Raspberry Pi –
Adding a LCD Display

Level of difficulty: Beginner / Intermediate
Hardware: Raspberry Pi Model B, LCD Display Module, wires, optional connectors
2 x Potentiometers and 1 Resistor
Tools required: Wire cutters, soldering iron
Project cost: Under $10-$20

Document Version: October 2013 v1.0.0

Written By: David Osborne raspberry_pi@hotmail.com

The point of this series of documents is to provide a very simple set of directions to add hardware or install software to the Raspberry Pi. These represent my findings. I'm not a 'professional', not an engineer and don't come close to being an expert on the Raspberry Pi (RPi). I am however familiar with a lot of the fundamentals of electronics, pretty good at soldering and have done a lot of programming in many different languages. You follow my advice at your own risk. Much of the information here is taken from other documents and from others who are in some cases much more knowledgeable than I am. One of the issues I found is that many articles are out of date. These documents are specifically for the Raspberry Pi Model B 512MB and based on what I could find in the Fall of 2013. For the most part I expect this information will apply to the original RPi model and may even carry forward. I hope you find them useful.

Introduction

A Liquid Crystal Display (LCD) Module can be a fun addition to your Raspberry Pi (RPi). It does not replace a monitor, and the type described here does not display graphics, but it can display basic textual information. One of the easiest ways to accomplish this is to install Python, a relatively simple yet powerful programming language. And send lines of text to the LCD. There are many examples available on the internet and it is likely that the more common uses have already been coded by someone so a little tweaking and you should be good to go.

Hardware

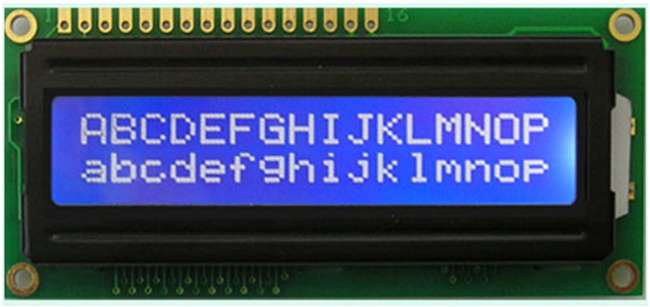
Installing a LCD Module is moderately harder than installing some other devices such as a Real Time Clock, but really only due to the number of connections that are required. It is a fairly straightforward process and can open a wide range of options for you to build a case, add switches, etc.

Advanced users can go directly to the Connection Summary if they are only interested in the connection details.

Keep an eye on eBay and the various other web sites and you can often find a really good deal on a starter display. The most common and simplest LCD module is a 16 character, 2 line display referred to as a 16x2 or 1602 LCD Module. This basic module can sometimes be found as cheap as $2 or may run up to $20. Although there seem to be slight manufacturing differences, most use the HD44780 Controller chip made by Hitachi (http://en.wikipedia.org/wiki/Hitachi_HD44780_LCD_controller). This not only makes them all compatible, but also allows for simplified code and cheaper costs.

Once again, the RPi community can take advantage of technology that has been around originally for other units such as the Arduino. In fact, it seems easier to add an LCD module to the RPi from what I have read.

Here is an image of a simple 1602 LCD Module.



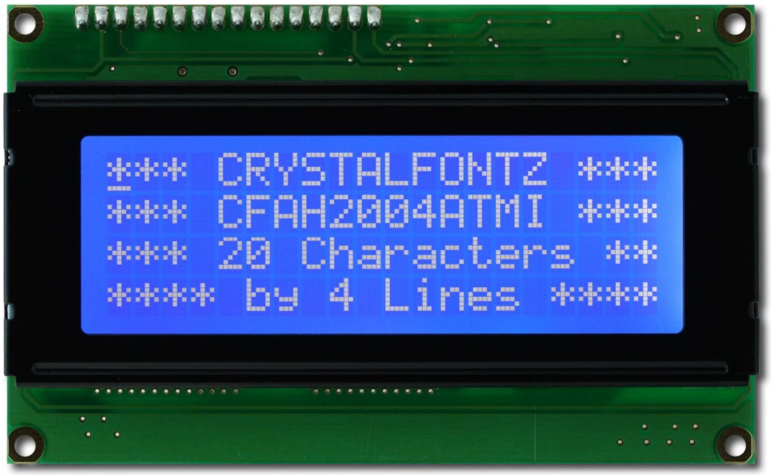
The main thing to note here is that the interface (top left of image) has 16 connection points, although we will not need to use them all.

There are many variations on this basic example. For instance, you can get a LCD Module that has buttons on it and you can program them to perform various operations or actions, basically whatever you decide to program them to do.

Here is an image of the SainSmart LCD with 6 front-mounted buttons.

You can also get LCD modules that can display more (4 lines x 20 characters) or less (1 line x 16 characters) information.

Example of the 2004A (20x4) LCD Module.



The fancier the module, the more expensive it will be. Fortunately though you can find good pricing if you shop around. I was able to find a 2004A for about $6 for example. Personally, I bought all 3 types shown here, and bought 3 of the basic 1602 Modules for $1.50 each. As it turns out it was a good thing I did as after soldering and desoldering the first one about a dozen times I ended up lifting some of the traces off the PCB which more or less made it useless. This was my intention though, I was using it to build my prototype as I was waiting for various parts / connectors to arrive in the mail.

In the end, I soldered a connector (1x16) to each LCD Module and built an interface on a small breadboard which the matching 1x16 interface so it is simply a matter of interface into the appropriate LCD Module and attaching the GPIO cable to the RPi and I can switch back and forth between them now. More on that later.

As you search to internet on this subject you will find various references to I2C, serial modules and other stuff. Fortunately, for the RPi Model B, this is not required and it is straightforward to connect the two devices.

For starters, we will of course need power, so hopefully you bought a 5V LCD Module. All the ones described here are, so you should be ok. Two connections, +5V and Ground will be used. We will use four (4) of the GPIO pins to send data the LCD Module. The rest of the connections will be for contrast, backlighting and LCD Module control. Out of the 16 connection points on the LCD module we actually only need 12, in 2 sets of 6 so you have a bit of flexibility in terms of the connectors you use. On one I used two 1x6 connectors, on another I used two 1x8 connectors and on the third I used a 1x16 connector. It all works out the same as only 12 connection points are used. It was simply what I had handy.

The LCD Module gives you access to control the backlighting (LED) and contrast. This allows you to set the display to an easily readable level based on the room lighting conditions. It is of course possible to hard-wire a couple of resistors in place to reduce the size of the module, but we will take a better approach and install a couple of variable resistors or potentiometers. This will allow us to control both the backlight and contrast whenever we want. They are cheap and easy to install. I would also recommend a precautionary resistor on the backlight circuit to ensure you do not burn it out prematurely, in case it does not have built in protection. Again, simple and cheap.

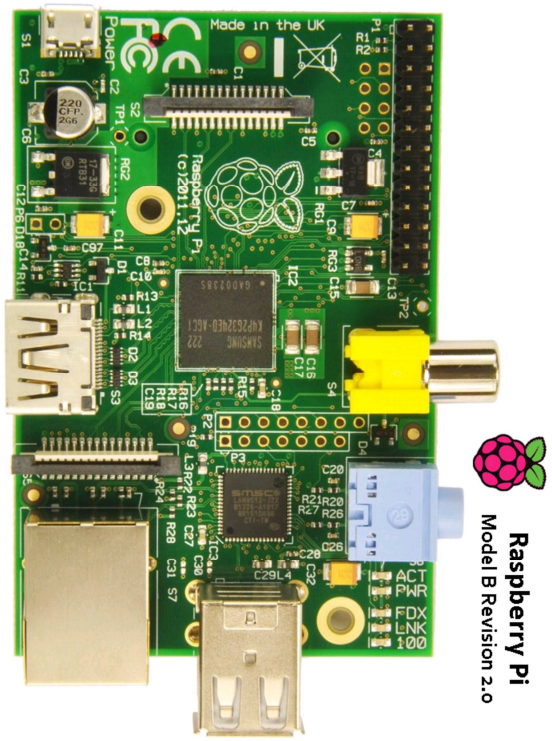
So what are we looking at in terms of parts (for a bare minimum):

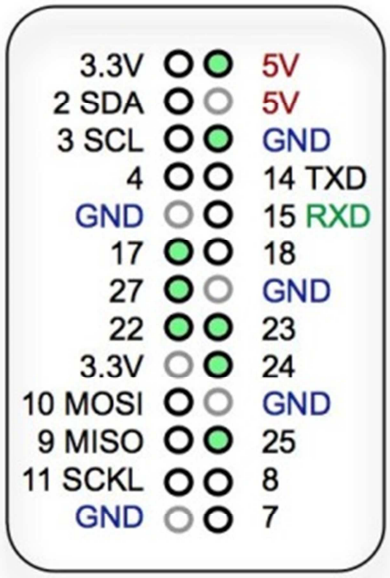
- (1) 1602 LCD Module – roughly $2-$5
- (2) 2K-10K Ohm Variable Resistors / Potentiometers - <$1
- (1) ~ 560 Ohm Resistor - <$1
- (1) 2x13 ribbon cable to attach to the RPi – roughly $5

I will assume you have a soldering iron, solder, solder-wick, wires etc. It would be a really good idea to get a small breadboard to solder all the parts to and therefore you will need a few connectors (2x13, 1x16, etc), some wires etc. You can also of course use a plastic project board and just push connecting wires into place. This is a great way to start and ensure it all works, then mount everything on a more permanent board.

I also opted to install a small slide on/off switch to control the LCD Modules power and backlight operation. This allowed me to install my Real Time Clock (RTC) module to the LCD interface I built but turn off the display if I wasn't using it and allow the RPi to use the RTC. This will become even more important as I add more and more devices and sensors. It is of course up to your imagination to decide what is right for you.

Ok, let's get down to the detail. Here are images showing the GPIO connector (top right of the RPi) and the function of the various pins. Pin #1 is at the top left.





The pins shown with the green circles are the ones we will be using in this project. Eight in total. +5V, Ground and six GPIO data and control pins.

On the other end, there are 16 connection points of which we will use 12.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Vss	Vdd	Vo	RS	R/W	E	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7	LEDA	LEDK

Again, the ones in green (1-6, 11-16) are the ones we will be using. At this point you are probably thinking, "Huh? How do I connect those, nothing matches up.". Right, so, let's look at what these do.

'.Interface Pin Function

Pin No.	Symbol	Level	Description
1	V_{SS}	0V	Ground
2	V_{DD}	5.0V	Supply Voltage for logic
3	VO	(Variable)	Operating voltage for LCD
4	RS	H/L	H: DATA, L: Instruction code
5	R/W	H/L	H: Read(MPU→Module) L: Write(MPU→Module)
6	E	H,H→L	Chip enable signal
7	DB0	H/L	Data bit 0
8	DB1	H/L	Data bit 1
9	DB2	H/L	Data bit 2
10	DB3	H/L	Data bit 3
11	DB4	H/L	Data bit 4
12	DB5	H/L	Data bit 5
13	DB6	H/L	Data bit 6
14	DB7	H/L	Data bit 7
15	A	-	LED +
16	K	-	LED -

Still confused? I know I was. Let's see where these connect.

LCD Pin #1 – Vss connects to Ground, or the neutral/negative power source.
LCD Pin #2 – Vdd connects to +5V power.
LCD Pin #3 – Vo controls the Contrast of the display and will connect to Potentiometer (Pot) #2
LCD Pin #4 – RS controls the LCD mode and will connect to GPIO 25 (RPi Pin # 22)
LCD Pin #5 – RW controls the Read/Write mode and connects to GND/Ground
LCD Pin #6 – E is another control connection and will connect to GPIO 24 (RPi Pin # 18)
LCD Pin #11 – DB4 Data connects to GPIO 23 (RPi Pin # 16)
LCD Pin #12 – DB5 Data connects to GPIO 17 (RPi Pin # 11)
LCD Pin #13 – DB6 Data connects to GPIO 27 (RPi Pin # 13)
LCD Pin #14 – DB7 Data connects to GPIO 22 (RPi Pin # 15)
LCD Pin #15 – A controls the LED + Backlight power and connects to Pot #1
LCD Pin #16 – K controls the LED – Backlight ground and connects to ground.

Ok, lots of numbers to look at, DB4, GPIO 23, RPI Pin 16, etc. Most of it you don't need to worry about, just look at what pin on the LCD connects to what pin on the RPi, at least for now. Installing using these GPIO ports and pin numbers will allow you to load some libraries and functions that are already coded, making the software control of the module really easy. You can use other ports/pins of course, but

hopefully only if you know what needs to be changed at the software end. This guide is really just about the hardware.

The key thing to remember is to take it one step at a time. Make one set of connections before moving on to the next or print out a diagram and highlight the wires as you go so you can track what has or has not been completed.

LCD Pin #5 – RW controls the Read/Write mode and connects to GND/Ground is shown in bold as it is very important. This tells the LCD Module that it is in Read mode and not to output or write and data out to the RPi. If it does, it would send out +5V on a data connection and the RPi is designed to accept only +3.3V on the data ports. Sending +5V would not be a good thing. The very first connection you make should be to join LCD Pin # 5 to LCD Pin # 1 or another ground point. All ground points should be connected together across the entire set of devices.

Let's make sure that the basics of your LCD Module will work. We will ignore the data connections for the moment and get everything else connected and test the LCD Module. For this test you can use the RPi or use any +5V power source.

To keep things simple, let's assume you have a separate power source and a simple on/off switch. It can be any kind of switch you want, DIP, toggle, slide, etc but since it will control power, it should be a latching switch of some sort i.e. you don't want a momentary push spring type switch, at least not for long term use. If you don't have a switch, then simply plug in or unplug the power source instead of turning the switch on/off as indicated.

Never make any connections with the power connected. Always remove the power source by unplugging from the wall or the RPi when connecting / disconnecting / soldering wires. Do not rely on a switch you are installing as part of this circuit as there is a live feed on one end and it would be easy to short something out. Enough said. You hopefully know what you are doing in this regard.

This project demonstrates the use of two (2) variable resistors or potentiometers (Pots). There are a number of different styles that can be used and of varying Ohm ratings. A Potentiometer allows you to "turn up" or "turn down" the settings that control the Brightness and Contrast in your project by varying the resistance to increase or decrease the actual voltage that goes to the device. I used 2 x 10K Ohm Pots in my project as they were the first to arrive in the mail. Using lower ratings mean more sensitivity but less range and using higher settings mean more range but less sensitivity. The guide I was basing my project on called for a 10K Ohm Pot and a 2K Ohm Pot, so it just means I have a little less sensitivity in the lighting control. Since this first one was my prototype I wasn't too concerned and it will not damage the LCD Module. Use what you can find as long as the values are close, they do not have to match 100%. Potentiometers have 3 connections or legs. The center connection, or the single pin on one side, is the Output and the pin/leg that connects to your "device", in this case the LCD Module. The other two pins/legs connect to +5V and Ground. By turning the screw or wheel on the Pot you increase or decrease the resistance from zero (0) to whatever the maximum rating is for the Pot. In my case the range is Zero to 10K Ohms. If the LCD Module does not have protection built into it then it may be possible to damage or burn out the LED. Knowing if there is protection in the LCD Module is next to impossible as getting a spec sheet for your particular one is unlikely. The safest thing to do therefore, other than never turning your Pot to Zero, is to install a resistor in the circuit. A resistor in the 500 Ohm range should be sufficient protection. I had a 560 Ohm resistor handy and used that.

Is it absolutely necessary to use a potentiometer to control the backlight? Of course not. It's your hardware and you know what you are going to use it for. If you do not use a potentiometer to control the brightness of the backlight I would suggest that you still use a resistor and a switch so that you can turn it off. For testing purposes you could easily skip these parts. You will need one Pot to control the contrast though.

Keep in mind that you want to minimize the number of times you connect / disconnect a wire or part of your circuit. I played around with my "design" quite a bit, trying to get parts to fit, have a logical layout, etc before I settled on a "final" layout. I suggest initially using a solderless breadboard to get everything working and then moving to a more permanent solution, whatever is appropriate for you.

Test the LCD Module Backlight

Connect LCD Pin #16 to Ground or the Negative connection on your power source (with the actual power disconnected of course).

Install your first Potentiometer (Pot1). If you have Pots with different values, this is likely the one you want the lower value on since you will also install a resistor. Connect Pot1 Pin # 2, or the center pin/leg, to LCD Pin # 15. Connect Pot1 Pin # 1 to a 560 Ohm resistor (R1). Connect the other end of the resistor R1 to the +5V Power connection point. If you want to install an on/off switch, then connect to that instead and connect the other side of the switch to the +5V Power. Connect Pot1 Pin # 3 to Ground.

Connect the power source and give it a try. Assuming your LCD Module has a backlight then you should be able to turn it up/down by adjusting Pot1.

Reversing Pot1 Pins # 1 and # 3 allow you to control the "direction" you use to increase/decrease the backlight. So, if you find that by adjusting your Pot1 clockwise decreases brightness, you can simply reverse the wires for Pot1 Pin # 1 and # 3. There is no right or wrong here, it is your choice. Connect it, try it, and then decide if you want to flip it the other way around.

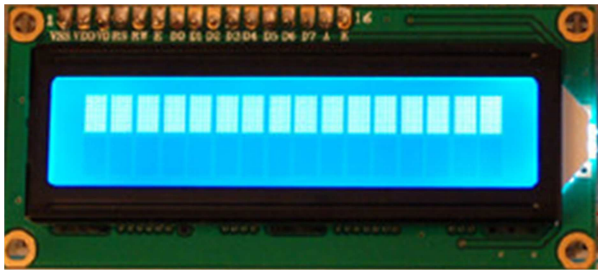
As mentioned above, this potentiometer could be considered optional. If you purchased a 5V display and it can accept +5V directly to the backlight, then go for it. Personally, I consider controlling the backlight as a non-optional function, but it is up to you.

Test the LCD Module Contrast

Install Pot2 to set the Contrast of the LCD Module. This one is a bit simpler since you do not need a resistor or switch. Connect Pot2 Pin # 2 or the center pin/leg, or the one on a side by itself (the output) to LCD Pin # 3. Connect Pot2 Pin 1 to the +5V power source (or the power switch as described earlier). Connect Pot2 Pin # 3 to Ground.

Connect the power source. Pot1 should still adjust the backlight and you should now be able to use Pot2 to control the Contrast. You should be able to adjust the two Pots to obtain a line of "black squares". This of course is dependent on your particular LCD Module. Colour may vary from this image too depending on what you purchased as some are green, some are blue, etc.

Example of a blue background LCD Module with Backlight (Brightness) and Contrast:



Connect the LCD Module Control Lines

Believe it or not, the hard part is done. The rest of the connections are all straightforward, no other components required other than wires or connectors. You should now have connections in place for LCD Pins # 1, # 2, # 3, # 15 and #16.

Connect LCD Module Pin # 5 to any available Ground point. This can be LCD Pin # 1, which is close by, either of the Ground connections for the Potentiometers that you installed, RPi GPIO Pin # 6 or the Ground plane on your breadboard. All of these should be electrically connected together so where you connect is up to you.

Connect LCD Pin #4 to GPIO 25 (RPi Pin # 22)

Connect LCD Pin #6 to GPIO 24 (RPi Pin # 18)

Connect the LCD Module Data Lines

Connect the four (4) Data lines between the LCD Module and the RPi GPIO port using whatever method is appropriate for you. Jumper wire, 2x13 connector, etc.

Connect:
LCD Pin #11 to GPIO 23 (RPi Pin # 16)
LCD Pin #12 to GPIO 17 (RPi Pin # 11)
LCD Pin #13 to GPIO 27 (RPi Pin # 13)
LCD Pin #14 to GPIO 22 (RPi Pin # 15)

Connection Summary

Note that most connections are listed twice as both the starting point and end points are listed.

Device Name: LCD Module
Device Type: 1602 LCD Module/Panel, 2004 LCD Module/Panel, etc
Device Connections:
LCD Pin #1 connects to Ground
LCD Pin #2 connects to +5V power.
LCD Pin #3 connects to Pot2 Pin # 2 (center)
LCD Pin #4 connects to GPIO 25 (RPi Pin # 22)
LCD Pin #5 connects to Ground
LCD Pin #6 connects to GPIO 24 (RPi Pin # 18)
LCD Pin #11 connects to GPIO 23 (RPi Pin # 16)
LCD Pin #12 connects to GPIO 17 (RPi Pin # 11)
LCD Pin #13 connects to GPIO 27 (RPi Pin # 13)
LCD Pin #14 connects to GPIO 22 (RPi Pin # 15)
LCD Pin #15 connects to Pot1 Pin # 2 (center)
LCD Pin #16 connects to ground.

Device Name: Potentiometer # 1 (Pot1)
Device Type: 2K Ohm Potentiometer
Other Options: Any 2K Ohm to 10K Ohm Potentiometer
Device Function: LCD Backlight / Brightness Control (Some may consider this an optional component)
Device Connections:
Pot1 Pin # 1 connects to Resistor # 1 (R1) Pin # 1
Pot1 Pin # 2 connects to LCD Module Pin # 3
Pot1 Pin # 3 connects to Ground

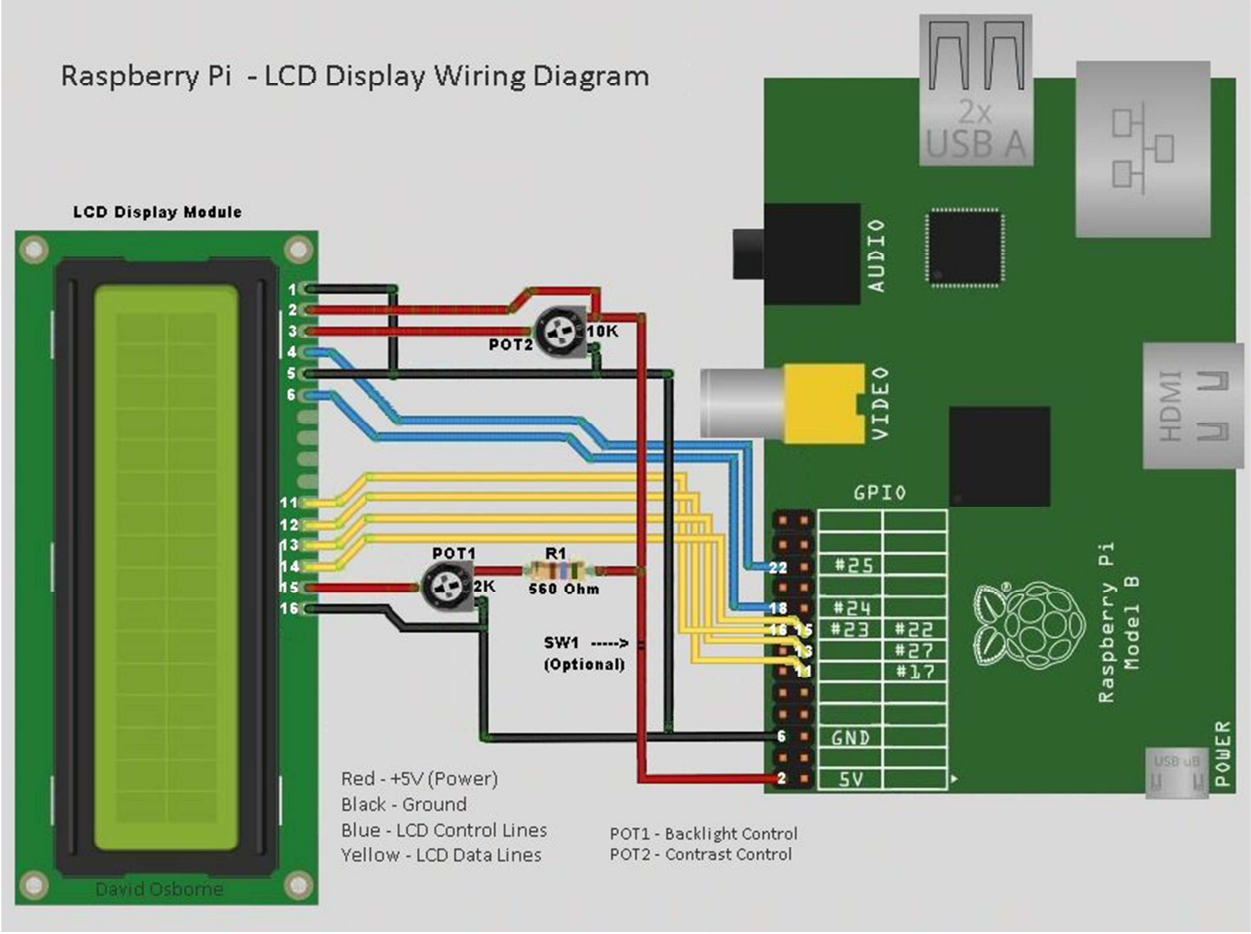
Device Name: Potentiometer # 2 (Pot2)
Device Type: 10K Ohm Potentiometer
Device Function: LCD Contrast Control
Device Connections:
Pot2 Pin # 1 connects to +5V Power
Pot2 Pin # 2 connects to LCD Module Pin # 3
Pot2 Pin # 3 connects to Ground

Device Name: Resistor # 1 (R1)
Device Type: 560 Ohm Resistor
Other Options: 560 Ohm Resistor
Device Connections:
R1 Pin # 1 connects to Pot1 Pin #1
R1 Pin # 2 connects to +5V Power

Optional but Recommended:
Device Name: Switch # 1 (SW1)
Device Type: On/Off Switch - Single Pole Single Throw (SPST)
Other Options: SPDT, DPST, DPDT, toggle, slide, etc
Device Connections:
SW1 Pin # 1 connects to LCD Pin #2 and R1 Pin #1
SW1 Pin # 2 connects to +5V Power

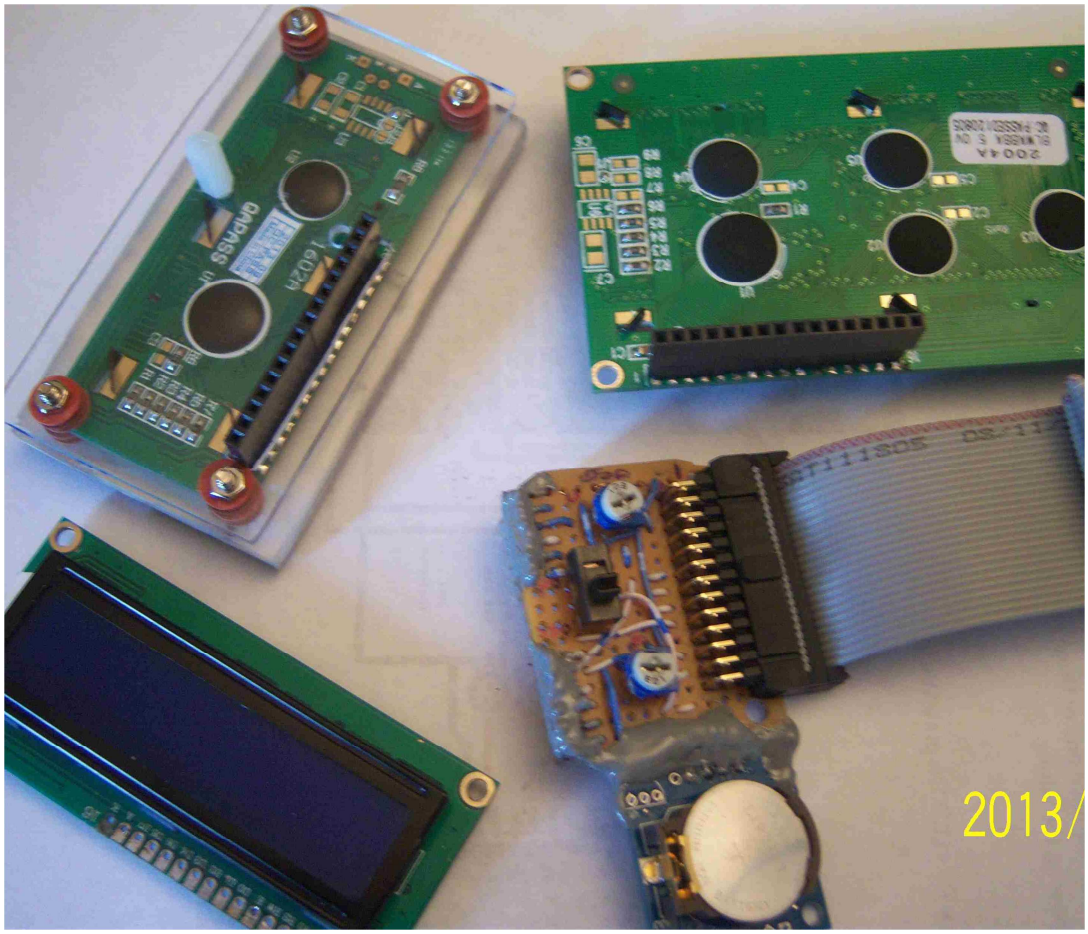
You can of course add just about anything to the optional list. You can get fancy and add transistors to control power, extra RPi GPIO controls to turn the LCD Module and / or Backlight on or off, etc.

Unfortunately, I do not have one of those fancy CAD programs some people use, so here is my attempt at a diagram showing the wiring.

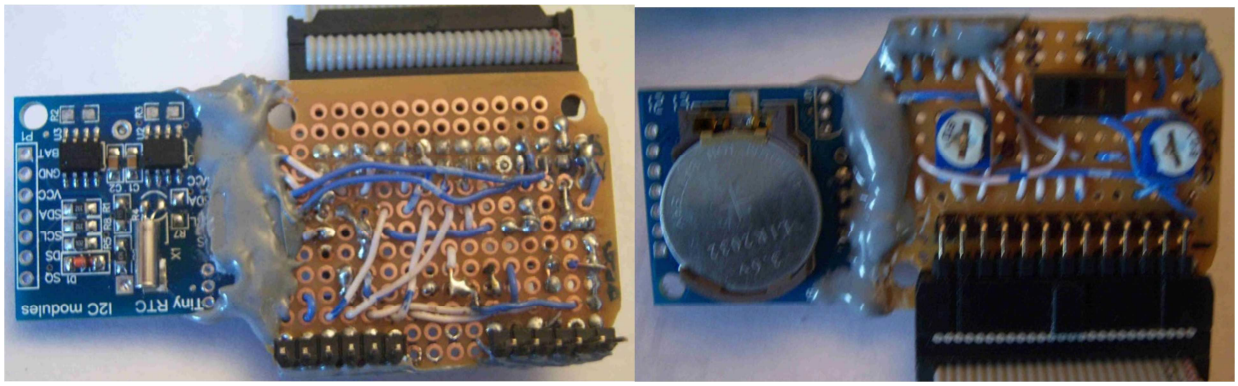


Since I had previously added a Real Time Clock (RTC), after I mounted the Pots and Resistor onto a small prototype board, I also mounted the RTC module and glued the two together. It was a simple matter to connect the four (4) clock connections to the nearest +5V and Ground sources and then run two wires to the RPi GPIO Port SDA Pin # 3 and SCL Pin # 5. With the switch I can leave it all connected so that the RTC is active all the time and use the switch to turn the LCD on or off depending on what I am working on.

Here is an image showing the module I created. This was my prototype / proof of concept build. I plan on making a new module after I play with some more sensors and other stuff.



The top two LCD Display Modules show the connectors (1x16) that I added. I also cut out some acrylic to make a frame for the display. You can see where I epoxied the RTC Module to the LCD Module. Not pretty, but it is functional.



Software Installation

In order to actually use the LCD Display you must install some software. At this point I will assume that you have a working RPi and you can access it either directly with a monitor / keyboard / mouse or are accessing it remotely through SSH, VNC or some other package. These instructions assume you have ***root*** (admin) user access or the password, and that the LCD Display is now connected.

You should also be familiar with a basic text editor that is used to modify the files as required below. There are many tutorials on Vi, ViM, Nano and other text editors that come with Raspbian. Other distribution packages of Linux for the RPi will be similar, but I will document the steps that I specifically followed for Raspbian.

The commands issued below represent what is required on a "clean" system. You may have already performed some of the steps installing other devices or software and if so, then you can skip doing them again.

Logging into your RPi as the root user simplifies the commands required but if necessary use the "sudo" command to preface the statements. Google search "sudo" or type "man sudo" for more information. This command allows you to run specific actions as if you were logged in as the root user which is necessary to make certain system level changes. For example, if you need to run the command "apt-get update" and are not logged in as the root user, then you can issue the command "sudo apt-get update" and the system will run the command with root privileges.

Although not technically required, I strongly suggest that you consider updating all your packages to the most current level. If for some reason you are unable to do so, you should at least consider updating the required packages. The commands and methods here reflect using "2013-09-10-wheezy-raspbian.img".

In Raspbian, and other Debian Linux-like distributions, you can use the "apt-get" command to install or update the software on your system. To update all your installed software packages, you can issue the following two commands:

apt-get update

apt-get upgrade

These commands will go to a predetermined web site and search for and install any updates that are available. It may take a few minutes to run and obviously requires a connection to the internet.

Adafruit (http://www.adafruit.com/) provide hardware and software for the RPi. If you are looking for peripherals, it is a good place to start.

You can use just about any programming language that the RPi supports to drive the LCD Display. This set of instructions is meant to represent the absolute bare minimum in order to test that the LCD

Display is wired correctly. There is a LOT of information on the internet around what you can code, much of the work is likely done for you as these modules are very popular.

If you have not previously installed Python, install it now. From a command prompt / terminal window, run the following commands:

```
apt-get install python-dev

apt-get install python-setuptools

easy_install -U distribute

apt-get install python-pip
```

Each one of these commands may take a few minutes to run and there may be one or more prompts you need to answer.

Python code for LCD Displays is available on Github at https://github.com/adafruit/Adafruit-Raspberry-Pi-Python-Code (http://adafru.it/aOg)

The easiest way to download and install the code onto your RPi is to use the 'git' command. Create and change to a directory where you want to install the Python code. For example, you can create and install to /home/LCD.

```
apt-get install git

git clone git://github.com/adafruit/Adafruit-Raspberry-Pi-Python-Code.git

cd Adafruit-Raspberry-Pi-Python-Code

cd Adafruit_CharLCD
```

The Python code was designed for the first release (Model A) of the RPi, so you need to make a tiny change before you can use it. Use any text editor you are familiar with and change the file **Adafruit_CharLCD.py**

Find the line of text that matches:

```
def __init__(self, pin_rs=25, pin_e=24, pins_db=[23, 17, 21, 22], GPIO = None):
```

and change it to:

```
def __init__(self, pin_rs=25, pin_e=24, pins_db=[23, 17, 27, 22], GPIO = None):
```

This change is necessary as the RPi Model A had GPIO Port 21 at the same hardware location as the RPi Model B has GPIO Port 27. If you have a RPi Model A then you do not need to change this entry.

Ensure that all the Python code file (end in .py) are executable, i.e. you can run them as programs. Issue the following command:

chmod +x *.py

Now, run the Python example program that was downloaded. It will display the IP Address of your RPi and the current Date and Time on the LCD Display. Be sure to use a (.) dot (/) slash as shown:

/Adafruit_CharLCD_IPclock_example.py

Your LCD Display should now show something like the following example. Obviously your IP Address and the Date and Time may be different ☺.

You may have to adjust the Contrast and Brightness in order to see the text, but if you were able to see the “black boxes” previously then the settings should be close.

You can now explore the various programming options and send text to the LCD Display.

Now that the LCD Display is working you can add other controls using more GPIO connections to turn the Backlight on/off, etc.

Project complete! I hope you found this information helpful.