Arduino Programming
Part 6: LCD Panel Output

EAS 199B, Winter 2010

Gerald Recktenwald
Portland State University
gerry@me.pdx.edu

Goals

Use the 20x4 character LCD display for output
❖ Overview of assembly — detailed instructions on the web
  ➔ http://web.cecs.pdx.edu/~gerry/class/EAS199B/howto/LCDwiring/
  ➔ http://www.ladyada.net/learn/lcd/charlcd.html
❖ Introduction to the LCD library
  ➔ http://www.arduino.cc/en/Tutorial/LiquidCrystal
❖ Simple demonstration
❖ Map the 20x4 character display for fish tank data

Breadboard connection via Adafruit Tutorial

http://www.ladyada.net/learn/lcd/charlcd.html
1. Header for electrical connections
2. Potentiometer for contrast adjustment
3. Panel on PCB

---

**Wiring diagram**

<table>
<thead>
<tr>
<th>Connector</th>
<th>Arduino</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Green</td>
</tr>
<tr>
<td>2</td>
<td>Red</td>
</tr>
<tr>
<td>3</td>
<td>Yellow</td>
</tr>
<tr>
<td>4</td>
<td>Brown</td>
</tr>
<tr>
<td>5</td>
<td>Green</td>
</tr>
<tr>
<td>6</td>
<td>White</td>
</tr>
<tr>
<td>7</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Blue</td>
</tr>
<tr>
<td>12</td>
<td>Black</td>
</tr>
<tr>
<td>13</td>
<td>Gray</td>
</tr>
<tr>
<td>14</td>
<td>Orange</td>
</tr>
<tr>
<td>15</td>
<td>Red</td>
</tr>
<tr>
<td>16</td>
<td>Green</td>
</tr>
</tbody>
</table>

---

**Step 1: Solder the header**
Step 2: Assemble the wiring harness

Cable bundle
Jumper wire for ground pins
Jumper wire for +5V pins
Crimp connectors
Connector block

Crimp Connectors: they are small!

Female
Strain relief
Electrical signal
Barb
Male

Use jumpers to avoid unnecessary wire

Connection to +5V on Arduino
+5V on pin 2 and pin 15

Connection to ground on Arduino
Ground on pin 1, pin 5 and pin 16
Locate the crimp connector in the tool

Crimp the strain relief

Finished crimping for the female connector
Finished female and male connectors

Female connector for LCD end

Male pins for Arduino end

Note: These male pins still need heat shrink to insulate pins from each other when they are inserted into a breadboard.

Programming Arduino for LCD Display

Refer to Adafruit tutorial

❖ http://www.ladyada.net/learn/lcd/charlcd.html

and Arduino documentation

❖ http://www.arduino.cc/en/Tutorial/LiquidCrystal

Breadboard connection via Adafruit Tutorial

http://www.ladyada.net/learn/lcd/charlcd.html
```cpp
// include the library code:
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
    // set up the LCD's number of columns and rows:
    lcd.begin(16, 2);
    // Print a message to the LCD.
    lcd.print("hello, world!");
}

void loop() {
    // set the cursor to column 0, line 1
    // Line 1 is the second row, because counting begins with 0
    lcd.setCursor(0, 1);
    // print the number of seconds since reset:
    lcd.print(millis()/1000);
}
```

Test the display

```cpp
// include the library code:
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
    // set up the LCD's number of columns and rows:
    lcd.begin(16, 2);
    // Print a message to the LCD.
    lcd.print("hello, world!");
}

void loop() {
    // set the cursor to column 0, line 1
    // Line 1 is the second row, because counting begins with 0
    lcd.setCursor(0, 1);
    // print the number of seconds since reset:
    lcd.print(millis()/1000);
}
```
Arduino code to write to the LCD panel

Include the LCD library

In the header:  
#include <LiquidCrystal.h>

Initialize the display by creating a LiquidCrystal object

Before using the display:  
LiquidCrystal lcd(p1,p2,p3,p4,p5,p6);
lcd.begin(20,4);

Send characters in a two-step process

Move the cursor:  
lcd.setCursor(column,row)

Display the message:  
lcd.print("message")

Character matrix on a 4 X 20 display

Row and column indices begin with zero

```
0123456789012345689
0
1
2
3
```

Character matrix on a 4 X 20 display

Row and column indices begin with zero

```
lcd.setCursor(6,2)
lcd.setCursor(0,0)
```

```
0123456789012345689
0
1
2
3
```
Display fish tank salinity

Modify the HelloWorld code to display the salinity
❖ “Salinity = ” and “Average of ” can be displayed once at the start
❖ x.xx and NNN values change, and are updated on the display.

<table>
<thead>
<tr>
<th>0</th>
<th>Salinity = x.xx%</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Average of NNN</td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

### Programming Paradigms

To think about styles of programming, we can organize programming languages into paradigms

<table>
<thead>
<tr>
<th>Paradigm</th>
<th>Representative Languages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Procedural or Sequential</td>
<td>Fortran, C, Basic</td>
</tr>
<tr>
<td>Object-oriented</td>
<td>C++, smalltalk</td>
</tr>
<tr>
<td>Parallel /Concurrent</td>
<td>occam, erlang</td>
</tr>
<tr>
<td>Dataflow</td>
<td>LabVIEW</td>
</tr>
<tr>
<td>Functional</td>
<td>Haskel, Lisp</td>
</tr>
<tr>
<td>Scripting</td>
<td>perl, python</td>
</tr>
</tbody>
</table>

Note that many modern programming languages have features of more than one paradigm

### Object-Oriented Programming (OOP)

As you might expect, Objects are central to OOP
❖ Objects have data  
❖ Objects have methods (like functions)  
❖ Objects can be assembled into other objects.

**Arduino Programming**
❖ Uses the object-oriented language C++
❖ Don’t get carried away with the OOP on Arduino
  ❖ Keep your Arduino programs from becoming too complex
  ❖ Basic structure of code, with setup() and loop() is sequential
❖ Libraries for the Serial Monitor and LCD output use OOP
  ❖ Know enough OOP to use existing libraries
  ❖ OOP can be handy when programming with new types of sensors
Create a new LiquidCrystal object:

```cpp
LiquidCrystallcd(p1,p2,p3,p4,p5,p6);
```

When a new object is created, the data passed to the constructor is stored in the object. Thus, whenever we use the variable `lcd` again in the program, the `lcd` object “knows” that it is connected to `p1`, `p2`, ..., `p6`.

Tell the `lcd` object about the size of the display:

```cpp
lcd.begin(20,4)
```

Objects have data and methods:

- Data are values associated with a particular “instance” of an object.
- Some data may be “public”. Programmers can view or change public data.
- Some data may be “private”, and therefore unavailable to programmers.
- Methods are functions that an object knows how to perform:
  - Methods can return values
  - Methods can change public data
  - Methods can perform computations and interact with the environment (sensors)

Change the current cursor position:

```cpp
lcd.setCursor(12,1)
```

The `setCursor` methods prepares `lcd` for its next action:

```cpp
lcd.print("Hello")
```

`lcd.print(...)` works because the `lcd` object “knows” about its current position (from `setCursor`), the size of the display (from `begin`), and from the pin assignments from the constructor. When the `lcd.print()` method runs, it unleashes action that is constrained by data stored in the object.