CG1111 Engineering Principles and Practice I

Arduino Workshop

<u>Challenge Activity</u>

Studio Objectives:

1. Learn to translate a schematic diagram to a breadboard prototype
2. Develop neat bread-boarding techniques
3. Develop a simple Arduino-based application
4. Learn about the Serial-Port Interface

Materials:

- Breadboard, connecting wires, Arduino Uno, USB cable
- 5% tolerance resistors (4 x 560 Ω per group)
- LEDs (2 Red, 2 Green)

Introduction:

In the Arduino workshop you have been introduced to the Arduino Uno Microcontroller and some of its basic features. The code that you develop makes extensive use of pre-built libraries that simplify the underlying hardware interface. It is now very easy to start developing an embedded application using such development tools.

To challenge you slightly, we are now going to build a simple project that will also test your HW prototyping skills, as well as basic SW development techniques.

We will develop this project progressively over the next four activities. Code snippets are provided to help you get started. Apply the C coding techniques that you have learned in your CS1010 to complete the project.

Activity 1: Running LED

In this activity you are going to focus on developing the HW and SW to create the LED running effect. The connections between the Uno and the LEDs are shown below in Figure 2.

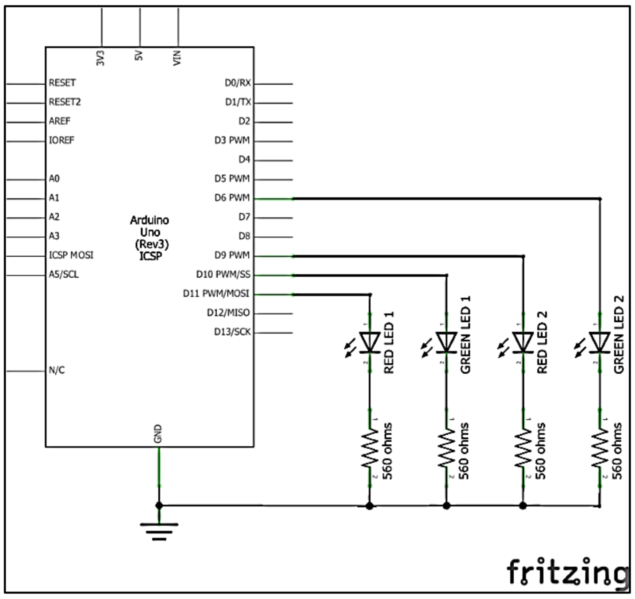


Figure 2: LED Connections to Arduino

The connections are as described below:

Arduino Pin	LED
D11	RED LED 1
D10	GREEN LED 1
D9	RED LED 2
D6	GREEN LED 2

A short video on its expected behaviour can be seen here:

https://youtu.be/li6EFvSMhEM

Code Snippet:

```
void setup() {
      pinMode(R1, OUTPUT);
      …
      // Add configuration for the other pins
}
```

In your `loop()` function, you may want to implement function calls to light-up the appropriate LED. It is important to add an appropriate delay after a LED is turned ON, so as to see the desired effect.

Activity 2: Fading LED

In this activity, you are going to implement a similar sequence as in Activity 1, but instead of just turning the LED's On/Off, you are going to create a Fading effect as shown here:

https://youtu.be/b5mP5cw8YWo

The setup() will be the same as before, but now you need to make use of the analogWrite() function to create this effect. Remember that only one LED should be fading at any time.

Activity 3: User-Interface Testing

Normally, we use buttons to control the LED's. Now, we are going to do something a bit more fun by using your laptop to send commands to control the LED's. You can view the demo here:

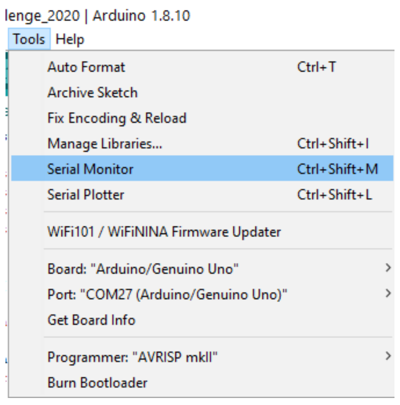
https://youtu.be/ZobellKXDcA

The Uno's USB Cable provides a Serial Link to the Computer. Yes, I can see the alarm bells ringing as you haven't been taught all these yet. Not to worry. We will guide you through the process.

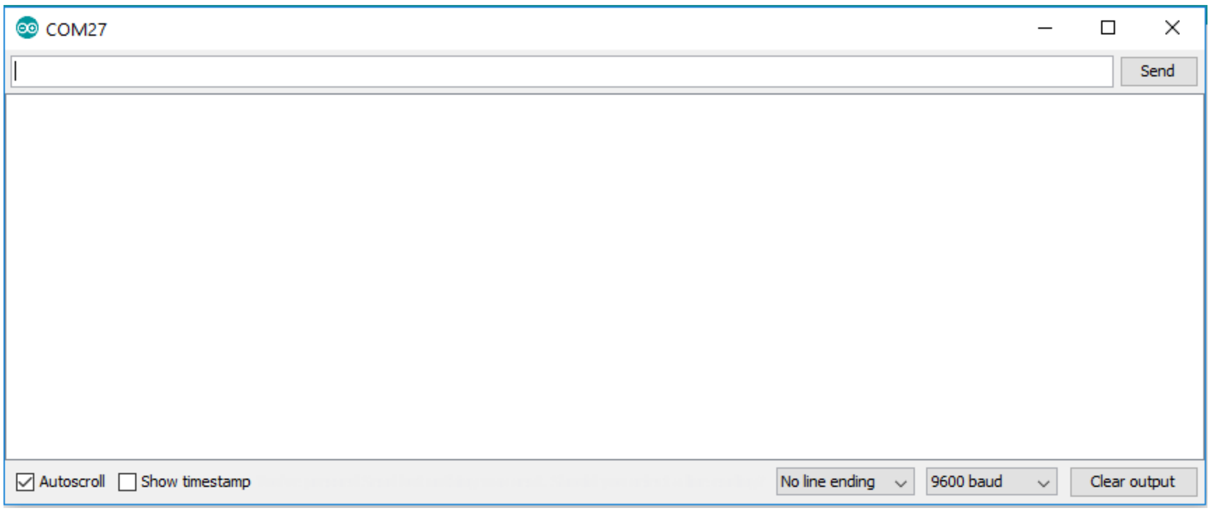
Step 1: Enable the Serial Port in your Setup().

```
void setup() {
  Serial.begin(9600);
  // Other Setup code…
}
```

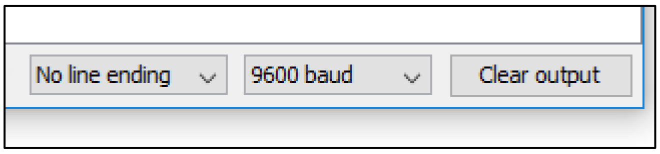
Step 2: Open up the Serial Monitor in your Arduino IDE


lenge_2020 | Arduino 1.8.10
Tools
Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Board: "Arduino/Genuino Uno"
Port: "COM27 (Arduino/Genuino Uno)"
Get Board Info
Programmer: "AVRISP mkII"
Burn Bootloader

The Serial Monitor will look like this:



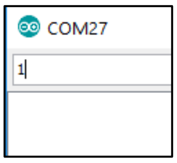
Make sure that at the bottom-tight corner of the monitor window, the settings follow as what is shown below.


No line ending
9600 baud

<u>Step 3:</u> Modify your loop() as shown below:

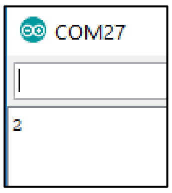
```
void loop() {

  if(Serial.available() > 0)
  {
    char data = Serial.read();
    Serial.write(data+1);
}
```

Step 4: Enter number '1' in the textbox and press <Enter>.



1

You will observe that number '2' is sent back as the reply.



```
2
```

If this is working fine, then your Serial Port is setup correctly. Well Done!

You can try character inputs as well and observe the response. When you send an ‘a’, you get ‘b’ as the response. If you are interested to understand more on this, you can read up on ASCII characters and their equivalent value, but that is not needed for now.

The detailed documentation on the Serial Port can be found here:

https://www.arduino.cc/reference/en/language/functions/communication/serial/

For now, we can see that Serial.read() captures the data that we send from the computer, and Serial.write() sends data from the Arduino back to the computer.

Activity 4: Controlling LED’s through the Serial Interface

In this, your objective is to send commands to the Uno through the serial interface and create the appropriate LED effect. The commands can be simple numeric digits like “1” or “2”.

In your code, you need to incorporate <if-else> or <switch-case> statements to check for the incoming data and then call the appropriate LED function.

For this activity, there is no need to send data back to the PC using the Serial.write().

Once you are done, you can extend it to make more LED effects together with more commands from the PC.

THE END