GEEN 1200 - Engineering Discovery 1

Electrical Engineering and Computer Engineering (EECE) Module
Laboratory 1 – Arduino First Steps

Cristinel Ababei and Susan C. Schneider
Dept. of Electrical and Computer Engineering, Marquette University
Fall 2017

1. Objectives

To become familiar with the Arduino development environment (IDE) by creating several projects.

2. Introduction

In this lab, you will work through three Arduino projects in which you will "drive" LEDs (light-emitting diodes). These projects were adapted from some of the projects discussed in [2]. These projects will allow you to become familiar with the use of the Arduino IDE to program the microcontroller as well as the construction of circuits on an experiment's breadboard – aka protoboard.

Materials Needed

Hardware (HW) –	Software (SW) –
Arduino Uno, USB mini to USB B cable Experimenter's wireless breadboard, connection wires 4 LEDs, 7-segment display, 220 Ohm resistors Pushbutton switch, momentary NO	Arduino IDE

3. Preparation for Laboratory 1

This is to be completed PRIOR TO your lab #1 session on 11/7/2017.

REQUIRED

- Download/extract the Arduino IDE software
  - Go to https://www.arduino.cc/en/Main/Software. Choose "Windows zip file for non admin installation". (Donate, or not). Download.
  - Extract the files in the `arduino-1.8.5-windows.zip` to a folder on your hard drive. Suggestion: Use a descriptive file name that you will remember is associated with the Arduino software. For example, I extracted the files to the folder M:/arduino185.
  - After the files are extracted, open the folder and locate the Arduino.exe file. Also note that a libraries folder has been created.
  - You will complete the installation process in the laboratory as access to an Arduino Uno board is required.

HIGHLY RECOMMENDED

- Read up on how to use the experiment's breadboard (protoboard, wireless breadboard)
    - http://www.tweaking4all.com/hardware/breadboard/
    - Be able to answer simple questions on breadboard.
- Read up on LEDs.
    - https://learn.sparkfun.com/tutorials/light-emitting-diodes-leds
    - http://www.electronics-tutorials.ws/diode/diode_8.html
    - Be able to answer simple questions on LEDs.
- Read up on 7 segment displays
    - https://en.wikipedia.org/wiki/Seven-segment_display
    - What's the difference between a common cathode and a common anode implementation of the 7-segment display?
- Read through the steps associated with the three projects given "In the Laboratory" section of this writeup.
- Type up the sketches for the three projects using the code in Appendix 1. Edit the comments ("notes") appropriately. Store on your laptop computer in the appropriate location.

4. In the Laboratory

Complete the following experiments. As each experiment is completed, show the working system to a teaching assistant for credit. Be prepared to answer questions about your work.

1) Complete the Arduino IDE installation

Connect the Arduino Uno board to your computer using the USB cable supplied.
Windows 8 and 10 will download the appropriate drivers. If you use Windows 7, you will need to manually install the driver by following the (excellent) instructions at https://www.arduino.cc/en/Guide/Windows#toc4

2) Experiment 1: Build and Test the "Blinky" project

a) Hardware setup

Follow the ***schematic diagram*** of this project shown in Fig.1 to connect the LED to the Arduino.

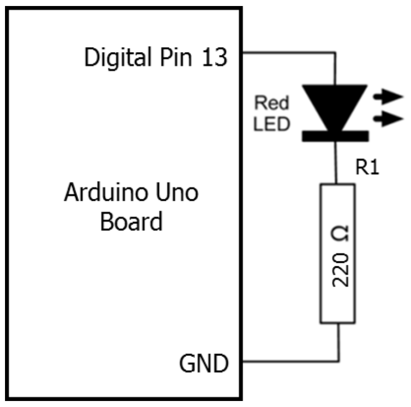

Figure 1: Schematic diagram of 1st experiment.

Your final hardware setup will look something like the ***protoboard or breadboard layout*** shown in Fig.2.

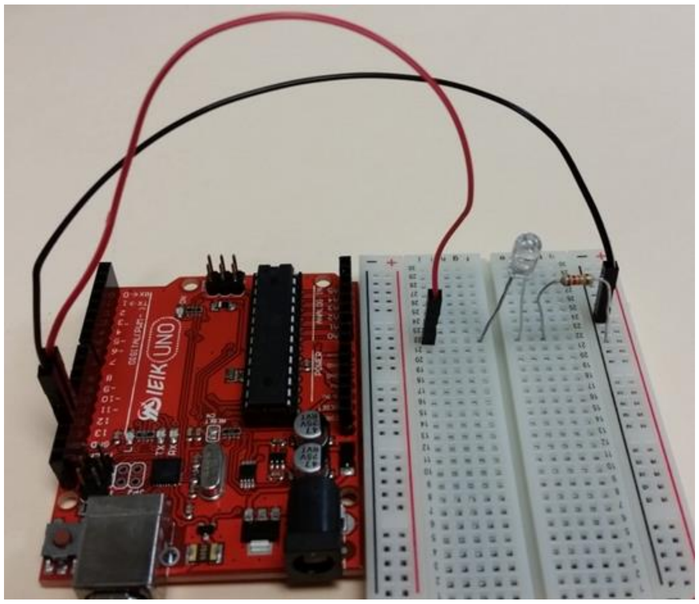

Figure 2: Protoboard layout of 1st experiment.

b) Software (i.e., sketch)
The software is the sketch or the program, which we enter using the Arduino IDE or any text editor. In this first project the sketch file is called **project_blink.ino;** see listing in Appendix A, at the end of this document. After the sketch is typed in, it must be saved in a new folder in your working (Arduino) directory with the same name as the file, **project_blink/**.

c) Verify and compile
Once you have finished writing the sketch, press the Verify icon within the Arduino IDE. If your sketch has errors, debug and fix them.

d) Upload the compiled sketch to Arduino's MCU
Use the given serial USB cable to connect the Arduino board to the host PC. Then, within the Arduino IDE, click the Upload icon. Your program should now be running.

e) Observe operation. Consider changes.
Observe the operation. For a properly operating project, the LED on the protoboard should be flashing.
Look at the sketch – what line(s) in the sketch do you think determine(s) the rate at which the LED flashes? What would you do to increase the length of time during which the LED is on? Try it!

3) Experiment 2: Build and Test the "LED Marquee" Project.

a) Hardware

In this project we control a set of four LEDs to make them blink one by one from left to right. The schematic diagram of this project is shown in Fig.3.

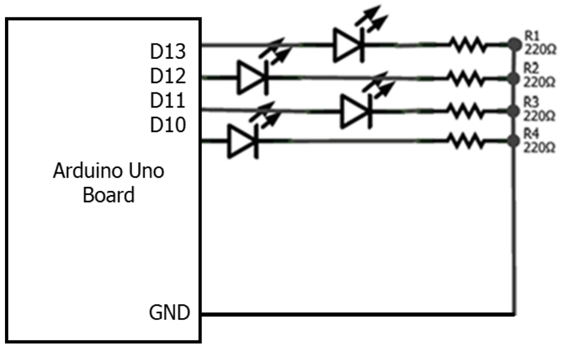

Figure 3: Schematic diagram of 2nd experiment.

The protoboard layout is shown in Fig.4 below.

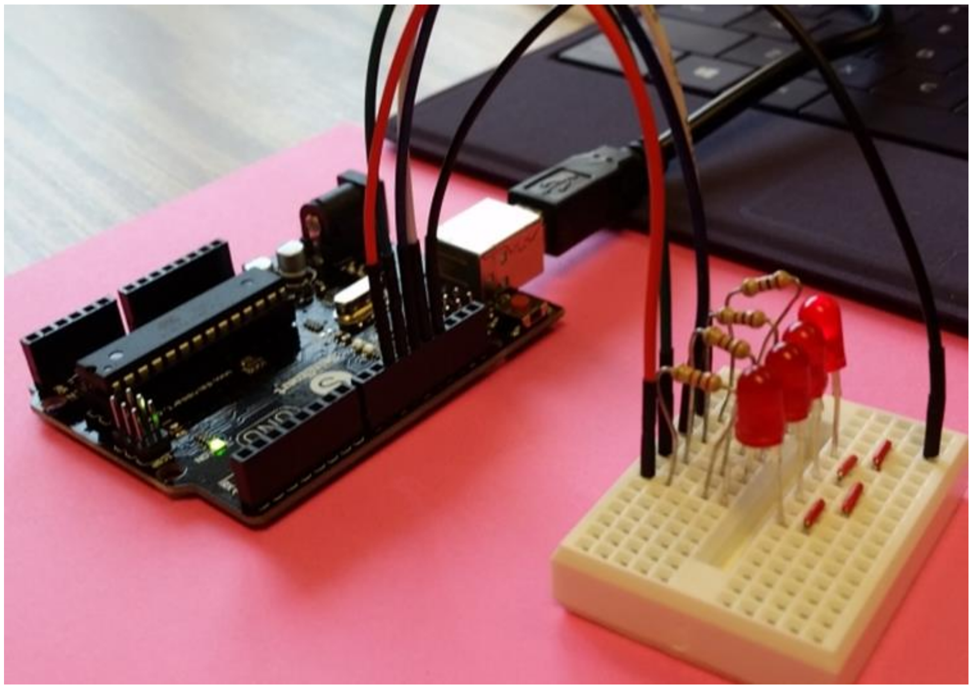

Figure 4: Protoboard layout of 2nd experiment.

b) Software
Use the listing for **project_marquee.ino** in Appendix B to create the sketch file. After the sketch is typed, save it in a new folder with the same name as the file.
c) Verify and compile. If your sketch has errors, debug and fix them.
d) Upload the compiled sketch to the Arduino MCU.
e) Observe operation. **Consider changes**.
Describe the LED lighting sequence of this sketch. Modify the project to make LEDs blink from left to right followed by right to left repeatedly. Modify the project to make LEDs blink all at the same time followed by a pause of 1 second. Propose your own style of blinking. Try them!

4) Experiment 3: Build and Test the 7-segment LED Project

a) Hardware
In this project, you will build a simple die. A push-button switch is used to "toss" the die. When the button is pushed, the microcontroller generates a random number between 1 and 6, which is then displayed on a single 7-segment LED display (see Fig.5).

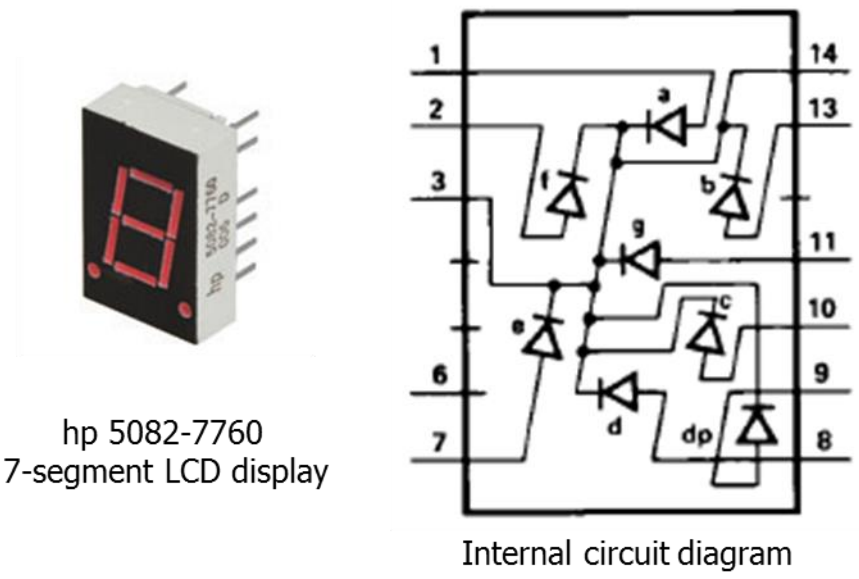

Figure 5: Example of 7-segment LED display.

The schematic diagram of this project is shown in Fig.6.

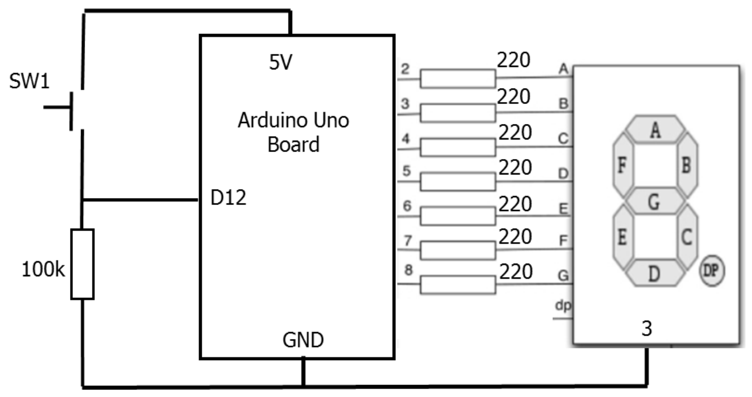

Figure 6: Schematic diagram of 3rd experiment.

The protoboard layout is shown in Fig.7.

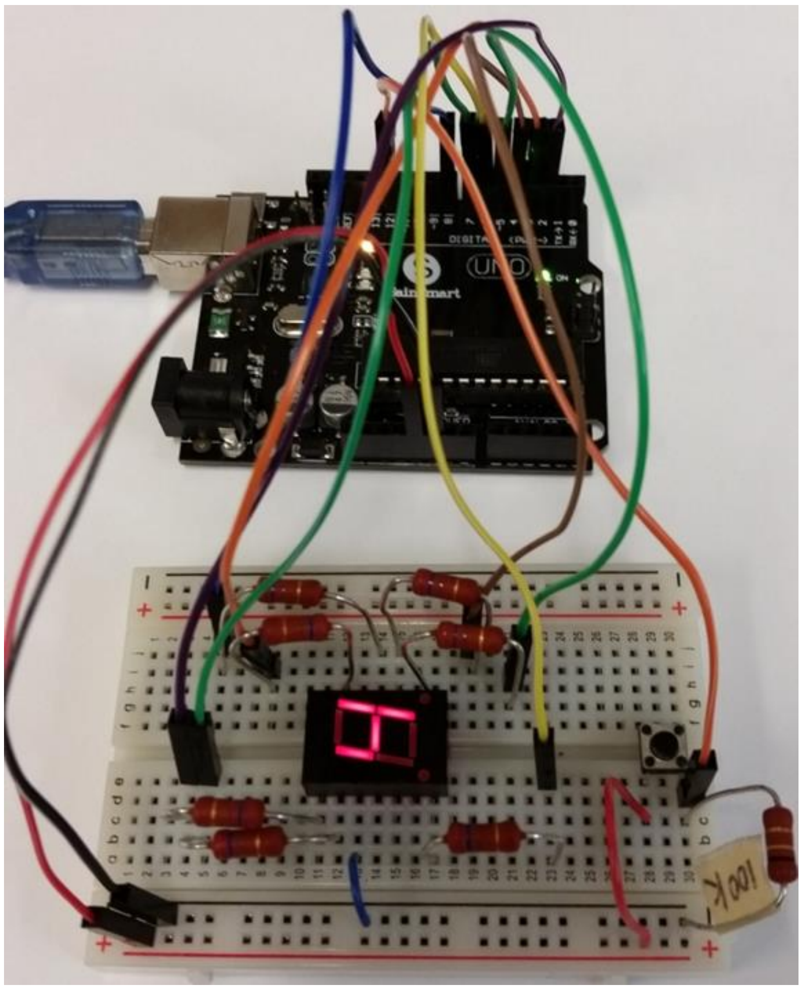

Figure 7: Protoboard layout of 3rd experiment

b) Software
Use the listing for **project_7segment.ino** in Appendix C to create the sketch file. After the sketch is typed, save it in a new folder with the same name as the file.
c) Verify and compile. If your sketch has errors, debug and fix them.
d) Upload the sketch to the Arduino.
e) Observe operation. Consider changes
Modify the sketch to generate a random number between 1 and 9 instead of a random number between 1 and 6. Modify the sketch to display randomly one of the following letters {A, C, E, F, L, U} instead of a random number between 1 and 6. Change this sketch to generate random numbers continuously. Try it!

References

[1] Simon Monk, 30 Arduino Projects for the Evil Genius, McGraw Hill, 2010.

Appendix A – Sketch for “Blink” Project

```
// Sketch:  project_blink.ino
// Turns on an LED on for one second, then off for one second, repeatedly

void setup() {
  // we will connect an LED to Pin 13, which we want to operate as an output
  // therefore, we initialize the digital Pin 13 as an output
  pinMode(13, OUTPUT);
}

void loop() {
  digitalWrite(13, HIGH);  // turn the LED on
  delay(1000);             // wait for a second
  digitalWrite(13, LOW);   // turn the LED off
  delay(1000);             // wait for a second
}
```

Appendix B – Sketch for “LED Marquee” Project

```
// Sketch:  project_marquee.ino
// turns on/off a set of four LEDs in different styles

void setup() {
  // initialize the digital pins 10,11,12,13 as outputs
  pinMode(13, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(11, OUTPUT);
  pinMode(10, OUTPUT);
}

void loop() {
  // turn LEDs on and off one by one from left to right
  digitalWrite(13, HIGH);
  delay(300);
  digitalWrite(13, LOW);
  digitalWrite(12, HIGH);
  delay(300);
  digitalWrite(12, LOW);
  digitalWrite(11, HIGH);
  delay(300);
  digitalWrite(11, LOW);
  digitalWrite(10, HIGH);
  delay(300);
  digitalWrite(10, LOW);
  // uncomment the following lines to turn the LEDs on/off
  // from left to right and then from right to left, repeatedly
  //digitalWrite(11, HIGH);
  //delay(300);
  //digitalWrite(11, LOW);
  //digitalWrite(12, HIGH);
  //delay(300);
  //digitalWrite(12, LOW);
}
```

Appendix C – Sketch for "7-Segment Driver" Project

```
// Sketch:  project_7segment.ino
//
// Single die displayed on 7-segment LED;

// Define the microcontroller pins which control the individual segments
// of the 7-segment LED
int segmentPins[] = {2, 3, 4, 5, 6, 7, 8};
// Define the pin connected to the push button
int buttonPin = 12;

// a 2D array (a matrix) to store hard coded control signals
// for the 7-segment LED; these codes make for the 7-segment LED
// to display the desired number;
byte digits[10][7] = {
  //a  b  c  d  e  f  g
  { 0, 0, 0, 0, 0, 0, 1 },  // 0
  { 1, 0, 0, 1, 1, 1, 1 },  // 1
  { 0, 0, 1, 0, 0, 1, 0 },  // 2
  { 0, 0, 0, 0, 1, 1, 0 },  // 3
  { 1, 0, 0, 1, 1, 0, 0 },  // 4
  { 0, 1, 0, 0, 1, 0, 0 },  // 5
  { 0, 1, 0, 0, 0, 0, 0 },  // 6
  { 0, 0, 0, 1, 1, 1, 1 },  // 7
  { 0, 0, 0, 0, 0, 0, 0 },  // 8
  { 0, 0, 0, 0, 1, 0, 0 }   // 9
};

void setup()
{
  // set microcontroller's pins to output mode; use a for loop;
  for (int i=0; i < 7; i++)
  {
    pinMode(segmentPins[i], OUTPUT);
  }
  pinMode(buttonPin, INPUT);
}

void loop()
{
  // declare a variable, which will store the randomly generated
  // value of the tossed die;
  static int die_value;
  if (digitalRead(buttonPin)) // "toss" die each time button is pushed;
  {
    die_value = random(1,7);
  }
  // call function setSegments(.) to set the value of each of the
  // output pins that drive the 7-segment LED accordingly, so that
  // the generated number is actually displayed;
  setSegments(die_value);
}

void setSegments(int n)
{
  for (int i=0; i < 7; i++)
  {
    digitalWrite(segmentPins[i],!digits[n][i]);
  }
}
```