Interactive Devices and Design
Yating Zhan (yz2237)

Part A. Revisiting Blink

1. Blinking LEDs with Arduino

Code:

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin 9 as an output.
  pinMode(9, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(9, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(9, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

a. What line(s) of code do you need to change to make the LED blink (like, at all)?

I changed three lines of code and changed “LED_BUILTIN” to pin 9.

b. What line(s) of code do you need to change to change the rate of blinking?

I can change the input for delay function to change the rate of blinking. For example, change delay(1000) to delay(500) will make the LED blink faster.

c. What circuit element would you want to add to protect the board and external LED?

A resistor with proper value should be added.

2. Digitally toggle LEDs on and off using the Arduino

a. Which lines do you need to modify to correspond with your button and LED pins?

I have to change the ledPin to 13, the buttonPin to 2, and change all LED_BUILTIN to pin 9.

b. Modify the code or the circuit so that the LED lights only while the button is depressed. Include your code in your lab write-up.

```
// constants won't change. They're used here to set pin numbers:
const int buttonPin = 2;     // the number of the pushbutton pin
const int ledPin =  13;      // the number of the LED pin

// variables will change:
int buttonState = 0;         // variable for reading the pushbutton status

void setup() {
  // initialize the LED pin as an output:
  pinMode(9, OUTPUT);
```

```
  // initialize the pushbutton pin as an input:
  pinMode(buttonPin, INPUT);
}

void loop() {
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed. If it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(9, HIGH);
  } else {
    // turn LED off:
    digitalWrite(9, LOW);
  }
}
```

3. Fading LEDs on and off using Arduino

Code:

```
int ledPin = 9;    // LED connected to digital pin 9
void setup() {
  // nothing happens in setup
}

void loop() {
  // fade in from min to max in increments of 5 points:
  for (int fadeValue = 0 ; fadeValue <= 255; fadeValue += 5) {
    // sets the value (range from 0 to 255):
    analogWrite(ledPin, fadeValue);
    // wait for 30 milliseconds to see the dimming effect
    delay(30);
  }

  // fade out from max to min in increments of 5 points:
  for (int fadeValue = 255 ; fadeValue >= 0; fadeValue -= 5) {
    // sets the value (range from 0 to 255):
    analogWrite(ledPin, fadeValue);
    // wait for 30 milliseconds to see the dimming effect
    delay(30);
  }
}
```

a. Which line(s) of code do you need to modify to correspond with your LED pin?

0, because the original code have ledPin set to 9, which is my setup for the LED.

b. How would you change the rate of fading?
I can change the rate of fading by changing delay values.
c. (Extra) Since the human eye doesn't see increases in brightness linearly and the diode brightness is also nonlinear with voltage, how could you change the code to make the light appear to fade linearly?
Change the code so that the fadeValue increases and decreases nonlinearly.

Part B. Advanced Inputs

1. Potentiometer

a. Post a copy of your new code in your lab write-up.

```
int sensorPin = A0;    // select the input pin for the potentiometer
int ledPin = 9;      // select the pin for the LED
int sensorValue;  // variable to store the value coming from the sensor

void setup() {
  // declare the ledPin as an OUTPUT:
  pinMode(ledPin, OUTPUT);
  pinMode(sensorPin, INPUT);
  Serial.begin(9600);
}

void loop() {
  // read the value from the sensor:
  sensorValue = analogRead(sensorPin);
  Serial.println(sensorValue);
  // turn the ledPin on
  analogWrite(ledPin, map(sensorValue, 0, 1023, 0, 255));
  delay(30);
}
```

2. Force Sensitive Sensor

a. What resistance values do you see from your force sensor?
Values from 0 to 1023.
b. What kind of relationship does the resistance have as a function of the force applied? (e.g., linear?)
Linear relationship, the lower the force, the higher the frequency.
c. Can you change the LED fading code values so that you get the full range of output voltages from the LED when using your FSR?
Yes, by mapping the sensor value from (0,1023) to (0,255).

Part C. Writing to the LCD

a. What voltage level do you need to power your display?
Power Supply Voltage : 5.0 V
b. What voltage level do you need to power the display backlight?

3.2V

b. What was one mistake you made when wiring up the display? How did you fix it?

I wired pin11 and pin12 to R/W and RS by mistake and the LCD shown digital blocks instead of text. I fixed it by made the two pins connected to correct corresponding ports on LCD board.

c. What line of code do you need to change to make it flash your name instead of "Hello World"?

Instead of lcd.print("Hello World!"), change it to lcd.print("Yating!");

d. Include a copy of your Lowly Multimeter code in your lab write-up.

```
#include <LiquidCrystal.h>

// initialize the library by associating any needed LCD interface pin
// with the arduino pin number it is connected to
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
int readPin = A0;
int readValue;
void setup() {
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  // Print a message to the LCD.
  lcd.print("Yating!");
}

void loop() {
  // set the cursor to column 0, line 1
  // (note: line 1 is the second row, since counting begins with 0):
  lcd.setCursor(0, 1);
  readValue = analogRead(readPin);
  lcd.print(readValue);
}
```

e. Include a copy of your FSR thumb wrestling code in your lab write-up.

```
#include <LiquidCrystal.h>
// initialize the library by associating any needed LCD interface pin
// with the arduino pin number it is connected to
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
int player1 = A0;
int player2 = A1;
int player1_score;
int player2_score;
void setup() {
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  // Print a message to the LCD.
```

```
  lcd.print("Winner is:");
}
void loop() {
  // set the cursor to column 0, line 1
  // (note: line 1 is the second row, since counting begins with 0):
  lcd.setCursor(0, 1);
  player1_score = analogRead(player1);
  player2_score = analogRead(player2);
  if (player1_score > player2_score){
    lcd.print("Player 1");
  }else if (player1_score < player2_score) {
    lcd.print("Player 2");
  }else {
    lcd.print("Tie!");
  }
  delay(100);
}
```

Part D. Timer

Code:

```
#include <LiquidCrystal.h>
// initialize the library by associating any needed LCD interface pin
// with the arduino pin number it is connected to
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
int pin = A0;
int pinRead;

void setup() {
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  // Print a message to the LCD.
  lcd.print("Time remain:");
  pinRead = analogRead(pin);
}

void loop() {
  // set the cursor to column 0, line 1
  // (note: line 1 is the second row, since counting begins with 0):
  lcd.setCursor(0, 1);
  lcd.print(pinRead);
  pinRead -= 1;
  delay(1000);
```

```
    if (pinRead <=0) {
      pinRead = analogRead(pin);
    }
  }
```

a. Make a short video showing how your timer works, and what happens when time is up!
https://youtu.be/PbBfb9BmFeY
b. Post a link to the completed lab report to the class Slack.