Minor Project Evidence Guide

Oanh PHAM
1014309

Aim
The aim of our minor project was to create a creative response to Microcontroller systems.

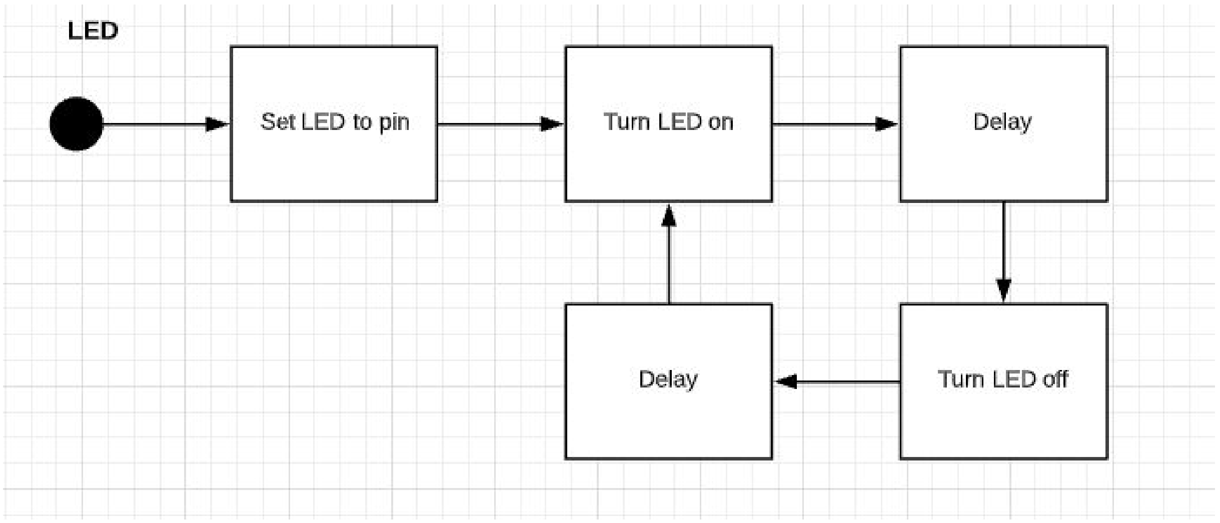
Unit Goals

- Demonstrate an understanding of microcontrollers and interfacing electronic hardware as it relates to a Microcontroller system
- Explore and examine the computer programming to effectively control Microcontroller systems
- Demonstrate an understanding of the Programming Design Cycle as it relates to Microcontroller Systems

Our Project
For this assignment, I collaborated with two other classmates and recreated an alarm system that will be used in a prison to indicate when someone has either broken in or out to the prison or when the system has been hacked. To recreate this system, we had to identify the individual components of the alarm with each group member researching one of the following components listed below.

- Flashing lights
- Sound of the alarm
- Opening and closing of doors

Flashing Lights
3mm Red LED
Diagram:



Code:

```
int led = 13;

void setup( )  {
     pinMode(led, OUTPUT);
     Serial.begin(9600);
}

void loop( )  {
     digitalWrite(led, HIGH);
     delay(100);
     digitalWrite(led, LOW);
     delay(100);
}
```

RGB LEDs

For the flashing lights, it was determined that a 3mm red LED would be used as it had a greater intensity. Whilst using 3mm red LED would work perfectly for this project, it was also recommended that I looked into other alternatives such as RGB LEDs.

Summary of findings

- 'RGB' is the acronym or abbreviated term for 'red, green and blue'
- There are two common types of common RGB LEDs, anode and cathode
    - Both have four pins, one for each colour and one for a shared positive or negative connection
        - Anodes have three LEDs that all share a positive connection
        - Cathodes have three LEDs that all share a negative connection
- Colours other than red, green or blue can be created by adjusting the intensity of the red, green and blue hues

Eg.

```
void loop() {
  setColor(255, 0, 0); // Red Color
  delay(1000);
  setColor(0, 255, 0); // Green Color
  delay(1000);
  setColor(0, 0, 255); // Blue Color
  delay(1000);
  setColor(255, 255, 255); // White Color
  delay(1000);
  setColor(170, 0, 255); // Purple Color
  delay(1000);
}

void setColor(int redValue, int greenValue, int blueValue) {
```

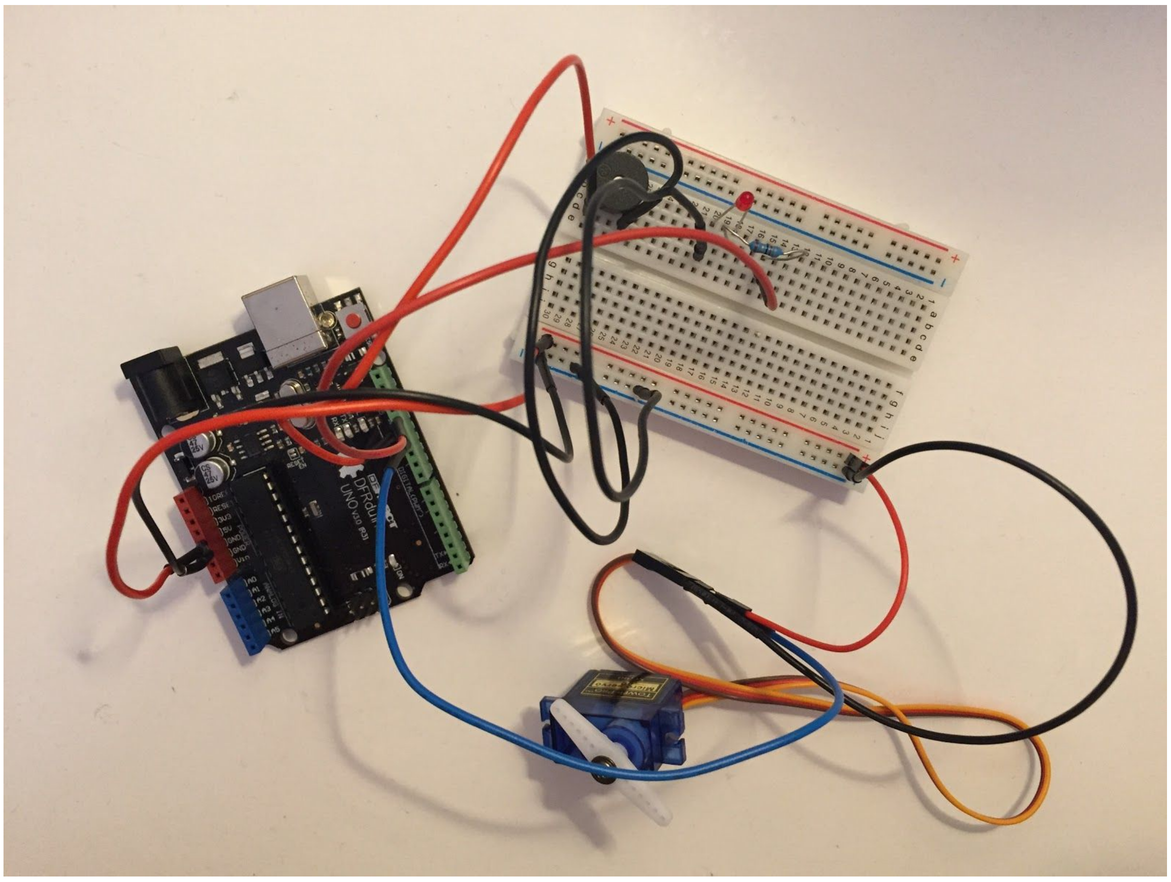
```
analogWrite(redPin, redValue);
analogWrite(greenPin, greenValue);
analogWrite(bluePin, blueValue);
}
```

In this piece of coding, the author has created a function that would set and change the intensity of the red, green and blue lights. This function is then used in the loop(). As the author as annotated, each colour is created differently. We know from colour theory that the three primary colours of light; red, green and blue; create the colour white. This is shown in the program with the author setting each colour pin to emit the pulse of 225.

References

- https://randomnerdtutorials.com/electronics-basics-how-do-rgb-leds-work/
- https://howtomechatronics.com/tutorials/arduino/how-to-use-a-rgb-led-with-arduino/
- https://www.youtube.com/watch?v=wqzfbImsrPE
- https://www.teachmemicro.com/arduino-rgb-led-tutorial/
- https://www.instructables.com/id/How-to-Blink-LED-Using-Arduino/

Physical Build



Final Code

```
#include <Servo.h>
Servo servo;
int angle = 10;

int ledPin =  13;
int buzzerPin = 12;
int ledState = LOW;
unsigned long previousMillis = 0;
long OnTime = 250;
long OffTime = 750;

void setup() {
  // set the digital pin as output:
  pinMode(ledPin, OUTPUT);
  pinMode(buzzerPin, OUTPUT);
  servo.attach(11);
}

void loop(){
  unsigned long currentMillis = millis();

  if((ledState == HIGH) && (currentMillis - previousMillis >= OnTime)){
    ledState = LOW;
    previousMillis = currentMillis;
    for(angle = 180; angle > 10; angle--) {
      servo.write(angle);
      noTone(buzzerPin);
      digitalWrite(ledPin, ledState);
    }
  }
  else if((ledState == LOW) && (currentMillis - previousMillis >= OffTime)){
    ledState = HIGH;
    previousMillis = currentMillis;
    for(angle = 10; angle < 180; angle++) {
      servo.write(angle);
      tone(buzzerPin, 1000);
      digitalWrite(ledPin, ledState);
    }
  }
}
```

References

- https://www.instructables.com/id/Arduino-Buzzer-With-LDR-and-LED/
- https://xavisolemora.wordpress.com/2014/04/30/arduino-led-piezo-servo-pez-dispenser/

- https://arduino.stackexchange.com/questions/35177/stop-leds-from-blinking
- https://learn.adafruit.com/multi-tasking-the-arduino-part-1/using-millis-for-timing