

# Drexel University

**To:** Dr. Christopher Peters

**From:** Zheren Gu

**cc:** Amirhosein Chahe

**Date:** 4/17/2023

**Re:** Timers and Interrupts

---

## Purpose

The purpose of this lab was to introduce us to Timers and Interrupts for our project and create the groundwork for the DigitalPin class that we will be using later in the term.

## Discussion

The main parts of this lab were to get the input from serial monitor and then compare to the correct pin to see how the LEDs should react. This was done by taking from serial monitor as a integer and then using modulus to separate each value into 4 values to compare to the 4 values from the correct password to see if it matches, and if it does not the rate of the LEDS will increase. This will be repeated 5 times and if the correct value is not found all the LEDs turn on and the system is considered locked.

## Recommendation

If I were to improve on what I did I would start by using the method of storing the possible pins that was shown in class that had all the possible combinations in arrays instead of just using the ones that were relevant to my breadboard setup. I also would have made better use of the Safe class as I feel like I didn't know what to do with that class.

```

#include <Arduino.h>
#include <Safe.h>
#include <DigitalPin.h>

#define RED1 11
#define RED2 44
#define RED3 6
#define RED4 5
DigitalPin LED[4] = {DigitalPin(RED1), DigitalPin(RED2), DigitalPin(RED3),
DigitalPin(RED4)};
int password = 1234;
Safe Pass(password);
// int password = random(10000);
int attempts = 99;
void setup()
{
    Serial.begin(9600);
    randomSeed(analogRead(0));
    noInterrupts();
    for (int x = 0; x < 4; x++)
    {
        LED[x].set_TCCRA(0);
        LED[x].set_TCCRB(0);
        LED[x].set_TCNT(0);
        LED[x].set_OCR(31248);
        LED[x].set_TIMSK(1);
    }
    interrupts();

    // put your setup code here, to run once:
}

void loop()
{
    if (attempts == 99)
    {
        Serial.print("This is the code");
        Pass.displayCode();
        Serial.println("Input Guess");
        attempts = 0;
    }
    if (attempts < 5)
    {

```

```

while (Serial.available() == 4)
{

    int Val = Serial.parseInt();
    int copy = Pass.getCode();

    Serial.println(Val, DEC);
    int Val4 = Val % 10;
    Val /= 10;
    int Val3 = Val % 10;
    Val /= 10;
    int Val2 = Val % 10;
    Val /= 10;
    int Val1 = Val % 10;
    int password4 = copy % 10;
    copy /= 10;
    int password3 = copy % 10;
    copy /= 10;
    int password2 = copy % 10;
    copy /= 10;
    int password1 = copy % 10;
    // Serial.println(Val4);
    // Serial.println(password4);

    if (Val4 == password4)
    {
        LED[3].set_TIMSK(0);
        LED[3].Off();
    }
    if (Val3 == password3)
    {
        LED[2].set_TIMSK(0);
        LED[2].Off();
    }
    if (Val2 == password2)
    {
        LED[1].set_TIMSK(0);
        LED[1].Off();
    }

    if (Val1 == password1)
    {
        LED[0].set_TIMSK(0);
    }
}

```

```

        LED[0].Off();
    }

    attempts++;
    Serial.println(attempts);
    for (int i = 0; i < 4; i++)
    {
        LED[i].factor_OCR(2);
    }
    Serial.println("Input Guess");
    if (attempts == 5)
    {
        Serial.println("TOO MANY ATTEMPS");
        for (int k = 0; k < 4; k++)
        {
            LED[k].set_TIMSK(0);
            LED[k].On();
        }
    }
}

ISR(TIMER1_COMPA_vect)
{
    digitalWrite(REDD1, !digitalRead(REDD1));
}

ISR(TIMER5_COMPA_vect)
{
    digitalWrite(REDD2, !digitalRead(REDD2));
}

ISR(TIMER4_COMPA_vect)
{
    digitalWrite(REDD3, !digitalRead(REDD3));
}

ISR(TIMER3_COMPA_vect)
{
    digitalWrite(REDD4, !digitalRead(REDD4));
}

```

```

#include <Arduino.h>
#include "Safe.h"

Safe::Safe(int code)
{
    _code = code;
}

int Safe::getCode(){
    return _code;
}

void Safe::displayCode(){
    Serial.println(_code);
}

```

```

#ifndef Safe_h
#define Safe_h

#include <Arduino.h>

class Safe
{
public:
    Safe(int code);
    int getCode();
    void displayCode();

private:
    int _code;
};

#endif

```

```

#ifndef DigitalPin_h
#define DigitalPin_h

```

```

#include <Arduino.h>

class DigitalPin
{
public:
    DigitalPin(int pin);
    void On();
    void Off();
    void set_TCCRA(int num);
    void set_TCCRB(int num);
    void set_TCNT(int num);
    void set_OCR(int num);
    void factor_OCR(int factor);
    void set_TIMSK(int num);
private:
    int _pin;
};

#endif
#include <Arduino.h>
#include "DigitalPin.h"

DigitalPin::DigitalPin(int pin)
{
    pinMode(pin, OUTPUT);
    _pin = pin;
}

void DigitalPin::On()
{
    digitalWrite(_pin, HIGH);
}

void DigitalPin::Off()
{
    digitalWrite(_pin, LOW);
}

void DigitalPin::set_TCCRA(int num)
{
    if (_pin == 11)
    { // timer 1
        TCCR1A = num;
    }
    else if (_pin == 44)

```

```

    { // timer 5
        TCCR5A = num;
    }
    else if (_pin == 6)
    { // timer 4
        TCCR4A = num;
    }
    else if (_pin == 5)
    { // timer 3
        TCCR3A = num;
    }
}

void DigitalPin::set_TCCRB(int num)
{
    if (_pin == 11)
    { // timer 1
        TCCR1B = num;
        TCCR1B |= (1 << WGM12);
        TCCR1B |= (1 << CS12);
    }
    else if (_pin == 44)
    { // timer 5
        TCCR5B = num;
        TCCR5B |= (1 << WGM52);
        TCCR5B |= (1 << CS52);
    }
    else if (_pin == 6)
    { // timer 4
        TCCR4B = num;
        TCCR4B |= (1 << WGM42);
        TCCR4B |= (1 << CS42);
    }
    else if (_pin == 5)
    { // timer 3
        TCCR3B = num;
        TCCR3B |= (1 << WGM32);
        TCCR3B |= (1 << CS32) | (0 << CS31) | (0 << CS30);
    }
}

void DigitalPin::set_TCNT(int num)
{
    if (_pin == 11)
    { // timer 1

```

```

        TCNT1 = num;
    }
    else if (_pin == 44)
    { // timer 5
        TCNT5 = num;
    }
    else if (_pin == 6)
    { // timer 4
        TCNT4 = num;
    }
    else if (_pin == 5)
    { // timer 3
        TCNT3 = num;
    }
}
void DigitalPin::set_OCR(int num)
{
    if (_pin == 11)
    { // timer 1
        OCR1A = num;
    }
    else if (_pin == 44)
    { // timer 5
        OCR5A = num;
    }
    else if (_pin == 6)
    { // timer 4
        OCR4A = num;
    }
    else if (_pin == 5)
    { // timer 3
        OCR3A = num;
    }
}
void DigitalPin::factor_OCR(int factor)
{
    if (_pin == 11)
    { // timer 1
        OCR1A = OCR1A / factor;
    }
    else if (_pin == 44)
    { // timer 5
        OCR5A = OCR5A / factor;
    }
}

```



```

else if (_pin == 6)
{ // timer 4
  OCR4A = OCR4A / factor;
}
else if (_pin == 5)
{ // timer 3
  OCR3A = OCR3A / factor;
}
}
void DigitalPin::set_TIMSK(int num)
{
  if (_pin == 11)
  { // timer 1
    if (num == 0)
    {
      TIMSK1 = 0;
    }
    else
    {
      TIMSK1 = 0;
      TIMSK1 |= (1 << OCIE1A);
    }
  }
  else if (_pin == 44)
  { // timer 5
    if (num == 0)
    {
      TIMSK5 = 0;
    }
    else
    {
      TIMSK5 = 0;
      TIMSK5 |= (1 << OCIE5A);
    }
  }
  else if (_pin == 6)
  { // timer 4
    if (num == 0)
    {
      TIMSK4 = 0;
    }
    else
    {
      TIMSK4 = 0;
    }
  }
}

```

```

        TIMSK4 |= (1 << OCIE4A);
    }
}
else if (_pin == 5)
{ // timer 3
    if (num == 0)
    {
        TIMSK3 = 0;
    }
    else
    {
        TIMSK3 = 0;
        TIMSK3 |= (1 << OCIE3A);
    }
}
}
}

```