

**MATERIA:**

Sistemas Programables

**CARRERA:**

Ingeniería en Sistemas Computacionales

**PRESENTA:**

Francisco Martínez González

**NOMBRE DEL MAESTRO:**

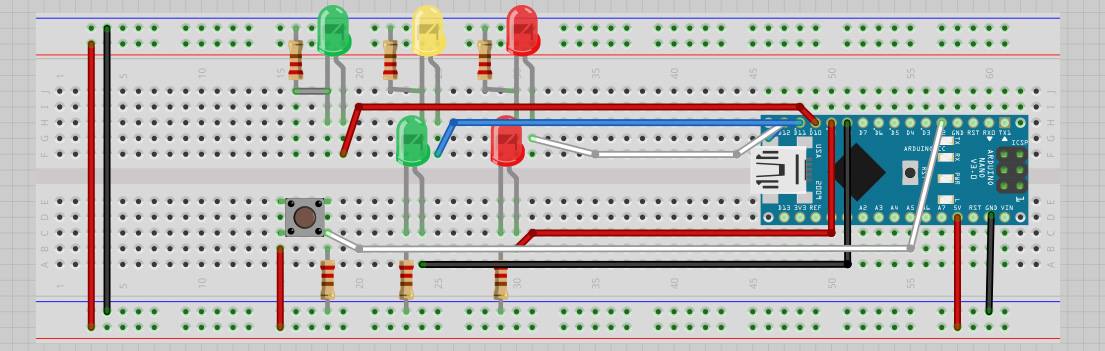
Ing. Levy Rojas Carlos Rafael

**PRACTICA:**

Semáforo con Arduino

**LEÓN, GUANAJUATO Periodo: Enero – Junio 2018**

**DIAGRAMA DE CONFIGURACIÓN**

****

**CÓDIGO FUENTE**

const int CAR\_RED = 12; //red light of vehicular traffic initialized in 12

const int CAR\_YELLOW = 11; //yellow light of vehicular traffic initialized in 11

const int CAR\_GREEN = 10; //green light of vehicular traffic initialized in 10

const int PED\_RED = 9; // red light of the pedestrian traffic initialized in 9

const int PED\_GREEN = 8; // green light of the pedestrian traffic initialized in 8

const int BUTTON = 2; // button pin initialized in 2

int crossTime = 10000; //time alloyoud to cross

unsigned long changeTime; //time since BUTTON pressed

int state = LOW; //if is push button

/\*

The method setup() configures the pin of the lights of the traffic lights to behave like an exit with the method pinMode()

Also the method digitalWrite() Write a HIGH or a LOW value to a digital pin.

\*/

void setup() {

pinMode(CAR\_RED, OUTPUT);

pinMode(CAR\_YELLOW, OUTPUT);

pinMode(CAR\_GREEN, OUTPUT);

pinMode(PED\_RED, OUTPUT);

pinMode(PED\_GREEN, OUTPUT);

pinMode(BUTTON, INPUT); // button on pin 2

digitalWrite(CAR\_GREEN, HIGH);

digitalWrite(PED\_RED, HIGH);

}

/\*

The method loop()has two conditions, one that verifies if the button is pressed, if it is true the variable "state" takes the value of "high",

the second validation is responsible for verifying if the variable "satate" has the value of "high" and if the time that has passed since

the button was pressed in greater than 10 seconds

\*/

void loop() {

if (digitalRead(BUTTON)) //Change of state is button is pressed

state = HIGH;

if (state == HIGH && (millis() - changeTime) > crossTime) {

changeLight();

state = LOW;

}

}

/\*

The mothod changeLight is responsible for changing the lights of the traffic lights with the digitalWrite() method

\*/

void changeLight() {

digitalWrite(CAR\_GREEN, LOW); //green off

digitalWrite(CAR\_YELLOW, HIGH); //yellow on

delay(2000); //wait 2 seconds

digitalWrite(CAR\_YELLOW, LOW); //green off

digitalWrite(CAR\_RED, HIGH); //yellow on

delay(2000); //wait 2 seconds

digitalWrite(PED\_RED, LOW); //green off

digitalWrite(PED\_GREEN, HIGH); //yellow on

delay(crossTime); //wait for

for (int i = 0; i < 10; i++) {

digitalWrite(PED\_RED, HIGH); //Turn on red pedestrian traffic light

delay(250);//wait 250 milliseconds

digitalWrite(PED\_GREEN, LOW);//Turn off green pedestrian traffic light

delay(250);//wait 250 milliseconds

}

digitalWrite(PED\_RED, HIGH);//Turn on red pedestrian traffic light

delay(500);//wait 500 milliseconds

digitalWrite(CAR\_YELLOW, HIGH);//turn on the yellow light of the traffic lights of the vehicles

digitalWrite(CAR\_RED, LOW);//turn on the red light of the traffic lights of the vehicles

delay(1000);//wait 500 milliseconds

digitalWrite(CAR\_GREEN, HIGH);//turn on the green light of the traffic lights of the vehicles

digitalWrite(CAR\_YELLOW, LOW);//turn on the yellow light of the traffic lights of the vehicles

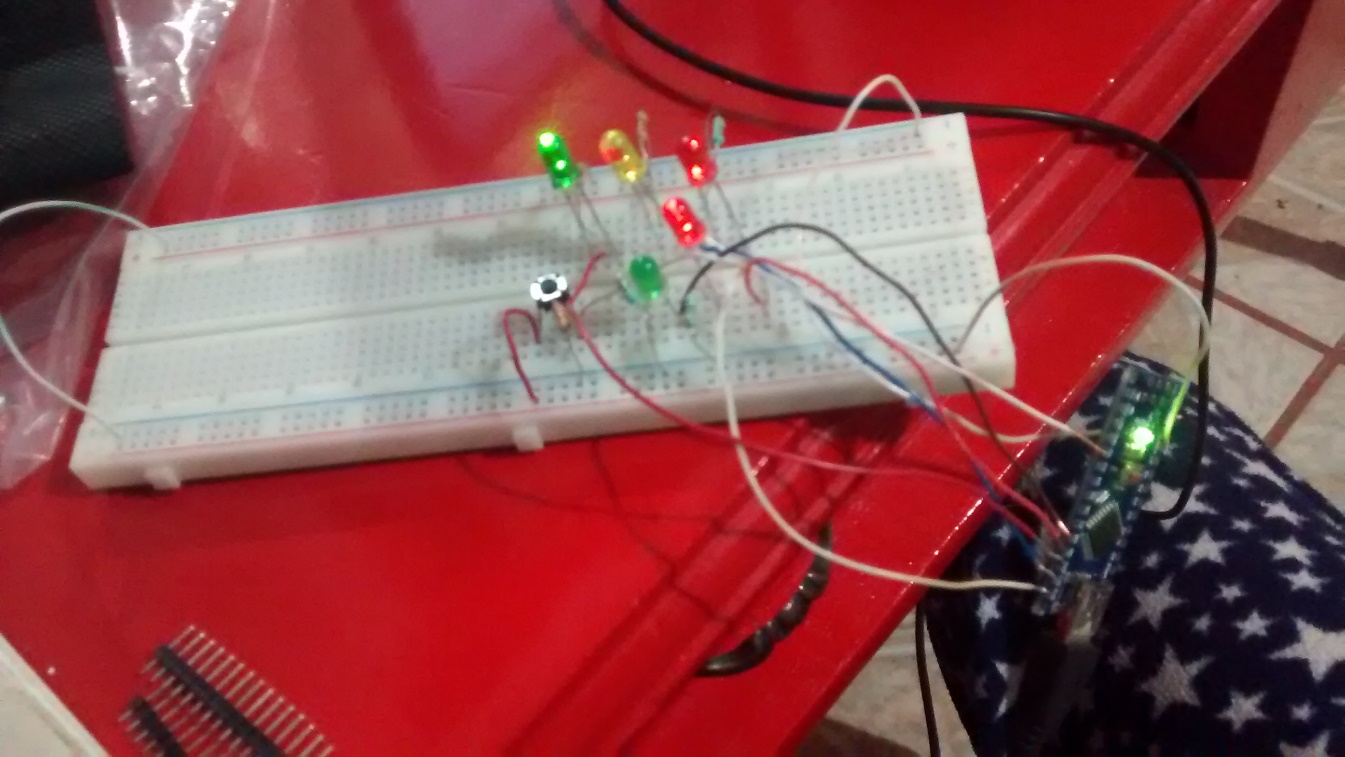
//record the time size last change of lights

changeTime = millis();

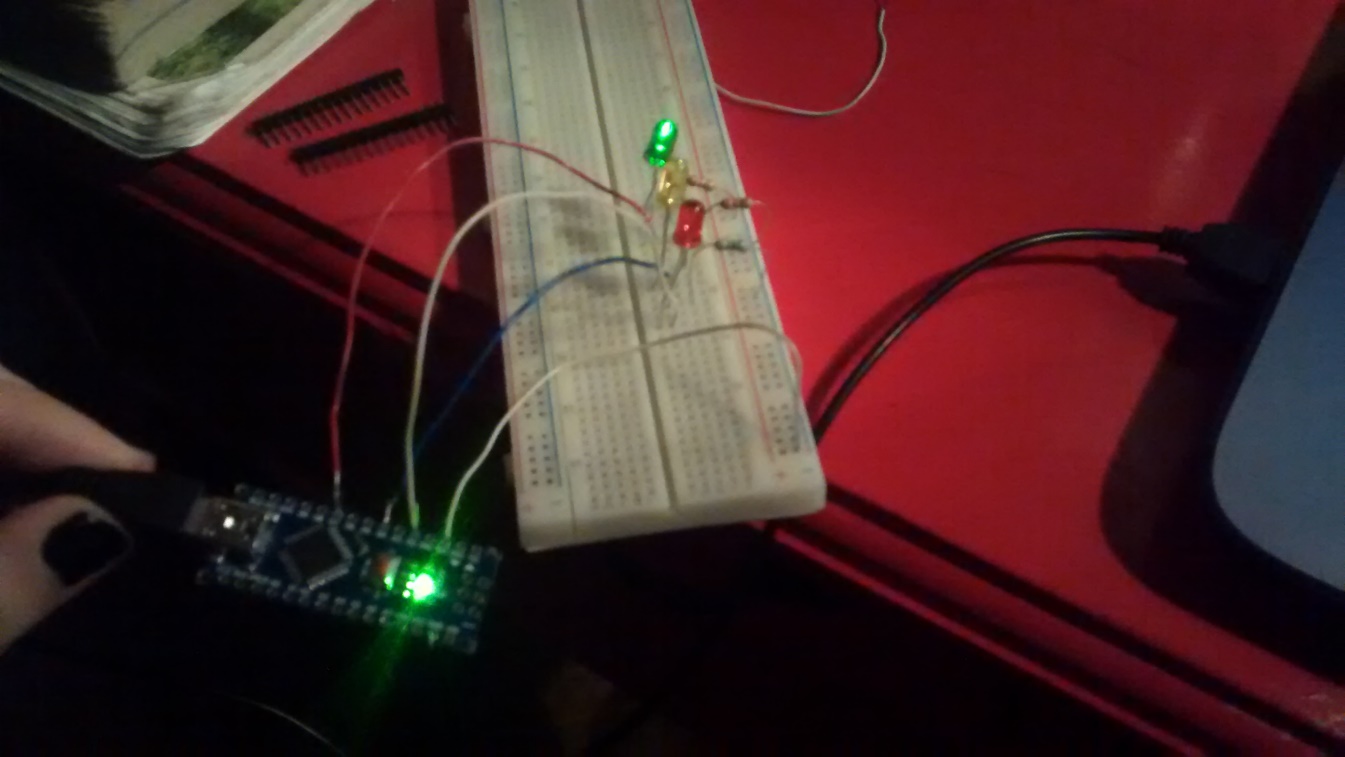
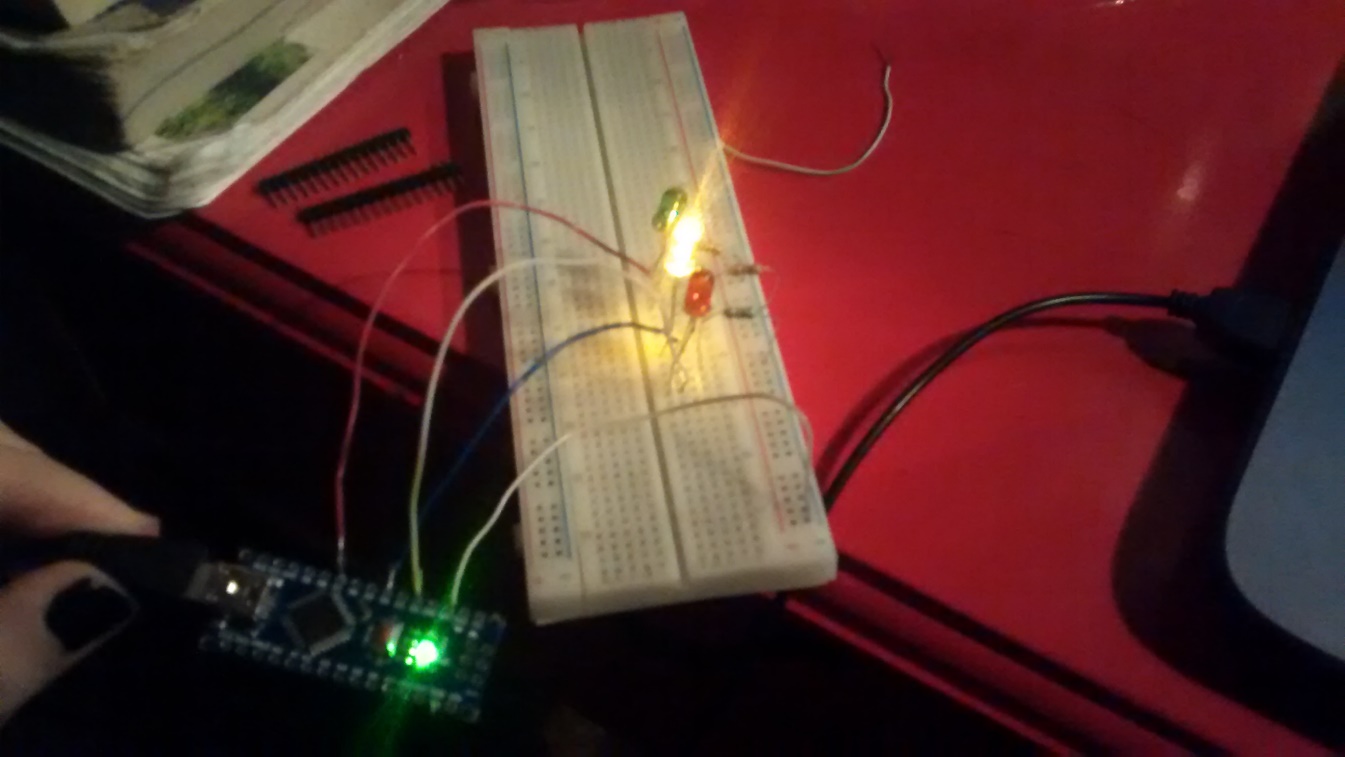
//then return to the my program loop

}

**RESULTADOS**

****

**L**

****

**CONCLUSIÓN**

En esta práctica pudimos implementar un semáforo vehicular y peatonal en base a la funcionalidad que se utiliza en el Reino Unido utilizando el ambiente de desarrollo de Arduino, el cual contiene instrucciones muy simples de utilizar y que nos permiten programar la funcionalidad en base a las necesidades que ocupemos. Al terminar de programar las instrucciones necesarias en el Arduino pudimos ensamblarlo en el Protoboard y probar que funciona el semáforo.