

# **INSTITUTO TECNOLÓGICO AUTÓNOMO DE MÉXICO**



---

Laboratorio de Principios de Mecatrónica

## **Práctica 2. Microcontroladores ( $\mu$ C) – Señales Digitales**

- Juan Pablo Macías Watson
- José Luis Gutiérrez Espinosa

Asignatura: Laboratorio de Principios de Mecatrónica

Docente: M.I. Sergio Hernández Sánchez

Grupo: 01

Semestre: Otoño 2022

## **1. Introducción**

El uso de microcontroladores nos permite atacar problemas que requieren o se facilitan a través del uso de la computación, sin ser tan caros como una PC.

Arduino nos proporciona las herramientas para resolver estos problemas con el uso de los microcontroladores.

En esta práctica utilizamos la IDE de Arduino junto con el microcontrolador Mega 2560 para controlar cómo y cuándo se prenden LED's dentro de un circuito.

## **2. Objetivos**

- Conocer las plataformas, herramientas y ambientes más comunes para el desarrollo de sistemas mecatrónicos basados en microcontroladores.
- Identificar los elementos básicos de la arquitectura de un microcontrolador digital.
- Identificar los módulos que componen a la tarjeta de desarrollo Arduino y sus interconexiones básicas.
- Realizar una comparativa entre las implementaciones en código de bajo y alto nivel, destacando las ventajas y pertinencia de cada una de las estrategias de programación de microcontroladores.

## **3. Marco Teórico**

Genere una breve investigación de los siguientes conceptos:

- Microcontrolador

Un microcontrolador es un circuito integrado programable, capaz de ejecutar las órdenes grabadas en su memoria. Está compuesto de varios bloques funcionales que cumplen una tarea específica. Un microcontrolador incluye en su interior las tres principales unidades funcionales de una computadora: unidad central de procesamiento, memoria y periféricos de entrada/salida.

- Placa de desarrollo Arduino MEGA

Arduino Mega es una placa de desarrollo open-source construida con un microcontrolador modelo Atmega2560 que posee pines de entradas y salidas. Esta tarjeta es programada en un entorno de desarrollo que implementa el lenguaje Processing/Wiring. Arduino puede utilizarse en el desarrollo de objetos interactivos autónomos o puede comunicarse a un PC a través del puerto serial (conversión con USB) utilizando lenguajes como C++, Ensamblador, entre otros.

- Características más importantes de la placa Arduino MEGA

ATmega2560 Processor

Up to 16 MIPS Throughput at 16MHz

256k bytes (of which 8k is used for the bootloader)

4k bytes EEPROM

8k bytes Internal SRAM

32 × 8 General Purpose Working Registers

Real Time Counter with Separate Oscillator

Four 8-bit PWM Channels

Four Programmable Serial USART

Controller/Peripheral SPI Serial Interface

- Configuración *Pull up* y *pull down*

#### Pull-up

Como su nombre indica esta resistencia tiene la función de “jalar” hacia “arriba”, lo que significa que polariza el voltaje hacia el voltaje de fuente (VDD) que puede ser +5V o +3.3V. De esta forma cuando el pulsador está abierto o en reposo, el voltaje en la entrada del Arduino siempre será de +5V. Las entradas del Arduino son de alta impedancia lo que significa que la corriente que circulará por esa línea sea mínima en el orden de los micro-amperios, por lo que el voltaje que “cae” en la resistencia pull-up es mínimo y tenemos casi el mismo voltaje de fuente en la entrada del Arduino.

#### Pull-down

De forma similar la resistencia pull-down “jala” el voltaje hacia “abajo” o “0V”. Cuando el pulsador está en reposo, el voltaje en la entrada del Arduino será 0V. Cuando presionamos el pulsador la corriente fluye de +5V por el pulsador hacia la resistencia y termina en 0V, de esa forma tenemos +5V en la entrada del Arduino.

## 4. Material y equipo utilizado

- a. 1 Arduino MEGA
- b. 1 cable USB A/B
- c. 1 protoboard grande
- d. 2 LED rojo
- e. 2 LED amarillo
- f. 2 LED verde
- g. 1 Push-Button
- h. 6 resistor  $220\ \Omega$
- i. 1 resistor  $10\ k\ \Omega$

## 5. Experimentos y simulaciones

### a. Actividad 1 – Blink

Para esta actividad, se busca encender y apagar un LED, el cual está conectado a una resistencia de  $220\ \Omega$  usando un Arduino MEGA, replique el circuito físico mostrado en la figura 1. Tenga en cuenta que en la figura se muestra utilizando una placa de desarrollo Arduino UNO.

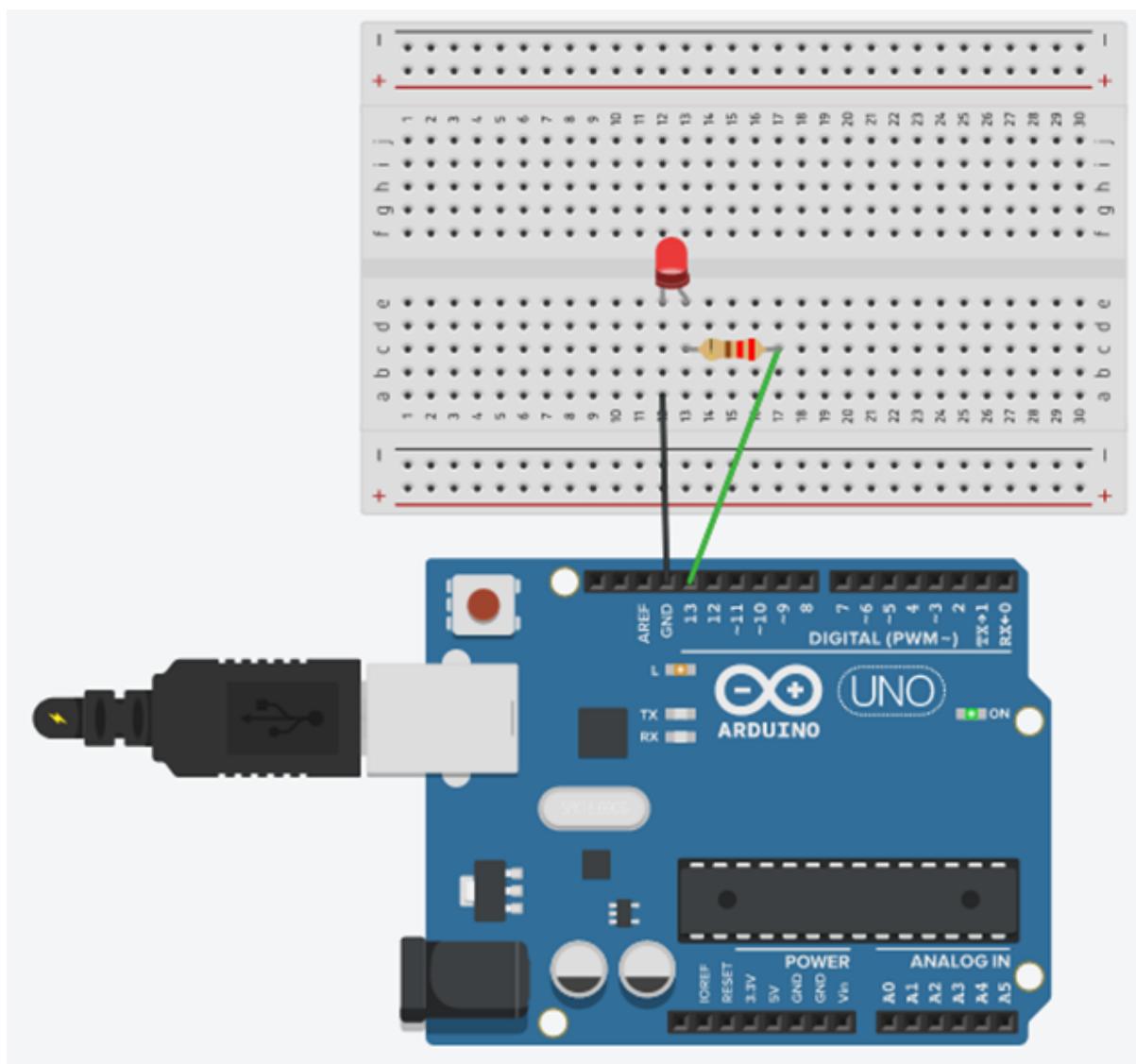


Figura 1. Circuito para probar el Blink.

A continuación, coloque una fotografía del circuito armado en la figura 2 donde se muestre el LED encendido.

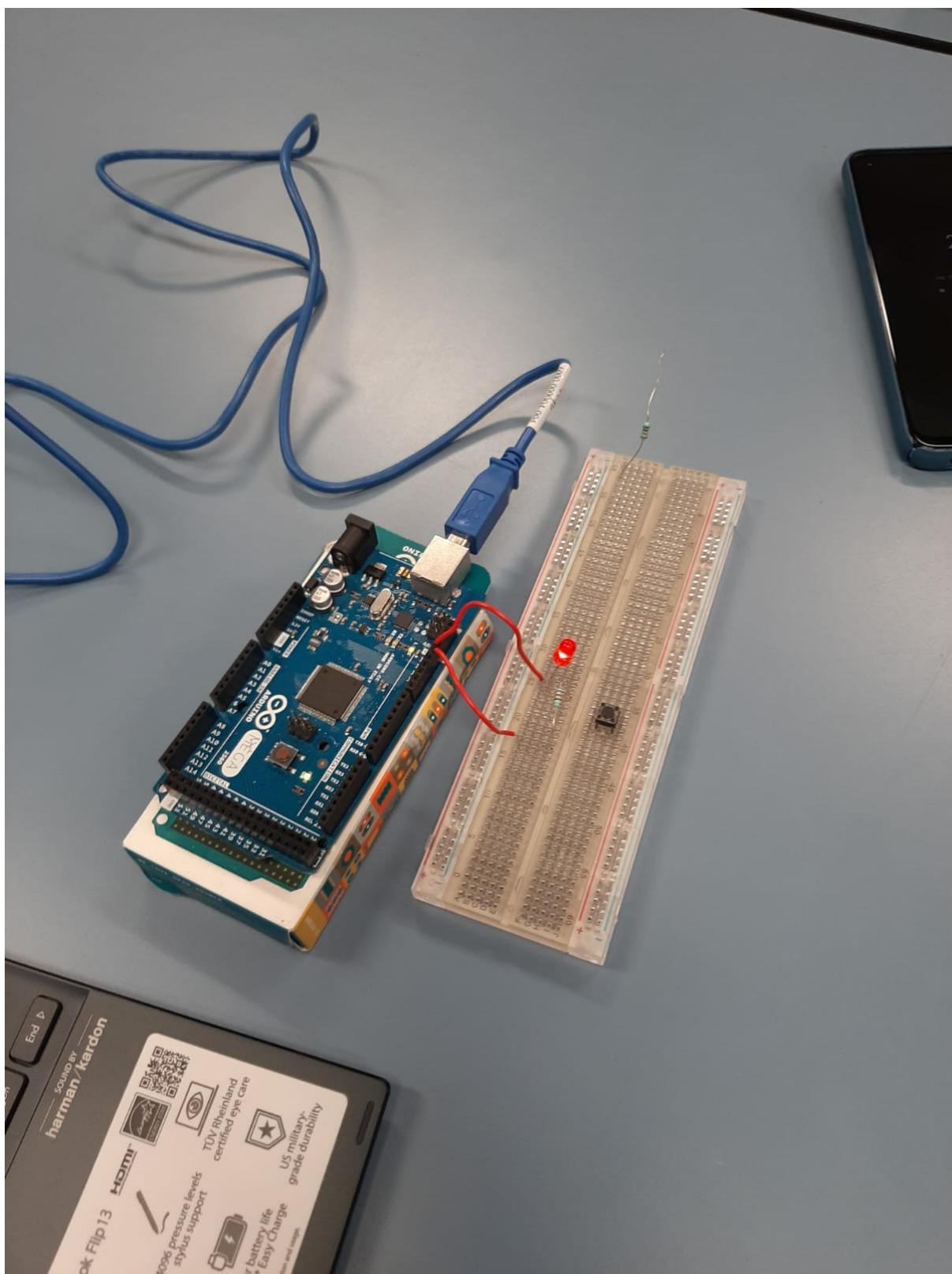


Figura 2. Circuito blink armado y funcionando.

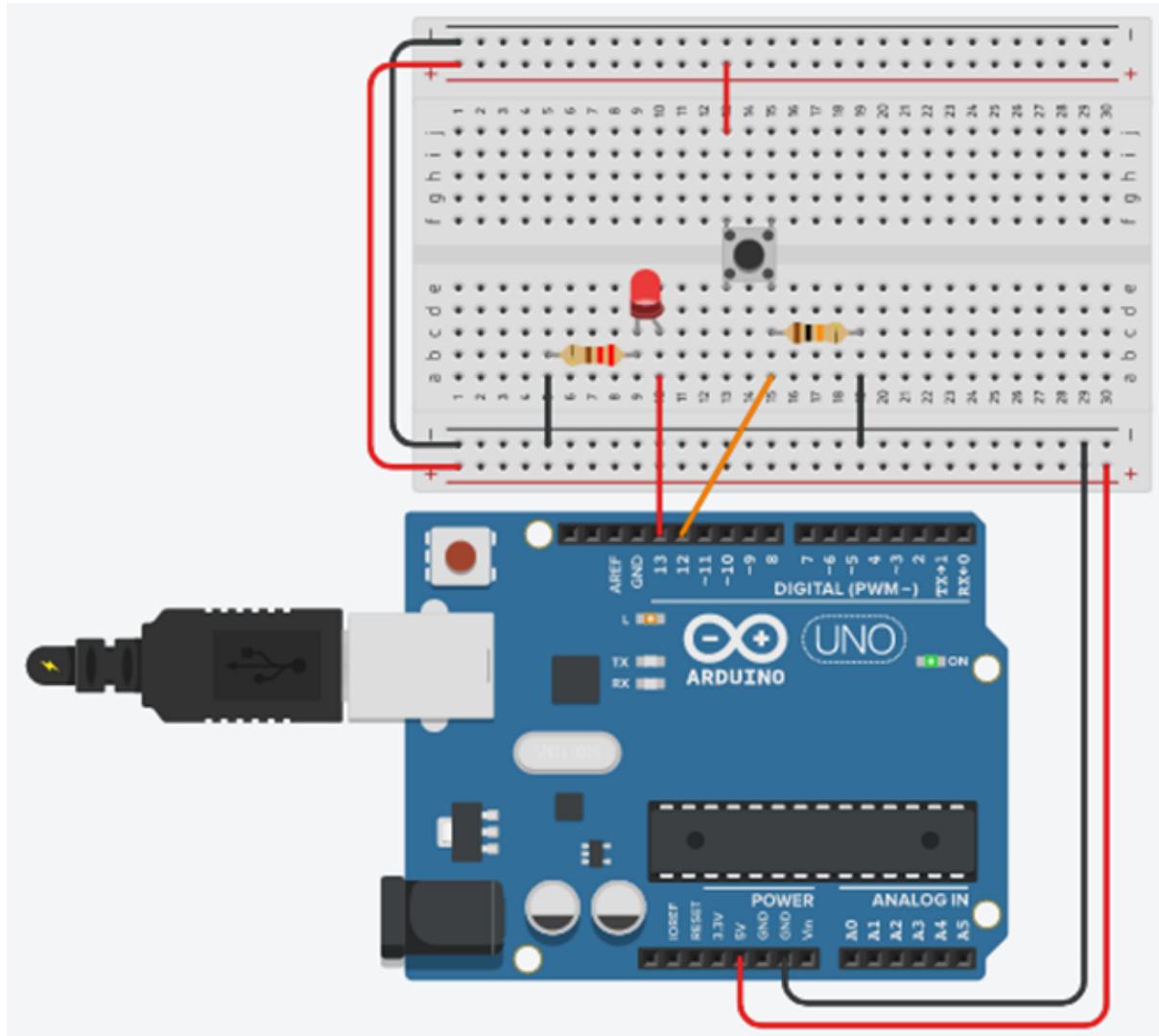
Enseguida, dentro del directorio **P\_Mecatrónica**, cree un directorio llamado Practica02 y guarde dentro el código generado cuyo nombre será **P2A1-Blink**, y actualice su repositorio tanto local como en línea de tal forma que el profesor pueda verlo y checar su código. Este

procedimiento descrito deberá hacerse para cada una de las actividades de cada práctica de laboratorio.

El código deberá encender un LED, dejarlo encendido por 1 segundo y posteriormente apagarlo por 1 segundo. Esto se deberá repetir indefinidamente. Deberá comentar cada línea de código de tal forma que explique su funcionamiento, recuerde que para comentar una línea se pondrá doble diagonal //.

### **b. Actividad 2 – Botón y Frecuencia**

Como segunda actividad, cree un nuevo sketch en el IDE de Arduino que deberá guardar como **P2A2-Boton-Freq** y guardar en su carpeta de Práctica02 de su repositorio. Se solicita que se conecte un *push-button* en la configuración *pull-down* utilizando una resistencia de 10 kΩ de tal manera que al ser presionado, envíe una señal de un uno lógico o de 5 V a la tarjeta Arduino y cuando no esté presionado, se envíe un cero lógico o 0 V. Utilice el diagrama de la figura 3 para poder armarlo.



*Figura 3.* Circuito para probar el push-button y frecuencia de LED.

Una vez armado el circuito, añada el código utilizado a su repositorio en Github, el cual deberá estar comentado en cada línea, de tal manera que cumpla la tabla mostrada a continuación y añada una fotografía de su circuito armado en la figura 4.

*Tabla 1.* Lógica que deberá seguir el LED.

Estado del Push-Button	Frecuencia de encendido del LED
LOW	1 Hz
HIGH	4 Hz

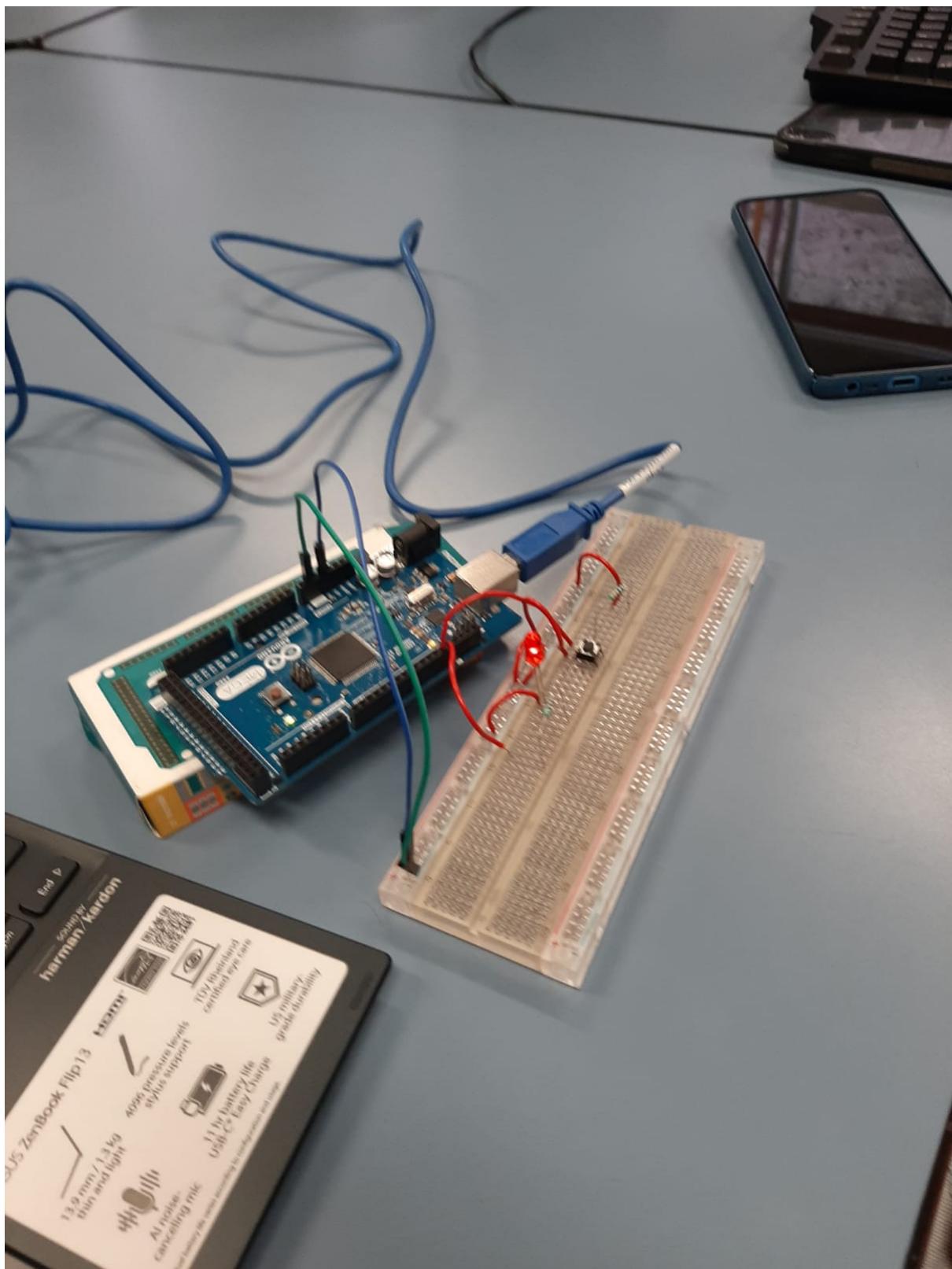


Figura 4. Circuito frecuencia real armado y funcionando.

### c. Actividad 3 – Semáforo

Finalmente, en esta última actividad conecte a su circuito 2 LEDs rojos, 2 LEDs amarillos y 2 verdes con sus respectivas resistencias de  $200\ \Omega$ , de forma que estén acomodados como dos configuraciones de semáforos. Conecte el ánodo del LED a alguno de los pines digitales de su Arduino MEGA, como se muestra en la figura 5.

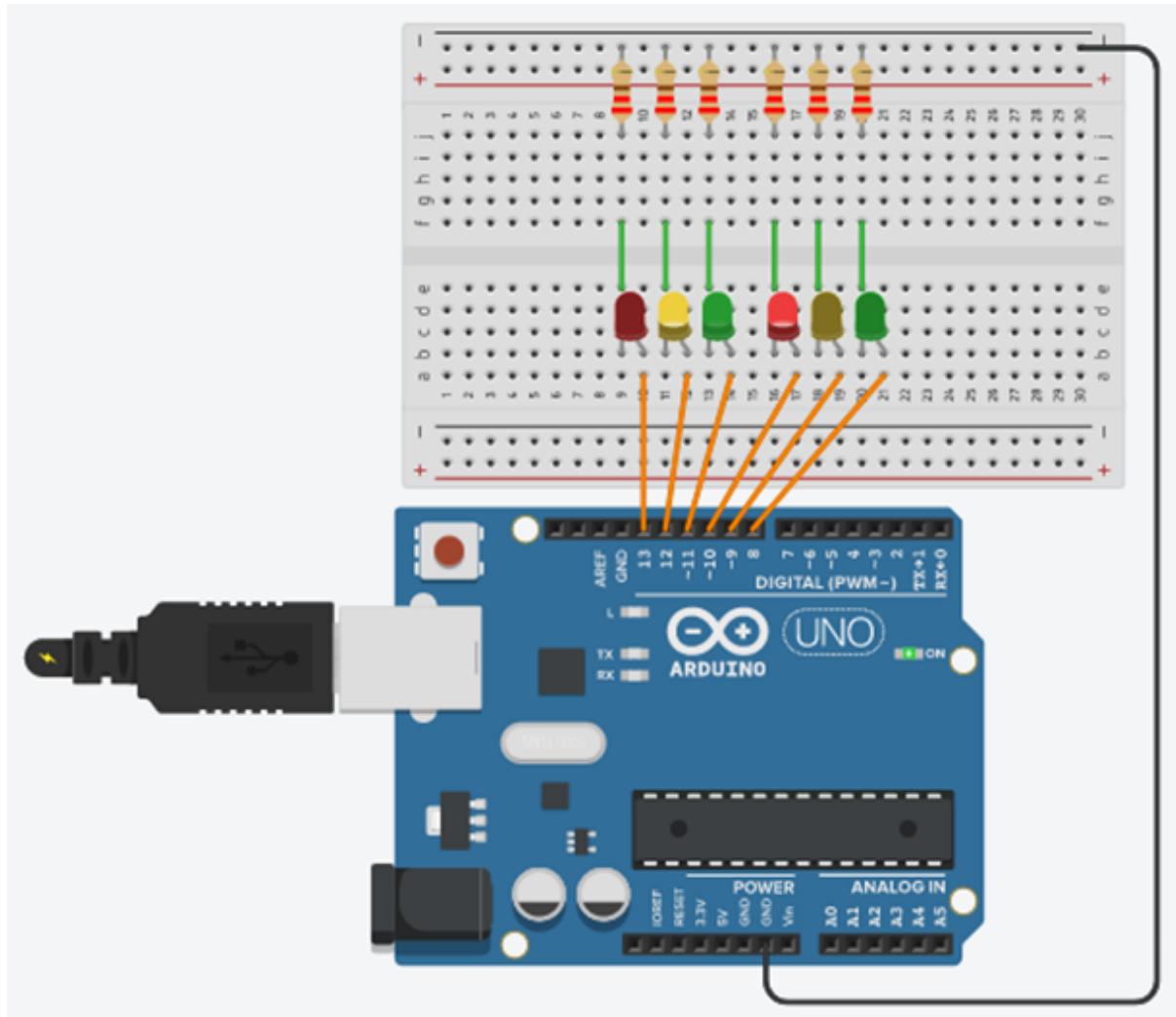


Figura 5. Circuito para probar el semáforo.

A continuación, genere el código necesario guardándolo como **P2A3-Semaforo**, donde deberá realizar lo la lógica descrita por la figura 6 para lograr el funcionamiento de 2 semáforos. Recuerde comentar su código.

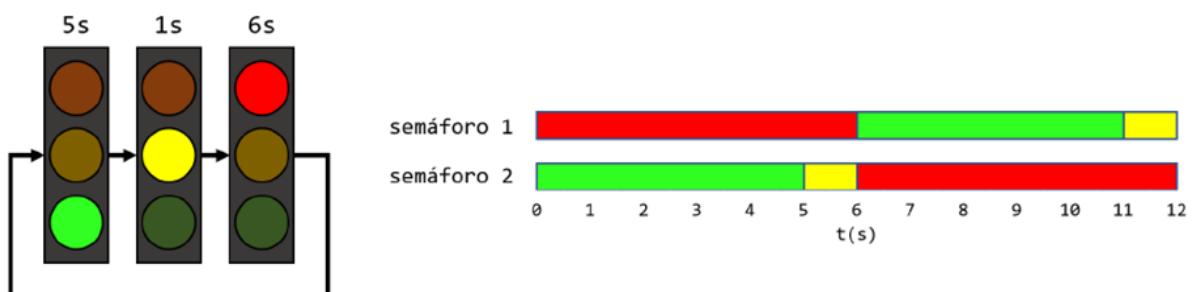


Figura 6. Lógica de encendido del semáforo.

A continuación, muestre una o más fotografías mostrando el circuito armado y funcionando según lo solicitado.

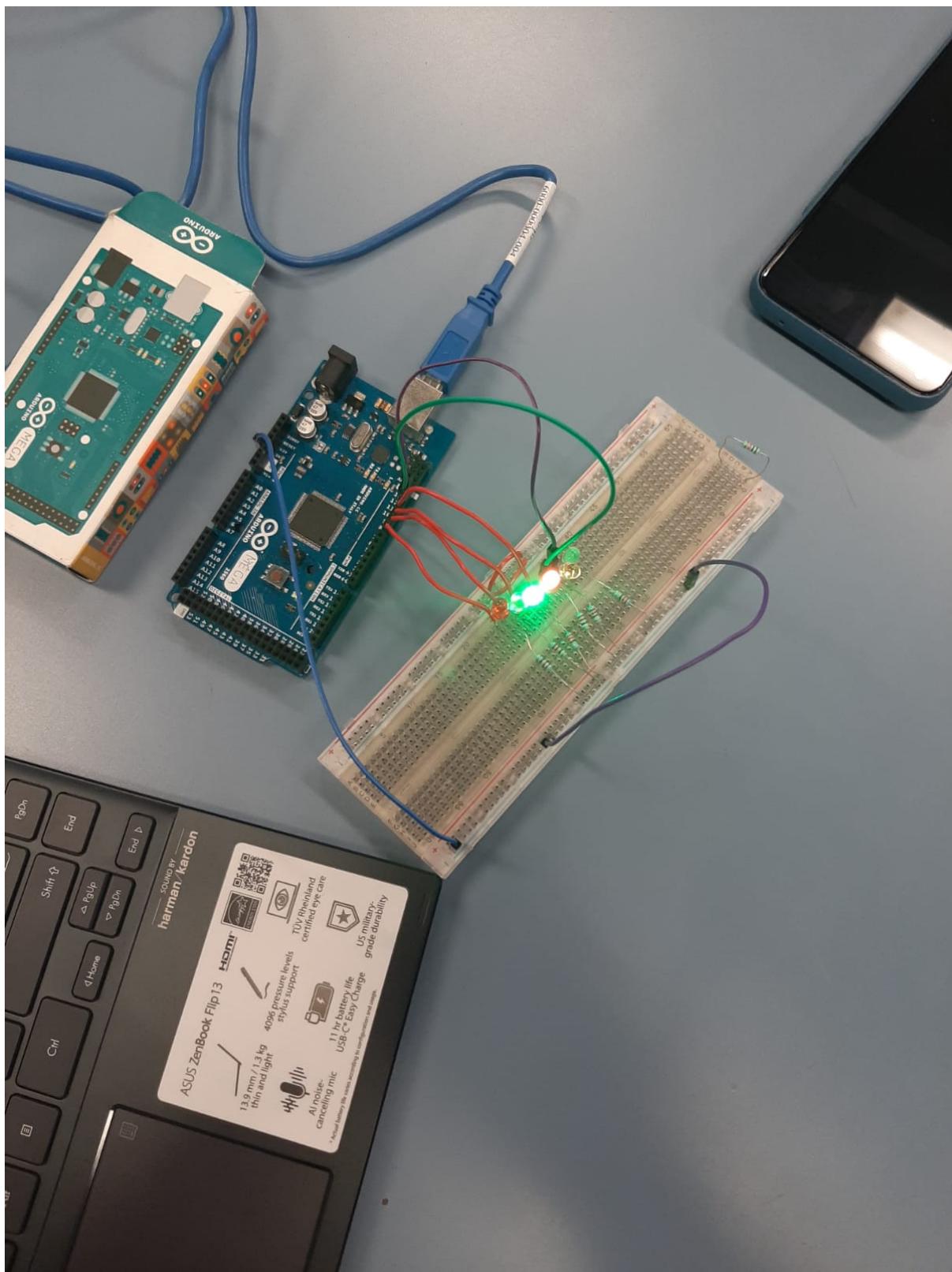


Figura 7. Circuito del semáforo armado y funcionando.

#### d. Actividad 4 – ASM

Finalmente, se muestra el código que deberá añadir a su Arduino, de tal forma que se programe en lenguaje ensamblador. Este código deberá ser subido a su repositorio de Github

comentando línea a línea explicando su funcionamiento (**P2A4-ASM**). Apóyese de la hoja de datos (*datasheet*) del ATMEGA2560.

```
void setup()
{
    DDRB = DDRB | B10000000; // Data Direction Register B: Inputs 0-6, Output 7
}

void loop()
{
    asm (
        "inicio: \n\t"
        "    sbi 0x05,0x07 \n\t"
        "    call tiempo \n\t"
        "    cbi 0x05,0x07 \n\t"
        "    call tiempo \n\t"
        "    jmp main \n\t"

        "tiempo: \n\t"
        "    LDI r22, 45 \n\t"
        "    LOOP_3: \n\t"
        "    LDI r21, 255 \n\t"
        "    LOOP_2: \n\t"
        "    LDI r20, 255 \n\t"
        "    LOOP_1: \n\t"
        "    DEC r20 \n\t"
        "    BRNE LOOP_1 \n\t"
        "    DEC r21 \n\t"
        "    BRNE LOOP_2 \n\t"
        "    DEC r22 \n\t"
        "    BRNE LOOP_3 \n\t"
        "    ret \n\t"
    );
}
```

Finalmente, mencione cuales son las ventajas y desventajas de programar en lenguaje ensamblador respecto a un lenguaje de alto nivel como el que utiliza Arduino.

Al usar ensamblador estamos dando instrucciones más fáciles de interpretar para la computadora, por lo que la capacidad computacional requerida para correrla es menor.

## 6. Conclusiones

Usar microcontroladores nos permite utilizar el poder de la computación sin el alto precio que tiene una PC, en esta práctica vimos un uso práctico controlando LEDs y simulando un semáforo. Esto nos muestra la gran utilidad del uso de microcontroladores.

Además, observamos que usando ensamblador en vez de C podemos reducir la cantidad de recursos utilizados por el programa, permitiendo hacer programas muy ligeros.

## 7. Referencias

Microcontrolador - Wikipedia, la enciclopedia libre. (2022). Retrieved 22 August 2022, from <https://es.wikipedia.org/wiki/Microcontrolador>

Resistencias Pull-Up y Pull-Down. (2022). Retrieved 23 August 2022, from [https://naylampmechatronics.com/blog/39\\_resistencias-pull-up-y-pull-down.html#:~:text=Las%20resistencias%20pull-up%20y,y%20asegurar%20una%20correcta%20lectura.](https://naylampmechatronics.com/blog/39_resistencias-pull-up-y-pull-down.html#:~:text=Las%20resistencias%20pull-up%20y,y%20asegurar%20una%20correcta%20lectura.)

(2022). Retrieved 25 August 2022, from <https://docs.arduino.cc/static/cd09dd6952ef2651be512dfdcf8b67f1/A000067-datasheet.pdf>