Control programado con Arduino

Práctica 3: El Pulsador Momentáneo (Funcionamiento Astable)

Hasta ahora, nuestros circuitos solo enviaban información hacia el exterior encendiendo LEDs (canales de salida). En esta práctica daremos un paso crucial: aprenderemos a recibir datos del exterior utilizando un pulsador (canal de entrada). Diseñaremos un programa inteligente que tome una decisión lógica y encienda un LED solo mientras mantengamos presionado el pulsador.

Objetivos de Aprendizaje

Al finalizar esta sesión, serás capaz de:

- Configurar pines digitales como canales de entrada de datos empleando `pinMode INPUT` .
- Leer y monitorizar estados digitales (valores lógicos HIGH y LOW) con la orden `digitalRead` .
- Aplicar estructuras de control condicionales mediante la instrucción `if else`.
- Entender la necesidad eléctrica de utilizar una resistencia de protección (Pull-Down) para evitar el estado flotante (ruido eléctrico).

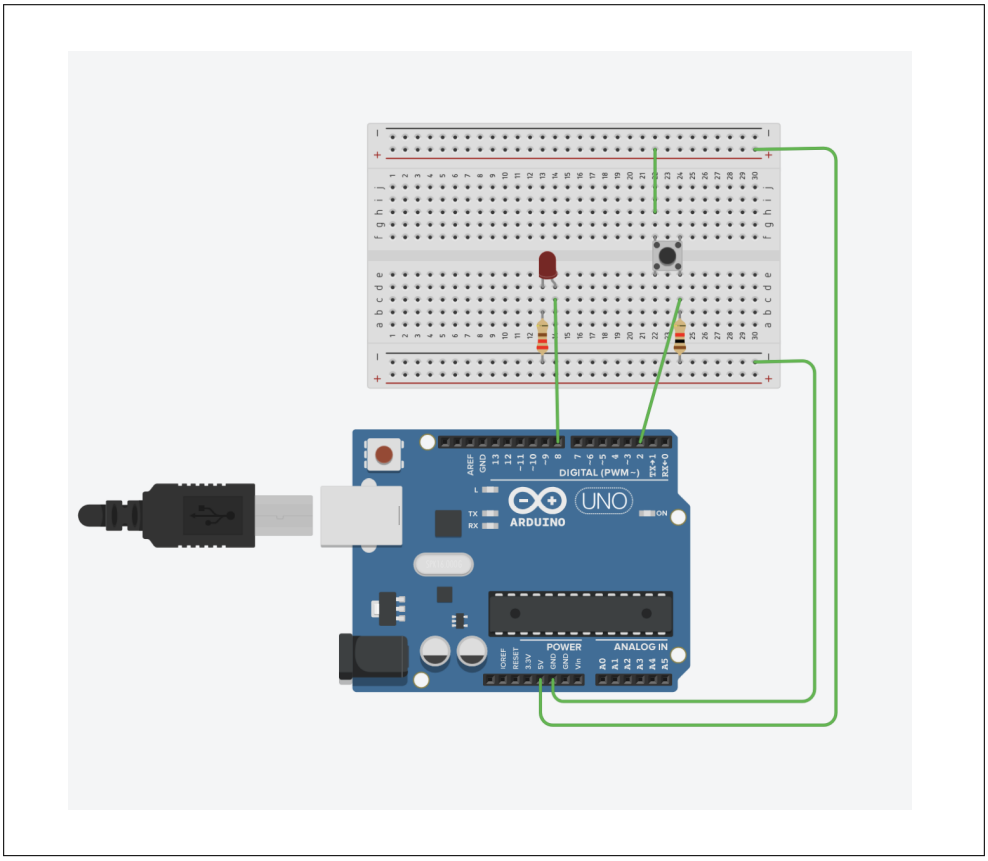
Componentes Necesarios

Busca y coloca los siguientes componentes en tu mesa de trabajo de TinkerCad:

- 1 Placa Arduino Uno.
- 1 Placa de pruebas pequeña (Protoboard).
- 1 Diodo LED y 1 resistencia de $220\,\Omega$ (para el LED).
- 1 Pulsador de cuatro patillas.
- 1 Resistencia de $10\,\text{k}\Omega$ (código de colores: marrón, negro, naranja; se usará como Pull-down).
- Cables de conexión virtuales.

Esquema de Montaje en la Protoboard

Es fundamental conectar el pulsador de forma adecuada. Para que Arduino lea un cero eléctrico (0V) cuando nadie presione el pulsador, utilizaremos una configuración de resistencia en **Pull-Down**.



Código Base: El condicional if/else

Escribe el siguiente programa en el editor de código de TinkerCad. Observa con atención cómo el código decide de manera continua el estado del LED leyendo el voltaje del pulsador.

```
// Declaracion de variables asignando los pines de conexion
int led = 8;         // El LED esta conectado en el pin digital 8
int pulsador = 2;    // El pulsador esta conectado en el pin digital 2

void setup() {
  pinMode(led, OUTPUT);       // Configuramos el LED como SALIDA de corriente
  pinMode(pulsador, INPUT);   // Configuramos el pulsador como ENTRADA de datos
}

void loop() {
  // Creamos una variable local para guardar el estado del pulsador (HIGH o LOW)
  int estadoPulsador = digitalRead(pulsador);

  // Condicional: Si el pulsador esta presionado (recibe 5V / HIGH)
  if (estadoPulsador == HIGH) {
    digitalWrite(led, HIGH);  // Enciende el LED
  }
  // En caso contrario (si no esta presionado / recibe 0V / LOW)
  else {
    digitalWrite(led, LOW);   // Apaga el LED
  }
}
```

Listing 1: Código para controlar un LED con un pulsador momentáneo.

¿Cómo funciona el código?

- `pinMode` `INPUT` : Le indica al microcontrolador que el pin 2 no va a suministrar corriente, sino que va a "escuchar" de forma pasiva el voltaje que le llegue del circuito exterior.
- `digitalRead` : Es un sensor lógico. Lee el voltaje en el pin indicado. Si detecta 5V nos devolverá el valor liso `HIGH` (alto), y si detecta 0V (GND) nos devolverá `LOW` (bajo).
- **La comparación** : En programación, un solo símbolo de igual ( ) sirve para guardar un valor dentro de una variable. Para comparar si dos cosas son matemáticamente iguales, debemos usar obligatoriamente el doble igual ( ).
- `if` `else` **(Si / Si no)**: Evalúa una condición lógica entre paréntesis. Si la condición se cumple, se ejecutan las líneas encerradas entre las llaves del `if`. Si la condición es falsa, el programa ignora ese bloque y ejecuta directamente lo que hay dentro de las llaves del `else`.

El Reto de la Práctica 3

Demuestra tu comprensión sobre el control del flujo del código superando estos tres retos:

1. **Lógica Inversa**: Modifica el programa en TinkerCad para que el LED esté siempre encendido por defecto, y se **apague** únicamente cuando mantengas presionado el pulsador.
2. **Semáforo Peatonal de dos posiciones**: Añade un segundo LED (por ejemplo, verde en el pin 9 y rojo en el pin 8). Haz que, por defecto, el LED rojo esté encendido y el verde apagado. Al presionar el pulsador, el rojo debe apagarse y el verde encenderse.
3. *Pregunta para reflexionar*: Si retiramos la resistencia de $10\,\mathrm{k}\Omega$ de la protoboard y conectamos el pulsador directamente a GND y a 5V sin ella, ¿qué tipo de anomalía eléctrica o física crees que podría ocurrirle a la placa de Arduino al pulsar el botón?