PROYECTO INGEGRADOR

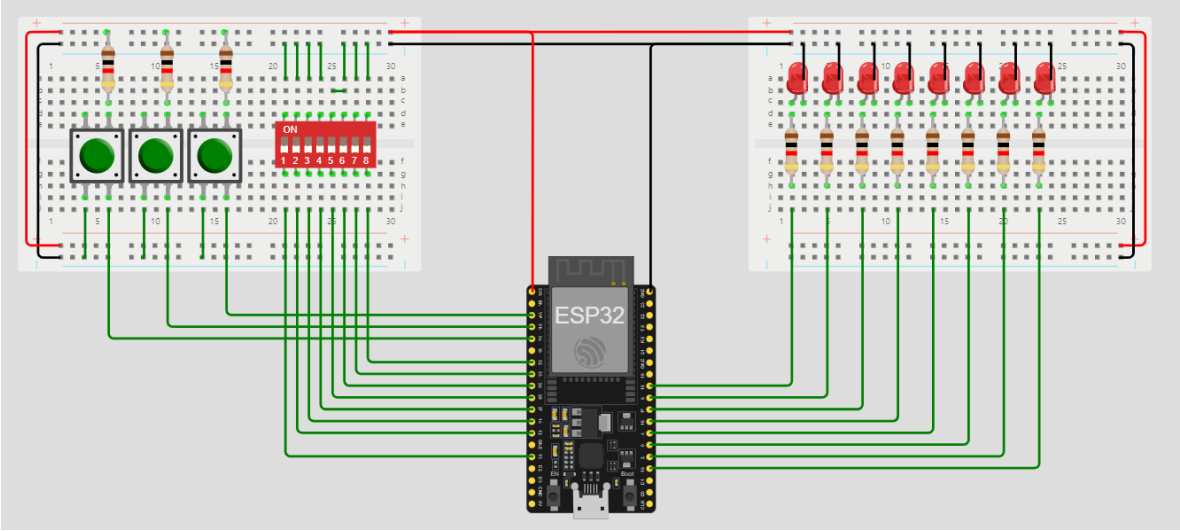
TP #3 Transductores binarios

Objetivos

- Practicas con el framework de Arduino en VsCode
- Primera aproximación a un entrenador básico
- Practica con sensores y actuadores digitales
- Primera aproximación a un controlador

Desarrollo

Sea el siguiente entrenador digital:



Definimos:

Pulsadores

- **btn1 (Pulsador 1)**: Conectado al pin **GPIO34**.
- **btn2 (Pulsador 2)**: Conectado al pin **GPIO39**.
- **btn3 (Pulsador 3)**: Conectado al pin **GPIO36**.

Dip Switch

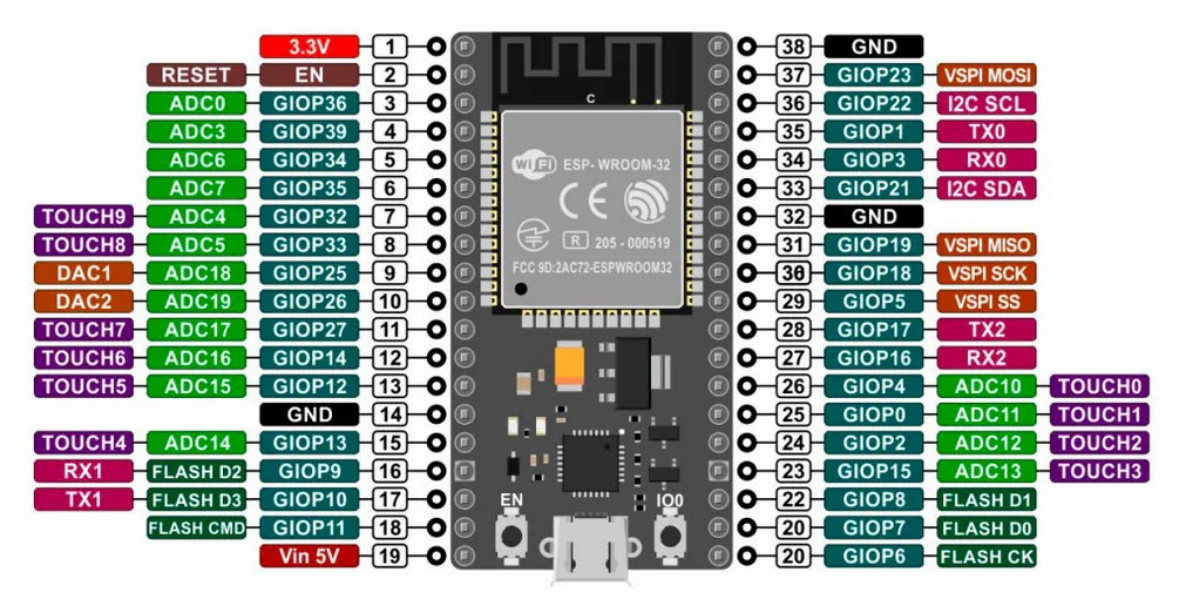
- **sw1.1 (Posición 1 del Dip Switch)**: Conectado al pin **GPIO13**.
- **sw1.2 (Posición 2 del Dip Switch)**: Conectado al pin **GPIO12**.
- **sw1.3 (Posición 3 del Dip Switch)**: Conectado al pin **GPIO14**.

- **sw1.4 (Posición 4 del Dip Switch)**: Conectado al pin **GPIO27**.
- **sw1.5 (Posición 5 del Dip Switch)**: Conectado al pin **GPIO26**.
- **sw1.6 (Posición 6 del Dip Switch)**: Conectado al pin **GPIO25**.
- **sw1.7 (Posición 7 del Dip Switch)**: Conectado al pin **GPIO33**.
- **sw1.8 (Posición 8 del Dip Switch)**: Conectado al pin **GPIO32**.

LEDs

- **led1 (LED 1)**: Conectado al pin **GPIO18**.
- **led2 (LED 2)**: Conectado al pin **GPIO5**.
- **led3 (LED 3)**: Conectado al pin **GPIO17**.
- **led4 (LED 4)**: Conectado al pin **GPIO16**.
- **led5 (LED 5)**: Conectado al pin **GPIO4**.
- **led6 (LED 6)**: Conectado al pin **GPIO0**.
- **led7 (LED 7)**: Conectado al pin **GPIO2**.
- **led8 (LED 8)**: Conectado al pin **GPIO15**.

Además, tener en cuenta que el pinout se refiere al modelo ESP32 WROOM de 38 pines.



Ejercicios a resolver:

Nivel Principiante

Ejercicio 1: Encender un LED

- Enciende el **led1** conectado al GPIO18 de forma continua.

Ejercicio 2: Parpadeo de un LED

- Programa el **led1** para que parpadee con un intervalo de 1 segundo.

Ejercicio 3: Secuencia de LEDs

- Crea una secuencia que encienda los LEDs del **led1** al **led3** de forma sucesiva, cada uno durante 500ms.

Ejercicio 4: Control de LED con botón

- Usa el **btn1** para encender el **led1** mientras se mantenga presionado.

Nivel Intermedio

Ejercicio 5: Uso de botón con estado

- Cambia el estado del **led1** cada vez que se presione y suelte el **btn1**.

Ejercicio 6: Debounce de botón

- Implementa una lógica de debounce en el **btn1** para evitar lecturas erróneas.

Ejercicio 7: Control de múltiples LEDs con botones

- Usa **btn1** y **btn2** para controlar el estado de **led1** y **led2** respectivamente.

Ejercicio 8: Uso de dip switches para control de LEDs

- Lee el estado de los dip switches **sw1.1** a **sw1.8** y refleja el estado en los **led1** a **led8**.

Nivel Avanzado

Ejercicio 9: Secuencia de LEDs con botón

- Crea una secuencia de luces que avance cada vez que se presione **btn1**.

Ejercicio 10: Control de velocidad de parpadeo con dip switch

- Utiliza los dip switches **sw1.1** a **sw1.3** para controlar la velocidad de parpadeo de **led1**, asignando distintas velocidades.

Ejercicio 11: Patrón de parpadeo de LEDs con dip switches

- Establece un patrón de parpadeo para los **led1** a **led8** basado en la combinación de estados de **sw1.1** a **sw1.4**. Por ejemplo, cada posición activa del switch puede representar un patrón diferente (como parpadeo rápido, lento, secuencial, etc.).

Ejercicio 12: Medidor de pulsaciones

- Programa un contador de pulsaciones utilizando **btn1**. El número de pulsaciones debe mostrarse en una secuencia de LEDs (por ejemplo, **led5** a **led8** donde cada LED representa una cantidad de pulsaciones).

Ejercicio 13: Contraseña con botones

- Implementa un sistema de contraseña usando **btn1**, **btn2**, y **btn3** donde una secuencia específica de pulsaciones activa **led1**. Si la secuencia es incorrecta, **led2** debería encenderse.

Ejercicio 14: Aplicación de timers para control de LEDs

- Utiliza el temporizador del ESP32 para controlar el parpadeo de **led1** a **led4** sin usar la función **delay()**, permitiendo que el programa ejecute otras tareas mientras los LEDs parpadean.

Ejercicio 15: Control de LEDs mediante comunicación serial

- Escribe un programa que reciba comandos a través del puerto serie para controlar los LEDs. Por ejemplo, enviar '1' podría encender **led1**, '2' apagar **led2**, etc.

Ejercicio 16: Secuencia de luces de emergencia

- Simula luces de emergencia con los LEDs, donde **led1** y **led2** parpadean alternativamente en un patrón rápido, mientras que **led3** y **led4** lo hacen en un patrón más lento.