

PROYECTO INTERGRADOR I

DOCENTE: GONZALO VERA

ALUMNO/A: KARINA JAZMIN BARBERO

CARRERA: TÉCNICO SUPERIOR EN TELECOMUNICACIONES

Trabajo Practico #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

Ejercicios a resolver:

Nivel Principiante

Ejercicio 1:

Encender un LED

- Enciende el led1 conectado al GPIO18 de forma continua.

WOKWI SAVE SHARE TP3 Ejercicio 1: Encender un LED Docs

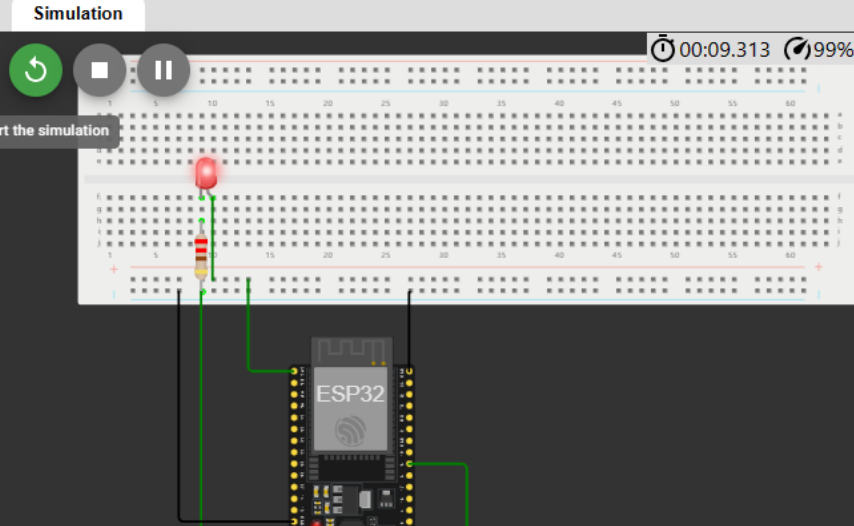
main.py diagram.json

```
1 print("Hello, ESP32!")
2 // La función setup se ejecuta una vez cuando presionas el botón de reinicio
3 void setup() {
4
5   // Inicializa el pin digital GPIO18 como salida
6   pinMode(18, OUTPUT);
7
8   // La función loop se ejecuta continuamente en un bucle
9
10 void loop() {
11   // Enciende el LED
12   digitalWrite(18, HIGH);
13
14   // Espera 500 milisegundos
15   delay(500);
16   // Apaga el LED
17   digitalWrite(18, LOW);
18
19   // Espera otros 500 milisegundos
20   delay(500);
21 }
22
23
24
```

Simulation

00:09.313 99%

Restart the simulation



mode:DIO, clock div:2
load:0x3fff0030,len:4728
load:0x40078000,len:14888
load:0x40080400,len:3368
entry 0x400805cc
Traceback (most recent call last):
 File "main.py", line 2
SyntaxError: invalid syntax
MicroPython v1.22.0 on 2023-12-27; Generic ESP32 module with ESP32
Type "help()" for more information.
>>>

<https://wokwi.com/projects/396822336674150401>

Ejercicio 2:

Parpadeo de un LED

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

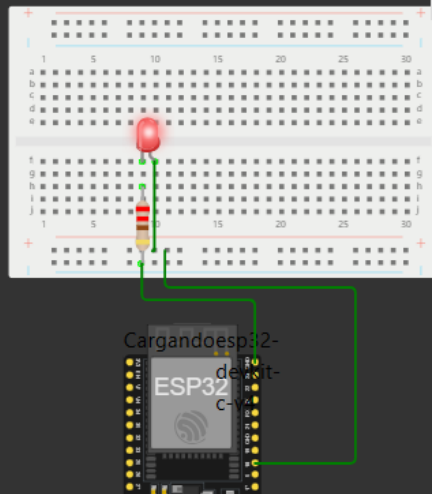
WOKWI SALVAR COMPARTIR TP3 Ejercicio 2: : Parpadeo de un LED Docs

sketch.ino diagram.json Administrador de la biblioteca

```
1 Configuración del pin del LED
2 #define LED_PIN 18
3
4 Configuración nula() {
5
6   Ponga su código de instalación aquí, para ejecutarlo una vez:
7   Serie.begin(115200);
8   Serie.println("¡Hola, ESP32!");
9   pinMode(LED_PIN, SALIDA);
10 }
11
12 Bucle vacío() {
13   Coloque su código principal aquí, para ejecutarlo repetidamente:
14   digitalWrite(LED_PIN, ALTO); Enciende el LED
15   retraso(1000); Espera 1 segundo
16   digitalWrite(LED_PIN, BAJO); Apaga el LED
17   retraso(1000); Espera 1 segundo nuevamente
18 }
19
20
```

Simulación

00:04.665 99%



Cargando esp32-

mode:DI0, clock div:2
load:0x3fff0030,len:1156
load:0x40078000,len:11456
ho 0 tail 12 room 4
load:0x40080400,len:2972
entry 0x400805dc
Hello, ESP32!

<https://wokwi.com/projects/396873154990287873>

Ejercicio 3:

Secuencia de LEDs

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

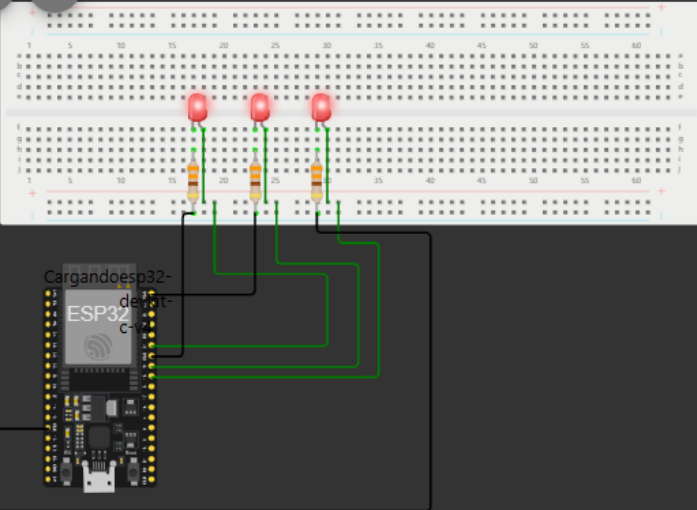
WOKWI SALVAR COMPARTIR TP3 Ejercicio 3: Secuencia de LEDs Docs

sketch.ino diagram.json Administrador de la biblioteca

```
1 // Definición de pines para los LEDs
2 const int led1Pin = 18;
3 const int led2Pin = 19;
4 const int led3Pin = 21;
5
6 Configuración nula() {
7   Configura los pines como salidas
8   pinMode(led1Pin, SALIDA);
9   pinMode(led2Pin, SALIDA);
10  pinMode(led3Pin, SALIDA);
11 }
12
13 Bucle vacío() {
14   Enciende el LED 1 durante 500 ms
15   digitalWrite(led1Pin, HIGH);
16   retraso(500); Espera 500 ms
17   digitalWrite(led1Pin, LOW);
18
19   Enciende el LED 2 durante 500 ms
20   digitalWrite(led2Pin, HIGH);
21   retraso(500); Espera 500 ms
22   digitalWrite(led2Pin, LOW);
23
24   Enciende el LED 3 durante 500 ms
25   digitalWrite(led3Pin, HIGH);
26   retraso(500); Espera 500 ms
27   digitalWrite(led3Pin, LOW);
28 }
29
30
```

Simulación

00:13.941 96%



Cargando esp32-
defini-
c

ets Jul 29 2019 12:21:46

rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)

config: 0, SPIWP:0xee

<https://wokwi.com/projects/396873321664606209>

Ejercicio 4:

Control de LED con botón

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

WOKWI SALVAR COMPARTIR TP Ejercicio 4: Control de LED con botón Docs

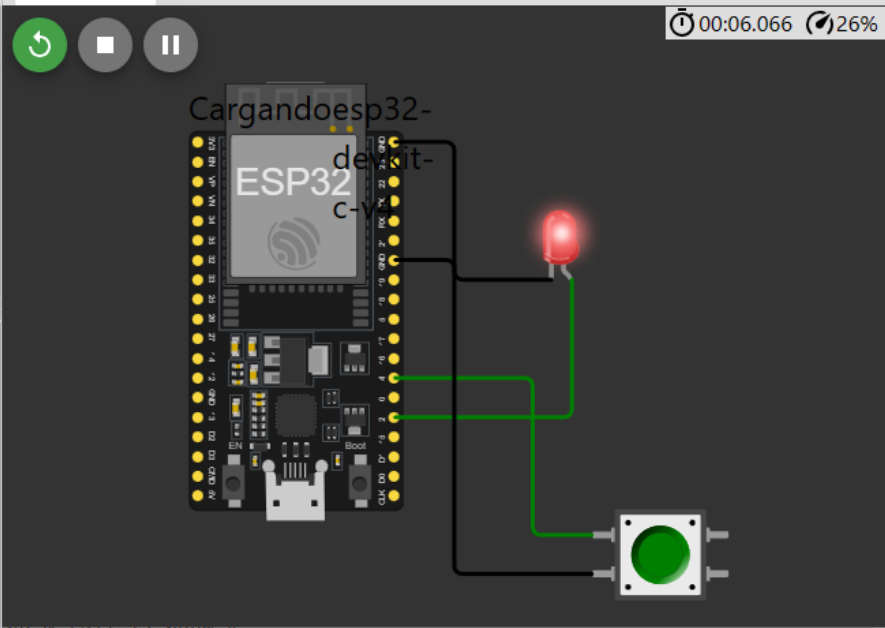
sketch.ino diagram.json ● Administrador de la biblioteca

```
1 const int ledPin = 2; Pin del LED
2 const int buttonPin = 4; Pin del botón
3
4
5 Configuración nula() {
6   Ponga su código de instalación aquí, para ejecutarlo una vez:
7   Serie.begin(115200);
8   Serie.println("¡Hola, ESP32!");
9
10  pinMode(ledPin, OUTPUT);
11
12  pinMode(buttonPin, INPUT_PULLUP); Configura el botón con resistencia pul
13 }
14
15 Bucle vacío() {
16   if (digitalRead(buttonPin) == LOW) { Si el botón está presionado
17     digitalWrite(ledPin, HIGH); Enciende el LED
18   } más {
19     digitalWrite(ledPin, LOW); Apaga el LED
20   }
21 }
22
23
```

Simulación

00:06.066 26%

Cargando esp32-
devkit-c



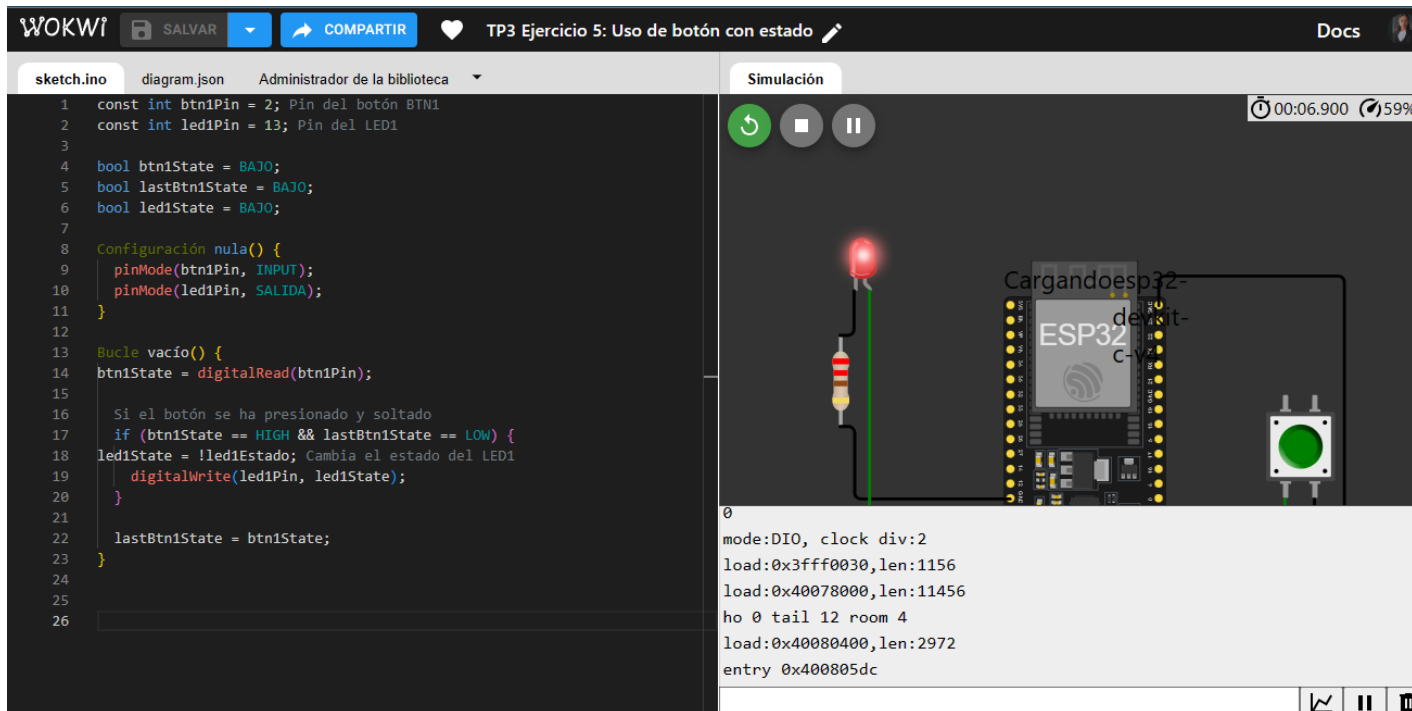
no 0 call 12 Room 4
load:0x40080400,len:2972
entry 0x400805dc
Hello, ESP32!

<https://wokwi.com/projects/396878487947781121>

Nivel Intermedio

Ejercicio 5: Uso de botón con estado

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



The image shows the Wokwi web IDE interface. The left pane displays a C++ sketch for an ESP32. The sketch defines two pins: btn1Pin (2) and led1Pin (13). It initializes btn1State, lastBtn1State, and led1State to LOW. The setup function configures btn1Pin as an input and led1Pin as an output. The loop function reads the button state and toggles the LED state when the button is pressed and released.

```
1 const int btn1Pin = 2; Pin del botón BTN1
2 const int led1Pin = 13; Pin del LED1
3
4 bool btn1State = BAJO;
5 bool lastBtn1State = BAJO;
6 bool led1State = BAJO;
7
8 Configuración nula() {
9   pinMode(btn1Pin, INPUT);
10  pinMode(led1Pin, SALIDA);
11 }
12
13 Bucle vacío() {
14   btn1State = digitalRead(btn1Pin);
15
16   Si el botón se ha presionado y soltado
17   if (btn1State == HIGH && lastBtn1State == LOW) {
18     led1State = !led1Estado; Cambia el estado del LED1
19     digitalWrite(led1Pin, led1State);
20   }
21
22   lastBtn1State = btn1State;
23 }
24
25
26
```

The right pane shows a simulation of the circuit. It includes an ESP32 module, a red LED, and a green button. The simulation is running, as indicated by the play button and the progress bar. The console output shows the following messages:

```
mode:DIO, clock div:2
load:0x3fff0030,len:1156
load:0x40078000,len:11456
ho 0 tail 12 room 4
load:0x40080400,len:2972
entry 0x400805dc
```

<https://wokwi.com/projects/396882436325402625>

Ejercicio 6:

Debounce de botón

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

WOKWI TP3 Ejercicio 6: Debounce de botón Docs

sketch.ino diagram.json Administrador de la biblioteca

```
1 const int buttonPin = 2; DONDE ESTÁ CONECTADO EL PIN DEL PULSADOR
2 const int ledPin = 13;
3
4 botón bool volátilPresionado = false;
5
6 manija void buttonPress() {
7   buttonPressed = true;
8 }
9
10 Configuración nula() {
11   pinMode(buttonPin, INPUT_PULLUP);
12   pinMode(ledPin, OUTPUT);
13   attachInterrupt(digitalPinToInterrupt(buttonPin), handleButtonPress, FALLING);
14 }
15
16 Bucle vacío() {
17   if (buttonPressed) {
18     digitalWrite(ledPin, !digitalRead(ledPin)); Cambia el estado del LED
19   }
20   buttonPressed = false;
21 }
22
23
```

Simulación

00:14.100 56%

Cargando esp32-
de kit-
c-

mode:DI0, clock div:2
load:0x3fff0030,len:1156
load:0x40078000,len:11456
ho 0 tail 12 room 4
load:0x40080400,len:2972
entry 0x400805dc

<https://wokwi.com/projects/397313341730031617>

Ejercicio 7:

Control de múltiples LEDs con botones

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

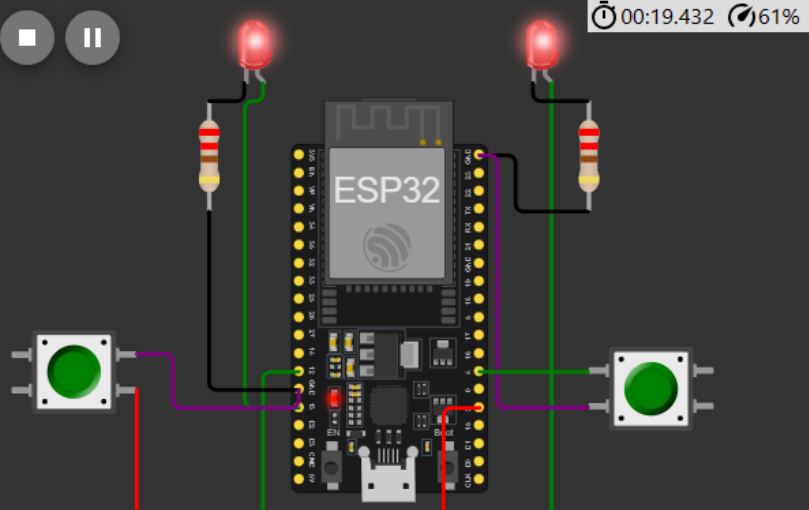
WOKWI SAVE SHARE TP3 Ejercicio 7: Control de múltiples LEDs con botones Docs

sketch.ino diagram.json Library Manager

```
1 const int led1Pin = 13; // Pin para el LED 1
2 const int led2Pin = 12; // Pin para el LED 2
3 const int btn1Pin = 2; // Pin para el botón 1
4 const int btn2Pin = 4; // Pin para el botón 2
5
6 bool led1State = false; // Estado inicial del LED 1 (apagado)
7 bool led2State = false; // Estado inicial del LED 2 (apagado)
8
9 void setup() {
10   pinMode(led1Pin, OUTPUT);
11   pinMode(led2Pin, OUTPUT);
12   pinMode(btn1Pin, INPUT_PULLUP);
13   pinMode(btn2Pin, INPUT_PULLUP);
14 }
15
16 void loop() {
17   // Lee el estado de los botones
18   bool btn1Pressed = !digitalRead(btn1Pin); // Invertimos el estado porque
19   bool btn2Pressed = !digitalRead(btn2Pin); // Invertimos el estado porque
20
21   // Actualiza el estado de los LEDs según los botones
22   if (btn1Pressed) {
23     led1State = !led1State; // Cambia el estado del LED 1
24     digitalWrite(led1Pin, led1State);
25   }
26
27   if (btn2Pressed) {
28     led2State = !led2State; // Cambia el estado del LED 2
29     digitalWrite(led2Pin, led2State);
30   }
31 }
```

Simulation

00:19.432 61%



mode:DI0, clock div:2
load:0x3fff0030,len:1156
load:0x40078000,len:11456
ho 0 tail 12 room 4
load:0x40080400,len:2972
entry 0x400805dc

<https://wokwi.com/projects/397315571978113025>

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.

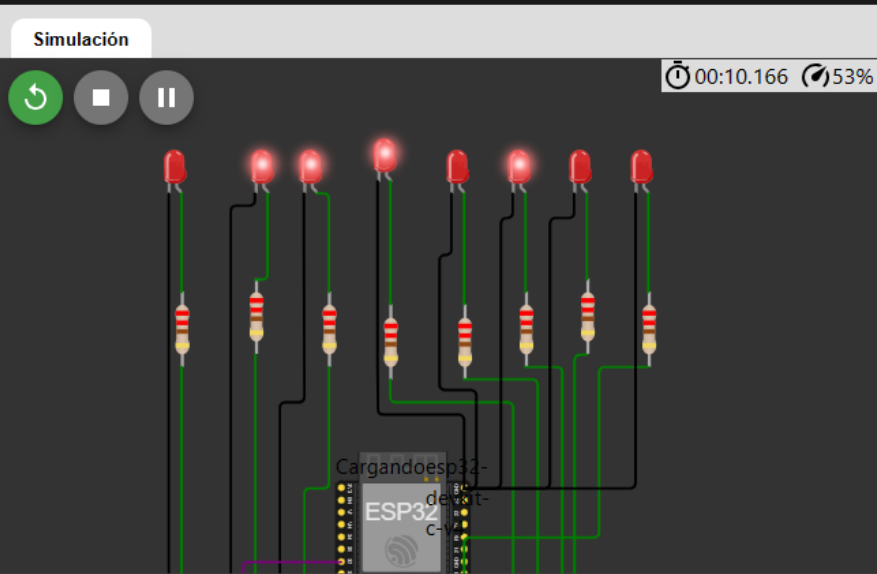
WOKWI SALVAR COMPARTIR TP3 Ejercicio 8: Uso de dip switches para control de LEDs Docs

sketch.ino diagram.json Administrador de la biblioteca

```
1 Definición de pines para los dip switches y los LEDs
2 const int dipSwitchPins[] = {2, 3, 4, 5, 25, 26, 27, 32, };
3 const int ledPins[] = { 12, 13, 14, 15, 16, 17, 18, 19,};
4
5 Configuración nula() {
6   Configura los pines de los dip switches como entradas
7   for (int i = 0; i < 8; i++) {
8     pinMode(dipSwitchPins[i], INPUT);
9   }
10
11   Configura los pines de los LEDs como salidas
12   for (int i = 0; i < 8; i++) {
13     pinMode(ledPins[i], SALIDA);
14   }
15 }
16
17 Bucle vacío() {
18   Lee el estado de los dip switches
19   for (int i = 0; i < 8; i++) {
20     int switchState = digitalRead(dipSwitchPins[i]);
21
22     Si el dip switch está encendido, activa el LED correspondiente
23     if (switchState == HIGH) {
24       digitalWrite(ledPins[i], HIGH);
25     } más {
26       digitalWrite(ledPins[i], LOW);
27     }
28   }
29 }
30
```

Simulación

00:10.166 53%



ets Jul 29 2019 12:21:46

rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)

config: 0, SPIWP:0xee

clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00

mode:DIO, clock div:2

<https://wokwi.com/projects/397321393877431297>

