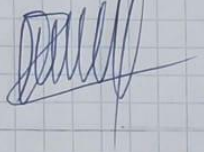
13-10

Integrantes: Oebe
Chipugsi
Chasi

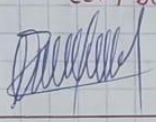
x2 x2

Fecha: 13 / 10 / 2025

Practica #1 LED / BCM



LED / BOARD

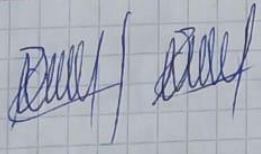


Práctica #1 LED + button / BCM Fecha: 14 / 10 / 2025

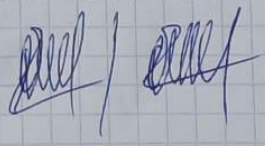
LED + button / BOARD

Práctica # 2 Fecha: 07 / 11 / 2025

funciones LED BCM / BOARD



funciones LED + button BCM / BOARD



Control de LED y Botón con Funciones

Práctica 2

Anthony Fabricio Chipugsi Pilicita
achipugsip@est.ups.edu.ec
Diego Mauricio Chasi Guamushig
dchasig@est.ups.edu.ec
Cinthya Aracelly Orbe Muñoz
corbem@est.ups.edu.ec

I. Introducción

El control de un LED mediante un botón es uno de los proyectos más sencillos y didácticos para iniciarse en la programación y manejo de hardware con la Raspberry Pi 3. Utilizando la conexión remota a través de PuTTY, es posible escribir y ejecutar programas en Python que gestionen los pines GPIO de la placa. A través de funciones, se organiza el código de manera modular y reutilizable, lo que facilita la comprensión y el mantenimiento del programa. En este ejercicio, el pulsador actúa como entrada digital que envía una señal al sistema, mientras que el LED funciona como salida, encendiéndose o apagándose según la interacción del usuario. Este tipo de práctica no solo introduce conceptos básicos de electrónica y programación, sino que también sienta las bases para proyectos más complejos de automatización y control.

II. Metodología

1. Conexión del circuito

- En el protoboard y con ayuda de puentes, se conectó un LED, el pin positivo al pin GPIO 12 (BCM) de la raspberry, y el pin negativo a tierra mediante una resistencia de 220 Ω.
- Luego, en el protoboard se conectó un pulsador al pin GPIO 18 (BCM) de la raspberry, del mismo pin alimentado a positivo mediante una resistencia de 10kΩ, y el otro alimentado a negativo.

2. Programación en PuTTY

- A continuación explicaré lo que hace cada línea de código de la función led en BCM, tomando en cuenta que se realiza la misma estructura para escribir las otras tres programaciones: función led BOARD, función botón BCM y función botón BOARD.

```
import RPi.GPIO as GPIO
import time
```

Figura 1. programación función led BCM.

- **Importa la librería RPi.GPIO**, que permite controlar los pines GPIO de la Raspberry Pi.

- Se le asigna el alias GPIO para facilitar su uso en el resto del código.
- **Importa la librería** time, que se usa para manejar pausas temporales (delays) en la ejecución del programa.

```
def lbo(pin, modo):
        GPIO.setwarnings(False)
        GPIO.setmode(modo)
        GPIO.setup(pin, GPIO.OUT)
```

Figura 2. programación función led BCM.

- La primera línea de la figura, define una función llamada lbo que recibe dos parámetros: pin, número del pin GPIO que se usará para el LED; y modo, que es el modo de numeración de los pines.
- La segunda línea, desactiva las advertencias que podrían aparecer si el pin ya fue configurado anteriormente.
- La tercera línea de la figura, establece el modo de numeración de los pines, ya sea BOARD(numeración física de los pines) o BCM(numeración lógica de la raspebrry).
- La última línea de la figura, configura el pin como salida digital, para poder enviarle señales al LED.

```
        while True:
                GPIO.output(pin,True)
                time.sleep(1)
                GPIO.output(pin, False)
                time.sleep(1)
```

Figura 3. programación función led BCM.

- La primera línea de la figura, inicia un bucle infinito, el LED se encenderá y apagará continuamente.

- La segunda línea de la figura, enciende el LED al enviar una señal de voltaje alto (True) al pin. [1]
- La tercera línea espera un segundo con el LED encendido.
- La cuarta línea apaga el LED al enviar una señal de voltaje bajo (False) al pin.
- La quinta línea espera un segundo con el LED apagado antes de repetir el ciclo. [1]
- La última línea llama a la función lbo, indicando que se usará el pin físico número 12 y el modo de numeración BOARD.

```
lbo(12, GPIO.BOARD)
```

Figura 4. programación función led BCM.

- Esta última línea llama a la función lbo, indicando que se usará el pin físico número 12 y el modo de numeración BOARD.
- A continuación explicaré el script del menú interactivo para llamar a las 4 funciones anteriormete mencionadas:

```
import time

def op1():
        import funledbcm as lbcm
        return
def op2():
        import funledboard as lboard
        return
def op3():
        import funbutbcm as bbcm
        return
def op4():
        import funbutboard as bboard
        return
while True:
        op=0
        print ("-----MENU-----")
        print ("1.LED BCM")
        print ("2.LED BOARD")
        print ("3.BOTON BCM")
        print ("4.BOTON BOARD")
        op=int(input("op= "))
        if op==1:
                print ("Opcion 1 seleccionada")
                op1()
                time.sleep (3)
                break
        if op==2:
                print ("Opcion 2 seleccionada")
                op2()
                time.sleep (3)
                break
        if op==3:
                print ("Opcion 3 seleccionada")
                op3()
                time.sleep (3)
                break
        if op==4:
                print ("Opcion 4 seleccionada")
                op4()
                time.sleep (3)
                break
```

Figura 5. programación menú interactivo.

Esta programación es un menú interactivo en consola donde el usuario podrá elegir cuatro opciones:

-1. Controlar un LED usando BCM
-2. Controlar un LED usando BOARD
-3. Controlar uun botón usando BCM
-4. Controlar un botón usando BOARD

Cada opción importa un módlo externo que contiene la lógica correspondiente como:

```
def op1():
        import funledbcm as lbcm
```

Figura 6. programación menú interactivo.

- Esto importa el archivo de la función del led BCM.

III. Resultados

- Este código hace que el Led conectado al pin 12 parpadee cada segundo de forma indefinida y al agregar el pulsador la programación cambia, el LED se queda encendido por defecto y cuando se presiona el pulsador el LED se apaga, y al soltar el botón, el LED se enciende nuevamente.

- En el menú interactivo, el usuario escribe un número del 1 al 4, el prohgrama imprime qué opción fué seleccionada, importando el módulo correspondiente, espera 3 segundos para ejecutar el código y luego sale del bucle con break.

IV. Discusión

Esta práctica permitió comprender de forma aplicada cómo interactuar con los pines GPIO de las Raspberry Pi3 para controlar dispositivos electrónicos simples como LEDs y botones. A través de uso de PuTTY, se logró ejecutar scripts de manera remota, lo que facilita el desarrollo sin necesidad de una interfaz gráfica.

Uno de los aspectos más relevantes fué la implementación de funciones modulares, que permiten organizar el código de forma clara y reutilizable. Esto se evidenció en el menú principal, que actúa como un controlador general para ejecutar distintos módulos según el modo de numeración, ya sea BCM o BOARD, y el tipo de componente, ya sea un LED o un Botón.

V. Conclusiones

- La implementación de funciones y módulos separados para cada configutación, BCM o BOARD, permitió una estructura de código más limpia y fácil de mantener.

- La práctica demostró cómo la Raspberry Pi3 puede controlar dispositivos electrónicos como LEDs y botones mediante la programación en Python, ejecutada desde PuTTY.
- El uso de if, while y funciones permitió controlar el comportamiento del LED en respuesta al estado del botón, mostrando cómo la lógica de la programación puede traducirse en acciones físicas concretas.

VI. Referencias

[1] Raspberry Pi Foundation, "GPIO and RPi.GPIO library," Raspberry Pi Docs, 2025. [Online]. Available: https://www.raspberrypi.com/documentation/computers/os.html#gpio

VII. Anexos

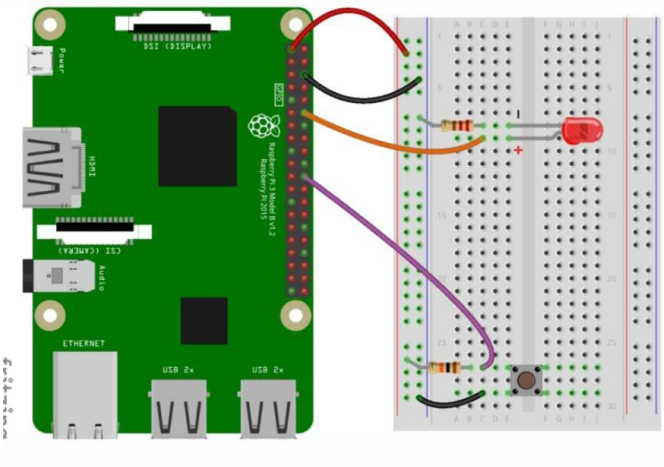


Figura 7. Diagrama de conexión.

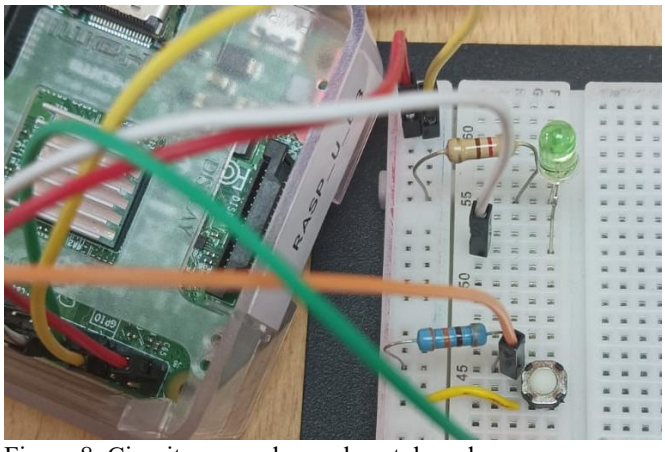


Figura 8. Circuito aramado en el protoboard.