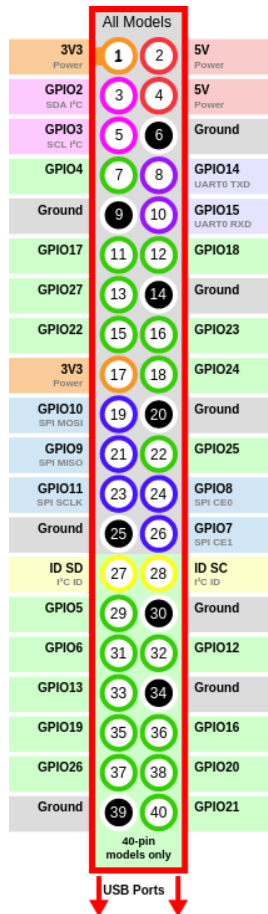


# COMPARACIÓN DE RPI.GPIO CON GPIOZERO

En el siguiente documento se muestran los ejemplos vistos en clase (codificados en forma tradicional) y la forma de codificación equivalente empleando la librería gpiozero.

## Antes de empezar

### Distribución de pines



### Empleando el nombre BCM del pin

```
led = LED(17)
led = LED("GPIO17")
led = LED("BCM17")
```

### Empleando el número del pin fisico

```
led = LED("BOARD11")
```

### Empleando el nombre Wiring Pi asociado al pin (numero 0)

```
led = LED("WPI0")
```

## Instalación

Actualizar la lista de repositorios

```
sudo apt update
```

Instalar el paquete para python 3:

```
sudo apt install python3-gpiozero
```

Instalar el paquete para python 2:

```
sudo apt install python-gpiozero
```

La documentación completa se puede encontrar en el siguiente [enlace](#) (version en [pdf](#))

## Ejemplos

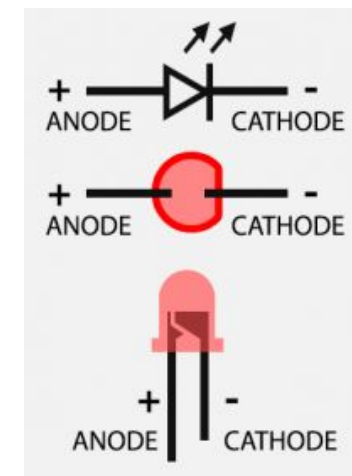
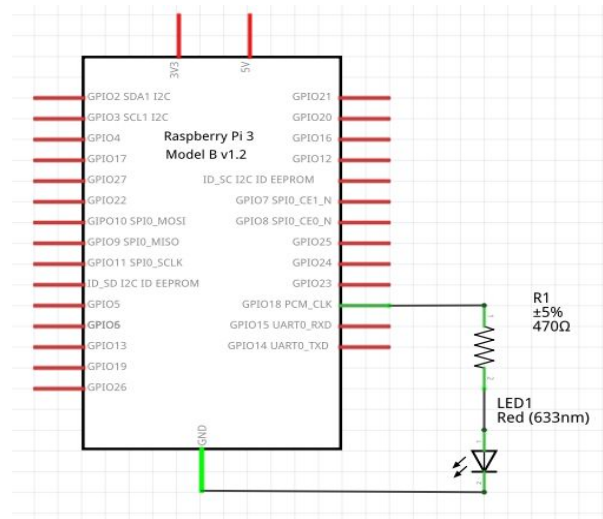
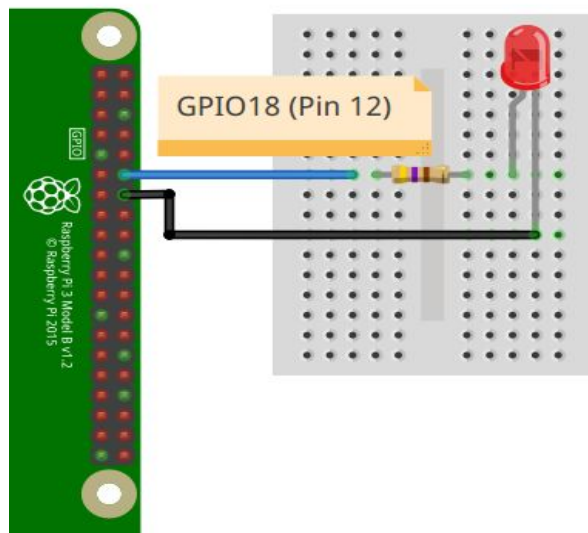
A continuación se muestran los enlaces a los ejemplos que vienen con la documentación:

1. [Basic Recipes](#)
2. [Advanced Recipes](#)
3. [Remote GPIO](#)

Para migrar ejemplos de rPI.GPIO a gpiozero (que fue lo que se hizo) puede consultar el siguiente [enlace](#).

**Ejemplo 1:** Realizar un programa que haga parpadear un led cada 1 segundo.

Para este caso se empleó la clase [LED](#).



Forma tradicional	Empleando la libreria gpiozero
<pre>led_blink_ex1.py  import RPi.GPIO as GPIO import time  # Pin Definitions ledPin = 18  # Pin Setup GPIO.setmode(GPIO.BCM) # Broadcom pin-numbering scheme GPIO.setup(ledPin,GPIO.OUT) # LED pin set as output try:     while True:         GPIO.output(ledPin, True)         time.sleep(1)         GPIO.output(ledPin, False)         time.sleep(1) except KeyboardInterrupt: # If CTRL+C is pressed, exit cleanly:     GPIO.cleanup() # cleanup all GPIO</pre>	<pre>gpio_zero_led_blink_ex1.py  from gpiozero import LED from time import sleep  ledPin = LED(18)  while True:     ledPin.on()     sleep(1)     ledPin.off()     sleep(1)</pre>

Pregunta:

Que hace el siguiente código

```
from gpiozero import LED
from signal import pause

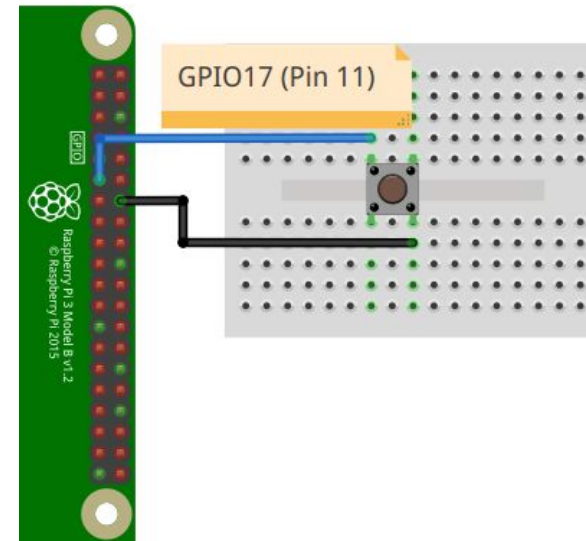
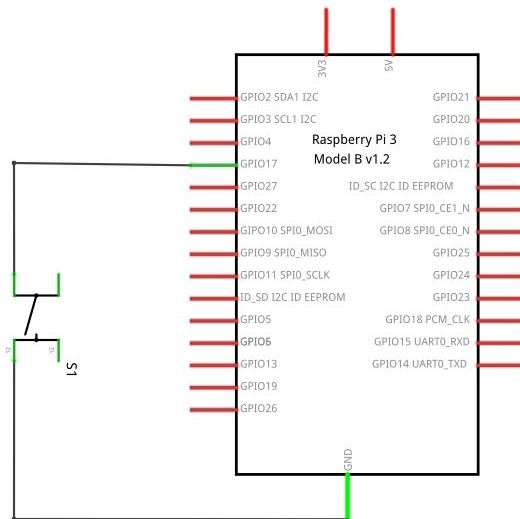
ledPin = LED(18)

ledPin.blink()

pause()
```

**Ejemplo 2:** Realizar un programa que imprima en consola “**botón presionado**” cada vez que se presione el botón. Configure al aplicación de modo que no sea necesario una resistencia de pull-up externa.

Para este caso se empleó la clase [Button](#)



Forma tradicional

Empleando la libreria gpiozero

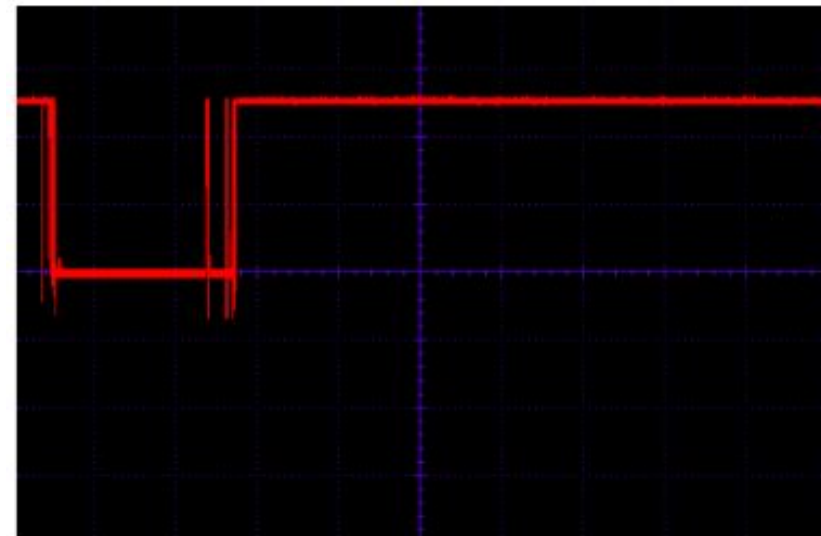
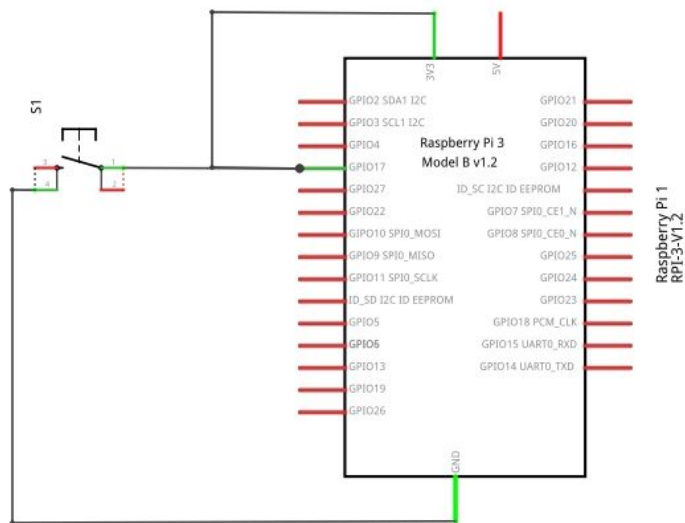
button_ex2.py	gpio_zero_button_ex2_1.py
<pre>import RPi.GPIO as GPIO import time  # Pin Definitons buttonPin = 17  # Pin Setup GPIO.setmode(GPIO.BCM) # Broadcom pin-numbering scheme GPIO.setup(buttonPin, GPIO.IN, pull_up_down=GPIO.PUD_UP) # LED pin set as output try:     while True:         buttonValue = GPIO.input(buttonPin)         if buttonValue == True:             print("Boton presionado") except KeyboardInterrupt: # If CTRL+C is pressed, exit cleanly:     GPIO.cleanup() # cleanup all GPIO</pre>	<pre>from gpiozero import Button  buttonPin = Button(17) buttonPin.pull_up()  while True:     if buttonPin.is_pressed:         print("Boton presionado")</pre>

Que hace el siguiente código:

<pre>from gpiozero import Button  buttonPin = Button(17)  buttonPin.wait_for_press() print("Button was pressed")</pre>
--

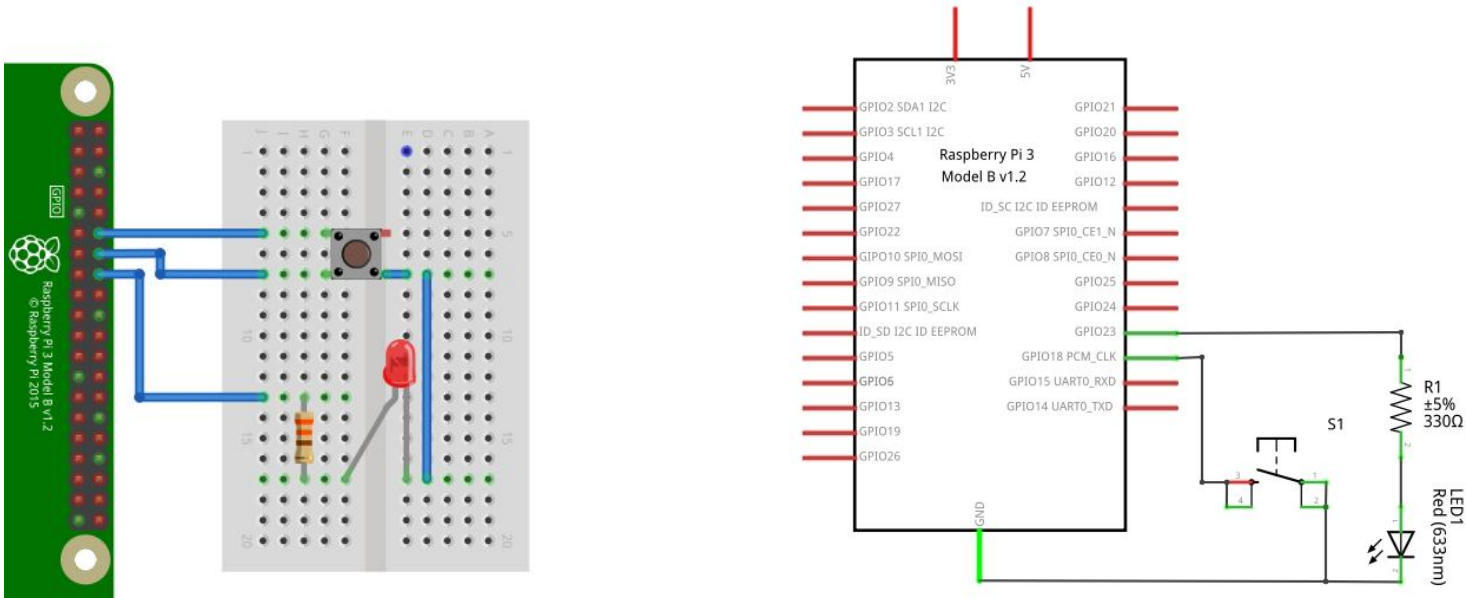
**Ejemplo 3:** Repita el ejemplo 2 pero sin usar resistencia de pull-up externa, el programa deberá solucionar el problema de los rebotes.

Para este caso se empleó la clase [Button](#).



Forma tradicional	Empleando la libreria gpiozero
button_anti_db_ex3.py	gpio_zero_button_anti_db_ex3.py
<pre>import RPi.GPIO as GPIO import time  # Pin Definitons buttonPin = 17  # Pin Setup GPIO.setmode(GPIO.BCM) # Broadcom pin-numbering scheme GPIO.setup(buttonPin,GPIO.IN) # LED pin set as output try:     while True:         buttonValue = GPIO.input(buttonPin)         if buttonValue == True:             print("Boton presionado")             time.sleep(0.2) except KeyboardInterrupt: # If CTRL+C is pressed, exit cleanly:     GPIO.cleanup() # cleanup all GPIO</pre>	<pre>from gpiozero import Button  buttonPin = Button(17, bounce_time=0.2) while True:     if buttonPin.is_pressed:         print("Boton presionado")</pre>

**Ejemplo 4:** Realizar un programa que cambie el estado de un led (de ON → OFF y viceversa) cada vez que se presiona un botón. (Aca se hizo uso de las clases [LED](#) y [Button](#)).



Forma tradicional	Empleando la libreria gpiozero
button_led_ex4.py	gpio_zero_button_led_ex4.py
<pre>import RPi.GPIO as GPIO import time  GPIO.setmode(GPIO.BCM)  switch_pin = 18 led_pin = 23  GPIO.setup(switch_pin, GPIO.IN, pull_up_down=GPIO.PUD_UP) GPIO.setup(led_pin, GPIO.OUT)</pre>	<pre>from gpiozero import Button, LED  switch_pin = Button(18, bounce_time=0.2) ledPin = LED(23) ledPin.off()  led_state = False old_input_state = True  while True:     new_input_state = switch_pin.is_pressed</pre>

```

led_state = False
old_input_state = True # pulled-up

try:
    while True:
        new_input_state = GPIO.input(switch_pin)
        if new_input_state == False and old_input_state == True:
            led_state = not led_state
            time.sleep(0.2)
        old_input_state = new_input_state
        GPIO.output(led_pin, led_state)
finally:
    print("Cleaning Up!")
    GPIO.cleanup()

```

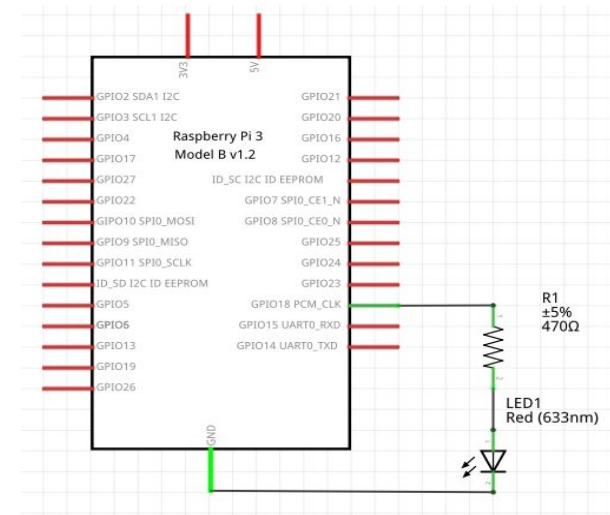
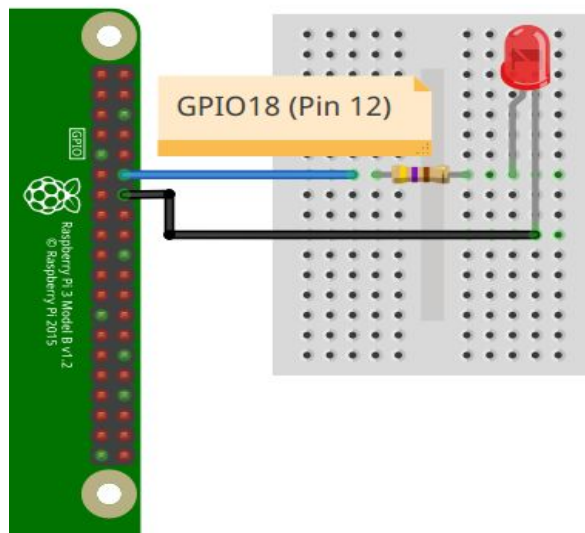
```

if new_input_state == False and old_input_state == True:
    led_state = not led_state
    old_input_state = new_input_state
ledPin.toggle()

```

**Ejemplo 5:** Hacer un programa que permita ingresar el PWM por teclado (0-100) para ir cambiando la intensidad de un led.

Para este caso se empleó la clase [PWMLed](#).



Forma tradicional

Empleando la librería gpiozero



pwm_ex5.py	gpio_zero_pwm_ex5.py
<pre> import RPi.GPIO as GPIO  led_pin = 18 GPIO.setmode(GPIO.BCM) GPIO.setup(led_pin, GPIO.OUT)  pwm_led = GPIO.PWM(led_pin, 500) pwm_led.start(100)  try:     while True:         duty_s = raw_input("Enter Brightness (0 to 100):")         duty = int(duty_s)         pwm_led.ChangeDutyCycle(duty) finally:     print("Cleaning Up!")     pwm_led.stop()     GPIO.cleanup() </pre>	<pre> from gpiozero import PWMLED  led_pin = PWMLED(18) while True:     duty_s = input("Enter Brightness (0 to 100):")     duty = int(duty_s)/100     led_pin.value = duty </pre>

Que hace el siguiente código:

```

from gpiozero import PWMLED
from signal import pause

led = PWMLED(18)

led.pulse()

pause()

```

## Referencias

1. [Simple Electronics with gpiozero](#)

2. [gpiozero](#)
3. [Python and the gpiozero Library](#)