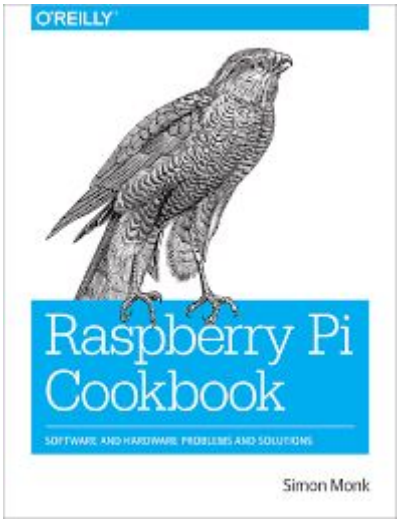
EJEMPLOS RPi

Los ejemplos aquí descritos fueron tomados del libro Raspberry Pi Cookbook: Software and Hardware Problems and Solutions (Puede descargarlo del siguiente link).



El código de este libro se encuentra disponible en el repositorio: https://github.com/iot-udea/raspberrypi_cookbook

Adicionalmente se usó información de la pagina del autor del libro a la cual puede acceder dando click en la siguiente imagen:

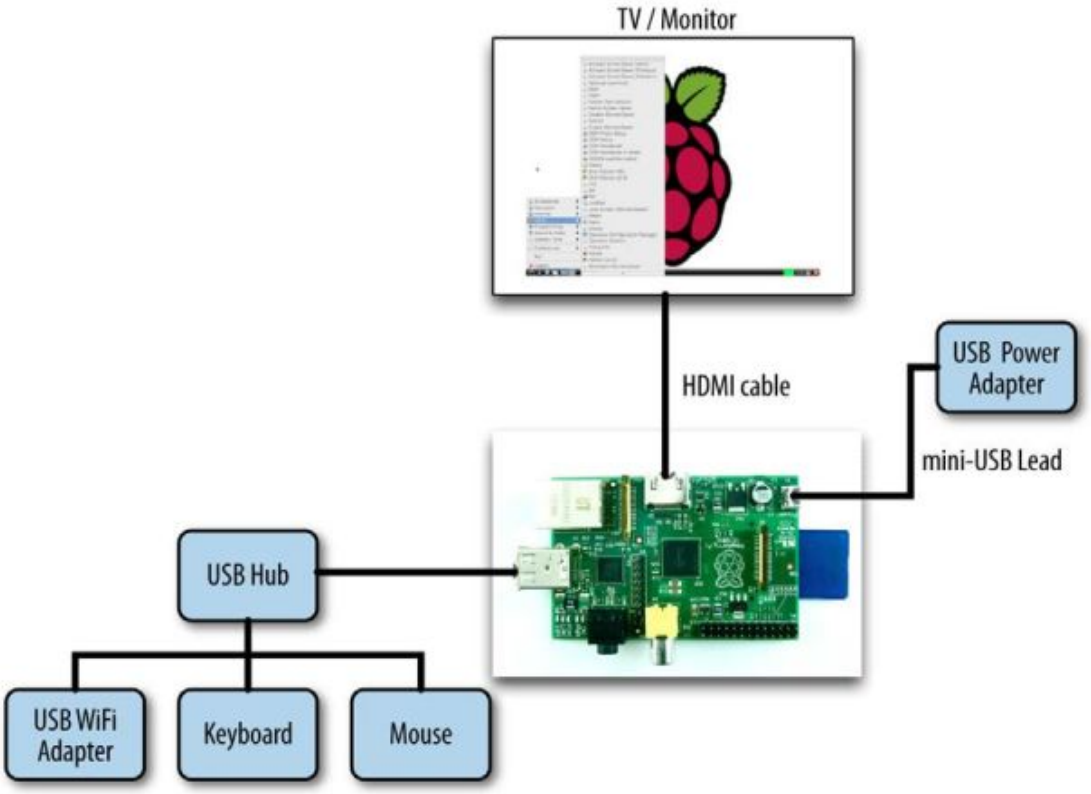
Material a revisar antes de empezar.

Sparkfun posee un material muy interesante (Raspberry gPIo) que resume muy bien todos los contenidos asociados a los puertos GPIO de la raspberry. El material del curso asociado a esta parte se basó en su gran mayoría en éste.

También a modo de resumen en el siguiente directorio se encuentran varias reference sheet de la raspberry, el entenderlas y aplicarlas le facilitará la tarea de programar la raspberry pi para solucionar problemas de la vida cotidiana.

Esquema de conexión para llevar a cabo las pruebas

A continuación se muestra el esquema de conexión de las rPi con los demás elementos necesarios para realizar pruebas típicas de prototipado. Inicialmente es necesario poner en marcha la raspberry (si no lo recuerda ver el siguiente video).



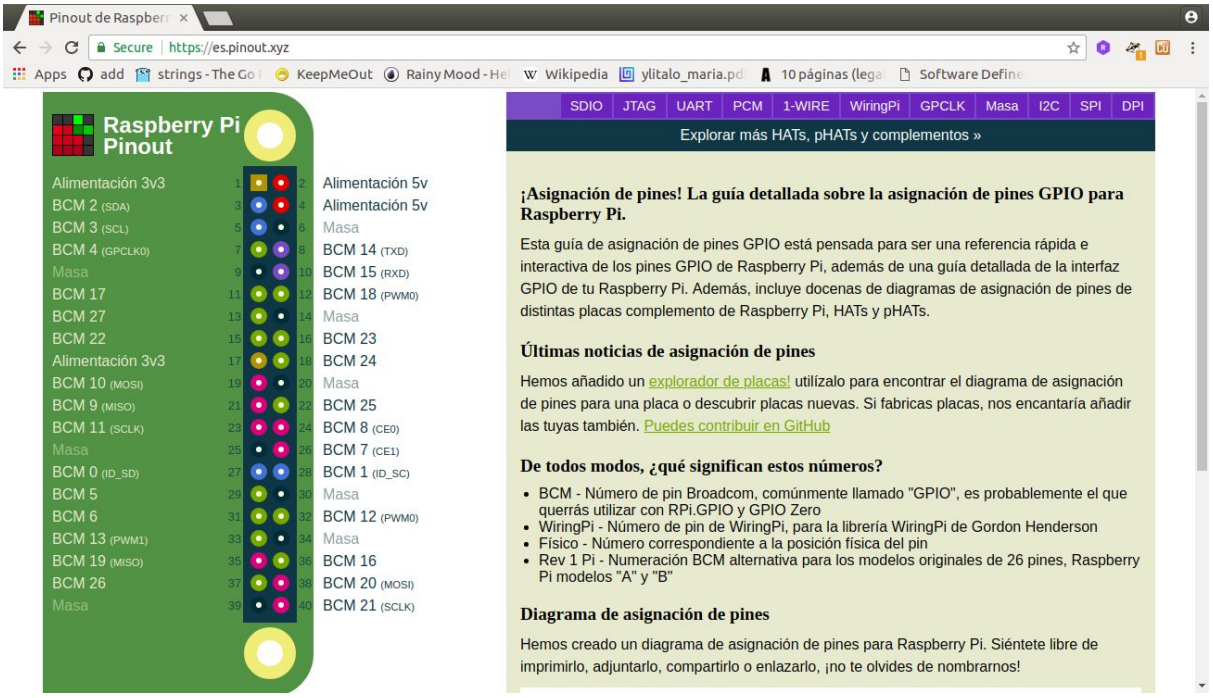
Discribución de pines.

Conocer la distribución de los pines es muy importante para ver cómo conectar dispositivos externos y poder poner a interactuar la rapsberry con el mundo exterior. Para ello se puede emplear el comando:

```
pinout
```

El cual es sumamente útil cuando no se tiene conexión a internet. La figura mostrada a continuación muestra el resultado:

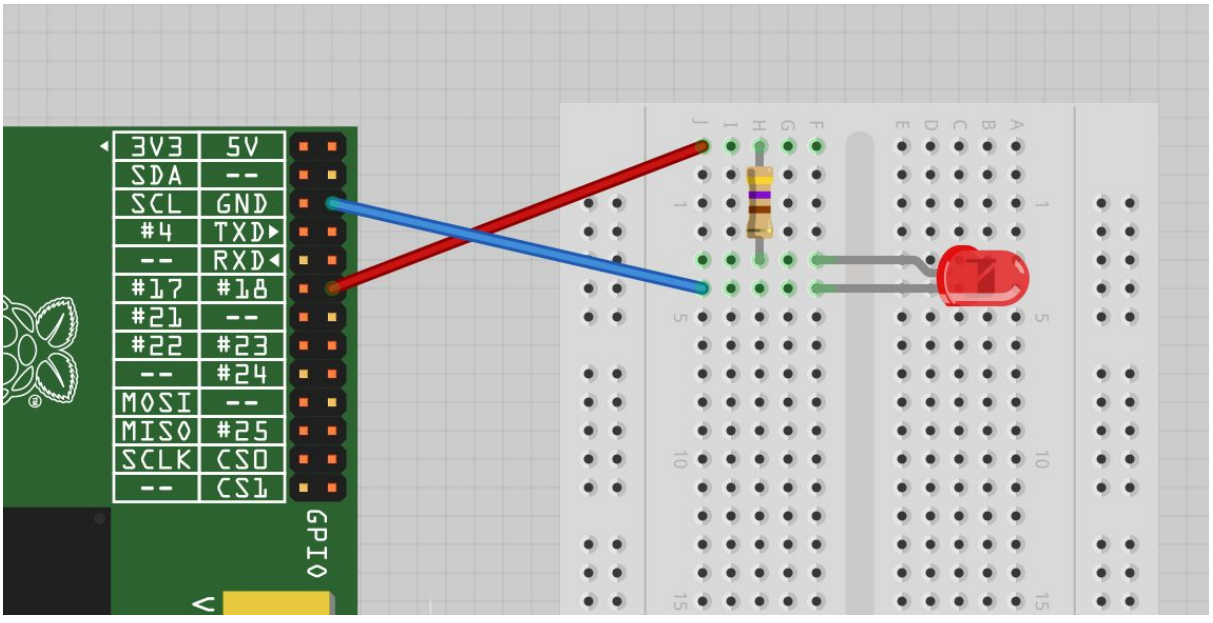
Sin embargo, la siguiente la pagina web https://es.pinout.xyz/ muestra de una manera más clara esta distribución:



Ejemplos

1. **Connecting an LED**: En el siguiente ejemplo se muestra cómo controlar un led (prendiendolo y apagandolo) desde el prompt de python. Para mejor comprensión ver el siguiente enlace: http://razzpisampler.oreilly.com/ch03.html

Hardware: A continuación se muestra la conexión del hardware:



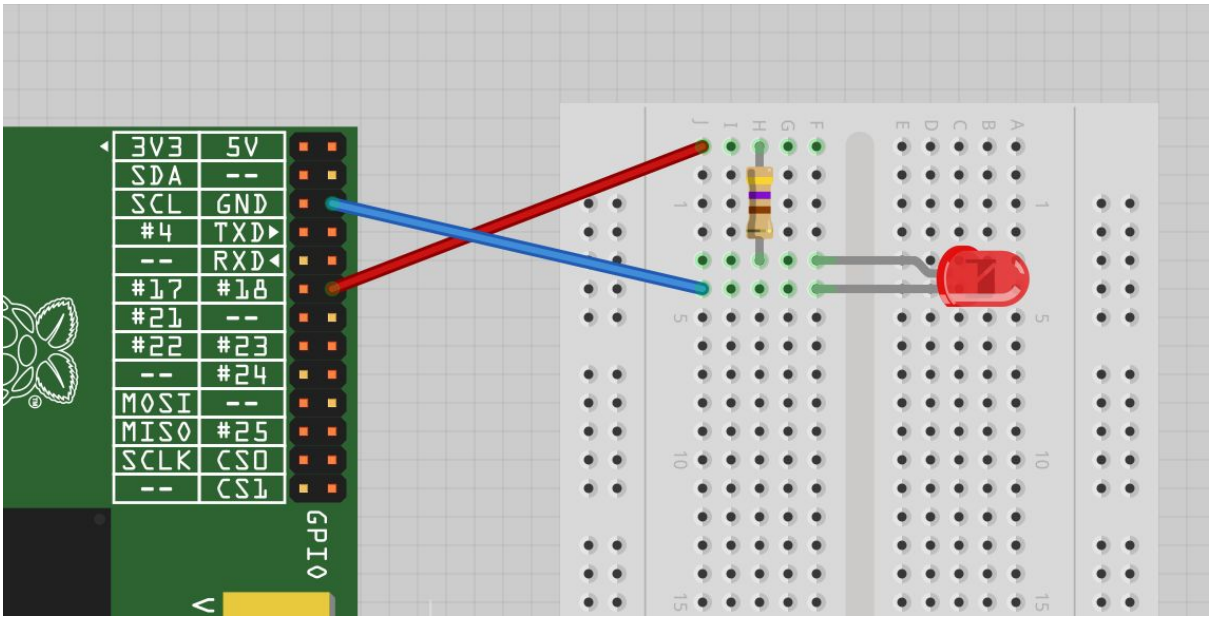
Software: Los comandos digitados en consola se muestran a continuación:

```
$ sudo python
>>> import RPi.GPIO as GPIO
>>> GPIO.setmode(GPIO.BCM)
>>> GPIO.setup(18, GPIO.OUT)
>>> GPIO.output(18, True)
>>> GPIO.output(18, False)
>>> GPIO.cleanup()
>>> exit()
```

Código ejemplo 1. Comandos en consola para el ejemplo del led.

2. **Leaving the GPIO Pins in a Safe State**: Se usa la función GPIO.cleanup() para este fin. El siguiente código muestra un led parpadeando cada 500 ms. A continuación se muestra el código:

Hardware: La conexión entre la rPi y el led se muestra a continuación:



Software: el script de python con la solución se muestra a continuación:

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

GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.OUT)

try:
  while (True):
    GPIO.output(18, True)
    time.sleep(0.5)
    GPIO.output(18, False)
    time.sleep(0.5)
finally:
  print("Cleaning Up!")
  GPIO.cleanup()
```

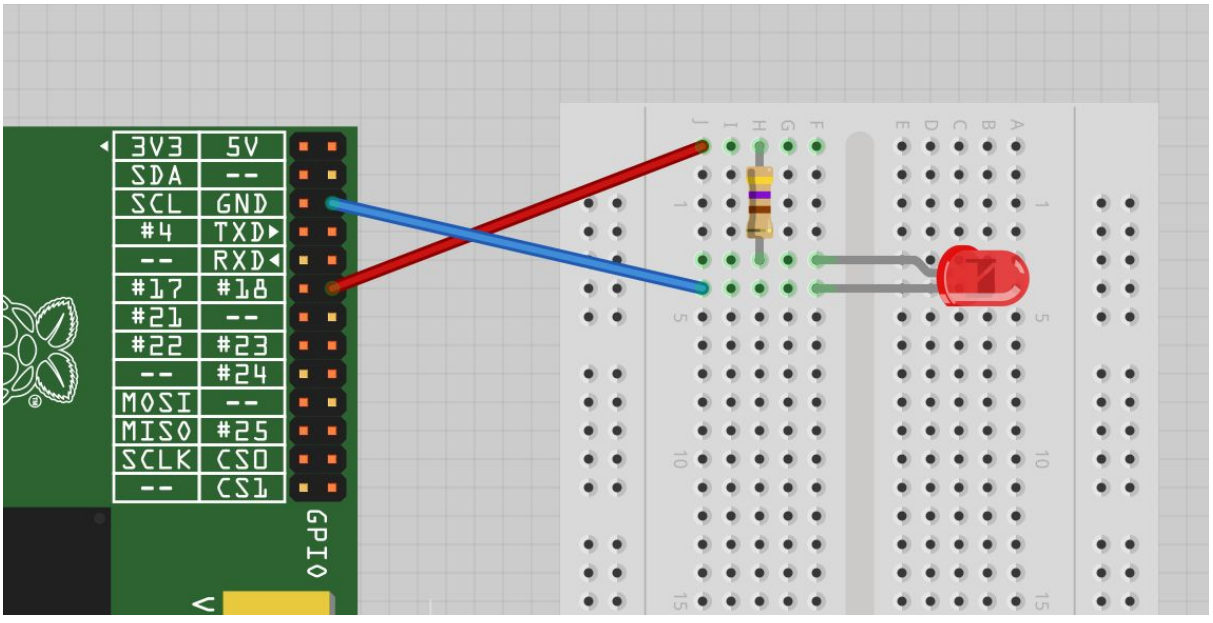
Código ejemplo 2. led_blink_safe.py

Para correr el script anterior se ejecuta en la rPi el siguiente comando:

```
sudo python led_blink_safe.py
```

3. **Controlling the Brightness of an LED**: En el siguiente ejemplo se muestra cómo controlar el brillo de un led mediante el uso de señales PWM empleando la rPi:

Hardware: La conexión se muestra a continuación:



Software: El código asociado se muestra a continuación:

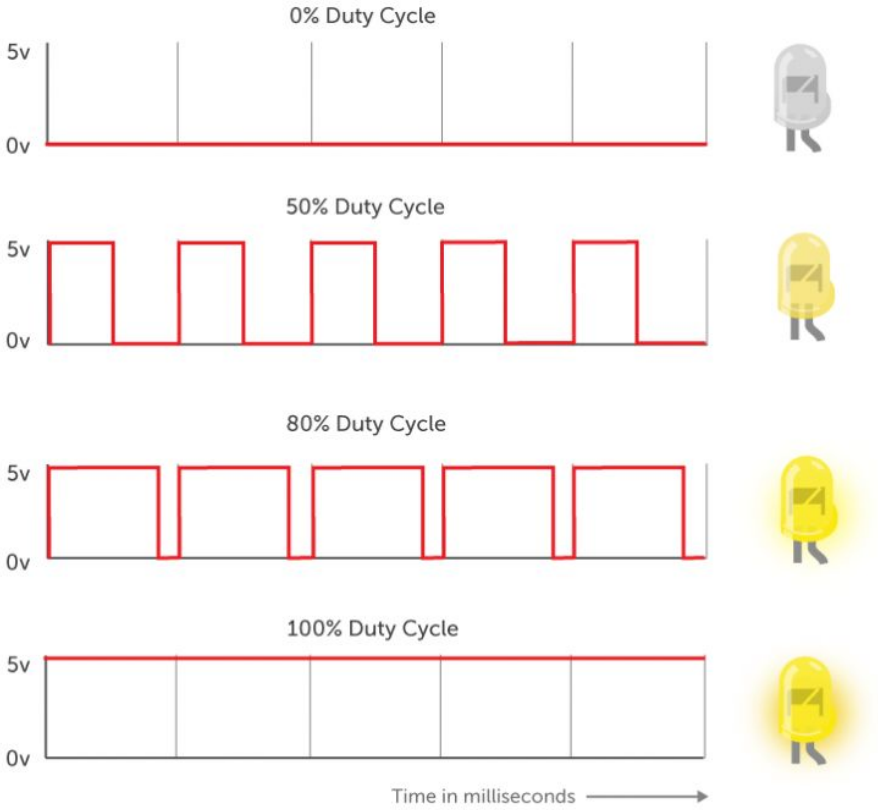
```
import RPi.GPIO as GPIO

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!")
  GPIO.cleanup()
```

Código ejemplo 3. led_brightness.py

Para correr el script anterior se ejecuta en la rPi el siguiente comando:

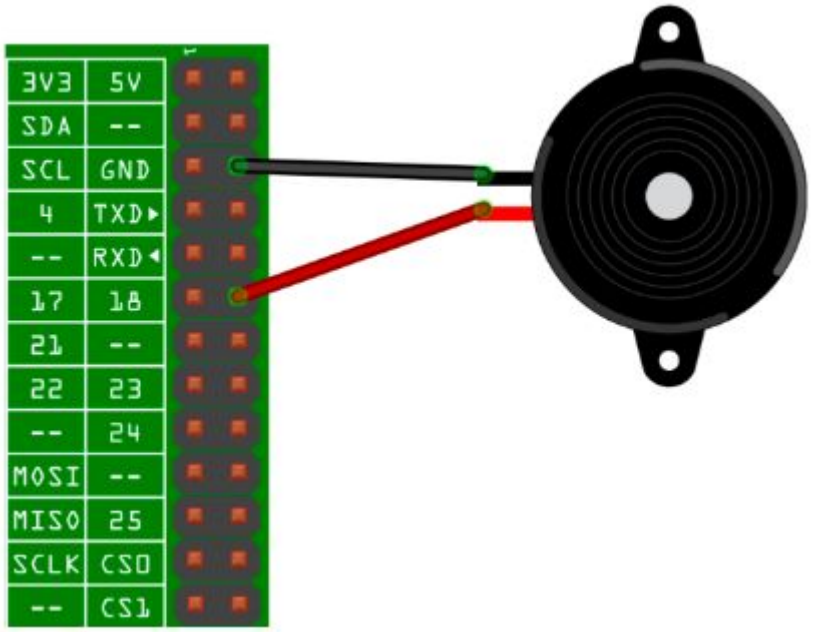
```
sudo python led_brightness.py
```

El resultado sería algo similar al que se muestra en la siguiente gráfica.



4. **Make a Buzzing Sound**: El siguiente ejemplo se emplea un buzzer para generar una señal sonora de acuerdo a la señal pwm aplicada a la salida del puerto.

Hardware:



Software:

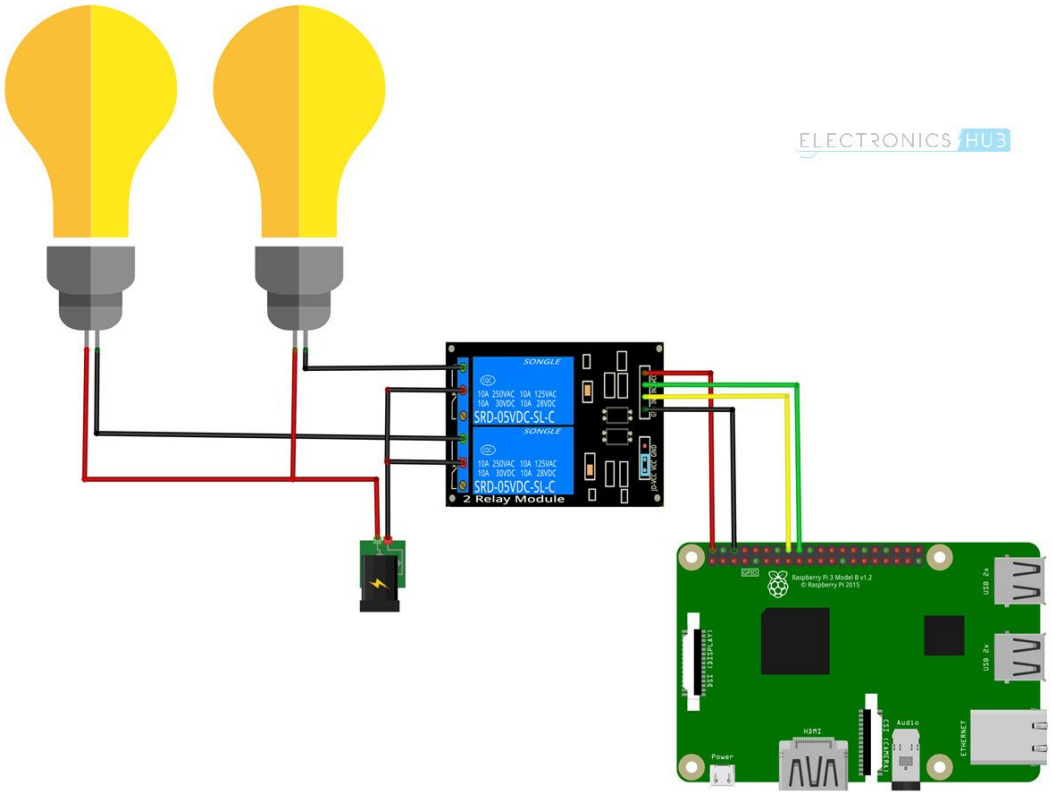
```
import RPi.GPIO as GPIO

buzzer_pin = 18
GPIO.setmode(GPIO.BCM)
GPIO.setup(buzzer_pin, GPIO.OUT)

pwm = GPIO.PWM(buzzer_pin, 500)
pwm.start(100)

try:
  while True:
    duty_s = raw_input("Enter Brightness (0 to 100):")
    duty = int(duty_s)
    pwm.ChangeDutyCycle(duty)
finally:
  print("Cleaning Up!")
  GPIO.cleanup()
```

Código ejemplo 4. buzzer.py

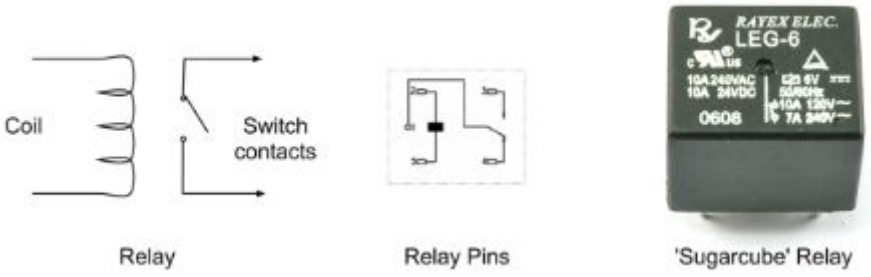
Para correr el script anterior se ejecuta en la rPi el siguiente comando:

```
sudo python buzzer.py
```

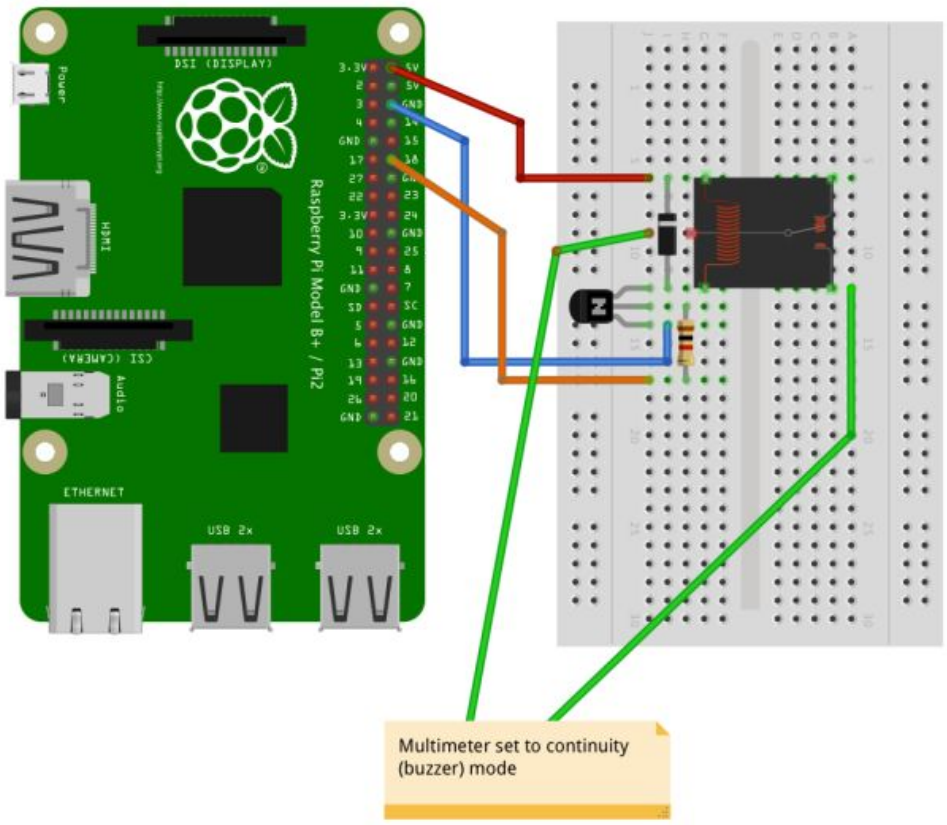
5. **Switching a High-Power Device Using a Relay**: A continuación se muestra un ejemplo que conecta la raspberry con un relay. Un relay es un switch electromagnético que permite que dispositivos que manejan bajas potencias puedan controlar dispositivos que manejan altas potencias. La siguiente figura muestra un caso típico:



Conocer la distribución de pines de un relay también es sumamente importante. A continuación se muestra esta:



Hardware: La conexión asociada al ejemplo se muestra a continuación:



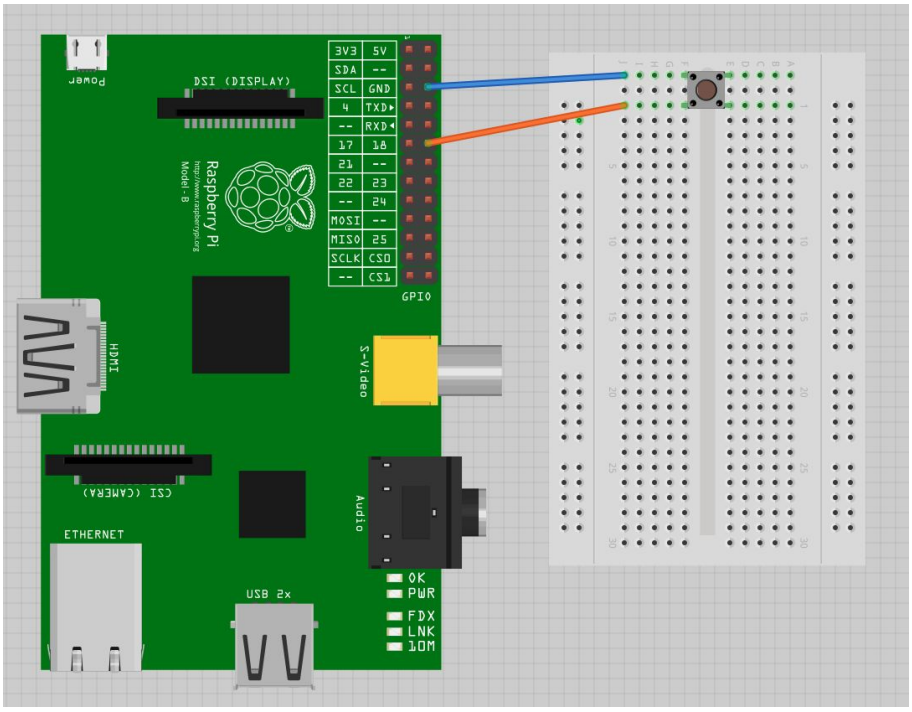
Software: El código de interacción desde la consola de python se muestra a continuación:

```
$ sudo python
>>> import RPi.GPIO as GPIO
>>> GPIO.setmode(GPIO.BCM)
>>> GPIO.setup(18, GPIO.OUT)
>>> GPIO.output(18, True)
>>> GPIO.output(18, False)
>>> GPIO.cleanup()
>>> exit()
```

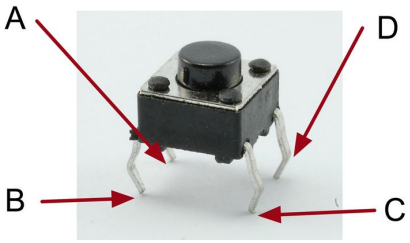
Código ejemplo 5. Comandos en consola para el ejemplo del relay.

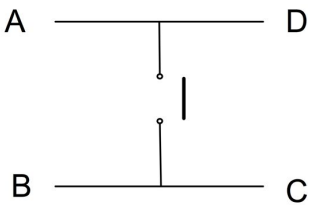
6. **Connecting a Push Switch**: En el siguiente ejemplo se muestra cómo interactúa la raspberry pi con hardware entrada (switch). Para mayor facilidad el ejemplo se encuentra explicado en: http://razzpisampler.oreilly.com/ch07.html#SEC11.1

Hardware: Para este caso el switch está conectado al pull up interno del puerto de la raspberry pi, por ello no se necesita resistencia externa.



El esquemático del switch se muestra a continuación:





Software: A continuación se muestra el software:

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

GPIO.setup(18, GPIO.IN, pull_up_down=GPIO.PUD_UP)

try:
  while True:
    input_state = GPIO.input(18)
    if input_state == False:
      print('Button Pressed')
      time.sleep(0.2)
    else:
      print('Button Released')
      time.sleep(0.2)
finally:
  print("Cleaning Up!")
  GPIO.cleanup()
```

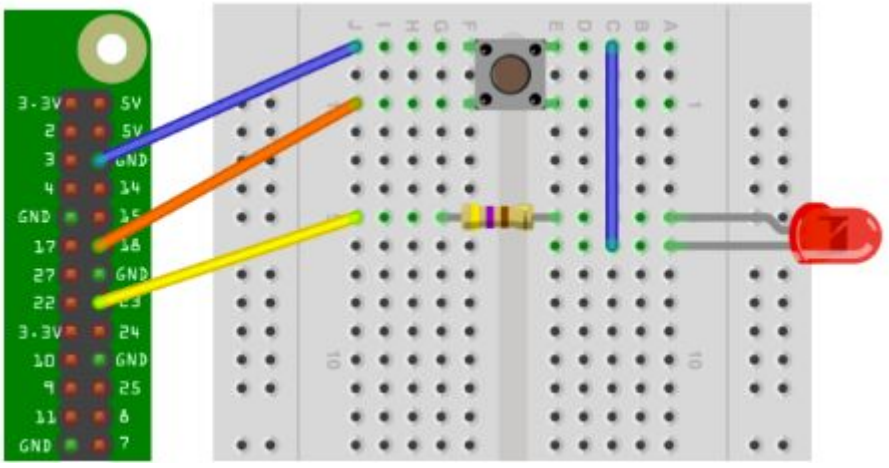
Código ejemplo 6. switch.py

Para correr el script anterior se ejecuta en la rPi el siguiente comando:

```
sudo python switch.py
```

7. **Toggling with a Push Switch**: El siguiente programa cambia el estado de un led (ON → OFF y OFF → ON) cada vez que el botón es presionado.

Hardware:



Software:

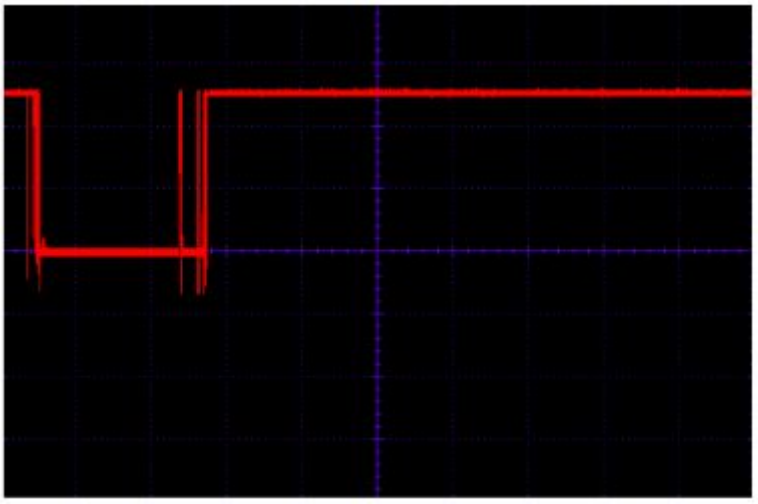
```
import RPi.GPIO as GPIO
import time

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)

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
    old_input_state = new_input_state
    GPIO.output(led_pin, led_state)
finally:
  print("Cleaning Up!")
  GPIO.cleanup()
```

Código ejemplo 7. switch_on_off.py

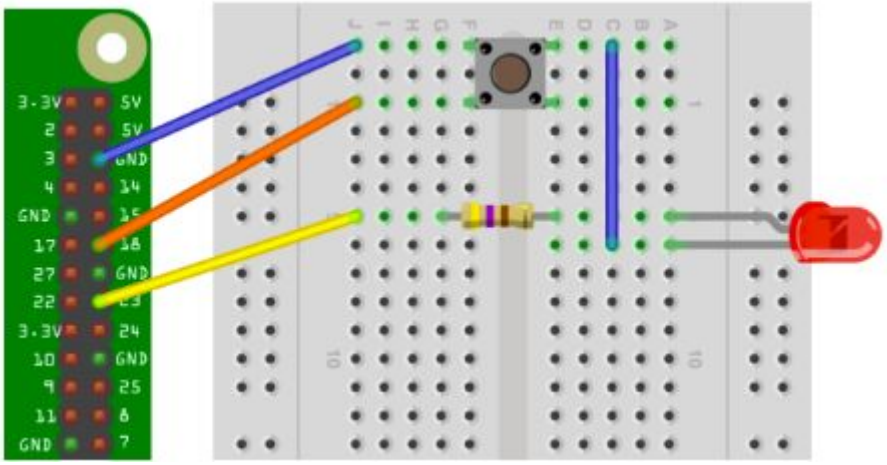
Para correr el script anterior se ejecuta en la rPi el siguiente comando:

```
sudo python switch_on_off.py
```

8. **Debouncing a Button Press**: En el ejemplo anterior, no se trata un problema que se genera cuando se hace uso de switches conocido como el rebote. Como su nombre lo dice, el rebote es un fenomeno que consiste en la oscilación de la señal electrica de entrada debido a las oscilaciones del switch mecanico tal y como se muestra en la siguiente figura:



Hardware: La siguiente figura muestra el diagrama de conexión que es el mismo del ejemplo anterior.



Software: El programa asociado a este problema soluciona el problema del rebote agregando al código del ejemplo anterior la línea resaltada en amarillo y que se muestra a continuación:

```
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)

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()
```

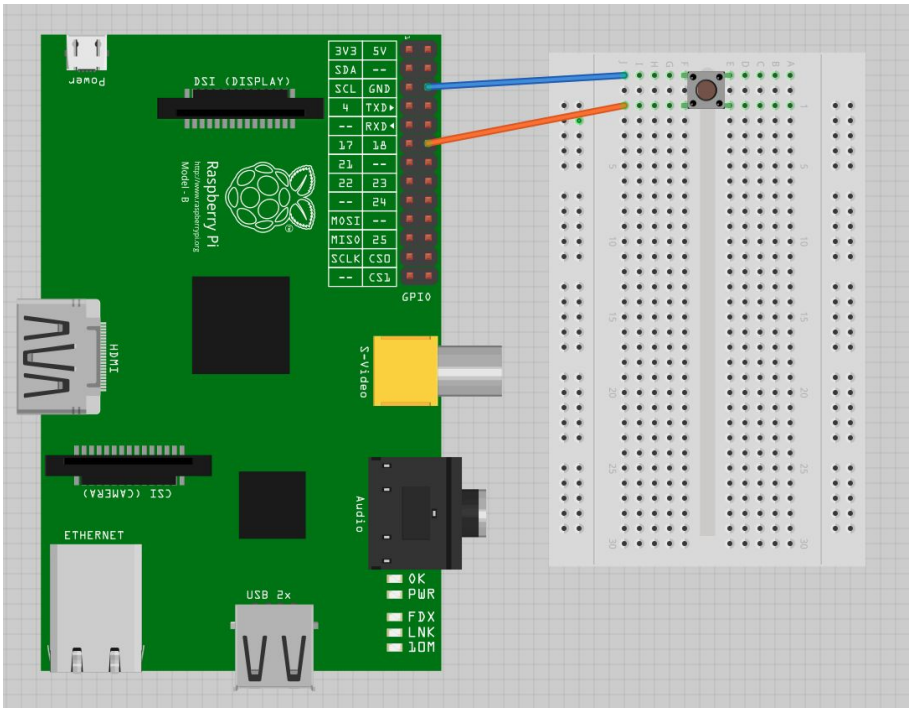
Código ejemplo 8. switch_on_off_no_bounce.py

Para correr el script anterior se ejecuta en la rPi el siguiente comando:

```
sudo python switch_on_off_no_bounce.py
```

9. **Programming with Interrupts**: Una interrupción es una señal que indica cuando se da cierto evento a lo largo de la ejecución del programa y que requiere la atención del procesador antes de que el programa. La atención a la interrupción es llevada a cabo por una función de atención a la interrupción. Para asociar la rutina de atención de interrupción al evento de botón presionado, se emplea la función **add_event_detect**. El ejemplo mostrado a continuación A continuación se muestra el código asociado.

Hardware:



Software: Observe con detalle las partes del código que se encuentran resaltadas.

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

def my_callback(channel):
  print('You pressed the button')

GPIO.setup(18, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.add_event_detect(18, GPIO.FALLING, callback=my_callback)

try:
  i = 0
  while True:
    i = i + 1
    print(i)
```

```
    time.sleep(1)
finally:
  print("Cleaning Up!")
  GPIO.cleanup()
```

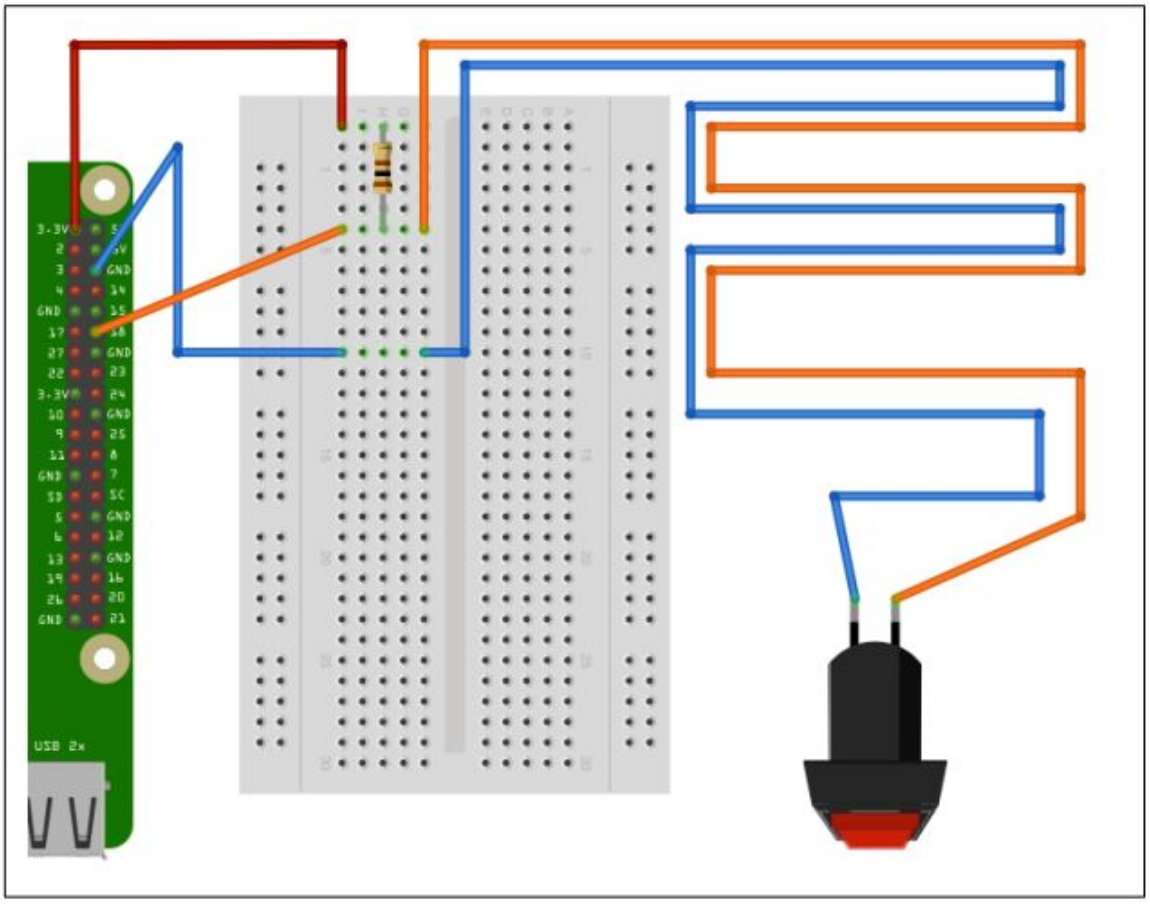
Código ejemplo 9. switch_count_irq.py

Para correr el script anterior se ejecuta en la rPi el siguiente comando:

```
sudo python switch_count_irq.py
```

10. **Using an External Pull-up Resistor**: Un resistor de pull-up sirve para asignar a una entrada un voltaje de entrada alto por default. El siguiente ejemplo muestra cada vez que el botón se presiona y libera mediante un mensaje en consola.

Hardware:



Software:

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

# GPIO.setup(18, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(18, GPIO.IN)

try:
  while True:
    input_state = GPIO.input(18)
    if input_state == False:
      print('Button Pressed')
      time.sleep(0.2)
    else:
      print('Button Released')
      time.sleep(0.2)
finally:
  print("Cleaning Up!")
  GPIO.cleanup()
```

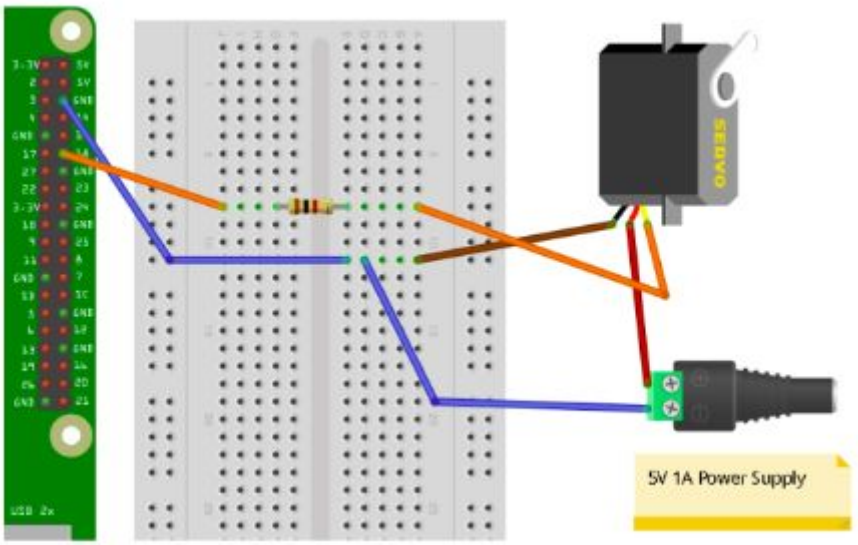
Código ejemplo 10. switch_pull_up_resistor.py

Para correr el script anterior se ejecuta en la rPi el siguiente comando:

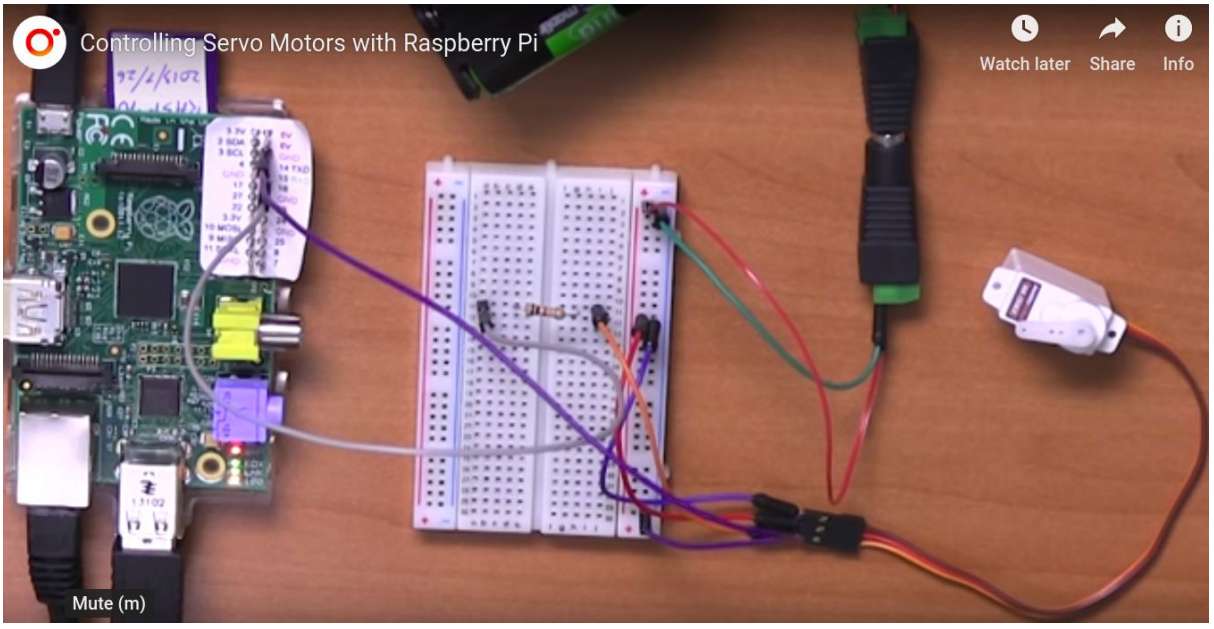
```
sudo python switch_pull_up_resistor.py
```

11. **Controlling Servo Motors**: En el siguiente ejemplo se muestra cómo controlar un servomotor desde una rPi. Para mas claridad en el enlace: http://razzpisampler.oreilly.com/ch05.html#SEC7.12 se explica con lujo de detalles.

Hardware: A continuación se muestra el diagrama de conexión del ejemplo:



En el siguiente video se muestra el sistema en funcionamiento:



Software: Para hacer más amigable el programa se hace uso de una interfaz gráfica codificada usando la librería tkinter para controlar el ángulo de movimiento del servomotor. En la siguiente figura se muestra la interfaz gráfica de usuario empleada para controlar el servo:

Por otro lado a continuación se muestra el codigo asociado al ejemplo:

```
from Tkinter import *
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.OUT)
pwm = GPIO.PWM(18, 100)
pwm.start(5)

class App:
  def __init__(self, master):
    frame = Frame(master)
    frame.pack()
    scale = Scale(frame, from_=0, to=180, \
                  orient=HORIZONTAL, command=self.update)
    scale.grid(row=0)


  def update(self, angle):
    duty = float(angle) / 10.0 + 2.5
    pwm.ChangeDutyCycle(duty)

root = Tk()
```

```
root.wm_title('Servo Control')
app = App(root)
root.geometry("200x50+0+0")
root.mainloop()
```

Código ejemplo 11. servo.py

Para correr el script anterior se ejecuta en la rPi el siguiente comando:

```
sudo python servo.py
```

Enlaces web

1. What are the min/max voltage/current values the gpio pins can handle?
2. GPIO Electrical Specifications - Raspberry Pi input and output pin voltage and current capability
3. Taller de Raspberry Pi
4. Power Supply
5. GPIO
6. What are the Electrical Specifications of GPIO pins?
7. How do I power my Raspberry Pi?
8. Raspberry Pi Servo Motor control

Enlaces web sin organizar

1. https://projects.raspberrypi.org/en/projects/physical-computing/3
2. http://simonmonk.org/
3. http://razzpisampler.oreilly.com/
4. https://github.com/simonmonk
5. https://github.com/simonmonk/raspberrypi_cookbook_ed2
6. https://learn.sparkfun.com/tutorials/raspberry-gpio/all
7. https://www.programoergosum.com/cursos-online/raspberry-pi/238-control-de-gpio-con-python-en-raspberry-pi/led-regulado
8. https://electronics.stackexchange.com/questions/104456/understanding-gpio-analog-and-digital
9. https://learn.sparkfun.com/tutorials/raspberry-gpio/c-wiringpi-setup
10. https://github.com/simonmonk/pi_analog
11. https://github.com/simonmonk/monk_raspberrypi
12. http://www.mosaic-industries.com/embedded-systems/microcontroller-projects/raspberry-pi/gpio-pin-electrical-specifications
13. https://www.adafruit.com/product/757
14. https://www.raspberrypi.org/forums/viewtopic.php?t=196531
15. https://github.com/raspberrypi/documentation
16. https://github.com/pubnub/workshop-raspberrypi

17. https://github.com/pubnub/workshop-raspberrypi/tree/master/projects-python/helloworld
18. https://github.com/pubnub/workshop-raspberrypi/tree/master/projects-python/led
19. https://github.com/pubnub/workshop-raspberrypi/tree/master/projects-python/remote-led
20. https://github.com/pubnub/workshop-raspberrypi/tree/master/projects-python/remote-led
21. https://github.com/pubnub/workshop-raspberrypi/tree/master/projects-python
22. https://github.com/googlecreativelab/coder
23. https://github.com/initialstate/pi-process-dashboard
24. https://github.com/gregtinkers/carspeed.py
25. https://tutorials-raspberrypi.com/raspberry-pi-servo-motor-control/
26. https://maker.pro/raspberry-pi/projects/how-to-build-an-internet-radio-station-with-raspberry-pi-darkice-and-icecast
27. https://www.instructables.com/id/Raspberry-Pi-Internet-Radio/
28. https://www.hackster.io/pranciskus/internet-radio-station-f0bdc1
29. https://stmllr.net/blog/streaming-audio-with-mpd-and-icecast2-on-raspberry-pi/
30. https://maker.pro/raspberry-pi/projects/how-to-build-an-internet-radio-station-with-raspberry-pi-darkice-and-icecast
31. https://stmllr.net/blog/streaming-audio-with-mpd-and-icecast2-on-raspberry-pi/
32. https://www.youtube.com/watch?v=g-jGM48xUrs
33. https://www.instructables.com/id/Arduino-Raspberry-Pi-Internet-Radio/
34. https://learn.adafruit.com/adabox004/internet-radio
35. https://www.hackster.io/user3330224/pi-arduino-internet-radio-b372fe
36. https://www.fossmint.com/best-radio-streaming-apps-for-linux/
37. https://www.ubuntupit.com/top-10-best-radio-streaming-software-for-linux-system/
38. https://www.cloudrad.io/radio-broadcasting-software/
39. http://ubuntuhandbook.org/index.php/2013/07/airtime-free-open-source-radio-automation-for-ubuntu-linux/
40. https://mediarealm.com.au/articles/open-source-broadcast-software-github/
41. https://www.linux.org/threads/create-your-own-linux-radio-station.4572/
42. https://www.vultr.com/docs/install-icecast-on-ubuntu-18-04
43. https://www.howtoforge.com/linux_webradio_with_icecast2_ices2
44. https://www.smarthomebeginner.com/icecast-ices2-music-server-for-raspberry-pi-raspberry-pi-streaming-audio-server/
45. https://maker.pro/raspberry-pi/projects/how-to-build-an-internet-radio-station-with-raspberry-pi-darkice-and-icecast
46. https://tutorials-raspberrypi.com/raspberry-pi-servo-motor-control/
47. https://www.instructables.com/id/Servo-Motor-Control-With-Raspberry-Pi/
48. http://fpaez.com/controlar-un-servomotor-con-raspberry-pi/
49. https://rpi.science.uoit.ca/lab/servo/
50. https://raspberrypi.stackexchange.com/questions/33845/controlling-servos-with-a-raspberry-pi