



INSTITUTO TECNOLÓGICO SUPERIOR  
ZACATECAS OCCIDENTE  
Manual de Usuario

**Manual de Usuario**  
**SemaforosMina**  
**Murma INC**

---

**HISTORIAL DE VERSIONES**

VERSIÓN	FECHA VIGENCIA	DETALLE DEL CAMBIO	SECCIÓN CAMBIADA	AUTOR	FECHA AUTORIZACIÓN
0.1	04/05/2023	Creación del documento	Ninguna	HDCS, RAPR Y RBMJ	04/05/2023
1.0	10/05/2023	Validación del documento	Ninguna	HDCS, RAPR Y RBMJ	10/05/2023



# INSTITUTO TECNOLÓGICO SUPERIOR ZACATECAS OCCIDENTE

## Manual de Usuario

### Manual de usuario

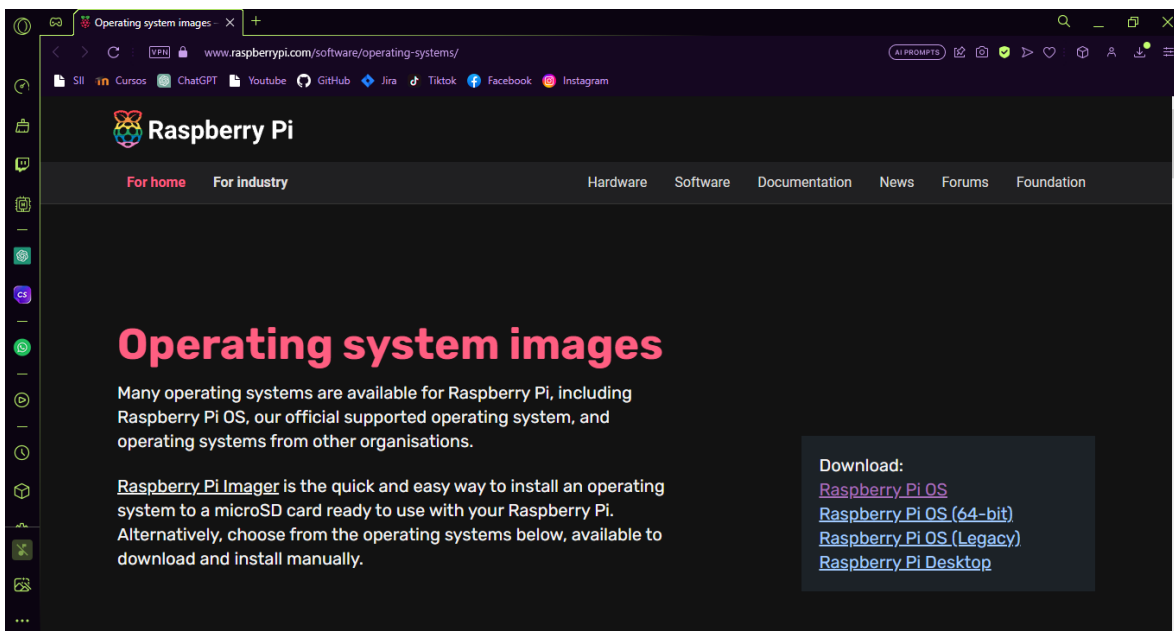
#### Introducción

Los semáforos de dos luces para interior de mina son dispositivos diseñados para mejorar la seguridad en entornos mineros, permitiendo regular el tráfico y la circulación de personas. Este manual proporciona instrucciones detalladas sobre cómo utilizar y configurar los semáforos utilizando una Raspberry Pi 4 y Python. Así como una breve explicación de cómo es que trabajan estos códigos.

Instalación de SO en raspberry:

Preparación de la tarjeta SD:

Descarga la imagen de Raspberry Pi OS desde el sitio web oficial de Raspberry Pi (<https://www.raspberrypi.org/software/operating-systems/>).



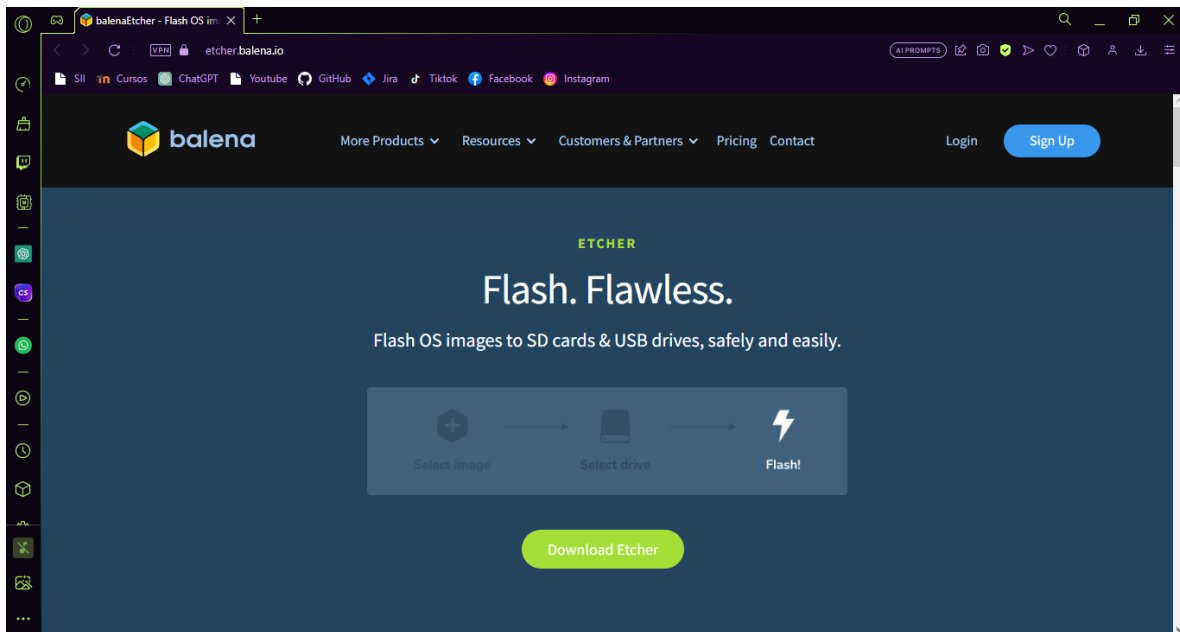


# INSTITUTO TECNOLÓGICO SUPERIOR ZACATECAS OCCIDENTE

## Manual de Usuario

Inserta la tarjeta SD en tu computadora.

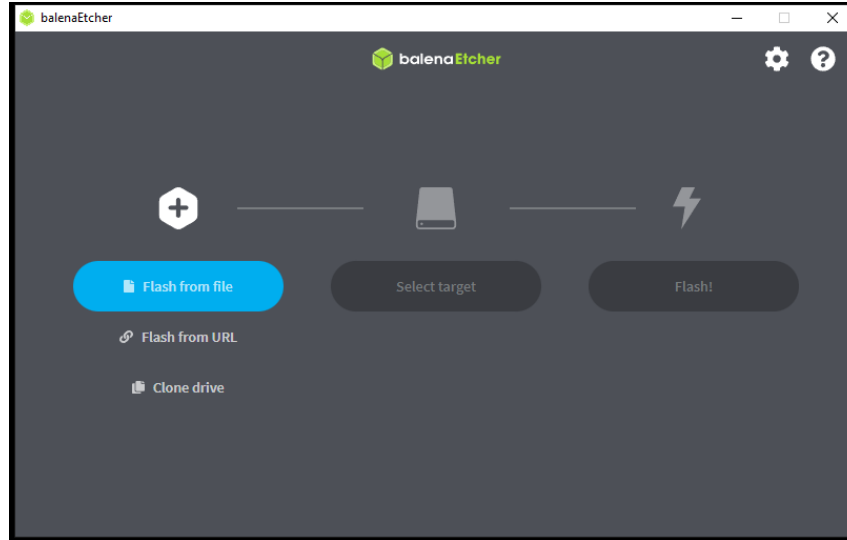
Utiliza un programa como Etcher (disponible en <https://www.balena.io/etcher/>) para escribir la imagen de Raspberry Pi OS en la tarjeta SD. Selecciona la imagen descargada y la tarjeta SD en Etcher, y haz clic en "Flash" para comenzar el proceso de escritura. Ten en cuenta que este proceso borrará todos los datos de la tarjeta SD, así que asegúrate de hacer una copia de seguridad de cualquier archivo importante.





# INSTITUTO TECNOLÓGICO SUPERIOR ZACATECAS OCCIDENTE

## Manual de Usuario



### 1. Configuración inicial de Raspberry Pi:

- Retira la tarjeta SD de tu computadora y colócala en la ranura correspondiente de la Raspberry Pi.
- Conecta el teclado, el mouse, el monitor y el cable de alimentación a la Raspberry Pi.
- Enciende la Raspberry Pi conectando el cable de alimentación.
- Raspberry Pi OS se iniciará y te llevará a la pantalla de configuración inicial.

### 1. Configuración de Raspberry Pi OS:

- Selecciona el idioma deseado y haz clic en "Siguiente".
- Asegúrate de seleccionar tu país o región correcta y haz clic en "Siguiente".
- Configura la conexión Wi-Fi si es necesario.



## INSTITUTO TECNOLÓGICO SUPERIOR ZACATECAS OCCIDENTE Manual de Usuario

- Establece una contraseña para el usuario "pi" (el nombre de usuario predeterminado) y haz clic en "Siguiente".
- Configura la configuración regional, como la zona horaria y el formato de fecha y hora.
- Haz clic en "Siguiente" y, si es necesario, configura las actualizaciones de software y la instalación de programas recomendados.
- Una vez completada la configuración, se te pedirá que reinicies la Raspberry Pi. Haz clic en "Finalizar" y la Raspberry Pi se reiniciará.

### Instalación de OpenCV en raspberry pi

Actualiza el sistema operativo de la raspberry pi ejecutando los siguientes comandos:

```
sudo apt update
```

```
sudo apt upgrade
```

Después instala las dependencias usando este comando:

```
sudo apt install libopencv-dev python3-opencv
```

Puedes verificar la instalación con el siguiente comando:

```
python3 -c "import cv2; print(cv2.__version__)"
```

Si no hubo percances, se imprimirá su versión de OpenCV

(la instalación de gpio se puede consultar en “manual de mantenimiento”)

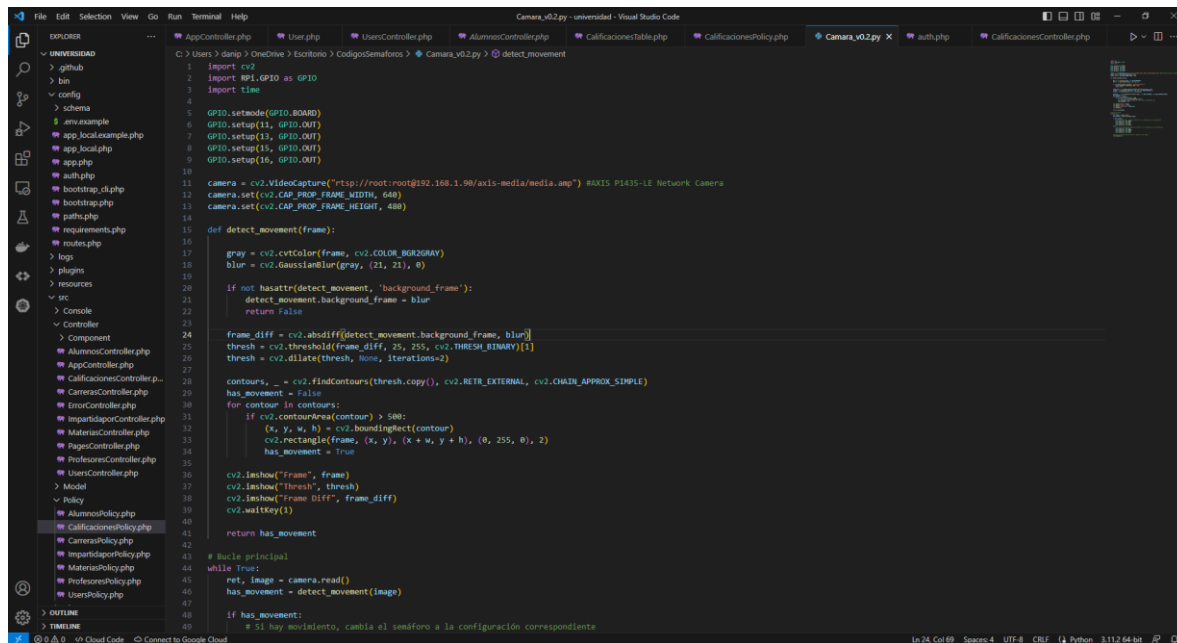


# INSTITUTO TECNOLÓGICO SUPERIOR ZACATECAS OCCIDENTE

## Manual de Usuario

### Códigos:

Este código utiliza la biblioteca OpenCV (cv2) y la biblioteca RPi.GPIO para detectar movimiento en un flujo de video proveniente de una cámara IP (AXIS P1435-LE Network Camera) y controlar el estado de pines GPIO en una Raspberry Pi. A continuación, se explica paso a paso:



```
1 import cv2
2 import RPi.GPIO as GPIO
3 import time
4
5 GPIO.setmode(GPIO.BOARD)
6 GPIO.setup(11, GPIO.OUT)
7 GPIO.setup(13, GPIO.OUT)
8 GPIO.setup(15, GPIO.OUT)
9 GPIO.setup(16, GPIO.OUT)
10
11 camera = cv2.VideoCapture("rtsp://root:root@192.168.1.90/axis-media/media.amp")
12 camera.set(cv2.CAP_PROP_FRAME_WIDTH, 640)
13 camera.set(cv2.CAP_PROP_FRAME_HEIGHT, 480)
14
15 def detect_movement(frame):
16     gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
17     blur = cv2.GaussianBlur(gray, (21, 21), 0)
18
19     if not hasattr(detect_movement, 'background_frame'):
20         detect_movement.background_frame = blur
21         return False
22
23     frame_diff = cv2.absdiff(detect_movement.background_frame, blur)
24     thresh = cv2.threshold(frame_diff, 25, 255, cv2.THRESH_BINARY)[1]
25     thresh = cv2.dilate(thresh, None, iterations=2)
26
27     contours, _ = cv2.findContours(thresh.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
28     has_movement = False
29     for contour in contours:
30         if cv2.contourArea(contour) > 500:
31             (x, y, w, h) = cv2.boundingRect(contour)
32             cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)
33             has_movement = True
34
35     cv2.imshow("Frame", frame)
36     cv2.imshow("Thresh", thresh)
37     cv2.imshow("Frame Diff", frame_diff)
38     cv2.waitKey(1)
39
40     return has_movement
41
42 # Bucla principal
43 while True:
44     ret, image = camera.read()
45     has_movement = detect_movement(image)
46
47     if has_movement:
48         # Si hay movimiento, cambia el semáforo a la configuración correspondiente
```

Se crea un objeto de captura de video utilizando cv2.VideoCapture para conectarse a la cámara IP. Se especifica la dirección RTSP de la cámara y se configura el ancho y alto del cuadro capturado.

```
camera = cv2.VideoCapture("rtsp://root:root@192.168.1.90/axis-media/media.amp")
```

```
camera.set(cv2.CAP_PROP_FRAME_WIDTH, 640)
```

```
camera.set(cv2.CAP_PROP_FRAME_HEIGHT, 480)
```

Se define una función llamada detect\_movement que toma un cuadro de imagen (frame) como entrada y realiza la detección de movimiento utilizando técnicas de procesamiento de imagen.

El cuadro de imagen se convierte a escala de grises y se aplica un desenfoque gaussiano.



# INSTITUTO TECNOLÓGICO SUPERIOR ZACATECAS OCCIDENTE

## Manual de Usuario

Si no se ha almacenado un cuadro de fondo previo (background\_frame), se guarda el cuadro actual en background\_frame y se devuelve False.

Se calcula la diferencia entre el cuadro de fondo y el cuadro actual (frame\_diff).

Se aplica un umbral a la diferencia para obtener una imagen binaria (thresh) que resalta las regiones de movimiento.

Se dilatan las regiones en la imagen binaria para conectar píxeles cercanos.

Se encuentran los contornos en la imagen binaria y se comprueba si tienen un área mayor a 500 píxeles. Si es así, se dibuja un rectángulo alrededor del contorno en el cuadro original (frame) y se establece has\_movement en True.

```
def detect_movement(frame):
```

```
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
```

```
    blur = cv2.GaussianBlur(gray, (21, 21), 0)
```

```
    if not hasattr(detect_movement, 'background_frame'):
```

```
        detect_movement.background_frame = blur
```

```
    return False
```

```
    frame_diff = cv2.absdiff(detect_movement.background_frame, blur)
```

```
    thresh = cv2.threshold(frame_diff, 25, 255, cv2.THRESH_BINARY)[1]
```

```
    thresh = cv2.dilate(thresh, None, iterations=2)
```

```
    contours, _ = cv2.findContours(thresh.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
```

```
    has_movement = False
```

```
    for contour in contours:
```

```
        if cv2.contourArea(contour) > 500:
```

```
            (x, y, w, h) = cv2.boundingRect(contour)
```

```
            cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)
```

```
            has_movement = True
```

```
    cv2.imshow("Frame", frame)
```

```
    cv2.imshow("Thresh", thresh)
```

```
    cv2.imshow("Frame Diff", frame_diff)
```

```
    cv2.waitKey(1)
```

```
    return has_movement
```



# INSTITUTO TECNOLÓGICO SUPERIOR ZACATECAS OCCIDENTE

## Manual de Usuario

Se inicia un bucle principal infinito (while True) que captura continuamente cuadros de imagen de la cámara y realiza la detección de movimiento.

- Se lee un cuadro de imagen de la cámara y se almacena en la variable image.
- Se llama a la función detect\_movement para verificar si hay movimiento en el cuadro.
- Si se detecta movimiento, se cambia el estado de los pines GPIO para indicar una configuración específica del semáforo.
- Si no se detecta movimiento, se cambia el estado de los pines GPIO para indicar otra configuración del semáforo.
- Se espera un breve período de tiempo (0.1 segundos) antes de capturar el siguiente cuadro de imagen.

En resumen, este código configura los pines GPIO, establece una conexión con una cámara IP, realiza la detección de movimiento en el flujo de video y controla el estado de los pines GPIO para simular un semáforo que responde al movimiento detectado.

```
while True:
```

```
    ret, image = camera.read()
```

```
    has_movement = detect_movement(image)
```

```
    if has_movement:
```

```
        GPIO.output(11, GPIO.HIGH)
```

```
        GPIO.output(13, GPIO.LOW)
```

```
        GPIO.output(15, GPIO.LOW)
```

```
        GPIO.output(16, GPIO.HIGH)
```

```
    else:
```

```
        GPIO.output(11, GPIO.LOW)
```

```
        GPIO.output(13, GPIO.HIGH)
```

```
        GPIO.output(15, GPIO.HIGH)
```





# INSTITUTO TECNOLÓGICO SUPERIOR ZACATECAS OCCIDENTE

## Manual de Usuario

```
GPIO.output(16, GPIO.LOW)
```

```
time.sleep(0.1)
```

```
Camara_v02.py - universidad - Visual Studio Code
File Edit Selection View Go Run Terminal Help
C:\Users\danip> danip > OneDrive > CódigosSemáforos > Camara_v02.py > detect_movement
1 import RPi.GPIO as GPIO
2 import time
3
4 # Configuramos los pines GPIO para los colores del semáforo 1
5 GPIO.setmode(GPIO.BCM)
6 GPIO.setwarnings(False)
7 ROJO_1_PIN = 18
8 VERDE_1_PIN = 23
9 GPIO.setup(ROJO_1_PIN, GPIO.OUT)
10 GPIO.setup(VERDE_1_PIN, GPIO.OUT)
11
12 # Configuramos los pines GPIO para los colores del semáforo 2
13 ROJO_2_PIN = 24
14 VERDE_2_PIN = 25
15 GPIO.setup(ROJO_2_PIN, GPIO.OUT)
16 GPIO.setup(VERDE_2_PIN, GPIO.OUT)
17
18 # Funciones para cambiar el color del semáforo 1
19 def cambiar_a_rojo_1():
20     GPIO.output(VERDE_1_PIN, GPIO.LOW)
21     GPIO.output(ROJO_1_PIN, GPIO.HIGH)
22
23 def cambiar_a_verde_1():
24     GPIO.output(ROJO_1_PIN, GPIO.LOW)
25     GPIO.output(VERDE_1_PIN, GPIO.HIGH)
26
27 # Funciones para cambiar el color del semáforo 2
28 def cambiar_a_rojo_2():
29     GPIO.output(VERDE_2_PIN, GPIO.LOW)
30     GPIO.output(ROJO_2_PIN, GPIO.HIGH)
31
32 def cambiar_a_verde_2():
33     GPIO.output(ROJO_2_PIN, GPIO.LOW)
34     GPIO.output(VERDE_2_PIN, GPIO.HIGH)
35
36 # Programa principal
37 while True:
38     print("seleccione el color del semáforo 1:")
39     print("1. Rojo")
40     print("2. Verde")
41     opcion = input("Ingrese el número de la opción que desea: ")
42     if opcion == "1":
43         cambiar_a_rojo_1()
44         cambiar_a_verde_2() # Cambiar semáforo 2 a rojo
45     elif opcion == "2":
46         cambiar_a_verde_1()
47         cambiar_a_rojo_2() # Cambiar semáforo 2 a verde
48     else:
49         print("Opción no válida")
50
51 # Detectar movimiento
52 def detect_movement():
53     frame = camera.read()
54     has_movement = detect_movement(frame)
55     if has_movement:
56         # Si hay movimiento, cambia el semáforo a la configuración correspondiente
57         GPIO.output(11, GPIO.HIGH)
58         GPIO.output(15, GPIO.LOW)
59         GPIO.output(16, GPIO.LOW)
60     else:
61         # Si no hay movimiento, cambia el semáforo a la configuración correspondiente
62         GPIO.output(11, GPIO.LOW)
63         GPIO.output(15, GPIO.HIGH)
64         GPIO.output(16, GPIO.HIGH)
65     # Espera un tiempo antes de volver a capturar otra imagen
66     time.sleep(0.1)
```

Este código muestra un ejemplo de cómo controlar dos semáforos utilizando la biblioteca RPi.GPIO en una Raspberry Pi. A continuación, se explica paso a paso:

```
Sem.py - Visual Studio Code
File Edit Selection View Go Run Terminal Help
C:\Users\danip> danip > Sem.py
1 import RPi.GPIO as GPIO
2 import time
3
4 # Configuramos los pines GPIO para los colores del semáforo 1
5 GPIO.setmode(GPIO.BCM)
6 GPIO.setwarnings(False)
7 ROJO_1_PIN = 18
8 VERDE_1_PIN = 23
9 GPIO.setup(ROJO_1_PIN, GPIO.OUT)
10 GPIO.setup(VERDE_1_PIN, GPIO.OUT)
11
12 # Configuramos los pines GPIO para los colores del semáforo 2
13 ROJO_2_PIN = 24
14 VERDE_2_PIN = 25
15 GPIO.setup(ROJO_2_PIN, GPIO.OUT)
16 GPIO.setup(VERDE_2_PIN, GPIO.OUT)
17
18 # Funciones para cambiar el color del semáforo 1
19 def cambiar_a_rojo_1():
20     GPIO.output(VERDE_1_PIN, GPIO.LOW)
21     GPIO.output(ROJO_1_PIN, GPIO.HIGH)
22
23 def cambiar_a_verde_1():
24     GPIO.output(ROJO_1_PIN, GPIO.LOW)
25     GPIO.output(VERDE_1_PIN, GPIO.HIGH)
26
27 # Funciones para cambiar el color del semáforo 2
28 def cambiar_a_rojo_2():
29     GPIO.output(VERDE_2_PIN, GPIO.LOW)
30     GPIO.output(ROJO_2_PIN, GPIO.HIGH)
31
32 def cambiar_a_verde_2():
33     GPIO.output(ROJO_2_PIN, GPIO.LOW)
34     GPIO.output(VERDE_2_PIN, GPIO.HIGH)
35
36 # Programa principal
37 while True:
38     print("seleccione el color del semáforo 1:")
39     print("1. Rojo")
40     print("2. Verde")
41     opcion = input("Ingrese el número de la opción que desea: ")
42     if opcion == "1":
43         cambiar_a_rojo_1()
44         cambiar_a_verde_2() # Cambiar semáforo 2 a rojo
45     elif opcion == "2":
46         cambiar_a_verde_1()
47         cambiar_a_rojo_2() # Cambiar semáforo 2 a verde
48     else:
49         print("Opción no válida")
50
```



# INSTITUTO TECNOLÓGICO SUPERIOR ZACATECAS OCCIDENTE

## Manual de Usuario

1. Se importa la biblioteca RPi.GPIO para acceder a las funcionalidades de los pines GPIO de la Raspberry Pi. (la instalación de gpio se puede consultar en “manual de mantenimiento”)
2. Se importa la biblioteca time para utilizar la función sleep y generar retrasos en el programa.
3. Se configuran los pines GPIO para los colores del semáforo 1 y 2:
4. Se utiliza el modo BCM para referirse a los pines GPIO.
5. Se desactivan las advertencias.
6. Se definen las variables para los pines de los colores ROJO y VERDE del semáforo 1 y 2.
7. Se configuran los pines GPIO como salidas.
8. Se definen las funciones para cambiar el color del semáforo 1 y 2:
9. Las funciones cambiar\_a\_rojo\_1() y cambiar\_a\_verde\_1() controlan los pines GPIO del semáforo 1.
10. Las funciones cambiar\_a\_rojo\_2() y cambiar\_a\_verde\_2() controlan los pines GPIO del semáforo 2.
11. Al llamar a estas funciones, se establece el estado de los pines correspondientes para cambiar el color del semáforo.
12. En el programa principal (dentro de un bucle while True), se muestra un menú para seleccionar el color del semáforo 1:
13. Se muestra el menú con las opciones "1. Rojo" y "2. Verde".
14. Se solicita al usuario que ingrese el número de la opción deseada.
15. Dependiendo de la opción seleccionada, se llama a las funciones correspondientes para cambiar el color del semáforo 1 y 2.
16. Si se ingresa una opción inválida, se muestra un mensaje de error.
17. Se utiliza la función sleep para esperar 1 segundo antes de volver a mostrar el menú y preguntar por el color del semáforo 1. Esto proporciona un intervalo de tiempo entre los cambios de color

## Especificaciones técnicas:

### Raspberry Pi 4:

- Procesador: Broadcom BCM2711, Quad-core Cortex-A72 (ARM v8) 64-bit SoC
- Memoria: 2GB/4GB/8GB LPDDR4-3200 SDRAM
- Conectividad: Wi-Fi 802.11ac, Bluetooth 5.0, Ethernet Gigabit
- Cuatro puertos USB 2.0
- Dos puertos USB 3.0



# INSTITUTO TECNOLÓGICO SUPERIOR ZACATECAS OCCIDENTE

## Manual de Usuario

- Un puerto HDMI
- Un puerto de audio de 3.5 mm
- Un puerto de video compuesto (RCA)
- Dos puertos micro-HDMI
- Un puerto de cámara (CSI)
- Un puerto de pantalla táctil (DSI)
- Un puerto de tarjeta microSD
- Un puerto GPIO (General Purpose Input/Output)
- Un puerto de alimentación USB-C

### AXIS P1435-LE Network Camera:

- Resolución de video: Full HD 1080p (1920 x 1080 píxeles).
- Sensor de imagen: CMOS de 1/2,8".
- Iluminación mínima: 0,08 lux en color, 0,04 lux en blanco y negro.
- Rango dinámico amplio (WDR) con tecnología Forensic Capture.
- Lente varifocal y enfoque remoto.
- Funcionalidad día/noche con filtro de corte infrarrojo removible (ICR).
- Compresión de video: H.264 y Motion JPEG.
- Conectividad: Ethernet y compatibilidad con PoE (Power over Ethernet).
- Protección contra condiciones ambientales: resistente al polvo y agua (IP66) y carcasa resistente a los impactos (IK10).

Código de conexión del semaforo con la aplicación



# INSTITUTO TECNOLÓGICO SUPERIOR ZACATECAS OCCIDENTE

## Manual de Usuario

```
C: > Users > hp > Desktop > main.py > InputB
1  from flask import Flask , request
2  import socket
3  import requests
4  from flask_cors import CORS
5  import RPi.GPIO as GPIO
6
7  GPIO.setmode(GPIO.BCM)
8  GPIO.setup(2, GPIO.OUT)
9  GPIO.setup(3, GPIO.OUT)
10
11 def obtener_direccion_ip():
12     hostname = socket.gethostname()
13     direccion_ip = socket.gethostbyname(hostname)
14     return direccion_ip
15 """
16 def ping ():
17     Nododata = {'Ip': obtener_direccion_ip(), 'Nombre':"REBAJE 720" }
18     resp = requests.post('http://192.168.100.31:7000/Control/1', data=Nododata)
19     print(resp)
20 """
21 direccion_ip = obtener_direccion_ip()
22
23
24 app = Flask(__name__)
25 cors = CORS(app, resources={r"*": {"origins": "*"}})
26
27 @app.route("/")
28 def home():
29     return "high"
```

```
30
31
32
33
34
35
36
37 @app.route("/Input/B")
38 def InputB():
39     GPIO.output(2, RELAY_OFF)
40     GPIO.output(3, RELAY_ON)
41
42     return "True"
43
44 if __name__ == "__main__":
45     #ping()
46     app.run(host='192.168.100.56', port=7000)
```

Al iniciar este código iniciaremos un servidor de slack, por medio de este servidor nos conectaremos a la raspberry usando su dirección IP y el puerto 7000 de la raspberry.



## INSTITUTO TECNOLÓGICO SUPERIOR ZACATECAS OCCIDENTE Manual de Usuario

Como podemos observar, se encuentran dos módulos que es el input (A) y el input (B), los cuales cuando el semaforo este en el input (A) estará encendido el verde y en el input (B) el rojo, cuando se cambie la indicación en el input (A) estará prendido el rojo y en el input (B) el verde