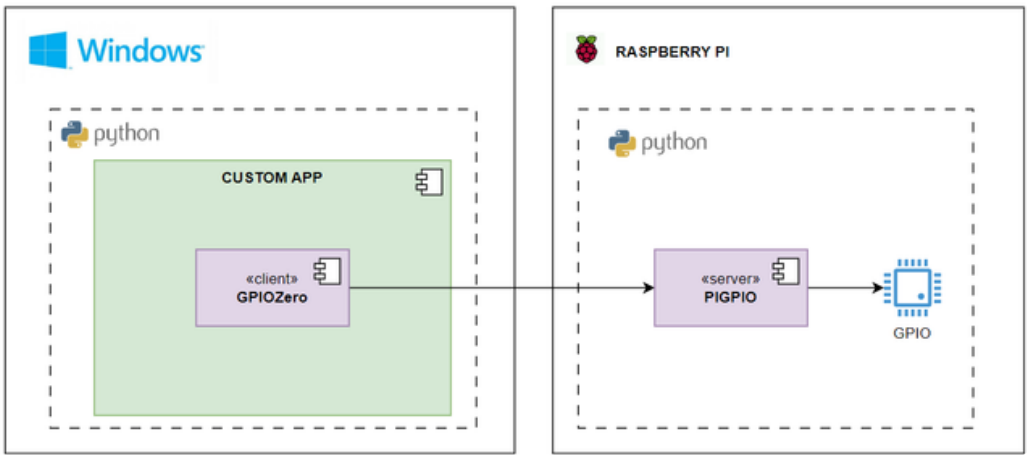
2. Instalar PIGPIO en Raspberry Pi

Resumen

Como primera etapa, se definió avanzar con la instalación de PIGPIO en Raspberry PI para comprobar que la librería funciona con el hardware real. En el siguiente diagrama se muestra de forma simplificada la tarea que se pretender llevar a cabo:



De esta manera, podremos validar que la biblioteca PIGPIO funciona correctamente dentro del Raspberry PI y que la biblioteca GPIOZero permite conectarse a PIGPIO para poder enviar o recibir información del GPIO físico del equipo.

Desarrollo

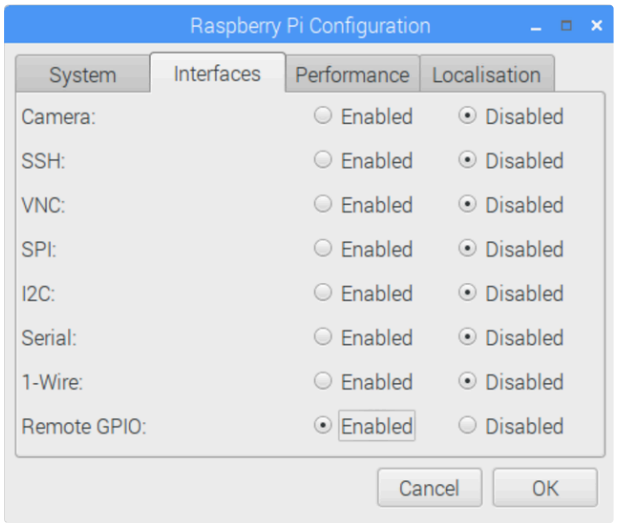
Instalación lado Servidor

Basado en la página de GPIOZero, el primer paso fue instalar PIGPIO en el Raspberry Pi ejecutando el siguiente comando:

```
sudo apt install pigpio
```

Luego habilitamos las conexiones remotas del Raspberry Pi. Esto se puede hacer de 2 maneras.

- **EN FORMA GRAFICA:**



- **DESDE LA LINEA DE COMANDOS:**

Desde la línea comandos se debe ejecutar lo siguiente:

```
sudo raspi-config
```

Al ejecutar esta instrucciones se abrira una pantalla de selccion de opciones. Luego elegir la opcion 3 *"Interface Options"* y despues habilitar la opcion de "Remote GPIO"

Una vez que habilitamos Remote GPIO, automatizamos la ejecución de PIGPIO al inicio:

```
sudo systemctl enable pigpiod
```

Luego ejecutamos el daemon de PIGPIO:

```
sudo systemctl start pigpiod
```

Si se desea ejecutar en primer plano, se debe usar

```
sudo pigpiod
```

Instalación lado Cliente

Basado en la página de GPIOZero, una vez que tenemos corriendo el servidor en Raspberry PI, el primer paso del lado del cliente fue actualizar los paquetes del SO:

```
sudo apt update
```

Luego instalamos GPIO Zero y la biblioteca PGPIO para Python 3:

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

Una vez realizada la instalación, procedemos a instalarlo en python:

```
sudo pip3 install gpiozero pigpio
```

Completada la instalación de la biblioteca, se realizaron algunas pruebas con diferentes programas python para verificar la funcionalidad.

El primero que se probó fue utilizando el `PiGPIOFactory` de la siguiente manera:

Blinkled.py

```python
from gpiozero import LED
from gpiozero.pins.pigpio import PiGPIOFactory
from time import sleep

factory = PiGPIOFactory(host='192.168.235.25')
led = LED(24, pin_factory=factory)

while True:
    print ("prende")
    led.on()
    sleep(1)
    print ("apaga")
    led.off()
    sleep(1)
```

Luego se probó de otra manera seteano variablees de entorno desde el código para intentar dejar el código lo más parecido al código fuente original:

ButtonLed-pgpio V1.py

```
from gpiozero import Button, LED
from gpiozero.pins.pigpio import PiGPIOFactory
import os
from time import sleep

os.environ["PIGPIO_ADDR"] = "192.168.235.25"
os.environ["GPIOZERO_PIN_FACTORY"] = "pigpio"
def main():

    def button_pressed():
        print("button pressed!")
        led2.toggle()

    led2 = LED(24)
    button = Button(23)
    button.when_pressed = button_pressed


    while True:
        sleep(0.1)

main()
```

ButtonLed-pgpio V2.py

```
from gpiozero import Button, LED
from gpiozero.pins.pigpio import PiGPIOFactory
import os
import time

os.environ["PIGPIO_ADDR"] = "192.168.235.25"
os.environ["GPIOZERO_PIN_FACTORY"] = "pigpio"

button = Button(23)
led = LED(24)

led.source = button

while True:
   time.sleep(1)
```

Por último, se probó la funcionalida de un servo motor con el siguiente código:

Servo_Test.py

```
from gpiozero import Servo
from time import sleep
import os

os.environ["GPIOZERO_PIN_FACTORY"] = "pigpio"
os.environ["PIGPIO_ADDR"] = "192.168.30.12"
```

```
7
8  myGPIO = 18
9
10 servo = Servo(myGPIO)
11 print("Rassberry Pi Servo")
12 while True:
13     servo.min()
14     print("min")
15     sleep(1)
16     servo.mid()
17     print("mid")
18     sleep(1)
19     servo.max()
20     print("max")
21     sleep(1)
```

Instalación de PIGPIO desde codigo fuente

Se probo el funcionamiento de pigpio en la Raspberry fisica(Solo en forma local), instalandolo desde el codigo fuente . Para ello los pasos realizados fueron los siguientes;

1. `git clone https://github.com/joan2937/pigpio`
2. dentro del directorio descargado usar el comando el make
3. Luego se inicio el demonio ejecutando comando creado en el mimo directorio

`./pigpiod`

4: Se probo el siguiente script en python, para prender y apagar un led en forma local

```
1  import pigpio
2  import time
3
4  # Conectar al demonio de pigpio
5  pi = pigpio.pi()
6
7  # Definir el número del pin GPIO al que está conectado el LED
8  led_pin = 17
9
10 # Configurar el pin como salida
11 pi.set_mode(led_pin, pigpio.OUTPUT)
12
13 # Función para encender el LED
14 def encender_led():
15     pi.write(led_pin, 1)
16     print("LED encendido")
17
18 # Función para apagar el LED
19 def apagar_led():
20     pi.write(led_pin, 0)
21     print("LED apagado")
22
23 try:
24     # Encender el LED por 5 segundos
25     encender_led()
26     time.sleep(5)
27
```

```
    # Apagar el LED por 5 segundos
    apagar_led()
    time.sleep(5)

finally:
    # Limpiar y desconectar
    pi.write(led_pin, 0)  # Asegúrate de que el LED esté apagado
    pi.stop()

```

5. Luego con el comando se chequeo desde la consola si prendia o se apagaba el led, observando el campo level

```
raspi-gpio get 17
```

Conclusiones 👇

Finalmente, los resultados de todas las pruebas fueron satisfactorios. Por lo que se decidió avanzar con el siguiente paso de la investigación.

Enlaces

Scripts:

emulador_raspberry/scripts/pgpio at qemu-gpio-zero · soa-pc-unlam/emulador_raspberry

Pinouts:

Raspberry Pi 3 Model B GPIO 40 Pin Block Pinout

W3Schools.com

Pasos instalacion pigpio

4. Configuring Remote GPIO — gpiozero 2.0.1 Documentation