

4. Probar pigpio en Qemu

Se siguieron los siguientes pasos:

1. De acuerdo a la URL [pigpio library](#) el port default es 8888, por lo que debemos dejar abierto ese puerto en el contenedor. Para eso se debe ejecutar el comando:

```
1 docker run -it -p 5022:5022 -p 8888:8888 soaunlam/emulador-raspberry-cmd:v2
```

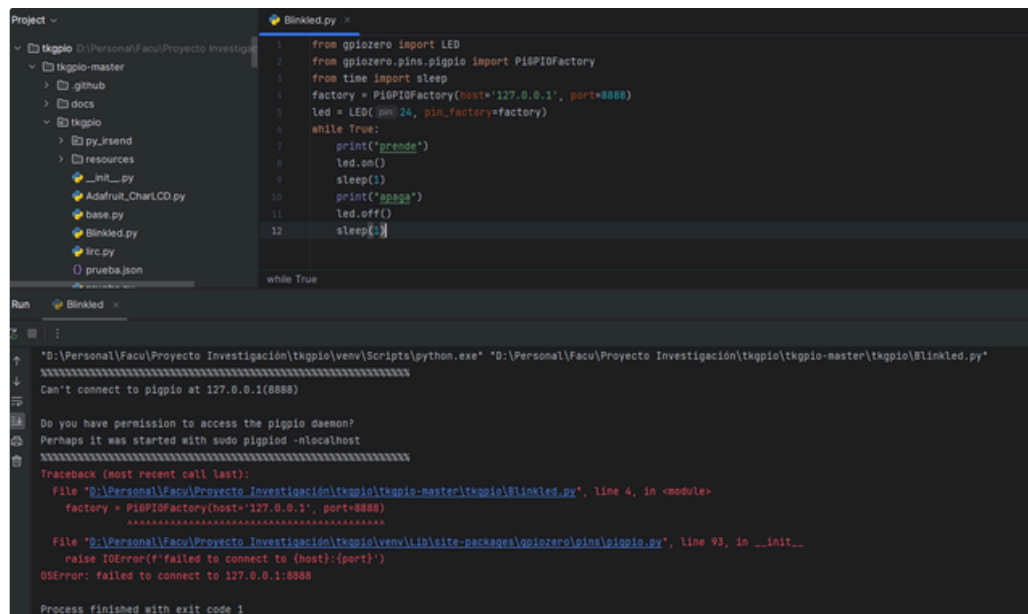
El conenedor deberá quedar con los puertos 5022 y 8888 expuestos como en la siguiente imagen:

	NAME	IMAGE	STATUS	PORT(S)
	admiring_borg 21e3613d3c36	soaunlam/emulador-raspber	Running	5022:5022 8888:8888

2. Realizar los pasos descritos en [2. Instalar PIGPIO en Raspberry Pi](#)
3. Una vez instalado chequeamos los estados de los puertos

```
1 raspi-gpio get
2 raspi-gpio get 0-18
3 raspi-gpio set 18 op
4 raspi-gpio set 18 op dh
5 raspi-gpio get 0-18
6
```

4. Por otro lado ejecuté el comando **ss** que permite ver los puertos usados de Raspbian, pero no vi el 8888.
5. Por las dudas, ejecuté el programa en Python por fuera de docker, pero sigue sin funcionar.



```
Project
└─ tkgpio
   └─ tkgpio-master
      ├── github
      ├── docs
      └─ tkgpio
         ├── py_arsend
         ├── resources
         ├── _init_.py
         ├── Adafruit_CharLCD.py
         ├── base.py
         ├── Blinkled.py
         ├── src.py
         └─ prueba.json

Blinkled.py
1 from gpiozero import LED
2 from gpiozero.pins.pigpio import PiGPIOFactory
3 from time import sleep
4 factory = PiGPIOFactory(host='127.0.0.1', port=8888)
5 led = LED(24, pin_factory=factory)
6 while True:
7     print('prende')
8     led.on()
9     sleep(1)
10    print('apaga')
11    led.off()
12    sleep(1)

while True

Run
Blinkled
"D:\Personal\Facu\Proyecto Investigación\tkgpio\venv\Scripts\python.exe" "D:\Personal\Facu\Proyecto Investigación\tkgpio\tkgpio-master\tkgpio\Blinkled.py"
Can't connect to pigpio at 127.0.0.1(8888)

Do you have permission to access the pigpio daemon?
Perhaps it was started with sudo pigpiod -nlocalhost

Traceback (most recent call last):
  File "D:\Personal\Facu\Proyecto Investigación\tkgpio\tkgpio-master\tkgpio\Blinkled.py", line 4, in <module>
    factory = PiGPIOFactory(host='127.0.0.1', port=8888)
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "D:\Personal\Facu\Proyecto Investigación\tkgpio\venv\lib\site-packages\gpiozero\pins\pigpio.py", line 93, in __init__
    raise IOError(f'failed to connect to (host):{port}')
OSError: failed to connect to 127.0.0.1:8888

Process finished with exit code 1
```

6. Ejecuté el comando que indican ahí dentro de QEMU y sale el siguiente mensaje:

```

pi@raspberrypi:~$ sudo pigpiod -nlocalhost
2024-05-15 12:11:27 gpioHardwareRevision: unknown revision=0
2024-05-15 12:11:27 initCheckPermitted:
+-----+
|Sorry, this system does not appear to be a raspberry pi. |
|aborting.                                                 |
+-----+

Can't initialise pigpio library
pi@raspberrypi:~$

```

7. También intenté ejecutar pigpio manualmente siguiendo [4. Configuring Remote GPIO — gpiozero 2.0.1 Documentation](#) :

```

pi@raspberrypi:~$ sudo pigpiod
2024-05-15 12:14:26 gpioHardwareRevision: unknown revision=0
2024-05-15 12:14:26 initCheckPermitted:
+-----+
|Sorry, this system does not appear to be a raspberry pi. |
|aborting.                                                 |
+-----+

Can't initialise pigpio library

```

8. Por otro lado buscando en la web encontré un post interesante que habla sobre simular el GPIO de QEMU: [D. - Simulating Raspberri y PI GPIO interaction with QEMU](#)

9. Se probaron los comandos, pero pertenecen a Raspberry Pi anteriores y no es compatible con la versión 3 que estamos usando.

10. De todas formas, funcione o no por ahora el PIGPIO parece que debemos abrir el puerto 8888 de qemu y redireccionarlo afuera como hicimos en el docker que tenía UI agregando el hostfwd:



```

script_start_raspbian.sh
~/raspberrypi
1 #!/bin/bash
2
3 echo "Iniciando Raspbian..."
4
5 cd /root/raspberry/images
6 qemu-system-arm -kernel kernel-qemu-4.19.50-buster -drive "file=raspbian-
  buster.qcow,index=0,media=disk,format=qcow2" -append "root=/dev/sda2 panic=1 filesystem=ext4 rw"
  -cpu arm1176 -m 256 -M versatilepb -dtb versatile-pb-buster.dtb -no-reboot -net
  user,hostfwd=tcp::2222-:22 -net nic

```

10. Buscando encontré que debemos modificar la imagen [rpivm/run.sh at master · braghetto/rpivm](#) agregando el puerto 8888 con el comando hostfwd=tcp::8888-:8888

```

1  #!/bin/sh
2
3  qemu-img resize rpi-os.img 16G
4
5  qemu-system-aarch64 \
6  --machine raspi3b \
7  --cpu cortex-a53 \
8  -smp 4 --m 1024m \
9  --drive format=raw,file=rpi-os.img -netdev user,id=net0,hostfwd=tcp::5022-:22, hostfwd=tcp::8888-:8888 -device
10 --dtb bcm2710-rpi-3-b-plus.dtb \
11 --kernel kernel18.img \
12 --append "rw earlyprintk loglevel=8 console=ttyAMA0,115200 dwc_otg.lpm_enable=0 root=/dev/mmcblk0p2 rootdelay
13 --no-reboot \
14 --display none \
15 --serial mon:stdio -usb -device usb-mouse -device usb-kbd

```