2024/07/23 15:14 1/5 RPi.GPIO for ODROID



'ODROID-N2' on this page refers to the ODROID-N2 series (N2, N2+, N2L).

RPi.GPIO for ODROID



- Tested on C1+/C2/N2/XU4 + Shifter Shield.
- Thanks to @ifath from our forum for porting this useful library.
 - You can visit its Github repository.
- You can use WiringPi and WiringPi-Python alternatively.
 - Visit the wiki page: {EACH PRODUCT} application note gpio wiringpi.

RPi.GPIO is a library for using GPIO on a Raspberry Pi with Python. It implements basic GPIO functions and it's easy to use.

Install

Clone the repository.

target

```
$ sudo apt-get install git python-dev
$ git clone https://github.com/hardkernel/RPi.GPIO-Odroid
```

Install by typing this command.

target

```
$ cd RPi.GPIO-Odroid
$ sudo python setup.py build install
```

Installation does automatically.

Example Code

Let's do an example.

Before you proceed, make sure you've connected a jump cable from physical pin

• C1+/C2/N2/XU4 with Shifter Shield: #13 to #31.

• XU4 without Shifter Shield: #13 to #19.

The Github repository has this example file in RPi.GPIO-Odroid/test/simplerw.py. You can run it by enter this command.

target

```
$ sudo python simplerw.py
# results
To read output correctly, jumper pin 13 (bcm27) to pin 31 (bcm6)
Press Ctrl-C to exit
('*****Input pin state (Output HIGH) ', 1, '*****\n')
('*****Input pin state (Output LOW) ', 0, '*****\n')
('*****Input pin state (Output HIGH) ', 1, '****\n')
('*****Input pin state (Output LOW) ', 0, '****\n')
('*****Input pin state (Output HIGH) ', 1, '****\n')
```

```
import RPi.GPIO as GPIO
import time
LedPinW = 27 # pin13, bcm27
LedPinR = 6 # pin31, bcm6
def setup():
  GPIO.setmode(GPIO.BCM) # Numbers GPIOs by chip numbering scheme
  GPIO.setup(LedPinR, GPIO.IN, pull up down=GPIO.PUD UP) # Set LedPin's
mode is input
  GPIO.setup(LedPinW, GPIO.OUT) # Set LedPin's mode is output
  GPIO.output(LedPinW, GPIO.HIGH) # Set LedPin high(+3.3V) to turn on led
def blink():
  while True:
    GPIO.output(LedPinW, GPIO.HIGH) # led on
    time.sleep(2)
    pstate=GPIO.input(LedPinR)
    print("*****Input pin state (Output HIGH) ", pstate, "****\n")
    time.sleep(2)
    GPIO.output(LedPinW, GPIO.LOW) # led off
    time.sleep(2)
    pstate=GPIO.input(LedPinR)
    print("*****Input pin state (Output LOW) ", pstate, "****\n")
    time.sleep(2)
def destroy():
  GPIO.output(LedPinW, GPIO.LOW) # led off
  GPIO.setup(LedPinW, GPIO.IN, pull up down=GPIO.PUD DOWN) # Set LedPin's
mode is input
  GPIO.cleanup()
                                 # Release resource
if name == ' main ': # Program start from here
```

https://wiki.odroid.com/ Printed on 2024/07/23 15:14

2024/07/23 15:14 3/5 RPi.GPIO for ODROID

```
print('To read output correctly, jumper pin 13 (bcm27) to pin 31 (bcm6)')
print('Press Ctrl-C to exit')
setup()
try:
   blink()
except KeyboardInterrupt: # When 'Ctrl+C' is pressed, the child program
destroy() will be executed.
   destroy()
```

As you can see, this example code confirms operation of pin **output/input** and controlling **pull up/down mode**.

About BCM numbering

Actually BCM numbering for GPIO pin map is for Raspberry Pi products. So some people might not be familiar with this.

But fortunately, there's a helpful website offers different kind of pin maps of Raspberry Pi, and of course we can refer to.

Raspberry Pi			
Pinout 3v3 Power		2	5v Power
	3	4	
BCM 2 (WiringPi 8)			5v Power
BCM 3 (WiringPi 9)	5 0		Ground
BCM 4 (WiringPi 7)	7 0		BCM 14 (WiringPi 15)
Ground	9 • (_	
BCM 17 (WiringPi 0)	11 0		
BCM 27 (WiringPi 2)	13 🔾		
BCM 22 (WiringPi 3)	15 🔾	16	BCM 23 (WiringPi 4)
3v3 Power	17 🔘 🤇	18	BCM 24 (WiringPi 5)
BCM 10 (WiringPi 12)	19 💽	20	Ground
BCM 9 (WiringPi 13)	21 🔘	22	BCM 25 (WiringPi 6)
BCM 11 (WiringPi 14)	23	24	BCM 8 (WiringPi 10)
Ground	25 • (26	BCM 7 (WiringPi 11)
BCM 0 (WiringPi 30)	27 🔾	28	BCM 1 (WiringPi 31)
BCM 5 (WiringPi 21)	29	30	Ground
BCM 6 (WiringPi 22)	31	32	BCM 12 (WiringPi 26)
BCM 13 (WiringPi 23)	33	34	
BCM 19 (WiringPi 24)	35 🔘	36	BCM 16 (WiringPi 27)
BCM 26 (WiringPi 25)	37 🔾	38	BCM 20 (WiringPi 28)
Ground	39 • (40	
	C		

Screenshot captured on pinout.xyz

ODROID's 40 pin structure is equal to Raspberry Pi has, at least in vout and ground pins. Futher, this page provides with WiringPi numbering, and our each product provides WiringPi numbering pin map (as well as WiringPi library) so that you can write a source code with this.

References

- [1] https://forum.odroid.com/viewtopic.php?f=97&t=30577
- [2] https://github.com/jfath/RPi.GPIO-Odroid
- [3] https://github.com/joshua-yang/RPi.GPIO-Odroid
- [3] https://sourceforge.net/p/raspberry-gpio-python/wiki/Home/
- [4] https://pinout.xyz/pinout/wiringpi

https://wiki.odroid.com/ Printed on 2024/07/23 15:14

From:

https://wiki.odroid.com/ - ODROID Wiki

Permanent link:

https://wiki.odroid.com/odroid-xu4/application_note/gpio/rpi.gpio

Last update: 2023/11/16 18:24

