소켓프로그래밍

라즈베리파이로 배우는 소켓 통신 프로그래밍

동양미래대학교

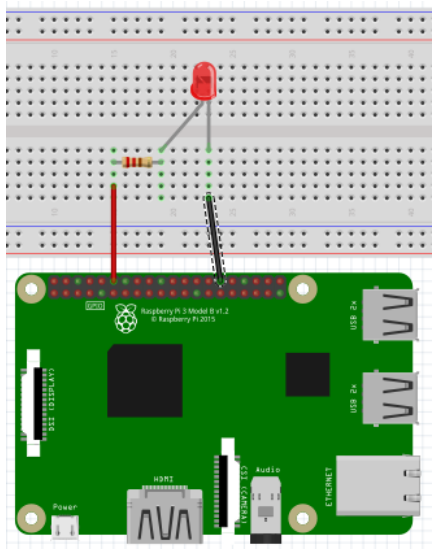
컴퓨터공학부 정석용

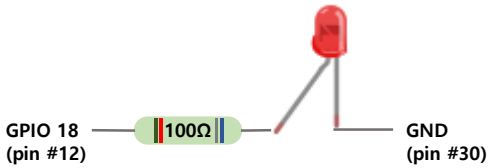
동양미래대학교

LED 원격 제어

(실습과제 1) gpio 명령을 이용한 LED 점멸

- 회로구성

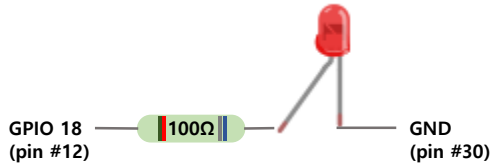




- LED의 Anode(+) 핀은 65 Ω 이상의 저항을 통해 라즈베리파이 GPIO 18(pin #12)에 연결
- LED의 cathod(-) 핀은 라즈베리파이 GND(pin #30)에 연결

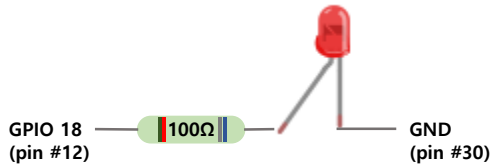
LED 반짝거리기

```
import RPi.GPIO as GPIO
import time

led_pin = 18

try :
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(led_pin, GPIO.OUT)

    while True:
            GPIO.output(led_pin, True)
            time.sleep(0.5)
            GPIO.output(led_pin, False)
            time.sleep(0.5)
finally:
      print('clean up')
      GPIO.cleanup()
```



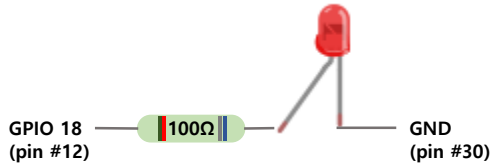
LED 원격제어 - client

```
import socket
import json

s = socket.socket()
host = '127.0.0.1'
port = 9000

s.connect((host, port))

data = {}
cmd = input('command : ')
led_no = input('led no : ')
act = input('on|off : ')

data['cmd'] = cmd
data['led_no'] = int(led_no)
data['act'] = act
body = json.dumps(data)
s.sendall(bytes(body, 'UTF-8'))
s.close()
```

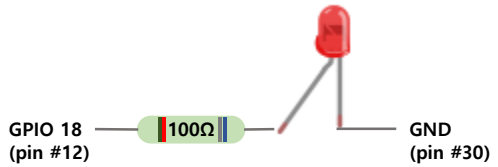


LED 원격제어 - server

```
import socket
import json

s = socket.socket()
s.setsocketopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)

host = '0.0.0.0'
port = 9000

s.bind((host, port))
s.listen(10)
while True:
    c. addr = s.accept()
    print('Got connection from', addr)
    data = c.recv(2048)
    msg = json.loads(data.decode())

    print('received data : ', msg)
    c.close()
s.close()
```



LED 원격제어 – client server



$ python server.py

$ python client.py

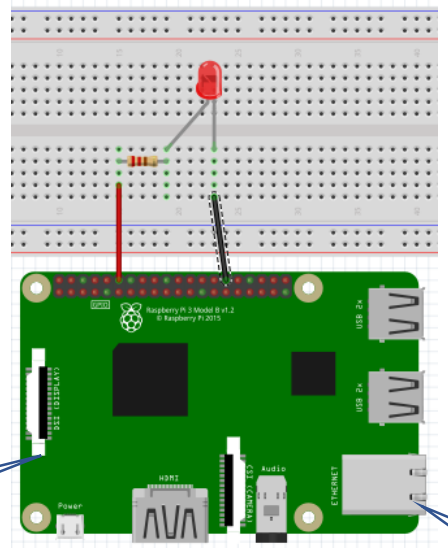
command : led

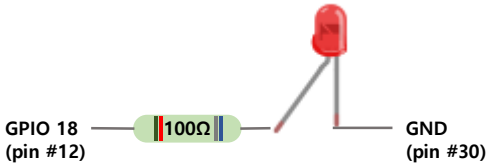
led_no : 10

on|off : on

실습과제 1 : LED 원격 제어

(실습과제 1) LED 원격 제어





클라이언트(client.py)

$ python client.py
cmd : led
led_no : 18
on|off : on

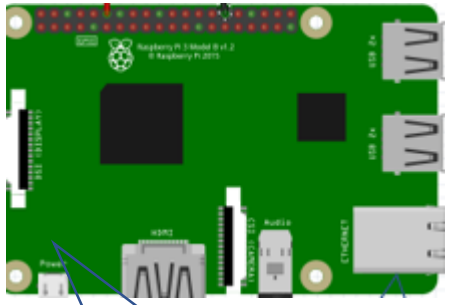
json

{ “cmd” : “led”, “led_no” : 18, “act” : ”on”}

서버(server.py)

GPIO.setmode(GPIO.BCM)
GPIO.setup(led_pin, GPIO.OUT)
GPIO.output(led_pin, True)

(실습과제 1) LED 원격 제어

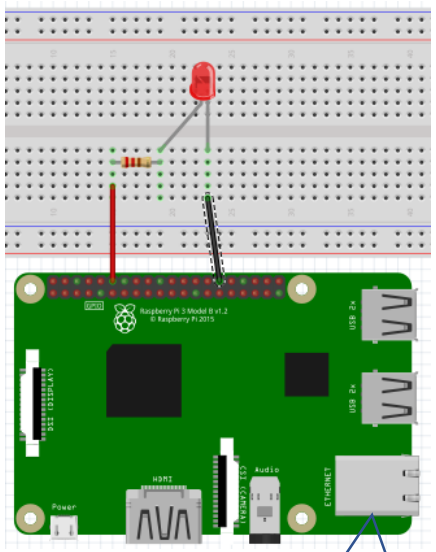


클라이언트(client.py)

$ python client.py
cmd : led
led_no : 18
on|off : on

json

{ “cmd” : “led”, “led_no” : 18, “act” : ”on”}

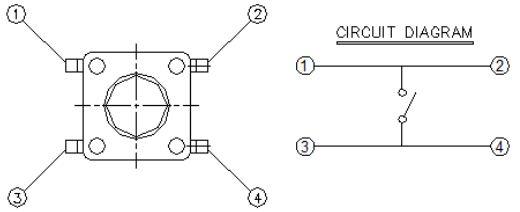


서버(server.py)

GPIO.setmode(GPIO.BCM)
GPIO.setup(led_pin, GPIO.OUT)
GPIO.output(led_pin, True)

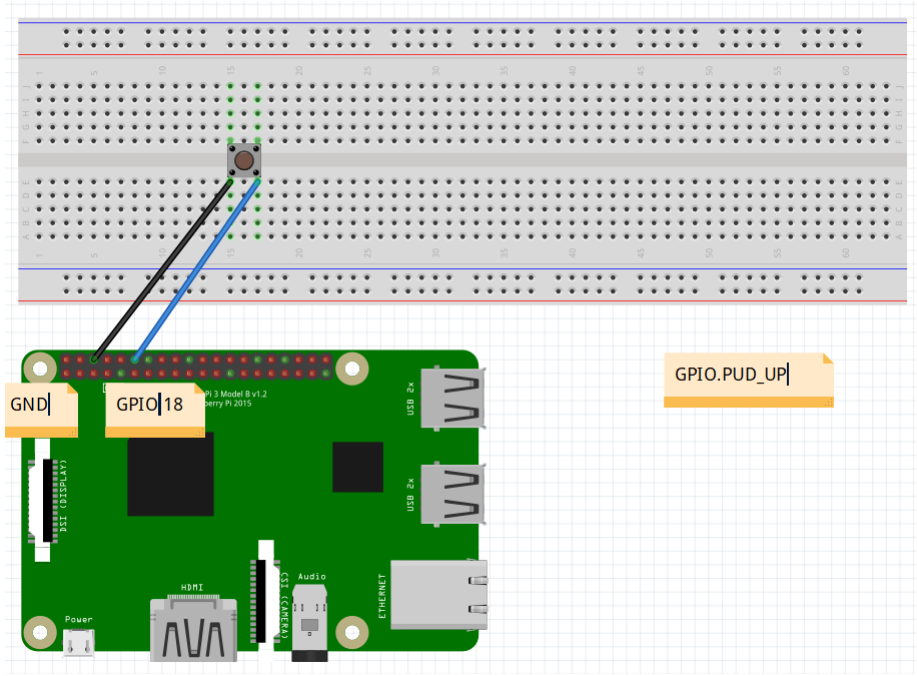
스위치 제어

Tack Switch 소자의 이해



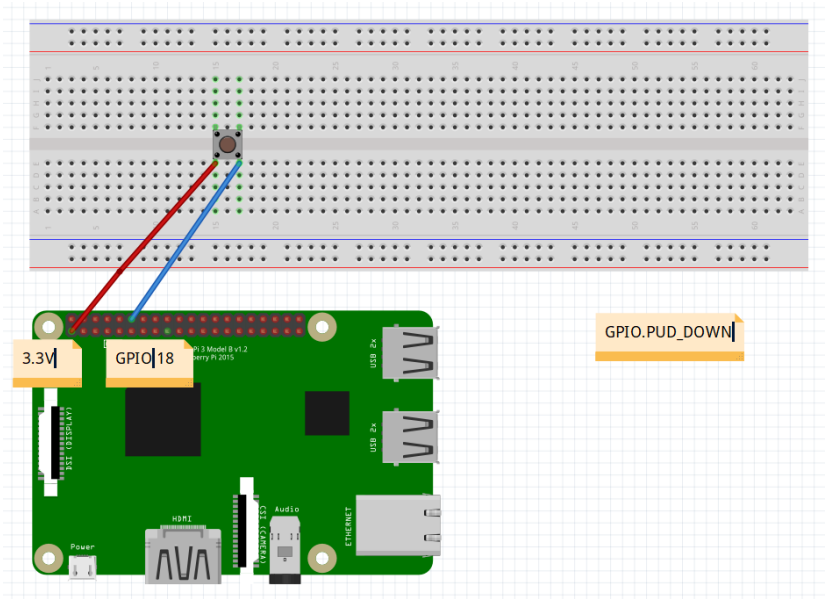
- 4개의 pin으로 구성되고, 2개씩 한 쌍으로 사용
- 한 쪽 pin에 3.3V 전원을 연결
- 반대 쪽 한 쌍의 pin을 각각 GPIO와 GND에 연결
- Switch 눌림에 따라 GPIO에 연결된 전원이 on/off

internal pull down/up : 내부적으로 풀업 저항을 제공



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

평상시 GPIO 18번 핀 값을 읽으면 – 1 값
스위치를 누르고 GPIO 18번 핀 값을 읽으면 – 0 값



GPIO.setup(sw_pin, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)

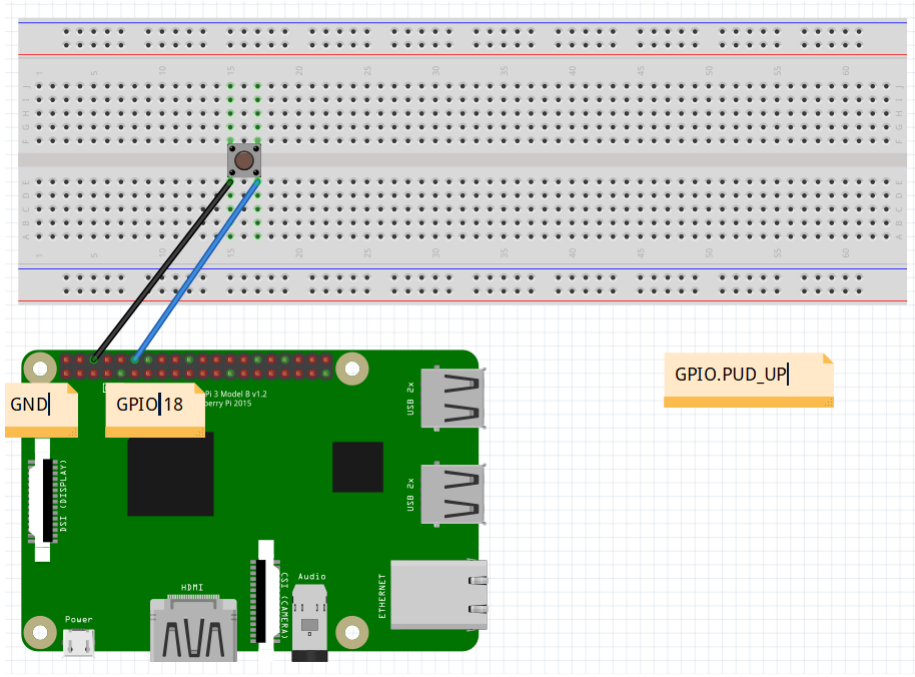
평상시 GPIO 18번 핀 값을 읽으면 – 0 값
스위치를 누르고 GPIO 18번 핀 값을 읽으면 – 1 값

(실습과제 2) pull-up 저항, 버튼 출력 확인

- 소스 코드

sw-up.py

```
import RPi.GPIO as GPIO

sw_pin = 18

try :
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(sw_pin, GPIO.IN, pull_up_down=GPIO.PUD_UP)
    while True:
            val = GPIO.input(sw_pin)
            print('switch : ', val)
finally:
    print('clean up')
    GPIO.cleanup()
```

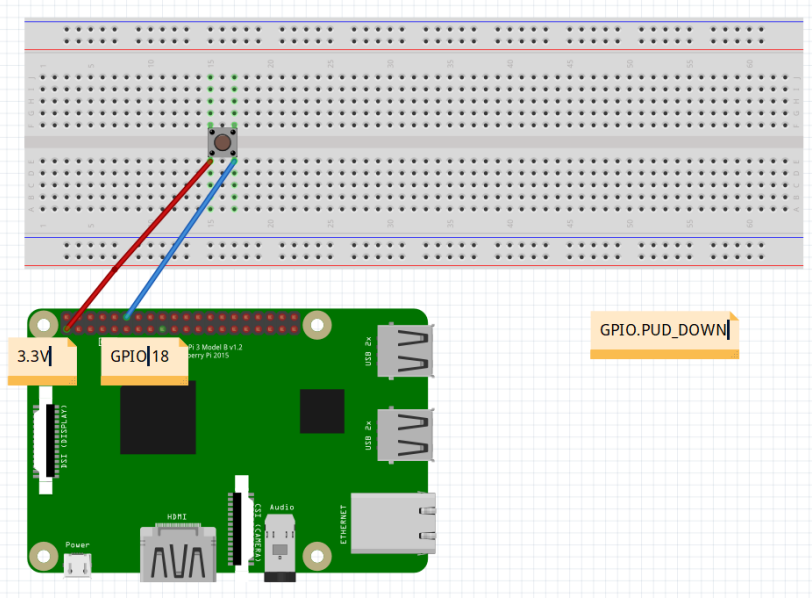


(실습과제 2) pull-up 저항, 버튼 출력 확인

- 소스 코드

sw-up.py

```
import RPi.GPIO as GPIO

sw_pin = 18

try :
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(sw_pin, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
    while True:
            val = GPIO.input(sw_pin)
            print('switch : ', val)
finally:
    print('clean up')
    GPIO.cleanup()
```



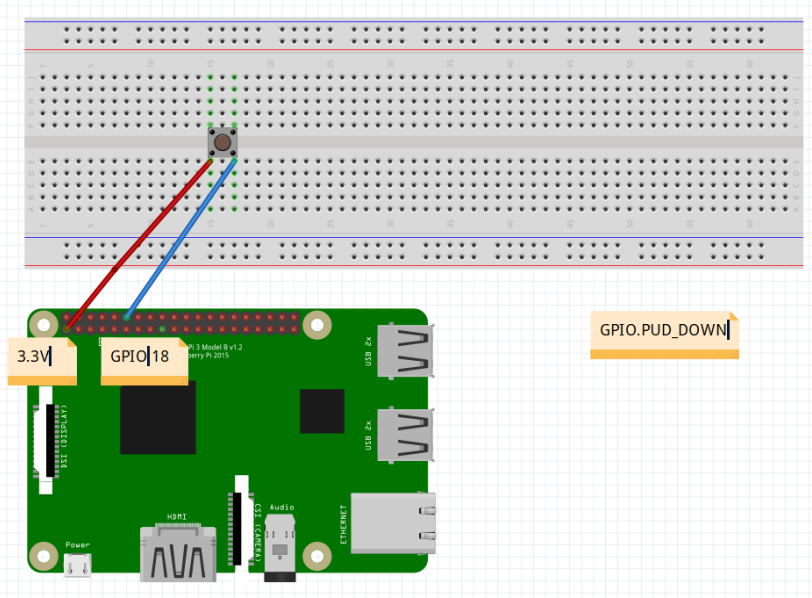
버튼 값이 계속 출력됨

(실습과제 3) pull-up 저항, 버튼 출력 확인 / 상태가 바뀔 때만 출력

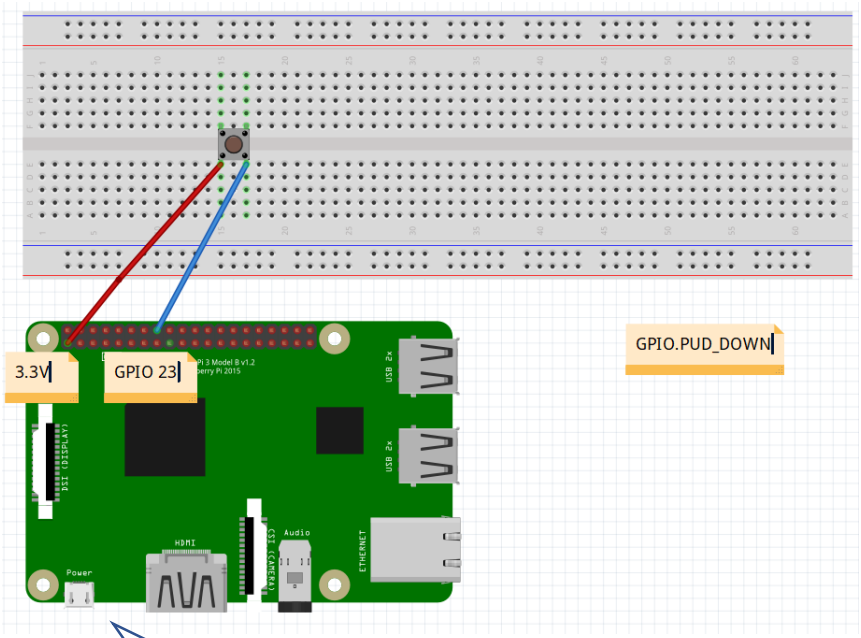
- 소스 코드

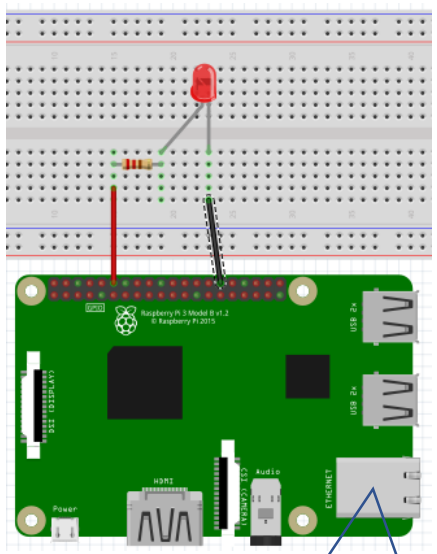
sw-up.py

```
import RPi.GPIO as GPIO

sw_pin = 18
before  = 0
try :
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(sw_pin, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
    while True:
            after = GPIO.input(sw_pin)
            while before == after :
                    after = GPIO.input(sw_pin)
            before = after
            print('switch : ', after)
finally:
    print('clean up')
    GPIO.cleanup()
```



(실습과제 4) switch를 이용한 led 원격 제어





서버(server.py)

클라이언트(client.py)

스위치를 누르면

json

{ “cmd” : “led”, “led_no” : 18, “act” : ”on”}

GPIO.setmode(GPIO.BCM)

GPIO.setup(led_pin, GPIO.OUT)

GPIO.output(led_pin, True)