소켓프로그래밍

라즈베리파이로 배우는 소켓 통신 프로그래밍

동양미래대학교

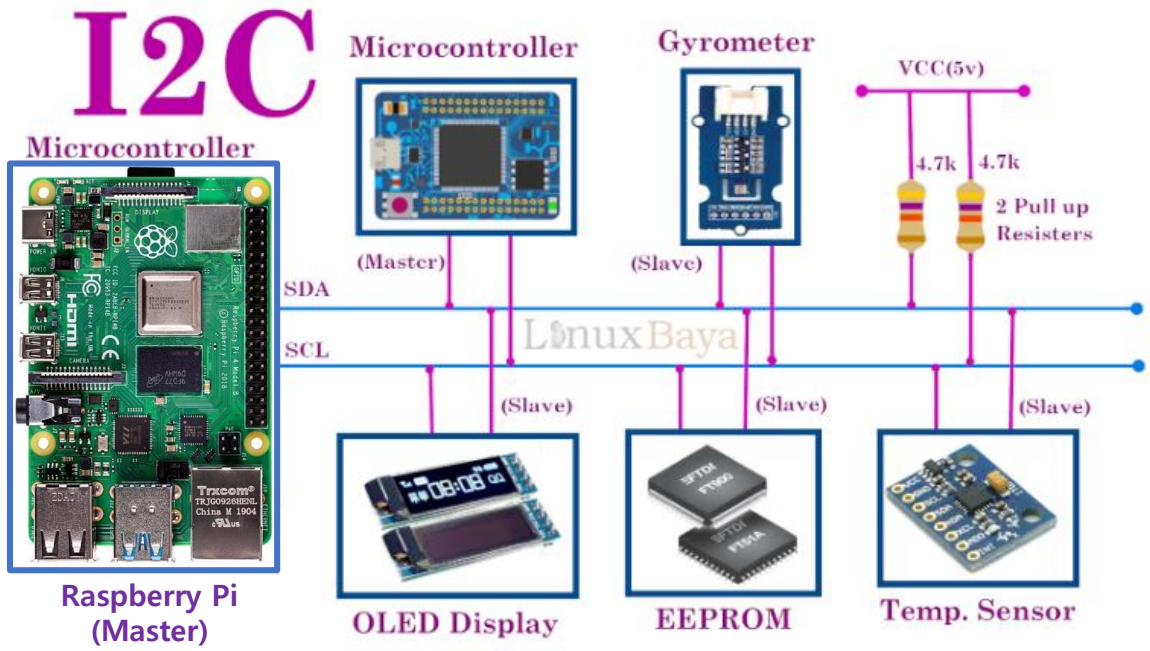
컴퓨터공학부 정석용

동양미래대학교

I^2C (Inter Integrated Circuit) 통신

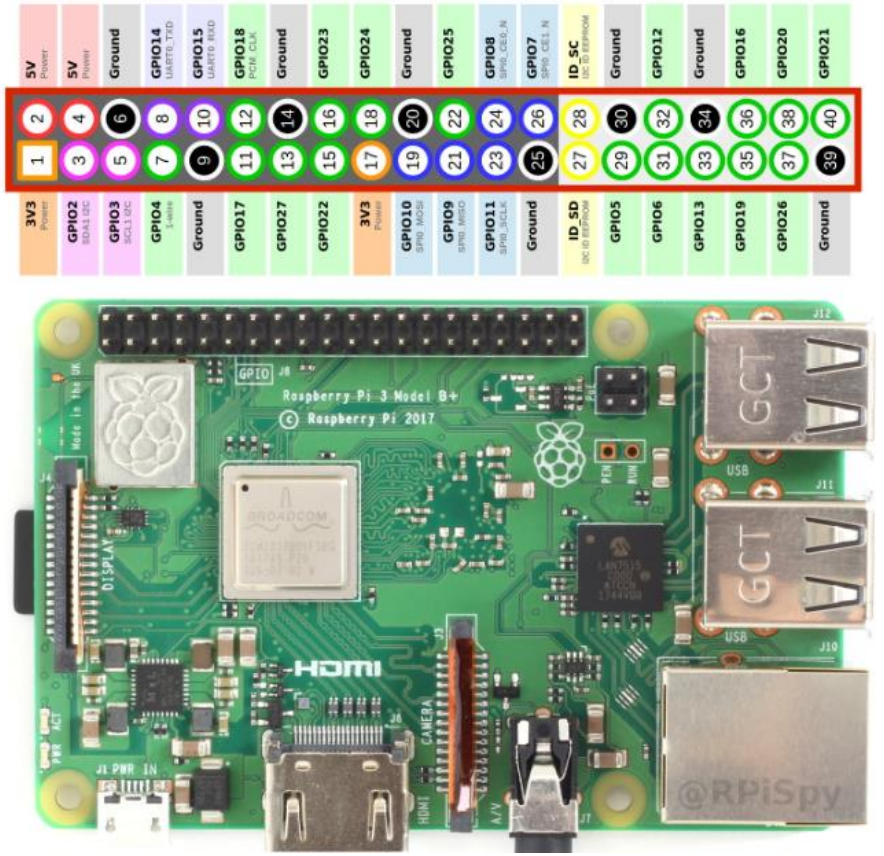
I²C (Inter Integrated Circuit) 통신

- MCU(Micro Control Unit)와 주변 장치와의 통신을 위한 프로토콜
- 2개 회선 송수신 타이밍 동기화를 위한 SCL(Serial Clock)과 데이터를 주고 받는 SDA(Serial Data)로 통신
- 장치 추가에 따른 회선 추가가 없고, 하나의 버스에 여러 장치(최대 128개)를 연결 가능
- 각 장치는 고유의 주소(7 비트)를 가짐



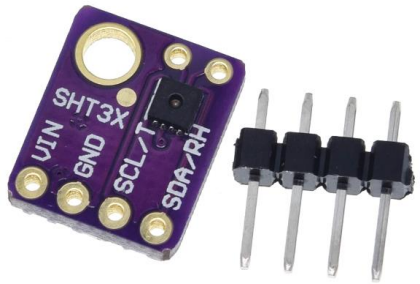
Raspberry Pi I²C (Inter Integrated Circuit) 통신

Raspberry Pi
Pin #3(SDA1)
Pin #5(SCL1)



Raspberry Pi에서 I2C 통신을 사용하려면 환경 설정에서 I2C를 활성화 해주어야 함

온습도 센서 STH31의 이해



- VIN : Pin # 1 (3.3V)
- GND : Pin # 6 (GND)
- SCL : Pin #3 (SDA1, GPIO2)
- SDA : Pin #5 (SCL1, GPIO3)

- 온도와 습도를 측정할 수 있음

I²C (Inter Integrated Circuit) 통신을 위한 Raspberry Pi 환경 설정

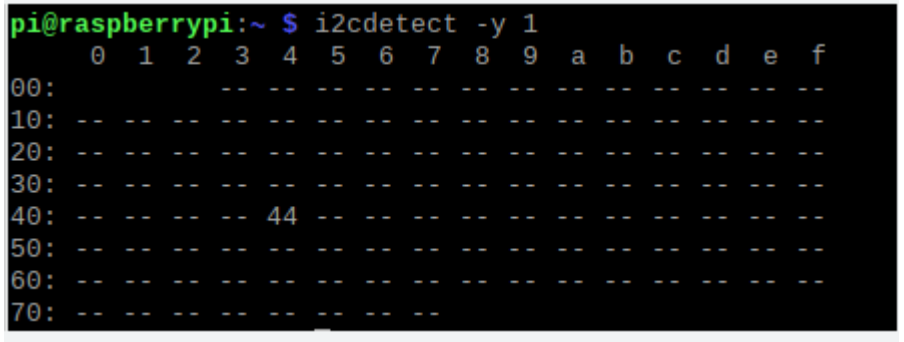
(1) Interface Option에서 I2C 활성화

$ sudo raspi-config

(2) I2C 동작 테스트 : 연결된 온습도 센서(STH31) 연결 상태 및 주소 확인

정상 연결되면 온습도 센서(STH31)의 I2C 주소가 0x44임을 알 수 있음

$ i2cdetect –y 1



(3) Python I2C 라이브러리 설치

$ sudo pip install smbus2

온습도 센서 STH31 Datasheet

Datasheet SHT3x-DIS

Please note that the I2C address is represented through the 7 MSBs of the I2C read or write header. The LSB switches between read or write header. The wiring for the default address is shown in Table 7 and Figure 11. The ADDR pin must not be left floating. Please note that only the 7 MSBs of the I2C Read/Write header constitute the I2C Address.

SHT3x-DIS	I2C Address in Hex. representation	Condition
I2C address A	0x44 (default)	ADDR (pin 2) connected to VSS
I2C address B	0x45	ADDR (pin 2) connected to VDD

Table 7 I2C device addresses.

4.5 Measurement Commands for Periodic Data Acquisition Mode

In this mode one issued measurement command yields *a stream of data pairs*. Each data pair consists of one 16 bit temperature and one 16 bit humidity value (in this order).

Datasheet SHT3x-DIS

Condition		Hex. code	
Repeatability	mps	MSB	LSB
High	0.5	0x20	32
Medium			24
Low			2F
High	1	0x21	30
Medium			26
Low			2D
High	2	0x22	36
Medium			20
Low			2B
High	4	0x23	34
Medium			22
Low			29
High	10	0x27	37
Medium			21
Low			2A

4.13 Conversion of Signal Output

Measurement data is always transferred as 16-bit values (unsigned integer). These values are already linearized and compensated for temperature and supply voltage effects. Converting those raw values into a physical scale can be achieved using the following formulas.

Relative humidity conversion formula (result in %RH):

$$RH = 100 \cdot \frac{S_{RH}}{2^{16} - 1}$$

Temperature conversion formula (result in °C & °F):

$$T[°C] = -45 + 175 \cdot \frac{S_T}{2^{16} - 1}$$

$$T[°F] = -49 + 315 \cdot \frac{S_T}{2^{16} - 1}$$

S_{RH} and S_T denote the raw sensor output for humidity and temperature, respectively. The formulas work only correctly when S_{RH} and S_T are used in decimal representation.

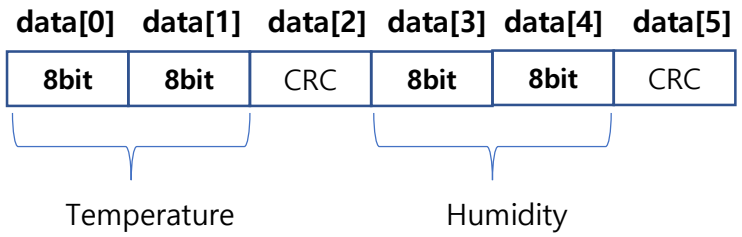
sht31.py.py

```
import smbus2
import time

bus = smbus2.SMBus(1)
bus.write_i2c_block_data(0x44, 0x22, [0x36])                    # SHT31(0x44)에게 반복적으로 온습도 측정 요청
while True :
        time.sleep(0.5)

        data = bus.read_i2c_block_data(0x44, 0x00, 6)           # SHT31(0x44)에서 0x00 주소 부터 6바이트 읽어옴
        temp_r  = data[0] * 256 + data[1]                       # 온도 : data[0] data[1]
        temp = round(-45 + (175 * temp_r /65535.0), 1)
        humidity = round(100 * (data[3] * 256 + data[4]) / 65535.0, 1)   # 습도 : data[3] data[4]

        print("Temperature : %.2f C " % temp)
        print("Humidity : %.2f RH" % humidity)
```



SSD1306 OLED 의 이해

- VCC : Pin # 1 (3.3V)
- GND : Pin # 6 (GND)
- SCL : Pin #3 (SDA1, GPIO2)
- SDA : Pin #5 (SCL1, GPIO3)

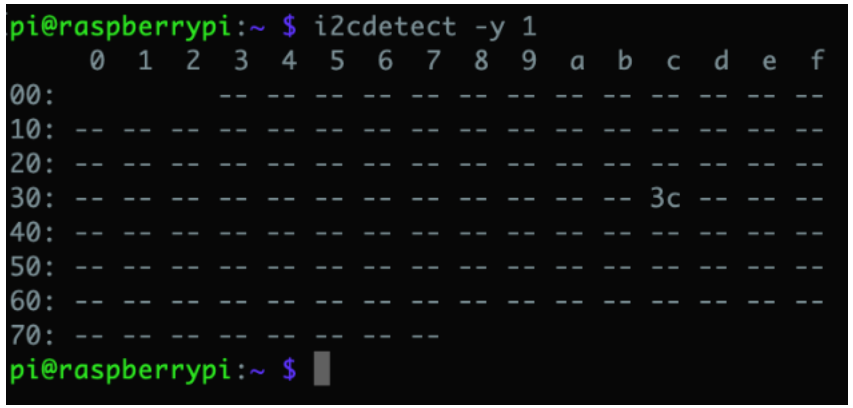
- Text나 이미지를 출력할 수 있음

SSD1306 OLED 설정 확인

(1) I2C 동작 테스트 : OLED(SSD1306) 연결 상태 및 주소 확인

정상 연결되면 OLED(SSD1306)의 I2C 주소가 0x3C임을 알 수 있음

$ i2cdetect –y 1



(2) Python에서 OLED를 제어하기 위한 모듈 설치

sudo pip3 install adafruit-circuitpython-ssd1306

(3) OLED 출력에 사용할 font 다운로드

```
wget https://github.com/adafruit/Adafruit_CircuitPython_framebuf/raw/main/examples/font5x8.bin
```

SSD1306 OLED에 “Hello” 출력

oled.py

```
import board
from busio import I2C
import adafruit_ssd1306

i2c = I2C(board.SCL, board.SDA)

display = adafruit_ssd1306.SSD1306_I2C(128, 64, i2c, addr=0x3C)

display.fill(255)     #display.fill(0)
display.show()

display.text('Hello', 5, 40, 1, font_name='font5x8.bin')
display.show()
```

display.rect(5, 15, 20, 10, 1)

display.circle(50, 15, 10, 1)

display.hline(5, 10, 20, 1)

SSD1306 OLED에 IP 주소 출력

oled-ip.py

```
import board
from busio import I2C
import adafruit_ssd1306
import socket

def get_ip_addr() :
        s = socket.socket(socket.AF_INET, socket.SOCK_DRAM)
        s.connect((“8.8.8.8”, 80))
        return s.getsockname()[80]

i2c = I2C(board.SCL, board.SDA)
display = adafruit_ssd1306.SSD1306_I2C(128, 64, i2c, addr=0x3C)
ip = get_ip_addr()
display.fill(255)    #display.fill(0)
display.show()

display.text(ip, 5, 40, 1, font_name=‘font5x8.bin’)
display.show()
```

실습과제 1 : 온습도 센서 값을 읽고, OLED에 표시하시오.

실습과제 2 : 친구와 짝을 이루어, 온습도 센서 값을 읽어 서버로 보내면
서버에 연결된 OLED에 값이 표시 되도록 하시오.

딕셔너리로 데이터 전송 body = { 'temp' : 12.7 , 'humi' : 27.8 }

실습과제 1

```
import smbus2
import time
import board
from busio import I2C
import adafruit_ssd1306

i2c = I2C(board.SCL, board.SDA)

display = adafruit_ssd1306.SSD1306_I2C(128, 64, i2c, addr=0x3C)

bus = smbus2.SMBus(1)
bus.write_i2c_block_data(0x44, 0x22, [0x36])
```

```
while True :
        display.fill(0)
        display.show()

        time.sleep(0.5)

        data = bus.read_i2c_block_data(0x44, 0x00, 6)

        temp_r  = data[0] * 256 + data[1]

        temp = round(-45 + (175 * temp_r /65535.0), 1)
        humidity = round(100 * (data[3] * 256 + data[4]) / 65535.0, 1)

        display.text(str(temp), 5, 20, 1, font_name='font5x8.bin')
        display.text(str(humidity), 5, 40, 1, font_name='font5x8.bin')
        display.show()

        print("Temperature : %.2f C " % temp)
        print("Humidity : %.2f RH" % humidity)
```

실습과제 2-1 온습도 측정 client : sht31_client.py

```
import smbus2
import time

import socket
import json

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

s.connect((host, port))

bus = smbus2.SMBus(1)
bus.write_i2c_block_data(0x44, 0x22, [0x36])
```

```
while True :
        time.sleep(0.5)

        data = bus.read_i2c_block_data(0x44, 0x00, 6)
        temp_r  = data[0] * 256 + data[1]
        temp = round(-45 + (175 * temp_r /65535.0), 1)
        humidity = round(100 * (data[3] * 256 + data[4]) / 65535.0, 1)

        print("Temperature : %.2f C " % temp)
        print("Humidity : %.2f RH" % humidity)

        data = {}
        data['temp'] = temp
        data['humi'] = humidity
        body = json.dumps(data)
        s.sendall(bytes(body, 'UTF-8'))
s.close()
```

실습과제 2-2 oled 온습도 출력 server : oled_server.py

```
import time
import board
from busio import I2C
import adafruit_ssd1306
import socket
import json

s = socket.socket()
s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
host = '0.0.0.0'
port = 9000
s.bind((host, port))
s.listen(10)

i2c = I2C(board.SCL, board.SDA)
display = adafruit_ssd1306.SSD1306_I2C(128, 64, i2c, addr=0x3C)
```

```
while True :
    c. addr = s.accept()
    print('Got connection from', addr)

    while True :
            data = c.recv(2048)
            msg = json.loads(data.decode())
            print('received data : ', msg)

            display.fill(0)
            display.show()

            display.text(str(msg['temp']), 5, 20, 1, font_name='font5x8.bin')
            display.text(str(msg['humi']), 5, 40, 1, font_name='font5x8.bin')
            display.show()
```

잠깐만 서버는 다른 요청을 받을 수 없음 – 쓰레드로 도입 필요