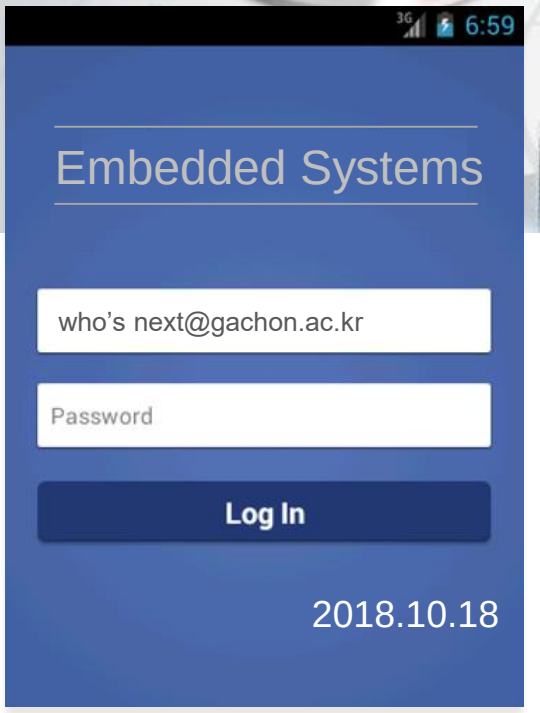


임베디드 시스템

컴퓨터공학과 이병문

강의일정(전반부)

01 강의소개, 강의일정소개, 평가소개

02 사물인터넷, 라즈베리파이3 설치/구축

03 임베디드 액츄레이터/센서 제어 1

04 임베디드 액츄레이터/센서 제어 2

05 임베디드 액츄레이터/센서 제어 3

06 임베디드 액츄레이터/센서 제어 4

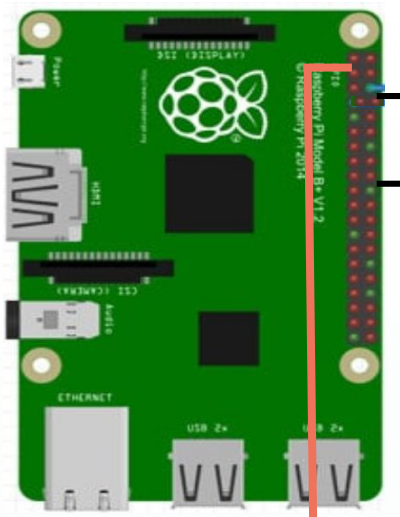
07 임베디드 액츄레이터/센서 제어 5

08 중간고사

- **아날로그 센서제어**
- SPI **전송방식**
- ADC(MCP3208)

■ 라즈베리파이와 다른 기기간의 연동

☑ 3가지 전송방식

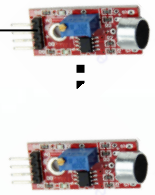


① Serial 통신 (serial to USB)

② SPI 통신

ADC

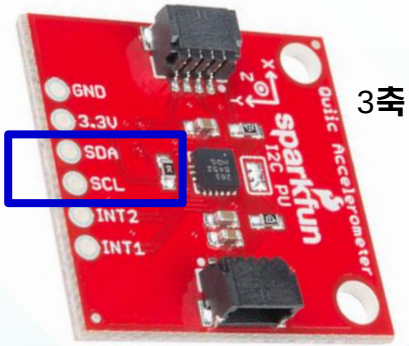
아날로그센서들…



9축 가속도센서

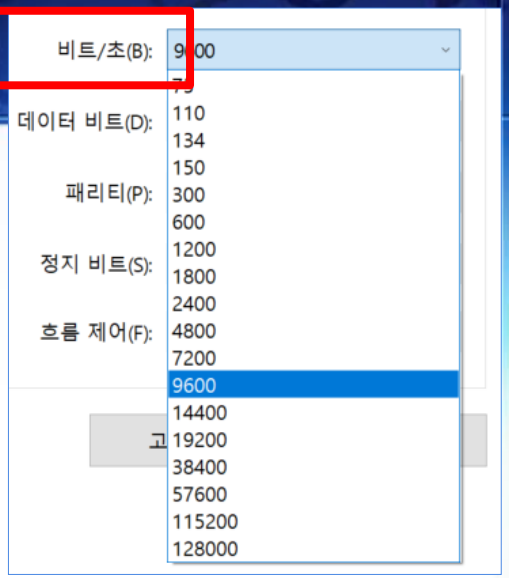
③ I²C 통신

미세먼지센서



3축 가속도센서

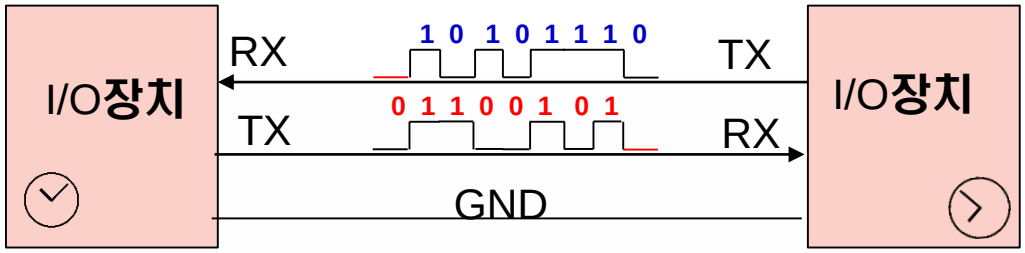
임베디드 액츄레이터/센서 제어



■ Asynchronous Serial Communication (비동기식 전송방식)

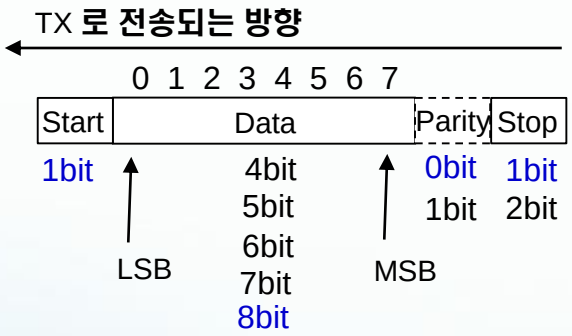
☑ 두개의 디바이스간의 시리얼(직렬)전송

☑ 각각 TX, RX 이용하여 데이터 송신/수신함 (**full duplex** 방식)



TX **송신전용 신호선**
RX **수신전용 신호선**
GND **접지**

☑ 전송데이터 포맷

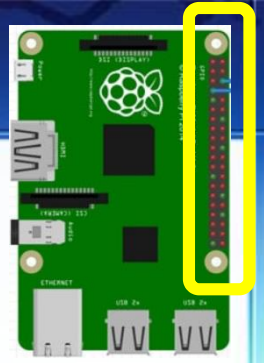


비동기 설정값

전송속도 ?
Data bit ?
Parity bit ?
Stop bit ?

```
$ gpio readall
all
+---Pi 3---+---+------+---------+--
| Physical | V | Mode | Name    | w
+----++----+---+------+---------+--
|   1 ||  2 |   |      | 5v      |
|   3 ||  4 |   |      | 5v      |
|   5 |  6 |   |      | 0v      |
|   7 |  8 | 0 | IN   | TxD     | 1
|   9 | 10 | 1 | IN   | RxD     | 1
|  11 || 12 | 0 | OUT  | GPIO. 1 | 1
```

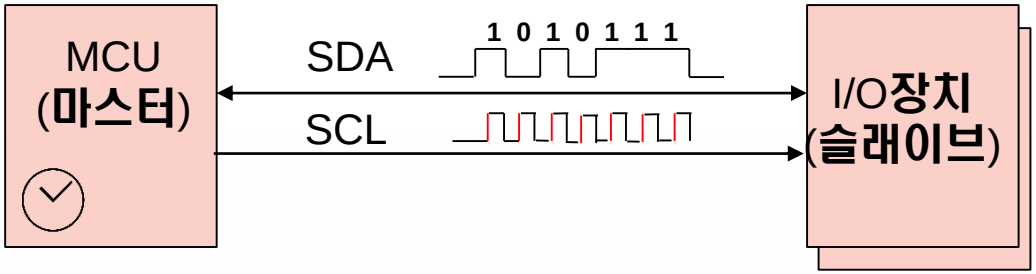
임베디드 액츄레이터/센서 제어



■ I²C, I2C 버스방식 (Inter-Integrated Circuit) … 동기식 전송방식

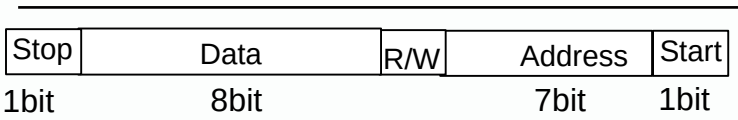
☑ MCU와 I/O 디바이스간의 양방향 시리얼(직렬)전송 버스

☑ 2개의 버스(SDA, SCL)를 이용하여 데이터 전송함 (**half duplex** 방식)



Serial DAta line
데이터 비트의 신호선
Serial Clock Line
동기용 클럭 신호선

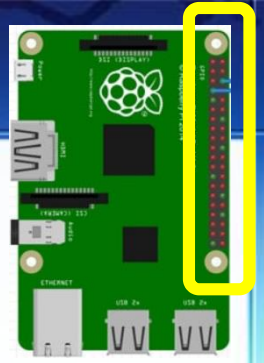
☑ SDA에서의 전송데이터 포맷



GPIO Pin layout (**실제시스템**, **$ gpi**

```
pi@raspberrypi:~/mysrc $ gpio readall
 +-----+-----+---------+------+---+---Pi
 | BCM | wPi |   Name  | Mode | V | Physi
 +-----+-----+---------+------+---+----+
 |     |     |    3.3v |      |   |  1 |
 |   2 |   8 |   SDA.1 |   IN | 1 |  3 |
 |   3 |   9 |   SCL.1 |   IN | 1 |  5 |
 |   4 |   7 | GPIO. 7 |   IN | 1 |  7 |
 |     |     |      0v |      |   |  9 |
 |   0 |  30 |   SDA.0 |   IN | 1 | 27 | 28 | 1 | IN | SCL.0 | 31 | 1 |
 |   5 |  21 | GPIO.21 |   IN | 1 | 29 | 30 |   |    |   0v  |    |   |
 |   6 |  22 | GPIO.22 |   IN | 1 | 31 | 32 | 0 | IN | GPIO.26 | 26 | 12 |
```

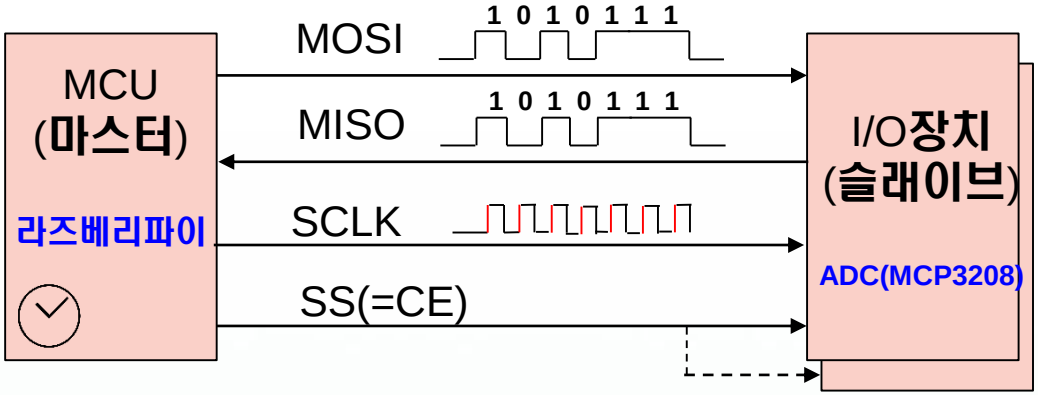
임베디드 액츄레이터/센서 제어



■ SPI (Serial Peripheral Interface) 버스방식 … 동기식 전송방식

☑ MCU와 I/O 디바이스간의 양방향 시리얼(직렬)전송 버스

☑ 4개의 버스(SCLK, MOSI, MISO, SS)를 이용하여 데이터 전송함 (full duplex 방식)



Master Output Slave Input
Master Input Slave Output
Serial CLocK
Slave Select (=Chip Enable)

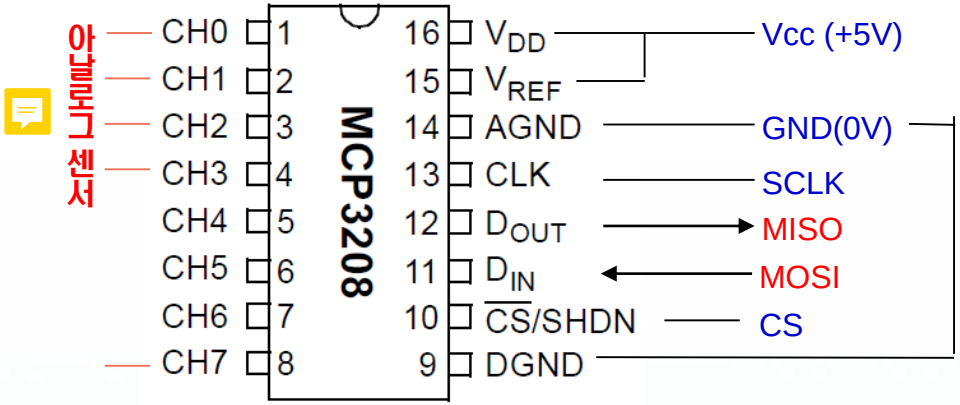
22	3	GPIO. 3	IN	1	15	16	0	IN	GPIO. 4	4	23
		3.3v			17	18	0	IN	GPIO. 5	5	24
10	12	MOSI	IN	0	19	20			0v		
9	13	MISO	IN	0	21	22	0	IN	GPIO. 6	6	25
11	14	SCLK	IN	0	23	24	1	IN	CE0	10	8
		0v			25	26	1	IN	CE1	11	7
0	30	SDA.0	IN	1	27	28	1	IN	SCL.0	31	1
5	21	GPIO.21	IN	1	29	30			0v		
6	22	GPIO.22	IN	1	31	32	0	IN	GPIO.26	26	12

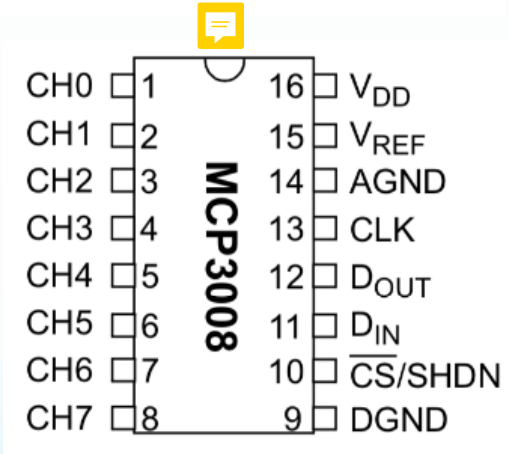
임베디드 액츄레이터/센서 제어

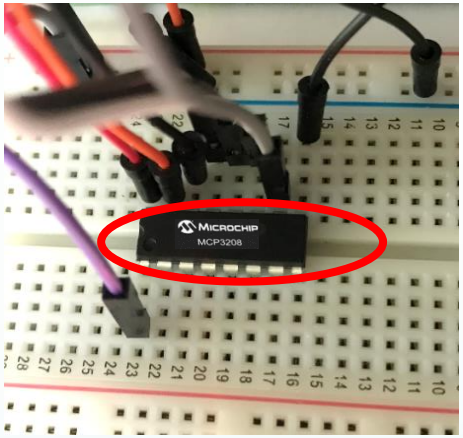
■ 하드웨어 구성

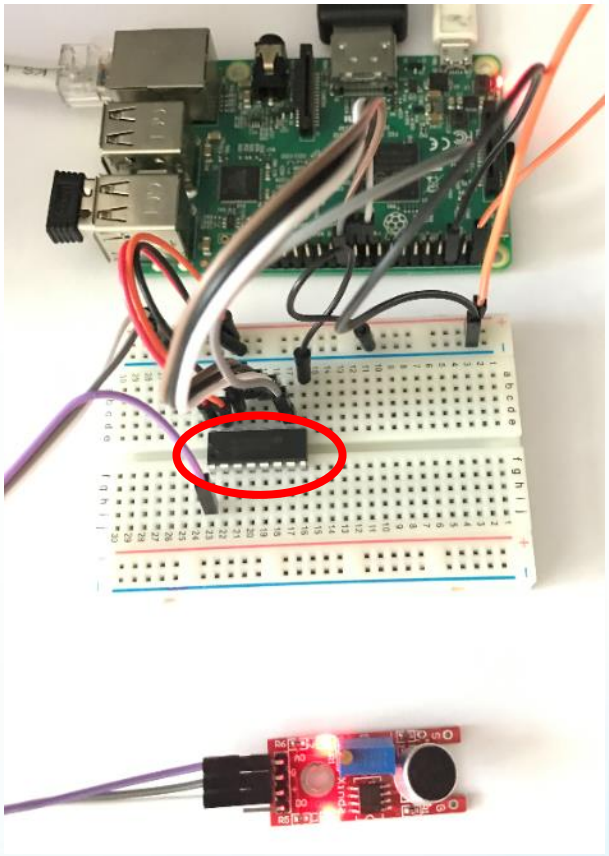
☑ **ADC 칩과 구성**

ADC (**A**nalog to **D**igical **C**onverter)



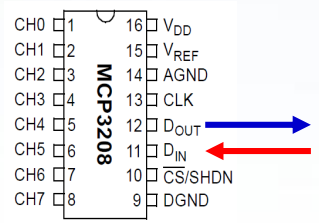




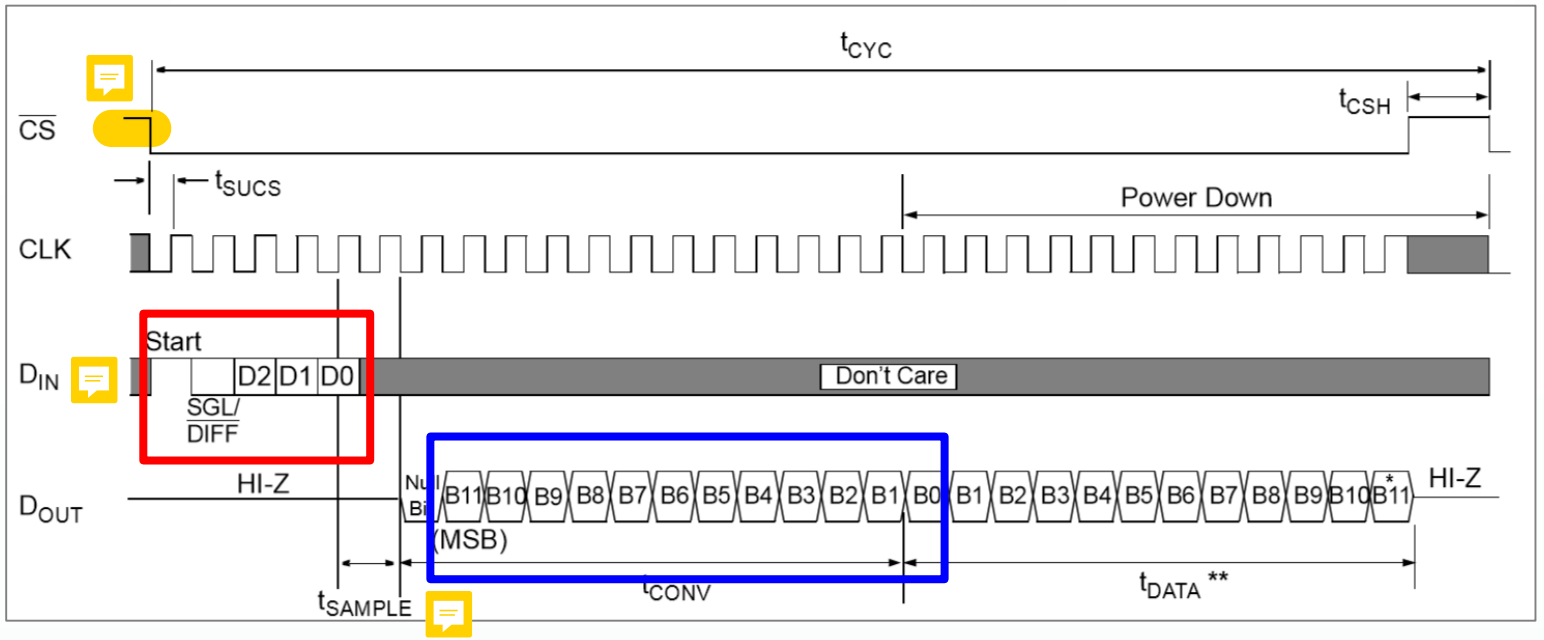


임베디드 액츄레이터/센서 제어

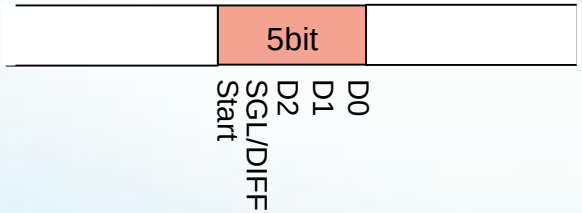
■ SPI통신 프로토콜



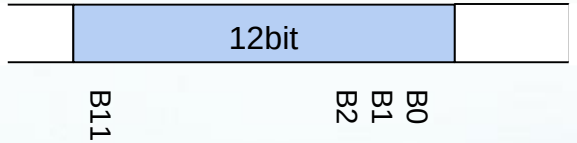
☑ 아날로그 사운드(소리) 센서데이터 **SPI**전송과정



RaspberryPi → MCP3208 ADC



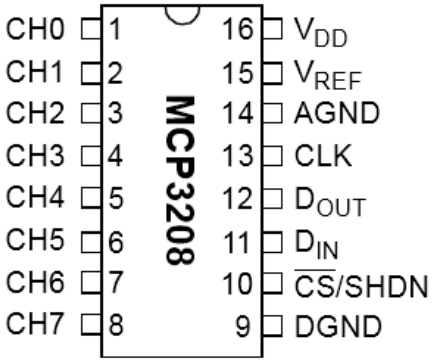
RaspberryPi ← MCP3208 ADC

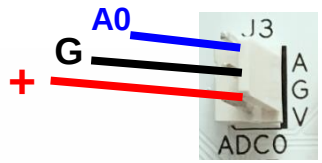


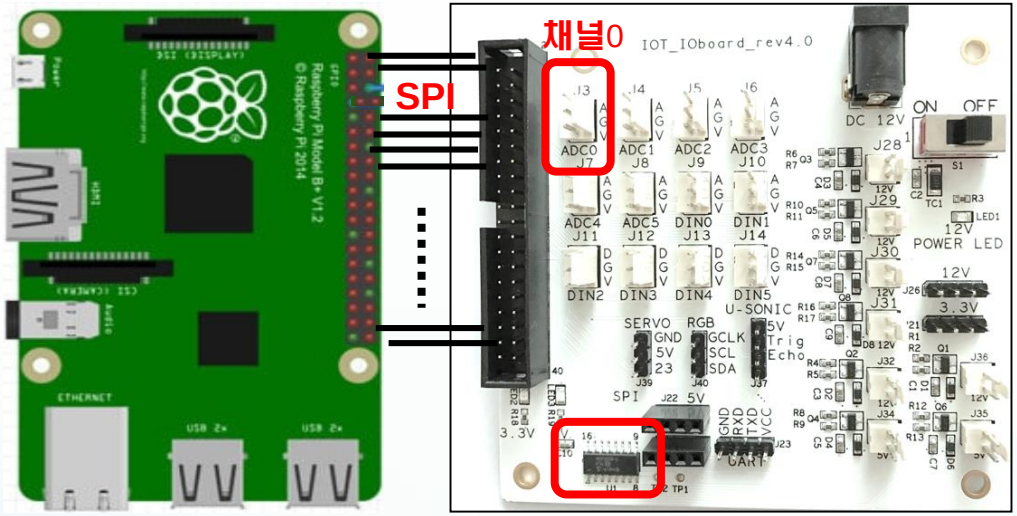
■ 아날로그 센서모듈 제어

☑ 아날로그 사운드(소리)센서모듈

ADC (Analog to Digical Converter)







ADC (MCP3208)

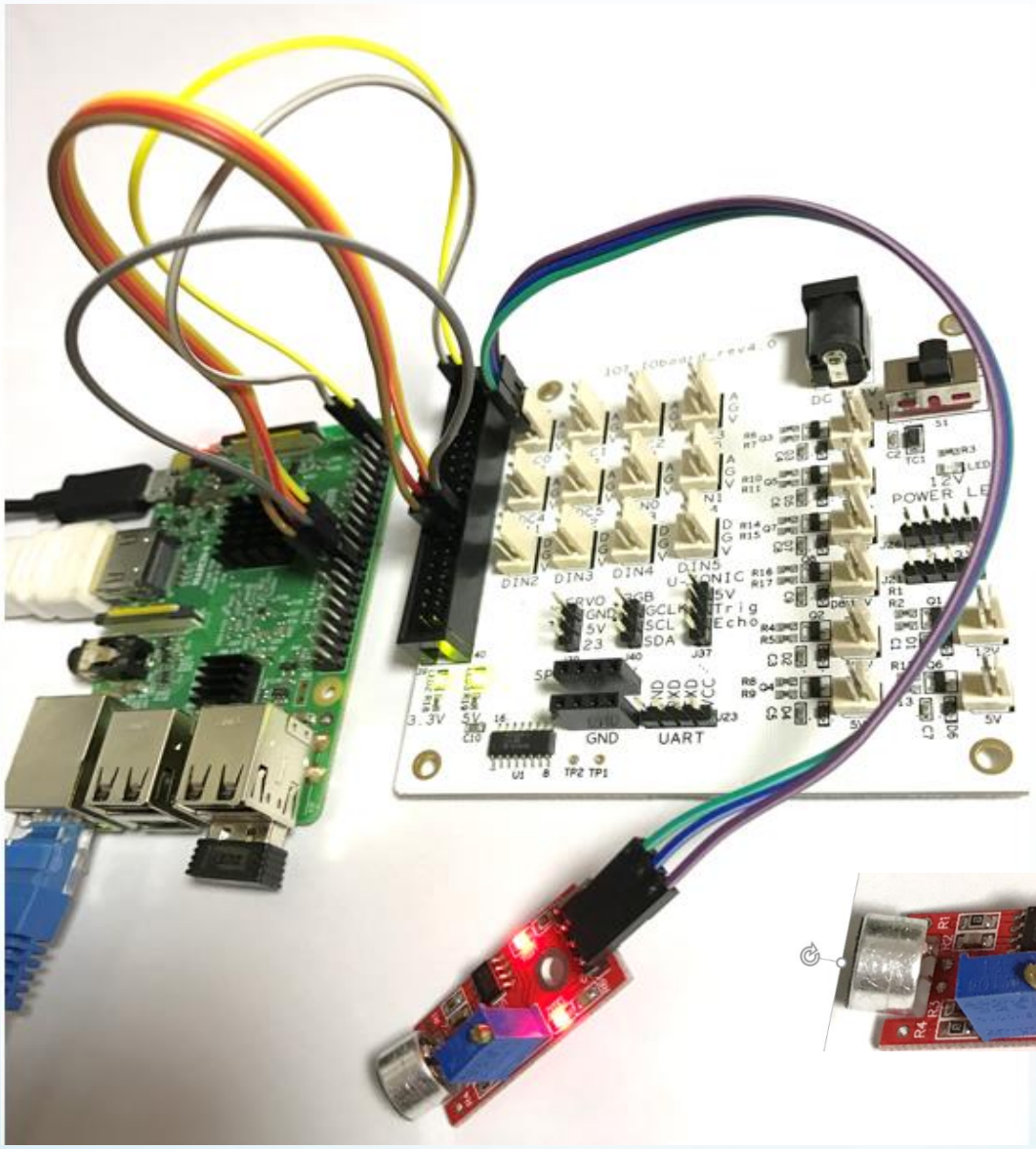
CE0(Chip Enable)

Vcc G

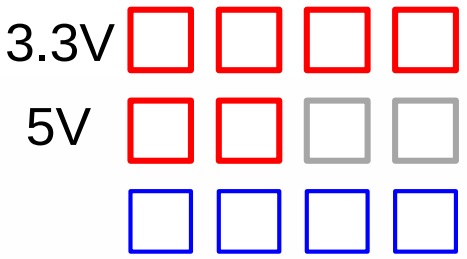
(디지털 경우만 사용함)

임베디드 액츄레이터/센서 제어

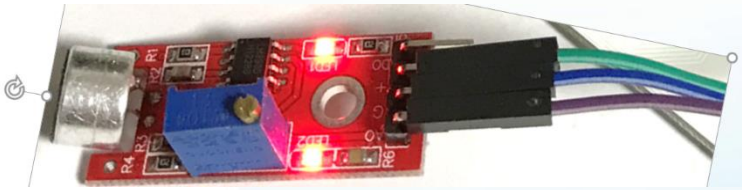
■ 하드웨어 구성



상단의 4개채널은 3.3V전원
하단의 4개채널은 5V전원

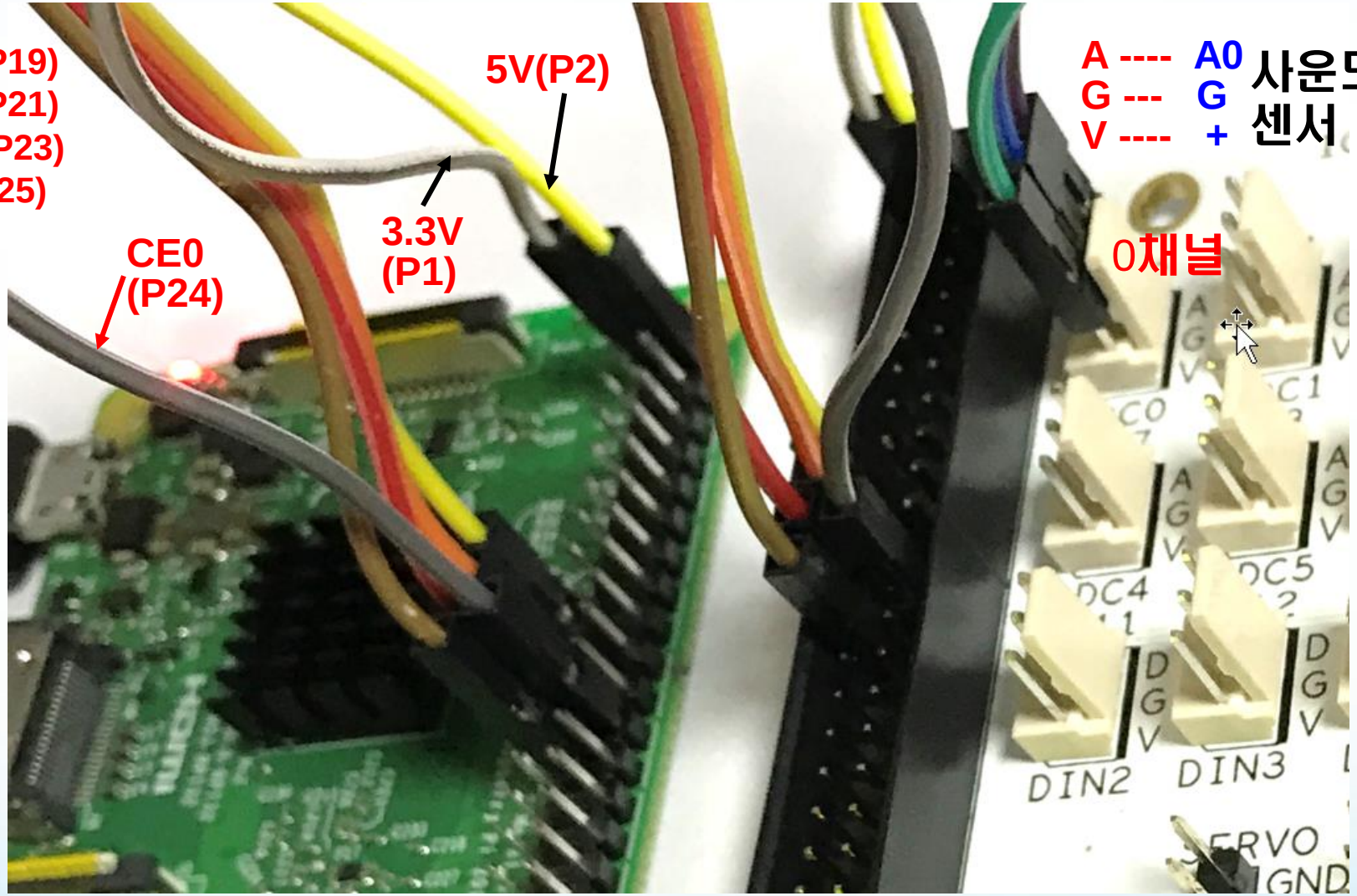


MCP3208은 8채널을 지원하나
I/O보드에서는 5채널만 연결되어 있음



임베디드 액츄레이터/센서 제어

■ 하드웨어 구성 (점퍼케이블 – 핀연결)



■ 프로그래밍 준비

☑ 라즈베리파이3의 **SPI** 기능 활성화 **(최초 1회만 설정함)**

```
$ sudo   raspi-config
```

Interface Option -> SPI enable **로 반드시 셋팅** (SPI **드라이버를 커널에 로딩**)

```
$ sudo  reboot
```

리부팅하여야 함!!

☑ 외부모듈 **(mcp-adc)** 설치

```
$ cd  mypgm
$ npm  install  mcp-adc
……
…… 설치완료!
$
```

mcp-adc **외부모듈 설치**!

■ Example code

예제1 (analog_sound.js)

☑ (SPI를 이용한) 아날로그 소리센서 측정 샘플코드

기준값은 주변소음/상황에 따라 적절한 값을 찾아 설정하여야 함!!
(또는, 가변저항기를 드라이버로 돌려서 기준값을 설정할수도 있음)

```
const  gpio = require('node-wiring-pi');
const  mcpadc = require('mcp-adc');
const  soundsensor = new   mcpadc.Mcp3208();

const  CS_MCP3208    = 10       // Chip Enable(CE0) is set
const  SPI_CHANNEL  = 0        // ADC 0번째 채널선택=아날로그센서
const  SPI_SPEED      = 1350000   // 1.3Mhz
const  QuietSound  = 1935;

const  SoundDetect = function() {
          soundsensor.readRawValue(SPI_CHANNEL, function(value) {
          if (value > QuietSound)
                    console.log("소리인식!   = %d", value);
          else  console.log(".(%d)",value);
        });
         setTimeout(SoundDetect, 500);
}
process.on('SIGINT', function() {
      console.log("Program Exit...");
      process.exit();
});
```

```
gpio.wiringPiSetup();
gpio.wiringPiSPISetup(SPI_CHANNEL, SPI_SPEED);
gpio.pinMode(CS_MCP3208, gpio.OUTPUT);
console.log("마이크에 대고 소리를 내주세요...\n");
console.log("프로그램에서 설정된 기준값이상 커야합니다...\n");
setTimeout(SoundDetect, 500);
```

■ 실습일지