

소켓프로그래밍

라즈베리파이로 배우는 소켓 통신 프로그래밍



동양미래대학교
컴퓨터공학부 정석용



동양미래대학교

GPIO (General Purpose Input Output)

GPIO 핀 x 40

- 주변장치와 통신하기 위해 범용으로 사용하는 입출력 포트
- 라즈베리파이 모델 2 이후 40개의 핀 제공
- 5V 전원용 핀 2개, 3.3V 전원용 핀 2개, 그라운드(GND) 핀 8개, 범용 3.3V 핀 28개

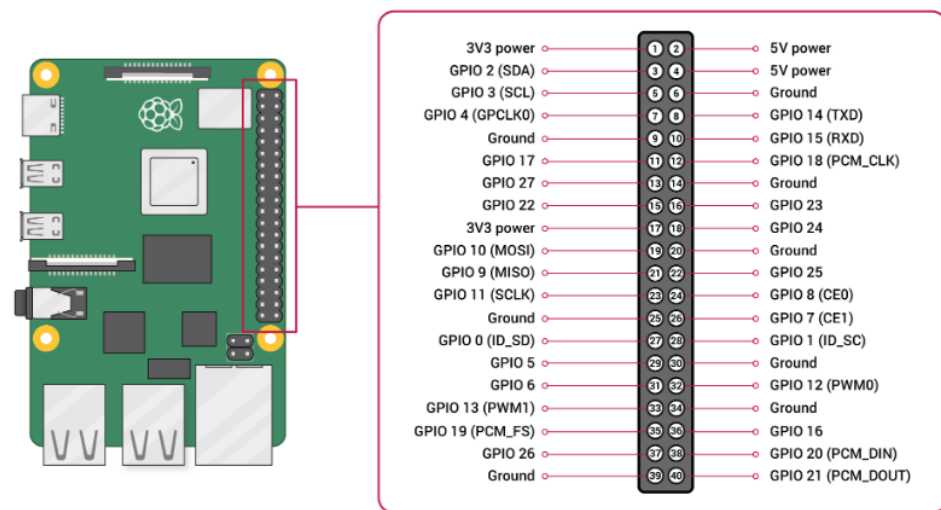


그림 출처:

<https://www.raspberrypi.org/documentation/usage/gpio/>

전원용 GPIO 핀 : 5V 전원용 핀 2개, 3.3V 전원용 핀 2개

- GPIO 핀의 전원은 라즈베리파이의 전원을 이용
- 상시 전원 공급용으로 5V 전원용 핀 2개, 3.3V 전원용 핀 2개 존재
- 많은 전기를 사용하는 디바이스를 연결할 때는 별도의 외부 전원 사용

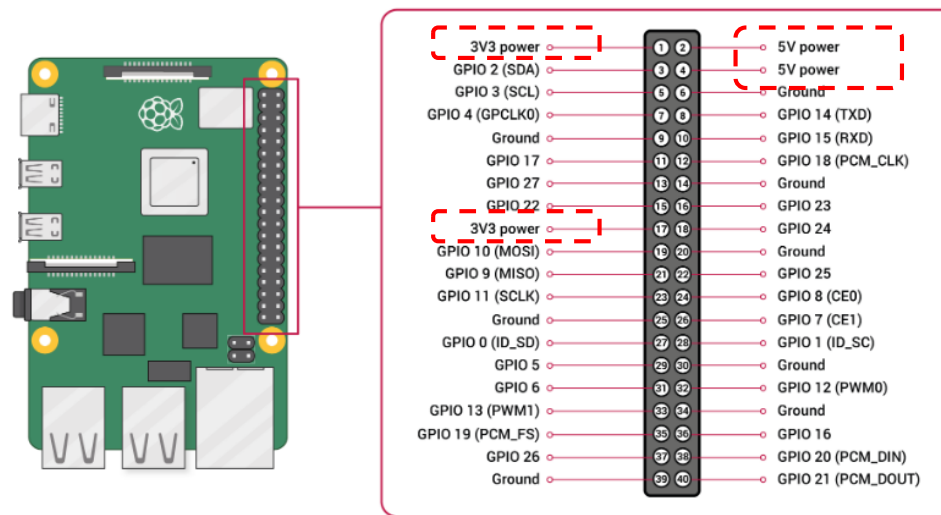


그림 출처:

<https://www.raspberrypi.org/documentation/usage/gpio/>

그라운드 GPIO 핀 : 그라운드(GND) 핀 8개

- 그라운드로 사용할 수 있는 그라운드(GND) 핀 8개 존재

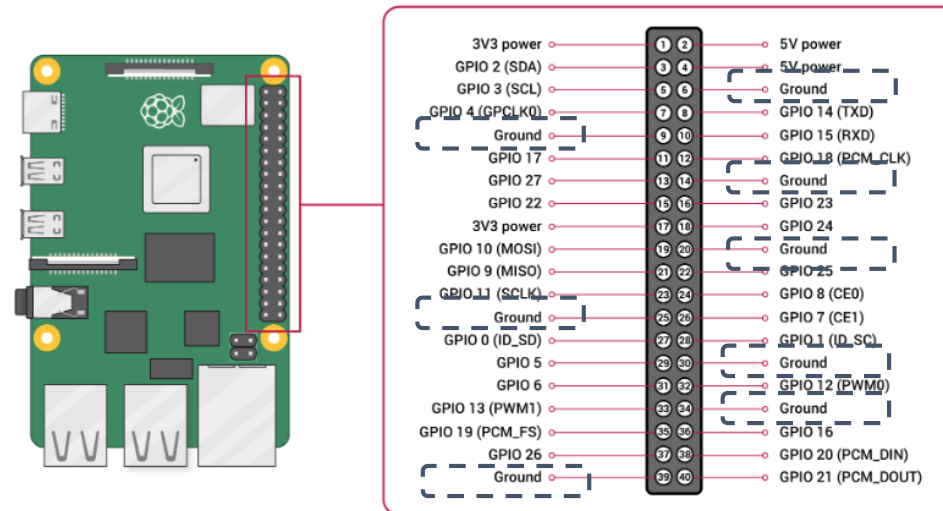


그림 출처:

<https://www.raspberrypi.org/documentation/usage/gpio/>

범용 GPIO 핀 x 28

- 3.3V를 기본 전압으로 사용
- 입력(Input) : 핀을 input type으로 설정,
핀에 걸린 전압이 3.3V이면 ON(High 또는 1), 0V 이면 OFF(low 또는 0) 상태로 파악
- 출력(Output) : 핀을 output type으로 설정,
3.3V 전압을 걸어 ON(High 또는 1), 0V 전압을 걸어 OFF(low 또는 0) 상태로 만들

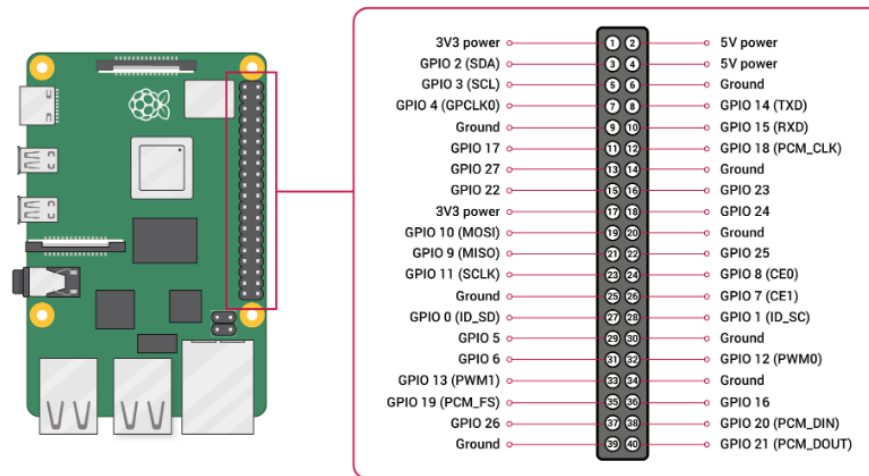
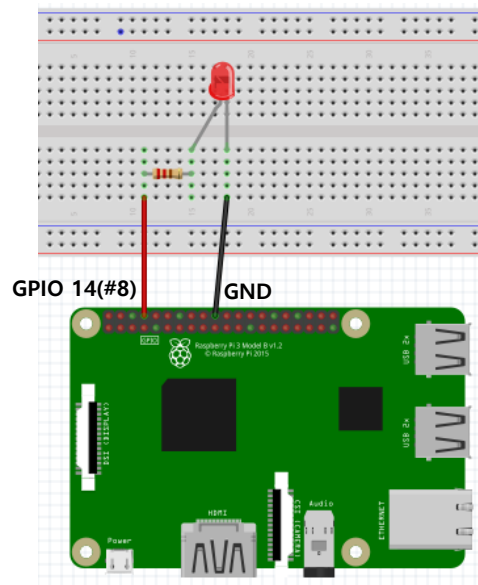


그림 출처:

<https://www.raspberrypi.org/documentation/usage/gpio/>

범용 GPIO 핀 x 28

- 입력 유형(Input type) 설정 :
스위치나 센서처럼 GPIO 핀으로 값을 받아들이는 경우
- 출력 유형(Output type) 설정:
LED나 모터 처럼 GPIO 핀으로 무언가를 동작시키는 Actuator의 경우



[LED 제어의 예]

- ① LED를 제어하는 회로를 구성
- ② 첫 줄 4번째 핀(8번)를 출력 유형(Output Type) 지정
- ③ 핀에 3.3V 전압을 걸어 ON으로 만들면, LED에 불이 들어옴

전자회로 구성

GPIO 제어프로그램으로 제어

범용 GPIO 핀 x 28

- GPIO 핀은 3.3V를 기본 전압으로 사용
- GPIO핀에 3.3V가 아닌 5V 전압을 연결할 때는 전압 레벨을 낮추어 사용해야 함
- GPIO 핀의 값을 설정(ON 또는 Off)로 설정하면 계속 설정된 값을 유지(래치)



GPIO 핀 넘버링

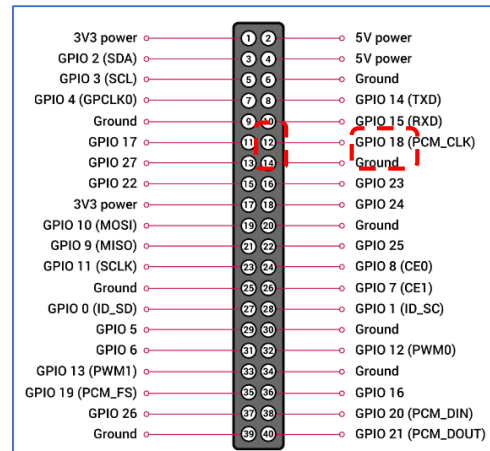
- ① 물리적 넘버링(보드 넘버링) : 기판의 핀 순서대로 일련번호를 붙여 사용
- ② GPIO 넘버링(BCM 넘버링) : 'GPIO + X'로 표기 된 번호
- ③ wiringPi 넘버링 : GPIO 핀을 제어할 수 있는 3rd 파티 C 라이브러리 wiringPi에서 부여한 번호

물리 번호 12 - GPIO 18 - wPi 1

①

②

③



```
pi@raspberrypi: /tmp $ gpio readall
```

BCM	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	BCM
2	8	3.3v			1	2		5v		
3	9	SDA.1	IN	1	3	4		5v		
4	7	SCL.1	IN	1	5	6		0v		
17	0	GPIO. 0	IN	0	11	12	1	IN	15	14
27	2	GPIO. 2	IN	0	13	14	0	IN	16	15
22	3	GPIO. 3	IN	0	15	16	0	IN	1	18
10	12	3.3v			17	18	0	IN	4	23
9	13	MOSI	IN	0	19	20	0	IN	5	24
11	14	MISO	IN	0	21	22	0	IN	6	25
0	30	SCLK	IN	0	23	24	1	IN	10	8
5	21	0v			25	26	1	IN	11	7
6	22	SDA.0	IN	1	27	28	1	IN	31	1
13	23	GPIO. 21	IN	1	29	30		0v		
19	24	GPIO. 22	IN	1	31	32	0	IN	GPIO. 26	26
26	25	GPIO. 23	IN	0	33	34		0v		
		GPIO. 24	IN	0	35	36	0	IN	GPIO. 27	27
		GPIO. 25	IN	0	37	38	0	IN	GPIO. 28	28
		0v			39	40	0	IN	GPIO. 29	29

그림 출처:

<https://www.raspberrypi.org/documentation/usage/gpio/>

GPIO 핀 넘버링 – 물리적 번호

- 기판의 핀 순서대로 일련번호를 붙여 사용
- 보드 넘버링이라고도 부르며, 기판 뒷부분을 보면 알 수 있음
- 항상 1번 핀은 **네모난 납 뿔**으로 되어 있음

Python에서 물리적 번호를 사용하기 위해서는

```
import Rpi.GPIO as GPIO
```

```
GPIO.setmode(GPIO.BOARD)
```

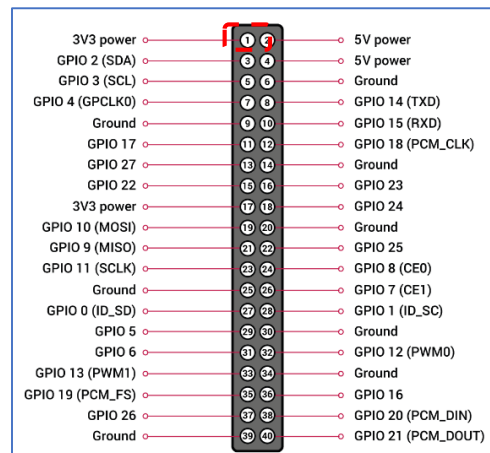


그림 출처:

<https://www.raspberrypi.org/documentation/usage/gpio/>



GPIO 핀 넘버링 – GPIO 넘버링

- 'GPIO + X'로 표기 된 번호
- BCM 넘버링이라고도 부르며, 프린팅해서 참조
- 핀 배열이 바뀌어도 프로그램 변경 없이 사용 가능

Python에서 물리적 번호를 사용하기 위해서는

```
import Rpi.GPIO as GPIO  
GPIO.setmode(GPIO.BCM)
```

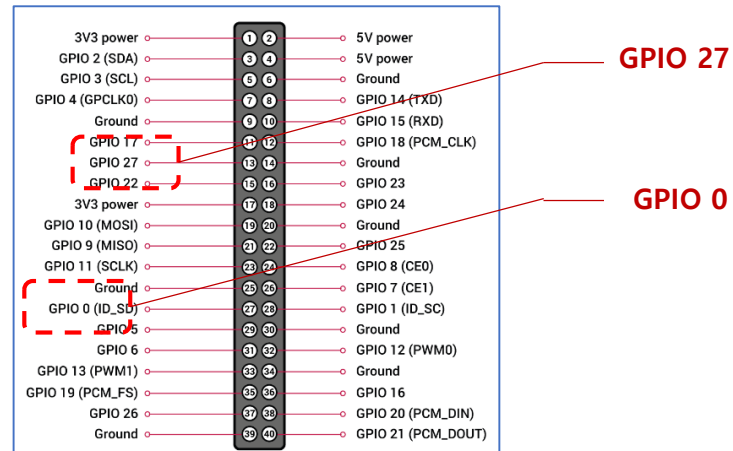


그림 출처:

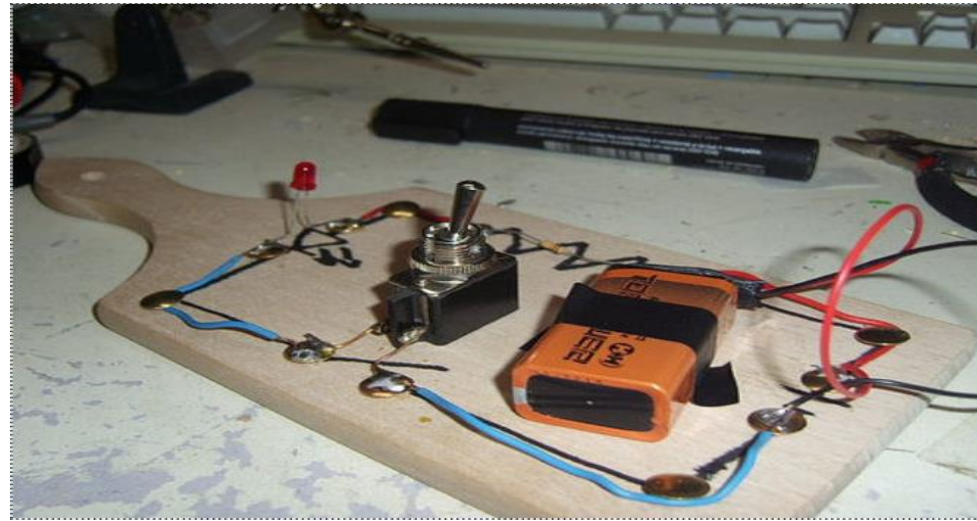
<https://www.raspberrypi.org/documentation/usage/gpio/>

GPIO 핀 넘버링 - GPIO 확장 연결 보드



그림 출처: 자체

회로 구성에 필요한 기본 소자 - 브레드보드

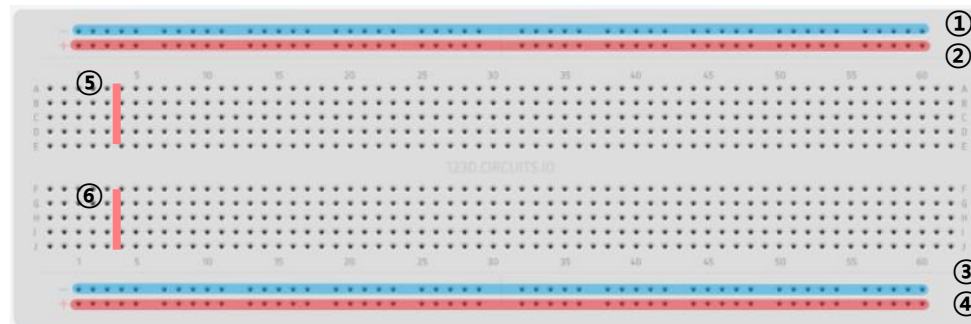


출처:

<https://www.instructables.com/Use-a-real-Bread-Board-for-prototyping-your-circui/>

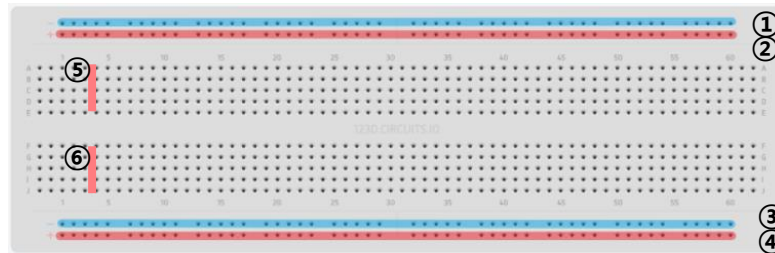
회로 구성에 필요한 기본 소자 - 브레드보드

앞면



- 부품의 핀을 꼽을 수 있는 구멍(hole)을 가짐
- 위 가로 2줄(①,②)과 아래 가로 2줄(③,④)의 구멍(hole)은 전기적으로 연결되어 있음
- 가운데 5개 구멍(hole)의 각각의 세로줄(⑤,⑥)도 전기적으로 연결되어 있음
- 가로 줄(①,②,③,④)은 전원을 공급하기 위해 사용
 - 가로 줄(①,③)에는 전원(-) 또는 GND를 연결
 - 가로 줄(②,④)에는 전원(+)을 연결

회로 구성에 필요한 기본 소자 - 브레드보드



[그림 1] 앞면



[그림 2] 뒷면

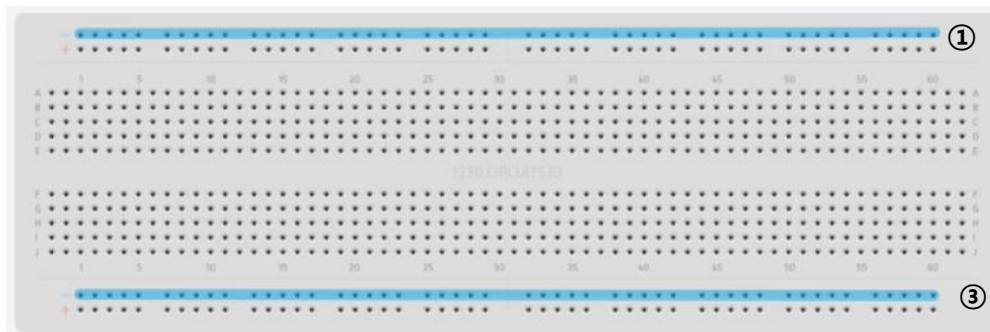
- 보드를 뒤집어 커버를 벗기면
- 가로, 세로 연결 상태를 볼 수 있음

주의

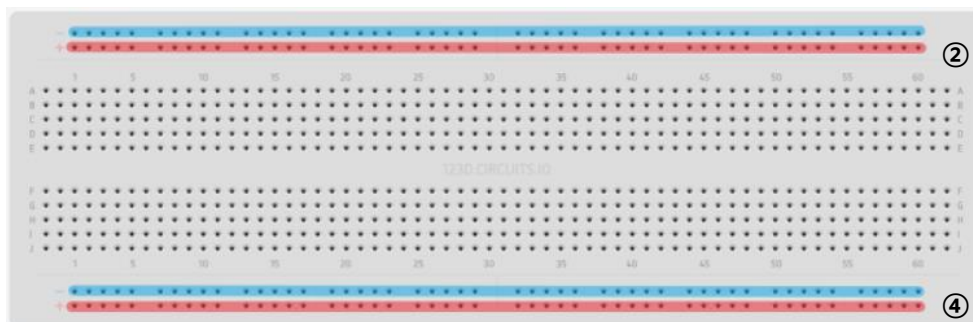
실습을 위해 뒷면 커버를 벗긴 것으로,
커버를 벗기면 보드가 망가질 수 있음

회로 구성에 필요한 기본 소자 - 브레드보드

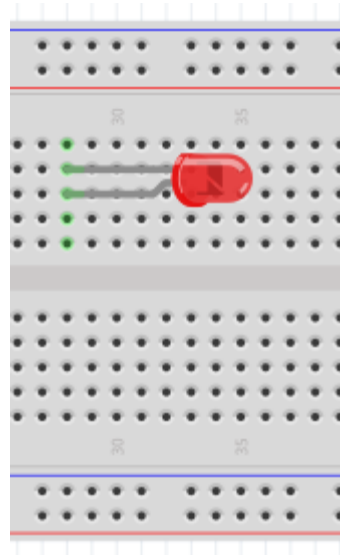
(GND) 가로 줄(①,③) 중 한 구멍에 GND를 연결하면 가로 방향의 모든 구멍을 GND로 사용



(+전원) 가로 줄(②,④) 중 한 구멍에 +전원을 연결하면 가로 방향의 모든 구멍은 +전원 연결

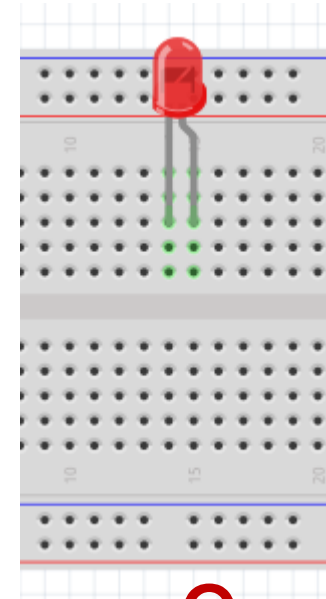


회로 구성에 필요한 기본 소자 - 브레드보드



[그림 1]

X

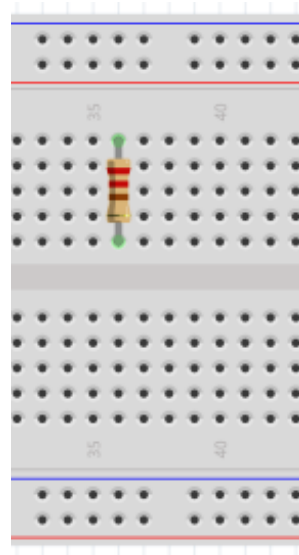


[그림 2]

O

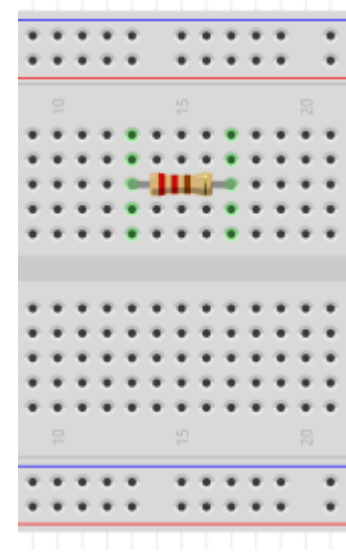
- LED 2개의 핀이 가운데 5개 구멍(hole)의 세로줄에 꼽힘
- LED 2개의 핀은 서로 전기적으로 연결됨 (Short)

회로 구성에 필요한 기본 소자 - 브레드보드



[그림 1]

X

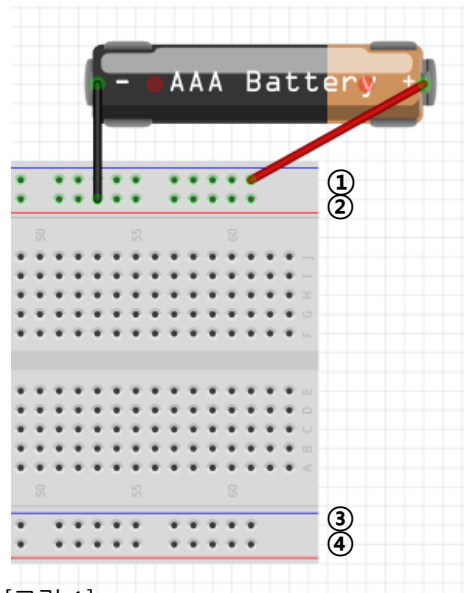


[그림 2]

O

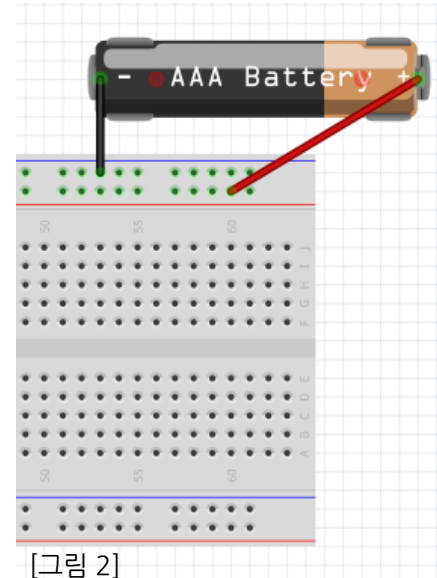
- 저항 2개의 핀이 가운데 5개 구멍(hole)의 세로줄에 꼽힘
- 저항 2개의 핀은 서로 전기적으로 연결됨 (Short)

회로 구성에 필요한 기본 소자 - 브레드보드



[그림 1]

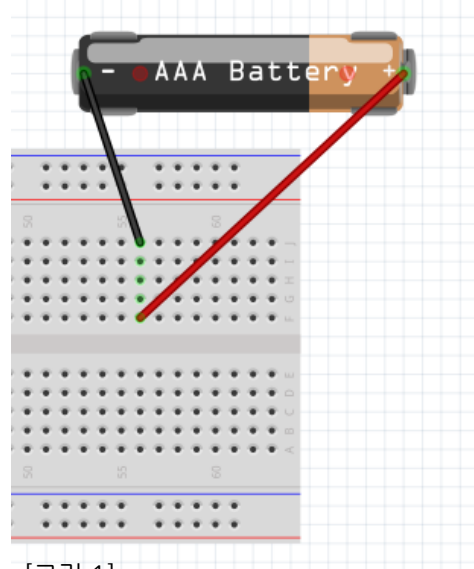
X



[그림 2]

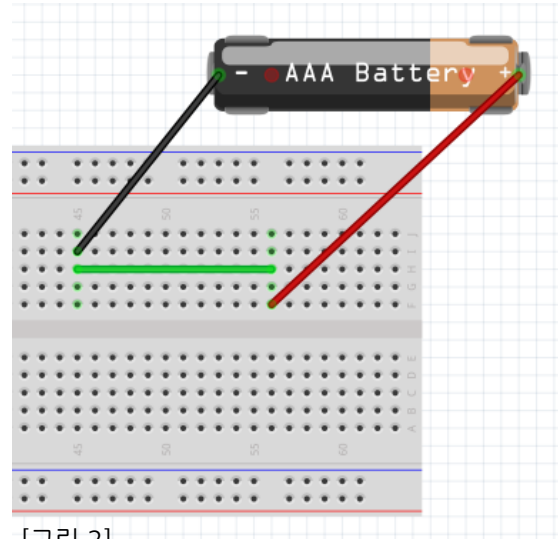
O

- 전통적으로, **빨간색** 케이블은 전원(+)에 연결하고 **검정색** 케이블은 전원(-)나 그라운드 연결에 사용
- 반대로 연결하면, 시각적 혼란을 초래함



[그림 1]

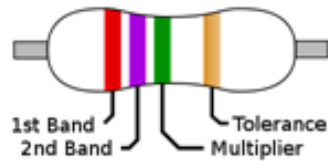
X



[그림 2]

X

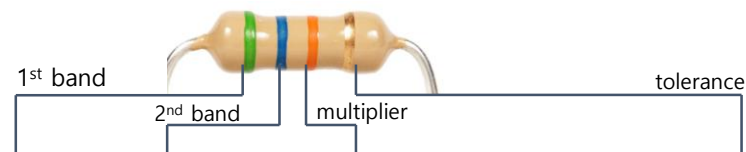
회로 구성에 필요한 기본 소자 - 저항



- 4개의 색 띠를 통해 저항 값을 읽을 수 있다.

- 1st band : 10의 자리 수
- 2nd band : 1의 자리 수
- Multiplier : 10의 제곱수
- Tolerance : 오차 범위

회로 구성에 필요한 기본 소자 - 저항

출처: <https://www.instructables.com/Resistors/>

컬러(Color)	1 st band	2 nd band	10의 제곱수	오차범위	
Black	0	0	$\times 10^0$	-	
Brown	1	1	$\times 10^1$	$\pm 1\%$	F
Red	2	2	$\times 10^2$	$\pm 2\%$	G
Orange	3	3	$\times 10^3$	-	
Yellow	4	4	$\times 10^4$	($\pm 5\%$)	-
Green	5	5	$\times 10^5$	$\pm 0.5\%$	D
Blue	6	6	$\times 10^6$	$\pm 0.25\%$	C
Violet	7	7	$\times 10^7$	$\pm 0.1\%$	B
Gray	8	8	$\times 10^8$	$\pm 0.05\%$ ($\pm 10\%$)	A
White	9	9	$\times 10^9$	-	
Gold	-	-	$\times 10^{-1}$	$\pm 5\%$	J
Silver	-	-	$\times 10^{-2}$	$\pm 10\%$	K
None	-	-	-	$\pm 20\%$	M

회로 구성에 필요한 기본 소자 - 저항

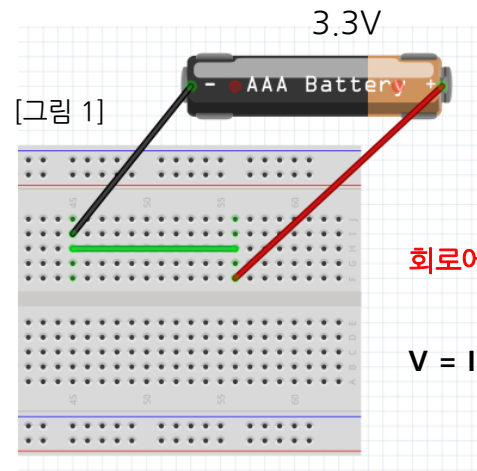


색띠 배열: Brown (1) Black (0) Yellow (10^4) Gold ($\pm 5\%$)

저항 값: $10 \times 10^4 = 100,000 = 100\text{k}\Omega$

저항 종류	저항 값
빨빨갈금	220Ω
청갈흑금	61Ω
갈흑빨금	$1\text{K}\Omega$
빨흑빨금	$2\text{K}\Omega$
갈흑오금	$10\text{k}\Omega$
갈흑청금	$1\text{M}\Omega$

회로 구성에 필요한 기본 소자 - 저항



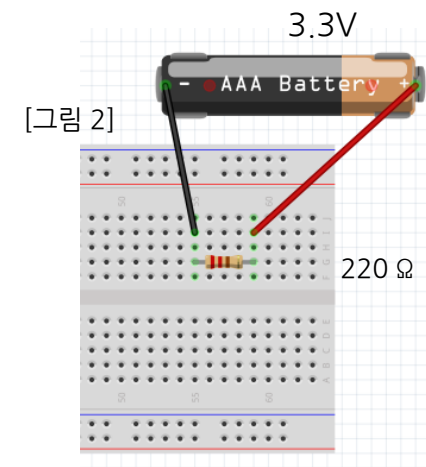
회로에 흐르는 전류는 ?

$$V = IR \quad I = V / R$$

$$I = V / R$$

$$I = 3.3 / 0 = \text{무한대}$$

회로에서 저항을 생략하면 과전류에 의해
회로가 급격히 뜨거워지고 고장



$$I = V / R$$

$$I = 3.3 / 220 = 0.015 \text{ A}$$

회로 구성에 필요한 기본 소자 - 단색 LED

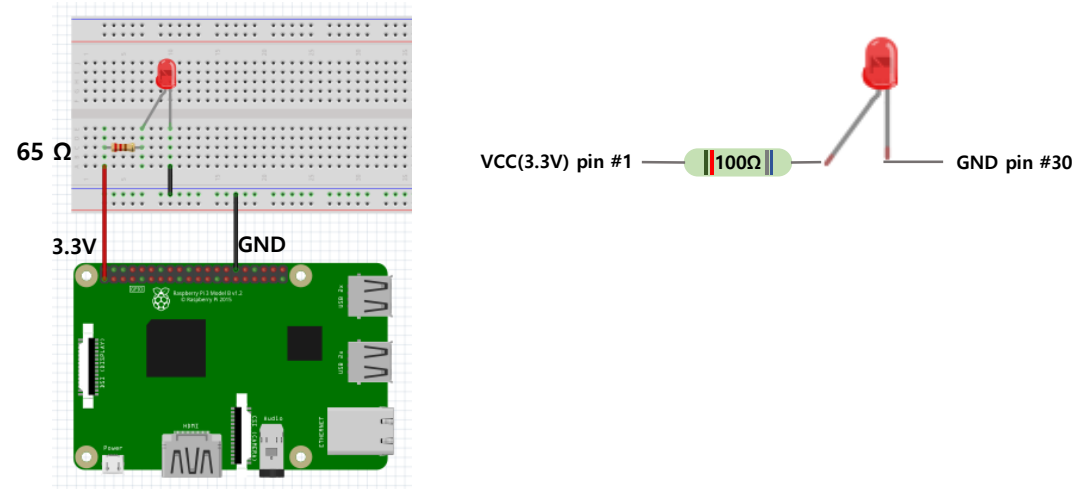


LED light Emitting Diode 발광 다이오드

- 전류가 흐르면 빛을 내는 다이오드
- 회로구성에서 동작 중임을 ON/OFF로 알려주는 인디케이터로 많이 사용
- 극성이 있어 반드시 극성에 맞추어 연결해 주어야 함
 - 긴 다리(+ Anode) : + 전원에 연결
 - 짧은 다리(- Cathod) : - 전원 또는 GND에 연결

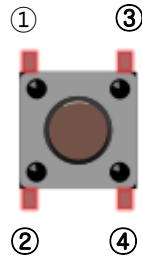
회로 구성에 필요한 기본 소자 - 단색 LED

[실습] LED에 불이 들어오도록 회로를 구성해 보자



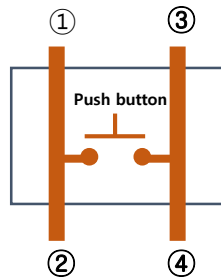
- LED의 Anode(+) 핀은 65 Ω 이상의 저항을 통해 라즈베리파이 3.3V 상시 전원(pin #1)에 연결
- LED의 cathod(-) 핀은 라즈베리파이 GND(pin #30)에 연결
- 라즈베리파이에 전원을 연결하면, LED에 불이 들어오는 것을 확인할 수 있음

회로 구성에 필요한 기본 소자 - 푸시 버튼(Push Button) 스위치

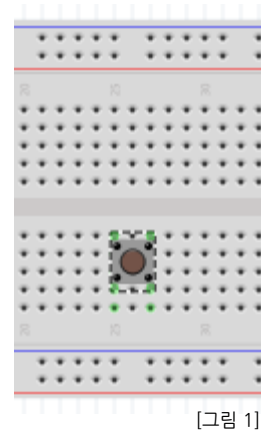
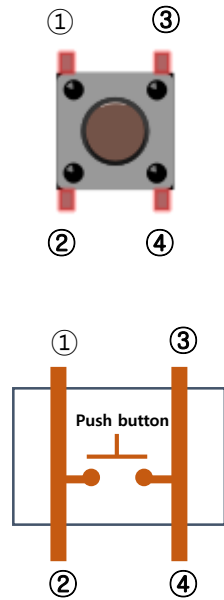


Push Button Switch 푸시 버튼 스위치

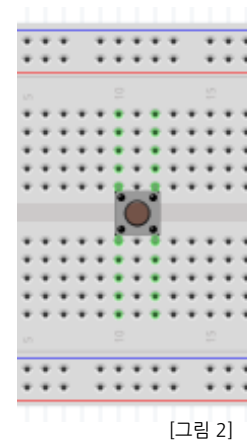
- 4개의 pin으로 구성되고, 2개씩 한 쌍으로 사용
- 거리가 먼 핀(①과②, ③과④)들은 서로 연결됨
- 스위치를 누르면, 모든 핀이 연결되는 구조



회로 구성에 필요한 기본 소자 - 브레드보드



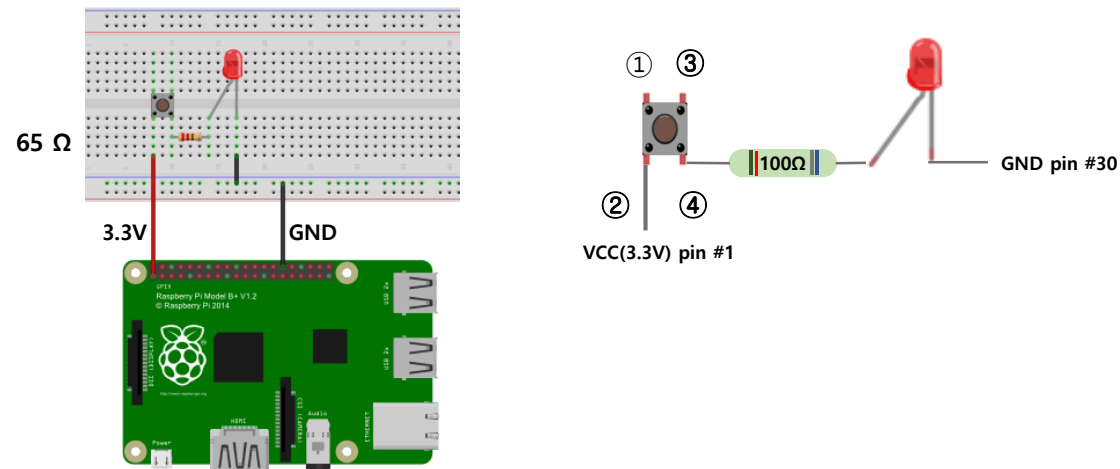
X



O

회로 구성에 필요한 기본 소자 – 푸시 버튼(Push Button) 스위치

[실습] 스위치를 추가하여 스위치를 누르면, LED에 불이 들어오도록 회로를 구성해 보자



- LED의 Anode(+) 핀은 65 Ω 이상의 저항을 통해 푸시 버튼 스위치에 연결
- 푸시 버튼 스위치는 라즈베리파이 3.3V 상시 전원(pin #1)에 연결
- LED의 cathod(-) 핀은 라즈베리파이 GND(pin #30)에 연결
- 푸시버튼을 누르면, LED에 불이 들어오는 것을 확인할 수 있음

WiringPi 설치

WiringPi 설치

- BCM2835/BCM2836 GPIO를 C언어로 제어할 수 있게 제공된 라이브러리
- wiringPi는 Arduino의 wiring 시스템을 사용했던 개발자에게 익숙하도록 설계
- wiringPi은 wiring에서 사용한 함수와 기능을 동일하게 제공

```
$ sudo apt purge wiringpi
```

```
$ hash -r
```

```
$ git clone https://github.com/WiringPi/WiringPi.git
```

```
$ ls
```

```
$ cd WiringPi
```

```
$ git pull origin
```

```
$ ./build
```

```
$ gpio -v
```

```
$ gpio readall
```

GPIO 핀 번호 확인

- wiringPi에서는 GPIO 핀에 대해 BroadCom에서 사용하는 번호 체계와는 다른 번호 체계를 사용
- wiringPi에서 사용하는 GPIO 번호체계는 gpio 명령을 통해 확인 가능

```
$ gpio readall
```

```

pi@raspberrypi: /tmp
pi@raspberrypi: /tmp $ gpio readall
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| BCM | wPi |   Name   | Mode | V  | Physical | V  | Mode |   Name   | wPi | BCM |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 2 | 8 | SDA.1 | IN | 1 | 3 || 4 | | | 5v | | | |
| 3 | 9 | SCL.1 | IN | 1 | 5 || 6 | | | 0v | | |
| 4 | 7 | GPIO. 7 | IN | 1 | 7 || 8 | | 1 | IN | TxD | 15 | 14 |
| | | 0v | | | 9 || 10 | | 1 | IN | RxD | 16 | 15 |
| 17 | 0 | GPIO. 0 | IN | 0 | 11 || 12 | | 0 | IN | GPIO. 1 | 1 | 18 |
| 27 | 2 | GPIO. 2 | IN | 0 | 13 || 14 | | | 0v | | |
| 22 | 3 | GPIO. 3 | IN | 0 | 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 |
| 13 | 23 | GPIO.23 | IN | 0 | 33 || 34 | | | 0v | | |
| 19 | 24 | GPIO.24 | IN | 0 | 35 || 36 | | 0 | IN | GPIO.27 | 27 | 16 |
| 26 | 25 | GPIO.25 | IN | 0 | 37 || 38 | | 0 | IN | GPIO.28 | 28 | 20 |
| | | 0v | | | 39 || 40 | | 0 | IN | GPIO.29 | 29 | 21 |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| BCM | wPi |   Name   | Mode | V  | Physical | V  | Mode |   Name   | wPi | BCM |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+

```


GPIO 핀 번호 확인

`$ gpio readall`

- ① Physical numbering
- ② BCM numbering (GPIO numbering)
- ③ wiringPi(wPi) Numbering

② ③ ① ③ ②

```
pi@raspberrypi: /tmp $ gpio readall
```

BCM	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	BCM
		3.3v			1 2			5v		
2	8	SDA.1	IN	1	3 4			5v		
3	9	SCL.1	IN	1	5 6			0v		
4	7	GPIO. 7	IN	1	7 8	1	IN	TxD	15	14
		0v			9 10	1	IN	RxD	16	15
17	0	GPIO. 0	IN	0	11 12	0	IN	GPIO. 1	1	18
27	2	GPIO. 2	IN	0	13 14			0v		
22	3	GPIO. 3	IN	0	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
13	23	GPIO.23	IN	0	33 34			0v		
19	24	GPIO.24	IN	0	35 36	0	IN	GPIO.27	27	16
26	25	GPIO.25	IN	0	37 38	0	IN	GPIO.28	28	20
		0v			39 40	0	IN	GPIO.29	29	21
BCM	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	BCM

② ③ ① ③ ②

\$ gpio readall

물리번호 12번 핀의 상태 확인

1. input/output mode ? **Input type**
2. 핀의 값? **0**

pi@raspberrypi: /tmp

```
pi@raspberrypi:/tmp $ gpio readall
```

Pi 4B											
BCM	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	BCM	
		3.3v			1	2		5v			
2	8	SDA.1	IN	1	3	4		5v			
3	9	SCL.1	IN	1	5	6		0v			
4	7	GPIO. 7	IN	1	7	8	1	IN	TxD	15	14
		0v			9	10	1	IN	RxD	16	15
17	0	GPIO. 0	IN	0	11	12	0	IN	GPIO. 1	1	18
27	2	GPIO. 2	IN	0	13	14		0v			
22	3	GPIO. 3	IN	0	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
13	23	GPIO.23	IN	0	33	34		0v			
19	24	GPIO.24	IN	0	35	36	0	IN	GPIO.27	27	16
26	25	GPIO.25	IN	0	37	38	0	IN	GPIO.28	28	20
		0v			39	40	0	IN	GPIO.29	29	21

gpio 명령으로 핀 상태 제어 (wPi 번호 사용) – mode 변경

```
$ gpio mode 1 out
```

```
$ gpio readall
```

```

pi@raspberrypi: ~
파일(F) 편집(E) 탭(T) 도움말(H)
pi@raspberrypi:~ $
pi@raspberrypi:~ $
pi@raspberrypi:~ $ gpio mode 1 out
pi@raspberrypi:~ $ gpio readall
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| BCM | wPi |   Name   | Mode | V | Physical | V | Mode |   Name   | wPi | BCM |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 2 | 8 | 3.3v |      |   | 3 | 4 |      | 5v |      |      | |
| 3 | 9 | SCL.1 | ALT0 | 1 | 5 | 6 |      | 5v |      |      |
| 4 | 7 | GPIO. 7 | IN | 1 | 7 | 8 | 1 | ALT5 | TxD | 15 | 14 |
|   |   | 0v |      |   | 9 | 10 | 1 | ALT5 | RxD | 16 | 15 |
| 17 | 0 | GPIO. 0 | IN | 0 | 11 | 12 | 0 | OUT | GPIO. 1 | 1 | 18 |
| 27 | 2 | GPIO. 2 | IN | 0 | 13 | 14 |      | 0v |      |      |
| 22 | 3 | GPIO. 3 | IN | 0 | 15 | 16 | 0 | IN | GPIO. 4 | 4 | 23 |
|   |   | 3.3v |      |   | 17 | 18 | 0 | IN | GPIO. 5 | 5 | 24 |
| 10 | 12 | MOSI | ALT0 | 0 | 19 | 20 |      | 0v |      |      |
| 9 | 13 | MISO | ALT0 | 0 | 21 | 22 | 0 | IN | GPIO. 6 | 6 | 25 |
| 11 | 14 | SCLK | ALT0 | 0 | 23 | 24 | 1 | OUT | CE0 | 10 | 8 |
|   |   | 0v |      |   | 25 | 26 | 1 | OUT | 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 |
| 13 | 23 | GPIO.23 | IN | 0 | 33 | 34 |      | 0v |      |      |
| 19 | 24 | GPIO.24 | IN | 0 | 35 | 36 | 0 | IN | GPIO.27 | 27 | 16 |
| 26 | 25 | GPIO.25 | IN | 0 | 37 | 38 | 0 | IN | GPIO.28 | 28 | 20 |
|   |   | 0v |      |   | 39 | 40 | 0 | IN | GPIO.29 | 29 | 21 |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| BCM | wPi |   Name   | Mode | V | Physical | V | Mode |   Name   | wPi | BCM |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
pi@raspberrypi:~ $

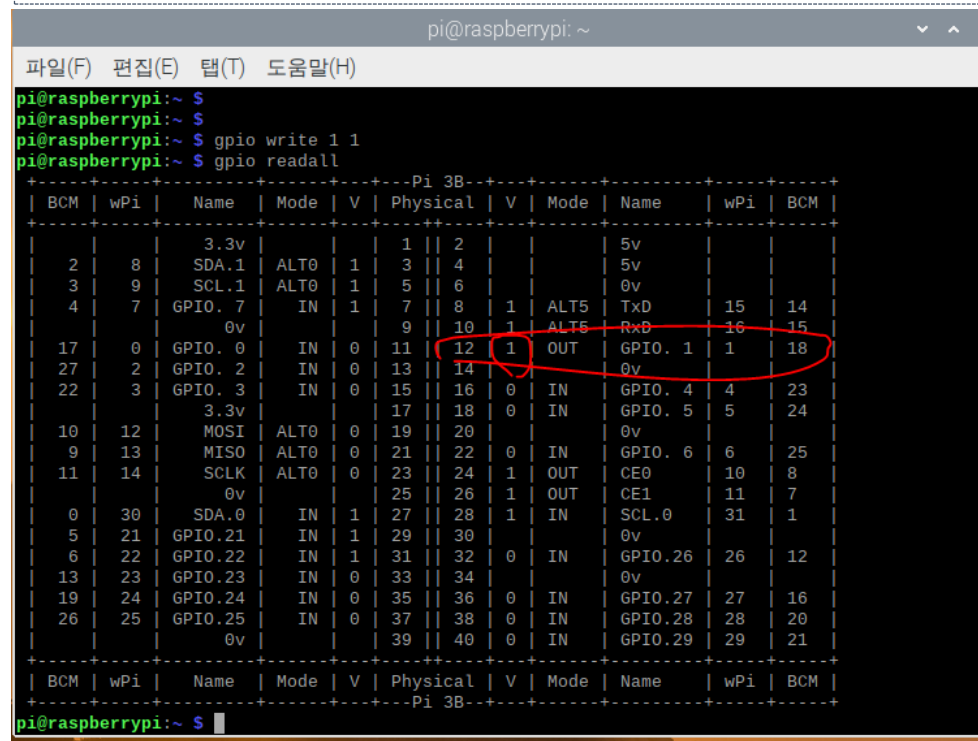
```

(토의 1) wPi 1번 핀의 모드가 IN(INPUT)에서 OUT(OUTPUT)으로 변경 된 것을 확인하자

gpio 명령으로 핀 상태 제어 (wPi 번호 사용) - 값 변경

```
$ gpio write 1 1
```

```
$ gpio readall
```



```
pi@raspberrypi: ~  
파일(F) 편집(E) 탭(T) 도움말(H)  
pi@raspberrypi:~ $  
pi@raspberrypi:~ $  
pi@raspberrypi:~ $ gpio write 1 1  
pi@raspberrypi:~ $ gpio readall  
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+  
| BCM | wPi | Name | Mode | V | Physical | V | Mode | Name | wPi | BCM |  
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+  
| 2 | 8 | 3.3v | | | 1 | 2 | | | 5v | | |  
| 3 | 9 | SCL.1 | ALT0 | 1 | 5 | 6 | | | 5v | | |  
| 4 | 7 | GPIO. 7 | IN | 1 | 7 | 8 | 1 | ALT5 | TxD | 15 | 14 |  
| | | 0v | | | 9 | 10 | 1 | ALT6 | RxD | 16 | 15 |  
| 17 | 0 | GPIO. 0 | IN | 0 | 11 | 12 | 1 | OUT | GPIO. 1 | 1 | 18 |  
| 27 | 2 | GPIO. 2 | IN | 0 | 13 | 14 | | | 0v | | |  
| 22 | 3 | GPIO. 3 | IN | 0 | 15 | 16 | 0 | IN | GPIO. 4 | 4 | 23 |  
| | | 3.3v | | | 17 | 18 | 0 | IN | GPIO. 5 | 5 | 24 |  
| 10 | 12 | MOSI | ALT0 | 0 | 19 | 20 | | | 0v | | |  
| 9 | 13 | MISO | ALT0 | 0 | 21 | 22 | 0 | IN | GPIO. 6 | 6 | 25 |  
| 11 | 14 | SCLK | ALT0 | 0 | 23 | 24 | 1 | OUT | CE0 | 10 | 8 |  
| | | 0v | | | 25 | 26 | 1 | OUT | 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 |  
| 13 | 23 | GPIO.23 | IN | 0 | 33 | 34 | | | 0v | | |  
| 19 | 24 | GPIO.24 | IN | 0 | 35 | 36 | 0 | IN | GPIO.27 | 27 | 16 |  
| 26 | 25 | GPIO.25 | IN | 0 | 37 | 38 | 0 | IN | GPIO.28 | 28 | 20 |  
| | | 0v | | | 39 | 40 | 0 | IN | GPIO.29 | 29 | 21 |  
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+  
| BCM | wPi | Name | Mode | V | Physical | V | Mode | Name | wPi | BCM |  
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+  
pi@raspberrypi:~ $
```

(토의 2) wPi 1번 핀의 값이 0에서 1로 변경 된 것을 확인하자.

gpio 명령으로 핀 상태 제어 (BCM 번호 사용) – mode 변경

```
$ gpio -g mode 18 in
```

```
$ gpio readall
```

```

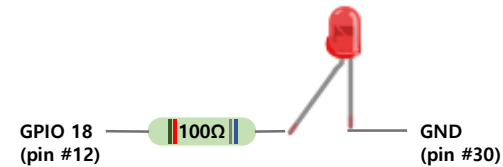
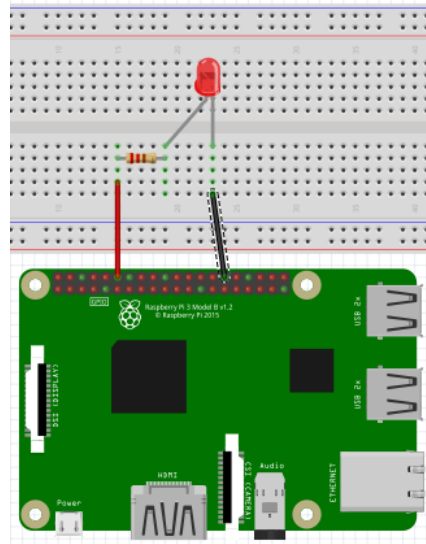
pi@raspberrypi: ~
파일(F) 편집(E) 탭(T) 도움말(H)
pi@raspberrypi:~ $
pi@raspberrypi:~ $
pi@raspberrypi:~ $ gpio -g mode 18 in
pi@raspberrypi:~ $ gpio readall
+-----+-----Pi 3B-----+-----+
| BCM | wPi |   Name   | Mode | V | Physical | V | Mode |   Name   | wPi | BCM |
+-----+-----+-----+-----+-----+-----+-----+-----+
|  2  |  8  |  SDA.1   | ALT0  | 1 |  3  |  4  |      |  5v      |      |      | |
|  3  |  9  |  SCL.1   | ALT0  | 1 |  5  |  6  |      |  0v      |      |      |
|  4  |  7  | GPIO. 7   | IN    | 1 |  7  |  8  |  1  | ALT5    | TxD   | 15  | 14  |
|      |      |  0v      |      |   |  9  | 10  |  1  | ALT5    | RxD   | 16  | 15  |
| 17  |  0  | GPIO. 0   | IN    | 0 | 11  | 12  |  0  | IN      | GPIO. 1 | 1  | 18  |
| 27  |  2  | GPIO. 2   | IN    | 0 | 13  | 14  |      |  0v      |      |      |
| 22  |  3  | GPIO. 3   | IN    | 0 | 15  | 16  |  0  | IN      | GPIO. 4 | 4  | 23  |
|      |      |  3.3v    |      |   | 17  | 18  |  0  | IN      | GPIO. 5 | 5  | 24  |
| 10  | 12  |  MOSI    | ALT0  | 0 | 19  | 20  |      |  0v      |      |      |
|  9  | 13  |  MISO    | ALT0  | 0 | 21  | 22  |  0  | IN      | GPIO. 6 | 6  | 25  |
| 11  | 14  |  SCLK    | ALT0  | 0 | 23  | 24  |  1  | OUT     | CE0    | 10  |  8  |
|      |      |  0v      |      |   | 25  | 26  |  1  | OUT     | 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  |
| 13  | 23  | GPIO.23   | IN    | 0 | 33  | 34  |      |  0v      |      |      |
| 19  | 24  | GPIO.24   | IN    | 0 | 35  | 36  |  0  | IN      | GPIO.27 | 27  | 16  |
| 26  | 25  | GPIO.25   | IN    | 0 | 37  | 38  |  0  | IN      | GPIO.28 | 28  | 20  |
|      |      |  0v      |      |   | 39  | 40  |  0  | IN      | GPIO.29 | 29  | 21  |
+-----+-----+-----+-----+-----+-----+-----+-----+
| BCM | wPi |   Name   | Mode | V | Physical | V | Mode |   Name   | wPi | BCM |
+-----+-----+-----+-----+-----+-----+-----+-----+
pi@raspberrypi:~ $

```

(토의 3) BCM 18번 핀의 모드가 OUT(OUTPUT)에서 IN(INPUT)으로 변경 된 것을 확인하자

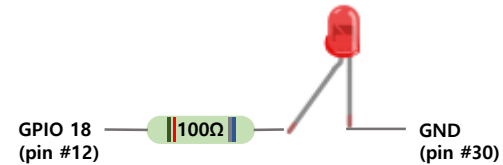
(실습과제 1) gpio 명령을 이용한 LED 점멸

- 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

- gpio 명령 실행



```
$ gpio -g mode 18 out  
$ gpio -g write 18 1  
$ gpio -g write 18 0  
$ gpio -g write 18 1  
$ gpio -g write 18 0
```

- LED에 신호를 주기위해 GPIO 18번 핀을 출력용으로 설정 (output type)
- GPIO 18번 핀을 1 값으로 설정하면, 핀에 3.3V 전압이 걸리고 LED가 켜짐
- GPIO 18번 핀을 0 값으로 설정하면, 핀에 0V 전압이 걸리고 LED가 꺼짐

LED 반짝거리기

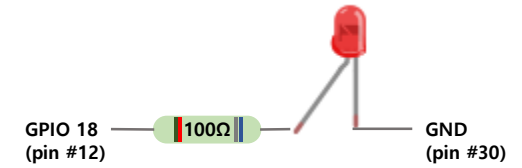
```
import RPi.GPIO as GPIO
import time

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

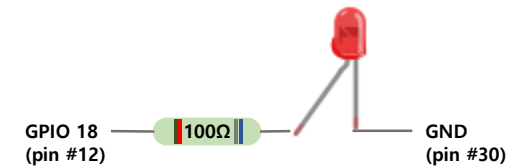
```
import socket
import json

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

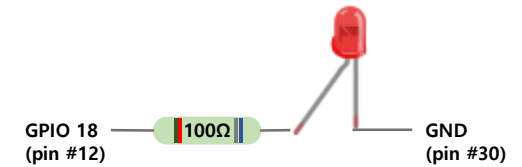
```
import socket
import json

s = socket.socket()
s.setsockopt(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
```

