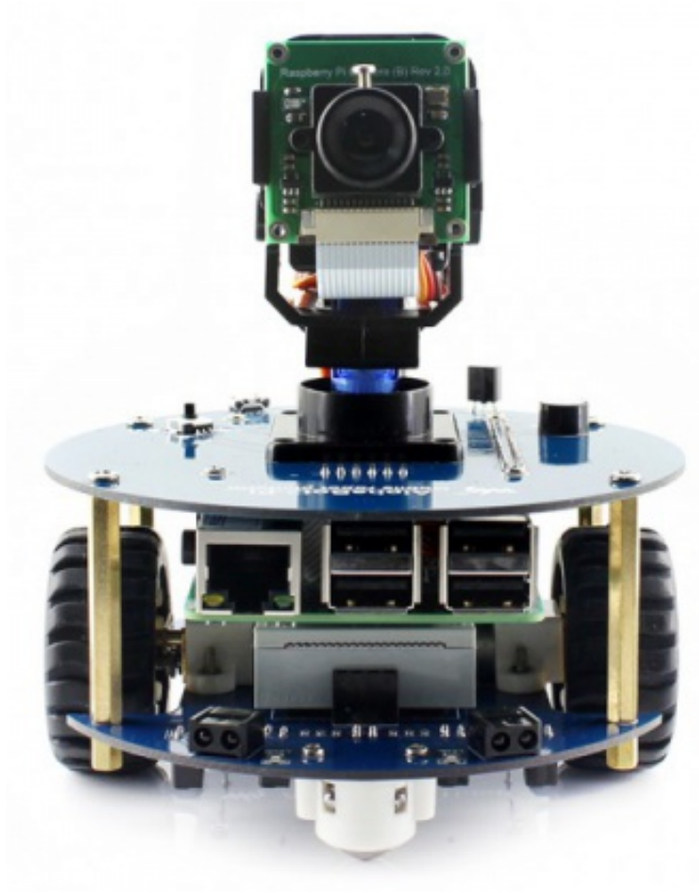
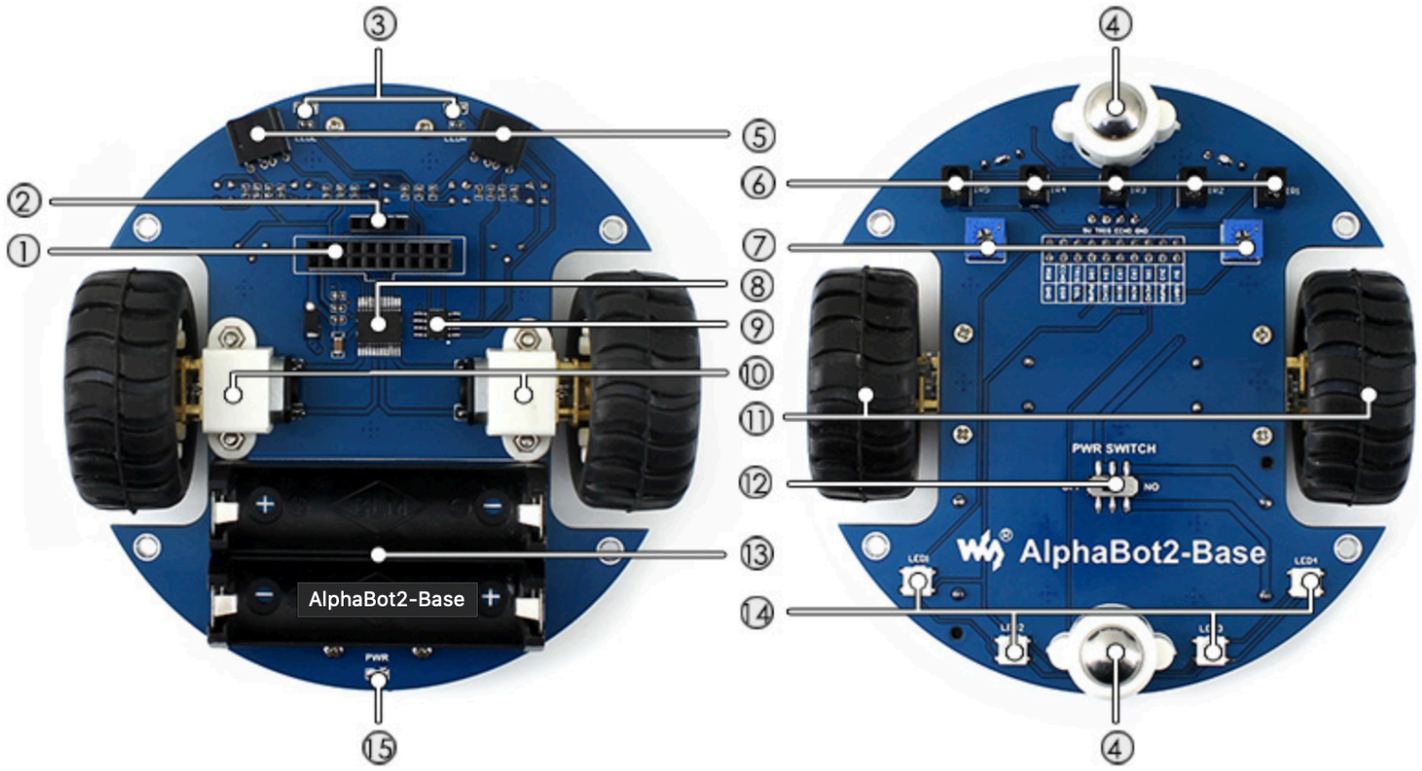


제2장

AlphaBot2-Pi

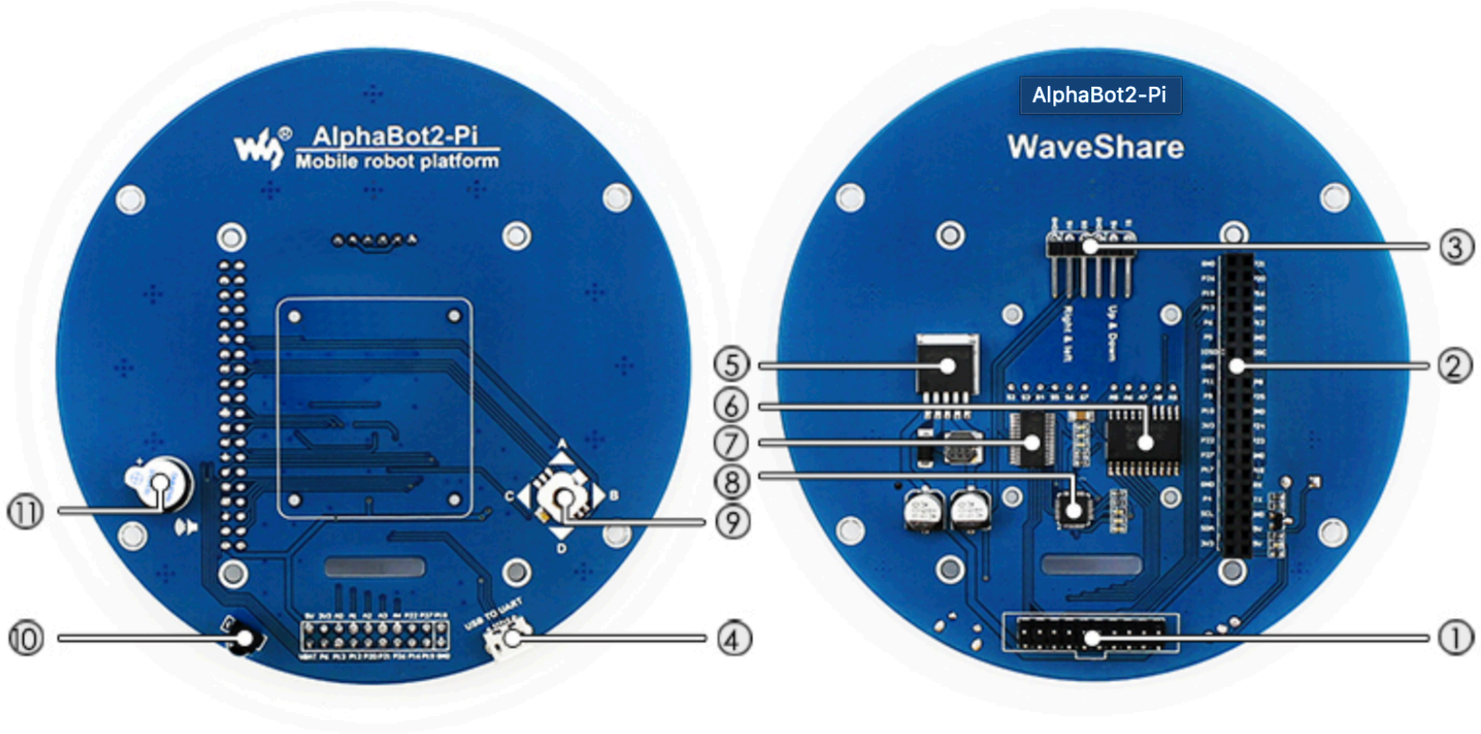
AlphaBot2-Pi

AlphaBot2-Base



1. Ultrasonic module interface
2. AlphaBot2 control interface: for connecting sorts of controller adapter board
3. Obstacle avoiding indicators
4. Omni-direction wheel
5. ST188: reflective infrared photoelectric sensor, for obstacle avoiding
6. ITR20001/T: reflective infrared photoelectric sensor, for line tracking
7. Potentiometer for adjusting obstacle avoiding range
8. TB6612FNG dual H-bridge motor driver
9. LM393 voltage comparator
10. N20 micro gear motor reduction rate 1:30, 6V/600RPM
11. Rubber wheels diameter 42mm, width 19mm
12. Power switch
13. Battery holder: supports 14500 batteries
14. WS2812B: true color RGB LEDs
15. Power indicator

AlphaBot2-Pi



1. AlphaBot2 control interface: for connecting AlphaBot2-Base
2. Raspberry Pi interface: for connecting Raspberry Pi 3 Model B
3. Servo interface
4. USB TO UART: easy for controlling the Pi via UART
5. LM2596: 5V voltage regulator
6. TLC1543: 10-bit AD acquisition chip, allows the Pi to use analog sensors
7. PCA9685: servo controller, make it more smoothly to rotate the pan head
8. CP2102: USB TO UART converter
9. Joystick
10. IR receiver
11. Buzzer

AlphaBot2 Assembly

- https://www.youtube.com/watch?v=ONg0qpxYWQo

예제 코드 다운로드

```
pi@raspberrypi:~ $ sudo apt update
...
Reading state information... Done
10 packages can be upgraded. Run 'apt list --upgradable' to see them.

pi@raspberrypi:~ $ wget https://www.waveshare.com/w/upload/7/74/AlphaBot2.tar.gz
pi@raspberrypi:~ $ tar zxvf AlphaBot2.tar.gz
pi@raspberrypi:~ $ cd AlphaBot2
```

AlphaBot 예제 코드를 다운로드하고 압축을 푼다.

Python Virtual Environment

RPi에 설치된 Python2와 3의 버전을 확인한다.

```
pi@raspberrypi:~/AlphaBot2 $ python --version
Python 2.7.16
pi@raspberrypi:~/AlphaBot2 $ python3 --version
Python 3.7.3
pi@raspberrypi:~/AlphaBot2 $ python3 -m venv .venv
pi@raspberrypi:~/AlphaBot2 $ source .venv/bin/activate
(.venv) pi@raspberrypi:~/AlphaBot2 $
(.venv) pi@raspberrypi:~/AlphaBot2 $ python --version
Python 3.7.3
```

~/AlphaBot2 디렉토리 내부에 .venv라는 이름의 virtual env를 **생성**하고 **활성화**한다.

Setup and Test

pip은 python package management tool이다.

RPi.GPIO 패키지를 설치한다.
(https://sourceforge.net/p/raspberry-gpio-python/wiki/Examples/)

```
(.venv) pi@raspberrypi:~/AlphaBot2 $ pip install RPi.GPIO
Looking in indexes: https://pypi.org/simple, https://www.piwheels.org/simple
Collecting RPi.GPIO
  Downloading https://www.piwheels.org/simple/rpi-gpio/RPi.GPIO-0.7.0-cp37-cp37m-
linux_armv7l.whl (69kB)
    100% |████████████████████████████████| 71kB 107kB/s
Installing collected packages: RPi.GPIO
Successfully installed RPi.GPIO-0.7.0
(.venv) pi@raspberrypi:~/AlphaBot2 $ cd python/
(.venv) pi@raspberrypi:~/AlphaBot2/python $ python AlphaBot2.py
(.venv) pi@raspberrypi:~/AlphaBot2/python $
```

예제 코드 AlphaBot2.py를 실행하여 AlphaBot이
정상적으로 동작하는지 확인한다.
Control-C를 누르면 종료한다.

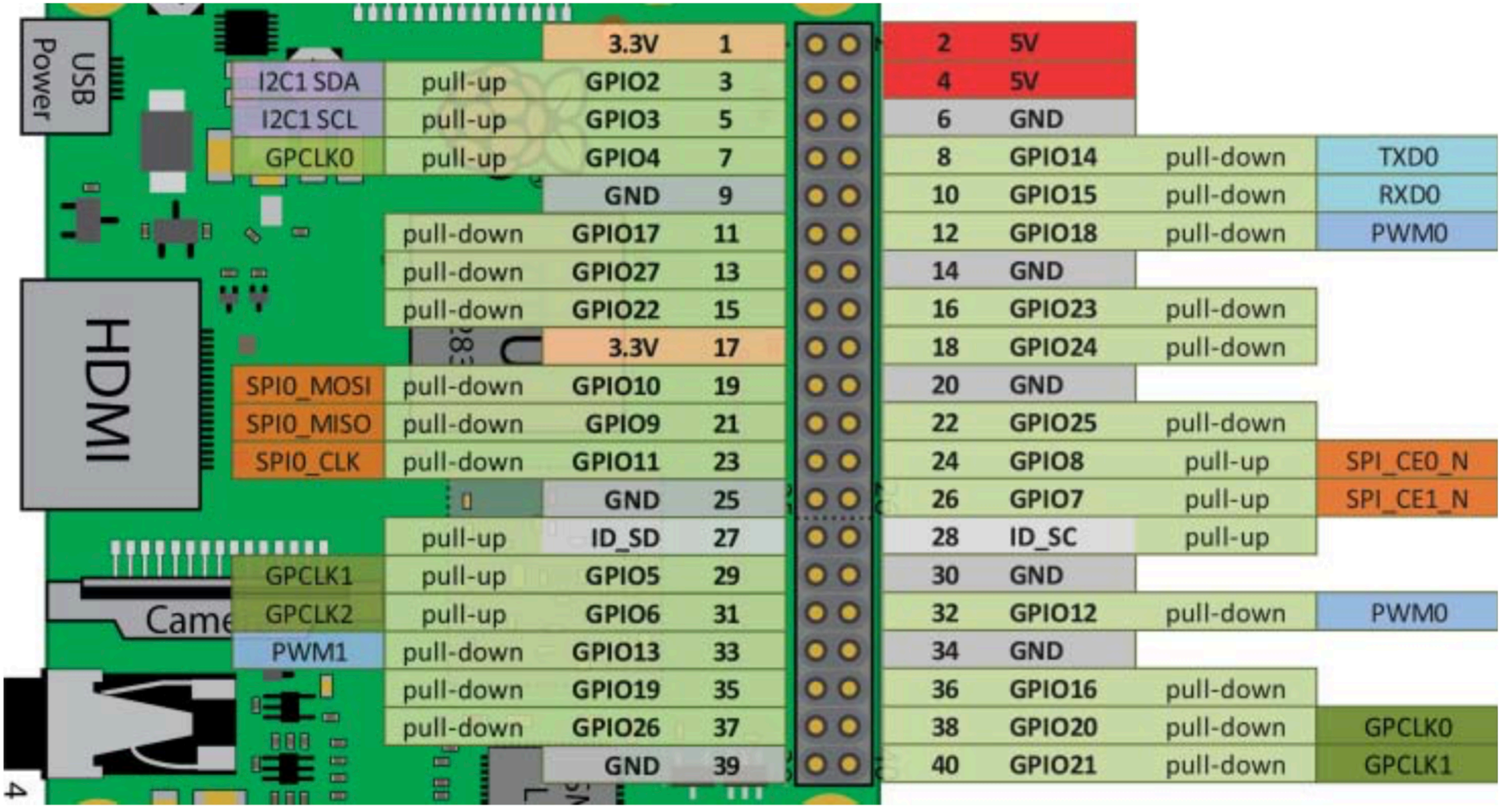
GPIO and PWM

GPIO

- **GPIO (general-purpose input/output)는 목적이 미리 정해져 있지 않는 마이크로컨트롤러의 핀**
- **사용자에 의해서 실행시간(run time)에 입력 혹은 출력 목적으로 선택하여 사용**

GPIOs	40 pin (or 26 pin) GPIO header	40 pins that are multiplexed to provide access to the features listed on the following table rows. Not all functionality is available at the same time. These inputs and outputs are described in detail in Chapter 6 and Chapter 8.
	26 x GPIOs	General purpose inputs outputs that are used for reading or writing binary data. The maximum number of GPIOs is 26 on the 40 pin RPi models. All GPIOs are 3.3V tolerant. Using buses and other interfaces reduces the number of available GPIOs.
	2 x I^2C bus	I^2C is a digital bus that allows you to connect several modules to each of the two-wire buses at the same time. One of these two buses is reserved for HAT support.
	SPI bus	Serial peripheral interface (SPI) provides a synchronous serial data link over short distances. It uses a master/slave configuration and requires 4 wires for communication. The RPi SPI bus has Linux support for two slave select lines.
	UART	Used for serial communication between two devices. The RPi typically (except the RPi 3) has one UART device that is allocated by default to providing a serial console connection.
	PWM	Pulse width modulation (PWM) outputs allow you to send a type of analog output that can be used to control devices (e.g., motors). There is at least one hardware PWM output on all RPi boards, and two on more recent boards.
	GPCLK	General purpose clocks (GPCLK) allow you to establish accurate timing signals.

GPIO Headers in RPi 2/3



RPi GPIO: 3가지 접근 방법

- **sysfs 파일시스템(/sys)을 통해서**

```
pi@raspberrypi:/sys/class/gpio $ echo 4 > export
pi@raspberrypi:/sys/class/gpio $ cd gpio4/
pi@raspberrypi:/sys/class/gpio/gpio4 $ ls
active_low  device  direction  edge  power  subsystem  uevent  value
pi@raspberrypi:/sys/class/gpio/gpio4 $ cat direction
in
pi@raspberrypi:/sys/class/gpio/gpio4 $ cat value
1
pi@raspberrypi:/sys/class/gpio/gpio4 $ cat value
0
```

- **Memory-mapped I/O를 이용해서**
- **wiringPi 라이브러리를 이용해서**

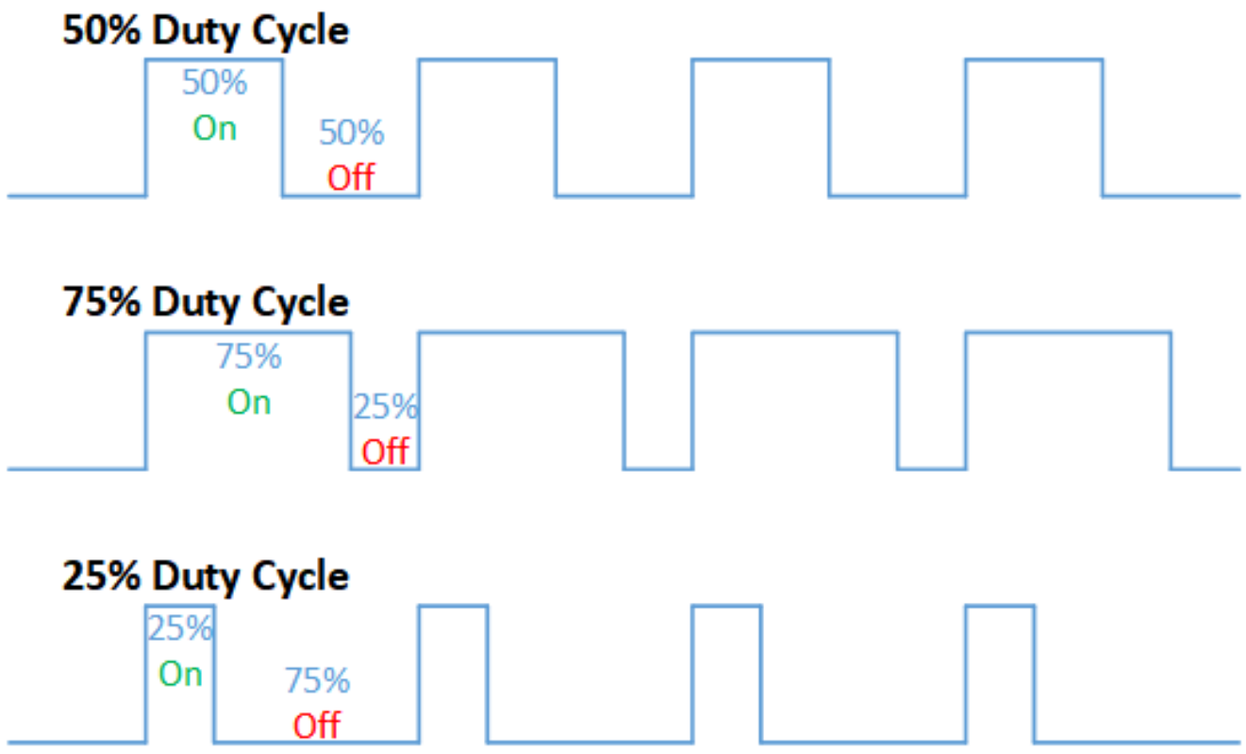
RPi GPIO Python Libraries

- **WiringPi (WiringPi2)**
- **RPi.GPIO** ← most popular
- **RPIO**
- **gpiozero**
- **pigpio**

RPi.GPIO tutorial:
https://sourceforge.net/p/raspberry-gpio-python/wiki/BasicUsage/

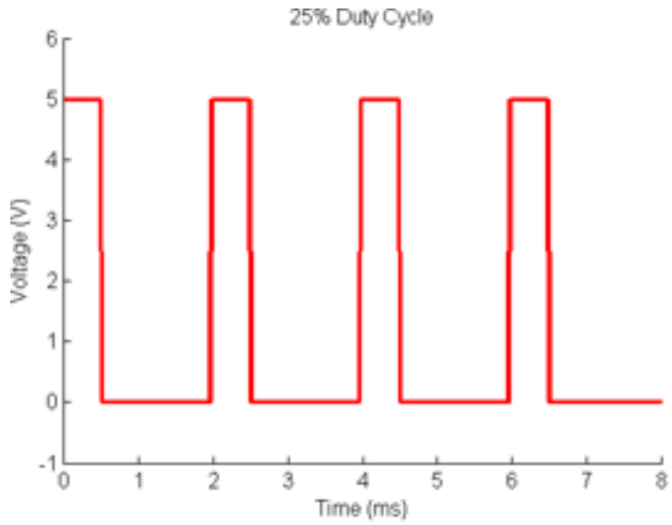
PWM (Pulse Width Modulation)

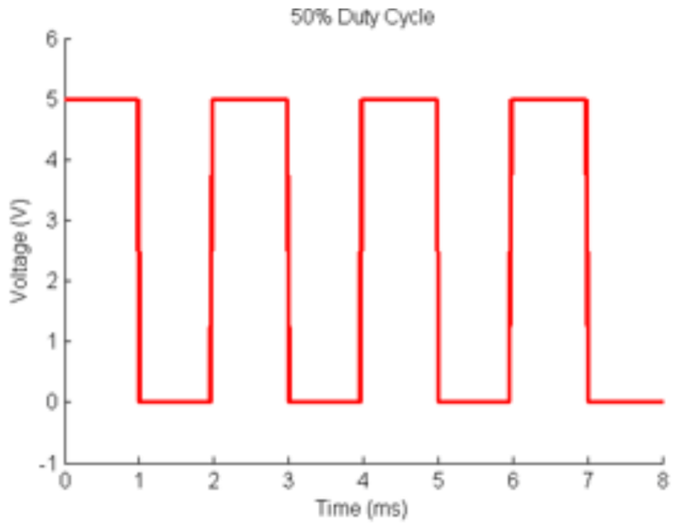
- **Pulse Width를 변경하여 전달되는 평균 전력을 조절하는 방법**

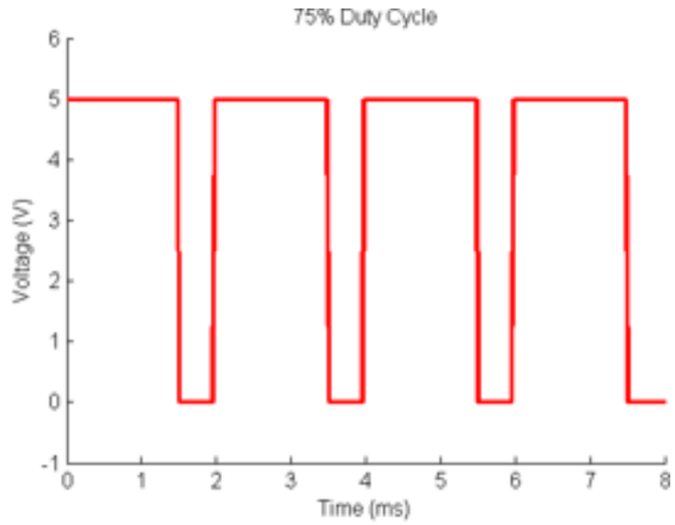


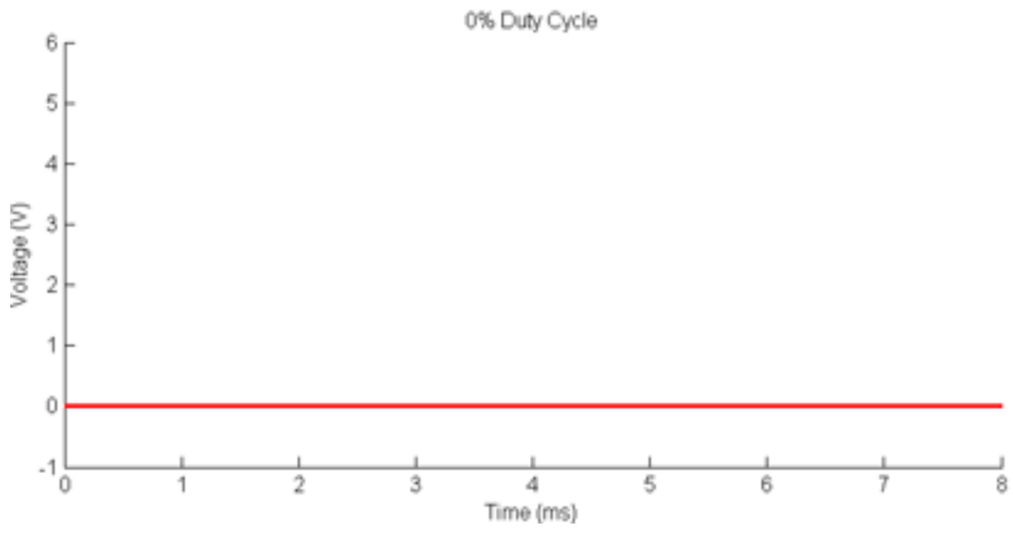
- **RPi는 DAC이 없으므로 아날로그 출력을 지원하지 않는다. 대신 PWM을 이용하여 유사한 효과를 얻을 수 있다.**
- **대표적인 응용: LED 전등의 밝기 조절 (Dimming), DC 모터의 속도 제어**

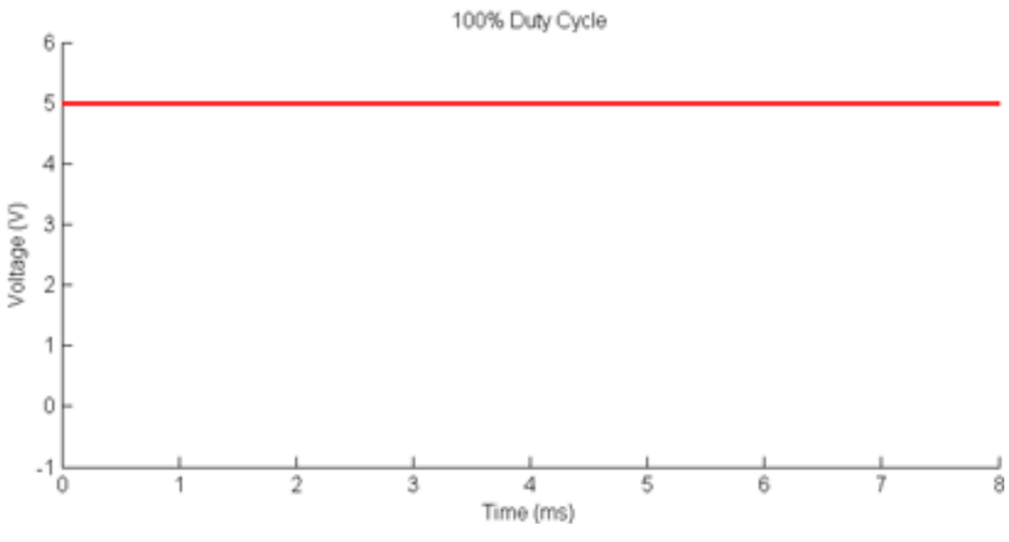
PWM with varying duty cycles









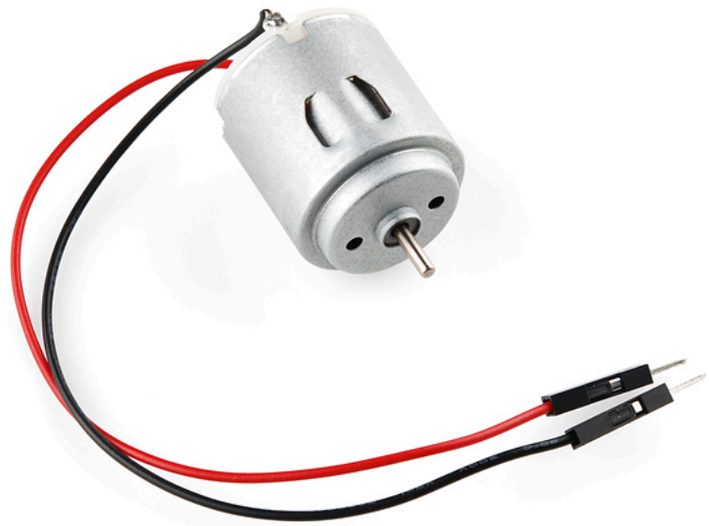


PWM in RPi

- **RPi 3 모델은 2개의 Hardware PWM 채널을 제공**
  - PWM0 on Pin 12 (GPIO18)
  - PWM1 on Pin 33 (GPIO13)
- **RPi.GPIO 라이브러리를 이용해 다른 핀들을 소프트웨어적으로 PWM 핀으로 사용 가능**
- **참고: RPi.GPIO의 PWM 사용법**

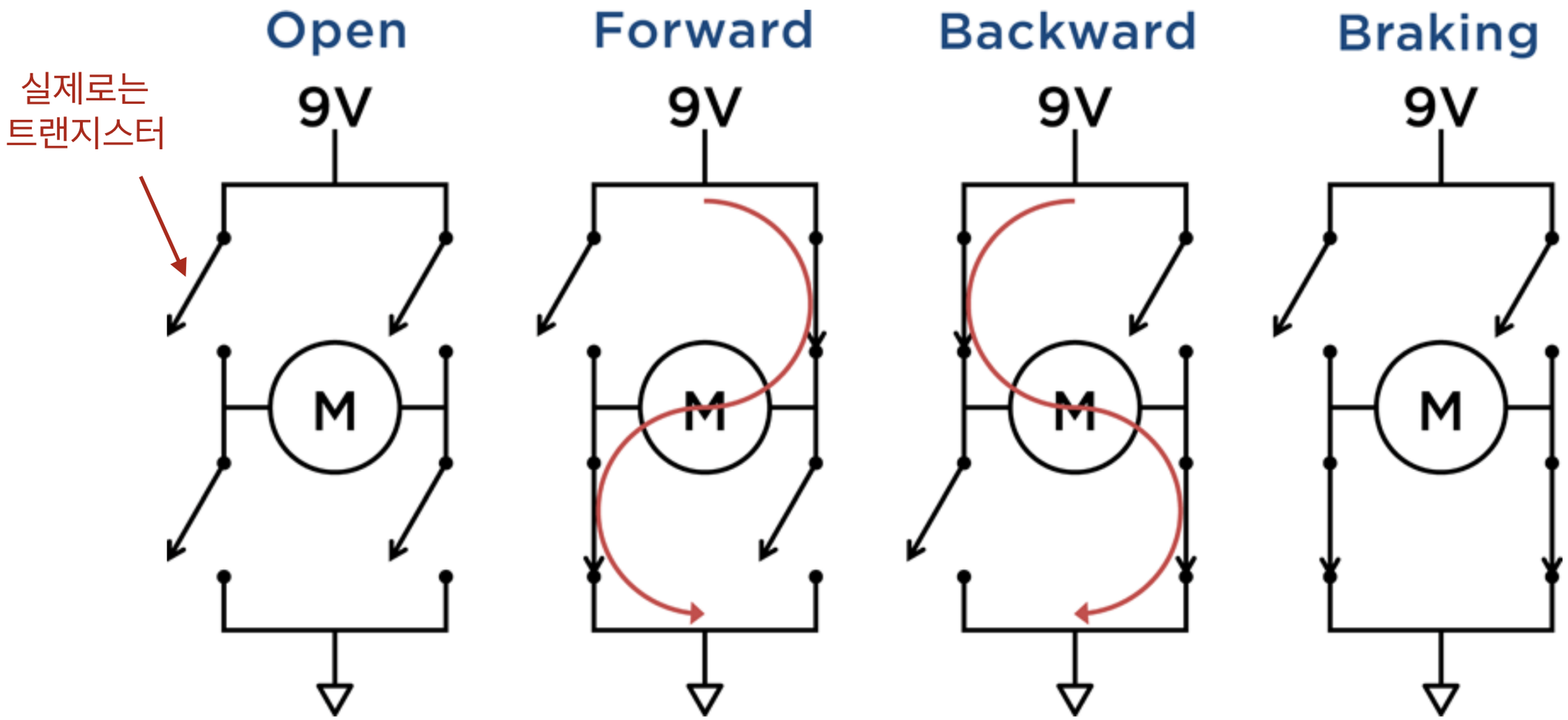
Driving DC Motors

DC Brush Motors



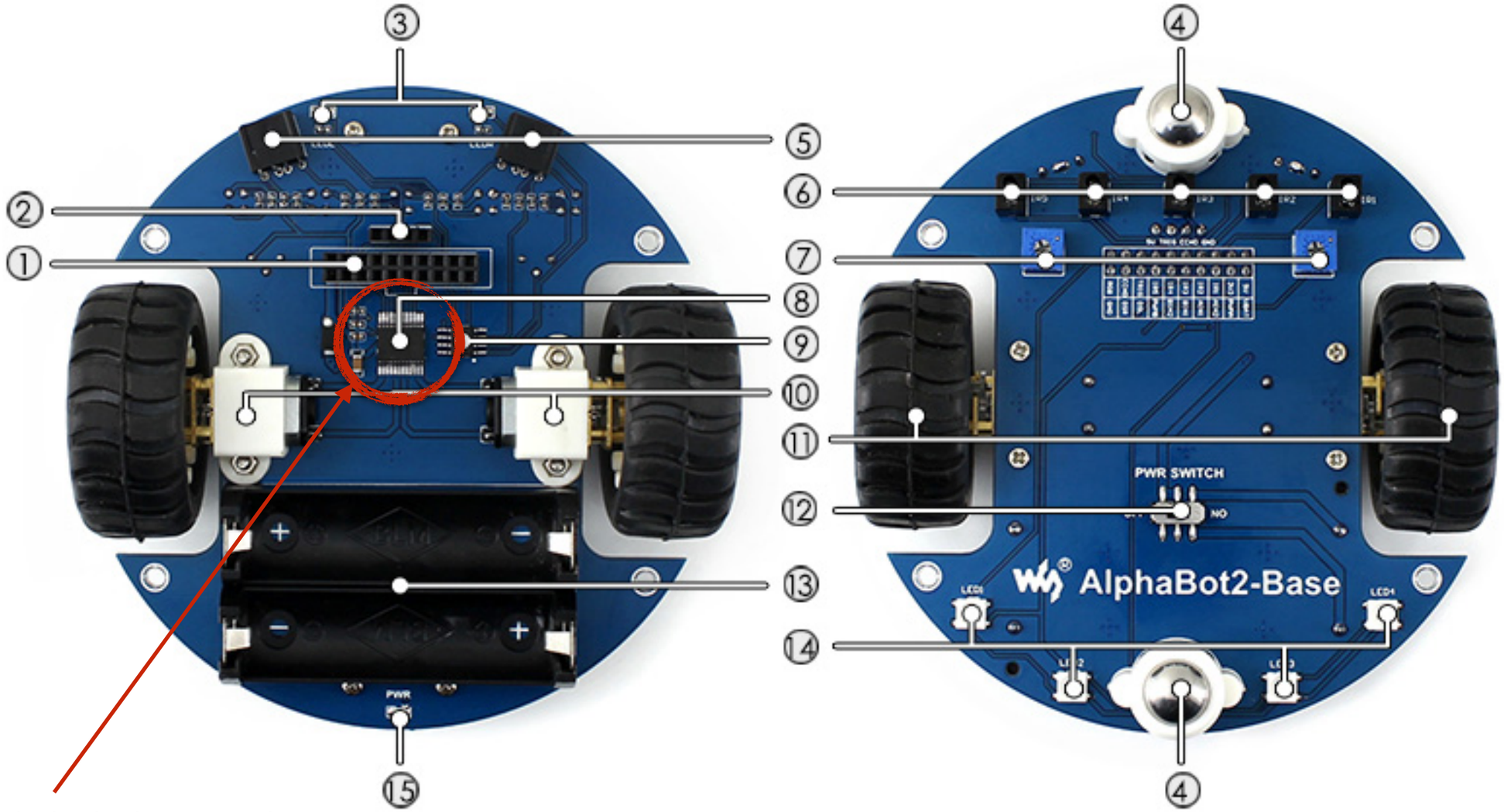
Motors and Selecting the Right One

H-bridge



내부적으로 4개의 스위치를 사용하여 모터의 구동 방향을 제어.
"모터 드라이버"라고 부르기도 함

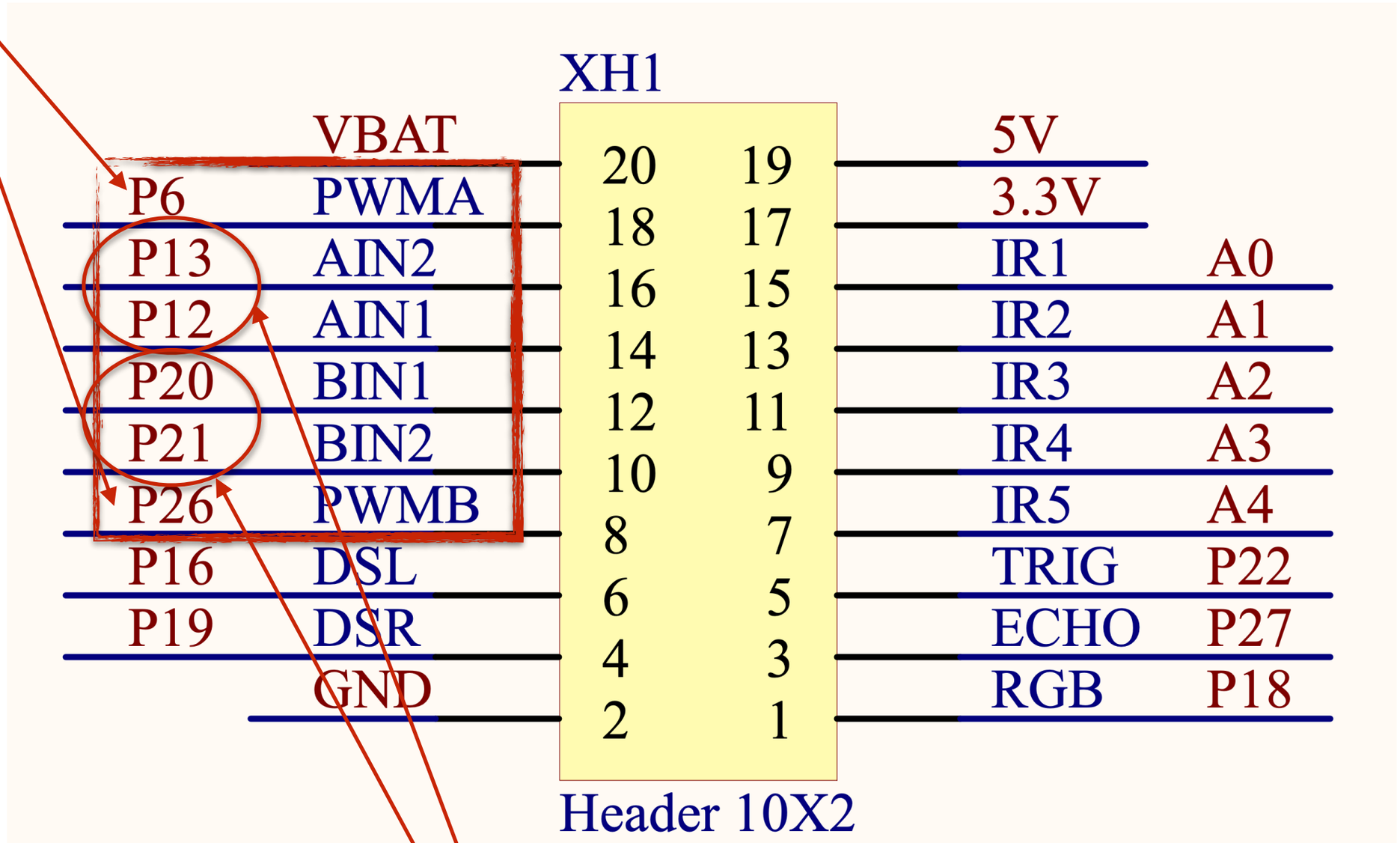
Motor Driver in AlphaBot



TB6612FNG dual H-bridge motor driver

TB6612FNG dual H-bridge motor driver

pwm으로 두 모터의 속도를 조절한다.



각 모터마다 2개의 핀으로 회전 방향을 조절한다.
(0,0) for stop, (0,1) for forward, (1,0) for backward

AlphaBot2.py

왼 쪽 바퀴를 구동하는 모터의 방향 제어(ain1, ain2) 및 pwm 시그널(ena)를 제공할 GPIO 핀 번호

오른 쪽 바퀴를 구동하는 모터의 방향 제어(bin1, bin2)
및 pwm 시그널(enb)를 제공할 GPIO 핀 번호

```
import RPi.GPIO as GPIO
import time

class AlphaBot2(object):

    def __init__(self, ain1=12, ain2=13, ena=6, bin1=20, bin2=21, enb=26):
        self.AIN1 = ain1
        self.AIN2 = ain2
        self.BIN1 = bin1
        self.BIN2 = bin2
        self.ENA = ena
        self.ENB = enb

        self.PA  = 50
        self.PB  = 50
```

양쪽 바퀴의 구동 속도(pwm의 duty cycle). 0에서 100 까지

AlphaBot2.py

RPi.GPIO에서는 2종류의 GPIO 핀 번호 체계를 사용할 수 있다. GPIO.BCM 모드는 MCU 내부에서 사용하는 번호 체계이며, GPIO.BOARD 모드는 RPi의 P1 해더에서의 핀 번호이다.

```
GPIO.setmode(GPIO.BCM)

GPIO.setwarnings(False)

GPIO.setup(self.AIN1, GPIO.OUT)
GPIO.setup(self.AIN2, GPIO.OUT)
GPIO.setup(self.BIN1, GPIO.OUT)
GPIO.setup(self.BIN2, GPIO.OUT)
```

AIN1, AIN2, BIN1, BIN2는 양쪽 바퀴의 회전 방향을 제어한다.

```
GPIO.setup(self.ENA, GPIO.OUT)
GPIO.setup(self.ENB, GPIO.OUT)
```

ENA, ENB는 PWM을 이용하여 양쪽 바퀴의 회전 속도를 제어한다.

```
self.PWMA = GPIO.PWM(self.ENA, 500)
self.PWMB = GPIO.PWM(self.ENB, 500)
```

ENA, ENB를 frequency 500의 PWM 핀으로 설정한다.

```
self.PWMA.start(self.PA)
self.PWMB.start(self.PB)
```

ENA, ENB의 duty cycle을 지정하고 시작한다.

```
self.stop()
```

PWM in RPi.GPIO

AlphaBot2.py

```
def forward(self):
    self.PWMA.ChangeDutyCycle(self.PA)
    self.PWMB.ChangeDutyCycle(self.PB)
    GPIO.output(self.AIN1,GPIO.LOW)
    GPIO.output(self.AIN2,GPIO.HIGH)
    GPIO.output(self.BIN1,GPIO.LOW)
    GPIO.output(self.BIN2,GPIO.HIGH)

def stop(self):
    self.PWMA.ChangeDutyCycle(0)
    GPIO.output(self.AIN1,GPIO.LOW)
    GPIO.output(self.AIN2,GPIO.LOW)
    GPIO.output(self.BIN1,GPIO.LOW)
    GPIO.output(self.BIN2,GPIO.LOW)

def backward(self):
    self.PWMA.ChangeDutyCycle(self.PA)
    self.PWMB.ChangeDutyCycle(self.PB)
    GPIO.output(self.AIN1,GPIO.HIGH)
    GPIO.output(self.AIN2,GPIO.LOW)
    GPIO.output(self.BIN1,GPIO.HIGH)
    GPIO.output(self.BIN2,GPIO.LOW)
```

두 바퀴 모두 전진

두 바퀴 모두 정지

두 바퀴 모두 후진

AlphaBot2.py

```
def left(self):
    self.PWMA.ChangeDutyCycle(30)
    self.PWMB.ChangeDutyCycle(30)
    GPIO.output(self.AIN1, GPIO.HIGH )
    GPIO.output(self.AIN2, GPIO.LOW  )
    GPIO.output(self.BIN1, GPIO.LOW  )
    GPIO.output(self.BIN2, GPIO.HIGH )

def right(self):
    self.PWMA.ChangeDutyCycle(30)
    self.PWMB.ChangeDutyCycle(30)
    GPIO.output(self.AIN1, GPIO.LOW  )
    GPIO.output(self.AIN2, GPIO.HIGH )
    GPIO.output(self.BIN1, GPIO.HIGH )
    GPIO.output(self.BIN2, GPIO.LOW  )

def setPWMA(self,value):
    self.PA = value
    self.PWMA.ChangeDutyCycle(self.PA)

def setPWMB(self,value):
    self.PB = value
    self.PWMB.ChangeDutyCycle(self.PB)
```

양쪽 바퀴의 속도를 30으로 줄인다.

왼쪽 바퀴는 후진, 오른쪽 바퀴는 전진하여 좌회전한다.

왼쪽 바퀴는 전진, 오른쪽 바퀴는 후진하여 우회전한다.

AlphaBot2.py

```
def setMotor(self, left, right):

    if((right >= 0) and (right <= 100)):
        GPIO.output(self.AIN1,GPIO.HIGH)
        GPIO.output(self.AIN2,GPIO.LOW)
        self.PWMA.ChangeDutyCycle(right)
    elif((right < 0) and (right >= -100)):
        GPIO.output(self.AIN1,GPIO.LOW)
        GPIO.output(self.AIN2,GPIO.HIGH)
        self.PWMA.ChangeDutyCycle(0 - right)

    if((left >= 0) and (left <= 100)):
        GPIO.output(self.BIN1,GPIO.HIGH)
        GPIO.output(self.BIN2,GPIO.LOW)
        self.PWMB.ChangeDutyCycle(left)
    elif((left < 0) and (left >= -100)):
        GPIO.output(self.BIN1,GPIO.LOW)
        GPIO.output(self.BIN2,GPIO.HIGH)
        self.PWMB.ChangeDutyCycle(0 - left)
```

left와 right는 양쪽 바퀴의 회전 방향과 속도를 지정한다. 양수이면 후진, 음수이면 전진, 그리고 절대값은 속도를 지정한다.
(좌회전시 left는 음수, right는 양수, 둘 다 양수이면 후진, 둘 다 음수이면 전진, 우회전 시 left는 양수, right는 음수).
좌회전이나 우회전시 left와 right의 절대값이 같으면 제자리 회전

AlphaBot2.py

```
if __name__=='__main__':

    Ab = AlphaBot2()
    Ab.forward()
    try:
        while True:
            time.sleep(1)

    except KeyboardInterrupt:
        GPIO.cleanup()
```

Control-C를 누르면
KeyboardInterrupt가 발생한다.

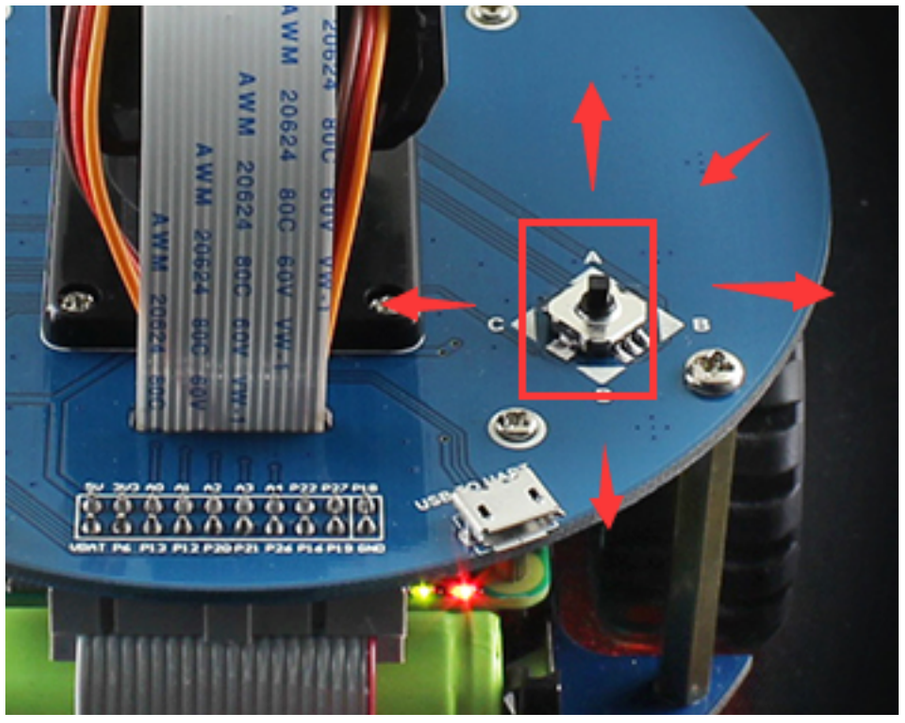
연습

- **Robot을 전후좌우로 다양하게 움직여보자.**
- **Robot이 사각형을 그리며 움직이도록 만들어 보자.**
- **Robot이 지름 1m 정도의 원을 그리면서 움직이도록 만들어 보자.**
- ~~**Robot이 5m이상 거의 완전한 직선으로 움직이도록 만들어 보자.**~~

5-Way Joystick

Joystick

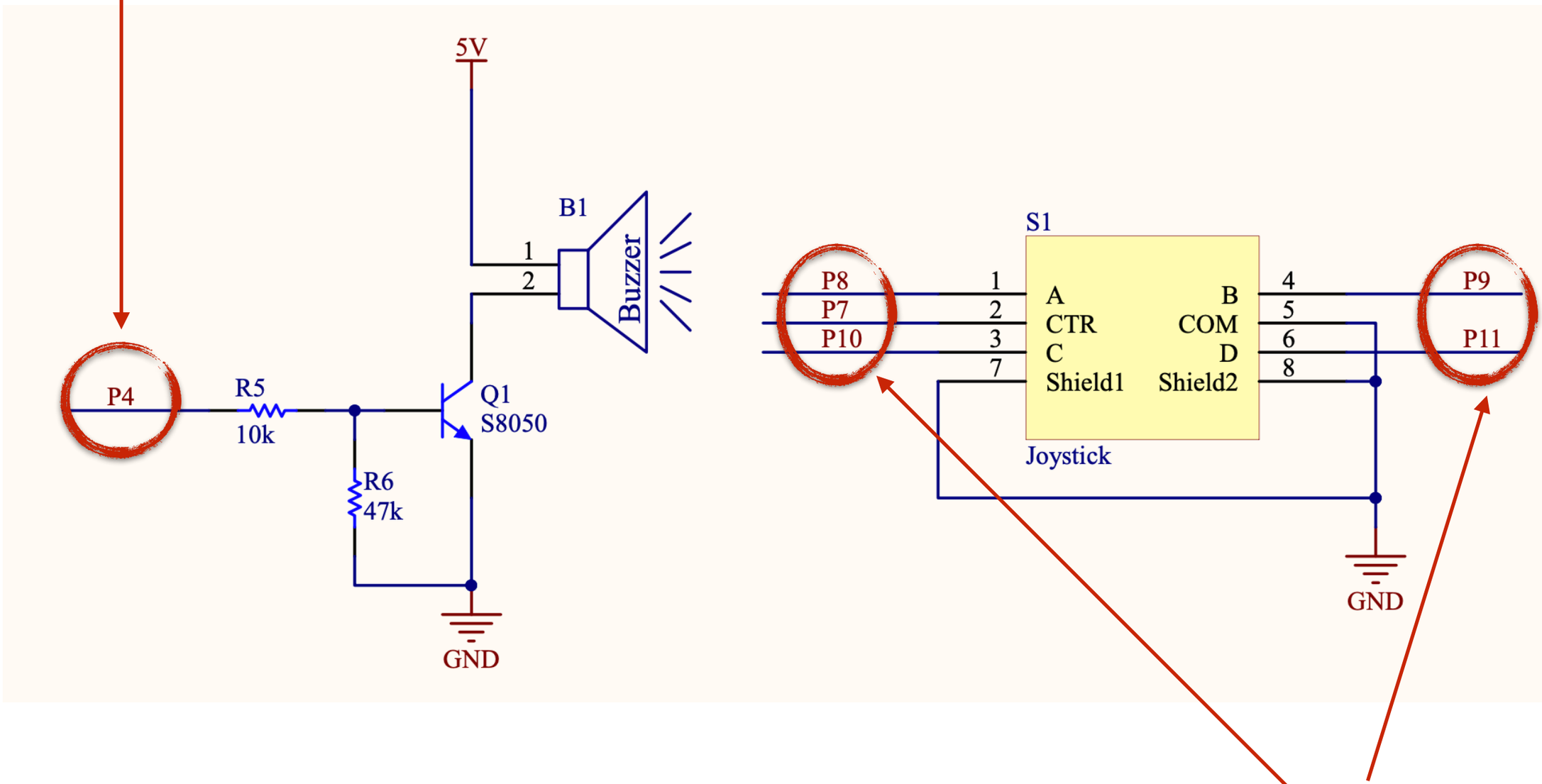
- **Joystick을 누르면 부저가 울리고 모터가 지정한 방향으로 회전한다.** 만약 잘못된 방향으로 회전하면 핀 설정을 변경한다.




```
pi@raspberrypi:~/AlphaBot2/python $ sudo python Joystick.py
up
right
left
down
left
right
center
```

Joystick and Buzzer

GPIO 4번 핀은 AlphaBot에
내장된 buzzer에 연결되어 있다.



GPIO 8~11번 핀은 AlphaBot Joystick의 4방향 버튼에,
7번 핀은 Joystick 중앙의 버튼에 연결되어 있다.

Joystick

```
import RPi.GPIO as GPIO
import time
from AlphaBot2 import AlphaBot2

Ab = AlphaBot2()

CTR = 7
A = 8
B = 9
C = 10
D = 11
BUZ = 4

def beep_on():
    GPIO.output(BUZ, GPIO.HIGH)
def beep_off():
    GPIO.output(BUZ, GPIO.LOW)

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

GPIO.setup(CTR, GPIO.IN, GPIO.PUD_UP)
GPIO.setup(A, GPIO.IN, GPIO.PUD_UP)
GPIO.setup(B, GPIO.IN, GPIO.PUD_UP)
GPIO.setup(C, GPIO.IN, GPIO.PUD_UP)
GPIO.setup(D, GPIO.IN, GPIO.PUD_UP)

GPIO.setup(BUZ, GPIO.OUT)
```

Joystick의 버튼들은 pull-up 모드로 설정한다.
(버튼을 누르면 Low, 떼면 High가 된다.)

Joystick

```
try:
    while True:
        if GPIO.input(CTR) == 0:
            beep_on();
            Ab.stop();
            print("center")
            while GPIO.input(CTR) == 0:
                time.sleep(0.01)
        elif GPIO.input(A) == 0:
            beep_on();
            Ab.forward();
            print("up")
            while GPIO.input(A) == 0:
                time.sleep(0.01)
        elif GPIO.input(B) == 0:
            beep_on();
            Ab.right();
            print("right")
            while GPIO.input(B) == 0:
                time.sleep(0.01)
```

Pull-up 모드이므로 버튼을 누르면 0(Low)가 된다.

Joystick

```
elif GPIO.input(C) == 0:
    beep_on();
    Ab.left();
    print("left")
    while GPIO.input(C) == 0:
        time.sleep(0.01)
elif GPIO.input(D) == 0:
    beep_on();
    Ab.backward();
    print("down")
    while GPIO.input(D) == 0:
        time.sleep(0.01)
    else:
        beep_off();

except KeyboardInterrupt:
    GPIO.cleanup()
```

AlphaBot을 그냥 손에 들고
코드를 실행한 후 Joystick으로 조종해보자.

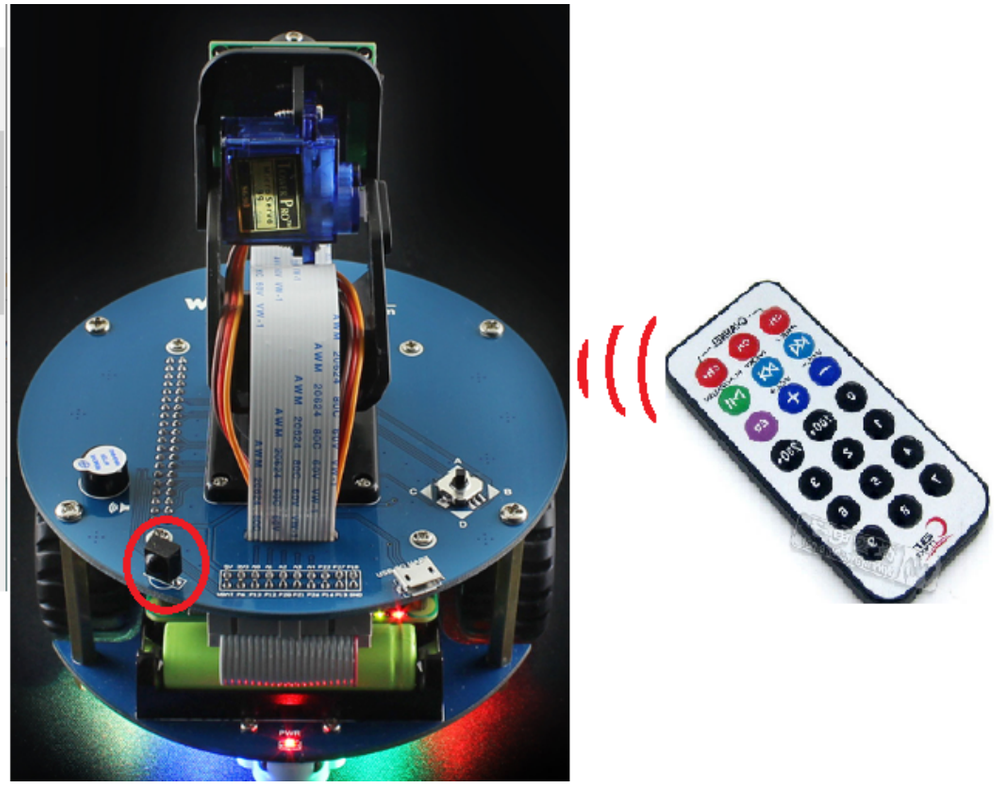
```
(.venv) pi@raspberrypi:~/AlphaBot2/python $ python Joystick.py
[Ctrl-C to stop]
```

Infrared Remote Control

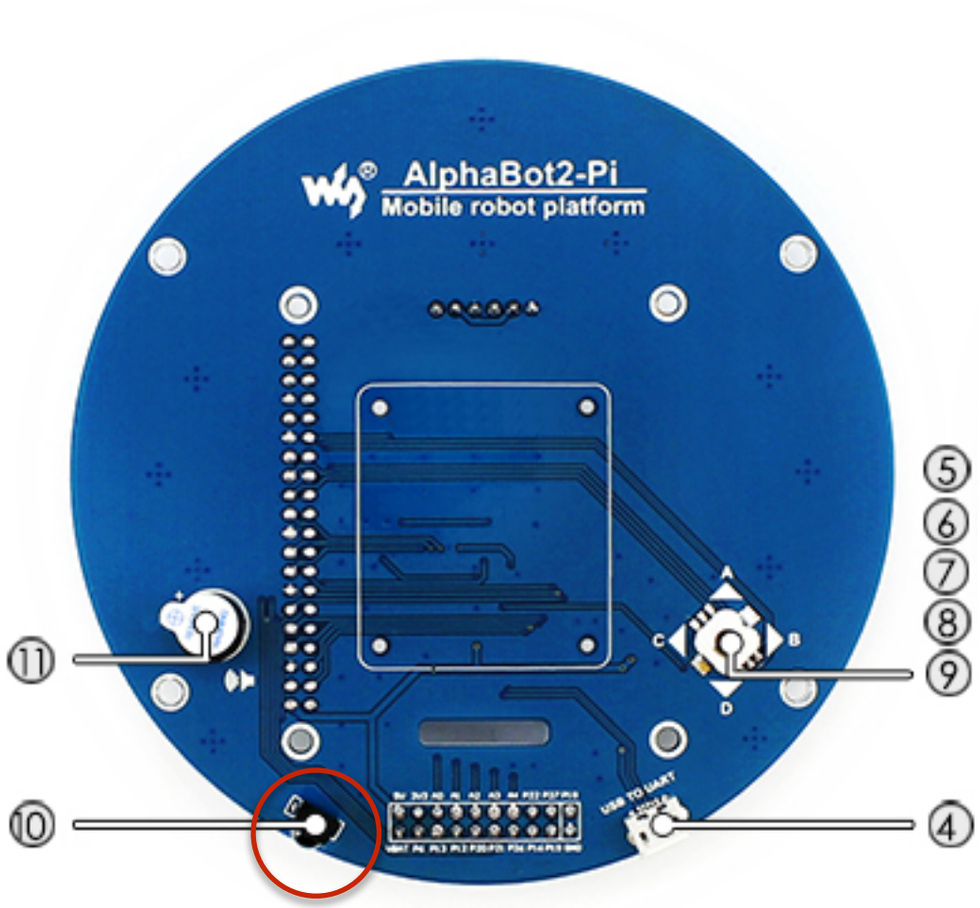
Infrared Remote Control

- 제공된 리모콘으로 **Alphabot**을 조종할 수 있다.
  - 2: go forwards
  - 8: go backwards
  - 4: turn left
  - 6: turn right
  - 5: stop
  - - and +: adjust the speed
  - EQ: restore the default setting

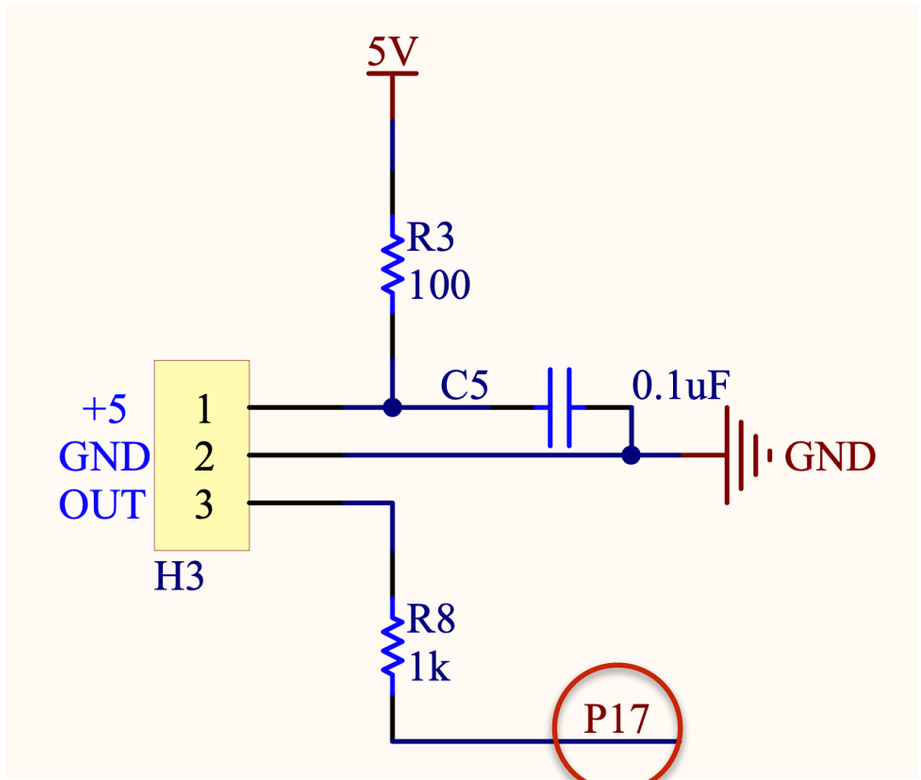
```
pi@raspberrypi: ~/AlphaBot2/python
pi@raspberrypi:~/AlphaBot2/python $ cd ~/AlphaBot2/python
pi@raspberrypi:~/AlphaBot2/python $ sudo python IRremote.py
IRremote Test Start ...
left
repeat
left
repeat
repeat
repeat
repeat
left
repeat
repeat
left
repeat
repeat
right
right
repeat
repeat
repeat
repeat
```



IR Receiver



IR receiver



IR receiver connected to P17

Infrared Remote Control

```
import RPi.GPIO as GPIO
import time
from AlphaBot2 import AlphaBot2

Ab = AlphaBot2()

IR = 17

PWM = 50
n=0

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

GPIO.setup(IR, GPIO.IN)
```

IR Receiver는 GPIO 17번 핀에 연결

GPIO 17번 핀을 입력 모드로 설정

Infrared Remote Control

```
def getkey():
    if GPIO.input(IR) == 0:
        count = 0
        while GPIO.input(IR) == 0 and count < 200:  #9ms
            count += 1
            time.sleep(0.00006)
    if(count < 10):
        return;
    count = 0
    while GPIO.input(IR) == 1 and count < 80:  #4.5ms
        count += 1
        time.sleep(0.00006)
```

리모콘과의 통신 프로토콜에 따라서 리모콘에서 눌려진 키를 전송받는다.

Infrared Remote Control

```
idx = 0
cnt = 0
data = [0,0,0,0]
for i in range(0,32):
    count = 0
    while GPIO.input(IR) == 0 and count < 15:     #0.56ms
        count += 1
        time.sleep(0.00006)
count = 0
while GPIO.input(IR) == 1 and count < 40:    #0: 0.56mx
    count += 1                                #1: 1.69ms
    time.sleep(0.00006)

if count > 7:
    data[idx] |= 1<<cnt
if cnt == 7:
    cnt = 0
    idx += 1
else:
    cnt += 1
```

```
#         print data
        if data[0]+data[1] == 0xFF and data[2]+data[3] == 0xFF:  #check
            return data[2]
        else:
            print("repeat")
            return "repeat"
```

Infrared Remote Control

```
print('IRremote Test Start ...')
Ab.stop()
try:
    while True:
        key = getkey()
        if(key != None):
            n = 0
            if key == 0x18:
                Ab.forward()
                print("forward")
            if key == 0x08:
                Ab.left()
                print("left")
            if key == 0x1c:
                Ab.stop()
                print("stop")
            if key == 0x5a:
                Ab.right()
                print("right")
            if key == 0x52:
                Ab.backward()
                print("backward")
```

Infrared Remote Control

```
if key == 0x15:
    if(PWM + 10 < 101):
        PWM = PWM + 10
        Ab.setPWMA(PWM)
        Ab.setPWMB(PWM)
        print(PWM)
if key == 0x07:
    if(PWM - 10 > -1):
        PWM = PWM - 10
        Ab.setPWMA(PWM)
        Ab.setPWMB(PWM)
        print(PWM)
    else:
        n += 1
        if n > 20000:
            n = 0
            Ab.stop()
except KeyboardInterrupt:
    GPIO.cleanup();
```

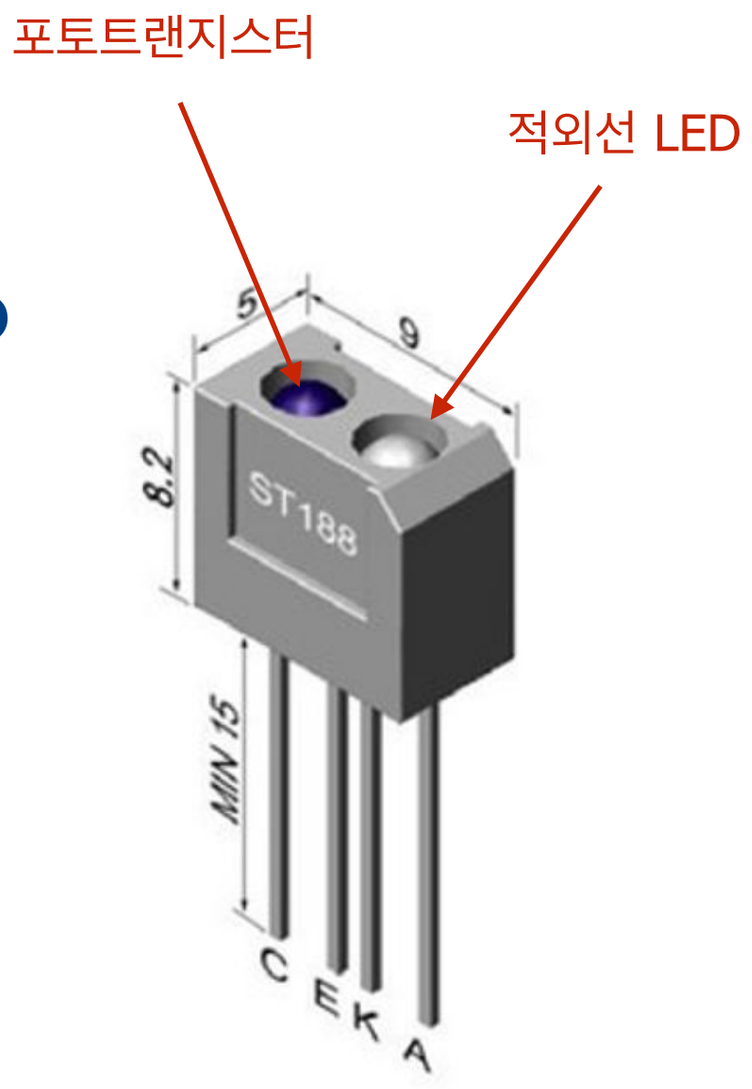
코드를 실행한 후 리모컨으로 조종해보자.

```
(.venv) pi@raspberrypi:~/AlphaBot2/python $ sudo python IRremote.py
[Ctrl-C to stop]
```

Infrared Obstacle Avoidance

ST188 적외선 센서

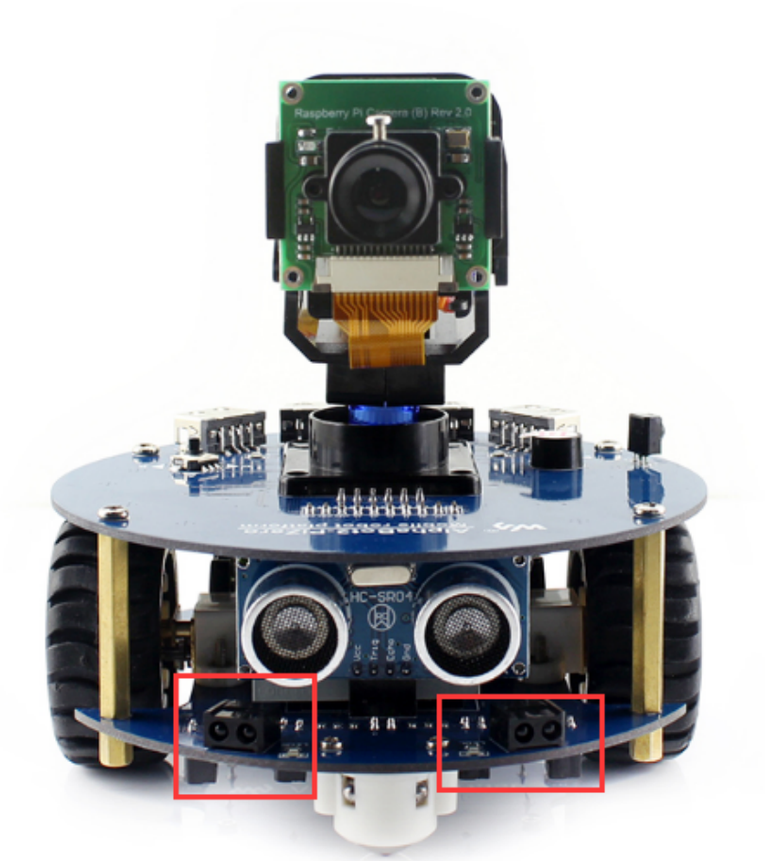
- 적외선 장애물 감지 센서는 적외선을 보내는 적외선 **LED** 와 물체에 닫아서 반사된 적외선을 감지하는 포토트랜지스터로 구성



• Combines high output GaAs IRED with high sensitive phototransistor.
• Wide detecting range: 4~13mm.
• Non-contact detecting manner

Infrared Obstacle Avoidance

- **Alphabot**에는 전방에 **2**개의 **ST188** 적외선 센서가 장착되어 전방의 장애물을 감지
- **ST188** 자체는 아날로그 센서이지만 **Voltage comparator**를 이용하여 **digital** 출력을 제공
- 로봇 하부의 가변저항**(potentiometer)**으로 센서의 감도를 조절



2개의 적외선 센서 ST188

Infrared Reflective Sensor with Voltage Comparator

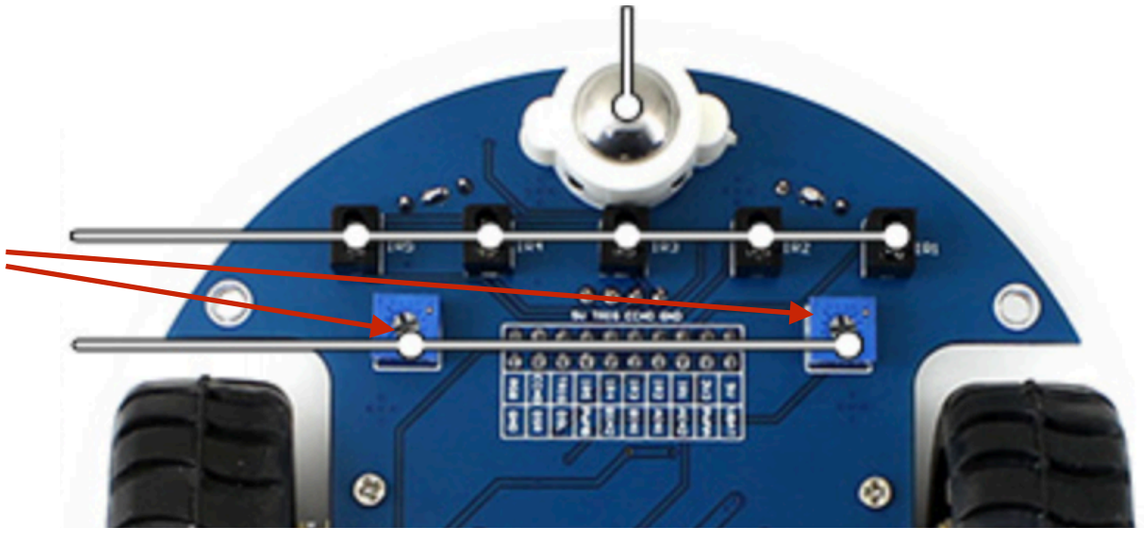
Sensor	ST188
Voltage comparator chip	LM393
Operating voltage	3.0V-5.3V
Dimensions	30.2mm*11.9mm
Fixing hole size	2.0mm

Pin No.	Symbol	Descriptions
1	DOUT	Digital output
2	AOUT	Analog output
3	GND	Power ground
4	VCC	Positive power supply (3.0V-5.3V)

Infrared Obstacle Avoidance

- 전방에 장애물이 없으면 녹색 **LED**가 꺼지고, 장애물이 감지되면 녹색 **LED**가 켜진다.
- 센서에 장애물이 감지되지 않으면 전진하고, 장애물이 감지되면 좌회전하도록 만들어보자.
- **LED**가 항상 켜져 있거나 혹은 꺼져 있으면 로봇 하부의 가변저항(**potentiometer**)을 조절하라.

적외선 센서 ST188의 감도조절을 위한 가변저항



Obstacle Avoidance

```
import RPi.GPIO as GPIO
import time
from AlphaBot2 import AlphaBot2

Ab = AlphaBot2()
```

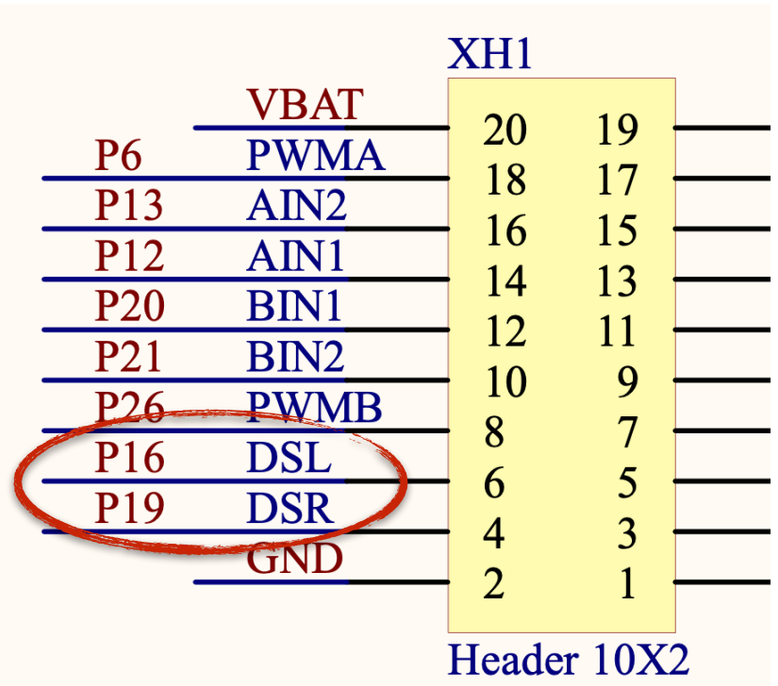
```
DR = 16
DL = 19
```

← 양쪽 IR Sensor는 각각 GPIO 16, 19번 핀에 연결

```
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
```

```
GPIO.setup(DR, GPIO.IN, GPIO.PUD_UP)
GPIO.setup(DL, GPIO.IN, GPIO.PUD_UP)
```

↑ GPIO 16, 19번 핀을 PULL-UP된 input 모드로 설정



Obstacle Avoidance

```
try:
    while True:
        DR_status = GPIO.input(DR)
        DL_status = GPIO.input(DL)
#       print(DR_status,DL_status)
        if((DL_status == 0) or (DR_status == 0)):
            Ab.left()
            #Ab.right()
            time.sleep(0.002)
            Ab.stop()
        #   print("object")
        else:
            Ab.forward()
        #   print("forward")

except KeyboardInterrupt:
    GPIO.cleanup();
```

어느 쪽이든 장애물이 감지되면 (pull-up)

어느 쪽에도 장애물이 감지되지 않으면

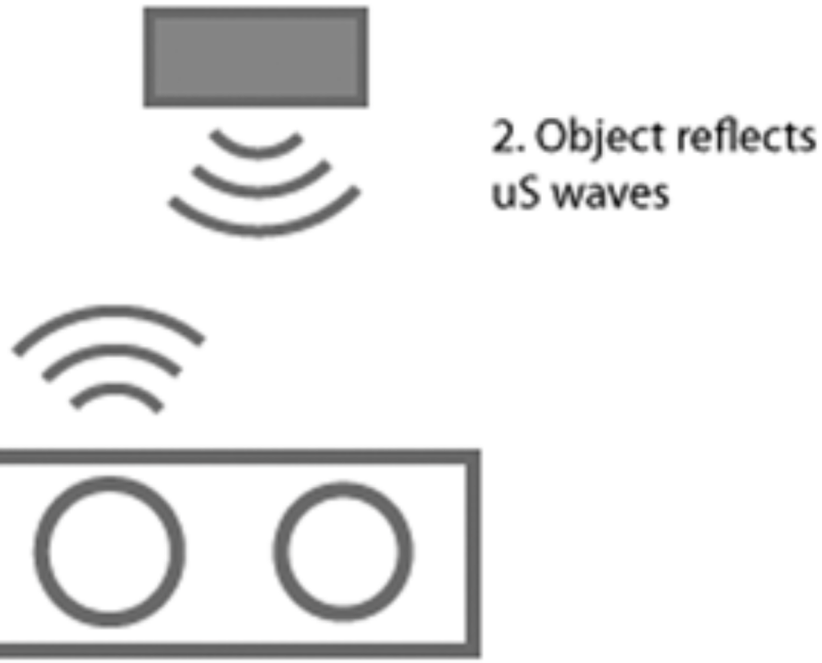
```
(.venv) pi@raspberrypi:~/AlphaBot2/python $ sudo python Infrared_Obstacle_Avoidance.py
[Ctrl-C to stop]
```

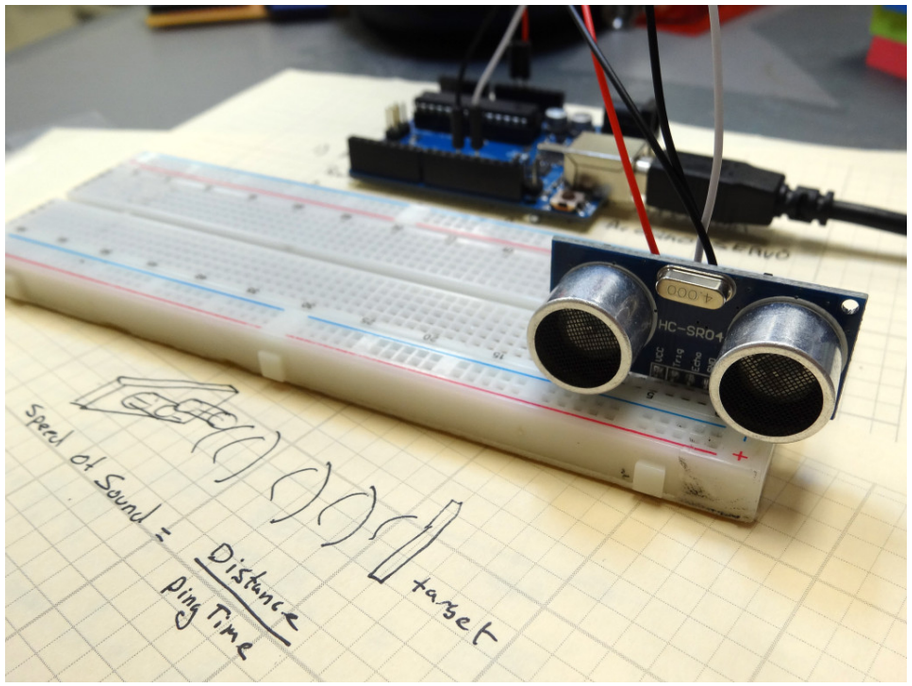
실습

- 장애물이 감지되면 무조건 왼쪽으로 회전하는 대신 다른 방법으로 움직이도록 만들어 보고 어떤 장단점이 있는지 생각해보자.
- 후진을 해서 다른 경로를 찾는 방법에 대해서도 생각해보자.

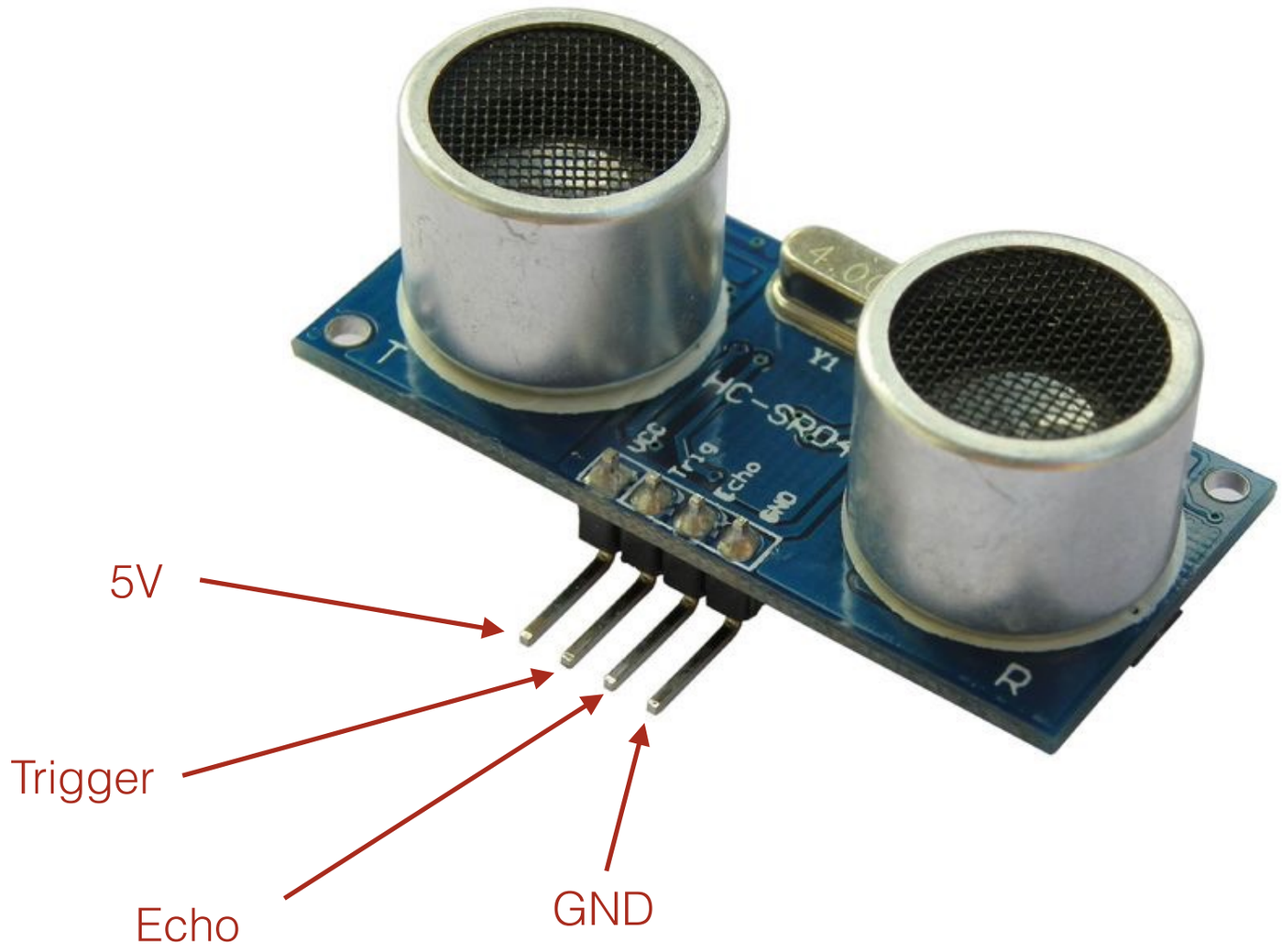
Ultrasonic Distance Measurement

초음파센서(ultrasonic sensor)





초음파센서(ultrasonic sensor)

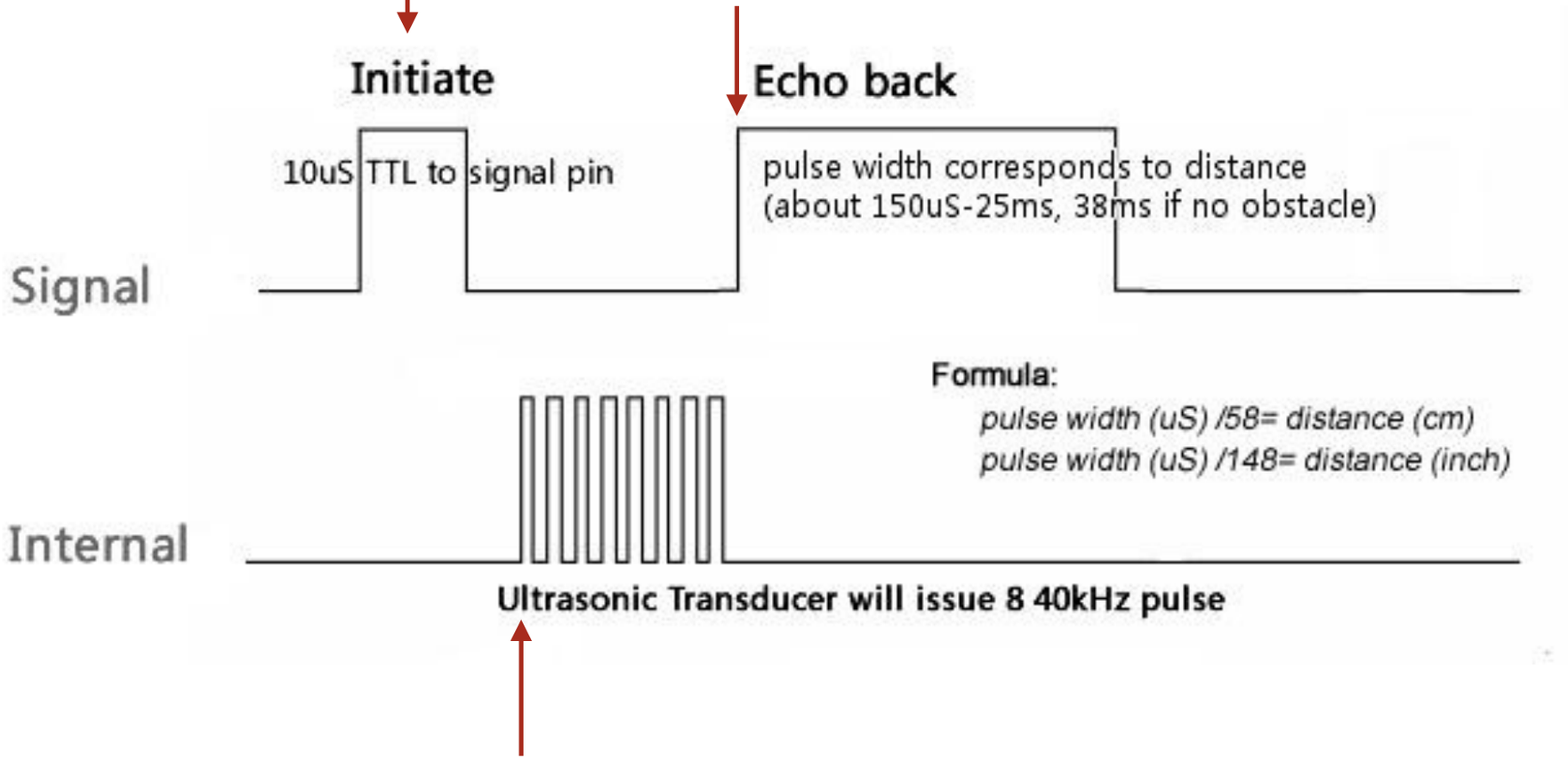


센서에 따라서 trigger와 echo가
하나의 핀을 공유하는 경우도 있음

초음파센서(ultrasonic sensor)

1. Trigger 핀으로 10μs width의 trigger 펄스를 준다.

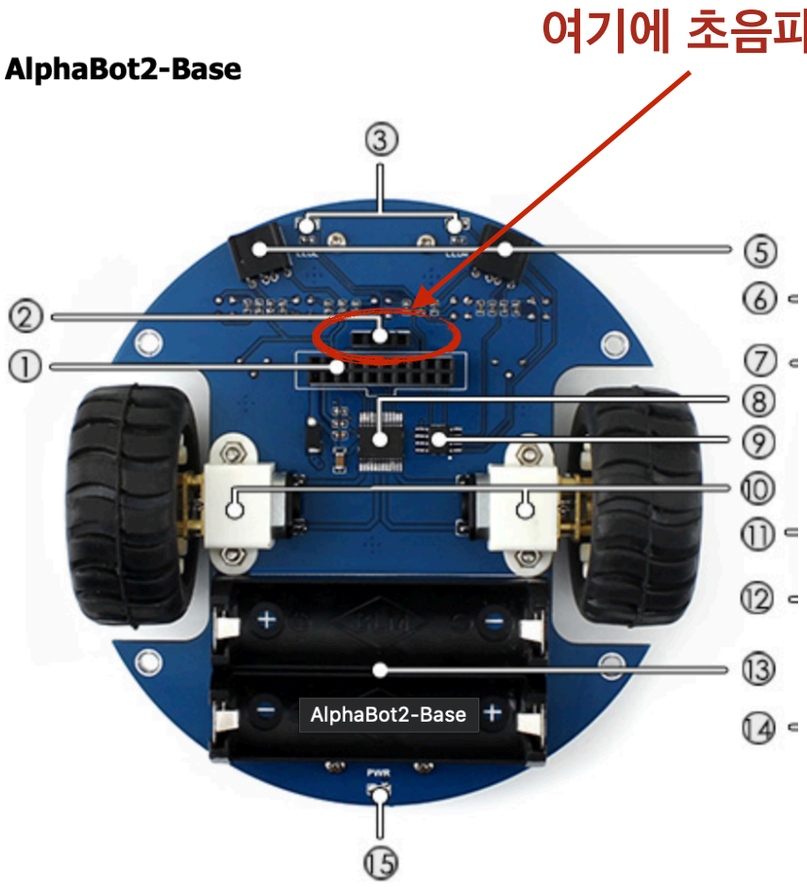
3. echo를 탐지한 후 소모된 시간에 비례하는 길이의 pulse를 발생한다. 이 펄스의 길이를 pulseIn 함수로 측정한다.

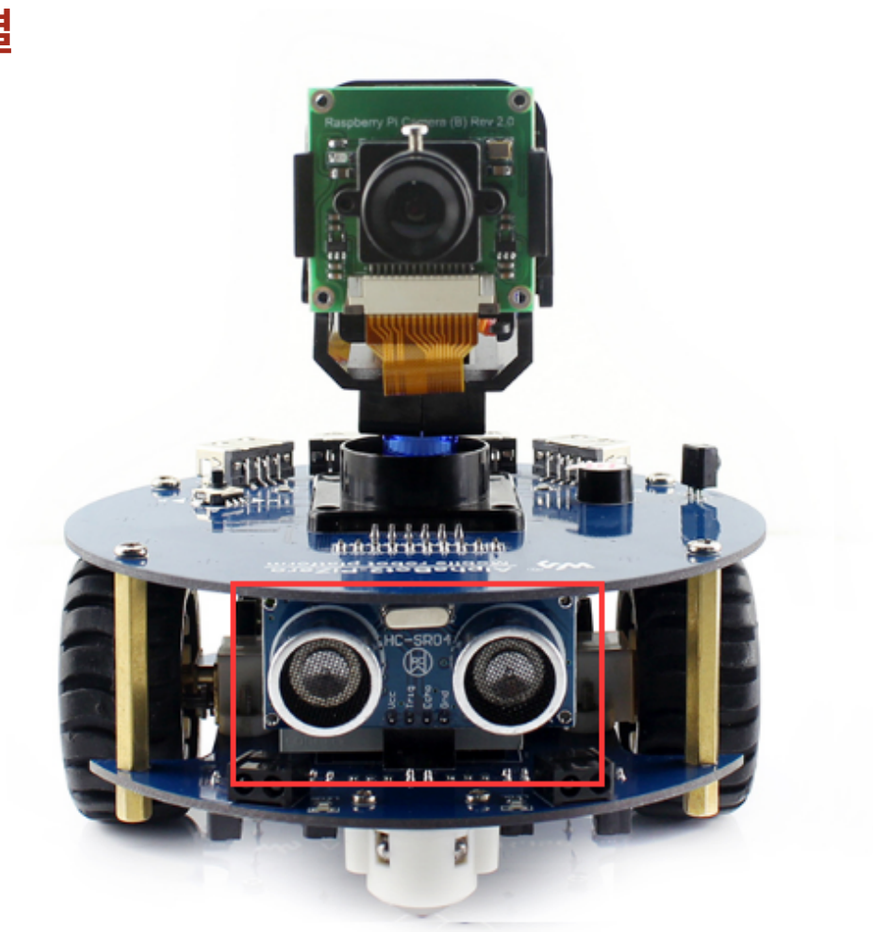


2. 8번의 40kHz의 펄스를 생성한다.

초음파 센서로 거리 측정

- 나사를 풀어 상판을 분리하고 초음파 센서를 연결한다 (상판이 고정되지 않음).





초음파 센서로 거리 측정

Ultrasonic_Ranging.py

```
import RPi.GPIO as GPIO
import time

TRIG = 22
ECHO = 27

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

GPIO.setup(TRIG, GPIO.OUT, initial=GPIO.LOW )

GPIO.setup(ECHO, GPIO.IN)
```

GPIO 출력 핀의 초기 상태를 이렇게 지정한다.

초음파 센서로 거리 측정

Ultrasonic_Ranging.py

```
def dist():
    GPIO.output(TRIG, GPIO.HIGH)
    time.sleep(0.000015)
    GPIO.output(TRIG,GPIO.LOW)
```

← 15μs width의 trigger 펄스를 준다.

```
    while not GPIO.input(ECHO):
        pass
    t1 = time.time()
    while GPIO.input(ECHO):
        pass
    t2 = time.time()
```

← ECHO 핀으로 입력되는 pulse의 width를 측정한다.

```
    return (t2-t1)*34000/2
```

← 시간을 거리(cm)로 환산한다.

```
try:
    while True:
        print "Distance:%0.2f cm" % dist()
        time.sleep(1)
except KeyboardInterrupt:
    GPIO.cleanup()
```

초음파 센서를 이용하여 장애물 회피하기

Ultrasonic_Obstacle_Avoidance.py

```
import RPi.GPIO as GPIO
import time
from AlphaBot2 import AlphaBot2

TRIG = 22
ECHO = 27

Ab = AlphaBot2()
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(TRIG,GPIO.OUT,initial=GPIO.LOW)
GPIO.setup(ECHO,GPIO.IN)

def Distance():
    GPIO.output(TRIG,GPIO.HIGH)
    time.sleep(0.000015)
    GPIO.output(TRIG,GPIO.LOW)
    while not GPIO.input(ECHO):
        pass
    t1 = time.time()
    while GPIO.input(ECHO):
        pass
    t2 = time.time()
    return (t2-t1)*34000/2
```

초음파 센서를 이용하여 장애물 회피하기

Ultrasonic_Obstacle_Avoidance.py

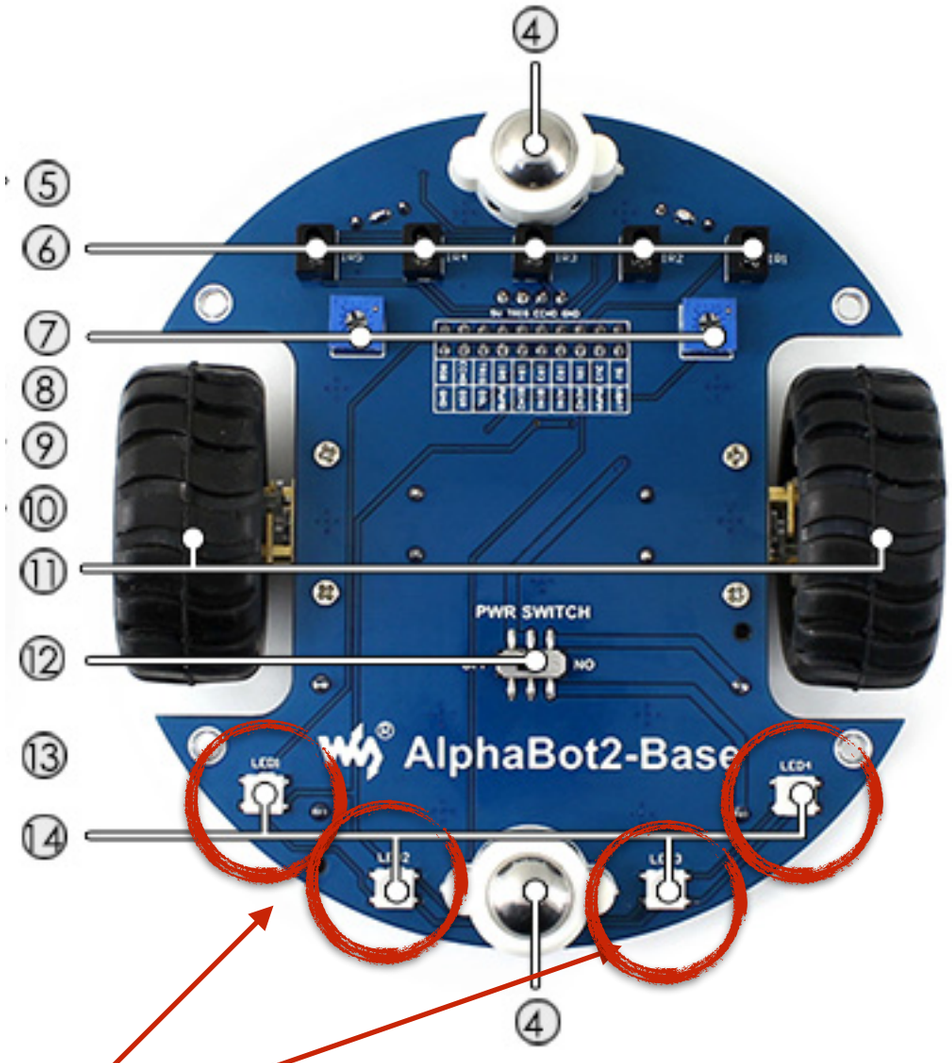
```
print("Ultrasonic_Obstacle_Avoidance")
try:
    while True:
        Dist = Distance()
        print("Distance = %0.2f cm"%Dist)
        if Dist <= 20:
            Ab.right()
#           Ab.left()
        else:
            Ab.forward()
        time.sleep(0.02)

except KeyboardInterrupt:
    GPIO.cleanup();
```

장애물과의 거리가 20cm 이내이면 우회전한다.

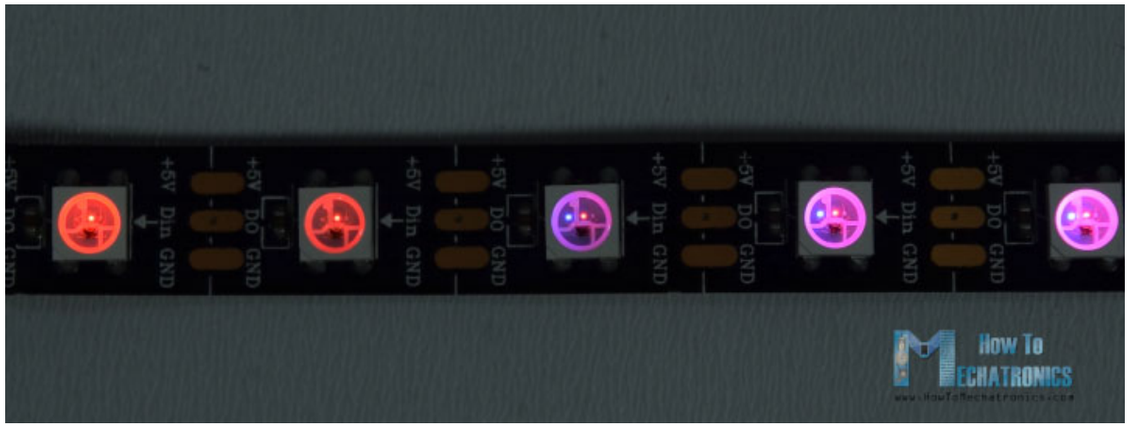
RGB LED

WS2812B RGB LEDs

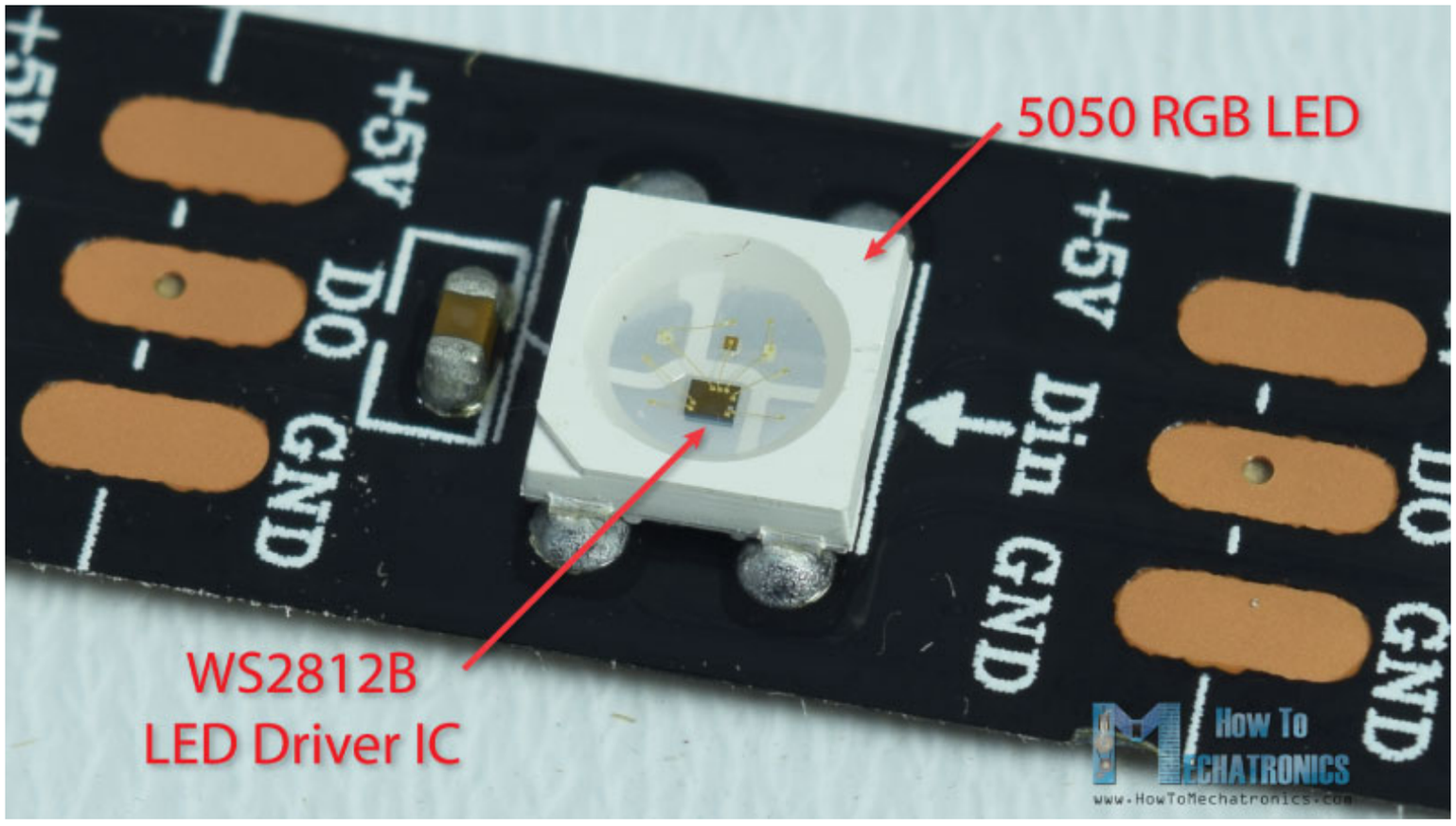


WS2812B RGB LEDs

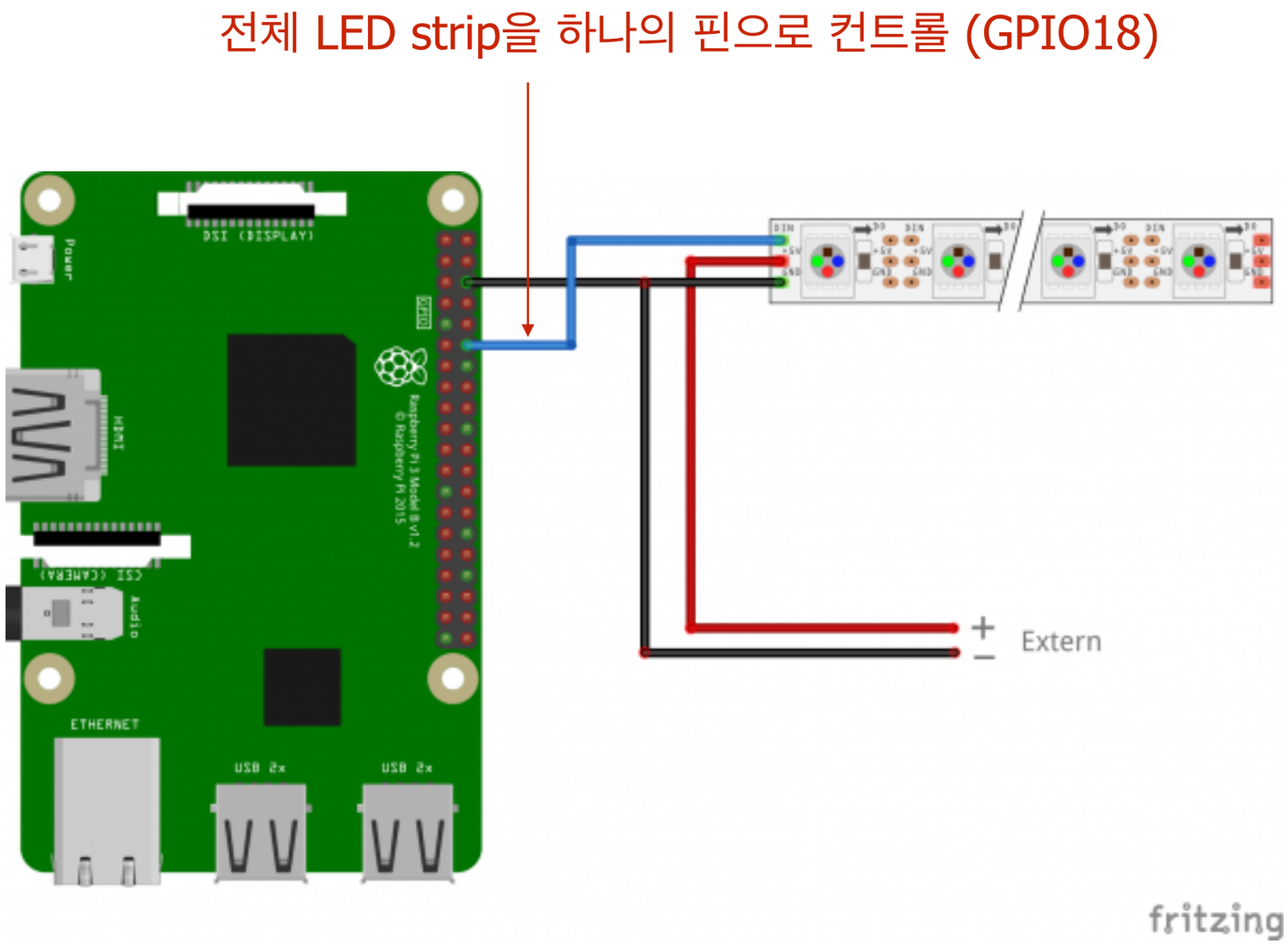
WS2812B RGB LEDs



LED strip



WS2812B RGB LEDs



WS281x 라이브러리 설치

```
(.venv) pi@raspberrypi:~ $ pwd
/home/pi
(.venv) pi@raspberrypi:~ $ wget https://www.waveshare.com/w/upload/c/c3/Rpi_ws281x-
master.zip
(.venv) pi@raspberrypi:~ $ unzip Rpi_ws281x-master.zip
(.venv) pi@raspberrypi:~ $ sudo apt install build-essential python-dev scons swig -y
(.venv) pi@raspberrypi:~ $ cd ~/rpi_ws281x-master
(.venv) pi@raspberrypi:~/rpi_ws281x-master $ sudo scons
...
(.venv) pi@raspberrypi:~/rpi_ws281x-master $ sudo ./test
...
[Ctrl-C to stop program]
(.venv) pi@raspberrypi:~/rpi_ws281x-master $ cd python
(.venv) pi@raspberrypi:~/rpi_ws281x-master/python $ sudo python setup.py build
(.venv) pi@raspberrypi:~/rpi_ws281x-master/python $ sudo python setup.py install
(.venv) pi@raspberrypi:~/rpi_ws281x-master/python $ cd examples/
(.venv) pi@raspberrypi:~/rpi_ws281x-master/python/examples $ sudo python lowlevel.py
...
[Ctrl-C to stop program]
```

ws2812.py

```
(.venv) pi@raspberrypi:~/AlphaBot2/python $ pwd
/home/pi/AlphaBot2/python
(.venv) pi@raspberrypi:~/AlphaBot2/python $ sudo python ws2812.py
[Ctrl-C to stop program]
```

ws2812.py

```
import time
from neopixel import *

# LED strip configuration:
LED_COUNT      = 4       # Number of LED pixels.
LED_PIN        = 18      # GPIO pin connected to the pixels (must support PWM!)
LED_FREQ_HZ    = 800000  # LED signal frequency in hertz (usually 800khz)
LED_DMA        = 5       # DMA channel to use for generating signal (try 5)
LED_BRIGHTNESS = 255     # Set to 0 for darkest and 255 for brightest
LED_INVERT     = False   # True to invert the signal (when using NPN transistor
level shift)
LED_CHANNEL    = 0       # set to '1' for GPIOs 13, 19, 41, 45 or 53
```

ws2812.py

```
# Create NeoPixel object with appropriate configuration.
strip = Adafruit_NeoPixel(LED_COUNT, LED_PIN, LED_FREQ_HZ, LED_DMA, LED_INVERT,
LED_BRIGHTNESS)

# Intialize the library (must be called once before other functions).
strip.begin()

strip.setPixelColor(0, Color(255, 0, 0))       #Red
strip.setPixelColor(1, Color(0, 255, 0))       #Green
strip.setPixelColor(2, Color(0, 0, 255))       #Blue
strip.setPixelColor(3, Color(255, 255, 0))     #Yellow
strip.show()

time.sleep(2)
strip.setPixelColor(0, Color(0, 0, 0))       #Red
strip.setPixelColor(1, Color(0, 0, 0))       #Green
strip.setPixelColor(2, Color(0, 0, 0))       #Blue
strip.setPixelColor(3, Color(0, 0, 0))       #Yellow
strip.show()
```

Test other examples in
"~/rpi_ws281x-master/python/examples"

Putting It All Together

장애물 감지 센서를 이용하여...

- 지금까지 다룬 모든 기능들을 함께 사용하여 강의실의 앞에서 뒤까지 장애물을 피해가면서 이동하도록 만들어 보자.