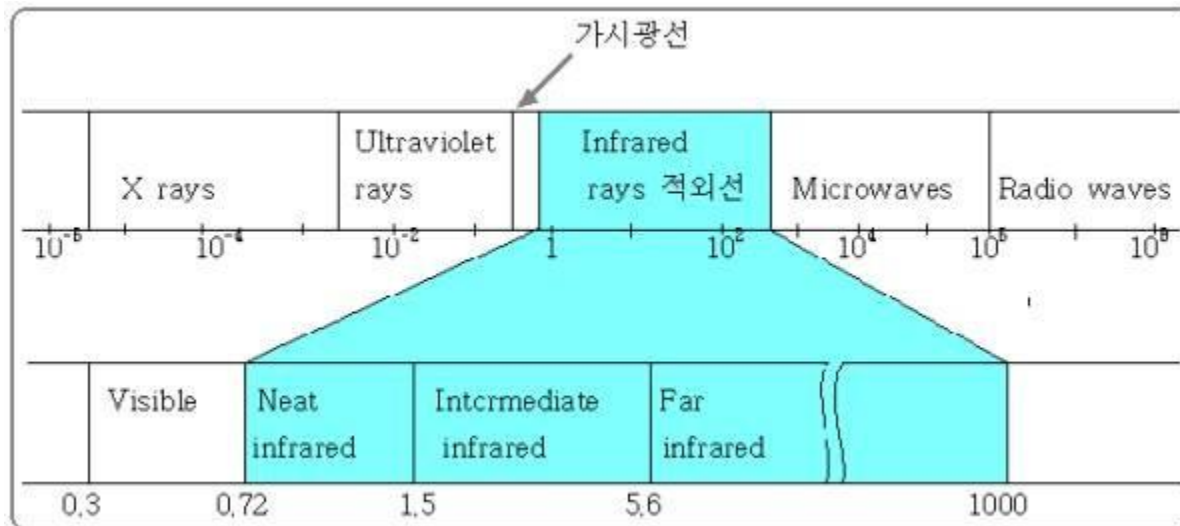


CH. 6. 라즈베리파이 주변 제어

Section 01 적외선 센서

□ 적외선이란?

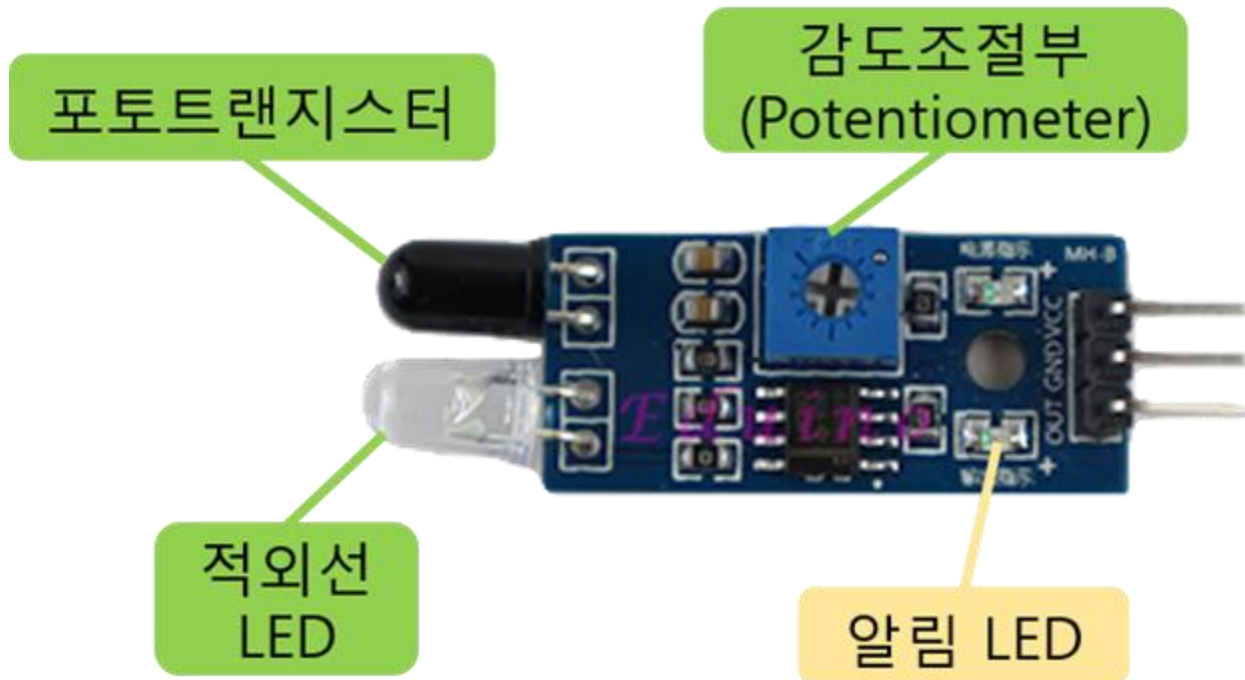
- 적외선은 가시광선이 빨,주,노,초,파,남,보로 프리즘을 통해 나타날 때, 빨강색 바깥쪽에 나타난다고 해서 적외선이라고 부르게 되었다.
- 태양이나 물체가 내는 복사열의 대부분은 이 적외선으로 이루어져있으므로 적외선을 열선이라고 한다.
- 적외선의 파장범위는 가시광선의 장파장 끝의 $0.76\sim 0.8\mu\text{m}$ 를 하단으로 하고, 상단은 1mm정도까지이다.



Section 01 적외선 센서

□ 적외선센서 사용방법

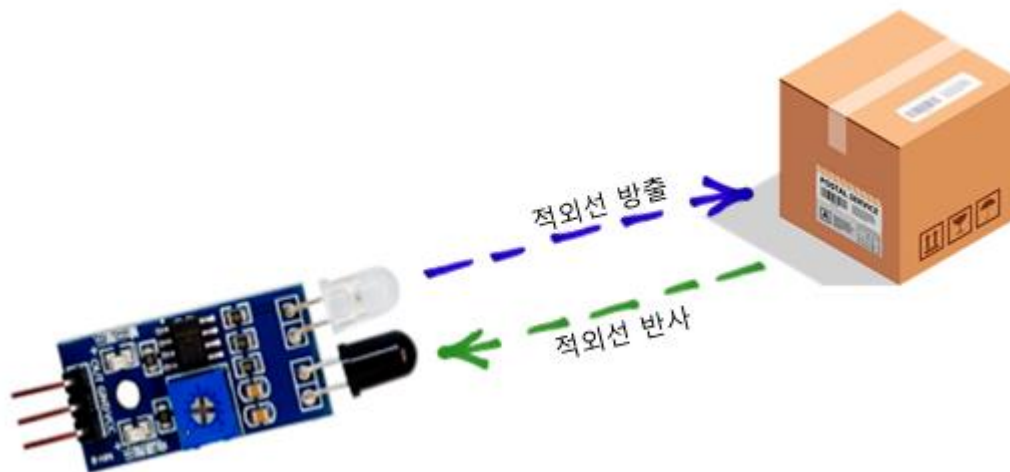
- 적외선도 빛의 한 종류이기 때문에 빛의 성질을 가지고 있다.
- 적외선은 직진하며 물체에 닿으면 반사하는 성질을 가지고 있다.
- 적외선 장애물 감지 센서는 적외선을 보내는 부분과 반사된 적외선을 감지하는 감지 부분으로 구성되어 있다.



Section 01 적외선 센서

□ 적외선센서 사용방법

- 적외선 LED에서 적외선을 보내게 되고, 물체에 닿아 반사되는 빛은 포토 트랜지스터에서 감지하게 된다.
- 적외선 장애물 감지 센서는 앞에 장애물이 놓이게 되면 센서에서 보낸 적외선이 장애물에 반사되어 수신부에 들어가게 되고, 이를 통해 센서가 장애물을 인식하게 된다.
- 10cm부근에서 최대의 전압값을 갖다가 거리가 멀어질수록 다시 전압이 감소하는 형태를 띄고 있다.



Section 01 적외선 센서

□ 적외선센서 수광부

- 적외선 센서의 수광부의 역할은 적외선을 인식하는 것이다.
- 우선 적외선을 인식하기 위해서는 포토트랜지스터(Phototransister)라는 것이 필요하다.
- 포토트랜지스터는 적외선을 인식하면 전류를 흐르게 하는 특징을 가지고 있다.
- 그러니까 적외선 빛이 베이스(Base)가 되고 긴다리가 콜렉터(Collector), 짧은 다리가 에미터(Emitter)가 되는 트랜지스터이다.



Section 01 적외선 센서

□ 파이썬 코드

- 다음과 같은 코드를 작성하여 적외선 센서의 측정의 값을 입력받아 보자.

```
1  from ctypes import *
2  import os
3  import time
4
5  WiringPi = CDLL("/home/pi/WiringPi/wiringPi/libwiringPi.so.2.70", mode=RTLD_G
  LOBAL)
6  swcar = cdll.LoadLibrary('/home/pi/swcar/libswcar.so')
7
8  swcar.SIO_Init(0)
9
10 print('press ctrl + c to terminate program')
11
```

Section 01 적외선 센서

□ 파이썬 코드

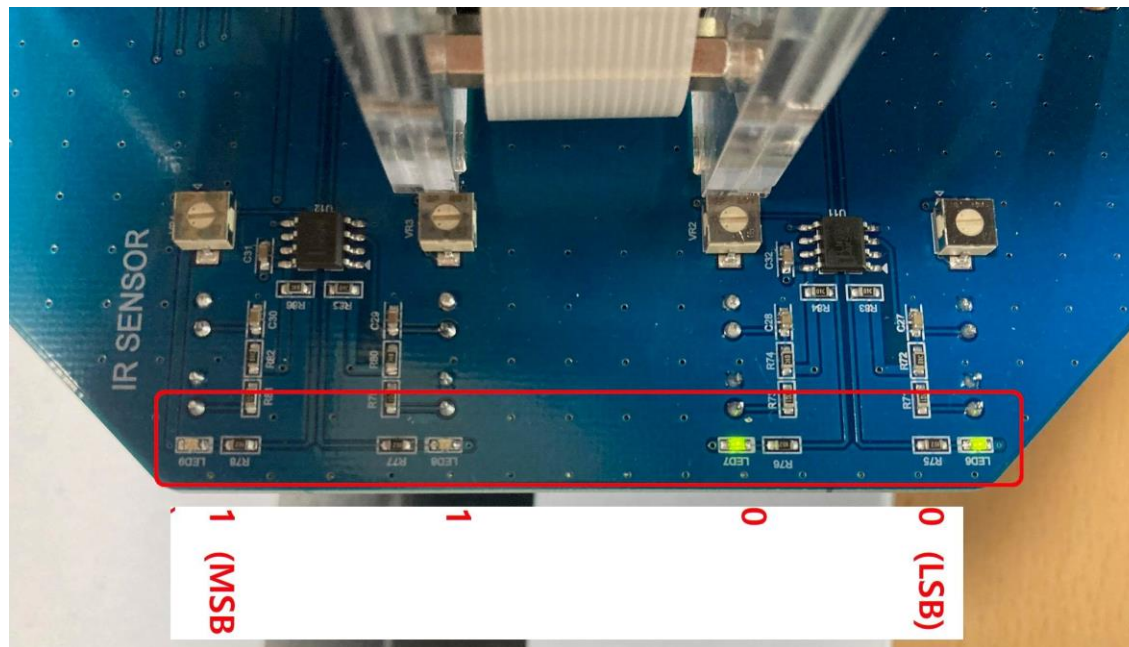
- 다음과 같은 코드를 작성하여 적외선 센서의 측정의 값을 입력받아 보자.

```
11
12 try:
13     while(True):
14         ir = swcar.SIO_ReadIR()
15         print(f'ir = {ir:04b}')
16         time.sleep(1)
17
18 except KeyboardInterrupt:
19     pass
```

Section 01 적외선 센서

□ 파이썬 코드

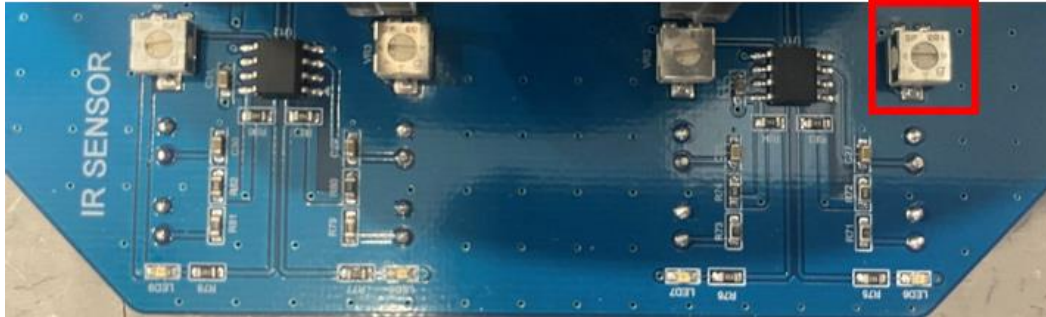
- 코드를 실행하면 화면에 0이 출력된다.
- 적외선 센서는 자동차 키트 전면부에 장착되어 있는데, 그 밑에 색이 있는 물체를 집어넣으면 출력되는 숫자가 바뀌는 것을 확인할 수 있다.



Section 01 적외선 센서

□ 파이썬 코드

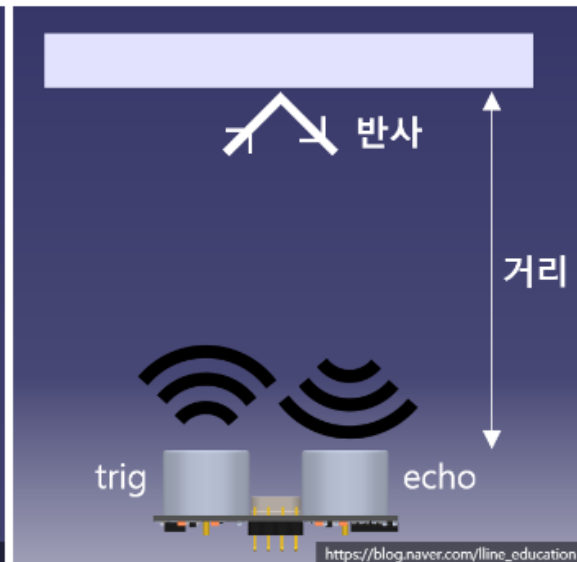
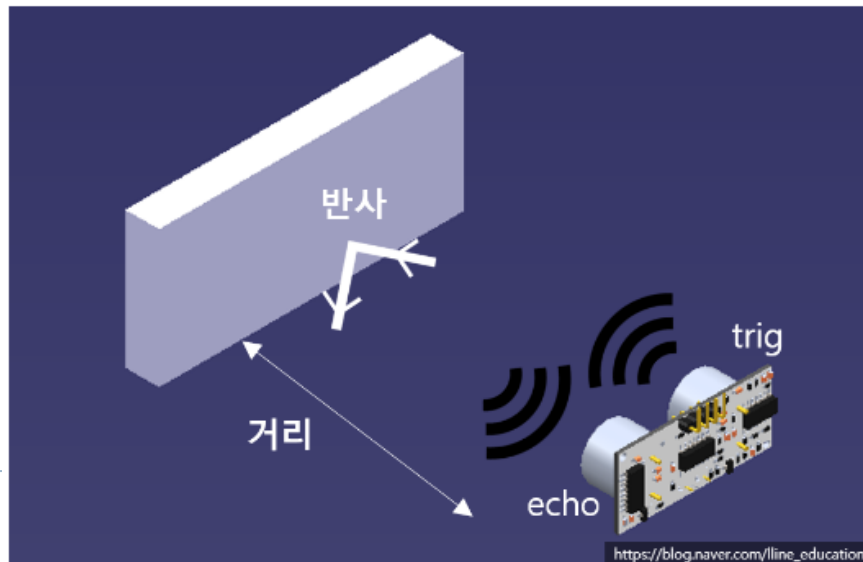
- 만약 IR 센서가 빛에 대해 적절히 반응하지 않는다면, IR 센서 보드의 가변저항을 적절히 조절함으로써 원하는 감도를 설정할 수 있다.
- 차량 메인보드의 IR 센서 상태 LED 를 참고하여 가변 저항을 조절하여 민감도를 조절할 수 있다.
- 적절히 조절하여, 밝은 상태와 어두운 상태를 구별할 수 있도록 설정한다.



Section 02 초음파 센서

□ 초음파 센서 US-015

- SmartCar에 사용되는 US-015는 거리 측정용으로 많이 사용하는 HC-SR04와 모양도 사용방법도 동일하지만 보다 나은 성능을 가진 고정밀 초음파 거리센서이다.
- US-015 센서는 로봇의 눈처럼 생겼다.
- 2개의 초음파 장치 중 하나에서는 40kHz의 초음파를 발사하고 나머지 하나의 센서에서는 반사되어 되돌아오는 초음파를 감지한다.
- 이 시간차를 이용해 거리를 측정하는 것이다.



Section 02 초음파 센서

□ 초음파 센서 US-015

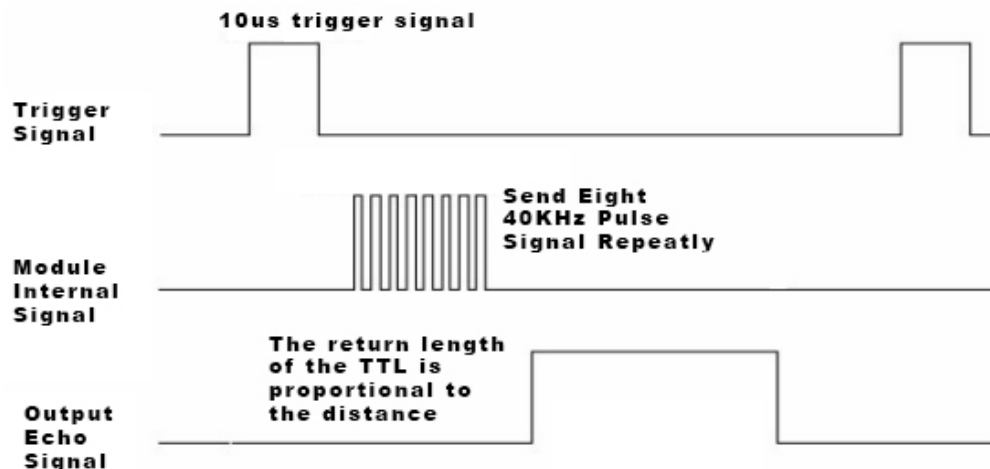
- 초음파 센서는 송신부와 수신부로 이루어져 있으며 송신부는 Transmitter 초음파를 발생하는 것이고, 수신부는 Receiver 초음파가 발생한 것이 물체에 부딪혀 다시 돌아오는 것이다.
- 이에 따라 초음파가 돌아오는 시간에 따라 물체와 초음파 사이에 거리를 알 수 있다.
- 공식으로 보자면 다음과 같다.

$$D(\text{거리}) = \frac{T(\text{시간})}{2} \times V_s(\text{속도})$$

Section 02 초음파 센서

□ 초음파 센서 US-015

- 측정은 다음과 같은 흐름으로 이루어진다.
 - 트리거 핀으로 10us 시간 동안 ON 신호를 주면 센서가 8개 펄스의 초음파를 발사한다.
 - 물체에서 반사된 초음파(echo)를 감지해 에코 핀을 통해 신호를 전달한다.
 - 초음파가 발사되는 순간 에코 핀은 ON 상태가 되고 반사파를 감지하는 순간 OFF로 변하는데, 이 시간차를 측정해서 해당 값을 초음파의 거리 속도 계산식에 넣어 거리를 구한다.



Section 02 초음파 센서

□ 파이썬 코딩

- 다음과 같은 코드를 작성하여 초음파 센서의 측정의 값을 입력받아 보자.

```
1  from ctypes import *
2  import os
3  import time
4
5  WiringPi = CDLL("/home/pi/WiringPi/wiringPi/libwiringPi.so.2.70", mode=RTLD_GLO
  OBAL)
6  swcar = cdll.LoadLibrary('/home/pi/swcar/libswcar.so')
7
8  swcar.SIO_Init(0)
9
10 print('press ctrl + c to terminate program')
11
```

Section 02 초음파 센서

□ 파이썬 코딩

- 다음과 같은 코드를 작성하여 초음파 센서의 측정의 값을 입력받아 보자.

```
11
12 try:
13     while(True):
15         us_rear = swcar.SIO_ReadDistUS(0)
17         print("rear = ", us_rear)
18         time.sleep(1)
19
20 except KeyboardInterrupt:
21     pass
```

Section 02 초음파 센서

□ 파이썬 코딩

- 코드를 실행하면 콘솔창에 센서 측정값이 출력된다.
- 초음파 센서에 장애물을 배치하여 값의 변화를 확인한다.



후면부 초음파 센서

```
Shell

>>> %Run ex5_sensor_us_read.py

Press ctrl + c to terminate program
rear = 154
rear = 154
rear = 154
```

콘솔 출력 결과

Section 03 라이다 센서

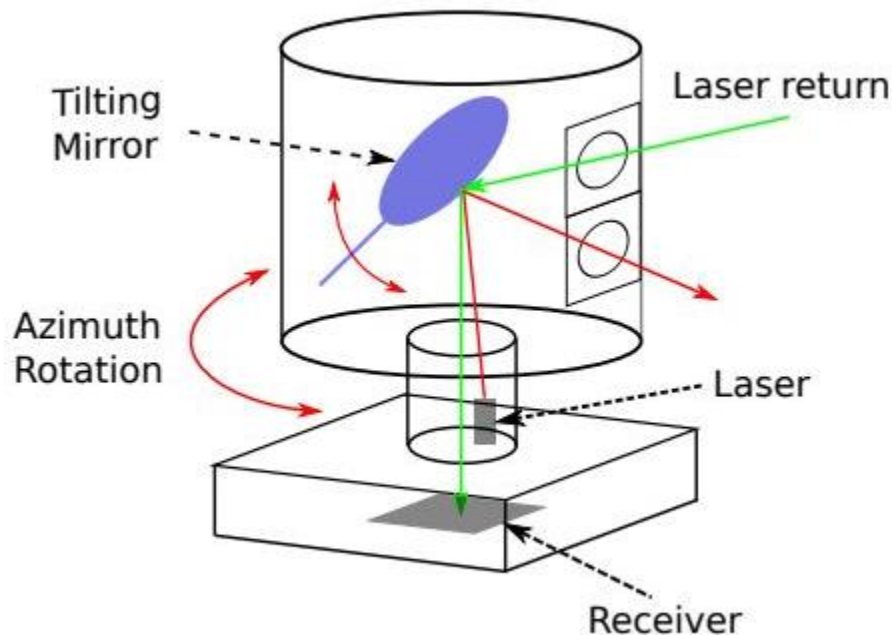
- 라이다(LiDAR : Light Detection And Ranging)는 에너지 밀도가 높은 고출력의 레이저(LASER : Light Amplification by Stimulated Emission of Radiation) 펄스를 송신하고 목표물에 맞고 되돌아오는 시간을 측정하여 사물까지의 거리, 방향 및 속도 등을 감지할 수 있는 센서
- 라이다 센서는 초당 수 백만 개의 달하는 레이저빔을 송수신하므로 높은 인지 해상도로 정확하고 빠른 신호 처리가 가능하지만 양산형 차량에 적용하기에는 아직까지 가격 경쟁력이 부족



Section 03 라이다 센서

□ 회전형 라이다

- 회전형 라이다 센서의 송신부는 레이저 다이오드를 활용하여 레이저 빔을 송신하고, 틸팅(Tilting) 거울을 통해 방향이 조정된 레이저를 포토다이오드를 통해 수신
- 라이다 센서 내부의 서보 모터를 이용하여 공간적으로 360° 회전이 가능
- 차량을 중심으로 원하는 측정 방향으로 레이저를 선택적으로 송수신 가능



Section 03 라이다 센서

□ 회전형 라이다

- 기구적으로 회전하는 라이다 센서의 경우 보통 차량의 지붕(Roof)에 장착하여 전방향(Omnidirectional)으로 사물의 유무와 거리 및 방향 등의 정보를 인지
- 넓은 수평 시야각을 갖고 상대적으로 좋은 데이터 정확도를 갖는다는 장점이 있다.
- 하지만 모터가 있기 때문에 가격이 비싸고 내구성이 약함.
- 360도를 봐야 해서 매립해서 사용할 수 없음.
- 도로 위의 물체를 인식하기 위해 각도를 하향 조정할 경우 인지 거리가 짧아지는 한계

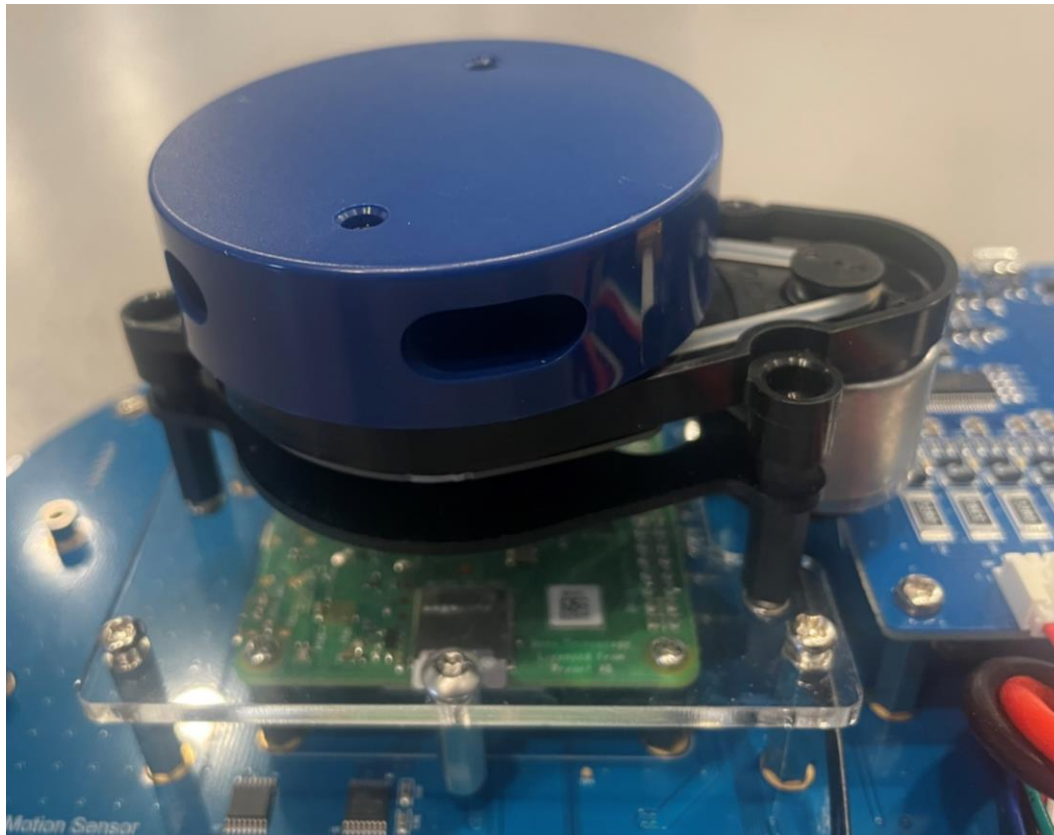
Section 03 라이다 센서

□ 고정형 라이다 (Solid State Lidar)

- 레이저 광원을 기구적으로 회전시키는 모터가 없으며, 수직 방향으로 30° 범위 안에서 16개 채널이 2° 단위의 해상도로 스캐닝(Scanning)하는 형태를 가지고 있다.
- 장점
 - 모터가 없어서 가격이 저렴하고 내구성이 좋음
 - 소형화 할 수 있음
 - 한 방향만 스캔하기 때문에 매립해서 사용할 수 있음.
 - 높은 해상도.
- 단점
 - 좁은 수평 시야각.
 - 상대적으로 떨어지는 데이터 정확도.

Section 03 라이다 센서

□ SmartCar의 라이다 센서 모듈



Section 03 라이다 센서

□ 파이썬 코드

- 다음과 같은 코드를 작성하여 라이다 센서의 측정의 값을 입력받아 보자.

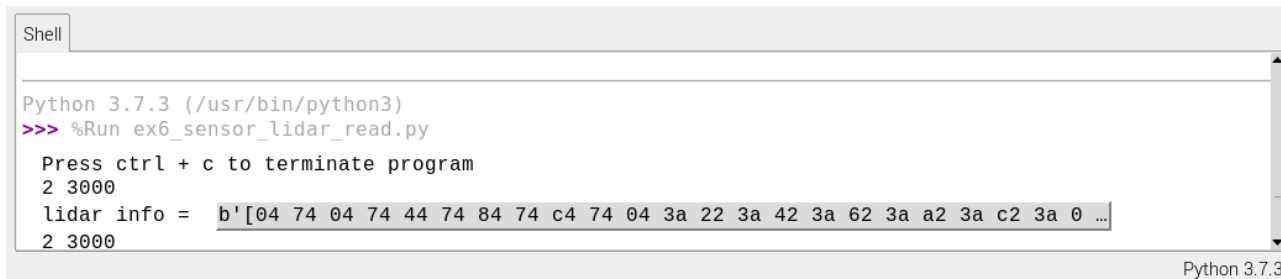
```
1  from ctypes import *
2  import os
3  import time
4
5  WiringPi = CDLL("/home/pi/WiringPi/wiringPi/libwiringPi.so.2.70", mode=RTLD_GLOBAL)
6  swcar = cdll.LoadLibrary('home/pi/swcar/libswcar.so')
7
8  swcar.SIO_Init(0)
9
10 print('press ctrl + c to terminate program')
11
12 swcar.SIO_ReadLidar.restype = c_char_p
13 swcar.SIO_ReadLidar.argtypes = None
14
```

Section 03 라이다 센서

□ 파이썬 코드

- 다음과 같은 코드를 작성하여 라이다 센서의 측정의 값을 입력받아 보자.

```
15 swcar.SIO_ActivateLidar(1);
16
17 while(True):
18     lidar = swcar.SIO_ReadLidar()
19     print("lidar info = ", lidar)
20     time.sleep(1)
```



```
Shell
Python 3.7.3 (/usr/bin/python3)
>>> %Run ex6_sensor_lidar_read.py
Press ctrl + c to terminate program
2 3000
lidar info = b'[04 74 04 74 44 74 84 74 c4 74 04 3a 22 3a 42 3a 62 3a a2 3a c2 3a 0 ...'
2 3000
Python 3.7.3
```

Section 04 서보 모터

□ 서보 모터란?

- 서보 모터는 정확한 각도 회전을 위해 사용된다는 점은 스텝 모터와 비슷하지만 구동되는 방식은 전혀 다르다.
- 서보 모터는 각도를 제어할 수 있는 DC 모터이다.
- 90도 회전하고 정지했다가 다시 90도 돌아가도록 설정할 수 있다.
- 자동으로 움직이는 부품에 대한 정밀도가 필요할 때 유용하다.
- 서보 모터 내부에는 DC 모터, 전위차계 및 모터를 제어하는 회로의 세 부분이 있다.
- 서보 모터의 전위차계는 LED를 켤 때 사용하는 저항과 마찬가지로 저항이다.
- 예외는 돌릴 때 저항 값을 변경할 수 있다는 것이다.

Section 04 서보 모터

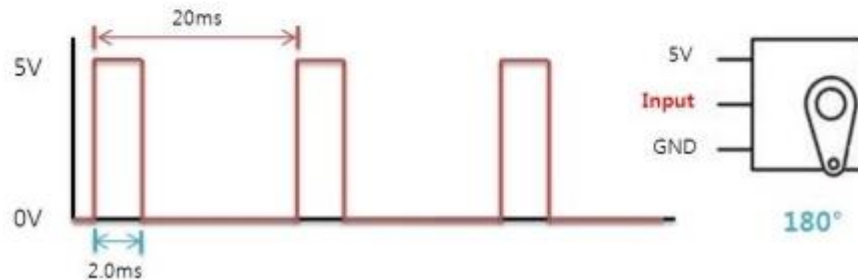
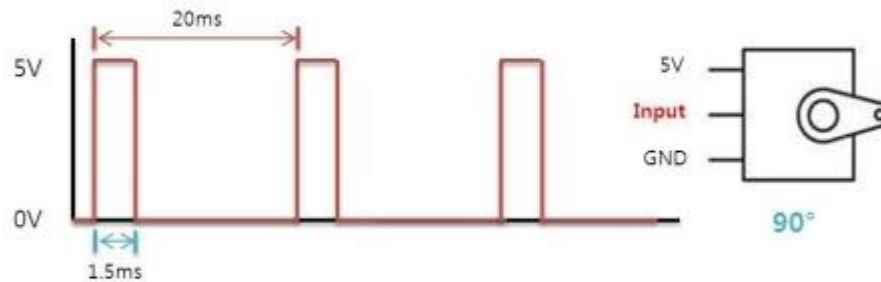
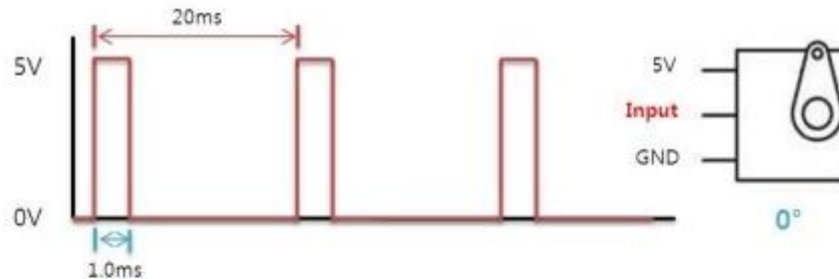
□ 서보 모터란?

- 서보 모터에서 전위차계는 DC 모터에 맞춰져 있으므로 DC 모터가 회전할 때 회전한다.
- 이를 통해 모터 샤프트의 각도를 알 수 있다.
- 컨트롤러 회로는 특정 각도에 도달하면 정지하도록 지시한다.
- 제어 방법은 DC 모터와 같은 PWM이지만, 서보 모터의 PWM은 주파수가 정해져 있으며, 듀티비라기보다는 신호의 유지 시간으로 회전 각도가 결정된다.
- 대부분의 서보 모터는 3개의 선으로 이루어져 있다.
 - 전원, GND, 입력 신호.

Section 04 서보 모터

□ 서보 모터란?

- 입력 신호에 다음과 같은 신호가 입력되면 신호에 맞는 각도로 모터가 회전



Section 04 서보 모터

□ 서보 모터란?

- 서보 모터를 제어하기 위해서는 50Hz의 주파수를 가지는 신호가 입력되어야 한다.
- 즉, 한 주기 당 20ms의 시간을 가진다는 것.
- 20ms의 주기 안에서 HIGH 신호의 폭(=시간)이 얼마인가에 따라 서보 모터의 회전 각도가 결정된다.
- 서보 모터의 종류에 따라 약간씩 다를 수는 있겠지만 보통은 1.0ms에서 0° , 1.5ms에서 90° , 2.0ms에서 180° 의 각도를 가진다.
- 사실 1.5ms에서 90° , 정중앙에 위치하는 것이고 1.5ms보다 작을 때 시계 반대 방향으로 이동, 1.5ms보다 클 때 시계 방향으로 이동하는 거라고 생각하면 된다.
- 모터가 각 방향으로 최대한 이동했을 때의 신호 시간 폭은 모터마다 다를 수 있다.
- 물론 최대한 이동했을 때의 각도도 0° 나 180° 보다 크거나 작을 수 있다.

Section 04 서보 모터

□ 서보 모터란?

- 서보 모터는 위치를 유지하기 위해서는 신호를 계속해서 보내야 한다.
- 신호를 주지 않으면 모터는 각도를 유지하지 못하고 흐물흐물 돌아가버린다.
- 서보 모터의 경우 DC 모터나 스텝 모터처럼 큰 전압을 요구하지는 않지만, 큰 전류를 요구하기 때문에 역시 외부 전압을 라즈베리 파이 보드에 연결해주는 것이 좋다.
- 게다가 서보는 회전하지 않아도 각도를 유지하기 위해 계속 소모하게 되므로, 만일 건전지를 이용한다면 건전지의 소모 시간이 예상 소모 시간보다 훨씬 짧을 수 있다.

Section 04 서보 모터

□ 파이썬 코드

- 다음과 같은 코드를 작성하여 서보모터의 각도를 제어해보자.

```
1  from ctypes import *
2  import os
3  import time
4
5  WiringPi = CDLL("/home/pi/WiringPi/wiringPi/libwiringPi.so.2.70", mode=RTLD_G
  LOBAL)
6  swcar = cdll.LoadLibrary('/home/pi/swcar/libswcar.so')
7
8  swcar.SIO_Init(0)
9
10 print('press ctrl + c to terminate program')
11 print("Running Servo Motor.")
```

Section 04 서보 모터

□ 파이썬 코드

- 다음과 같은 코드를 작성하여 서보모터의 각도를 제어해보자.

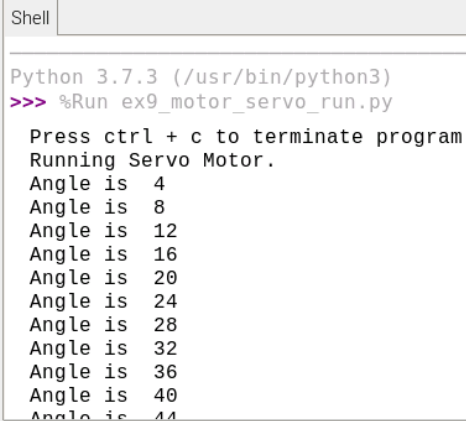
```
12
13 iAngle = 0
14 Iinc = 4
15 try:
16     while True:
17         iAngle += Iinc
18         if (iAngle >= 100) :
19             iInc = -4
20         if (iAngle <= 0) :
21             iInc = 4
22
```

Section 04 서보 모터

□ 파이썬 코드

- 다음과 같은 코드를 작성하여 서보모터의 각도를 제어해보자.

```
23     swcar.SIO_WriteServo(100, iAngle)
24     print("Angle is " , iAngle)
25     time.sleep(0.10)
26
27 except KeyboardInterrupt:
28     swcar.SIO_WriteServo(100, 50)
```



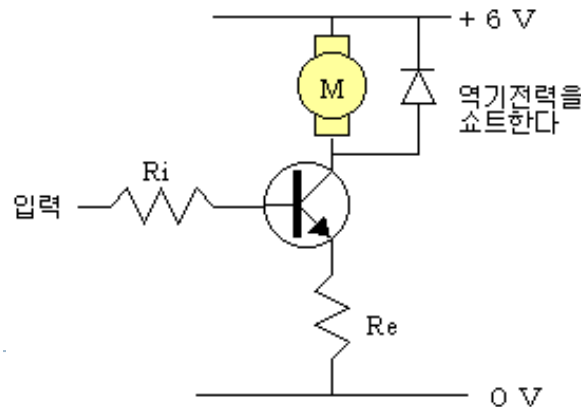
Shell

```
Python 3.7.3 (/usr/bin/python3)
>>> %Run ex9_motor_servo_run.py
Press ctrl + c to terminate program
Running Servo Motor.
Angle is 4
Angle is 8
Angle is 12
Angle is 16
Angle is 20
Angle is 24
Angle is 28
Angle is 32
Angle is 36
Angle is 40
Angle is 44
```

Section 05 구동 모터

□ DC 모터

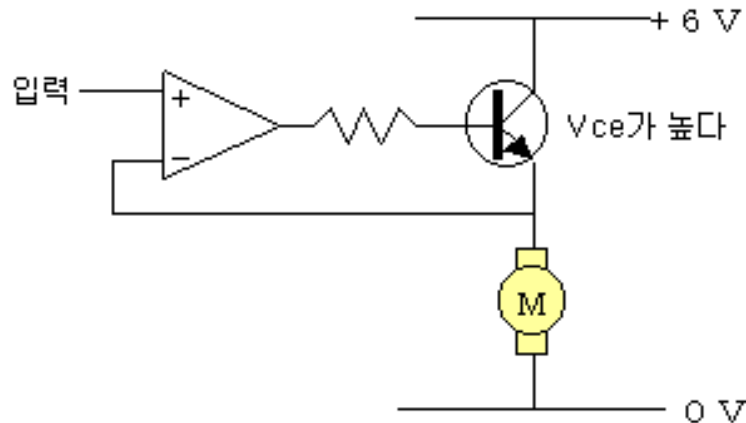
- DC모터는 DC(직류)전원으로 작동하는 모터를 말한다.
- 선풍기나 RC카와 같이 빠르고 연속적인 회전이 필요할 때 사용한다.
- DC모터는 (+)극과 (-)극에 전원을 입력하여 작동시킬 수 있으며, 극을 반대로 연결하면 회전 방향을 바꿀 수 있다.
- DC 모터의 속도를 연속적으로 바꾸려는 경우에는 어떻게 하는가?
- 기본적으로는 DC 모터에 가하는 전압을 바꾸면 속도는 변화한다.
- 단순히 모터의 코일에 흐르는 전류와 속도가 정비례하기 때문에 아래 그림과 같이 하여 모터의 구동전압을 변화시키면 속도를 가변으로 할 수 있다.



Section 05 구동 모터

□ DC 모터

- 이 구동전압을 변화시키는 방법으로 아날로그 방식과 펄스폭 변조방식의 두 가지 방법이 있다.
- 아날로그 방식은 직접 구동전압 그 자체를 변화시키는 것으로, 기본회로는 아래 그림과 같다.

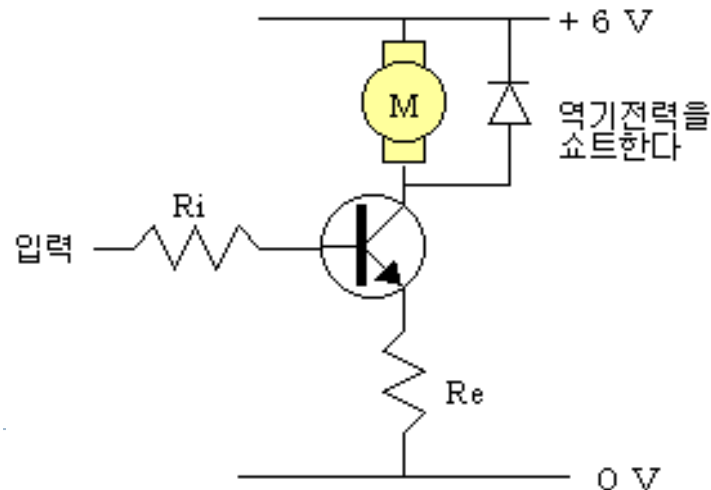


- 저속으로 할 때, 전력 사용 효율이 나빠지고 만다.
- 그러나, 소형 모터이고, 게다가 속도의 가변폭이 작아도 좋은 경우에는 손실을 작게 할 수 있다는 점과, 제어회로가 간단하기 때문에 흔히 사용되고 있다.

Section 05 구동 모터

□ DC 모터

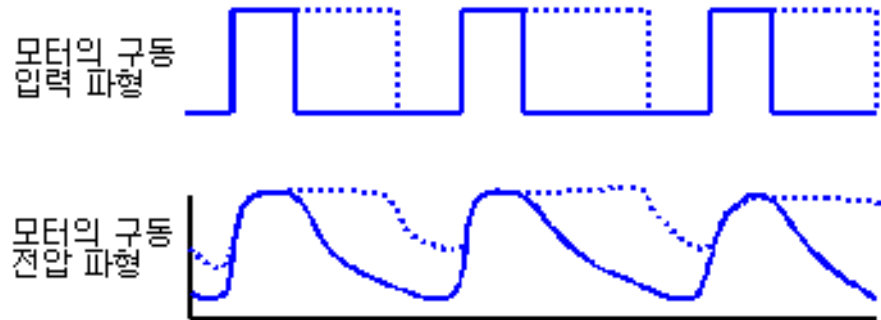
- PWM 방식은 결과적으로는 구동전압을 바꾸고 있는 것과 같은 효과를 내고 있지만, 그 방법이 펄스폭에 따르고 있으므로 펄스폭 변조(PWM: Pulse Width Modulation)라 부르고 있다.
- 구체적으로는 모터 구동전원을 일정 주기로 On/Off 하는 펄스 형상으로 하고, 그 펄스의 duty비(On 시간과 Off 시간의 비)를 바꿈으로써 실현하고 있다.
- 기본회로는 아래 그림과 같으며, 그림에서 트랜지스터를 일정시간 간격으로 On/off하면 구동전원이 On/Off 되는 것이다.



Section 05 구동 모터

□ DC 모터

- 이 펄스 형상의 전압으로 DC 모터를 구동했을 때의 실제 모터에 가해지는 전압 파형은 아래 그림과 같이 되며, 평균전력, 전압을 생각하면 외관상, 구동전압이 변화하고 있는 것이다.



Section 05 구동 모터

□ 파이썬 코딩

- 다음과 같은 코드를 작성하여 구동모터의 속도를 제어해보자.

```
1  from ctypes import *
2  import os
3  import time
4
5  WiringPi = CDLL("/home/pi/WiringPi/wiringPi/libwiringPi.so.2.70", mode=RTLD_GL
   OBAL)
6  swcar = cdll.LoadLibrary('/home/pi/swcar/libswcar.so')
7
8  swcar.SIO_Init(0)
9
10 print('press ctrl + c to terminate program')
11 print("Running Driving Motor.")
12
13 swcar.SIO_MaxMotorSpeed(50)
14
```

Section 05 구동 모터

□ 파이썬 코딩

- 다음과 같은 코드를 작성하여 구동모터의 속도를 제어해보자.

```
15 for i in range(1, 10):
16     speed = i * 10
17     swcar.SIO_ForwardMotor(speed)
18     print("Forward Speed is %d" %speed)
19     time.sleep(1)
20
21 swcar.SIO_ForwardMotor(0)
22 print("Forward Speed is %d" %(0))
23 time.sleep(3)
24
25 for i in range(1, 10):
26     speed = i * 10
27     swcar.SIO_ReverseMotor(speed)
```

Section 05 구동 모터

□ 파이썬 코딩

- 다음과 같은 코드를 작성하여 구동모터의 속도를 제어해보자.

```
28     print("Reverse Speed is %d" % speed)
29     time.sleep(1)
30
31     swcar.SIO_ReverseMotor(0)
32     print("Reverse Speed is %d" %(0))
33     time.sleep(3)
34
35     print('Stop Driving Motor.')
```

Section 05 구동 모터

□ 파이썬 코딩

- 이 예제는 드라이빙 모터를 구동하여 바퀴가 회전하는 것을 보여 주는 예제이다.
- `SIO_ForwardMotor(int iSpeed)`, `SIO_ReverseMotor(speed)` 함수를 사용하며, Parameter 값으로 바퀴의 구동 속도를 조절한다.
- `iSpeed` 의 값은 0 ~ 100 사이의 값을 갖는다.
- 0 은 stop 상태가 되며, 100 은 최대값이 된다. `SIO_MaxMotorSpeed()` 함수를 이용하면, Speed 의 최대값을 제한할 수 있다.
- 예를 들어, `SIO_MaxMotorSpeed(50)` 으로 설정하면, `SIO_ForwardMotor()` 나 `SIO_ReverseMotor()` 함수에서 50을 넘는 값을 입력해도, 최대값이 50으로 제한된다.

Section 06 GPS

□ GPS 개요

- 현재 위치를 인식하는 다양한 방법 중에 가장 많이 사용하는 것이 GPS 수신기일 것이다.
- GPS(Global Positioning System)는 미국 국방부에서 개발한 범지구 위치결정 시스템이다.
- 쉽게 설명하면 지구상에 떠 있는 24개 이상의 인공위성으로부터 전파를 수신받은 GPS 수신기가 삼각측량(trilateration)과 같은 방식으로 자신의 위치를 계산하는 방식을 말한다.
- 실외에서 사용할 때 그 오차가 많아야 1~10여 미터이므로 군사용, 측량, 자동차 내비게이션용으로 널리 쓰이며, 모바일 폰 및 차량 등에 대부분 장착되어 있다.

Section 06 GPS

□ NEO-6M 제품 사양과 회로연결



Supply Voltage: 3.3~5V

Module Size: 35 x 26 x 3mm

Antenna Size: 50 x 25mm

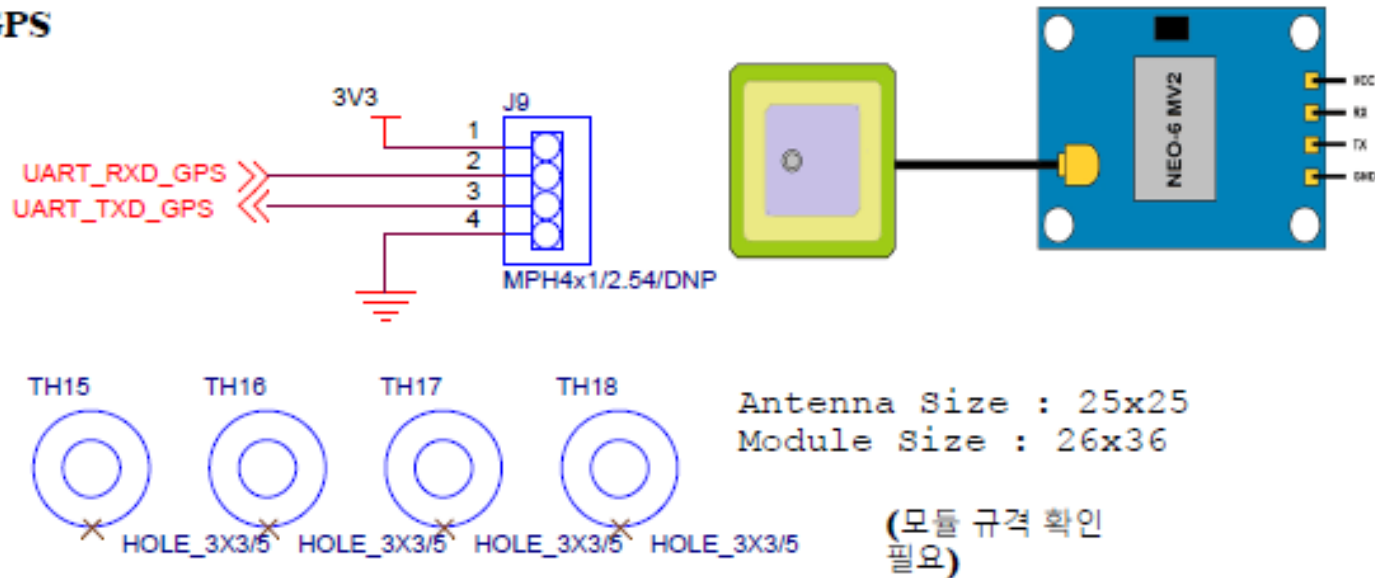
UART Interface

Baud Rate : 9600bps

Section 06 GPS

□ NEO-6M 제품 사양과 회로연결

GPS



Section 06 GPS

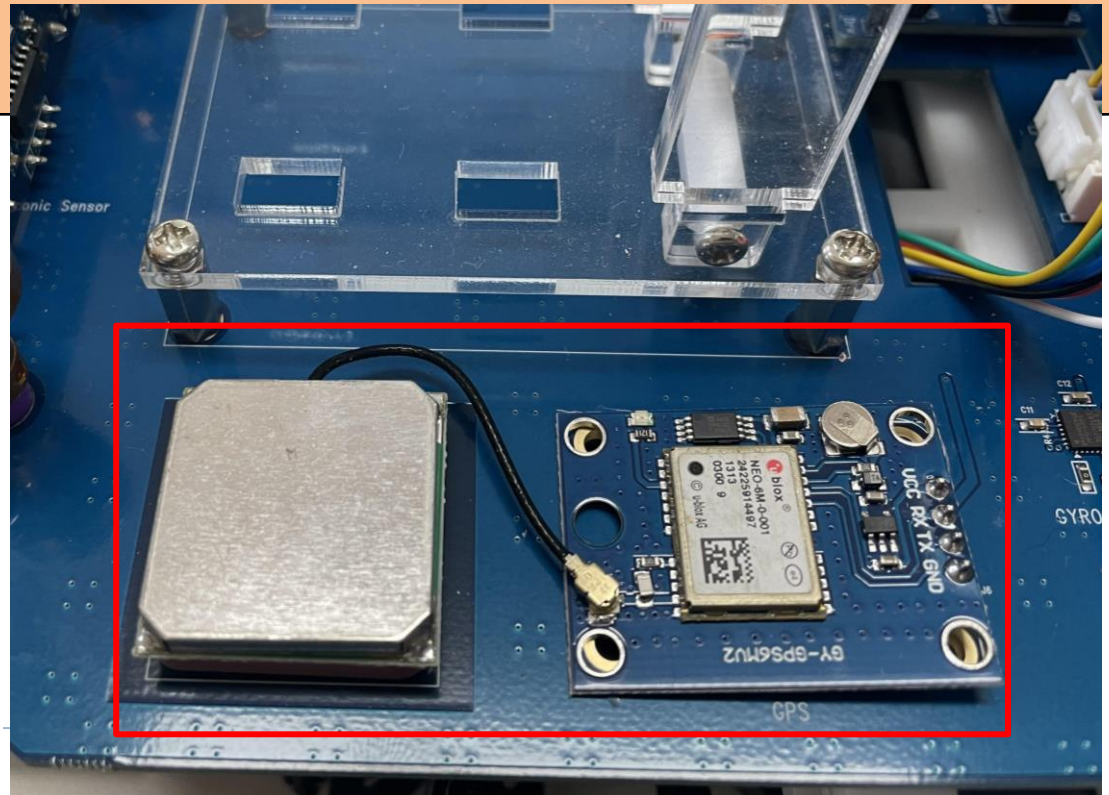
□ GPS 실습 코드

```
1  from ctypes import *
2  import os
3  import time
4
5  WiringPi = CDLL("/home/pi/WiringPi/wiringPi/libwiringPi.so.2.70", mode=RTLD_GLO
  BAL)
6  swcar = cdll.LoadLibrary('home/pi/swcar/libswcar.so')
7
8  swcar.SIO_Init(0)
9
10 print('press ctrl + c to terminate program')
11
12 swcar.SIO_ReadGPS.restype = c_char_p
13 swcar.SIO_ReadGPS.argtypes = None
14
```

Section 06 GPS

□ GPS 실습 코드

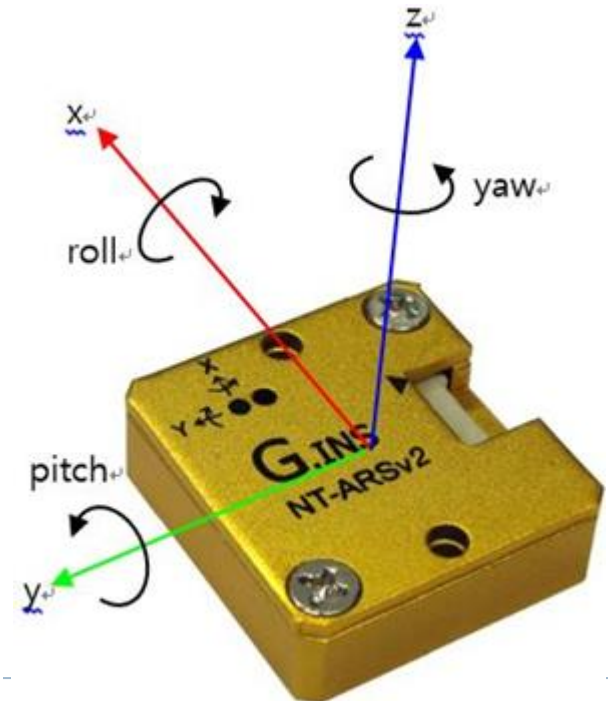
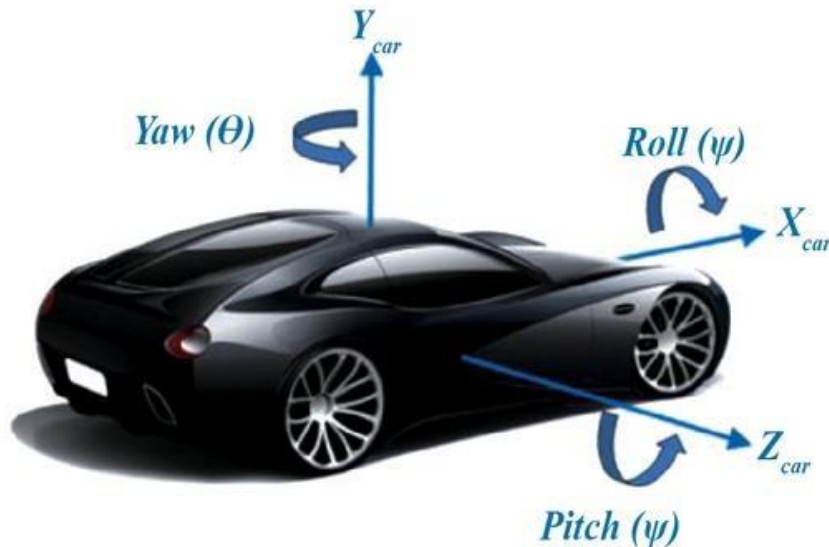
```
15  
16 while True:  
17     gps = swcar.SIO_ReadGPS()  
18     print("gps = ", gps)  
19     time.sleep(1)
```



Section 07 가속도 센서

□ IMU 센서

- 관성 센서(IMU : Inertial Measurement Unit)는 보통 차량의 무게 중심(C.G : Center of Gravity)에 장착되어 현재 차량에 발생한 가속도와 각속도를 계측하는 목적으로 사용
- 초기에는 아날로그 형태로 개발되었지만, 현재는 MEMS(Micro Electro Mechanical System) 기술이 적용



Section 07 가속도 센서

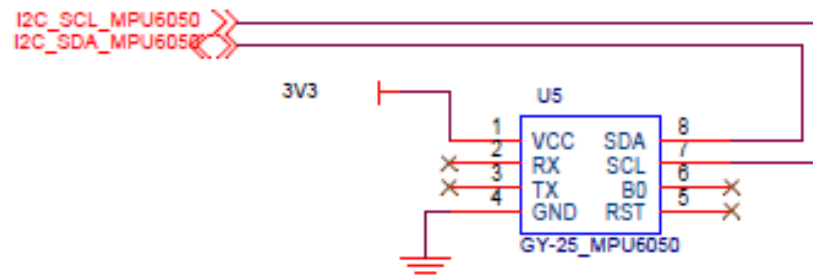
□ IMU 센서 구조

- IMU 센서는 차량의 진행 방향인 x축 방향의 가속도, 좌우 방향인 y축 방향에 대한 가속도, 위쪽 방향인 z축 방향의 회전 각속도를 계측
- z축의 회전 각속도를 요 레이트(Yaw Rate), x축의 회전 각속도는 롤 레이트(Roll Rate), y축의 회전 각속도를 피치 레이트(Pitch Rate)라고 한다.
- 일반적인 자동차에서 IMU 센서의 출력 중 요 레이트만 사용

Section 07 가속도 센서

□ 가속도 센서를 이용한 각도 계산

- 이러한 가속도센서를 이용한 각도계산은, 힘이 일정하다면 정확한 각도 값을 얻어낼 수 있지만 진동과 같은 떨림에 약하다는 단점이 있다.
- 다음 그림은 자이로 센서의 회로도이다.



- 여기까지 가속도센서를 통해 기울어진 각도를 계산해보았는데, 다음으로 이 식이 적용된 파이썬 코드를 작성하여 MPU6050센서를 기울였을 때 각도가 계산되는지 확인하는 실습을 해보자.

Section 07 가속도 센서

□ 파이썬 코드

```
1  from ctypes import *
2  import os
3  import time
4
5  WiringPi = CDLL("/home/pi/WiringPi/wiringPi/libwiringPi.so.2.70", mode=RTLD_GLOBAL)
6  swcar = cdll.LoadLibrary('home/pi/swcar/libswcar.so')
7
8  swcar.SIO_Init(0)
9
10 print('press ctrl + c to terminate program')
11
12 swcar.SIO_ReadGyroAccel.restype = c_char_p
13 swcar.SIO_ReadGyroAccel.argtype = [POINTER(c_int), POINTER(c_int),
14                                     POINTER(c_int)]
15 swcar.SIO_ReadGyroRotate.restype = c_char_p
```

Section 07 가속도 센서

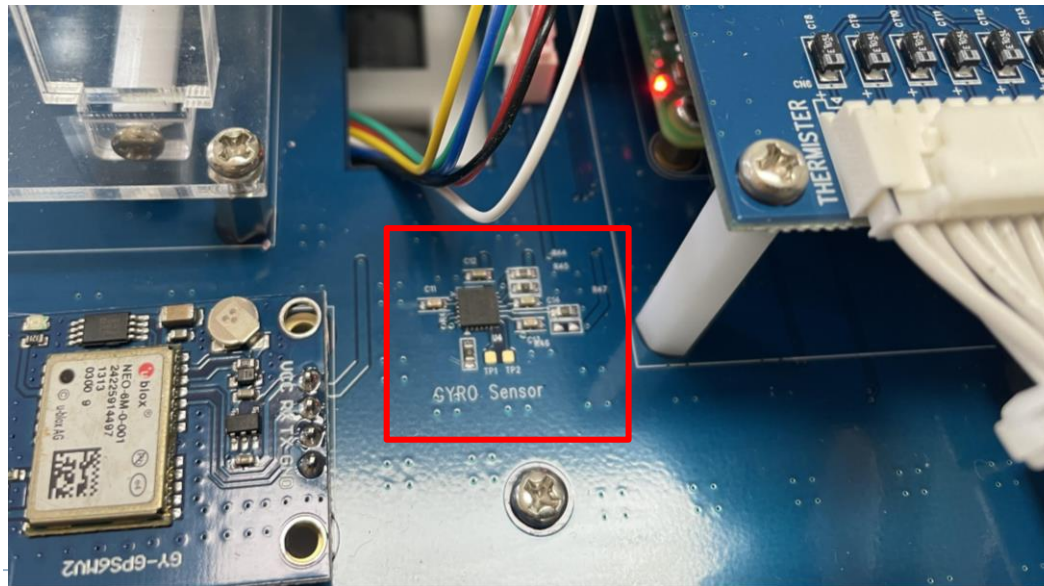
□ 파이썬 코드

```
15 swcar.SIO_ReadGyroRotate.argtype = [POINTER(c_int), POINTER(c_int),  
    POINTER(c_int)]  
16  
17 acc_x = pointer(c_int(0))  
18 acc_y = pointer(c_int(0))  
19 acc_z = pointer(c_int(0))  
20  
21 rot_x = pointer(c_int(0))  
22 rot_y = pointer(c_int(0))  
23 rot_z = pointer(c_int(0))  
24  
25  
26 while True:  
27     swcar.SIO_ReadGyroAccel(acc_x, acc_y, acc_z)  
28     swcar.SIO_ReadGyroRotate(rot_x, rot_y, rot_z)
```


Section 07 가속도 센서

□ 파이썬 코드

```
29  
30     print("acc = ", acc_x[0], acc_y[0], acc_z[0])  
31     print("rot = ", rot_x[0], rot_y[0], rot_z[0])  
32  
33     time.sleep(1)
```



Q&A

