

## CH. 4. 라즈베리파이 모터 제어

# Section 01 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 01 서보 모터

---

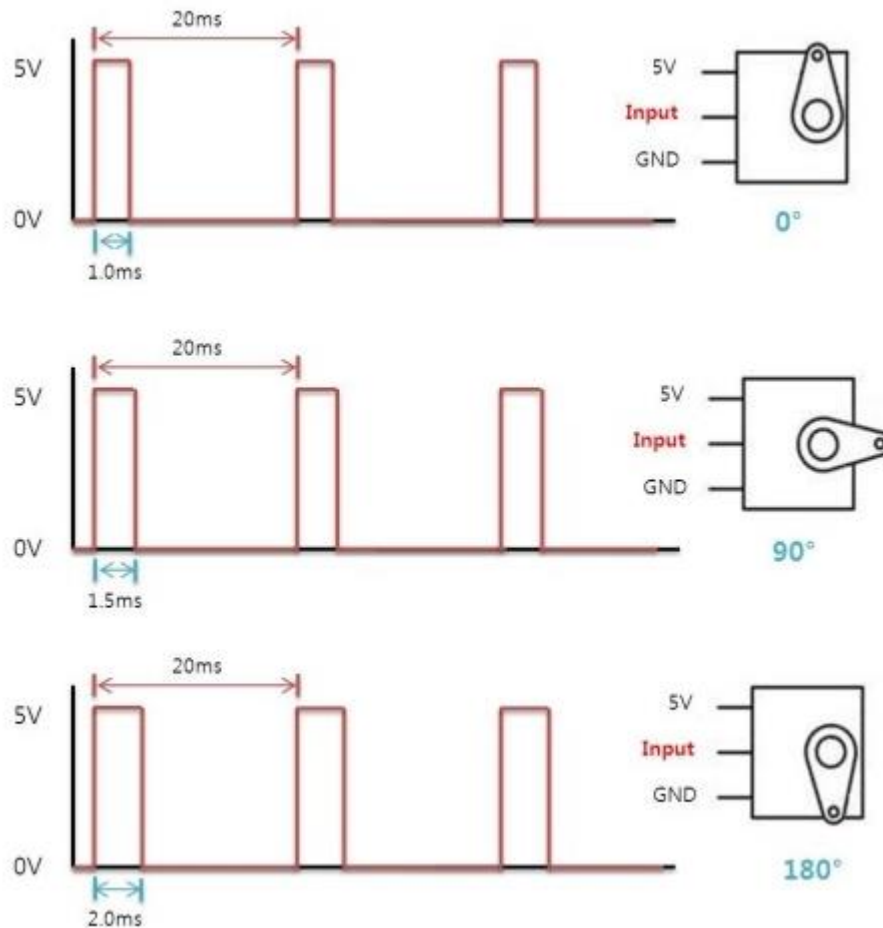
## □ 서보 모터란?

- 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 01 서보 모터

## □ 서보 모터란?

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



# Section 01 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  $0^{\circ}$ , 1.5ms에서  $90^{\circ}$ , 2.0ms에서  $180^{\circ}$ 의 각도를 가진다.
- 사실 1.5ms에서  $90^{\circ}$ , 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  $0^{\circ}$ 나  $180^{\circ}$ 보다 크거나 작을 수 있다.

# Section 01 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 01 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(1)
9
10 print('press ctrl + c to terminate program')
11 print("Running Servo Motor.")
```

# Section 01 서보 모터

---

## □ 파이썬 코드

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

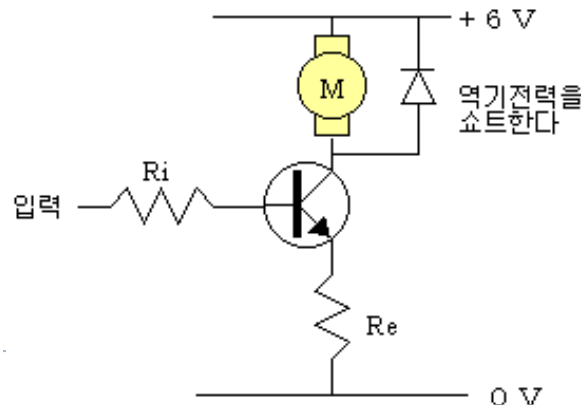
```
12
13 iAngle = -100
14 while(True):
15     iAngle = iAngle + 1
16     if (iAngle > 100) :
17         iAngle = -100
18
19     swcar.SIO_WriteServo(iAngle)
20     time.sleep(0.02)
```



## Section 02 구동 모터

### □ DC 모터

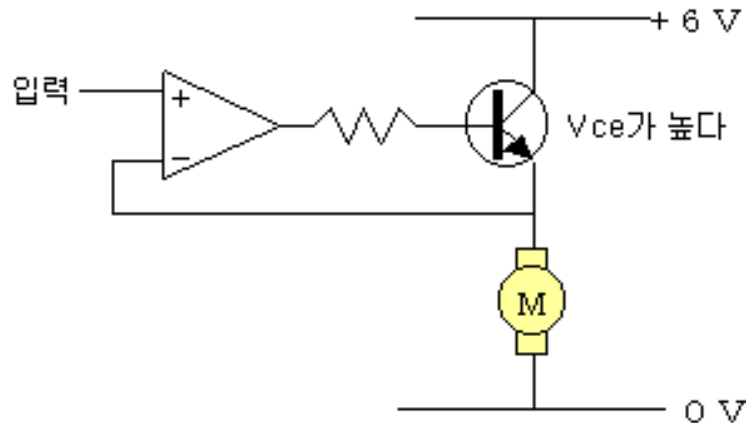
- DC모터는 DC(직류)전원으로 작동하는 모터를 말한다.
- 선풍기나 RC카와 같이 빠르고 연속적인 회전이 필요할 때 사용한다.
- DC모터는 (+)극과 (-)극에 전원을 입력하여 작동시킬 수 있으며, 극을 반대로 연결하면 회전 방향을 바꿀 수 있다.
- DC 모터의 속도를 연속적으로 바꾸려는 경우에는 어떻게 하는가?
- 기본적으로는 DC 모터에 가하는 전압을 바꾸면 속도는 변화한다.
- 단순히 모터의 코일에 흐르는 전류와 속도가 정비례하기 때문에 아래 그림과 같이 하여 모터의 구동전압을 변화시키면 속도를 가변으로 할 수 있다.



## Section 02 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.

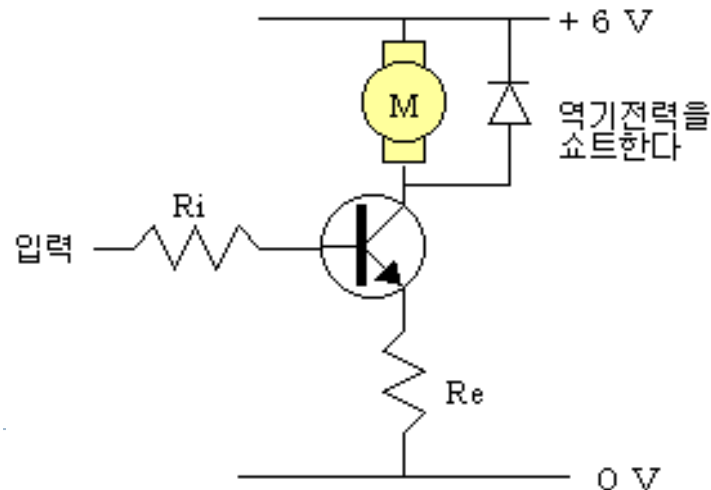


- 저속으로 할 때, 전력 사용 효율이 나빠지고 만다.
- 그러나, 소형 모터이고, 게다가 속도의 가변폭이 작아도 좋은 경우에는 손실을 작게 할 수 있다는 점과, 제어회로가 간단하기 때문에 흔히 사용되고 있다.

## Section 02 구동 모터

### □ DC 모터

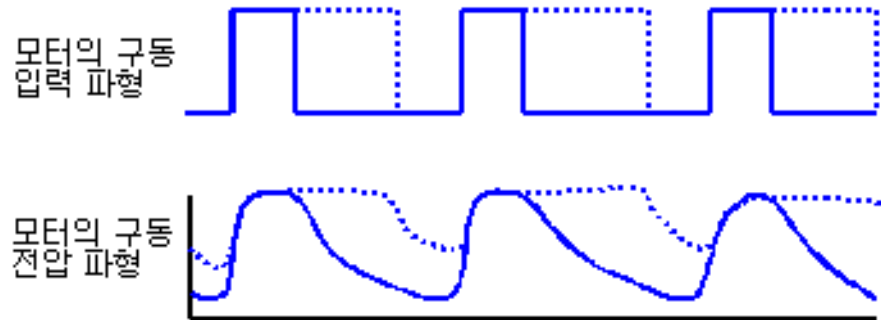
- PWM 방식은 결과적으로는 구동전압을 바꾸고 있는 것과 같은 효과를 내고 있지만, 그 방법이 펄스폭에 따르고 있으므로 펄스폭 변조(PWM: Pulse Width Modulation)라 부르고 있다.
- 구체적으로는 모터 구동전원을 일정 주기로 On/Off 하는 펄스 형상으로 하고, 그 펄스의 duty비(On 시간과 Off 시간의 비)를 바꿈으로써 실현하고 있다.
- 기본회로는 아래 그림과 같으며, 그림에서 트랜지스터를 일정시간 간격으로 On/off하면 구동전원이 On/Off 되는 것이다.



## Section 02 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 02 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(1)
9
10 print('press ctrl + c to terminate program')
11 print("Running Driving Motor.")
12
```

## Section 02 구동 모터

---

### □ 파이썬 코딩

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

```
13 while(True):  
14     swcar.SIO_WriteMotor(-20)  
15     time.sleep(2)  
16     swcar.SIO_WriteMotor(0)  
17     time.sleep(2)  
18     swcar.SIO_WriteMotor(20)  
19     time.sleep(2)  
20     swcar.SIO_WriteMotor(0)  
21     time.sleep(2)
```

---

# Q&A

