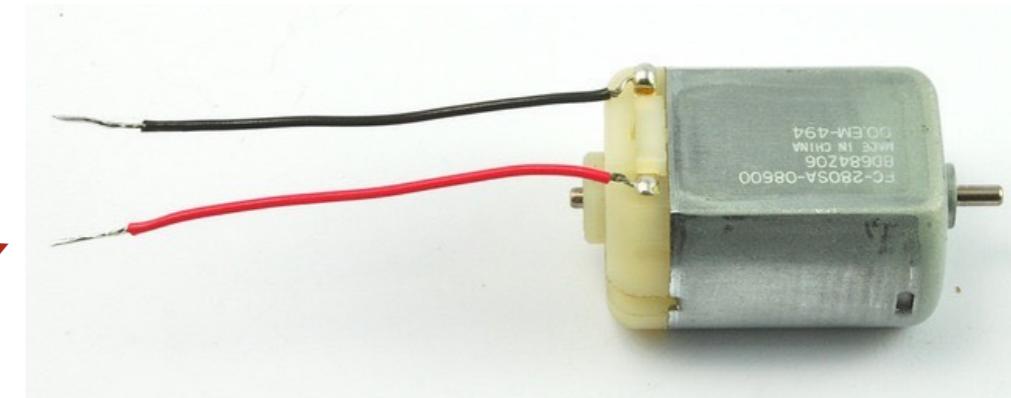


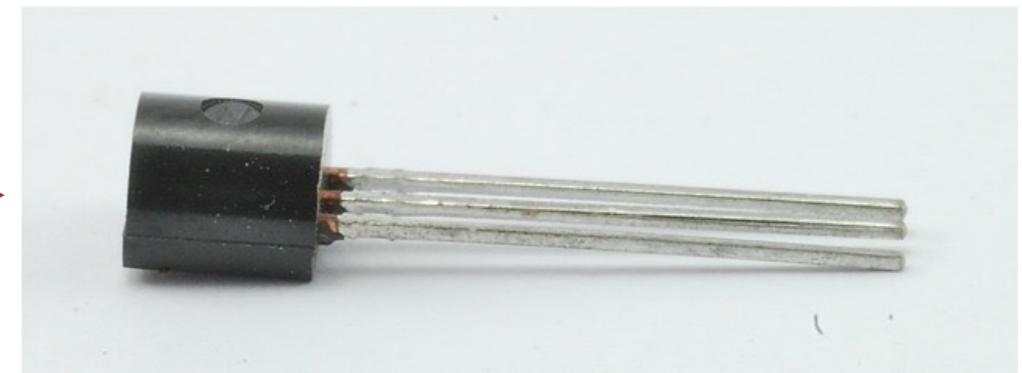
Motors

Lesson 03

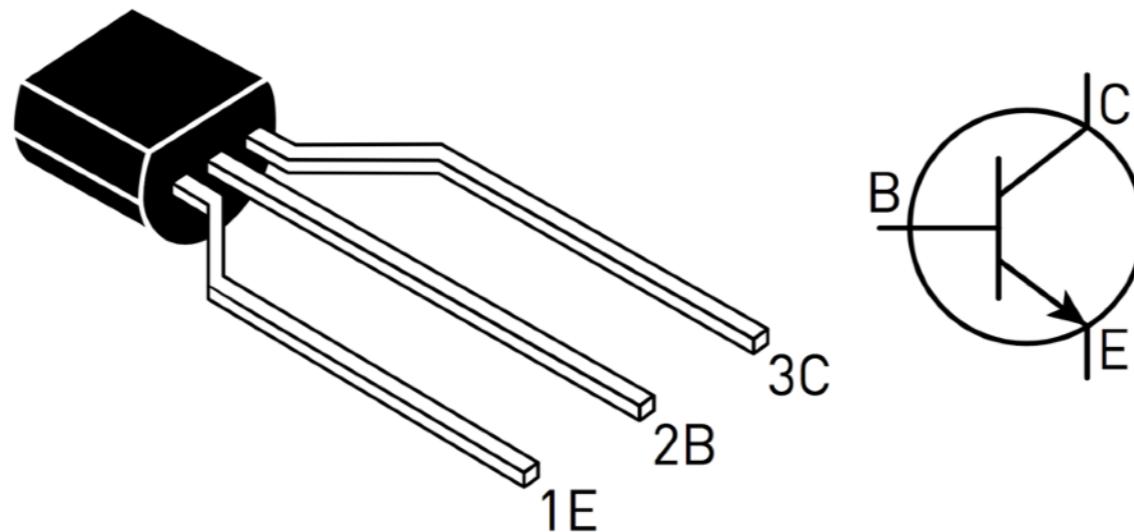
DC Motor 제어



- **DC motor**
- **Transistor**
- **Diode**
- **Resistor**



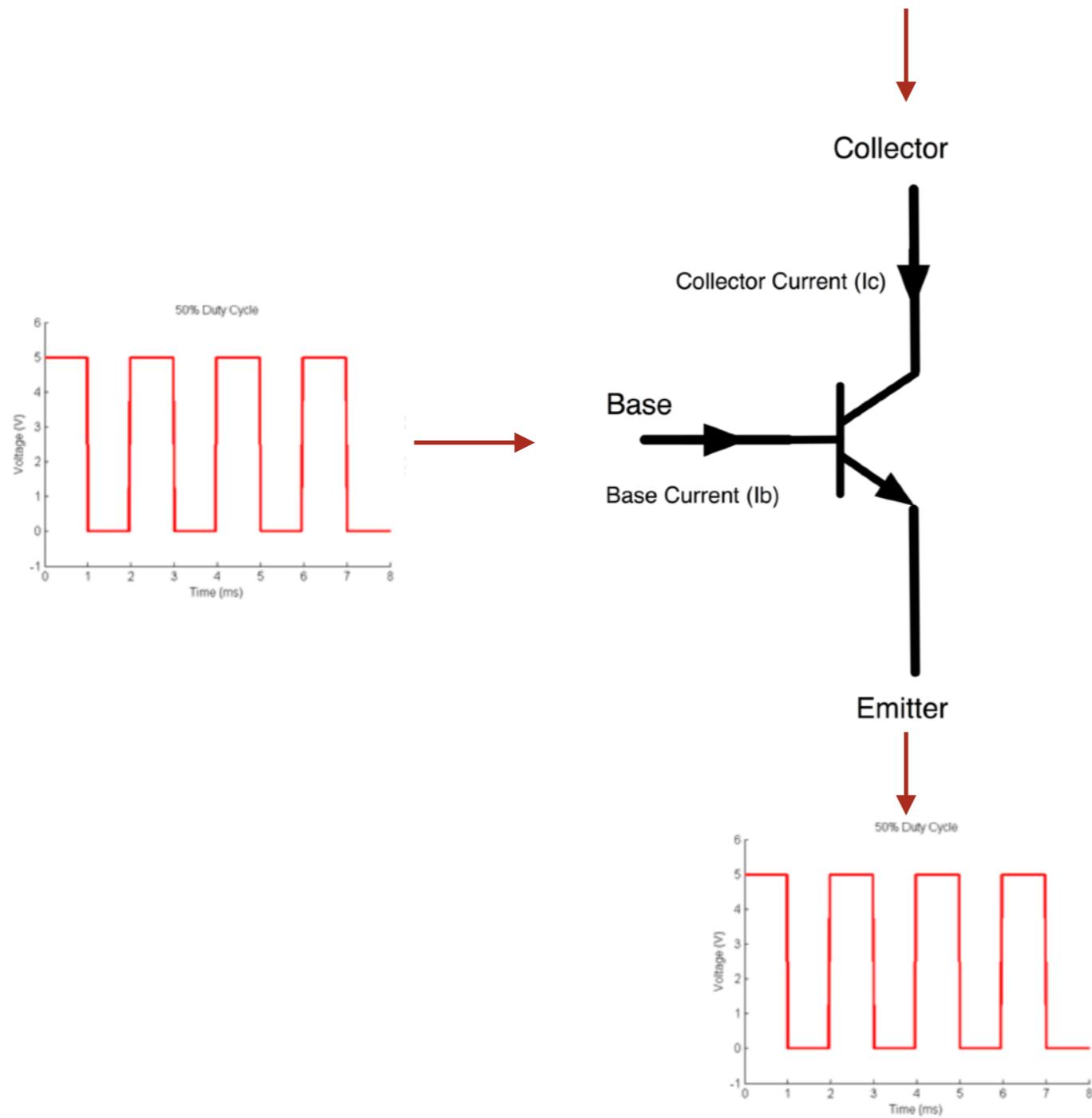
- 트랜지스터를 **electrically controlled switch**로 사용
- BJT (접합형 트랜지스터)는 3개의 핀: **emitter(E), collector(C), base(B)**



- 전류는 **collector**에서 **emitter**로 흐름
- base**에 일정값 이상의 전압(5V)이 걸릴때만 C에서 E로 전류가 흐름
- base** 핀에 **PWM** 출력을 이용하여 모터의 속도를 조절

트랜지스터

5V 혹은 그 이상의 외부 배터리



DC모터를 직접 arduino의
디지털 출력핀에 연결하면 ?

Arduino Uno Current Limitations

<http://playground.arduino.cc/Main/ArduinoPinCurrentLimitations>

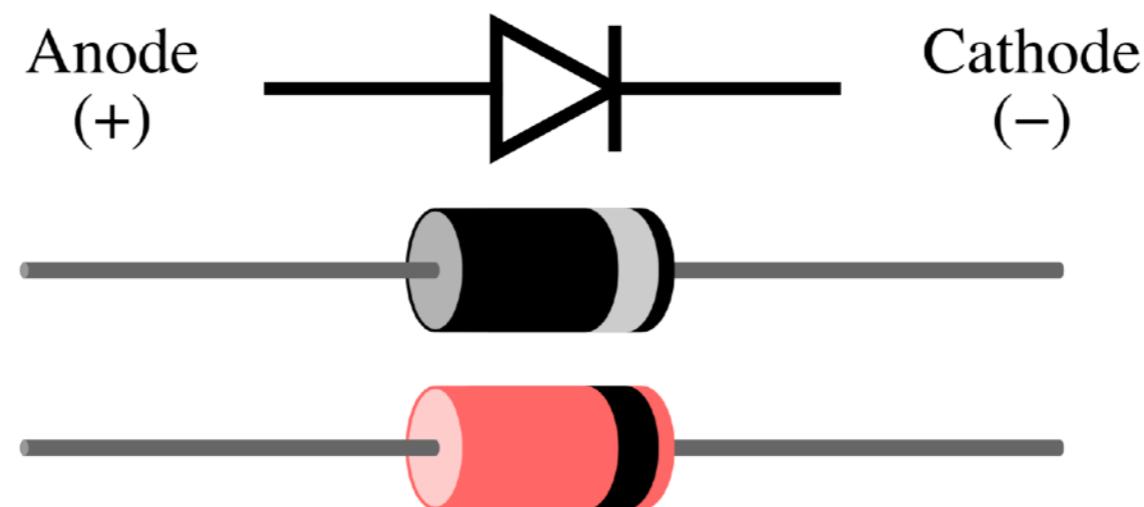
- DC Current per I/O Pin: 40.0 mA
- DC Current VCC and GND Pins: 200.0 mA



TECHNICAL DETAILS

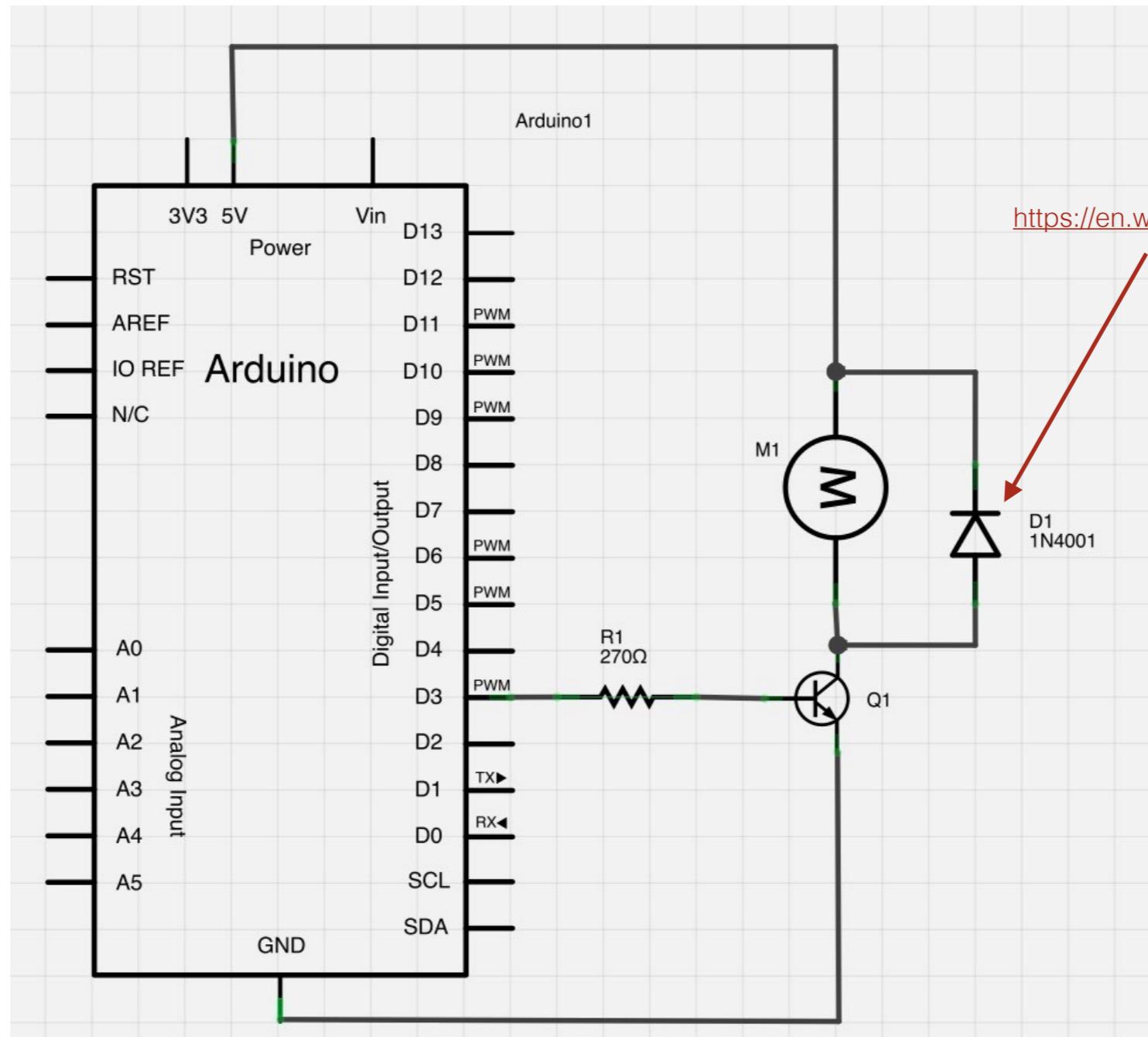
- Operating Temperature: -10°C ~ +60°C
- Rated Voltage: 6.0VDC
- Rated Load: 10 g*cm
- No-load Current: 70 mA max
- No-load Speed: 9100 ±1800 rpm
- Loaded Current: 250 mA max
- Loaded Speed: 4500 ±1500 rpm
- Starting Torque: 20 g*cm
- Starting Voltage: 2.0
- Stall Current: 500mA max
- Body Size: 27.5mm x 20mm x 15mm
- Shaft Size: 8mm x 2mm diameter
- Weight: 17.5 grams

- **다이오드(diode)**는 한 방향으로만 전류가 흐르는 것을 허용하는 소자



- DC 모터가 **inductor**와 같은 효과를 가짐. 따라서 DC의 전원이 끊기는 순간 일시적으로 배터리와 같은 역할을 하게됨
- 이 전류가 다른 회로로 흐르는 것을 방지하기 위한 용도로 다이오드를 사용

실습 01: DC 모터



flyback diode

https://en.wikipedia.org/wiki/Flyback_diode

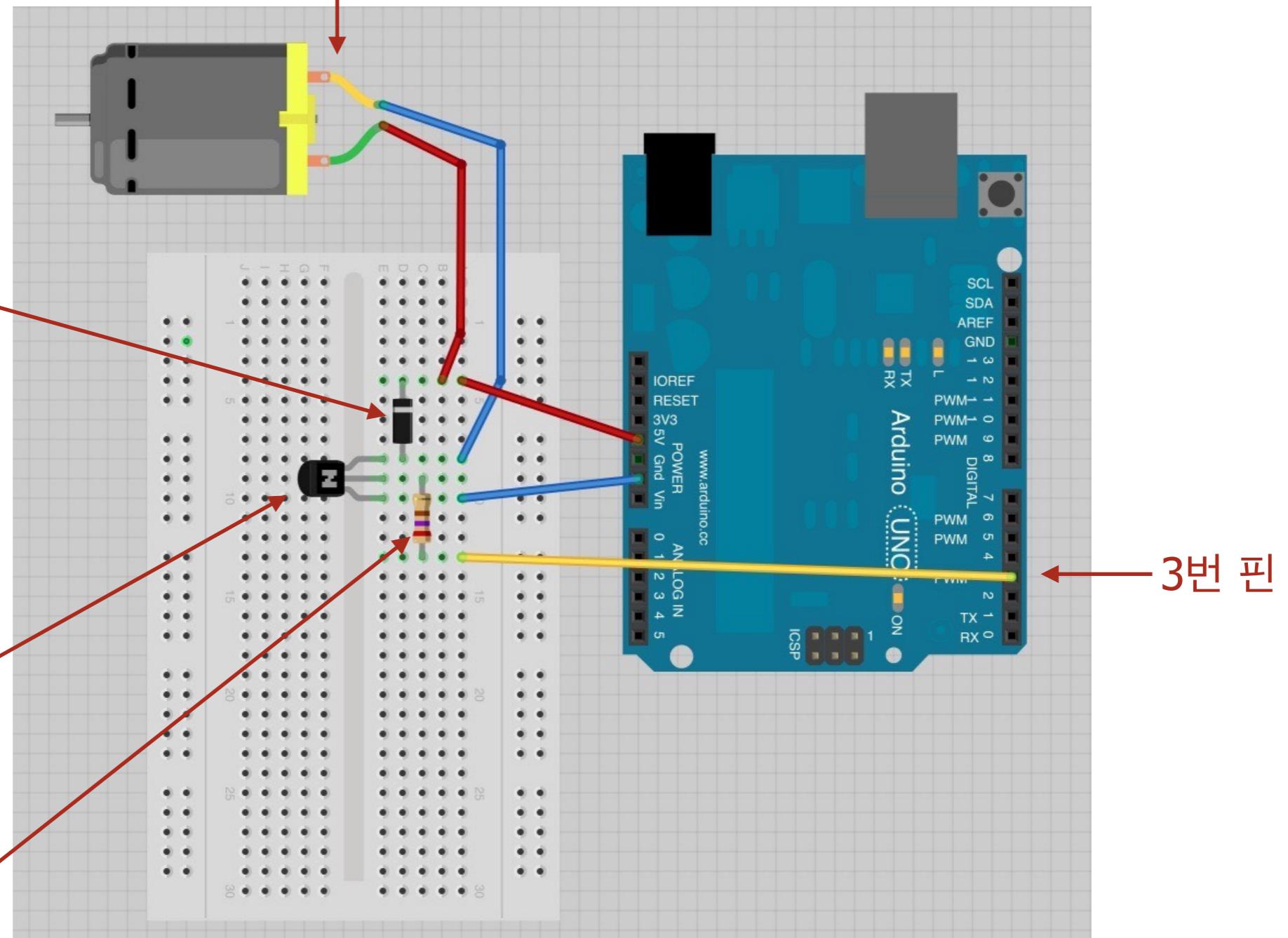
실습 01: DC 모터

모터에 연결되는 핀은 극성이 없음
(반대로 연결하면 반대로 회전함)

다이오드는
회색 띠를 보고
연결 방향을 구분

다음 페이지 참조

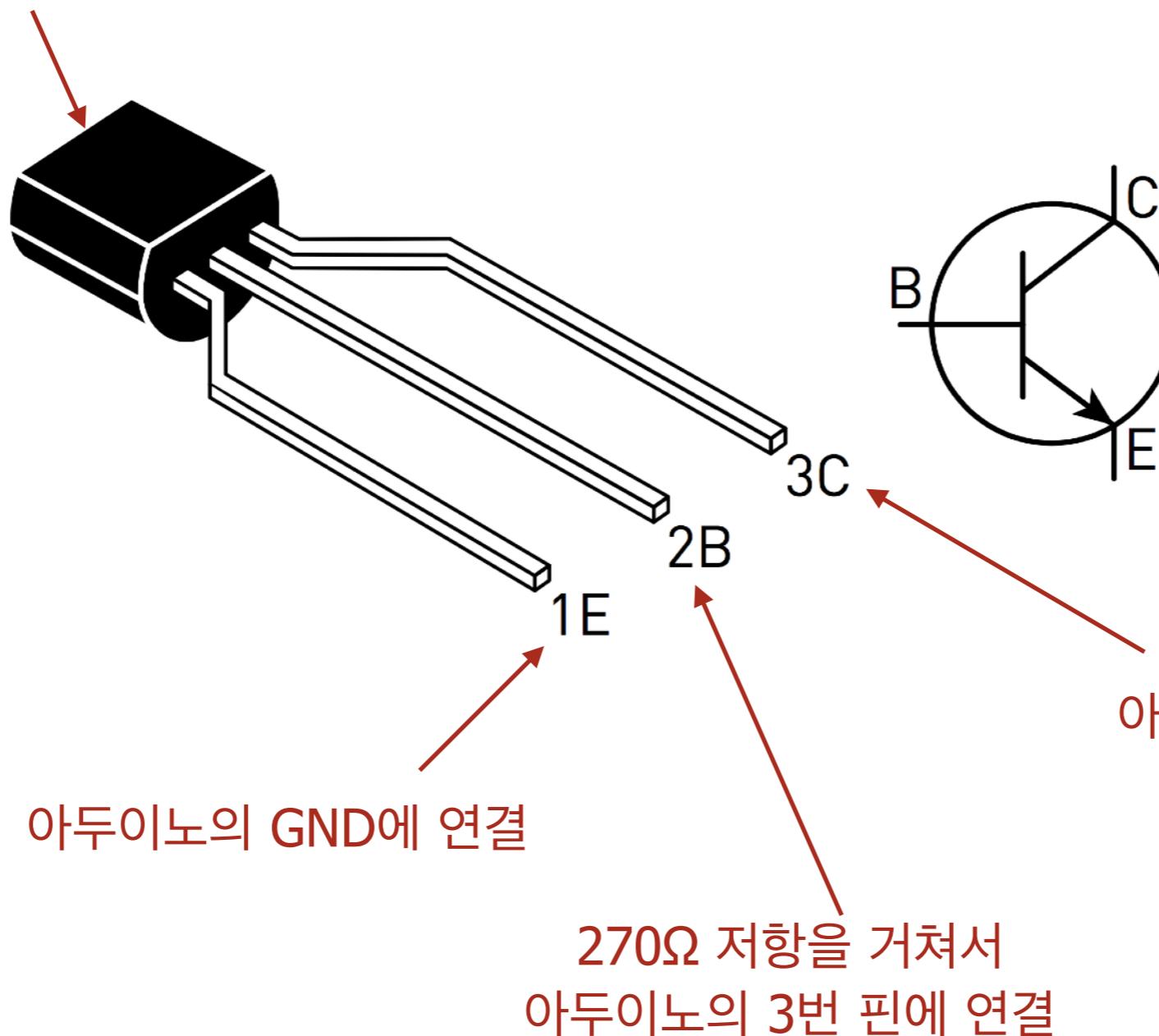
270Ω 저항



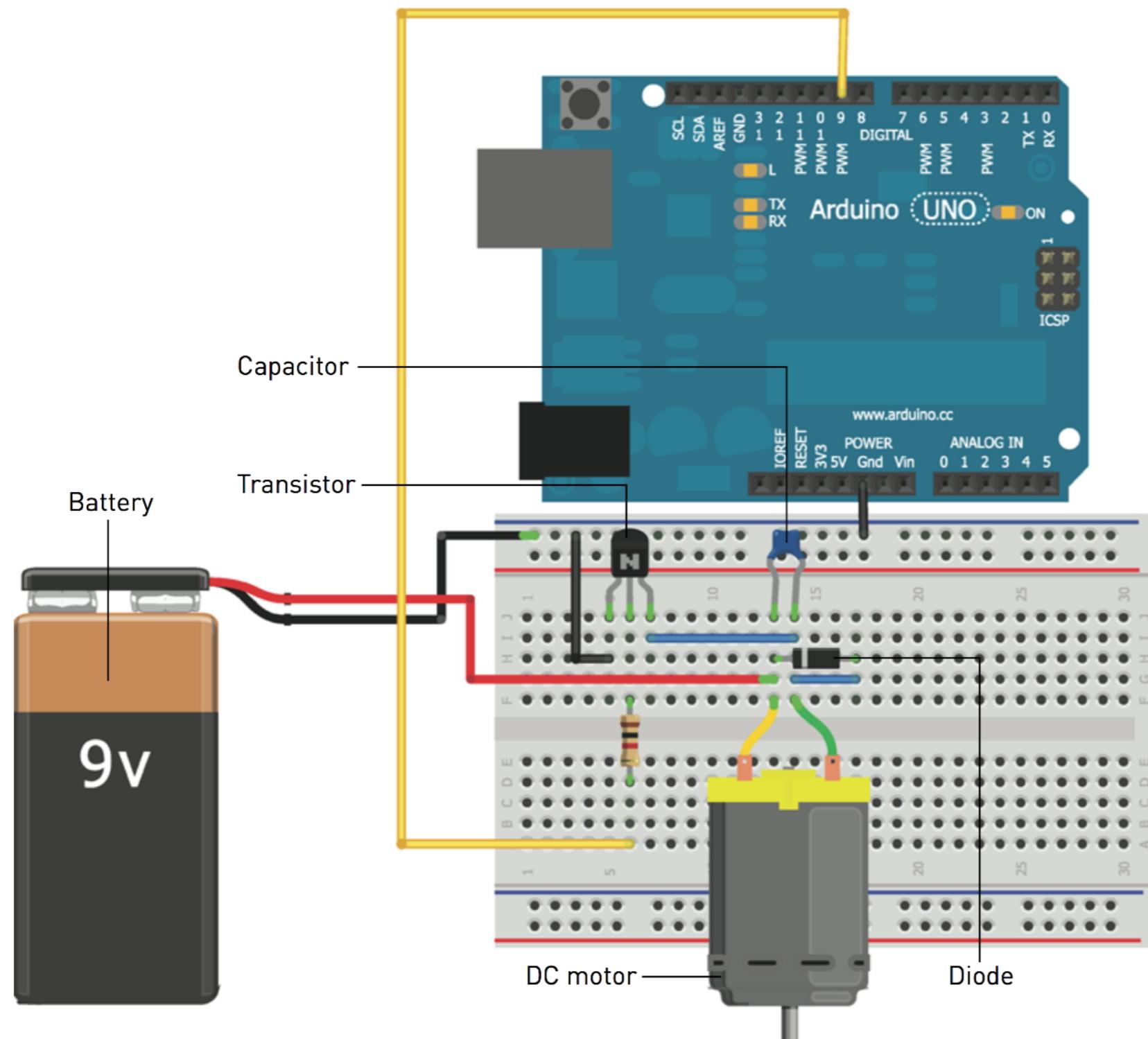
트랜지스터 연결

평평한 면을 위쪽으로

놓았을 때



참조: 외부 배터리를 사용하는 경우



Sketch 01: DC motor

```
int motorPin = 3;

void setup()
{
    pinMode(motorPin, OUTPUT);
    Serial.begin(9600);
    while (! Serial);
    Serial.println("Speed 0 to 255");
}

void loop()
{
    if (Serial.available())
    {
        int speed = Serial.parseInt();
        if (speed >= 0 && speed <= 255)
        {
            analogWrite(motorPin, speed);
        }
    }
}
```

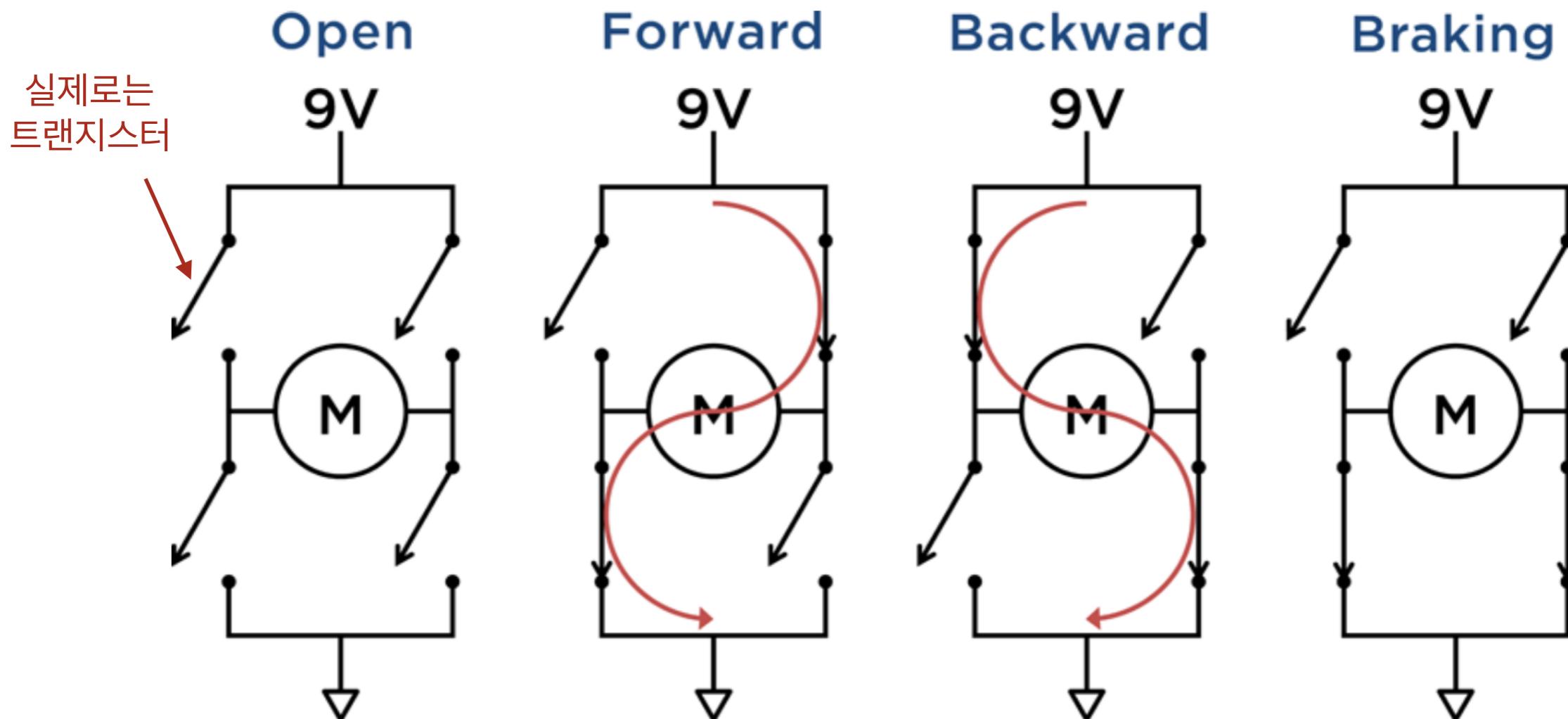
모터를 한 방향으로만
구동할 수 있다.

실습 과제 01: potentiometer를 이용해 모터의 속도 조절

- ☞ 시리얼 입력 대신 **potentiometer**를 이용해 모터의 속도를 조절하라.

H-Bridge를 사용하여 DC Motor 방향 제어하기

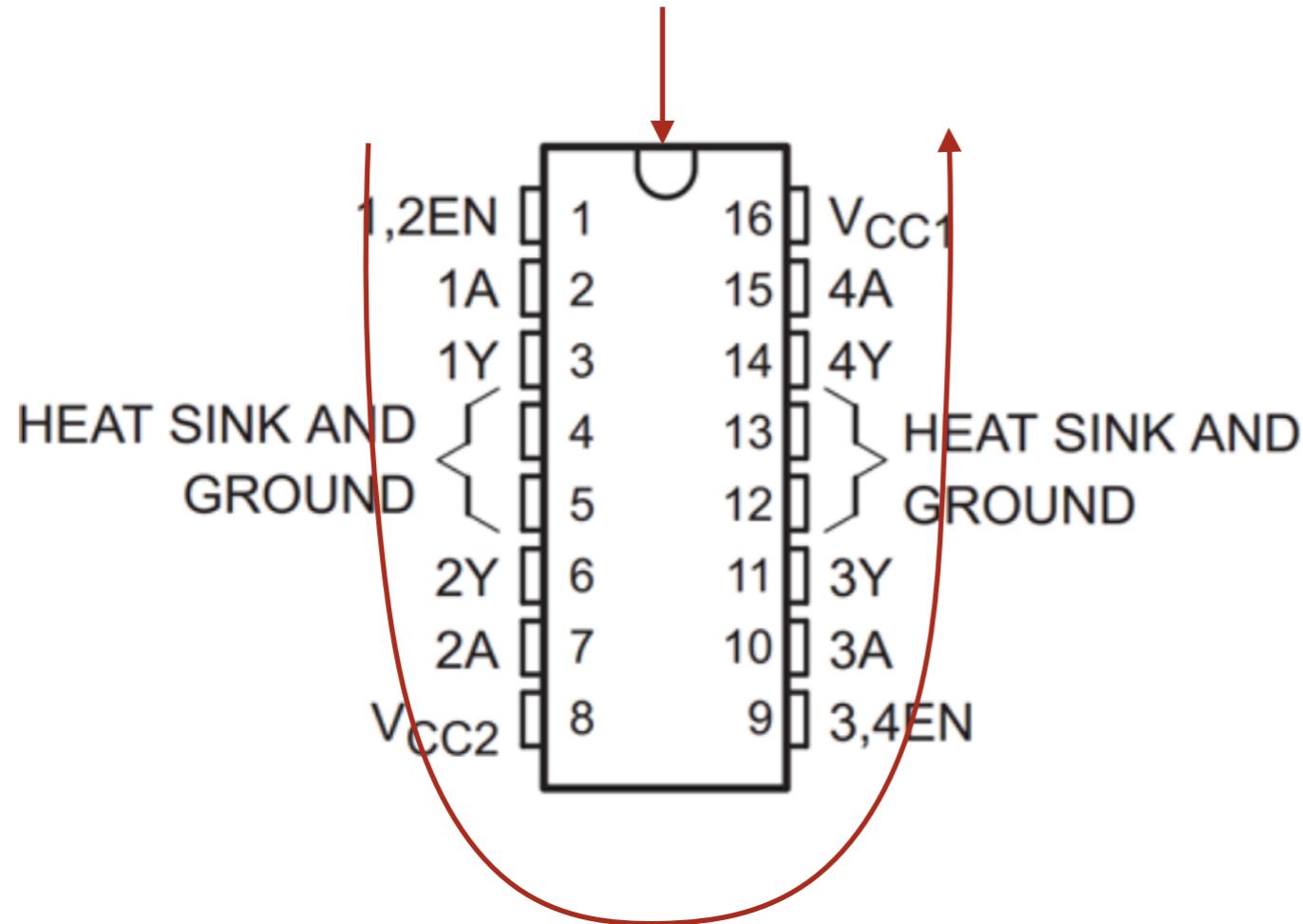
H-bridge



내부적으로 4개의 스위치를 사용하여 모터의 구동 방향을 제어. 보통
“모터 드라이버”라고 부르기도 함
(일반적으로 protection diode도 포함)

H-bridge pin-out and logic table

1번 핀의 위치한 방향은 이렇게 반원이나
혹은 점 등 어떤 식으로든 표시된다.

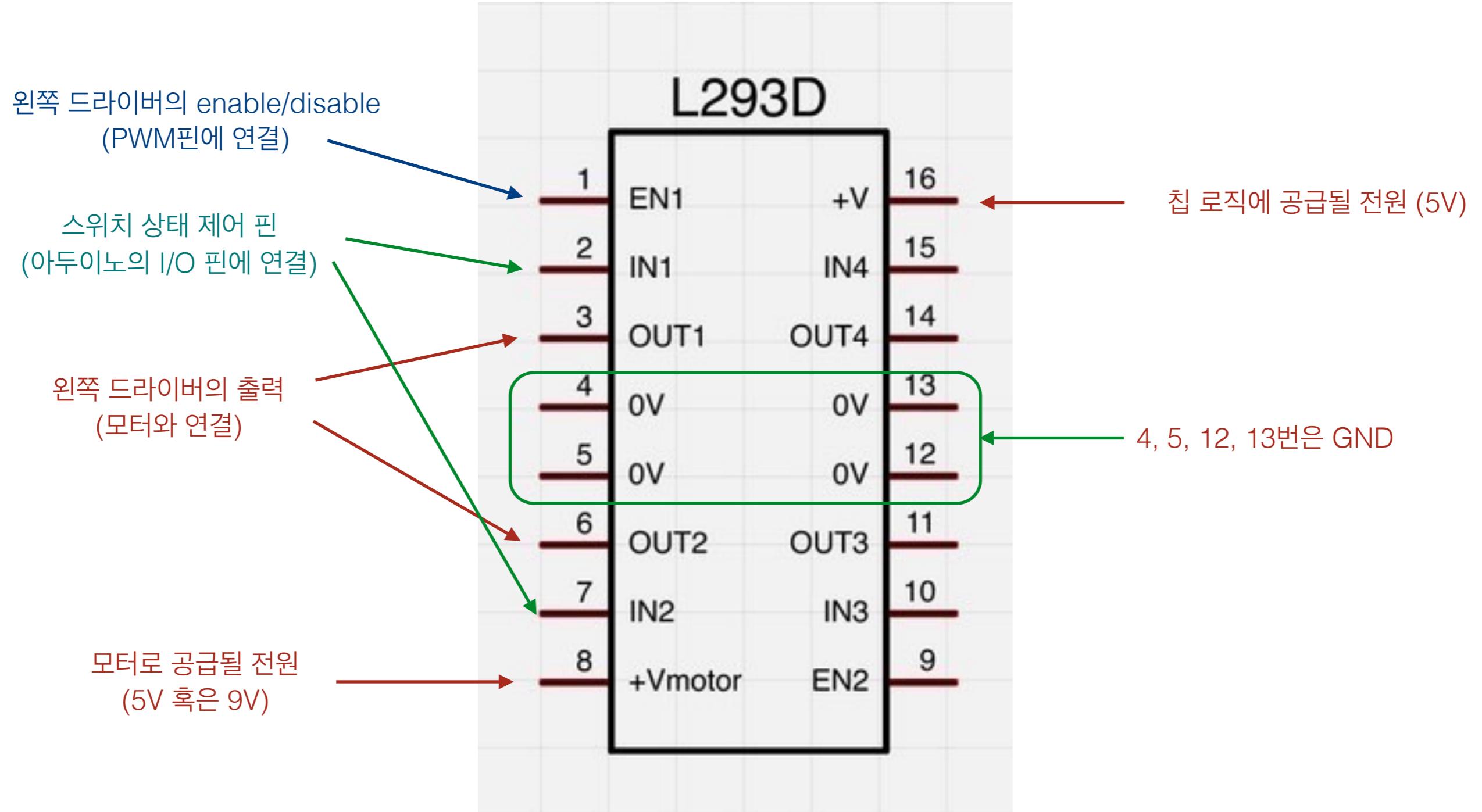


대부분의 칩에서 핀 번호는
이렇게 왼쪽 위에서 시작하여 반시계방향으로 매겨진다.

1,2EN	1A	2A	State
L	-	-	disable
H	L	L	brake
H	L	H	reverse
H	H	L	forward

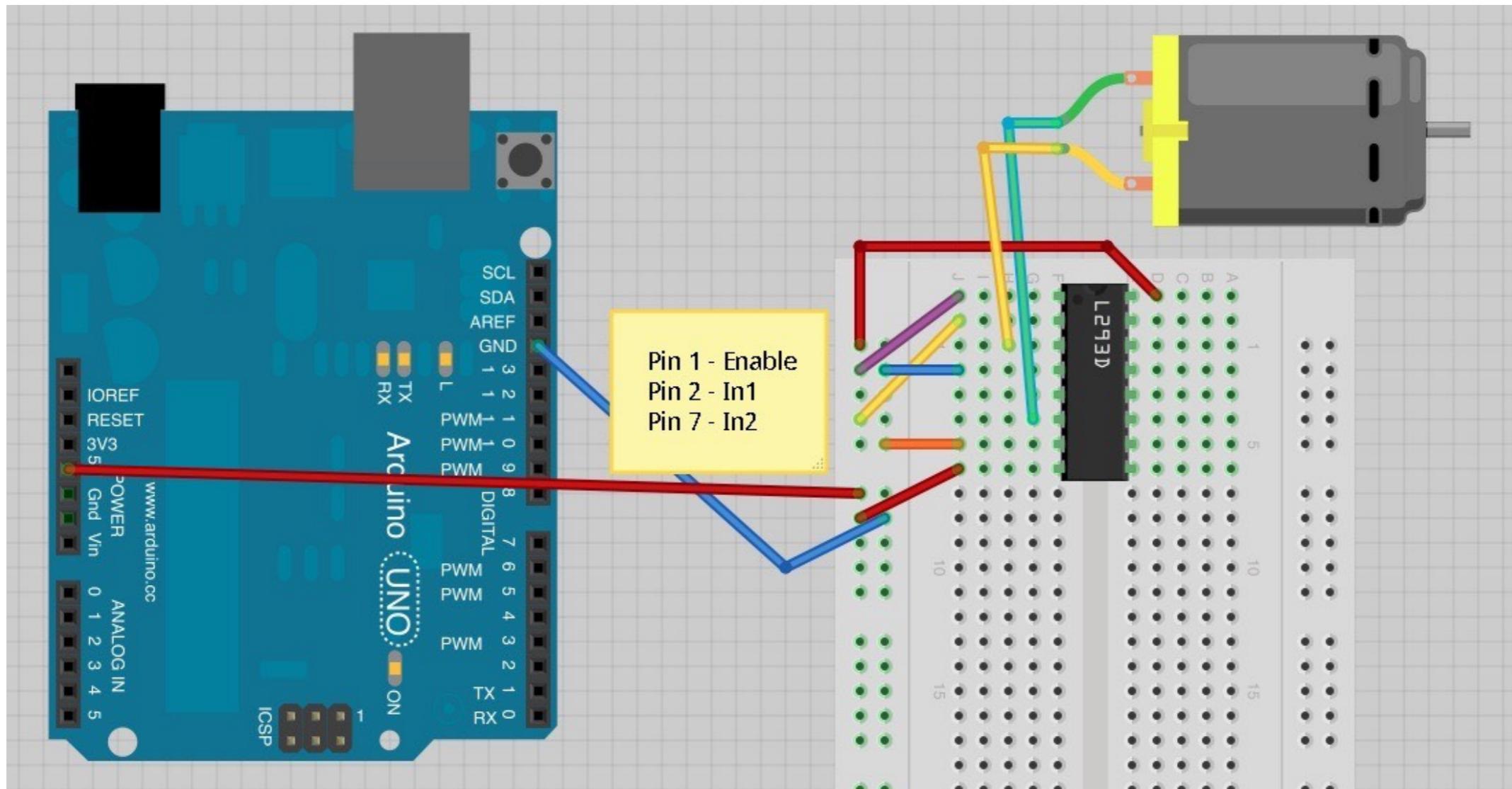
보통 모터 드라이버 칩은 2개의 H-bridge로 구성된다. 즉 2개의 DC 모터를 구동할 수 있다.

H-bridge pin-out and logic table



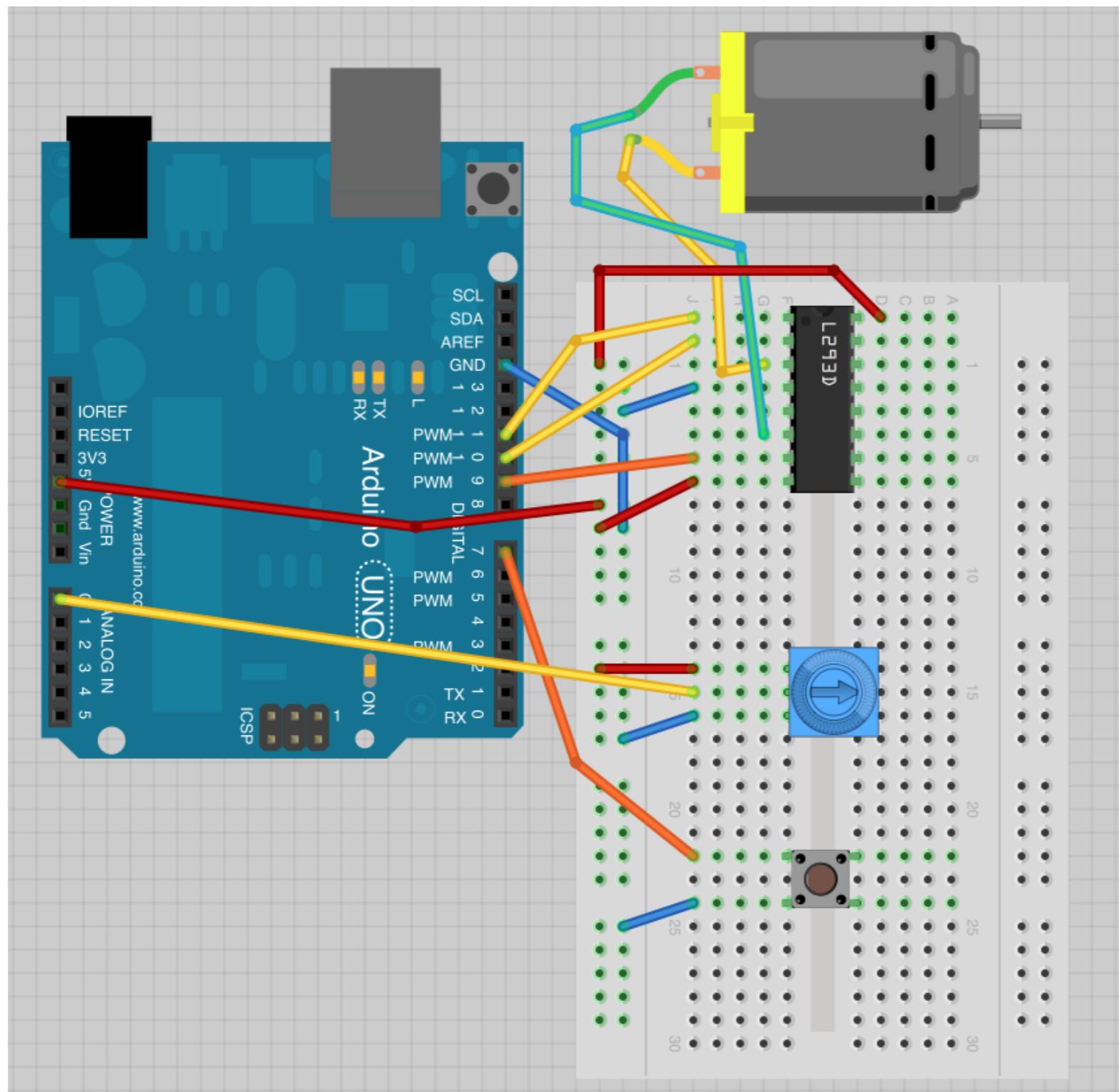
1번 핀의 위치한 방향은 이렇게 반원이나 혹은 점 등 어떤 식으로든 표시된다.

실습 02 (스케치 없음)



1. 핀 1을 GND에 연결하면 다른 핀이 어떻게 연결되든 상관없이 모터가 stop됨
2. 핀 2와 7을 각각 5V와 GND에 바꾸어가면서 연결해본다 (즉 둘 다 5V, 둘 다 GND, 한 쪽은 5V 다른쪽은 GND 등)

실습 03



Sketch 02: H-bridge

```
int enablePin = 11;
int in1Pin = 10;
int in2Pin = 9;
int switchPin = 7;
int potPin = 0;

void setup()
{
    pinMode(in1Pin, OUTPUT);
    pinMode(in2Pin, OUTPUT);
    pinMode(enablePin, OUTPUT);
    pinMode(switchPin, INPUT_PULLUP);
}

void loop()
{
    int speed = analogRead(potPin) / 4;
    boolean reverse = digitalRead(switchPin);
    setMotor(speed, reverse);
}

void setMotor(int speed, boolean reverse)
{
    analogWrite(enablePin, speed);
    digitalWrite(in1Pin, !reverse);
    digitalWrite(in2Pin, reverse);
}
```

Sketch 03: H-bridge 2

```
//Hbridge Motor Control
const int EN=11; //Half Bridge 1 Enable
const int MC1=10; //Motor Control 1
const int MC2=9; //Motor Control 2
const int POT=0; //POT on Analog Pin 0

int val = 0; //for storing the reading from the POT
int velocity = 0; //For storing the desired velocity (from 0-255)

void setup()
{
    pinMode(EN, OUTPUT);
    pinMode(MC1, OUTPUT);
    pinMode(MC2, OUTPUT);

    brake(); //Initialize with motor stopped
}
```

Sketch 03: H-bridge 2

```
void loop() {
    val = analogRead(POT);
    //go forward
    if (val > 562)
    {
        velocity = map(val, 563, 1023, 0, 255);
        forward(velocity);
    }
    //go backward
    else if (val < 462)
    {
        velocity = map(val, 461, 0, 0, 255);
        reverse(velocity);
    }
    //brake
    else
    {
        brake();
    }
}
```

Sketch 03: H-bridge 2

```
//Motor goes forward at given rate (from 0-255)
void forward (int rate)
{
    digitalWrite(EN, LOW);
    digitalWrite(MC1, HIGH);
    digitalWrite(MC2, LOW);
    analogWrite(EN, rate);
}

//Motor goes backward at given rate (from 0-255)
void reverse (int rate)
{
    digitalWrite(EN, LOW);
    digitalWrite(MC1, LOW);
    digitalWrite(MC2, HIGH);
    analogWrite(EN, rate);
}

//Stops motor
void brake ()
{
    digitalWrite(EN, LOW);
    digitalWrite(MC1, LOW);
    digitalWrite(MC2, LOW);
    digitalWrite(EN, HIGH);
}
```

실습 과제 02: potentiometer를 이용해 모터의 속도 조절

- 버튼을 토클(toggle) 스위치로 이용하여 누를 때마다 모터가 “정방향”, “정지”, “순방향”, “정지”의 네 가지 상태로 스위치되도록 스케치를 작성하라.

Driving Servo Motors

④ DC 모터 :

- 주변에서 흔히 보는 모터. 입력 전류(+, -) 방향으로 회전 방향을 제어할 수 있음.
- 상대적으로 고회전에 유리. 회전 움직임을 사용하는 RC카, 쿼드콥터 등.
- 회전수와 방향을 자유롭게 제어하기 위해서는 별도의 드라이버 모듈이 필요.

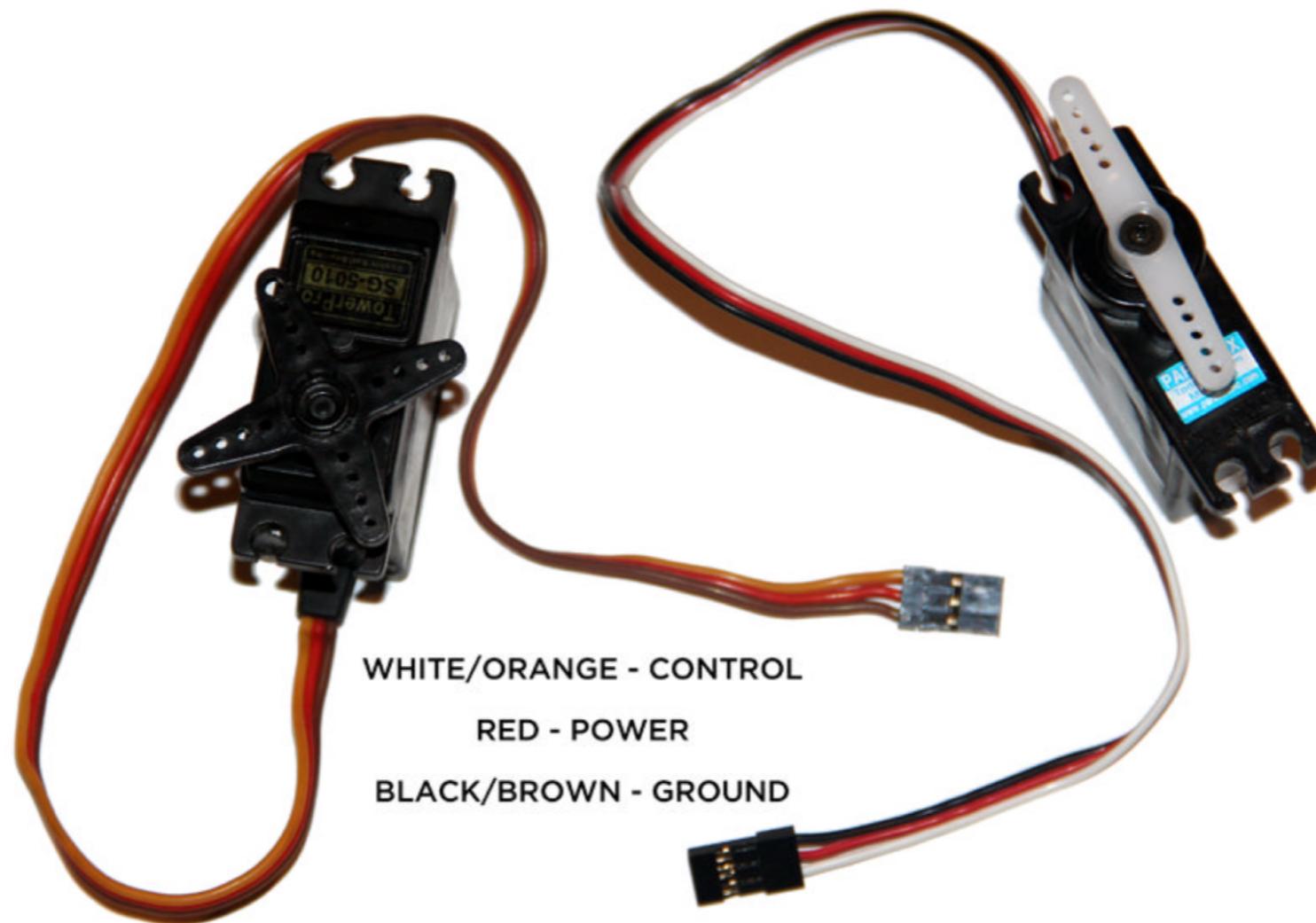
⑤ 서보 모터 :

- 보통 0~180 사이를 움직이며, 해당 회전 범위 안에서의 위치를 사용자가 설정 가능.
- 동작 범위가 제한적이지만 정확한 위치 제어가 가능. 제어 방법도 간단함. (PWM 신호로 간단히 위치제어)
- RC카의 방향타, 로봇 관절 등 회전각 제어가 필요한 곳에 광범위하게 사용.

⑥ 스텝 모터 :

- 회전 방향과 속도 뿐 아니라 회전각을 정밀히 제어할 수 있음.
- DC 모터와 서보 모터의 장점을 합친 모터. 하지만 제어가 복잡함. 그래서 보통 스텝모터 드라이버 모듈을 이용해서 제어.
- 상대적으로 고회전이 필요치 않으면서 정밀한 제어가 필요한 곳에 사용. 3D 프린터 움직임을 만드는 핵심 모터이기도 함.

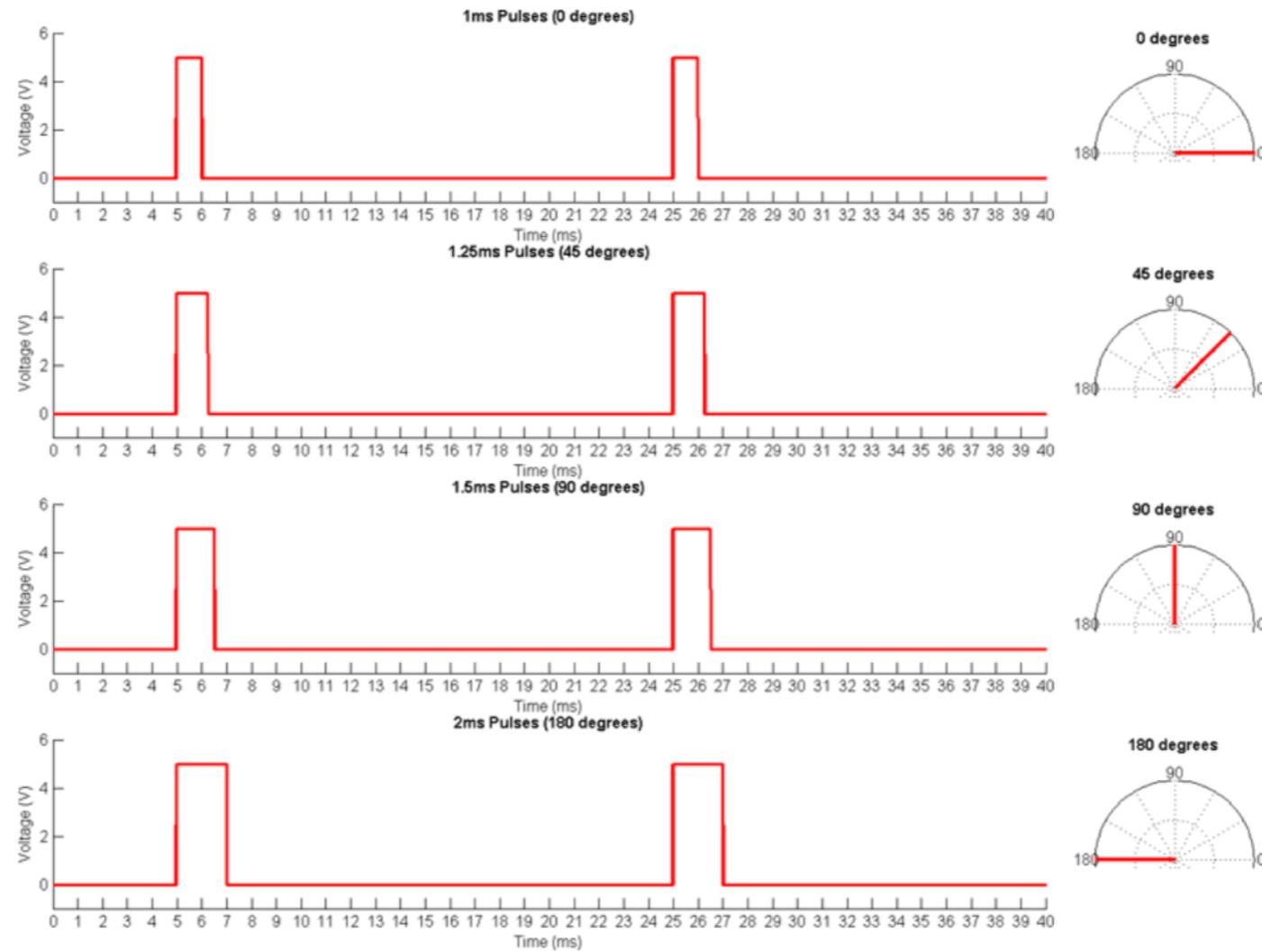
Servo Motors



3 pins:

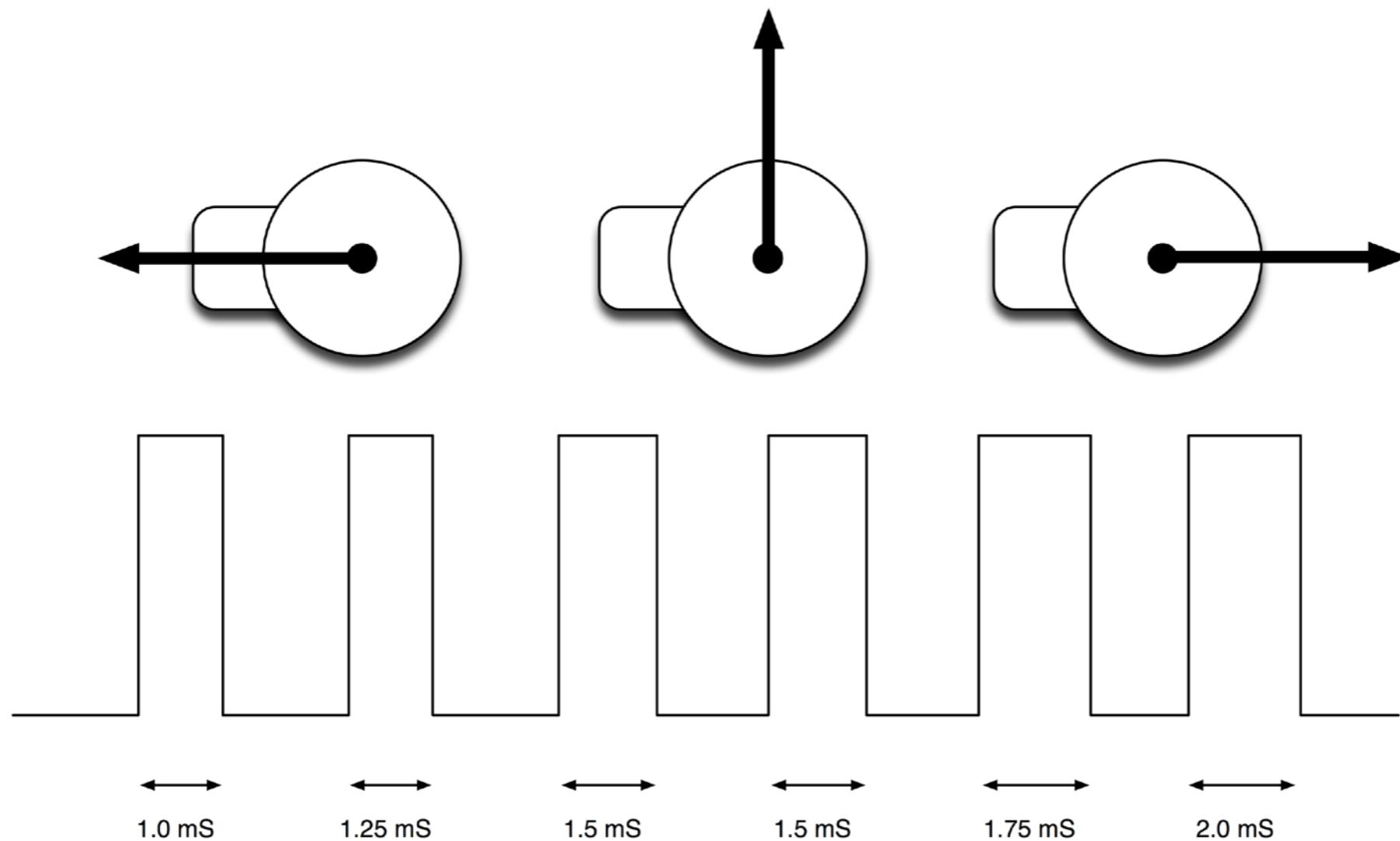
- power (usually red)
- ground (usually brown or black)
- control signal (usually white or orange)

Control Signal



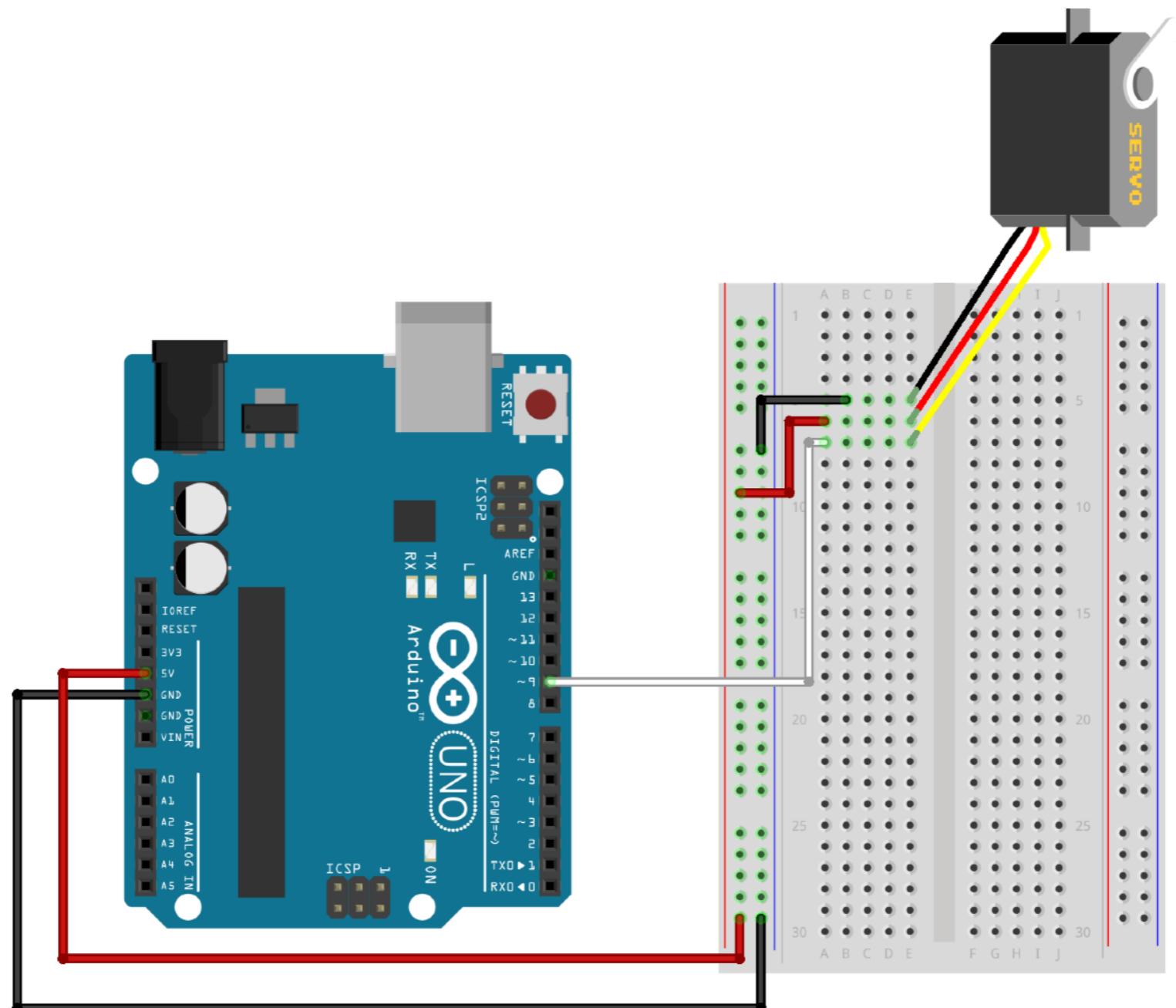
- ◎ 20ms 주기로 펄스를 전송
- ◎ 펄스의 길이를 최소 1ms에서 최대 2ms까지 변화시켜서 회전 각도를 0도에서 180도까지 지정

Control Signal



저렴한 servo의 경우 실제로는 대략 170도 정도만 돌아가기도 함
(최대 회전 각도를 실험적으로 알아내는게 좋음)

실습 04: Servo 제어



Sketch 04: 간단한 Servo 제어

Servo 라이브러리를 include한다.

```
#include <Servo.h>
```

```
int servoPin = 9;
```

```
Servo servol;
```

```
void setup()
```

```
{
```

```
    servol.attach(servoPin);
```

Servo 객체를 9번 핀과 연결한다.

```
}
```

```
void loop()
```

```
{
```

```
    int position;
```

```
    servol.write(90);
```

Servo가 90도를 가리키도록 한다.

```
    delay(1000);
```

```
    servol.write(180);
```

Servo가 180도를 가리키도록 한다.

```
    delay(1000);
```

```
    servol.write(0);
```

Servo가 0도를 가리키도록 한다.

```
    delay(1000);
```

```
    for(position = 0; position < 180; position += 2) {
```

```
        servol.write(position);
```

```
        delay(20);
```

```
}
```

```
    for(position = 180; position >= 0; position -= 1) {
```

```
        servol.write(position);
```

```
        delay(20);
```

```
}
```

```
}
```

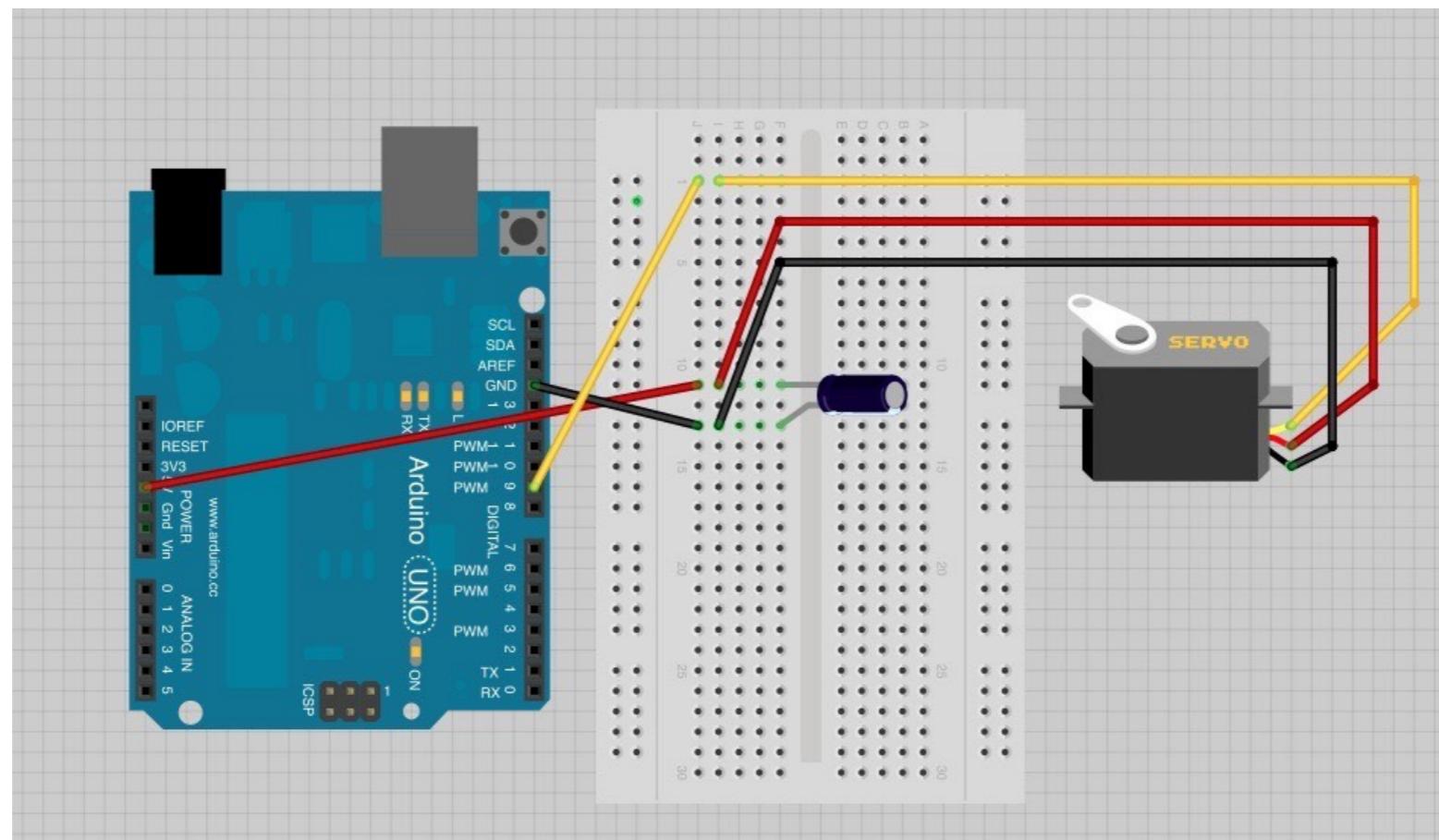
Servo 모터를 9번 PWM 핀에 연결한다.

Servo 객체를 생성한다.

Servo가 지정된 각도로 움직이려면 시간여유가 필요하다.

만약 서보가 제대로 작동하지 않으면

- 서보의 동작이 시작될 때 일시적으로 많은 전력이 소모됨. 이로 인해 아두이노 보드에 공급되는 전압이 순간적으로 낮아져 보드가 **reset**될 수 있음
- 이런 경우 **high value capacitor (470uF or greater)**를 **GND와 5V 사이에 배치**하여 해결한다. **capacitor**가 충전기의 역할을 함
- capacitor**를 연결할 때 긴 다리를 **5V쪽에**, 짧은 다리를 **GND 쪽에 연결함**

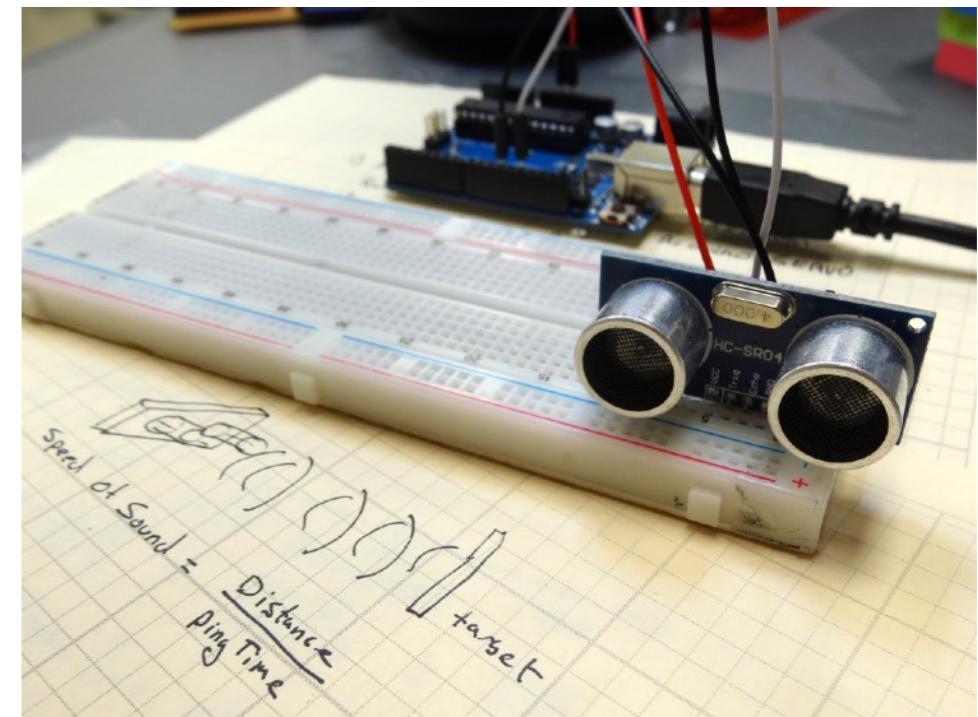
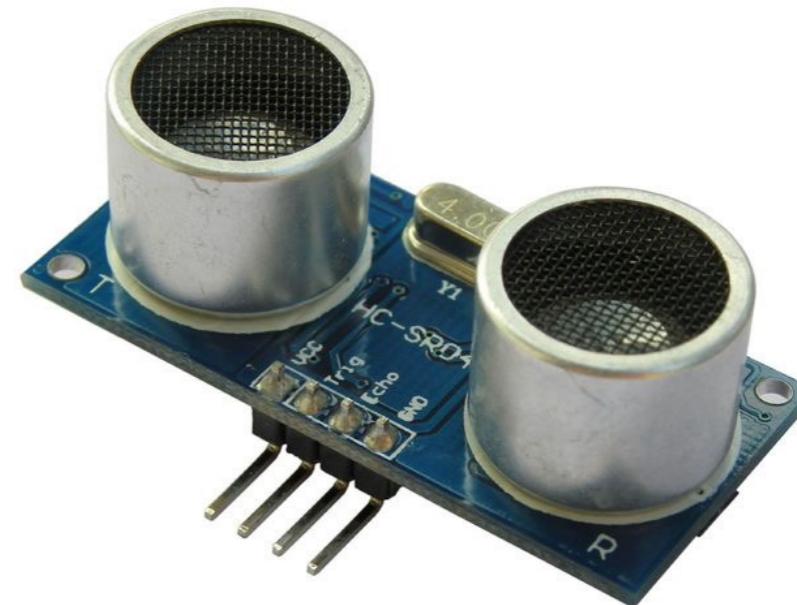
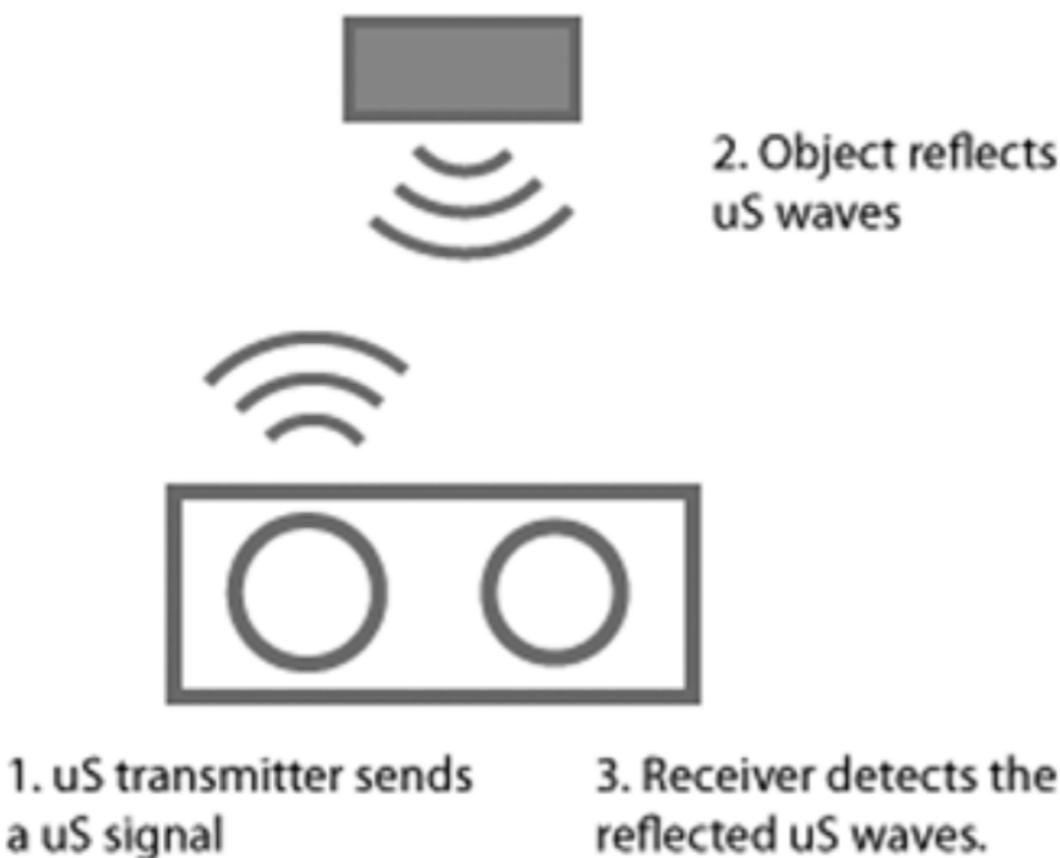


실습 과제 03: 시리얼 모니터로부터 서보의 각도 입력

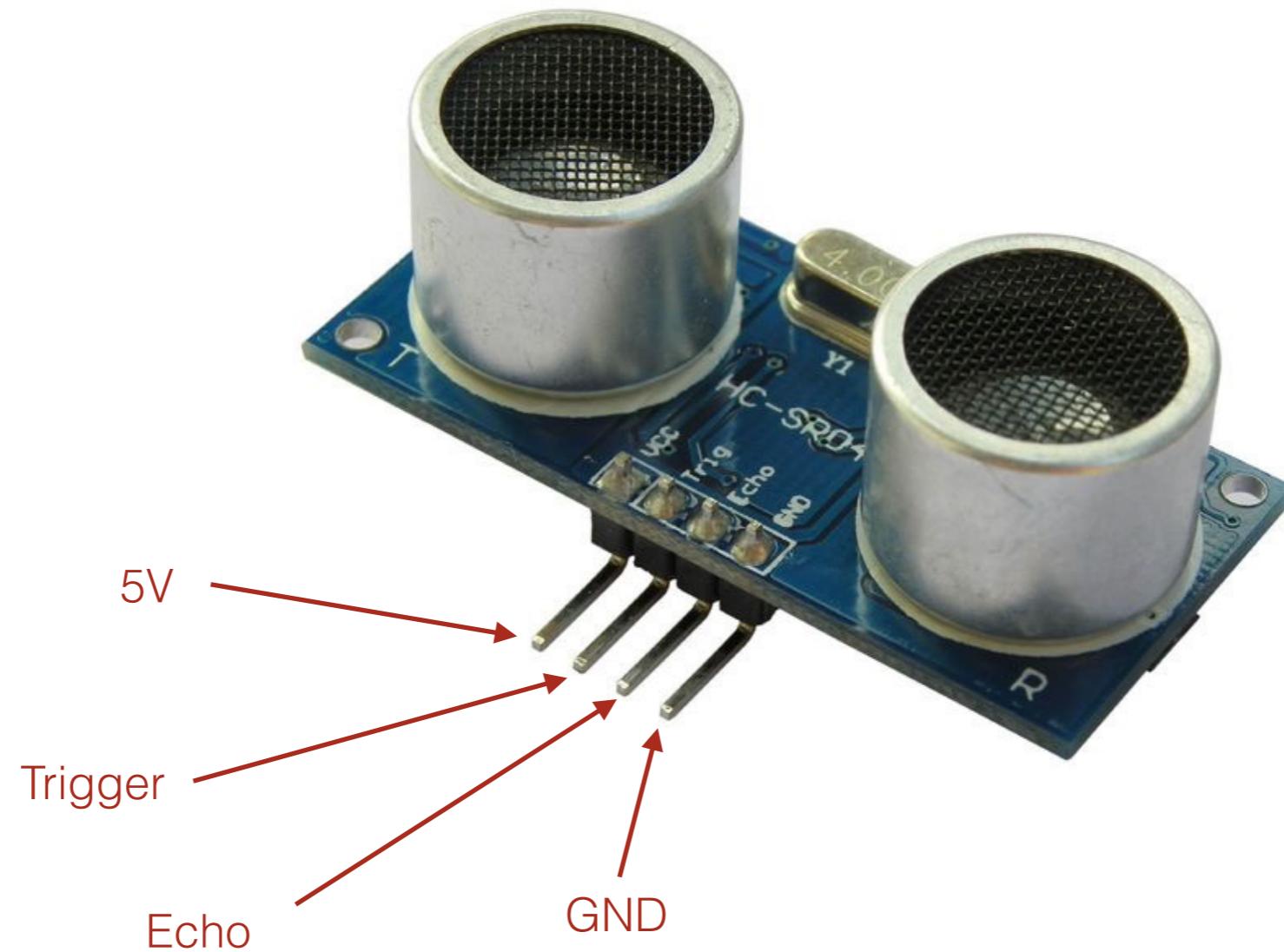
- 서보의 각도값을 시리얼 모니터에서 입력하도록 하라. 이 경우 시리얼 모니터에서 전송한 값을 정수로 변환해주는 **Serial.parseInt()** 함수를 사용하라.

Sweeping Distance using Servo and Ultrasonic Sensor

초음파센서(ultrasonic sensor)



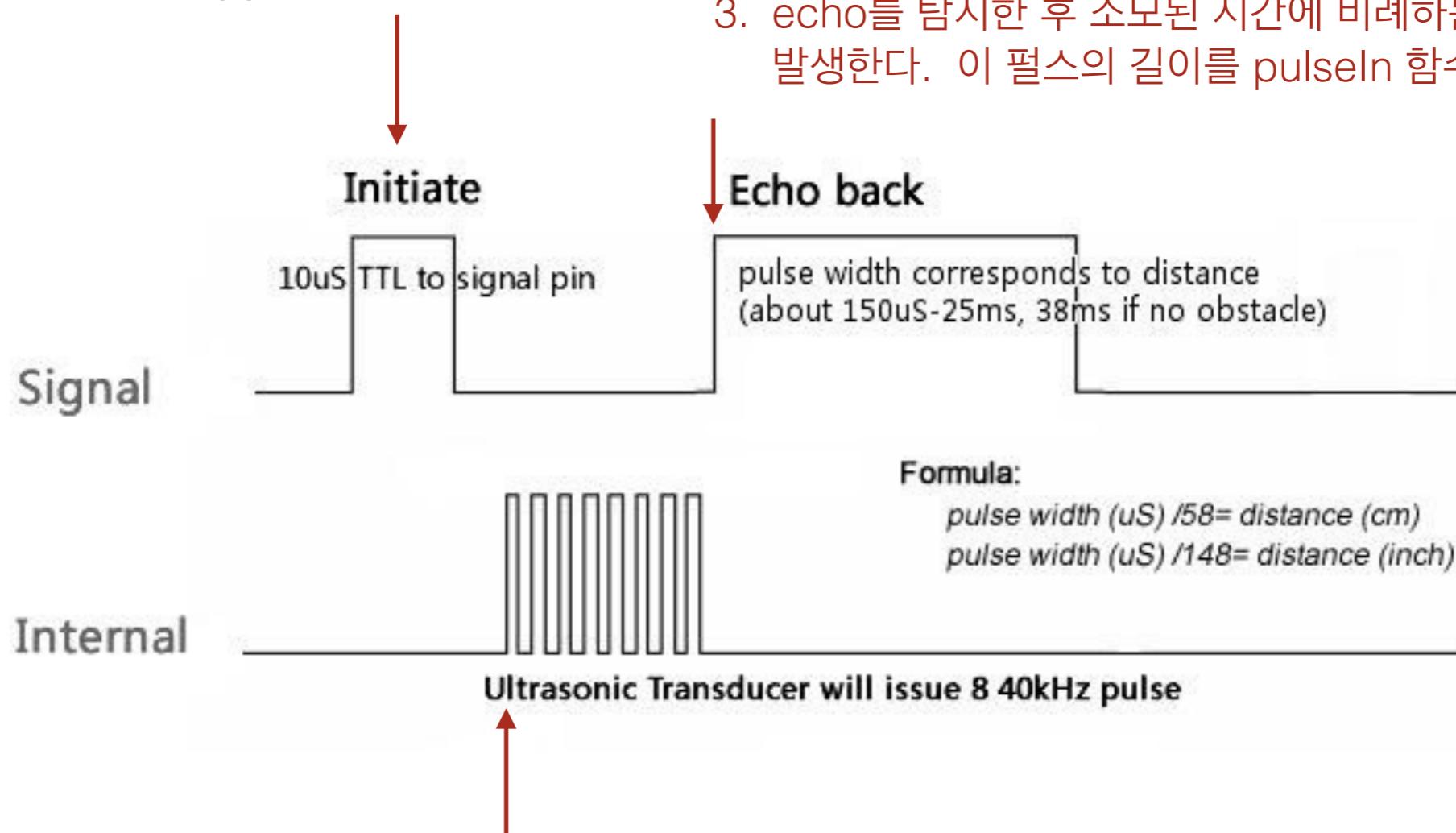
초음파센서(ultrasonic sensor)



센서에 따라서 trigger와 echo가
하나의 핀을 공유하는 경우도 있음

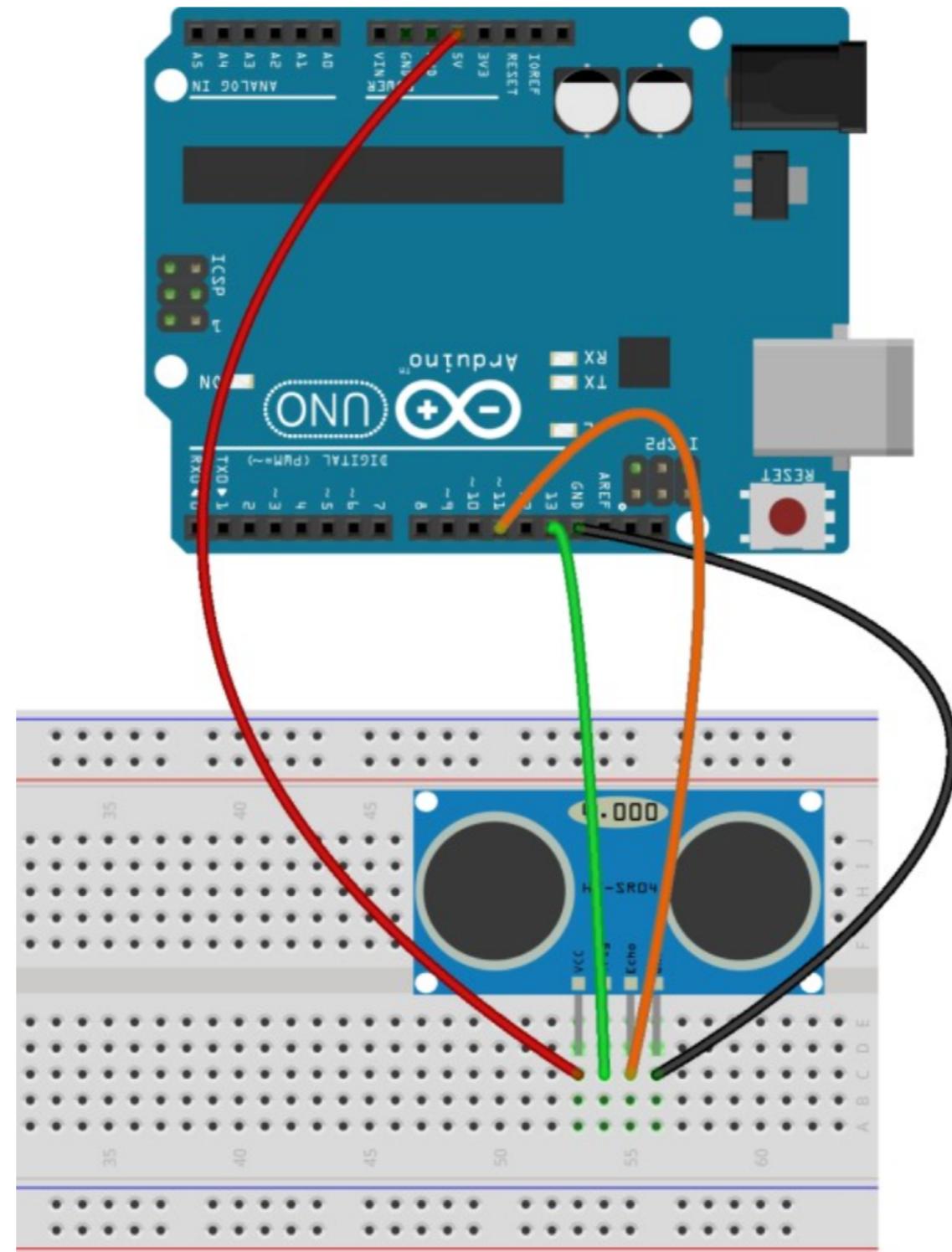
초음파센서(ultrasonic sensor)

1. Trigger 핀으로 10us 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.

초음파센서



Sketch 05: Ultrasonic Range Finder

```
int trigPin=13; //Sensor Trip pin connected to Arduino pin 13
int echoPin=11; //Sensor Echo pin connected to Arduino pin 11
long duration;
long distance; //Distance to Target

void setup() {
  Serial.begin(9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
}

void loop() {
  digitalWrite(trigPin, LOW); //Set trigger pin low
  delayMicroseconds(2); //Let signal settle
  digitalWrite(trigPin, HIGH); //Set trigPin high
  delayMicroseconds(10); //Delay in high state
  digitalWrite(trigPin, LOW); //ping has now been sent

  duration = pulseIn(echoPin, HIGH); //duration is presented in microceconds

  distance = microsecondsToCentimeter(pingTime);
  Serial.print("The distance to the target is: ");
  Serial.print(distance);
  Serial.println(" centimeters.");
  delay(1000);
}

long microsecondsToCentimeter(long time) {
  return time/29/2; // The speed of sound is 340m/s or 29 microseconds/centimeter.
}
```

실습 과제 04: 초음파센서를 서보모터에 부착하여 스캔하기

- 아래 그림과 같이 초음파 센서를 서보모터에 부착하여 회전하면서 거리를 측정하라. **0도에서 180도까지 1도 단위로** 거리를 측정하여 시리얼 모니터로 전송하라.

