

Chapter 219. 모터

모터는 전기를 이용하여 회전력을 얻는 기기로 움직이는 장치를 구현하기 위해 필수적인 부품이다. 모터는 전기장의 변화에 따라 자기장의 변화가 발생하고 자기장에서 인력과 척력에 의해 회전 운동을 발생시키게 되며 제어 방식에 따라 여러 가지 종류의 모터가 존재한다. 마이크로컨트롤러에서는 DC 모터, 스텝핑 모터, 서보 모터 등이 많이 사용되며 각각의 모터들은 그 특징이 조금씩 다르므로 사용하고자 하는 목적에 맞는 모터를 선택하는 것이 중요하다. 각 모터의 특징은 다음과 같다.

- DC (Direct Current) 모터 : 축이 연속적으로 회전하는 형태의 모터로, 전원이 연결되면 회전하고 전원이 끊어지는 경우에 정지하는 간단한 구조로 되어 있어 제어가 간단한 장점이 있다. 정지 시에는 관성으로 인해 정확한 정지 위치를 지정하기 어렵다.
- 스텝핑 (stepping) 모터 : 전원이 공급되면 축은 일정 각도를 회전하고 멈춘다. 축을 연속적으로 회전시키기 위해서는 모터로 펄스열이 전달되어야 하며 하나의 펄스에 반응하여 회전하는 양을 분할 각도라고 한다. 분할 각도가 1도인 경우 모터를 10도 회전시키기 위해서는 10개의 펄스를 전송해주어야 한다. 스텝핑 모터는 원하는 각도만큼 회전시킬 수 있는 장점은 있지만 분할 각도 단위로만 제어된다는 단점이 있다.
- 서보 (servo) 모터 : DC 모터의 한 종류로 DC 모터에 귀환 제어 회로를 추가하여 위치 제어가 가능하도록 구성된 모터이다. 제어 회로로 인해 가격이 비싼 단점은 있지만 정밀 제어가 가능한 점은 DC 모터에 비해 우수하며 속도 면에서도 스텝핑 모터에 비해 빠른 장점이 있다.

1. DC 모터

DC 모터는 자기장 속의 도체에 전류를 흐르게 함으로써 발생하는 전자기력을 이용하여 회전력을 발생시키는 모터로 모터로서 최초로 발명된 모터에 해당한다. DC 모터는 구조가 간단하고, 제어가 쉬우며, 가격이 저렴한 등 여러 가지 장점으로 인해 널리 사용되고 있다. DC 모터는 2개의 연결선을 가지며 연결하는 전원의 극성에 따라 회전 방향이 결정된다. 또한 PWM (Pulse Width Modulation) 신호를 사용하면 간단히 속도 제어가 가능하다. 하지만 DC 모터를 사용하는 경우에는 두 가지 점에 유의하여야 한다. 첫 번째는 일반적으로 DC 모터는 많은 전력을 필요로 한다는 점이다. 아두이노의 디지털 출력 핀에 DC 모터를 연결하면 DC 모터를

제어할 수 있지만 아두이노의 디지털 출력 핀으로 출력되는 최대 전류는 40mA로 이는 소형 DC 모터를 구동하기에도 충분하지 않다. 또한 DC 모터는 전기 잡음을 유발하므로 아두이노의 디지털 출력 핀에 직접 연결하여 제어하는 것은 추천되지 않는다. 따라서 DC 모터를 사용하는 경우에는 아두이노 구동을 위한 전원과는 별도로 전용 전원을 사용하며 아두이노의 디지털 출력 핀은 전원을 on/off시키는 스위치 역할을 하도록 구성하는 것이 일반적이다. 하지만 전용 전원의 사용만으로는 충분하지 않다. DC 모터 사용에서 두 번째 문제는 DC 모터의 회전 방향이 전원을 연결하는 방향에 따라 결정되기 때문에 모터를 연결한 이후에는 회전 방향을 변경할 수 없다는 점이다. 전진만 가능한 차는 매력이 없지 않은가? 회전 방향을 제어하기 위해서는 일반적으로 H 브리지(bridge) 회로가 사용되며 아두이노의 디지털 출력 핀으로 회전 방향을 제어한다. H 브리지 회로는 그림 1에서와 같이 4개의 스위치로 표현할 수 있다. 스위치 1과 4가 연결되는 경우와 스위치 2와 3이 연결되는 경우 모터에 가해지는 전압은 반대가 되는 원리를 사용한다.

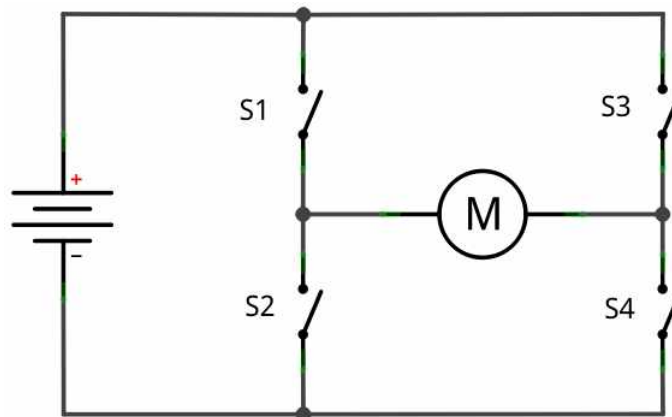


그림 1. H 브리지 회로

이러한 두 가지 문제점을 해결해줄 수 있는 칩을 모터 드라이버(motor driver)라고 한다. 모터 드라이버 칩은 H 브리지 회로를 사용하고 전용 전원을 연결할 수 있도록 만들어져 있으며 L298 칩이 흔히 사용된다. L298 칩은 아두이노의 공식 모터 쉴드에서도 사용되는 칩이지만 공식 모터 쉴드의 경우 외부 전원 연결이 직관적이지 않아 사용이 쉽지 않다. 이 장에서는 L298 칩을 사용하여 제작된 그림 2의 DC 모터 제어 모듈을 사용하여 DC 모터를 제어한다.

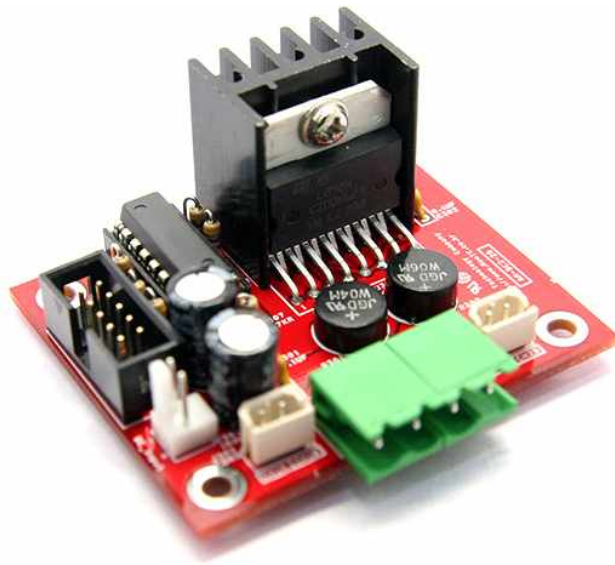
그림 2. DC 모터 제어 모듈¹⁾

그림 2의 모터 제어 모듈은 두 개의 DC 모터를 제어할 수 있으며 L298 칩 구동을 위한 5V 전원과 모터 전용의 12V 전원을 연결하여 사용한다. 각각의 모터 제어를 위해서는 3개의 제어선 연결을 필요로 하며 각 제어선의 역할은 표 1과 같다.

제어선	설명
Enable	모터 제어 활성화
DIR	모터 방향 제어
PWM	모터 속도 제어

표 1. 모터 제어 모듈 제어선

제어선 중 Enable은 입력이 LOW인 상태에서만 모터의 구동 및 제어가 가능하도록 해주며, HIGH인 상태에서 모터는 정지하게 된다. DIR은 모터의 회전 방향을 제어하기 위해 사용되며 LOW인 경우 역방향, HIGH인 경우 정방향으로 회전한다. PWM은 듀티 사이클에 따라 속도를 조절하기 위해 사용하므로 마이크로컨트롤러에서 PWM 출력이 가능한 핀에 연결하여야 한다. 모터 제어 모듈의 연결 방법은 그림 3과 같다.

1) 그림 출처: <http://www.newtc.co.kr>

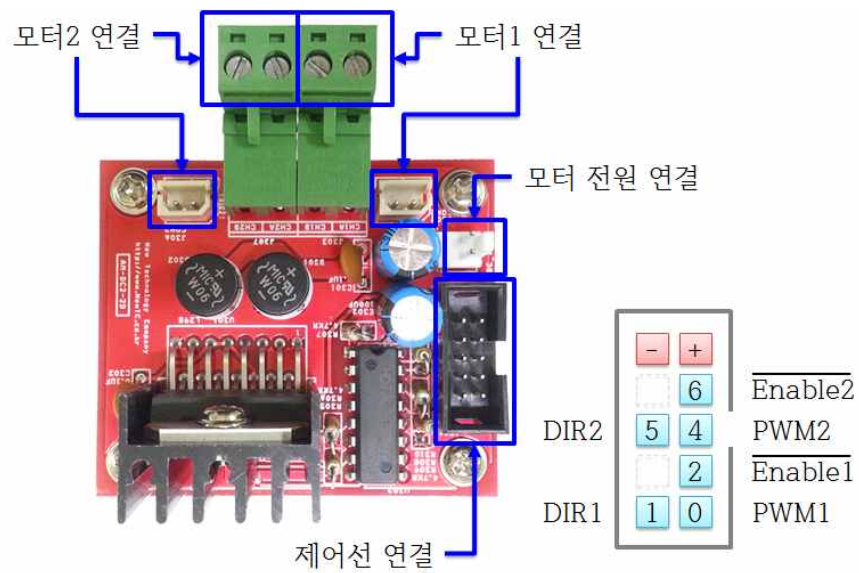


그림 3. 모터 제어 모듈 연결

그림 4와 같이 모터 제어 모듈을 사용하여 아두이노에 모터를 연결해 보자.

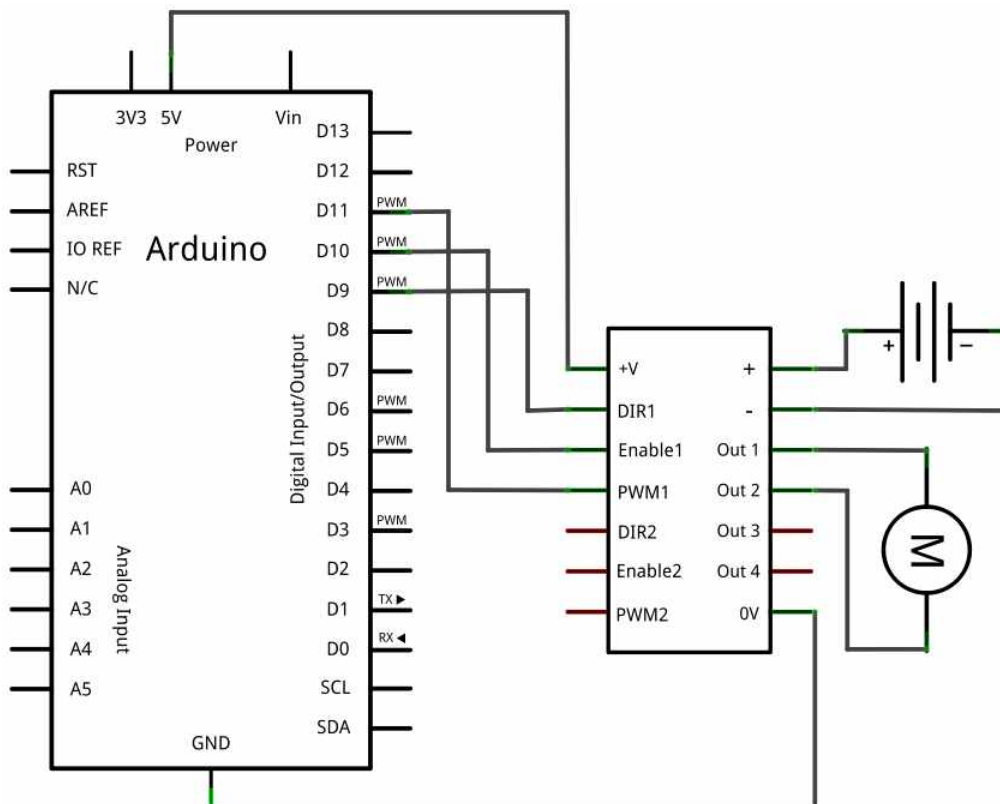


그림 4. 모터 제어 모듈 연결 회로도

스케치 1은 모터를 정방향으로 최저 속도에서 최고 속도로 서서히 속도를 증가시킨 후 1초 동안 정지했다가, 역방향으로 최저 속도에서 최고 속도로 속도를 서서히 증가시키는 스케치의 예이다.

스케치 1. DC 모터 제어

```
int PWM1 = 11;           // 모터 1 제어를 위한 연결 핀
int Enable1 = 10;
int DIR1 = 9;

void setup(){
  // 모터 제어를 위한 핀들을 출력으로 설정
  pinMode(PWM1, OUTPUT);
  pinMode(Enable1, OUTPUT);
  pinMode(DIR1, OUTPUT);

  digitalWrite(Enable1, HIGH);           // 초기 상태는 비활성화 상태
}

void loop(){
  digitalWrite(Enable1, LOW);           // 모터 1 활성화

  digitalWrite(DIR1, HIGH);           // 정방향 회전
  for(int i = 0; i < 256; i++){        // 속도 조절을 위한 PWM 신호 출력
    analogWrite(PWM1, i);
    delay(25);
  }
  analogWrite(PWM1, 0);                 // 모터 정지

  delay(1000);

  digitalWrite(DIR1, LOW);             // 역방향 회전
  for(int i = 0; i < 256; i++){        // 속도 조절을 위한 PWM 신호 출력
    analogWrite(PWM1, i);
    delay(25);
  }
}
```

```
}  
analogWrite(PWM1, 0);           // 모터 정지  
  
digitalWrite(Enable1, HIGH);     // 모터 1 비활성화  
  
while(true);  
}
```

2. 서보 모터

서보 모터는 DC 모터의 한 종류로 PWM 신호를 사용하여 원하는 만큼 모터를 회전 시킨 후 위치를 유지할 수 있는 특징을 가진다. 서보 모터는 표준 서보 모터와 연속 회전 서보 모터로 나눌 수 있다. 표준(standard) 서보 모터는 회전 범위가 제한된 서보 모터로 일반적으로 $0^{\circ} \sim 180^{\circ}$ 범위에서만 회전할 수 있어 회전 범위가 제한된 기기의 움직임을 제어하기 위해 사용된다. 이에 비해 연속 회전 (continuous rotation) 서보 모터는 DC 모터와 마찬가지로 360° 를 회전할 수 있다. 이 장에서 사용할 서보 모터는 표준 서보 모터이다.

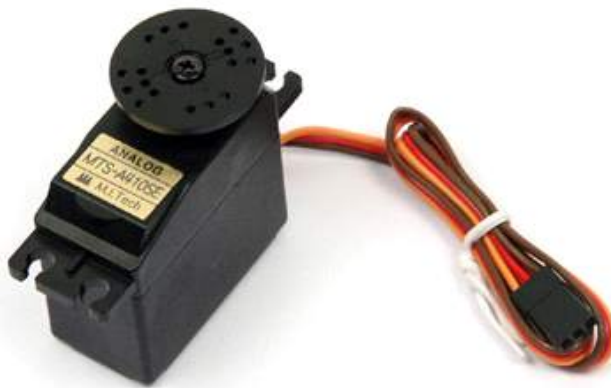


그림 5. 표준 서보 모터²⁾

서보 모터는 PWM 신호를 통해 회전 각도를 조절하므로 DC 모터와 같이 회전 방향을 변경하기 위해 모터 드라이버가 필요하지는 않다. 또한 아두이노 우노 보드의 레귤레이터는 1A 까지 전류를 공급할 수 있으므로 대부분의 서보 모터는 직접 제어가 가능하다. DC 모터에서와

2) 이미지 출처: <http://www.newtc.co.kr>

마찬가지로 모터는 전기 잡음을 유발할 수 있으므로 마이크로컨트롤러의 안정적인 동작을 위해 모터에는 전용 전원을 공급하는 것이 추천되지만, 이 장에서는 서보 모터를 아두이노에 직접 연결하여 제어하는 방법을 알아본다.

서보 모터는 일반적으로 3개의 연결선을 (VCC, GND 그리고 제어선) 가지며 VCC는 붉은색, GND는 검정색이나 갈색, 제어선은 노란색, 주황색 또는 흰색으로 구별된다. DC 모터의 경우 VCC와 GND의 구별이 없으며 반대로 연결하는 경우 반대 방향으로 모터가 회전하지만, 서보 모터의 경우에는 전원을 반대로 연결하면 모터가 파손될 수 있으므로 주의하여야 한다.

서보 모터의 위치 제어는 PWM 신호에 의해 이루어지며, 서보 모터가 PWM 신호를 받으면 내부적으로 현재 위치와 입력된 신호를 비교하여 입력된 PWM 신호에 맞는 위치로 모터를 회전시킨다. 즉, 듀티 사이클에 따라 서보 모터의 위치가 정해진다. 서보 모터는 50Hz의 PWM 주파수, 즉 주기 $\frac{1}{50Hz} = 20ms$ 의 PWM 신호를 필요로 한다. 20ms 중 서보 모터의 위치를 결정하는 구간은 1~2ms로, 1ms에서는 반시계 방향으로 최대로 회전한 상태(0°)이며 2ms에서는 시계 방향으로 최대로 회전한 상태(180°)에 있게 된다. 서보 모터의 종류에 따라 회전할 수 있는 각도의 범위와 이에 따른 듀티 사이클에 차이가 있을 수 있으므로 사용하고자 하는 서보 모터의 데이터시트를 확인하여야 한다.

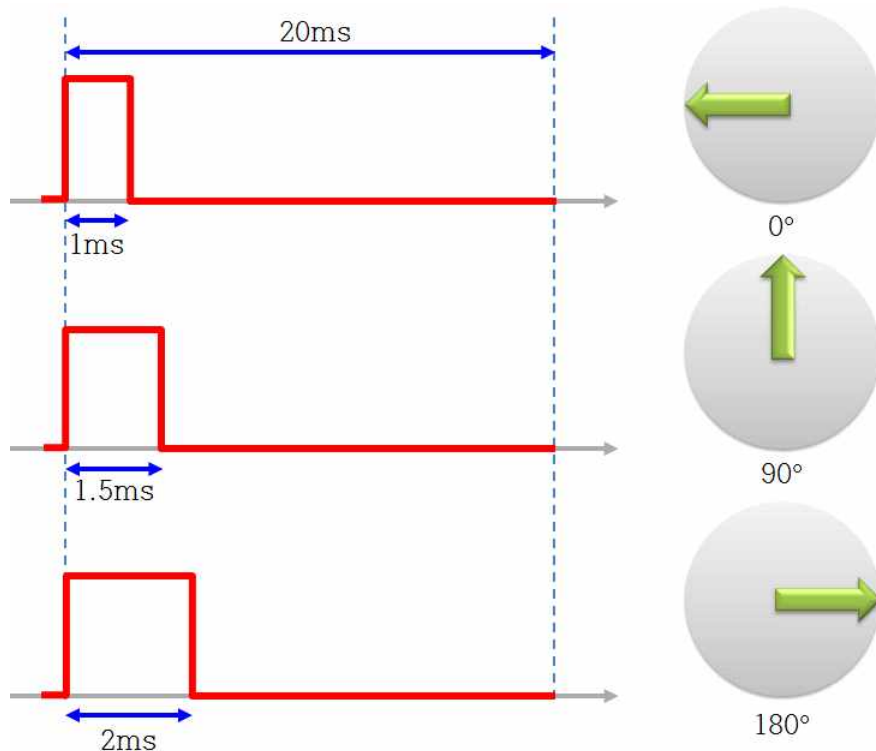


그림 6. 서보 모터 제어

서보 모터 제어를 위해 아두이노에는 기본 라이브러리 중 하나로 Servo 라이브러리가 포함되어 있다. Servo 라이브러리를 사용하면 아두이노 우노에서는 최대 12개까지의 서보 모터를 제어할 수 있다. Servo 라이브러리를 사용하기 위해서는 먼저 각 서보 모터 제어를 위한 객체를 생성하고 attach 함수를 통해 서보 모터 제어 핀을 서보 모터와 연결시켜 주어야 한다.

```
uint8_t attach(int pin)
uint8_t attach(int pin, int min, int max)
- 매개변수
  pin : 서보의 데이터 선이 연결된 핀 번호
  min : 0°에 해당하는 마이크로초 단위의 펄스 폭 (디폴트값은 544)
  max : 180°에 해당하는 마이크로초 단위의 펄스 폭 (디폴트값은 2400)
- 반환값 : 현재 서보 모터가 연결된 채널
```

attach 함수에서 반환되는 값은 1에서 12 사이의 값으로 연결에 실패하면 255를 반환한다. 서보 모터가 연결되면 write 함수를 통해 각도 제어가 가능하다.

```
void write(int value)
- 매개변수
  value : 서보 모터의 제어 값
- 반환값 : 없음
```

write 함수의 매개변수는 0에서 180 사이의 값으로 표준 서보 모터의 경우 각도를, 연속 회전 서보 모터의 경우 회전 속도를 나타낸다. 그림 7과 같이 서보 모터를 연결해 보자.

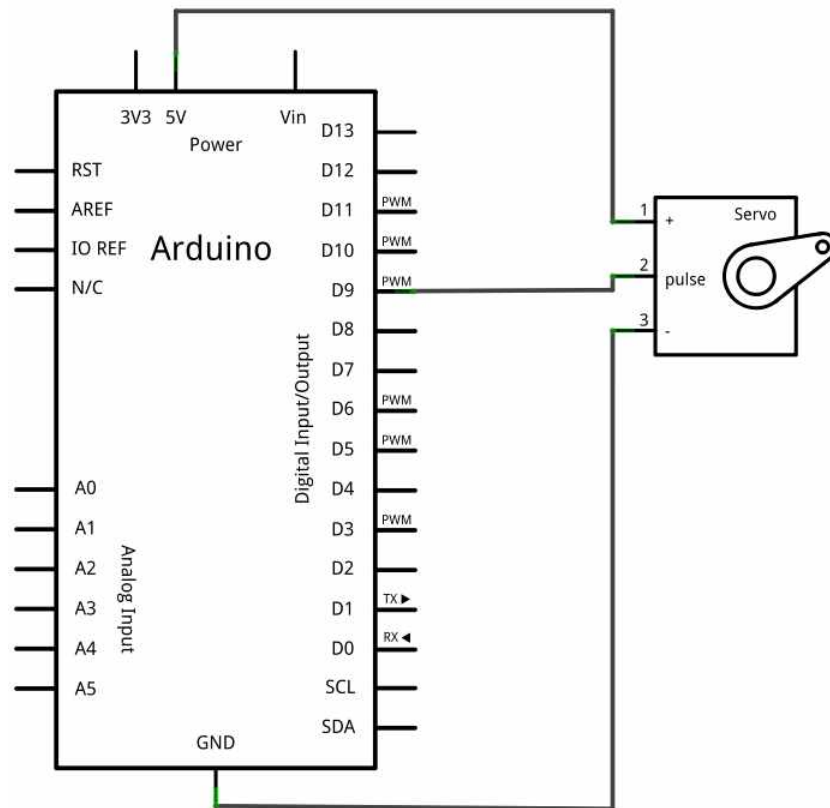


그림 7. 서보 모터 연결 회로도

스케치 2는 서보 모터를 0°에서 180°로 회전시키고, 다시 0°로 회전시키는 스케치의 예이다.

스케치 2. 서보 모터 제어

```
#include <Servo.h>

Servo microServo;
int servoPin = 9;

void setup(){
    microServo.attach(servoPin);    // 서보 모터 연결
}

void loop(){
    int angle;                      // 축 제어 값
```

```

for(angle = 0; angle < 180; angle++){    // 0도에서 180도로
    microServo.write(angle);
    delay(20);
}

for(angle = 180; angle > 0; angle--){    // 180도에서 0도로
    microServo.write(angle);
    delay(20);
}
}

```

3. 스텝핑 모터

스텝핑 모터(steping motor)는 스텝 모터(step motor), 펄스 모터(pulse motor) 등으로도 불리는 모터로 펄스에 의해 모터의 회전을 제어할 수 있어 각도 제어에 유리하다. 서보 모터에 비해 일반적으로 가격이 저렴하므로 정확한 위치 제어가 필요한 경우에 많이 사용된다. 하지만 분할각 단위로만 제어가 가능하므로 필요한 정밀도에 따라 모터를 선택하여야 한다. 스텝핑 모터의 기본 구조는 그림 8과 같다.

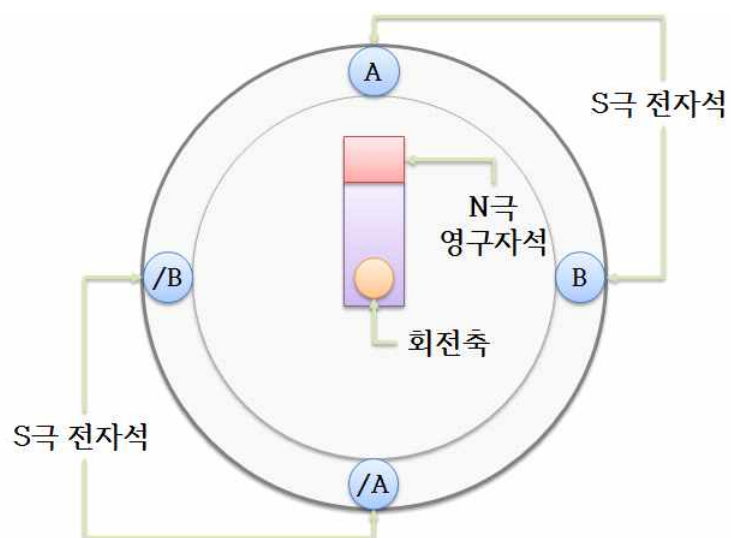


그림 8. 스텝핑 모터의 구조

그림 8에서와 같이 스텝핑 모터는 4개의 고정된 전자석과 회전하는 영구자석으로 이루어진다. 4개의 전자석 중 하나의 코일에만 전류를 흘려 S극의 전자석으로 동작하도록 하면 회전축에 연결된 N극의 영구자석은 전자석 쪽으로 회전하게 되고 4개의 전자석에 순서대로 전류를 흘리면 회전력을 얻을 수 있다. 그림 8에서 $A \rightarrow B \rightarrow /A \rightarrow /B$ 순서로 전류를 흘리면 축은 정방향으로 회전하고, $A \rightarrow /B \rightarrow /A \rightarrow B$ 순서로 전류를 흘리면 축은 역방향으로 회전한다. 그림 8의 모터의 경우 전자석의 상태 변화에 따라 90° 를 회전하는 구조를 가지고 있다, 즉, 90° 분할각을 가진다. 하지만 한 번에 90° 씩만 회전 가능한 모터는 사용할 수 있는 곳이 그리 많지 않다. 따라서 회전자는 톱니바퀴 형태로 N극과 S극이 번갈아 배치된 형태로 만들어지는 것이 일반적이며, 코일의 수도 4개 이상이 흔히 사용된다. 톱니의 수와 코일의 수는 분할각에 반비례하며 일반적으로 1.8° 의 분할각을 가지는 스텝핑 모터가 흔히 사용된다.

그림 8에서 회전자를 회전시키기 위해서는 코일이 S극만을 만들어낼 수 있으면 되지만 안정적인 회전을 위해 N극과 S극을 만들어 낼 수 있도록 구성된다. 이 때 N극과 S극을 만들어 내기 위해서는 코일에 가하는 전원의 극성을 조절하여야 하며, 코일에 전원을 가하는 방식에 따라 스텝핑 모터는 단극(unipolar) 모터와 양극(bipolar) 모터로 나눌 수 있다. 그림 9는 단극 모터와 양극 모터에서 전원을 가하는 방법을 나타낸 것이다.

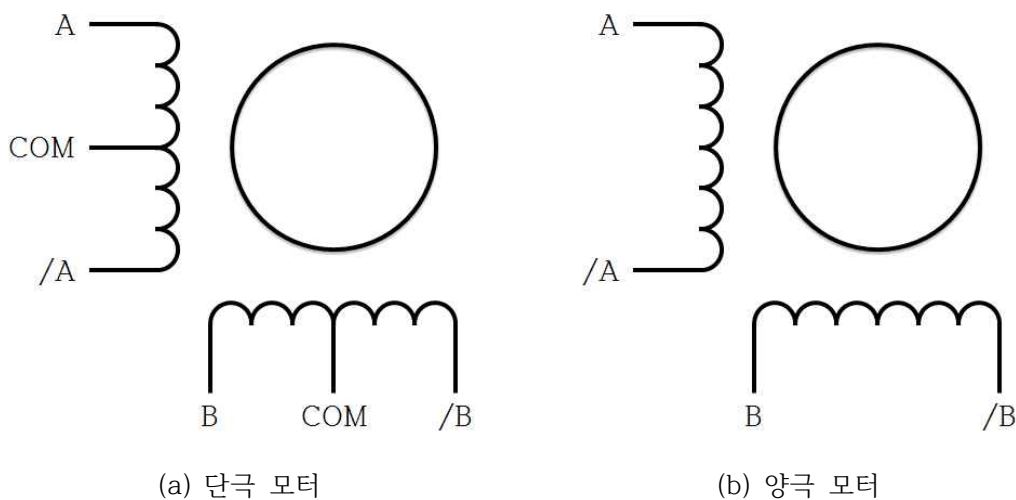


그림 9. 단극 및 양극 모터의 구조

단극 모터의 경우 공통 연결선에 전원(또는 GND)을 연결하고, A나 /A중 하나에 GND(또는 전원)를 연결함으로써 전자석의 극성을 결정할 수 있다. 반면 양극 모터의 경우에는 A와 /A에 전원을 연결하는 방향에 따라 극성을 결정할 수 있다. 따라서 일반적으로 단극 모터의 경우 6개의 연결선을, 양극 모터의 경우 4개의 연결선을 가진다. 이 장에서는 그림 10의 단극

방식 스텝핑 모터를 사용한다.



그림 10. 스텝핑 모터³⁾

모터의 구동 방식 역시 1상 여자 방식(1 phase excitation)과 2상 여자 방식의 두 가지로 나눌 수 있다. 1상 여자 방식은 하나의 코일에만 전류를 흘려 하나의 전자석만을 사용하는 방식이다. 하나의 코일만을 사용하는 경우 그림 8에서 활성화되는 코일은 표 2와 같다.

스텝	1	2	3	4	5	6	7	8	9
A	1	0	0	0	1	0	0	0	1
B	0	1	0	0	0	1	0	0	0
/A	0	0	1	0	0	0	1	0	0
/B	0	0	0	1	0	0	0	1	0

표 2. 1상 여자 방식의 구동

이에 비해 2상 여자 방식은 인접한 2개 코일을 동시에 사용하는 방식이다. 2상 여자 방식의 경우 1상 여자 방식에 비해 2배의 전류가 필요하지만 토크가 크고 진동이 적어 널리 사용되고 있다.

3) 이미지 출처: <http://www.newtc.co.kr/>

스텝	1	2	3	4	5	6	7	8	9
A	1	0	0	1	1	0	0	1	1
B	1	1	0	0	1	1	0	0	1
/A	0	1	1	0	0	1	1	0	0
/B	0	0	1	1	0	0	1	1	0

표 3. 2상 여자 방식의 구동

두 가지 방식을 함께 사용하는 1-2상 여자 방식은 전류 소모량이 1상 여자 방식의 1.5배가 되며, 1상 여자 방식과 2상 여자 방식에 비해 분할각이 $1/2$ 이므로 정밀 제어에 사용된다.

스텝	1	2	3	4	5	6	7	8	9
A	1	1	0	0	0	0	0	1	1
B	0	1	1	1	0	0	0	0	0
/A	0	0	0	1	1	1	0	0	0
/B	0	0	0	0	0	1	1	1	0

표 4. 1-2상 여자 방식의 구동

그림 10의 모터를 포함하여 스텝핑 모터 중 흔히 사용되는 모터는 한 스텝 당 1.8° 를 회전하며 1바퀴 회전을 위해서는 200 단계를 필요로 한다. 1상 및 2상 여자 방식의 경우 풀 스텝(full step) 방식이라고 하며 한 스텝에 1.8° 를 회전한다. 반면 1.5상 여자 방식의 경우 하프 스텝(half step) 방식이라고 하며 한 번에 1.8° 의 절반인 0.9° 회전한다. 전자석의 극성뿐만이 아니라 전자석에 흐르는 전류의 양을 조절하여 분할각을 더 줄이는 방식을 마이크로 스텝(micro step) 방식이라고 하며 $1/256$ 까지 분할각을 줄일 수 있는 것으로 알려져 있다.

스텝핑 모터 제어를 위해 아두이노에서는 전용 라이브러리인 Stepper 라이브러리를 제공하고 있다. Stepper 라이브러리에는 다음과 같은 멤버 함수들이 정의되어 있다.

- Stepper

```
Stepper(int number_of_steps, int pin_1, int pin_2)
Stepper(int number_of_steps, int pin_1, int pin_2, int pin_3, int pin_4)
- 매개변수
  number_of_steps : 1회전을 위한 스텝 수
  pin_1, pin_2, pin_3, pin_4 : 모터 연결 핀
- 반환값 : 없음
```

Stepper 클래스의 객체를 생성한다. number_of_steps는 1회전을 위한 스텝 수를 지정하며, 모터 제어를 위해서는 2개 또는 4개의 핀을 사용할 수 있다. 그림 10에서 볼 수 있듯이 스테핑 모터에는 최소 4개의 제어선을 필요로 하며 스테핑 모터 제어를 위한 전압은 마이크로컨트롤러 구동을 위한 전압보다 높은 것이 일반적이다. 따라서 DC 모터 제어에서와 같이 구동을 위한 회로가 필요하다. 단극 모터의 경우 전원을 on/off하는 기능만 있으면 제어가 가능하므로 전용 전원과 달링턴 회로(Darlington circuit)를 통한 스위치 기능을 통해 제어할 수 있다. 두 개의 연결핀으로 제어하기 위해서는 A와 /A, B와 /B에 전원이 반대로 가해진다는 사실을 이용하여 제어할 수 있다. 물론 구성해야 하는 회로는 좀 더 복잡해진다. 양극 모터의 경우에는 on/off 기능이 아니라 VCC나 GND를 선택적으로 가해주어야 하므로 DC 모터에서와 마찬가지로 H-bridge 회로가 필요하다.

- setSpeed

```
void setSpeed(long whatSpeed)
- 매개변수
  whatSpeed : 분당 회전 수
- 반환값 : 없음
```

모터의 분당 회전수(RPM, Rounds per Minute)를 설정한다.

- step

```
void step(int number_of_steps)
- 매개변수
  number_of_steps : 스텝 수
- 반환값 : 없음
```

모터를 지정한 스텝 수만큼 회전시킨다. 양수나 음수를 지정함으로써 정방향 또는 역방향 회

전을 지정할 수 있다. step 함수는 블로킹(blocking) 함수에 해당한다. 즉, step 함수는 지정된 스텝을 진행할 때까지 제어권을 넘기지 않는다. 따라서 많은 수의 스텝을 한꺼번에 회전시키기 보다는 작은 수의 스텝을 여러 번 회전시키는 것이 바람직하다.

Stepper 라이브러리의 4선 제어 방식을 사용하여 스텝핑 모터를 제어해 보자. 스텝핑 모터 제어를 위해서는 아두이노 우노의 전압인 5V보다 높은 전압이 필요한 경우가 대부분이므로 DC 모터의 경우와 마찬가지로 모터 드라이버 칩을 사용한 전용 제어 모듈을 사용하는 것이 일반적이다. 스텝핑 모터 제어를 위해 이 장에서는 그림 11의 스텝핑 모터 제어 모듈을 사용한다.

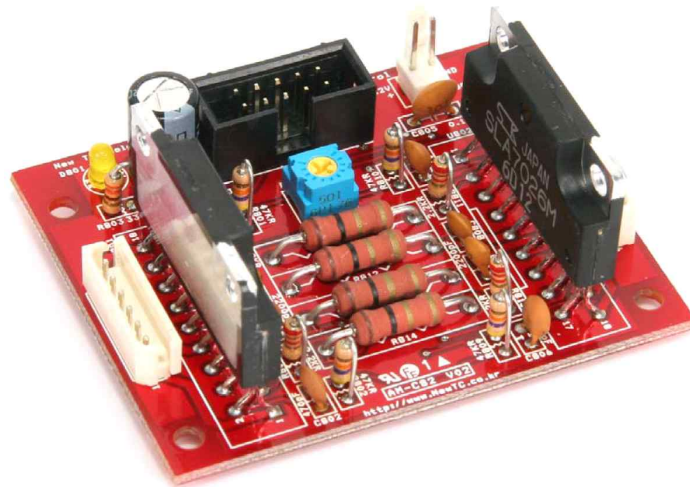


그림 11. 스텝핑 모터 제어 모듈 14)

그림 11의 스텝핑 모터 제어 모듈은 모터 드라이버로 SLA7026M 칩을 사용하고 있으며 2개의 스텝핑 모터를 연결하여 사용할 수 있다. 각각의 모터 제어를 위해서는 A, /A, B, /B에 해당하는 4개의 제어선 연결을 필요로 한다. 모터 제어 모듈의 연결 방법은 그림 12와 같다. 그림 10의 스텝핑 모터를 제어하기 위해 모터 제어 모듈의 모터 전원은 12V 이상을 연결하여야 하며, 모터가 정상적으로 회전하지 않는다면 가변저항을 조절하여 모터에 유입되는 전류를 조절하면 된다.

4) 이미지 출처: <http://www.newtc.co.kr>

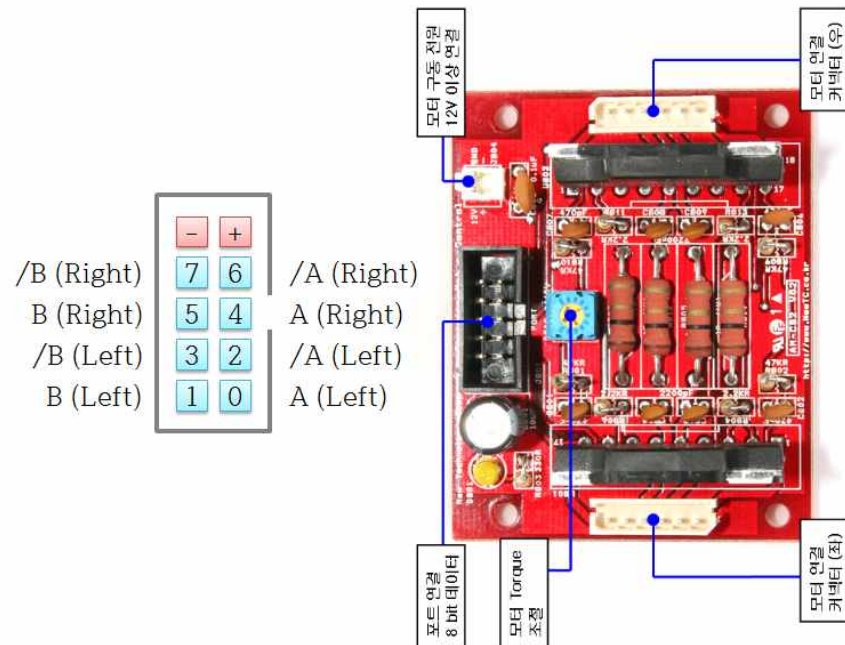


그림 12. 모터 제어 모듈 연결 1

Stepper 클래스의 객체를 생성할 때 생성자의 각 핀은 A, /A, B, /B의 순서에 해당한다.

```
Stepper(int number_of_steps, int pin_1, int pin_2, int pin_3, int pin_4)
Stepper(int number_of_steps, int pin_A, int pin_slash_A,
        int pin_B, int pin_slash_B)
```

그림 14와 같이 모터 제어 모듈을 사용하여 아두이노에 스텝핑 모터를 연결해 보자. 스텝핑 모터는 6핀의 연결선을 가지고 있으며, 그림 14에서 '(C)'는 컨트롤러의 제어핀과 연결하는 부분을 나타낸다.

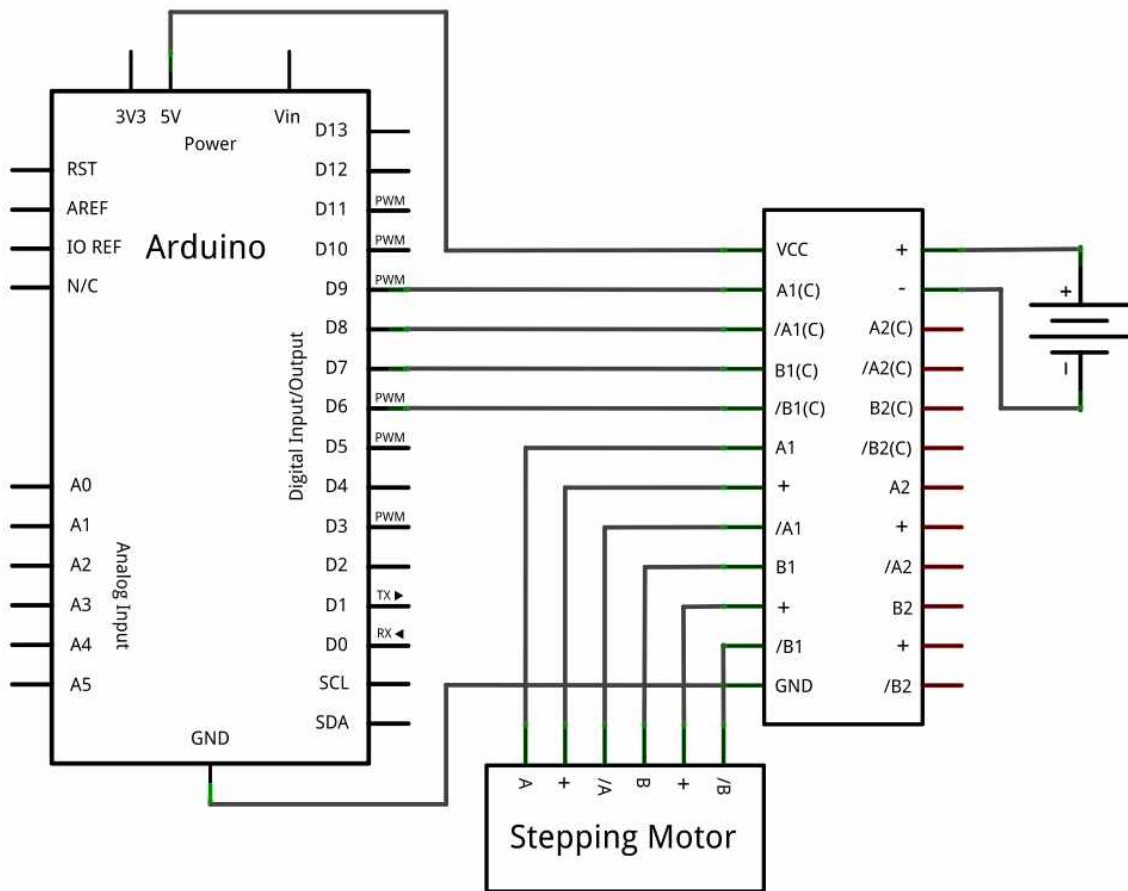


그림 14. 스텝핑 모터 연결 회로도

스케치 3은 Stepper 라이브러리를 사용하여 스텝핑 모터를 제어하는 스케치의 예이다. 스케치 3은 정방향으로 1바퀴 회전하고 0.5초 정지한 후, 역방향으로 1바퀴 회전하고 0.5초 정지하기를 반복하는 스케치이다.

스케치 3. 스텝핑 서보 모터 - Stepper 라이브러리

```
#include <Stepper.h>

const int stepsPerRevolution = 200;      // 1바퀴 회전을 위해 필요한 스텝 수

// 아두이노의 디지털 9, 8, 7, 6번 핀은
// 스텝핑 모터 제어 모듈의 A, /A, B, /B 핀으로 연결
Stepper myStepper(stepsPerRevolution, 9, 8, 7, 6);
```

```
void setup() {  
    myStepper.setSpeed(60);           // 분당 60회전, 초당 1회전  
}  
  
void loop() {  
    // 양수인 경우 정방향 회전  
    // 200 스텝 진행 → 1회전에 해당  
    myStepper.step(stepsPerRevolution);  
    delay(500);                       // 0.5초 대기  
  
    myStepper.step(-stepsPerRevolution); // 음수인 경우 역방향 회전  
    delay(500);  
}
```