

Chapter 113. DC 모터

1. DC 모터

DC 모터는 고정자로 영구자석을, 회전자로 코일을 사용하여 만들어진 간단한 구조의 모터로 회전자에 전류를 흘려줌으로써 자력이 발생하고 고정자와의 반발 및 흡인력으로 회전을 발생시킨다. DC 모터는 제어가 간단하고, 회전 특성이 전류나 전압에 비례하며, 가격이 저렴하여 널리 사용되고 있다. DC 모터의 가장 큰 단점은 구조상 브러시(brush)와 정류자(commutator) 사이에 접점이 존재하며, 모터 구동 시에 접점에서 스파크와 소음이 발생할 수 있으며 이로 인해 수명이 짧다는 점이다. 이러한 잡음으로 인해 DC 모터로 정밀 제어는 불가능하다는 점도 단점이라 할 수 있다. 이러한 단점을 개선한 DC 모터가 브러시를 없앤 브러시리스(brushless) DC 모터로, DC 모터는 코일이 회전하는 반면 브러시리스 DC 모터는 영구자석이 회전하는 형태로 만들어진 점에서 차이가 있다.

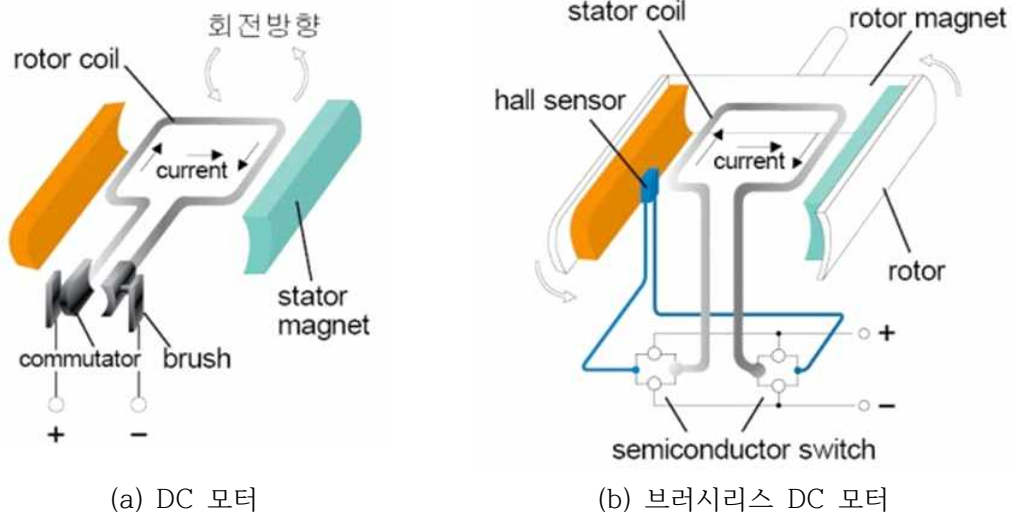


그림 1. DC 모터와 브러시리스 DC 모터의 구조

DC 모터는 두 개의 연결선만을 가지며 VCC와 GND에 연결하면 연결 방향에 따라 정회전 또는 역회전 하는 모터로 별도의 제어선을 필요로 하지 않는다. 따라서 모터의 VCC에 디지털 출력 핀을 연결하여 HIGH 또는 LOW 값을 출력함으로써 간단히 모터의 on/off를 제어할 수 있다. 또는 PWM 신호 출력이 가능한 핀에 연결하여 PWM 신호를 통해 속도 제어도 가능하

다. 하지만 DC 모터를 제어하기 위해서는 두 가지 점에 유의하여야 한다. 첫 번째는 ATmega328의 디지털 출력 핀에서 공급할 수 있는 최대 전류는 40mA로 이는 소형 DC 모터를 구동시키기에도 충분하지 않다는 점이다. 따라서 큰 전류를 요하는 DC 모터의 경우 모터 전용의 전원을 사용하고 디지털 핀의 출력은 스위치 용도로 사용된다. P2N2222 트랜지스터의 경우 ATmega328의 디지털 출력 핀을 베이스에 연결하면 콜렉터와 에미터 사이에 흐르는 큰 전류를 제어할 수 있도록 해주므로 DC 모터 제어를 위한 스위치로 사용할 수 있다.



그림 2. P2N2222 트랜지스터

큰 전류를 제어할 수 있는 방법은 트랜지스터를 사용하면 된다. 하지만 여전히 문제점은 남아 있다. 두 번째 문제점은 모터를 회로에 연결하고 나면 한 방향으로만 회전이 가능하다는 점이다. 모터는 일반적으로 속도 조절 이외에도 정회전과 역회전이 필요하다. 따라서 모터 회전 방향을 바꾸기 위해서는 모터 드라이버(motor driver)라 불리는 전용의 칩을 사용한다. 모터 드라이버 칩은 제어하고자 하는 모터가 필요로 하는 전압 및 전류에 따라 여러 가지 종류가 존재하므로 모터에 맞게 선택하여 사용할 수 있다. 그림 3은 아두이노의 공식 모터 쉴드에 사용된 L298N 칩을 나타낸다¹⁾.

1) 아두이노 공식 모터 쉴드에 사용된 칩은 SMD 타입의 L298P 칩으로 20개의 핀을 가지고 있지만 기능적으로는 동일하다.

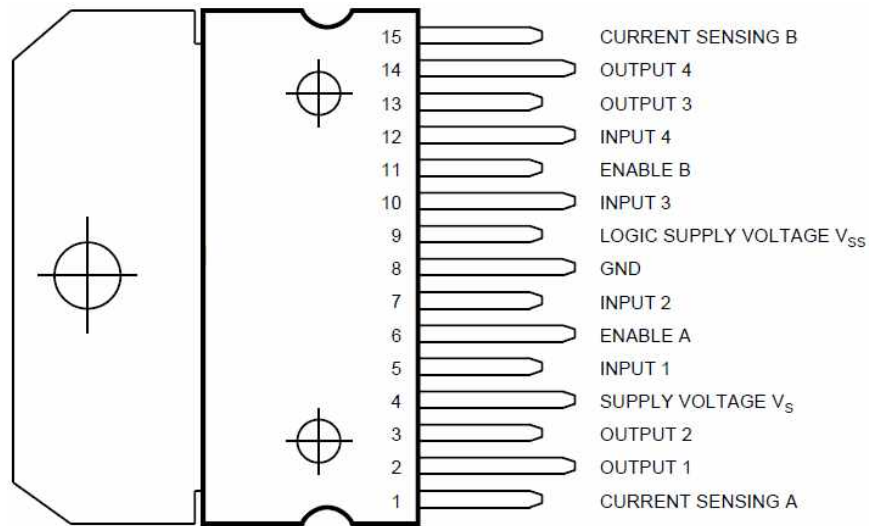


그림 3. L298N 모터 드라이버 칩

L298N 칩은 두 개의 H 브리지(bridge) 회로로 구성되며 15 개의 핀을 가지고 있다. H 브리지 회로는 전압이 정방향 또는 역방향으로 가해질 수 있도록 구성된 회로로 DC 모터와 함께 사용되면 정방향 또는 역방향 회전을 선택하기 위해 사용할 수 있다. H 브리지 회로는 그림 4에서와 같이 4개의 스위치로 표현할 수 있다. 스위치 1과 4가 연결되는 경우와 스위치 2와 3이 연결되는 경우 모터에 가해지는 전압은 반대가 된다.

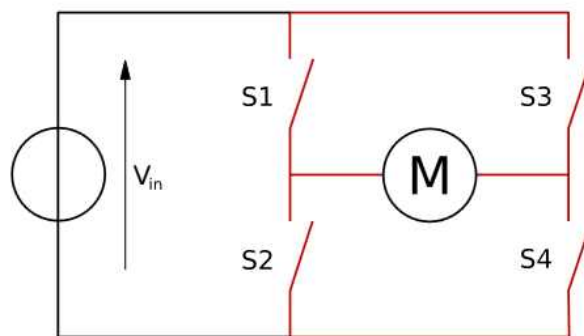


그림 4. H 브리지 회로

H 브리지 회로를 이용하면 모터의 회전 방향을 제어할 수 있지만, 이를 위해서는 L298N 칩 이외에도 여러 가지 추가적인 부품이 필요하다. 따라서 DC 모터 제어를 위해서는 DC 모터 제어를 위해 만들어진 전용의 보드를 사용하는 것이 일반적이며 아두이노 공식 모터 쉴드 역시 그 중 하나이다.

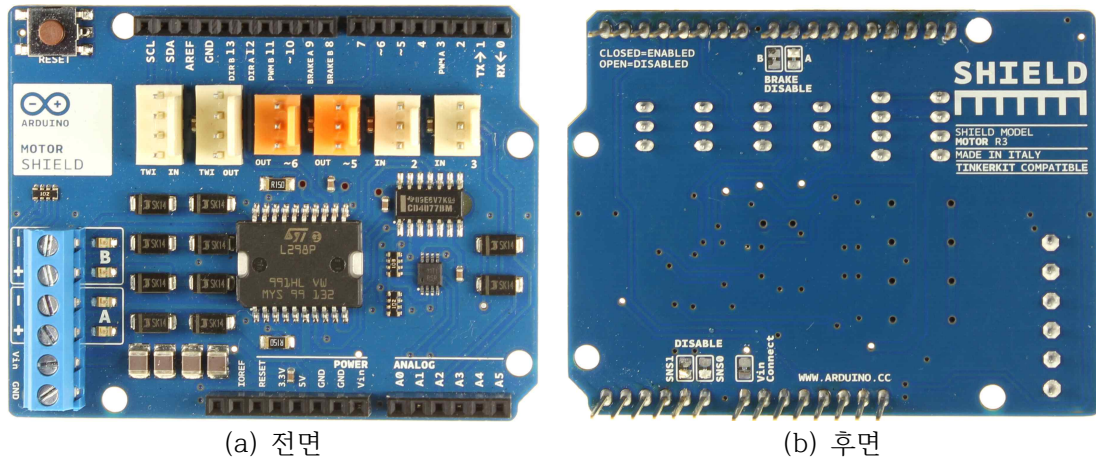


그림 5. 아두이노 공식 모터 쉴드²⁾

이 장에서 사용할 모터 제어 모듈은 그림 6으로 아두이노 공식 모터 쉴드와 동일한 L298N 칩을 사용하고 있다.

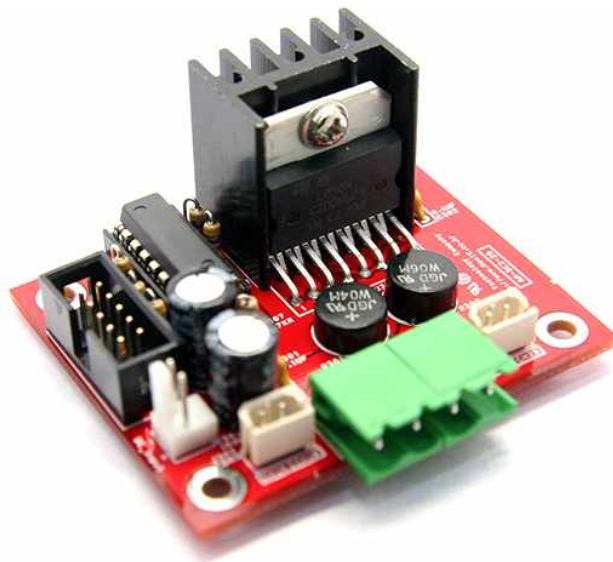


그림 6. 모터 제어 모듈³⁾

그림 6의 모터 제어 모듈은 4개의 전원 연결, 6개의 모터 제어선 연결, 4개의 모터 연결을 필요로 한다.

2) 그림 출처: <http://arduino.cc/en/Main/ArduinoMotorShieldR3>

3) 그림 출처: <http://www.newtc.co.kr>

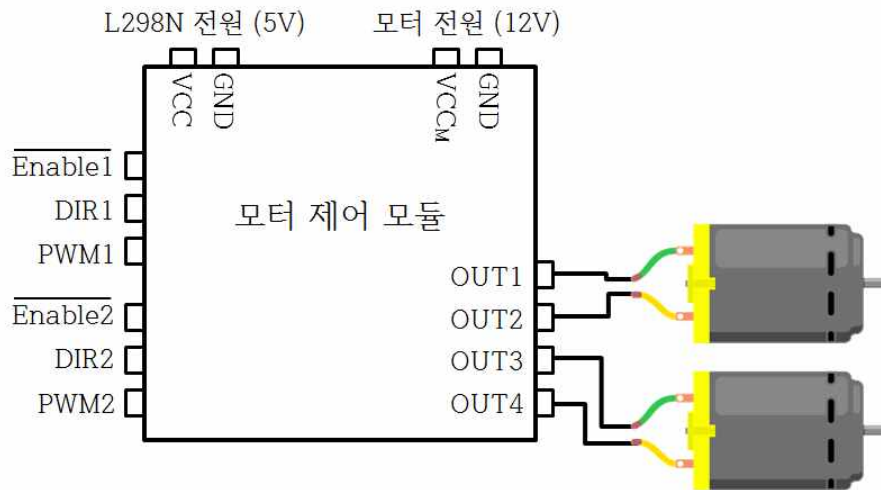


그림 7. 모터 제어 모듈과 모터 연결

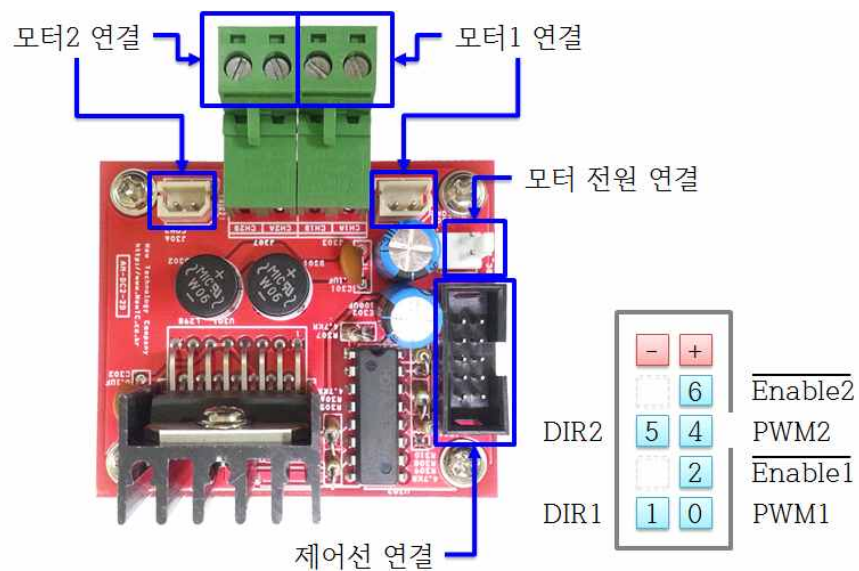


그림 8. 모터 제어 모듈 연결 커넥터

전원의 경우 L298N 칩의 동작을 위한 전원과 모터 동작을 위한 전원이 별개로 주어진다. 6개의 제어선은 두 개의 모터를 개별적으로 제어하기 위해 필요하다. Enable은 입력이 LOW인 상태에서만 모터의 구동 및 제어가 가능하며, HIGH인 상태에서는 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.

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

표 1. 모터 제어 모듈 제어선

2. 모터 제어

그림 9와 같이 모터 제어 모듈을 사용하여 모터를 연결해 보자. 모터 제어 모듈에는 두 개까
지 모터를 연결할 수 있지만 제어 방법은 동일하므로 하나의 모터만을 연결하였다.

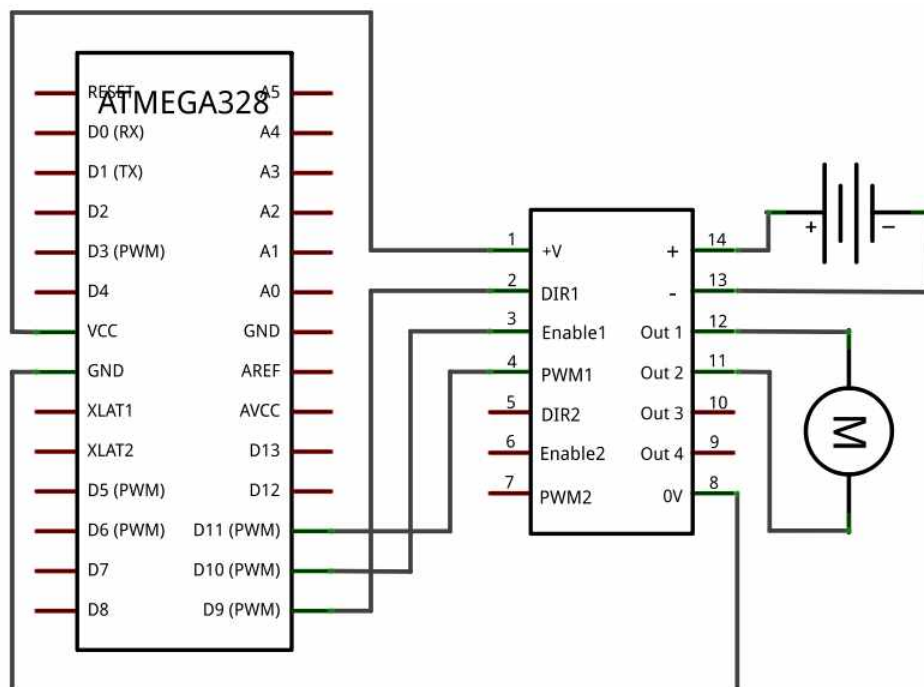


그림 9. 모터 제어 모듈 연결 회로도

코드 1은 모터를 정방향으로 최저 속도에서 최고 속도로 회전시킨 후 1초 동안 정지했다가, 역방향으로 최저 속도에서 최고 속도로 회전시키는 코드이다. PWM 출력을 위해서는 타이머/카운터 2번을 고속 PWM 모드로 사용하였다.

코드 1. 모터 제어

```
#define F_CPU 16000000L
#include <avr/io.h>
#include <util/delay.h>

int main(void)
{
    DDRB |= 0x0E; // 모터 제어를 위한 핀들을 출력으로 설정

    // 타이머/카운터 2번을 고속 PWM 모드로 설정
    TCCR2A |= (1 << WGM21) | (1 << WGM20);
    TCCR2A |= (1 << COM2A1); // 비반전 모드
    TCCR2B |= (1 << CS22); // 분주비 64

    PORTB |= 0x04; // 초기 상태는 비활성화 상태

    while(1)
    {
        PORTB &= ~0x04; // 모터 1 활성화

        PORTB |= 0x02; // 정방향 회전
        for(int i = 0; i < 256; i++){ // 속도 조절을 위한 PWM 신호 출력
            OCR2A = i;
            _delay_ms(25);
        }
        OCR2A = 0; // 모터 정지

        _delay_ms(1000);

        PORTB &= ~0x02; // 역방향 회전
        for(int i = 0; i < 256; i++){ // 속도 조절을 위한 PWM 신호 출력
            OCR2A = i;
            _delay_ms(25);
        }
        OCR2A = 0; // 모터 정지
```

```
PORTB |= 0x04;                // 모터 1 비활성화

while(1);

}
```

아두이노 환경에서 모터를 제어하는 방법 역시 동일하다. 다만 아두이노에서는 PWM 신호 출력을 위해 별도의 설정이 필요하지 않으며 analogWrite 함수를 사용하여 지정한 분주비를 가지는 PWM 신호를 출력할 수 있으므로 편리하다.

코드 2. 모터 제어 - 아두이노

```
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);
}
```