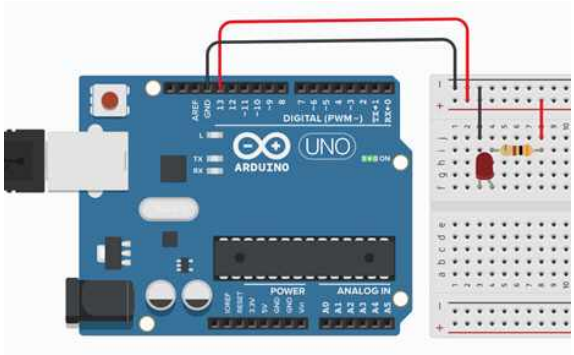
< 정보과학프로젝트 >

1. Physical Computing (p.2)

- 디지털 기술을 통해 사용자로부터 물리적인 방식으로 정보 입력 (Sensor)
- 처리한 결과를 물리적 방식으로 출력 (Actuator)

2. LED 켜는 코드 (p.5,13)



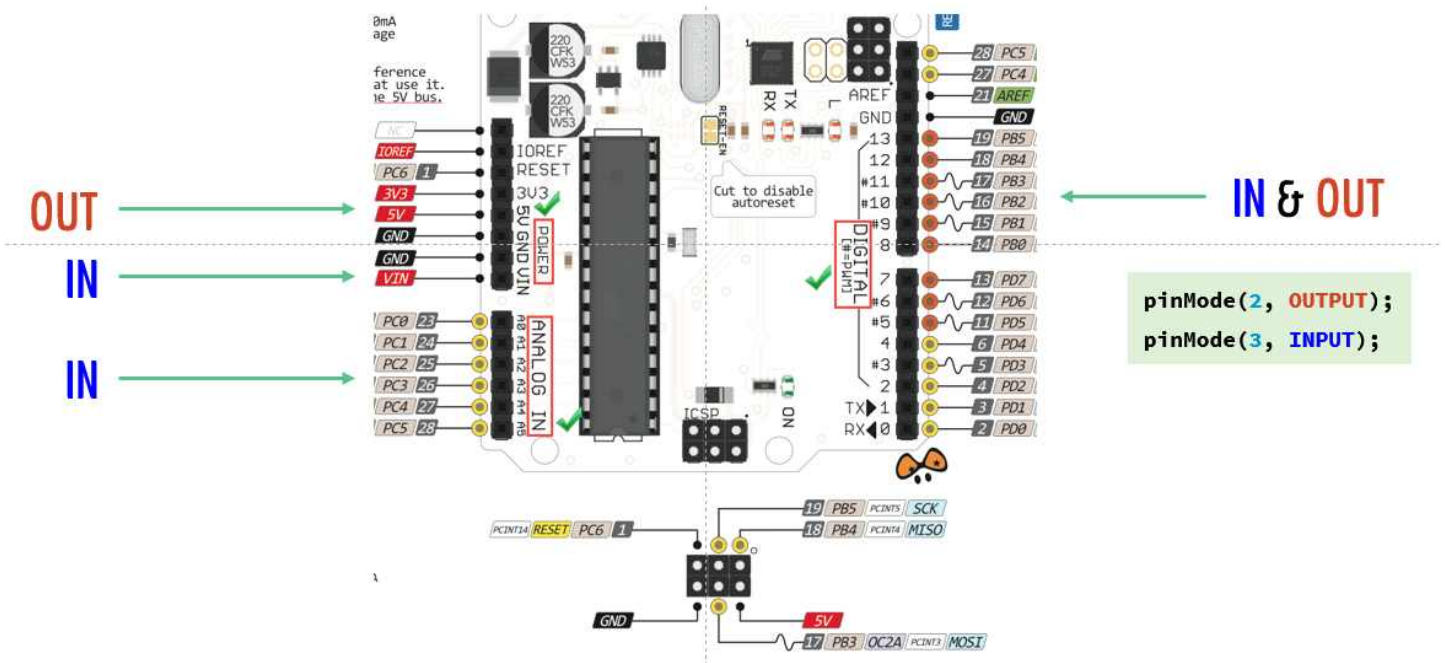
```
void setup()  //Run code once
{   //Set the pin 13 to an output
    pinMode(13, OUTPUT);
}

void loop()  //Run code forever
{
    digitalWrite(13, HIGH);
    delay(1000);
    digitalWrite(13, LOW);
    delay(1000);
}
```

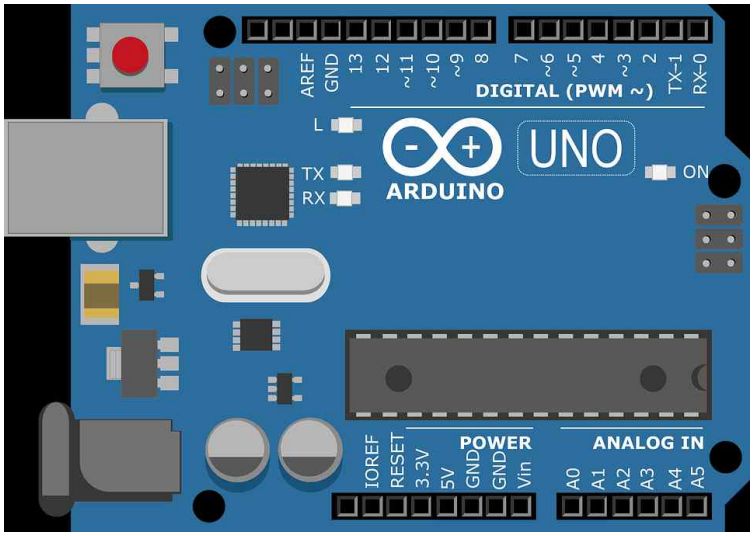
3. Base Knowledges (p.6)

- 센서(Sensor) : 자극을 받아들이고 이를 통해 외부의 상태를 알아내는 장치
  ex) 소리감지센서, 온도센서, LED, RTC, 조이스틱 등
- 액추에이터(Actuator) : 시스템을 움직이거나 제어하는데 쓰이는 기계 장치
  ex) 모터, 스피커, LCD 등

4. Push Button Switch (p.9)

5. Arduino 핀 번호 배열 (p.12)





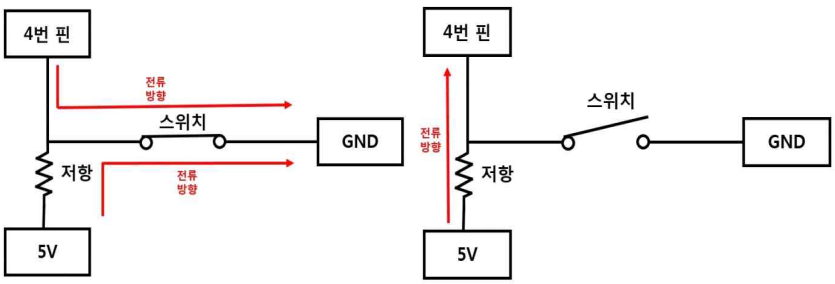
- Digital : 전기의 IN & OUT, 이산적
- PWM (Pulse Width Modulation) : analog out (아날로그 형태의 액추에이터 연결, '~'에 연결)
- POWER : 전기 나오는 부분
- ANALOG IN : 전기 받는 부분 (in) → 센서의 연결 → 물리적 값을 받아들임 (연속적)

6. Photoresistor (LDR), analogRead/analogWrite (p.17~19)

- analogRead()는 0~1023의 값
- analogWrite()는 0~255의 값 : PWM(~)이 허용된 곳에서만
- map(val, fromLow, fromHigh, toLow, toHigh);
  ex) map(val, 0, 1023, 0, 255);

7. Pull Up/Pull Down (p.21~22)

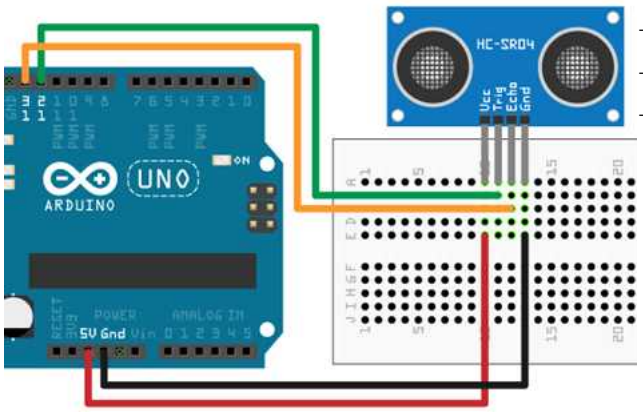
- 풀 업/다운을 사용하는 이유 : 이상적인 스위치 상태가 아니기 때문. 플로팅 상태가 생김.
  아날로그 핀이 고정이 안되기 때문에 사용.
- Pull Up : 입력 핀을 VCC 입력 상태고 묶어두는 것 : 기본 값이 HIGH, 작동이 있으면 LOW
  : 스위치가 닫히면 그라운드 쪽으로 전류가 흘러 회로의 전압이 0V가 됨.
- Pull Down : 0V 상태로 묶어두는 것 : 기본 값이 LOW, 작동이 있으면 HIGH (풀업 반대)



- 플로팅 : 아무것도 연결하지 않은 상태에도 특정 값이 넘어옴.
  보드를 손으로 만지거나 미세한 전류 변화에 의해 입력.
  이런 증상을 없애기 위하여 풀업/풀다운을 사용

```
int pushbt = 4;

void setup(){
  Serial.begin(9600);
  pinMode(pushbt, INPUT_PULLUP);
}

void loop(){
  int btstate = digitalRead(pushButton);
  Serial.println(btstate);
  delay(1);
}
```

8. Ultrasonic (p.23~24)

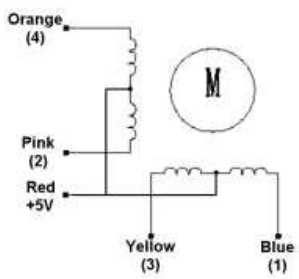


- Pin 순서 : VCC, Trig, Echo, GND
- Trig 핀 HIGH : 40kHz 음파 발사 (10us 이상 유지 권장)
- Echo 핀 : 초음파 수신 시간 계산
- Distance = Time / 29.0 / 2.0

```
int pin=7;
long duration;

void setup(){ pinMode(pin, INPUT); }

void loop(){ duration = pulseIn(pin,HIGH); }
```

```
//출력핀(trig)과 입력핀(echo) 연결 설정
int TrigPin = 7;
int EchoPin = 6;
int OutPin = 13;

//시리얼 속도설정, trigPin을 출력
//echoPin을 입력으로 설정
void setup() {
  Serial.begin(9600);
  pinMode(TrigPin, OUTPUT);
  pinMode(EchoPin, INPUT);
  pinMode(OutPin, OUTPUT);
}

void loop() {
  long duration;
  double distance;

  //Trig 초음파 발사^ ^; Echo 대기
  digitalWrite(TrigPin, HIGH);
  delay(10);
  digitalWrite(TrigPin, LOW);

  // Echo가 초음파를 수신할 동안의 시간 계산
  duration = pulseIn(EchoPin, HIGH);

  // 340m/s --> 1cm/29us.
  // 왕복거리이므로 2로 나눠준다.
  distance = (double)duration / 29.0 / 2.0;

  //시리얼모니터에 Echo가 HIGH인
  //시간 및 거리를 표시해준다.
  Serial.print("Duration:");
  Serial.print(duration);
  Serial.print("\nDIstance:");
  Serial.print(distance);    Serial.println("cm\n");
  if (distance < 5) {
    digitalWrite(OutPin, HIGH);
    delay(100);
    digitalWrite(OutPin, LOW);
    delay(200);
  }
  else {
    digitalWrite(OutPin, LOW);
  }
  delay(500); // 0.5 초 간격...
}
```

9. Stepper Motor (p.26~29)

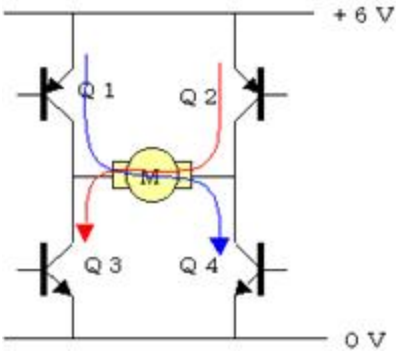


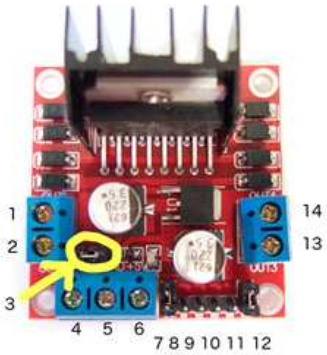
Half-Step Switching Sequence

Lead Wire Color	---> CW Direction (1-2 Phase)							
	1	2	3	4	5	6	7	8
4 Orange	-	-						-
3 Yellow		-	-	-				
2 Pink				-	-	-		
1 Blue						-	-	-

10. L298N (p.30~33)

- easily and independently control two motors of up to 2A each in both directions.





8~11 : 방향 조절
7, 12 : 속도 조절 (PWM)

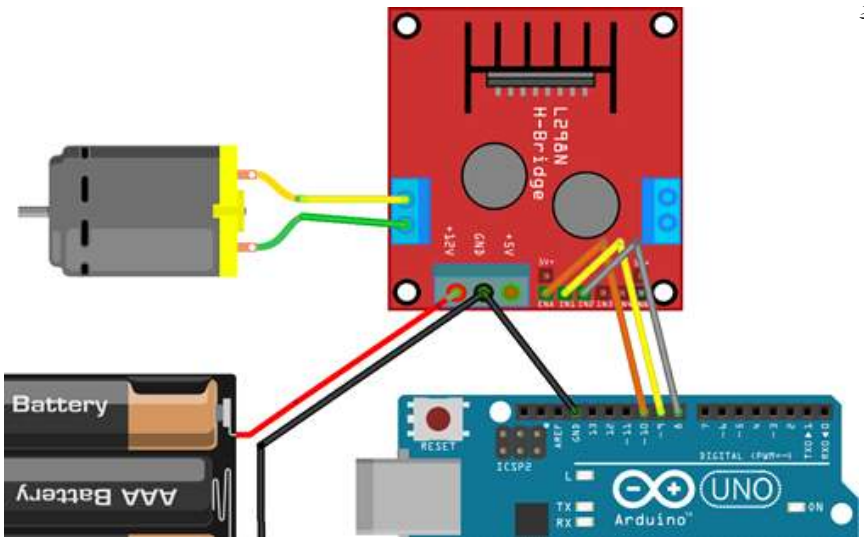
4, 6 : GND
5 : 5V

1, 2, 13, 14 : 모터

1	DC Motor 1 + / Stepper Motor A+	8	IN1
2	DC Motor 1 − / Stepper Motor A−	9	IN2
3	12V jumper (remove if >12V)	10	IN3
4	motor supply voltage (max 35V)	11	IN4
5	GND	12	DC motor 2 enable jumper Connect to PWM output for DC motor speed control
6	5V output if 12V jumper in place	13	DC Motor 2 + / Stepper Motor B+
7	DC motor 1 enable jumper Connect to PWM output for DC motor speed control	14	DC Motor 2 − / Stepper Motor B−

DC motor	rotate	IN1	IN2	IN3	IN4	speed adjust PWM signal	
						end	end
M1	corotation	high	low	/	/	high	/
	reversal	low	high	/	/	high	/
	stop	low	low	/	/	high	/
M2	corotation	/	/	high	low	/	high
	reversal	/	/	low	high	/	high
	stop	/	/	low	low	/	high

(corotation : 동시 회전, reversal : 전환)



주의점

- 회로가 되게 만들어야 함
- 회로의 전압차가 동일해야 함
- 아두이노와 모터의 그라운드가 연결되어야 함.

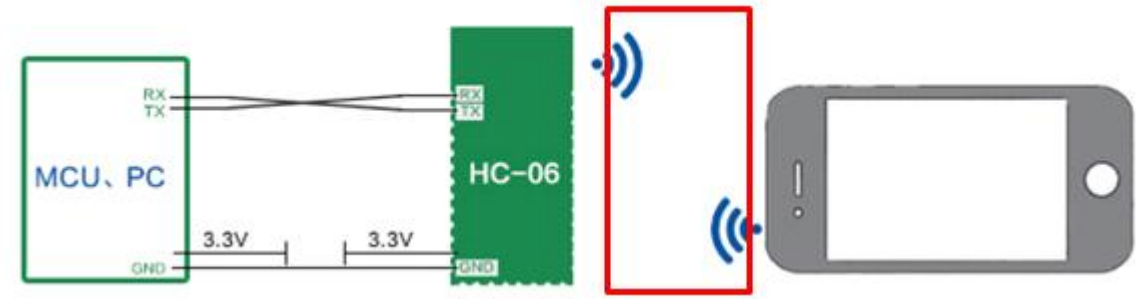
```
// motor one
int enA = 10;
int in1 = 9;
int in2 = 8;

void setup()
{
  // set all the motor control pins to outputs
  pinMode(enA, OUTPUT);
  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);
}

void demoOne() //속도 조절 코드
{
  // this function will run the motors in both directions at a fixed speed
  // turn on motor A
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);
  // set speed to 200 out of possible range 0~255
  analogWrite(enA, 200);
  delay(3000);
  // now turn off motors
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);
  delay(250);
  // now change motor directions
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
  delay(3000);
  // now turn off motors
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);
}
```

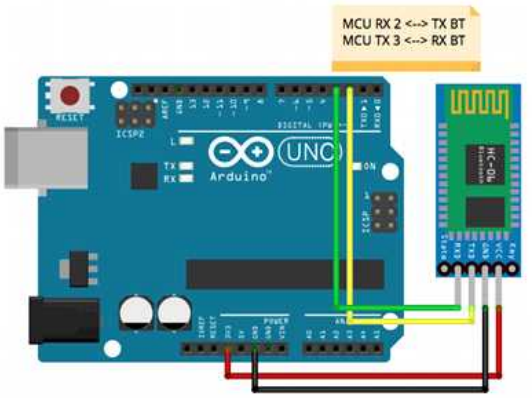
```
void demoTwo() //방향 조절 코드
{
  // this function will run the motors across the range of possible speeds
  // note that maximum speed is determined by the motor itself and the operating voltage
  // the PWM values sent by analogWrite() are fractions of the maximum speed possible
  // by your hardware
  // turn on motors
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
  // accelerate from zero to maximum speed
  for (int i = 0; i < 256; i++)
  {
    analogWrite(enA, i);
    delay(20);
  }
  // decelerate from maximum speed to zero
  for (int i = 255; i >= 0; --i)
  {
    analogWrite(enA, i);
    delay(20);
  }
  // now turn off motors
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);
}

void loop()
{
  demoOne();
  delay(1000);
  demoTwo();
  delay(1000);
}
```

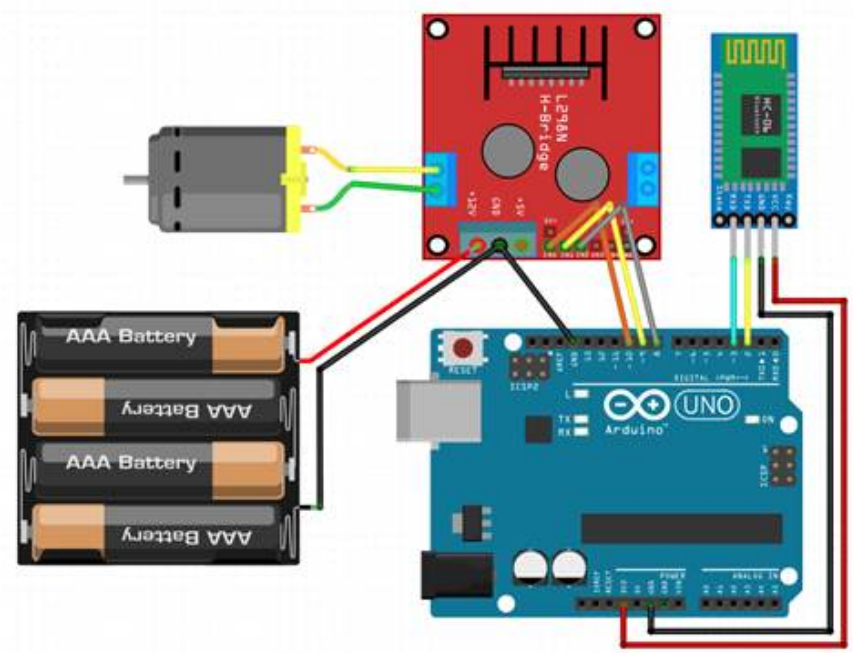
11. HC06 (p.34~39)

- Bluetooth Slave UART Module
- UART : Universal Asynchronous Receiver/Transmitter == Serial Communication

- UART 통신 방법 = 사람이 대화하는 원리
- Rx(데이터 수신), Tx(데이터 송신), GND 연결
- Rx-Tx 교차 연결
- 비동기 통신 : Baud Rate 일치 필요








```
//블루투스 설정 코드
#include <SoftwareSerial.h>

SoftwareSerial BT(2, 3);

void setup() {
  Serial.begin(9600);
  BT.begin(9600);
}

void loop() {
  if (BT.available())
    Serial.write(BT.read());
  if (Serial.available())
    BT.write(Serial.read());
}
```

Command	Response	Comment
AT	OK	Used to verify communication
AT+VERSION	OKlinvorV1.8	The firmware version (version might depend on firmware)
AT+NAMExyz	OKsetname	Sets the module name to “xyz”
AT+PIN1234	OKsetPIN	Sets the module PIN to 1234
AT+BAUD1	OK1200	Sets the baud rate to 1200
AT+BAUD2	OK2400	Sets the baud rate to 2400
AT+BAUD3	OK4800	Sets the baud rate to 4800
AT+BAUD4	OK9600	Sets the baud rate to 9600

- SoftwareSerial mySerial(RX pin, TX pin)

12. Motor (p.40~42)

DC 모터: 주변에서 흔히 보는 모터. 입력 전류(+, −) 방향으로 회전 방향 제어. 상대적으로 고회전에 유리. 회전 움직임을 사용하는 RC카, 쿼드콥터 등 사용처가 매우 다양. 회전수와 방향을 자유롭게 제어하기 위해서 별도 드라이버 모듈 필요.

스텝 모터: 회전 방향과 속도 뿐 아니라 회전각을 정밀히 제어 가능. DC 모터와 서보 모터의 장점을 합친 모터. 제어가 복잡함. 보통 스텝모터 드라이버 모듈 이용해 제어. 상대적으로 고회전이 필요치 않으면서 정밀한 제어가 필요한 곳에 사용. 3D 프린터 움직임을 만드는 핵심 모터.

서보 모터: 보통 0~180도 사이를 움직이며, 해당 회전 범위 안에서 위치 설정 가능. 동작 범위가 제한적이지만 정확한 위치 제어 가능. 제어 방법도 간단 (PWM 신호로 위치제어). RC카의 방향타, 로봇 관절 등 회전각 제어 필요한 곳에 광범위하게 사용.