

# 아두이노를 이용한 스마트 센서 제어 실습1

한백전자 기술연구소



HANBACK ELECTRONICS CO.,LTD



## 학습3

# 아두이노를 이용한 스마트 센서제어 실습1

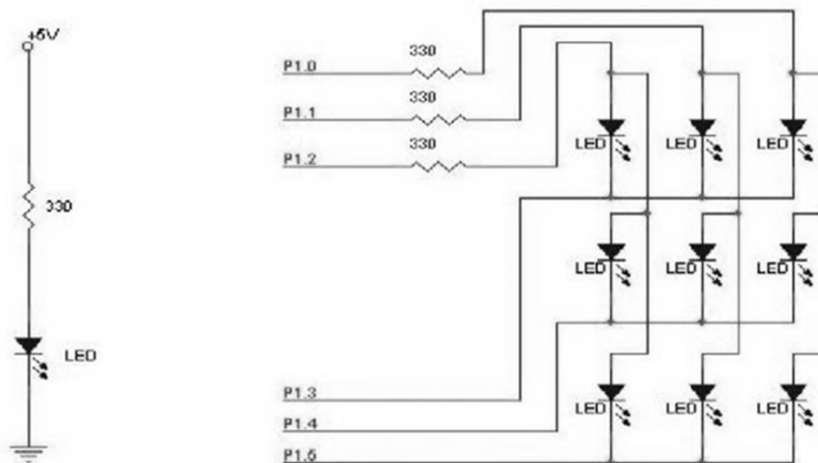


- LED제어 실습하기
- 인체감지 센서 제어 실습하기
- 사운드센서 제어 실습하기
- Buzzer 제어 실습하기
- DC Motor 제어 실습하기
- Step Motor 제어 실습하기



## 1.LED구조

- LED는 Light-emitting diode의 약자로서, 발광다이오드라 부른다.  
이는 빛을 발산하는 반도체 소자를 말하며, 방출하는 빛의 색깔은 크리스털 도핑 (Crystal Doping)의 양과 종류에 따라 적색, 녹색 또는 황색등이 될 수 있다.
- ED는 Pn 접합을 하는 다이오드이고 순방향에 전류를 흘리는 것에 따라 전자와 정공이 재결합하여 발광하는 소자이다.
- LED 의 모양은 다리가 긴 부분이 양극 (Anode), 짧은 쪽이 음극 (Cathode) 이다.



LED의 정적구동 방식(좌) 및 동적구동 방식(우)



LED 모듈 외형	모듈 항목	모듈 항목의 내용
	구성	RED, GREEN, BLUE LED 5PI 각각 1개
	허용전압 및 소비전류	2.0~5V, 20mA
	동작 전압	3.3V~5V
LED를 이용한 출력 가능한 모듈		



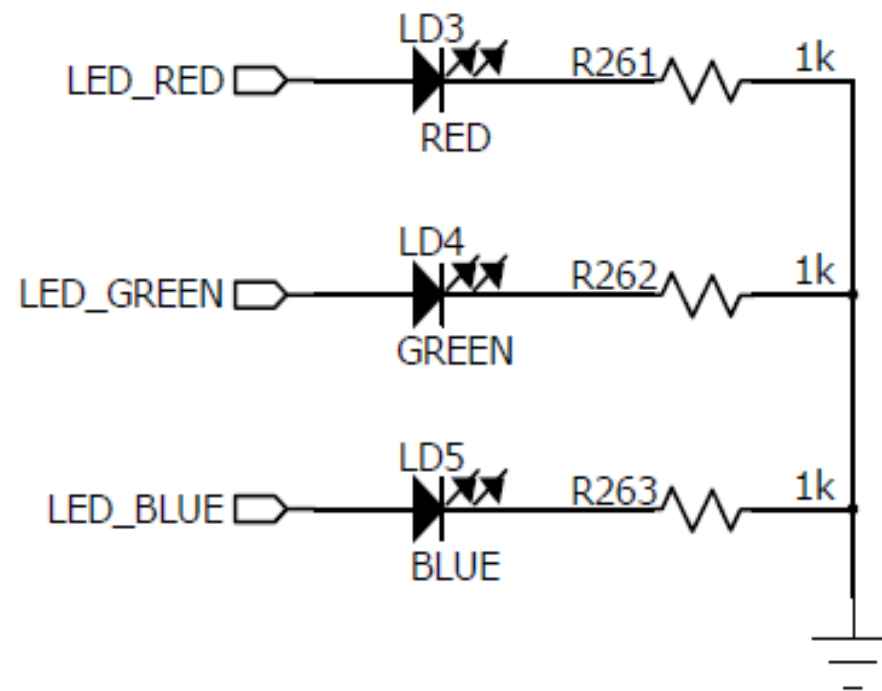
# HBE-ADK-2560과 LED 모듈의 연결 방법



HANBACK ELECTRONICS.,LTD

- 두 모듈을 연결하기 위해서는 3핀 케이블로 그림과 같이 HBE-ADK-2560과 LED 모듈을 연결해야 한다.  
J24 커넥터의 D11, D12, D13 과 J15번 커넥터의 LED\_RED, LED\_GREEN, LED\_BLUE에 각각 연결한다.
- HBE-ADK-2560과 LED/스위치 모듈의 핀 연결 정보 와 회로도

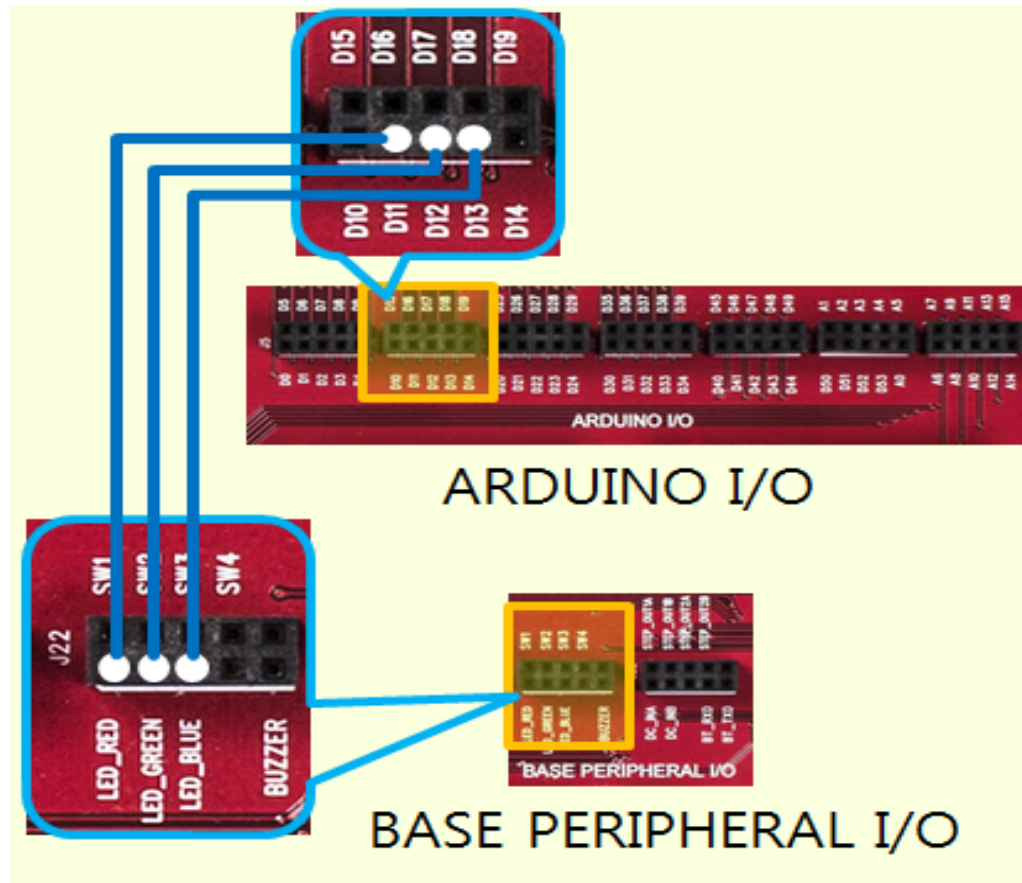
HBE-ADK-2560 모듈 핀 번호	핀 정보	LED 모듈 핀 번호
D11	Digital pin(PWM)	LED_RED
D12	Digital pin(PWM)	LED_GREEN
D13	Digital pin(PWM)	LED_BLUE





## • LED 모듈 연결

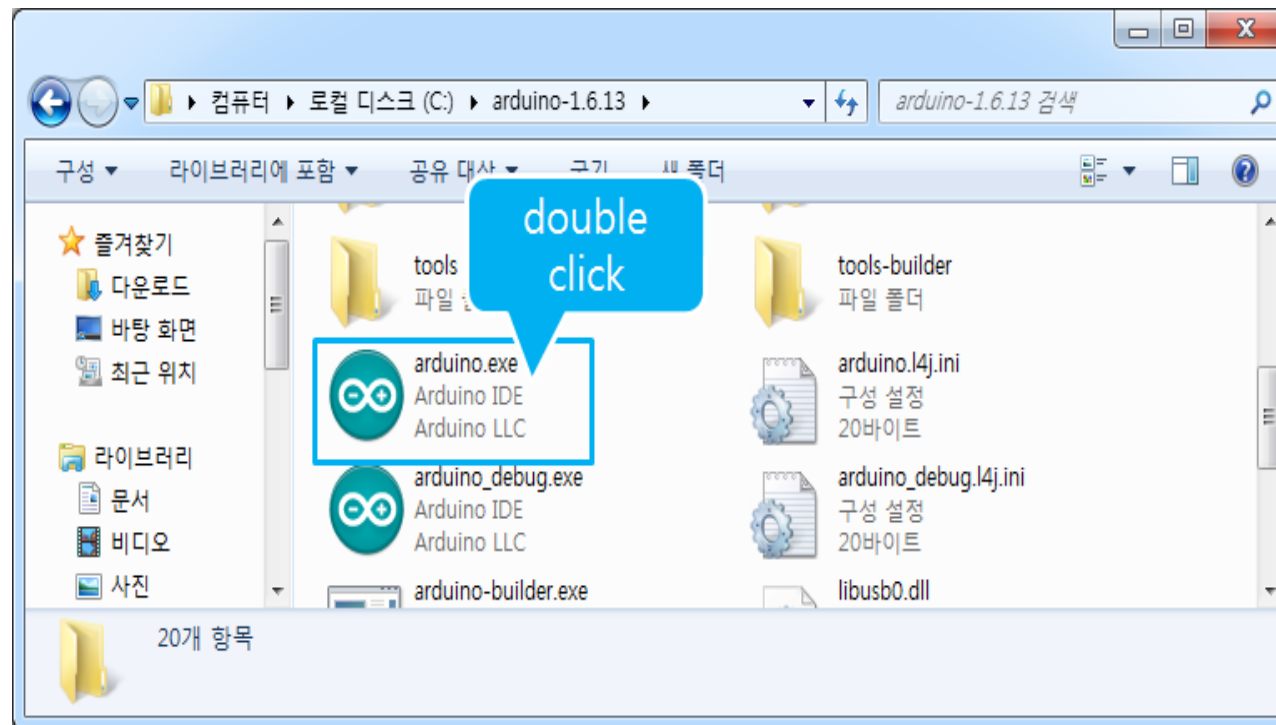
HBE-ADK-2560에 전원을 인가하지 않은 상태에서 그림과 같이 LED 모듈을 연결한다.





## 1. 아두이노를 실행한다.

그림과 같이 C:\Warduino-1.6.13\Warduino.exe 를 실행한다.

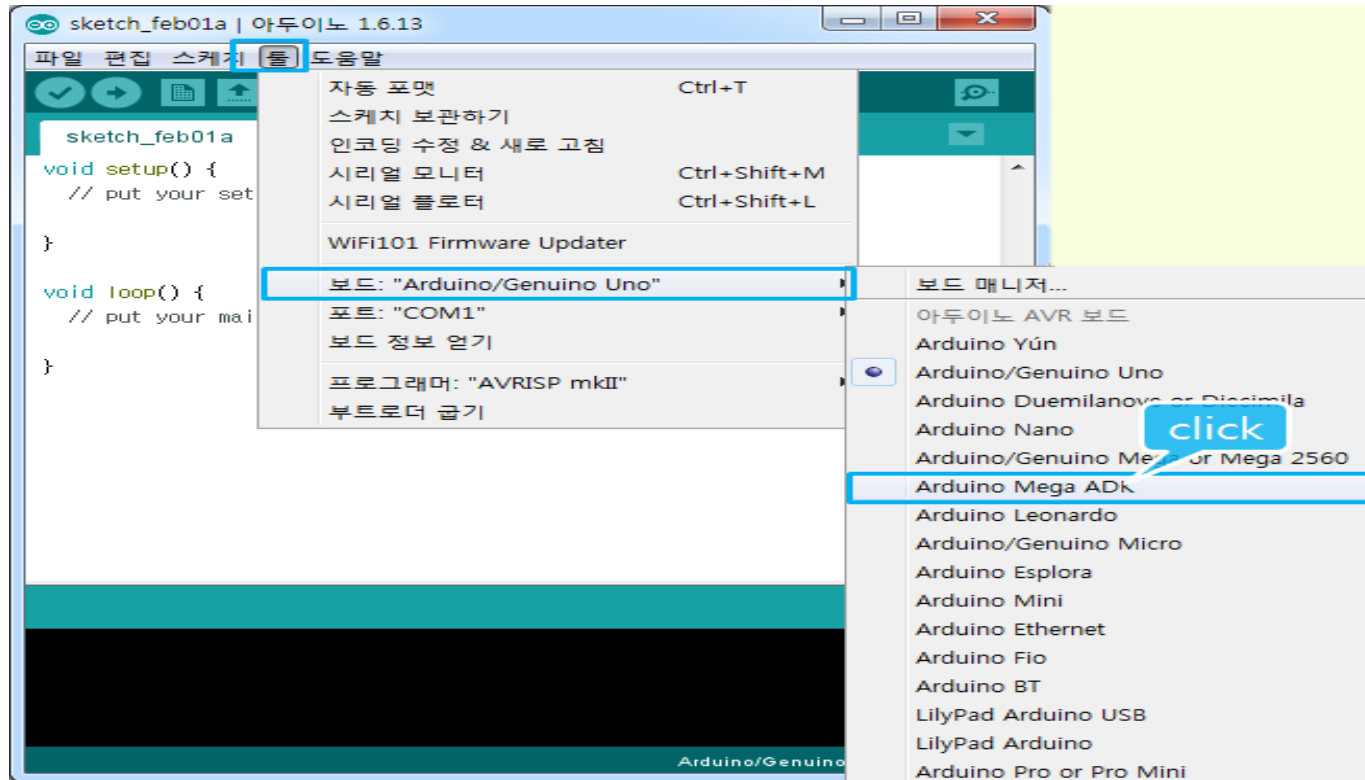






## 2. 아두이노를 설정한다.

그림과 같이 툴에서 보드를 Arduino Mega ADK로 설정한다.





# LED 모듈 실습 프로그램 기본 설정 및 확인



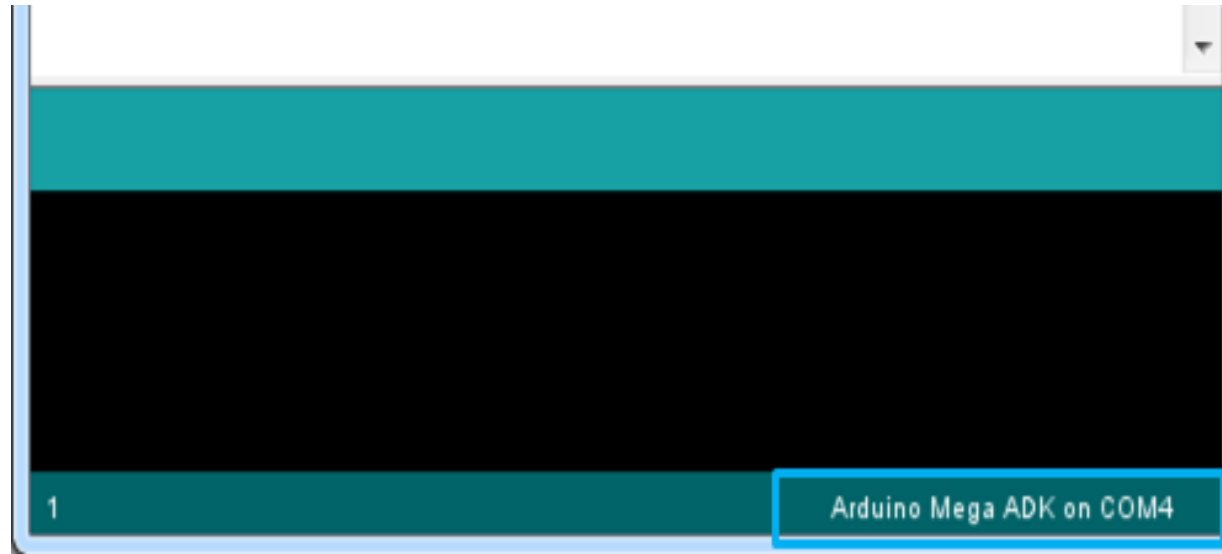
HANBACK ELECTRONICS.,LTD

- 그림과 같이 툴에서 포트를 COM4(장치관리자에서 확인)로 설정한다.





- 만약 아래 그림과 같이 아두이노(Arduino)에 표시된 다음 부분의 설정 값과 같으면 보드 및 시리얼 포트 설정은 생략해도 된다.





## 1. LED RED 제어 프로그램 작성 PIONEER\_LIGHT\_LED\_RED.ino

```
int pin_LED_R = 11;

int pin_LED_G = 12;

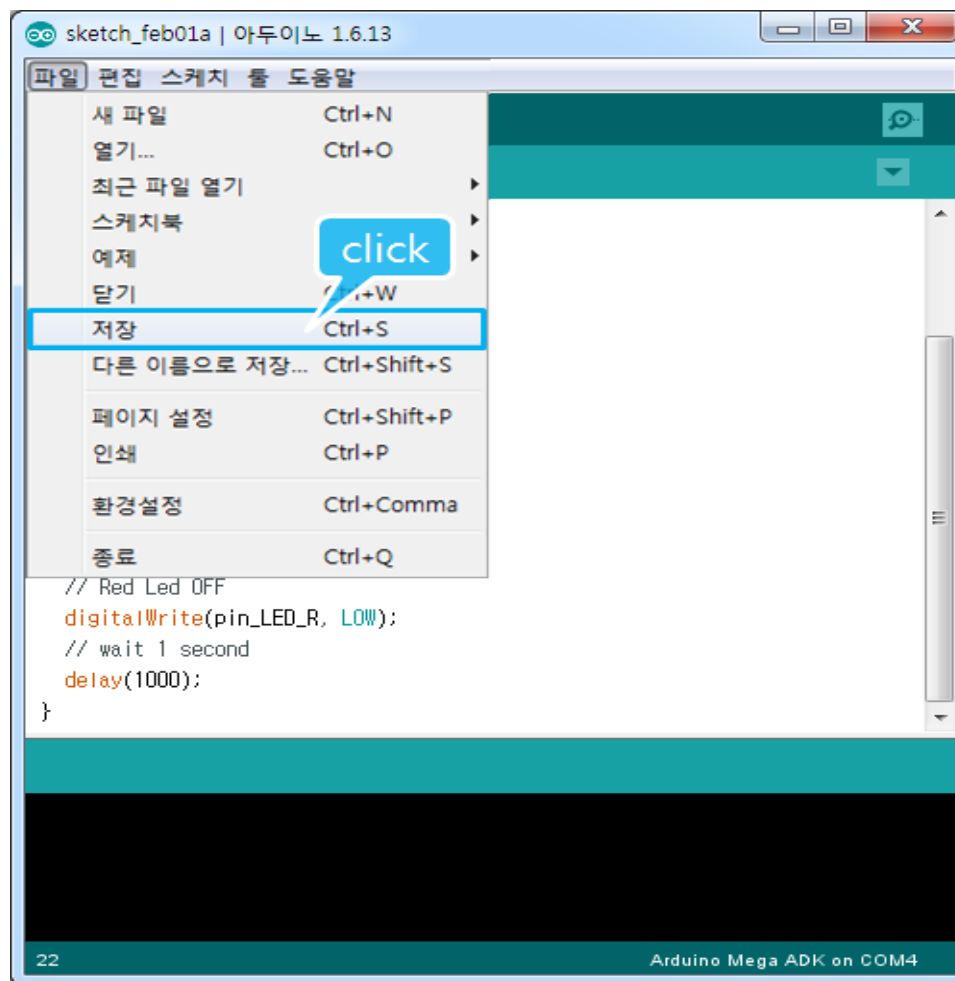
int pin_LED_B = 13;


void setup() {
  pinMode(pin_LED_R, OUTPUT);
}


void loop() {
  digitalWrite(pin_LED_R, HIGH);
  delay(1000);
  digitalWrite(pin_LED_R, LOW);
  delay(1000);
}
```

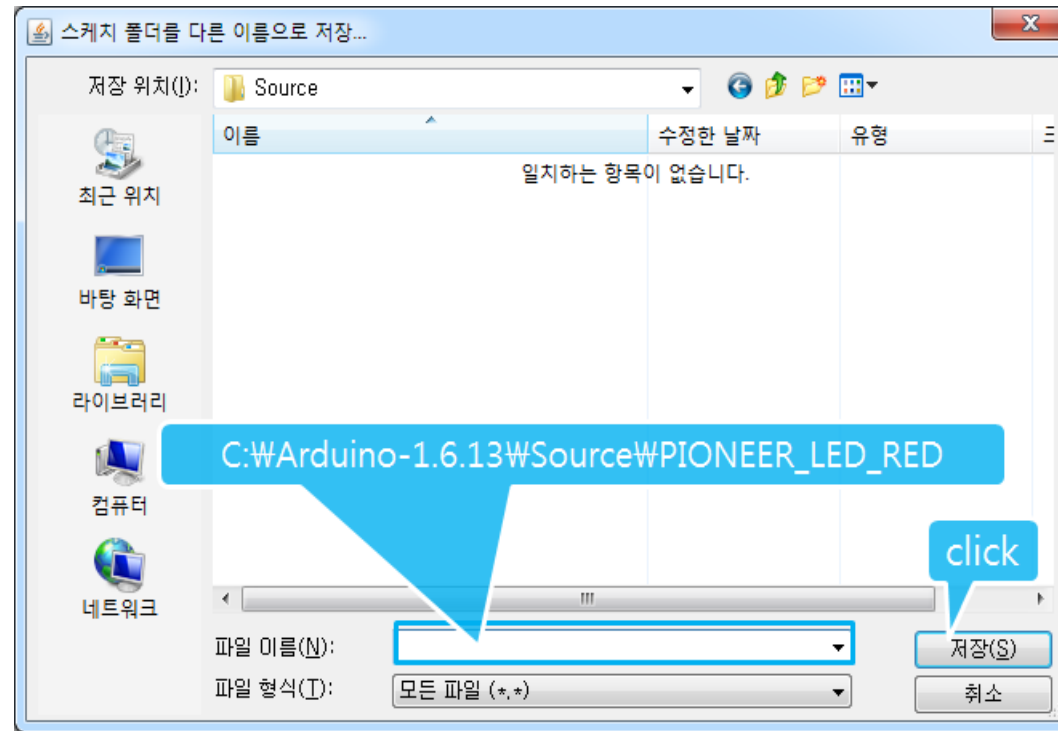


## 2. LED 제어 프로그램 저장(작성완료후 파일 → 저장을 클릭한다.)





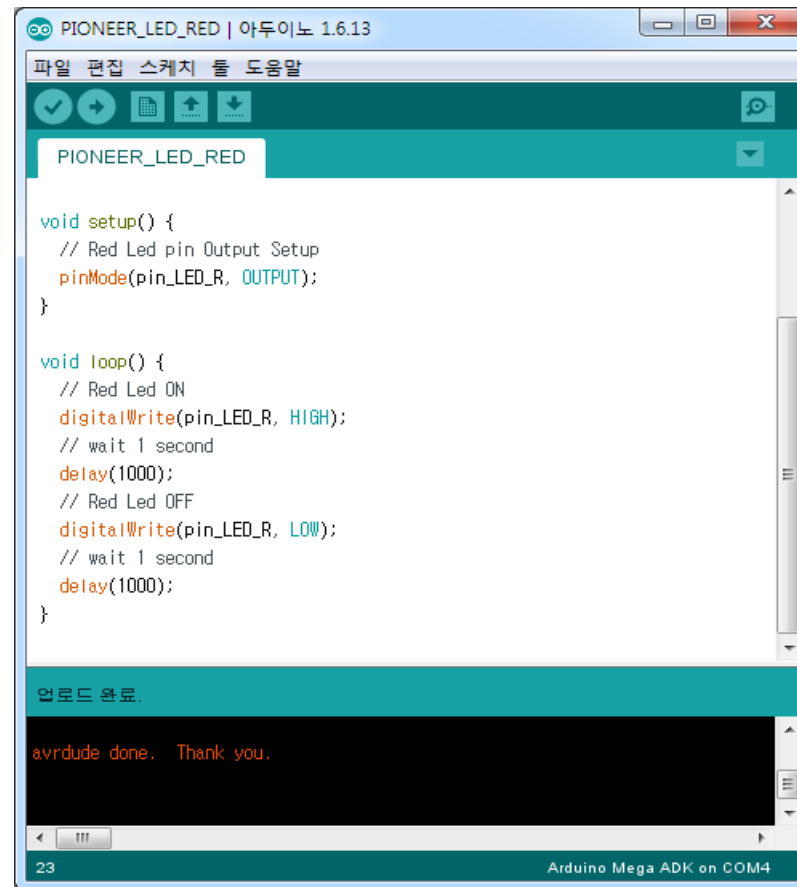
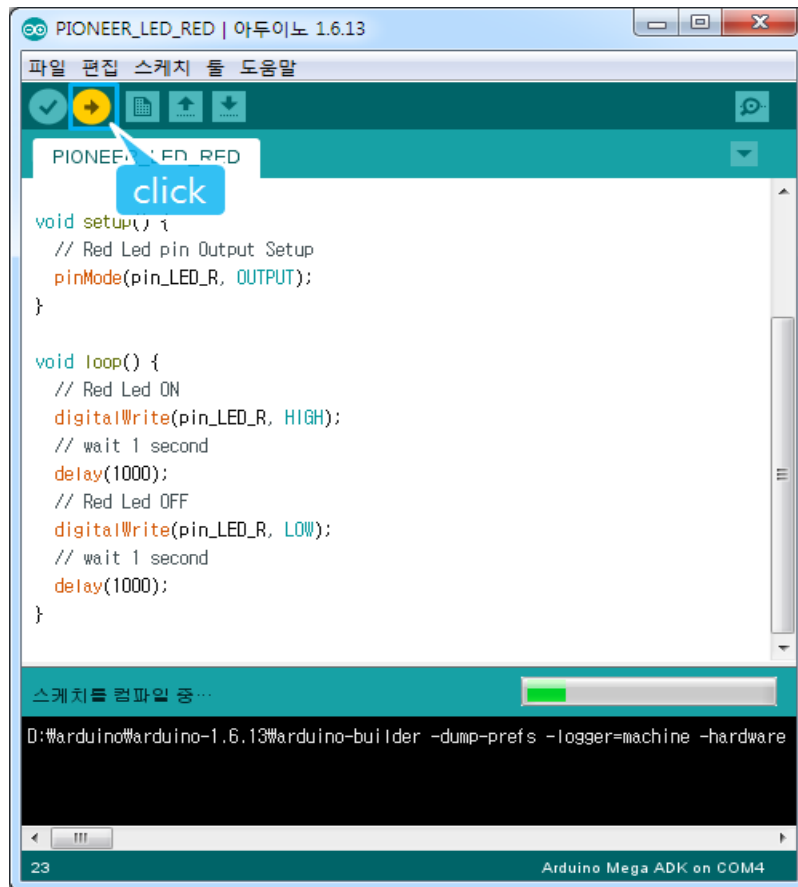
- 아래와 같이 저장하고자 하는 드라이브 및 폴더를 선택하고 PIONEER\_LED\_RED 저장한다. 여기서는 C:\WArduino-1.6.13\Source\WPIONEER\_LED\_RED로 저장하여 C:\WArduino-1.6.13\Source 폴더에 저장하도록 하였다.





## 1.LED 프로그램의 컴파일 및 업로드

LED 프로그램의 저장이 완료되면 그림과 같이 업로드 버튼을 클릭하여 LED 프로그램의 컴파일 및 업로드를 실행한다.





## 1. PIONEER\_LED\_RED.ino

이 프로그램은 빨간색 LED를 ON/OFF하는 프로그램이다.

### 1) 핀 설정

사용할 핀의 번호를 알아보기 쉽도록 변수에 저장한다.

```
int pin_LED_R = 11;  
int pin_LED_G = 12;  
int pin_LED_B = 13;
```

### 2) 초기화 구문

빨간색 LED 핀은 출력으로 설정한다.

```
// Red Led pin Output Setup  
pinMode(pin_LED_R, OUTPUT);
```





## 3) LED 제어

빨간색 LED 핀을 HIGH를 출력하여 LED를 켜다.

```
// Red Led ON  
digitalWrite(pin_LED_R, HIGH);  
  
// wait 1 second  
delay(1000);
```

빨간색 LED 핀을 LOW를 출력하여 LED를 끈다.

```
// Red Led OFF  
digitalWrite(pin_LED_R, LOW);  
  
// wait 1 second  
delay(1000);
```

## 1. PIONEER\_LED\_RED 동작 확인

프로그램 업로드가 완료된 후 빨간색 LED가 1초 간격으로 주기적으로 ON/OFF 한다.



## 1.LED ALL 제어 프로그램 작성

PIONEER\_LIGHT\_LED\_ALL.ino

```
int pin_LED[3] = {11, 12, 13};  
  
void setup() {  
  char i;  
  for(i=0; i<3; i++)  
  {pinMode(pin_LED[i], OUTPUT);}  
}  
  
void loop() {  
  digitalWrite(pin_LED[0],1);  
  digitalWrite(pin_LED[1],1);  
  digitalWrite(pin_LED[2],1);  
  delay(500);  
  digitalWrite(pin_LED[0],0);  
  digitalWrite(pin_LED[1],0);  
  digitalWrite(pin_LED[2],0);  
  delay(500);  
}
```



## 1. PIONEER\_LED\_ALL.ino

이 프로그램은 빨간색, 녹색 및 파란색 LED를 ON/OFF하는 프로그램이다.

### 1) 핀 설정

배열 변수를 선언하여 R,G,B의 LED 핀 번호를 저장한다.

```
// Red, Green, Blue LED  
int pin_LED[3] = {11, 12, 13};
```

### 2) 초기화 구문

R, G, B LED 핀은 출력으로 설정한다.

```
void setup() {  
  char i;  
  for(i=0; i<3; i++)  
  {  
    pinMode(pin_LED[i], OUTPUT);  
  }  
}
```



## 3) LED ON/OFF

LED를 0.5초주기로 ON/OFF 한다.

```
void loop() {  
  digitalWrite(pin_LED[0],1);  
  digitalWrite(pin_LED[1],1);  
  digitalWrite(pin_LED[2],1);  
  delay(500);  
  digitalWrite(pin_LED[0],0);  
  digitalWrite(pin_LED[1],0);  
  digitalWrite(pin_LED[2],0);  
  delay(500);  
}
```

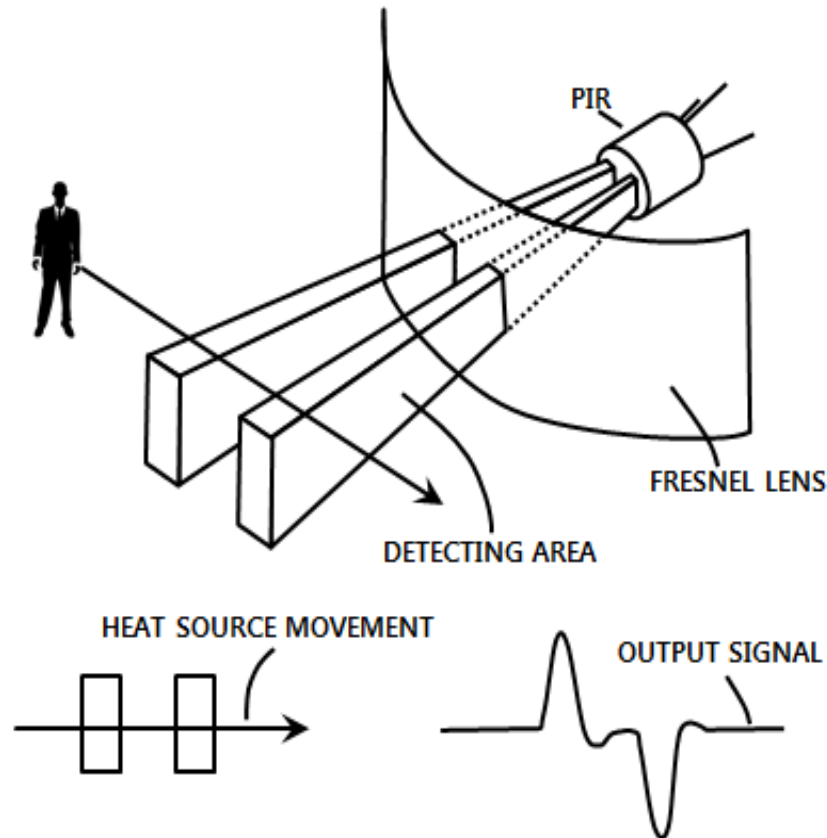
### 1. PIONEER\_LED\_ALL 동작 확인

프로그램 업로드가 완료된 후 빨간색, 녹색 및 파란색 LED가 0.5초 간격으로 주기적으로 ON/OFF 한다.



## 1. 인체감지 센서의 구조와 구동 방법

인체 감지 센서(PIR Sensor: Passive Infrared Sensor)는 적외선 센서의 한 종류로써 인체 적외선 센서라고 한다. 이 센서는 인간의 몸이나 동물에서 나오는 적외선을 감지하였을 때 사용자에게 신호를 보내준다.





## 2. 인체감지 센서 모듈의 사양

인체감지 센서의 하드웨어 사양은 <표 3-5>와 같다.

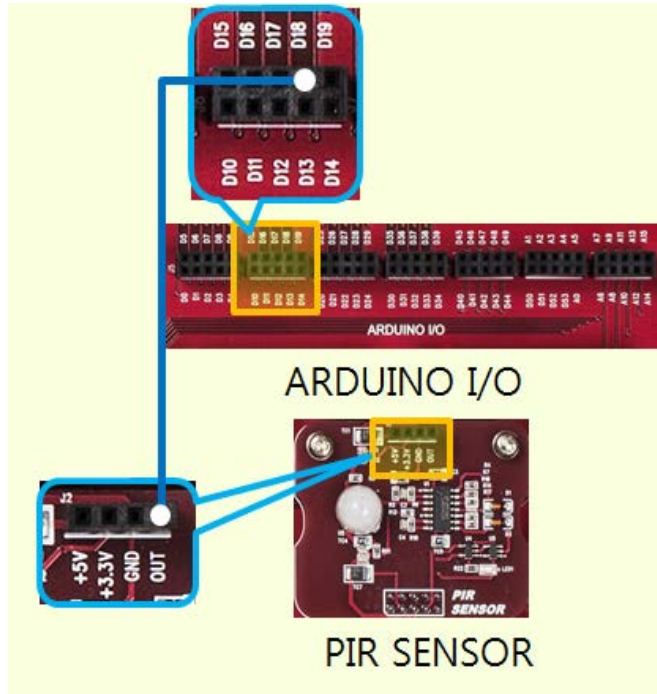
인체 감지 센서 외형	모듈 항목	모듈 항목의 내용
	적외선 센서	RE200B
	센싱 범위	110 degree
	동작 전압	3.3V
	입력/출력 핀	3 핀 헤더 1개(2.54mm 간격)
	크기	27x33mm

<표 3-5> 인체감지 센서 모듈의 사양



# HBE-ADK-2560과 인체감지 센서의 연결 방법

- 두 모듈을 연결하기 위해서는 1핀 케이블로 [그림 3-16]과 같이 HBE-ADK-2560과 인체 감지 센서 모듈을 연결해야 한다.  
J6 커넥터의 D18과 PIR SENSOR 모듈의 커넥터의 OUT에 연결한다.  
<표 3-4>는 HBE-ADK-2560과 인체 감지 센서 모듈의 핀 연결을 정리한 것이다.



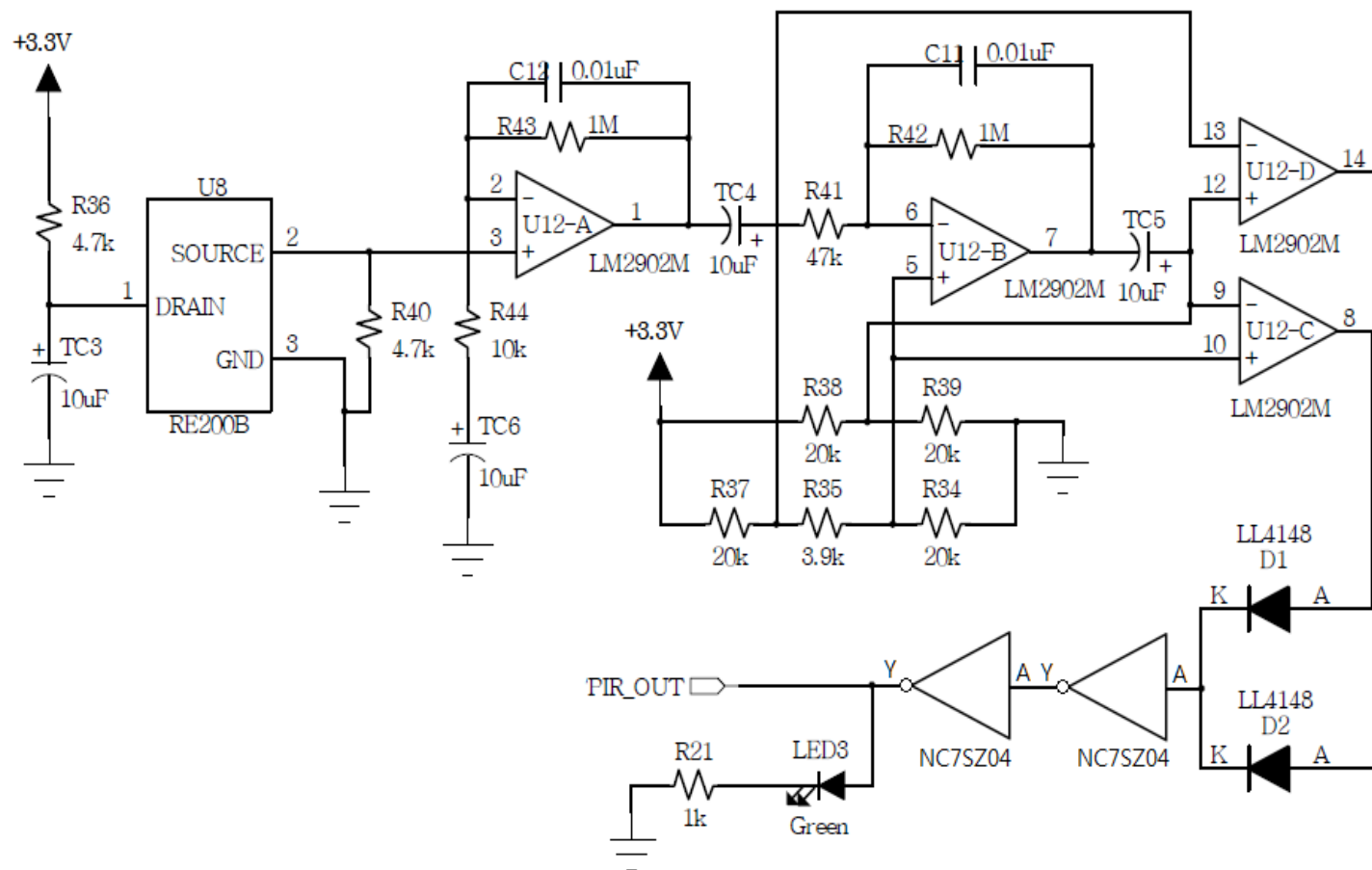
< 그림3-16 >

HBE-ADK-2560 모듈 핀 번호	핀 정보	사운드 센서 핀 번호
D18	Digital pin(EXT INT)	PIR_OUT

< 표3-4 >



# 인체감지 센서 모듈의 회로도







## 1.인체 감지 프로그램 작성

PIONEER\_LIGHT\_PIR.ino

```
int pin_PIR = 18;
uint16_t PIR_flag = 0;

void setup() {
  Serial.begin(115200);
  pinMode(pin_PIR, INPUT);
  attachInterrupt(digitalPinToInterrupt(pin_PIR), PIR_ISR, RISING);
}

void loop() {
  if(PIR_flag == 1){
    Serial.println("Detected");
    PIR_flag = 0;

  }else{

    Serial.println("Not detected");
  }
  delay(500);
}

void PIR_ISR(void){
  PIR_flag = 1;
}
```



## 1.PIONEER\_LIGHT\_PIR.ino

### 1) 핀 설정

사용할 핀의 번호를 알아보기 쉽도록 변수에 저장한다.

```
int pin_PIR = 18; // Pir pin number 18 setup  
uint16_t PIR_flag = 0;
```

### 2) 초기화 구문

시리얼 통신 속도를 115200으로 설정한다.

```
void setup() {  
  Serial.begin(115200);
```

사용할 핀을 입력으로 설정한다.

```
  pinMode(pin_PIR, INPUT);
```

PIR 핀이 상승엣지에서 인터럽트 발생 및 인터럽트 발생시 PIR\_ISR 함수가 실행되도록 설정한다.

```
  attachInterrupt(digitalPinToInterrupt(pin_PIR), PIR_ISR, RISING);
```



## 3) 실행 구문

PIR\_flag 값이 1인 경우에 인체가 감지되었다고 판단하고, "Detected" 문자열을 시리얼로 전송한다.

또한 PIR\_flag 값이 0이면 인체가 감지되지 않았다고 판단하고, "Not detected" 문자열을 시리얼로 전송한다.

```
void loop() {  
  if(PIR_flag == 1)  
  {  
    // Pir Detected !!  
    Serial.println("Detected");  
    PIR_flag = 0;  
  }  
  else  
  {  
    // Pir Not detected !!  
    Serial.println("Not detected");  
  }  
  delay(500);  
}
```



## 3) 외부 인터럽트 함수

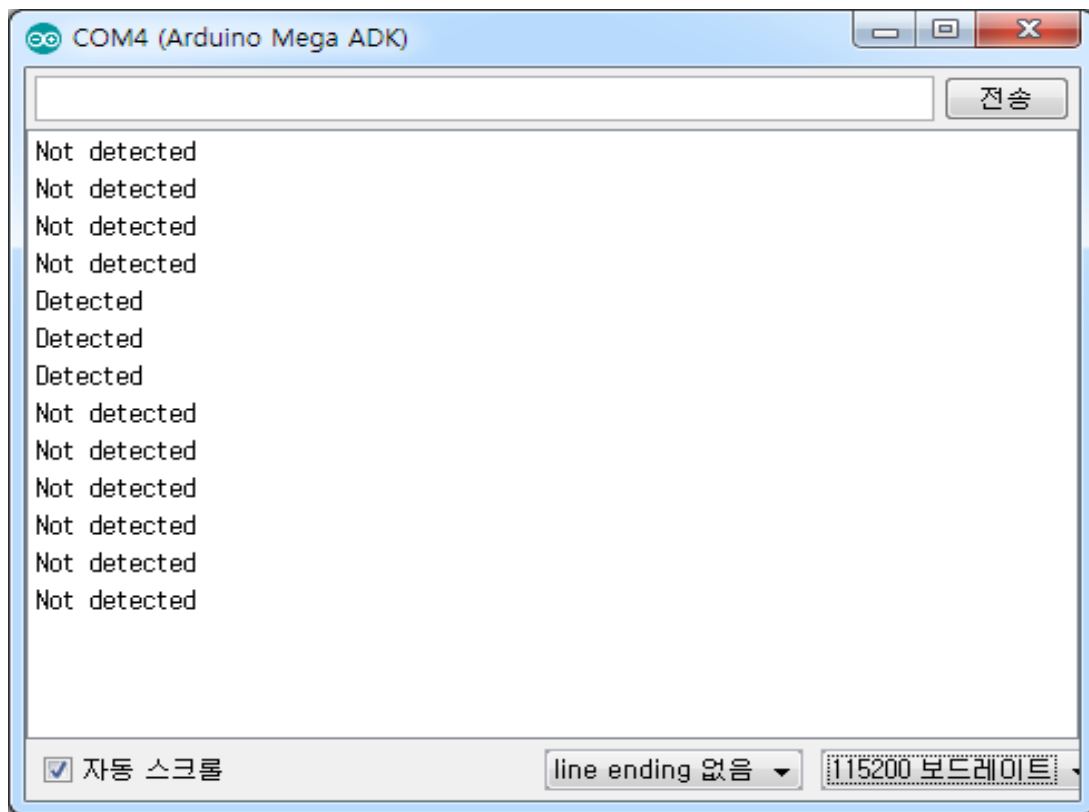
외부 인터럽트가 발생하면 실행되는 함수로 PIR\_flag 값을 1 증가시킨다.

```
void PIR_ISR(void){  
  
    PIR_flag = 1;  
  
}
```



## 1.시리얼 모니터 실행 및 동작 확인

- 프로그램 업로드가 완료되면 시리얼 모니터를 활성화 한 후 통신 속도를 115200으로 설정한다.
- 시리얼 모니터에 인체 감지 센서의 동작이 감지되면 "Detected", 감지되지 않았을 때는 "Not Detected"이 출력된다.



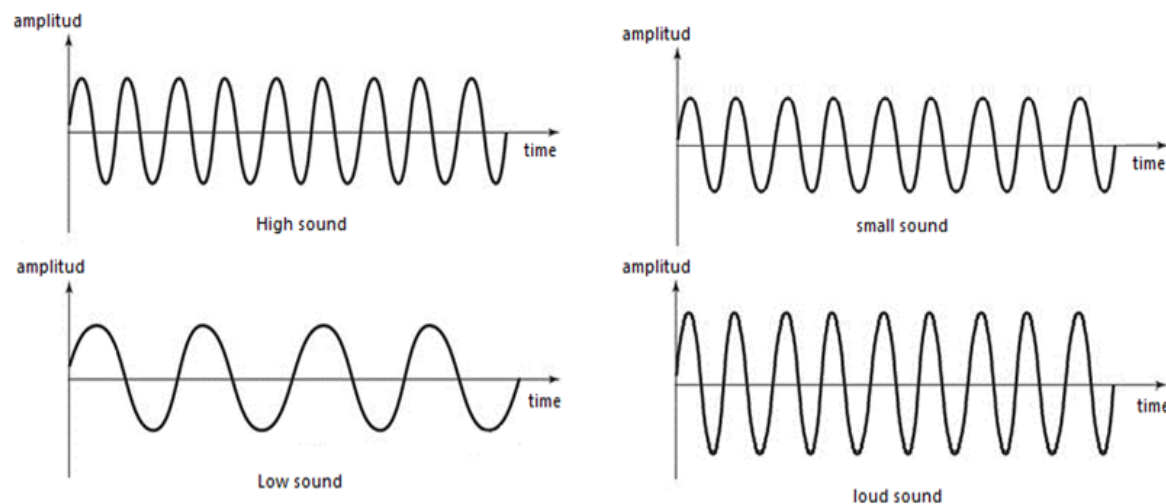


## 1. 사운드 센서의 구조와 구동 방법

사운드(Sound)는 물체의 진동이나 기체의 흐름에 의하여 발생하는 파동의 일종이다.

소리의 세기는 소리 진폭에 따라 결정된다.

진폭이 클수록 큰 소리이고 진폭이 작을수록 작은 소리이다.




사운드 센서(Sound Sensor) 는 주변의 소리를 감지하는 센서이다.

센서의 전면에 마이크가 연결되어 있다. 이 마이크를 통해서 주변의 소리를 감지하게 되며, 우측의 저항 값을 조절하여 센서의 민감도를 조절해 줄 수 있다.



## 2. 사운드 센서 모듈의 사양

사운드 센서의 하드웨어 사양은 <표 3-5>와 같다.

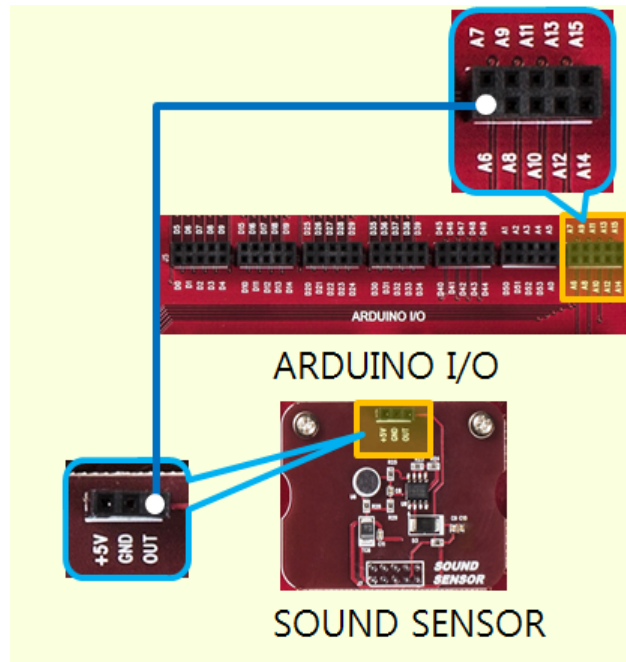
사운드 센서 외형	모듈 항목	모듈 항목의 내용
	사운드 센서	Microphone
	동작 전압	5V
	I/O Interface	1 Analog , 1 Digital OUTPUT
소리를 감지하는 센서		

<표 3-5> 사운드 센서 모듈의 사양



# HBE-ADK-2560과 사운드 센서의 연결 방법

- 두 모듈을 연결하기 위해서는 1핀 케이블로 [그림 3-21]과 같이 HBE-ADK-2560과 사운드 센서 모듈을 연결해야 한다.  
J29번 커넥터의 A6을 J19번 커넥터의 SOUND\_A에 연결한다.  
<표 3-6>는 HBE-ADK-2560과 사운드 센서 모듈의 핀 연결을 정리한 것이다.



< 그림3-21 >

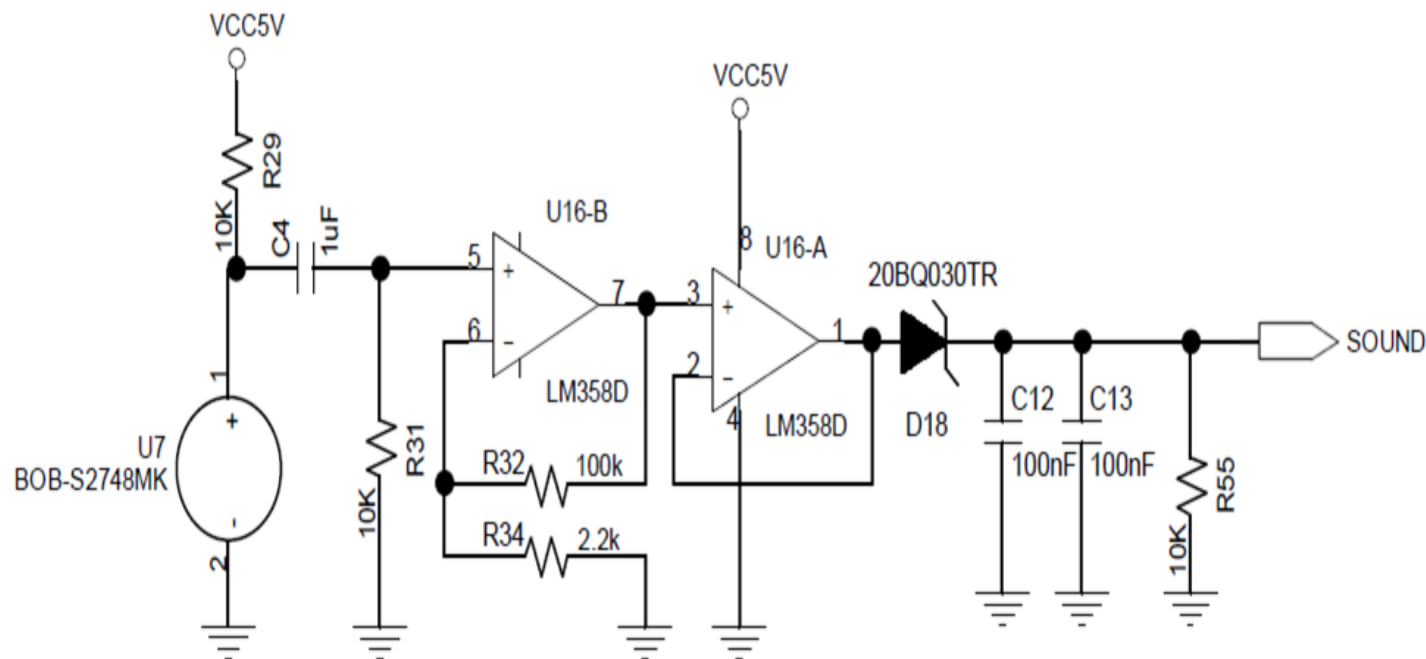
HBE-ADK-2560 모듈 핀 번호	핀 정보	사운드 센서 핀 번호
A6	Analog pin	SOUND_A

< 표3-6 >





# 사운드센서 모듈의 회로도





## 1.소리 감지 프로그램 작성

PIONEER\_LIGHT\_SOUND.ino

```
int pin_SOUND = A6;

void setup() {
  Serial.begin(115200);
  pinMode(pin_SOUND, INPUT);
}

void loop() {
  int ADC_data = analogRead(pin_SOUND);
  Serial.print("ADC Data : ");
  Serial.print(ADC_data);
  if(ADC_data > 511)
  {Serial.println(" Noise");}
  else
  {
    Serial.println(" Quiet");
  }
  delay(500);
}
```



## 1.PIONEER\_LIGHT\_SOUND.ino

### 1) 핀 설정

사용할 핀의 번호를 알아보기 쉽도록 변수에 저장한다.

```
int pin_SOUND = A6;
```

### 2) 초기화 구문

시리얼 통신 속도를 115200으로 설정한다.

```
void setup() {  
  Serial.begin(115200);
```

사용할 핀을 입력으로 설정한다.

```
// Sound pin Input Setup  
pinMode(pin_SOUND, INPUT);
```



## 3) 실행 구문

Sound ADC핀을 읽어 측정 값을 시리얼로 전송한다.

```
void loop() {  
  int ADC_data = analogRead(pin_SOUND);  
  Serial.print("ADC Data : ");  
  Serial.print(ADC_data);
```

측정값이 511 보다 크면 소리가 감지되었다고 판단하고,  
"Noise" 문자열을 시리얼로 전송한다.

```
  if(ADC_data > 511)  
  {  
    Serial.println(" Noise");  
  }
```

측정 값이 511 보다 작거나 같으면 소리가 감지되지 않았다고 판단하고,  
"Quiet" 문자열을 시리얼로 전송한다.

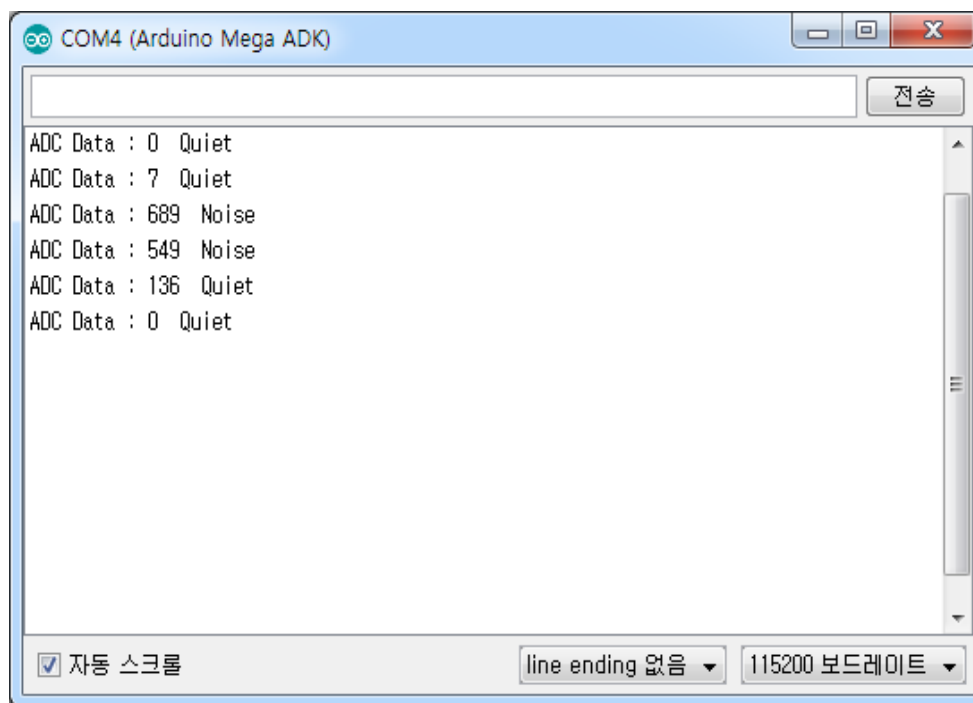
```
  else  
  {Serial.println(" Quiet");}  
  delay(500);}
```



## 1.시리얼 모니터 실행 및 동작 확인

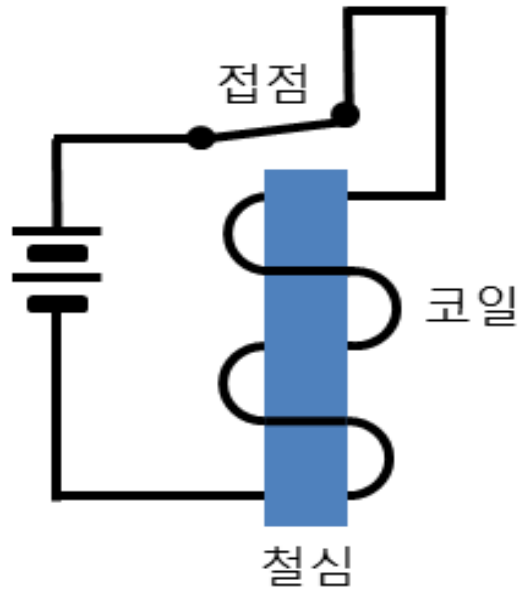
- 프로그램 업로드가 완료되면 시리얼 모니터를 활성화 한 후 통신 속도를 115200으로 설정한다.
- 그림과 같이 Sound 센서에서 소리를 감지하여 아날로그 데이터를 변환한 데이터와 현재 상태를 출력한다.

소리가 감지되면 "Noise"를 감지되지 않는다면 "Quiet"를 출력한다.

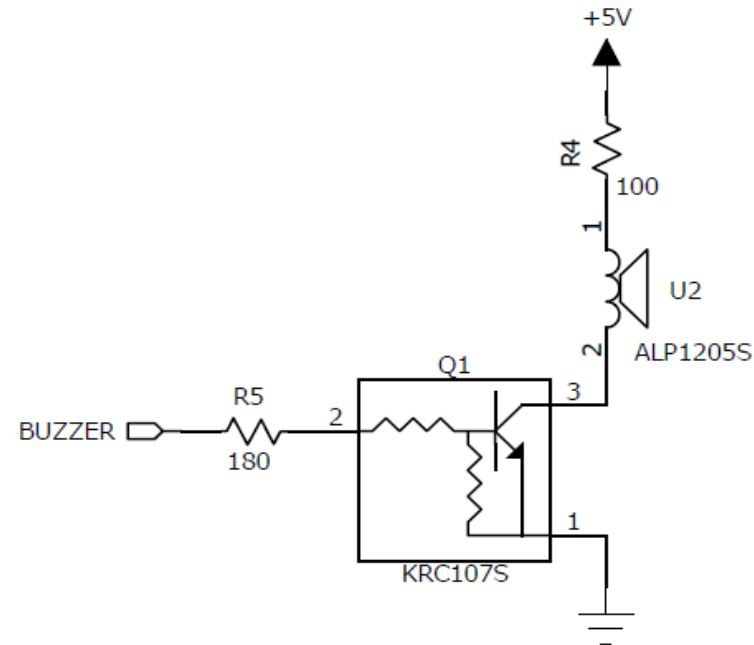


## 1. Buzzer의 구조와 구동 방법

- Buzzer는 신호를 주면 소리를 발생 시키는 모듈이다.  
이 모듈은 사용자에게 특정 상황 등을 인지시키기 위해서 사용한다.
- Buzzer는 내부에 코일을 감은 철심이 들어 있어서 릴레이 모듈과 같이 사용자가 전기 신호를 주면 전류가 흐르면서 자기화 되어 접점이 떨어지게 된다.  
하지만 접점이 떨어지면 전류가 흐르지 않게 되어 다시 접점이 붙는다.  
이런 현상이 빠르게 반복되면서 소리를 발생시킨다.




<Buzzer 내부구조 >



<회로도>



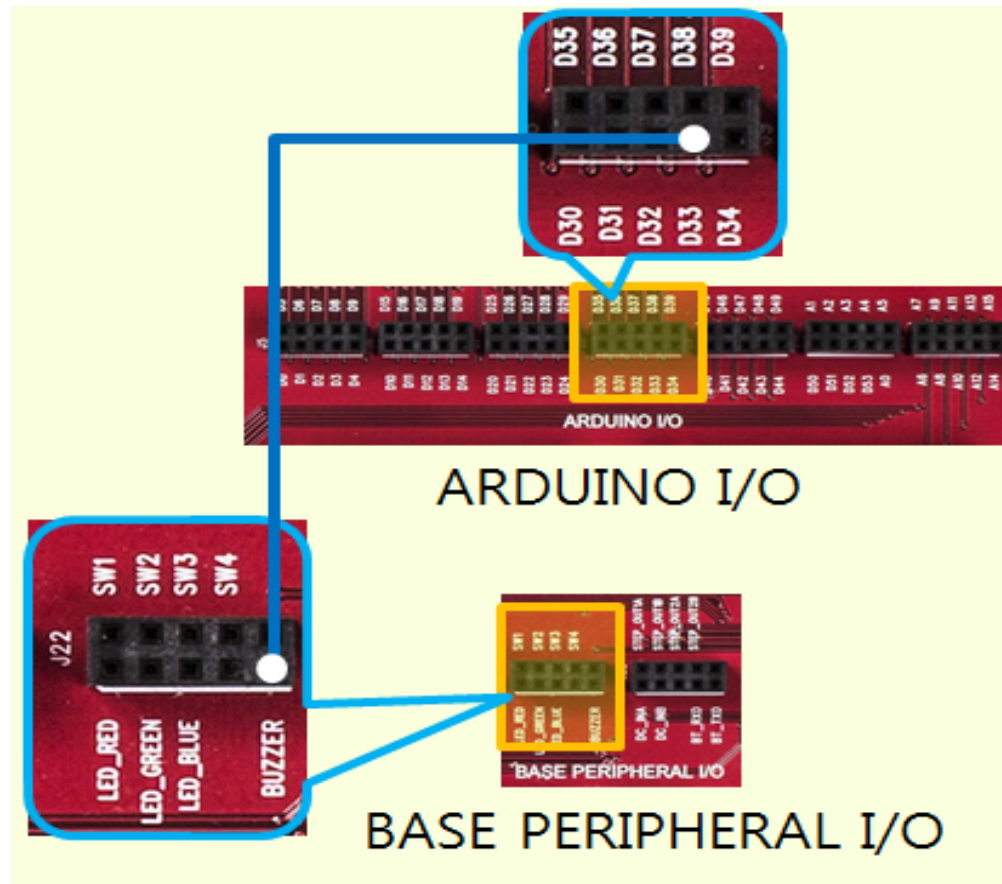
## 2. Buzzer 모듈의 사양

센서 모듈 외형	모듈 항목	모듈 항목의 내용
	동작 전압	5V
	소리 레벨	88dB



## 1. Buzzer 센서 연결

HBE-ADK-2560에 전원을 인가하지 않은 상태에서 그림과 같이 Buzzer 모듈 을 연결한다.







## 2. 부저 프로그램 작성

### PIONEER\_LIGHT\_BUZZER.ino

```
// BUZZER pin number 33
int pin_BUZZER = 33;
void setup() {
  // buzzer pin Output Setup
  pinMode(pin_BUZZER, OUTPUT);
}
void loop() {
  // Buzzer ON
  digitalWrite(pin_BUZZER, HIGH);
  // wait 1 second
  delay(1000);
  // Buzzer OFF
  digitalWrite(pin_BUZZER, LOW);
  // wait 1 second
  delay(1000);
}
```



## 1. PIONEER\_LIGHT\_BUZZER.ino

1) 핀 설정 : 사용할 핀의 번호를 변수에 저장한다.

```
// BUZZER pin number 33  
int pin_BUZZER = 33;
```

2) 초기화 구문 : 사용할 핀을 출력으로 설정한다.

```
void setup {  
  // buzzer pin Output Setup  
  pinMode(pin_BUZZER, OUTPUT);  
}
```



**3) 실행구문 : 앞서 출력으로 설정한 핀을 소리가 발생하도록 HIGH 로 설정한다.**

```
void loop() {  
  // Buzzer ON  
  digitalWrite(pin_BUZZER, HIGH);  
  // wait 1 second  
  delay(1000);
```

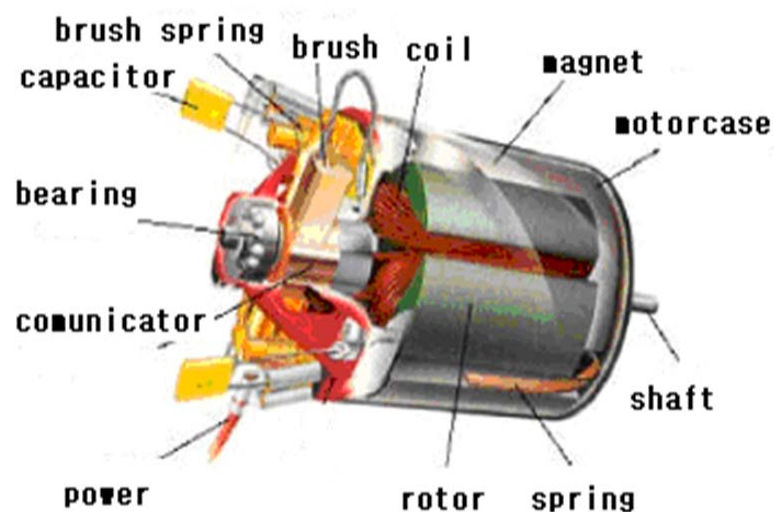
**발생중인 소리를 끄기 위하여 핀의 설정을 LOW로 변경한다.**

```
  // Buzzer OFF  
  digitalWrite(pin_BUZZER, LOW);  
  // wait 1 second  
  delay(1000);  
}
```



## 1. DC Motor의 특징

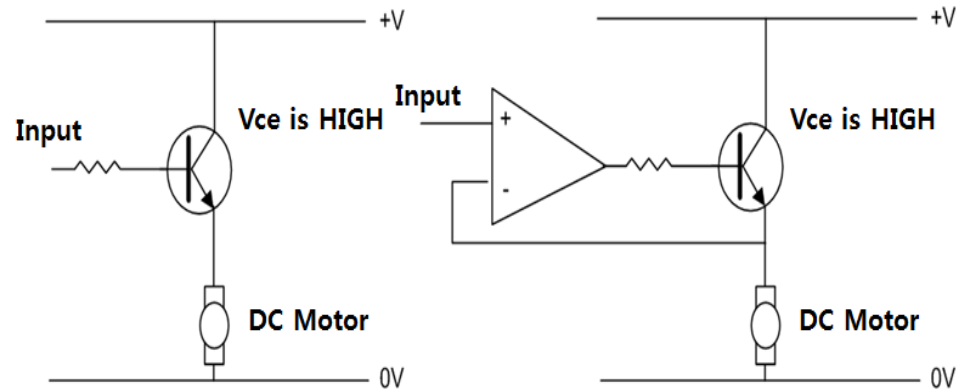
- 기동 토크가 크다.
- 인가전압에 대하여 회전특성이 직선적으로 비례한다.
- 입력전류에 대하여 출력 토크가 직선적으로 비례하며, 또한 출력 효율이 양호하다
- 가격이 저렴하다.
- 제어회로가 매우 단순하고, 제어가 쉽다.
- 구조상 브러시(brush)와 정류자(commutator)에 의한 기계식 접점을 가지고 있어, 전기불꽃(spark), 회전 소음, 수명 단축 등의 단점이 있다.





## 1) 트랜지스터 구동(에미터 부하)

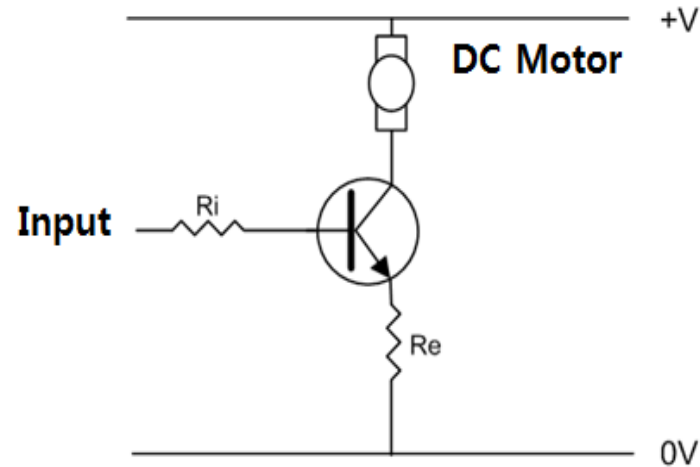
- 그림의 회로에 의해 트랜지스터를 On/Off함으로써 모터를 On/Off한다.  
그러나, 이 회로는 트랜지스터가 완전히 포화되는 On 상태로 할 수 없고,  
 $V_{ce}$ 가 커서 전압손실이 크다.  
자동적으로 부귀환이 동작하기 때문에 동작은 안정적이다.  
그래서, 간단한 속도제어를 위해 OP 앰프를 추가한 회로가 사용된다.





## 2) 트랜지스터 구동(컬렉터 부하)

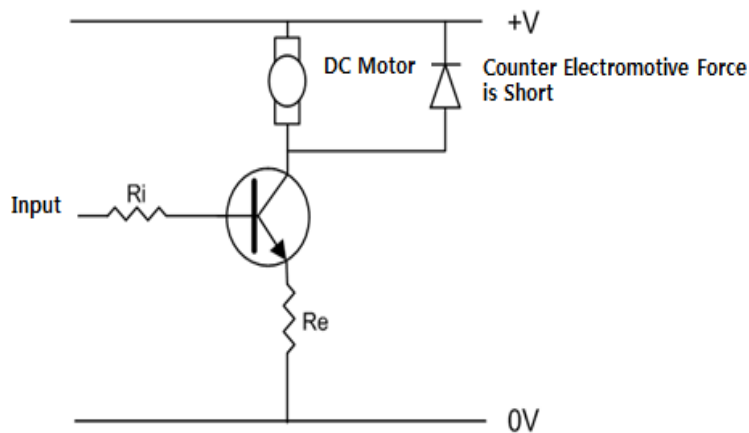
- 모터를 트랜지스터 컬렉터의 부하로 이용한 것으로, 트랜지스터가 완전히 포화된 On 상태로 구동할 수 있기 때문에 드라이브 능력이 크고 전압손실도 적게 할 수 있다.





## 3) 역기전력의 처리

- 트랜지스터가 On으로 되어 모터가 회전하고 있는 동안에는 모터의 코일에 에너지가 축적되어 있다. 트랜지스터가 Off로 되면 그 에너지를 방출하려고 하기 때문에, 모터 코일의 양단에는 플러스, 마이너스가 역방향의 기전력이 발생한다.
- 이에 대한 대책으로 코일을 쇼트 시켜 남아 있는 에너지를 순간적으로 전류로 흘려 버리는 방법으로 역기전력을 억제하도록 한다.

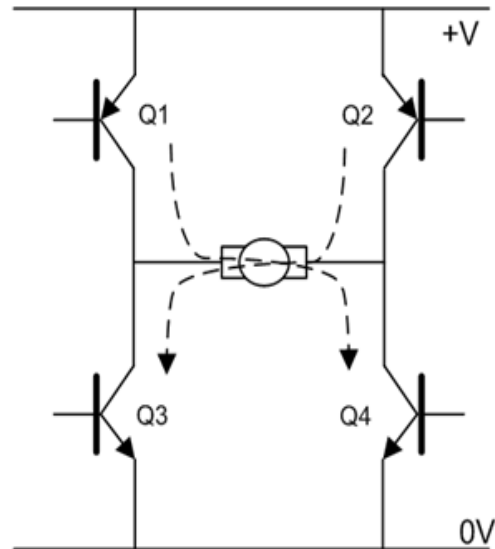




## ● 4) 회전 방향을 바꾸는 H 브리지 제어회로

본동작은 Q1과 Q4의 트랜지스터만 동시에 On으로 하면 Q1에서 Q4쪽으로 전류가 흐르고, 모터는 정회전한다.

반대로 Q2와 Q3만 On으로 하면 Q2에서 Q3쪽으로 전류가 흐르고, 모터는 역회전하게 된다. 그리고, Q3과 Q4만 동시에 On으로 하면 모터에 브레이크를 거는 동작으로 된다.





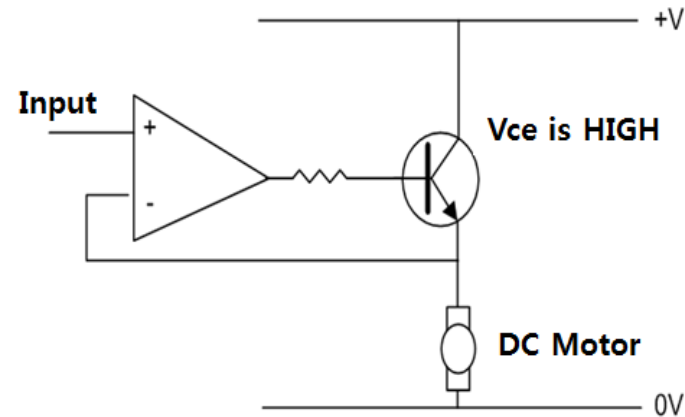


## 5) DC 모터의 가변속 제어법

- DC 모터의 속도를 연속적으로 바꾸려는 경우에는 기본적으로는 DC 모터에 가하는 전압을 바꾸면 속도는 변화한다.

### ① 아날로그 방식의 가변속 제어

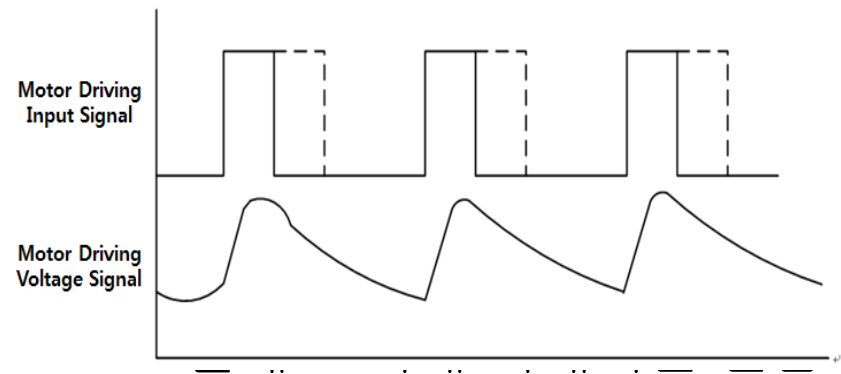
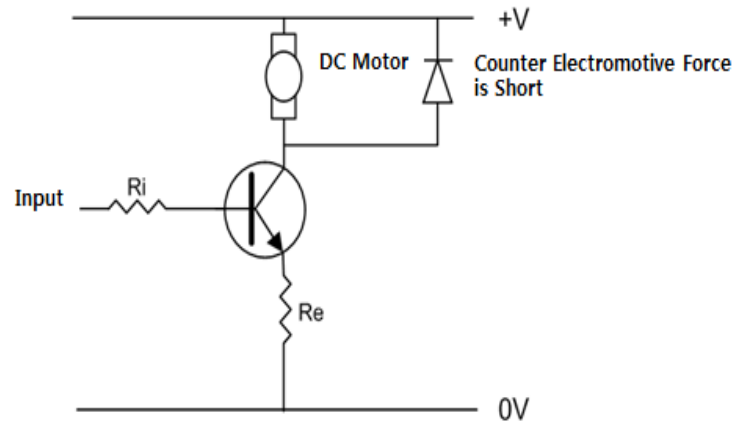
트랜지스터로 전압 dropper를 구성하고, 컬렉터 에미터간의 드롭 전압을 바꿈으로써 모터에 가해지는 구동전압을 가변으로 한다.





## ② 펄스폭 변조(PWM : Pulse Width Modulation)

PWM 방식은 구동전압을 바꾸고 있는 것과 같은 효과를 내고 있지만, 그 방법이 펄스폭에 따르고 있으므로 펄스폭 변조(PWM)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.





## 2. DC Motor 모듈의 사양

- DC Motor 모듈 의 하드웨어 사양은 <표 3-9>와 같다.

DC 모터 외형	모듈 항목	모듈 항목의 내용
	모터 드라이버	BA6208F
	모터	Micro Type DC Motor
	동작 전압	5V
액추에이터로 활용할 수 있는 DC Motor 모듈		

<표 3-9> DC Motor 모듈의 사양



## ● HBE-ADK-2560과 DC Motor의 연결 방법

두 모듈을 연결하기 위해서는 2핀 케이블로 다음과 같이

HBE-ADK-2560과 DC Motor 모듈을 연결해야 한다. J23 커넥터의

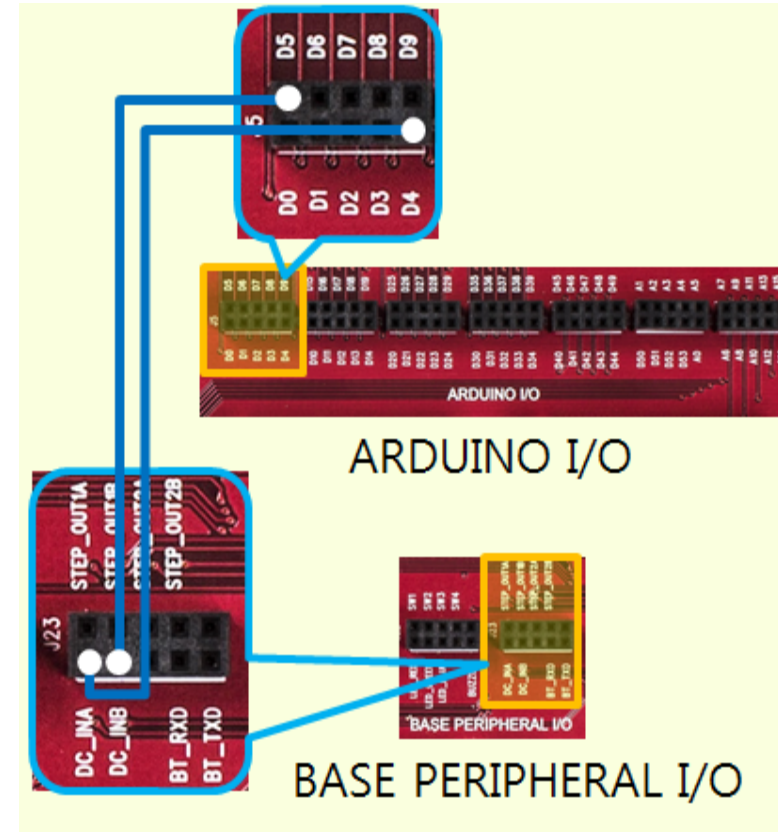
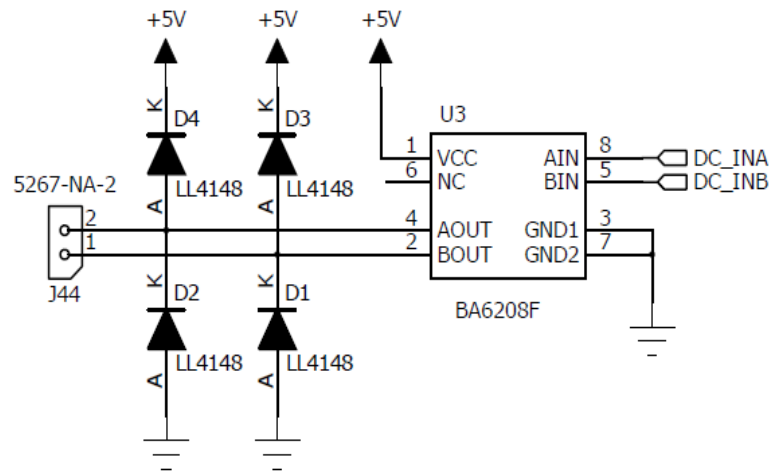
D4과 D5번 핀을 J14 커넥터의 DC\_INA와 DC\_INB 에 연결한다.

<표 3-10>는 HBE-ADK-2560과 DC Motor 모듈의 핀 연결을 정리한 것이다.

ADK-2560 모듈 핀 번호	핀 정보	DC Motor 모듈 핀 번호
D4	Digital pin(PWM)	DC_INA
D5	Digital pin(PWM)	DC_INB

<표 3-10> HBE-ADK-2560 모듈과 DC Motor 모듈의 핀 연결 정보

## 1. DC Motor 모듈의 모터 드라이브연결 및 회로도





## 1. 모터 제어 프로그램 작성

PIONEER\_LIGHT\_DCMOTOR.ino

```
int pin_DC_A = 4;
int pin_DC_B = 5;
void setup() {
  pinMode(pin_DC_A, OUTPUT);
  pinMode(pin_DC_B, OUTPUT);
}
void loop() {
  int Speed;
  for(Speed=0; Speed <256; Speed++)
  {
    Motor_CW(Speed);
    delay(20);
  }
  delay(1000);
  Stop();
  delay(200);
```

```
for(Speed=0; Speed <256; Speed++)
{
  Motor_CCW(Speed);
}
void Motor_CCW(unsigned char Speed)
{
  digitalWrite(pin_DC_B, LOW);
  analogWrite(pin_DC_A, Speed);
}
void Stop(void)
{
  digitalWrite(pin_DC_A, LOW);
  digitalWrite(pin_DC_B, LOW);
}
int pin_DC_A = 4;
int pin_DC_B = 5;
void setup()
```



# DC Motor 제어 프로그램 작성



HANBACK ELECTRONICS.,LTD

```
{
pinMode(pin_DC_A, OUTPUT);
pinMode(pin_DC_B, OUTPUT);
}

void loop() {
int Speed;
for(Speed=0; Speed <256; Speed++)
{
Motor_CW(Speed);
delay(20);
}
delay(1000);
Stop();
delay(200);
for(Speed=0; Speed <256; Speed++)
{
Motor_CCW(Speed);
delay(20);}
```

```
delay(1000);
Stop();
delay(200);
}

void Motor_CW(unsigned char Speed)
{
digitalWrite(pin_DC_A, LOW);
analogWrite(pin_DC_B, Speed);}

void Motor_CCW(unsigned char Speed)
{
digitalWrite(pin_DC_B, LOW);
analogWrite(pin_DC_A, Speed);
}

void Stop(void)
{
digitalWrite(pin_DC_A, LOW);
digitalWrite(pin_DC_B, LOW);
}
```



## 1.PIONEER\_LIGHT\_DCMOTOR.ino

1) 사용 핀 선언 : 변수를 선언하여 모터 구동에 필요한 핀 번호를 저장한다.

```
// DC A pin number 4  
  
int pin_DC_A = 4;  
  
// DC B pin number 5  
  
int pin_DC_B = 5;
```

2) 초기화 구문 : 모터 제어에 사용할 핀을 출력으로 설정한다.

```
void setup( ){  
  
// DC A pin Output Setup  
pinMode(pin_DC_A, OUTPUT);  
  
// DC B pin Output Setup  
pinMode(pin_DC_B, OUTPUT);  
  
}
```





## 3) DC 모터 제어

DC 모터를 시계방향으로 회전시키고, 최고속도가 될 때까지 속도를 점점 빠르게 변화 시킨다.

```
void loop() {  
    uint16_t Speed;  
    for(Speed=0; Speed <256; Speed++)  
    {  
        Motor_CW(Speed);  
        delay(20);  
    }  
    delay(1000);  
    Stop();  
    delay(200);  
}
```



- DC 모터를 반시계 방향으로 회전시키고, 최고속도가 될 때까지 속도를 점점 빠르게 변화시키고 나서 모터를 정지시킨 후 0.2초의 시간지연을 갖도록 한다.

```
for(Speed=0; Speed <256; Speed++)  
{  
    Motor_CCW(Speed);  
    delay(20);  
}  
delay(1000);  
Stop();  
delay(200);  
}
```



- DC 모터를 시계방향으로 회전시키는 함수이다. Speed 값에 따라 속도가 제어 된다.

```
// DC Motor control  
// Motor CW  
void Motor_CW(uint8_t Speed)  
{  
    digitalWrite(pin_DC_A, LOW);  
    analogWrite(pin_DC_B, Speed);  
}
```



- DC 모터를 반시계방향으로 회전시키는 함수이다. Speed 값에 따라 속도가 제어 된다.

```
// Motor CCW  
  
void Motor_CCW(uint8_t Speed)  
{  
    digitalWrite(pin_DC_B, LOW);  
    analogWrite(pin_DC_A, Speed);  
}
```

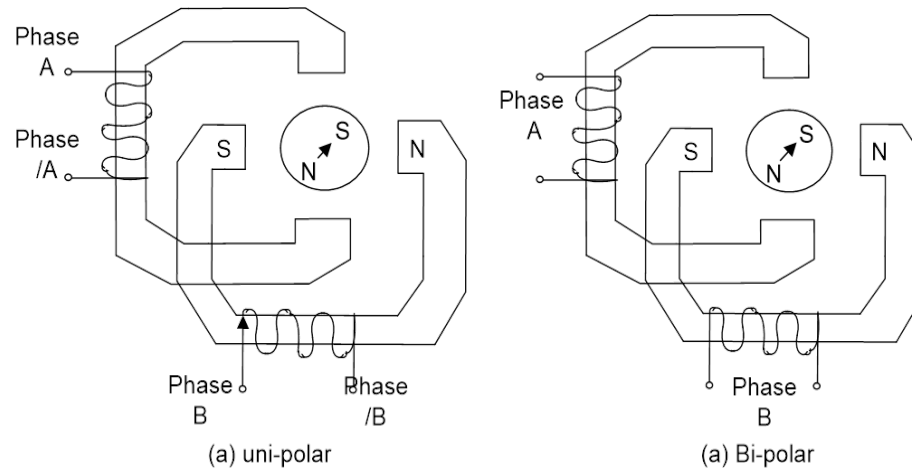
- DC 모터를 정지시키는 함수이다.

```
// Stop  
  
void Stop(void)  
{  
    digitalWrite(pin_DC_A, LOW);  
    digitalWrite(pin_DC_B, LOW);  
}
```



## 1. Step 모터의 동작

- Step Motor는 산업용뿐 아니라 아날로그시계에 이르기까지 광범위하게 사용되고 있다. 스텝 모터는 일반적인 위치 제어에 많이 사용하고 있는 모터로서 위치의 정지 정밀도를 요구하는 곳에 알맞은 모터라고 할 수 있다.

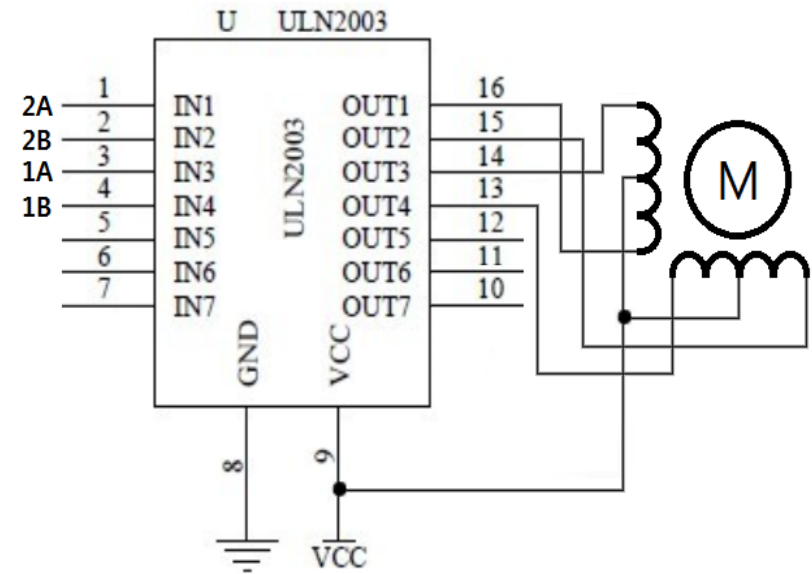


<Step Motor 내부구조>



# Step 모터의 동작

	1상 여자 방식	2상 여자 방식	1,2상 여자 방식
특징	항상 1개의 위상에만 전류가 흐른다.	항상 2개의 위상에만 전류가 흐른다.	1개, 2개의 위상에 전류가 교차되어 흐른다.
통전방식	A -> B -> /A -> /B	A,B -> B,/A -> /A,/B -> /B,A	A -> A,B -> B -> B,/A -> /A -> /A,/B -> /B -> B,/A
토크	작다	크다	변동이 있다.
스텝 각	Full Step : 90도	Full Step : 90도	Full Step : 45도




<구동방식에 따른 특성 비교>

< 회로도 >



## 2. Step Motor 모듈의 사양

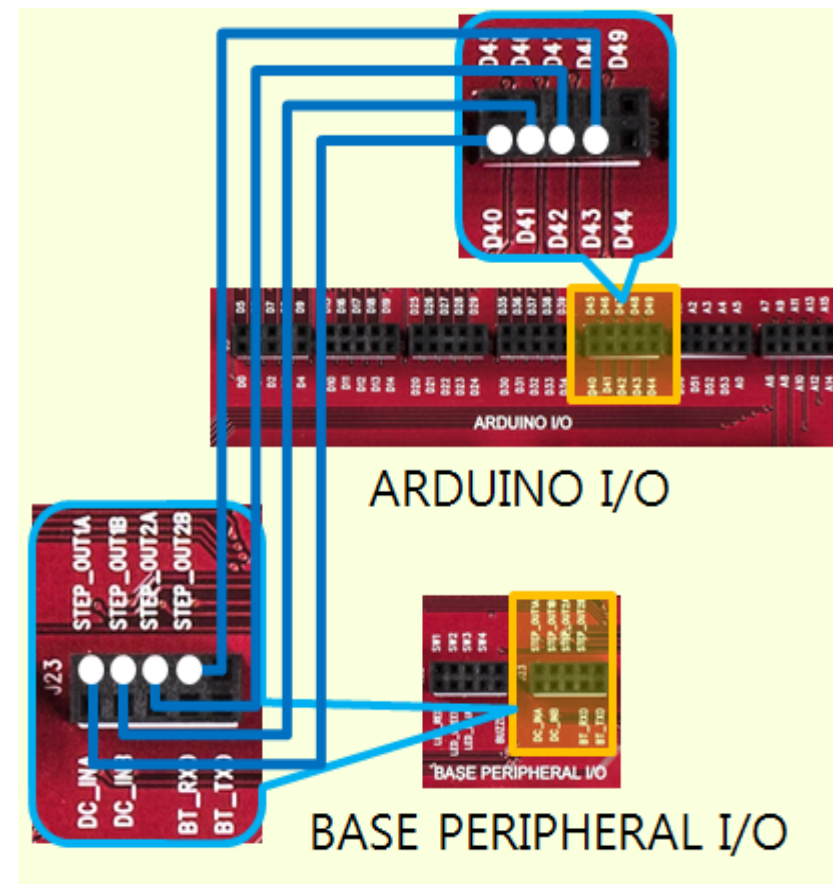
- Step Motor 모듈 의 하드웨어 사양은 다음과 같다.

Step 모터 외형	모듈 항목	모듈 항목의 내용
	모터 드라이버	ULN2003
	모터	Step Motor(32 Step , 1/64Gear)
	동작 전압	5V
액츄에이터로 활용할 수 있는 Step Motor 모듈		

## 1. Step Motor 모듈의 연결

두 모듈을 연결하기 위해서는 4핀 케이블로 그림과 같이 HBE-ADK-2560과 Step Motor 모듈을 연결해야 한다. J27 커넥터의 D40, D41, D42, D43 핀을 J14 커넥터의 DC\_INA, DC\_INB, DC\_ENCA, DC\_ENCB에 각각 연결한다.

ADK-2560 모듈 핀 번호	핀 정보	DC Motor 모듈 핀 번호
D40	Digital pin	STEP_OUT1A
D41	Digital pin	STEP_OUT1B
D42	Digital pin	STEP_OUT2A
D43	Digital pin	STEP_OUT2B







## • 1. Step 모터 제어 프로그램 작성 : 1상 여자방식

PIONEER\_LIGHT\_STEP.ino

```
int pin_STEP[4] = {40, 41, 42, 43};  
void setup()  
{  
  for(int i=0; i<4; i++)  
    pinMode(pin_STEP[i], OUTPUT);  
}  
void loop()  
{  
  for(int i=0; i<512 ; i++)  
  {  
    digitalWrite(pin_STEP[3], 1);  
    digitalWrite(pin_STEP[2], 0);  
    digitalWrite(pin_STEP[1], 0);  
    digitalWrite(pin_STEP[0], 0);  
    delay(3);
```

```
    digitalWrite(pin_STEP[3], 0);  
    digitalWrite(pin_STEP[2], 1);  
    digitalWrite(pin_STEP[1], 0);  
    digitalWrite(pin_STEP[0], 0);  
    delay(3);  
    digitalWrite(pin_STEP[3], 0);  
    digitalWrite(pin_STEP[2], 0);  
    digitalWrite(pin_STEP[1], 1);  
    digitalWrite(pin_STEP[0], 0);  
    delay(3);  
    digitalWrite(pin_STEP[3], 0);  
    digitalWrite(pin_STEP[2], 0);  
    digitalWrite(pin_STEP[1], 0);  
    digitalWrite(pin_STEP[0], 1);  
    delay(3);  
  }  
}
```



## Step 모터 제어 프로그램 작성



HANBACK ELECTRONICS.,LTD

```
digitalWrite(pin_STEP[3], 0);  
digitalWrite(pin_STEP[2], 0);  
digitalWrite(pin_STEP[1], 0);  
digitalWrite(pin_STEP[0], 0);  
delay(2000);  
for(int i=0; i<512; i++)  
{  
    digitalWrite(pin_STEP[3], 0);  
    digitalWrite(pin_STEP[2], 0);  
    digitalWrite(pin_STEP[1], 0);  
    digitalWrite(pin_STEP[0], 1);  
    delay(3);  
    digitalWrite(pin_STEP[3], 0);  
    digitalWrite(pin_STEP[2], 0);  
    digitalWrite(pin_STEP[1], 1);  
    digitalWrite(pin_STEP[0], 0);  
    delay(3);  
}
```

```
digitalWrite(pin_STEP[3], 0);  
digitalWrite(pin_STEP[2], 1);  
digitalWrite(pin_STEP[1], 0);  
digitalWrite(pin_STEP[0], 0);  
delay(3);  
digitalWrite(pin_STEP[0], 1);  
digitalWrite(pin_STEP[1], 0);  
digitalWrite(pin_STEP[2], 0);  
digitalWrite(pin_STEP[3], 0);  
delay(3);  
}  
digitalWrite(pin_STEP[3], 0);  
digitalWrite(pin_STEP[2], 0);  
digitalWrite(pin_STEP[1], 0);  
digitalWrite(pin_STEP[0], 0);  
delay(2000);  
}
```



## 1. PIONERR\_LIGHT\_STEP.ino

### 1) 사용 핀 선언

- 변수를 선언하여 STEP MOTOR의 핀 번호를 저장한다.

```
// 1A, 1B, 2A, 2B  
  
int pin_STEP[4] = {40, 41, 42, 43};
```

### 2) 초기화 구문

- STEP Motor 핀을 출력으로 설정한다.

```
for(int i=0; i<4; i++)  
{  
    pinMode(pin_STEP[i], OUTPUT);  
}
```



## 3) STEP 모터 제어- 1상여자 방식

- STEP Motor를 시계방향으로 회전한다. A 상을 ON한다.

```
digitalWrite(pin_STEP[3], 1);  
digitalWrite(pin_STEP[2], 0);  
digitalWrite(pin_STEP[1], 0);  
digitalWrite(pin_STEP[0], 0);  
delay(3);
```

B 상을 ON한다.

```
digitalWrite(pin_STEP[3], 0);  
digitalWrite(pin_STEP[2], 1);  
digitalWrite(pin_STEP[1], 0);  
digitalWrite(pin_STEP[0], 0);  
delay(3);
```



/A 상을 ON한다.

```
digitalWrite(pin_STEP[3], 0);  
digitalWrite(pin_STEP[2], 0);  
digitalWrite(pin_STEP[1], 1);  
digitalWrite(pin_STEP[0], 0);  
delay(3);
```

/B 상을 ON한다.

```
digitalWrite(pin_STEP[3], 0);  
digitalWrite(pin_STEP[2], 0);  
digitalWrite(pin_STEP[1], 0);  
digitalWrite(pin_STEP[0], 1);  
delay(3);
```



- STEP Motor를 반시계방향으로 회전한다.  
/B 상을 ON한다.

```
digitalWrite(pin_STEP[3], 0);  
digitalWrite(pin_STEP[2], 0);  
digitalWrite(pin_STEP[1], 0);  
digitalWrite(pin_STEP[0], 1);  
delay(3);
```

/A 상을 ON한다.

```
digitalWrite(pin_STEP[3], 0);  
digitalWrite(pin_STEP[2], 0);  
digitalWrite(pin_STEP[1], 1);  
digitalWrite(pin_STEP[0], 0);  
delay(3);
```



B 상을 ON한다.

```
digitalWrite(pin_STEP[3], 0);  
digitalWrite(pin_STEP[2], 1);  
digitalWrite(pin_STEP[1], 0);  
digitalWrite(pin_STEP[0], 0);  
delay(3);
```

A 상을 ON한다.

```
digitalWrite(pin_STEP[0], 1);  
digitalWrite(pin_STEP[1], 0);  
digitalWrite(pin_STEP[2], 0);  
digitalWrite(pin_STEP[3], 0);  
delay(3);
```