

아두이노를 이용한 스마트 센서제어 실습2

한백전자 기술연구소



HANBACK ELECTRONICS CO.,LTD



학습3

아두이노를 이용한 스마트 센서제어 실습2

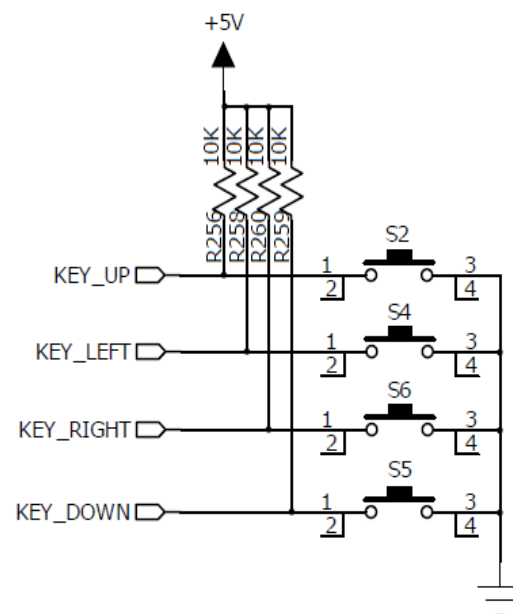


- Input 실습하기
- Light Sensor 실습하기
- Ultra Sonic Sensor 실습하기
- 온/습도 Sensor 실습하기
- 가변저항 실습하기
- Bluetooth 실습하기



1. Input(JoyStck) 모듈

- Input 모듈은 푸쉬 버튼이다. 버튼은 평소에는 연결된 PIN의 값이 “HIGH”상태를 유지하며 버튼이 눌렸을 때 “LOW”상태로 변경된다. 다음 그림은 Input 모듈 회로도이다.

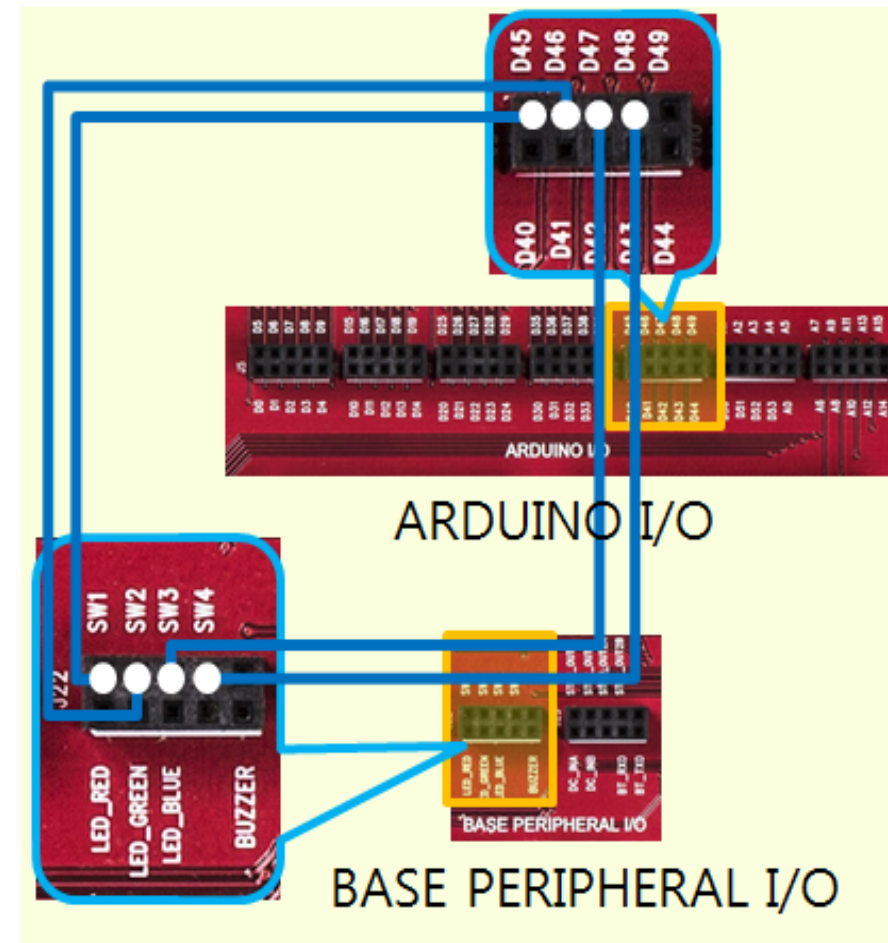




1. Input 모듈의 핀 연결

- 두 모듈을 연결하기 위해서는 7핀 케이블로 그림과 같이 HBE-ADK-2560과 Input 모듈을 연결해야 한다.

ADK-2560 모듈 핀 번호	핀 정보	Input 모듈 핀 번호
D45	Digital pin	KEY_UP
D46	Digital pin	KEY_DOWN
D47	Digital pin	KEY_LEFT
D48	Digital pin	KEY_RIGHT





1. Input 모듈 프로그램 작성

PIONEER_LIGHT_SWITCH.ino

```
int pin_SW1 = 45; // SW1 pin number
45 setup
int pin_SW2 = 46; // SW2 pin number
46 setup
int pin_SW3 = 47; // SW3 pin number
47 setup
int pin_SW4 = 48; // SW4 pin number
48 setup
void setup() {
// UART setup baud 115200, data bit
8, parity None, stop bit 1
Serial.begin(115200); // same
Serial.begin(115200, SERIAL_8N1)
// SW1 pin Input Setup
pinMode(pin_SW1, INPUT);
// SW2 pin Input Setup
pinMode(pin_SW2, INPUT);
// SW3 pin Input Setup
pinMode(pin_SW3, INPUT);
// SW4 pin Input Setup
pinMode(pin_SW4, INPUT);
}
```

```
void loop() {
Serial.print("SW1 :");
Serial.print(digitalRead(pin_SW
1));
Serial.print(", SW2 :");
Serial.print(digitalRead(pin_SW
2));
Serial.print(", SW3 :");
Serial.print(digitalRead(pin_SW
3));
Serial.print(", SW4 :");
Serial.println(digitalRead(pin_S
W4));
delay(500);
}
```



1) PIONEER_LIGHT_SWITCH.ino

- 사용 핀 선언 : 변수를 선언하여 Input 모듈의 핀 번호를 저장한다.

```
int pin_SW1 = 45; // SW1 pin number 45 setup  
int pin_SW2 = 46; // SW2 pin number 46 setup  
int pin_SW3 = 47; // SW3 pin number 47 setup  
int pin_SW4 = 48; // SW4 pin number 48 setup
```

2) 초기화 구문

- Serial port(UART 0)를 전송속도 115200, 데이터 비트 8, 패리티 없음, 스톱 비트 1로 설정하고 시작한다. 통신 및 프로그램 동작을 확인하기 위해 연결한다.

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



- Up, Down, Left, Right 버튼을 입력으로 설정한다.

```
// SW1 pin Input Setup  
pinMode(pin_SW1, INPUT);  
// SW2 pin Input Setup  
pinMode(pin_SW2, INPUT);  
// SW3 pin Input Setup  
pinMode(pin_SW3, INPUT);  
// SW4 pin Input Setup  
pinMode(pin_SW4, INPUT);
```





3) Input 모듈의 상태 값 출력

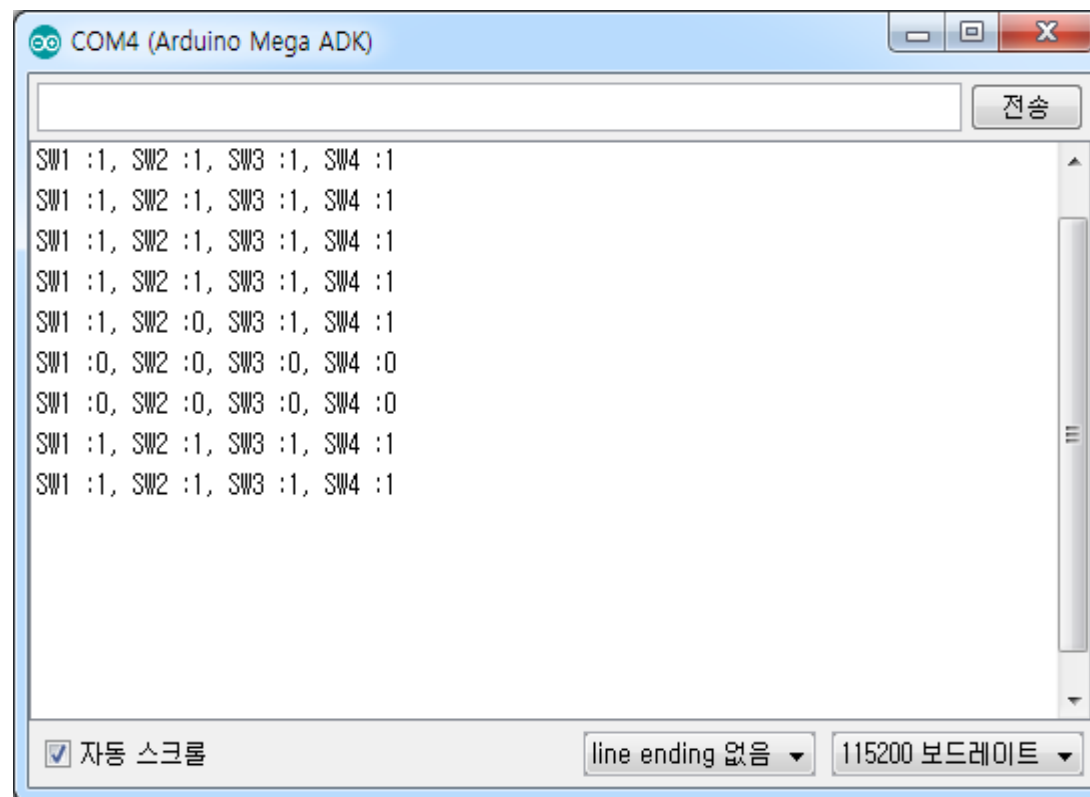
- SW1, SW2, SW3, SW4 버튼의 핀을 읽어 그 값을 출력한다.

```
Serial.print("SW1 :");  
Serial.print(digitalRead(pin_SW1));  
Serial.print(", SW2 :");  
Serial.print(digitalRead(pin_SW2));  
Serial.print(", SW3 :");  
Serial.print(digitalRead(pin_SW3));  
Serial.print(", SW4 :");  
Serial.println(digitalRead(pin_SW4));
```



1. 시리얼 모니터를 및 동작 확인

- 업로드가 완료된 후 시리얼 모니터 활성화() 를 한 후 통신 속도를 115200으로 설정한다. 그림과 같이 Input 모듈의 SW1, SW2, SW3, SW4 버튼에 대한 값을 Serial로 출력한다. 보통 상태는 1이 출력되고 버튼을 누르면 0이 출력된다.



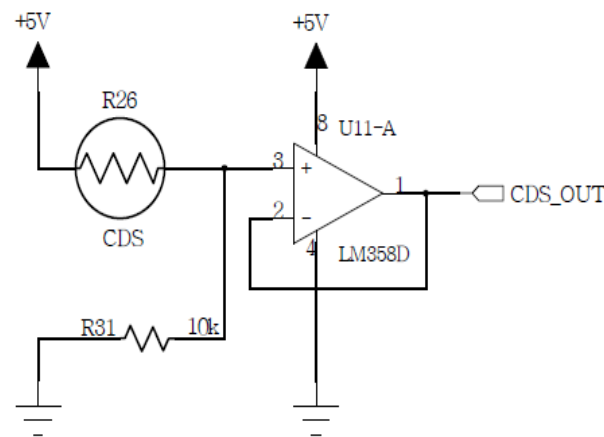


1. Light Sensor

- CdS 센서는 반도체에 빛이 닿으면 전자와 정공이 증가하고 조사된 빛에너지에 비례하여 전도도가 변화하고 이에 따라 전기저항이 변화하는 현상을 이용한 소자를 말한다. CdS 센서는 두 전극 사이에 광 도전체인 CdS (황화카드뮴)가 삽입되어 있어 빛의 세기에 따라 내부 저항이 변하는데 밝은 곳에서는 내부 저항이 작아지는 광 가변 저항기다.



< Light 센서 >



< 회로도 >



2. Light 센서의 사양

- Light 센서 의 하드웨어 사양은 <표 3-17>과 같다.

Light 센서 외형	모듈 항목	모듈 항목의 내용
	센서	CDS
	인터페이스	1pin Analog OUTPUT
	동작 전압	5V

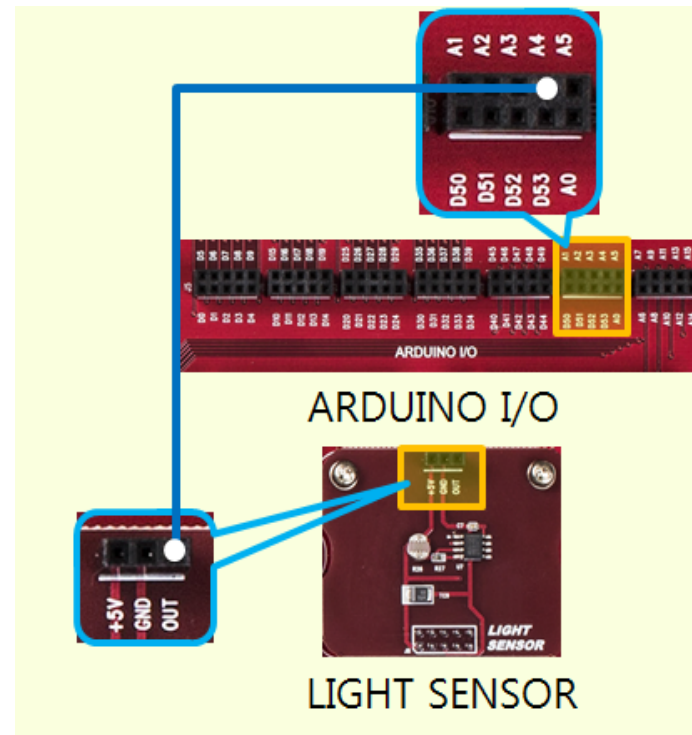
< 표 3-17 >Ligth 센서 모듈의 사양



1. Light 센서의 핀 연결

- 두 모듈을 연결하기 위해서는 1핀 케이블로 그림과 같이 HBE-ADK-2560과 Light 센서 모듈을 연결해야 한다.

ADK-2560 모듈 핀 번호	핀 정보	Light Sensor 핀 번호
A4	Analog pin	CDS_A





1. Light 센서 프로그램 작성

PIONEER_LIGHT_CDS.ino

```
int pin_CDS = A4; // CDS pin number A4 setup

void setup() {
  Serial.begin(115200); // same Serial.begin(115200, SERIAL_8N1)
  pinMode(pin_CDS, INPUT);
}

void loop() {
  Serial.print("ADC Data :");
  Serial.println(analogRead(pin_CDS));
  delay(500);
}
```



1. PIONEER_LIGHT.ino

1) 사용 핀 선언 : 변수를 선언하여 Light Sensor 모듈의 핀 번호를 저장한다.

```
int pin_CDS = A4; // CDS pin number A4 setup
```

2) 초기화 구문

- Serial port(UART 0)를 전송속도 115200, 데이터 비트 8, 패리티 없음, 스톱 비트 1로 설정하고 시작한다. 통신 및 프로그램 동작을 확인하기 위해 연결한다.

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

CDS 핀을 입력으로 설정한다.

```
// CDS analog pin Input Setup  
pinMode(pin_CDS, INPUT);  
}
```




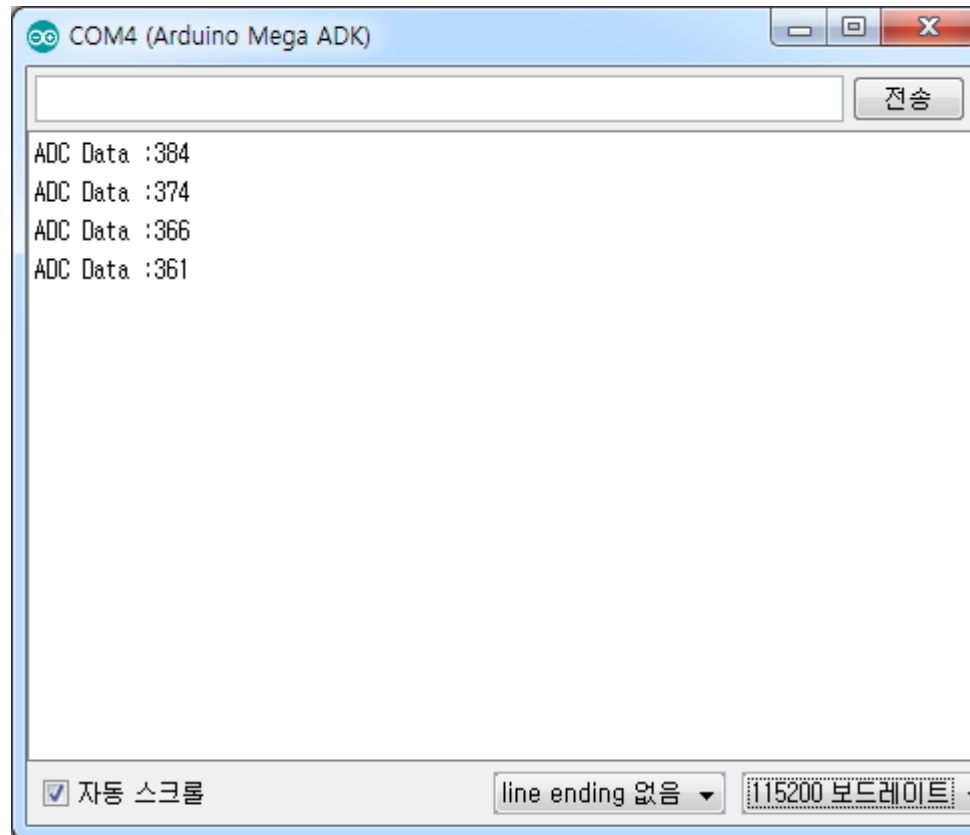
3) Light 모듈의 ADC 값 출력 : CDS 핀의 ADC 값을 시리얼로 출력한다.

```
void loop() {  
  Serial.print("ADC Data :");  
  Serial.println(analogRead(pin_CDS));  
  delay(500);  
}
```




1. 시리얼 모니터를 및 동작 확인

- 업로드가 완료된 후 시리얼 모니터 활성화()를 한 후 통신 속도를 115200으로 설정한다. 그림과 같이 Light Sensor 모듈의 CDS핀에 대한 ADC 값을 Serial로 출력한다.





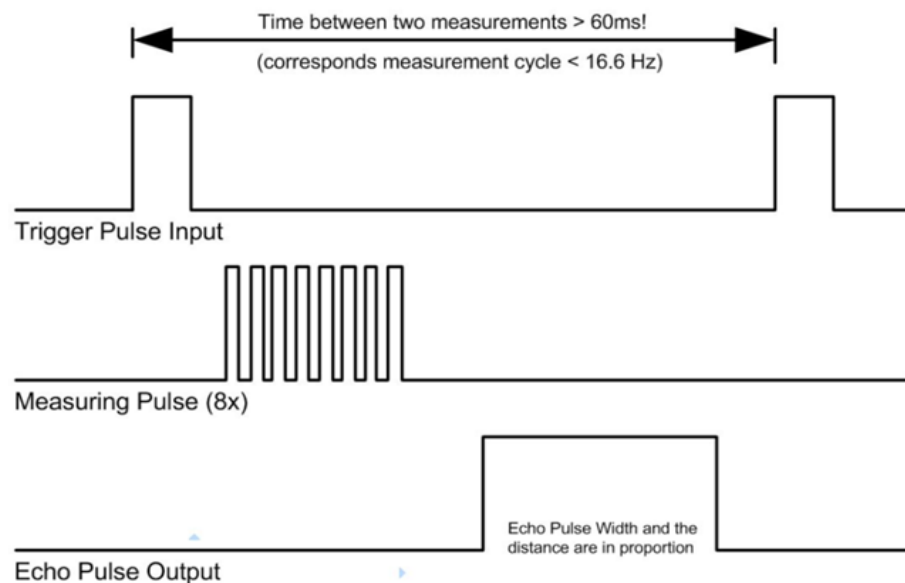
1. UltraSonic Sensor

Ultrasonic Sensor는 초음파를 이용하여 거리를 측정하는 모듈이다.

센서 모듈로 거리를 측정하기 위해서 먼저 초음파를 발생시켜준다.

발생된 초음파는 전방으로 진행한다. 전방으로 진행하는 초음파가 물체에 부딪치면 반사되어 모듈로 돌아온다.

이 과정에서 초음파가 발사되어 돌아온 시간을 측정하여 거리를 계산한다.



<Ultrasonic Sensor 거리 측정 과정>



2. UltraSonic 센서의 사양

- UltraSonic 센서의 하드웨어 사양은 다음과 같다.

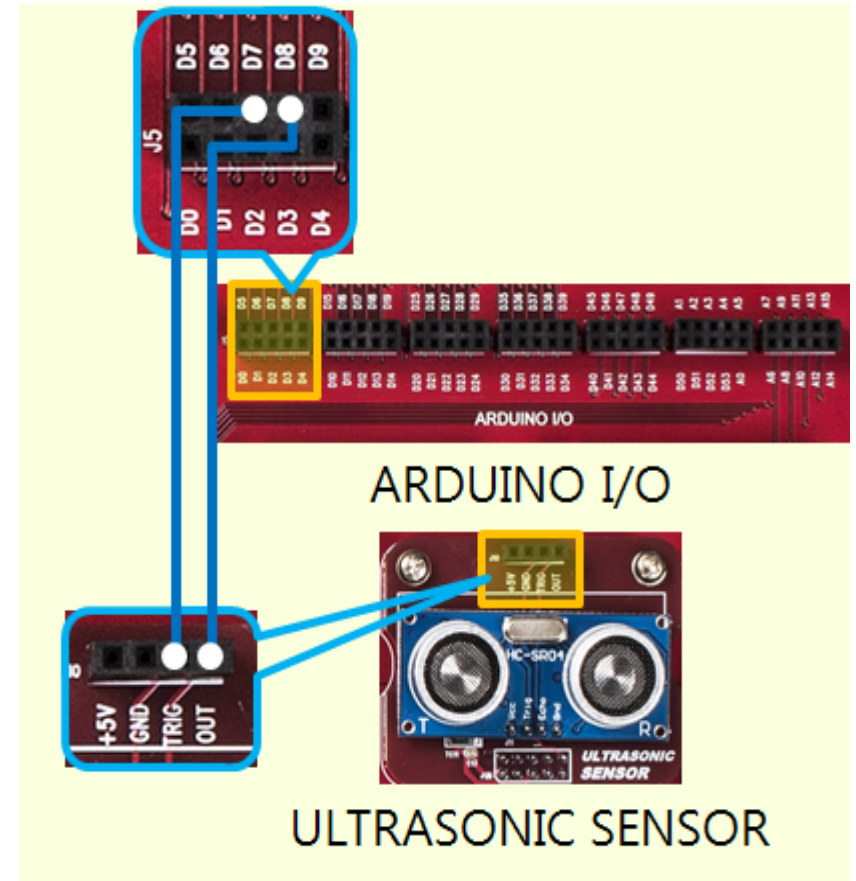
UltraSonic 센서 외형	모듈 항목	모듈 항목의 내용
	측정 거리	2 ~ 500cm
	인터페이스	1pin Digital Input , 1pin Digital O UTPUT
	동작 전압	5V



1. UltraSonic 센서의 핀 연결

- 두 모듈을 연결하기 위해서는 케이블로 그림과 같이 HBE-ADK-2560과 Ultra Sonic 센서 모듈을 연결해야 한다.

ADK-2560 모듈 핀 번호	핀 정보	UltraSonic Sensor 핀 번호
D7	Digital pin	UL_TRIG
D8	Digital pin	UL_OUT





1. UltraSonic 센서 프로그램 작성

PIONEER_ULTRASONIC.ino

```
int pin_UL_TRIG = 7;
int pin_UL_OUT = 8;
void setup() {
  Serial.begin(115200); // same Serial.begin(115200
, SERIAL_8N1)
  pinMode(pin_UL_OUT, INPUT);
  pinMode(pin_UL_TRIG, OUTPUT);
  digitalWrite(pin_UL_TRIG, 0);
}
void loop() {
  unsigned long microseconds, distance_cm;
  digitalWrite(pin_UL_TRIG, 0); delayMicroseconds(2);
```

```
  digitalWrite(pin_UL_TRIG, 1);
  delayMicroseconds(10);
  digitalWrite(pin_UL_TRIG, 0);
  microseconds = pulseIn(pin_UL_OUT, 1 , 24000);
  distance_cm = microseconds * 17/1000;
  Serial.print("Time :");
  Serial.print(microseconds);
  Serial.print("[us], Dist :");
  Serial.print(distance_cm);
  Serial.println("[cm]");
  delay(1000);
}
```



1. PIONEER_ULTRASONIC.ino

- 1) 사용 핀 선언 : 변수를 선언하여 UltraSonic Sensor 모듈의 핀 번호를 저장한다.

```
int pin_UL_TRIG = 7;  
int pin_UL_OUT = 8;
```

- 2) 초기화 구문

Serial port(UART 0)를 전송속도 115200, 데이터 비트 8, 패리티 없음, 스톱 비트 1로 설정하고 시작한다. 통신 및 프로그램 동작을 확인하기 위해 연결한다.

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

Out 핀을 입력, Trig 핀을 출력으로 설정한다.

```
  pinMode(pin_UL_OUT, INPUT);  
  pinMode(pin_UL_TRIG, OUTPUT);  
  digitalWrite(pin_UL_TRIG, 0);  
}
```



- 3) 거리 측정 : Trig 핀에 10us의 HIGH 펄스를 주어 초음파가 발사 되도록 한다.

```
void loop() {  
    unsigned long microseconds, distance_cm;  
    digitalWrite(pin_UL_TRIG, 0);  
    delayMicroseconds(2);  
    digitalWrite(pin_UL_TRIG, 1); // Output pin_ULTRASONIC_T to HIGH  
    delayMicroseconds(10);  
    digitalWrite(pin_UL_TRIG, 0); // Output pin_ULTRASONIC_T to LOW
```

초음파가 돌아오는데 걸리는 시간을 측정하여 microseconds에 저장한다.
0.024초 내에 돌아오는 신호가 없으면 0을 반환한다.

```
microseconds = pulseIn(pin_UL_OUT, 1 , 24000);
```



- 시간 값을 거리 값으로 변환하여 distance_cm에 저장한다.


```
// time to dist  
distance_cm = microseconds * 17/1000;
```

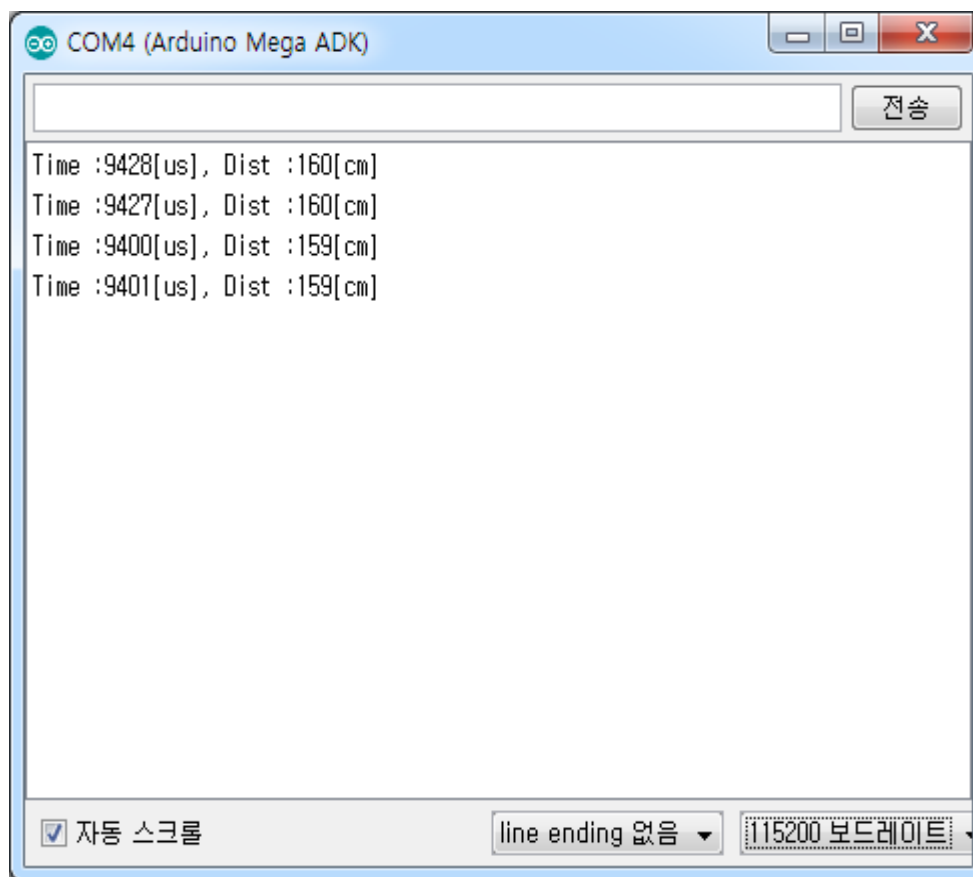
시간 및 거리 값을 시리얼로 전송한다.

```
Serial.print("Time :");  
Serial.print(microseconds);  
Serial.print("[us], Dist :");  
Serial.print(distance_cm);  
Serial.println("[cm]");  
delay(1000);  
}
```




1. 시리얼 모니터를 및 동작 확인

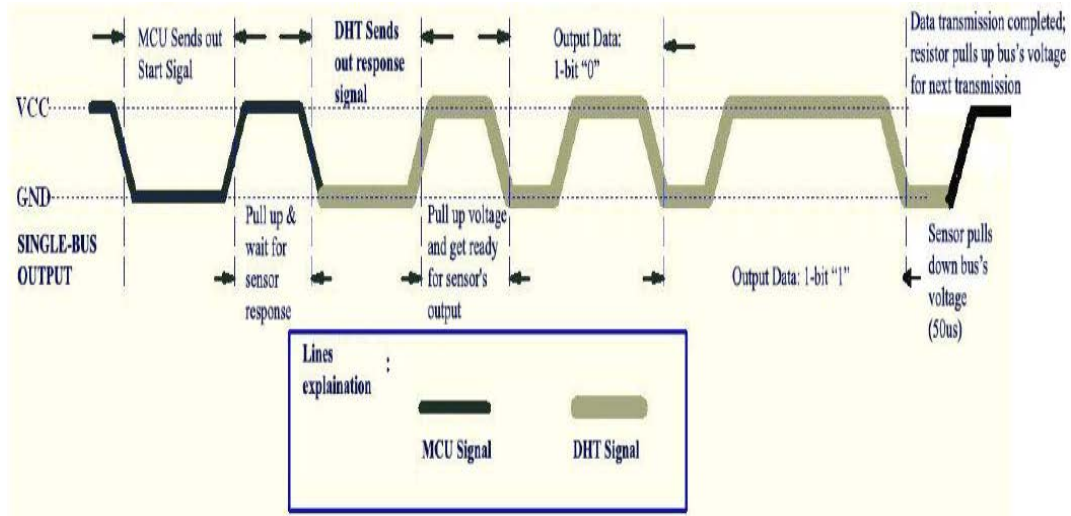
- 업로드가 완료된 후 시리얼 모니터 활성화()를 한 후 통신 속도를 115200으로 설정한다. 그림과 같이 Ultrasonic 센서로 초음파가 다시 돌아오는데 걸리는 시간 및 거리 값을 Serial로 출력한다.



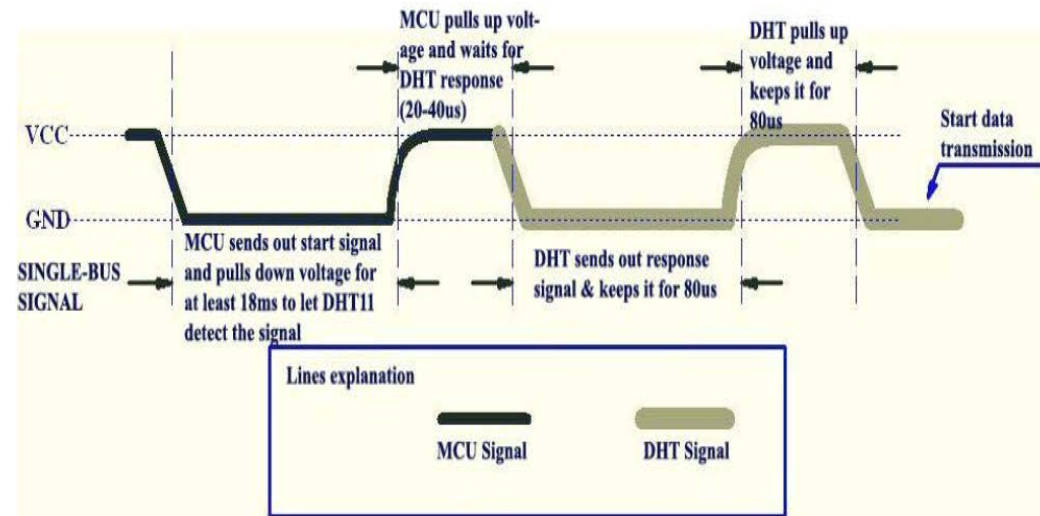


1. DHT11 센서의 통신 방식

- DHT11 온/습도 센서는 0~50°C, 20~90%의 온/습도를 측정할 수 있으며 1-wire bus로 통신한다. 데이터 구성은 습도 8bit 정수 데이터, 습도 8bit 소수 데이터, 온도 8bit 정수 데이터, 온도 8bit 소수 데이터, 8bit 패리티 비트의 40bit로 구성되어 있다.



< 데이터 타이밍도 >



< 시작 신호 및 응답 신호 >



2. DHT11 센서의 사양

- DHT11 센서의 하드웨어 사양은 <표 3-27>과 같다.

DHT11 센서 외형	모듈 항목	모듈 항목의 내용
	인터페이스	1pin Digital OUTPUT
	동작 전압	5V

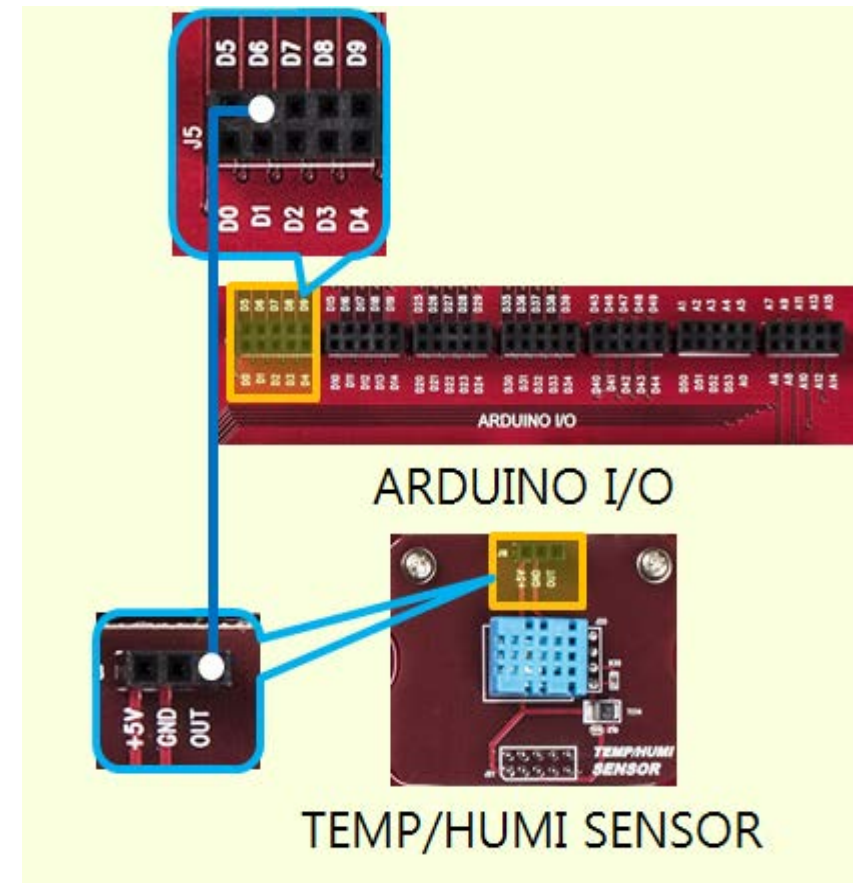
< 표 3-27 >



1. 온/습도 센서의 핀 연결

- 두 모듈을 연결하기 위해서는 케이블로 그림과 같이 HBE-ADK-2560과 온/습도 센서 모듈을 연결해야 한다.

ADK-2560 모듈 핀 번호	핀 정보	DHT11 Sensor 핀 번호
D6	Digital pin	DHT11_D





1. 온/습도 센서 프로그램 작성

PIONEER_LIGHT_DHT11.ino

```
#include <dht11.h>

dht11 DHT11;

int pin_DHT11 = 6;

void setup()
{
  Serial.begin(115200); void loop()
  {
    int chk = DHT11.read(pin_DHT11);
    switch (chk)
    {
      case DHTLIB_OK:
        Serial.print("Temperature : ");
        Serial.print(DHT11.temperature);
```

```
        Serial.print("[C] Humidity : ");
        Serial.print(DHT11.humidity);
        Serial.println("[%]");
        break;
      case DHTLIB_OK:
        Serial.print("Temperature : ");
        Serial.print(DHT11.temperature);
        Serial.print("[C] Humidity : ");
        Serial.print(DHT11.humidity);
        Serial.println("[%]");
        break;
```



```
case DHTLIB_ERROR_CHECKSUM:  
    Serial.println("Checksum error");  
    break;  
  
case DHTLIB_ERROR_TIMEOUT:  
    Serial.println("Time out error");  
    break;  
  
default:  
    Serial.println("Unknown error");  
    break;  
}  
  
delay(1000);  
}
```



1. PIONEER_LIGHT_DHT11.ino

1) 라이브러리 추가

- 프로그램에서 DHT11 함수를 사용할 수 있도록 선언한다. 이는 CD에 Source\WLibraries 폴더에서 DHT11 폴더를 arduino가 설치된 폴더 내에 Wlibraries 폴더로 복사한다.

```
#include <dht11.h>
```

2) 사용 핀 선언

- 변수를 선언하여 DHT11 모듈의 핀 번호를 저장한다.

```
int pin_DHT11 = 6;
```

3) 초기화 구문

- Serial port(UART 0)를 전송속도 115200, 데이터 비트 8, 패리티 없음, 스톱 비트 1로 설정하고 시작한다. 통신 및 프로그램 동작을 확인하기 위해 연결한다.

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



4) 온/습도 측정

- 온도 및 습도를 DHT11 센서로부터 읽어온다.

```
void loop(){  
  int chk = DHT11.read(pin_DHT11);
```

읽어온 센서 값이 정상이면 온도와 습도를 Serial로 전송한다.

```
  switch (chk)  
  {  
    case DHTLIB_OK:  
      Serial.print("Temperature : ");  
      Serial.print(DHT11.temperature);  
      Serial.print("[C] Humidity : ");  
      Serial.print(DHT11.humidity);  
      Serial.println("[%]");  
      break;
```




- 읽어온 센서 값이 정상이 아니면 error 원인에 따라 그 내용을 Serial로 전송한다.

```
case DHTLIB_ERROR_CHECKSUM:
    Serial.println("Checksum error");
    break;


case DHTLIB_ERROR_TIMEOUT:
    Serial.println("Time out error");
    break;

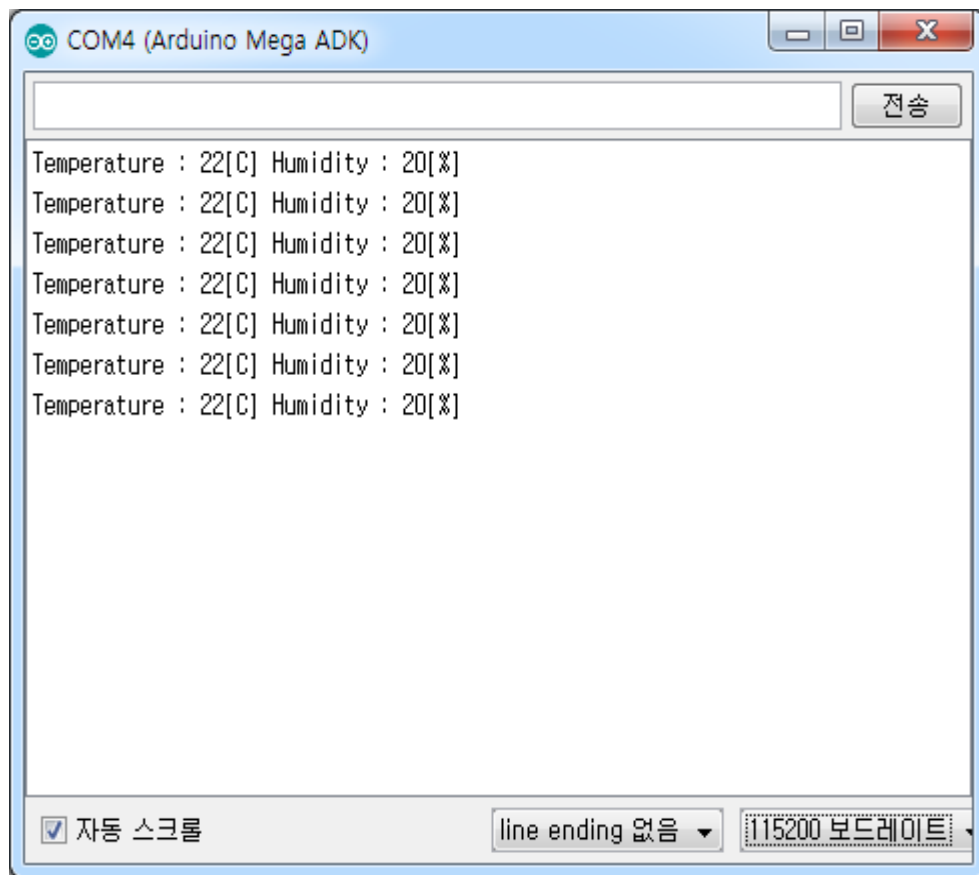
default:
    Serial.println("Unknown error");
    break;
}

delay(1000);
}
```



1. 시리얼 모니터를 및 동작 확인

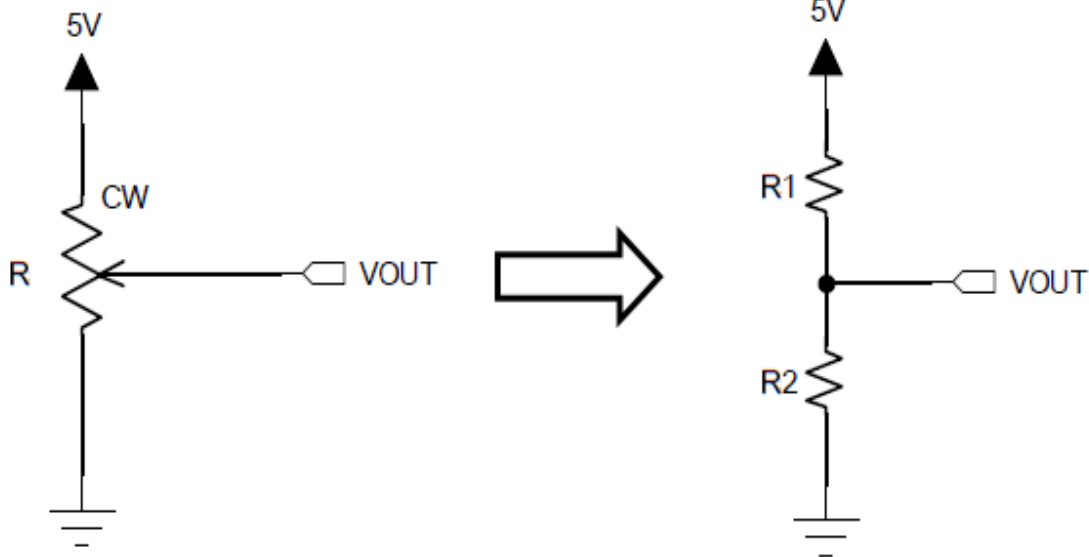
- 업로드가 완료된 후 시리얼 모니터 활성화()를 한 후 통신 속도를 115200으로 설정한다. 그림과 같이 DHT11 센서에서 온도 및 습도를 읽어 출력한다.





1. 가변저항

- 가변저항이란 전자회로에서 저항 값을 임의로 바꿀 수 있는 저항기이다.
- 여기서 저항 값을 바꾸는 방법은 접동자를 저항체 위로 슬라이딩 시켜서 저항 값을 연속 변화된다.
- 반시계 방향으로 회전 시 R1이 커지면 R2가 작아져 전압 값이 작아지고, 시계방향으로 회전 시 R1이 작아지고 R2가 커져 전압이 커진다.





2. 가변저항 모듈의 사양

- 가변저항의 하드웨어 사양은 다음과 같다.

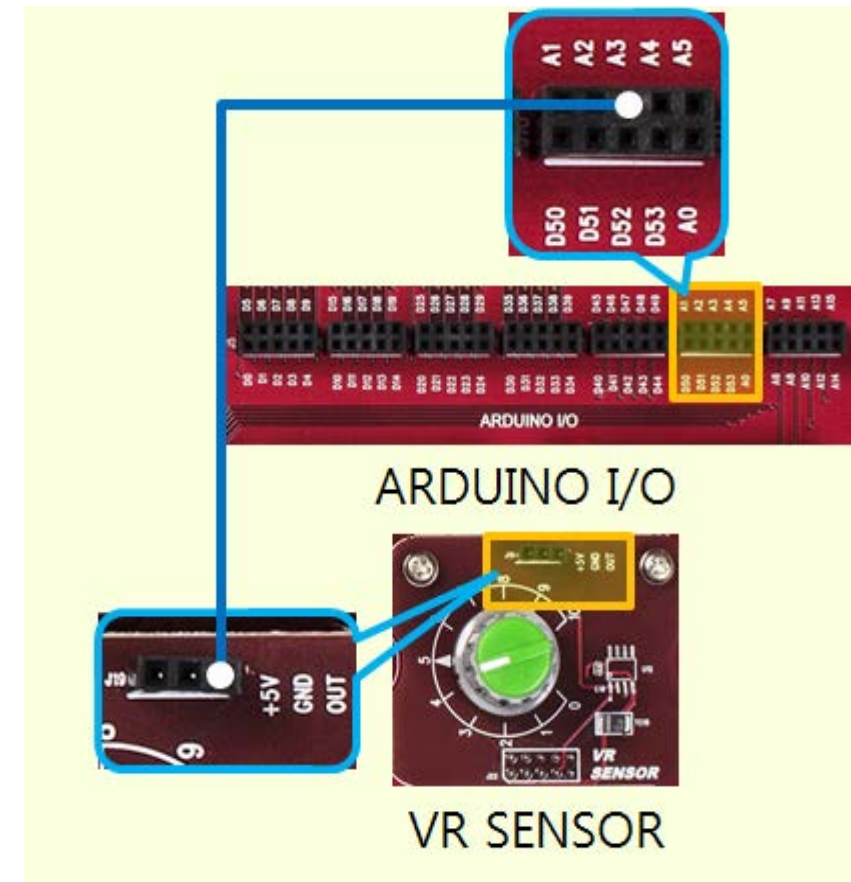
가변저항 모듈 외형	모듈 항목	모듈 항목의 내용
	센서	가변저항
	동작 전압	5V
	I/O Interface	1 Analog OUTPUT



1. 가변저항과의 핀 연결

두 모듈을 연결하기 위해서는 1핀 케이블로 그림과 같이 HBE-ADK-2560과 가변저항 모듈을 연결해야 한다.

ADK-2560 모듈 핀 번호	핀 정보	가변저항(VR) 핀 번호
A3	Analog pin	OUT





1. 가변저항 프로그램 작성

PIONEER_LIGHT_VR.ino

```
int pin_VR = A3; // VR pin number A7 setup
void setup() {
  Serial.begin(115200);
  pinMode(pin_VR, INPUT);
}
void loop() {
  float Volt = 0.0;
  unsigned int ADC_data = analogRead(pin_VR);}
```

```
Volt = (ADC_data *5.0) / 1023;
Serial.print("ADC Data : ");
Serial.print(ADC_data);
Serial.print(", Voltage : ");
Serial.print(Volt);
Serial.println("[V]");
delay(500);
}
```



1. PIONEER_VR.ino

1) 사용 핀 선언 : 변수를 선언하여 가변저항 모듈의 핀 번호를 저장한다.

```
int pin_VR = A7; // VR pin number A7 setup
```

2) 초기화 구문

- Serial port(UART 0)를 전송속도 115200, 데이터 비트 8, 패리티 없음, 스톱 비트 1로 설정하고 시작한다. 통신 및 프로그램 동작을 확인하기 위해 연결한다.

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

가변저항 핀을 입력으로 설정한다.

```
// VR analog pin Input Setup  
pinMode(pin_VR, INPUT);  
}
```



3) 가변저항 모듈의 전압 측정

- 가변저항의 ADC 핀을 읽어 ADC 값을 전압으로 변환한다.

```
void loop() {  
    float Volt = 0.0;  
    unsigned int ADC_data = analogRead(pin_VR);  
    Volt = (ADC_data * 5.0) / 1023;
```


4) 가변저항 모듈의 전압 값 출력

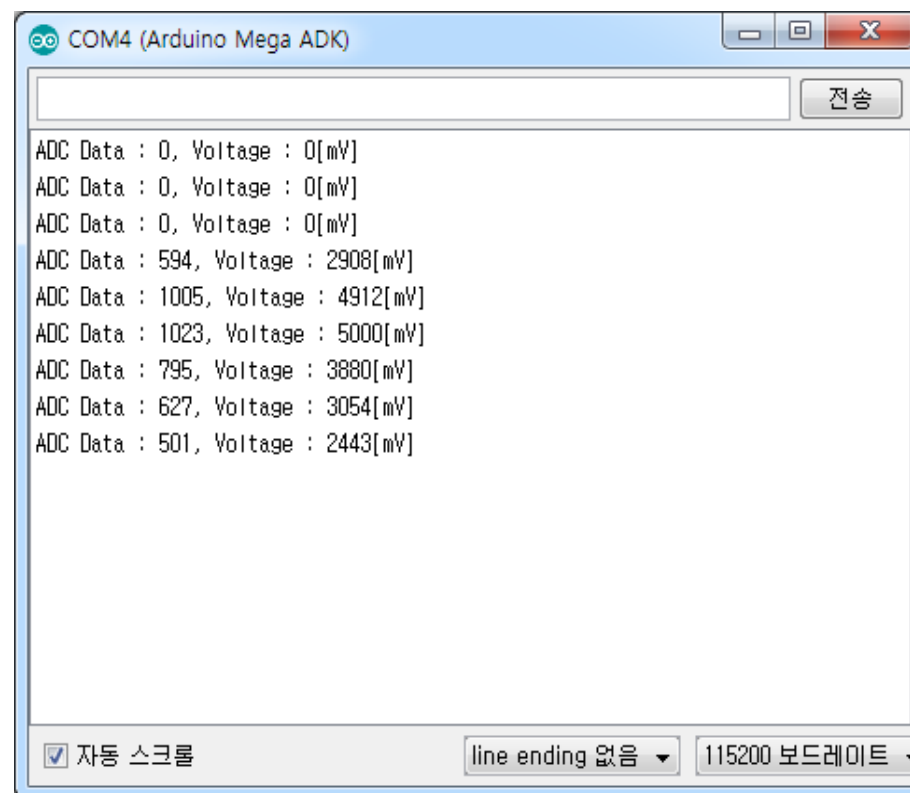
- ADC 및 전압 값을 시리얼로 전송한다. 여기서 출력 전압의 값의 단위는 [V]이다.

```
Serial.print("ADC Data : ");  
Serial.print(ADC_data);  
Serial.print(", Voltage : ");  
Serial.print(Volt);  
Serial.println("[V]");
```




1. 시리얼 모니터를 및 동작 확인

- 업로드가 완료된 후 시리얼 모니터 활성화()를 한 후 통신 속도를 115200으로 설정한다. 그림과 같이 가변저항 모듈에서 전압을 측정하여 ADC 값 및 전압을 Serial로 출력한다.





1. Bluetooth

- 블루투스(Bluetooth)는 휴대폰, 노트북, 이어폰·헤드폰 등의 휴대기기를 서로 연결하여 데이터통신하는 근거리 무선 기술을 말한다. 주로 10미터 안팎의 초단거리에서 저 전력 무선 연결이 필요할 때 사용된다.

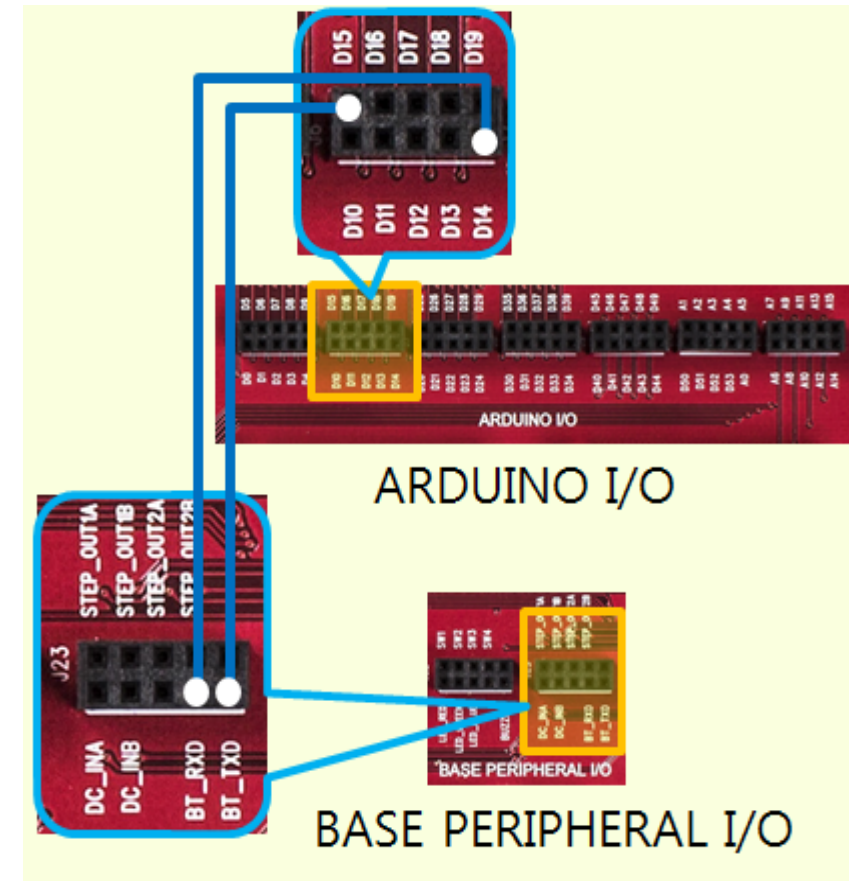




1. Bluetooth의 핀 연결

두 모듈을 연결하기 위해서는 케이블로 그림과 같이 HBE-ADK-2560과 Bluetooth 모듈을 연결해야 한다.

ADK-2560 모듈 핀 번호	핀 정보	Bluetooth 모듈 핀 번호
D14	TXD3	BT_RXD
D15	RXD3	BT_TXD





1. Bluetooth 프로그램 작성

PIONEER_LIGHT_BLUETOOTH.ino

```
void setup() {  
  Serial3.begin(9600);  
}  
  
void loop() {  
  if(Serial3.available() > 0)  
  {  
    char recv = Serial3.read();  
    Serial3.write(recv);  
  }  
}
```



1) 초기화 구문

Serial3 port(UART 3)를 전송속도 9600, 데이터 비트 8, 패리티 없음, 스톱 비트 1로 설정하고 시작한다. Bluetooth 통신을 하기 위해 연결한다.

```
Serial3.begin(9600); // same Serial3.begin(9600, SERIAL_8N1)
```

2) Bluetooth 데이터 수신 및 전송

블루투스로부터 수신된 데이터가 있으면 다음을 실행한다.
블루투스로 수신된 데이터를 읽어 recv 변수에 저장한다.

```
char recv = Serial3.read();
```

- recv 변수에 저장된 값을 다시 블루투스로 전송한다.

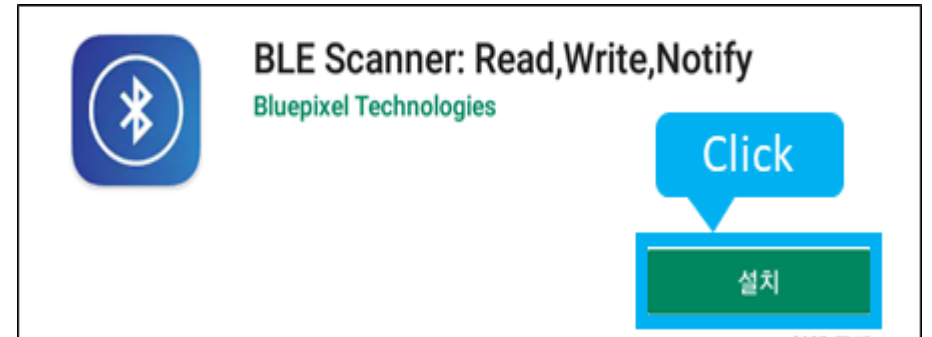
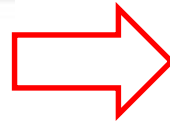
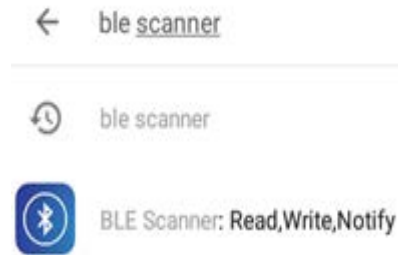
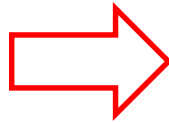
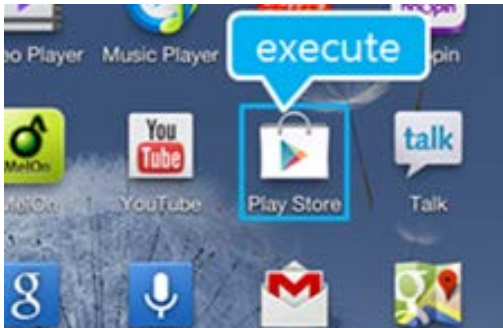
```
Serial3.write(recv);
```



1) Bluetooth 확인용 안드로이드 앱 설치

여기서는 안드로이드OS 기반의 스마트폰 등으로 시리얼 통신을 위해 프로그램 설치하는 과정을 설명한다.
(iOS에서도 동일한 어플리케이션이 사용가능하다.)

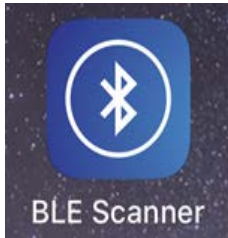
다음은 Bluepixel Technologies의 “BLE Scanner”라는 프로그램을 설치하는 과정이다.
다음 그림과 같이 Play Store를 실행하여 BLE Scanner를 검색하고 설치한다.



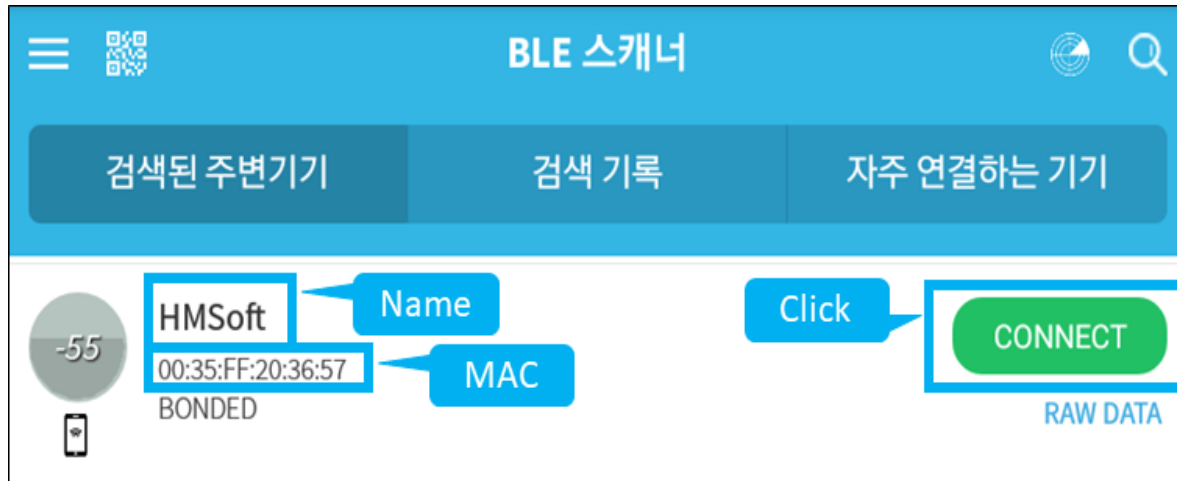


1. 프로그램 동작 확인

- 스마트폰의 블루투스 기능을 ON한 후, BLE Scanner 어플리케이션 설치가 완료되면 프로그램을 실행한다.



- BLE Scanner 어플리케이션이 실행되면 다음 그림과 같이 장치를 검색하게 되며, 검색된 HMSoft의 CONNECT를 클릭한다.
만약 여러개의 HMSoft가 검색된다면 블루투스 모듈에 써진 MAC 값과 비교하여 같은 MAC 주소의 블루투스를 선택하면 된다.





1. 프로그램 동작 확인

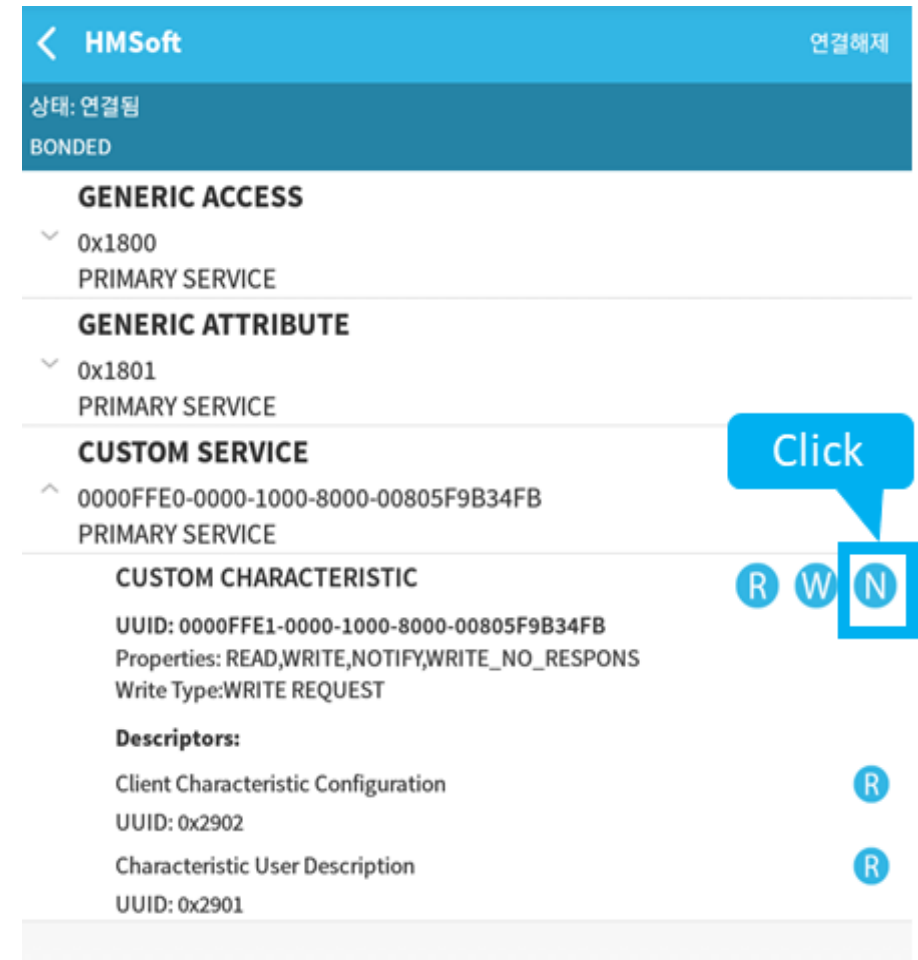
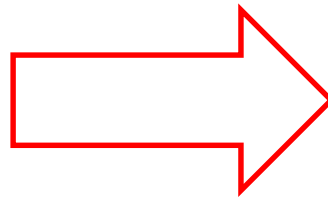
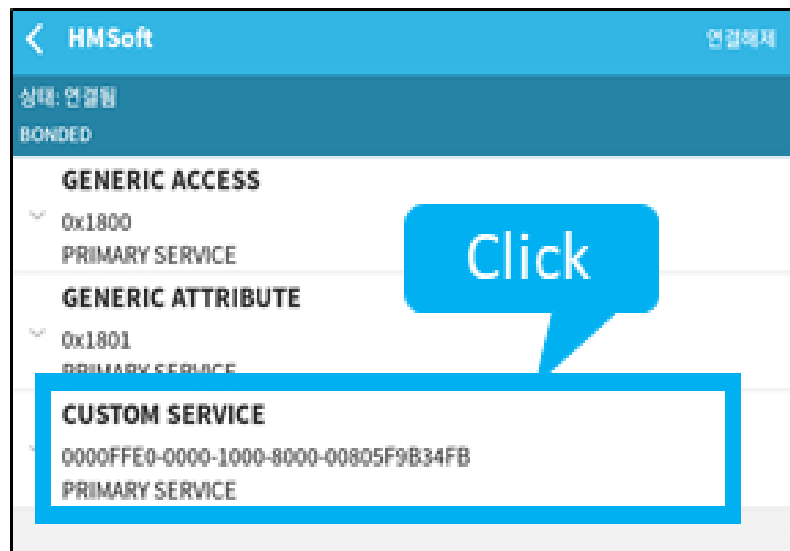
- 비밀번호를 요청한다면 숫자 '000000'을 입력한다.

The screenshot shows the HMSoft application interface. At the top, there's a blue header with a back arrow, the text 'HMSoft', and a '연결해제' (Disconnect) button. Below the header, the status is '상태: 연결됨' (Status: Connected) and 'NOT BONDED'. The main content area lists services under 'GENERIC ACCESS' and 'GENERIC ATTRIBUTE'. A modal dialog box is overlaid in the center, titled '블루투스 연결 요청' (Bluetooth connection request). It contains the text 'HMSoft' and '을(를) 등록하려 합니다.' (Attempting to register). Below this, it says '해당 디바이스의 PIN을 입력하세요:' (Enter the PIN of the device:). There is a text input field with six dots, and a blue callout bubble points to it with the text 'Input: "000000"'. Below the input field, it says '(0000 또는 1234 입력)' (Enter 0000 or 1234). There is a checkbox labeled '문자 또는 기호가 포함된 PIN' (PIN containing letters or symbols) which is currently unchecked. Below the checkbox, it says '다른 디바이스에도 PIN을 입력하세요.' (Enter PIN for other devices too.). At the bottom of the dialog, there are two buttons: '취소' (Cancel) and '확인' (Confirm). A blue callout bubble points to the '확인' button with the text 'Click'.



1. 프로그램 동작 확인

- 다음 그림과 같이 CUSTOM SERVICE를 클릭한다.
- 'N'을 클릭하여 Notification을 활성화 한다.





1. 프로그램 동작 확인

- 'W'을 클릭한다.

< HMSoft 연결해제

상태: 연결됨
BONDED

GENERIC ACCESS
0x1800
PRIMARY SERVICE

GENERIC ATTRIBUTE
0x1801
PRIMARY SERVICE

CUSTOM SERVICE
0000FFE0-0000-1000-8000-00805F9B34FB
PRIMARY SERVICE

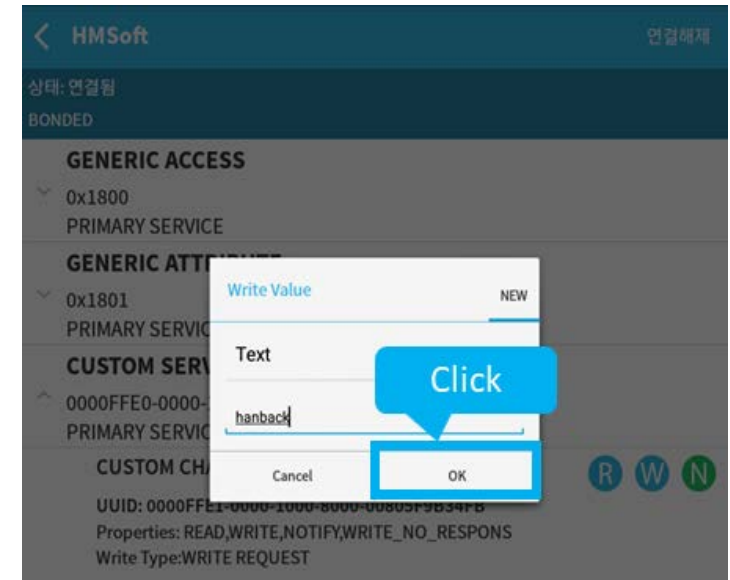
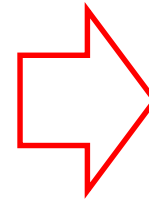
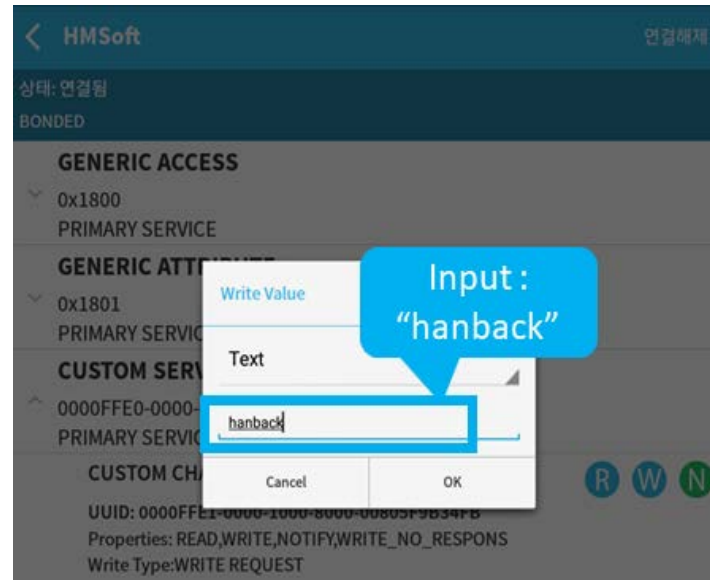
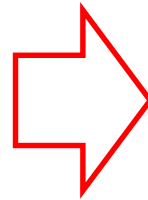
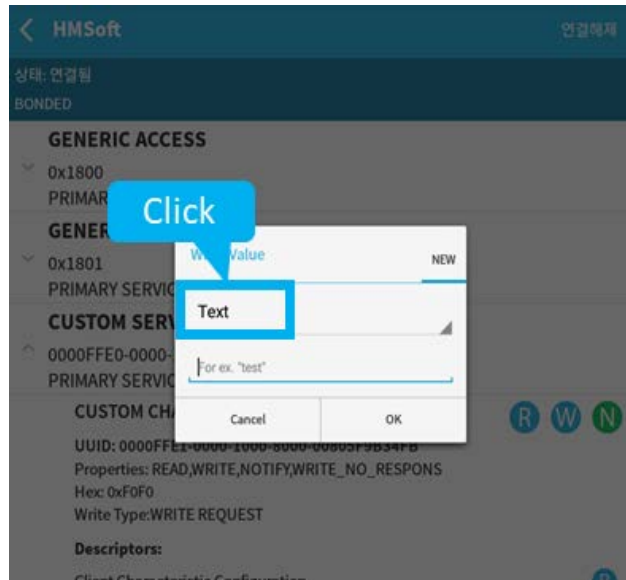
CUSTOM CHARACTERISTIC
UUID: 0000FFE1-0000-1000-8000-00805F9B34FB
Properties: READ,WRITE,NOTIFY,WRITE_NO_RESPONS
Write Type:WRITE REQUEST

Descriptors:
Client Characteristic Configuration (R)
UUID: 0x2902
Notifications 활성화됨
Characteristic User Description (R)
UUID: 0x2901



1. 프로그램 동작 확인

- Text를 선택하고 “hanback”을 입력한 후 OK를 클릭하여 데이터를 전송한다.





1. 프로그램 동작 확인

- 입력한 문자를 Bluetooth 로 전송하면 입력한 문자를 다시 전달 받아 동일 한 문자가 출력된다.

<

HMSoft

연결해제

상태: 연결됨

BONDED

GENERIC ACCESS

< 0x1800

PRIMARY SERVICE

GENERIC ATTRIBUTE

< 0x1801

PRIMARY SERVICE

CUSTOM SERVICE

< 0000FFE0-0000-1000-8000-00805F9B34FB

PRIMARY SERVICE

CUSTOM CHARACTERISTIC

UUID: 0000FFE1-0000-1000-8000-00805F9B34FB

Properties: READ,WRITE,NOTIFY,WRITE_NO_RESPONSE

Value:hanback

Hex: 0x00010E0261636B

Write Type:WRITE REQUEST

R

W

N