2020-08-05 ~ 2020-08-07

센서 활용 및

사물인터넷(IoT) 클라우드 서비스

〈 2020 충청지역 IoT 전문가 Summer School – Track B 〉

〈 센서 추가 매뉴얼 〉





강현주

센서 활용 III

초음파 거리센서 (HC-SR04)

부록. 센서 활용 III

초음파 센서 (HC-SR04)

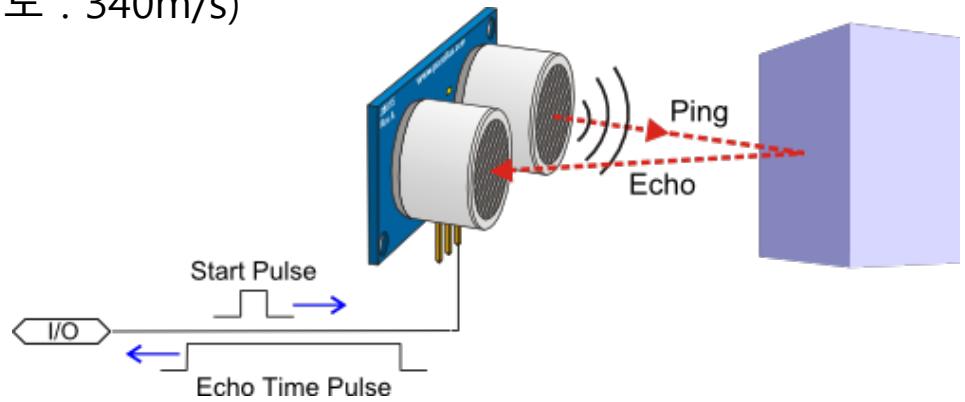
- 초음파란?
  - 사람의 귀에 들리지 않을 정도로 높은 주파수(약 20KHz 이상)의 소리
  - 특징 : 파장이 짧아 지향성과 직진성이 높으며 공기중 340m/s 의 일정한 속도로 진행

- 초음파 센서란?
  - 초음파를 쏜 후에 그 음파가 물체에 부딪혀 반사되어 오는 시간을 이용하여 거리를 측정하는 센서
  - 발신부 (Trig)
    - 함수 발생기에서 (+)와 전압을 번갈아 압전 소자에 가해주면,
    - 압전 소자의 변형에 의해 진동이 발생
    - 진동에 의해 초음파가 발생하는 역압전 현상을 이용
  - 수신부 (Echo)
    - 발신부에서 발생한 초음파가 물체에 반사되어 돌아오는 파동에 의해 압전소자가 진동
    - 진동에 의해 전압이 발생되는 정압전 현상을 이용
    - 반사되어 돌아오는 시간을 기초로 거리를 측정

부록. 센서 활용 III

초음파 센서 (HC-SR04)

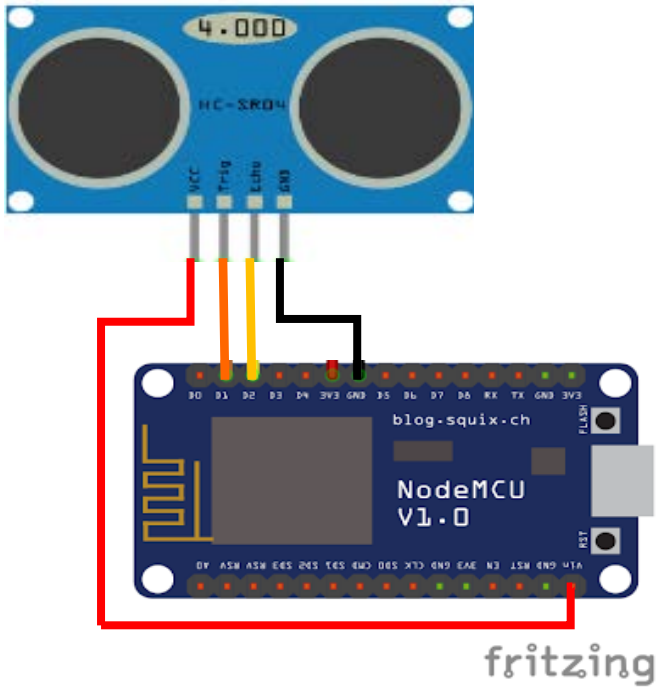
- 측정 거리 : 최대 4m
- 측정 각도 : 약15도
- 4pin : Vcc, GND, Trig, Echo
- Trig 핀에 최소 10us 펄스를 입력 → 물체에 반사된 펄스가 Echo핀으로 돌아옴
- 거리측정 :
  - Trig 핀에 펄스를 입력한 시점에서부터
  - Echo핀으로 펄스가 출력되는 시점까지의 시간을 측정한 후
  - 거리 계산 (거리 = 속도*시간, 초음파의 공기 중 속도 : 340m/s)
  - 1cm 이동하는데 약 29us의 시간이 걸림
  - 이동 거리 = 왕복시간 / 1cm 이동시간 / 2
- 주의: 작동 전압 5V



부록. 센서 활용 III

초음파 센서 (HC-SR04)

- 회로연결
  - HC-SR04 Vcc ←→ NodeMCU Vin
  - HC-SR04 GND ←→ NodeMCU GND
  - HC-SR04 Trig ←→ NodeMCU D1
  - HC-SR04 Echo ←→ NodeMCU D2



예제. NodeMCU_HCSR04_distance.ino

```
#define trigPin D1    //Trig 핀 할당
#define echoPin D2    //Echo 핀 할당

void setup()
{
   Serial.begin(115200);    //시리얼 초기화
   Serial.println("Hello Arduino!!");

   pinMode(trigPin, OUTPUT);    //Trig 핀 output으로 세팅
   pinMode(echoPin, INPUT);    //Echo 핀 input으로 세팅
}

void loop()
{
   long duration, distance;    //기본 변수 선언

   //Trig 핀으로 10us의 pulse 발생
   digitalWrite(trigPin, LOW);       //Trig 핀 Low
   delayMicroseconds(2);           //2us 유지
   digitalWrite(trigPin, HIGH);    //Trig 핀 High
   delayMicroseconds(10);           //10us 유지
   digitalWrite(trigPin, LOW);       //Trig 핀 Low

   //Echo 핀으로 들어오는 펄스의 시간 측정
   duration = pulseIn(echoPin, HIGH);     //us단위로 값을 리턴.
   //음파가 반사된 시간을 거리로 환산
   distance = duration / 29 / 2;           //센치미터로 환산

   Serial.print(distance);    Serial.print("cm");    Serial.println();
   delay(100);
}
```

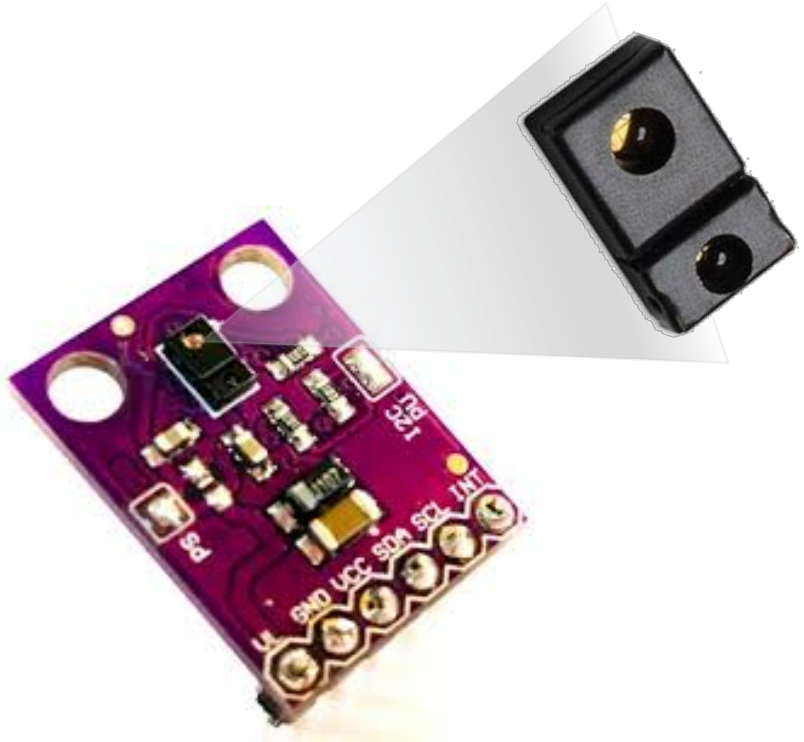
센서 활용 III

거리인식 근접센서 (APDS-9960)

부록. 센서 활용 III

거리인식 근접 센서 (APDS-9960)

- I2C 통신 소자
- 가까운 거리의 물체까지의 거리를 측정 (10cm)
- 동작 전압 : 2.4~3.6V → 3.3V만 사용 가능
- 동작 온도 : -30°C ~ +85°C
- 최대 스위칭 주파수 : 400KHz
- #include <Wire.h>
- #include <SparkFun_APDS-9960.h>
- 라이브러리 (컬러인식 센서에서) 이미 설치 되어 있어야 함



부록. 센서 활용 III

거리인식 근접 센서 (APDS-9960)

- 회로연결
  - APDS-9960 Vcc ←→ NodeMCU 3.3v
  - APDS-9960 GND ←→ NodeMCU GND
  - APDS-9960 SCL ←→ NodeMCU D1
  - APDS-9960 SDA ←→ NodeMCU D3

예제. NodeMCU_APDS9960_proximity.ino (1/3)

```
#include <Wire.h>
#include <APDS9960.h>

#define APDS9960_SDA    D3 // GPIO0
#define APDS9960_SCL    D1 // GPIO5

// Global Variables
APDS9960 apds = APDS9960();
uint8_t proximity_data = 0;
```

정수 자료형 signed	값 범위	문자열 표현시 문자수
int8_t	-128~ 127	4
int16_t	-32,768 ~ 32,767	6
int32_t	-2,147,483,648 ~ 2,147,483,647	11
int64_t	-9,223,372,036,854,775,808 ~ 9,223,372,036,854,775,807	21

정수 자료형 unsigned	값 범위	문자열 표현시 문자수
uint8_t	0 ~ 255	3
uint16_t	0 ~ 65,535	5
uint32_t	0 ~ 4,294,967,295	10
uint64_t	0 ~ 18,446,744,073,709,551,615	20

부록. 센서 활용 III

거리인식 근접 센서 (APDS-9960)

- 회로연결
  - APDS-9960 Vcc ←→ NodeMCU 3.3v
  - APDS-9960 GND ←→ NodeMCU GND
  - APDS-9960 SCL ←→ NodeMCU D1
  - APDS-9960 SDA ←→ NodeMCU D3

예제. NodeMCU_APDS9960_proximity.ino (2/3)

```
void setup() {
  Wire.begin(APDS9960_SDA, APDS9960_SCL);

  // Initialize Serial port
  Serial.begin(9600);
  Serial.println(F("SparkFun APDS-9960 - ProximitySensor"));

  // Initialize APDS-9960 (configure I2C and initial values)
  if ( apds.init() ) {
    Serial.println(F("APDS-9960 initialization complete"));
  } else {
    Serial.println(F("Something went wrong during APDS-9960 init!"));
  }

  // Adjust the Proximity sensor gain
  if ( !apds.setProximityGain(PGAIN_2X) ) {
    Serial.println(F("Something went wrong trying to set PGAIN"));
  }

  // Start running the APDS-9960 proximity sensor (no interrupts)
  if ( apds.enableProximitySensor(false) ) {
    Serial.println(F("Proximity sensor is now running"));
  } else {
    Serial.println(F("Something went wrong during sensor init!"));
  }

}
```

부록. 센서 활용 III

거리인식 근접 센서 (APDS-9960)

- 회로연결
  - APDS-9960 Vcc ←→ NodeMCU 3.3v
  - APDS-9960 GND ←→ NodeMCU GND
  - APDS-9960 SCL ←→ NodeMCU D1
  - APDS-9960 SDA ←→ NodeMCU D3

예제. NodeMCU_APDS9960_proximity.ino (3/3)

```
void loop() {

  // Read the proximity value
  if ( !apds.readProximity(proximity_data) ) {
    Serial.println("Error reading proximity value");
  } else {
    Serial.print("Proximity: ");
    Serial.println(proximity_data);
  }

  // Wait 250 ms before next reading
  delay(1000);
}
```

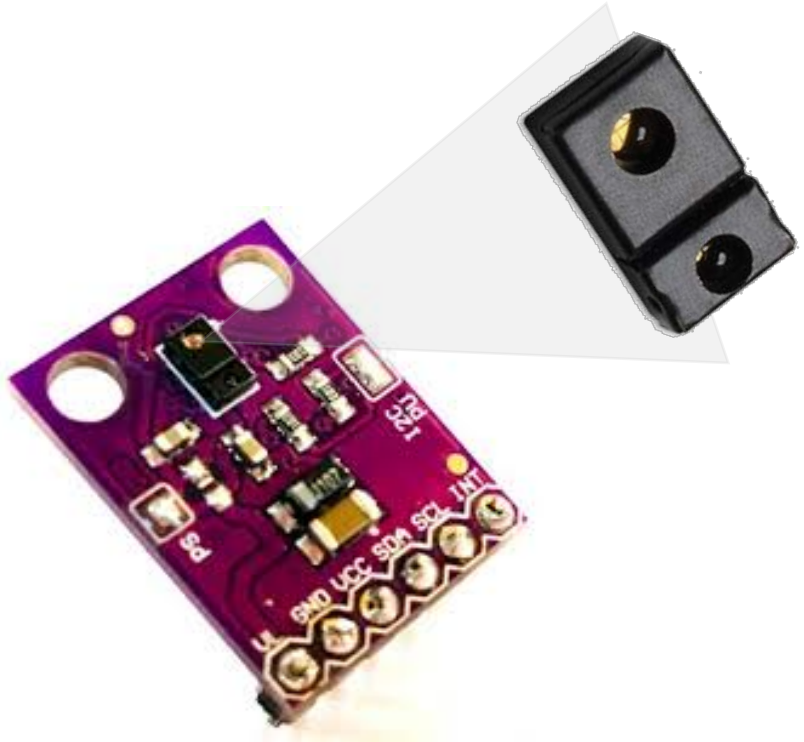
센서 활용 III

제스처 센서 (APDS-9960)

부록. 센서 활용 III

제스쳐 센서 (APDS-9960)

- I2C 통신 소자
- 위, 아래, 왼쪽, 오른쪽으로 스와이프(swipe) 동작 인식
- 멀리에서 가까이 Near 동작 인식 (2인치 이내)
- 가까이에서 멀리 Far 동작 인식 (2인치 이내)
- 적어도 1초 정도 동작 유지 (너무 빠른 속도는 인식 불가)
- 동작 전압 : 2.4~3.6V → 3.3V만 사용 가능
- #include <Wire.h>
- #include <SparkFun_APDS-9960.h>
- 라이브러리 (컬러인식 센서에서) 이미 설치 되어 있어야 함



부록. 센서 활용 III

제스쳐 센서 (APDS-9960)

- 회로연결
  - APDS-9960 Vcc ←→ NodeMCU 3.3v
  - APDS-9960 GND ←→ NodeMCU GND
  - APDS-9960 SCL ←→ NodeMCU D1
  - APDS-9960 SDA ←→ NodeMCU D3
  - APDS-9960 INT ←→ NodeMCU D6

예제. NodeMCU_APDS9960_gesture.ino (1/3)

```
#include <Wire.h>
#include <APDS9960.h>

// Pins
//#define APDS9960_INT    D6 // GPIO12 Needs to be an interrupt pin
#define APDS9960_SDA    D3 // GPIO0
#define APDS9960_SCL    D1 // GPIO5

// Constants

// Global Variables
APDS9960 apds = APDS9960();
//int isr_flag = 0;
```

부록. 센서 활용 III

제스쳐 센서 (APDS-9960)

- 회로연결
  - APDS-9960 Vcc ←→ NodeMCU 3.3v
  - APDS-9960 GND ←→ NodeMCU GND
  - APDS-9960 SCL ←→ NodeMCU D1
  - APDS-9960 SDA ←→ NodeMCU D3
  - APDS-9960 INT ←→ NodeMCU D6

예제. NodeMCU_APDS9960_gesture.ino (2/3)

```
void setup() {
  Wire.begin(APDS9960_SDA, APDS9960_SCL);
  // Set interrupt pin as input
  //  pinMode(APDS9960_INT, INPUT);

  // Initialize Serial port
  Serial.begin(9600);

  // Initialize interrupt service routine
  //  attachInterrupt(0, interruptRoutine, FALLING);

  // Initialize APDS-9960 (configure I2C and initial values)
  if ( apds.init() ) {
    Serial.println(F("APDS-9960 initialization complete"));
  } else {
    Serial.println(F("Something went wrong during APDS-9960 init!"));
  }

  // Start running the APDS-9960 gesture sensor engine
  if ( apds.enableGestureSensor(true) ) {
    Serial.println(F("Gesture sensor is now running"));
  } else {
    Serial.println(F("Something went wrong during gesture sensor init!"));
  }

}
```

부록. 센서 활용 III

제스쳐 센서 (APDS-9960)

- 회로연결
  - APDS-9960 Vcc ←→ NodeMCU 3.3v
  - APDS-9960 GND ←→ NodeMCU GND
  - APDS-9960 SCL ←→ NodeMCU D1
  - APDS-9960 SDA ←→ NodeMCU D3
  - APDS-9960 INT ←→ NodeMCU D6

예제. NodeMCU_APDS9960_gesture.ino (3/3)

```
void loop() {
  if ( apds.isGestureAvailable() ) {
      switch ( apds.readGesture() ) {
        case DIR_UP:
          Serial.println("UP");
          break;

        case DIR_DOWN:
          Serial.println("DOWN");
          break;

        case DIR_LEFT:
          Serial.println("LEFT");
          break;

        case DIR_RIGHT:
          Serial.println("RIGHT");
          break;

        case DIR_NEAR:
          Serial.println("NEAR");
          break;

        case DIR_FAR:
          Serial.println("FAR");
          break;

        default:
          Serial.println("NONE");
      }
    }
  delay(1000);
}
```

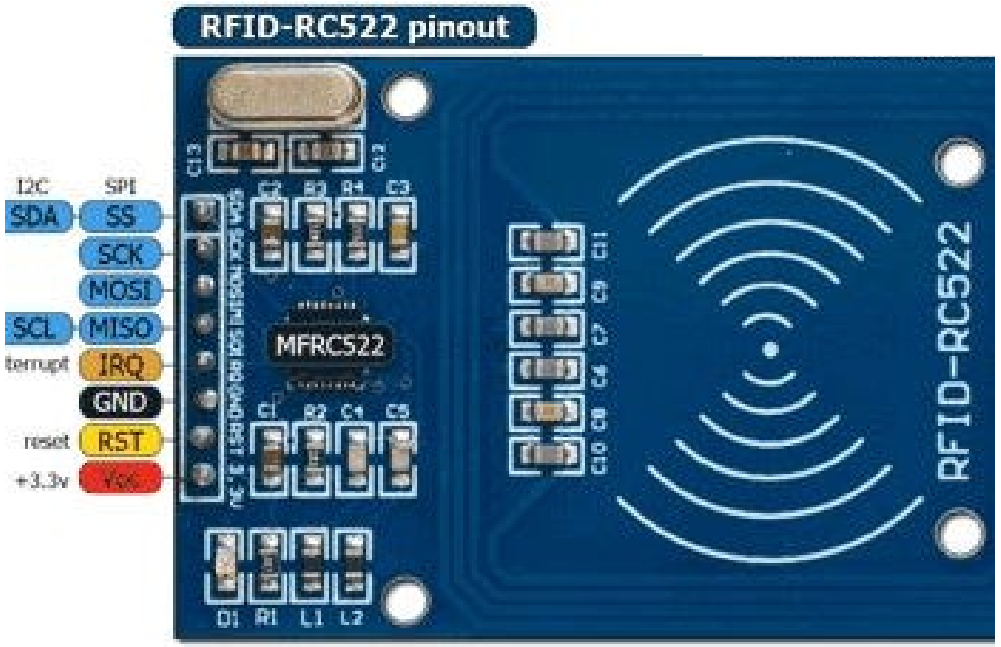
센서 활용 III

RFID IC 카드리더 (MFRC5200)

부록. 센서 활용 III

RFID IC 카드 리더 (MFRC 5200)

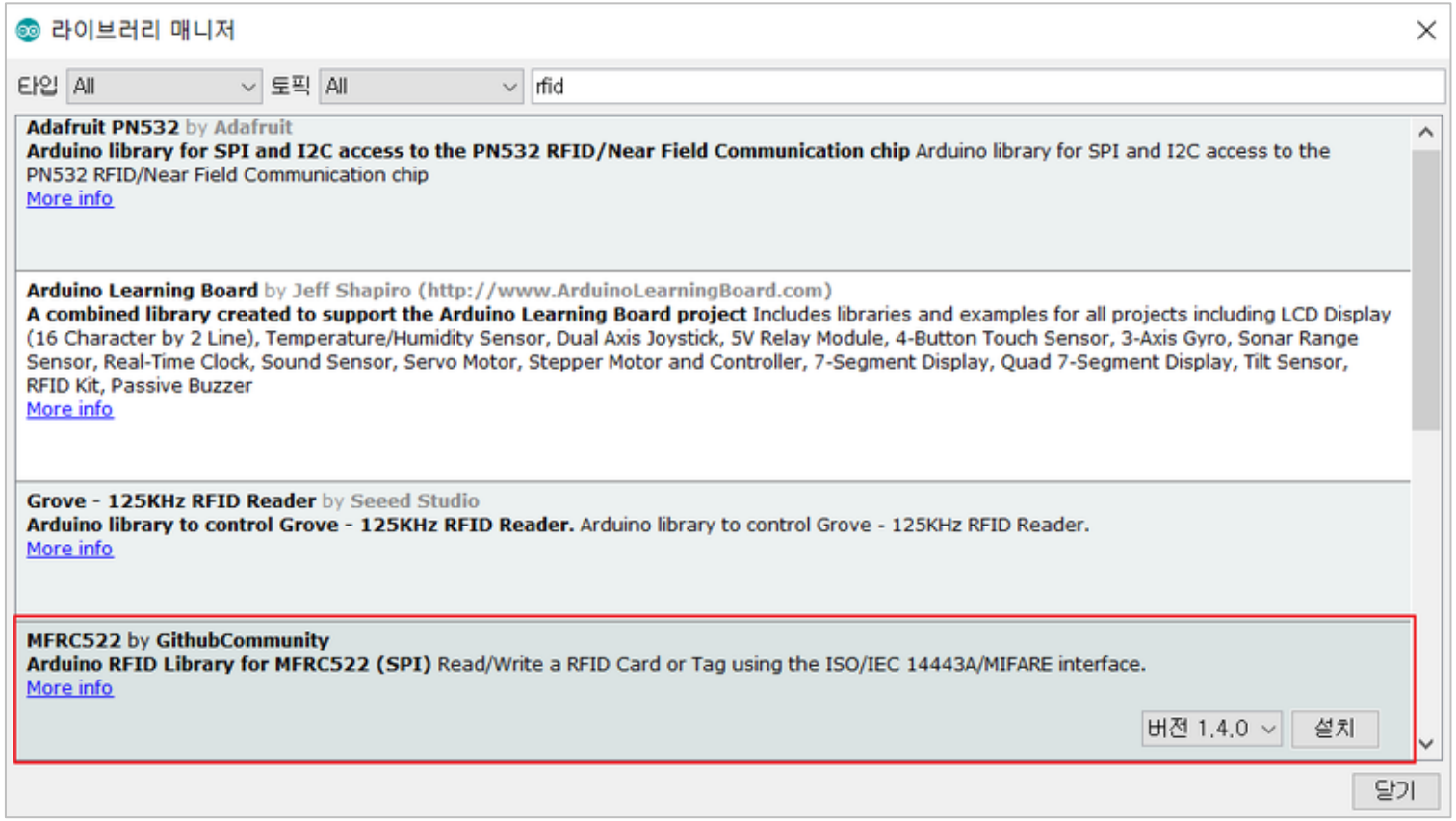
- RFID (Radio-Frequency Identification)
- 자기장과 안테나를 통해 데이터 통신
- 아주 적은 전류를 사용
- 태그와 리더기가 필요함



부록. 센서 활용 III

RFID IC 카드 리더 (MFRC 5200)

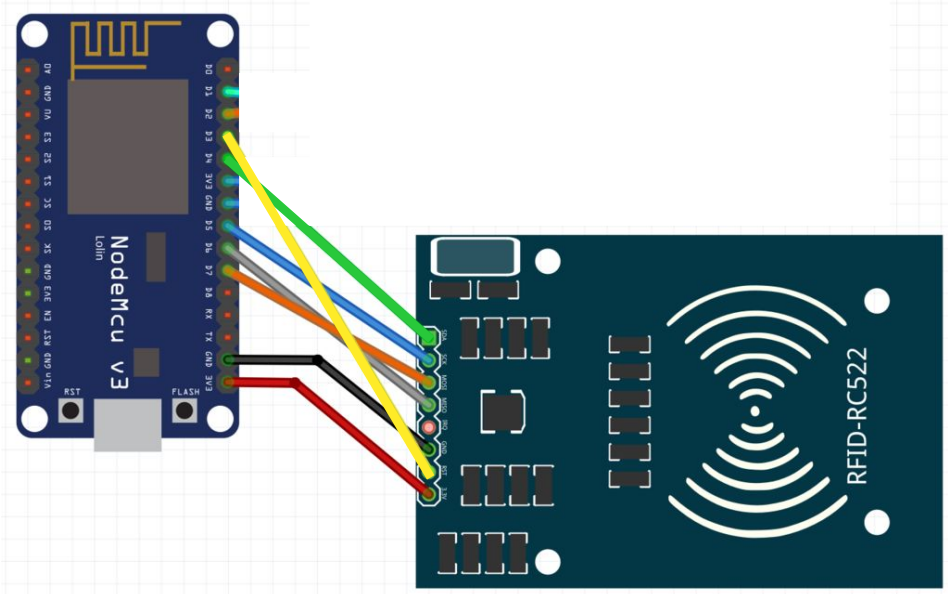
- 라이브러리 설치
  - 아두이노 IDE의 메뉴에서, [스케치/라이브러리 포함하기/라이브러리 관리...] 실행
  - 라이브러리 매니저 창에서 RFID 검색
  - MFRC522 설치
    - 마우스 커서를 해당 영역에 올려 놓으면 '설치 ' 버튼이 나타남 → 설치 버튼 클릭



부록. 센서 활용 III

RFID IC 카드 리더 (MFRC 5200)

- 회로연결
  - NodeMCU D4(2) ←→ MFRC522 SDA(SS)
  - NodeMCU D5(14) ←→ MFRC522 SCK
  - NodeMCU D7(13) ←→ MFRC522 MOSI
  - NodeMCU D6(12) ←→ MFRC522 MISO
  - NC ←→ MFRC522 IRQ
  - NodeMCU GND ←→ MFRC522 GND
  - NodeMCU D3(0) ←→ MFRC522 RST
  - NodeMCU 3.3v ←→ MFRC522 Vcc



예제. MFRC522 > DumpInfo.ino (1/2)

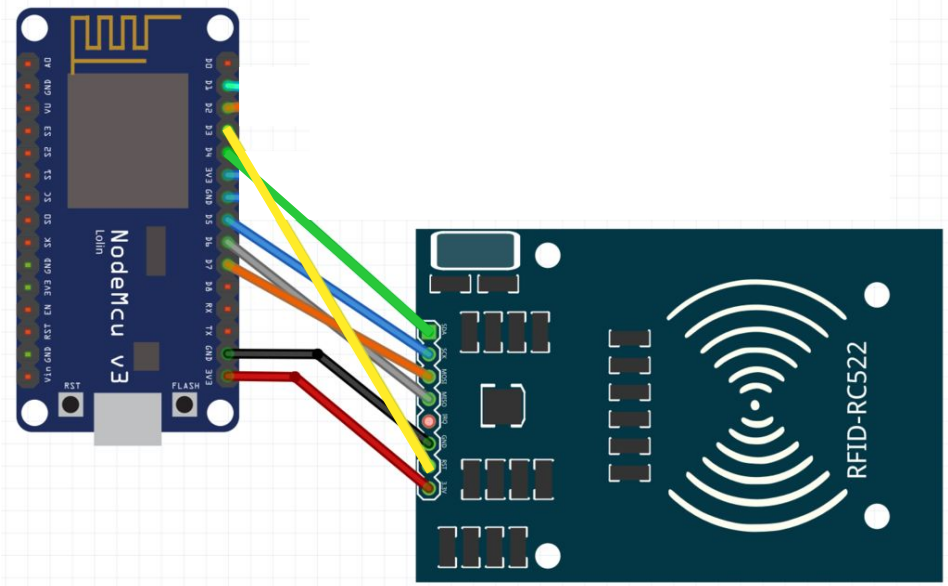
```
#include <SPI.h>
#include <MFRC522.h>

#define RST_PIN  0          // Configurable, see typical pin layout above
#define SS_PIN   2          // Configurable, see typical pin layout above

MFRC522 mfrc522(SS_PIN, RST_PIN);  // Create MFRC522 instance

void setup() {
  // Initialize serial communications with the PC
  Serial.begin(9600);

  // Do nothing if no serial port is opened
  while (!Serial);

  // Init SPI bus
  SPI.begin();

  // Init MFRC522
  mfrc522.PCD_Init();

  // Optional delay.
  delay(4);

  // Show details of PCD - MFRC522 Card Reader details
  mfrc522.PCD_DumpVersionToSerial();
  Serial.println(F("Scan PICC to see UID, SAK, type, and data blocks..."));
}
```

부록. 센서 활용 III

예제. MFRC522 > DumpInfo.ino (2/2)

RFID IC 카드 리더 (MFRC 5200)

- 회로연결
  - NodeMCU D4(2) ←→ MFRC522 SDA(SS)
  - NodeMCU D5(14) ←→ MFRC522 SCK
  - NodeMCU D7(13) ←→ MFRC522 MOSI
  - NodeMCU D6(12) ←→ MFRC522 MISO
  - NC ←→ MFRC522 IRQ
  - NodeMCU GND ←→ MFRC522 GND
  - NodeMCU D3(0) ←→ MFRC522 RST
  - NodeMCU 3.3v ←→ MFRC522 Vcc

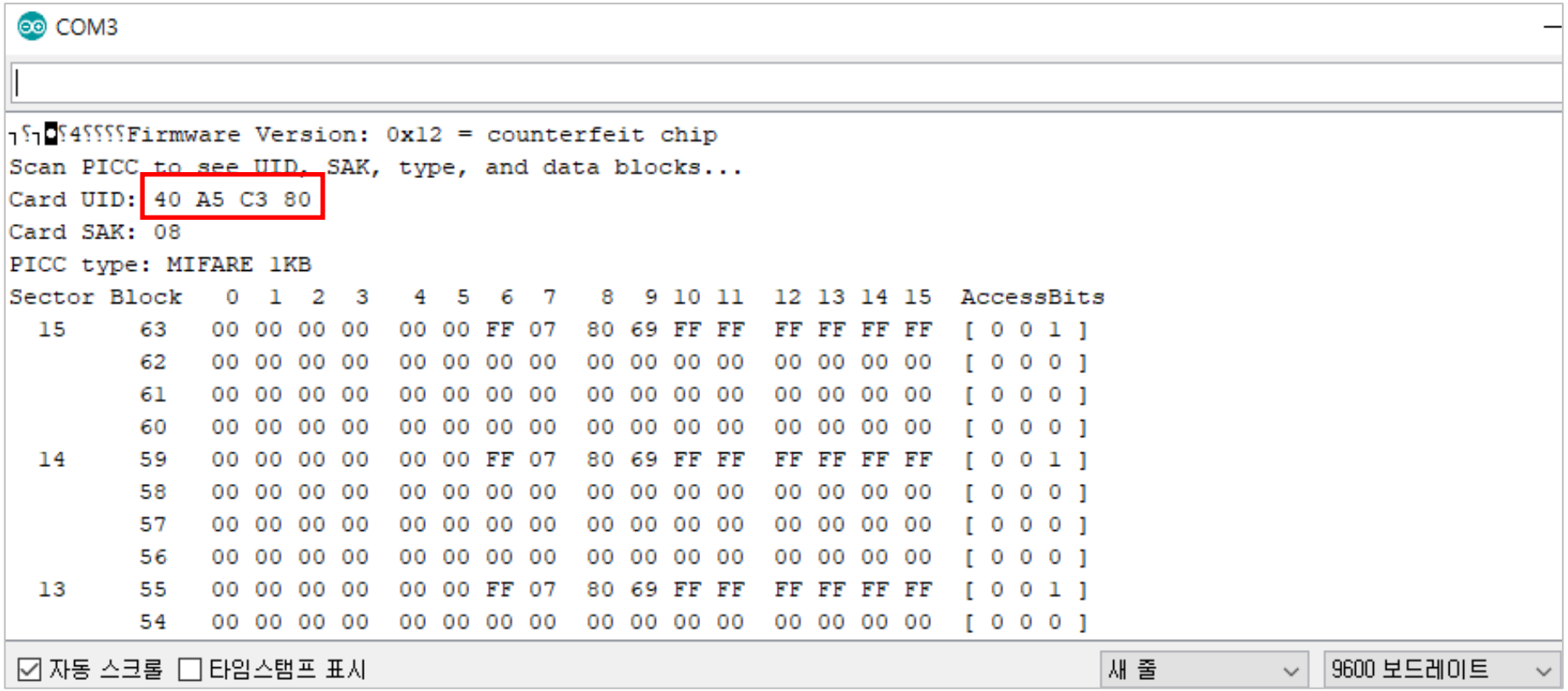



```
void loop() {
   // Reset the loop if no new card present on the sensor/reader.
   //This saves the entire process when idle.
   if ( ! mfrc522.PICC_IsNewCardPresent()) {
      return;
   }

   // Select one of the cards
   if ( ! mfrc522.PICC_ReadCardSerial()) {
      return;
   }

   // Dump debug info about the card; PICC_HaltA() is automatically called
   mfrc522.PICC_DumpToSerial(&(mfrc522.uid));
}
```

부록. 센서 활용 III

RFID IC 카드 리더 (MFRC 5200)

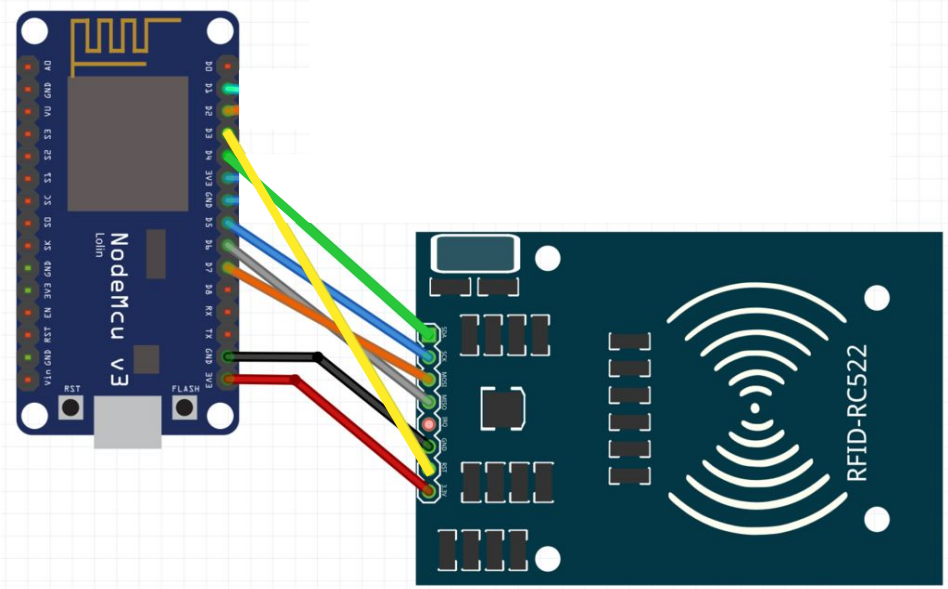
- 코드 업로드 후 시리얼 모니터 확인
  - 열쇠고리형 Card UID : 40 A5 C3 80
  - 카드형 Card UID : 00 95 9D A3



부록. 센서 활용 III

RFID IC 카드 리더 (MFRC 5200)

- 회로연결
  - NodeMCU D4(2) ←→ MFRC522 SDA(SS)
  - NodeMCU D5(14) ←→ MFRC522 SCK
  - NodeMCU D7(13) ←→ MFRC522 MOSI
  - NodeMCU D6(12) ←→ MFRC522 MISO
  - NC ←→ MFRC522 IRQ
  - NodeMCU GND ←→ MFRC522 GND
  - NodeMCU D3(0) ←→ MFRC522 RST
  - NodeMCU 3.3v ←→ MFRC522 Vcc



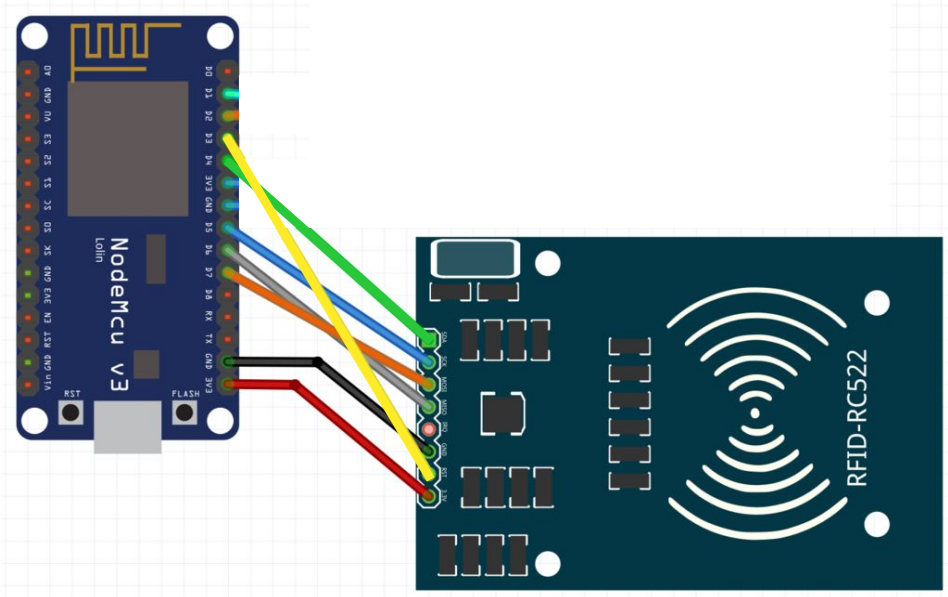
예제. NodeMCU_MFRC_accept.ino (1/2)

```
#include "SPI.h"
#include "MFRC522.h"

#define RST_PIN 0 //D3
#define SS_PIN 2 // D4

MFRC522 mfrc522(SS_PIN, RST_PIN); //Create MFRC522 instance
int status = 0;


void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200); //Initiate a serial communication
  SPI.begin(); //Initiate SPI bus
  mfrc522.PCD_Init(); // Initiate MFRC522

  Serial.println("카드를 리더기에 가까이 대 주세요");
  Serial.println();

}
```

부록. 센서 활용 III

RFID IC 카드 리더 (MFRC 5200)

- 회로연결
  - NodeMCU D4(2) ←→ MFRC522 SDA(SS)
  - NodeMCU D5(14) ←→ MFRC522 SCK
  - NodeMCU D7(13) ←→ MFRC522 MOSI
  - NodeMCU D6(12) ←→ MFRC522 MISO
  - NC ←→ MFRC522 IRQ
  - NodeMCU GND ←→ MFRC522 GND
  - NodeMCU D3(0) ←→ MFRC522 RST
  - NodeMCU 3.3v ←→ MFRC522 Vcc



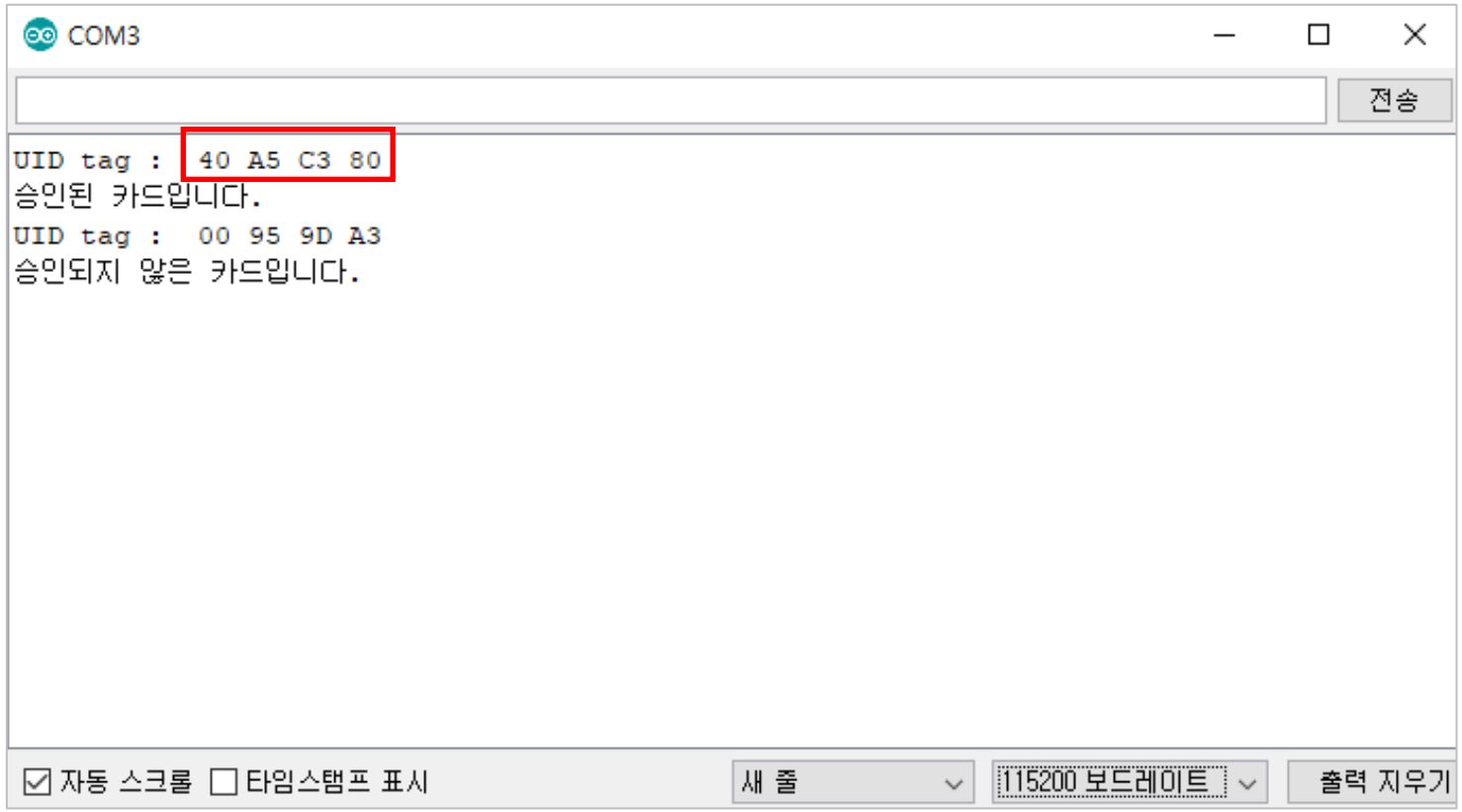
예제. NodeMCU_MFRC_accept.ino (2/2)

```
void loop() {
  //Look for new cards
  if(!mfrc522.PICC_IsNewCardPresent()){
    return;
  }
  //Select one of cards
  if(!mfrc522.PICC_ReadCardSerial()){
    return;
  }

  //Show UID on Serial monitor
  Serial.print("UID tag : ");
  String content= "";
  byte letter;
  for (byte i = 0; i < mfrc522.uid.size; i++)
  {
     Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
     Serial.print(mfrc522.uid.uidByte[i], HEX);
     content.concat(String(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " "));
     content.concat(String(mfrc522.uid.uidByte[i], HEX));
  }
  content.toUpperCase();
  Serial.println();

  if(content.substring(1) == "40 A5 C3 80") {
    Serial.println("승인된 카드입니다.");
    status =1;
    delay(3000);
  } else {
    Serial.println("승인되지 않은 카드입니다.");
    delay(3000);
  }
}
```

부록. 센서 활용 III

RFID IC 카드 리더 (MFRC 5200)

- 코드 업로드 후 시리얼 모니터 확인
  - 열쇠고리형 Card UID : 40 A5 C3 80
  - 카드형 Card UID : 00 95 9D A3

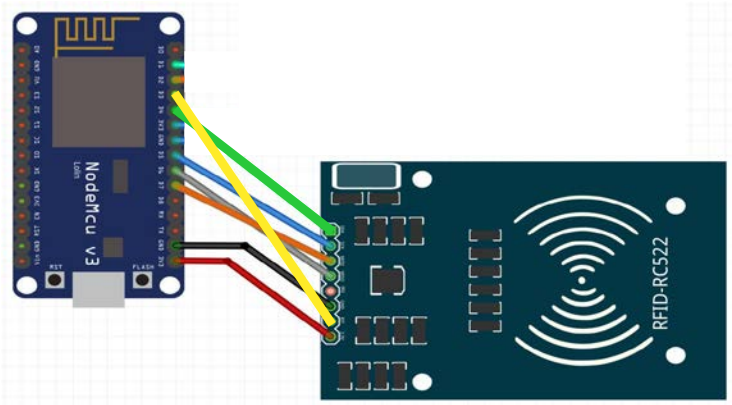


부록. 센서 활용 III

RFID IC 카드 리더 (MFRC 5200)

- 회로연결
  - NodeMCU D4(2) ←→ MFRC522 SDA(SS)
  - NodeMCU D5(14) ←→ MFRC522 SCK
  - NodeMCU D7(13) ←→ MFRC522 MOSI
  - NodeMCU D6(12) ←→ MFRC522 MISO
  - NC ←→ MFRC522 IRQ
  - NodeMCU GND ←→ MFRC522 GND
  - NodeMCU D3(0) ←→ MFRC522 RST
  - NodeMCU 3.3v ←→ MFRC522 Vcc

  - NodeMCU D8 ←→ 부저 (+)
  - NodeMCU GND ←→ 부저 (-)



예제. NodeMCU_MFRC_accept_buzzer.ino (1/3)

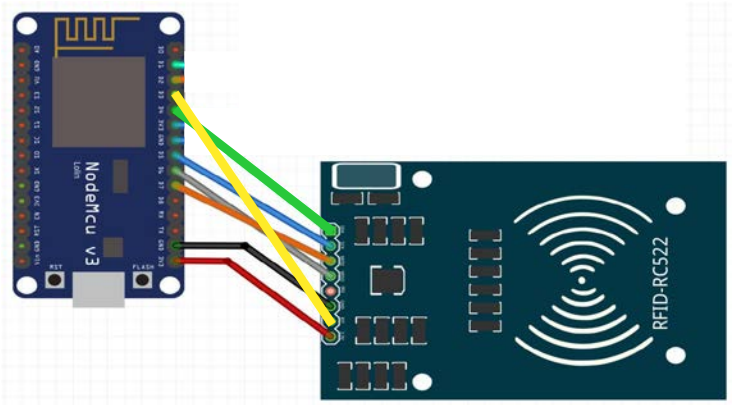
```
#include "SPI.h"
#include "MFRC522.h"

#define RST_PIN 0 //D3
#define SS_PIN 2 // D4

MFRC522 mfrc522(SS_PIN, RST_PIN); //Create MFRC522 instance
int status = 0;

int buzzer = 15; //D8
int tone_accept = 440;
int tone_deny = 330;


void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200); //Initiate a serial communication
  SPI.begin(); //Initiate SPI bus
  mfrc522.PCD_Init(); // Initiate MFRC522

  Serial.println("카드를 리더기에 가까이 대 주세요");
  Serial.println();

  pinMode(buzzer, OUTPUT);
}

//.....
//.....
```

부록. 센서 활용 III

RFID IC 카드 리더 (MFRC 5200)

- 회로연결
  - NodeMCU D4(2) ←→ MFRC522 SDA(SS)
  - NodeMCU D5(14) ←→ MFRC522 SCK
  - NodeMCU D7(13) ←→ MFRC522 MOSI
  - NodeMCU D6(12) ←→ MFRC522 MISO
  - NC ←→ MFRC522 IRQ
  - NodeMCU GND ←→ MFRC522 GND
  - NodeMCU D3(0) ←→ MFRC522 RST
  - NodeMCU 3.3v ←→ MFRC522 Vcc

  - NodeMCU D8 ←→ 부저 (+)
  - NodeMCU GND ←→ 부저 (-)



예제. NodeMCU_MFRC_accept_buzzer.ino (2/3)

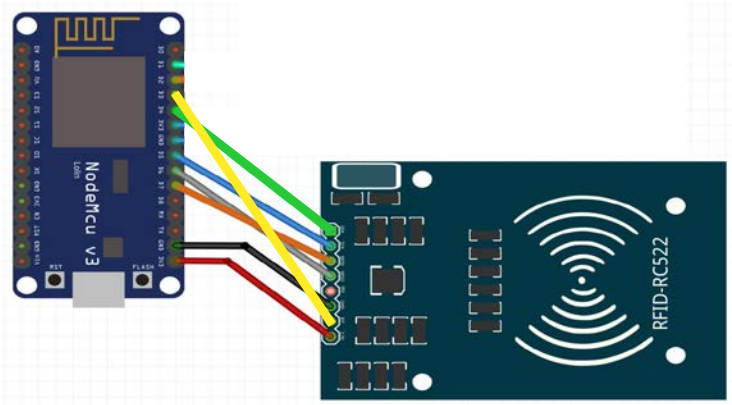
```
//.....
//.....

void loop() {
  // put your main code here, to run repeatedly:

  //Look for new cards
  if(!mfrc522.PICC_IsNewCardPresent()){
    return;
  }
  //Select one of cards
  if(!mfrc522.PICC_ReadCardSerial()){
    return;
  }

  //Show UID on Serial monitor
  Serial.print("UID tag : ");
  String content= "";
  byte letter;
  for (byte i = 0; i < mfrc522.uid.size; i++)
  {
     Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
     Serial.print(mfrc522.uid.uidByte[i], HEX);
     content.concat(String(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " "));
     content.concat(String(mfrc522.uid.uidByte[i], HEX));
  }
  content.toUpperCase();
  Serial.println();


//.....
//.....
```

부록. 센서 활용 III

RFID IC 카드 리더 (MFRC 5200)

- 회로연결
  - NodeMCU D4(2) ←→ MFRC522 SDA(SS)
  - NodeMCU D5(14) ←→ MFRC522 SCK
  - NodeMCU D7(13) ←→ MFRC522 MOSI
  - NodeMCU D6(12) ←→ MFRC522 MISO
  - NC ←→ MFRC522 IRQ
  - NodeMCU GND ←→ MFRC522 GND
  - NodeMCU D3(0) ←→ MFRC522 RST
  - NodeMCU 3.3v ←→ MFRC522 Vcc

  - NodeMCU D8 ←→ 부저 (+)
  - NodeMCU GND ←→ 부저 (-)



예제. NodeMCU_MFRC_accept_buzzer.ino (3/3)

```
//.....
//.....

  int tones;
  if(content.substring(1) == "40 A5 C3 80") {
    beep(tone_accept);
    Serial.println("승인된 카드입니다.");
    status =1;
    delay(3000);

  } else {
    beep(tone_deny);
    Serial.println("승인되지 않은 카드입니다.");
    delay(3000);
  }

} //end of loop()

void beep(int tones){
  tone(buzzer, tones);
  delay(300);
  noTone(buzzer);
  delay(100);
}
```

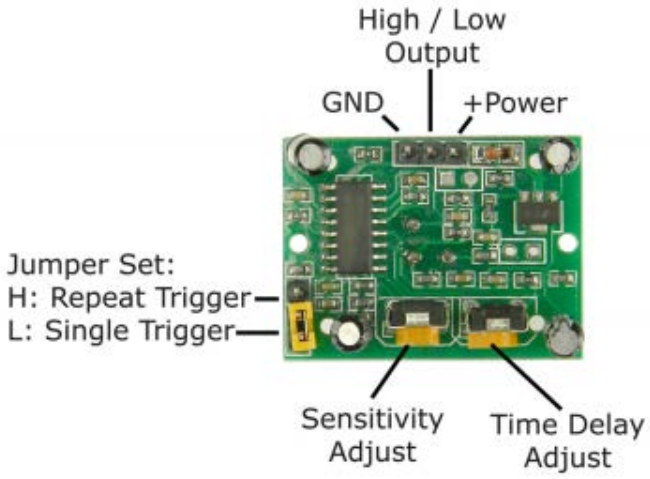
센서 활용 III

PIR 적외선 모션감지 센서 (HC-SR501)

부록. 센서 활용 III

PIR (적외선 모션 감지) 센서 (HC-SR501)

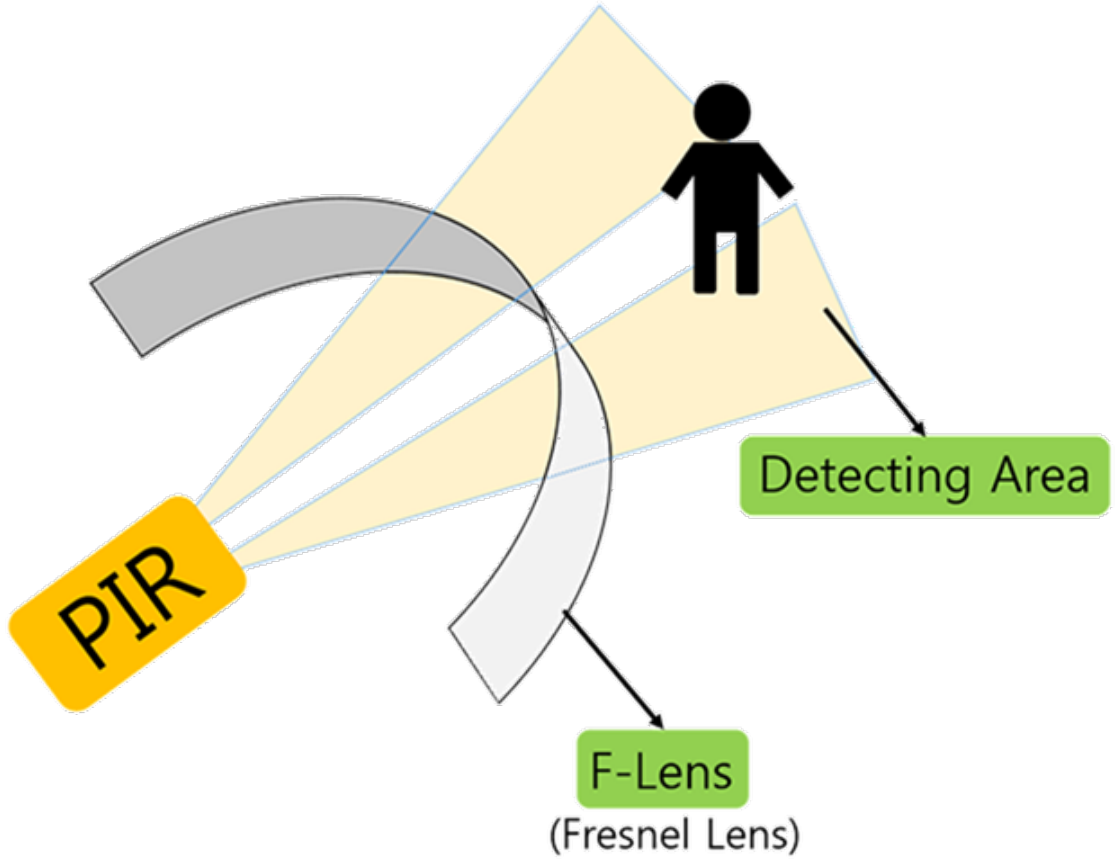
- Passive Infrared sensor
- 사람의 몸에서 방사되는 적외선 움직임(모션)의 유무를 판단
- 측정 거리 : 7m 이하
- 감지 범위 : 정면 기준 최대 110도
- 동작전압 : 5~20V
- 출력전압 : 3.3V
- 지연 시간 : 약 0.3~5분
- 덮개 : 편광 필터
- 특징 : 가변저항으로 센서의 감도와 지연 시간 조절



부록. 센서 활용 III

PIR (적외선 모션 감지) 센서 (HC-SR501) 동작원리

- 인체에서 약 9um~11um 정도의 적외선이 방출
- 방출된 적외선이 집광렌즈인 Fresnel Lens를 통과
- 센서 표면부에 위치한 Window에 도달
- 적외선 신호가 전압으로 출력
- 내장된 증폭기로 전달
- 조명 On/Off

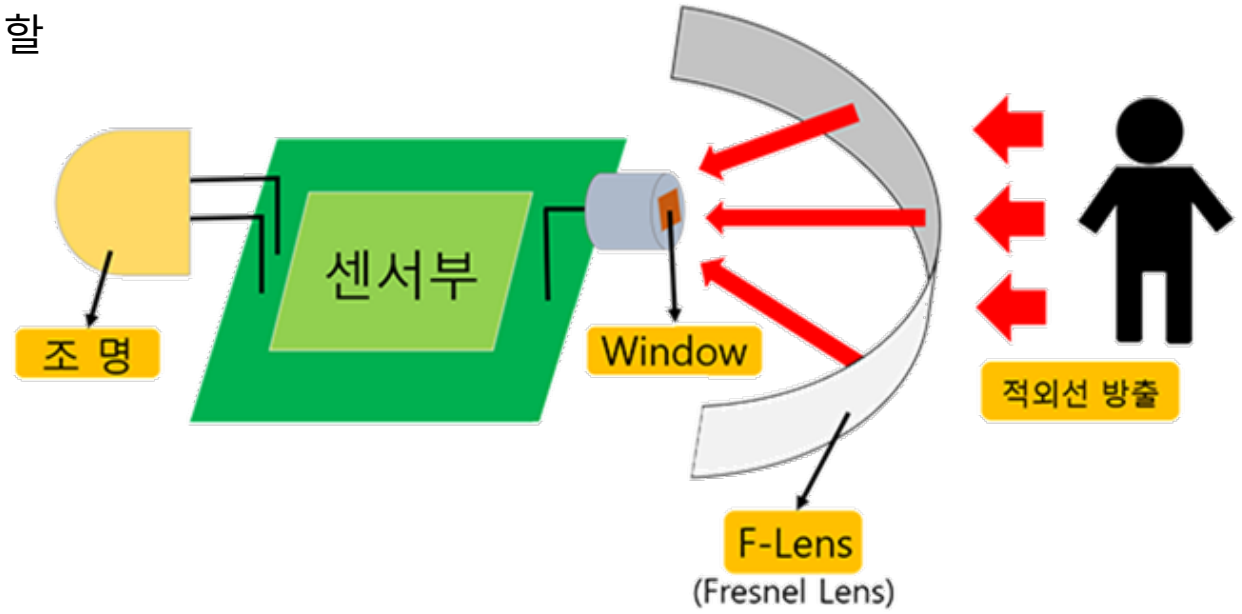


부록. 센서 활용 III

PIR (적외선 모션 감지) 센서 (HC-SR501) 동작원리

- Fresnel Lens (집광렌즈)
  - 인체로부터 발생된 적외선을 통과시키고 센서표면부에 위치한 Window에 적외선을 모아주는 역할
  - 감지거리 확대, 감도 극대화
  - 외부환경 요소로부터 센서를 보호

- Window (편광필터)
  - 표면부의 직사각형 형태의 편광 필터
  - 일정한 주파수 대역의 적외선만 통과시키는 역할

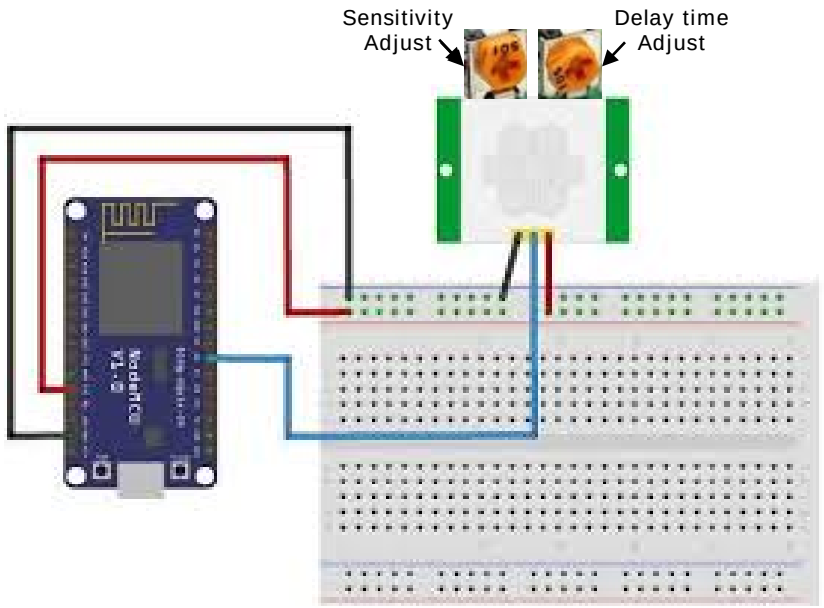


부록. 센서 활용 III

PIR (적외선 모션 감지) 센서 (HC-SR501)

- 회로연결
  - NodeMCU D2 ←→ HC-SR501 Output
  - NodeMCU Vcc ←→ HC-SR501 Vcc
  - NodeMCU GND ←→ HC-SR501 GND

  - 코드 업로딩 후 센서 감도와 지연시간 조절



예제. NodeMCU_HC_SR501_PIR_Motion.ino

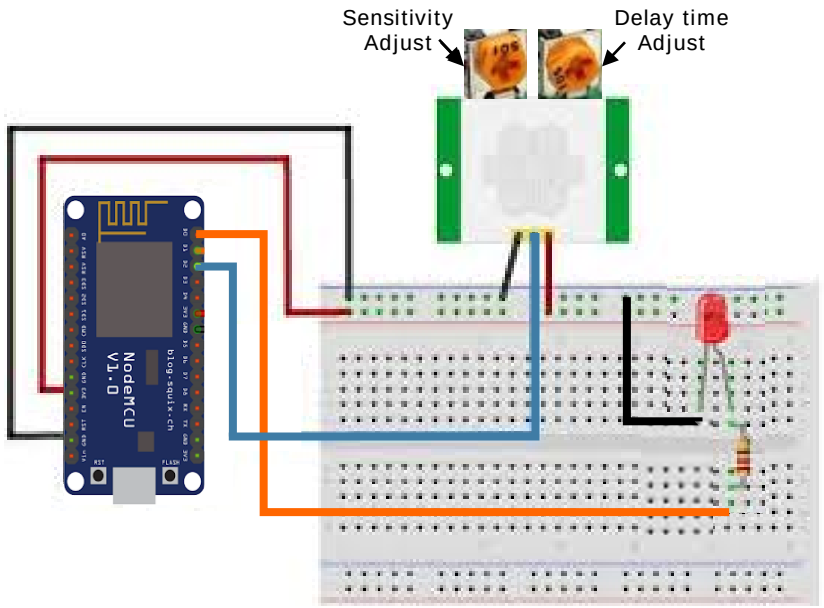
```
int inputPin = 4;  //GPIO4, 센서 시그널 D2에 연결
int ledPin = 16;    //LED D0에 연결


void setup() {
  pinMode(inputPin, INPUT);
  Serial.begin(9600);
}


void loop(){

 delay(500);

 int valueRead = 0;
 valueRead = digitalRead(inputPin);

 Serial.println(valueRead);

 if (valueRead == HIGH) {  // 인체 감지시
    Serial.println("Motion detected!");
  }else{   // 평상시
    Serial.println("Normal state...");
  }
}
```

부록. 센서 활용 III

PIR (적외선 모션 감지) 센서 (HC-SR501)

- 회로연결
  - NodeMCU D2 ←→ HC-SR501 Output
  - NodeMCU Vcc ←→ HC-SR501 Vcc
  - NodeMCU GND ←→ HC-SR501 GND

  - NodeMCU D0 ←→ 저항(220) ←→ LED (+)
  - NodeMCU GND ←→ LED (-)

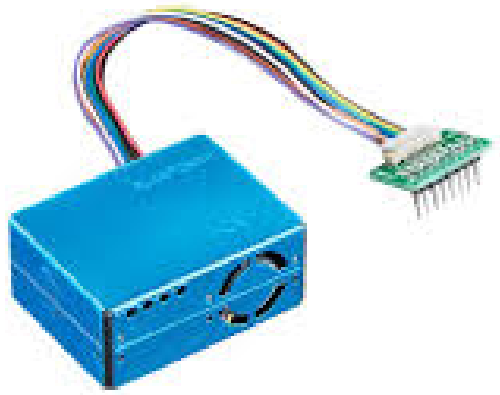


예제. NodeMCU_HC_SR501_PIR_Motion_Led.ino

```
int inputPin = 4;  //GPIO4, 센서 시그널 D2에 연결
int ledPin = 16;    //LED D0에 연결


void setup() {
  pinMode(inputPin, INPUT);
  pinMode(ledPin, OUTPUT);
  Serial.begin(9600);
}

void loop(){
 delay(500);
 int valueRead = 0;
 valueRead = digitalRead(inputPin);

 Serial.println(valueRead);

 if (valueRead == HIGH) {  // 인체 감지시
    digitalWrite(ledPin, HIGH); // LED ON
    Serial.println("Motion detected!");
  }else{
    digitalWrite(ledPin, LOW); // LED OFF
    Serial.println("Normal state!");
  }
}
```

센서 활용 III

미세먼지 감지 센서 (PMS5003)



부록. 센서 활용 III

미세먼지

- 미세먼지란?
  - 눈에 보이지 않을 정도로 입자가 작은 먼지
  - 지름이 10μm 보다 작은 미세먼지 (PM_{10})
  - 지름이 2.5μm 보다 작은 초미세먼지 ($PM_{2.5}$)
  - 지름이 1.0μm 보다 작은 극초미세먼지 ($PM_{1.0}$)

- 미세먼지를 이루는 성분
  - 대기오염물질이 공기중에서 반응하여 형성된 덩어리
  - 석탄, 석유 등 화학연료를 태우는 과정에서 발생하는 탄소류
  - 검댕, 지표면 흙먼지 등에서 생기는 광물 등
  - 발생한 지역이나 계절, 기상 조건에 따라 달라짐

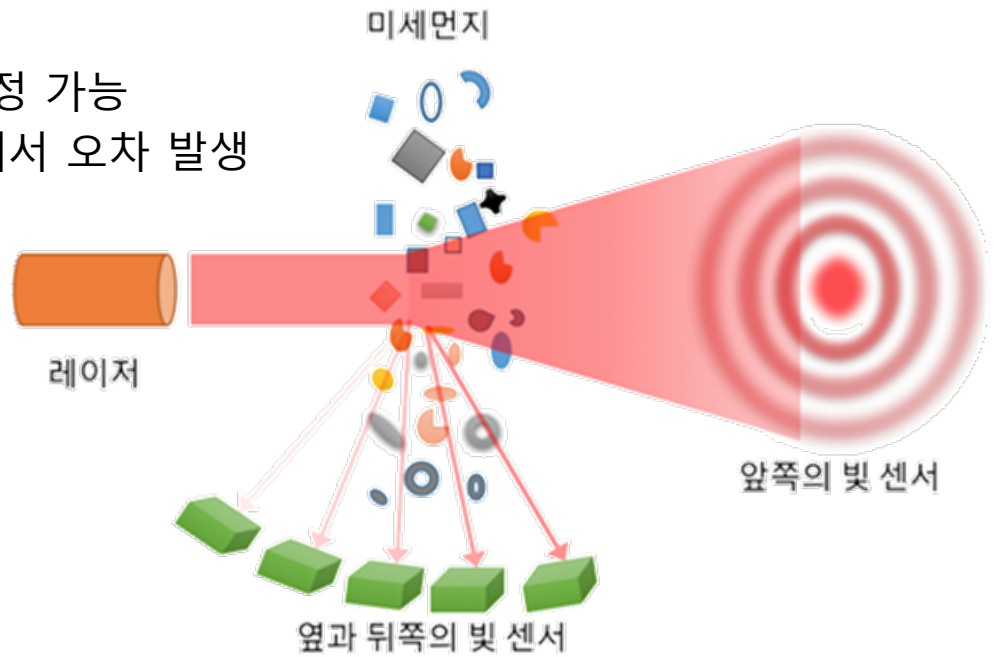
※ 출처 : 미국 환경보호청(EPA)

부록. 센서 활용 III

미세먼지 측정방식

- 광산란 방식
  - 구멍을 통해 외부로부터 유입된 공기에 빛을 쏜 후, 미세먼지에 의해 산란된 빛의 양을 수광 소자에서 검출
  - 빛을 입자에 비추면 산란, 굴절, 반사, 흡수되는 원리를 이용
  - 입자가 작으면 빛이 많이 산란되고, 입자가 크면 빛이 앞에 집중됨
  - 광원 : LED 또는 레이저 등
  - 수광소자 : 광다이오드(Photodiode) 또는 광트랜지스터(Phototransister)

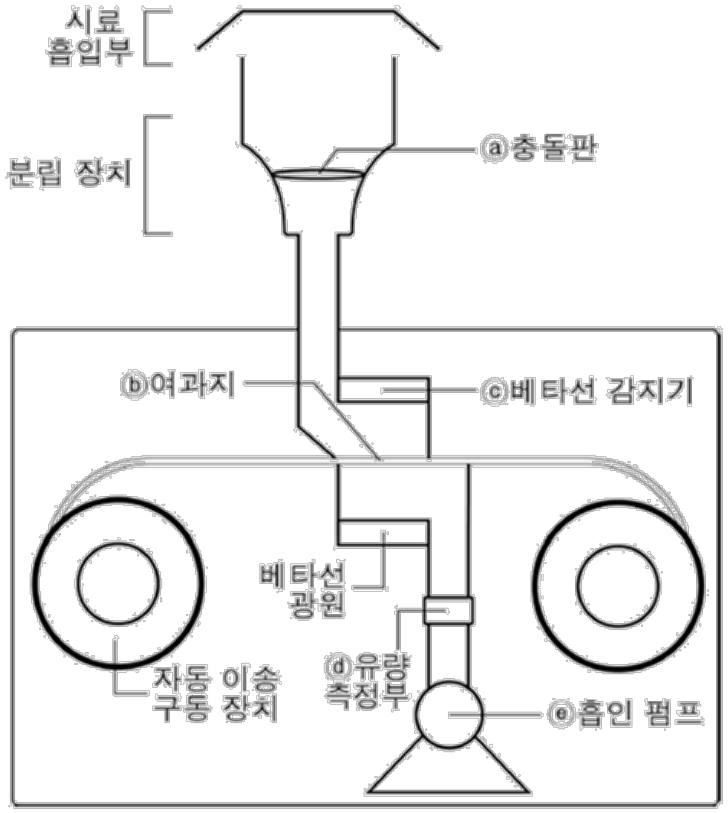
  - 장점 : 실시간 측정 가능, 휴대성, 한 개의 장치로 입자 크기별 측정 가능
  - 단점 : 입자의 개수농도를 측정하여 질량 농도로 전환하는 과정에서 오차 발생
  - 간이 측정기에 많이 사용



부록. 센서 활용 III

미세먼지 측정방식

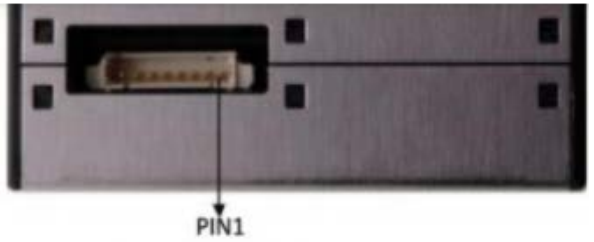
- 베타선 방식
  - 미세먼지를 채취한 여과지(필터)에 흡수되는 베타선의 양으로 미세먼지 농도를 자동 측정
  - 방사선인 베타선이 어떤 물질을 통과할 때 그 물질의 질량이 클수록 더 많이 흡수되는 성질을 이용
  - 1시간 동안 측정한 평균을 알려줌
  - 정부의 초미세먼지 측정기



부록. 센서 활용 III

미세먼지 감지 센서 (PMS5003)

- 광산란 방식 미세먼지 감지 센서 (레이저사용)
- 공기중의 입자를 감지하여 UART로 데이터 출력
- 구동전압 : 5V
- 감지범위 : 0.3 ~ 10μm (PM2.5/PM10)
- 유효 범위 : 0.3μm는 50%, 0.5μm 이상은 98%
- 내부에 소형 팬 부착

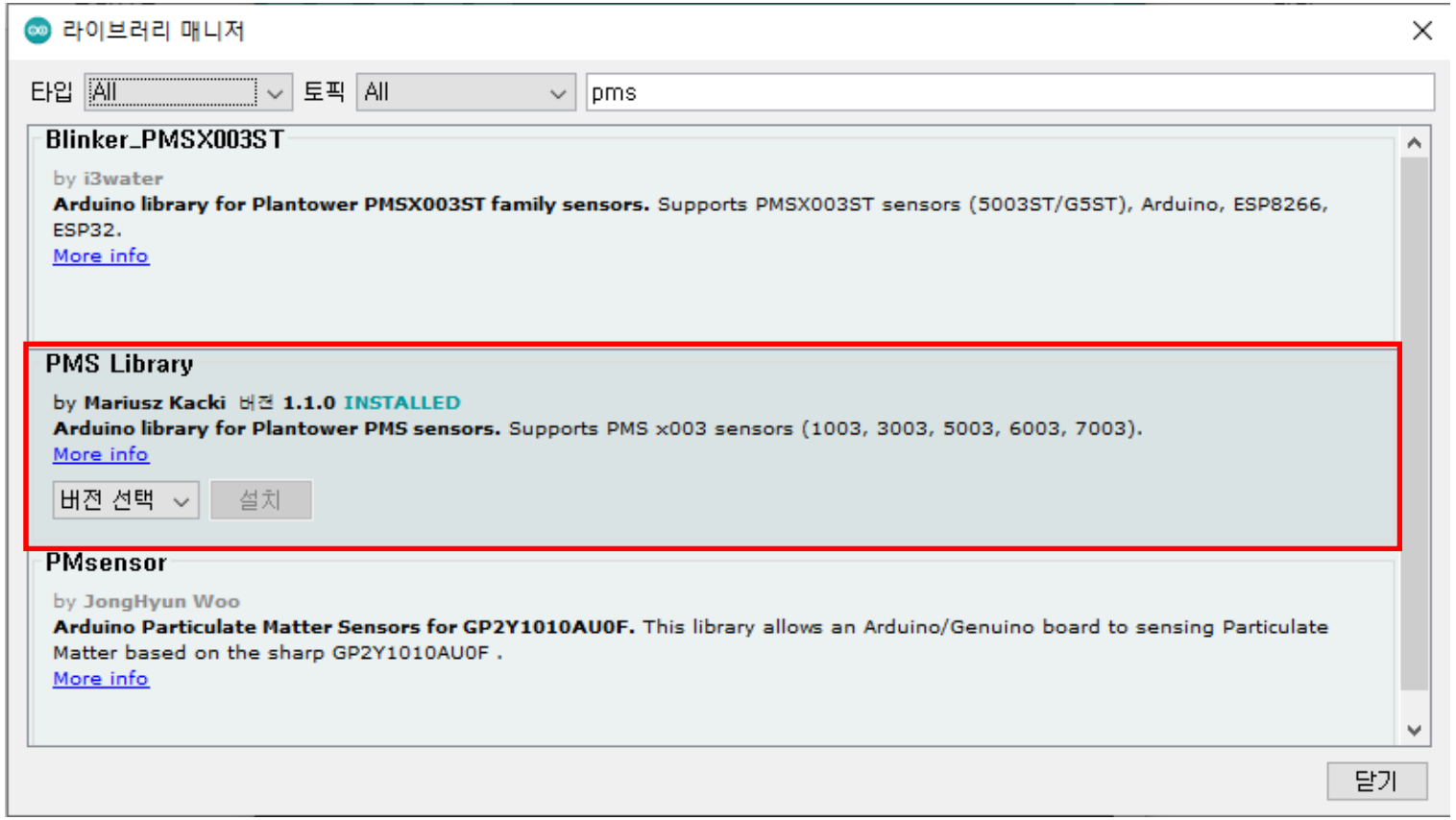


PIN1	VCC	Positive power 5V
PIN2	GND	Negative power
PIN3	SET	Set pin /TTL level@3.3V，high level or suspending is normal working status, while low level is sleeping mode.
PIN4	RX	Serial port receiving pin/TTL level@3.3V
PIN5	TX	Serial port sending pin/TTL level@3.3V
PIN6	RESET	Module reset signal /TTL level@3.3V，low reset.
PIN7/8	NC	

부록. 센서 활용 III

미세먼지 감지 센서 (PMS5003)

- 라이브러리 설치
  - 아두이노 IDE의 메뉴에서, [스케치/라이브러리 포함하기/라이브러리 관리...] 실행
  - 라이브러리 매니저 창에서 PMS 검색
  - PMS Library 설치



부록. 센서 활용 III

미세먼지 감지 센서 (PMS5003)

- 회로연결
  - PMS5003 Vcc ←→ NodeMCU Vin
  - PMS5003 GND ←→ NodeMCU GND
  - PMS5003 Pin5 Tx ←→ Rx (GPIO3)

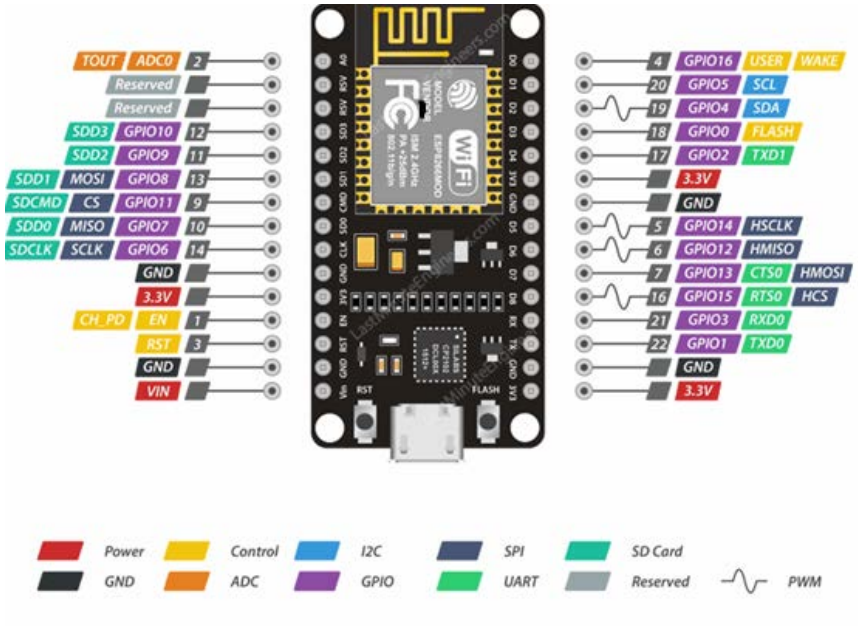
예제. NodeMCU_PMS_dust.ino

```
#include "PMS.h"

PMS pms(Serial);
PMS::DATA data;

void setup()
{
  Serial.begin(9600);   // GPIO1, GPIO3 (TX/RX pin on ESP-12E Development Board)
}

void loop()
{
  if (pms.read(data))
  {
    Serial.print("PM 2.5 (ug/m3): ");
    Serial.println(data.PM_AE_UG_2_5);

    Serial.println();
  }
}
```

부록. 센서 활용 III

미세먼지 감지 센서 (PMS5003)

- 회로연결
  - PMS5003 Vcc ←→ NodeMCU Vin
  - PMS5003 GND ←→ NodeMCU GND
  - PMS5003 Pin5 Tx ←→ NodeMCU D8
  - PMS5003 Pin4 Rx ←→ NodeMCU D7



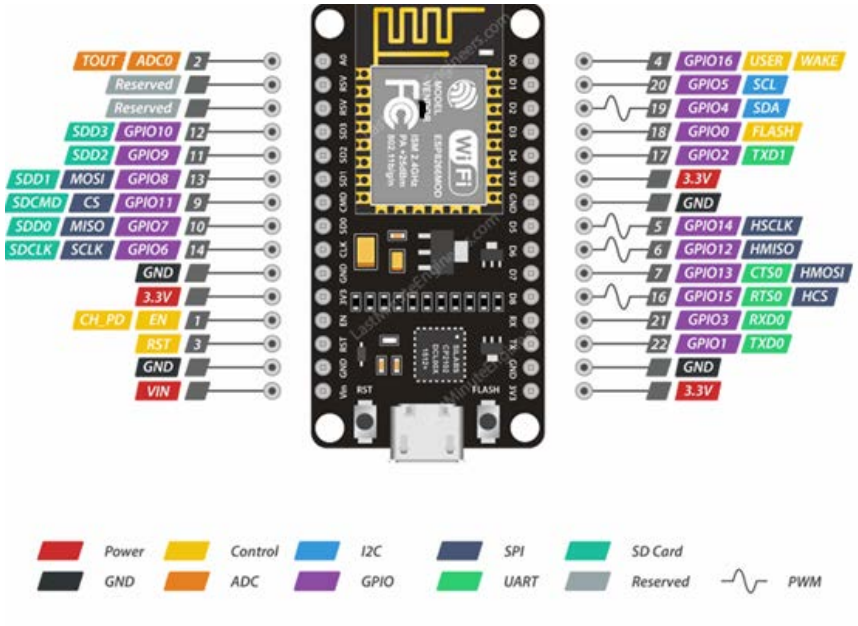
예제. NodeMCU_PMS_dust_02.ino

```
#include <SoftwareSerial.h>
#include <PMS.h>

SoftwareSerial PmsSerial(D8, D7);   // (UART2 RX, UART2 TX)

PMS pms(PmsSerial);
PMS::DATA data;

void setup()
{
  PmsSerial.begin(9600);
  Serial.begin(9600);
  delay(4000);
}

void loop()
{
  if (pms.read(data))
  {
    Serial.println("Dust Concentration");
    //Serial.println("PM1.0 :" + String(data.PM_AE_UG_1_0) + "(ug/m3)");
    Serial.println("PM2.5 :" + String(data.PM_AE_UG_2_5) + "(ug/m3)");
    Serial.println("PM10 :" + String(data.PM_AE_UG_10_0) + "(ug/m3)");
    Serial.println("₩n");
    delay(1000);
  }
}
```

부록. 센서 활용 III

미세먼지 감지 센서 (PMS5003)

- 회로연결
  - PMS5003 Vcc ←→ NodeMCU Vin
  - PMS5003 GND ←→ NodeMCU GND
  - PMS5003 Pin5 Tx ←→ NodeMCU D8
  - PMS5003 Pin4 Rx ←→ NodeMCU D7



예제. NodeMCU_PMS_dust_03.ino (1/2)

```
#include <SoftwareSerial.h>
#include <PMS.h>

SoftwareSerial PmsSerial(D8, D7); // (UART2 RX, UART2 TX)

PMS pms(PmsSerial);
PMS::DATA data;

void setup()
{
  PmsSerial.begin(9600);
  Serial.begin(9600);
  delay(4000);
}

void loop()
{
  if (pms.read(data))
  {
    Serial.println("Dust Concentration");
    //Serial.println("PM1.0 :" + String(data.PM_AE_UG_1_0) + "(ug/m3)");
    Serial.println("PM2.5 :" + String(data.PM_AE_UG_2_5) + "(ug/m3)");
    Serial.println("PM10 :" + String(data.PM_AE_UG_10_0) + "(ug/m3)");


//.....
//.....
```

부록. 센서 활용 III

미세먼지 감지 센서 (PMS5003)

- 회로연결
  - PMS5003 Vcc ←→ NodeMCU Vin
  - PMS5003 GND ←→ NodeMCU GND
  - PMS5003 Pin5 Tx ←→ NodeMCU D8
  - PMS5003 Pin4 Rx ←→ NodeMCU D7

- 환경부(에어코리아) 미세먼지 예보등급 (μm/m^3)

단계	미세먼지	초/극초 미세먼지
좋음	0 ~ 30	0 ~ 15
보통	31 ~ 80	16 ~ 50
나쁨	81 ~ 150	51 ~ 100
매우나쁨	151 이상	101 이상

예제. NodeMCU_PMS_dust_03.ino (2/2)

```
//.....
//.....

    if(0 <= data.PM_AE_UG_10_0 && data.PM_AE_UG_10_0 <= 30){
      Serial.print("미세먼지 좋음, ");
    }else if(31<=data.PM_AE_UG_10_0 && data.PM_AE_UG_10_0 <= 80){
      Serial.print("미세먼지 보통, ");
    }else if(81<=data.PM_AE_UG_10_0 && data.PM_AE_UG_10_0 <= 150){
      Serial.print("미세먼지 나쁨, ");
    }else if(151<=data.PM_AE_UG_10_0){
      Serial.print("미세먼지 매우 나쁨, ");
    }

    if(0 <= data.PM_AE_UG_2_5 && data.PM_AE_UG_2_5 <= 15){
      Serial.println("초미세먼지 좋음");
    }else if(16<=data.PM_AE_UG_2_5 && data.PM_AE_UG_2_5 <= 50){
      Serial.println("초미세먼지 보통");
    }else if(51<=data.PM_AE_UG_2_5 && data.PM_AE_UG_2_5 <= 100){
      Serial.println("초미세먼지 나쁨");
    }else if(101<=data.PM_AE_UG_2_5){
      Serial.println("초미세먼지 매우 나쁨");
    }

    Serial.println("₩n");
    delay(1000);
  }
}//end of loop()
```