무선통신

Week 11
2023-05-11

Handong Global University
Smart Sensors and IoT Devices

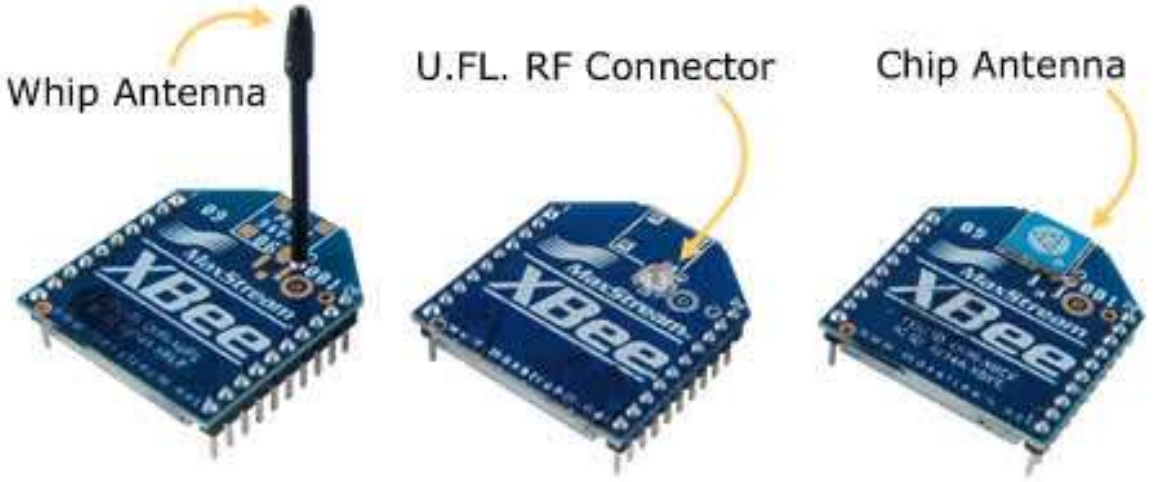
Arduino Xbee

1. Arduino Xbee

- **Xbee**
  - ➢ 근거리 무선 통신 장치
  - ➢ 비교적 넓은 범위를 저전력으로 통신 가능
  - ➢ 다양한 안테나 타입의 모듈 존재
  - ➢ Xbee 모듈 통신 위해 Xbee 장치의 펌웨어 설정

이름	Wi-Fi	Xbee
응용 분야	웹, 이메일, 비디오	모니터링 & 제어
시스템 자원	1MB+	25KB - 50KB
배터리 수명 (일)	.5 - 5	100 - 1,000+
최대 채널 수	**14**	32,000
통신속도 (Kb/s)	11,000+	20 - 250
전송범위 (meters)	1 - 100	1 - 100
중요 특성	속도, 유연성	신뢰성, 전력, 비용

http://www.digi.com/products/wireless-wired-embedded-solutions/zigbee-rf-modules/xctu



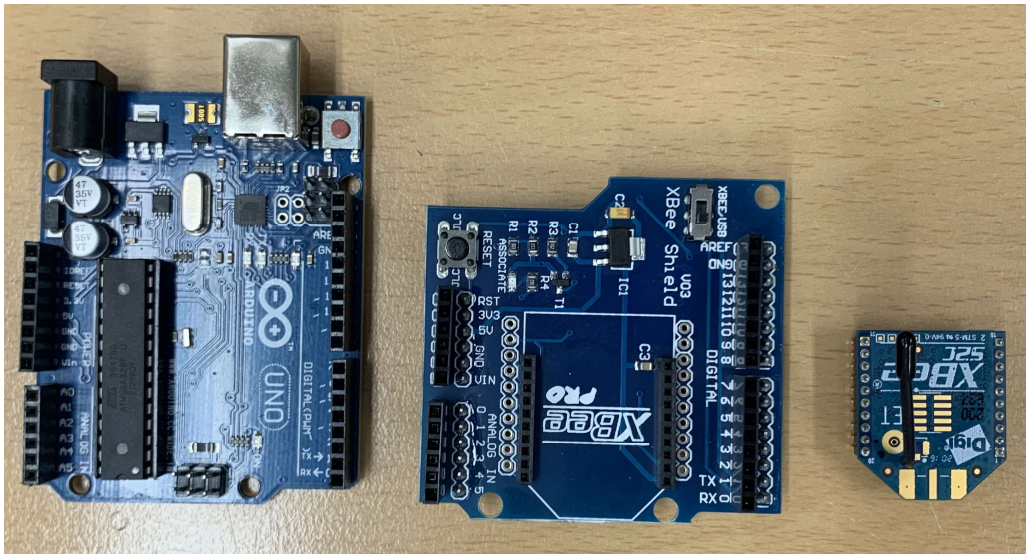
1. Arduino Xbee

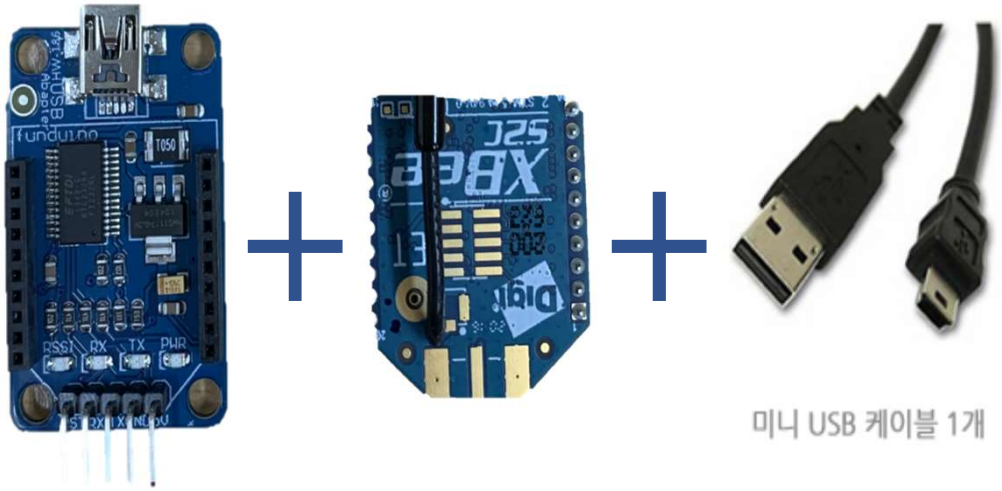
- **Xbee를 이용한 무선 통신**

  미니 USB로 연결될 아두이노 Xbee 모듈 연결

  아두이노 장치에 Xbee 장치가 부착된 Xbee 쉴드를 연결

  *** Xbee와 아두이노가 직접 연결되어 통신하기 때문에 PC에서 시리얼 모니터 사용 불가**





1. Arduino Xbee

- **Xbee를 이용한 무선 통신 1**

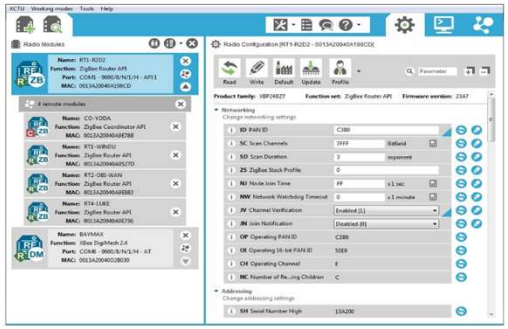
  Xbee 모듈 통신 위해 Xbee 장치의 펌웨어 설치

1 http://www.digi.com/products/wireless-wired-embedded-solutions/zigbee-rf-modules/xctu

Next Generation Configuration Platform for XBee/RF Solutions

- XCTU is a **free, multi-platform** application compatible with Windows, MacOS and Linux
- **Graphical Network View** for simple wireless network configuration and architecture
- **API Frame Builder** is a simple development tool for quickly building XBee API frames
- **Firmware Release Notes Viewer** allows users to explore and read firmware release notes

VISIT SUPPORT TO DOWNLOAD XCTU



Drivers & Patches

» Linux and Mac OS X Drivers (provided by FTDI)

» Drivers Installer for Windows (XP, Vista, 7 and 8)

Resources & Utilities

» XCTU v. 6.5.9 MacOS X

2 » XCTU v. 6.5.9 Windows x86:x64

» XCTU v. 6.5.9 Linux x86

» XCTU v. 6.5.9 Linux x64

» XCTU (legacy 5.2.8)

XCTU v. 6.5.9 Windows x86:x64

XCTU v. 6.5.9 Windows x86:x64

EXE (187.52 MB) - v. 6.5.9 | 2/16/2022 download release notes | license agreement |

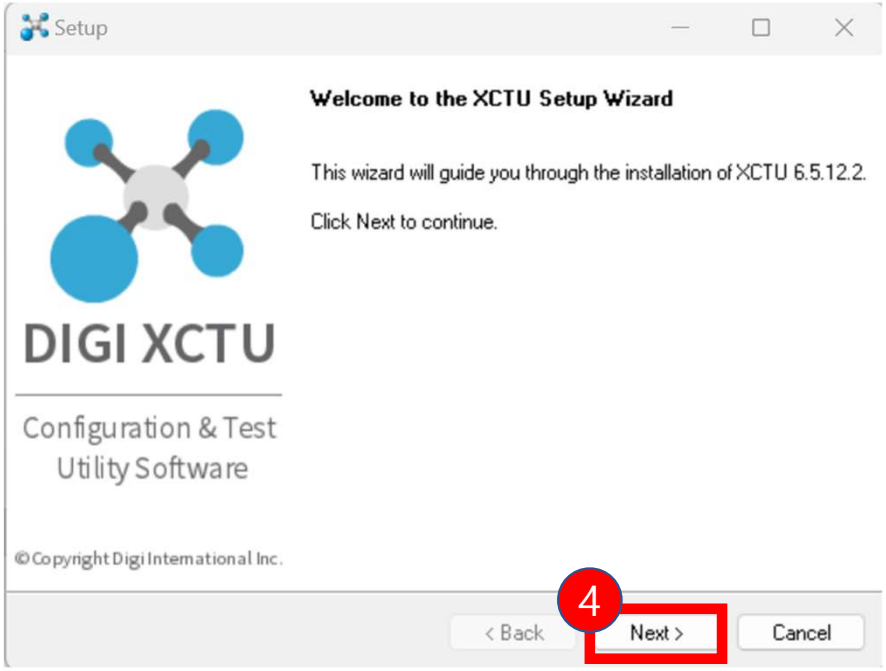
+ subscribe | previous

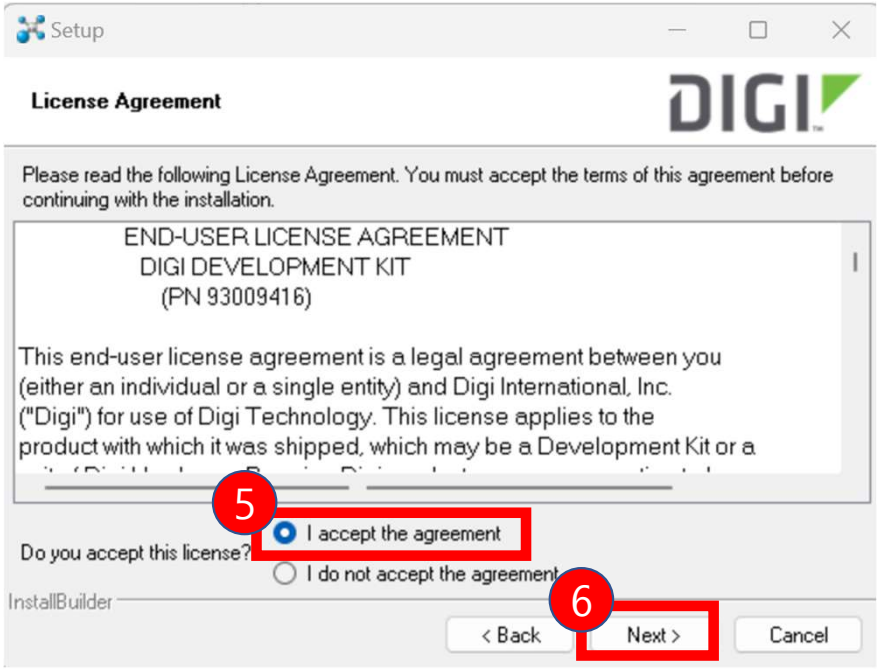
3

1. Arduino Xbee

- **Xbee를 이용한 무선 통신 1**

  Xbee 모듈 통신 위해 Xbee 장치의 펌웨어 설치

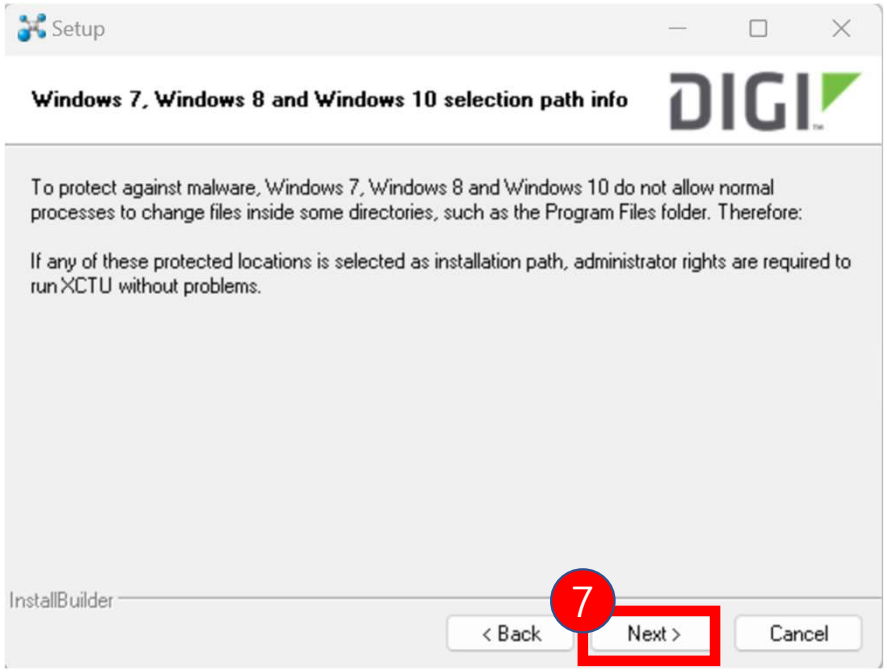




1. Arduino Xbee

- **Xbee를 이용한 무선 통신 1**

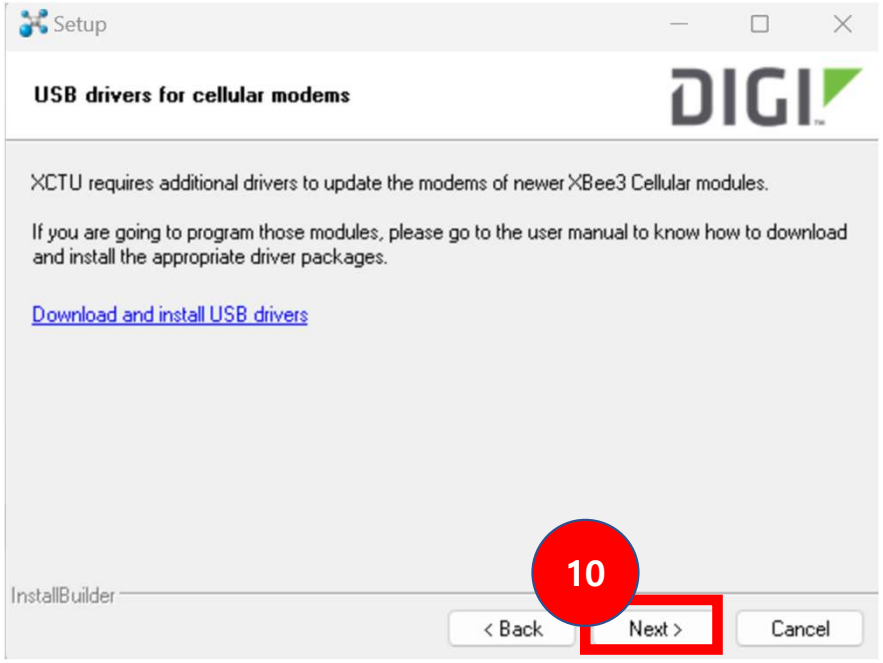
  Xbee 모듈 통신 위해 Xbee 장치의 펌웨어 설치

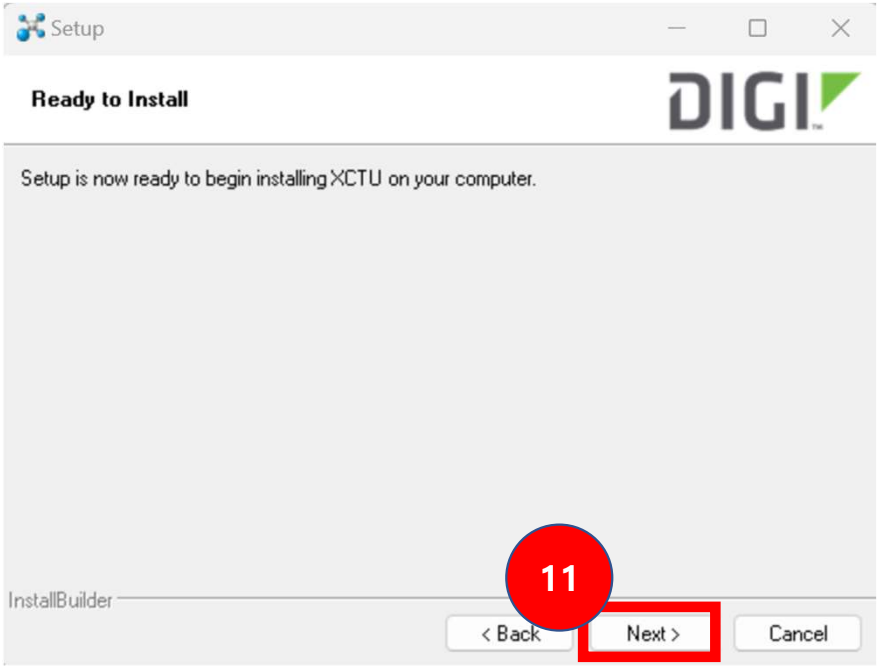


1. Arduino Xbee

- **Xbee를 이용한 무선 통신 1**

  Xbee 모듈 통신 위해 Xbee 장치의 펌웨어 설치

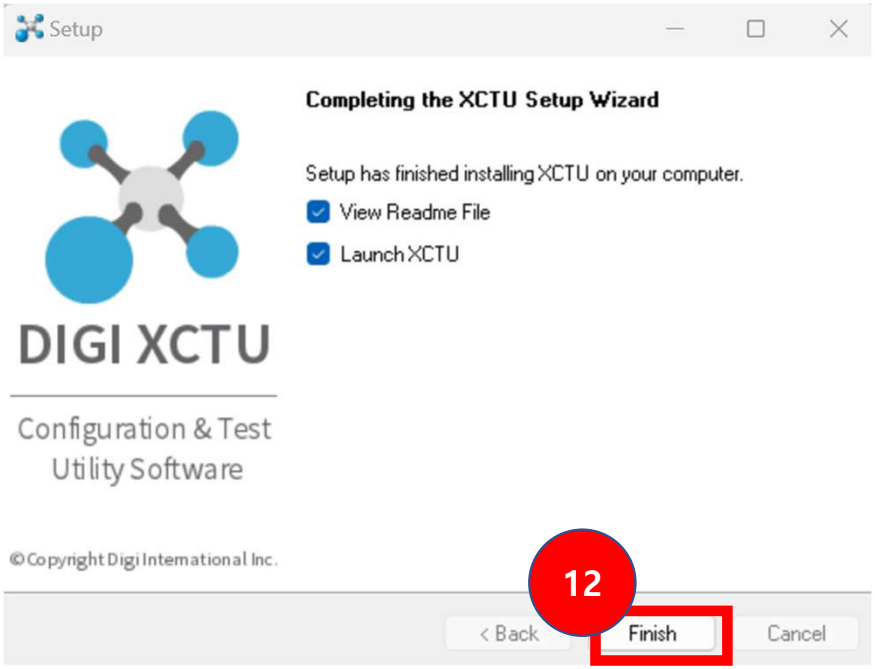


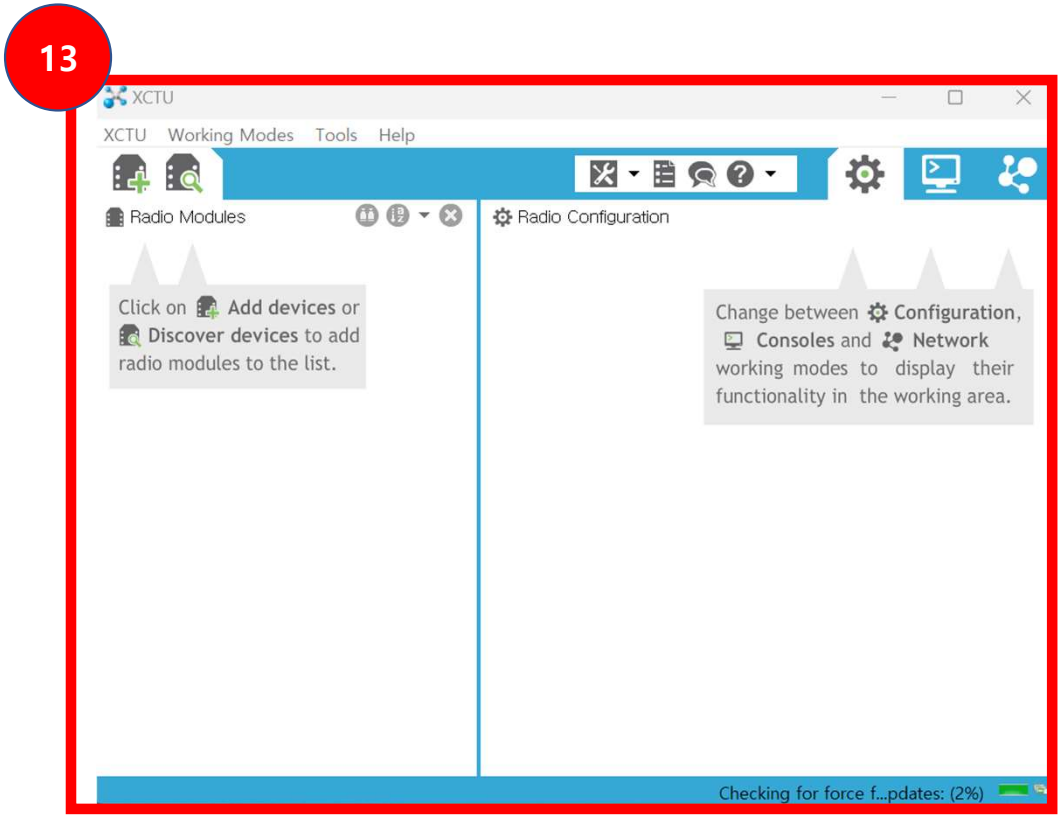


1. Arduino Xbee

- **Xbee를 이용한 무선 통신 1**

  Xbee 모듈 통신 위해 Xbee 장치의 펌웨어 설치

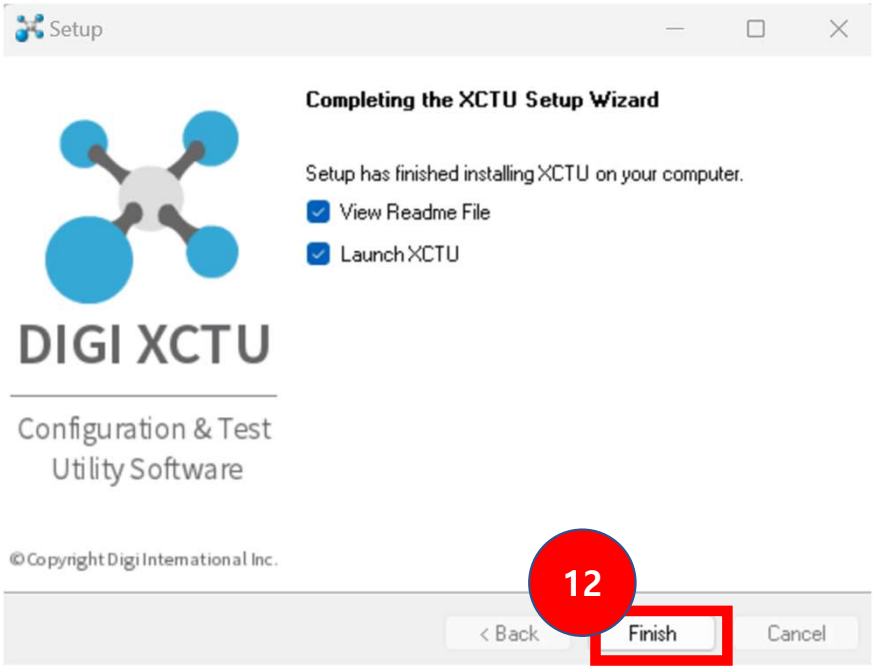


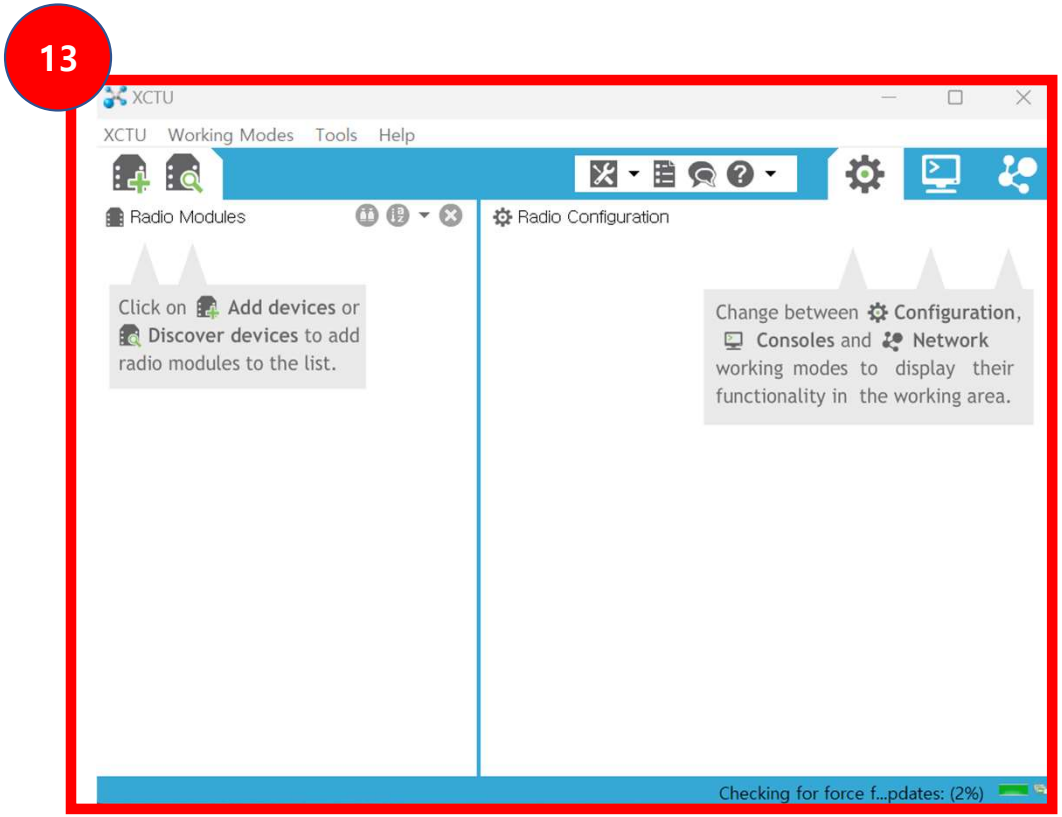


1. Arduino Xbee

- **Xbee를 이용한 무선 통신 2**

  Xbee 모듈 통신 위해 Xbee 장치의 펌웨어 설치

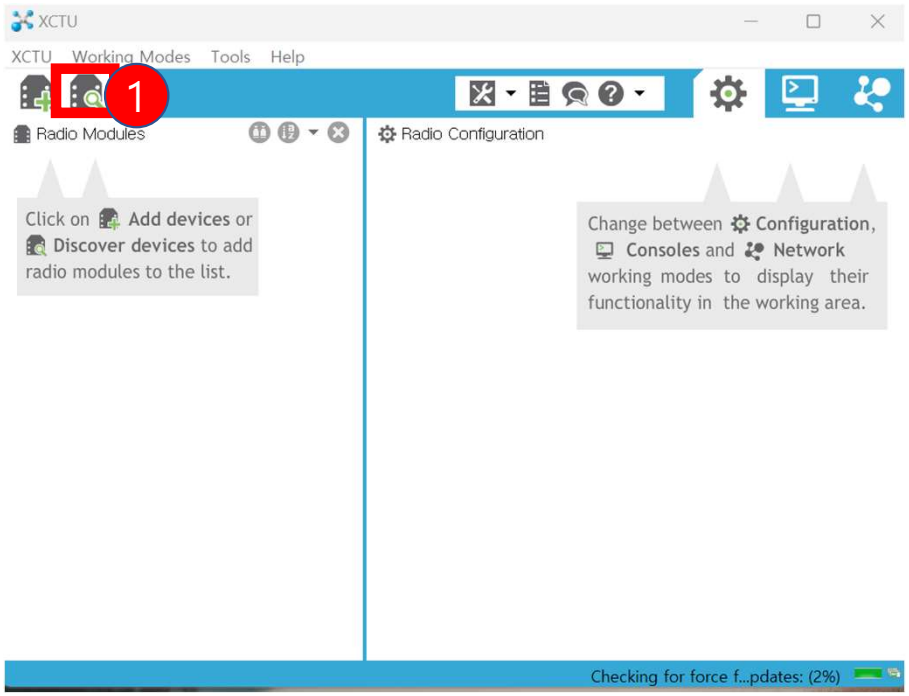


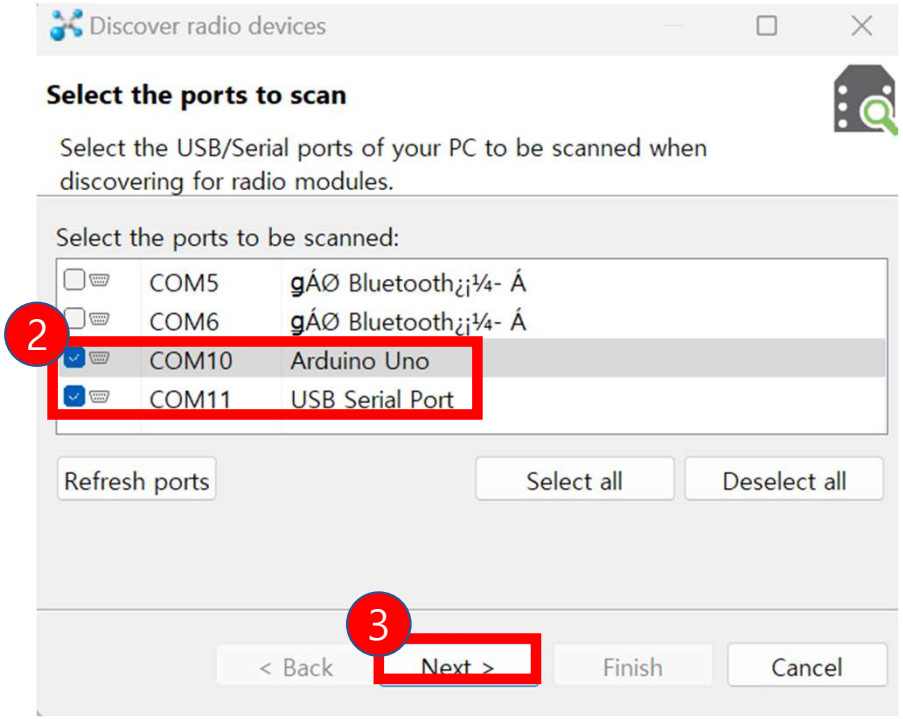


1. Arduino Xbee

- **Xbee를 이용한 무선 통신 2**

  Xbee 모듈 통신 위해 Xbee 장치의 펌웨어 설정

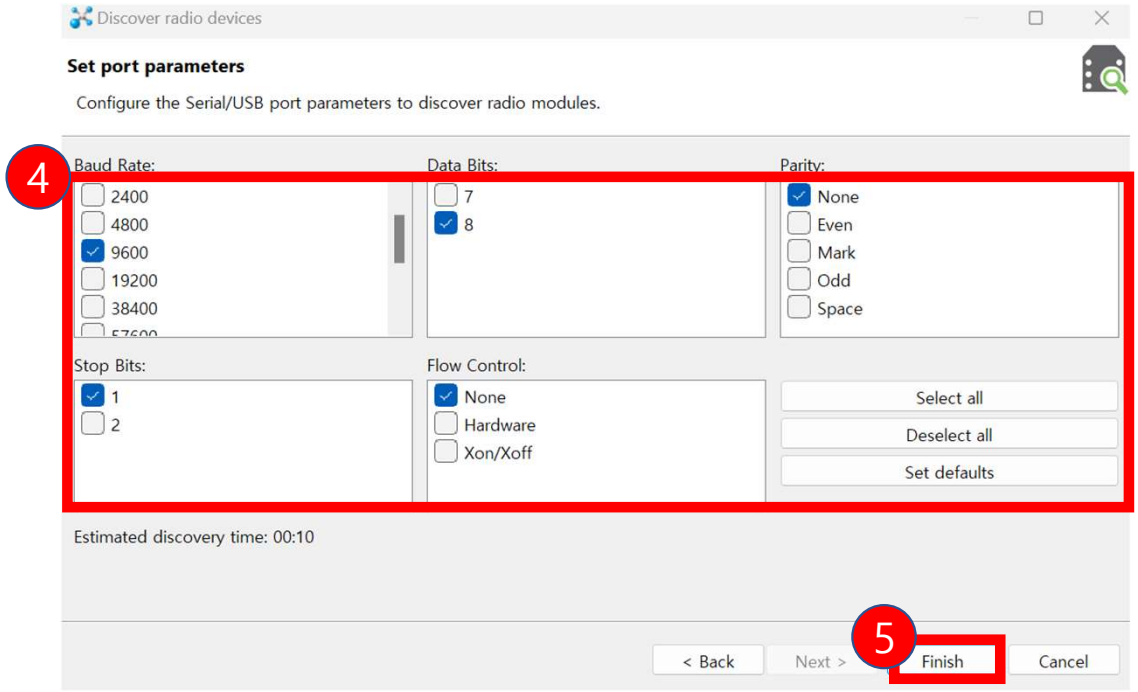


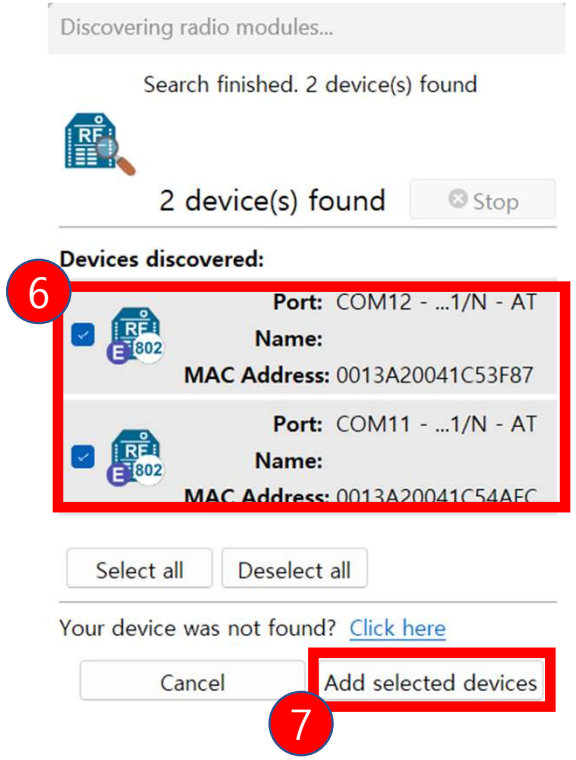


1. Arduino Xbee

- **Xbee를 이용한 무선 통신 2**

  Xbee 모듈 통신 위해 Xbee 장치의 펌웨어 설정

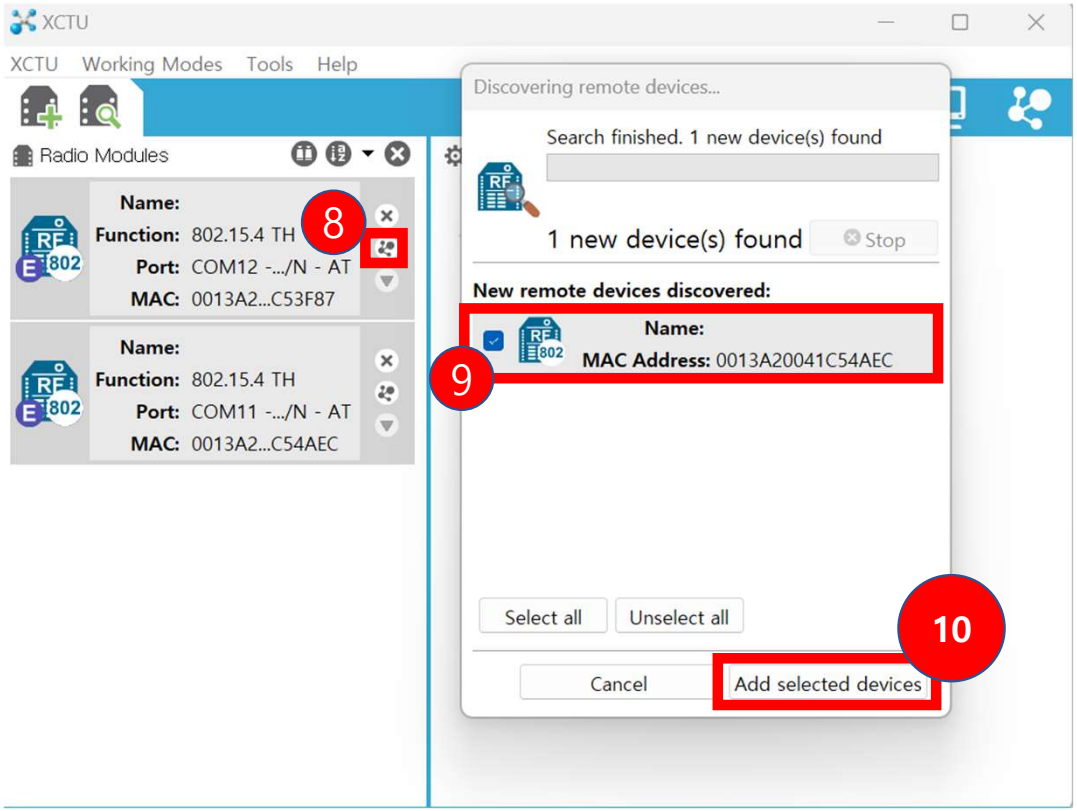




1. Arduino Xbee

- **Xbee를 이용한 무선 통신 2**

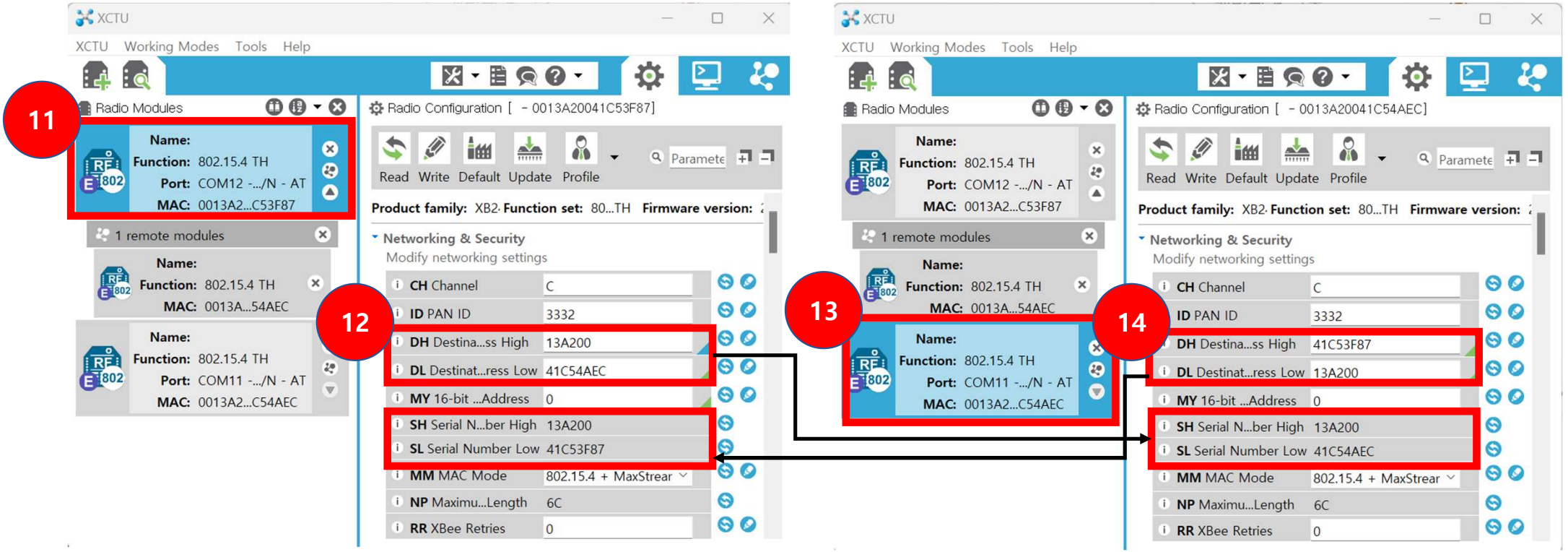
  Xbee 모듈 통신 위해 Xbee 장치의 펌웨어 설정



1. Arduino Xbee

- **Xbee를 이용한 무선 통신 2**

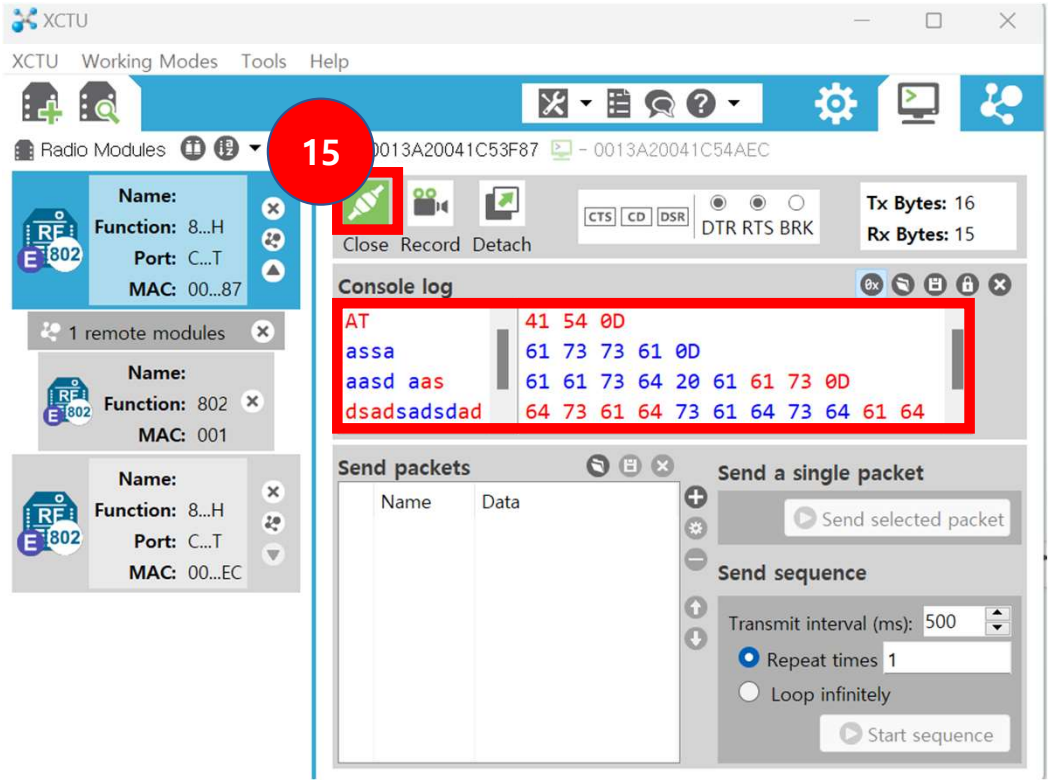
  Xbee 모듈 통신 위해 Xbee 장치의 펌웨어 설정

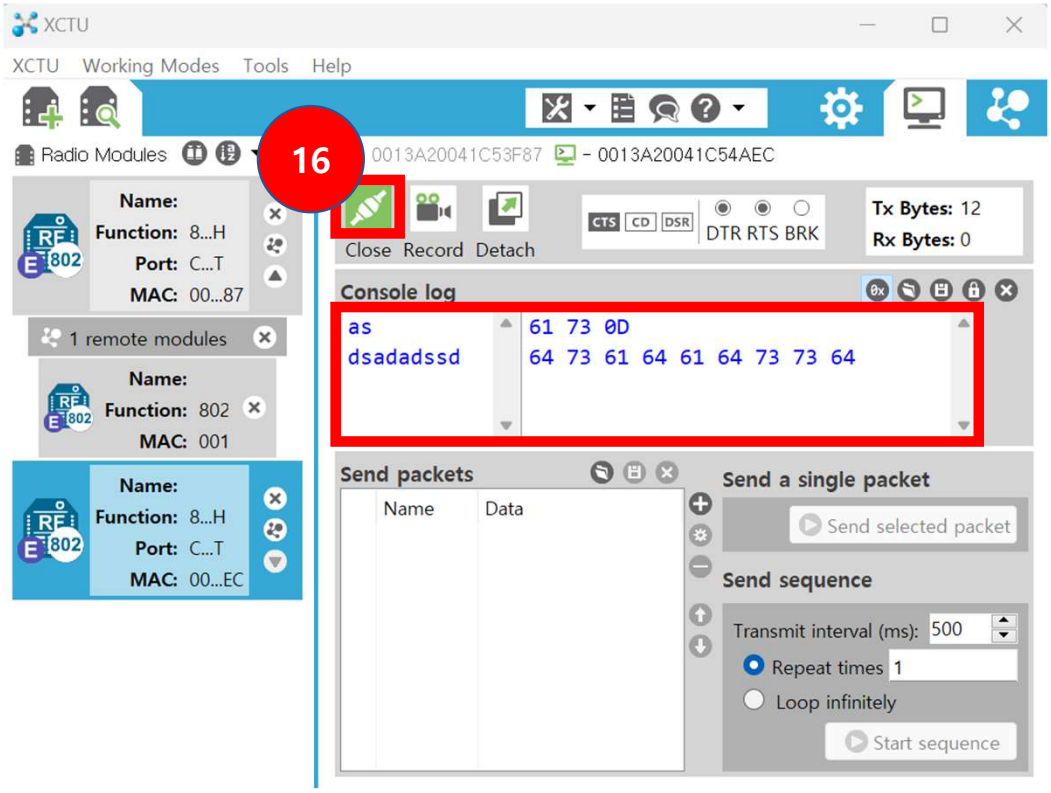


1. Arduino Xbee

- **Xbee를 이용한 무선 통신 2**

  Xbee 모듈 통신 위해 Xbee 장치의 펌웨어 설정



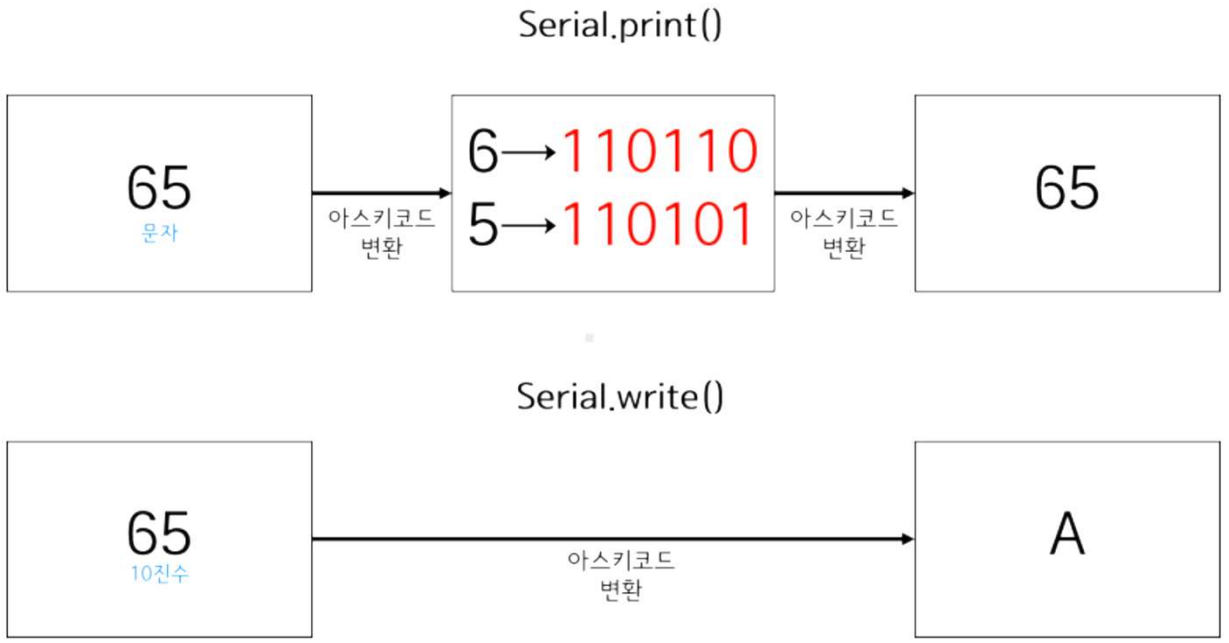


1. Arduino Xbee

- 시리얼 통신시 Serial.write 혹은 Serial.print를 이용

  Serial.print -> 문자를 하나씩 아스키 코드로 변환하여 전송

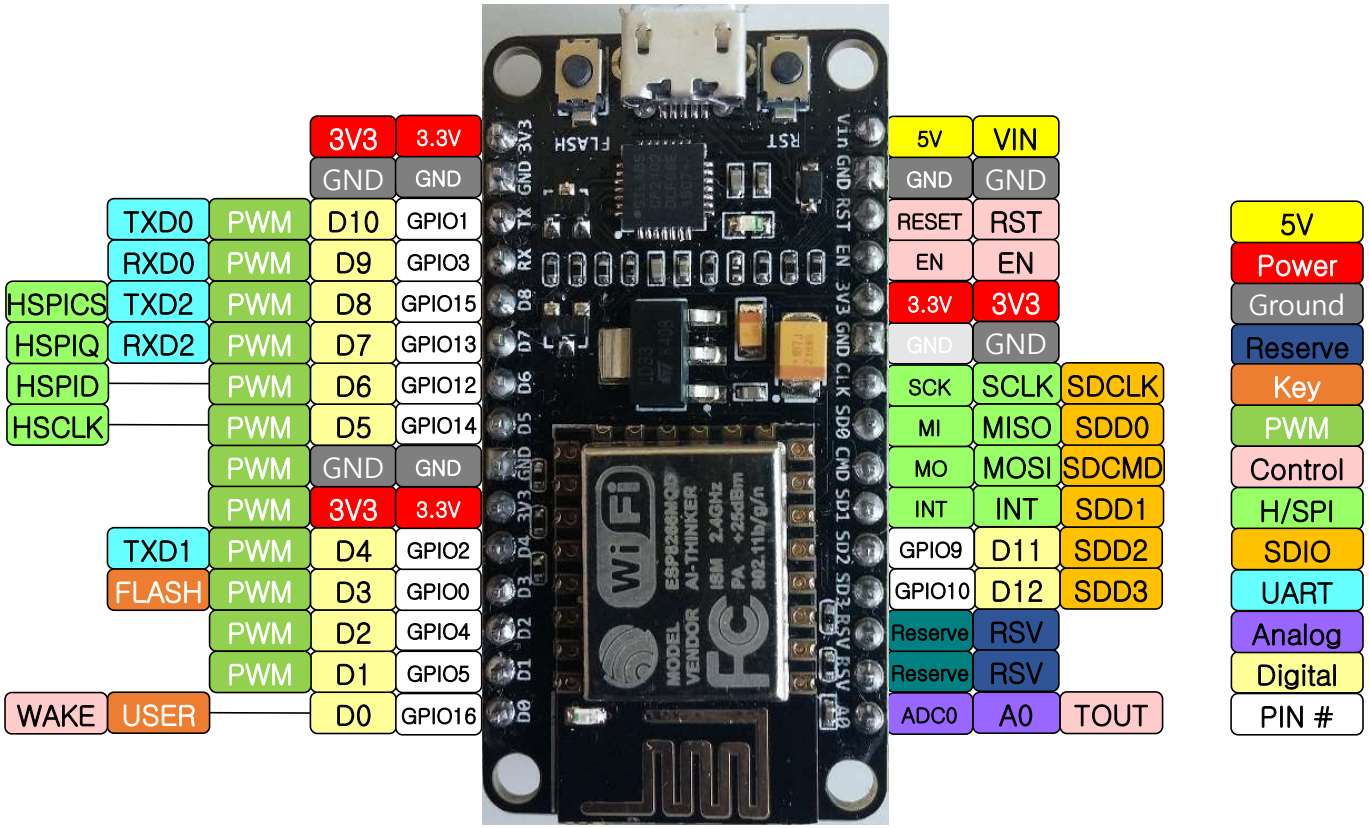
  Serial.write ->문자열 자체를 아스키코드로 변환 후 전송 (*255까지가 한계)



NodeMCU

2. NodeMCU

- **NodeMCU**
  - ➢ 와이파이 기능이 구현된 MCU 개발보드 / 오픈소스 사물인터넷 (IoT) 플랫폼의 일부
  - ➢ 작은 크기와 저렴한 가격으로 네트워크 기능 구현



2. NodeMCU

- **NodeMCU**

  Arduino IDE 내 NodeMCU 기기 설정

  1) Arduino 호환 보드에 대한 설치 지원 URLs

  Unofficial list of 3rd party boards support urls

  https://github.com/Arduino/Arduino/wiki/Unofficial-list-of-3rd-party-boards-support-urls

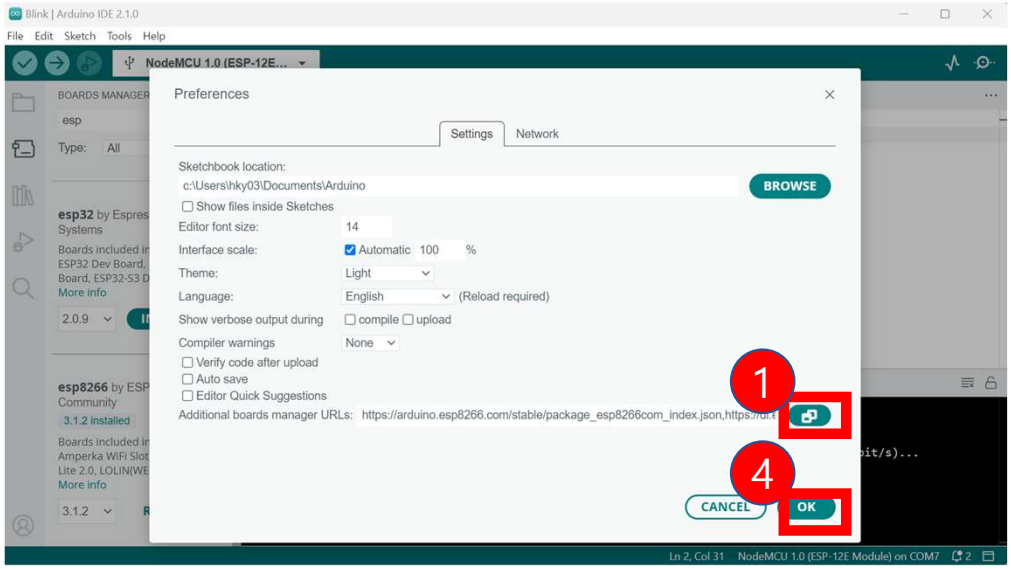
  2) Arduino IDE -> File -> Preferences -> Additional boards manager URLs

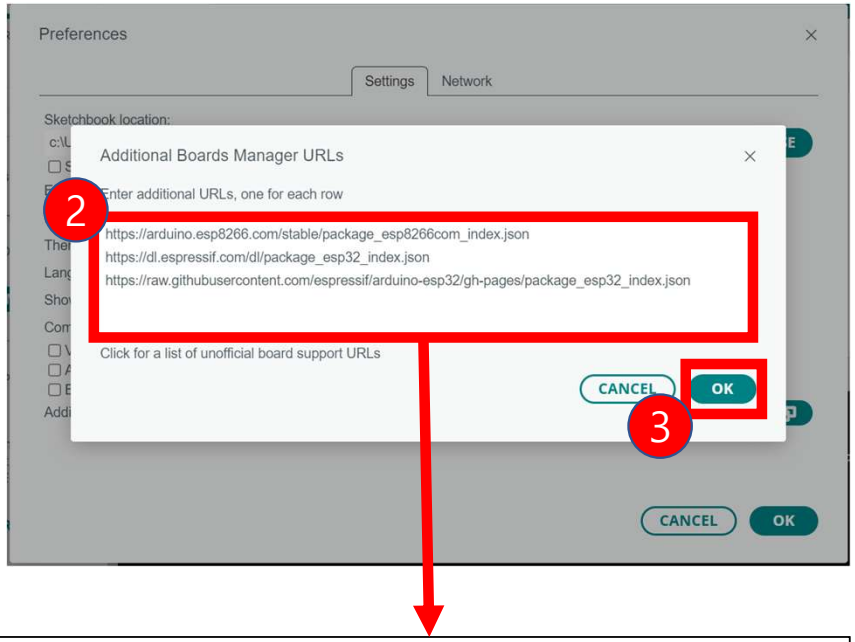
  필요한 호환 보드의 설치 지원 URL을 작성

2. NodeMCU

- **NodeMCU**

Arduino IDE 내 NodeMCU 기기 설정





https://arduino.esp8266.com/stable/package_esp8266com_index.json
https://dl.espressif.com/dl/package_esp32_index.json
https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json

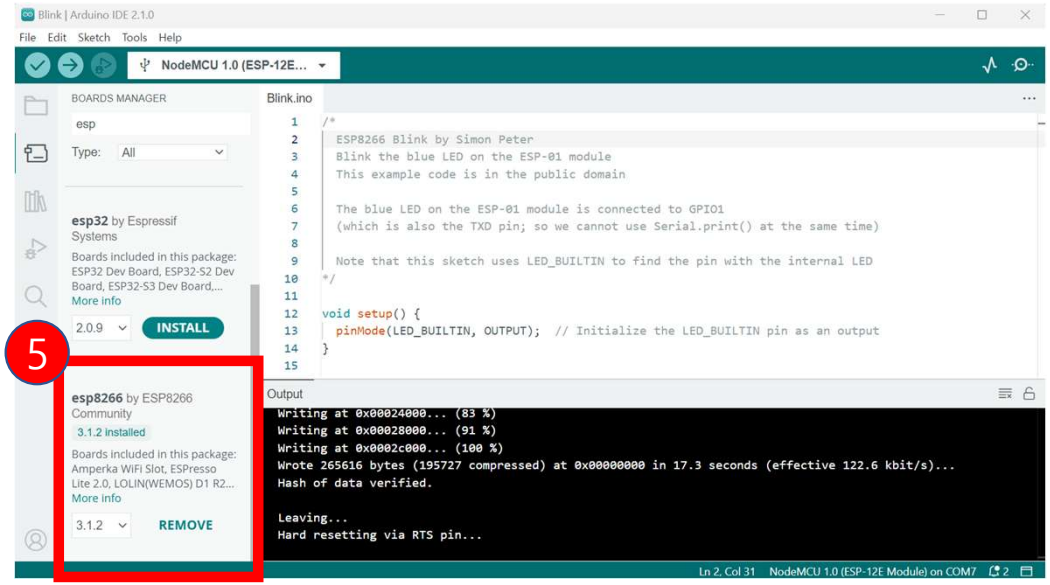
2. NodeMCU

- **NodeMCU**

Arduino IDE 내 NodeMCU 기기 설정

3) Arduino IDE -> Tools -> Board -> Boards Managers

검색해서 'esp8266 by ESP8266 Community'를 설치



2. NodeMCU

- **NodeMCU**

  Arduino IDE 내 NodeMCU 기기 설정

  4) Arduino IDE -> Tools -> Board -> esp8266 -> NodeMCU 1.0 (ESP-12E...)

  Board에서 'NodeMCU 1.0 (ESP-12E Module)' 선택

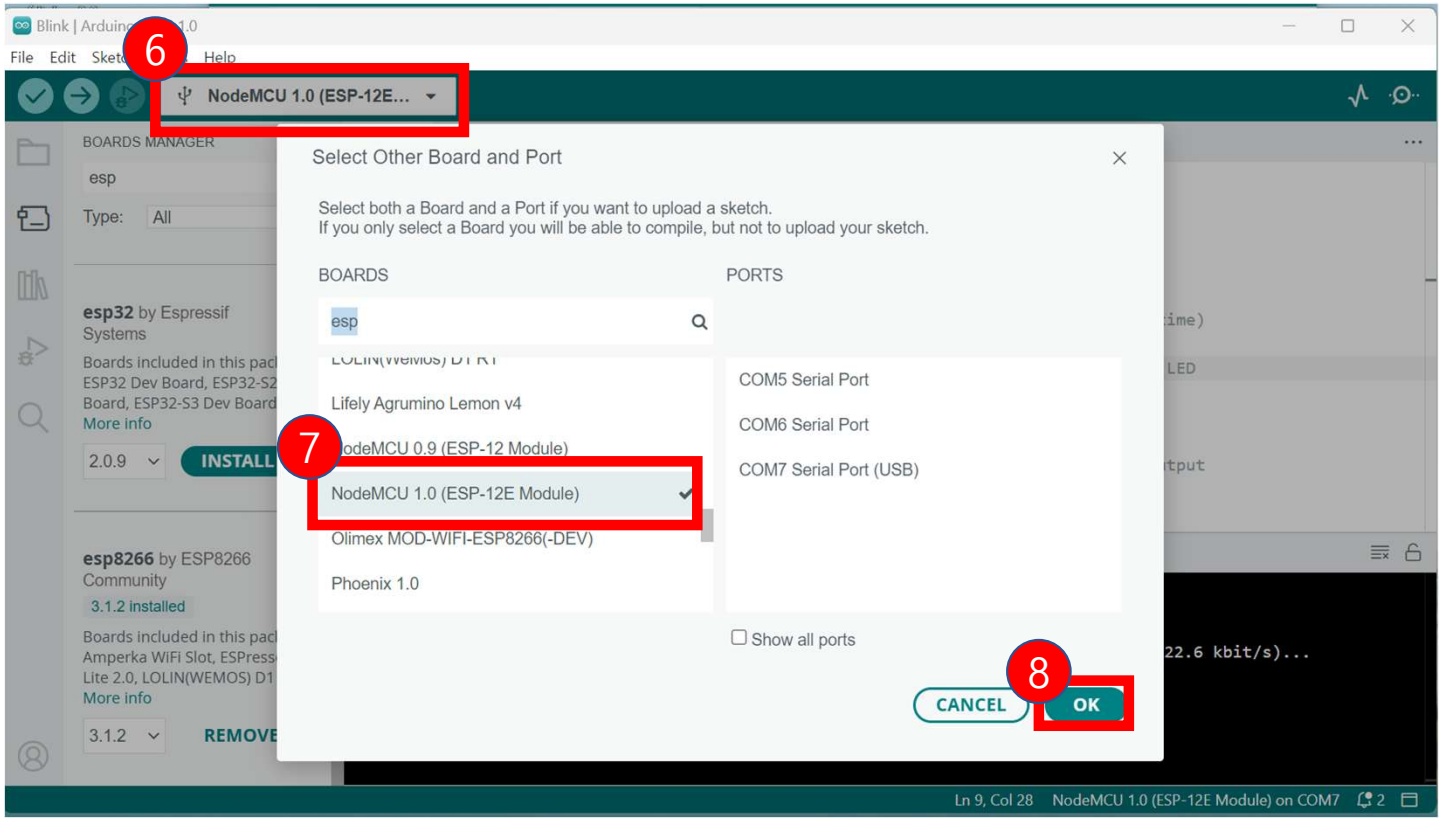
  5) Arduino IDE 2.1.0 이상의 버전의 경우

  port 선택 -> Select other board and port... -> NodeMCU 연결된 port (com) 선택 -> 해당 port(com)를 NodeMCU 1.0 (ESP-12E Module)로 연결

2. NodeMCU

- **NodeMCU**

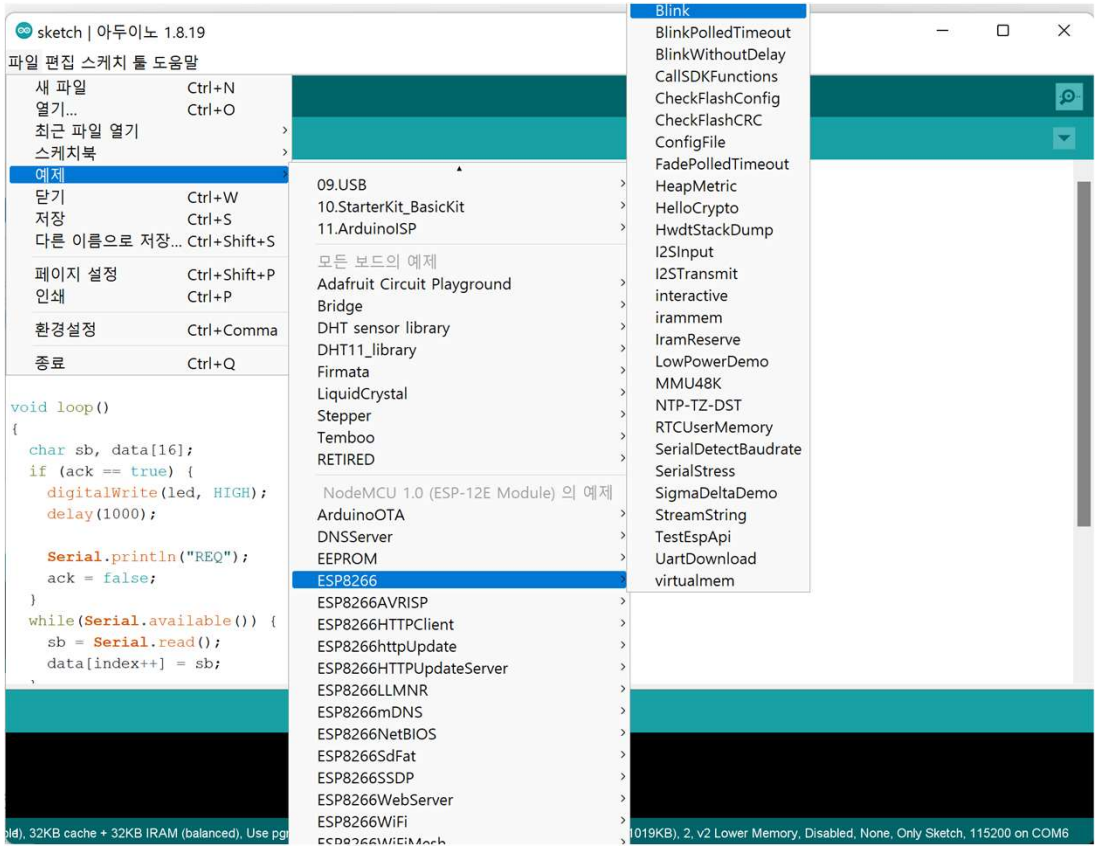
Arduino IDE 내 NodeMCU 기기 설정



2. NodeMCU

- **NodeMCU**

Arduino IDE -> File -> Example -> ESP8266 -> Blink (반짝이는지 확인)



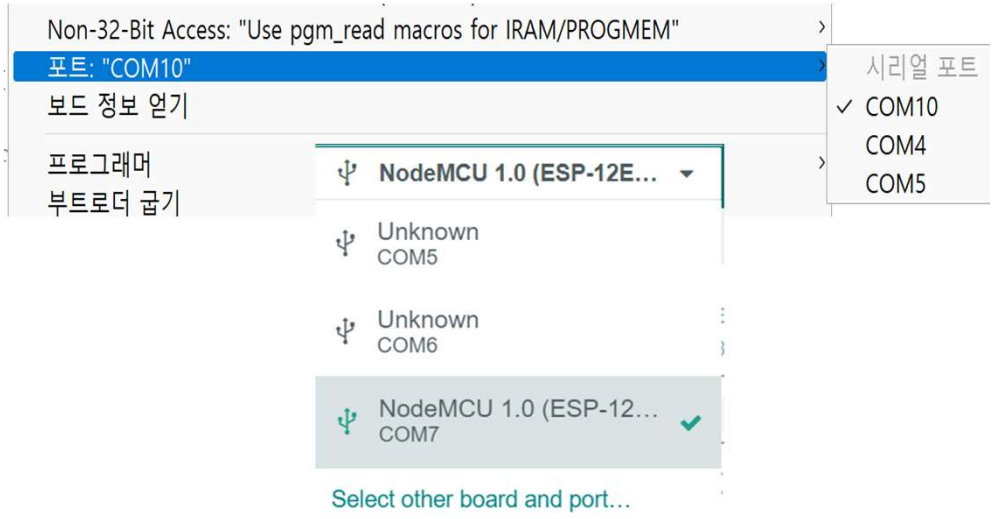
2. NodeMCU

- **NodeMCU**

  오류 발생시 우선 포트에 COM을 연결 (ex. COM10)

  -> 그 후, 시리얼 모니터로 이동 baudrate를 74880으로 맞춘 후 NodeMCU에 있는 FLASH를 누르고 RESET버튼 누르기

  -> 다음과 같이 boot mode가 나오면 문제 해결! Blink 예제문 다시 업로드 실행

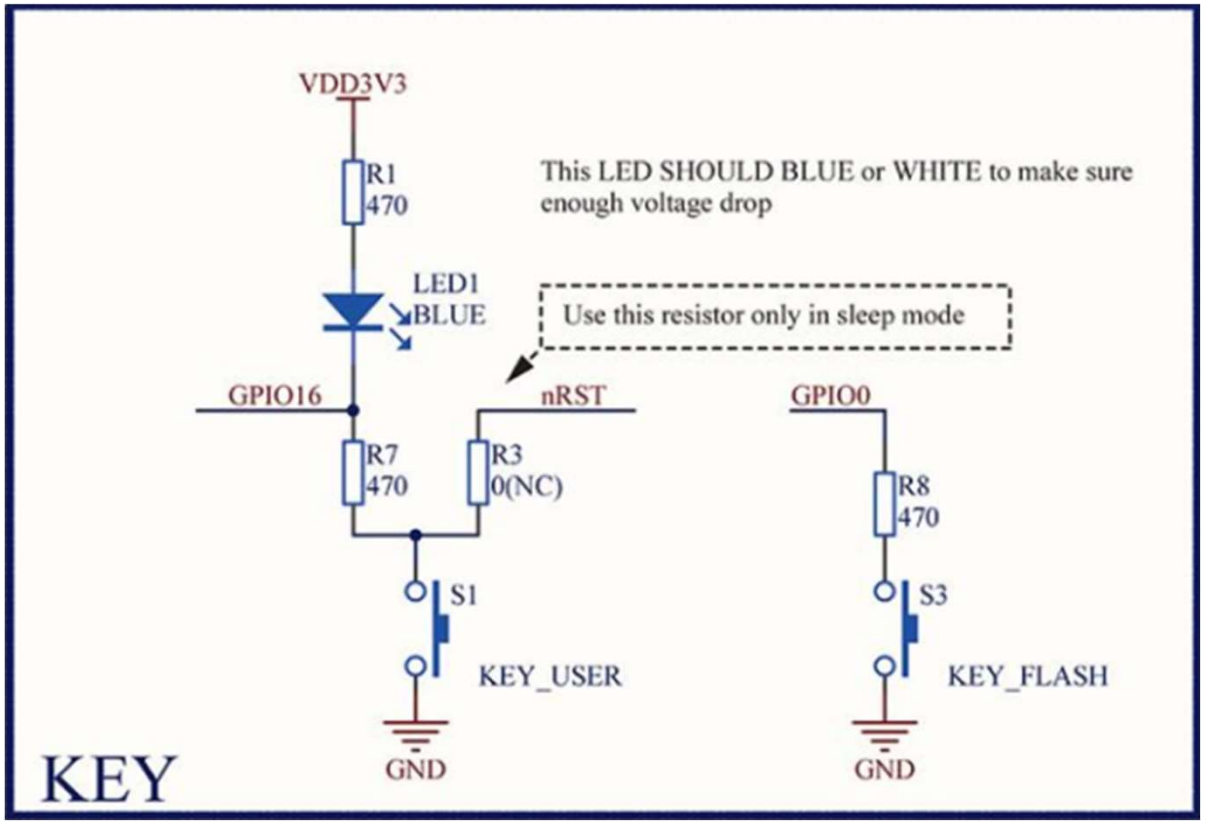



```
ets Jan  8 2013,rst cause:2, boot mode:(3,6)

load 0x40100000, len 2408, room 16
tail 8
chksum 0xe5
load 0x3ffe8000, len 776, room 0
tail 8
chksum 0x84
load 0x3ffe8310, len 632, room 0
tail 8
chksum 0xd8
csum 0xd8

2nd boot version : 1.6
  SPI Speed      : 40MHz
  SPI Mode       : QIO
  SPI Flash Size & Map: 32Mbit(512KB+512KB)
jump to run user1 @ 1000

rf cal sector: 1017
rf[112] : 00
rf[113] : 00
rf[114] : 01
```

2. NodeMCU

- **NodeMCU 활용**

1) NodeMCU의 내장 LED 제어하기




```
int LED_pin = 16;
int turn_on = 0;
int turn_off = 1;

void setup() {
 // put your setup code here, to run once:
 pinMode(LED_pin, OUTPUT);
 digitalWrite(LED_pin, turn_off);
}

void loop() {
 // put your main code here, to run repeatedly:
 digitalWrite(LED_pin, turn_on);
 delay(1000);
 digitalWrite(LED_pin, turn_off);
 delay(1000);
}
```

2. NodeMCU

- **NodeMCU 활용**

2) NodeMCU 활용하여 웹에 “HELLO WORLD!” 출력하기

```
#include <ESP8266WiFi.h>

const char* ssid = "AndroidHotspot9462";
const char* password = "hkyred3344";

WiFiServer server(80);
// 80은 내부에서 사용될 포트번호, 가급적 변경 X

void setup() {
  Serial.begin(9600);
  delay(10);

// Connect to WiFi network
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);
```

```
while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");

  // Start the server
  server.begin();
  Serial.println("Server started");

  // Print the IP address
  Serial.print("Use this URL to connect: ");
  Serial.print("http://");
  Serial.print(WiFi.localIP()); // 접속된 IP주소 출력
  Serial.println("/");
}
```

2. NodeMCU

- **NodeMCU 활용**

2) NodeMCU 활용하여 웹에 “HELLO WORLD!” 출력하기

```
void loop() {
// Check if a client has connected
  WiFiClient client = server.available();
  if (!client) {
   // client는 웹브라우저를 통해 접속한 유저
    return;
  }

// Wait until the client sends some data
  Serial.println("new client");
  while(!client.available()){
  delay(1);
  }

// Read the first line of the request
  String request = client.readStringUntil('₩r');
 //client의 url 알기
  Serial.println(request);
  client.flush();
```

```
// Return the response

  client.println("HTTP/1.1 200 OK");
  client.println("Content-Type: text/html");
  client.println(""); // 빈 줄을 삽입하여 헤더와 뒷 내용을 구분
  client.println("<!DOCTYPE HTML>");
// HTML5로 만들어진 문서 선언

  client.println("<html>");
  client.print("HELLO WORLD!");
  client.println("</html>");
  delay(1);

  Serial.println("Client disonnected");
  Serial.println("");
}
```

2. NodeMCU

- **NodeMCU 활용**

  2) NodeMCU 활용하여 웹에 “HELLO WORLD!” 출력하기

  시리얼 모니터를 열고 115200로 baud rate 변경 후 보드의 RESET 버튼 누르기

```
Connecting to iptime_human1
.........
WiFi connected
Server started
Use this URL to connect: http://192.168.0.4/
```

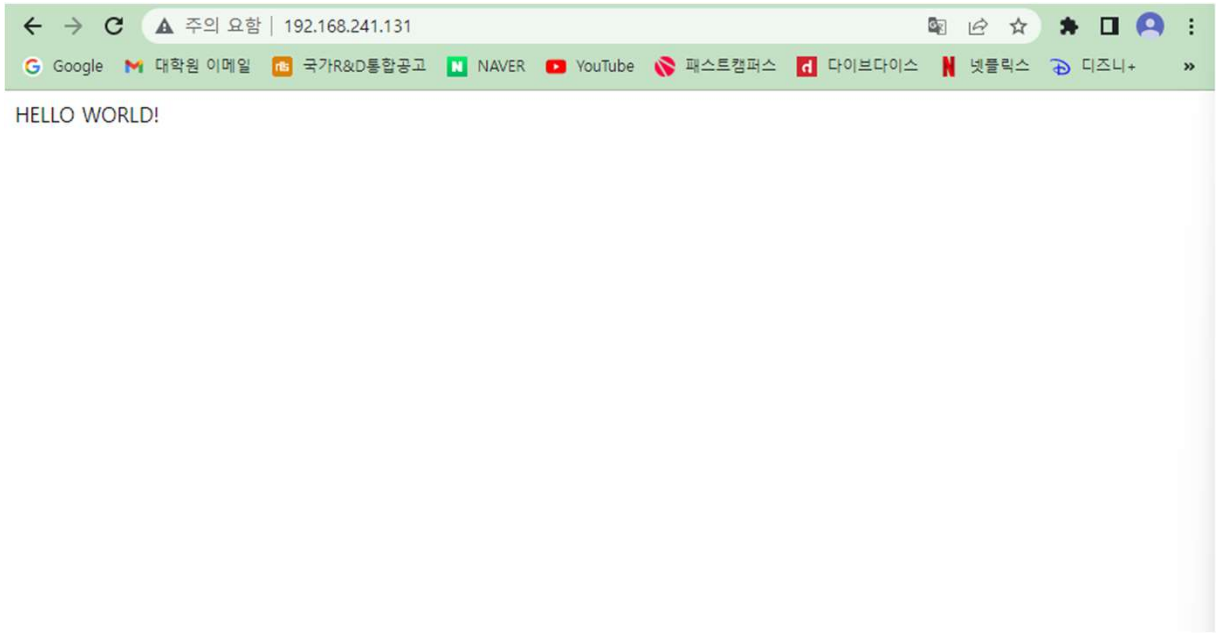
```
WiFi connected
Server started
Use this URL to connect: http://192.168.241.131/
```

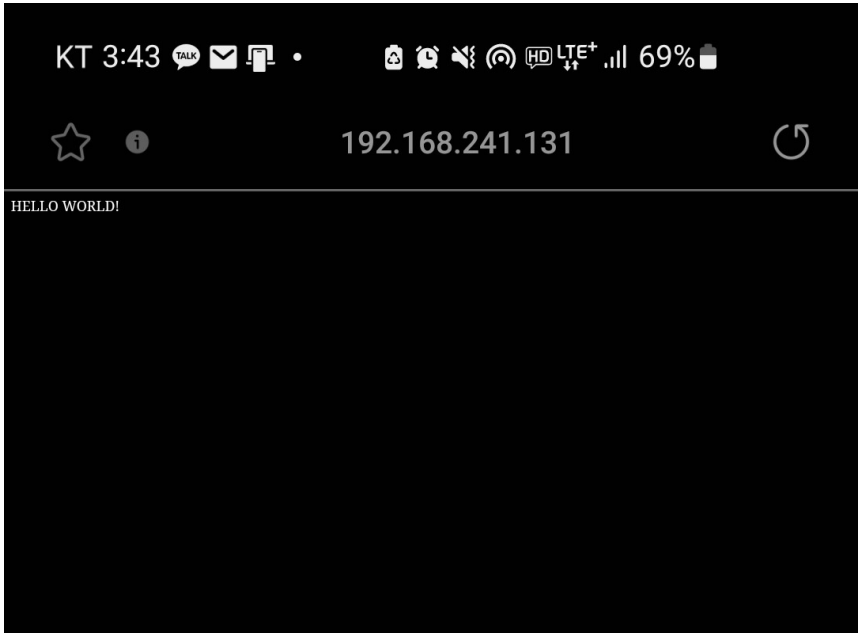
IP 확인 후 http://192.168.0.~을 스마트폰으로 접속하면 HELLO WORLD! 텍스트를 확인할 수 있다.

2. NodeMCU

- **NodeMCU 활용**

  2) NodeMCU 활용하여 웹에 “HELLO WORLD!” 출력하기

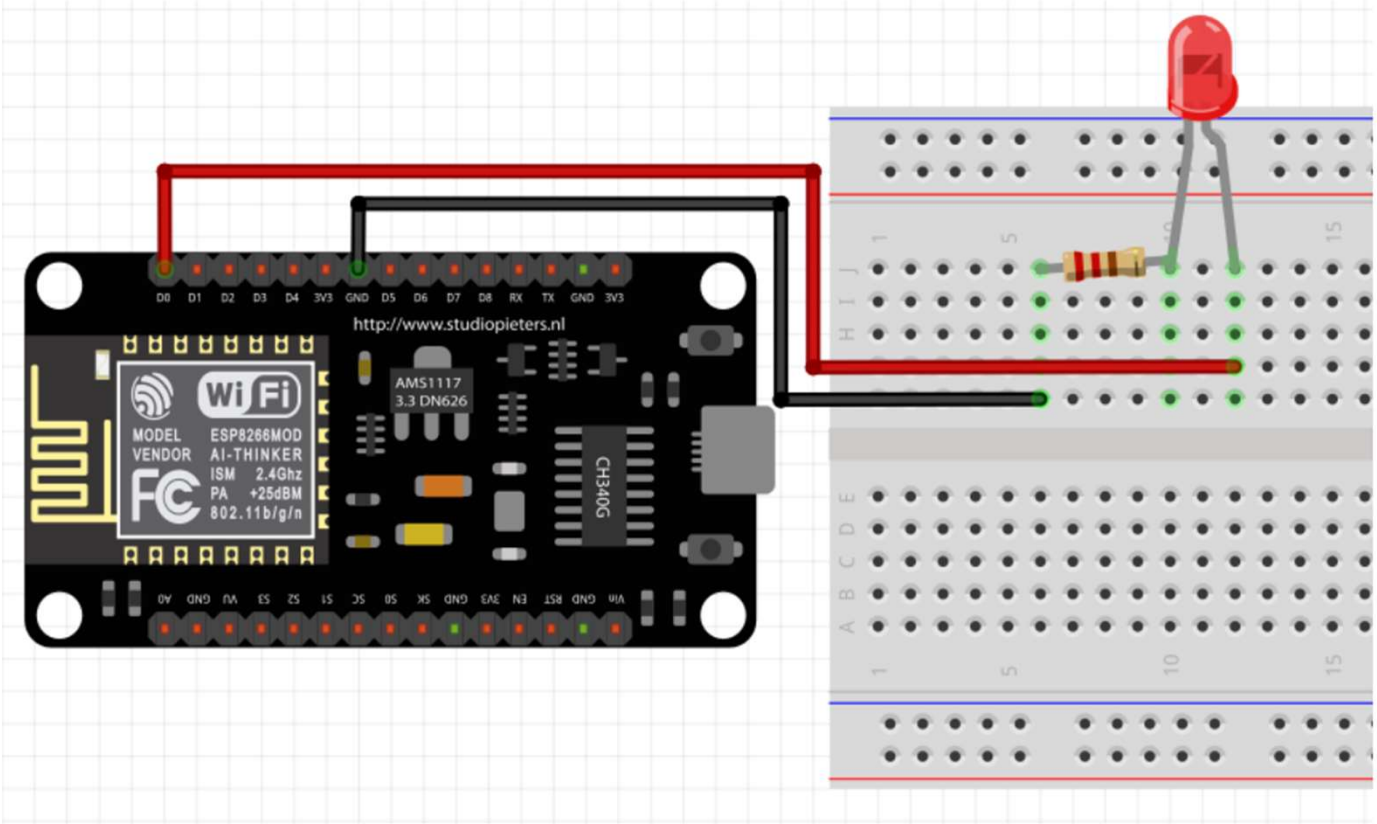




2. NodeMCU

- **NodeMCU 활용**

3) NodeMCU 활용하여 원격으로 LED 제어하기



2. NodeMCU

- **NodeMCU 활용**

3) NodeMCU 활용하여 원격으로 LED 제어하기

```
#include <ESP8266WiFi.h>
#define PIN_LED D0

const char* ssid = "AndroidHotspot9462";
const char* password = "hkyred3344";

WiFiServer server(80);

void setup() {
  pinMode(PIN_LED, OUTPUT);
  digitalWrite(PIN_LED, LOW);

  Serial.begin(115200);

  WiFi.mode(WIFI_STA);
// 다른 기기가 이 모듈을 통하여 접속 금지

  WiFi.begin(ssid, password);
```

```
  while (WiFi.status() != WL_CONNECTED) { //Wifi 접속까지
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.print("Connecting to ");
  Serial.println(ssid);
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());

  server.begin();
  Serial.println("Server started");
}
```

2. NodeMCU

- **NodeMCU 활용**

3) NodeMCU 활용하여 원격으로 LED 제어하기

```
void loop() {
  WiFiClient client = server.available();
// 웹사이트에 접속했을 때
  if(!client) return;
  Serial.println("새로운 클라이언트");
  client.setTimeout(5000);
// 클라이언트 전송 후 5초 초과 시 타임 아웃

  String request = client.readStringUntil('₩r');
// 전송받은 데이터 즉,URL을 알기 위해 사용

  Serial.println("request: ");
  Serial.println(request);
  if(request.indexOf("/ledoff") != -1) {
// LEDOFF를 누르면 LED, OFF

    digitalWrite(PIN_LED, LOW);
  }
```

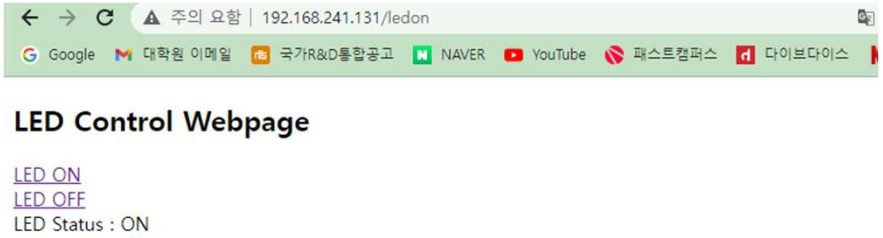
```
else if(request.indexOf("/ledon") != -1) {
    digitalWrite(PIN_LED, HIGH);
  }
  else {
    Serial.println("invalid request"); //
    digitalWrite(PIN_LED, digitalRead(PIN_LED));
// 현재 상태 유지
  }

  while(client.available()) {
    client.read(); // 버퍼 비움 효과
  }

  client.print("HTTP/1.1 200 OK");
  client.print("Content-Type: text/html₩r₩n₩r₩n"); // 해더
& 구분
  client.print("<!DOCTYPE HTML>");//HTML5로 만들어진 문
서 선언
  client.print("<html>");
```

2. NodeMCU

- **NodeMCU 활용**

3) NodeMCU 활용하여 원격으로 LED 제어하기

```
client.print("<head>");  //
  client.print("<meta charset=₩"UTF-8₩">");
  client.print("<title>LED Control Webpage</title>");
  client.print("</head>");

  client.print("<body>");
  client.print("<h2>LED Control Webpage</h2>");//제목
  client.print("<a href='/ledon'>LED ON</a>"); // 클릭 생성
  client.print("<br>"); // 줄 바꿈
  client.print("<a href='/ledoff'>LED OFF</a>"); // 클릭 생성
  client.print("<br>"); // 줄 바꿈
  client.print("LED Status : ");
  client.print((digitalRead(PIN_LED)) ? "ON" : "OFF"); //조건문
  client.print("</body>");

  client.print("</html>");

  Serial.println("클라이언트 연결 해제");
}
```




```
client.println("<br>");
client.println("<a href='/ledon'><button>LED On </button></a>");
client.println("<a href='/ledoff'><button>LED Off </button></a><br />");
```

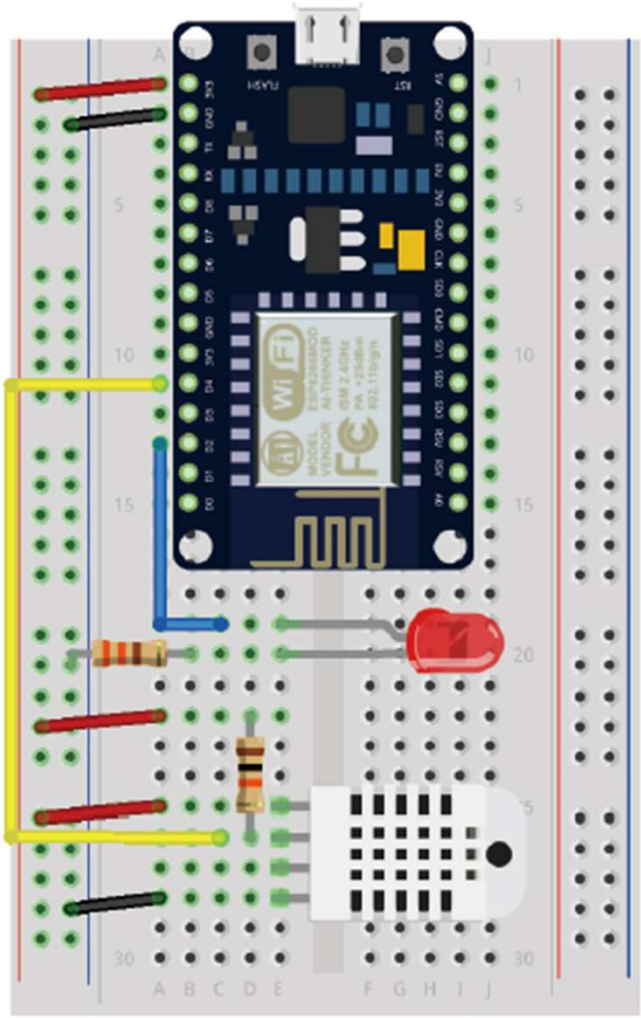
2. NodeMCU

- **NodeMCU 활용**

  4) NodeMCU 활용하여 DHT로 온습도 확인하기

  -> LED는 D2핀에 연결

  -> DHT11의 DOUT은 D4 핀에 연결



2. NodeMCU

- **NodeMCU 활용**

4) NodeMCU 활용하여 DHT로 온습도 확인하기

```
#include <ESP8266WiFi.h>
#include <DHT.h>

#define PIN_DHT D2

const char* ssid = "AndroidHotspot9462";
const char* password = "hkyred3344";


WiFiServer server(80);
WiFiClient client;
DHT DHTsensor(PIN_DHT, DHT11);

void setup() {
  DHTsensor.begin();
  Serial.begin(115200);
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
```

```
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

 Serial.println("");
  Serial.print("Connecting to ");
  Serial.println(ssid);
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());

  server.begin();
  Serial.println("Server started");
}

void loop() {
  client = server.available();
  if(!client) return;
```

2. NodeMCU

- **NodeMCU 활용**

4) NodeMCU 활용하여 DHT로 온습도 확인하기

```
  Serial.println("새로운 클라이언트");
  client.setTimeout(5000);
  String request = client.readStringUntil('₩r');
  Serial.println("request: ");
  Serial.println(request);
  while(client.available()) {
    client.read();
  }
float humidity = DHTsensor.readHumidity();
  float temp = DHTsensor.readTemperature();
  Serial.print("Humidity : ");
  Serial.print(humidity);
  Serial.print(" Temperature : ");
  Serial.print(temp);
  Serial.println(" ºC");
  client.print("HTTP/1.1 200 OK");
  client.print("Content-Type: text/html₩r₩n₩r₩n");
  client.print("<!DOCTYPE HTML>");
  client.print("<html>");
```

```
client.print("<head>");
  client.print("<meta charset=₩"UTF-8₩" http-
equiv=₩"refresh₩" content=₩"1₩">");
  client.print("<title>DHT senrsor test Webpage</title>");
  client.print("</head>");
  client.print("<body>");
  client.print("<h2>DHT senrsor test Webpage</h2>");
  client.print("<br>");
  client.print("Temperature : ");
  client.print(temp);
  client.print(" °C");
 client.print("<br>");
  client.print("Humidity : ");
  client.print(humidity);
  client.print(" %");
  client.print("</body>");
  client.print("</html>");

  Serial.println("클라이언트 연결 해제");
}
```

2. NodeMCU

- **NodeMCU 활용**

  4) NodeMCU 활용하여 DHT로 온습도 확인하기

  -> 실시간으로 온도와 습도가 변하는지 관찰

과제

2. 과제

- **NodeMCU 활용**

  NodeMCU와 Servo motor를 사용하기

  (1) Left 를 누르면 왼쪽으로 90도 회전

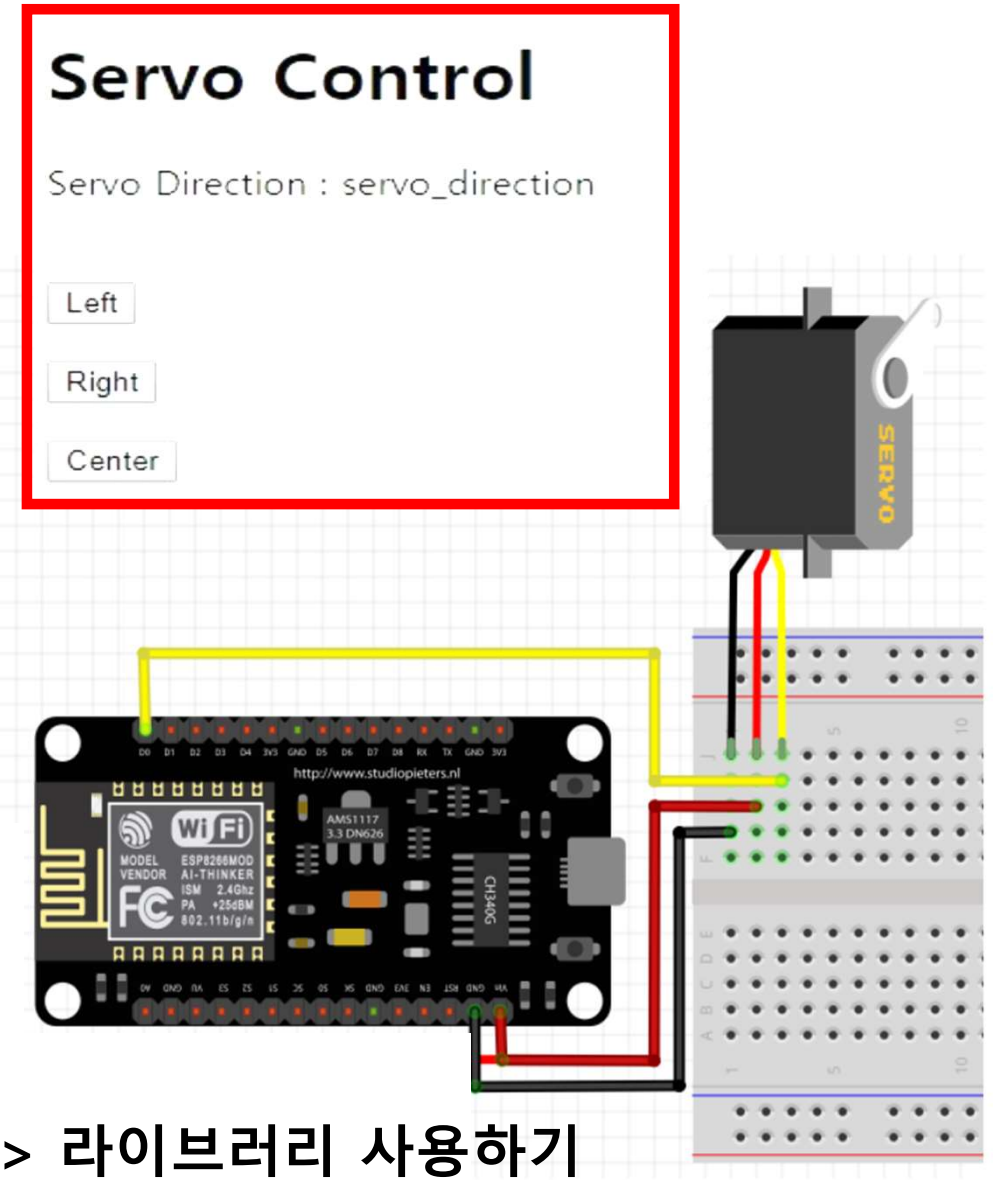
  (2) Right를 누르면 오른쪽으로 90도 회전

  (3) Center를 누르면 정중앙으로 돌아오기

  (4) 주황 선 -> 보드 D0

  빨간 선 -> 보드 Vin

  갈색 선 -> 보드 GND



Arduino에서 서보 모터제어 할 때 <Servo.h> 라이브러리 사용하기