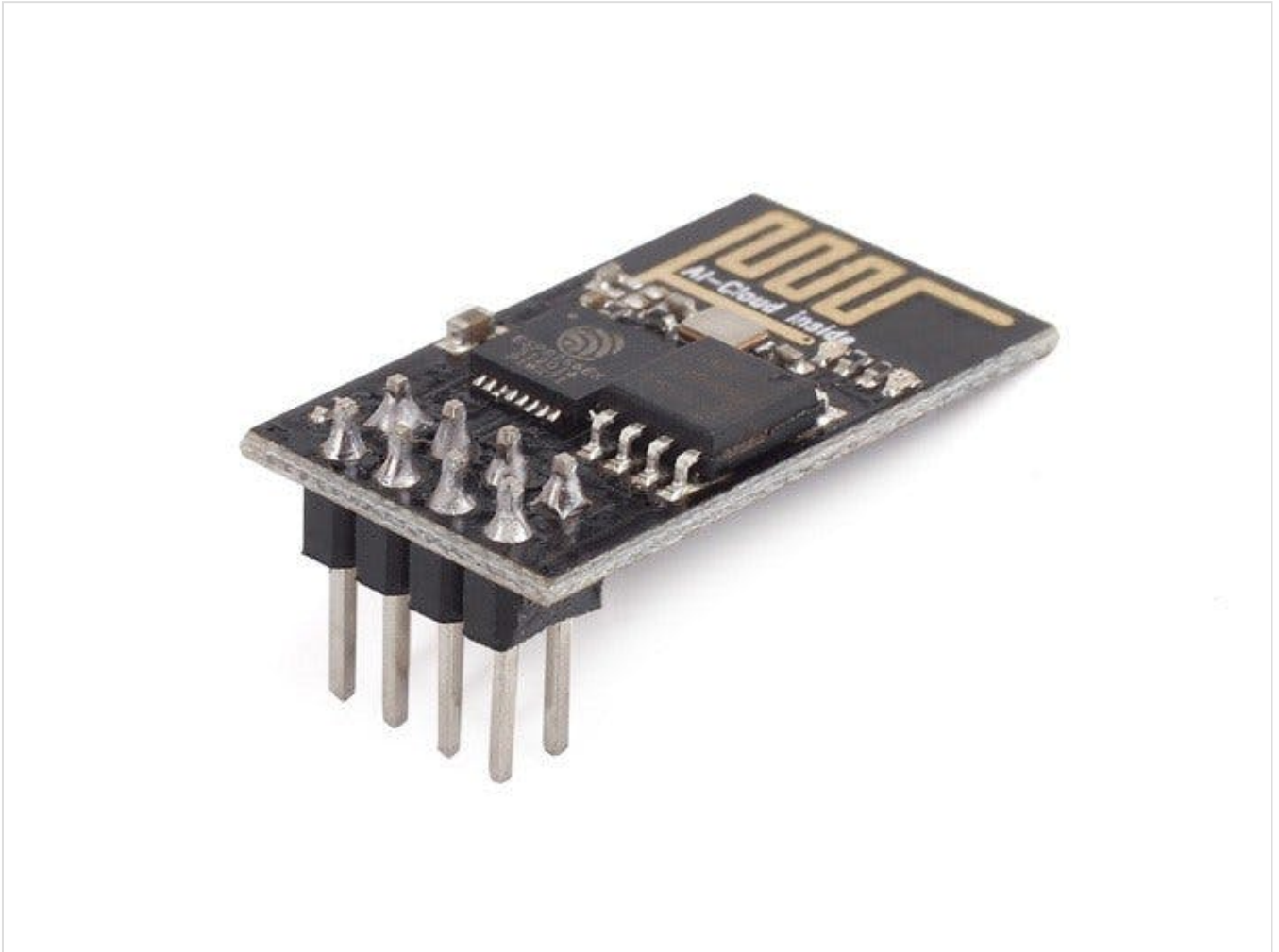


초저가 wifi모듈이라 생각하면 된다. 사실은 단순히 wifi모듈 이상, 직접 프로그래밍 할 수 있는 MCU이다.

아두이노에 연결해 wifi기능을 사용할 수 도 있지만, 칩 자체에 아두이노 부트로더를 올리고 아두이노로 개발할 수도 있고, 혹은 micropython이나 기타 여거가지 개발환경을 사용할 수도 있다. ESP32도 새롭게 소개되었는데, wifi + BLE 내장되어있다.

esp8266은 칩의 이름이고 이를 사용한 개발 보드가 많이 나와있는데, ESP-01 (작고 심플), ESP-12가 많이 사용된다.



ESP-01



ESP-12

제조사 페이지: <https://www.espressif.com/products/hardware/esp8266ex/overview/>

데이터시트: <https://nurdspace.nl/ESP8266>

커뮤니티 포럼: <https://www.esp8266.com/>

AT command :<https://cdn.sparkfun.com/datasheets/Wireless/WiFi/ESP8266ModuleV1.pdf>

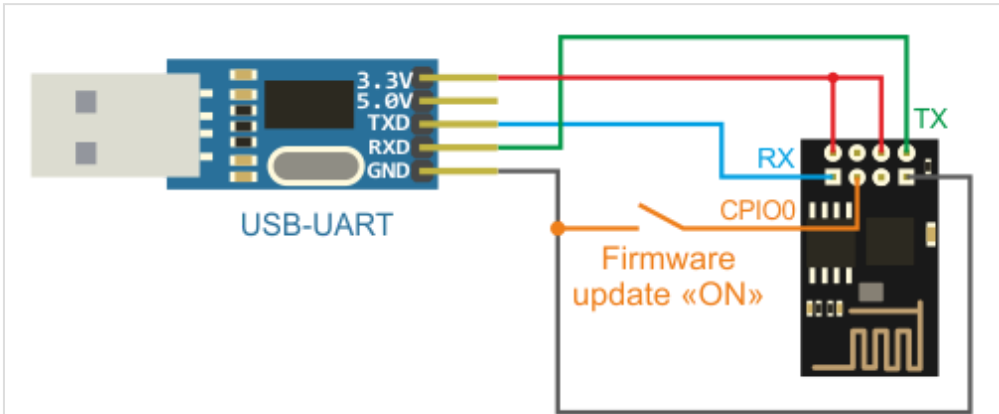
한글자료: [http://www.hardcopyworld.com/gnuboard5/bbs/board.php?](http://www.hardcopyworld.com/gnuboard5/bbs/board.php?bo_table=lecture_esp&wr_id=1)

[bo_table=lecture_esp&wr_id=1](http://www.hardcopyworld.com/gnuboard5/bbs/board.php?bo_table=lecture_esp&wr_id=1)

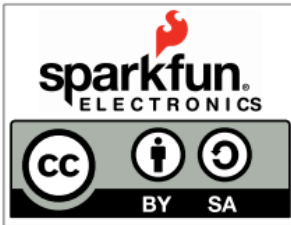
warning! ESP계열 사용시 잘 까먹는 꼭주의할 점은 모든 입출력은 3.3v라는 점!!

연결테스트

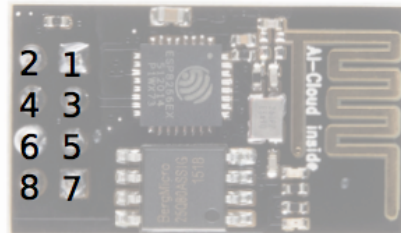
[↑]



ESP8266 Module (WRL-13678)



D7	GPIO1	TX	2- TXO
	Chip Enable		4- CHPD
	Reset		6- RST
	3.3V		8- 3V
	GND		1- GND
D2/SDA	GPIO2		3- GPIO2
D0	GPIO0		5- GPIO0
D8	GPIO3	RX	7- RXI



PCB Antenna

Power
VCC-3.0-3.6V
Standby ~ 0.9uA
Running ~60-215mA,
Average ~ 80mA

Wifi Features
802.11 b/g/n
2.4GHz
WPA/WPA2
Wifi Direct

+20dBm output power (802.11b)

I/O Features
Integrated TCP/IP
Integrated TR switch, LNA,
balun

Memory/Speed Features
80MHz
64KB instruction RAM
96KB data RAM
64K boot ROM
1MB* Flash Memory

Basic Connection
VCC - 3.3V
GND - GND
TX - RX on Arduino or FTDI
RX - TX on Arduino or FTDI
Chip Enable - 3.3V

Default Baud Rate
11520* 8N1

LEDs
Red: Power
Blue: TX

*milage may vary on different version of the board

AT Command Usage

Commands are case sensitive and should end with `/r/n`

Commands may use 1 or more of these types
Set = AT+<x>=<y> - Sets the value
Inquiry = AT+<x>? - See what the value is set at
Test = AT+<x>=? - See the possible options
Execute = AT+<x> - Execute a command

Commands with * have been deprecated in favor of COMMAND_CUR and COMMAND_DEF_CUR will not write the value to flash, DEF will write the value to flash and be used as the default in the future.

AT Command List

AT - Attention
AT+RST - Reset the board
AT+GMR - Firmware version
AT+CWMODE* - Operating Mode
1. Client
2. Access Point
3. Client and Access Point
AT+CWJAP=<ssid>,<pwd> - Join network
AT+CWLAP - View available networks
AT+CWQAP - Disconnect from network
AT+CWSAP=<ssid>,<pwd>,<chl>,<ecn> - Set up access point
0. Open. No security
1. WEP
2. WPA_PSK
3. WPA2_PSK
4. WPA_WPA2_PSK
AT+CWLIF - Show assigned IP addresses as access point
AT+CIPSTATUS - Show current status as socket client or server
AT+CIPSTART=<type>,<addr>,<port> - Connect to socket server
IP is fixed at 192.168.4.1, mask is fixed at 255.255.255.0
if CIPMUX is set to multichannel add <id> to beginning of string
AT+CIPCLOSE - Close socket connection
AT+CIFSR - Show assigned IP address when connected to network
AT+CIPMUX=<mode> - Set connection
0. Single Connection
1. Multi-Channel Connection
AT+CIPSERVER=<mode>,<port>[AT+CIPMUX=1] - Default port is 333
0. Close the Socket Server
1. Open the Socket Server
AT+CIPMODE=<mode> - Set transparent mode
Data received will be sent to serial port as
0. +IPD,<connection channel>,<length>format (AT+CIPMUX=[0,1])
1. Data stream (AT+CIPMUX=0)
AT+CIPSTO=<time> - Set auto socket client disconnect timeout from 1-28800s

Example commands
AT+CWMODE=? //View options for mode (test)
AT+CWMODE=3 //Set mode to client and access modes (set)
AT+CWLAP //View available networks (execute)
AT+CWJAP = "ssid","password" //Join network (set)
AT+CWJAP? //View the current network (inquiry)
AT+CIFSR //Show IP address (execute)
AT+CWQAP //Disconnect from network (execute)
AT+CWSAP="apoint","pass",11,0//Setup an open access point (set)
AT+CWLIF //Show devices connected to access point

일단 잘 작동하는지 컴퓨터에 UART연결해 테스트 해보자.USB-UART 컨버터를 사용해 노트북에 연결한다.

위 이미지에서 처럼

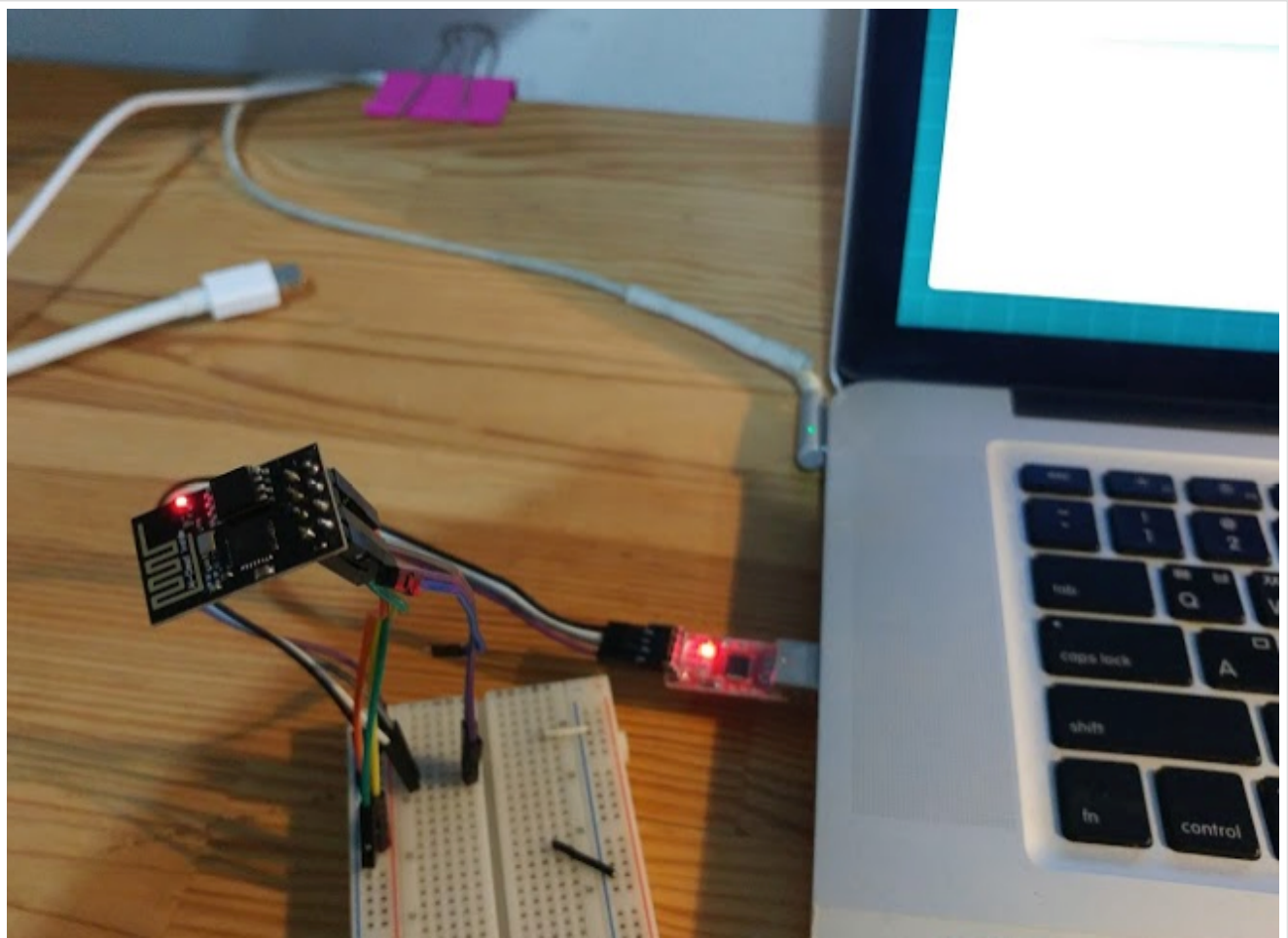
usb-uart	esp8266
3.3v	4번 (3.3v), 8번(CHPD-chip enable)

usb-uart	esp8266
TX	7번(RX)
RX	2번(TX)
GND	1번(GND)

으로 연결하고, 혹시나 펌웨어 업데이트가 필요할 때에는 한시적으로


usb-uart	esp8266
GND	5번(GPIO0)

을 연결한다.



coolterm 등 시리얼 터미널 프로그램을 사용해 연결한다.(물론 아두이노 시리얼 모니터도 좋다) 이때 초기 설정된 baudrate는 115200.아래의 커맨드를 넣어서 연결확인해보자.note: 명령어를 보낼 때에는 대소문자를 구분한다는 점, 그리고 리턴문자(both NL + CR)까지 보내야 한다는 점!

아래 명령어를 차례로 입력해 작업실의 wifi에 연결하고 확인해보자.



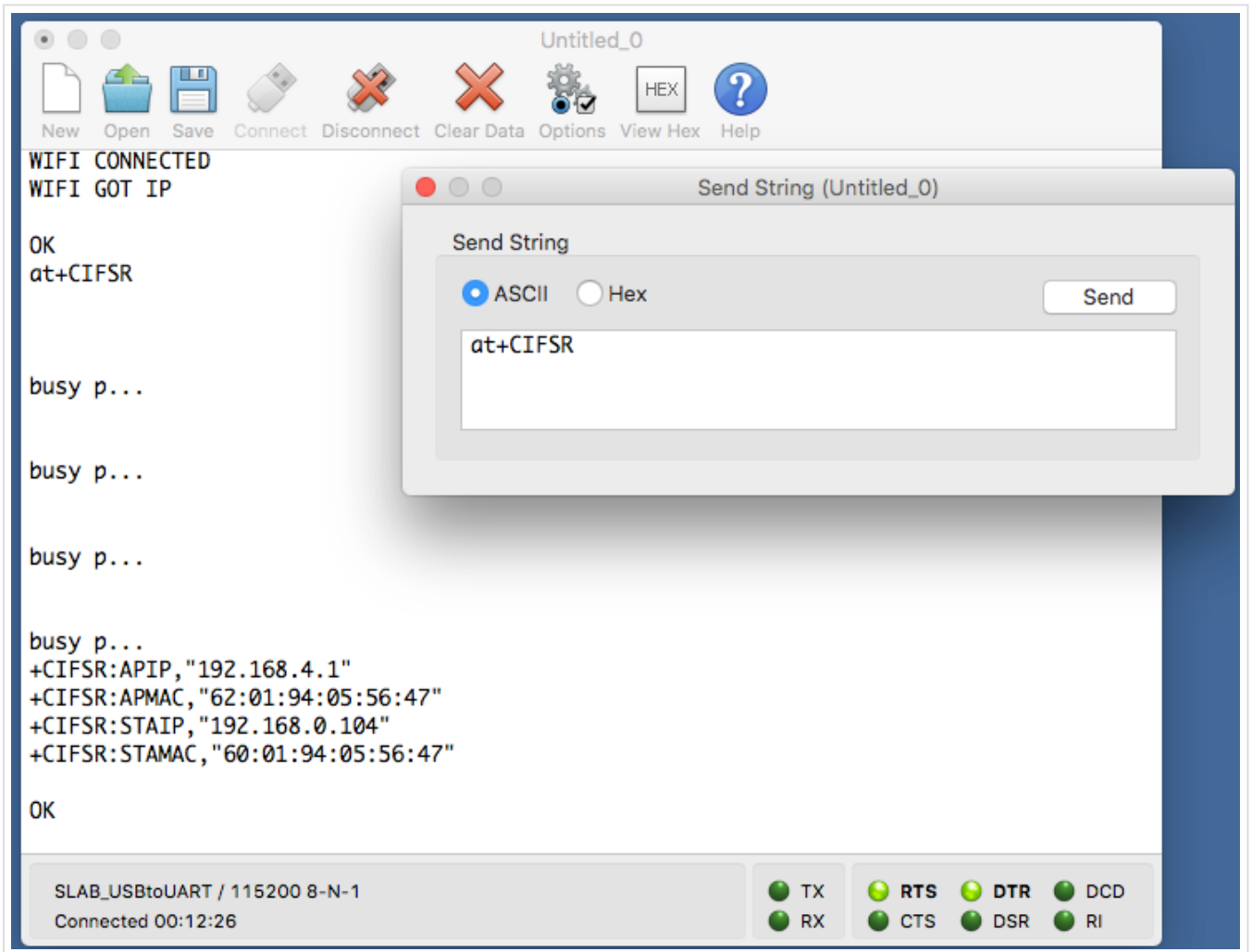
Goodnight moon!
Hello, world?
ERROR
at
OK
at+GMR
AT version:0.40.0.0(Aug 8 2015 14:45:58)
SDK version:1.3.0
Ai-Thinker Technology Co.,Ltd.
Build:1.3.0.2 Sep 11 2015 11:48:04
OK
AT+CIFSR
+CIFSR:STAIP,"0.0.0.0"
+CIFSR:STAMAC,"60:01:94:05:56:47"
OK
AT+CWJAP="doguin_studio","hellodoguin1!"
WIFI CONNECTED
WIFI GOT IP
OK

☒ 자동 스크롤 Both NL & CR 9600 보드레이트 출력 지우기

AT+CWMODE=3
(Response) OK

AT+CWJAP="ssid","password"
(Response)
WIFI CONNECTED
WIFI GOT IP
OK

AT+CIFSR
(Response)
+CIFSR:APIP,"192.168.4.1"
+CIFSR:APMAC,"62:01:94:05:56:47"
+CIFSR:STAIP,"192.168.0.104"
+CIFSR:STAMAC,"60:01:94:05:56:47"



잘된다.

baudrate 변경

[↑]

많은 예전 자료에서 8266을 사용하려면 가장먼저 펌웨어를 바꾸어 baudrate를 낮추어주어야 한다고 하는데, 기본 펌웨어로도 잘 된다.(사실 항상 잘 되지는 않고 데이터를 자꾸 잃어버린다. 펌웨어를 바꿀 필요는 없지만 아래 방법으로 속도를 낮추어 사용하자.)

아두이노 software Serial은 19200 bps까지밖에 지원하지 않기 때문이라는 설이 있으나 arduino uno에서는 115200까지 지원된다. 다만 attiny85라든가 클락수가 낮은 칩에서는 115200의 속도를 지원하지 않기 때문에 기본 115200으로 되어있는 시리얼 속도는 낮춰주어야 통신할 수 있다.

~~AT+CI0BAUD=9600~~ 명령어로 속도를 조절한다.

이 명령어는 ~~AT+GMR~~ 로 버전 확인 했을 때

```
AT version:0.40.0.0(Aug 8 2015 14:45:58)
SDK version:1.3.0
Ai-Thinker Technology Co.,Ltd.
Build:1.3.0.2-Sep 11 2015 11:48:04
```

이 이상에서 잘 작동한다.

AT+UART_DEF=9600,8,1,0,0 명령어로 속도 및 기타 시리얼 통신을 설정한다.

물론 전원 뽑았다가 다시 연결해도 변경된 9600으로 잘 작동되고, 앞서 설정한 wifi ap에 자동으로 연결된다.

펌웨어 업데이트

[↑]

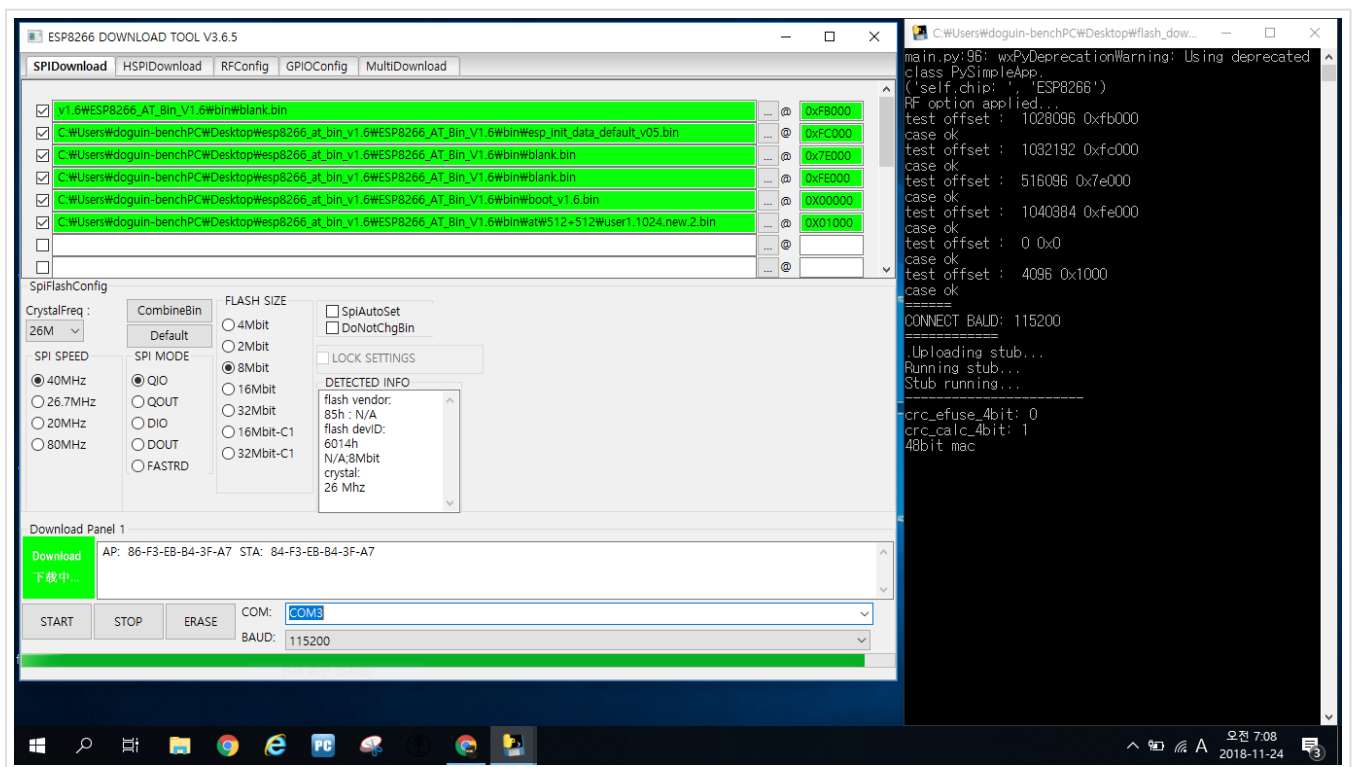
필요한 경우가 생겼다면 아래 페이지를 참고한다.

펌웨어업데이트 참고: <https://www.allaboutcircuits.com/projects/flashing-the-ESP-01-firmware-to-SDK-v2.0.0-is-easier-now/>

note: 업데이트시에는 GPIO0번 핀을 GND에 연결한상태여야 한다는 점!!

펌웨어 바이너리 파일은 이곳에서: <https://www.espressif.com/en/support/download/at> v1.6이 잘 작동하는 것 확인하였다.

바이너리파일을 모듈에 복사해주는 flash tool은 이곳에
서: <https://www.espressif.com/en/support/download/other-tools>



Arduino와 연결

[↑]

세가지 정도 주의하면 되겠다.

Baudrate 위에서 설정한대로 맞춰주기 (기본은 115200), 리인 끝에 'both NL & CR' ("WnWr") 반드시 붙여 메시지 보내기

* esp로의 입력은 3.3v로 맞춰주기 - 레벨 시프터를 사용하거나 10K, 4.7K 저항을 사용해 voltage

divider를 만들어 사용한다. (그냥 rx에 5v 신호 꽂아도 잘 된다는 자료도 있는데 고장날까 싶어 테스트해보지는 못했다.)

AT 명령어

[↑]

https://www.espressif.com/sites/default/files/documentation/4A-ESP8266_AT_Instruction_Set_EN.pdf

주요한 명령어 몇개만 보자.

명령어	기능	예제
AT	test	
AT+RST	모듈 리셋	
AT+GMR	펌웨어 버전 체크	
AT+CIIOBAUD AT+UART_DEF	uart Baudrate 변경	AT+CIIOBAUD=9600 AT+UART_DEF=9600,8,1,0,0 참 고: https://arduino.stackexchange.com/questions/24156/how-to-change-baudrate-of-esp8266-12e-permanently
AT+CWMODE	wifi mode 변경 1: station mode (공유기에 접속) 2: softAT mode (esp가 공유기가 됨 (인트라넷 구성)) 3: both	AT+CWMODE=3
AT+CWJAP	ap에 연결	AT+CWJAP="wifi-ssid","wifi-password"
AT+CWQAP	ap 연결 종료	
AT+CIPMUX	동시연결 기능 0: 단일연결 (클라이언트로 사용시) 1~4: 동시연결 가능수 (서버로 사용시)	AT+CIPMUX=0
AT+CIPSTART	TCP, UDP, SSL 연결 시작	AT+CIPSTART="TCP","teachmemicro.com",8080 AT+CIPSTART="UDP",192.168.10.110",1000,1001,1
AT+CIPSEND	데이터 보내기. parameter= 보내는 바이트	AT+CIPSEND=8
AT+CIPCLOSE	tcp 연결 종료	

HTTP 리퀘스트

[↑]

참고: <https://www.linkedin.com/pulse/how-make-rest-api-http-post-call-using-arduino-uno-esp8266-taha-ali/>

제조사에서 제공하는 공식 예

제: <https://www.espressif.com/en/products/hardware/esp8266ex/resources>

제조사 자료 참고:

https://www.espressif.com/sites/default/files/documentation/4b-esp8266_at_command_examples_en.pdf

1. set Wifi mode

```
AT+CWMODE = 3
```

```
response:
ok
```

1. Connect to a router

```
AT+CWJAP="SSID", "PASS"
```

```
response:
ok
```

1. ESP8266 connects to the server as a TCP client

```
AT+CIPSTART="TCP", "192,168,4,1", 8080 // protocol, server IP, port
```

```
response:
ok
```

1. ESP8266 sends data to the server

```
AT+CIPSEND = 4 // 보낼 데이터의 바이트수
```

```
response:
```

```
> // 데이터 보낼 준비 완료 프롬프트
```

```
>test // no CR
```

```
response:
```

```
Recv 4 bytes
```

```
SEND OK
```

1. 서버에서 응답받으면 표시됨

```
response:
```

```
+IPD,n:xxxxxxxxxx // received n bytes, data=xxxxxxxxxx
```

1. 연결 끊기



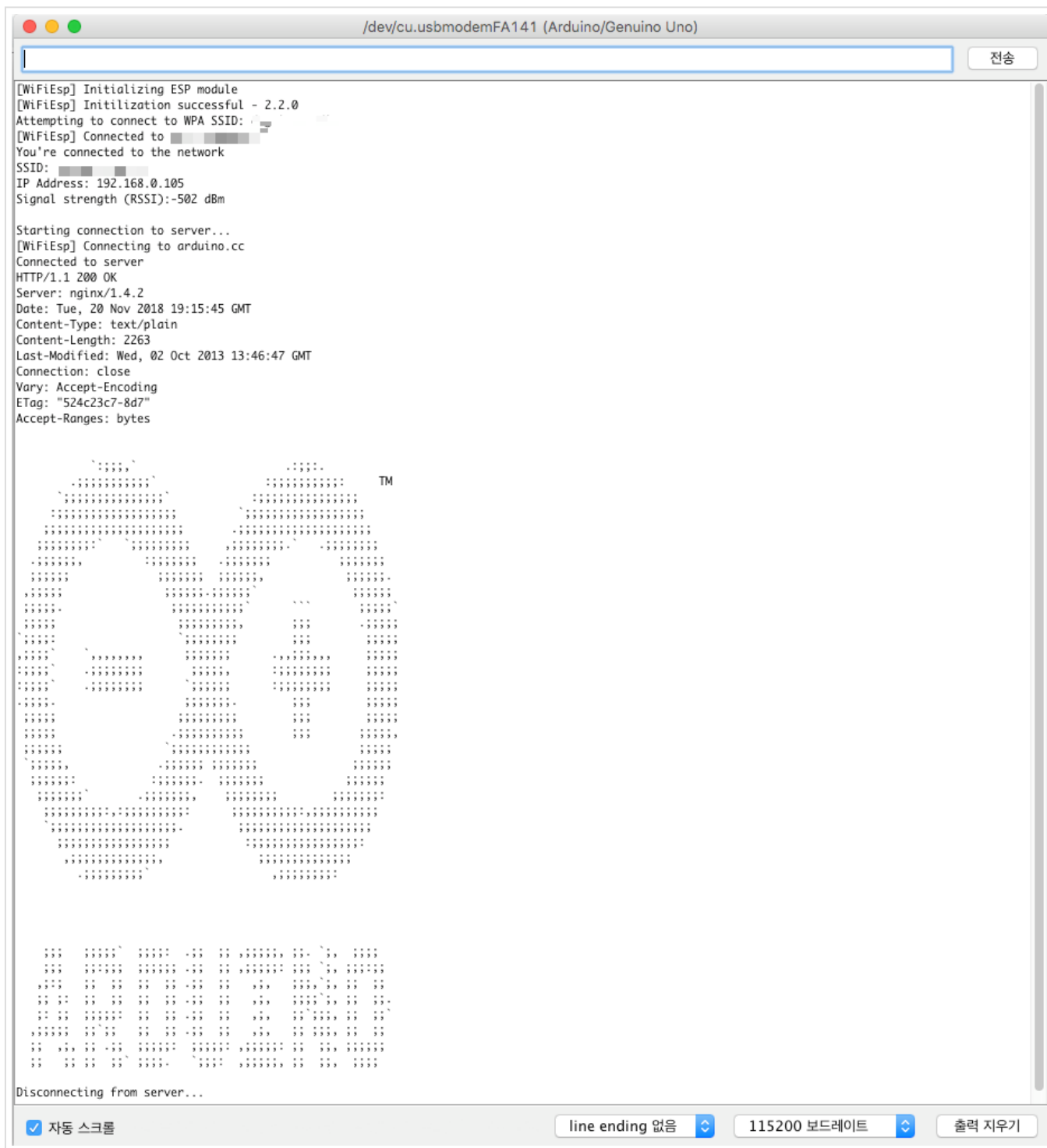
라이브러리 매니저에서 'WiFiEsp'를 검색해 설치한다.

예제 실행하기

라이브러리 설치시 함께 복사된 'WebClient' 예제를 사용해 Adafruit Io에 데이터 보내보자.

1. 아두이노의 파일 > 예제 > wifesp > webclient 메뉴를 선택해 예제 불러오기
2. softwareserial 핀연결 수정
3. 시리얼 통신 속도 수정
3. ssid[] 와 pass[] 수정
4. 아두이노에 스케치 업로드하고 시리얼 모니터 실행시켜 테스트

잘된다~!



Adafruit IO에 데이터 보내고 받기

[↑]

WiFiEsp 라이브러리와 함께 응답으로 오는 json을 손쉽게 처리하기 위해 ArduinoJson 라이브러리를 사용하였다.

arduinojson 참고: <https://arduinojson.org/>

adafruit io 참고

```

/* WiFiEsp-adafruit_io.ino
 *
 * ESP8266 + http request (rest) 사용해 Adafruit IO에 데이터 주고받기
 * Written by 도구의 인간 박상현
 * 2018.Nov
 *
 * 키보드로 '?'를 입력받으면 adafruit.io 에 저장된 마지막 feed data 값 읽어오기
 * 이외의 입력은 adafruit.io에 저장하기.
 *
 * POST request 보낼 때 서버로 값 전달이 잘 된다면 시리얼모니터에 뜨는 에러 메시지는 무시하자.
 * 숫자만 전달이되고 영문자가 전달되지 않는데, adafruit io쪽에서 손질해주어야하는 것 같다. 문자
사용필요하다면 좀 더 연구해보자
 *
 */

#include "WiFiEsp.h"
#include "ArduinoJson.h"
#include "SoftwareSerial.h"

SoftwareSerial Serial1(10, 11); // RX, TX

//////////////////// 사용자 값 설정
////////////////////
char ssid[] = "mySSID";           // your network SSID (name)
char pass[] = "myPASS";           // your network password
char server[] = "io.adafruit.com";
char ADAFRUIT_USERNAME[] = "myID";
char ADAFRUIT_FEED_KEY[] = "sensor1";
char X_AIO_KEY[] = "a34eb5d---myKEY---b7e3959238";
////////////////////

int status = WL_IDLE_STATUS;      // the Wifi radio's status

// Initialize the Ethernet client object
WiFiEspClient client;

void setup()
{
  // initialize serial for debugging
  Serial.begin(115200);
  // initialize serial for ESP module
  Serial1.begin(9600);
  // initialize ESP module
  WiFi.init(&Serial1);

  // check for the presence of the shield
  if (WiFi.status() == WL_NO_SHIELD) {
    Serial.println("WiFi shield not present");
    // don't continue
    while (true);
  }

  // attempt to connect to WiFi network
  while ( status != WL_CONNECTED) {

```

```

    Serial.print("Attempting to connect to WPA SSID: ");
    Serial.println(ssid);
    // Connect to WPA/WPA2 network
    status = WiFi.begin(ssid, pass);
}

// you're connected now, so print out the data
Serial.println("You're connected to the network");

}

void loop()
{
    // 시리얼 키입력을 받아 값이 '?' 라면 최근 데이터 가져오기
    // '?'외의 입력은 adafruit io로 보내 기록하기
    String input;
    if (Serial.available()){
        input = Serial.readString();
        Serial.print("key input: "); Serial.println(input);

        if (input.equals("?")){
            String lastValue = readAdafruitData();
            Serial.print("last value:"); Serial.println(lastValue);
        }else{
            writeAdafruitData(input);
        }
    }

    // // 서버로부터의 응답 모니터링
    // while (client.available()) {
    //     char c = client.read();
    //     Serial.write(c);
    // }

}

// adafruit io의 가장 최근 값 가져오기
char* readAdafruitData(){

    Serial.println("Starting connection to server...");
    // if you get a connection, report back via serial
    if (client.connect(server, 80)) {
        Serial.println("Connected to server");
        // Make a HTTP request
        client.println("GET /api/v2/" + String(ADAFRUIT_USERNAME) + "/feeds/" + String(ADAFRUIT_FEED_KEY) + "/data/last HTTP/1.1");
        //header
        client.println("Host: " + String(server));
        client.println("X-AIO-Key: " + String(X_AIO_KEY));
        client.println("Connection: close");
        client.println();

        //delay(200); // 상황에 따라 시간 조절필요..
    }
}

```

```

// arduinoJson Library 사용해 값 읽기
char endOfHeaders[] = "\r\n\r\n";
client.find(endOfHeaders);

client.read(); // 왜인지 모르겠는데 응답의 body 앞에 d5\r\n, d7\r\n하는 식으로 4byte가
불어온다.
client.read();
client.read();
client.read();

DynamicJsonBuffer jsonBuffer(100);
JsonObject& root = jsonBuffer.parseObject(client);

return(root["value"].as<char*>());

}
}

// adafruit io에 값 쓰기
void writeAdafruitData(String data){

    String content = "{\"value\":\"" + data + "\"}";

    Serial.println("Starting connection to server...");
    // if you get a connection, report back via serial
    if (client.connect(server, 80)) {
        Serial.println("Connected to server");
        // Make a HTTP request
        client.println("POST /api/v2/" + String(ADAFRUIT_USERNAME) + "/feeds/" + String(ADAFRUIT_FEED_KEY) + "/data HTTP/1.1");
        //header
        client.println("Host: " + String(server));
        client.println("X-AIO-Key: " + String(X_AIO_KEY));
        client.println("Content-Length: " + String(content.length()));
        client.println("Content-Type: application/json");
        client.println("Connection: close");
        client.println();
        //body
        client.println(content);

    }
}

```