ESP-01 WIFI MODULE

Handong university

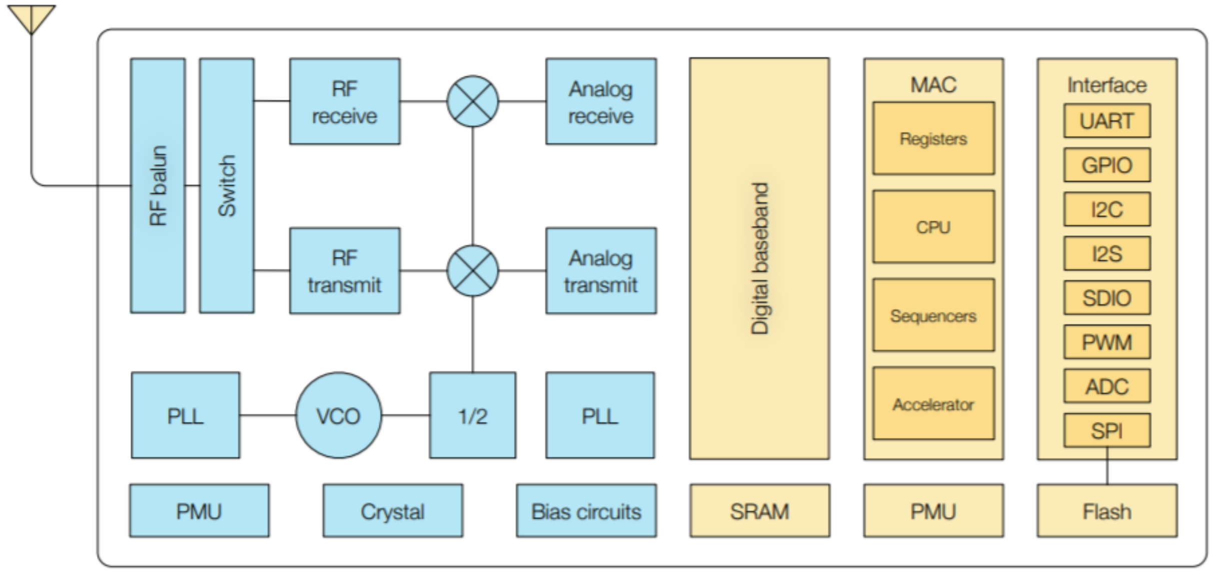
Jong-won Lee

2022.05.18

ESP8266

ESP8266EX

- ESP8266EX
  - A low-cost Wi-Fi chip with full TCP/IP stack and MCU produced by Espressif Systems (Chinese manufacturer)
- Block diagram



ESP8266EX

- Features
  - Based on Tensilica L106 32-bit RISC processor
    - Max. Clock: 160 MHz
  - Support IEEE 802.11 b/g/n (2.4 GHz)
  - No programmable ROM in the SoC
    - Use external SPI Flash (up to 16 MB: typically 1M – 4MB)
  - SRAM: 160 kB
    - 32 KiB instruction RAM, 32 KiB instruction cache RAM
    - 80 KiB user-data RAM, 16 KiB ETS system-data RAM
    - SRAM size available to user application: < 50 kB
  - 16 GPIO, SPI, I2C, I2S, UART, 10-bit ADC

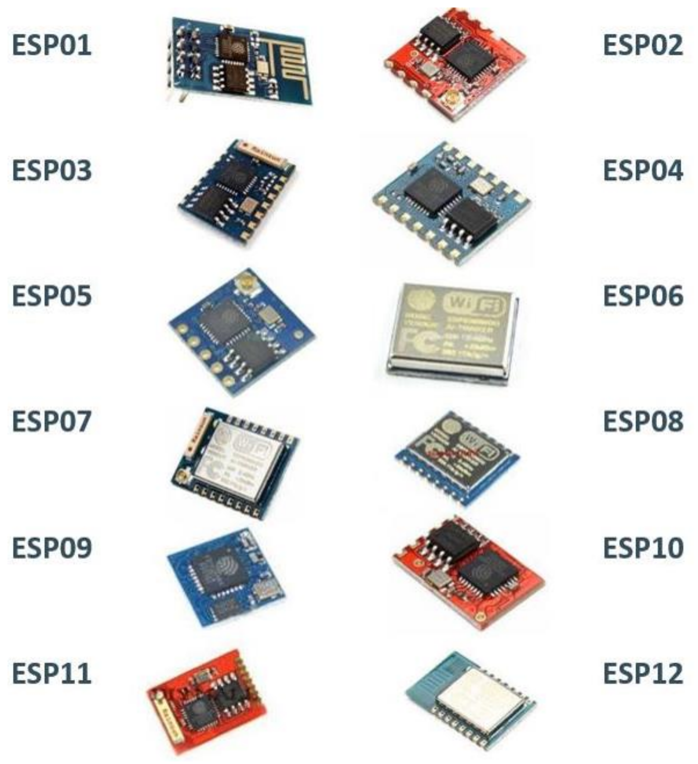
ESP8266EX

- Modules by Espressif Systems

Module	Description	Chip Embedded	Dimensions (mm)	Pins	Flash (MB)	PSRAM (MB)	Antenna	Development Board
ESP-WROOM-02D	ESP-WROOM-02D is an ESP8266EX-based module that have optimized RF performance.	ESP8266EX	18x20x3.2	18	2,4	N/A	PCB antenna	ESP8266-DevKitC
ESP-WROOM-02U	ESP-WROOM-02U is an ESP8266EX-based module that has optimized RF performance. It integrates a U.FL connector.	ESP8266EX	18x14.3x3.2	18	2,4	N/A	IPEX antenna	ESP8266-DevKitC
ESP-WROOM-02	ESP-WROOM-02 is based on ESP8266EX, measuring as small a size as 18x20x3 mm.	ESP8266EX	18x20x3	18	2,4	N/A	PCB antenna	N/A
ESP-WROOM-S2	ESP-WROOM-S2 can work as the SDIO/SPI slave, with the SPI speed being up to 8 Mbps.	ESP8266EX	16x23x3	20	2,4	N/A	PCB antenna	N/A

ESP8266

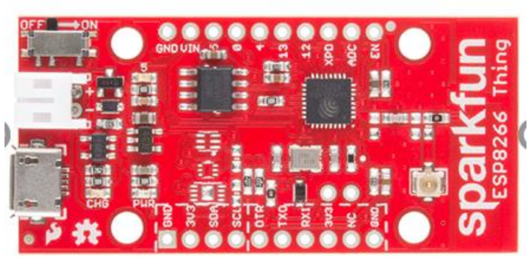
- Many modules by AI-Thinker
  - Series of ESP-xx
    - ESP-01. ESP-01M, ESP-02, …, ESP14



ESP8266

- Development boards
  - With USB interface
    - WEMOS D1/D1 mini/…

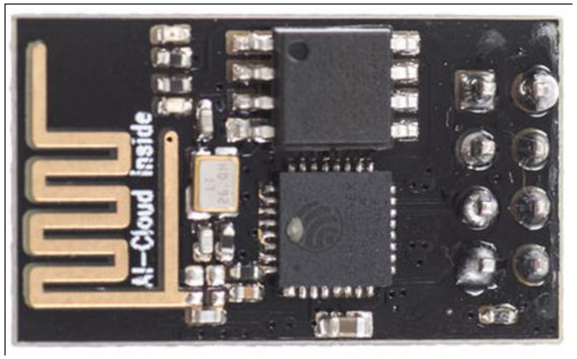
    - SparkFun ESP8266 Thing

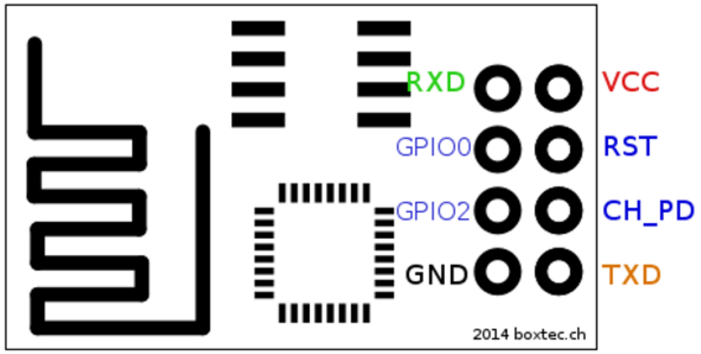


    - nodeMCU Devkit

ESP-01

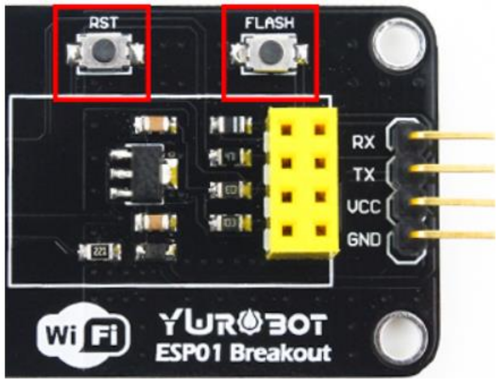
- ESP-01
  - By AI-Thinker
  - Integrated with 3 dBi PCB antenna
  - Integrated with **1MB** SPI Flash memory
  - 3.3V power
  - Pin map
    - GND, VCC (3.3V)
    - TXD, RXD (UART)
    - GPIO0, GPIO2 (internally pull-up)
    - RST: reset, active low
    - CH_PD: chip enable pin, active high

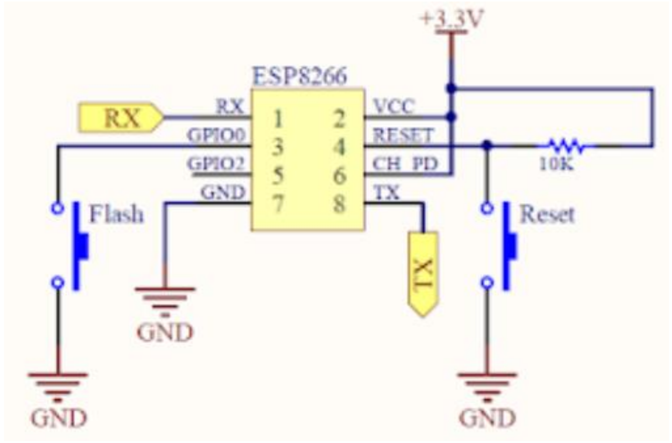


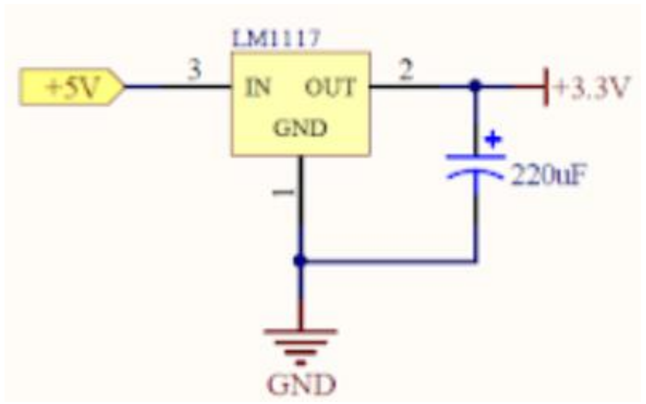


ESP-01 Breakout Board

- 외형, 연결 방법 및 주요 회로 부분





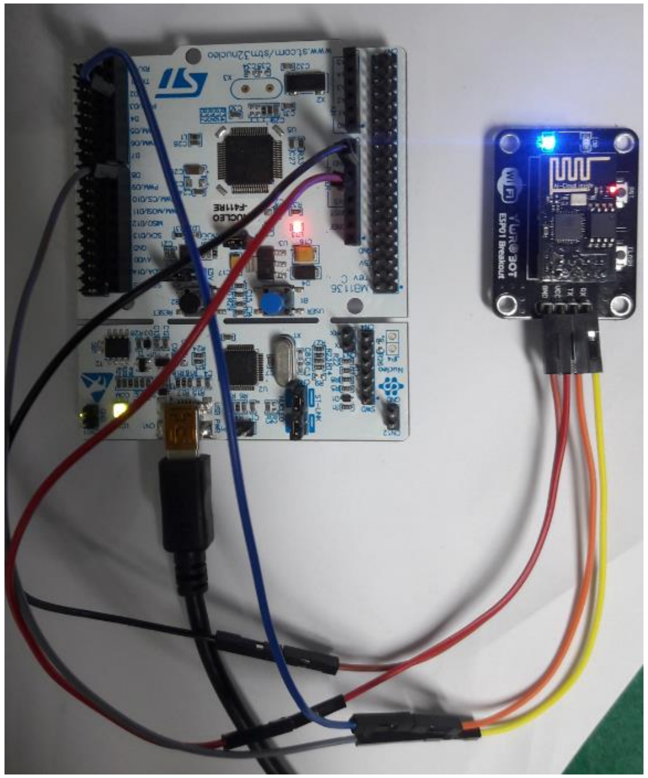


Firmware Update

- Firmwares for ESP-01
  - Two major firmwares: from **Espressif** and from **AT Thinker**
  - Theses two firmwares are not compatible.
- **For mbed-os, the firmware of Espressif** is supported.
- The firmware update process
  - https://os.mbed.com/teams/ESP8266/wiki/Firmware-Update

Lab13-1: Firmware Update

- Exp: Check which firmware is installed.
  - Pin connections (Nucleo-F411 ⇔ ESP-01 break board)
    - F411 5V ⇔ VCC (ESP-01 break board)
    - F411 GND ⇔ GND (ESP-01 break board)
    - F411 CN9-RX/D2 ⇔ TX (ESP-01 break board)
    - F411 CN9-TX/D8 ⇔ RX (ESP-01 break board)



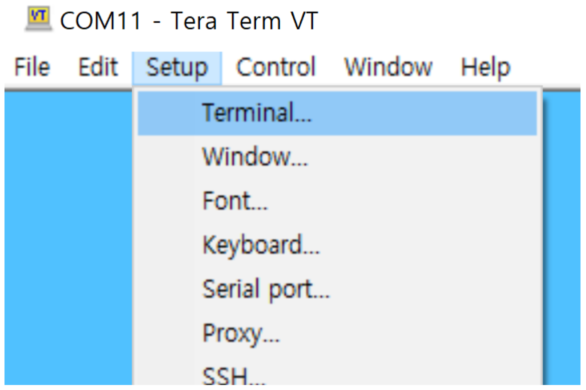
Lab13-1: Firmware Update

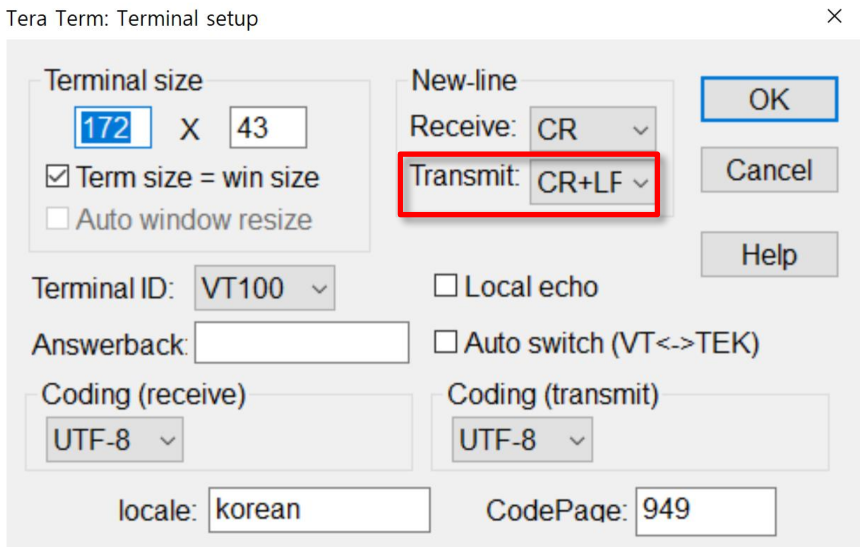
- A sample code: passthrough program (PC ⇔ ESP-01)

```
#include "mbed.h"

UnbufferedSerial pc(CONSOLE_TX, CONSOLE_RX, 115200);
UnbufferedSerial esp8266(ARDUINO_UNO_D8, ARDUINO_UNO_D2, 115200);

// main() runs in its own thread in the OS
int main() {
    char ch;

    while (true) {
        if (pc.readable()) {
            pc.read(&ch, 1);
            esp8266.write(&ch, 1);
        }

        if (esp8266.readable()) {
            esp8266.read(&ch, 1);
            pc.write(&ch, 1);
        }
    }
}
```

Lab13-1: Firmware Update

- TeraTerm Setup
  - Terminal
    - Transmit: CR -> CR+LF





Tera Term: Terminal setup

Terminal size: 172 X 43

☑ Term size = win size

☐ Auto window resize

New-line

Receive: CR

Transmit: CR+LF

Terminal ID: VT100

Answerback:

☐ Local echo

☐ Auto switch (VT<->TEK)

Coding (receive): UTF-8

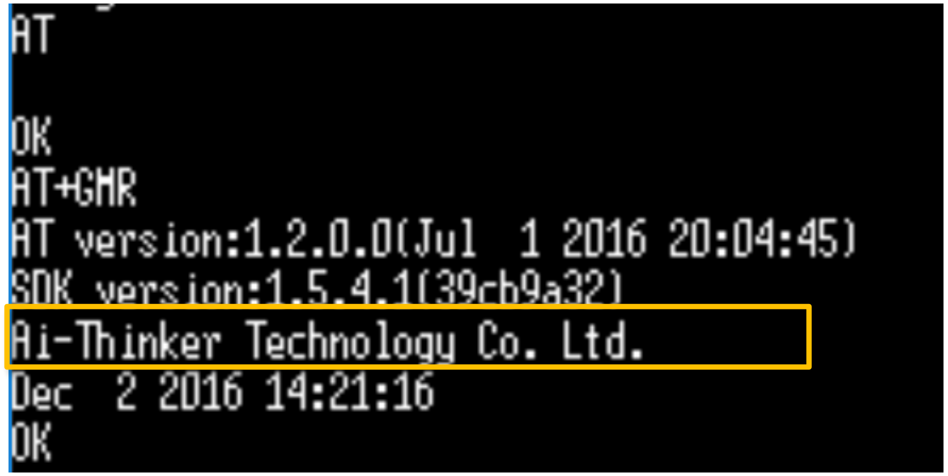
Coding (transmit): UTF-8

locale: korean

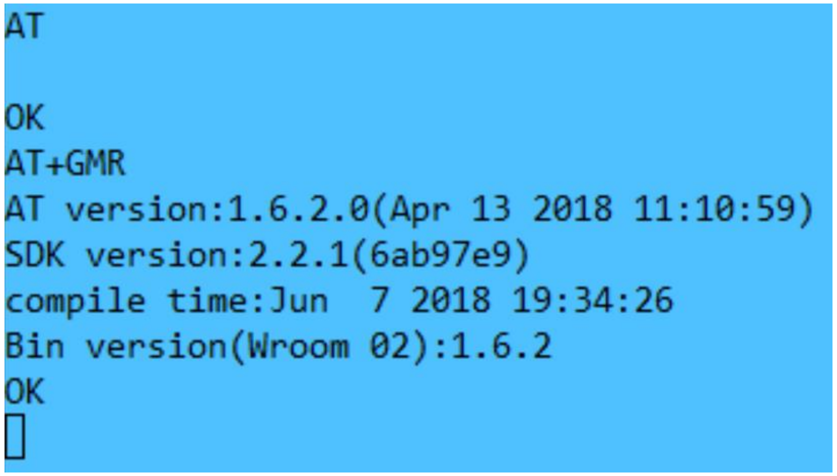
CodePage: 949

Lab13-1: Firmware Update

- Exp: Check which firmware is installed.
  - For a command **AT**
  - For a command **AT+GMR** (view version Info)



```
AT

OK
AT+GMR
AT version:1.2.0.0(Jul  1 2016 20:04:45)
SDK version:1.5.4.1(39cb9a32)
Ai-Thinker Technology Co. Ltd.
Dec  2 2016 14:21:16
OK
```



```
AT

OK
AT+GMR
AT version:1.6.2.0(Apr 13 2018 11:10:59)
SDK version:2.2.1(6ab97e9)
compile time:Jun  7 2018 19:34:26
Bin version(Wroom 02):1.6.2
OK
```

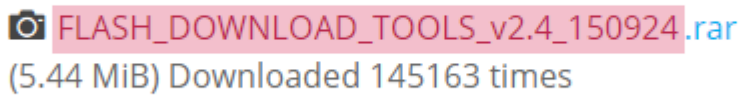
  - Close TeraTerm!!

Lab13-1: Firmware Update

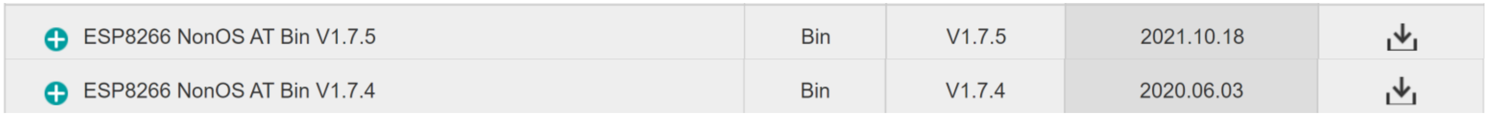
- ESP8266 WiFi driver for Mbed OS ('21.5.)
  - This driver supports AT firmware versions **1.3.0 to 1.7.0**.
  - We advise updating the AT firmware to at least version 1.7.0.

Lab13-1: Firmware Update

- Flash download tool
  - Download the flash download tool (v2.4)
    - https://bbs.espressif.com/viewtopic.php?f=57&t=433&hilit=FLASH_DOWNLOAD_TOOLS_v2.4_150924


FLASH_DOWNLOAD_TOOLS_v2.4_150924.rar
(5.44 MiB) Downloaded 145163 times

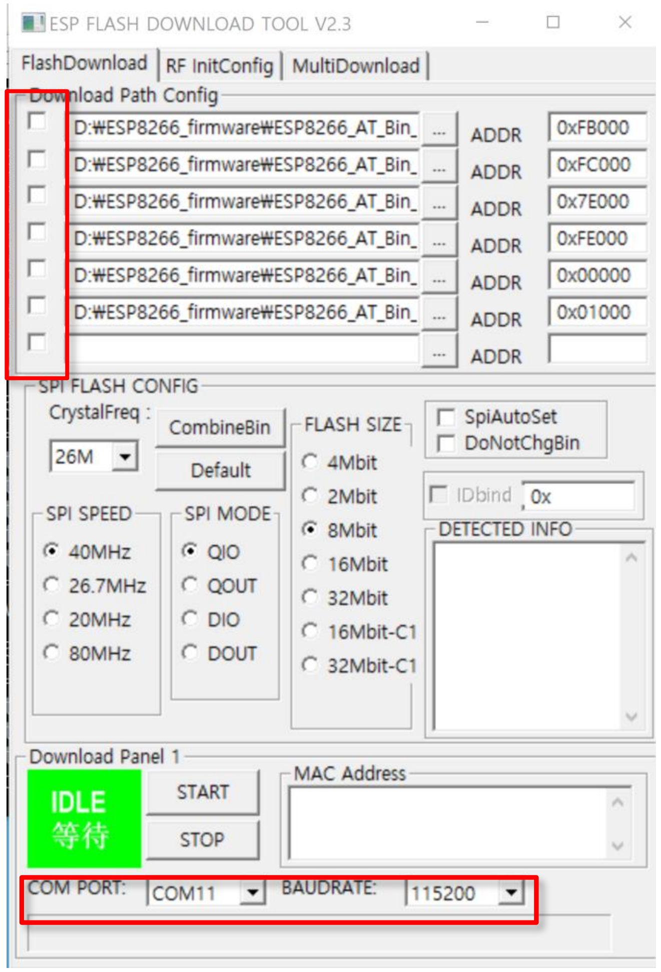
- Firmware
  - Download a firmware
    - https://www.espressif.com/en/products/sdks/esp-at/resource



ESP8266 NonOS AT Bin V1.7.5	Bin	V1.7.5	2021.10.18	
ESP8266 NonOS AT Bin V1.7.4	Bin	V1.7.4	2020.06.03	

Lab13-1: Firmware Update

- Run ESP Flash download tool
- Uncheck the boxes in “Download Path Config”
- Select the suitable “COM PORT”
  - PC ⇔ Nucleo-F411
- Select BAUDRATE = 115200



ESP FLASH DOWNLOAD TOOL V2.3

FlashDownload | RF InitConfig | MultiDownload

Download Path Config

☐	D:₩ESP8266_firmware₩ESP8266_AT_Bin_	ADDR	0xFB000
☐	D:₩ESP8266_firmware₩ESP8266_AT_Bin_	ADDR	0xFC000
☐	D:₩ESP8266_firmware₩ESP8266_AT_Bin_	ADDR	0x7E000
☐	D:₩ESP8266_firmware₩ESP8266_AT_Bin_	ADDR	0xFE000
☐	D:₩ESP8266_firmware₩ESP8266_AT_Bin_	ADDR	0x00000
☐	D:₩ESP8266_firmware₩ESP8266_AT_Bin_	ADDR	0x01000
☐		ADDR	

SPI FLASH CONFIG

CrystalFreq : 26M

CombineBin

Default

SPI SPEED: [✓ 40MHz] 26.7MHz 20MHz 80MHz

SPI MODE: [✓ QIO] QOUT DIO DOUT

FLASH SIZE: 4Mbit 2Mbit [✓ 8Mbit] 16Mbit 32Mbit 16Mbit-C1 32Mbit-C1

☐ SpiAutoSet

☐ DoNotChgBin

☐ IDbind 0x

DETECTED INFO

Download Panel 1

IDLE 等待

START

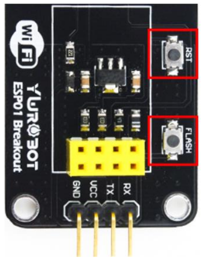
STOP

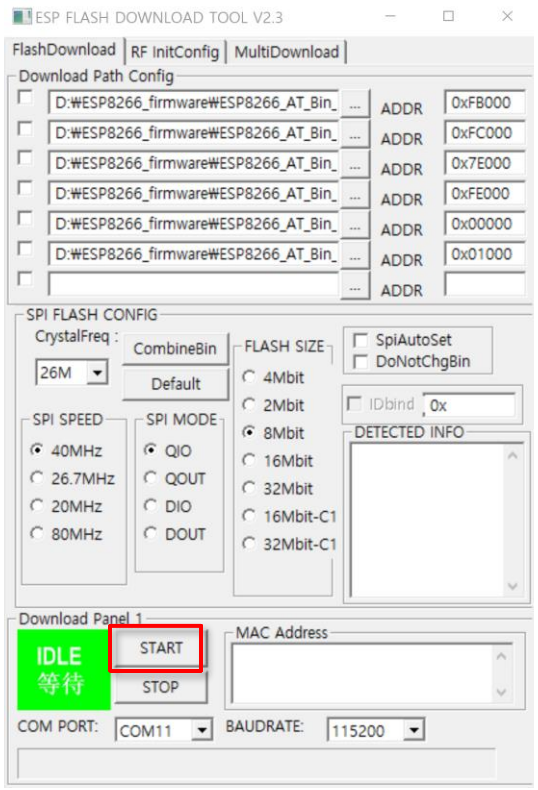
MAC Address

COM PORT: COM11 BAUDRATE: 115200

Lab13-1: Firmware Update

- **Press and hold the RST button** on ESP01 breakout board.
- **Press and hold the FLASH button** on ESP01 breakout board.
- **Release the RST button.**
- **Release the FLASH button.**
- Click the **START** button at the lower left portion of the download tool window.





ESP FLASH DOWNLOAD TOOL V2.3

FlashDownload | RF InitConfig | MultiDownload

Download Path Config

☐ D:₩ESP8266_firmware₩ESP8266_AT_Bin_	...	ADDR	0xFB000
☐ D:₩ESP8266_firmware₩ESP8266_AT_Bin_	...	ADDR	0xFC000
☐ D:₩ESP8266_firmware₩ESP8266_AT_Bin_	...	ADDR	0x7E000
☐ D:₩ESP8266_firmware₩ESP8266_AT_Bin_	...	ADDR	0xFE000
☐ D:₩ESP8266_firmware₩ESP8266_AT_Bin_	...	ADDR	0x00000
☐ D:₩ESP8266_firmware₩ESP8266_AT_Bin_	...	ADDR	0x01000
☐	...	ADDR	

SPI FLASH CONFIG

CrystalFreq : 26M | CombineBin | Default

FLASH SIZE: 4Mbit, 2Mbit, [● 8Mbit], 16Mbit, 32Mbit, 16Mbit-C1, 32Mbit-C1

☐ SpiAutoSet ☐ DoNotChgBin ☐ IDbind 0x

SPI SPEED: [● 40MHz], 26.7MHz, 20MHz, 80MHz

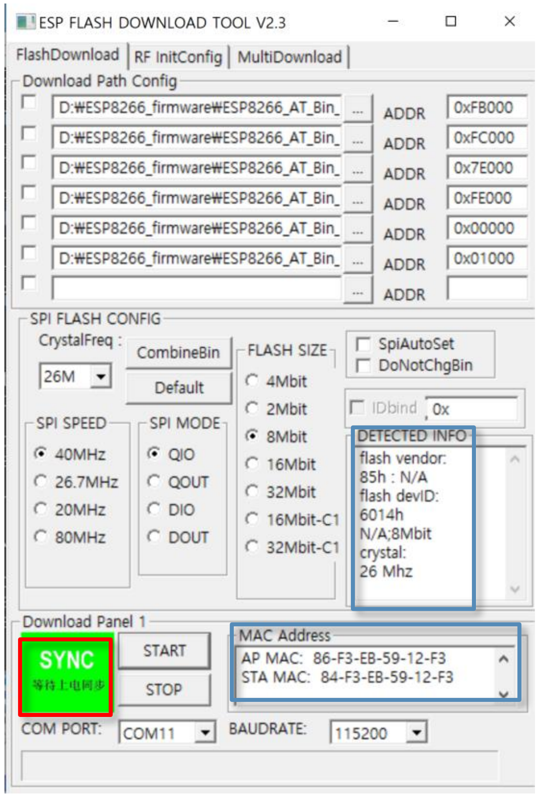
SPI MODE: [● QIO], QOUT, DIO, DOUT

DETECTED INFO

Download Panel 1

IDLE 等待 | START | STOP | MAC Address

COM PORT: COM11 BAUDRATE: 115200



ESP FLASH DOWNLOAD TOOL V2.3

FlashDownload | RF InitConfig | MultiDownload

Download Path Config

☐ D:₩ESP8266_firmware₩ESP8266_AT_Bin_	...	ADDR	0xFB000
☐ D:₩ESP8266_firmware₩ESP8266_AT_Bin_	...	ADDR	0xFC000
☐ D:₩ESP8266_firmware₩ESP8266_AT_Bin_	...	ADDR	0x7E000
☐ D:₩ESP8266_firmware₩ESP8266_AT_Bin_	...	ADDR	0xFE000
☐ D:₩ESP8266_firmware₩ESP8266_AT_Bin_	...	ADDR	0x00000
☐ D:₩ESP8266_firmware₩ESP8266_AT_Bin_	...	ADDR	0x01000
☐	...	ADDR	

SPI FLASH CONFIG

CrystalFreq : 26M | CombineBin | Default

FLASH SIZE: 4Mbit, 2Mbit, [● 8Mbit], 16Mbit, 32Mbit, 16Mbit-C1, 32Mbit-C1

☐ SpiAutoSet ☐ DoNotChgBin ☐ IDbind 0x

SPI SPEED: [● 40MHz], 26.7MHz, 20MHz, 80MHz

SPI MODE: [● QIO], QOUT, DIO, DOUT

DETECTED INFO

```
flash vendor:
85h : N/A
flash devID:
6014h
N/A;8Mbit
crystal:
26 Mhz
```

Download Panel 1

SYNC 等待上电同步 | START | STOP

MAC Address

```
AP MAC:  86-F3-EB-59-12-F3
STA MAC:  84-F3-EB-59-12-F3
```

COM PORT: COM11 BAUDRATE: 115200

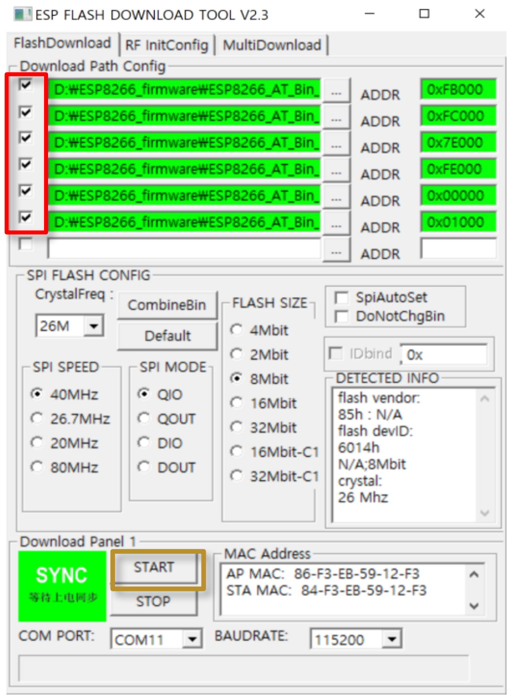
Lab13-1: Firmware Update

- Select the binary files to be downloaded from their storage locations as follows
  - ESP8266_NonOS_AT_Bin_V1.7.5\bin

Binary File Name	ESP-01 Flash Address
blank.bin	0xFB000
esp_init_data_default_v08.bin	0xFC000
blank.bin	0x7E000
blank.bin	0xFE000
boot_v1.7.bin	0x00000
user1.1024.new.2.bin	0x01000

  - at\512+512\user1.1024.new.2.bin

- Check the boxes in “Download Path Config”
- Click the **START** button at the lower left portion of the download tool window.



Download Path Config

[✓]	D:₩ESP8266_firmware₩ESP8266_AT_Bin_	ADDR	0xFB000
[✓]	D:₩ESP8266_firmware₩ESP8266_AT_Bin_	ADDR	0xFC000
[✓]	D:₩ESP8266_firmware₩ESP8266_AT_Bin_	ADDR	0x7E000
[✓]	D:₩ESP8266_firmware₩ESP8266_AT_Bin_	ADDR	0xFE000
[✓]	D:₩ESP8266_firmware₩ESP8266_AT_Bin_	ADDR	0x00000
[✓]	D:₩ESP8266_firmware₩ESP8266_AT_Bin_	ADDR	0x01000
[ ]		ADDR	

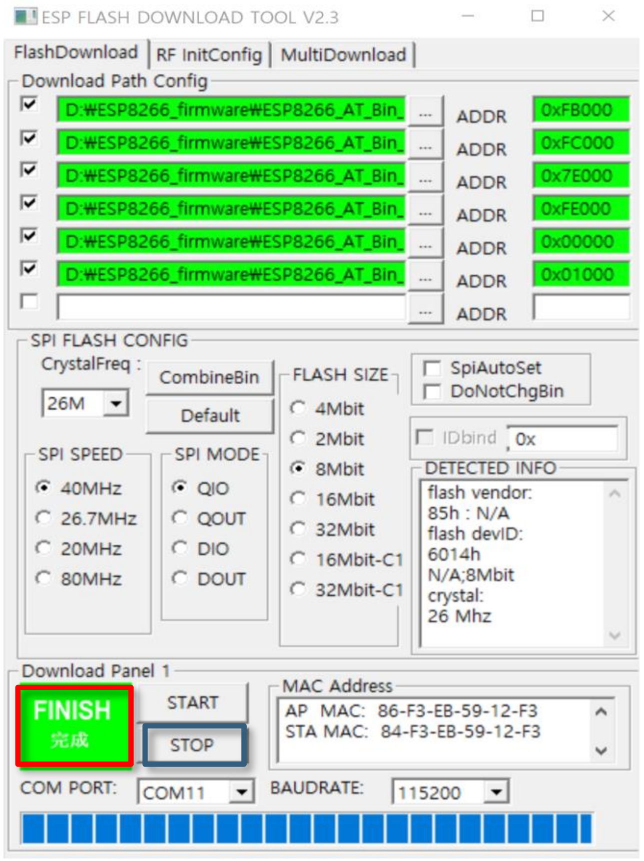
SPI FLASH CONFIG
CrystalFreq : 26M
SPI SPEED: [✓ 40MHz] 26.7MHz 20MHz 80MHz
SPI MODE: [✓ QIO] QOUT DIO DOUT
FLASH SIZE: 4Mbit 2Mbit [✓ 8Mbit] 16Mbit 32Mbit 16Mbit-C1 32Mbit-C1
[ ] SpiAutoSet
[ ] DoNotChgBin
[ ] IDbind 0x

DETECTED INFO
```
flash vendor:
85h : N/A
flash devID:
6014h
N/A;8Mbit
crystal:
26 Mhz
```

Download Panel 1
MAC Address
```
AP MAC:  86-F3-EB-59-12-F3
STA MAC:  84-F3-EB-59-12-F3
```
COM PORT: COM11 BAUDRATE: 115200

Lab13-1: Firmware Update

- After download is finished, click **STOP** button and then close the download tool.



✓ D:₩ESP8266_firmware₩ESP8266_AT_Bin_	ADDR	0xFB000
✓ D:₩ESP8266_firmware₩ESP8266_AT_Bin_	ADDR	0xFC000
✓ D:₩ESP8266_firmware₩ESP8266_AT_Bin_	ADDR	0x7E000
✓ D:₩ESP8266_firmware₩ESP8266_AT_Bin_	ADDR	0xFE000
✓ D:₩ESP8266_firmware₩ESP8266_AT_Bin_	ADDR	0x00000
✓ D:₩ESP8266_firmware₩ESP8266_AT_Bin_	ADDR	0x01000
	ADDR	

SPI FLASH CONFIG

CrystalFreq : 26M

FLASH SIZE: 4Mbit, 2Mbit, [✓ 8Mbit], 16Mbit, 32Mbit, 16Mbit-C1, 32Mbit-C1

SpiAutoSet ☐ DoNotChgBin ☐ IDbind ☐ 0x

SPI SPEED: [✓ 40MHz], 26.7MHz, 20MHz, 80MHz

SPI MODE: [✓ QIO], QOUT, DIO, DOUT

DETECTED INFO

```
flash vendor:
85h : N/A
flash devID:
6014h
N/A;8Mbit
crystal:
26 Mhz
```

Download Panel 1

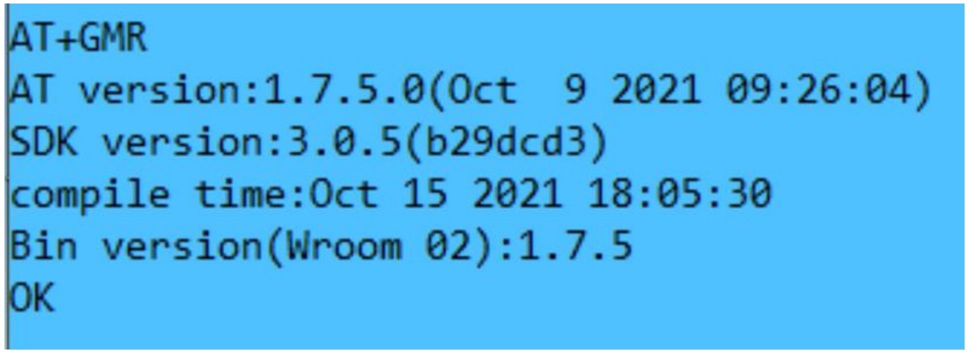
MAC Address

```
AP  MAC:  86-F3-EB-59-12-F3
STA MAC:  84-F3-EB-59-12-F3
```

COM PORT: COM11 BAUDRATE: 115200

Lab13-1: Firmware Update

- Press RST button on ESP01 breakout board.
- Open the TeraTerm.
  - Setup-> Terminal: Transmit: CR+LF
- On the TeraTerm
  - Execute the following AT commands
    - AT
    - AT+GMR



```
AT+GMR
AT version:1.7.5.0(Oct  9 2021 09:26:04)
SDK version:3.0.5(b29dcd3)
compile time:Oct 15 2021 18:05:30
Bin version(Wroom 02):1.7.5
OK
```

TCP AND UDP CLIENT USING AT COMMANDS - WITH ESP-01 -

Handong university

Jong-won Lee

2022.05.18

TCP & UDP Client using AT Command

Passthrough Program 2

- A sample program for TCP & UDP using AT commands

```
#include "mbed.h"

UnbufferedSerial pc(CONSOLE_TX, CONSOLE_RX, 230400);
UnbufferedSerial wifi(ARDUINO_UNO_D8, ARDUINO_UNO_D2, 115200);

char buffer[80];

// main() runs in its own thread in the OS
int main() {
    char ch;

    sprintf(buffer, "\r\n      ***** key <---> WiFi *****\r\n");
    pc.write(buffer, strlen(buffer));
    sprintf(buffer, "\r\n TCP and UDP Operation Using AT Commands\r\n");
    pc.write(buffer, strlen(buffer));
```

Passthrough Program 2

- A sample program for TCP & UDP using AT commands

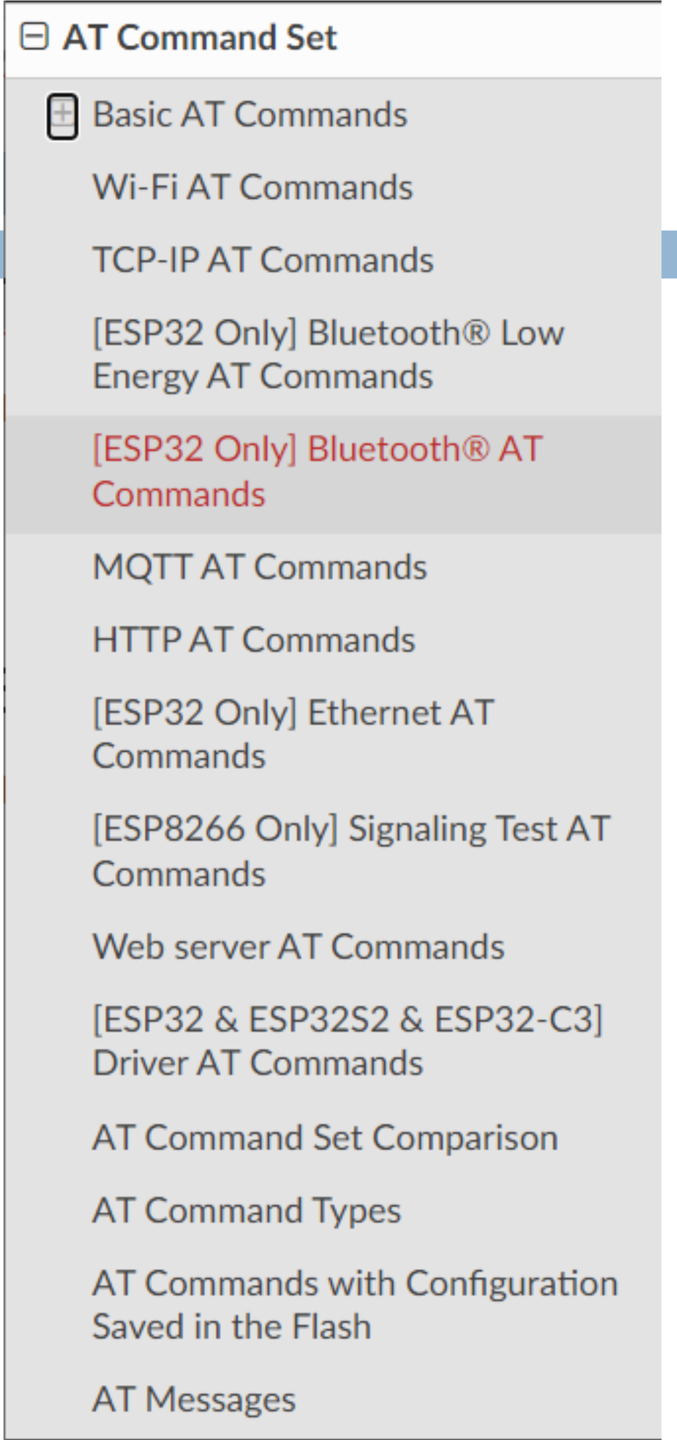
```
  while (true) {
      if (pc.readable()) {
          pc.read(&ch, 1);
          if (ch == 0x03) {  //ctrl+c to exit from transparent mode ("+++")
                             // that is, Back to a normal mode)
              wifi.write("+++", 3);
          } else {
              wifi.write(&ch, 1);
          }
      }

      if (wifi.readable()) {
          wifi.read(&ch, 1);
          pc.write(&ch, 1);
      }
  }
}
```

Passthrough Program

- TeraTerm configuration
  - Setup-> Terminal: Transmit: CR+LF

- ESP8266 AT command documents
  - https://docs.espressif.com/projects/esp-at/en/release-v2.2.0.0_esp8266/AT_Command_Set/index.html

The document gives some examples of the ESP8266 AT commands that are based on ESP8266_NONOS_SDK. However, this SDK is no longer updated, **so it is recommended to use *[ESP8266 IDF AT Bin]()* for new product designs.**

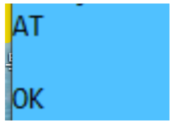


AT Command Set

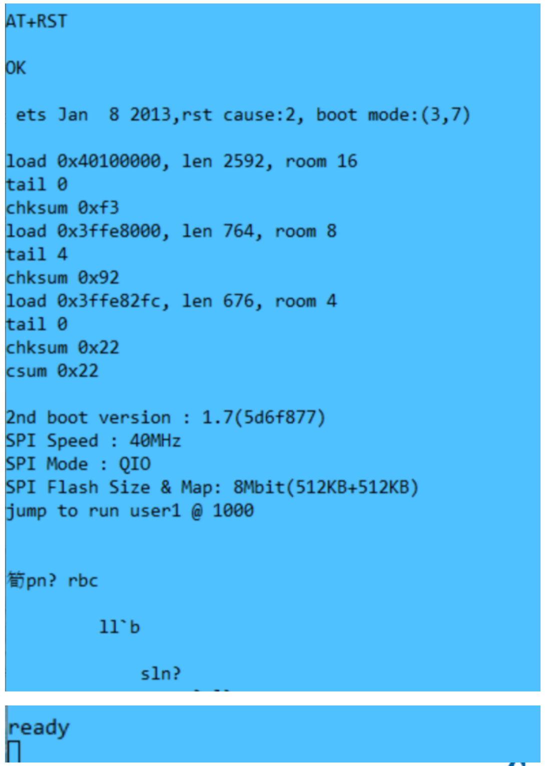
- Basic AT Commands
- Wi-Fi AT Commands
- TCP-IP AT Commands
- [ESP32 Only] Bluetooth® Low Energy AT Commands
- [ESP32 Only] Bluetooth® AT Commands
- MQTT AT Commands
- HTTP AT Commands
- [ESP32 Only] Ethernet AT Commands
- [ESP8266 Only] Signaling Test AT Commands
- Web server AT Commands
- [ESP32 & ESP32S2 & ESP32-C3] Driver AT Commands
- AT Command Set Comparison
- AT Command Types
- AT Commands with Configuration Saved in the Flash
- AT Messages

AT Commands

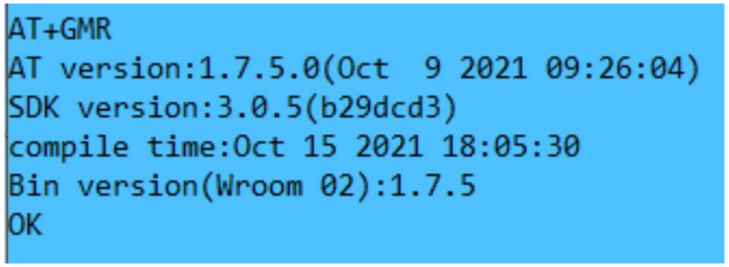
- Basic AT commands
  - **AT** : Test command
    - Response: <cr><lf>OK<cr><lf>
  - **AT+RST**: Restart module
    - Response: <cr><lf>OK<cr><lf> …
  - **AT+GMR** : view version info
    - Response: version info, OK



```
AT

OK
```



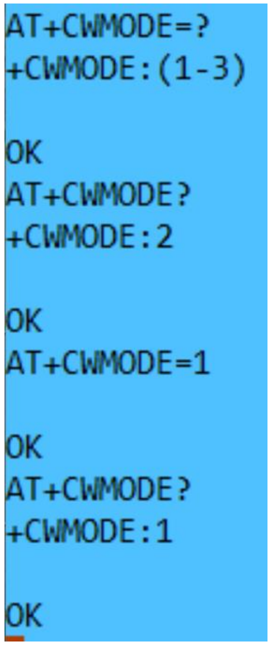
```
AT+RST

OK

 ets Jan  8 2013,rst cause:2, boot mode:(3,7)

load 0x40100000, len 2592, room 16
tail 0
chksum 0xf3
load 0x3ffe8000, len 764, room 8
tail 4
chksum 0x92
load 0x3ffe82fc, len 676, room 4
tail 0
chksum 0x22
csum 0x22

2nd boot version : 1.7(5d6f877)
SPI Speed : 40MHz
SPI Mode : QIO
SPI Flash Size & Map: 8Mbit(512KB+512KB)
jump to run user1 @ 1000


䈏pn? rbc

         ll`b

             sln?

ready
```



```
AT+GMR
AT version:1.7.5.0(Oct  9 2021 09:26:04)
SDK version:3.0.5(b29dcd3)
compile time:Oct 15 2021 18:05:30
Bin version(Wroom 02):1.7.5
OK
```

AT Commands

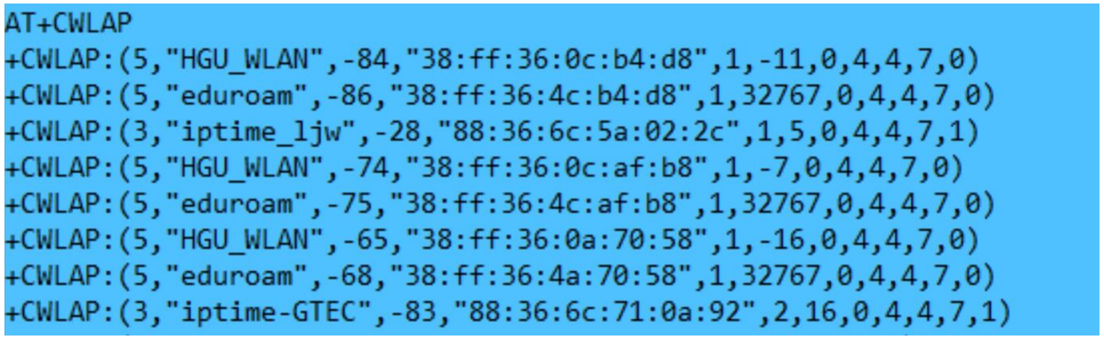
- WiFi AT commands
  - **AT+CWMODE** : WiFi mode (station, AP, station+AP)
    - Ex: AT+CWMODE=? : list valid modes (1-3)
    - Ex.: AT+CWMODE? : Query current WiFI mode
    - Ex.: AT+CWMODE=mode ; set WiFi mode to station mode
    - mode:
      - 1 = Station mode
      - 2 = softAP mode
      - 3 = softAP + Station mode



```
AT+CWMODE=?
+CWMODE:(1-3)

OK
AT+CWMODE?
+CWMODE:2

OK
AT+CWMODE=1

OK
AT+CWMODE?
+CWMODE:1

OK
```

AT Commands

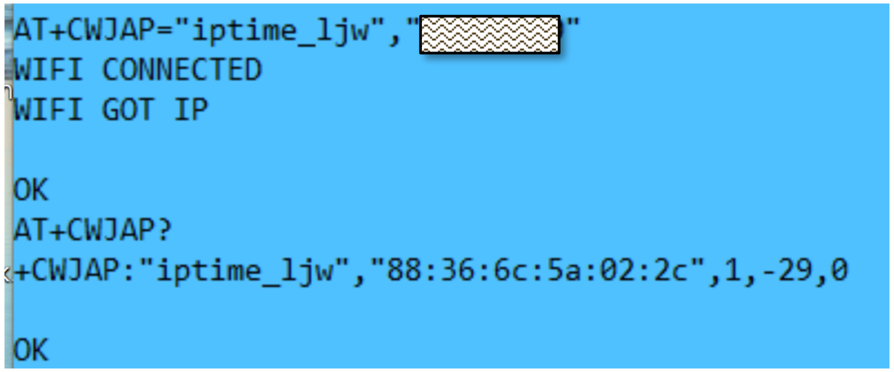
- WiFi AT commands
  - **AT+CWLAP** : list available APs



```
AT+CWLAP
+CWLAP:(5,"HGU_WLAN",-84,"38:ff:36:0c:b4:d8",1,-11,0,4,4,7,0)
+CWLAP:(5,"eduroam",-86,"38:ff:36:4c:b4:d8",1,32767,0,4,4,7,0)
+CWLAP:(3,"iptime_ljw",-28,"88:36:6c:5a:02:2c",1,5,0,4,4,7,1)
+CWLAP:(5,"HGU_WLAN",-74,"38:ff:36:0c:af:b8",1,-7,0,4,4,7,0)
+CWLAP:(5,"eduroam",-75,"38:ff:36:4c:af:b8",1,32767,0,4,4,7,0)
+CWLAP:(5,"HGU_WLAN",-65,"38:ff:36:0a:70:58",1,-16,0,4,4,7,0)
+CWLAP:(5,"eduroam",-68,"38:ff:36:4a:70:58",1,32767,0,4,4,7,0)
+CWLAP:(3,"iptime-GTEC",-83,"88:36:6c:71:0a:92",2,16,0,4,4,7,1)
```

AT Commands

- WiFi AT commands
  - **AT+CWJAP** : connect to an AP
    - AT+CWJAP=ssid,passwd : connect a SSID with the password
      - Ex.: AT+CWJAP="iptime_ljw","1234test"
    - AT+CWJAP? ; query which AP is connected

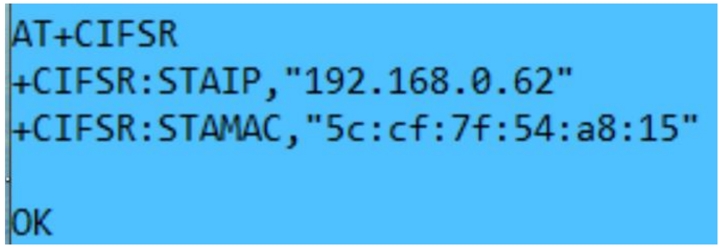


```
AT+CWJAP="iptime_ljw","[redacted]"
WIFI CONNECTED
WIFI GOT IP

OK
AT+CWJAP?
+CWJAP:"iptime_ljw","88:36:6c:5a:02:2c",1,-29,0

OK
```

  - **AT+CWQAP** : disconnect from the connected AP

AT Commands

- WiFi AT commands
  - **AT+CIFSR** : get local IP and MAC address



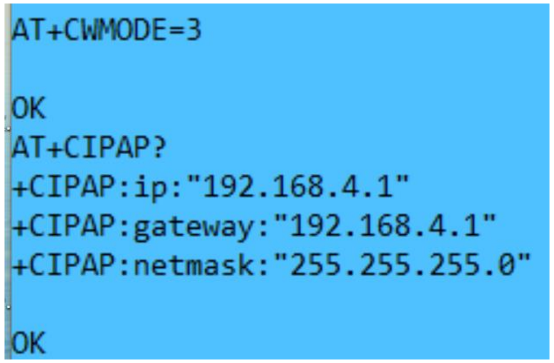
```
AT+CIFSR
+CIFSR:STAIP,"192.168.0.62"
+CIFSR:STAMAC,"5c:cf:7f:54:a8:15"

OK
```

AT Commands

- WiFi AT commands
  - **AT+CWSAP** : configuration of softAP mode
    - AT+CWSAP=ssid,pwd,ch,ecn
      - ssid: string, softAP SSID
      - pwd: string, password
      - ch: WiFi channel ID
      - ecn: security mode
        - 0 = open
        - 2 = WPA_PSK
        - 3 = WPA2_PSK
        - 4 = WPA_WPA2_PSK
    - Ex.: AT+CWSAP="esp_ap","1234test",5,3
  - **AT+CWLIF** : list clients connected to its softAP

AT Commands

- WiFi AT commands
  - **AT+CIPSTA** : set IP address of the station
  - **AT+CIPAP** : set IP address of the softAP
    - Ex.: AT+CIPAP="192.168.0.1"



```
AT+CWMODE=3

OK
AT+CIPAP?
+CIPAP:ip:"192.168.4.1"
+CIPAP:gateway:"192.168.4.1"
+CIPAP:netmask:"255.255.255.0"

OK
```

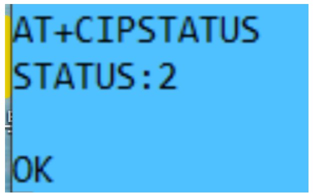
  - **AT+CIPSTAMAC** : set MAC address of the station
  - **AT+CIPAPMAC** : set MAC address of the softAP
    - Ex.: AT+CIPSTAMAC="2c:aa:35:97:d4:7b"

AT Commands

- WiFi AT commands
  - **AT+CIPSTATUS** : information about connection
    - Response:

```
STATUS: status
+CIPSTATUS: id , type , addr , port , tetype
OK
```

      - Parameters
        - Status: 2: Got IP, 3: Connected, 4: Disconnected
        - Id: id of the connection (0~4), for multi-connect
        - Type: String, “TCP” or “UDP”
        - Port: port number
        - Tetype: 0 = ESP8266 runs as a client,
          1 = ESP8266 runs as a server



```
AT+CIPSTATUS
STATUS:2

OK
```

TCP and UDP Client

TCP/UDP Client

- To connect a TCP or to register UDP server
  - **AT+CIPSTART=[id],type,addr,port**
    - Ex.: **AT+CIPSTART=0,"TCP","192.168.0.61",50000**
    - Ex.: **AT+CIPSTART="TCP","192.168.0.61",50000**
    - **Ex.: AT+CIPSTART=0,"UDP","192.168.0.61",50000**
    - **Ex.: AT+CIPSTART="UDP","192.168.0.61",50000**
    - **Ex.: AT+CIPSTART="UDP","192.168.0.61",50000,50001**
      - **Local port number: 50001**
    - **Parameters:**
      - Id: 0-4: id of connection
      - Type: string, "TCP" or "UDP"
      - Addr: string, server IP address
      - Port: server port number
  - To use multi-connection (id), set AT+CIPMUX=1

TCP/UDP Client

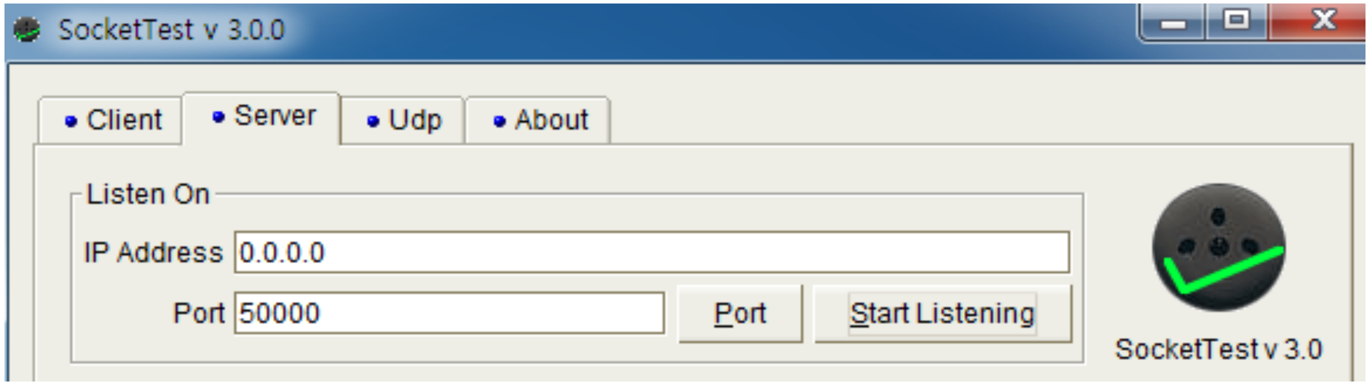
- To send data
  - **AT+CIPSEND=[id],length**
    - Ex.: **AT+CIPSEND=0,10**
    - Ex.: **AT+CIPSEND=10**
    - **Parameters:**
      - Id: 0-4: id of TX connection
      - Length: data length, MAX. 2048
- Display received data
  - +IPD, len:data : receive data from a single connection
  - +IPD, id, len:data : receive data from the connection of ID

TCP/UDP Client

- Close a connection
  - **AT+CIPCLOSE** : close the single TCP connection or unregister the single UDP server
  - **AT+CIPCLOSE=id** : close the TCP connection with id or unregister the UDP server with id.
- Normal mode vs. Transpararent mode
  - **AT+CIPMODE=0** : noraml mode
  - **AT+CIPMODE=1** : transparent mode (only in single connection)
  - **In transparent mode**, to send data, give the AT command, **"AT+CIPSEND" (with no specified length**). After that, all input data is transmitted to remote. To exit the transparent mode, three consecutive "+", that is, "+++"are given without time spacing between "+" characters.

Lab. 13-2: TCP Client

- Exp.: TCP Client test using SocketTest program
  - Get the program from https://sourceforge.net/projects/sockettest/
  - Run a TCP server on SocketTest program in PC

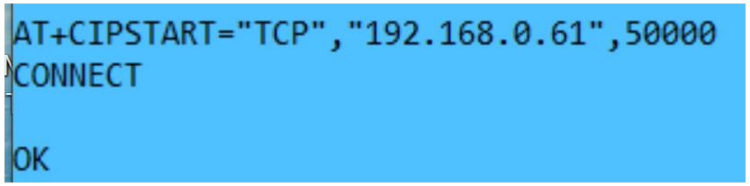


IP Address 0.0.0.0
Port 50000

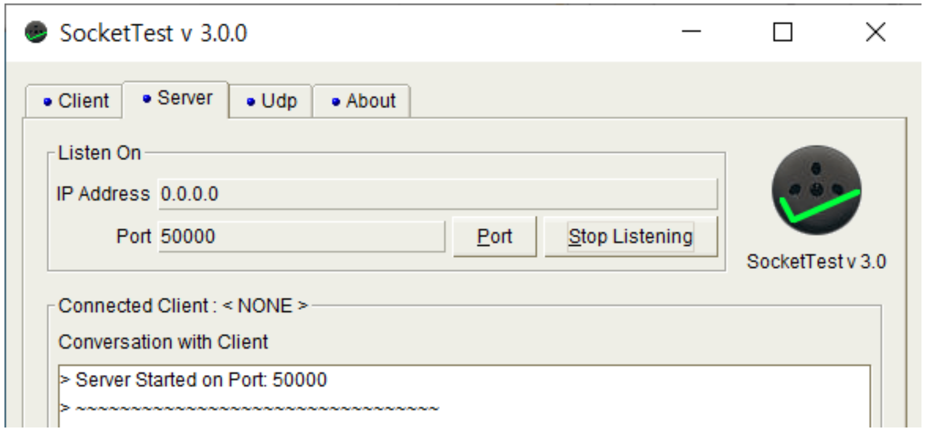
  - Click 'Start Listening'

TCP Client

- Exp.: TCP Client test using SocketTest program
  - Connect to the server using AT command
    - AT+CIPSTART="TCP","192.168.x.x",50000

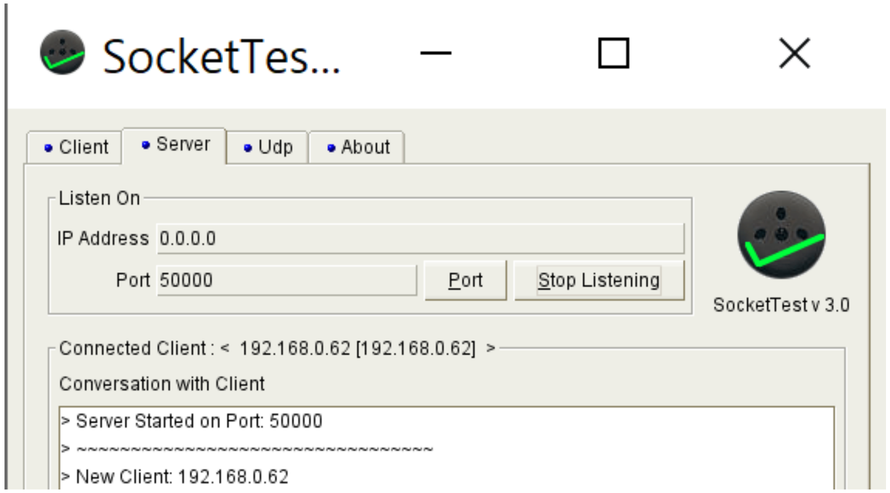


```
AT+CIPSTART="TCP","192.168.0.61",50000
CONNECT

OK
```



Listen On
IP Address 0.0.0.0
Port 50000
Connected Client : < NONE >
Conversation with Client

```
> Server Started on Port: 50000
> ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```



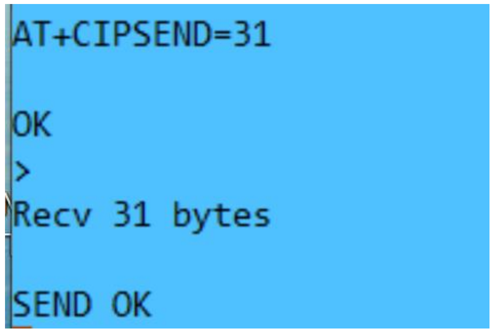
Listen On
IP Address 0.0.0.0
Port 50000
Connected Client : < 192.168.0.62 [192.168.0.62] >
Conversation with Client

```
> Server Started on Port: 50000
> ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
> New Client: 192.168.0.62
```

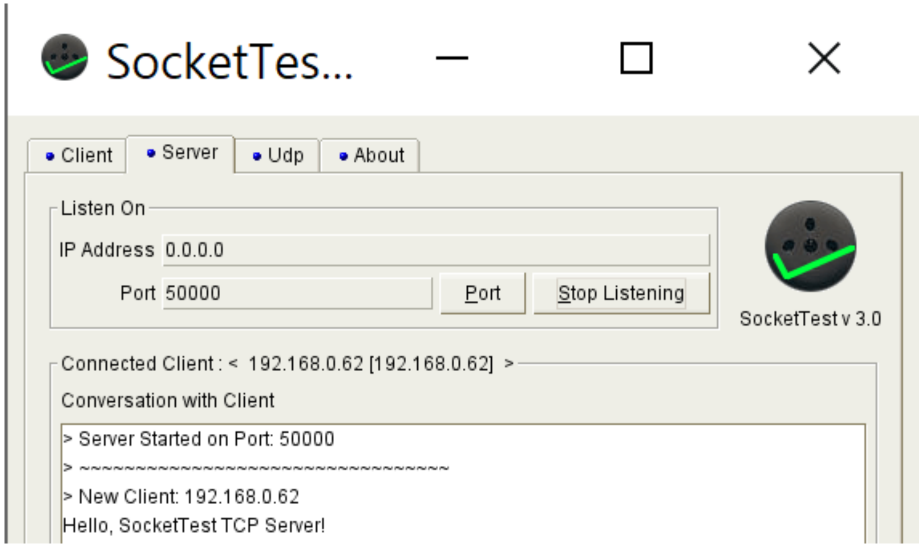
TCP Client

- Exp.: TCP Client test using SocketTest program
  - To send data to the server
    - AT+CIPSEND=31

      >Hello, SocketTest TCP Server!



```
AT+CIPSEND=31

OK
>
Recv 31 bytes

SEND OK
```



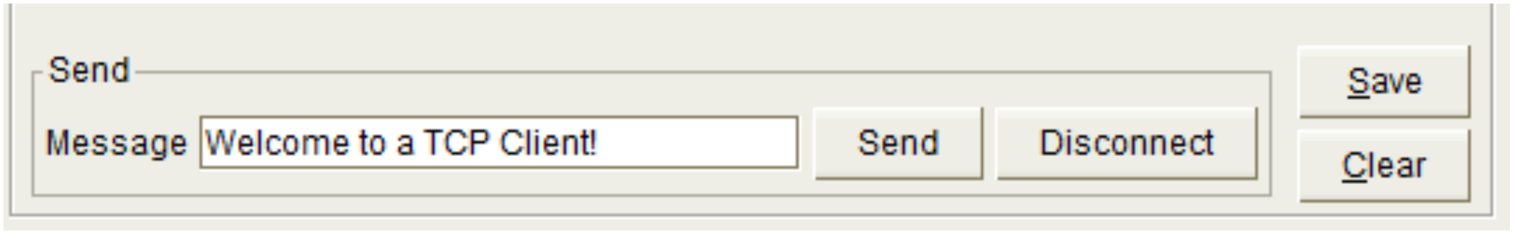
Listen On
IP Address 0.0.0.0
Port 50000

Connected Client : < 192.168.0.62 [192.168.0.62] >
Conversation with Client

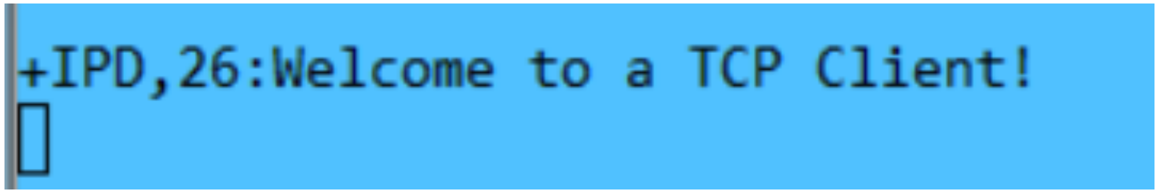
```
> Server Started on Port: 50000
> ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
> New Client: 192.168.0.62
Hello, SocketTest TCP Server!
```

TCP Client

- Exp.: TCP Client test using SocketTest program
  - To send data from the server
    - Enter data to Message box, and then click Send



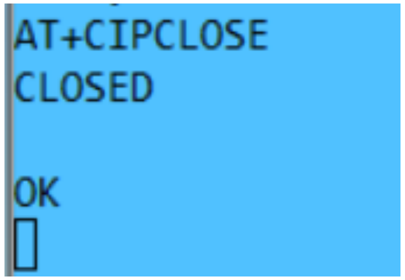
Message: Welcome to a TCP Client!



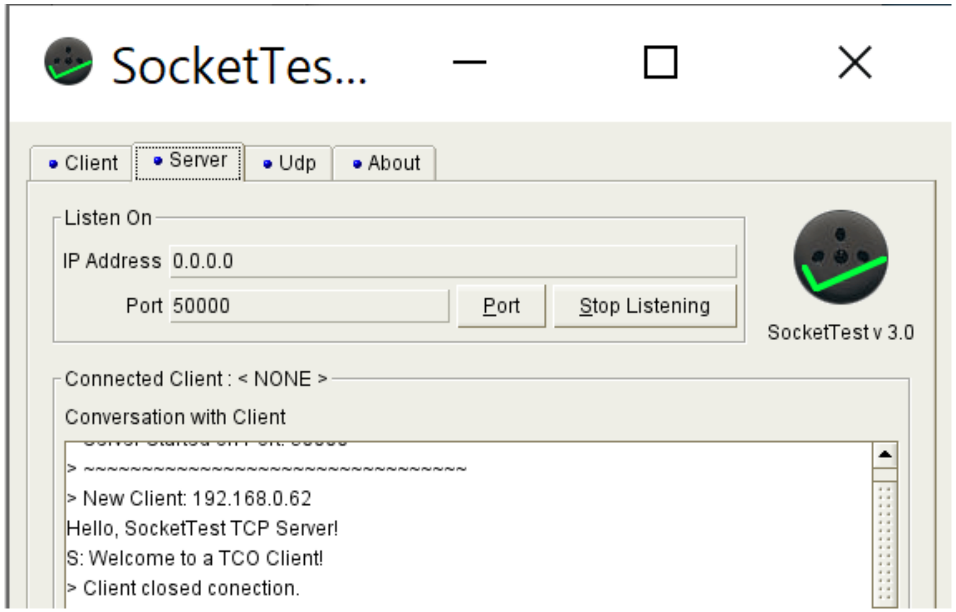
```
+IPD,26:Welcome to a TCP Client!
```

TCP Client

- Exp.: TCP Client test using SocketTest program
  - To close the TCP connection
    - AT+CIPCLOSE



```
AT+CIPCLOSE
CLOSED

OK
```



Listen On

IP Address 0.0.0.0

Port 50000

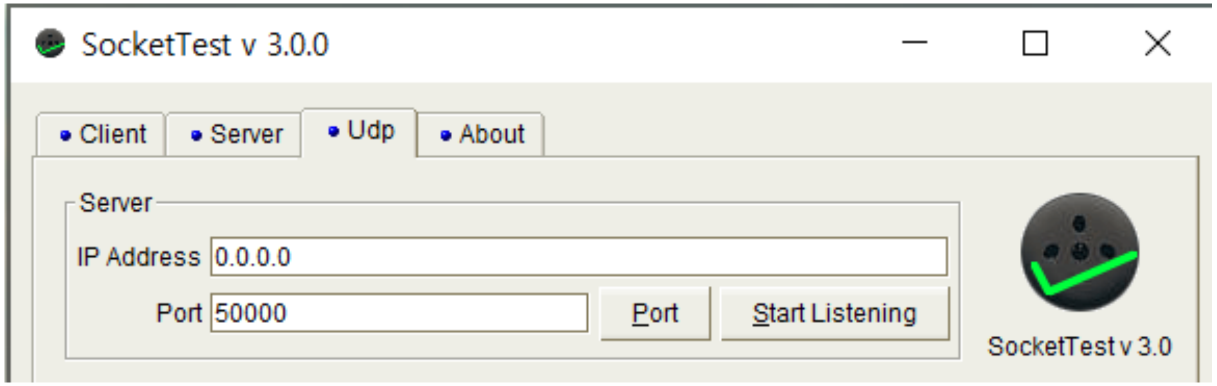
Connected Client : < NONE >

Conversation with Client

```
> ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
> New Client: 192.168.0.62
Hello, SocketTest TCP Server!
S: Welcome to a TCO Client!
> Client closed conection.
```

Lab. 13-3: UDP Client

- Exp.: UDP Client test using SocketTest program
  - Run a UDP server on SocketTest program in PC

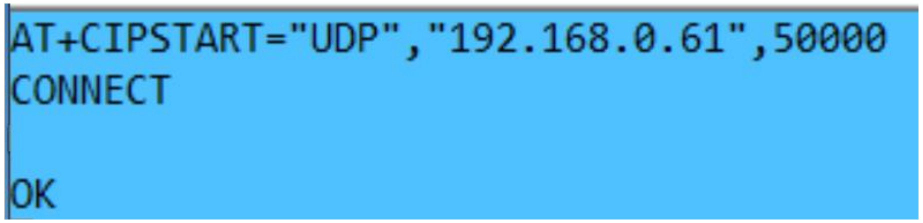


Server
IP Address 0.0.0.0
Port 50000

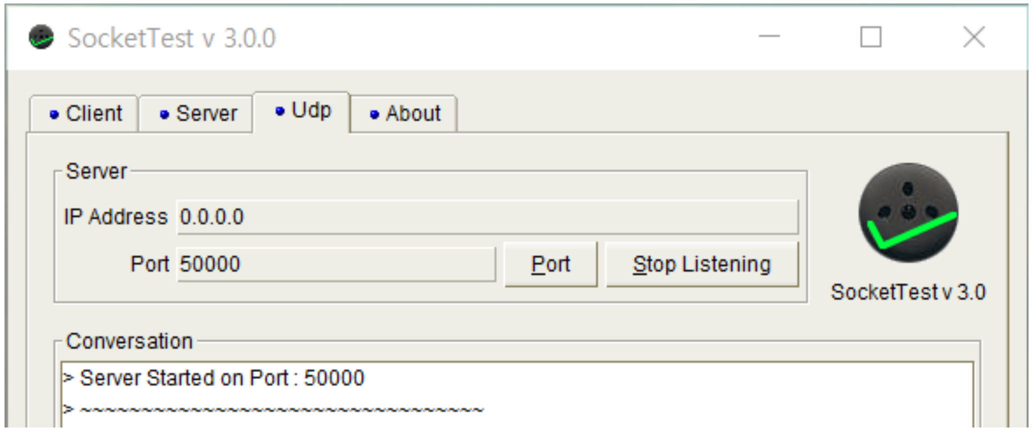
  - Click ‘Start Listening’

UDP Client

- Exp.: UDP Client test using SocketTest program
  - Register the UDP server using AT command
    - AT+CIPSTART="UDP","192.168.0.61",50000



```
AT+CIPSTART="UDP","192.168.0.61",50000
CONNECT

OK
```



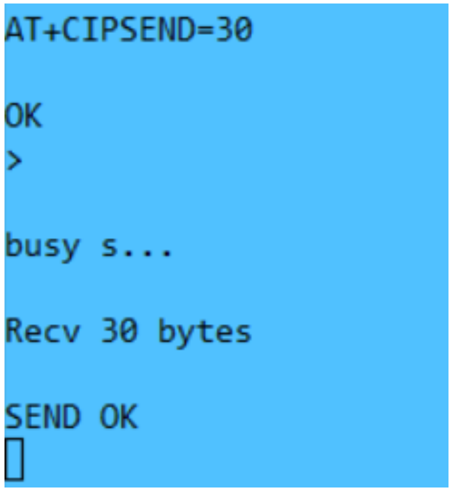
IP Address 0.0.0.0
Port 50000

```
> Server Started on Port : 50000
> ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

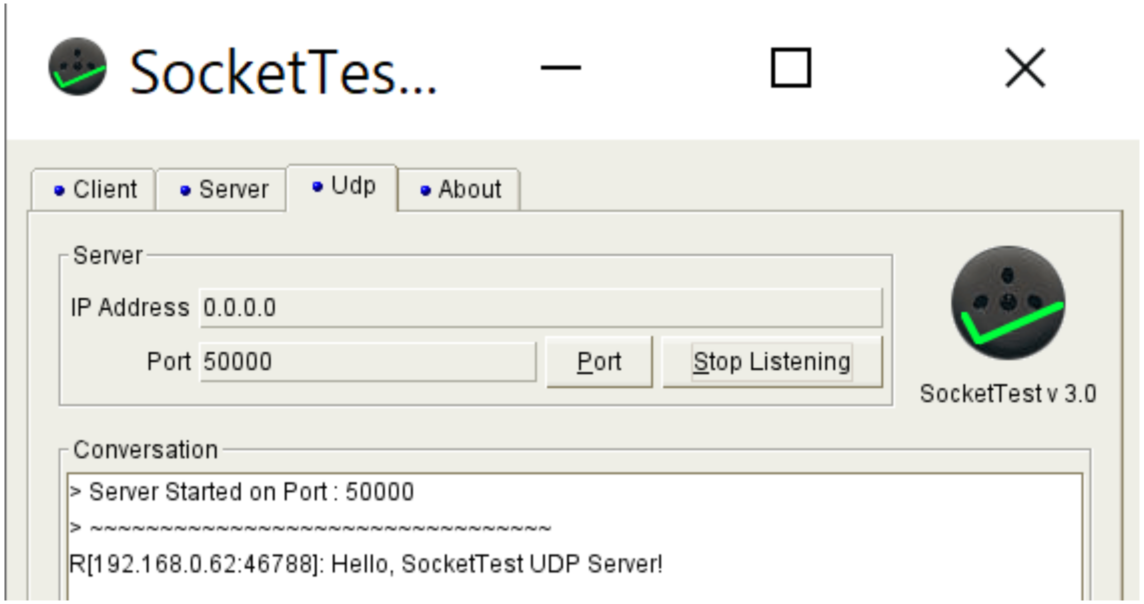
UDP Client

- Exp.: UDP Client test using SocketTest program
  - To send data to the server
    - AT+CIPSEND=30

      >Hello, SocketTest UDP Server!



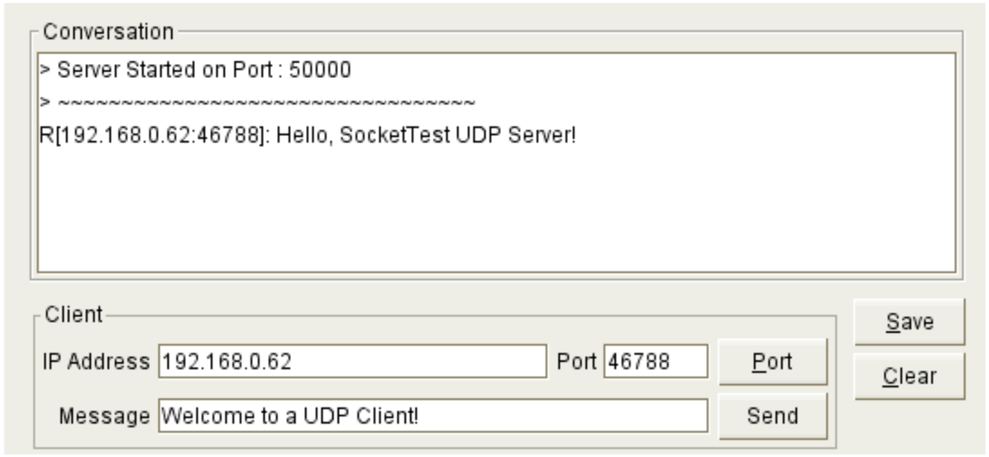
```
AT+CIPSEND=30

OK
>

busy s...

Recv 30 bytes

SEND OK
```



```
> Server Started on Port : 50000
> ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
R[192.168.0.62:46788]: Hello, SocketTest UDP Server!
```

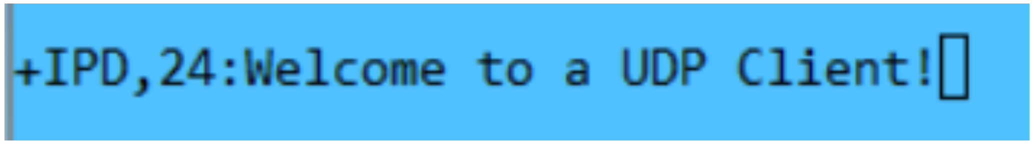
UDP Client

- Exp.: UDP Client test using SocketTest program
  - To send data from the server
    - Enter Client IP address and port number
    - Enter data to Message box, and then click Send



```
> Server Started on Port : 50000
> ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
R[192.168.0.62:46788]: Hello, SocketTest UDP Server!
```

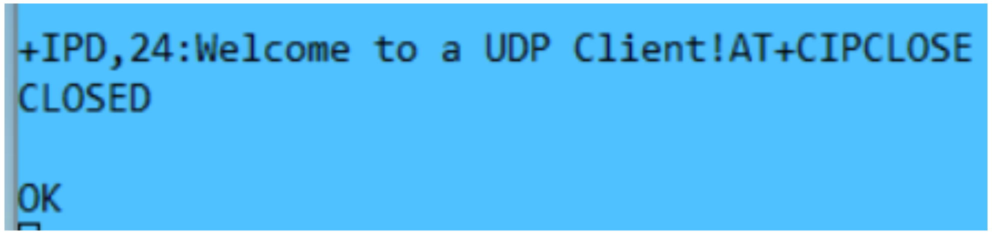
IP Address: 192.168.0.62
Port: 46788
Message: Welcome to a UDP Client!



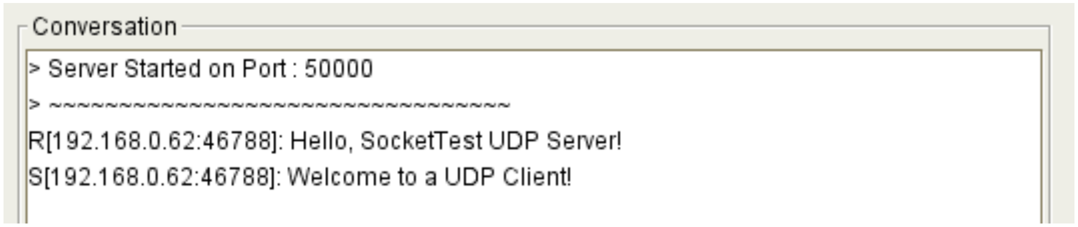
```
+IPD,24:Welcome to a UDP Client!
```

UDP Client

- Exp.: UDP Client test using SocketTest program
  - To unregister the UDP server
    - AT+CIPCLOSE



```
+IPD,24:Welcome to a UDP Client!AT+CIPCLOSE
CLOSED

OK
```

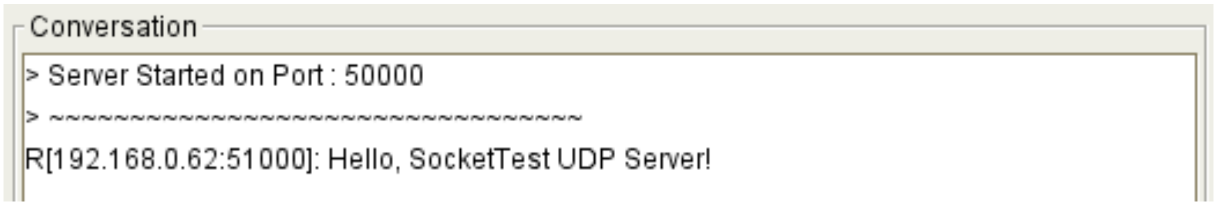


```
> Server Started on Port : 50000
> ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
R[192.168.0.62:46788]: Hello, SocketTest UDP Server!
S[192.168.0.62:46788]: Welcome to a UDP Client!
```

UDP Client

- Exp.: UDP Client test using SocketTest program
  - Register the UDP server with a fixed local port using AT command
    - AT+CIPSTART="UDP","192.168.0.61",50000, 51000
  - And then, send data to the server
    - AT+CIPSEND=30

      >Hello, SocketTest UDP Server!



```
Conversation
> Server Started on Port : 50000
> ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
R[192.168.0.62:51000]: Hello, SocketTest UDP Server!
```

Lab. 13-4: TCP Client in Transparent Mode

TCP

- Lab. 13-2와 동일한 동작 환경에서
  - Transparent mode에서 data 송수신을 실행하고,
  - Normal mode로 동작 모드를 변경한 다음 data 송수신을 수행하시오.

TCP CLIENT PROGRAM IN MBED-OS

Handong university

Jong-won Lee

2022.05.21

WiFI Interface

Class WiFiInterface

- Ref.: https://os.mbed.com/docs/mbed-os/v6.15/mbed-os-api-doxy/_wi_fi_interface_8h_source.html

```
class WiFIInterface : public virtual NetworkInterface
{
        static WiFiInterface *get_default_instance();
         …
}
```

- To generate an instance of WiFiInterface

```
WiFiInterface *wifi;
…
wifi = WiFiInterface::get_default_instance();
```

Class WiFiInterface

함수	nsapi_error_t connect(const char *ssid, const char *pass, nsapi_security_t security = NSAPI_SECURITY_NONE, uint8_t channel = 0);
동작	주어진 SSID 지닌 AP에 접속을 시도한다. 인자: ssid – 접속할 AP의 SSID 포인터. pass – 접속할 AP의 패스워드. Security – AP에서 사용하는 암호화 방법, 디폴트는 사용하지 않는 것임. 종류로는 NSAPI_SECURITY_WEP과 NSAPI_SECURITY_WPA, NSAPI_SECURITY_WPA2, NSAPI_SECURITY_WPA_WPA2가 있다. Channel – 사용되지 않음. 반드시 0이어야 함. 반환값: 성공작으로 접속이 되면 0 (NSAPI_ERROR_OK), 실패 시에는 음수 값 반환. (error type은 nsapi_types.h 참조)
함수	nsapi_error_t disconnect();
동작	접속된 AP와의 연결을 종료시킴. 반환값: 성공이면 0, 실패면 음수값

Class WiFiInterface

함수	nsapi_error_t **get_ip_address** (SocketAddress *address)
동작	인터페이스의 IP 주소를 얻음. (현재 권장하는 함수)
함수	nsapi_error_t **get_netmask** (SocketAddress *address)
동작	서브넷 마스크 값을 얻음. (현재 권장하는 함수)
함수	nsapi_error_t **get_gateway** (SocketAddress *address)
동작	게이트웨이의 IP 주소를 얻음. (현재 권장하는 함수)

Class WiFiInterface

함수	nsapi_size_or_error_t **scan**(WiFiAccessPoint *res, unsigned count);
동작	사용 가능한 AP들을 찾음. 인자: res – 찾은 AP에 대한 정보를 저장할 버퍼 포인터. count – 찾을 AP의 최대 개수. 0이 주어지면 찾은 AP의 수가 반환됨. 반환값 – 저장된 AP의 개수. 에러 발생 시 음수값.
함수	int8_t **get_rssi**();
동작	수신 신호의 세기를 나타냄. 단위는 dBm 임.

Class SocketAddress

- Constructor

SocketAddress (const nsapi_addr_t &addr, uint16_t port=0)	
addr	Raw IP address
port	Optional 16-bit port, defaults to 0

SocketAddress (const char *addr, uint16_t port=0)	
addr	Null-terminated representation of the IP address
port	Optional 16-bit port, defaults to 0

SocketAddress (const void *bytes, nsapi_version_t version, uint16_t port = 0)	
bytes	Raw IP address in big-endian order
version	IP address version, NSAPI_IPv4 or NSAPI_IPv6
port	Optional 16-bit port, defaults to 0

Class SocketAddress

- Methods

함수	const char * **get_ip_address ()** const
동작	String 형태 IP 주소 반환함.
함수	const void * **get_ip_bytes ()** const
동작	Raw IP 반환함. (big-endian 형태)
함수	nsapi_version_t **get_ip_version ()** const
동작	IP 주소 버전을 반환함. (NSAPI_IPv4 or NSAPI_IPv6)
함수	nsapi_addr_t **get_addr ()** const
동작	Raw IP 주소 반환함.
함수	uint16_t **get_port ()** const
동작	Port number 반환함.

Class SocketAddress

- Methods

함수	bool **set_ip_address** (const char *addr)
동작	IP 주소를 string 형태로 설정함.
함수	void **set_ip_bytes** (const void *bytes, nsapi_version_t version)
동작	IP 주소와 버전을 설정함. (raw IP 주소를 이용.)
함수	void **set_addr** (nsapi_addr_t addr)
동작	Raw IP 형태로 IP 주소를 설정함.
함수	void **set_port** (uint16_t port)
동작	Port number를 설정함.

TCP Socket

Class TCPSocket

- Constructor

	TCPSocket ( )
동작	초기화되지 않은 TCP 소켓을 생성함. 추후에 반드시 open() 함수를 이용하여 사용할 네트워크 인터페이스의 네트워크 스택을 지정하여야 함.

Class TCPSocket

- Methods

nsapi_errot_t	connect (const SocketAddress& address);
동작	주어진 host의 port의 TCP 서버에 커넥션을 요청한다. address: 접속하고자 하는 TCP 서버 IP와 포트 번호를 지닌 객체. 성공이면 0, 실패일 경우 음수값 반환.

Class TCPSocket

- Methods

nsapi_size_or_error_t	send (const void *data, nsapi_size_t size);
동작	TCP 소켓으로 데이터를 전송함. 소켓이 블러킹 모드에 있을 경우, 모든 데이터가 TX 버퍼로 넘겨질 때까지 블럭됨. 만약 넌블러킹 소켓이거나 타임아웃이 설정되어 있으면, 일부 데이터만 TX 버퍼로 넘겨질 수 있음. data: 전송할 데이터가 저장된 버퍼 포인터. size: 전송할 데이터 크기. (단위: 바이트) 성공시 TX 버퍼로 넘겨진 데이터 크기. 실패시 음수값 반환. 넌블러킹 소켓이거나 타임아웃이 설정되어 있을 경우 아무런 데이터도 넘겨지지 않게 되면 NSAPI_ERROR_WOULD_BLOCK 값이 반환됨.

Class TCPSocket

- Methods

nsapi_size_or_error_t	recv (void *data, nsapi_size_t size);
동작	TCP 소켓으로 전송된 데이터를 수신함. 디폴트 상태인 블러킹 소켓에서는 데이터를 수신할 때까지 블럭됨. data: 수신할 데이터를 저장할 버퍼 포인터. size: 버퍼의 최대 크기. 수신한 데이터의 크기. (단위는 바이트임.) 실패시 음수값이 반환됨. 넌블로킹 소켓이거나 타임아웃이 설정되어 있을 경우 수신한 데이터가 없으면 NSAPI_ERROR_WOULD_BLOCK(-3001) 값이 반환됨.
nsapi_error_t	close ();
동작	커넥션을 닫고, 생성된 소켓을 제거함. 성공 시에 0, 실패 시에 음수 값 반환함.

Class TCPSocket

Methods

void	set_blocking (bool blocking);
동작	소켓을 블러킹 혹은 넌블러킹 소켓으로 설정함. 디폴트 상태는 블러킹 소켓임. true 면 블러킹 모드, false 면 넌블러킹 모드임.
void	set_timeout (int timeout);
동작	소켓에 타임아웃 값을 설정함. 디폴트 타임아웃 값은 무한대 값임. 즉 set_timeout(-1)임. set_timeout(0) 은 set_blocking(false) 즉 넌블러킹 소켓인 경우와 동일함. set_timeout(-1)은 set_blocking(true) 즉 블러킹 소켓과 동일함. timeout: 타임아웃 값. (단위 밀리초)
nsapi_error_t	open (NetworkStack * stack);
동작	초기화되지 않은 소켓에 주어진 네트워크 인터페이스의 네트워크 스택 *stack에서 소켓을 오픈한다.

Class TCPSocket

- Methods

nsapi_error_t	bind (uint16_t port);
동작	소켓에 주어진 포트 번호를 바인드함. 성공 시에 0, 실패 시에 음수 값 반환함.
nsapi_error_t	bind (const SocketAddress& address);
동작	소켓에 주어진 IP 주소와 포트 번호를 바인드함. 주로 서버로 동작할 때에 사용됨. 성공 시에 0, 실패 시에 음수 값 반환함.
nsapi_error_t	bind (const char *address, uint16_t port);
동작	소켓에 주어진 IP 주소와 포트 번호를 바인드함. 주로 서버로 동작할 때에 사용됨. 성공 시에 0, 실패 시에 음수 값 반환함.

Class TCPSocket

- Methods

TCPSocket*	accept (nsapi_error_t *error=NULL)
동작	Client로부터 접속 요청을 받아 들임. Client와 정보 교환을 할 새로운 TCPSocket 포인터를 반환함.
nsapi_error_t	listen (int backlog=1)
동작	Client로부터 접속 요청을 대기함. 성공 시에 0, 실패 시에 음수 값 반환함.

TCP LABS IN MBED-OS

Handong university

Jong-won Lee

2022.05.21

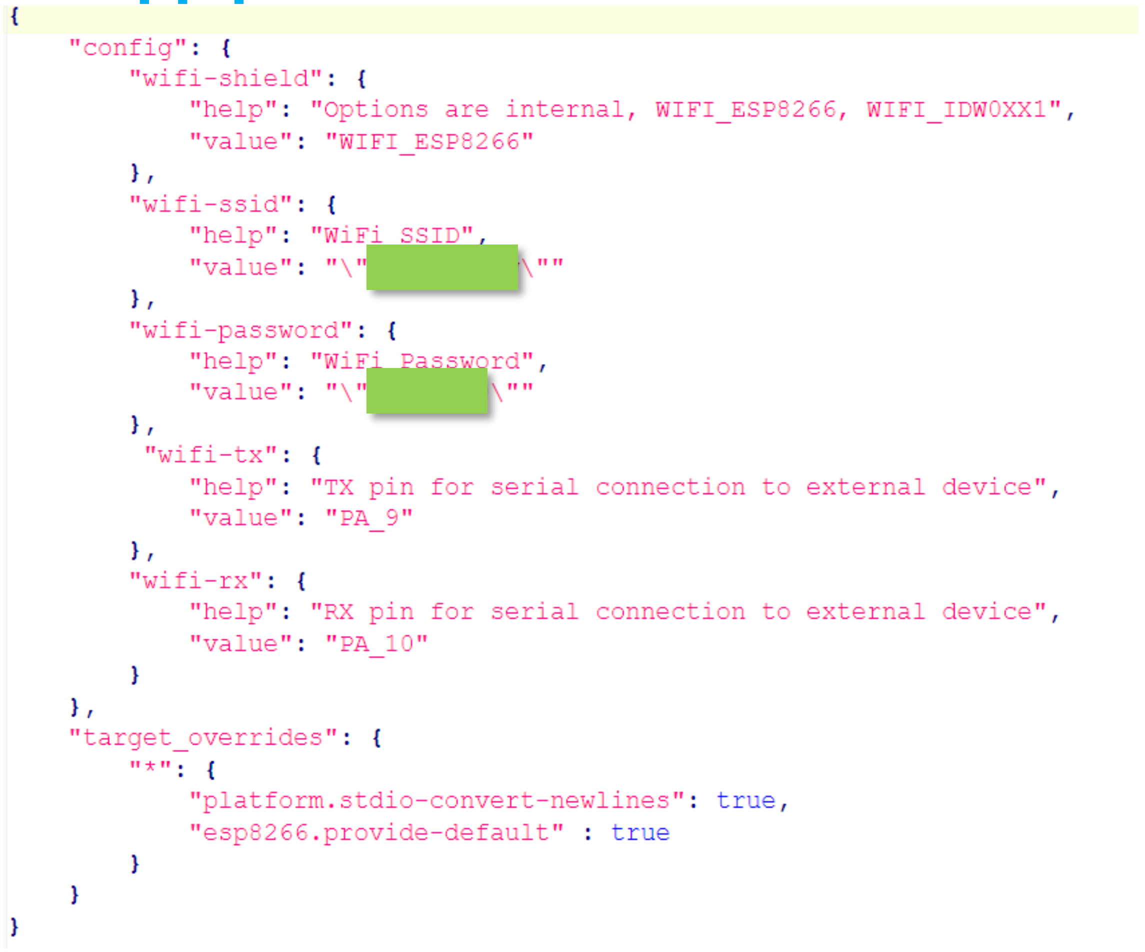
Simple TCP Client

Lab13-5: Simple TCP Client

- Scenario
  - Connect to a TCP server.
    - You can use a SocketTest tool for a TCP server.
  - Receive a line from the terminal.
  - If the input data is neither 'q' or 'Q', the data is transmitted to the server.
  - If the input data is 'q' or 'Q', the TCP socket is closed.
  - Display the data received from the server on the terminal.

Lab13-5: Simple TCP Client

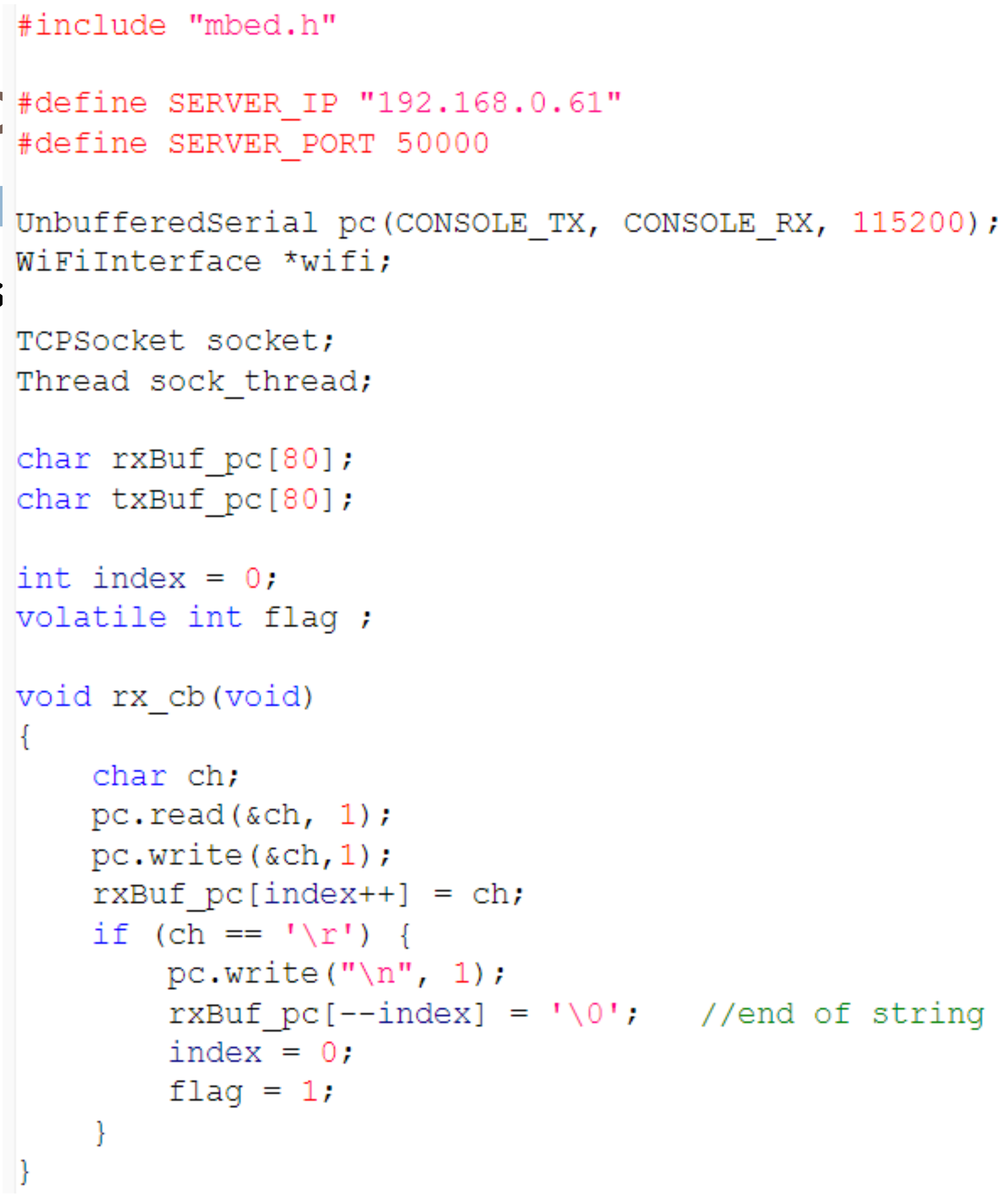
- Import a template project and then add a new file: **mbed_app.json**.



```
{
    "config": {
        "wifi-shield": {
            "help": "Options are internal, WIFI_ESP8266, WIFI_IDW0XX1",
            "value": "WIFI_ESP8266"
        },
        "wifi-ssid": {
            "help": "WiFi SSID",
            "value": "\"[redacted]\""
        },
        "wifi-password": {
            "help": "WiFi Password",
            "value": "\"[redacted]\""
        },
         "wifi-tx": {
            "help": "TX pin for serial connection to external device",
            "value": "PA_9"
        },
        "wifi-rx": {
            "help": "RX pin for serial connection to external device",
            "value": "PA_10"
        }
    },
    "target_overrides": {
        "*": {
            "platform.stdio-convert-newlines": true,
            "esp8266.provide-default" : true
        }
    }
}
```

Lab

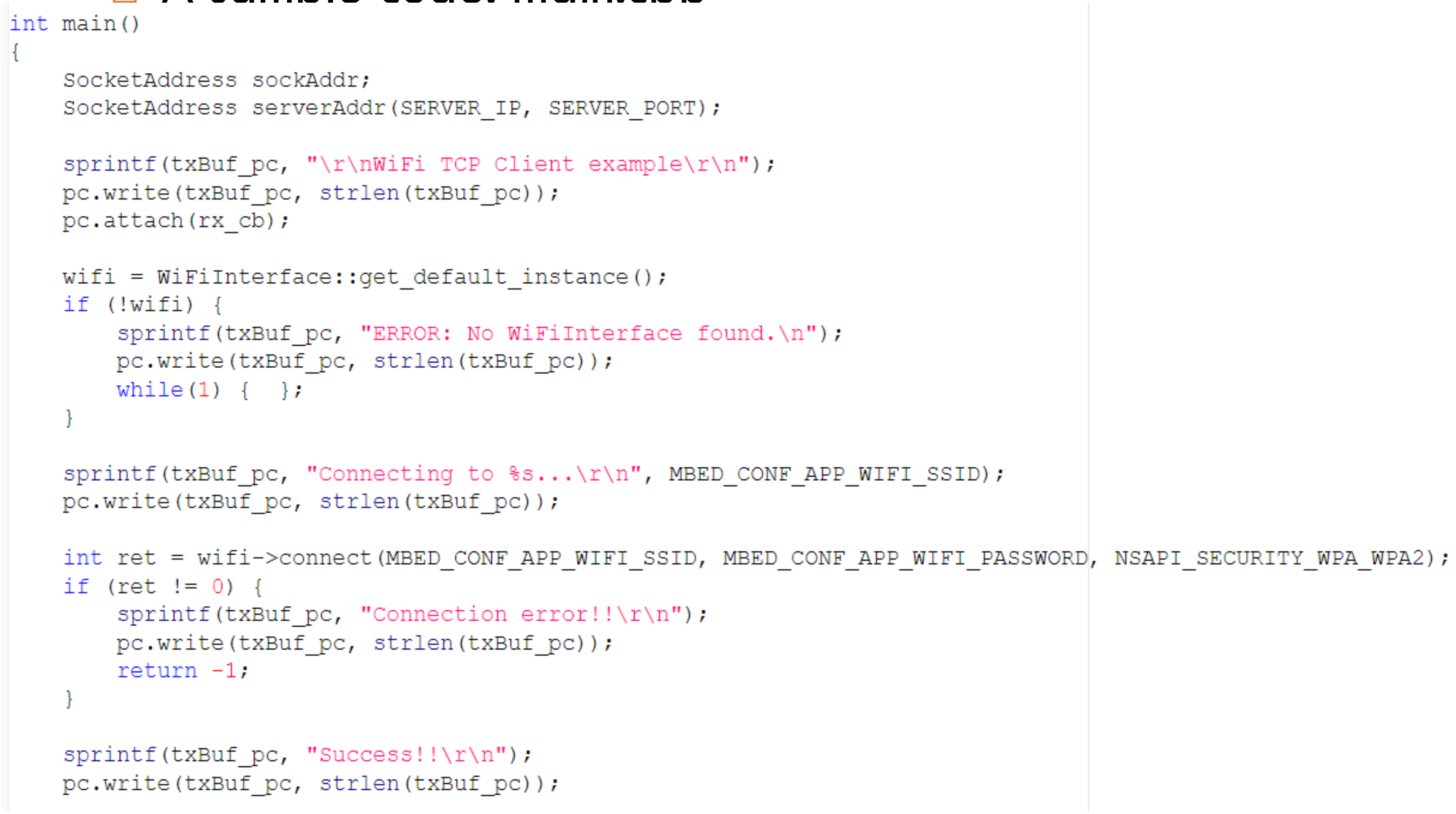
- A s



```
#include "mbed.h"

#define SERVER_IP "192.168.0.61"
#define SERVER_PORT 50000

UnbufferedSerial pc(CONSOLE_TX, CONSOLE_RX, 115200);
WiFiInterface *wifi;

TCPSocket socket;
Thread sock_thread;

char rxBuf_pc[80];
char txBuf_pc[80];

int index = 0;
volatile int flag ;

void rx_cb(void)
{
    char ch;
    pc.read(&ch, 1);
    pc.write(&ch,1);
    rxBuf_pc[index++] = ch;
    if (ch == '\r') {
        pc.write("\n", 1);
        rxBuf_pc[--index] = '\0';   //end of string
        index = 0;
        flag = 1;
    }
}
```

Lab13-5: Simple TCP Client

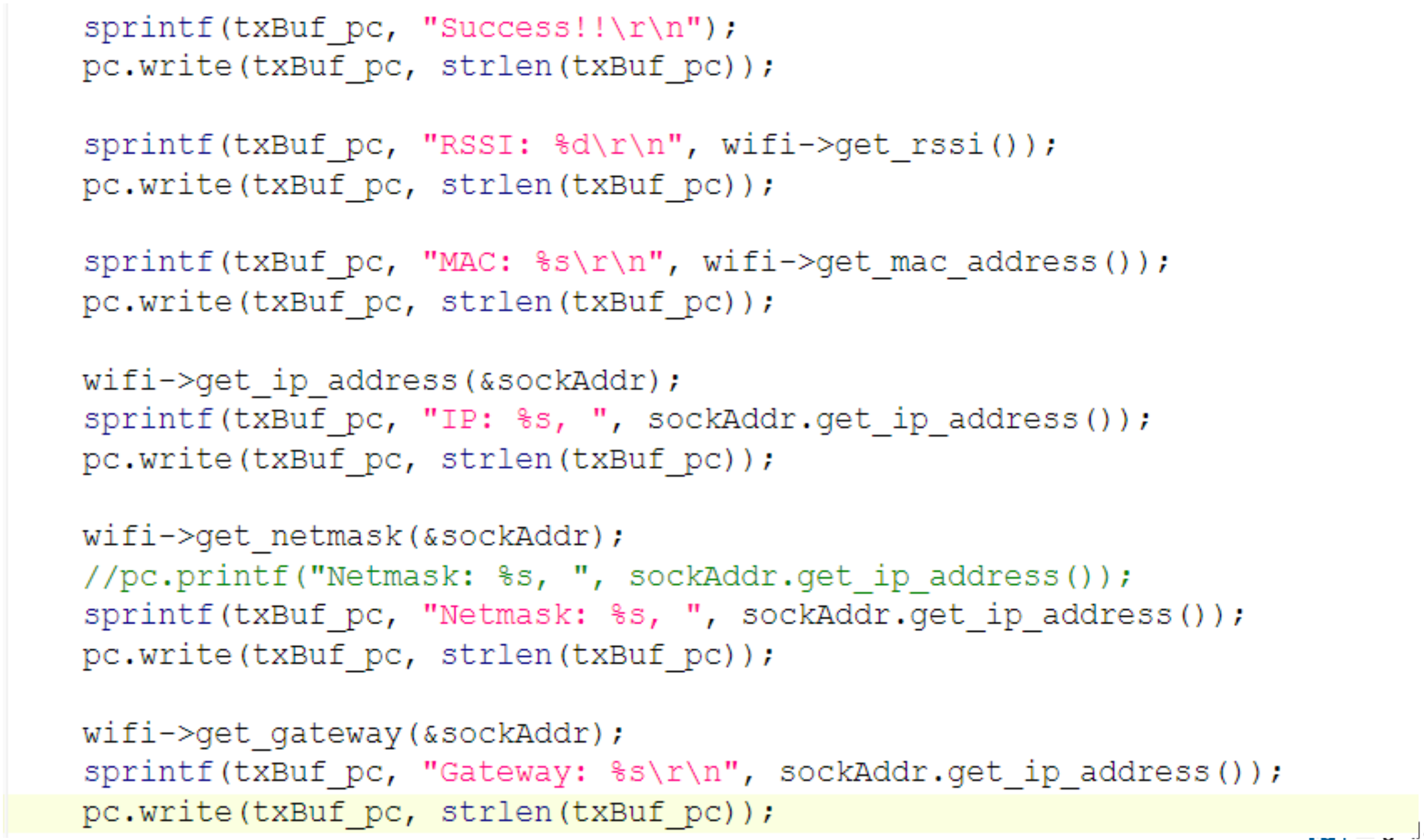
- A sample code: main.cpp



```
int main()
{
    SocketAddress sockAddr;
    SocketAddress serverAddr(SERVER_IP, SERVER_PORT);

    sprintf(txBuf_pc, "\r\nWiFi TCP Client example\r\n");
    pc.write(txBuf_pc, strlen(txBuf_pc));
    pc.attach(rx_cb);

    wifi = WiFiInterface::get_default_instance();
    if (!wifi) {
        sprintf(txBuf_pc, "ERROR: No WiFiInterface found.\n");
        pc.write(txBuf_pc, strlen(txBuf_pc));
        while(1) {   };
    }

    sprintf(txBuf_pc, "Connecting to %s...\r\n", MBED_CONF_APP_WIFI_SSID);
    pc.write(txBuf_pc, strlen(txBuf_pc));

    int ret = wifi->connect(MBED_CONF_APP_WIFI_SSID, MBED_CONF_APP_WIFI_PASSWORD, NSAPI_SECURITY_WPA_WPA2);
    if (ret != 0) {
        sprintf(txBuf_pc, "Connection error!!\r\n");
        pc.write(txBuf_pc, strlen(txBuf_pc));
        return -1;
    }

    sprintf(txBuf_pc, "Success!!\r\n");
    pc.write(txBuf_pc, strlen(txBuf_pc));
```

Lab13-5: Simple TCP Client

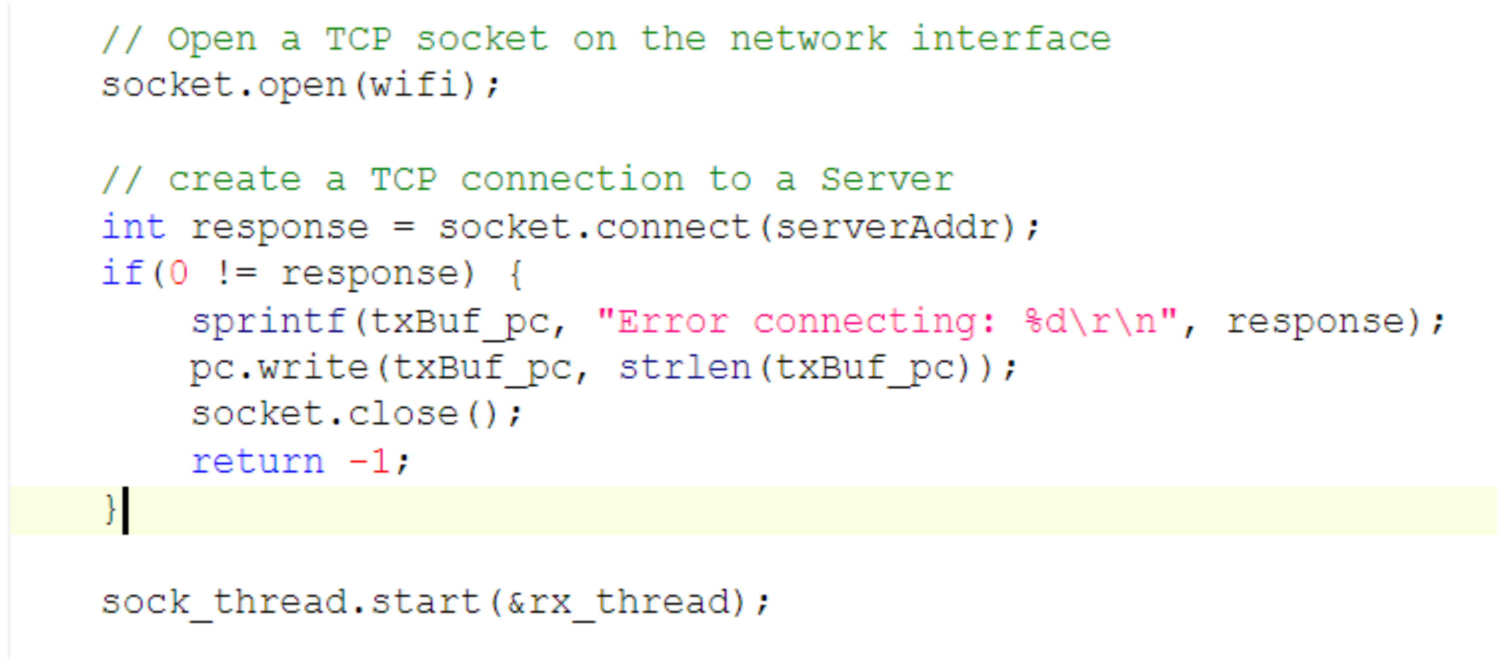
- A sample code: main.cpp



```
sprintf(txBuf_pc, "Success!!\r\n");
pc.write(txBuf_pc, strlen(txBuf_pc));

sprintf(txBuf_pc, "RSSI: %d\r\n", wifi->get_rssi());
pc.write(txBuf_pc, strlen(txBuf_pc));

sprintf(txBuf_pc, "MAC: %s\r\n", wifi->get_mac_address());
pc.write(txBuf_pc, strlen(txBuf_pc));

wifi->get_ip_address(&sockAddr);
sprintf(txBuf_pc, "IP: %s, ", sockAddr.get_ip_address());
pc.write(txBuf_pc, strlen(txBuf_pc));

wifi->get_netmask(&sockAddr);
//pc.printf("Netmask: %s, ", sockAddr.get_ip_address());
sprintf(txBuf_pc, "Netmask: %s, ", sockAddr.get_ip_address());
pc.write(txBuf_pc, strlen(txBuf_pc));

wifi->get_gateway(&sockAddr);
sprintf(txBuf_pc, "Gateway: %s\r\n", sockAddr.get_ip_address());
pc.write(txBuf_pc, strlen(txBuf_pc));
```

Lab13-5: Simple TCP Client

- A sample code: main.cpp

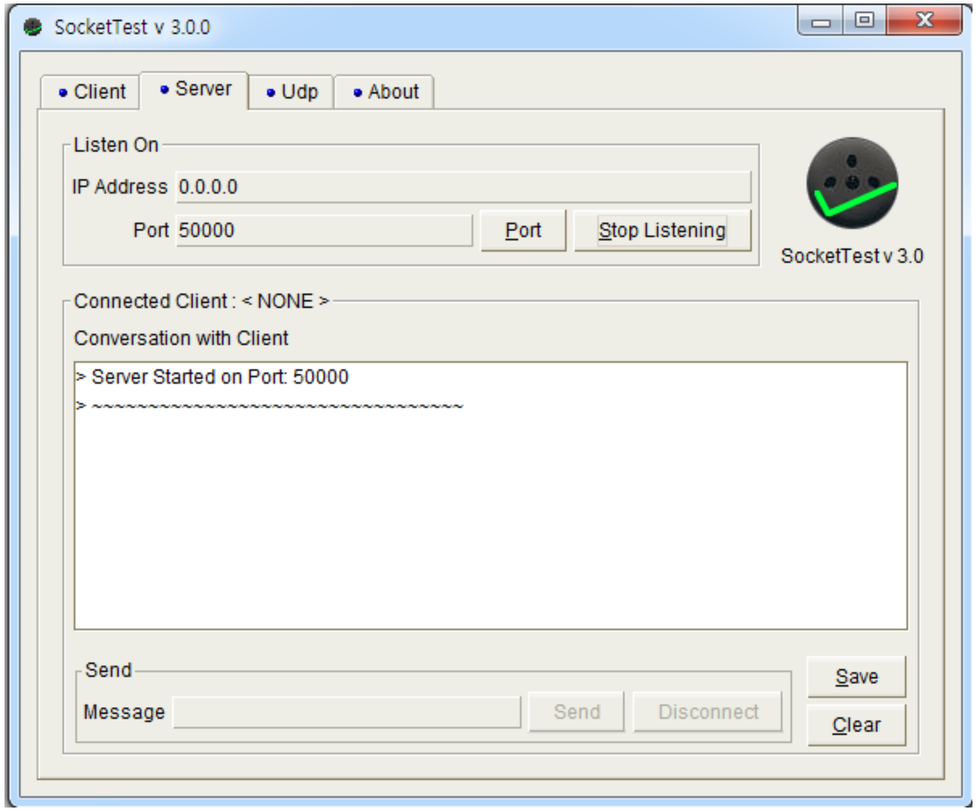


```
// Open a TCP socket on the network interface
socket.open(wifi);

// create a TCP connection to a Server
int response = socket.connect(serverAddr);
if(0 != response) {
    sprintf(txBuf_pc, "Error connecting: %d\r\n", response);
    pc.write(txBuf_pc, strlen(txBuf_pc));
    socket.close();
    return -1;
}

sock_thread.start(&rx_thread);
```

```
    while (true) {
        flag = 0;
        sprintf(txBuf_pc, "Enter characters to send to a server: ");
        pc.write(txBuf_pc, strlen(txBuf_pc));
        while (flag != 1) {
        }









     }

     socket.close();
     wifi->disconnect();

     sock_thread.join();
     sprintf(txBuf_pc, "RX sock_thread joined!!\r\n");
     pc.write(txBuf_pc, strlen(txBuf_pc));
     sprintf(txBuf_pc, "\nDone\r\n");
     pc.write(txBuf_pc, strlen(txBuf_pc));

     while(1) {  };
}
```

Lab13-5: Simple TCP Client

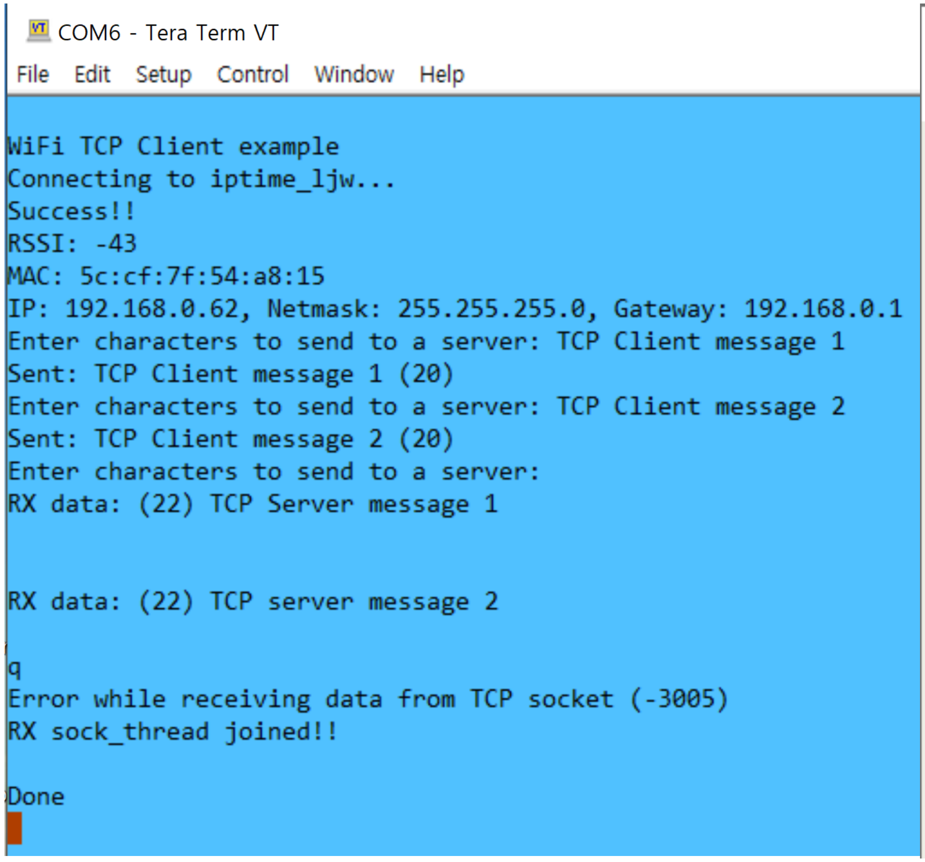
- Run SocketTest tool for a TCP server
  - Start listening for port number 50000



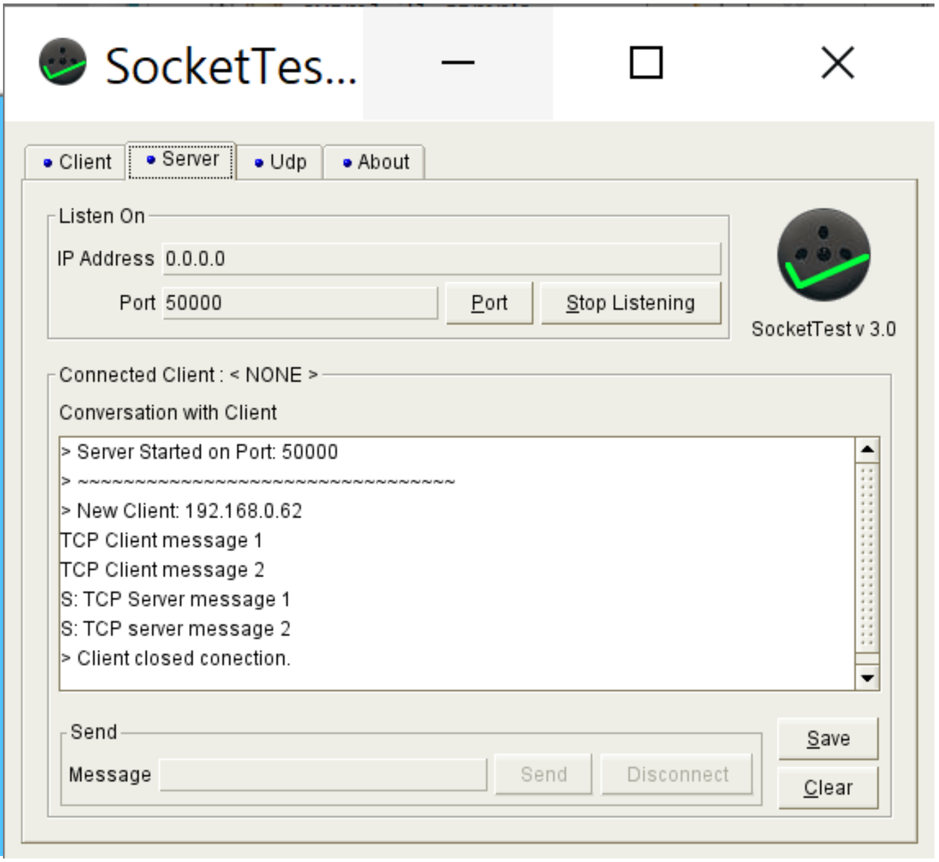
```
> Server Started on Port: 50000
> ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

Lab13-5: Simple TCP Client

- Download the program on mbed & run
  - It will send a message to the server.



```
WiFi TCP Client example
Connecting to iptime_ljw...
Success!!
RSSI: -43
MAC: 5c:cf:7f:54:a8:15
IP: 192.168.0.62, Netmask: 255.255.255.0, Gateway: 192.168.0.1
Enter characters to send to a server: TCP Client message 1
Sent: TCP Client message 1 (20)
Enter characters to send to a server: TCP Client message 2
Sent: TCP Client message 2 (20)
Enter characters to send to a server:
RX data: (22) TCP Server message 1


RX data: (22) TCP server message 2


q
Error while receiving data from TCP socket (-3005)
RX sock_thread joined!!

Done
```



Listen On
IP Address 0.0.0.0
Port 50000

Connected Client : < NONE >
Conversation with Client

```
> Server Started on Port: 50000
> ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
> New Client: 192.168.0.62
TCP Client message 1
TCP Client message 2
S: TCP Server message 1
S: TCP server message 2
> Client closed conection.
```

Send
Message

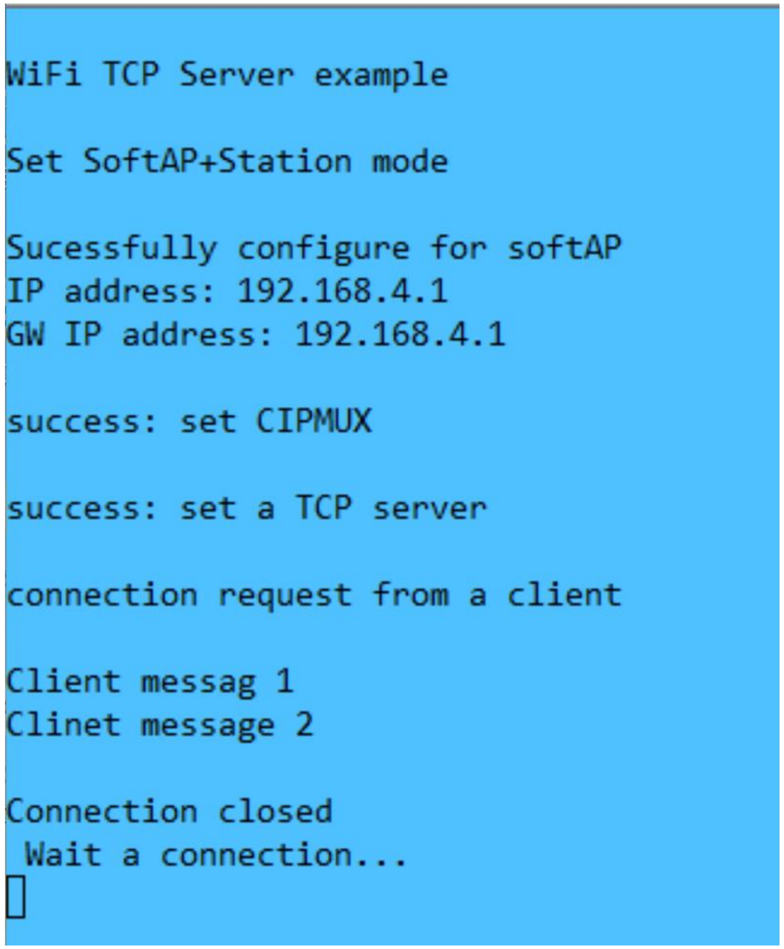
TCP Server

Lab.13-6: TCP Server

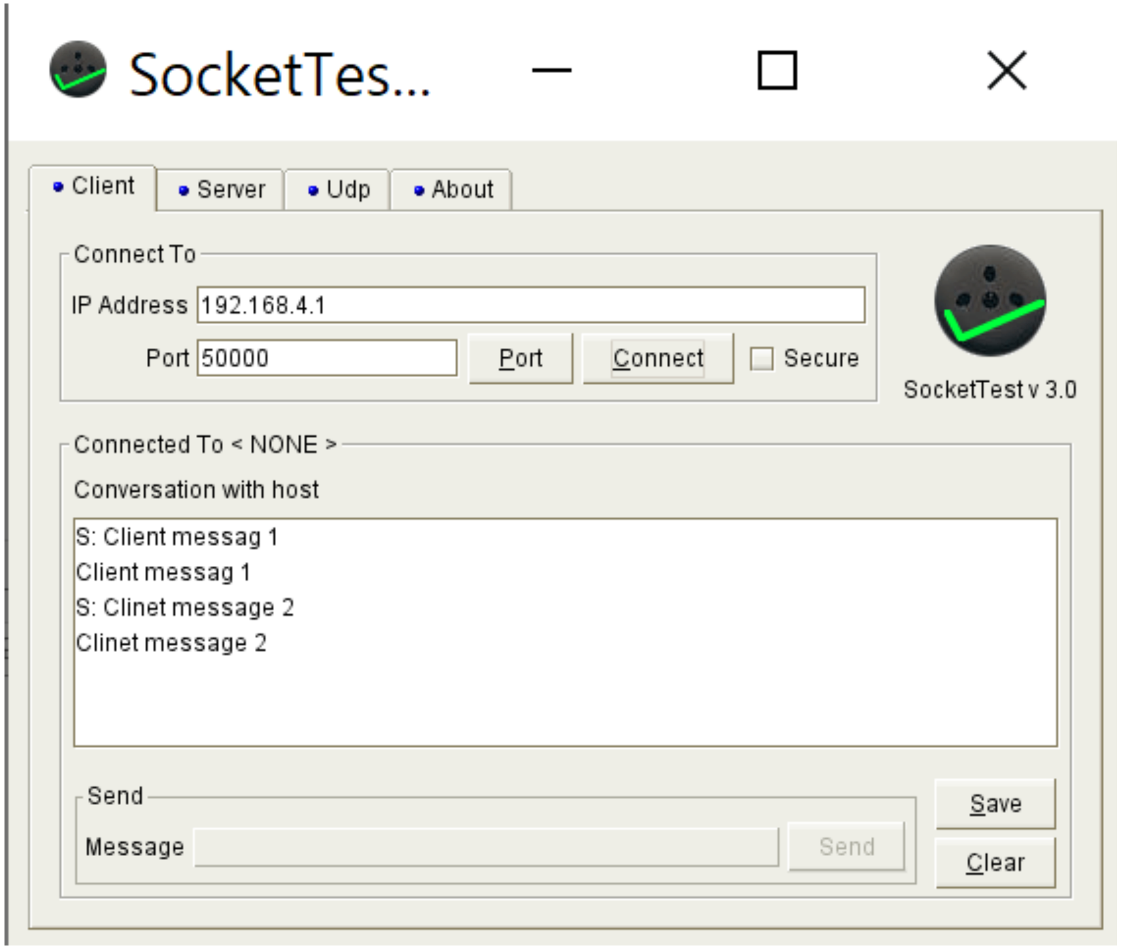
- 현재 mbed-os에서는 WiFi에 대해서는 TCP server 부분을 구현하지 않은 상태이다. 하지만 ESP8266 AT command를 보면, TCP server 기능이 firmware로 구현되어 있음을 알 수 있다.
- ESP-01 WiFi module을 Station+softAP 모드로 동작시킨다.
- AT command를 활용하여 다음과 같은 TCP echo server를 구현하시오.
  - TeraTerm과 같은 터미널을 이용하여 AT command를 주는 것이 아니라 program에서 AT command를 활용하는 프로그램을 하여야 한다.
  - Client가 보낸 data를 terminal에 표시하고, 동일한 데이터를 client로 전송한다.
  - Client가 connection을 종료하면 이를 표시하고, 다시 다른 client가 connection 하기를 대기한다.
- PC를 ESP-01 softAP에 접속을 한 다음, SocketTest의 TCP client 기능을 이용하여 Nucleo 보드에 접속을 한다.
- ATCmdParser Class를 이용하여 program할 수 있다.

Lab.13-6: TCP Server

- A sample result



```
WiFi TCP Server example

Set SoftAP+Station mode

Sucessfully configure for softAP
IP address: 192.168.4.1
GW IP address: 192.168.4.1

success: set CIPMUX

success: set a TCP server

connection request from a client

Client messag 1
Clinet message 2

Connection closed
 Wait a connection...
```



Connect To
IP Address 192.168.4.1
Port 50000

Connected To < NONE >
Conversation with host

```
S: Client messag 1
Client messag 1
S: Clinet message 2
Clinet message 2
```

Message