M480 WiFi TCP服務器

NuMicro® 32 位系列微控制器範例代碼介紹

文件資訊

代碼簡述	本範例使用 M480 UART 驅動 ESP8266，實現 TCP 服務器功能
BSP 版本	M480 Series BSP CMSIS V3.04.000
開發平台	NuMaker-IoT-M487 v1.3

1 功能介紹

1.1 簡介

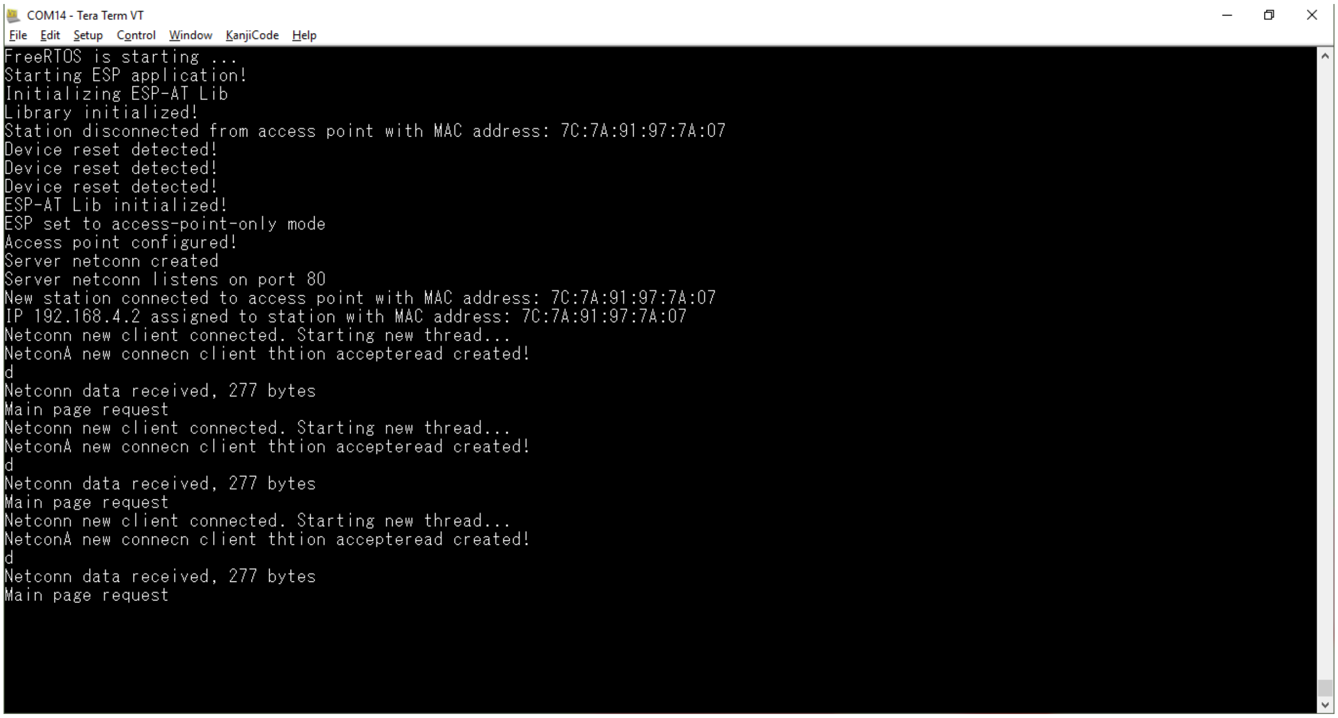
本範例程式碼使用M480系列晶片，通過UART1驅動ESP8266 WiFi模組，建立無線存取點，並提供TCP服務器功能。當電腦透過WiFi連接到ESP8266無線存取點後，可以開啟瀏覽器觀察NuMaker-IoT-M487 SW2(GPIO F11) 和 SW3(GPIO G5) 按鍵的狀態。

1.2 原理

範例程式核心使用了ESP-AT Lib函式庫。ESP-AT Lib是一個AT命令解析器，用於使用AT命令與ESP8266 WiFi模組通信。ESP-AT Lib提供了Station和Access point無線網路管理API，並提供了BSD socket相似的API，方便應用程式建立主從式架構的網路應用。

程式裡一開始讓ESP8266模組進入Access point模式，接著開啟HTTP 80埠，等待網路連接。此時可以讓電腦接入SSID為”ESP_AccessPoint”且Key為”12345678”的無線存取點，並自動取得IP。最後打開瀏覽器，連接80埠，取得目前NuMaker-IoT-M487 SW2和SW3按鍵的狀態。

1.3 執行結果



```
FreeRTOS is starting ...
Starting ESP application!
Initializing ESP-AT Lib
Library initialized!
Station disconnected from access point with MAC address: 7C:7A:91:97:7A:07
Device reset detected!
Device reset detected!
Device reset detected!
ESP-AT Lib initialized!
ESP set to access-point-only mode
Access point configured!
Server netconn created
Server netconn listens on port 80
New station connected to access point with MAC address: 7C:7A:91:97:7A:07
IP 192.168.4.2 assigned to station with MAC address: 7C:7A:91:97:7A:07
Netconn new client connected. Starting new thread...
NetconA new connecn client thtion accepteread created!
d
Netconn data received, 277 bytes
Main page request
Netconn new client connected. Starting new thread...
NetconA new connecn client thtion accepteread created!
d
Netconn data received, 277 bytes
Main page request
Netconn new client connected. Starting new thread...
NetconA new connecn client thtion accepteread created!
d
Netconn data received, 277 bytes
Main page request
```

2 代碼介紹

設置 ESP8266 模組為 Access point 模式：

```
    /* Enable access point only mode */
    if ((res = esp_set_wifi_mode(ESP_MODE_AP, 1, NULL, NULL, 1)) == espOK)
    {
        printf("ESP set to access-point-only mode\r\n");
    }
    else
    {
        printf("Problems setting ESP to access-point-only mode: %d\r\n", (int)res);
    }

    /* Configure access point */
    res = esp_ap_configure("ESP_AccessPoint", "12345678", 10, ESP_ECN_WPA2_PSK, 5, 0, 0,
NULL, NULL, 1);

    if (res == espOK)
    {
        printf("Access point configured!\r\n");
    }
```

開啟 80 埠服務：

```
    /*
     * First create a new instance of netconn
     * connection and initialize system message boxes
     * to accept clients and packet buffers
     */
    server = esp_netconn_new(ESP_NETCONN_TYPE_TCP);

    if (server != NULL)
    {
        printf("Server netconn created\r\n");

        /* Bind network connection to port 80 */
        res = esp_netconn_bind(server, 80);
        if (res == espOK)
        {
            printf("Server netconn listens on port 80\r\n");
            /*
             * Start listening for incoming connections
             * on previously binded port
             */
            res = esp_netconn_listen(server);

            while (1)
            {
                /*
                 * Wait and accept new client connection
                 *
                 * Function will block thread until
                 * new client is connected to server
                 */
                res = esp_netconn_accept(server, &client);

                if (res == espOK)
                {
                    printf("Netconn new client connected. Starting new thread...\r\n");

                    /*
                     * Start new thread for this request.
                     *
                     * Read and write back data to user in separated thread
                     * to allow processing of multiple requests at the same time
                     */
                    if (esp_sys_thread_create(NULL, "client",
(esp_sys_thread_fn)netconn_server_processing_thread, client, 512, ESP_SYS_THREAD_PRIO))
                    {
                        printf("Netconn client thread created\r\n");
                    }
                    else
                    {
                        printf("Netconn client thread creation failed!\r\n");

                        /* Force close & delete */
                        esp_netconn_close(client);
                        esp_netconn_delete(client);
                    }
                }
                else
                {
                    printf("Netconn connection accept error!\r\n");
                    break;
                }
            }
        }
        else
        {
            printf("Netconn server cannot bind to port\r\n");
        }
    }
    else
    {
        printf("Cannot create server netconn\r\n");
    }
```

3 軟體與硬體環境

- **軟體環境**
    - BSP 版本
        - M480 Series BSP CMSIS V3.04.000
    - IDE 版本
        - Keil uVersion 5.26
- **硬體環境**
    - 電路元件
        - NuMaker-IoT-M487
        - ESP8266 模組
    - 示意圖

M480 通過 UART1 接口與 ESP8266 模組連接。

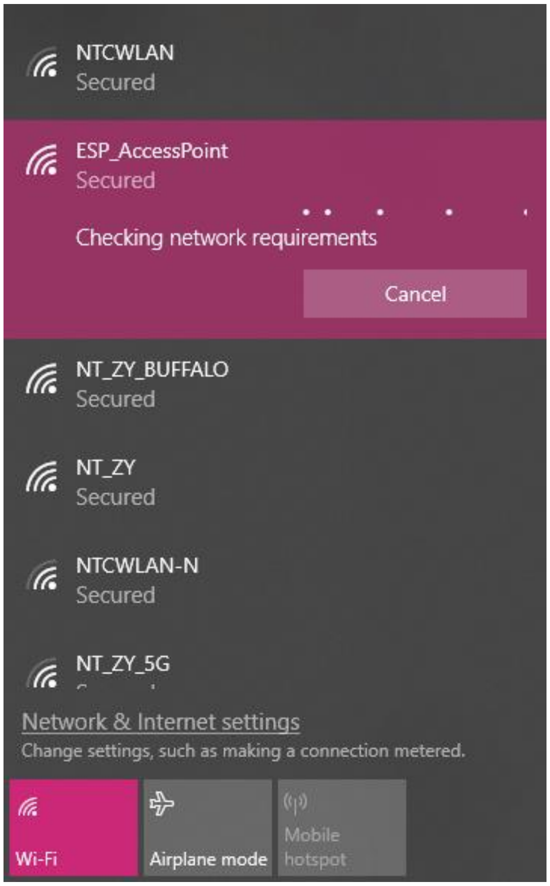
VCC	↔	VCC
PH.3	→	RESET
UART1_TXD(PH.8)	→	RXD
UART1_RXD(PH.9)	←	TXD
UART1_RTS(PB.8)	→	CTS
UART1_CTS(PB.9)	←	RTS
GND	↔	GND
M480		ESP8266

4 目錄資訊

🗁 EC_M480_WiFi_TCPServer_V1.00	
🗁 Library	Sample code header and source files
🗁 CMSIS	Cortex® Microcontroller Software Interface Standard (CMSIS) by Arm® Corp.
🗁 Device	CMSIS compliant device header file
🗁 StdDriver	All peripheral driver header and source files
🗁 SampleCode	
🗁 ExampleCode	Source file of example code
🗁 ThirdParty	
🗁 FreeRTOS	A real time operating system available for free download. Its official website is: http://www.freertos.org/
🗁 ESP_AT_Lib	ESP_AT Library commands parser is generic, platform independent, library for communication with ESP8266 WiFi module using AT commands. Its official website is https://majerle.eu/documentation/esp_at/html/index.html

5 如何執行範例程式

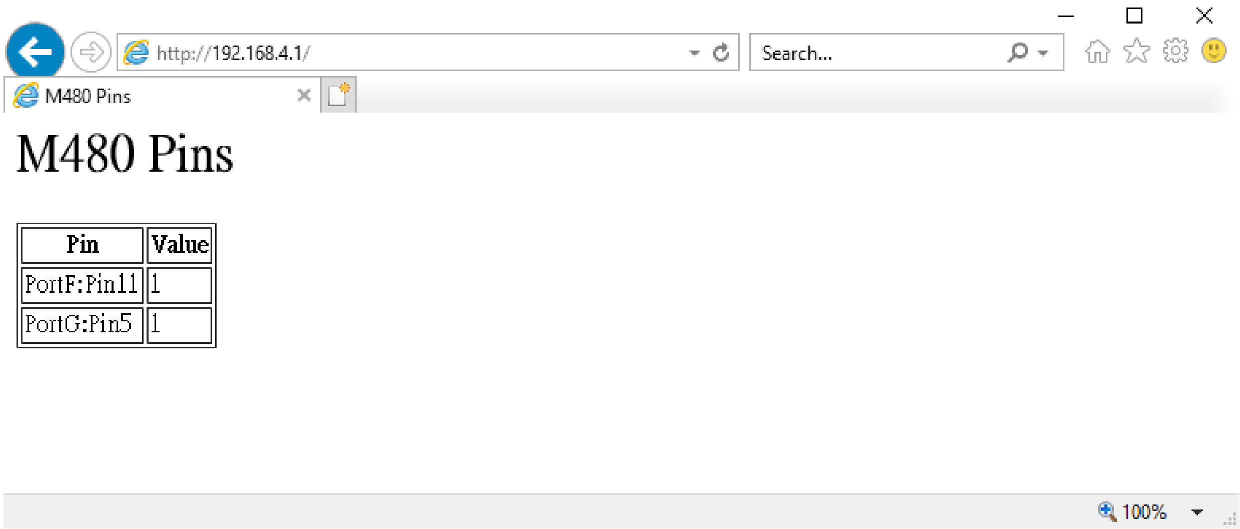
1. 根據目錄資訊章節進入 ExampleCode 路徑中的 KEIL 資料夾，雙擊 M480_WiFi_TCPServer.uvproj。
2. 進入編譯模式介面
    a. 編譯
    b. 下載代碼至記憶體
    c. 進入 / 離開除錯模式
3. 進入除錯模式介面
    a. 執行代碼
4. 電腦連接“ESP_AccessPoint”無線存取點 (key:12345678)



Checking network requirements

Change settings, such as making a connection metered.

5. 打開瀏覽器，並輸入 http://192.168.4.1/



M480 Pins

Pin	Value
PortF:Pin11	1
PortG:Pin5	1

6 修訂紀錄

Date	Revision	Description
Oct. 7, 2019	1.00	1. 初始發布.

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, “Insecure Usage”.

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer’s risk, and in the event that third parties lay claims to Nuvoton as a result of customer’s Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.