Reference Manual

mxchipWNet-DTU

UART <> Wi-Fi Firmware

4.0 **Date : 2013-09-25** **Reference manual**

Overview

mxchipWNet™ firmware is a software system running run on EMW Wi-Fi modules developed by MXCHIP. These firmware embedded with multiple M2M applications, TCP/IP stack and Wi-Fi driver can greatly reduce your development time and improve competitiveness on your M2M products.

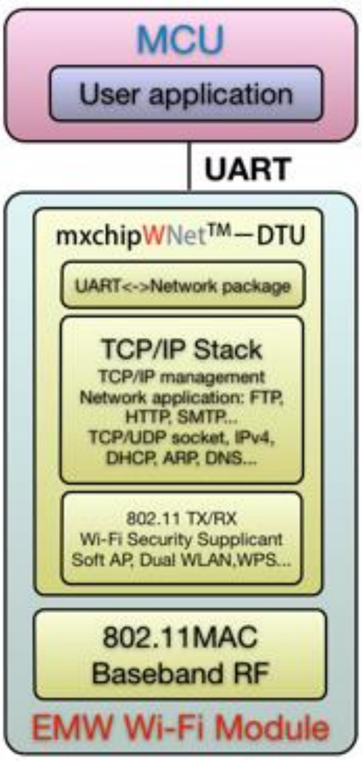
Firmware: mxchipWNet™-DTU is used to implement the Wi-Fi data transmission on serial devices. Two primary functions are provided: EMSP command and direct data transmission between UART and Wi-Fi. It is widely used in establishing wireless communication on serial devices.

Main functions:

- Two working modes: EMSP command mode and direct data transmission mode
- Multiple configuration interface
- Build-in web site
- Firmware updating using serial and web

Wi-Fi driver features:

- WLAN standard: IEEE 802.11 b/g/n
- RF frequency: channel 1-13 on 2.4GHz
- Greatly reduce power consumption
- Roaming between multiple access points
- Support station mode,, soft AP mode and Wi-Fi direct
- Security: WEP and WPA/WPA2 PSK enterprise
- Auto detect security mode



TCP/IP features:

- DHCP client and server
- DNS, mDNS (bonjour)
- Two sockets working at the same time
- TCP client/server with keep-alive detection and auto reconnection
- UDP unicast/broadcast
- Support 8 clients in TCP server mode
- HTTP, FTP and SMTP client
- Sock4/5 proxy

UART features:

- Baud rate: 2400-3686400
- CTS/RTS hardware flow control
- Multiple UART data package mode
- High-speed data transfer by DMA

Contents

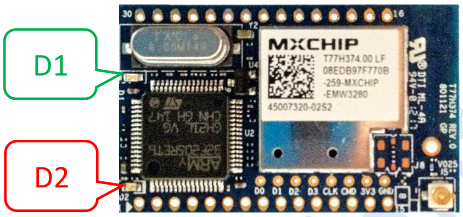
1 Introduction

mxchipWNet™ firmware is a software system running run on EMW Wi-Fi modules developed by MXCHIP. These firmware embedded with multiple M2M applications, TCP/IP stack and Wi-Fi driver can greatly reduce your development time and improve competitiveness on your M2M products.

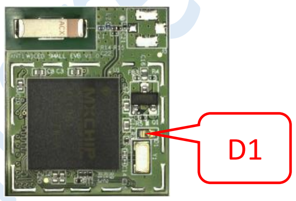
One of the firmware: mxchipWNet™-DTU is used to implement the Wi-Fi data transmission on serial devices. Two primary functions: EMSP command and direct data transmission between UART and Wi-Fi are provided. It is widely used in establishing wireless communication on serial devices.

mxchipWNet™-DTU can run on: EMW3280, EMW3161, and EMW3162. LEDs and pins on these module are defined to a specified function which is described in 1.1 and 1.2:

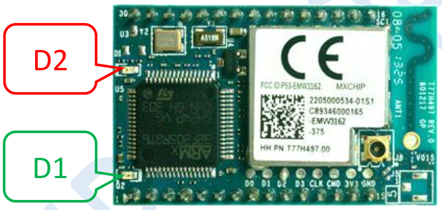
1.1 LED Functions Controlled by Firmware



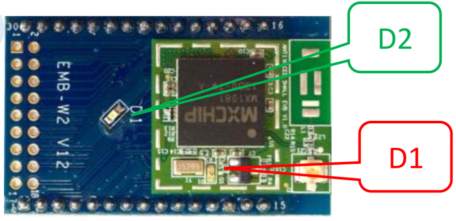
EMW3280



EMW3161



EMW3162



EMW3160

Table 1.1. LED function list

Name	Function description	
D1	On	Firmware initialize successful
	Off	Firmware uninitialized, or deep sleep mode
	Flash	Firmware update mode
D2*	On	Wi-Fi connected, DHCP negotiate success
	Off	Wi-Fi disconnected
	Keep Flashing	Data transmission/WPS negotiating/EasyLink mode

* D2 led is available on EMW3280, EMW3162 and EMW3160, and you can also connect an external led from pin 16. D2 is not existed on EMW3161 module, so you should connect an external LED from pin 36 on EMW3161.

1.2 Pin Definition Under mxchipWNet™-DTU

1.2.1 EMW3280, EMW3160 and EMW3162

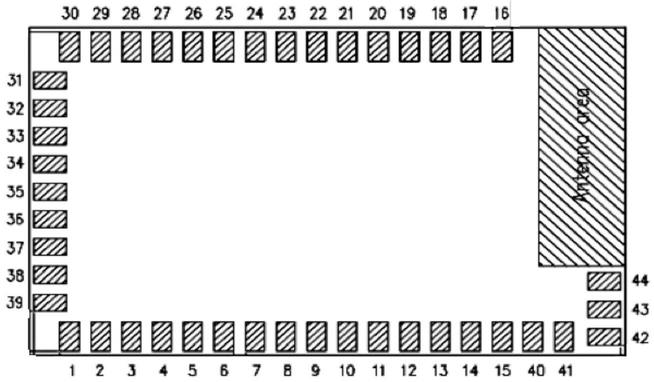
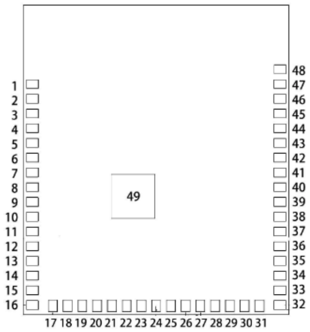


Table 1.2. EMW3280, EMW3160 and EMW3162 Pinouts

Pins	Pin Name	Pins	Pin Name
1	NFC_SCL	16	LED (OUT) BOOT(IN)
2	NFC_SDA	17	nRESET(IN)
3	NC	18	IO1
4	NC	19	NC
5	WPS/Default (IN)	20	nUART_RTS(OUT)
6	NC	21	nUART_CTS(IN)
7	NC	22	UART_TXD(OUT)
8	NC	23	UART_RXD(IN)
9	NC	24	VDD
10	NC	25	GND
11	EasyLink/Default (IN)	26	NC
12	NC	27	NC
13	NFC_INT	28	NC
14	LoPow(IN) *	29	nWAKE_UP(IN）
15	GND	30	STATUS(IN)

* LoPow function is only available on EMW316x modules.

1.2.2 EMW3161



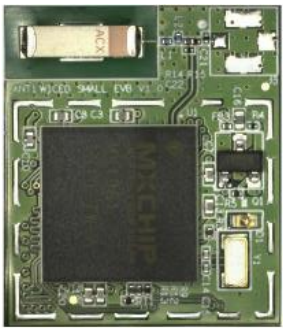


Table 1.3. EMW3161 Pinouts

Pins	Pin Name	Pins	Pin Name
11	WPS/Default (IN)	41,44,49	GND
12	EasyLink/Default (IN)	45	nUART_CTS(IN)
24	LoPow(IN) *	47	UART_TXD(OUT)
36	LED (OUT) BOOT(IN)	48	UART_RXD(IN)
40	nRESET(IN)	42,43	VDD
21	IO1	38	nWAKE_UP(IN)
46	nUART_RTS(OUT)	39	STATUS(IN)
Other	NC		

1.3 Function Description of Each Pin

Table 1.4. Function description

Pin	Type	Function
VDD		3.3V DC power input.
GND		Grounding.
NFC_SCL	O	IIC clock (connect to NFC tag EMF2104)
NFC_SDA	I/O	IIC data (connect to NFC tag EMF2104)
NFC_INT	I	Interrupt input (connect to NFC tag EMF2104)
WPS/Default	I	• Pull down and up: enter Wi-Fi protected setup(WPS) • Double click: enter one step configuration method: EasyLink • Pull down and keep for 5 seconds: restore all settings to default.
EasyLink/ Default	I	This pin is only available on EMW316x modules • Pull down and up: enter one step configuration method: EasyLink • Pull down and keep for 5 seconds: restore all settings to default.
UART_RXD	I	UART Data input.

Pin	Type	Function
UART_TXD	O	UART Data output.
nUART_CTS[5]	I	UART clear to send, active low.
nUART_RTS[5]	O	UART is ready to receive, active low.
STATUS	I, PU[1]	Set the operation mode: • Pull up: enter the direct data transmission mode • Pull down: enter the EMSP command mode Refer chapter 2 for details.
WAKE_UP	I, PU	Module enter deep sleep mode if the WAKE_UP pin is pulled low, and wake up while the WAKE_UP pin is pulled high. Reset is needed to wake up EMW3161 from deep sleep mode.
nRESET[6]	I, PU	Pull down this pin for 1μs to reboot the module.
LED/ BOOT	I, PU	BOOT signal is detected to enter the different working mode when module is powered on, Refer chapter 2 for details. • Pull up: Boot to normal working mode. • Pull down: Boot to special working modes (firmware update mode or test mode).
	O	LED pin has the same function as led D2, This signal is active low.
IO1	NC[7]	No function. (IO1 function is configured by firmware)
	I[7]	FC mode: Used as a serial data frame controller in direct data transmission mode. • Pull down: Module stores the received serial data in RAM • Pull up: Build up TCP/UDP package with the serial data in RAM
	O[7]	HDC mode: Used as a Half-duplex Controller. Output a high level signal while sending a serial data, otherwise, output a low level This function can also be used to wake up host before send any UART data.
NC		Undefined IOs. Leave them floating or grounding.

1. PU: The pin is at high level if no external signal is asserted.
2. PD: The pin is at low level if no external signal is asserted.
3. UART signals includes UART_TXD，UART_RXD，UART_RTS and UART_CTS。
4. Only VDD, GND UART_TXD and UART_RXD are needed in a simplest connection.
5. It is recommend that using UART_RTS and UART_CTS signal in serial interface. When using hardware flow control, firmware would not lost any serial data under TCP connection.
6. nRESET signal should not be forced to high by external circuit, internal watch

dog function would not work correctly in this condition, if nRESET is controlled by an IO signal, this IO should be set to open drain mode。

7. Use EMSP command to set the different function on IO1, “NC” is the default state.

1.4 Typical hardware connections

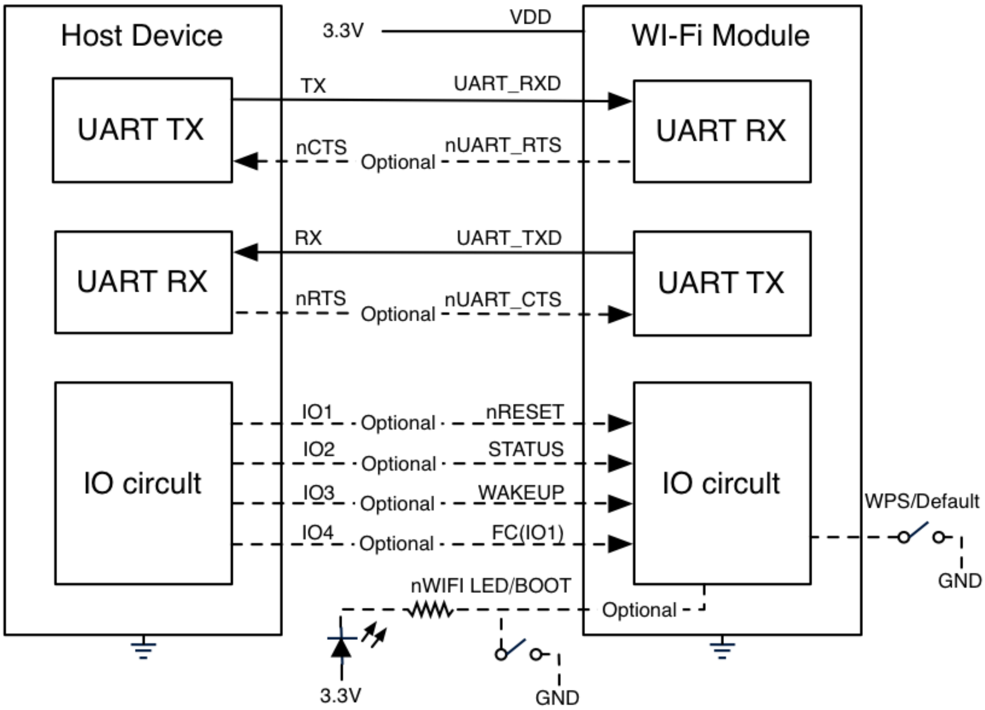


Figure 1.1 TTL/CMOS UART serial interface

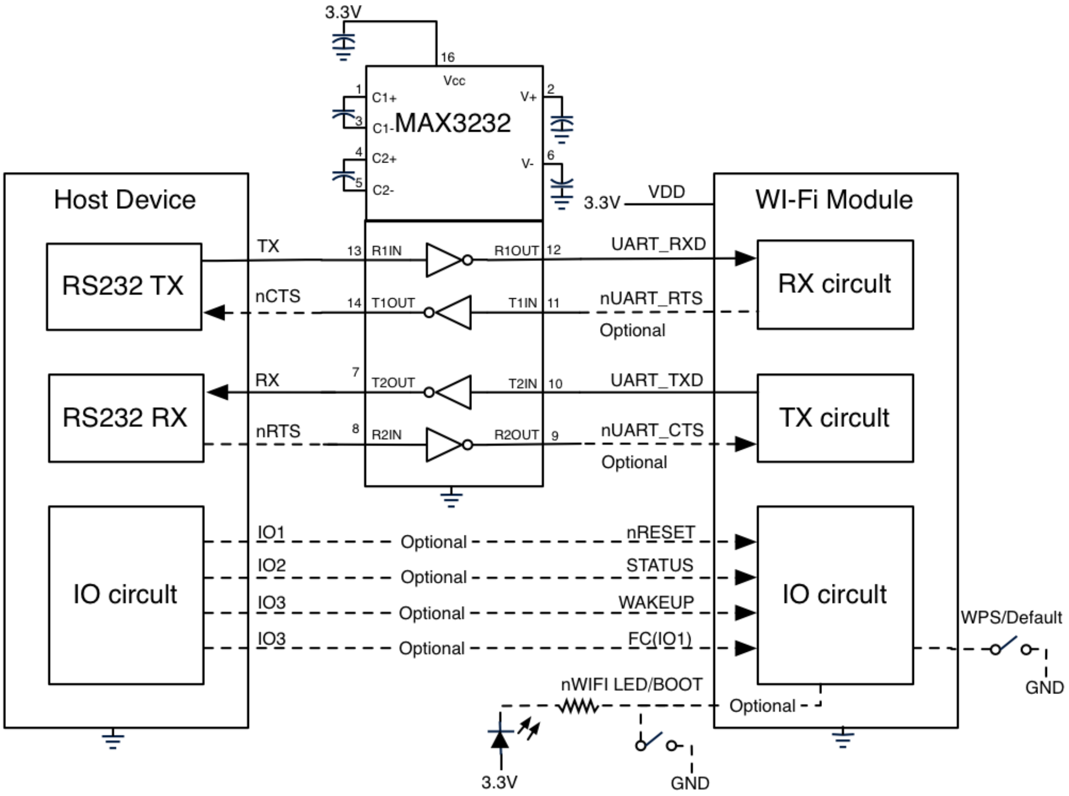


Figure 1.2 RS232 UART serial interface

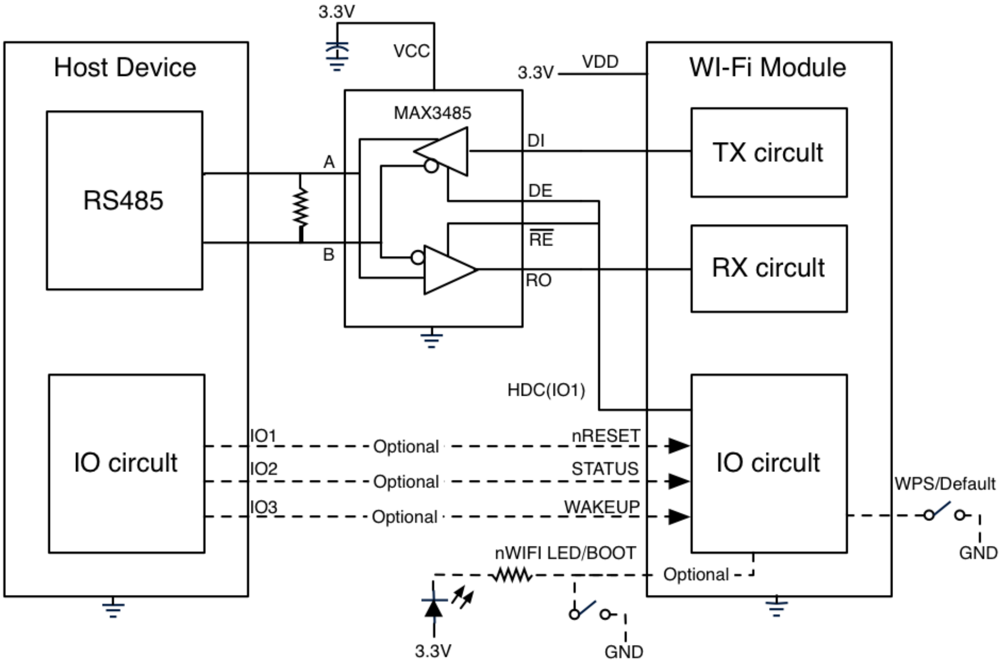
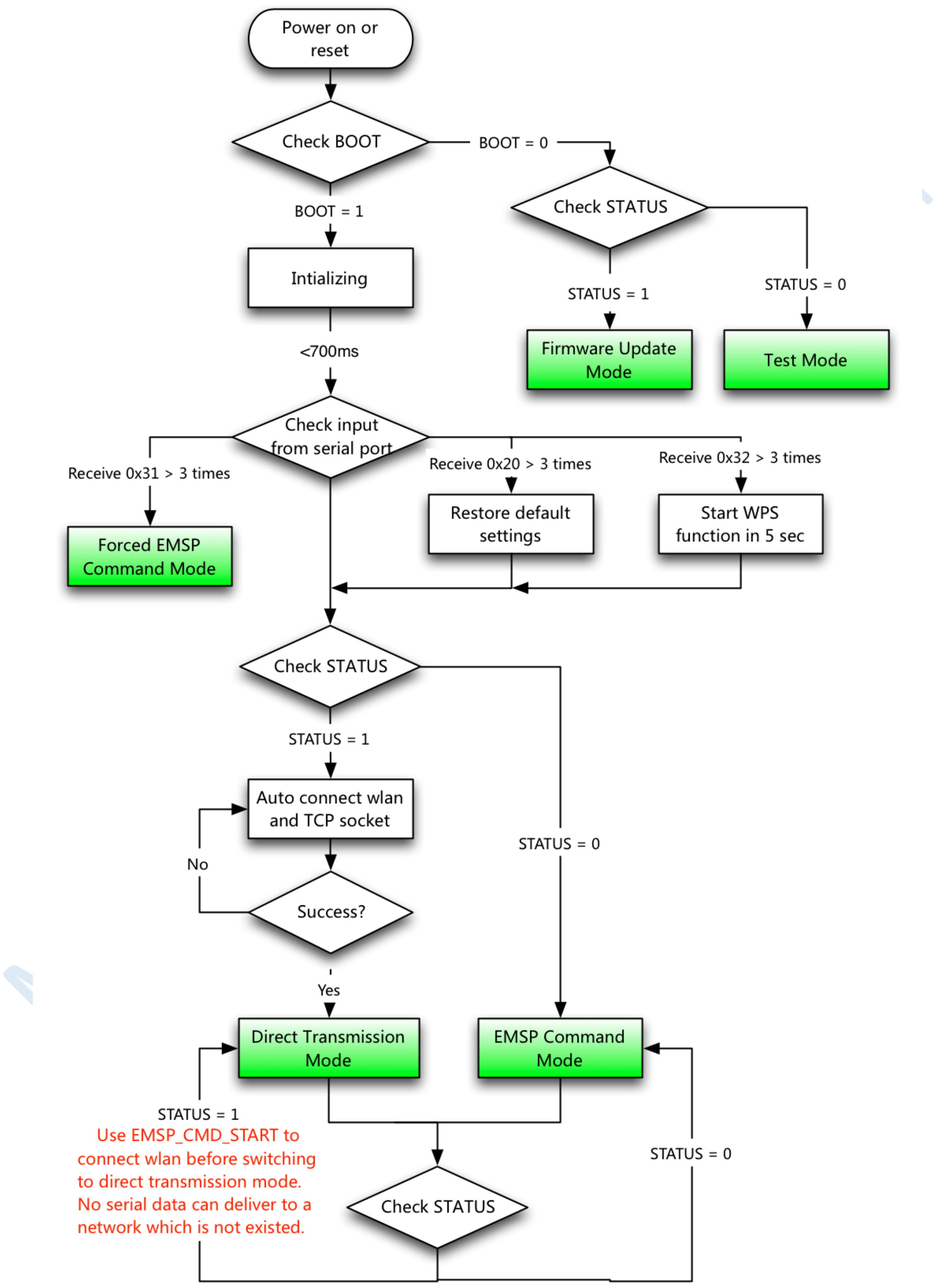


Figure 1.3 RS485 serial interface

2 Firmware Working Modes and Switching

2.1 Working Flowchart



2.2 Working Modes

Direct Transmission Mode：

Firmware automatically connects the wlan according to the predefined settings, then package the serial data into TCP/UDP packets and send them to wlan automatically. It can also receive TCP/UDP packets from wlan and send them to serial interface. Firmware enter this mode when STATUS signal is high.

EMSP Command Mode：

In this mode, you can use EMSP commands to control module and configure firmware' s parameters from serial port interface. Enter this mode when the STATUS pin is low. Please refer chapter 4 for details.

Forced EMSP Command Mode：

This mode has the same functions as EMSP command mode, but firmware doesn't check STATUS pin before entering this mode, and you can't quit this mode by assert high on STATUS pin of course. Use EMSP command EMSP_CMD_RESET or assert the nRESET pin to quit this mode.

Use this mode if you want to use EMSP command but STATUS pin is not connected.

Firmware Update Mode：

Firmware automatically connect the wlan according to the predefined settings, then package the serial data into TCP/UDP packets and send them to wlan automatically using TCP/UDP protocol. It can also receive TCP/UDP packets from wlan and send them to serial interface. Firmware enters this mode when STATUS signal is high.

Test Mode：

Firmware automatically connect the wlan according to the predefined settings, then package any data received from serial interface into TCP/UDP packets and send them to wlan automatically. It can also receive TCP/UDP packets from wlan and send them to serial interface. Firmware enters this mode when STATUS signal is high.

2.3 Working Mode after Power On

Firmware boot to different working depends on BOOT/STATUS signals.

Table 2.1. Boot Options

BOOT	STATUS	Working Mode
0	0	Test Mode
0	1（Default）	Firmware Update Mode
1（Default）	0	EMSP command Mode
1（Default）	1（Default）	Direct Transmission Mode

If BOOT=1, firmware will also check the continuous serial input during initializing with default serial settings: 115200/8/n/1. Initializing time is different depends on module’ s model, but would not exceed 750ms.

Table 2.2. Special serial input when booting

BOOT	STATUS	UART INPUT	Function
1	X	Receive three 0x31	Enter forced EMSP command mode
1	X	Receive three 0x20	Restore all settings to default
1	X	Receive three 0x32	Use WPS to negotiate with AP in 5 seconds

2.4 Switching between Working Modes

Direct Transmission Mode → EMSP Command Mode

1. Pull down the STATUS pin.
2. Send any EMSP command, until correct respond is returned.
3. Firmware enters EMSP Command Mode.

EMSP Command Mode → Direct Transmission mode

1. Send EMSP_CMD_START command to connect the network. (If connected, skip step 1)
2. Pull up the STATUS pin
3. Firmware enters Direct Transmission mode.

2.5 Default Settings

Customer can have their own factory settings when purchasing the module, contact MXCHIP for further information.

Default settings：

WLAN settings：SSID: “MXCHIP_XXXX” (XXXX=the last 2 bytes of module’s MAC address) , Soft AP mode

UART settings：115200/8/n/1

IP Address: 192.168.1.1/255.255.255.0, DHCP server enabled. Bonjour service enabled.

TCP Server mode, Port：8080

If you need to restore the default settings, you can

★ Send 0x20 to module’s serial port 3 times when firmware is initializing
Or

★ Push down WPS/Default or EasyLink/Default pin for 5 seconds

3 Direct Transmission Mode

Direct Transmission Mode adds the wireless data transmission on serial devices, and simplify the user's development significantly.

Details of the Direct Transmission Mode:

Transferring data in a reliable network must follow a certain kind of format and protocol. For example, when transferring data in WLAN, we need to pack data into TCP data packet, and then according to receiver's address, pack the TCP package into the TCP/IP package with IP information. Therefore, to transfer data in the WLAN, we need to develop the network protocol stack first. These protocol stacks exist in PC's operation system but is hard to run on embedded systems, which have very limited resources.

mxchipWNet™-DTU can pack the serial data into TCP/IP package automatically, and also unpack the payload from the TCP/IP package, send the payload to serial interface. So firmware does not require a specific data format on serial interface. Firmware hides the complicate network transmission function from the user's applications, it looks like that sender and receiver are connected with a traditional serial cable. And this is why it's called the Direct Transmission Mode.

Several major functions are adopted in this mode:

- All of the serial data received are transferred to payload of the TCP/UDP package on a specific port and delivered automatically.
- Fetch the entire payload from the TCP/UDP package on a specific port and send them to serial port.
- All Wi-Fi network connections and services are established automatically according to the predefined settings.
- Auto recovers from any network failure.

Two ways to enter the Direct Transmission Mode:

- Pull up the STATUS signal and reboot.
- Start the network connections in EMSP command mode, and then pull up the STATUS pin.

3.1 QUICK START

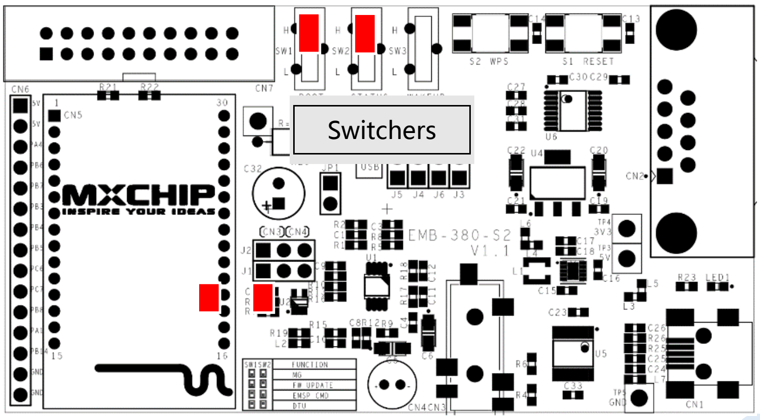
The following steps are listed to present how to demonstrate the Direct Transmission Mode on module, which has been configured with a factory setting from MXCHIP.

3.1.1 Hardware Connection

1. Set the BOOT pin and STATUS pin to high before powered on

According to Table 2.1, we need to set the correct signal: BOOT=1, STATUS=1 to enter the Direct Transmission Mode. The required inputs are the default state on the two pins, and if you are using an EMB-380-S test board, set the switchers as follow.

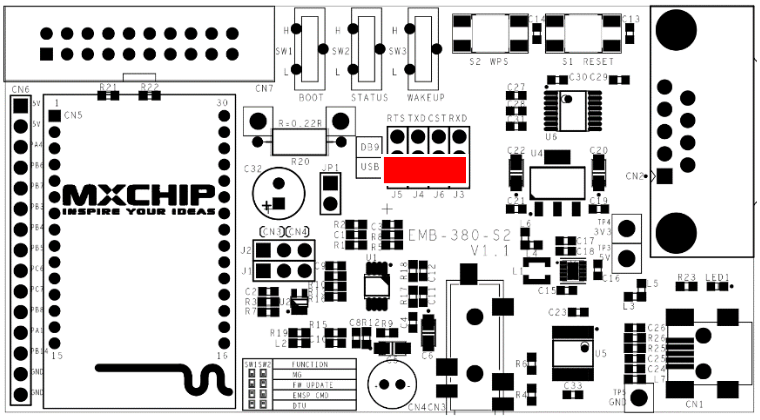
Figure 3.1 EMB-380-S switchers



2. **Connect module to PC using a serial cable.**

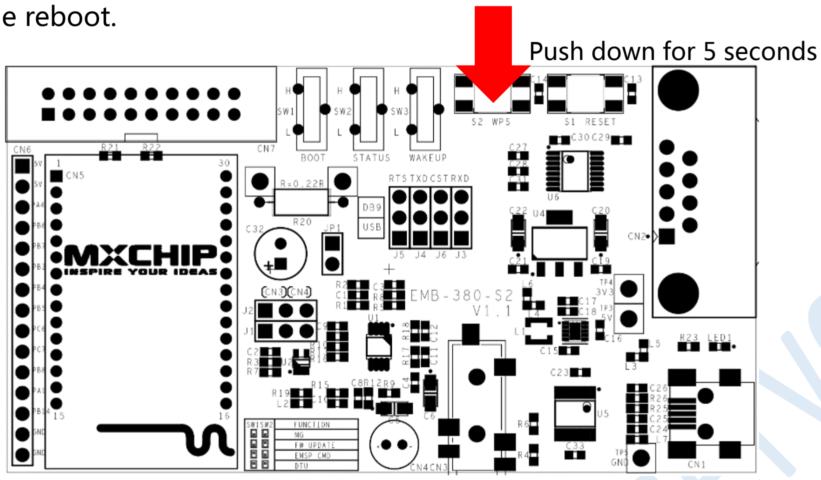
An USB-serial converter is available on EMW-380-S2 test board. It’ s very easy to connect the test board to PC via a mini USB cable. Set the jump J3, J4, J5, J6 to USB side and download the USB driver from http://www.ftdichip.com/Drivers/VCP.htm

Figure 3.2 EMB-380-S2 switchers to use serial/USB converter



3. Power on and restore the module with default settings.

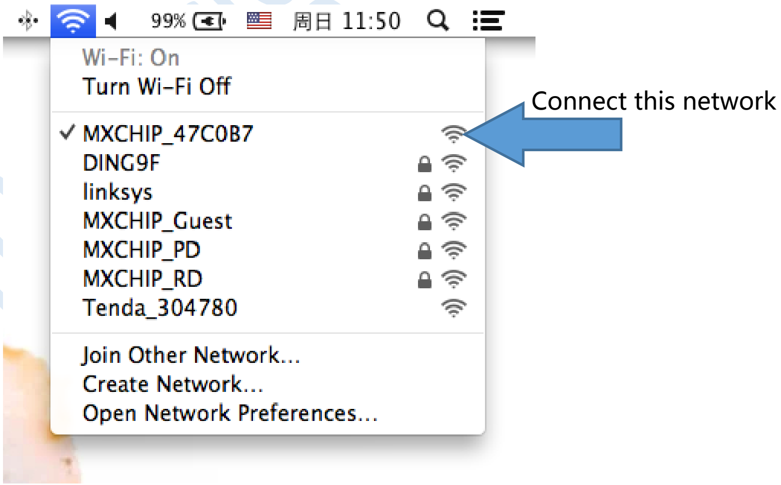
Use the methods listed in chapter 2.5 to restore the default settings. If you are using test board EMB-380-S2, it is easy to push down WPS/Default button for 5 seconds until the module reboot.



3.1.2 Connect to module using Wi-Fi

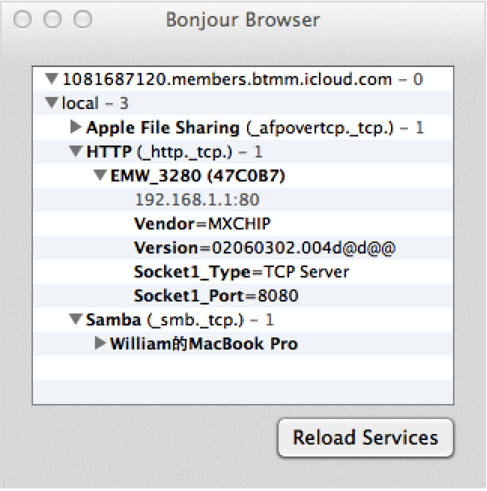
Open the Wireless connection window on PC, you can find a Wi-Fi network named "MXCHIP_XXXXXX" by searching available network nearby. "XXXXXX" is the last 3 bytes of module' s MAC address. Connect this network!

Figure 3.3 Connect to module

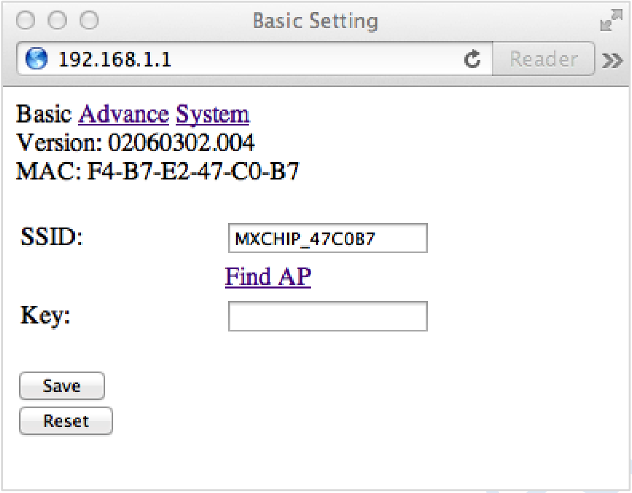


Once connected to the network, your pc would be assigned an IP address which within the scope of 192.168.1.XX. Now you can communicate with module, which has the static IP address: 192.168.1.1, and you can also locate module' s address by bonjour service. Login to module' s build-in web pages if you want to change module' s configuration, but we do not need to do this in this simple demonstration

Figure 3.4 Bonjour and WEB service



▼1081687120.members.btmm.icloud.com – 0
▼local – 3
▶Apple File Sharing (_afpovertcp._tcp.) – 1
▼HTTP (_http._tcp.) – 1
▼EMW_3280 (47C0B7)
192.168.1.1:80
Vendor=MXCHIP
Version=02060302.004d@d@@
Socket1_Type=TCP Server
Socket1_Port=8080
▼Samba (_smb._tcp.) – 1
▶William的MacBook Pro



192.168.1.1

Basic Advance System
Version: 02060302.004
MAC: F4-B7-E2-47-C0-B7

SSID: MXCHIP_47C0B7

Key:

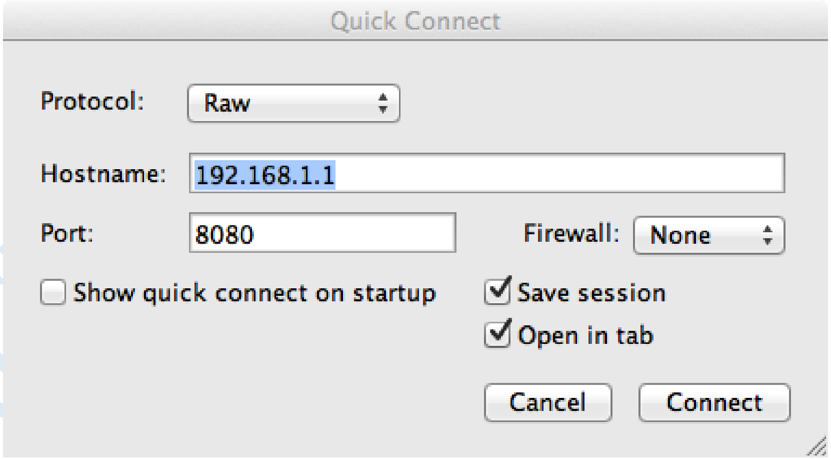
3.1.3 Data transmission between serial port and Wi-Fi

Use TCP&UDP debugger or terminal software to create a TCP client and connect to module on port 8080.

Typical Terminal software: HyperTerminal on Windows XP or SecureCRT on both Windows and MAC.

Note: download TCP&UDP debugger software at http://www.mxchip.com/uploadfiles/soft /EMW/TCP&UDP_Debugger_Setup.zip.

Figure 3.5 Create TCP conection to module on SecureCRT



Protocol: Raw

Hostname: 192.168.1.1

Port: 8080

Firewall: None

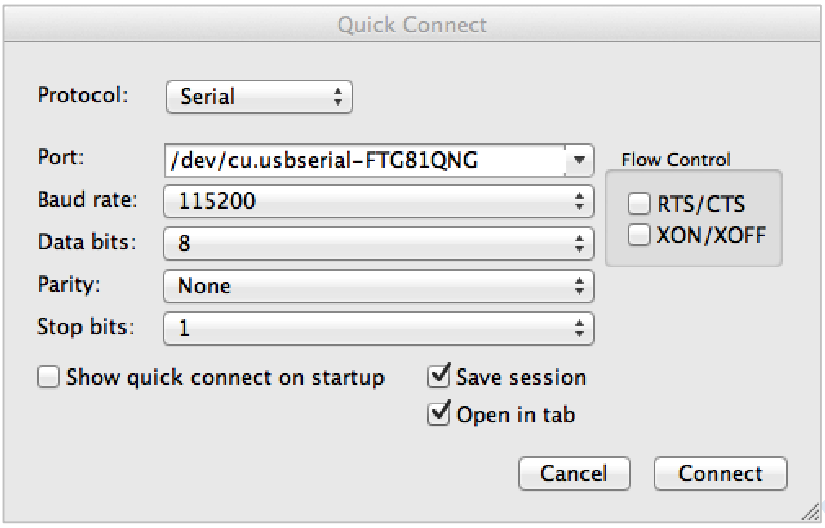
□ Show quick connect on startup

☑ Save session

☑ Open in tab

Open terminal software and connect to module from serial port with 115200 baud rate, 8 data length, 1 stop bit, and no flow control.

Figure 3.6 Create serial conection to module on SecureCRT



Quick Connect

Protocol: Serial

Port: /dev/cu.usbserial-FTG81QNG

Baud rate: 115200

Data bits: 8

Parity: None

Stop bits: 1

Flow Control

□ RTS/CTS

□ XON/XOFF

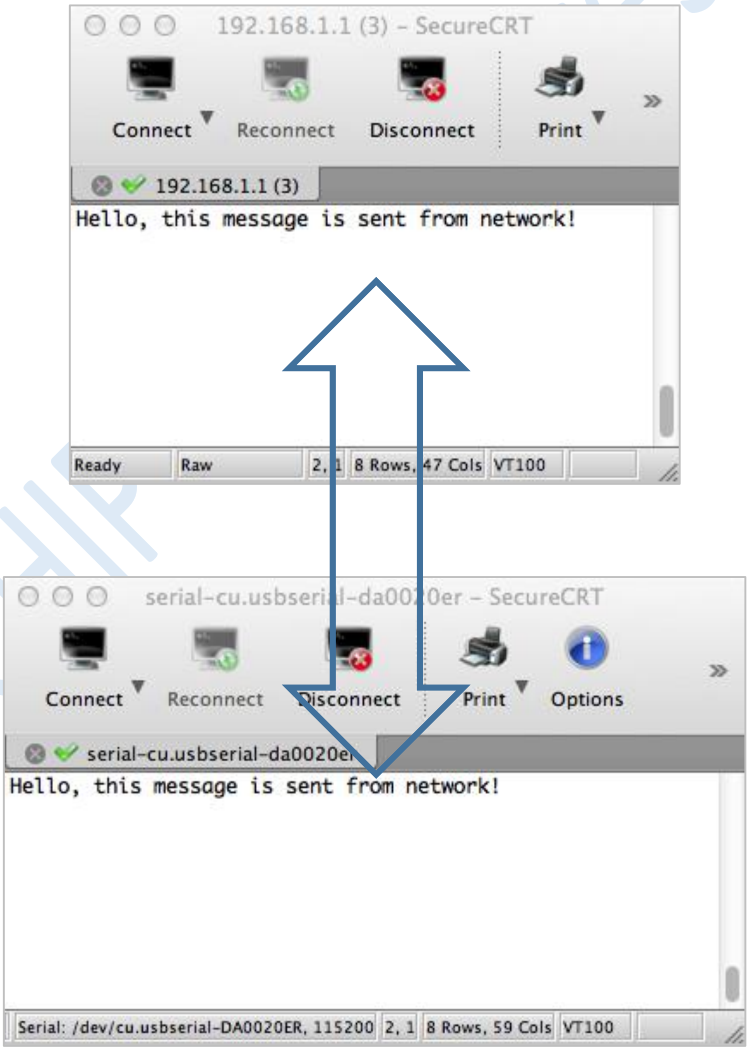
□ Show quick connect on startup

☑ Save session

☑ Open in tab

You may find that: any input in the serial terminal would be displayed on network terminal and any input on network terminal would be transferred to serial terminal.

Figure 3.6 Data transimission between two type of interface



192.168.1.1 (3) – SecureCRT

```
Hello, this message is sent from network!
```

Ready | Raw | 2, 1 | 8 Rows, 47 Cols | VT100

serial-cu.usbserial-da0020er – SecureCRT

```
Hello, this message is sent from network!
```

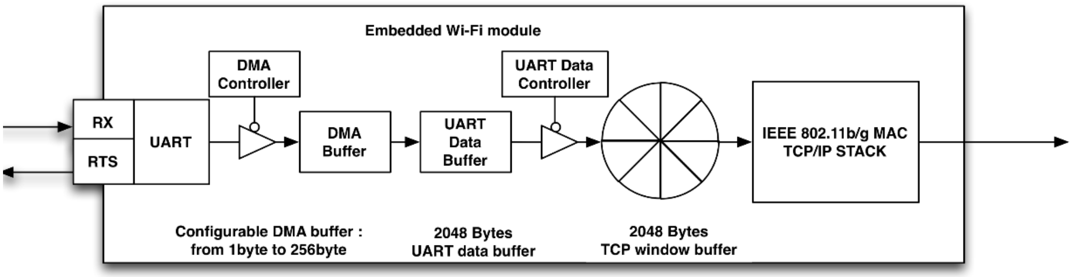
Serial: /dev/cu.usbserial-DA0020ER, 115200 | 2, 1 | 8 Rows, 59 Cols | VT100

3.2 Internal mechanism of Direct Transmission Mode

Under Direct Transmission Mode, all of the data conversion between serial port and network is performed automatically by firmware, and no need to be concerned in an ordinary use. Connect this network!

3.2.1 Serial Port=>Wireless Network

A flowchart of forwarding serial data to network:

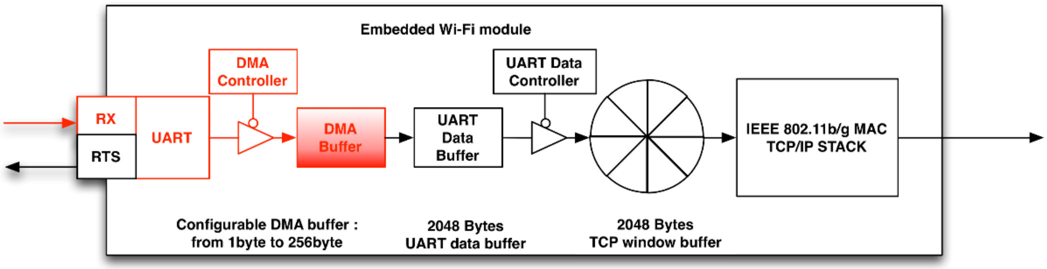

Embedded Wi-Fi module
DMA Controller
UART Data Controller
RX
RTS
UART
DMA Buffer
UART Data Buffer
IEEE 802.11b/g MAC TCP/IP STACK
Configurable DMA buffer : from 1byte to 256byte
2048 Bytes UART data buffer
2048 Bytes TCP window buffer

Firmware has three buffer used for temporary storage for serial data:

1. DMA Buffer: Temporary store the serial data received from DMA controller
2. UART Data Buffer: Firmware copy data from DMA buffer if DMA buffer is full
3. TCP Window Buffer: The data in TCP windows size are ready to be transferred to network handled by TCP/IP stack

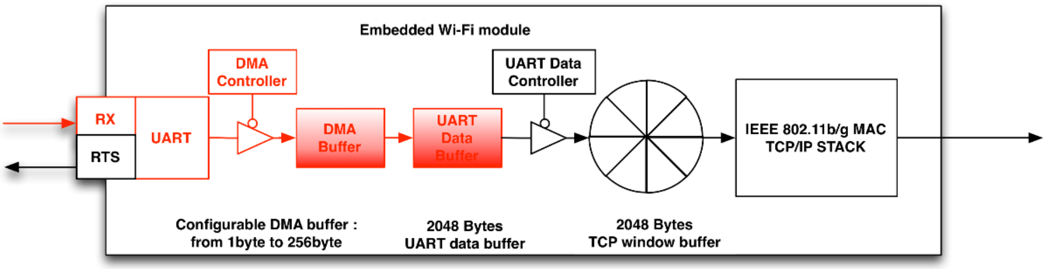
A brief description of the processing transfer data from serial port to network:

1. Hardware DMA controller deliver the received serial data to its DMA buffer. This part of the operation is completed by the DMA controller, and does not cost any CPU processing time.


Embedded Wi-Fi module
DMA Controller
UART Data Controller
RX
RTS
UART
DMA Buffer
UART Data Buffer
IEEE 802.11b/g MAC TCP/IP STACK
Configurable DMA buffer : from 1byte to 256byte
2048 Bytes UART data buffer
2048 Bytes TCP window buffer

2. Firmware copy serial data from DMA buffer to UART data buffer when DMA buffer is full or timeout. DMA buffer size and timeout can be configured by *DMA buffer size* in order to balance the performance and delay. A larger size of DMA buffer size reduce the frequency of the copy operation and give more CPU time to processing network stacks that improve the system’ s MAX of data

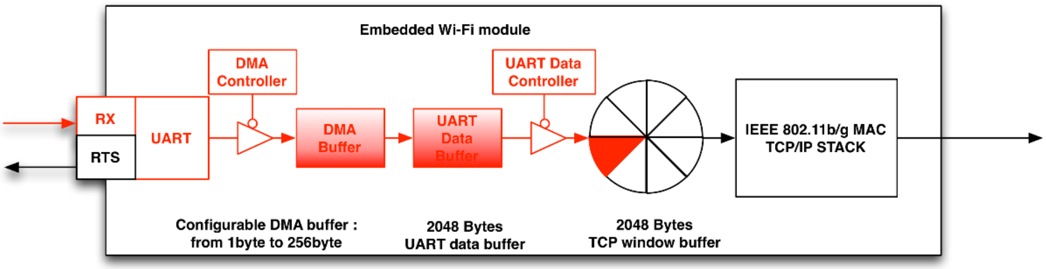
transmission speed in but reduce the system's real-time.


Embedded Wi-Fi module
DMA Controller
UART Data Controller
RX
RTS
UART
DMA Buffer
UART Data Buffer
IEEE 802.11b/g MAC TCP/IP STACK
Configurable DMA buffer : from 1byte to 256byte
2048 Bytes UART data buffer
2048 Bytes TCP window buffer

DMA buffer size

- 256 bytes: Copy immediately if receive 256 bytes or delay 100ms
- 128 bytes: Copy immediately if receive 128 bytes or delay 100ms
- 64 bytes: Copy immediately if receive 64 bytes or delay 100ms
- 32 bytes: Copy immediately if receive 32 bytes or delay 100ms
- 16 bytes: Copy immediately if receive 16 bytes or delay 100ms
- 8 bytes: Copy immediately if receive 8 bytes or delay 100ms
- No DMA buffer (default): Copy immediately if receive any serial data

3. Copy the data from UART data buffer to TCP window buffer according to rules defined by *Conversion Mode*. Data in TCP window buffer can be packaged in to the payload of one TCP/IP frame.


Embedded Wi-Fi module
DMA Controller
UART Data Controller
RX
RTS
UART
DMA Buffer
UART Data Buffer
IEEE 802.11b/g MAC TCP/IP STACK
Configurable DMA buffer : from 1byte to 256byte
2048 Bytes UART data buffer
2048 Bytes TCP window buffer

Conversion Mode has defined several conversion rules and will add more in future.

- Data Flow :

Firmware do not analyze data in UART data buffer but send them immediately if data exist. The content and length of the TCP/IP packages generated in this mode cannot be predicted but have the best real-time performance.

- Delay 20ms
- Delay 50ms
- Delay 100ms(default)
- Delay 150ms
- Delay 200ms

Similar in Data Flow mode, firmware does not analyze the data stored in UART data buffer, but send them in a delay if UART Data Buffer is not full. More serial data existed in one TCP/IP package in this mode, and the total quantity of the TCP/IP packages is reduced that improve the performance of network.

- Package Mode 1
- Package Mode 2

Firmware analyzes data in UART data buffer and packs the serial data which fit the predefined package structure. In this mode, a predefined serial data frame is only existed in one TCP/IP package, so it simplify the data analyzing operation on the other side of the network.

Data structure in Package Mode 1:

0x7E+Len (1 byte) +data+0xCE, (Length = data size + 1).

Firmware packs the full data frame into one TCP/IP package.

Example: Serial data: 7E 06 11 22 33 44 55 CE. Data in TCP/IP package: 7E 06 11 22 33 44 55 CE

Data structure in Package Mode 2:

0x7E+Len1 (1 byte) + Len2 (1 byte) +data+0xCE, (len1<<8+len2 = data size + 1).

Firmware only packs the data part into one TCP/IP package.

Example: Serial data: 7e 00 09 11 22 33 DD CE AA 12 DD CE. Data in TCP/IP package: 11 22 33 DD CE AA 12 DD

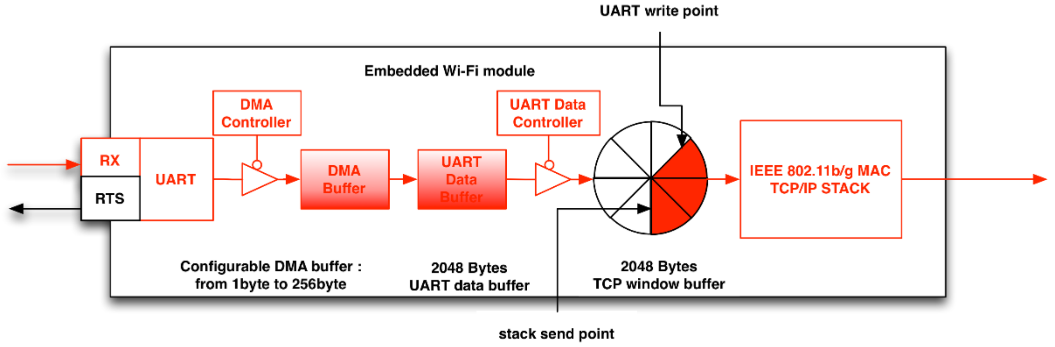
- FC mode

Pin: IO1 has configurable function defined by *IO1Mode.* When IO1 is defined to FC mode, IO1 detect the input signal:

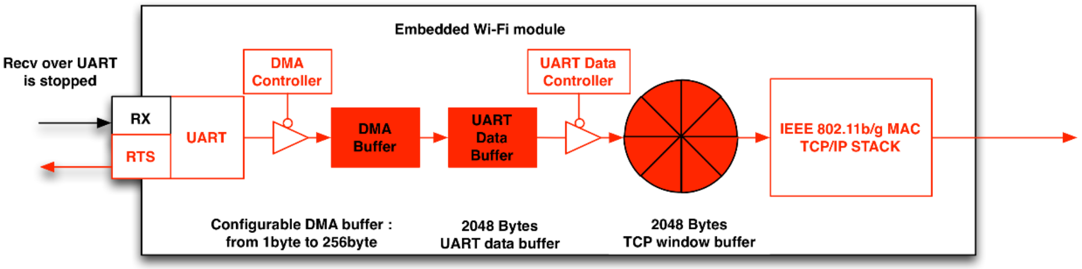
- IO1=0: Firmware stores the received serial data in UART Data Buffer
- IO1=1: Build up TCP/UDP package with the serial data in UART Data Buffer

Note: Set *DMA Buffer Size* to Zero, while using HC mode on IO1.

4. Step 1-3 are circled, firmware store the serial data to UART Data Buffer, then pack the data to TCP/IP package according to different method, TCP/IP stack and the IEEE 802.11 MAC/PHY send these data in wireless network at last.


UART write point
Embedded Wi-Fi module
DMA Controller
UART Data Controller
RX
RTS
UART
DMA Buffer
UART Data Buffer
IEEE 802.11b/g MAC TCP/IP STACK
Configurable DMA buffer : from 1byte to 256byte
2048 Bytes UART data buffer
2048 Bytes TCP window buffer
stack send point

5. If network is blocked, it result the TCP window Buffer filled with data, then no more data can be packed into TCP/IP package, it make the DMA Buffer and UART Data Buffer also full of data. Under this situation, serial port cannot receive any more data that lead to the loss of data on serial port.


Recv over UART is stopped
Embedded Wi-Fi module
DMA Controller
UART Data Controller
RX
RTS
UART
DMA Buffer
UART Data Buffer
IEEE 802.11b/g MAC TCP/IP STACK
Configurable DMA buffer : from 1byte to 256byte
2048 Bytes UART data buffer
2048 Bytes TCP window buffer

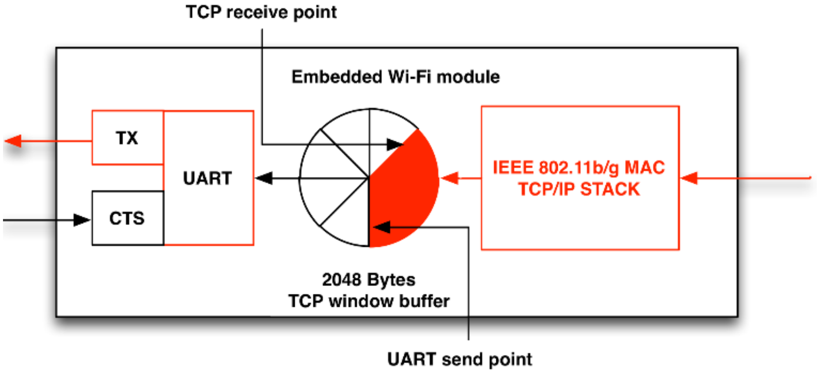
If *Hardware Flow Control* function is enabled, module would assert RTS signal under such situation to announce that sender should not send any more serial data. It can prevent the loss of data when network is blocked.

Note: Set the DMA Buffer Size above 16 bytes when you enable *hardware flow control* function on serial port.

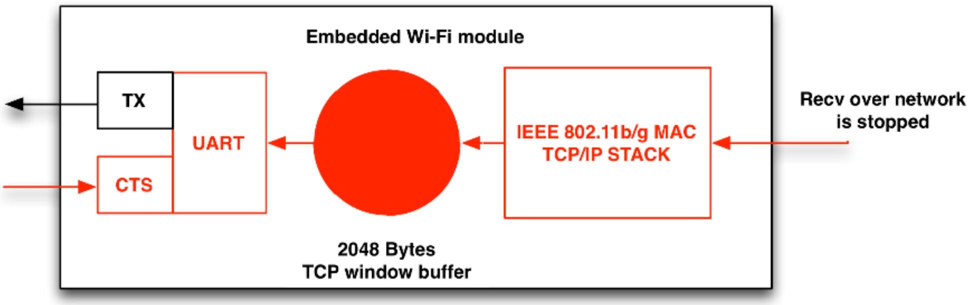
3.2.2 Wireless Network => Serial Port

It is quite simple to deliver data from network to serial port, because a TCP/IP network can adjust the data traffic automatically according to the data rate on serial port.。The flow chart is as below:

TCP/IP stack store the data in buffer and send them to serial port using DMA controller.



The UART data transmission is much slower than network data. In hardware flow control mode, the receiver may block the data sending on module by asserting CTS signal. Therefore, it is very common to see the TCP Window Buffer becomes full. When this happens, the receiving of the network data is blocked; it will reduce the network transmission rate. However, due to the TCP' s re-sending mechanism, the network data will not be lost. The following diagram shows an example where the module automatically stops sending the data to UART due to a CTS signal.

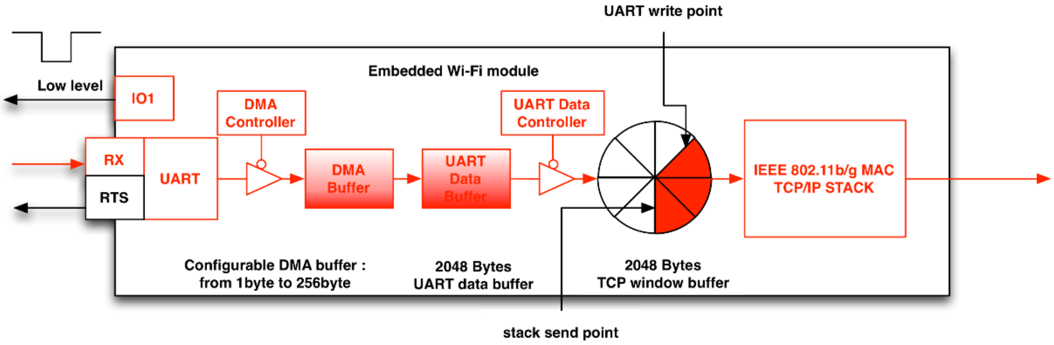


3.3 Half Duplex Mode (HDC mode)

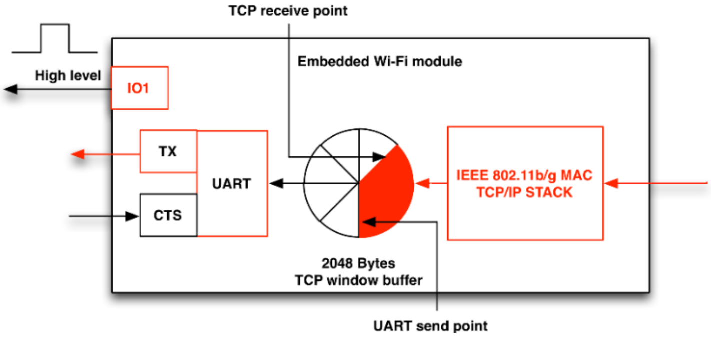
Half Duplex Mode is primarily used in RS485 communication. When the module is attached to a RS485 voltage converter, this mode controls the input and output state of the converter. A typical RS485 communication hardware connection is shown in Figure 1.3.

We can achieve this function by setting IO1 to HDC. IO1 in HDC Mode has the following function:

1. When the module receives UART data, IO1 outputs low (default state)



2. When the module sends out UART data, I01 outputs high.


TCP receive point
Embedded Wi-Fi module
High level
IO1
TX
UART
CTS
IEEE 802.11b/g MAC
TCP/IP STACK
2048 Bytes
TCP window buffer
UART send point

4 EMSP Command Mode

EMSP command is an ideal tool to communicate with mxchipWNet™-DTU firmware. Not like traditional ASCII code (like AT command), EMSP command is fast, reliable and easy programming for embedded device. Use EMSP commands, you can:

- Set the firmware' s configuration
- Execute Wi-Fi and TCP/IP operations
- Send or receive data on TCP/IP network, execute HTTP, FTP protocol operations

Any data input from serial must meet the specifications of EMSP command format, if not, firmware will not respond and the serial data is lost.

To enter the EMSP command mode, you just need to pull the module' s STATUS pin to low. If you are using EMW-380-S series test board, please set the STATUS switch to low.

Communication model

The command mode uses the typical master-slave communication protocol. The sender acts as the master/host, while the module is the slave/client. The communication starts with sender sending request to module, followed by module responding the request.

All the requests and responses go through verification and calculation to ensure the integrity and reliability of communication.

When the firmware initialization completes, the serial device can used the protocol provided by the specification to communicate with the module. Because each command in the communication protocol has different function, the processing time is not always the same. Therefore, you should wait for module' s return value after sending the next request.

Communication interface

1. Serial port interface

In default settings, the parameters of serial port interface are: 8 bits, no parity check, 1 stop bit, 115200 baud rate. Users can modify these parameters based on their requirement.

2. Wi-Fi interface

You can send EMSP commands to module from UDP protocol on port 8089 when module is connected to a wireless network. So you can send a command to all of the local modules using a UDP broadcast, and each module can return the result by an UDP unicast package.

4.1 QUICK START

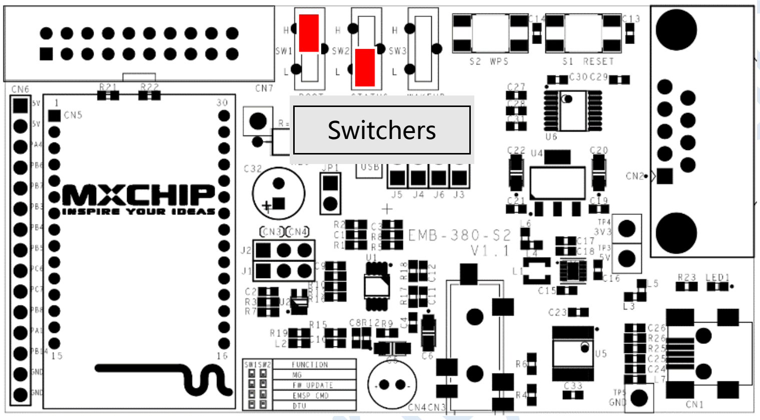
The following steps are listed to present how to demonstrate the EMSP command mode using a serial port.

4.1.1 Hardware Connection

1. Set BOOT pin = 1, STATUS = 0 before powered on

According to Table 2.1, we need to set the correct signal: BOOT=1, STATUS=0 to enter EMSP Command Mode. If you are using an EMB-380-S test board, set the switchers as follow.

Figure 4.1 EMB-380-S switchers



2. Connect module to PC using a serial cable.

Please refer to chapter 3.1.1 for details.

4.1.2 Send EMSP command using EMW Tool Box

EMW Tool Box is a software running on Windows PC, it can generate the required EMSP command according to the function selected on its window. Download this software at

http://www.mxchip.com/uploadfiles/soft/EMW/EMWToolBox_Setup.zip.

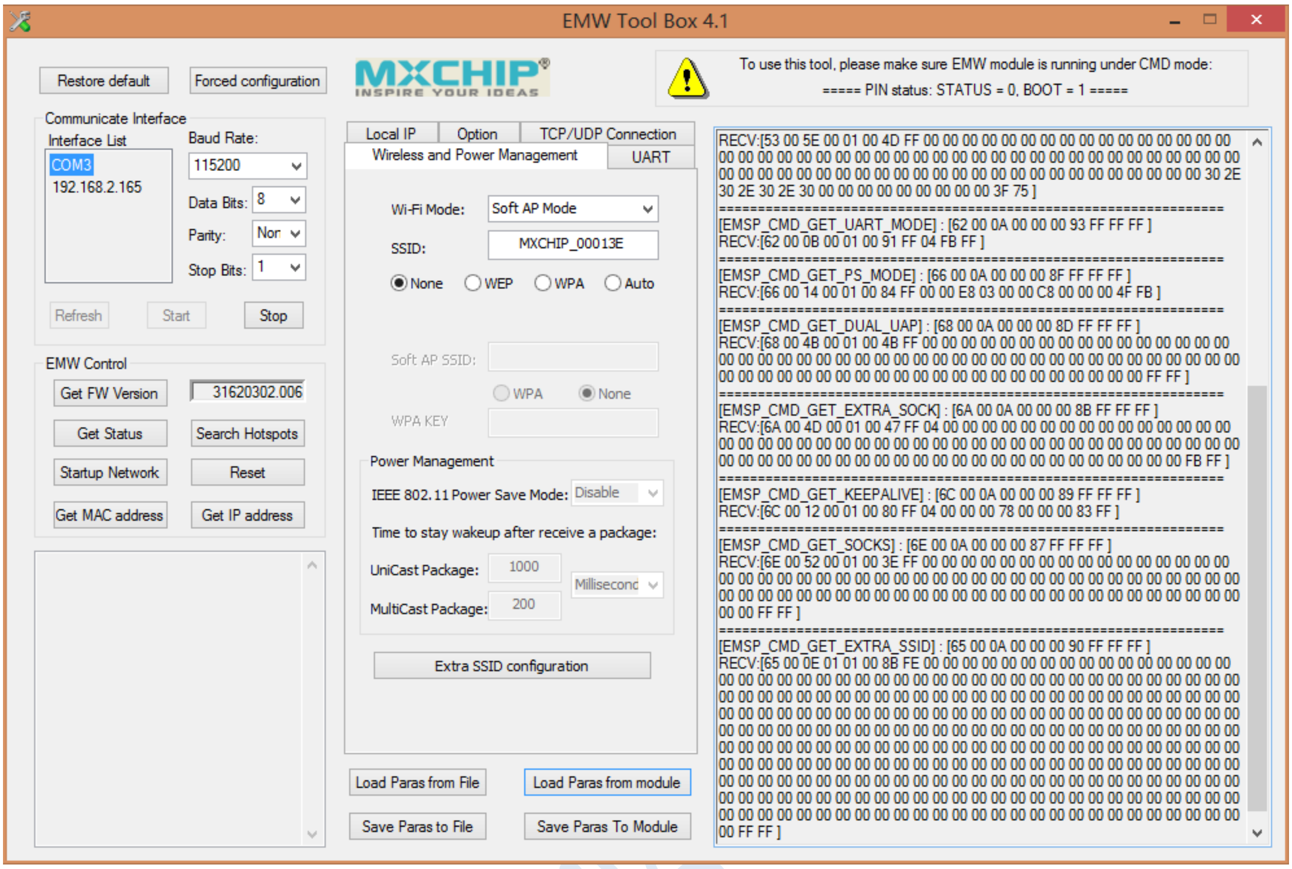
First, you need to open the correct COM port that is connected to module. And then click the "Get FW version" button, and the firmware version will be displayed on the right side of the button.

The EMSP commands sent and received will be displayed on the right side of the software; they are very useful if you are writing your own module controller software.

If you are trying to change settings, click the "Load Paras from Module" button first to read the parameters from module, then change the target parameters, and click "Save Paras to Module" button to save your settings.

Click "Startup Network" button to start Wi-Fi connection and TCP/UDP data link.

Figure 4.2 EMW Tool Box



EMW Tool Box 4.1

To use this tool, please make sure EMW module is running under CMD mode:
===== PIN status: STATUS = 0, BOOT = 1 =====

Communicate Interface
Interface List: COM3, 192.168.2.165
Baud Rate: 115200
Data Bits: 8
Parity: Non
Stop Bits: 1

EMW Control
Get FW Version: 31620302.006

Wireless and Power Management
Wi-Fi Mode: Soft AP Mode
SSID: MXCHIP_00013E
● None ○ WEP ○ WPA ○ Auto
Soft AP SSID:
○ WPA ● None
WPA KEY

Power Management
IEEE 802.11 Power Save Mode: Disable
Time to stay wakeup after receive a package:
UniCast Package: 1000
MultiCast Package: 200
Millisecond

```
RECV:[53 00 5E 00 01 00 4D FF 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 30 2E
30 2E 30 2E 30 00 00 00 00 00 00 00 00 00 3F 75 ]
==============================================================
[EMSP_CMD_GET_UART_MODE] : [62 00 0A 00 00 00 93 FF FF FF ]
RECV:[62 00 0B 00 01 00 91 FF 04 FB FF ]
==============================================================
[EMSP_CMD_GET_PS_MODE] : [66 00 0A 00 00 00 8F FF FF FF ]
RECV:[66 00 14 00 01 00 84 FF 00 00 E8 03 00 00 C8 00 00 00 4F FB ]
==============================================================
[EMSP_CMD_GET_DUAL_UAP] : [68 00 0A 00 00 00 8D FF FF FF ]
RECV:[68 00 4B 00 01 00 4B FF 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 FF FF ]
==============================================================
[EMSP_CMD_GET_EXTRA_SOCK] : [6A 00 0A 00 00 00 8B FF FF FF ]
RECV:[6A 00 4D 00 01 00 47 FF 04 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 FB FF ]
==============================================================
[EMSP_CMD_GET_KEEPALIVE] : [6C 00 0A 00 00 00 89 FF FF FF ]
RECV:[6C 00 12 00 01 00 80 FF 04 00 00 00 78 00 00 00 83 FF ]
==============================================================
[EMSP_CMD_GET_SOCKS] : [6E 00 0A 00 00 00 87 FF FF FF ]
RECV:[6E 00 52 00 01 00 3E FF 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 FF FF ]
==============================================================
[EMSP_CMD_GET_EXTRA_SSID] : [65 00 0A 00 00 00 90 FF FF FF ]
RECV:[65 00 0E 01 01 00 8B FE 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 FF FF ]
```

4.2 EMSP Command Specification

EMSP command consists of a protocol header (8 bytes) and a data block. The length of the data block is not fixed, but can be no longer than 256 bytes. The command format is shown as follows

[<command><length><result><head checksum>][<data><data checksum>]

Protocol Header

A protocol header consists of one command segment (2 bytes), one length segment (2 bytes), one result segment (2 bytes, returned by the module), and a header checksum (2 bytes). The format is shown as follows

[<command><length><result><head checksum>]

<command> : Command segment, 2 bytes

<length> : Length segment, 2 bytes. This is the length of the whole EMSP command, including the protocol header and the data block.

<result> : Result segment, 2 bytes. The request and response should use the same protocol header; this segment is only effective in the response packet.

<head checksum>： Header checksum, 2 bytes. It is used to verify the integrity of the protocol header.

Data block

The data block includes all the data used by the current command, and one data checksum (2 bytes) at the end. The format is shown as below:

[<data><data checksum>]

<data>：Data. Its length is not fixed, can be calculated from the length segment in the protocol header.

<data checksum>：Data checksum, 2 bytes. It is used to verify the integrity of the data block.

Note: The checksums of the protocol header and data block are independent. The checksum in protocol header only verifies the protocol header while the checksum in the data block only verifies the data block.

Verification algorithm

Please reference the following C code:

```
u16 calc_sum(void *data, u32 len)
{
   u32 cksum=0;
   __packed u16 *p = data;
   while (len > 1){
      cksum += *p++;
      len -=2;
   }
   if (len){
      cksum += *(u8 *)p;
   }
   cksum = (cksum >> 16) + (cksum & 0xffff);
   cksum += (cksum >>16);
   return ~cksum;
}
```

4.3 Command Description

There are two kinds of EMSP commands:

Static configuration commands

These commands are used to configure and read module’s parameters. These parameters will be written into module’s Flash, hence it normally requires reboot to make these commands take effect. If the module enters the data mode after reboot, the module will automatically establish the network communication network based on the parameters set by these commands.

Dynamic control commands

These commands can dynamically control module’s different functions, temporarily change working parameters, send or receive data. However, any changes resulted by these commands will not be saved in module. Once rebooted, the changes are lost. The dynamic control commands are introduced to increase module’s flexibility.

The following are the definitions to the date transmitted in the commands

typedef unsigned char	u8 :	u8 represent 8bit data
typedef unsigned short int	u16 :	u16 represents 16bit data
typedef unsigned int	u32 :	u32 represents 32bit data

4.3.1 Static Configuration Commands

EMSP_CMD_GET_CONFIG (COMMAND ID 0002)

This command retrieves basic configuration information from the module.

Host sends: 02 00 0A 00 00 00 F3 FF FF FF

Module returns: 02 00 A9 00 01 00 53 FF <data><data checksum>

<data> structure:

Table 4.1. Basic Parameters

NAME	TYPE	LENGTH	FUNCTION
Wi-Fi mode	U8	1	1: Station mode 2: Soft AP mode 3: Station mode + Soft AP mode(the Soft AP parameters in this mode is defined in command: EMSP_CMD_SET_DUAL_UAP)
SSID	U8	32	Wi-Fi network name
WEP Key	U8	16	WEP security key is available if Security Mode is set to WEP
WEP key length	U8	1	The key length of WEP security key available if Security Mode is set to WEP 0: The same as setting SECURITY MODE to NONE 5: WEP security key is 10 hex numbers (5 ACSII characters) 13: WEP security key is 26 hex numbers (13 ACSII characters)
Local IP address	U8	16	Module’s static IP address, it will be overridden if firmware get an address successfully from DHCP server. It is presented by string using dot-decimal notation. Example: “192.168.0.1”

<table>
<tr><th>NAME</th><th>TYPE</th><th>LENGTH</th><th>FUNCTION</th></tr>
<tr><td>Remote IP address</td><td>U8</td><td>16</td><td>The IP address where module is trying to connect or send data if module is setting to TCP client mode and UDP unicast mode. It is ignored if DNS is enabled.
The format is the same as Local IP address.</td></tr>
<tr><td>Net mask</td><td>U8</td><td>16</td><td>Net mask while using a static IP address, same format as Local IP address.</td></tr>
<tr><td>Gateway</td><td>U8</td><td>16</td><td>Gateway IP address while using a static IP address, same format as Local IP address.</td></tr>
<tr><td>Port_H</td><td>U8</td><td>1</td><td>Upper byte of the TCP/UDP socket port</td></tr>
<tr><td>Port_L</td><td>U8</td><td>1</td><td>Lower byte of the TCP/UDP socket port</td></tr>
<tr><td>Protocol_1</td><td>U8</td><td>1</td><td><table>
<tr><th>Protocol_1</th><th>Protocol_2</th><th>Function</th></tr>
<tr><td>0</td><td>0</td><td>TCP Server Mode</td></tr>
<tr><td>0</td><td>1</td><td>UDP Broadcast Mode</td></tr>
<tr><td>1</td><td>0</td><td>TCP Client Mode</td></tr>
<tr><td>1</td><td>1</td><td>UDP Unicast Mode</td></tr>
</table></td></tr>
<tr><td>DHCP</td><td>U8</td><td>1</td><td>0: Use static local IP address
1: Use DHCP service to get local IP address</td></tr>
<tr><td>Protocol_2</td><td>U8</td><td>1</td><td>Refer to Protocol_1</td></tr>
<tr><td>Baud Rate</td><td>U8</td><td>1</td><td>Serial port baud rate:
0: 9600 1: 19200 2: 38400 3: 57600 4:115200
5:230400 6: 460800 7:921600 8:1843200 9: 3686400
10: 4800 11: 2400 12: 1200</td></tr>
<tr><td>DMA Buffer Size</td><td>U8</td><td>1</td><td>DMA Buffer Size:
0: No DMA buffer 1: 8 bytes 2: 16bytes 3: 32bytes
4: 64 bytes 5:128 bytes 6:256 bytes</td></tr>
<tr><td>Flow Control</td><td>U8</td><td>1</td><td>0: Disable 1: Enable</td></tr>
<tr><td>Parity</td><td>U8</td><td>1</td><td>0: None 1: Even 2: Odd</td></tr>
<tr><td>Data Bits</td><td>U8</td><td>1</td><td>0: 8bits</td></tr>
<tr><td>Stop Bits</td><td>U8</td><td>1</td><td>0: 1bits 1: 0.5bits 1: 2bits 1: 1.5bits</td></tr>
<tr><td>IO1</td><td>U8</td><td>1</td><td>0: None 1: FC Mode 2: HDC Mode</td></tr>
<tr><td>Security Mode</td><td>U8</td><td>1</td><td>0: WEP 1: WPA/WPA2 PSK 2: None 3: WEP Hex
4: Auto</td></tr>
<tr><td>Key</td><td>U8</td><td>32</td><td>WPA，WPA2 PSK Security Key</td></tr>
</table>

EMSP_CMD_SET_CONFIG (COMMAND ID 0003)

This command configures module’s parameters. Once changed, the module has to be rebooted in order for these parameters to take effect.

Host sends: 03 00 A9 00 00 00 53 FF <data><data checksum>

Module returns: 03 00 A9 00 01 00 52 FF <data><data checksum>

Check Table 4.1 for details in <data> structure.

EMSP_CMD_SET_DNS (COMMAND ID 0052)

This command enables DNS resolution function and configures DNS related settings.

Host sends: 52 00 5E 00 00 00 4F FF <data><data checksum>

Module returns: 52 00 0A 00 01 00 A2 FF FF FF

<data> structure:

Table 4.2. DNS Parameters

NAME	TYPE	LENGTH	FUNCTION
ENABLE	U32	1	0: DNS disabled 1: DNS enabled
Domain Name	U8	64	The string of domain name where module is trying to connect or send data if module is setting to TCP client mode and UDP unicast mode.
DNS Server IP Address	U8	16	DNS server IP address, it will be overridden if firmware get an address successfully from DHCP server. It is presented by string using dot-decimal notation. Example: “192.168.0.1”

EMSP_CMD_GET_DNS (COMMAND ID 0053)

This command gets the module’ s DNS related parameters.

Host sends: 53 00 0A 00 00 00 A2 FF FF FF

Module returns: 53 00 5E 00 01 00 4F FF <data><data checksum>

Check Table 4.2 for details in <data> structure.

EMSP_CMD_SET_EXTRA_SSID (COMMAND ID 0064)

This command sets extra four Wi-Fi network parameters. Module can roam in these predefined Wi-Fi networks (Wi-Fi network defined in EMSP_CMD_GET_CONFIG is also included) under station mode.

Host sends: 64 00 0E 01 00 00 8D FE <data><data checksum>

Module returns: 64 00 0A 00 01 00 90 FF FF FF

<data> structure:

Table 4.3. Extra Wi-Fi networks Parameters

NAME	TYPE	LENGTH	FUNCTION
SSID_1	U8	32	Wi-Fi network 1 name
Key_1	U8	32	Security key of Wi-Fi network 1
Security Mode_1	U8	1	Wi-Fi security 1 mode 0: WEP 1: WPA/WPA2 PSK 2: None 4: Auto
SSID_2	U8	32	Wi-Fi network 2 name
Key_2	U8	32	Security key of Wi-Fi network 2
Security Mode_2	U8	1	Wi-Fi security 2 mode 0: WEP 1: WPA/WPA2 PSK 2: None 4: Auto
SSID_3	U8	32	Wi-Fi network 3 name
Key_3	U8	32	Security key of Wi-Fi network 3
Security Mode_3	U8	1	Wi-Fi security 3 mode 0: WEP 1: WPA/WPA2 PSK 2: None 4: Auto
SSID_4	U8	32	Wi-Fi network 4 name
Key_4	U8	32	Security key of Wi-Fi network 4
Security Mode_4	U8	1	Wi-Fi security 4 mode 0: WEP 1: WPA/WPA2 PSK 2: None 4: Auto

EMSP_CMD_GET_EXTRA_SSID (COMMAND ID 0065)

This command gets extra four Wi-Fi network parameters.

Host sends: 65 00 0A 00 00 00 90 FF FF FF

Module returns: 65 00 0E 01 01 00 8B FE <data><data checksum>

Check Table 4.3 for details in <data> structure.

EMSP_CMD_SET_EXTRA_SOCKET (COMMAND ID 0069)

This command sets an extra socket connection. Serial data can be delivered by both main socket and this extra socket, and any data on these sockets can be sent to serial port.

Host sends: 69 00 4D 00 00 00 49 FF <data><data checksum>

Module returns: 69 00 0A 00 01 00 8B FF FF FF

<data> structure:

Table 4.4. Extra Wi-Fi networks Parameters

NAME	TYPE	LENGTH	FUNCTION
Protocol	U8	1	0: TCP Server Mode 1: TCP Client Mode 2: UDP Unicast Mode 3: UDP Broadcast Mode 4: Disable

NAME	TYPE	LENGTH	FUNCTION
Port	U16	1	Socket port number
Remote address	U8	64	Remote server address used in TCP client mode and UDP unicast mode. Input IP address or domain name.

EMSP_CMD_GET_EXTRA_SSID (COMMAND ID 006A)

This command gets configuration of the extra socket connection.

Host sends: 6A 00 0A 00 00 00 8B FF FF FF

Module returns: 6A 00 4D 00 01 00 47 FF <data><data checksum>

Check Table 4.5 for details in <data> structure.

EMSP_CMD_SET_KEEPALIVE (COMMAND ID 006B)

When working as a TCP and data is not transmitting, module sends keep-alive package periodically, and the other side should returns to module. If module does not receive the returned keep-alive package, module will start to count the number of failures. If the count exceeds the max retry number, module will terminate the current TCP link, recycle the resources and try to reconnect the TCP server. The count will return to zero if the keep-alive package is returned or data transmission is successful.

Time needed to detect a broken TCP link: Retry number x Retry interval

Host sends: 6B 00 12 00 00 00 82 FF <data><data checksum>

Module returns: 6B 00 0A 00 01 00 89 FF FF FF

<data> structure:

Table 4.5. TCP Keep-alive Parameters

NAME	TYPE	LENGTH	FUNCTION
Retry number	U32	1	Max TCP keep-alive package retry number
Retry interval	U32	1	Time interval between two retry (Unit: second)

EMSP_CMD_GET_KEEPALIVE (COMMAND ID 006C)

This command gets TCP keep-alive parameters

Host sends: 6C 00 0A 00 00 00 89 FF FF FF

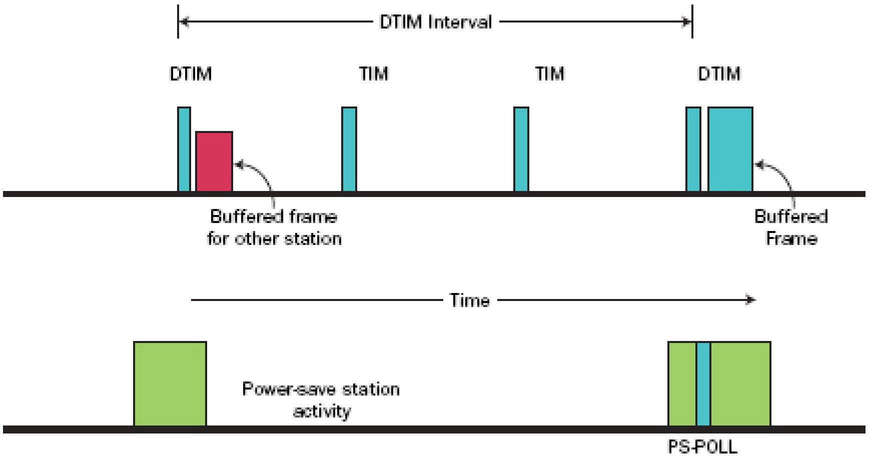
Module returns: 6C 00 12 00 01 00 80 FF <data><data checksum>

Check Table 4.4 for details in <data> structure.

EMSP_CMD_SET_PS_MODE (COMMAND ID 0063)

This command set IEEE 802.11 power save mode parameters. IEEE 802.11 power save mode can reduce the RF power consumption when data is not transmitting. When IEEE power save mode is enabled, 802.11 MAC and PHY will enter sleep mode between two DTIM intervals.

The DTIM is about 100ms -500ms depends on the settings on the AP.



Every time module wakes up, it will check if AP has buffered data for module and receive them immediately. After has received them successfully, module will continue to sleep. But if more data is arrived when module is asleep, module can receive only at the next DTIM time point. Delay is generated in this situation.

So, after module has received the buffered data, it should not go to sleep immediately but wait a predefined time. When data is received during this time, no delay would be generated. The time that module remains awake after receive a Wi-Fi data package is defined by: Interval x Interval Unit.

Host sends: 63 00 14 00 00 00 88 FF 00 00 E8 03 00 00 C8 00 00 00 4F FB <data><data checksum>

Module returns: 63 00 0A 00 01 00 91 FF FF FF

<data> structure:

Table 4.6. TCP Keep-alive Parameters

NAME	TYPE	LENGTH	FUNCTION
ENABLE	U8	1	0: Disable 1: Enable Power Save Mode
Interval Unit	U8	1	0=Millisecond 1=Time between two beacon (Defined in AP) Default is 0, Millisecond

NAME	TYPE	LENGTH	FUNCTION
Unicast Interval	U32	1	Stay awake after receive a unicast Wi-Fi package, default is 1000 (1000ms)
Multicast Interval	U32	1	Stay awake after receive a multicast Wi-Fi package, default is 100 (100ms)

EMSP_CMD_GET_PS_MODE (COMMAND ID 0066)

This command gets IEEE power save mode parameters.

Host sends: 66 00 0A 00 00 00 8F FF FF FF

Module returns: 66 00 14 00 01 00 84 <data><data checksum>

Check Table 4.5 for details in <data> structure.

EMSP_CMD_SET_DUAL_UAP (COMMAND ID 0067)

In Station + Soft AP mode, module can communicate with cloud service on the internet through a wireless router, and accept local controller in soft AP mode. Wi-Fi parameters in station mode are defined by command: EMSP_CMD_GET_CONFIG. This command defines the soft AP parameters in this mode.

Host sends: 67 00 4B 00 00 00 4D FF <data><data checksum>

Module returns: 67 00 0A 00 01 00 8D FF FF FF

<data> structure:

Table 4.7. Soft AP Parameters in DUAL Mode

NAME	TYPE	LENGTH	FUNCTION
SSID	U8	32	Wi-Fi network name
Key	U8	32	Security key
Security Mode	U8	1	Wi-Fi security 1 mode(WEP is not supported in this mode) 1: WPA/WPA2 PSK 2: None

EMSP_CMD_GET_DUAL_UAP (COMMAND ID 0068)

This command gets TCP keep-alive parameters

Host sends: 68 00 0A 00 00 00 8D FF FF FF

Module returns: 68 00 4B 00 01 00 4B FF <data><data checksum>

Check Table 4.6 for details in <data> structure.

EMSP_CMD_GET_UART_MODE (COMMAND ID 0062)

This command gets the Conversion Mode which package serial data to TCP/IP package.

Host sends: 62 00 0A 00 00 00 93 FF FF FF

Module returns: 62 00 0B 00 01 00 91 FF <data><data checksum>

<data> structure: (Refer chapter 3.2.1 for the detailed function of each mode)

Table 4.8. Conversion Mode Parameters

NAME	TYPE	LENGTH	FUNCTION
Conversion Mode	U8	1	0: Data Flow 1: Package Mode 1 2: Delay 20ms 3: Delay 50ms 4: Delay 100ms 5: Delay 150ms 6: Delay 200ms 7: Package Mode 2

EMSP_CMD_SET_UART_MODE (COMMAND ID 0061)

This command sets the Conversion Mode, which package serial data to TCP/IP package.

Host sends: 61 00 0B 00 00 00 93 FF <data><data checksum>

Module returns: 61 00 0A 00 01 00 93 FF FF FF

Check Table 4.3 for details in <data> structure.

EMSP_CMD_GET_NAME (COMMAND ID 0046)

This command gets module’ s name, the name will be presented by bonjour service.

Host sends: 46 00 0A 00 00 00 AF FF FF FF

Module returns: 46 00 32 00 01 00 86 FF <data><data checksum>

Table 4.9. Bonjour Name Parameters

NAME	TYPE	LENGTH	FUNCTION
Name	U8	40	The module’ s name string with a fixed size of 40 bytes, it is used by bonjour service.

EMSP_CMD_SET_NAME (COMMAND ID 0047)

This command sets module’ s name, the name will be presented by bonjour service.

Host sends: 47 00 32 00 00 00 86 <data><data checksum>

Module returns: 47 00 0A 00 01 00 AD FF FF FF

Check Table 4.4 for details in <data> structure.

EMSP_CMD_GET_VER (COMMAND ID 006F)

This command gets module’ s firmware version.

Host sends: 6F 00 0A 00 00 00 86 FF FF FF

Module returns: 6F 00 1A 00 01 00 75 FF <data><data checksum>

<data>: Current firmware’ s version(12 characters)

EMSP_CMD_GET_MAC_ADDR (COMMAND ID 000C)

This command gets module’ s MAC address.

Host sends: 0C 00 0A 00 00 00 E9 FF FF FF

Module returns: 0C 00 10 00 01 00 E2 FF <data><data checksum>

<data>: The MAC address of the module. It is a hex number with a fixed length of 6 bytes.

4.3.2 Dynamic Control Commands

EMSP_CMD_RESET (COMMAND ID 0001)

This command soft resets the module.

Host sends: 01 00 0A 00 00 00 F4 FF FF FF

Module returns: 01 00 0A 00 01 00 F3 FF FF FF

EMSP_CMD_START (COMMAND ID 0005)

This command enables module’ s Wi-Fi connection and TCP connection.

Host sends: 05 00 0A 00 00 00 F0 FF FF FF

Module returns: 05 00 0A 00 01 00 EF FF FF FF

EMSP_CMD_GET_STATUS (COMMAND ID 0008)

This command is used to get the module’ s current running status

Host sends: 08 00 0A 00 00 00 ED FF FF FF

Module returns: 08 00 0E 00 01 00 E8 FF <data><data checksum>

<data> structure:

Table 4.10. Current Running Status

NAME	TYPE	LENGTH	FUNCTION
System	U8	1	Reversed

Wi-Fi	U8	1	Bit0=0: Disconnected in station mode
			Bit0=1: Connected in station mode
			Bit1=0: Soft AP mode not established
			Bit1=1: Soft AP mode established

Socket1	U8	1	0: Socket1 is disabled 1: Socket1 disconnected (Under TCP client mode) 2: No TCP client is connected (Under TCP server mode) 3: Socket1 connected
Socket2	U8	1	0: Socket2 is disabled 1: Socket2 disconnected (Under TCP client mode) 2: No TCP client is connected (Under TCP server mode) 3: Socket2 connected

EMSP_CMD_GET_IP (COMMAND ID 0040)

This command gets the current IP address.

Host sends: 40 00 0A 00 00 00 B5 FF FF FF

Module returns: 40 00 3A 00 01 00 84 FF <data><data checksum>

<data>: ……, The data structure of the data block is shown as follows

Table 4.11. Current IP Address

NAME	TYPE	LENGTH	FUNCTION
Local IP address	U8	16	Current IP address. It is presented by string using dot-decimal notation. Example: "192.168.0.1"
Net mask	U8	16	Current net mask
Gateway	U8	16	Current gateway

EMSP_CMD_OPEN_SOCKET (COMMAND ID 0045)

This command establishes a TCP/UDP socket1 connection. The remote server address in the command could be either an IP address or a domain name.

Host sends: 45 00 4E 00 00 00 6C FF <data><data checksum>

Module returns: 45 00 0A 00 01 00 AF FF FF FF

<data> structure:

Table 4.12. Open Socket Parameters

<table>
<tr><th>NAME</th><th>TYPE</th><th>LENGTH</th><th>FUNCTION</th></tr>
<tr><td>Protocol_2</td><td>U8</td><td>1</td><td rowspan="2"><table>
<tr><th>Protocol_1</th><th>Protocol_2</th><th>Function</th></tr>
<tr><td>0</td><td>0</td><td>TCP Server Mode</td></tr>
<tr><td>0</td><td>1</td><td>UDP Broadcast Mode</td></tr>
<tr><td>1</td><td>0</td><td>TCP Client Mode</td></tr>
<tr><td>1</td><td>1</td><td>UDP Unicast Mode</td></tr>
</table></td></tr>
<tr><td>Protocol_1</td><td>U8</td><td>1</td></tr>
<tr><td>Port_H</td><td>U8</td><td>1</td><td>Upper byte of the TCP/UDP socket port</td></tr>
<tr><td>Port_L</td><td>U8</td><td>1</td><td>Lower byte of the TCP/UDP socket port</td></tr>
<tr><td>Server address</td><td>U8</td><td>64</td><td>The address of the TCP server or UDP send target.
Either domain name or IP address can write to this part.
Example: “192.168.0.1” or “www.google.com”</td></tr>
</table>

EMSP_CMD_CLOSE_SOCKET (COMMAND ID 0044)

This command closes the current TCP/UDP socket1.

Host sends: 44 00 0A 00 00 00 B1 FF FF FF

Module returns: 44 00 0A 00 01 00 B0 FF FF FF

EMSP_CMD_WIFI_STOP (COMMAND ID 004A)

This command stops the Wi-Fi connection.

Module returns: 4A 00 0A 00 01 00 AA FF FF FF

EMSP_CMD_WIFI_CONNECT (COMMAND ID 004B)

This command starts the Wi-Fi connection based on the predefined parameters.

Host sends: 4B 00 0A 00 00 00 AA FF FF FF

Module returns: 4B 00 0A 00 01 00 A9 FF FF FF

EMSP_CMD_SCAN_AP (COMMAND ID 0004)

This command gets all the available AP and their signal strength in the detectable range.

Host sends: 04 00 0A 00 00 00 F1 FF FF FF

Module returns: 04 00 <length><result><head checksum><data><data checksum>

<result>: The number of access points

<data> structure: result x Scan_Result_Type

Table 4.13. Scan Result Type

NAME	TYPE	LENGTH	FUNCTION
SSID	String	End with 0x0	Name of the Wi-Fi network
RSSI	String	End with 0x0	Signal strength

EMSP_CMD_SEND_DATA (COMMAND ID 0006)

This command sends data to the network through socket1

Host sends: 06 00 <length> 00 00 <head checksum><data><data checksum>

<length>: The actual command length

<data>: The data to be sent

Module returns: 06 00 0A 00 00 00<result><head checksum> FF FF

<result>: Successfully sent <result> bytes of data to the destination through network. 0 means “failed to send data” .

EMSP_CMD_RECV_DATA (COMMAND ID 0007)

This command is used to receive data from socket1, module returns immediately after receive data from socket1.

Module returns: 07 00 <length><result><head checksum><data><data checksum>

<length>: The length of the return command.

<result>: Successfully received <result> bytes of data through network. 0 means no data received.

<data>: The data received.

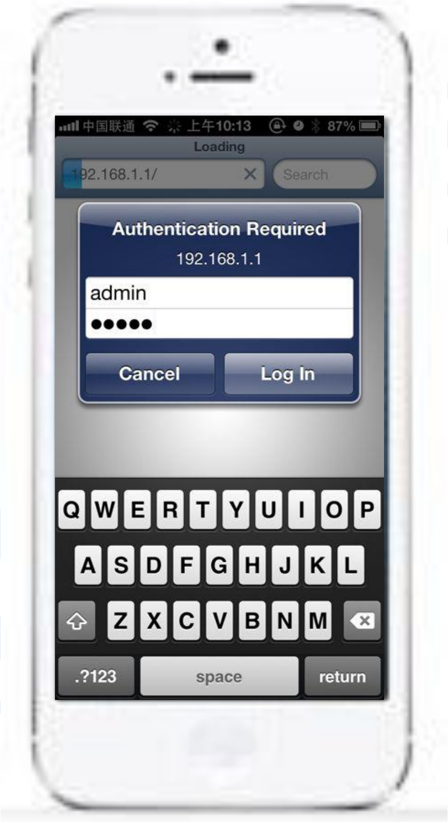
5 Methods of Configuration

5.1 Build-in Web Pages Method

We can use any PC or Smartphone with Wi-Fi function to configure the module by visiting module’s IP address or mDNS address on the web browser. However, you would have to make sure the module has connected to WLAN first.

Operation procedure:

1. Set the STATUS pin to high, and power on the Wi-Fi module.
2. Wait the Wi-Fi connection to be established (LED2 will be turned on). If failed, you should restore the module to factory settings according to chapter 2.5, and connect to network: “MXCHIP_XXXXXX” established by Wi-Fi module.
3. Open the web browser on PC, type in any web address at the browser’s address field. You will see the following configuration pages in the browser, input user name and password “admin/admin”.



Loading

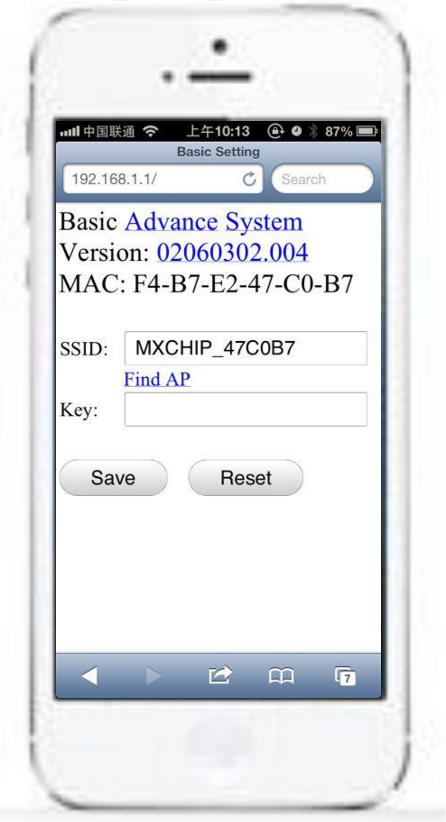
192.168.1.1/

Authentication Required

192.168.1.1

admin

●●●●●



Basic Setting

192.168.1.1/

Basic Advance System

Version: 02060302.004

MAC: F4-B7-E2-47-C0-B7

SSID: MXCHIP_47C0B7

Find AP

Key:

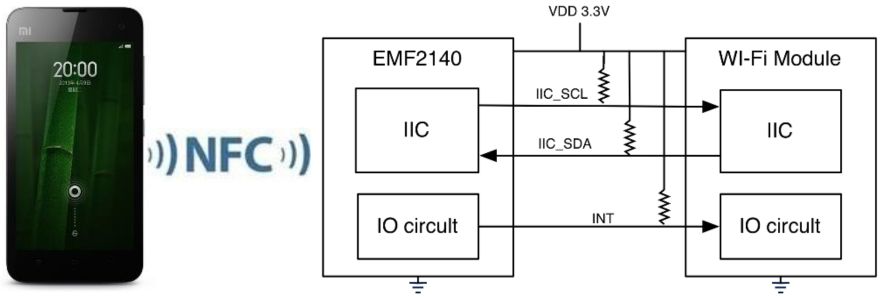
5.2 EMSP Commands Method

Refer chapter 4.1 for details.

5.3 Near-Field-Communication Method

EMF2104 NFC Tag can be read or write by any smart phone has NFC function. EMW module connects to EMW2104 by IIC interface and read back any information written by smartphone.

Hardware connection:



Operation procedure:

1. Download and install NFC configuration demo: Nfc-XPG-Demo, at http://www.mxchip.com/uploadfiles/soft/EMW/NfcXPGDemo.apk.zip .

2. Open the software and type in the Wi-Fi SSID and security Key. Then press the "Write" button.

3. Put your smart phone close to the NFC tag, and wait the writing procedure completed. The module will reboot with the new settings in a few seconds.

5.4 Wi-Fi Protected Setup (WPS)

Wi-Fi Protected Setup (WPS; originally Wi-Fi Simple Configuration) is a computing standard that attempts to allow easy establishment of a secure wireless home network.

The standard emphasizes usability and security, and allows up to four usage modes aimed at a home network user adding a new device to the network:

1. PIN Method (Supported by firmware but not opened to user)

In which a personal identification number (PIN) has to be read from either a sticker or the display on the new wireless device. This PIN must then be entered at the "representant" of the network, usually the access point of the network. Alternately, a PIN on the Access Point may be entered into the new device. The PIN Method is the mandatory baseline mode; every Wi-Fi Protected Setup certified product must support it.

2. Push-Button-Method (Supported by firmware and WPS button is available on module)

In which the user simply has to push a button, either an actual or virtual one, on both the access point (or a registrar of the network) and the new wireless client device. Support of this mode is mandatory for access points and optional for connecting devices.

3. Near-Field-Communication Method (Not supported, but has a better solution described chapter 5.3)

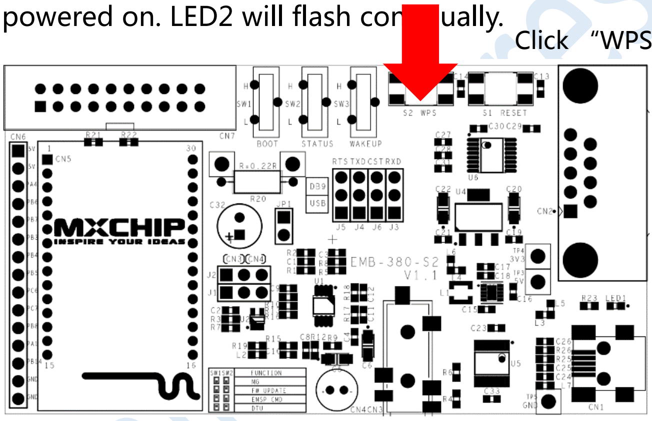
In which the user simply has to bring the new client close to the access point to allow a near field communication between the devices. NFC Forum compliant RFID tags can also be used. Support of this mode is optional.

4. USB Method (Not supported)

In which the user uses a USB flash drive to transfer data between the new client device and the access point of the network. Support of this mode is optional, but deprecated.

Operation procedure (Push button method):

1. Press down the "WPS/ Default" button or input character "2" from serial port when powered on. LED2 will flash continually.



Click "WPS/Default" button

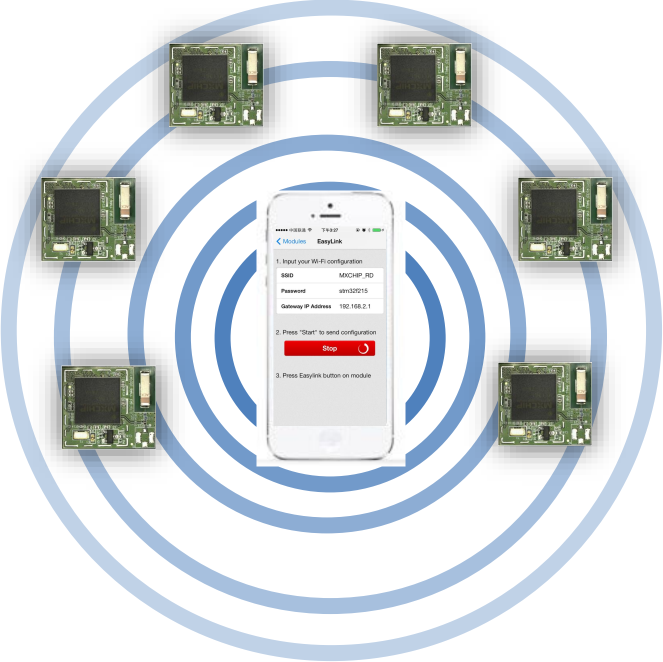
2. Press down the WPS button on the Access Point.

3. Wait the WPS negotiation complete, LED2 stop flashing and the module will reboot with the new settings. WPS negotiation timeout is 2 minutes. If WPS is failed, module would keep and use the old settings.

5.5 Easy Link Method

To create a great user experience, MXCHIP has created a one-step and one-time process to connect EMW316X modules to the home wireless network. This greatly stands apart from other methods require multiple steps to configure a device onto the network.

Easy link leverages the standard mechanisms present in Wi-Fi to configure an EMW316x's association information on the fly, regardless of whether user-interface is available. In this process a Wi-Fi enabled device such as a smartphone, tablet or a laptop is used to send the association information to the EMW316x.



●●●●● 中国联通 下午3:27

Modules EasyLink

1. Input your Wi-Fi configuration

SSID	MXCHIP_RD
Password	stm32f215
Gateway IP Address	192.168.2.1

2. Press "Start" to send configuration

Stop

3. Press Easylink button on module

This function only available on EMW316x modules.

Operation procedure:

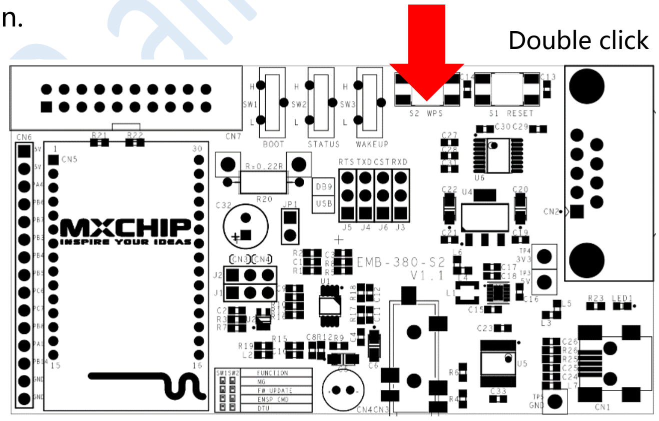
1. Download and install easylink demo app on android or iOS devices. APP and source code can be download at

http://mxchip.com/uploadfiles/soft/EMW/EasyLink_Android.zip

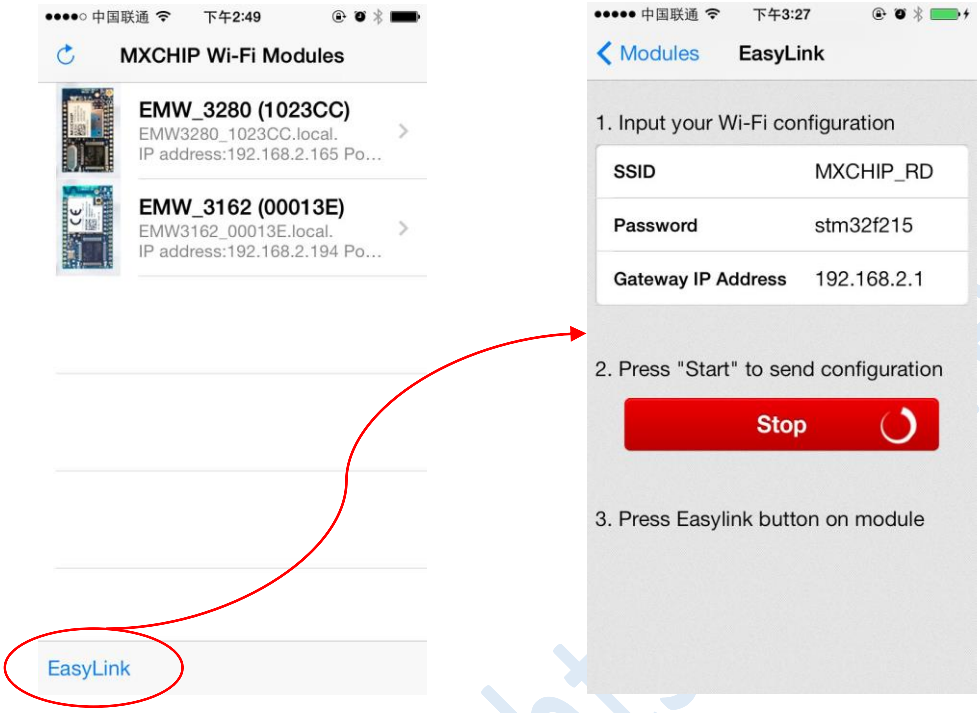
http://mxchip.com/uploadfiles/soft/EMW/EasyLink_iOS.zip

2. Double click the “WPS/ Default” button or press the “EasyLink/ Default” button.

Double click “WPS/Default” button



3. Run EasyLink APP on the devices, and input your AP’ s name and password and press “Start” button.



●●●●○ 中国联通 下午2:49

MXCHIP Wi-Fi Modules

EMW_3280 (1023CC)
EMW3280_1023CC.local.
IP address:192.168.2.165 Po...

EMW_3162 (00013E)
EMW3162_00013E.local.
IP address:192.168.2.194 Po...

EasyLink

●●●●● 中国联通 下午3:27

Modules EasyLink

1. Input your Wi-Fi configuration

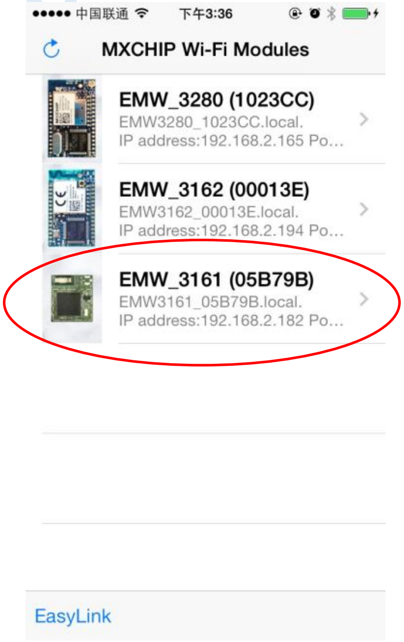
SSID	MXCHIP_RD
Password	stm32f215
Gateway IP Address	192.168.2.1

2. Press "Start" to send configuration

3. Press Easylink button on module

4. Wait the EasyLink negotiation complete, LED2 stop flashing and the module will reboot with the new settings. EasyLink negotiation timeout is 2 minutes. If EasyLink was failed, module would keep and use the old settings.

5. Now you can find the new module in local network.



●●●●● 中国联通 下午3:36

MXCHIP Wi-Fi Modules

EMW_3280 (1023CC)
EMW3280_1023CC.local.
IP address:192.168.2.165 Po...

EMW_3162 (00013E)
EMW3162_00013E.local.
IP address:192.168.2.194 Po...

EMW_3161 (05B79B)
EMW3161_05B79B.local.
IP address:192.168.2.182 Po...

EasyLink

6 Network Services

6.1 Use mDNS (Bonjour) Service to find module in local network

The multicast Domain Name System (mDNS) is a zero configuration host name resolution service. It uses essentially the same programming interfaces, packet formats and operating semantics as the unicast Domain Name System (DNS) to resolve host names to IP addresses within small networks that do not include a local name server, but can also be used in conjunction with such servers.

The mDNS protocol is published as RFC 6762, uses IP multicast User Datagram Protocol (UDP) packets, and is implemented by the Apple Bonjour and Linux nss-mdns services.

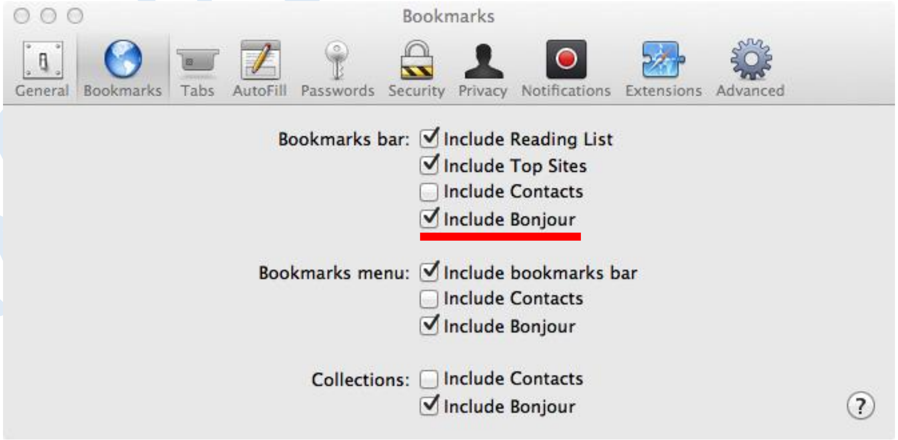
Bonjour is Apple's implementation of Zero configuration networking (Zeroconf), a group of technologies that includes service discovery, address assignment, and hostname resolution. Bonjour locates devices such as printers, other computers, and the services that those devices offer on a local network using multicast Domain Name System (mDNS) service records.

Find more information about Bonjour at: http://www.apple.com/support/bonjour/ .

When module is connected to the network, any device on the network can use bonjour service to locate module’ s IP address and available service on the module.

6.1.1 MAC/iOS Example:

Select “Include Bonjour” under “Bookmarks” tag in the safari’ s preference

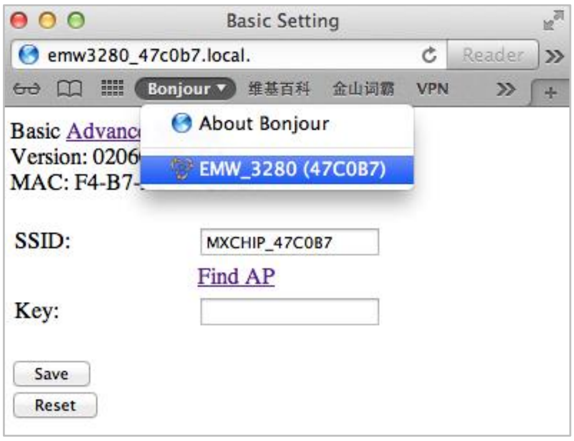


Bookmarks bar: ☑ Include Reading List
☑ Include Top Sites
☐ Include Contacts
☑ Include Bonjour

Bookmarks menu: ☑ Include bookmarks bar
☐ Include Contacts
☑ Include Bonjour

Collections: ☐ Include Contacts
☑ Include Bonjour

Module will be shown in the bookmarks bar of Safari.



Basic Setting

emw3280_47c0b7.local.

Bonjour ▾ 维基百科 金山词霸 VPN

About Bonjour

EMW_3280 (47C0B7)

Basic Advanc
Version: 0206
MAC: F4-B7-

SSID: MXCHIP_47C0B7

Find AP

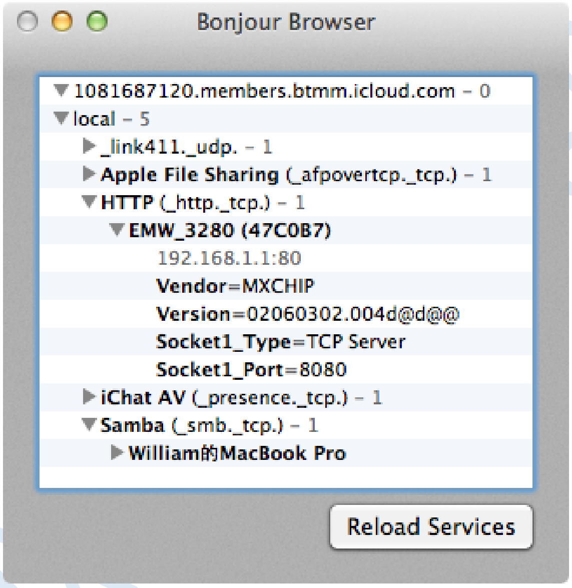
Key:

Save

Reset

Also, if you have install software: "Bonjour Browser", you may also find module and its information about IP address, firmware version, accessible port…:

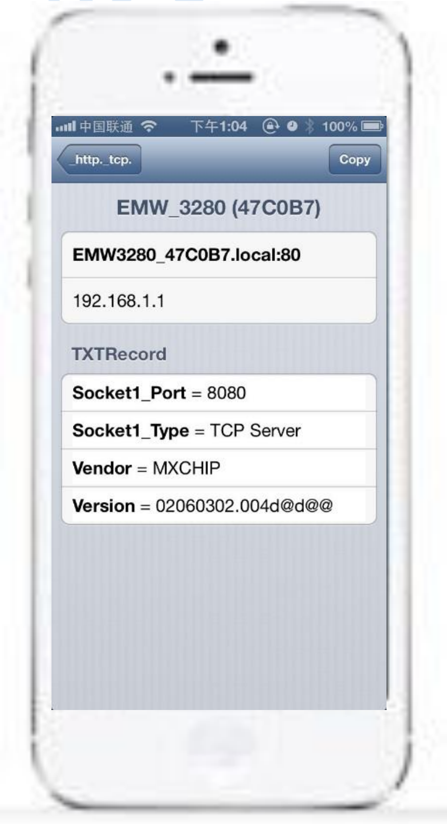
Download "Bonjour Browser" at http://www.tildesoft.com, or download an iOS version at https://itunes.apple.com/cn/app/discovery-bonjour-browser/id305441017?l=en&mt=8.



Bonjour Browser

```
▼ 1081687120.members.btmm.icloud.com – 0
▼ local – 5
   ▶ _link411._udp. – 1
   ▶ Apple File Sharing (_afpovertcp._tcp.) – 1
   ▼ HTTP (_http._tcp.) – 1
      ▼ EMW_3280 (47C0B7)
           192.168.1.1:80
           Vendor=MXCHIP
           Version=02060302.004d@d@@
           Socket1_Type=TCP Server
           Socket1_Port=8080
   ▶ iChat AV (_presence._tcp.) – 1
   ▼ Samba (_smb._tcp.) – 1
      ▶ William的MacBook Pro
```

Reload Services



中国联通 下午1:04 100%

http._tcp. Copy

EMW_3280 (47C0B7)

EMW3280_47C0B7.local:80

192.168.1.1

TXTRecord

Socket1_Port = 8080

Socket1_Type = TCP Server

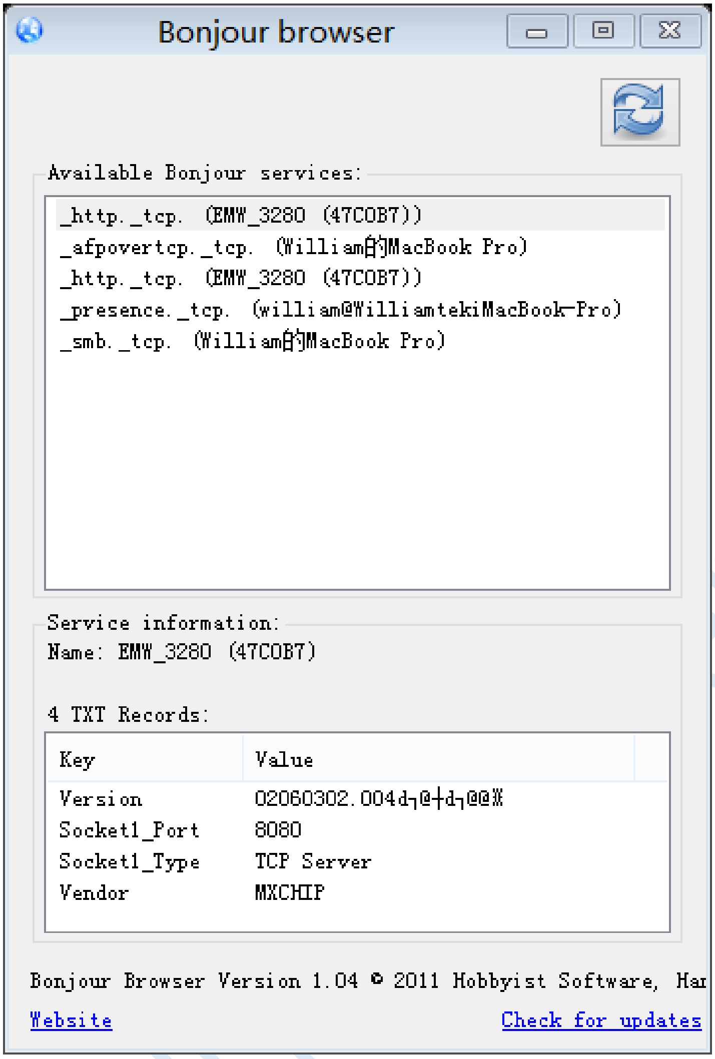
Vendor = MXCHIP

Version = 02060302.004d@d@@

6.1.2 Windows Example:

First, you should install bonjour service, which can be download at address: http://www.mxchip.com/uploadfiles/soft/EMW/BonjourSetup.zip.

Second, run "Bonjour browser for windows" which can be download at http://hobbyistsoftware.com/bonjourBrowser. Find module's information in bonjour service list.



Available Bonjour services:

```
_http._tcp. (EMW_3280 (47C0B7))
_afpovertcp._tcp. (William的MacBook Pro)
_http._tcp. (EMW_3280 (47C0B7))
_presence._tcp. (william@WilliamtekiMacBook-Pro)
_smb._tcp. (William的MacBook Pro)
```

Service information:

Name: EMW_3280 (47C0B7)

4 TXT Records:

Key	Value
Version	02060302.004d┐@┼d┐@@※
Socket1_Port	8080
Socket1_Type	TCP Server
Vendor	MXCHIP

Bonjour Browser Version 1.04 © 2011 Hobbyist Software, Har

6.1.3 Android example:

Download bonjour browser software to get the same function.

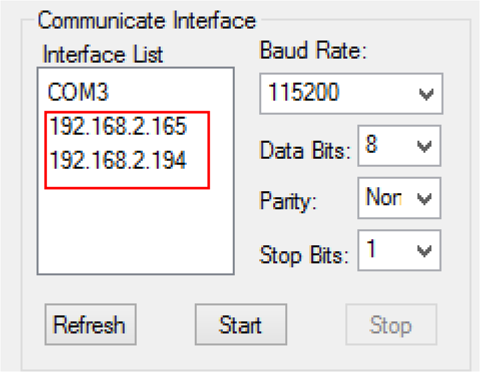
https://play.google.com/store/apps/details?id=com.grokkt.android.bonjour&hl=us_EN.

6.2 Use UDP broadcast to find modules in local network

EMSP commands can be delivered by network using UDP protocol on port 8089.

Firmware listen UDP packages on port 8089. So if one device needs to find available EMW modules, just send an EMSP command on port 8089 use UDP broadcast method (Target address: 255.255.255.255), and any module receive this EMSP command will send the return data to its sender.

The latest version EMW Tool Box use this function to locate the available modules in local network:



Communicate Interface

Interface List

COM3
192.168.2.165
192.168.2.194

Baud Rate: 115200

Data Bits: 8

Parity: Nor

Stop Bits: 1

6.3 DHCP Server

Firmware will automatically enable the DHCP server function in soft AP mode. This function can provide IP address for other devices connected to the module. The IP address range assigned to other devices is associated with the module’s own IP address.

Starting address: module’s IP address + 10

Ending address: module’s IP address +50

For example, if the module’s IP address is 192.168.1.1. The IP address range for other devices is 192.168.1.11-192.168.1.51

Note: When the module enables DHCP server function, it automatically disables DHCP client function. Therefore, in Soft AP mode, module only uses a static IP address.

7 Sales Information

If you need to buy this product, please call MXCHIP during the working hours. (Monday ~ Friday A.M.9:00~12:00; P.M. 1:00~6:00)

Telephone: +86-21-52655026 / 52655025

Address: Room 811, Tongpu Building, No.1220 Tongpu Road, Shanghai

Post Code: 200333

Email: sales@mxchip.com

8 Technical Support

If you need to get the latest information on this product or our other product information, you can visit: http://www.mxchip.com/

If you need to get technical support, please call us during the working hours:

ST ARM technical support

+86 (021)52655026-822 Email: support@mxchip.com

Wireless network technical support

+86 (021)52655026-812 Email: support@mxchip.com

Development tools technical support

+86 (021)52655026-822 Email: support@mxchip.com