MXCHIP
INSPIRE YOUR IDEAS

Datasheet

EMB-380-C/EMW-380-I1/EMW-380-I3

工业级多功能串口/Wi-Fi模块

Rev:01040154 Date：2010/6/15 产品数据手册

概述

EMB-380-C/EMW-380-I1/EMW-380-I3 是上海庆科信息技术有限公司开发的三款工业级多功能串口转 Wi-Fi 模块,它内部集成了 TCP/IP 协议栈和 Wi-Fi 通讯模块驱动，用户利用它可以轻松实现串口设备的无线网络功能，节省开发时间，使产品更快地投入市场，增强竞争力。

该产品用于串口与 Wi-Fi 无线网络之间的数据传输,可方便的为串口设备增加无线网络接口。可用于串口设备与 PC 机之间,或者多个串口设备之间的远程通信。


EMB-380-C


EMW-380-I1


EMW-380-I3

产品应用

- 楼宇/门禁/保安控制系统
- 医疗/保健自动化系统
- 金融证券交易系统
- 工业自动化系统
- 销售点系统(POS)
- 信息家电
- 汽车电子

产品特性

- 实现 UART 的 TTL 电平接口与 Wi-Fi 接口双向透明转换，UART 最大波特率为 921600；
- 易于使用的 11 个 EMSP 命令集，实现对模块的各种控制和参数配置；
- 支持通过 HTML 网页进行配置；
- 支持 IEEE802.11b/g 两种无线传输协议；
- 两种无线连接方式：AP 模式和 Ad-Hoc 模式；
- 可靠数据传输模式：TCP 服务器模式和 TCP 客户端模式， 保证数据在网络中可靠传输；
- 简单数据传输模式：UDP 数据传输模式；
- 支持 DHCP；
- 支持数据在 Internet 上的传输；
- 支持无线 WEP 加密技术；
- 通过休眠和唤醒，可以大幅降低功耗；
- 支持 CTS/RTS 硬件流控制

订购信息

型号	备注
EMB-380-C	UART/Wi-Fi module
EMW-380-I1	
EMW-380-I3	

上海庆科信息技术有限公司

无线设备开发部

----------典型应用模型----------

通过 AP 组成星形无线拓扑，多个嵌入式设备和 PC 或者智能手机交换数据

、

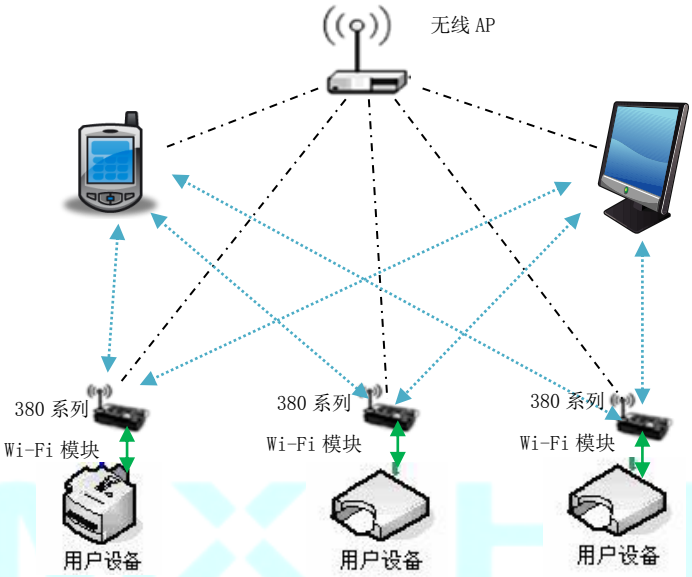


图 A 星形无线拓扑图

通过 Ad-Hoc 模式进行设备之间直接互联，实现嵌入式设备和 PC 或者智能手机的直接数据交换

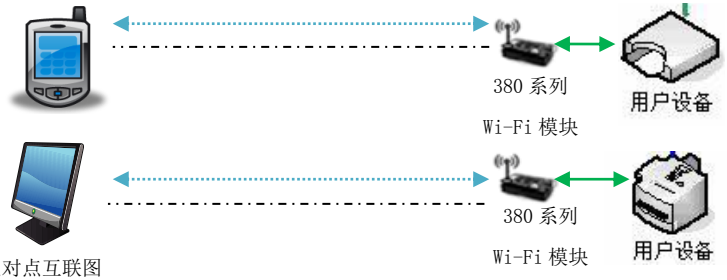


图 B 点对点互联图

图示说明：

网络数据虚拟流向

无线信号链路

UART 电缆链接

目录

1 文档约定

为方便用户阅读该使用说明书，特对以下常用名词作如下约定：

模块：指本文介绍的 EMB-380-C/EMW-380-I1/EMW-380-I3 工业级多功能串口转 Wi-Fi 模块。

嵌入式设备：指与模块相连，需要通过模块与网络上的其他设备进行通讯的嵌入式设备。

网络设备：指与模块通过网络相连，通过模块与嵌入式设备进行网络通讯的各种设备。

2 功能简介

EMB-380-C/EMW-380-I1/EMW-380-I3 是三款多功能嵌入式 UART/WiFi 数据转换模块，它内部集成了 TCP/IP 协议栈和 Wi-Fi 驱动，用户利用它可以轻松实现嵌入式设备的无线网络功能，节省开发时间，使产品更快投入市场，增强竞争力。

该系列模块可以工作在-40℃～85℃的温度范围内。串口通信最高波特率为 921600，具有 TCP，UDP 两种数据传输模式，并且支持命令来配置参数，方便使用。

2.1 特点

- WLAN 标准：IEEE 802.11b/g，Wi-Fi 兼容；
- 单操作电压：3.3V；
- 功耗电流：TX/RX ON <240mA，待机电流<1mA；
- 工作频率：2.4G ISM 频段；
- 输出功率：典型 15dbm +/-1.5dbm；
- 接收灵敏度：典型 802.11b：-91dbm；802.11g：-85dbm；
- 支持连接速率 54Mbps、48Mbps、36 Mbps、24 Mbps、18 Mbps、12 Mbps、9 Mbps、6Mbps、11 Mbps、5.5 Mbps、2 Mbps、1 Mbps；
- 支持 AP Client 和 Ad-Hoc 组网方式；
- 支持 WEP40 和 WEP104 加密（64/128 bit）；
- 媒体访问协议（MAC）：CSMA/CA 带 ACK；
- 波特率 9600-921600；
- 最高传输速率 60kbytes/s（双向传输）90kbytes/s（发送或者接收）；
- 两种工作模式：命令控制模式和数据传输模式；
- 模块网络数据通讯链路模式可选择 TCP，UDP 传输模式；
- TCP 模式下，模块可以作为服务器或者客户端；
- 具有断线自动重连机制，保证数据传输链路稳定可靠；
- 模块处于服务器模式时，允许 1 个客户端的连接，多个客户端的连接模式，需要定制；
- 灵活的外部配置方式：HTML 页面方式和 PC 端配置软件（含源代码）；
- 在嵌入式端通过 UART 接口使用 EMSP 命令控制模块和实现数据传输，提供 API 函数源代码；

2.2 产品规范

2.2.1 电气参数

绝对最大参数：电压

在此范围之内，保证不会损坏模块，但不能保证正常工作。正常工作的电器参数请见静态参数章节。

标号	项目	最小	最大	单位
VDD - VSS	供电电压	- 0.3	4.0	V
VIN	数字引脚输入电压	VSS -0.3	VDD+0.3	V

绝对最大参数：电流

在此范围之内，保证不会损坏模块，但不能保证正常工作。正常工作的电器参数请见静态参数章节。

标号	项目	最大	单位
I_{VDD}	Total current into VDD power lines (source)	320	mA
I_{VSS}	Total current out of VSS ground lines (sink)	320	
I_{IO}	IO 输出最大输出电流	25	
	IO 输出最大输入电流	-25	

绝对最大参数：温度

标号	项目	最大	单位
TSTG	存储温度	- 65 to +150	° C
T_A	正常工作温度范围	-40 to +85	° C

绝对最大参数：电磁环境

Electrostatic discharge (ESD)

Symbol	Ratings	Conditions	Class	Maximum value	Unit
VESD(HBM)	Electrostatic discharge voltage (human body model)	TA = +25 ° C conforming to JESD22-A114	2	2000	V
VESD(CDM)	Electrostatic discharge voltage (charge device model)	TA = +25 ° C conforming to JESD22-C101	II	500	

Static latch-up

经测试，完全通过 EIA/JESD 78A IC 的 latch-up 标准

Symbol	Parameter	Conditions	Class
LU	Static latch-up class	TA = +105 ° C conforming to JESD78A	II level A

静态参数：电源输入

标号	类别	条件	规格				说明
			最小	典型	最大	单位	
VDD	模块电压		2.6	3.3	3.6	V	
IVDD	模块电流	VDD=3.3V	210	230	240	mA	正常工作
IVDD	模块电流	VDD=3.3V	0.8	1	3	mA	休眠状态

静态参数：数字引脚

引脚输出

标号	类别	项目	条件	规格		
				最小	最大	单位
VOL	串口及 IO 信号引脚	低电平输出电压	I_{IO} = +8 mA 2.7 V < VDD < 3.6 V		0.4	V
VOH		高电平输出电压		VDD-0.4		V
VOL		低电平输出电压	I_{IO} = +20 mA 2.7 V < VDD < 3.6 V		1.3	V
V_{OH}		高电平输出电压		VDD-1.3		V

引脚输入

标号	类别	项目	条件	规格		
				最小	最大	单位
VIL	串口及 IO 信号引脚	低电平输入电压	TTL 电平	-0.5	0.8	V
VIH		标准高电平输入		2	VDD+0.5	V
		允许 5V 输入		2	5.5	V
VIL		低电平输入电压	CMOS 电平	-0.5	0.35VDD	V
VIH		高电平输入电压		0.65VDD	VDD+0.5	V

/RESET 引脚电气参数

/RESET 引脚的输入驱动电路采用 CMOS 电路。模块内部含有阻容复位电路，可以保证在模块上电时准确复位。如果需要控制其复位，只需要通过外部控制信号直接和复位引脚相连就可以了。

此外，通过 EMSP 命令，软件也可以控制其复位。

标号	项目	条件	最小	典型	最大	单位
VIL(NRST)	/RESET 输入低电平		-0.5		0.8	V
VIH(NRST)	/RESET 输入高电平		2		VDD+0.5	

R_{PU}	上拉电阻	V_{IN} = VSS	7.5	8	8.3	kΩ
C_{PD}	复位充电电容			100	1000	pF

注意，EMW-380-I1 的 RESET 引脚特殊，默认是高电平复位。

无线网络电气参数

频率范围	2.4 GHz ISM radio band
无线通道	802.11b: USA, Canada and Taiwan - 11 Most European Countries – 13 France - 4, Japan - 14 802.11g: USA and Canada - 11 Most European Countries - 13
调制方式	DSSS, OFDM, DBPSK, DQPSK, CCK, 16-QAM, 64-QAM
输出功率	802.11b: typical 17dBm +/- 2dBm 802.11g: typical 15dBm +/- 2dBm
天线接口	Dual IPEX antenna port
接收灵敏度	802.11b: typical -86 +/- 3dBm at 11Mbps 802.11g: typical -71 +/- 3dBm at 54Mbps
MAC 协议	CSMA/CA with ACK
数据速率	802.11b: 1, 2, 5.5, 11Mbps 802.11g: 6, 9, 12, 18, 24, 36, 48, 54Mbps
安全性	WEP 64-bit and 128-bit encryption with H/W TKIP processing
无线网络互存	Cell phone(GSM/DCS/WCDMA/UMTS/3G) co-existence

2.2.2 机械尺寸

EMB-380-C 机械尺寸

用户如需安装 EMB-380-C，或者制作 EMB-380-C 底板（主板），可参考图 1.2 和图 1.3 所提供的外观机械尺寸（公制单位表示），图中规定了产品的长、宽、高，以及部分机械结构。

- 模块尺寸：32 x39 x6.52mm
- 引出插针：1.27 间距双排针
- 模块引脚排列（顶视图），见下图 1.2

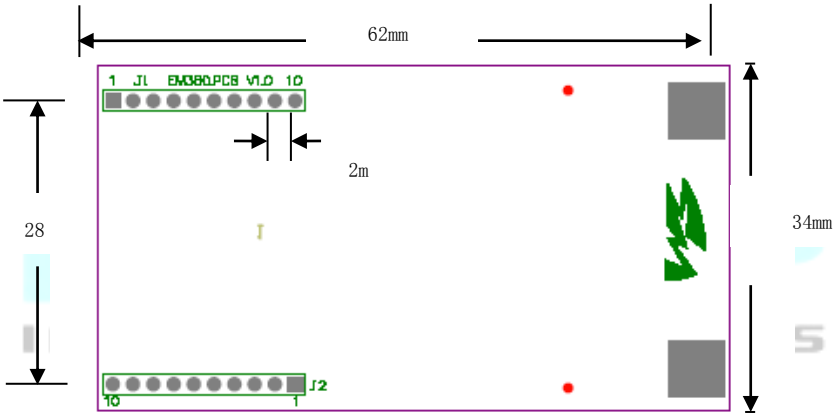


图 1.2 EMB-380-C 引脚排列顶视图

侧面视图，参考图 1.3

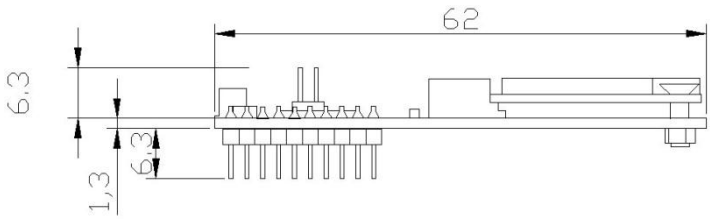


图 1.3EMB-380-C 侧面视图

EMW-380-I1 机械尺寸

用户如需安装 EMW-380-I1，或者制作 EMW-380-I1 底板（主板），可参考图 1.4 和图 1.5 所提供的外观机械尺寸（公制单位表示），图中规定了产品的长、宽、高，以及部分机械结构。

- 模块尺寸：32 x39 x6.52mm
- 引出插针：1.27 间距双排针
- 模块引脚排列（顶视图）

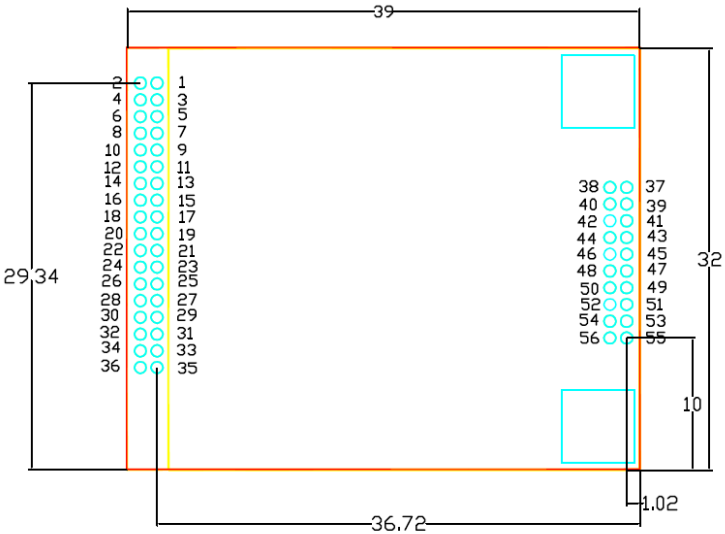


图 1.4 EMW-380-I1 引脚排列（顶视图）

侧面视图，参考图 1.5

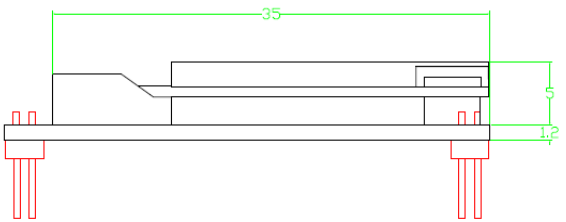


图 1.5 EMW-380-I1 侧面视图

EMW-380-I3 机械尺寸

用户如需安装 EMW-380-I3，或者制作 EMB-380-I3 底板（主板），可参考图 1.6 与图 1.7 所提供的外观机械尺寸（公制单位表示），图中规定了产品的长、宽、高，以及部分机械结构。

图中规定了排针与排针的距离是 35mm（1378 mil），另外排针的间距是 2.54mm（100mil）。

- 模块尺寸：40 x39.47 x6.52 mm
- 引出插针：一排 14-Pin（2.54mm 脚距），一排 13-Pin（2.54mm 脚距）排距 35mm（1378 mil）
- 模块引脚排列（顶视图），见图 1.6
- 注意：两排排针位置不在同一水平上，位置相差 2.0mm,具体见图 1.6

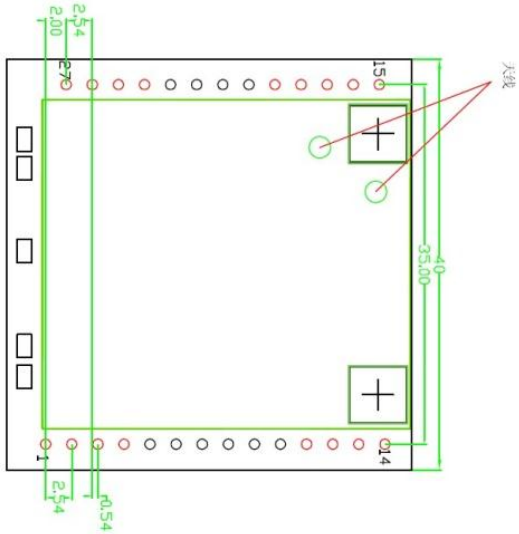


图 1.6 EMW-380-I3 引脚排列（顶视图）

侧面视图，参考图 1.7

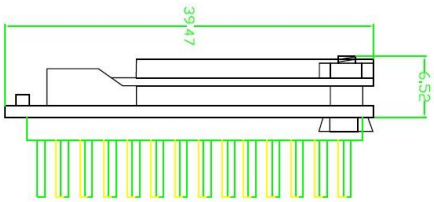


图 1.7 EMW-380-I3 侧面视图

3 硬件说明

3.1 天线接口和 LED 灯

EMB-380-C/EMW-380-I1/EMW-380-I3，天线需接 PCB 板上丝印 J2 接线柱处（三款模块 PCB 丝印标识），见下图 2.1（以 EMB-380-C 模块为例），功能表见下表 2.1.

图 2.1 模块丝印 J2、J3 标识图

表 2.1 天线接线柱功能表

No.	FUN.	No.	FUN.
J2	天线接线柱	J3	NG（暂不使用）

模块上有两个 LED 灯用来直观地指示模块当前的状态。用户也可以在配制模式下通过命令来获得模块当前的状态。LED 功能如下表 2.2.

表 2.2 EMB-380-C LED 指示灯状态

名称	颜色	含义	功能
D1	红色	WiFi 成功连接指示灯	闪烁：数据传输 常亮：已连接 常暗：断开
D2	绿色	系统初始化成功指示灯	常亮：模块初始化正常，正常工作 常暗：模块初始化失败，或处于休眠状态

3.2 模块外形及接口

3.2.1 EMB-380-C

模块的外形如 图 2.2 所示，从俯视图图 2.3 ，我们可以看出 EMB-380-C 模块有两排外引管脚，J1 和 J2。J1 的左边是 1 号引脚，向右依次增加。J2 则正好相反，右边是 1 号引脚，向左依次增加，引脚定义 见表 2.3.

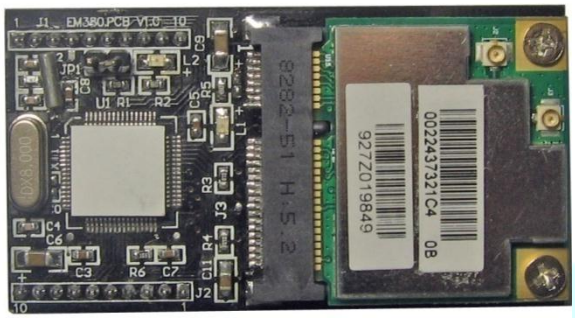


图 2.2 EMB-380-C 模块外形图

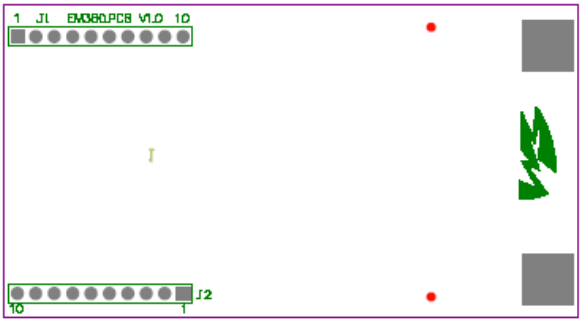


图 2.3 EMB-380-C 引脚外形图

3.2.2 EMW-380-I1

EMW-380-I1 有两排 1.27 间距的双排针。

模块的外形如图 2.4 所示，引脚排列如图 2.5.，引脚功能表见表 2.4

图 2.4 EMW-380-I1 外形

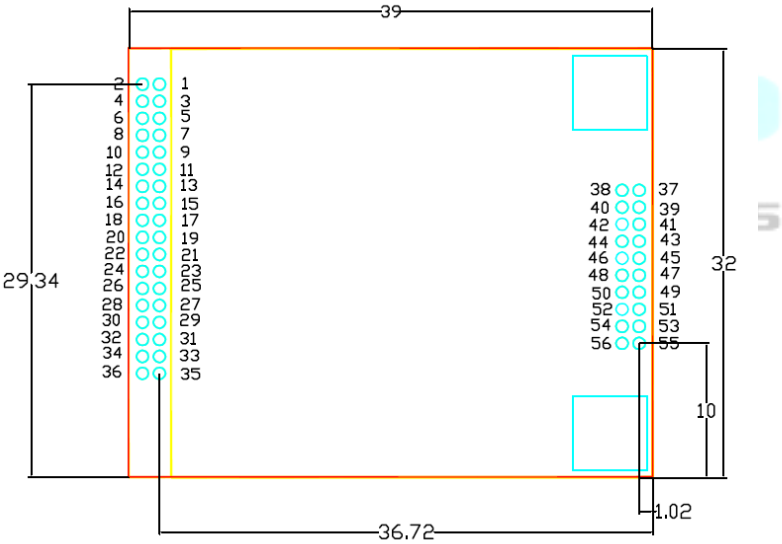


图 2.5 EMW-380-I1 引脚分布

3.2.3 EMW-380-I3

模块的外形如图 2.6 所示，从俯视图图 2.7 我们可以看出 EMB-380-I3 模块有两排外引管脚，右边是 1 号引脚，向上依次增加。左则正好相反，左上边是 15 号引脚，向下依次增加，引脚功能表见表 2.5.

图 2.6 EMW-380-I3 外形

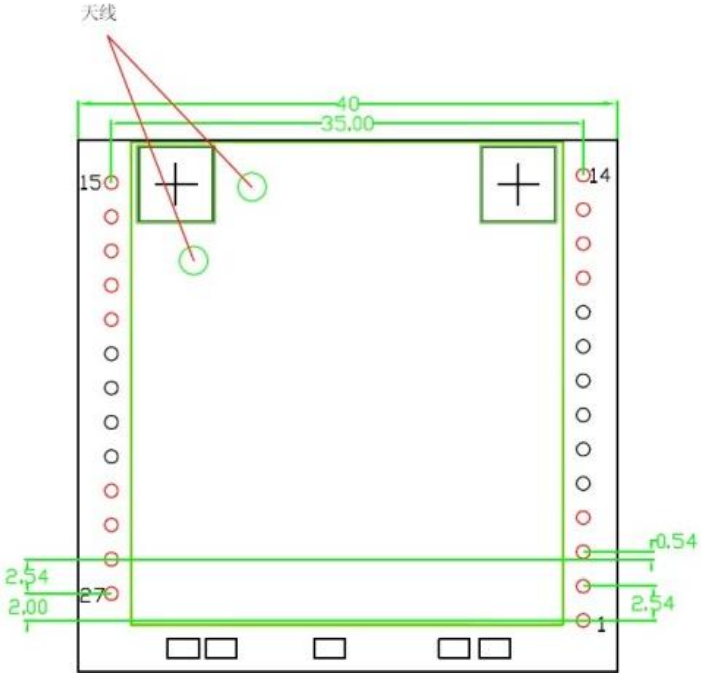


图 2.7 EMW-380-I3 引脚俯视图引脚定义

3.3 引脚定义

表 2.3 EMB-380-C 引脚功能表

J1 引脚	功能	FT	J2 引脚	功能	FT
1	Wakeup（IN）		1	UART_RTS(OUT)	
2	/RESET(IN)		2	UART_CTS(IN)	√
3	STATUS（IN）	√	3	UART_RXD(IN)	√
4	/INT(OUT)		4	UART_TXD(OUT)	
5	GND		5	GND	
6	NC		6	NC	
7	NC		7	NC	
8	NC		8	NC	
9	NC		9	NC	
10	NC		10	VDD	

表 2.4 EMW-380-I1 引脚功能表

No.	FUN.	FT	No.	FUN.	FT
1	UART_CTS(IN)	√	11	GND	
2	UART_RTS(OUT)		12	GND	
3	WAKE_UP（IN）		15	WI-FI LED（OUT）	
4	STATUS(IN)	√	17	RESET（IN）	
6	/INT（OUT）		19	GND	
7	UART_TXD(OUT)		20	GND	
8	UART_RXD(IN)	√	55	VDD	
9，10	VDD		56	GND	

表 2.5 EMW-380-I3 引脚功能表

引脚	功能	FT	引脚	功能	FT
1	NC		15	NC	
2	NC		16	NC	
3	/INT(OUT)		17	NC	
4	NC		18	NC	
5	VDD		19	Wakeup（IN）	
6	GND		20	STATUS(IN)	√
7	UART_RXD(IN)	√	21	NC	
8	UART_TXD(OUT)		22	WI-FI LED（OUT）	
9	NC		23	/RESET(IN)	
10	NC		24	NC	
11	NC		25	UART_RTS(OUT)	
12	NC		26	UART_CTS(IN)	√
13	NC		27	VDD	
14	GND				

引脚说明：

1. VDD 是电源输入引脚。
2. FT 脚表示该引脚允许 5V 电压输入。
3. NC 目前保留，用户无须连接。
4. 模块内部含有必要的上下拉电阻，因此模块外围电路无需额外的电阻来保证模块确定的工作状态。
5. RESET（IN）,低电平复位。（注意：EMW-380-I1 采用高电平复位）
6. STATUS 用来设置模块的工作状态。详细参阅第三章。
7. WAKE_UP 引脚如果产生下降沿使模块进入休眠状态，上升沿使模块进入正常工作状态，模块对该引脚有去噪功能，避免非正常信号的干扰。

8. UART 连接包括：UART_TXD，UART_RXD，UART_RTS 和 UART_CTS。
9. 基本应用中只需要连接电源和 UART_TXD、UART_RXD 以及 STATUS。
10. 建议连接 UART_RTS 和 UART_CTS 来使能 UART 的硬件流控制。这样，网络阻塞时，模块可以对嵌入式设备发送给 UART 接口的数据流量进行自动控制。
11. HOST 端可以通过查询 EMW-380 的/INT 引脚来确定初始化是否完成。当模块加电后，/INT 会保持为高，当初始化完成后，/INT 会被拉低，此后设备即可向 EMW-380 发送命令并进行相关操作了。在 EMW-380 中/INT 引脚没有其他功能。
12. WIFI_LED 引脚功能同模块上的 D1，用于指示模块的 Wi-Fi 连接状态和数据通讯状态，低电平有效。可以直接外接 LED 灯或者信号线，无需外接上拉电阻。

3.4 典型硬件连接

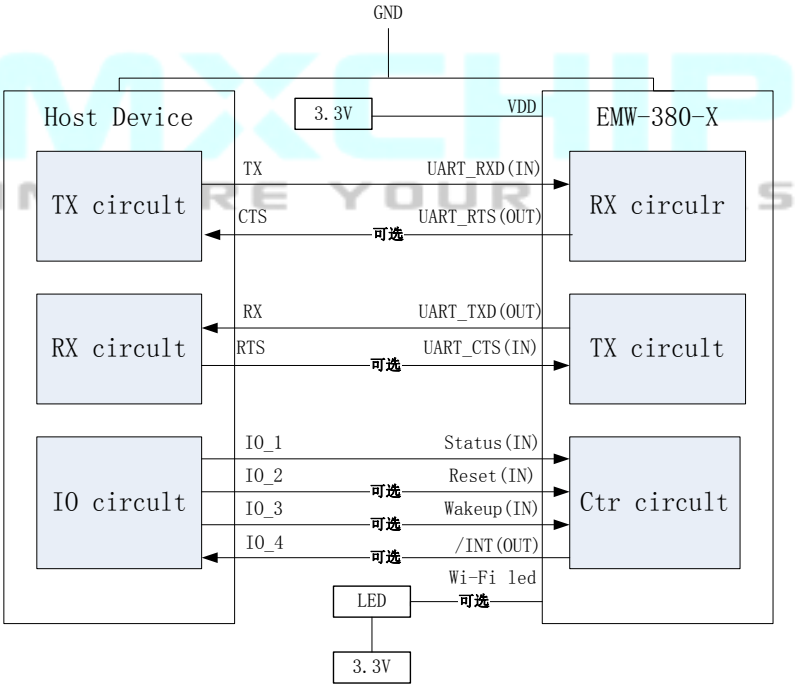


图 2.8 典型 UART 连接图

4 工作状态的控制

EMW-380 模块，具有相同的工作状态控制。

两类工作状态：命令控制模式和数据通讯模式。具体的功能如下：

命令控制模式：

在这种状态下用户可以使用 EMSP 通讯协议来对模块进行操作，配制，简单测试收发数据。模块的一切工作都处于 EMSP 命令的控制之下。命令控制模式的功能请见第 5 章。

数据通讯模式：

模块按照配制好的参数自动进行网络连接并进行数据传输，其工作模式可以分为：TCP 服务器模式，TCP 客户端模式和 UDP 模式。数据通讯模式下的功能请见第 6 章。

4.1 STATUS 引脚的功能

Status 引脚下拉至低电平时，模块进入命令控制模式。

Status 引脚上拉至高电平时，模块进入数据通讯模式。

4.2 模块上电后的工作状态

当模块电源接通时，会进行必要的初始化操作，这个过程需要延迟一段时间，此后模块开始响应嵌入式设备的请求。这个延迟时间是必须的，是为了使模块能充分完成初始化。

HOST 端有两种方法来检测模块的初始化是否正常完成：

1. 可以通过查询模块的/INT 引脚来确定初始化是否完成。当模块加电后，/INT 会保持为高，当初始化完成后，/INT 会被拉低，此后 HOST 即可向模块发送命令并进行相关操作了。
2. 如果 Status 引脚为低电平，也可以通过持续地向模块发送命令。如正常返回命令，则模块已正常启动。

模块上电后自动检测 Status 引脚的状态。

Status 引脚处于低电平：

模块进入命令控制模式。

初始化完成后，嵌入式设备可以通过 EMSP 命令来控制 Wi-Fi 模块。

Status 引脚处于高电平：

初始化完成后，模块自动按照预先设定好的参数尝试连接 Wi-Fi 网络，如果成功连接，模块将进入原先设定好的工作模式。如果连接失败，模块将不停尝试连接。

如果无线网络信号和参数配置都正确，上电后 1.5 秒钟后，模块可以正确进入正常工作状态。

4.3 模块工作中的状态转换

数据通讯模式→命令控制模式

1. 将 Status 引脚拉低
2. UART 输入任意 EMSP 串行命令，直至有数据正常返回

命令控制模式→数据通讯模式

1. 使用 EMSP_CMD_START 命令启动网络连接（如果已经连接，可以省略）
2. 通过 EMSP_CMD_GET_STATUS 命令查看 Wi-Fi 连接和 TCP 连接状态
3. 连接正常后，将 Status 引脚拉高
4. 模块进入数据通讯模式

5 命令控制模式

模块的命令控制模式主要用于对模块进行操控和配置。

将模块的 Status 引脚拉低，即可进入命令控制模式。如果您使用 EMW-380-S 系列测试底板，请用跳线将 Status 引脚接至低电平。

在命令控制模式下，嵌入式设备通过预定义的 EMSP 指令集来控制模块。具体来讲有两种方法来进行控制：

1、 使用提供的“EM380-配置工具”来进行配置。用户在了解 EMSP 命令集以后，也可以自行开发相应的 PC 配置软件；

2、 通过在应用中连接到模块的嵌入式设备上使用 EMSP 命令来控制模块。我们在此基础上封装了 EMSP 接口函数库，我们也在 MDV-STM32-107 评估板上提供了如何使用函数库的相应的示例程序；

此外，如果模块与 PC 的网络可以连通，那么模块也能够登录到模块的 HTML 页面，通过 IE 浏览器对模块的参数进行配置。这种配置方式通常在数据通讯模式下对模块进行配置。但是我们将这种配置方法移到这一章节介绍，方便用户完整地了解模块的所有配置方式。

5.1 EM380-配置工具

“EM380-配置工具”是一个专用于控制处于命令控制模式下的 EMW-380 模块的一套 PC 软件。

基本流程：

1、 将模块置于配套的测试底座上（如 EMW-380-S），将底座上的 Status 跳线连接至 GND。

2、 通过一根串口直连线（TX，RX 信号线交叉）连接测试底座和 PC 机。

3、 接通测试底板的电源，如果正常，可以看到模块上的绿色 LED 灯亮。

4、 在 PC 上打开“EM380-配置工具”软件，按照和模块匹配的串口参数打开串口（点击“打开串口”按钮），如果不清楚模块的串口配置，可以使用软件左上角的“恢复默认配置”功能，按照软件的提示进行操作，将模块恢复到默认参数。然后以 115200，8/n/1 的参数，将 PC 的串口打开。

5、 按下“获取设备型号按钮”，可以看到模块当前的功能型号显示在“连接设备型号” 中，接下来就可以使用软件提供的各项功能，对模块进行配置和操作了。

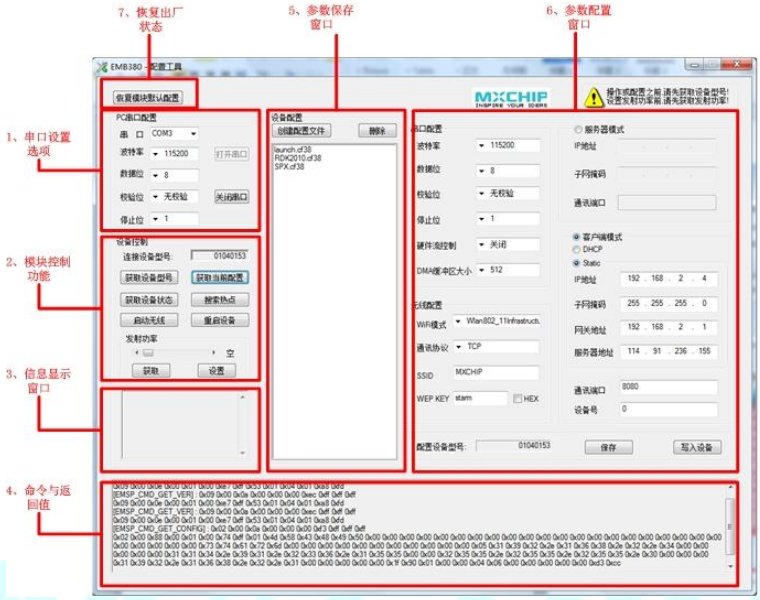


图 4.1 EMW380-配置工具界面

1. 串口设置选项

   用于选择您的 PC 和模块相连的串口的参数，这些参数应与模块的配置参数一致，否则将不能进行通讯，如果修改了模块的参数，则应该先关闭串口，将参数修改为与模块一致后，再打开。

2. 模块控制功能

3. 信息显示窗口

4. 命令和返回值

   配置软件实际与模块交互的命令与返回值的 16 进制编码，用户可以以此熟悉 EMSP 命令的编码和交互方式。

5. 参数保存窗口

   用户可以将常用的配置参数进行归档，并且保存。保存之后，只需要双击文件名就可以读出保存的所有配置参数。

6. 模块参数列表

   通过“获取当前配置”按钮可以将这些参数读出，“写入设备”按钮可以把这些参数写入到模块中。

7. 恢复出厂状态

5.1.1 模块配置参数详述

无线网络部分

WIFI 模式

Wi-Fi 连接模式，有以下选项

Wlan802_11IBSS：即 Ad-Hoc 模式，通常用于模块和网络设备的直接连接；

Wlan802_11Infrastructure：用于模块通过 AP 和其他网络设备组成星形结构的无线网络；

TIPS：

根据网络是否存在 AP，无线局域网有两种结构：INFRASTRUCTURE 类型和 AD HOC 类型。

INFRASTRUCTURE 型网络由若干个 STA 和一个 AP 组成，STA 之间通信通过 AP 转发，AP 连接到有线骨干网上。AD HOC 型由多个 STA 组成，相互之间直接通信，不存在 AP。这种方式不需要访问有线网络中的资源，只在无线通信设备之间相互通信。

无线网络的拓扑结构主要有：无中心的分布对等方式（AD HOC）、有中心的集中控制方式（INFRASTRUCTURE）、以及上述方式的混合方式。在分布式对等方式下，无线网中的任意两站之间可以直接通信，无需设中心转接站。这时， MAC 控制功能由各站分布管理。 这种方式同 IEEE802.3 局域网类似，网上的站共享一个无线通道，通常使用 CSMA/CA 作为 MAC 协议。这种方式的特点是结构简单易维护。由于采用分布控制方式，某一站的故障不会影响整个网络的运行。

在集中控制方式情况下，无线网中设置一个中心控制站，主要完成 MAC 控制及信道的分配等功能。网中的其它站在该中心的协调下与其它各站通信。由于对信道资源分配、 MAC 控制采用集中控制的方式，这样使信道利用率大大提高， 网络的吞吐性能优于分布式对等方式。当然引入中心站也使得无线网的结构复杂。但目前的无线产品都把这些复杂的功能作成透明的， 无需用户干预。

SSID

加入 WiFi 网络的网络名称，长度应小于 32 个字节。

WEP KEY

加入 WiFi 网络的 WEP 密钥，WEP 密钥的长度可以分为 5 字节或 13 字节，其他长度不接受，可以输入 ASICII 值或者 16 进制数，这个可以在 HEX 选项中设置。

如 5 字节的 ASICII 值：EM380，或者 16 进制数：454D333830 产生的密钥是一样的

通讯协议

有两个选项，分别是 TCP 和 UDP。

模块处于 TCP 模式时，通讯双方需按照主从模式建立数据传输通道之后，才能进行数据交换。

模块处于 UDP 模式时，通讯双方只需要向对方地址发送数据即可，不需要建立传输链路。

TIPS：

- *面向连接的 TCP*

“面向连接”就是在正式通信前必须要与对方建立起连接。比如你给别人打电话，必须等线路接通了、对方拿起话筒才能相互通话。

TCP（TRANSMISSION CONTROL PROTOCOL，传输控制协议）是基于连接的协议，也就是说，在正式收发数据前，必须和对方建立可靠的连接。一个 TCP 连接必须要经过三次“对话”才能建立起来，其中的过程非常复杂，我们这里只做简单、形象的介绍，你只要做到能够理解这个过程即可。我们来看看 TCP 三次对话的简单过程：主机 A 向主机 B 发出连接请求数据包：“我想给你发数据，可以吗？”，这是第一次对话；主机 B 向主机 A 发送同意连接和要求同步（同步就是两台主机一个在发送，一个在接收，协调工作）的数据包：“可以，你什么时候发？”，这是第二次对话；主机 A 再发出一个数据包确认主机 B 的要求同步：“我现在就发，你接着吧！”，这是第三次对话。三次“对话”的目的是使数据包的发送和接收同步，经过三次“对话”之后，主机 A 才向主机 B 正式发送数据。

TCP 协议能为应用程序提供可靠的通信连接，使一台计算机发出的字节流无差错地发往网络上的其他计算机，对可靠性要求高的数据通信系统往往使用 TCP 协议传输数据。

- 面向非连接的 UDP 协议

“面向非连接”就是在正式通信前不必与对方先建立连接，不管对方状态就直接发送。这与现在风行的手机短信非常相似：你在发短信的时候，只需要输入对方手机号就 OK 了。

UDP（User Data Protocol，用户数据报协议）是与 TCP 相对应的协议。它是面向非连接的协议，它不与对方建立连接，而是直接就把数据包发送过去！

UDP 适用于一次只传送少量数据、对可靠性要求不高的应用环境。比如，我们经常使用“ping”命令来测试两台主机之间 TCP/IP 通信是否正常，其实“ping”命令的原理就是向对方主机发送 UDP 数据包，然后对方主机确认收到数据包，如果数据包是否到达的消息及时反馈回来，那么网络就是通的。例如，在默认状态下，一次“ping”操作发送 4 个数据包。大家可以看到，发送的数据包数量是 4 包，收到的也是

4 包（因为对方主机收到后会发回一个确认收到的数据包）。这充分说明了 UDP 协议是面向非连接的协议，没有建立连接的过程。正因为 UDP 协议没有连接的过程，所以它的通信效果高；但也正因为如此，它的可靠性不如 TCP 协议高。QQ 就使用 UDP 发消息，因此有时会出现收不到消息的情况。

表 4.2 tcp 协议和 udp 协议差别

	TCP	UDP
是否连接	面向连接	面向非连接
传输可靠性	可靠的	不可靠的
应用场合	传输大量的数据	少量数据
速度	慢	快

TCP/IP 协议栈部分

IP 地址

模块的本地 IP 地址。如果使用 DHCP，模块的 IP 地址可以自动获得。

当模块处于 TCP 服务器模式下，其他客户端网络设备就需要访问该地址来建立 TCP 连接。因此在 TCP 服务器模式下时，不应该使用 DHCP 功能，以防止服务器地址的变化。在配置软件中，TCP 服务器是不允许使用 DHCP 功能的

服务器地址

远端服务器的 IP 地址。当模块的工作方式是 TCP 客户端时，模块将自动连接符合这个 IP 地址的网络设备，并进行数据传输。

子网掩码

注意和需要通讯的网络设备的掩码保持一致。

网关地址

模块需要访问 Internet 上的 IP 地址时，数据包需要通过局域网的网关发送到远端 IP，因此必须正确设置网关地址，一般来讲网关地址就是路由器的 IP 地址，或者是 AP 的 IP 地址。如果只需实现局域网内部的访问，则该地址可以任意。

如果使用 DHCP，网关地址可以自动获得。

通讯端口

TCP，UDP 通讯的网络端口，只有使用同一个端口的两个网络设备才能够相互通讯。

很多通讯端口是网络通用的，如 http 端口：80，telnet 端口：21 等。在设置的时候注意不要使用这些端口。此外，和 PC 通讯时，要设置 PC 的防火墙，以允许这些端口的通讯。

服务器模式/客户端模式

表示的是模块在 TCP 通讯过程中扮演的角色。

模块处于 TCP 模式并处于客户端模式时，会自动地和“服务器地址”匹配的网络设备连接。

模块处于 TCP 模式并处于服务器模式时，模块不会主动地和其他网络设备相连，其他网络设备可以主动地连接到模块。

模块处于 UDP 模式时，虽然不存在服务器/客戶端的概念。但是配置軟件上应设置成客户端模式，这样才能设置与模块通讯的远程计算机的地址。

DHCP/STATIC

打开或者关闭模块的 DHCP 功能，DHCP 功能可以使模块通过网络中的 DHCP 服务器自动获得 IP 地址，子网掩码和网关地址。

串行接口部分

波特率

用于设置模块和嵌入式设备 UART 通讯时的波特率。

校验位

用于设置模块和嵌入式设备 UART 通讯时的校验位

数据位

用于设置模块和嵌入式设备 UART 通讯时的数据位长度

停止位

用于设置模块和嵌入式设备 UART 通讯时的停止位长度

DMA 缓冲区大小

DMA 缓冲区仅用于嵌入式设备通过模块发送数据，模块对嵌入式设备发出的数据进行的缓冲。

用于设置模块接收嵌入式设备传入的 UART 数据时的缓冲。模块在达到缓冲半满时，会将缓冲中的数据进行网络数据包的封装，并通过网络发送到特定的目的地。如果在**特定时间内**没有达到半满，也会强制触发网络传输，将缓冲中的数据打包传走。

通过 DMA buffer 缓冲，允许用户在实时性和传输性能这两个特性之间进行调节。DMA buffer 越大，缓冲的数据越多，数据传输性能就越好，大数据量的传输速度就越快，但是如果数据量较少，在**特定时间**内无法达到缓冲的半满，就会影响网络数据的实时性：延时**特定时间**才能触发网络传输。如果需要进行小数据量，高实时性的数据传输，应该减少 DMA buffer 的大小。最极限的情况，使用 2 字节的缓冲，其实就是完全实时的数据传输。但是在这种情况下，每一个字节的串口数据都会被封装成一个独立的 TCP/IP 数据包

DMA buffer 有以下参数可选：

2bytes，16bytes，32bytes，64bytes， 128bytes，256bytes，512bytes。

特定时间：

- **buffer size 512 字节：未满 256 字节延时 500ms；满 256 字节立即发送**

- buffer size 256 字节：未满 128 字节延时 250ms；满 128 字节立即发送
- buffer size 128 字节：未满 64 字节延时 125ms；满 64 字节立即发送
- buffer size 64 字节：未满 32 字节延时 60ms；满 32 字节立即发送
- buffer size 32 字节：未满 16 字节延时 30ms；满 16 字节立即发送
- buffer size 16 字节：未满 8 字节延时 15ms；满 8 字节立即发送
- buffer size 2 字节：无延时

硬件流控制

该参数用于控制是否使用 UART 的硬件流控制，硬件流控制功能使得在高速串口数据收发时，对串口流量进行自动控制（仅在 TCP 模式下可以达到应有的效果）。

由于使用无线网络进行传输，很容易受到不稳定的无线信号的干扰，可能在短时间内阻塞网络通讯。模块可以使用 UART 的硬件流控制。在网络阻塞的情况下，通过拉高模块的 CTS 脚暂时停止嵌入式设备上 UART 的数据发送。这样可以在任何网络条件下，保证数据的可靠传输。

同样，如果嵌入式设备比较繁忙，不能及时接收模块发送的串口数据时，可以可将 RTS 拉高，这样模块就可以暂时中止对嵌入式设备发送串口数据。

使用硬件流控制后，模块可以控制嵌入式设备的数据收发，将串口速率控制在合适的范围之内。而且这种控制完全是由硬件控制的，不需要软件的干预，不会影响系统的性能。当然，和模块相连的嵌入式设备也需要打开硬件流控制功能，才能正确使用该功能。

TIPS：在硬件流控制下的 UART 时序

当 UART 接收器可以接收数据时，在自己的 RTS 引脚上会产生高电平，而数据正在接收中，或者正在进行数据处理，不能接收下一个 UART 数据时，就会将 RTS 拉高。如下如图 4.2。

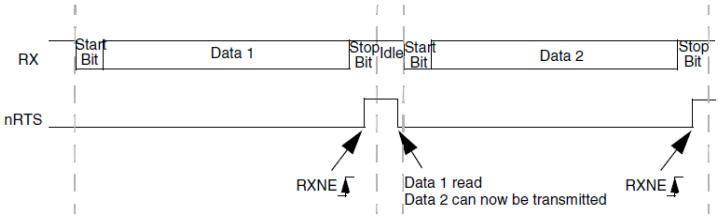


图 4.2 在硬件流控制下的 UART 时序图

RTS 引脚和发送端的 CTS 引脚相连，当 CTS 引脚为高电平时，发送端将不会发送数据。如下图 4.3，DATA2 和 DATA3 这两个数据的传输过程中会有一段 IDLE 时间。

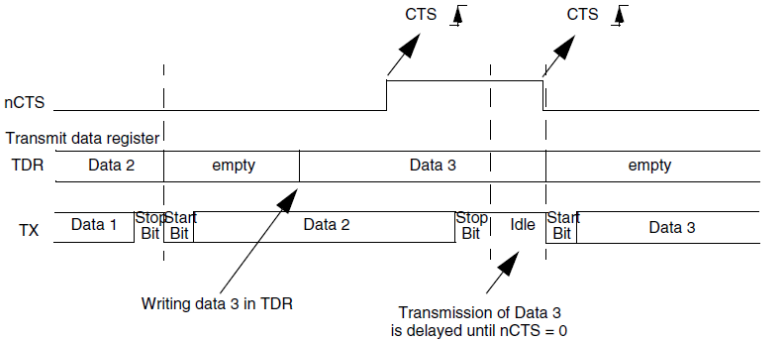


图 4.3 在硬件流控制下的 UART 时序图

5.2 通过 IE 浏览器配置

使用浏览器进行配置，使得任何 PC 机，或者带有 Wi-Fi 功能的智能手机都能够对模块进行配置。

要通过 IE 浏览器进行配置，需要保证与模块可以进行网络通讯，在模块的数据通讯模式下，模块可以自动建立网络通讯。因此，通常在这种模式下通过浏览器对模块进行配置。

操作流程：

1、 将模块置于配套的测试底座上（如 EMW-380-S），将底座上的 Status 跳线连接至 VCC。

2、 建立网络连接

将模块通电，模块将自动连接到 Wi-Fi 网络。如果不清楚模块的网络设置，可以通过上一节介绍的“380 配置工具”软件，将模块恢复到出厂状态。在出厂状态下的模块的网络设置为：Ad-Hoc 无线模式，网络名称为“MXCHIP”，IP 地址为 192.168.4.55，子网掩码为 255.255.255.0。

在 PC 上使用无线管理工具，加入与模块同一个网络中，并将 PC 的 IP 地址设在与模块同一个网段中。如果模块刚恢复了出厂状态，PC 可以找到一个名叫“MXCHIP”的网络，加入这个网络，并将 IP 地址设成 192.168.4.XX（XX 是任意值）。这时，模块上的红色 LED 灯会点亮。

3、 在 PC 上打开网络浏览器，在地址栏中输入模块的 IP 地址（默认为 192.168.4.55）.

在浏览器中可以看到以下界面，用户可以根据上一节讲述的各个参数的意义来进行配置。

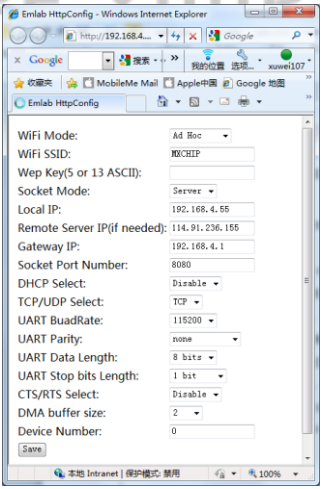


图 4.4 通过浏览器对模块进行配置

如果客户需要自己编写串口配置软件，或者需要通过嵌入式设备的串口对模块进行配置。请参阅以下关于 EMSP 命令的章节。

5.3 EMSP 命令协议规约

5.3.1 命令控制模式下的通讯模型

EMW-380 模块的命令控制模式使用的协议是典型的主从方式的通讯协议，即嵌入式设备充当 master，而 EMW-380 模块充当 slave。当嵌入式设备与 EMW-380 进行通讯时，由嵌入式设备发送请求给模块，模块返回响应。嵌入式设备充当 master，发送所有请求，模块充当 slave，响应这些请求。

所有的请求和应答都进行了校验和计算，从而确保数据交换的完整性和可靠性。

在模块初始化完毕后，嵌入式设备可以使用本规范中提供的协议与模块进行通讯。由于通讯协议中各命令的功能不同，所以处理时间也不完全相同。因此，发送完命令后，因该是一段时间，等待串口接收模块的返回值。

接口时序

出厂状态下模块的 UART 通讯接口使用的帧尺寸为 8 位数据位、无校验、1 位停止位、115200 波特率。如果用户修改过这些参数，则使用修改后的参数。也可以随时将模块恢复到出厂设置。

协议规约

每个发送到 EMW-380 模块的命令由一个协议头（8 个字节）和数据段（不定长，最大 256 个字节）组成的。其格式如下：

[<command><length><result><head_checksum>][<data><data_checksum>]

协议头

协议头由一个命令字段（2 字节）、一个长度字段（2 字节）、一个结果码（2 个字节，由模块返回）和一个头部校验和（2 个字节）构成。其格式如下：

[<command><length><result><head_checksum>]

<command>：命令字段，2 个字节。

<length>：长度字段，2 个字节。表示当前整个请求包的长度，包括协议头和数据段两部分。

<result>：结果码，2 个字节。请求和应答使用相同的协议头，此字段仅在应答包中有效。

<head_checksum>：协议头校验和，2 个字节。它用于校验协议头是否完整。

数据段

数据段中包括当前命令中的所有数据，最后再加上一个数据校验和（2 字节）。其格式如下：

[<data><data_checksum>]

<data>：数据，它的长度不确定，可以由协议头中的长度字段计算得到数据的长度。

<data_checksum>：数据段校验和，2 个字节。它用于校验数据段的完整性。

注：协议头和数据段的校验和相互独立，即头部校验和是对整个协议头的校验。数据段的校验和是对整个数据段的校验。

校验方法

请按照以下 C 代码进行校验。

```
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;
}
```

5.3.2 命令详述

整个协议包含 9 条命令。完成模块的控制、网络控制和网络通讯。

EMSP_CMD_RESET

此命令用来对模块进行复位操作。也可以使用模块的 J1-PIN2 进行硬件复位。

当执行完此命令后，模块重启，并重新初始化（需要 2 秒），此后可以重新开始对模块进行操作。

嵌入式设备发送的请求为：01 00 0A 00 00 00 F4 FF FF FF

<command>：0x0001

<length>：0x000A

<result>：0x0000

<head_checksum>：0xFFF4

<data>：由于请求中不需要附带任何数据，因此数据段中的数据长度为 0，即无任何数据

<data_checksum>：0xFFFF

模块返回的应答为：01 00 0A 00 01 00 F3 FF FF FF

<command>：0x0001

<length>：0x000A

<result>：0x00001，结果字段值为 0x0001，表示操作成功

<head_checksum>： 0xFFF3

<data>：由于请求中不需要附带任何数据，因此数据段中的数据长度为 0，即无任何数据

<data_checksum>：0xFFFF

EMSP_CMD_GET_CONFIG

此命令用于获取模块中的配置信息，包括 TCP/IP，WIFI，串行接口的配置信息。

嵌入式设备发送的请求为：02 00 0A 00 00 00 F3 FF FF FF

<command>： 0x0002

<length>： 0x000A

<result>： 0x0000

<head_checksum>： 0xFFF3

<data>： 由于请求中不需要附带任何数据，因此数据段中的数据长度为 0，即无任何数据

<data_checksum>： 0Xffff

模块返回的应答为：02 00 88 00 01 00 74 FF …… XX XX

<command>： 0x0002

<length>： 0x0088

<result>：0x0001，结果字段值为 0x0001，表示操作成功

<head_checksum>： 0xFF74

<data>：……，数据段的结构如下

```
typedef struct
{
 // WIFI
 u8 wifi_mode;          // Wlan802_11IBSS(0), Wlan802_11Infrastructure(1)
 u8 wifi_ssid[32];      //
 u8 wifi_wepkey[16];    // 40bit and 104 bit
 u8 wifi_wepkeylen;     // 5, 13

 // TCP/IP
 u8 local_ip_addr[16];       // if em380 is server, it is server's IP;    if em380 is client, it is em380's default IP(when DHCP is disable)
 u8 remote_ip_addr[16];      // if em380 is server, it is NOT used;       if em380 is client, it is server's IP
```

```
    u8 net_mask[16]; // 255.255.255.0
    u8 gateway_ip_addr[16];        // gateway ip address
    u8 portH;              // High Byte of 16 bit
    u8 portL;              // Low Byte of 16 bit
    u8 connect_mode; // 0:server  1:client
    u8 use_dhcp;           // 0:disale, 1:enable
    u8 use_udp;            // 0:use TCP,1:use UDP
    // COM
    u8 UART_buadrate;      //0:9600, 1:19200, 2:38400, 3:57600, 4:115200, 5:230400, 6: 460800,
7：921600
    u8 DMA_buffersize;     // 0:2, 1:16, 2:32, 3:64, 4:128, 5:256, 6:512
    u8 use_CTS_RTS;        // 0:disale, 1:enable
    u8 parity;             // 0:none, 1:even parity, 2:odd parity
    u8 data_length;        // 0:8, 1:9
    u8 stop_bits;          // 0:1, 1:0.5, 2:2, 3:1.5
    // DEVICE
    u8 device_num;         // 0 - 255
  } EM380C_parm_TypeDef;
```

<data_checksum>：0xXXXX，不同的数据段有不同的校验和

EMSP_CMD_SET_CONFIG

此命令用来设置模块的参数。参数修改后，应调用复位命令，使这些参数生效。。

嵌入式设备发送的请求为：03 00 88 00 00 00 74 FF …… XX XX

<command>： 0x0003

<length>： 0x0088

<result>： 0x0000

<head_checksum>： 0xFF74

<data>：……， 需要写入模块的参数，参数结构请参阅 EMSP_CMD_SET_CONFIG 命令的描述

<data_checksum>： 0xXXXX，不同的数据段有不同的校验和

模块返回的应答为：03 00 88 00 01 00 73 FF …… XX XX

<command>： 0x0003

<length>： 0x0088

<result>：0x0001，结果字段值为 0x0001，表示操作成功

<head_checksum>： 0xFF73

<data>：……， 写入参数写入模块之后，模块返回参数值，参数结构请参阅 EMSP_CMD_SET_CONFIG 命令的描述

<data_checksum>： 0xXXXX，不同的数据段有不同的校验和

EMSP_CMD_SCAN_AP

此命令用于获取模块可识别范围内的 AP。

嵌入式设备发送的请求为：04 00 0A 00 00 00 F1 FF FF FF

<command>： 0x0004

<length>： 0x000A

<result>： 0x0000

<head_checksum>： 0xFFF1

<data>： 由于请求中不需要附带任何数据，因此数据段中的数据长度为 0，即无任何数据

<data_checksum>： 0xFFFF

模块返回的应答为：04 00 YY YY ZZ ZZ VV VV …… XX XX

<command>： 0x0004

<length>： 0xYYYY，根据不同的搜索结果，长度也会不同

<result>：0xZZZZ，表示有效 AP 数为 ZZZZ 个

<head_checksum>： 0xVVVV，根据 head 的不同而有所不同

<data>： 数据段数据此处省略，它包含有效 AP 的 SSID 及其信号强度 RSSI 值，如果无有效 AP，则此处无数据。 数据中的 AP SSID 及 RSSI 值，都是以\0 为结束符的字符串。

<data_checksum>： 0xXXXX，不同的数据段有不同的校验和

EMSP_CMD_START

启动模块的 Wi-Fi 连接和 TCP 连接。命令执行之后，模块会自动连接 Wi-Fi 链路和 TCP 数据通道。可以通过数据收发命令来测试数据的传输，或者直接拉高 status 引脚，直接将模块切换到数据传输模式来进行数据透明传输。

嵌入式设备发送的请求为： 05 00 0A 00 00 00 F0 FF FF FF

<command>：0x0005

<length>：0x000A

<result>：0x0000

<head_checksum>：0xFFF0

<data>：由于请求中不需要附带任何数据，因此数据段中的数据长度为 0，即无任何数据

<data_checksum>：0xFFFF，根据<data>字段进行校验

模块返回的应答为：05 00 0A 00 01 00 EF FF FF FF

<command>：0x0005

<length>：0x000A

<result>：0x00001，结果字段值为 0x0001，表示操作成功

<head_checksum>： 0xFFEF

<data>：由于请求中不需要附带任何数据，因此数据段中的数据长度为 0，即无任何数据

<data_checksum>：0xFFFF

EMSP_CMD_SEND_DATA

此命令用于通过模块向网络上发送数据（建议使用数据传输模式来收发数据）。

嵌入式设备发送的请求为： 06 00 XX XX 00 00 YY YY …… ZZ ZZ

<command>： 0x0006

<length>： 0xXXXX，根据实际的长度填写

<result>： 0x0000

<head_checksum>： 0xYYYY,根据实际的 head 进行校验

<data>：……， 将需要发送的数据填入此处

<data_checksum>： 0xZZ ZZ，根据<data>字段进行校验

模块返回的应答为：06 00 0A 00 XX XX YY YY FF FF

<command>： 0x0006

<length>： 0x000A

<result>：0xXXXX，表示成功通过网络向对方发送了 0xXXXX 个字节数据，0 表示发送失败

<head_checksum>： 0xYYYY，根据实际的 head 进行校验

<data>：据由于请求中不需要附带任何数据，因此数据段中的数据长度为 0，即无任何数据

<data_checksum>： 0xFFFF

EMSP_CMD_RECV_DATA

此命令用于通过模块从网络上接收数据(建议使用数据传输模式来收发数据）。

嵌入式设备发送的请求为： 07 00 0A 00 00 00 EE FF FF FF

<command>： 0x0007

<length>： 0x000A

<result>： 0x0000

<head_checksum>： 0xFFEE

<data>： 由于请求中不需要附带任何数据，因此数据段中的数据长度为 0，即无任何数据

<data_checksum>： 0xFFFF

模块返回的应答为：07 00 XX XX YY YY ZZ ZZ …… VV VV

<command>： 0x0007

<length>： 0xXXXX，返回的命令长度

<result>：0xYYYY，表示成功通过网络从对方收到 0xYYYY 个字节数据，0 表示无数据

<head_checksum>： 0xZZZZ，根据实际的 head 进行校验

<data>：：……， 接收到的数据

<data_checksum>： 0xVVVV，根据<data>字段进行校验

EMSP_CMD_GET_STATUS

该命令主要用于读取模块在数据传输模式的网络状态，应答中的数据为 0x000X000Y，它的 X 位表示 TCP 的状态（1 为侦听，2 为已连接），Y 位表示 WiFi 的工作状态，5 表示处于工作状态，0 表示还没有启动 WiFi。

嵌入式设备发送的请求为： 08 00 0A 00 00 00 ED FF FF FF

<command>：0x0008

<length>：0x000A

<result>：0x0000

<head_checksum>：0xFFED

<data>：由于请求中不需要附带任何数据，因此数据段中的数据长度为 0，即无任何数据

<data_checksum>：0xFFFF

模块返回的应答为：08 00 0E 00 01 00 E8 FF XX XX YY YY ZZ ZZ

<command>：0x0008

<length>：0x000E

<result>：0x00001，结果字段值为 0x0001，表示操作成功

<head_checksum>： 0x FFE8

<data>：0xXXXX 表示 TCP 的状态（1 为侦听，2 为已连接），0xYYYY 表示 WiFi 的工作状态，5 表示处于工作状态，0 表示还没有启动 Wi-Fi。

<data_checksum>：0xZZZZ，根据<data>字段进行校验

EMSP_CMD_GET_VER

此命令用于获取模块版本号，包括硬件版本和固件版本。

嵌入式设备发送的请求为： 09 00 0A 00 00 00 EC FF FF FF

<command>： 0x0009

<length>： 0x000A

<result>： 0x0000

<head_checksum>： 0xFFEC

<data>： 由于请求中不需要附带任何数据，因此数据段中的数据长度为 0，即无任何数据

<data_checksum>： 0xFFFF

模块返回的应答为：09 00 0E 00 01 00 E7 FF XX XX XX XX ZZ ZZ

<command>： 0x0009

<length>： 0x000E

<result>：0x00001，结果字段值为 0x0001，表示操作成功

<head_checksum>： 0x FFE7

<data>：0xXXXXXXXX 版本，该版本应与模块标签上的版本号码保持一致

<data_checksum>： 0xZZZZ，根据<data>字段进行校验

EMSP_CMD_GET_MF_INFO

此命令用于获取模块的生产厂家信息。

嵌入式设备发送的请求为： 20 00 0A 00 00 00 D5 FF FF FF

<command>： 0x0020

<length>： 0x000A

<result>： 0x0000

<head_checksum>： 0xFFD5

<data>： 由于请求中不需要附带任何数据，因此数据段中的数据长度为 0，即无任何数据

<data_checksum>： 0xFFFF

模块返回的应答为：09 00 0E 00 01 00 E7 FF XX XX XX XX ZZ ZZ

<command>： 0x0020

<length>： 0x0010

<result>：0x00001，结果字段值为 0x0001，表示操作成功

<head_checksum>： 0x FFCE

<data>：0x6d 78 63 68 69 70，表示“MXCHIP”

<data_checksum>： 0xAEC5，根据<data>字段进行校验

EMSP_CMD_GET_RF_POWER

此命令用于获取模块无线收发功率。

嵌入式设备发送的请求为： 09 00 0A 00 00 00 EB FF FF FF

<command>： 0x0009

<length>： 0x000A

<result>： 0x0000

<head_checksum>： 0xFFEB

<data>： 由于请求中不需要附带任何数据，因此数据段中的数据长度为 0，即无任何数据

<data_checksum>： 0xFFFF

模块返回的应答为：09 00 0E 00 01 00 E6 FF XX YY ZZ 00 VV VV

<command>： 0x0009

<length>： 0x000E

<result>：0x00001，结果字段值为 0x0001，表示操作成功

<head_checksum>： 0x FFE6

<data>： XX YY ZZ，XX 表示模块支持的最小功率，YY 表示模块支持的最大功率，ZZ 表示模块当前的功率

<data_checksum>： 0xVVVV，根据<data>字段进行校验

EMSP_CMD_SET_RF_POWER

此命令用来设置模块发射无线功率。

嵌入式设备发送的请求为：0B 00 0B 00 00 00 E9 FF XX YY YY

<command>：0x000B

<length>：0x000B

<result>：0x0000

<head_checksum>：0xFFE9

<data>：模块的发射功率

<data_checksum>：0xYYYY，不同的数据段有不同的校验和

模块返回的应答为：0B 00 0B 00 01 00 E8 FF XX YY YY

<command>：0x000B

<length>：0x000B

<result>：0x0001，结果字段值为 0x0001，表示操作成功

<head_checksum>： 0xFFE8

<data>：XX，模块的发射功率

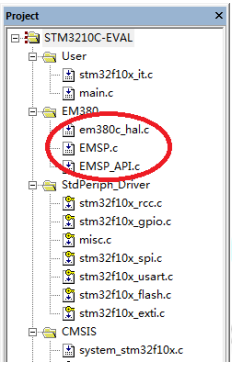
<data_checksum>：0xYYYY，不同的数据段有不同的校验和

5.4 EMSP 接口函数库

EMSP 接口函数库提供了一系列 API 函数，用户通过调用这些函数可以轻松地在各种嵌入式设备上实现对 EMW-380 模块的控制和参数配置。现在，该接口函数库随 MDV-STM32-107 开发板的 Wi-Fi 示例程序一并提供下载，下载地址如下：

MDV-STM32-107 开发板

下载对应的 Wi-Fi WLAN 示例程序即可。如果购买了 MDV-STM32-107 和 EMB-380-C 模块，用户可以在 MDV-STM32-107 板上调试这些示例程序。



EMSP 接口函数库由标准 C 编写而成，可以直接加入到常用的嵌入式开发环境，如 KEIl,IAR 等。

EMSP 接口函数库由以下三个 C 语言文件及其对应的头文件构成。

1. em380c_hal.c

   该代码实现了 EMW-380 模块和嵌入式设备之间的硬件接口。用户需要根据自己的硬件环境实现相应的函数。

2. EMSP.c

   该代码实现了 EMSP 命令的协议处理。

3. EMSP_API.c

   该代码提供给用户用于操控模块的 API 函数，用户只需要调用这些函数，就可以对模块实现配置和操作。

5.4.1 API 函数一览

vs8 EM380C_Init(uint32_t BaudRate, uint16_t WordLength, uint16_t StopBits,uint16_t Parity,uint16_t CTSRTS,uint16_t buffer);

函数功能：用于初始化 EM380C 模块,直接写入串口波特率，可以快速的进行初始化

参数 1（OUT）：串口波特率:9600, 19200, 38400, 57600, 115200，230400，460800，921600

参数 2（OUT）：数据位长度：USART_WordLength_8b, USART_WordLength_9b

参数 3（OUT）：停止位长度：USART_StopBits_1, USART_StopBits_0_5, USART_StopBits_2, USART_StopBits_1_5

参数 4（OUT）：校验方法： USART_Parity_No, USART_Parity_Even, USART_Parity_Odd

参数5(OUT):硬件流控制：USART_HardwareFlowControl_None, USART_HardwareFlowControl_RTS_CTS

参数 6（OUT）：串口缓冲长度：USART_HardwareFlowControl_None,USART_HardwareFlowControl_RTS_CTS

返回值：-1：执行命令失败

0：执行命令成功

vs8 EM380C_InitWithDefault(void);

初始化 EM380C 模块，并将模块的参数都置于初始状态。该函数的执行速度比 EM380C_Init 慢。一般用于首次对模块进行操作。

返回值：-1：执行命令失败

0：执行命令成功

vs8 EM380C_Get_ver(u32* version)

函数功能：用于获得 EM380C 的固件版本号

参数 1（IN）：用于存放获得的 EM380C 的固件版本号的地址

返回值：-1：执行命令失败

0：执行命令成功

vs8 EM380C_Get_status(EM380C_status_TypeDef* EM380C_status)

函数功能：用于获得 EM380 的网络连接状态

参数 1（IN）：用于存放 EM380 的网络连接状态结构体的地址

```
typedef struct
{
  EM380C_TCPstatus_TypeDef TCPstatus; // TCP_stop,TCP_listening,TCP_connected,TCP_unknown
  EM380C_WiFistatus_TypeDef  WiFistatus;// wifi_disconnected,wifi_connected,wifi_unknown
} EM380C_status_TypeDef;
```

返回值：-1：执行命令失败

0：执行命令成功

vs8 EM380C_Get_APList(EM380C_APLst_TypeDef* EM380C_APLst);

函数功能：用于获得区域内无线 AP 的 SSID 号和相应的信号强度

参数 1（IN）：用于存放无线 AP 的 SSID 号和相应的信号强度的线性表的起始地址

```
typedef struct
{
  char   AP_NAME[20];
  float   AP_signal;
} EM380C_APLst_TypeDef;
```

返回值：-1：执行命令失败

>=0：执行命令成功，获得的 AP 信息的数量

vs8 EM380C_Startup(void)

函数功能：启动 EM380 的 Wi-Fi 连接和 TCP/IP 网络连接

返回值：-1：执行命令失败

0：执行命令成功

vs8 EM380C_Get_RF_POWER(EM380C_RF_POWER_TypeDef* RF_POWER)

函数功能：获得 EM380 的 Wi-Fi 功率

参数 1（IN）：用于存放获得的 EM380 的 Wi-Fi 功率最小值，最大值和当前值

返回值：-1：执行命令失败

0：执行命令成功

vs8 EM380C_Set_RF_POWER(EM380C_RF_POWER_TypeDef* RF_POWER);

函数功能：设置 EM380 的 Wi-Fi 功率

参数 1（OUT）：用于存放设置的的 EM380 的 Wi-Fi 功率最小值，最大值和当前值（最大值和最小值不可修改）

返回值：-1：执行命令失败

0：执行命令成功

vs8 EM380C_Get_Config(EM380C_parm_TypeDef* EM380C_Parm)

函数功能：用于获得 EMW-380 模块当前的配置参数

参数 1（IN）：参数结构体的地址，成功执行命令之后，模块当前的参数会写入这个地址。EM380C_parm_TypeDef 结构中各参数的意义见相应的 EMSP_Get_Config 命令

返回值：-1：执行命令失败

0：执行命令成功

vs8 EM380C_Set_Config(EM380C_parm_TypeDef* EM380C_Parm)

函数功能：用于设置 EM380C 的配置参数

参数 1（OUT）：参数结构体的地址，成功执行命令之后，会将该地址上的数据写入到 模块中。

返回值：-1：执行命令失败

0：执行命令成功

5.4.2 硬件接口函数一览

这些函数需要用户根据 EMW-380 模块和用户自己的嵌入式设备的连接情况，实现这些函数的功能，以支持 API 函数的运行。具体的函数实现可以参考我们提供的示例程序。

vs8 EM380C_HAL_Init(uint32_t BaudRate,uint16_t WordLength,uint16_t StopBits,uint16_t Parity,uint16_t CTSRTS)

函数功能：初始化硬件接口，并且复位 EMW-380 模块

参数 1（OUT）：串口波特率:9600, 19200, 38400, 57600, 115200,230400,460800,921600

参数 2（OUT）：数据位长度：USART_WordLength_8b, USART_WordLength_9b

参数 3（OUT）：停止位长度：USART_StopBits_1, USART_StopBits_0_5, USART_StopBits_2, USART_StopBits_1_5

参数 4（OUT）：校验方法： USART_Parity_No, USART_Parity_Even, USART_Parity_Odd

参数5(OUT):硬件流控制：USART_HardwareFlowControl_None, USART_HardwareFlowControl_RTS_CTS

返回值：-1：执行命令失败

0：执行命令成功

vs8 EM380C_HAL_InitWithDefault(void)

函数功能：以默认值初始化硬件接口，并将 EMW-380 模块恢复到默认设置，然后复位 EMW-380 模块

返回值：-1：执行命令失败

0：执行命令成功

void UART_send_buf(u8 *buf, int len)

函数功能：通过串口发送一段存储区中的数据

参数 1（OUT）：数据存储区的起始地址

参数 2（OUT）：发送数据的数据长度

返回值：无

int UART_receive_buf(u8 *buf)

函数功能：通过串口接收数据，并将接收到的数据放在存储区中

参数 1（IN）：数据存储区的起始地址

返回值：接收到数据的长度

void Delay(__IO uint32_t nTime)

函数功能：提供一段时间的延时

参数 1（IN）：延时的时间，以毫秒为单位

返回值：无

6 数据通讯模式（透明传输模式）

在数据通讯模式下，模块可以实现 UART 接口和 Wi-Fi 无线网络接口中数据的透明传输。这是 EMW-380 系列嵌入式 Wi-Fi 模块所特有的传输模式，可以最大限度地降低用户的开发成本。

什么是透明传输？

数据在可靠的网络中进行传输必须遵循一定的数据包格式，比如在网络中进行数据传输，应该将数据封装成 TCP 数据包，然后根据传输的目的，再封装成 IP 数据包，等等。TCP/IP 协议栈就是实现对这些数据包进行封装的功能的软件。因此要实现数据在网络中的传输，首先要实现的就是网络协议栈。这些协议栈在普通的 PC 操作系统中是已经实现的，但是针对资源有限的嵌入式系统，PC 上复杂的网络协议栈是很难实现的，需要花费较多的软件开发成本。

EMW-380 无线模块内部实现了在网络中传输数据所需要的网络协议栈，可以自动地将串口的数据进行封装，实现这些数据在网络中的传输，也可以将网络中获得的数据包转换成串口可以识别的数据。网络中的其他网络设备可以直接获取到嵌入式设备通过串口发出的数据，嵌入式设备通过串口也可以直接得到网络设备所发出的数据，而无须进行数据格式的调整。连接两台通讯设备之间的是复杂的网络，而 EMW-380 无线模块可以将数据如何在网络中进行传输的功能隐藏起来，就好像把两台通讯的设备直接连接起来， 这就是所谓的透明传输。

简单来讲，就是下面两点：

- 嵌入式设备通过串口发送的数据被完整的发送到远程网络设备所接受到的 TCP/UDP 数据包的负载中。
- 远程网络设备所发送的 TCP/UDP 数据包的负载完整的发送到嵌入式设备的串口上。

以下两种情况下，模块可以进入数据通讯模式

1、将模块的 Status 引脚拉高，然后对模块硬件复位，或者重新上电。

2、在命令控制模式下，调用 EMSP_CMD_START 命令启动网络连接，然后将 Status 引脚拉高。

如果您使用 EMW-380-S 系列测试底板，请用跳线将 Status 引脚接至高电平。

6.1 简要操作流程

首先要保证模块的 STATUS 引脚接到高电平，才能启动模块的数据通讯模式。在这种模式下重新打开模块的电源，模块会按照您设置的参数连接 Wi-Fi 网络，并尝试实现数据通讯。你无需再对模块进行控制，一切都是自动运行的。

在测试数据通讯的时候请先保证关闭 PC 上的所有网络防护墙，或者单独设置允许模块通讯端口的数据通过，放心，模块是不会攻击您的 PC 的。

1、 建立 Wi-Fi 网络无线链路，保证网络设备与模块之间能够相互访问

该过程就好比在普通的有线以太网络中连接网线的过程。Wi-Fi 网络分为需要 Access Point 的星型结构网络和不带 Access Point 的 Ad-Hoc 网络。正确建立好 Wi-Fi 网络后，模块上的红色指示灯会常亮。如果模块的 IP 地址设置正确，那么在一个网络中的其他网络设备可以用 ping 命令查看和模块之间的网络通讯是否正常。

2、 建立 TCP/UDP 网络通讯链路

TCP 模式下的通讯好比在已经建成的电话网络中，拨通的对方电话。在 TCP 模式下，通讯的双方一边是客户端，另一边是服务器。客户端可以主动去“拨通”服务器的“电话”，而服务器则必须始终保证可以接受客户端的连接请求，在电话拨通之后，这个“神奇的电话”可以保证您传递的数据完整不丢失。在 UDP 模式下，如果需要向网络设备发送数据是不需要先“拨通电话”的，就好比你直接和另一个人说话，而不用管那一个人是否听到了你说的内容。

直接说话比打电话方便多了，不是么？

如果您在上面的两个步骤中遇到了问题，请重新回到第四章，看看 “模块配置参数详述”，检查一下，是否有哪一个参数设置错误了。

您可以在 PC 上使用我们提供的一个名叫“TCP/UDP 测试工具”的一个共享软件，这个软件可以在 PC 上建立 TCP 服务器或者客户端，或者进行 UDP 通讯。此外，还可以通过这个软件和与模块连接的嵌入式设备进行数据通讯，体验一下透明传输带给我们的方便和快捷。

通常您要设计自己的应用程序和模块进行网络通讯，“TCP/UDP 测试工具”也是一个很好的范例，同时，编写软件的时候首先应该在当前的平台上熟悉如何进行 socket 编程。比如，在标准 C 下面有 socket.h 提供的 socket API 函数，MFC 下面提供的 CSocket 类等等，如果您需要开发 iPhone 平台上的应用程序，就还需要熟悉 BSD socket 接口。

下面，就几种典型的应用模型，介绍一下模块在数据通信模式下的使用方法。

注意，以下所提到的测试软件都为网络上的免费下载版本，在性能上不能完全体现出模块的最大传输带宽，尤其是大多数 PC 上的串口调试软件。因此，以下的测试方法仅为演示模块的基本功能，实际应用中的上位机通讯软件均需要用户自行开发，以适应不同的应用。

如果用户需要就精确评估模块的传输性能，请使用我们提供的模块评估板 MDV-STM32-107。

6.2 使用前的准备工作

6.2.1 准备硬件环境

1、 一台安装有无线网卡的 PC

一般笔记本电脑都会安装无线网卡，普通的台式机可以购买 USB 无线网卡，如果需要更好的效果可以购买带天线的 PCI 接口无线网卡

2、 一台带串口的 PC

建议使用带真实串口的 PC。通过 USB 转出的串口功能由于采用的 USB 转串口芯片的型号不同导致功能各不相同，并且存在很多的兼容性问题，有些不支持硬件流控制，因此不能够测试模块的完整功能。

3、 一个 Access Point（AP）或者无线路由器

什么是 Access Point？

各种文章或厂家在面对无线 AP 时的称呼目前比较混乱，但随着无线路由器的普及，目前的情况下如没有特别的说明，我们一般还是只将所称呼的无线 AP 理解为单纯性无线 AP，以便和无线路由器加以区分。**它主要是提供无线工作站对有线局域网和从有线局域网对无线工作站的访问，在访问接入点覆盖范围内的无线工作站可以通过它进行相互通信。**

多数单纯性无线 AP 本身不具备路由功能，包括 DNS、DHCP、Firewall 在内的服务器功能都必须有独立的路由或是计算机来完成。目前大多数的无线 AP 都支持多用户（30-100 台电脑）接入，数据加密，多速率发送等功能，在家庭、办公室内，一个无线 AP 便可实现所有电脑的无线接入。

它的作用其实就类似于我们常用的有线网络中的集线器。

什么是无线路由器？

无线路由器其实是普通的有线路由器加入了 AP 的功能，使得之前的路由器能够通过无线连接 Wi-Fi 设备。相比于普通的 AP，通过无线路由器可以实现多种网络功能，如路由，PPPOE 拨号上网，DHCP 自动分配 IP 地址，网络流量控制和网络安全等。

在应用中，大型企业或者运营商一般采用无线 AP 和其他网络设备路分离的策略，这样可以最优化各个网络功能。在家庭和小企业的应用中，出于成本和方便的考虑，通常使用集成的无线路由器。

使用本模块，只需要处于 AP 的无线信号覆盖下即可。如果需要使用 DHCP 功能，则需要额外的能够提供该功能的网络设备，或者直接使用无线路由器。

在一下的功能演示中，我们采用集成的无线路由器：TL-WR541G+。

图 6.1 TL-WR541G+

4、 本模块

5、 NULL 串口线

NULL 串口线必须保证 TX 和 RX 信号线交叉，CTS 和 RTS 信号线交叉。

6、 EMW-380-S 测试底板

详见相应的产品说明书

7、 5V 直流电源

需要保证 1A 以上的额定电流

6.2.2 准备软件环境

1、 串口调试助手

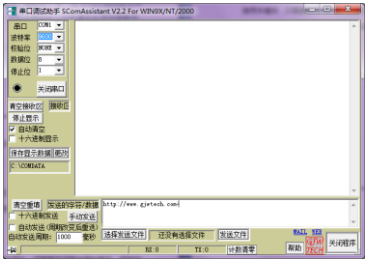


图 6.2 串口调试助手界面

- 作用：用来测试简单的串口数据收发功能。
- 优点：可以收发 16 进制数，允许周期发送。软件界面简明易懂。
- 缺点：但是该软件对中文支持不佳，并且在同时收发数据时会占用大量 CPU 时间，导致丢失数据包。

2、 TCP&UDP 测试工具

图 6.3 TCP&UDP 测试工具

- 作用：用来测试简单的网络 TCP/IP 数据包收发功能。
- 优点：可以演示各种模式下的网络数据收发，软件界面清楚。
- 缺点：显示接收到的数据内容功能会降低网络的传输性能，如果要进行大数据流量的测试，请关闭软件上的显示功能（点击“暂停显示”按钮）。

接下来，我们举两个常用的应用模型来解说模块在数据通讯模式的使用方法。

6.3 测试情景 1：数据服务器和数据终端模式

测试情景 1：各嵌入式设备作为数据终端节点通过 380 模块连接到 Wi-Fi 网络，并且和同一个数据服务器进行数据的交换。在这种模型下，嵌入式设备作为数据终端和模块通过串口相连，PC 或者其他网络设备作为数据服务器采用固定的 IP 地址和模块用过无线网络相连。

见封面的典型通讯模型 A。

6.3.1 建立 WI-FI 网络链路

由于我们需要多个嵌入式设备和服务器进行通讯，因此通过 AP 组成星形拓扑网络比较合适。

配置 AP 的无线参数

无线路由器也是作为我们所要组成的星形网络中的一员存在的，因此它也有自己的 IP 地址，我们访问无线路由器都是通过这个 IP 地址进行的。如果不知道这个无线路由器的 IP 地址，可以参照说明书将它恢复成出厂默认设置。

一般的无线路由器都可以通过 IE 浏览器来进行参数的配置，在不清楚无线网络的状态的情况下，可以使用有线网络连接 PC 和无线路由器的以太网口，在 IE 浏览器中输入无线路由器的 IP 地址进行配置。详情请参考无线路由器的说明手册。

虽然在正常的网络通讯中，无线路由器的 IP 地址并没有实际用处，但是由于无线路由器的 DHCP 功能所能自动分配的 IP 地址都是和它本身的 IP 地址在同一个网段中的，所以我们还是尽量将所需要通讯的网络设备和无线路由器的 IP 地址都放在同一个网段中，以方便我们的调试。无线路由器的 IP 地址如图 4.4 所示，在这个例子中，我们将路由器的 IP 地址设置成 192.168.4.1，所有该网段中的网络设备的 IP 地址都应该是 192.168.4.X。

图 6.4 LAN 口设置页面

故障排除

无法连接到无线路由器的参数配置页面：先将无线路由器恢复到出厂状态，然后连接 PC 和路由器的 LAN 口，在 IE 浏览器的地址栏里面输入路由器的默认地址。

如果还是无法连接，请咨询无线路由器生产厂商

无线参数配置页面如图 6.5 所示

图 6.5 无线参数配置页面

参数解释：

SSID 号：无线网络的名称

频段：自由选择，尽量不要和其他网络使用同一个频段；

模式：自由选择，模块支持 802.11b 或者 802.11g；

安全类型：WEP，EMW-380-X 支持该类型加密方式（可选参数）；

安全选项：自动就可以了（可选参数）；

密钥格式选择：选择 ASCII 码便于记忆（可选参数）；

密钥 1：输入网络使用的密钥（可选参数）

故障排除

如果搜索不到无线网络信号，请确认 AP 是否供电正常，配置是否正确。并且重新配置一次无线路由器。

配置无线路由器的 DHCP 功能（可选）

Wi-Fi 模块可以通过无线路由器的 DHCP 功能自动获得 IP 地址，但是数据服务器却不应该使用 DHCP，因为模块会自动连接设定好的数据服务器的 IP 地址，如果数据服务器 IP 地址发生变化，会导致数据通讯异常。

图 6.6 DHCP 设置页面

参数解释：

DHCP：选择启用。

地址池开始地址，地址池结束地址：分配给模块的 IP 地址范围，注意要和无线路由器的 IP 地址在一个网段内。

将 PC 连接到 Wi-Fi 网络

通过 PC 上的网络工具，搜索无线网络，可以搜索到我们刚才配置的无线路由器所创建的网络：Wifi_Test，点击连接即可。PC 会自动连接到 AP。不同的操作系统的界面有所不同，以实际操作界面为准，图 6.7 显示的是 windows 7 的连接界面。

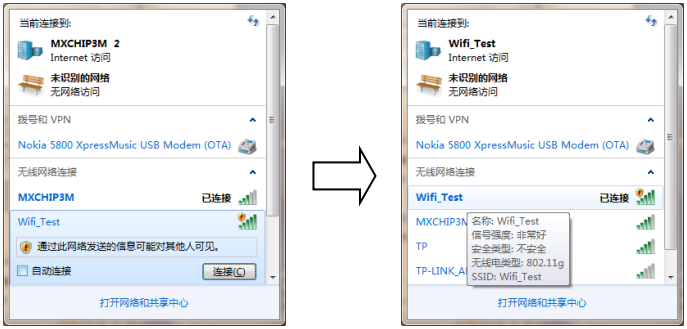


图 6.7 无线网络连接操作界面

故障排除

如果搜索不到无线网络信号，请确认 AP 是否供电正常，配置是否正确。并且重新配置一次无线路由器。

1、 设置 PC 的 IP 地址

在本例中，我们将 PC 的 IP 地址设置成 192.168.4.17，Wi-Fi 模块将会自动连接这个 IP 地址进行数据通信。

图 6.8 PC 机 IP 地址设置操作流程图

设置好之后，我们测试一下，打开 windows 的命令行，输入 ping 192.168.4.1，看是否可以进行通讯。

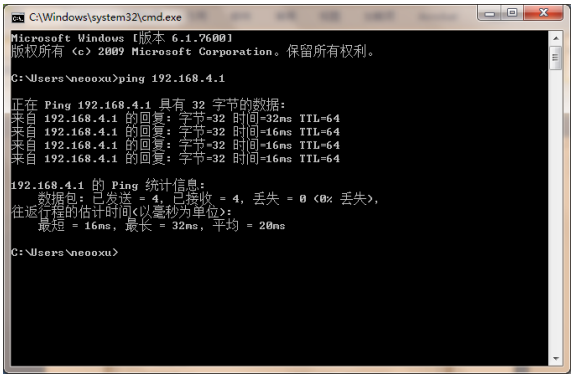


图 6.9 ping 界面

设置模块的参数

将模块插在对应的测试底座上，将 Status 引脚（J2）接地，然后插上 5V 直流电源。

观察模块上的 LED 指示灯，绿灯应该常量，红灯不应有动作。

故障排除

红灯常亮：应检查 EMW-380-S 上的 STATUS 引脚。

绿灯不亮：应检查电源和模块是否已经插好。复位引脚上点评是否正常。

通过 NULL 串口线连接 EMW-380-S 测试底座和 PC 机。打开模块配置软件。如图 6.10。

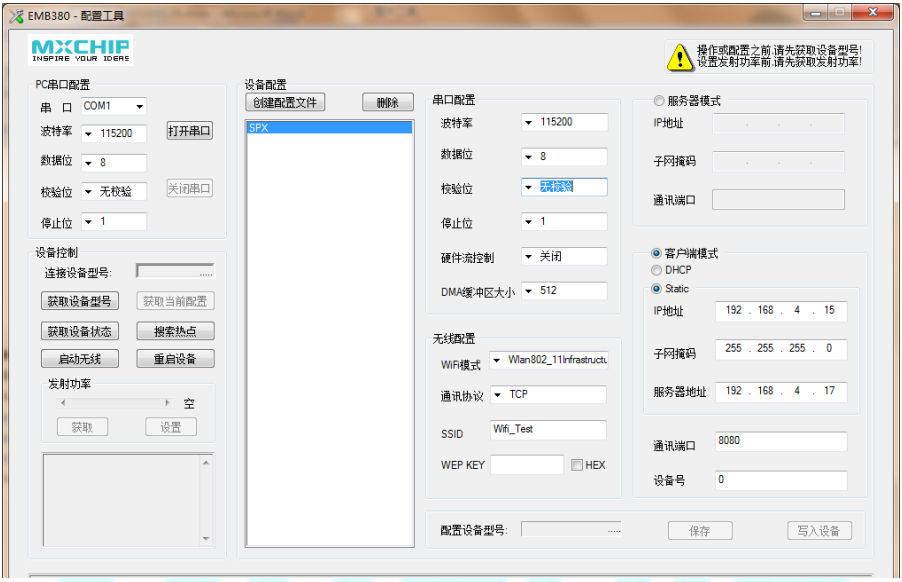


图 6.10 EMW-380 配置软件界面

选择您的 PC 的串口号和预设的 115200 波特率，点击“打开串口”。

故障排除

- ***如果返回“不能打开选择的串口”请确定选择的串口号是否正确。***

点击获取设备型号，看是否能够获得模块的固件版本号，和模块标签上的代码是否一致。

故障排除

没有获得固件版本号。

- ***检查串口线是否连接正常，串口线是否是 TX 和 RX 信号交叉的 NULL 串口线。如果不是 NULL 串口线，可以将 EMW-380-S 上的跳线 J7 和 J11 旋转 90 度。***
- ***可能模块非初次使用，尝试其他串口参数进行连接。***
- ***RS232 芯片和 PC 的串口不兼容，请更换 PC，并且确认使用的串口是真实串口，而不是 USB 转串口。***

版本号和标签上的不匹配。

- ***联系销售代表，更换模块***

点击“获取当前配置”，如果读出的参数和图 6.10 所示的值不一样，请按照如图所示的参数修改，然后选择“写入设备”，进行配置，再读出配置。看参数是否正确写入。

将模块连接到 Wi-Fi 网络

断开模块的电源，将 Status 引脚（J2）接到 VCC，然后插上 5V 直流电源。

观察模块上的 LED 指示灯，绿灯应该常量，过一段时间后红灯常亮。

故障排除

红灯不亮

- ***模块配置参数中的 SSID 号和 WEP 密码和无线路由器上的配置不匹配。***
- ***没有安装 2.4G 天线？天线应该插在模块的 J2 天线插座上。***
- ***检查 EMW-380-S 上的 STATUS 引脚。***

绿灯不亮，应检查电源和模块是否已经插好。复位引脚上电平是否正常。

接下来，可以测试一下，看我们的 IP 地址配置是否正确：

打开 windows 的命令行，输入 ping 192.168.4.15，看是否可以进行通讯。如下图 6.11。

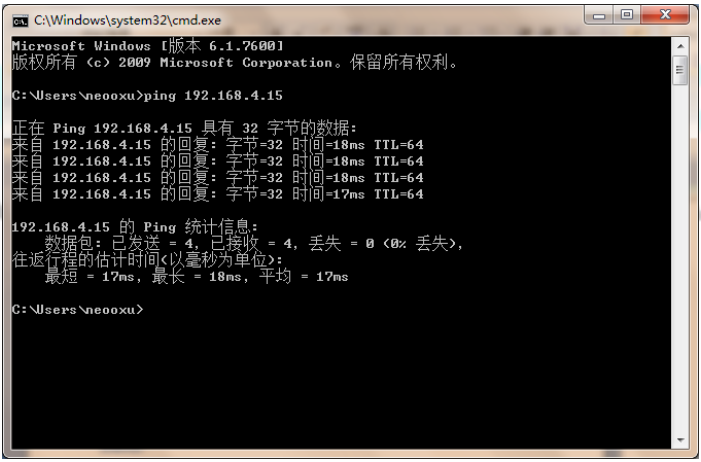


图 6.11 ping 界面

6.3.2 建立 TCP 连接

通过上面的步骤，我们已经成功地将模块和 PC 通过无线路由器连接在一起了，IP 地址也已经正确分配。接下来，我们需要建立 TCP 网络数据通讯通道。

在 PC 上打开 TCP&UDP 测试工具，按照下图的步骤进行操作。

图 6.12 TCP&UDP 测试工具操作流程图

点击启动服务器按钮，模块会自动登录到 PC 服务器，出现以下界面。说明 TCP 连接已经正确建立了。

图 6.13 TCP 建立正确连接界面

故障排除

出现如图 6.14 界面，模块不能自动登录

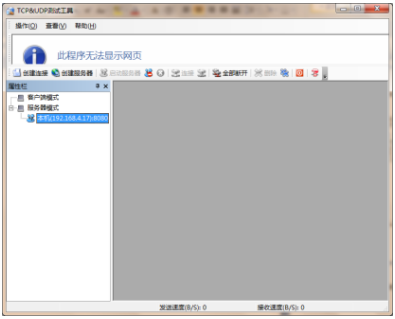


图 6.14 TCP 建立错误连接界面

- *PC 的 IP 地址应该和模块设置的“服务器 IP 地址“参数一致.*
- *PC 是否可以 PING 通模块的 IP 地址？如果不能 PING 通，请重新建立 WI-FI 链路。*
- *模块是否选择 TCP 连接方式和客户端方式？*
- *模块的 PORT NUMBER 参数是否和使用 TCP&UDP 测试工具上建立服务器使用的端口号一致？*
- *PC 的防火墙有没有允许您设置的网络端口进行通讯？尝试关闭所有防火墙。*

6.3.3 测试串口到网络的数据透明传输

关闭 UART configure 配置软件，打开串口调试助手。使用模块配置的波特率（本例是 115200）打开串口。如图 6.15

图 6.15 串口助手设置界面图

测试 1、网络数据发送到串口

我们在 TCP&UDP 测试工具的发送区输入一些数据，然后点击发送，观察串口调试助手的接收区，是否收到同样的数据？

测试 2、串口数据发送到网络

我们在串口调试助手的发送区输入一些数据，然后点击“手动发送”按钮，观察 TCP&UDP 测试工具的接收区，是否收到同样的数据？

测试 3、双向数据收发

分别启动两个测试工具中的自动发送功能，看数据是否传输正确。

数据传输时，模块上的红色 LED 灯会闪烁。

小提示：注意比对两个测试软件发送和接收到数据的长度是否一致，不要单纯用眼观察发送和接收区的数据。此外高速收发数据时，串口调试软件由于本身的性能问题，会导致数据接收丢包。如果要测试模块的网络传输性能，请不要使用该软件。

6.4 测试情景 2：点对点连接和通讯

测试情景 2：PC 或者移动网络设备不定期地连接到嵌入式设备上，以实现相关应用数据的交换。

见封面的典型通讯模型图 B。

在这种模式下，模块的无线模式采用点对点互联的 Ad-Hoc 连接模式.，PC 可以随时连接网络进行通讯，也可以很方便地断开网络。

6.4.1 建立 WI-FI 网络链路

设置模块的参数

通过配置软件配置模块。

按照图 6.16 的参数进行修改，主要修改参数：

WiFi 模式：选择 Wlan802_11IBSS。

选择服务器模式。

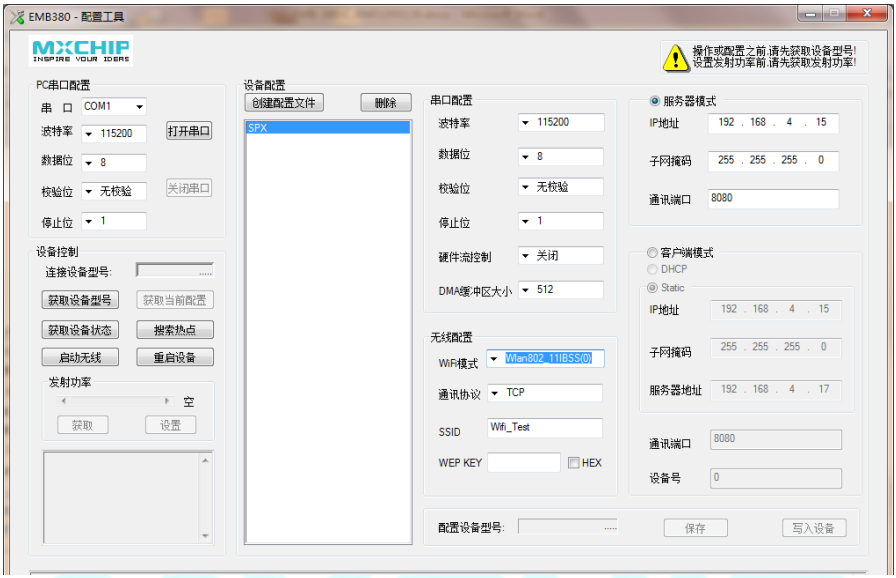


图 6.16 配置界面设置（点对点通讯模式）

连接 Wi-Fi 网络

断开 EMW-380-S 的电源，将 Status 引脚（J2）接到 VCC，然后重新插上 5V 直流电源。

观察模块上的 LED 指示灯，绿灯应该常亮。

故障排除

绿灯不亮，应检查电源和模块是否已经插好，复位信号是否正常。

然后我们配置 PC 的无线连接，使得 PC 和模块相连接。不同的操作系统的界面有所不同，以实际操作界面为准，图 6.17 显示的是 windows 7 的连接界面，注意 ad-Hoc 连接的图标和普通 AP 连接的图标是不同的。

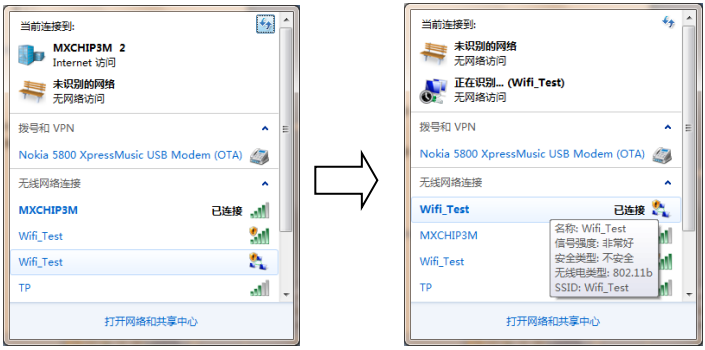


图 6.17 无线网络连接图

成功连接到模块之后，模块上的红色 LED 灯会常亮。

故障排除

没有搜索到这个 AD-HOC 网络

- *没有安装 2.4G 天线？天线应该插在模块的 J2 天线插座上。*
- *模块是否配置成 WLAN802_11IBSS 模式。*

不能连接到这个 AD-HOC 网络

- *是不是设置了网络密钥？如果有网络密钥的话，请保证模块和 PC 使用同一个网络密钥。*

设置 PC 的 IP 地址

设置 PC 的 IP 地址：192.168.4.X，X 表示 1-254 之间任意的整数。并且通过 ping 命令检查是否正常连接。

小贴士：这一次，PC 作为客户端连接到模块，这样 PC 的 IP 地址可以是任意的，只需要和模块的 IP 地址保持在同一个网段中即可。我们总是认为，作为服务器的一方，IP 地址应该是固定的。

故障排除

无法 PING 通模块的 IP 地址：

- *请将模块重新上电，重新进行 WI-FI 连接。*
- *确认模块的 IP 地址配置正确。*

确认模块上的红色 LED 灯在 PC 连接网络后常亮。

6.4.2 建立 TCP 连接

通过以上步骤，我们已经成功地将模块和 PC 以 Ad-Hoc 模式连接在一起了，IP 地址也已经正确分配。接下来，我们需要建立 TCP 网络数据通讯通道。

在 PC 上打开 TCP&UDP 测试工具，按照图 6.18 的步骤进行操作，连接的目标的 IP 地址是 192.168.4.15（模块的 IP 地址），端口是 8080。

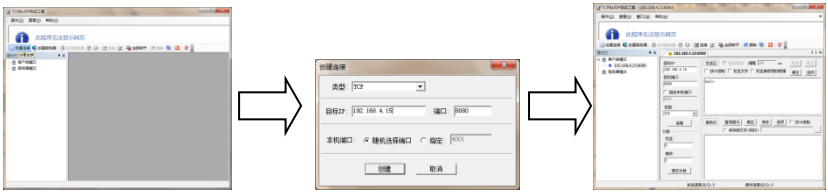


图 6.18 TCP&UDP 测试工具设置流程图

最后我们点击“连接”按钮，“连接”按钮变成“断开连接”，表示 TCP 的连接正常了。

故障排除

“连接”按钮编程“正在连接”

- ***用测试软件连接的远程 IP 地址是否是模块的 IP 地址，在这个例子中，应该是 192.68.4.15.。***
- ***PC 是否可以 PING 通模块的 IP 地址？不能 PING 通的话，请重新建立 WI-FI 连接。***
- ***模块是否设置成 TCP 模式和服务器模式？***
- ***模块的“通讯端口“是否和使用 TCP&UDP 测试工具上建立服务器使用的端口号一致？***
- ***PC 的防火墙有没有允许您设置的网络端口进行通讯？***

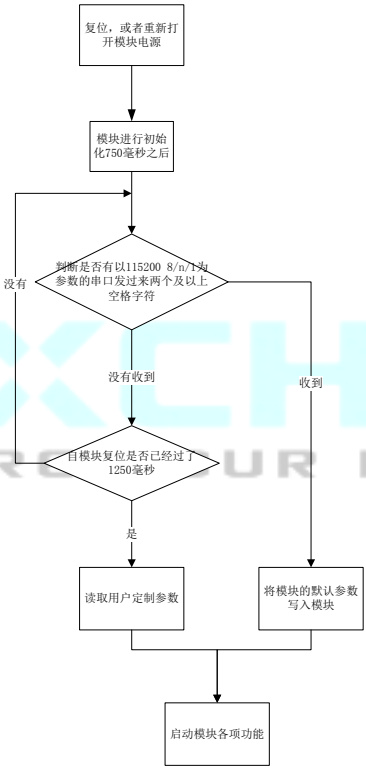
6.4.3 测试串口到网络的数据透明传输

测试方法和情景 1 中的测试方法一致。

7 附录 1. 恢复出厂设置

注意，该功能仅适用于 01040150 之后的版本。之前版本的模块修改之后，用户自己无法将模块恢复出厂参数。

恢复出厂设置的机制如下



在实际操作中，我们提供了下面两个方法

1、 使用“EM380 配置工具”软件的回复出厂参数的功能。

2、 调用 EMSP 协议 API 函数 vs8 EM380C_InitWithDefault(void)。

3、 用串口连接模块和 PC，打开 windows xp“超级终端”软件，以 115200 8/n/1 参数打开串口，按住空格键不放，然后将模块重新上电，直至模块上的绿灯亮，才松开空格键。这时，模块已经恢复到出厂参数了。

8 附录 2. 命令列表

- EMSP_CMD_RESET：此命令用来对模块进行复位操作。
- EMSP_CMD_GET_CONFIG：此命令用于获取模块中的配置信息。
- EMSP_CMD_SET_CONFIG：此命令用于设置模块的配置信息。
- EMSP_CMD_SCAN_AP：此命令用于获取模块可识别范围内的 AP。
- EMSP_CMD_START：此命令用于启动模块，即使用当前配置信息运行模块。
- EMSP_CMD_SEND_DATA：此命令用于通过模块向网络上发送数据。
- EMSP_CMD_RECV_DATA：此命令用于通过模块从网络上接收数据。
- EMSP_CMD_GET_STATUS：此命令用于获取模块当前工作状态。
- EMSP_CMD_GET_VER：此命令用于获取模块版本号，包括硬件版本和固件版本。
- EMSP_CMD_GET_RF_POWER：此命令用于获取模块无线收发功率。
- EMSP_CMD_SET_RF_POWER：此命令用于设置模块无线收发功率。

9 附录 3. 模块上的产品标签

- 产品标签一般位于模块的背面。样例如下图 7.1.

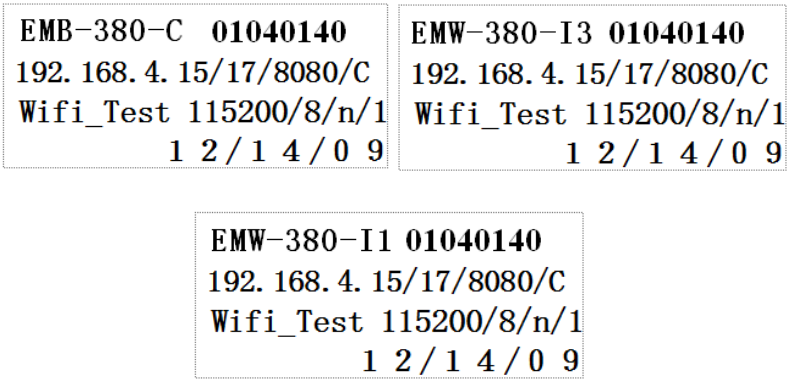


图 9.1 EMB-380-C/EMW-380-I1/EMW-380-I3 产品标签样品图

含义如下：

- EMB-380-C/EMW-380-I1/EMW-380-I2：产品型号。
- 01040140：固件版本，应该和通过 EMSP 命令读出的版本号一致，同时使用的 PC 配置软件的版本也需要和固件版本匹配，否则可能导致配置失败。
- 192.168.4.15/17/8080 /c：表示模块的 IP 地址是 192.168.4.15，服务器的 IP 地址是 192.168.4.17，通讯端口是 8080，模块处于客户端模式。（出厂设置，用户可以修改）
- Wifi_Test：默认连接的无线网络 SSID 号。（出厂设置，用户可以修改）
- 115200/8/n/1：串口配置参数。（出厂设置，用户可以修改）
- 12/14/10：出厂日期。

销售信息

如果需要购买本产品，请在办公时间（星期一至五上午 9:00~12：00；下午 1:00~6:00；）拨打电话咨询上海沁科信息技术有限公司。

联系电话：+86-21-52655026/52655025

联系地址：上海市普陀区千阳路 271 弄 9 号楼 2 楼

邮 编：200333

Email： sales@mxchip.com

技术支持

购买工业无线网络产品后，如果需要获得本产品的最新信息或者我公司其他产品信息，

你可以访问我们的网站：

http://www.mxchip.com/

如果需要电话技术支持，请在办公时间拨打电话：

ST ARM 技术支持

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

无线网络 技术支持

+86 (021)58655026-812 Email：support@mxchip.com

开发工具 技术支持

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

如果需要其他产品的网络技术支持，您可以访问：

http://www.mxchip.com