深圳四博智联科技有限公司

WiFiMCU 新手入门及应用开发案例



二〇一五年十二月

doit.am

目 录

一 WiFiMCU 简介 2
1.1 WiFiMCU 开源项目 2
1.2 WiFiMCU 硬件资源 3
二 上手入门 5
2.1 安装 USB 驱动 5
2.2 快速开始 7
2.3 使用串口调试工具交互 15
三 进阶开发 18
3.1 LED 闪烁 18
3.2 呼吸灯 19
3.3 SOCKET 编程 20
3.4 串口透传 22
四 升级固件 24
4.1 获取最新固件 24
4.2 WiFiMCU STUDIO 升级固件 24
4.3 使用 SecureCRT 升级固件 28
4.4 使用 SWD 工具升级固件 31
附录 资源链接 31

www.doit.am

一 WiFiMCU 简介

众所周知，智能硬件以及相关的创客类产品，越来越多需要通过WiFi模块连接互联网。从2013年至今，整合嵌入式处理器和无线的单芯SOC等产品和方案全线开花。正是在此背景下WiFiMCU应运而生，WiFiMCU以单芯片WiFi为载体，创造性的将Lua解释器移植到嵌入式模块中，为开发者提供一个极其便捷的物联网应用开发平台。WiFiMCU开发初衷是为了帮助广大创客、电子爱好者、学生、IT技术人员以及其他对物联网感兴趣的人士快速实现将设备连接到互联网，完成自己的创意，将有限的精力放在创新本身，而不用花太多时间研究嵌入式技术以及协议实现细节。

1.1 WiFiMCU 开源项目

WiFiMCU由深圳四博智联科技有限公司发起，基于EMW3165模块,以MICO操作系统为基础开发而成。内置Lua解释器，可直接运行脚本代码，操作硬件资源，实现智能硬件创客类产品快速原型开发。

- 比Arduino更简单的交互：通过UART/Telnet交互，屏蔽MCU寄存器设置，脚本操作硬件，无需编译直接运行；
- 硬件可靠，接口丰富：STM32 MCU品质，可外接多种类型传感器、显示器和执行器；
- 基于全实时操作系统MICO，保证网络底层数据交换的实时和稳定；
- 用Lua编写IOT应用：30秒实现WiFi模块连接无线路由器，发送数据到远程服务器；提供丰富的WiFi和网络操作函数，实现复杂网络应用；
- 开源，交互式，可编程，低成本，简单，智能，WiFi硬件；
- 提供硬件的高级接口，可以将应用开发者从繁复的硬件配置、寄存器操作中解放出来。用交互式Lua脚本，像arduino一样编写硬件代码！
- 事件驱动型API极大的方便了用户进行网络应用开发，使用类似Nodejs的方式编写代码，并运行于5mm*5mm大小的MCU之上，加快您的物联网开发进度；

WiFiMCU开发板可用于快速原型开发，为创客开发产品提供了性价比最高的物联网应用开发平台。

www.doit.am

1.2 WiFiMCU 硬件资源

WiFiMCU开发板如下图所示。



图1.1 WiFiMCU开发板照片

WiFiMCU的软硬件特点如下：

- 包含EMW3165模块几乎所有硬件功能；
- 基于上海庆科的互联网实时操作系统MICO开发；
- 18路GPIO可复用，两个UART；
- Micro USB接口，集成CP2102可直接供电和调试；
- SWD调试接口；
- 基于Lua 5.1.4和elua 0.9编写，包含基础模块package、string、table、math等；
- 内置模块：mcu、gpio、timer、wifi、net、file、pwm、uart、adc、spi、i2c、bit等；
- 提供interger和float两个版本的固件以及源代码；
- 用户可用剩余RAM大于48kb字节。
- 用户代码可以的Flash存储空间为1M+256K字节

WiFiMCU开发板的GPIO对应关系见下表。

www.doit.am

表1.1 WiFiMCU开发板管脚

GPIO索引	复用功能	说明
D0	GPIO/BOOT	WiFiMCU模块在上电时，检测D0，若为低电平，模块进入Bootloader模式。
D1	GPIO/PWM/ADC	
D2	GPIO	
D3	GPIO/PWM	
D4	GPIO	
D5	GPIO	SWD调试模式，SWCLK管脚
D6	GPIO	SWD调试模式，SWDIO管脚
D7	GPIO	
D8	GPIO/PWM	串口Uart1，RX1管脚
D9	GPIO/PWM	串口Uart1，TX1管脚
D10	GPIO/PWM	I2C接口SCL
D11	GPIO/PWM	I2C接口SDA
D12	GPIO/PWM	
D13	GPIO/PWM/ADC	
D14	GPIO/PWM	
D15	GPIO/PWM ADC	
D16	GPIO/PWM/ADC	
D17	GPIO/ADC	板载LED连接到此管脚

WiFiMCU的功能在不断增加，目前仍然在开发中。WiFiMCU地址：http://www.wifimcu.com；WiFiMCU开源地址：https://github.com/SmartArduino/WiFiMCU。

www.doit.am

二 上手入门

2.1 安装 USB 驱动

WiFiMCU 开发板使用 CP2102 作为 TTL 转 USB 接口芯片，USB 接口为 Micro USB。USB 接口一方面为开发板供电，另一方面作为 USB 模拟串口提供调试接口。

CP2102 芯片是 silabs 公司产品，其 USB 驱动可在 silabs 公司官网下载：

http://www.silabs.com/products/mcu/Pages/USBtoUARTBridgeVCPDrivers.aspx

下面以 Windows 7 操作系统下安装 USB 驱动为例进行说明。

第一步，打开下载的软件包，根据系统类型选择安装文件。32 位系统选择“CP210xVCPInstaller_x86.exe”，64 位系统选择“CP210xVCPInstaller_x64.exe”。根据软件提示安装即可。

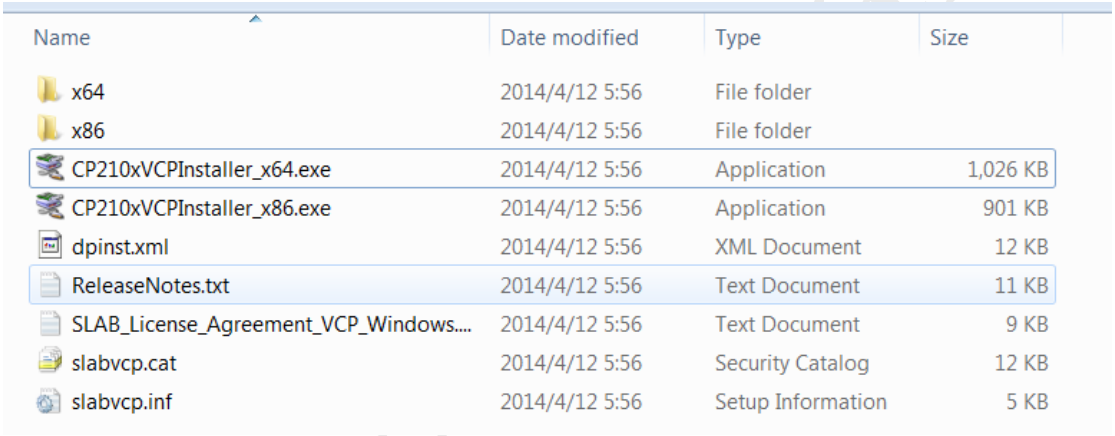
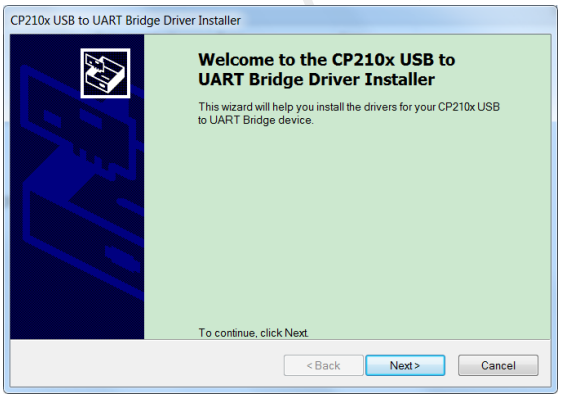
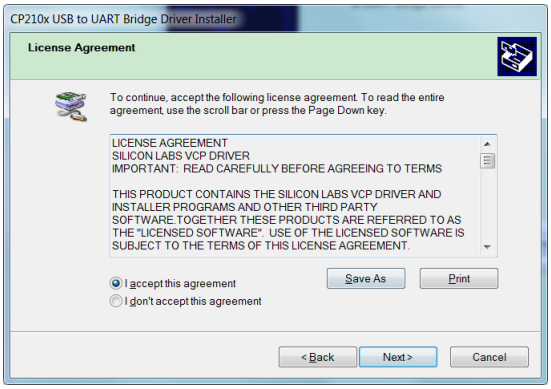


图 2.1 USB 驱动文件列表





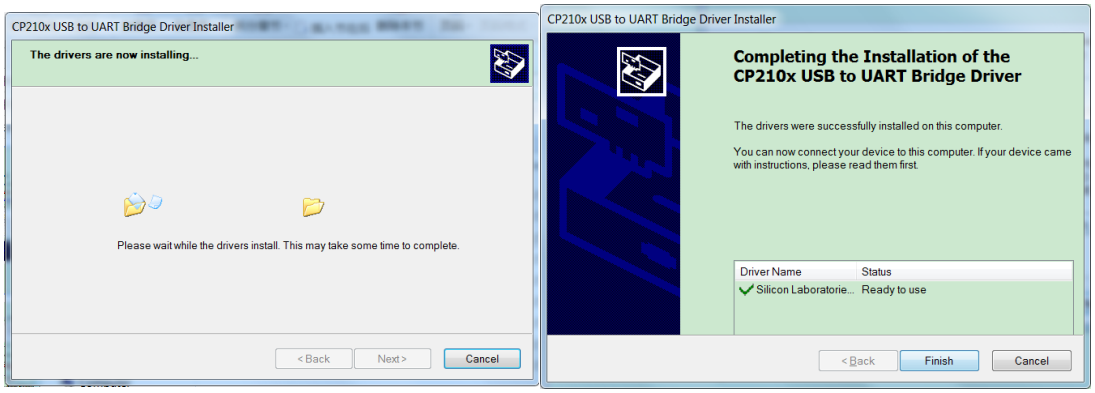


图 2.2 USB 驱动安装过程

第二步，安装完成后，使用 Micro USB 线连接开发板和电脑，在电脑的设备管理器中查看串口号。电脑设备管理器可通过“开始”->“控制面板”->“硬件和声音”->“设备管理器”进入。如下图所示，在“端口（COM 和 LPT）”中显示“Silicon Labs CP210x USB to UART Bridge”字样，表示安装成功。

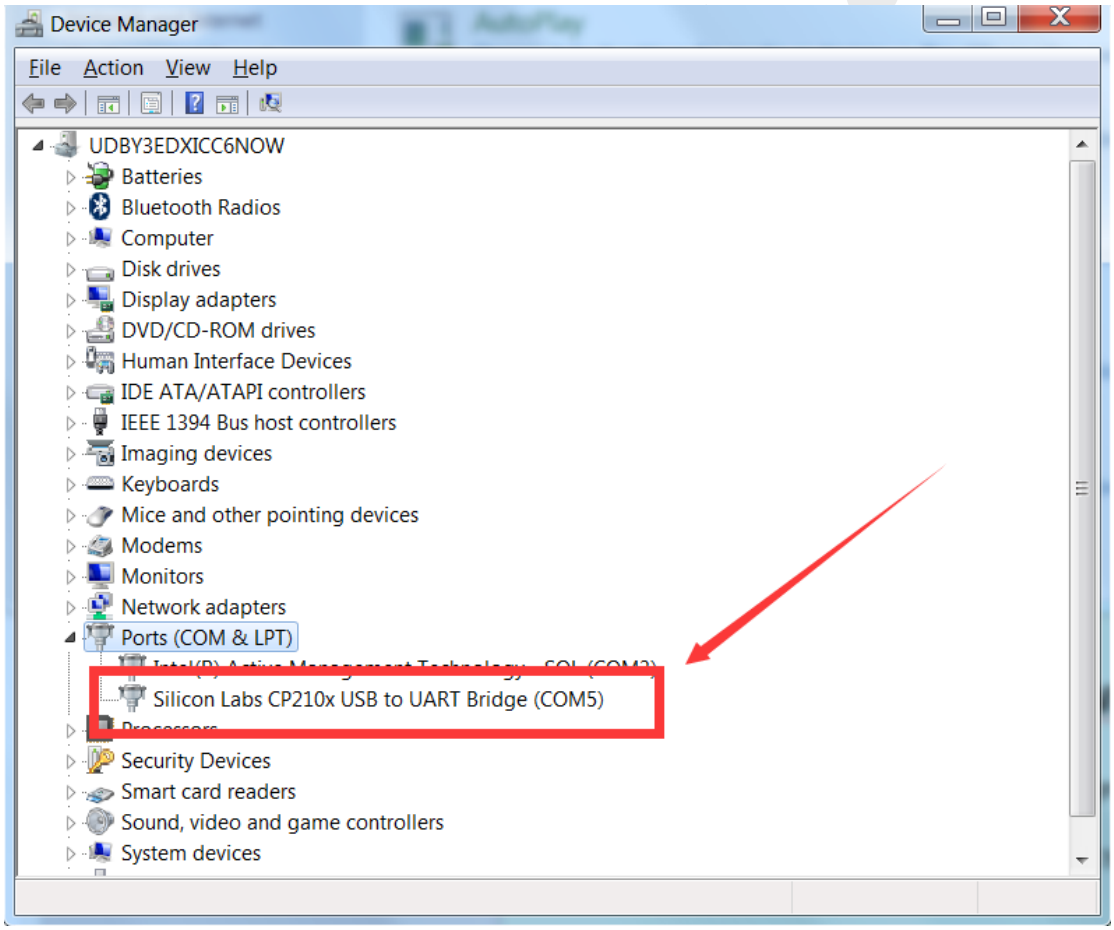


图 2.3 USB 启动安装完成

2.2 快速开始

WiFiMCU 通过串口交互，执行脚本代码，也可以通过串口下载 Lua 脚本文件，执行复杂功能。不同平台下的串口工具均可用于 WiFiMCU 开发板的调试。在 Windows 平台下，WiFiMCU 提供了 WiFiMCU STUDIO 快捷开发调试工具。使用 WiFiMCU STUDIO，开发者可快速熟悉 WiFiMCU 各模块基本功能，上传 Lua 文件到开发板并执行，更新 WiFiMCU 固件等。

WiFiMCU STUDIO 工具的下载地址：

https://github.com/SmartArduino/WiFiMCU-STUDIO/releases

WiFiMCU STUDIO 的源代码地址：

https://github.com/SmartArduino/WiFiMCU-STUDIO

2.2.1 上电

WiFiMCU 开发板使用 Micro USB 供电，如下图所示。“Reset”按钮用于直接模块复位。“Boot”按钮用于在启动模块时，进入到 Bootloader 模式。在正常模式下，Boot 按钮可作为普通输入按键。

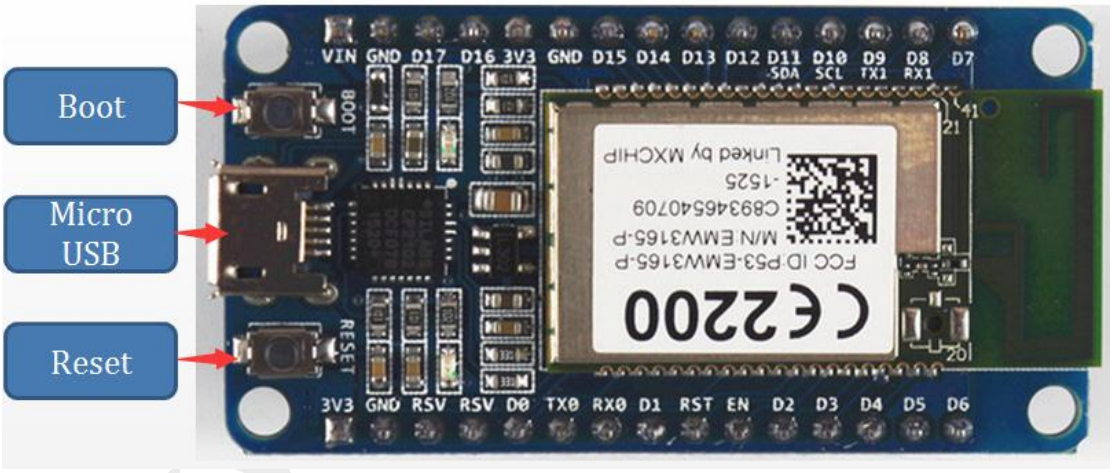


图 2.4 WiFiMCU 供电接口

2.2.2 选择串口

在电脑的设备管理器中查看 WiFiMCU 开发板对应的串口号并记下该串口号。

2.2.3 运行 WiFiMCU STUDIO

WiFiMCU STUDIO 是免安装的，可直接运行。双击 WiFiMCU STUDIO 图标运行软件。



图 2.5 WiFiMCU STUDIO 软件界面

在软件左上角“Scan Port”按钮可扫描当前电脑可用的串口。选择 WiFiMCU 开发板对应的串口号，波特率使用 115200bps。然后点击“Open”打开串口。如果成功打开，在 WiFiMCU STUDIO 的右侧串口交互框中将会出现输入提示符“>”。

在“串口交互区”输入命令“mcu.reboot()”，重启模块。效果如下：

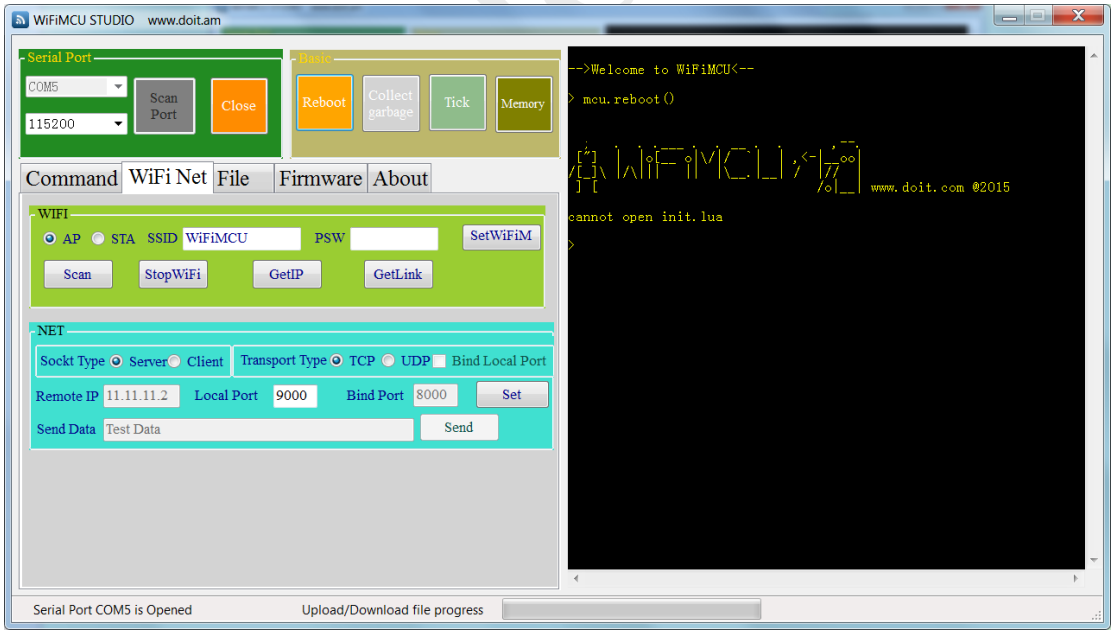


图 2.6 重启 WiFiMCU 开发板

使用“常用命令”中的“Reboot”按钮，可以实现同样的功能。

其他常用命令：

“collectgarbage()”主动回收内存碎片。WiFiMCU 内置的 Lua 解释器具备内存碎片自动回收功能。开发者可在程序中主动使用 collectgarbage()显示回收，提高内存利用率。

“Tick”等同于命令“=mcu.tick()”，打印 WiFiMCU 模块自启动以来的时间。单位为毫秒。

“Memory”等同于命令“mcu.mem()”，打印当前内存信息。
其他指令的具体参数和用法可参考 WiFiMCU 的函数手册：
https://github.com/SmartArduino/WiFiMCU/tree/master/Document/

2.2.4 LED 测试

在 WiFiMCU 开发板上，有一个 LED 连接到 D17，通过 GPIO 模块可直接控制该 LED 的亮、灭。

步骤一，在 WiFiMCU STUDIO 中，切换到“Command”选项卡。
步骤二，在 GPIO 组下选择 Pin 为 17，Mode 选为“gpio.OUTPUT”。

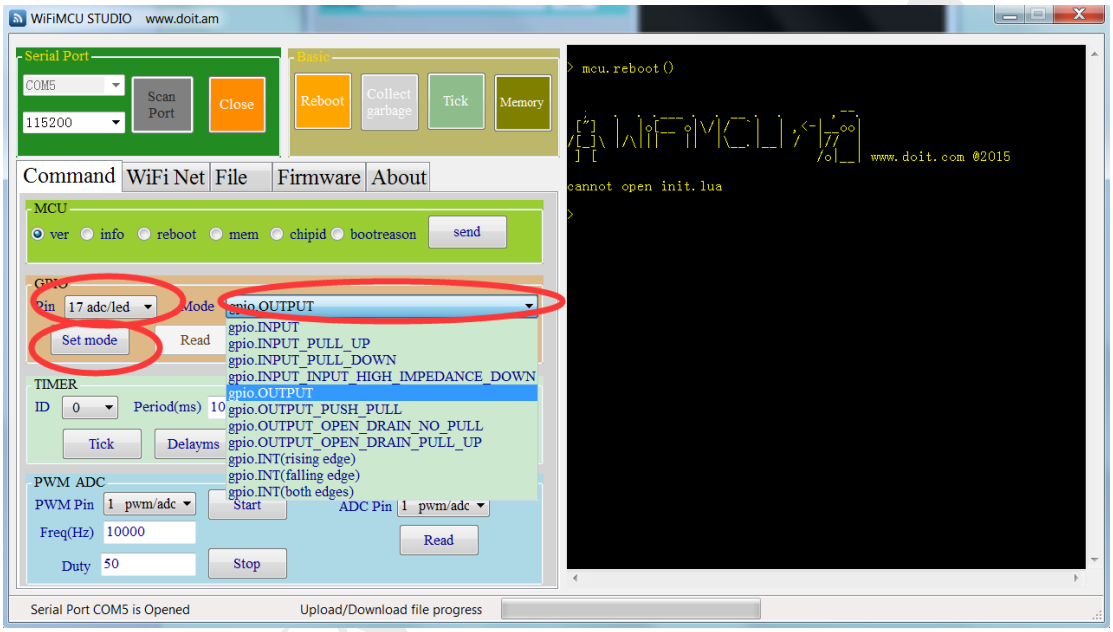


图 2.7 设置 GPIO17 为输出模式

步骤三，点击“Set mode”，。然后点击“Toggle”按钮，LED 将会根据当前状态，被点亮或熄灭。点击“Set mode”等同于通过串口向 WiFiMCU 开发板发送字符串命令“gpio.mode(17,gpio.OUTPUT)”。点击“Toggle”等同于发送字符串命令“gpio.toggle(17)”。

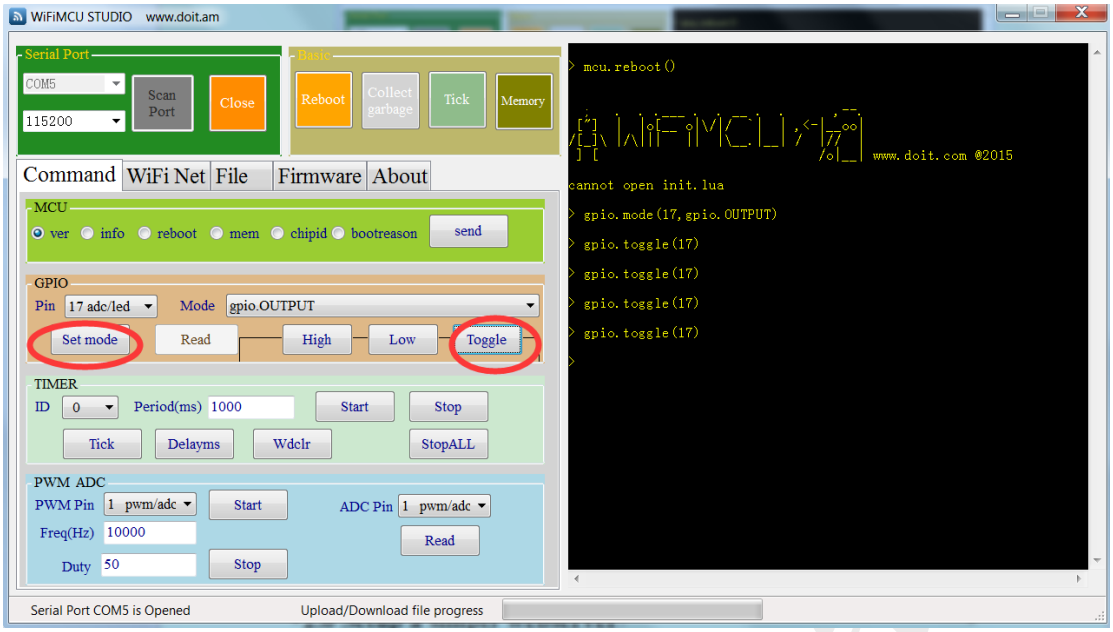


图 2.8 点击 Toggle 按钮



图 2.9 LED 测试照片（左为 LED 亮，右为 LED 灭）

2.2.5 开启热点模式

WiFiMCU 内置的 WiFi 芯片可工作在热点模式（Access Point，AP），接入点模式（Station，STA）或者 AP+STA 共存模式。下面以开启热点模式为例介绍。

步骤一，WiFiMCU STUDIO 中切换到“WiFi Net”选项卡。

步骤二，如下图，选择“AP”，在“SSID”输入框填写 AP 的广播名称，例如“WiFiMCU_Wireless”,“PSW”输入框中填写密码，若留空，表示不需要密码。然后点击“Set”设置。

上述操作等同于在串口输入如下指令：

```
1. cfg={}
2. cfg.ssid='WiFiMCU_Wireless'
3. cfg.pwd=''
4. wifi.startap(cfg)
5. cfg=nil
```

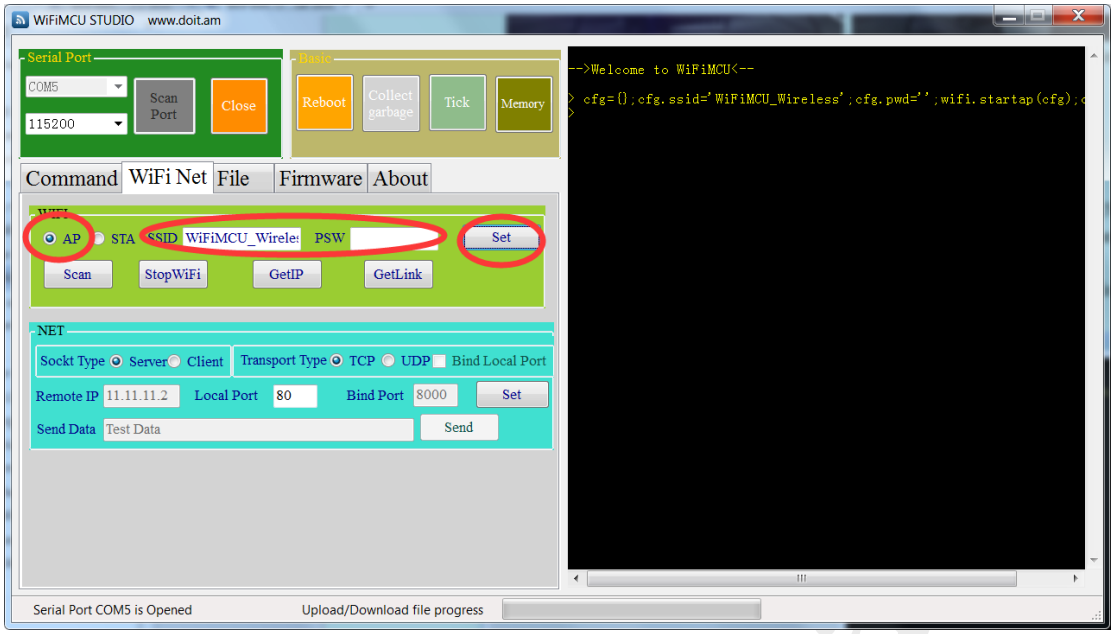


图 2.10 设置 AP

步骤三，名为“WiFiMCU_Wireless”的 AP 热点将会建立。



图 2.11 “WiFiMCU_Wireless”热点

步骤 4，利用手机或者笔记本电脑连接到“WiFiMCU_Wireless”，如果没有设置密码，可直连。通过电脑可查看 IP 地址分配情况。默认情况下，WiFiMCU 开发板的 IP 地址为 11.11.11.1，开发者可通过 Lua 内置的 wifi 功能模块设置自己需要的 ip 地址。

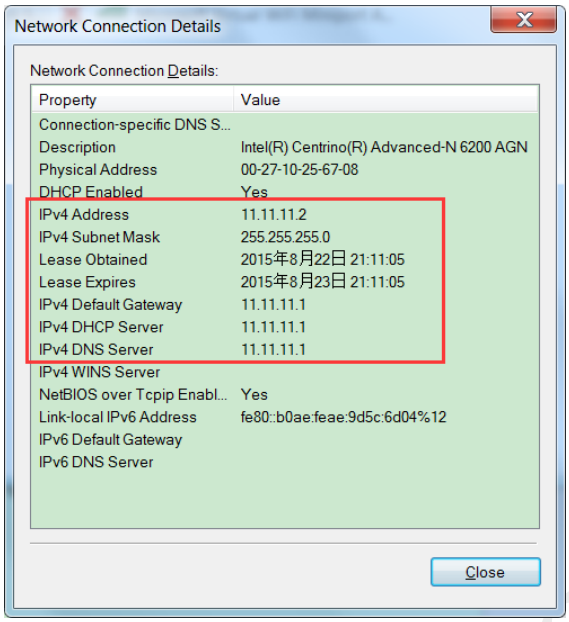


图 2.12 查看 IP

2.2.6 建立简单的 Web 服务器

利用 WiFiMCU 内置的 WiFi 和 Net 模块，可以快速建立 TCP/UDP 服务器和客户端。本例介绍了一个利用 WiFiMCU 建立最简单的 Web 服务器过程。

步骤 1，确保 WiFiMCU 开发板运行在 AP 模式，详细方法可参考上一节“开启热点模式”。

步骤 2，将下列代码复制到一个文本文件中，并保存为文件：“webserver.lua”。

```
1.  skt = net.new(net.TCP,net.SERVER)
2.  net.on(skt,"accept",function(clt,ip,port)
3.  print("accept ip:"..ip.." port:"..port.." clt:"..clt)
4.  net.send(clt,[[HTTP/1.1 200 OK
5.  Server: WiFiMCU
6.  Content-Type:text/html
7.  Content-Length: 28
8.  Connection: close
9.
10.
11. <h1>Welcome to WiFiMCU!</h1>]])
12. end)
13. net.start(skt,80)
```

上述代码中，第 1 行是创建了一个 TCP SERVER 的 socket。

第 2 行注册了这个 tcp server 的“accept”事件回调函数，当有 tcp client 连接到此 socket 的时候，将会调用“accept”事件回调函数。

第 4 行是在“accept”回调函数中，向指定的 clt 发送 http 文本，该文本包含了 http 头和页面文字。

第 11 行在 80 端口启动 tcp server。

步骤 3，在 WiFiMCU STUDIO 中，切换到“File”选项卡。

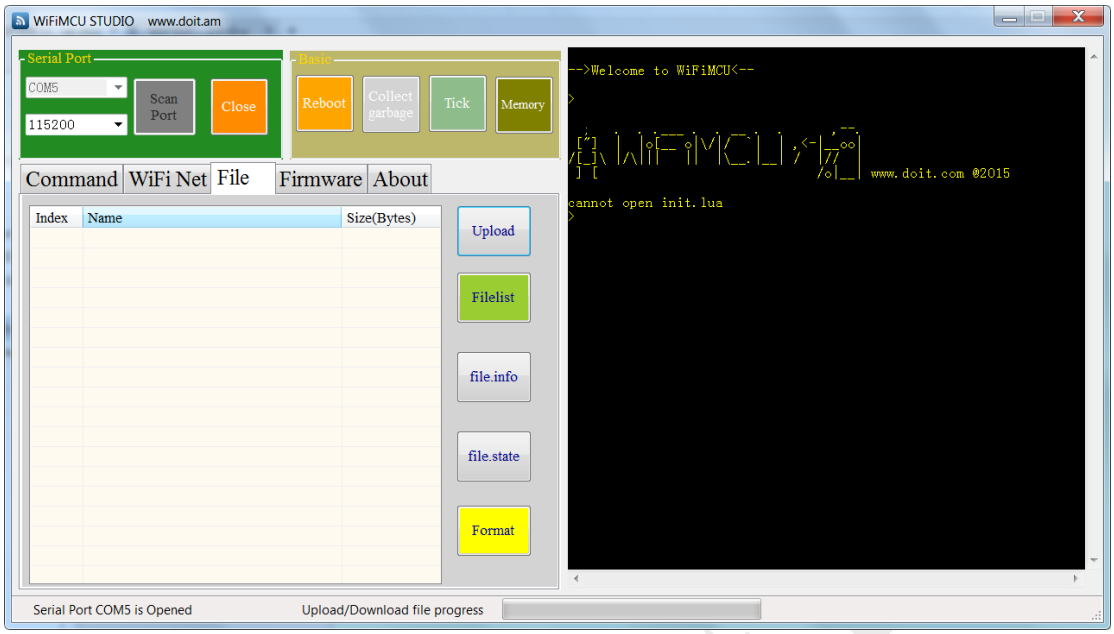


图 2.13 上传文件

步骤 4，点击“Upload”按钮，选择“webserver.lua”文件，上传文件。

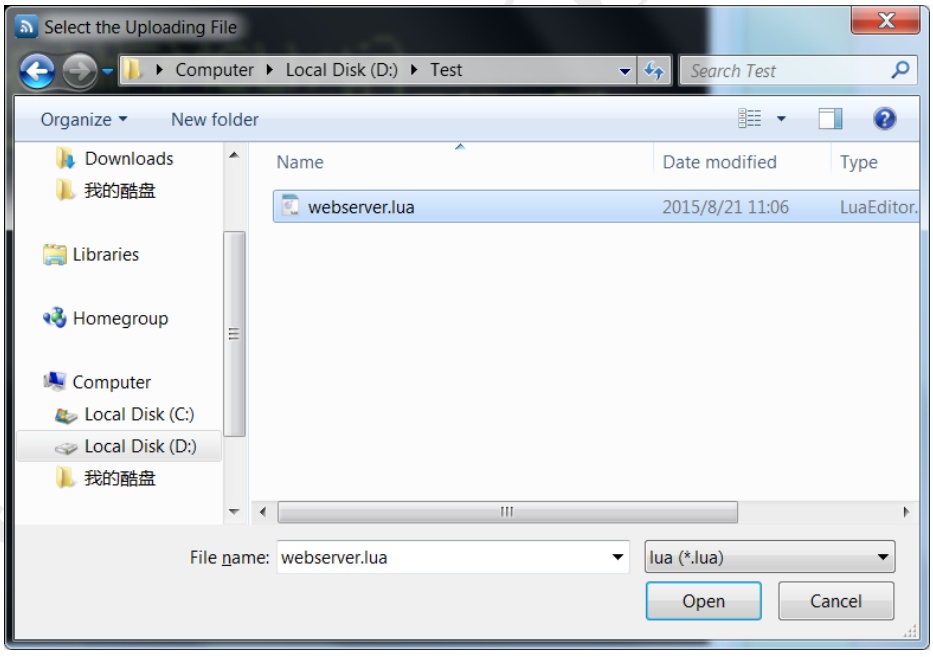


图 2.14 选择“webserver.lua”

步骤 5，等待上传完成。上传完成后，WiFiMCU STUDIO 的文件列表框自动更新文件列表。

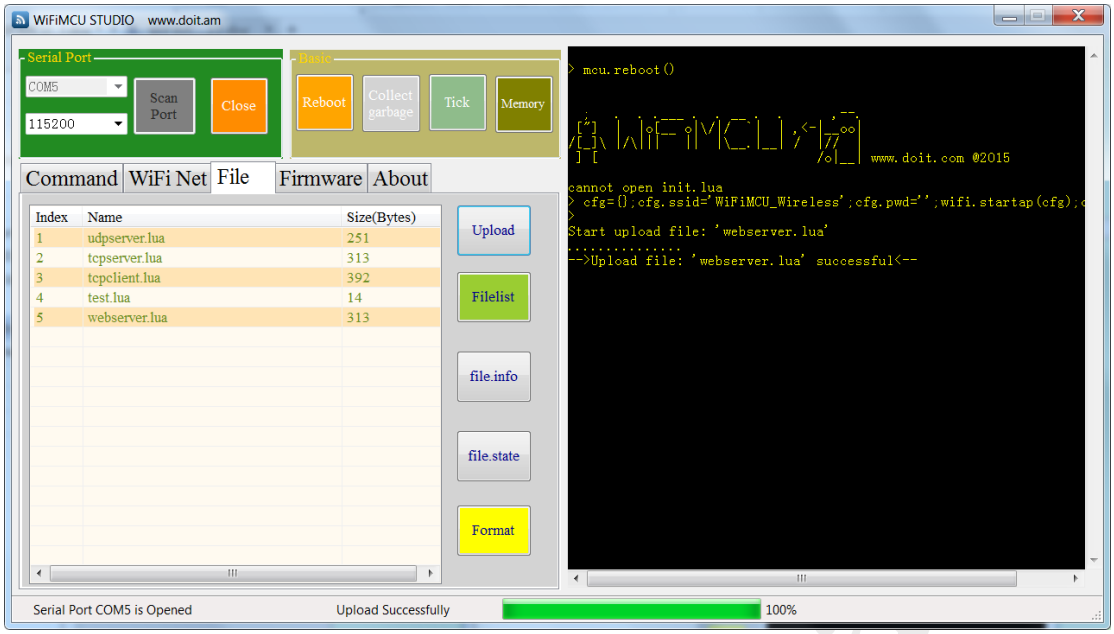


图 2.15 上传成功

步骤 6，在列表框中，选择“webserver.lua”，点击鼠标右键，出现快捷菜单。选择“Run”在 WiFiMCU 中运行该文件。等同于使用指令：“dofile('webserver.lua')”。

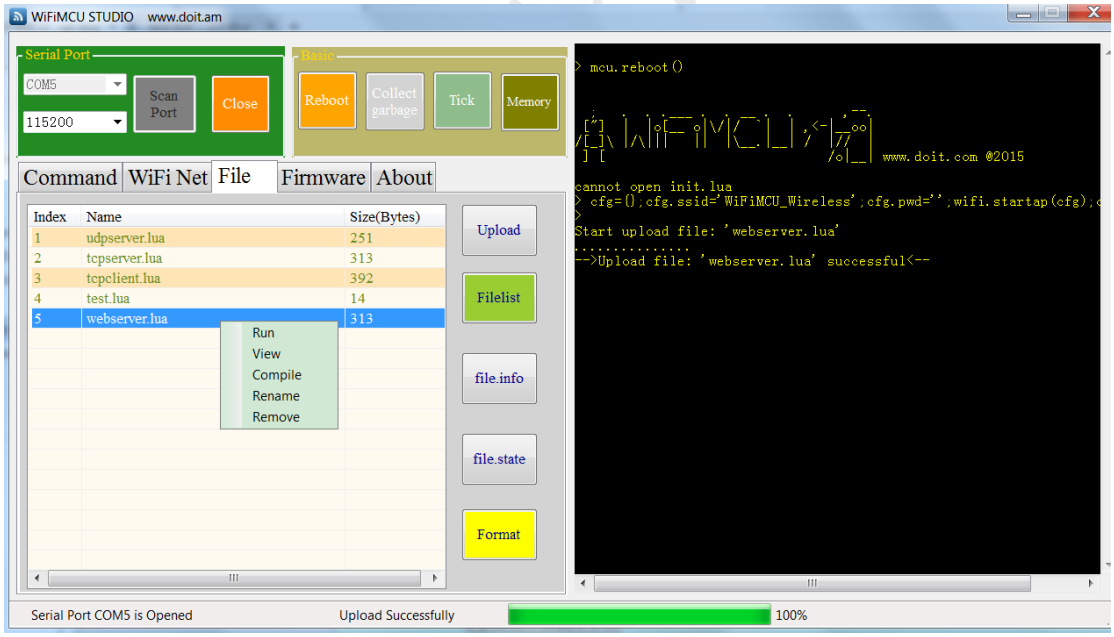


图 2.16 运行文件

步骤 6 使用智能手机或者笔记本连接到 WiFiMCU 开发板的 AP，然后在浏览器中输入地址：“11.11.11.1”，可打开页面。

最简单的 Web 页面完成。

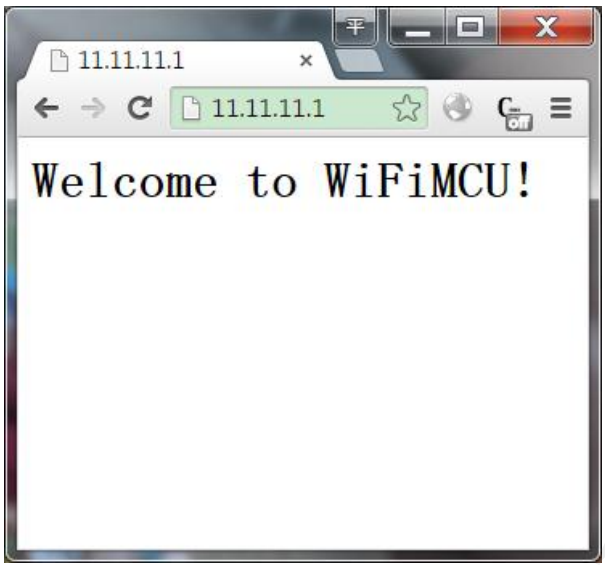
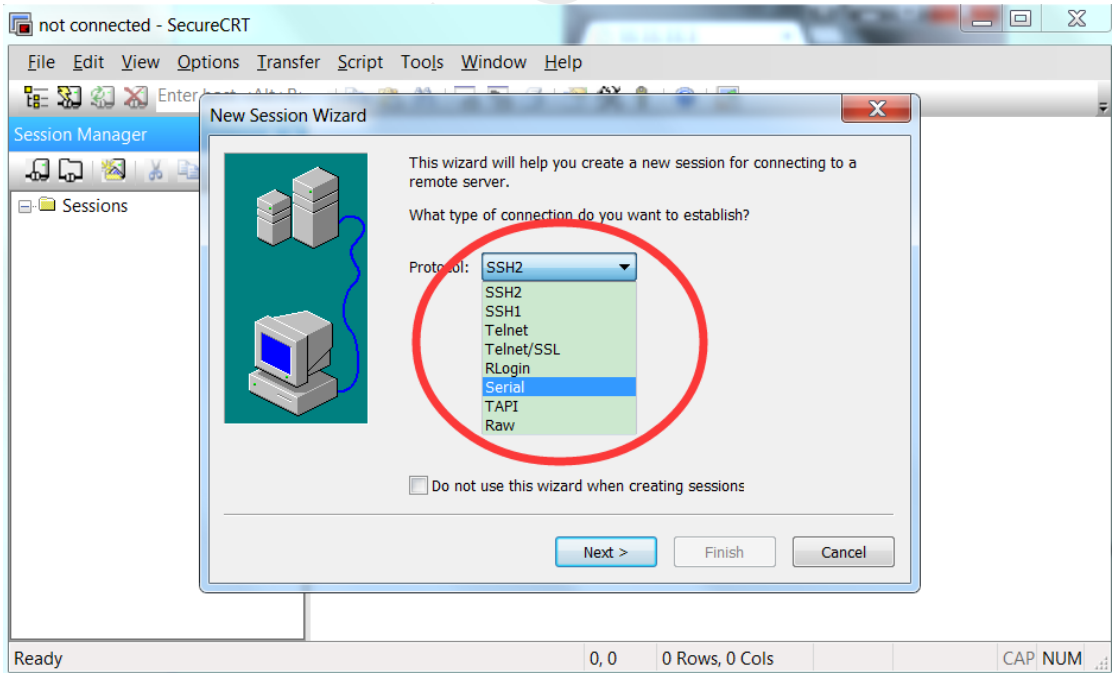


图 2.17 Web 页面

2.3 使用串口调试工具交互

WiFiMCU 通过串口交互，开发者可以使用任何平台下的串口工具进行调试。在此以 Windows 操作系统中的 SecureCRT 为例说明。

步骤 1，启动“SecureCRT”软件，新建一个会话。串口参数设置为 115200，8，n，1。选择 WiFiMCU 开发板对应的串口。



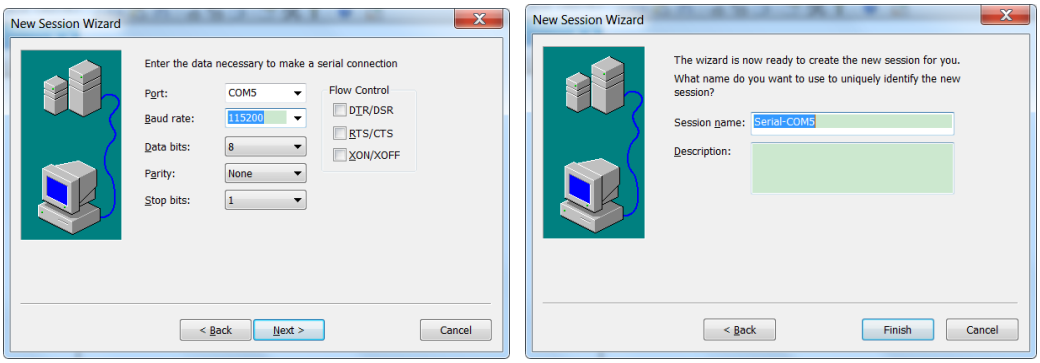


图 2.18 新建一个会话

步骤 2，打开会话。使用 WiFiMCU 开发板上的 Reset 重启模块，可以看到 WiFiMCU 启动信息。

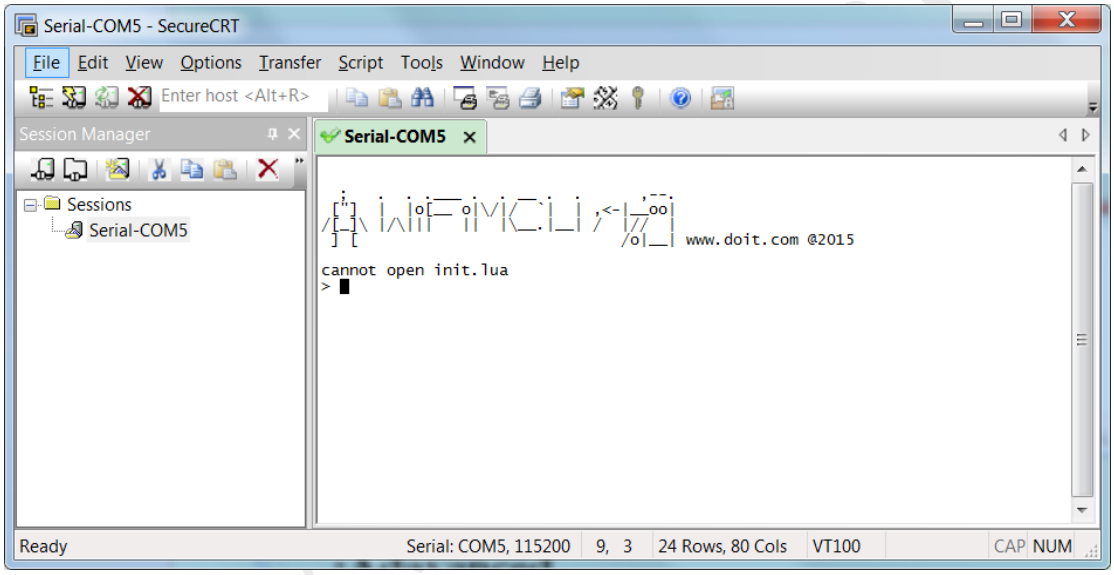


图 2.19 打开会话交互

步骤 3，使用串口指令实现与 WiFiMCU 开发板的交互。

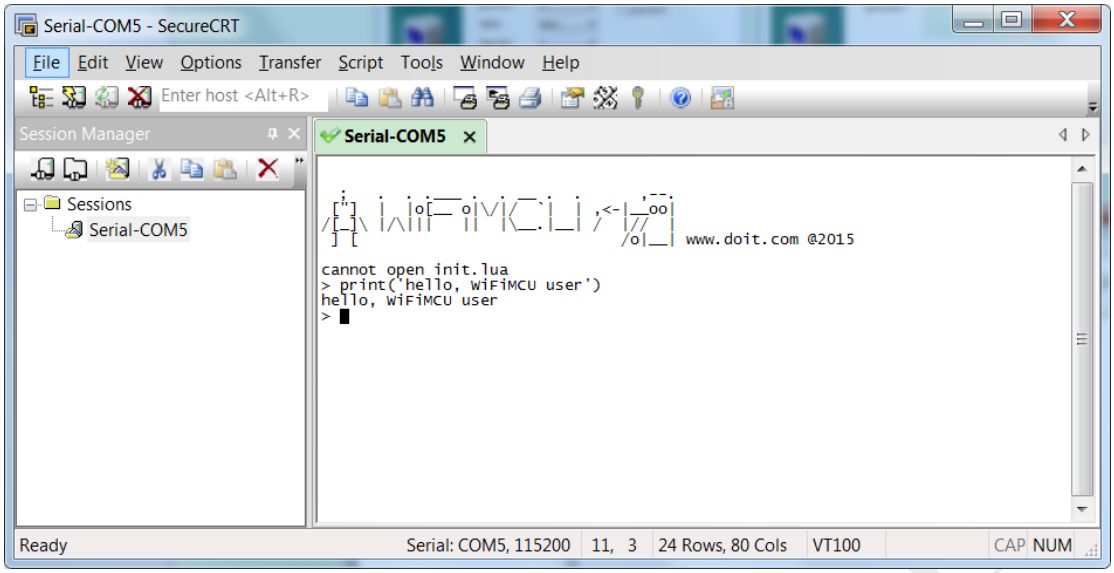


图 2.20 使用指令交互

www.doit.am

三 进阶开发

本节利用四个案例讲解 WiFiMCU 的定时器模块、PWM 模块、Net 模块、串口模块等的使用方法。

3.1 LED 闪烁

本例使用定时器产生 500 毫秒定时中断，在中断中对 WiFiMCU 开发板上的 LED 灯进行取反操作，实现 LED 灯闪烁。

源代码如下：

```
1. print("---WiFiMCU Demo---")
2. print("Flash LED")
3. pin = 17
4. gpio.mode(pin,gpio.OUTPUT)
5. tmr.start(0,500,function()
6.     gpio.toggle(pin)
7. end)
```

程序第 1、2 行向调试串口打印字符串。

第 3、4 行设置开发板上 LED 灯所对应的管脚 D17 为输出模式。

第 5~7 行启动定时器 0，以 500ms 周期调用回调函数。回调函数中，通过“gpio.toggle(pin)”反转 D17。

将上述代码保存为文本文件，文件名为“Flash_LED.lua”。通过 WiFiMCU STUDIO 下载到开发板中运行。运行后可以看到 LED 每隔 0.5 秒闪一次。

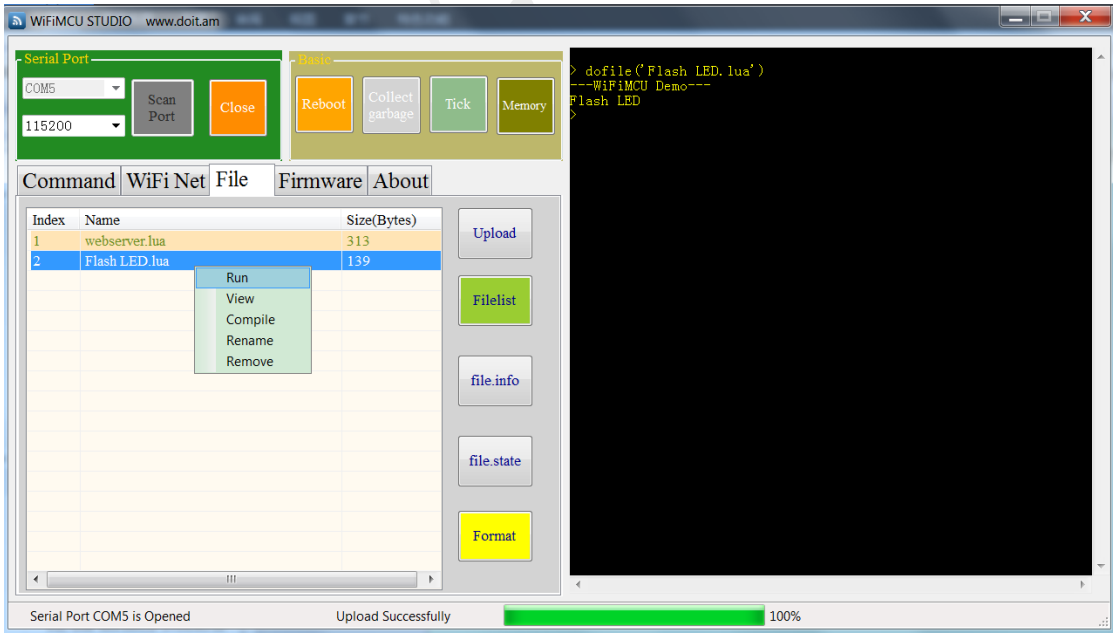


图 3.1 下载到开发板中运行

3.2 呼吸灯

本节利用 WiFiMCU 开发板上支持 PWM 输出的管脚实现 PWM 输出。

源代码如下：

```
1.  print("---WiFiMCU Demo---")
2.  print("Breathing LED")
3.  pin = 1
4.  freq =10000
5.  duty =0
6.  dir = 1
7.  tmr.start(1,50,function()
8.      if dir ==1 then
9.          duty = duty + 5
10.         if duty>100 then
11.             duty =100
12.             dir = 0
13.         end
14.     else
15.         duty = duty - 5
16.         if duty< 0 then
17.             duty = 0
18.             dir = 1
19.         end
20.     end
21.     pwm.start(pin,freq,duty)
22. end)
```

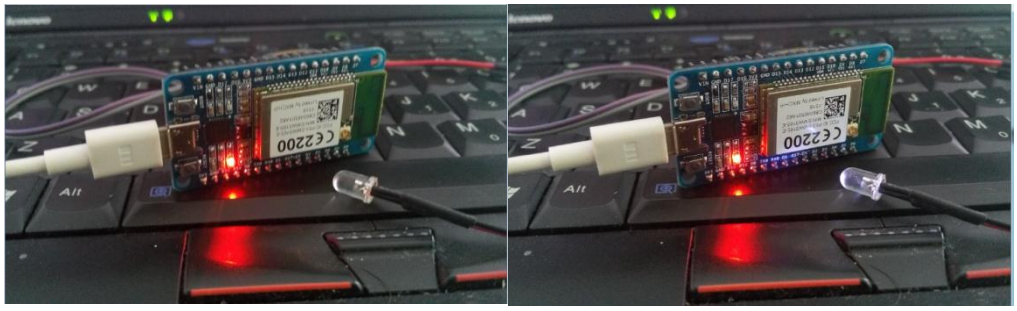
程序第 1、2 行向调试串口打印字符串。

第 3~6 行定义变量，定义 D1 为 PWM 输出口。

第 7~22 行启动定时器 1，以 50ms 周期调用回调函数。回调函数中，循环对占空比“duty”进行计算。

第 21 行是修改 PWM 的占空比，实现占空比调节。

将上述代码保存为文本文件，文件名为“Breathing_LED.lua”。通过 WiFiMCU STUDIO 下载到开发板中。使用一个 LED 连接到 WiFiMCU 开发板，其阳极接 D1，阴极接 GND。运行“Breathing_LED.lua”运行后可以看到 LED 从暗到明，然后从明到暗变化变化。



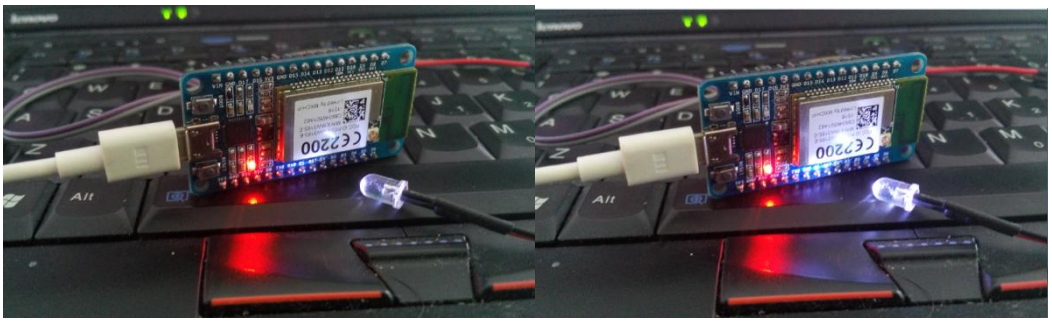


图 3.2 呼吸灯

3.3 SOCKET 编程

本节使 WiFiMCU 开发板工作在 STA 模式，建立 TCP Client 连接，连接到云端服务器。源码如下：

```
1.  print("------net demo------")
2.
3.  --setup a wifi interface first
4.  ap_ssid="Doit" --change to your ap ssid
5.  ap_pwd="123456789" --change to your ap pwd
6.  cfg={ssid = ap_ssid,pwd = ap_pwd}
7.  wifi.startsta(cfg)
8.  cfg=nil
9.  ap_pwd=nil
10.
11. --check if connect to ap
12. stat,_,_,_ = wifi.sta.getlink()
13. while stat ~= 'connected'   do
14.     print("Wating for connecting to AP:"..ap_ssid)
15.     tmr.delayms(1000)
16.     stat,_,_,_ = wifi.sta.getlink()
17. end
18. print("WiFiMCU connected to ap successful")
19.
20. print("tcp client started")
21. skt = net.new(net.TCP,net.CLIENT)
22. net.on(skt,"connect",function(skt) print("connect: skt:"..skt) end)
23. net.on(skt,"dnsfound",function(skt,ip) print("dnsfound:
    skt:"..skt.." ip:"..ip) end)
24. net.on(skt,"sent",function(skt) print("sent:skt:"..skt) end)
25. net.on(skt,"disconnect",function(skt) print("disconnect:skt:"..skt)
    end)
26. net.on(skt,"receive",function(skt,d)
27.     print("Receive:"..d)
```

```
28.     net.send(skt,'From Client:'..d.."\\n")
29. end)
30. --connect to server(ip:115.29.109.104) at port 6520
31. net.start(skt,6520,"115.29.109.104")
```

程序第 4~9 行设置模块工作在 sta 模式，并连接到无线路由器，无线路由器的 ssid 为“Doit”，密码为“123456789”。开发者可根据自己的无线路由器设置情况，响应修改。

程序第 12~17 行使用 repeat 循环检测 WiFiMCU 开发板是否连接到指定的无线路由器，通过 wifi.sta.getlink()获取连接状态。

程序第 21 行，建立 TCP Client socket。

程序第 22~26 行，分别注册该 socket 的“connect”、“sent”，“disconnect”、“receive”等事件的回调函数。在“recieve”事件回调函数中将接收到的数据返回给发送方。

程序第 31 行，连接到远程服务器“115.29.109.104”，端口为 6520。该服务器为免费服务器，开发者可通过在浏览器中访问“tcp.doit.am”获得动态的 ip 和端口。根据页面提示操作。该服务器功能是提供一个公网数据转发功能。例如：两个不同的 tcp client 均连接到该服务器，这两个 tcp client 将会实现透传功能。

将上述代码保存为文本文件，文件名为“tcp client.lua”。通过 WiFiMCU STUDIO 下载到开发板中运行。同时使用一个“TCP&UDP 调试工具”，也建立一个 tcp client 连接，与 WiFiMCU 开发板一样，连接到“115.29.109.104”的端口 6520。此时通过“TCP&UDP 调试工具”发送字符串“Hello, WiFiMCU”，在 WiFiMCU STUDIO 中可以看到开发板收到数据，并向服务器返回“From Client：Hello，WiFiMCU”。表明开发板程序正常工作。

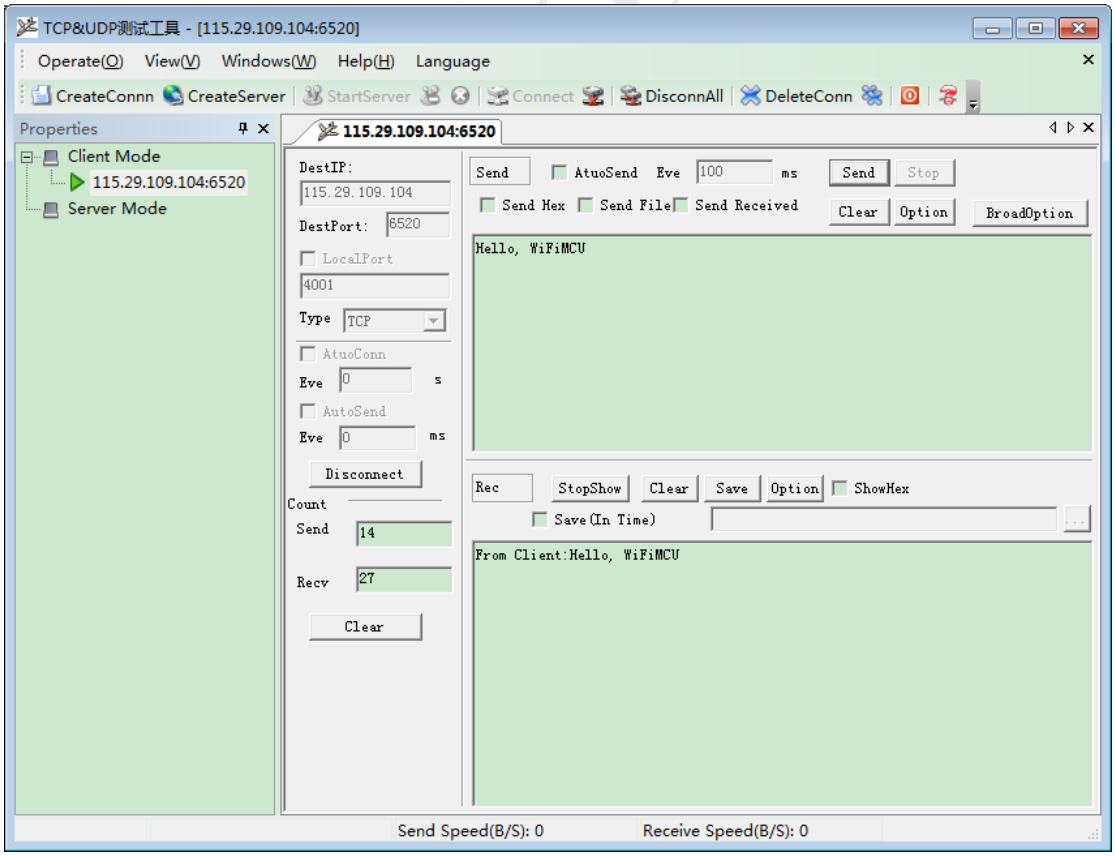


图 3.3 TCP&UDP 调试工具

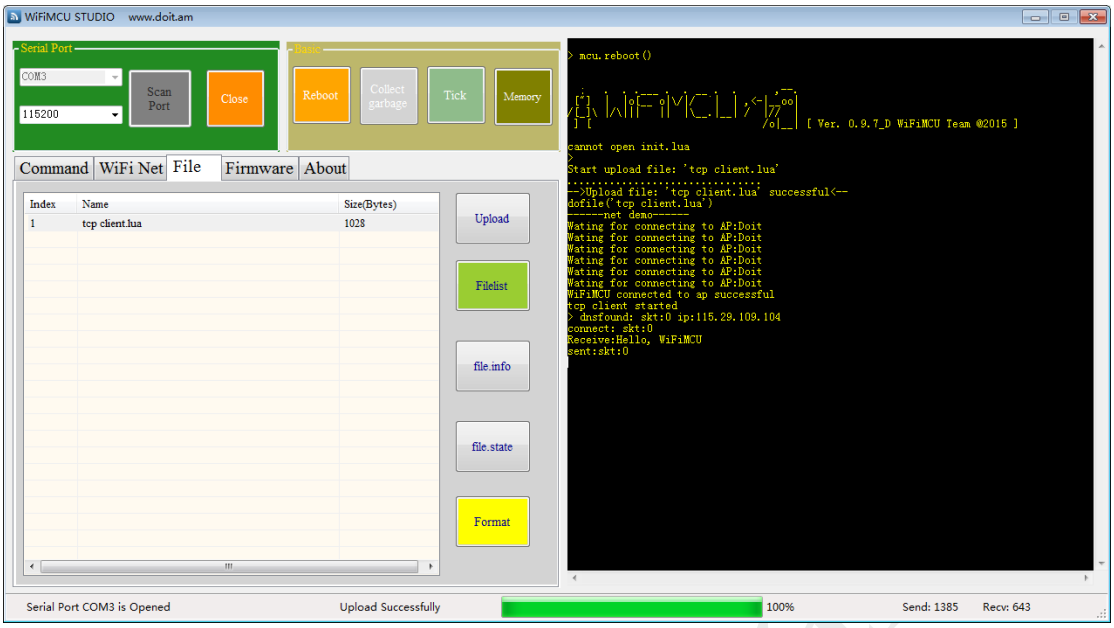


图 3.4 程序执行 log

程序打印信息：

```
1.  ------net demo------
2.  Wating for connecting to AP:Doit
3.  Wating for connecting to AP:Doit
4.  Wating for connecting to AP:Doit
5.  Wating for connecting to AP:Doit
6.  Wating for connecting to AP:Doit
7.  Wating for connecting to AP:Doit
8.  WiFiMCU connected to ap successful
9.  tcp client started
10.
    > dnsfound: skt:0 ip:115.29.109.104
11. connect: skt:0
12. Receive:Hello, WiFiMCU
13. sent:skt:0
```

更多关于 Net 模块使用的代码请参考：
https://github.com/SmartArduino/WiFiMCU/tree/master/Document/demos/5%20net

3.4 串口透传

本节使用 WiFiMCU 开发板打开 AP 热点，建立一个 TCP Server，实现 wifi 转串口功能。
源码如下：

```
1.  --uart demo
2.  print("------uart demo------")
3.  --setup a ap
4.  cfg={ssid = 'WiFiMCU_Wireless',pwd = ''}
5.  wifi.startap(cfg)
```

```
6.  cfg=nil
7.
8.  print("tcp server started")
9.  skt = net.new(net.TCP,net.SERVER)
10. uartClt = nil --save the clt socket
11. net.on(skt,"accept",function(clt,ip,port) uartClt = clt; end)
12. net.on(skt,"disconnect",function(clt) uartClt=nil end)
13. net.on(skt,"receive",function(clt,d)
14.     print("Receive:"..d)
15.     uartClt = clt
16.     --send to uart 1
17.     uart.send(1,d)
18. end)
19.
20. net.start(skt,9000)
21.
22. --uart pin:D8(RX1), D9(TX1)
23. --setup uart
24. uart.setup(1,9600,'n','8','1')
25.
26. function uartReceive(d)
27.     print("len(bytes)"..string.len(d).." data:"..d)
28.     if uartClt ~=nil then
29.         net.send(uartClt,d)
30.     end
31. end
32. uart.on(1, 'data',uartReceive)
```

程序第 4~6 行，建立一个名为“WiFiMCU_Wireless”的热点。密码为空。

第 9~20 行，建立 TCP SERVER，监听 9000 端口。分别注册 accept、disconnect 和 receive 事件。其中，使用变量 uartClt 来记录当前连接的 tcp client。当接收到数据，通过“uart.send(1,d)”发送到串口 1.

第 22~32 行，设置串口 1 的参数为 9600，n，8，1，注册接收回调函数 uartReceive。在收到串口数据后，发送到 uartClt。

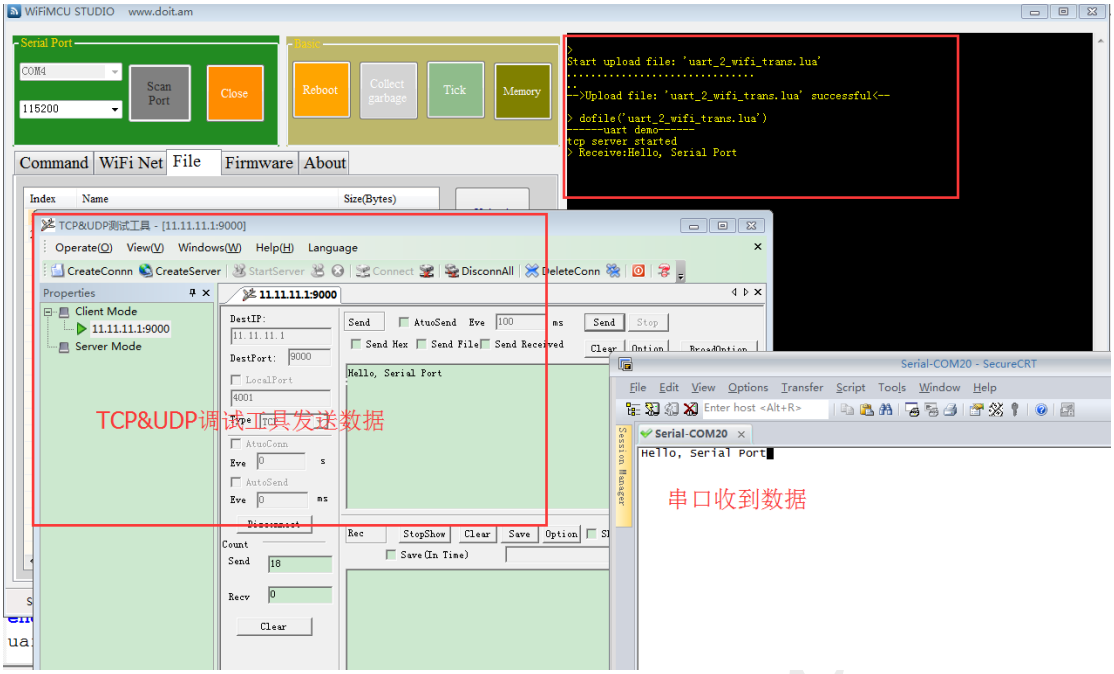


图 3.5 串口透传

更多的示例代码请参考：
https://github.com/SmartArduino/WiFiMCU/tree/master/Document/demos

四 升级固件

WiFiMCU 开发板的固件分为三部分：Firmware、Bootloader 和 Wlan Driver。“Fimware”为 WiFiMCU 包含 Lua 解释器的固件，“Bootloader”为引导固件用于升级，“Wlan Driver”为无线驱动芯片的驱动。WiFiMCU 提供三种升级方式：利用 WiFiMCU STUDIO 升级、使用 SecureCRT 软件升级、使用 SWD Flash 编程工具升级。

4.1 获取最新固件

开发者可以在 WiFiMCU 的开源项目地址下载 Firmware、Bootloader 和 Wlan Driver 三种固件的最新版本。

https://github.com/SmartArduino/WiFiMCU/releases

源代码地址：

https://github.com/SmartArduino/WiFiMCU

4.2 WiFiMCU STUDIO 升级固件

本节说明如何通过 WiFiMCU STUDIO 对固件进行升级。

步骤 1，WiFiMCU 上电，并通过 Micro USB 连接到电脑。运行 WiFiMCU STUDIO。

步骤 2，在 WiFiMCU STUDIO 中切换到“Firmware”页面。

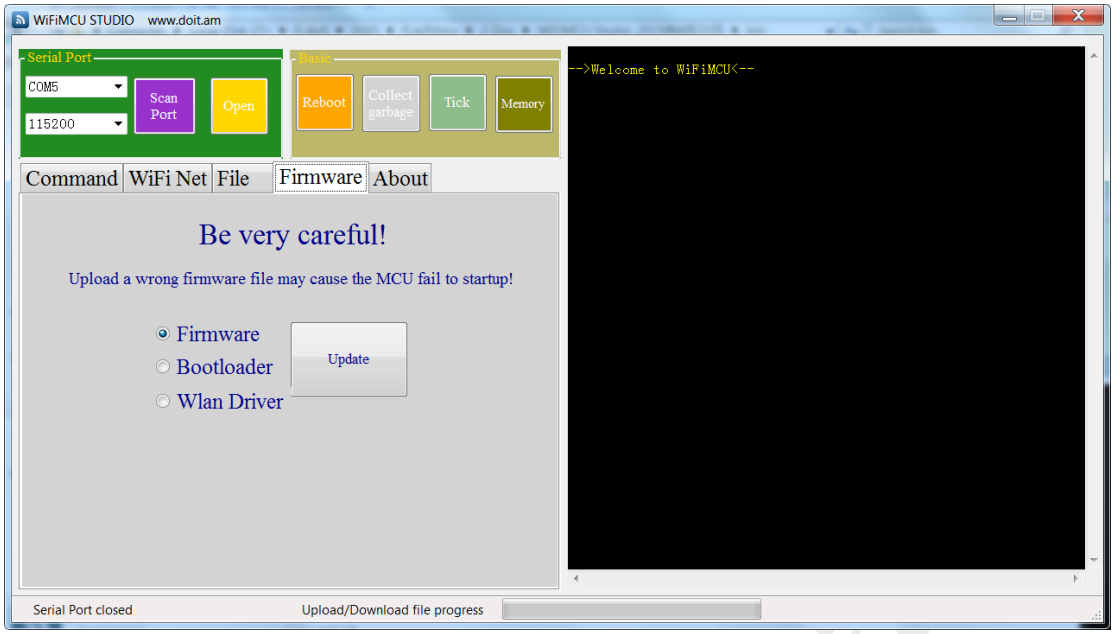


图 3.6 固件升级页面

步骤 3，在 WiFiMCU 开发板上，按住“Boot”按钮，然后按“Reset”，重启 WiFiMCU 开发板，重启后，松开“Boot”按钮。此时开发板进入到 Bootloader 模式。

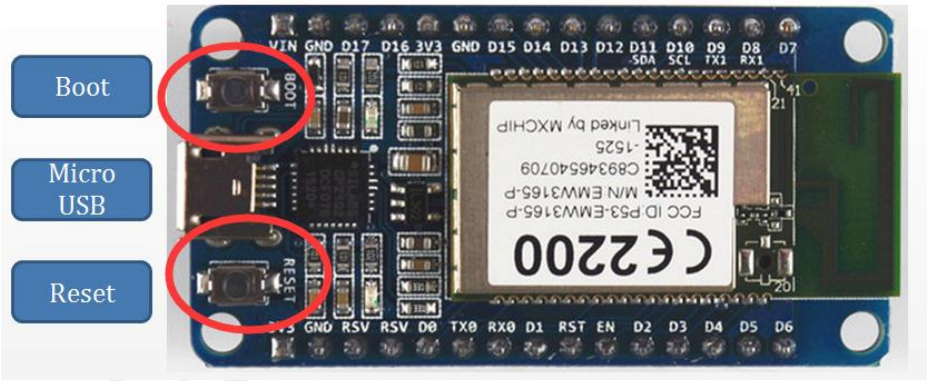


图 3.7 开发板进入 Bootloader 模式

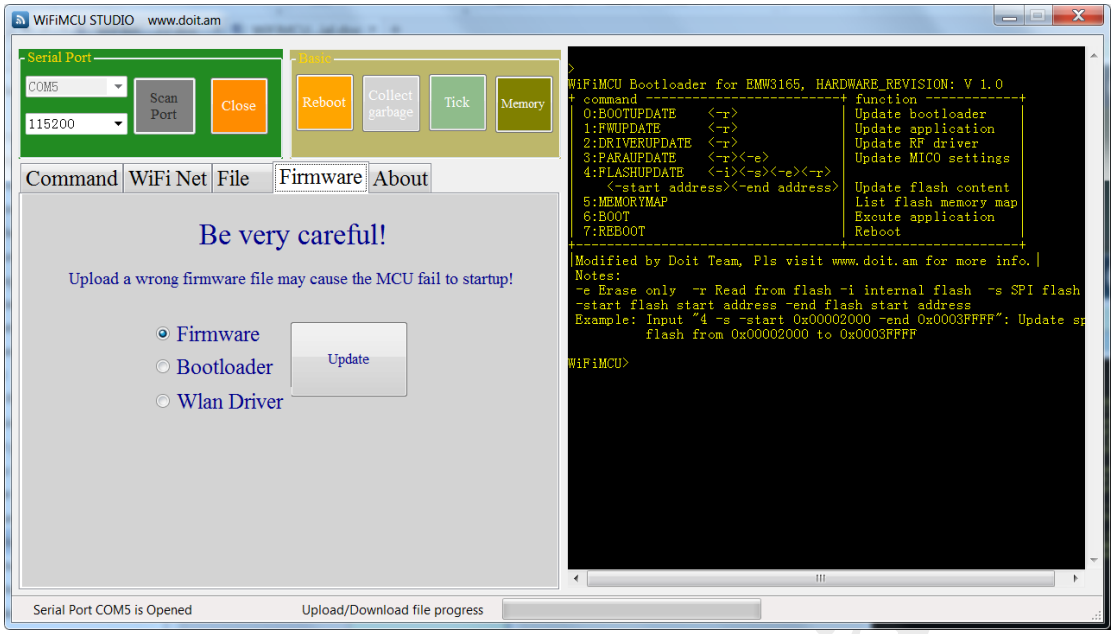


图 3.8 进入到 Bootloader 模式

步骤 4，选择固件升级。分别有“Firmware”， “Bootloader”， “Wlan Driver”三种可选，通常升级 WiFiMCU 固件选择“Firmware”。单击“Update”后，选择对应文件，稍等片刻升级自动开始。

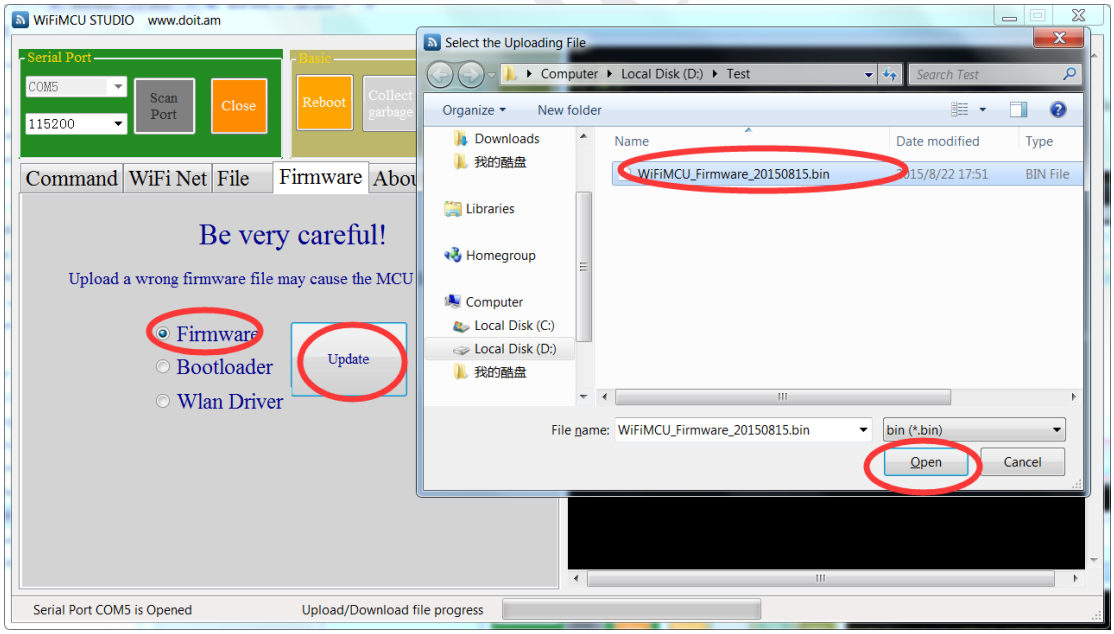


图 3.9 选择 Firmware 升级

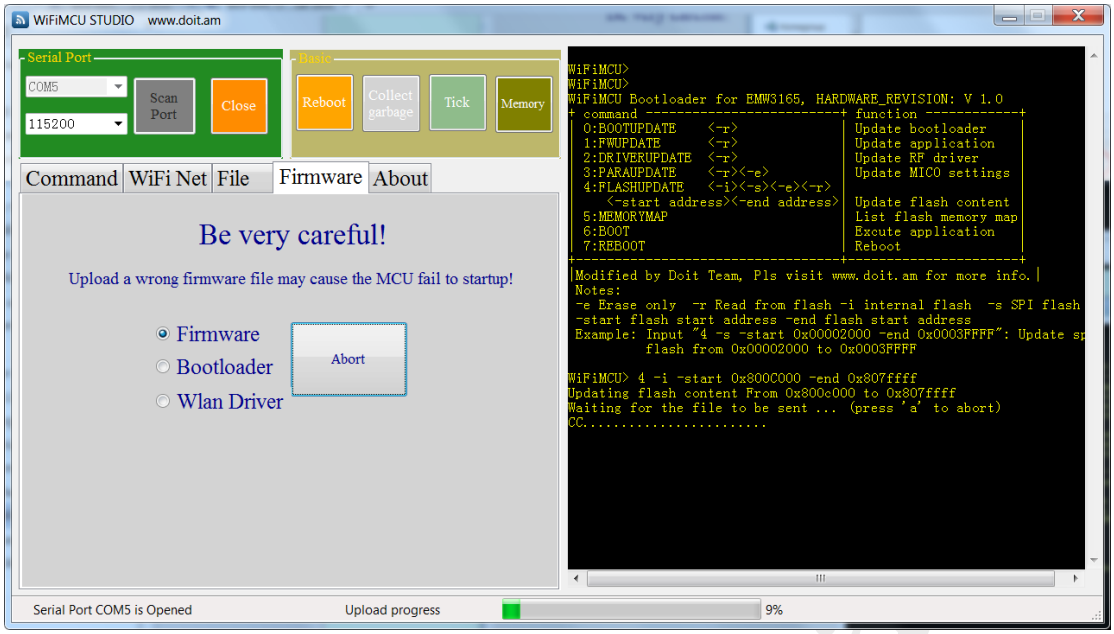


图 3.10 升级自动进行

步骤 5，等待升级完成。升级过程中，可使用“Abort”按钮，终止升级过程。如果升级失败，或者 WiFiMCU 开发板中途意外断电，可能导致模块程序损坏，下一次启动失败。遇到这种情况，可按照本章第三节使用 SWD 编程工具恢复。

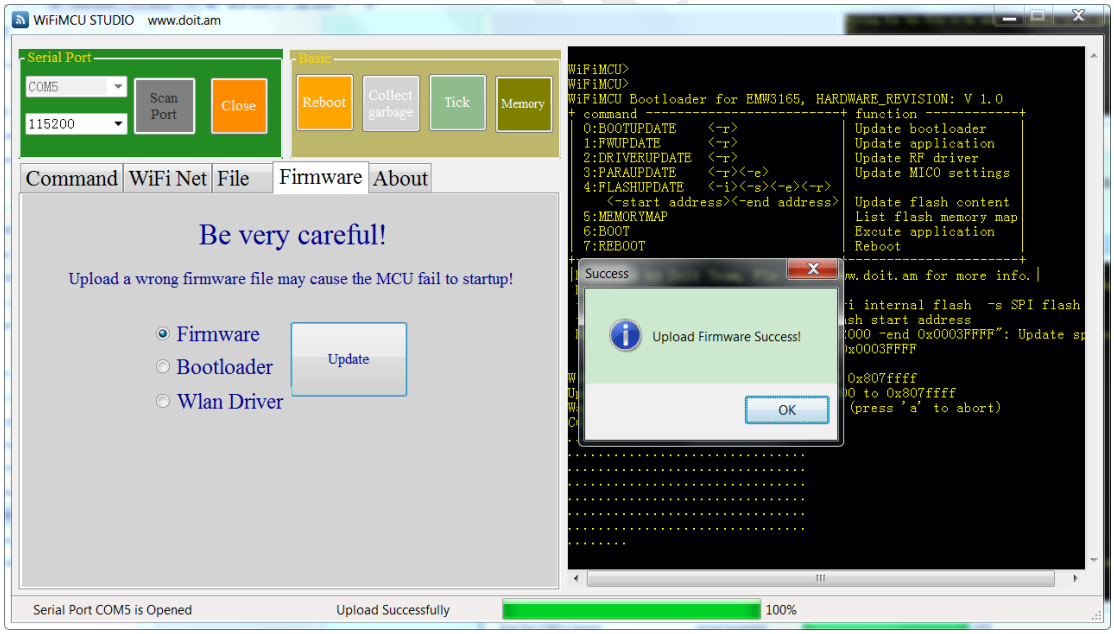


图 3.11 升级成功

步骤 6，在 WiFiMCU STUDIO 上点击“Reboot”按钮，重启 WiFiMCU 开发板。使用“=mcu.ver()”命令可以查看升级固件版本号。

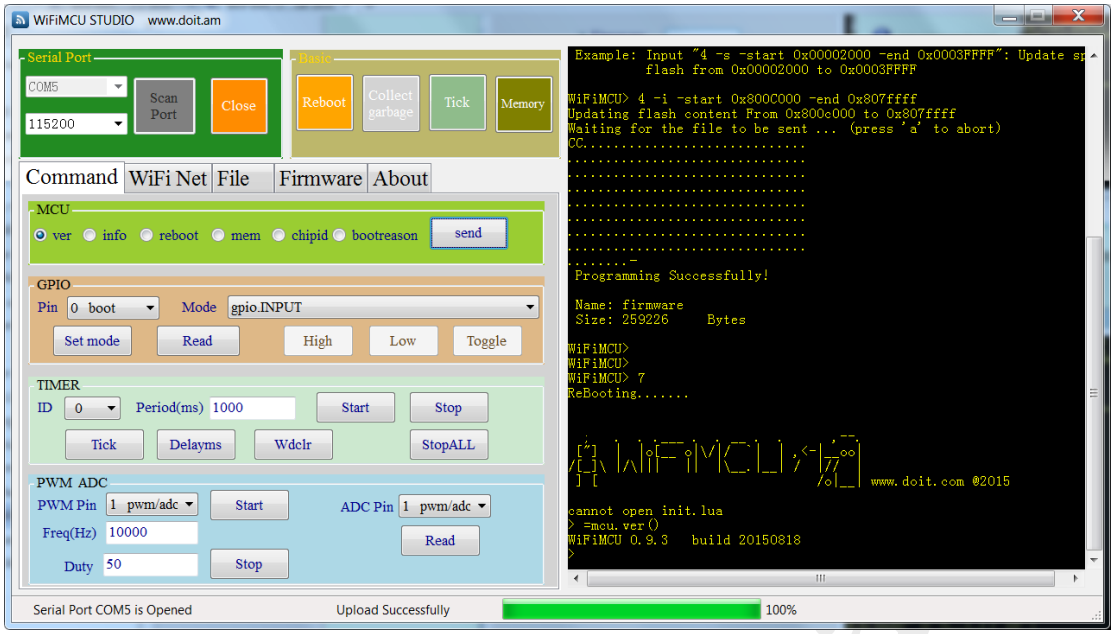


图 3.12 查看升级固件版本

4.3 使用 SecureCRT 升级固件

WiFiMCU 固件可通过其他支持 Y modem 传输协议的串口工具升级。以 SecureCRT 软件为例。

步骤 1，在 SercureCRT 中建立一个会话，如“上手入门”章节中的“使用串口调试工具交互”一节所示。

步骤 2，在 WiFiMCU 开发板上，按住“Boot”按钮，然后按“Reset”，重启 WiFiMCU 开发板，重启后，松开“Boot”按钮。此时开发板进入到 Bootloader 模式。

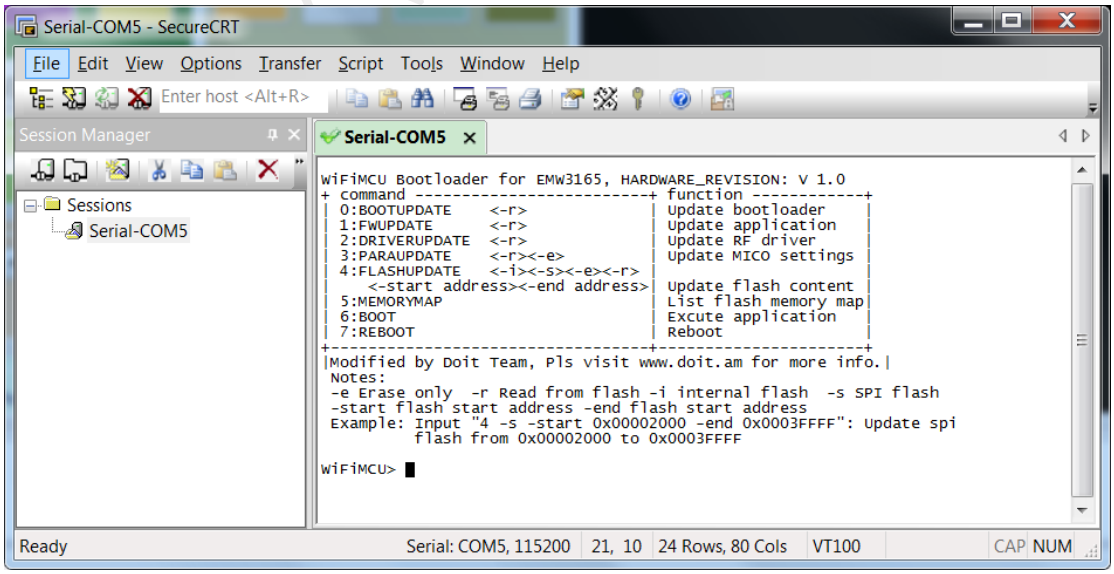


图 3.13 WiFiMCU 开发板进入 Bootloader 模式

步骤 3，使用如下命令升级不同的固件

WiFiMCU Firmware 固件：“4 -i -start 0x800C000 -end 0x807ffff”

Bootloader 固件：“4 -i -start 0x8000000 -end 0x8007fff”

Wlan 固件：“4 -s -start 0x00002000 -end 0x0003FFFF”

步骤 4，以 WiFiMCU Firmware 固件升级为例。输入“4 -i -start 0x800C000 -end 0x807ffff”。

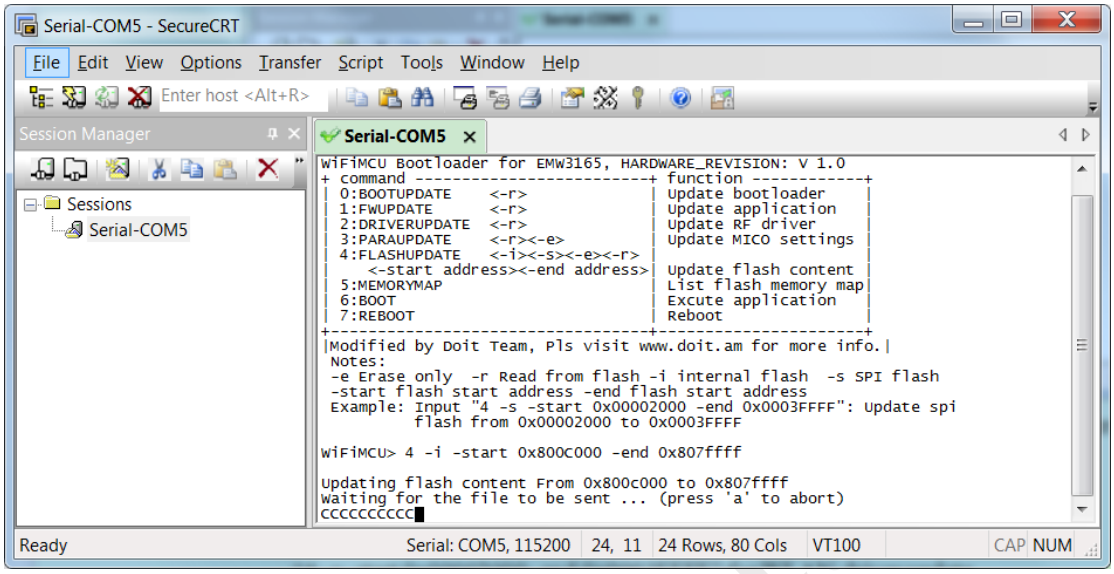


图 3.14 WiFiMCU Firmware 固件升级

步骤 5，在菜单“Transfer”中选择“Send Ymodem”，然后选择固件。确定后，升级开始。

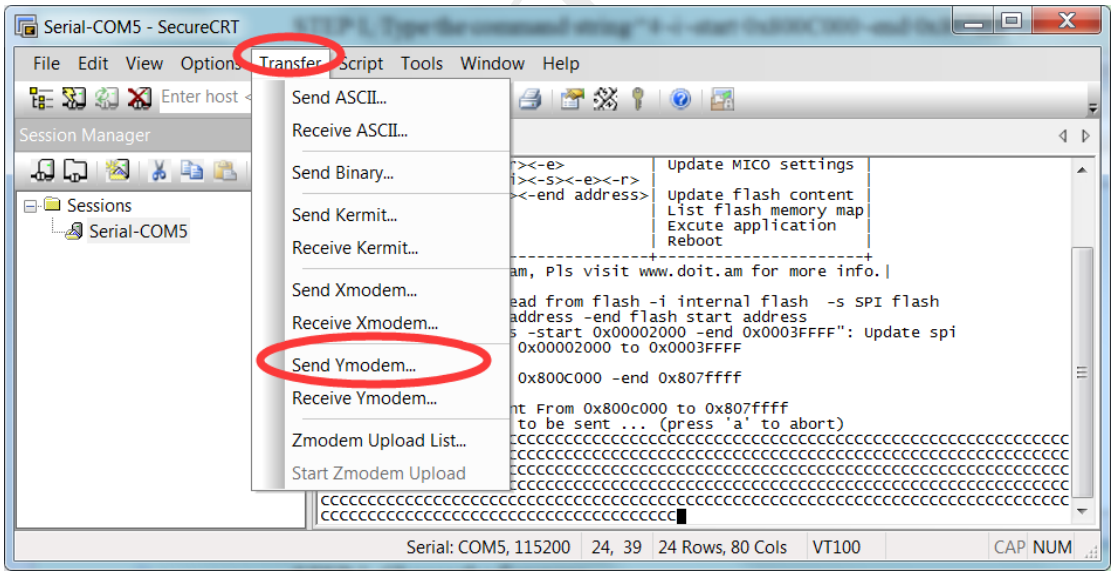


图 3.15 选择 Y modem 传输

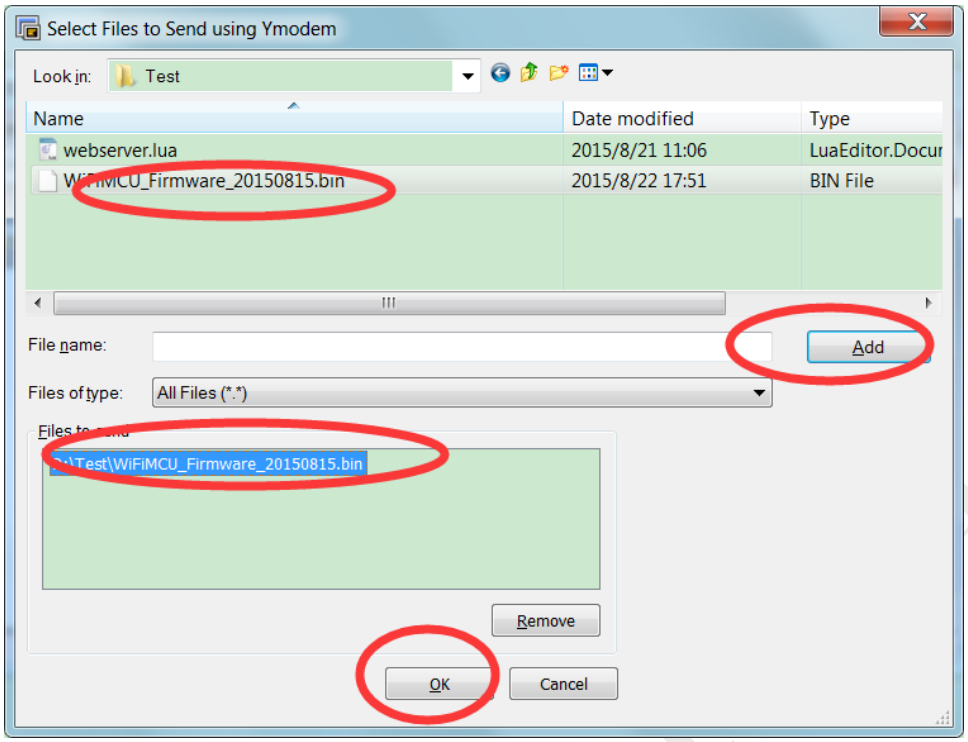


图 3.16 选择文件

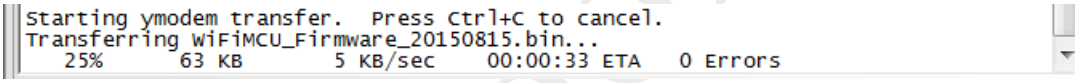


图 3.17 文件上传中

步骤 6 升级完成后，重启 WiFiMCU，使用“=mcu.ver()”命令可以查看升级固件版本号。

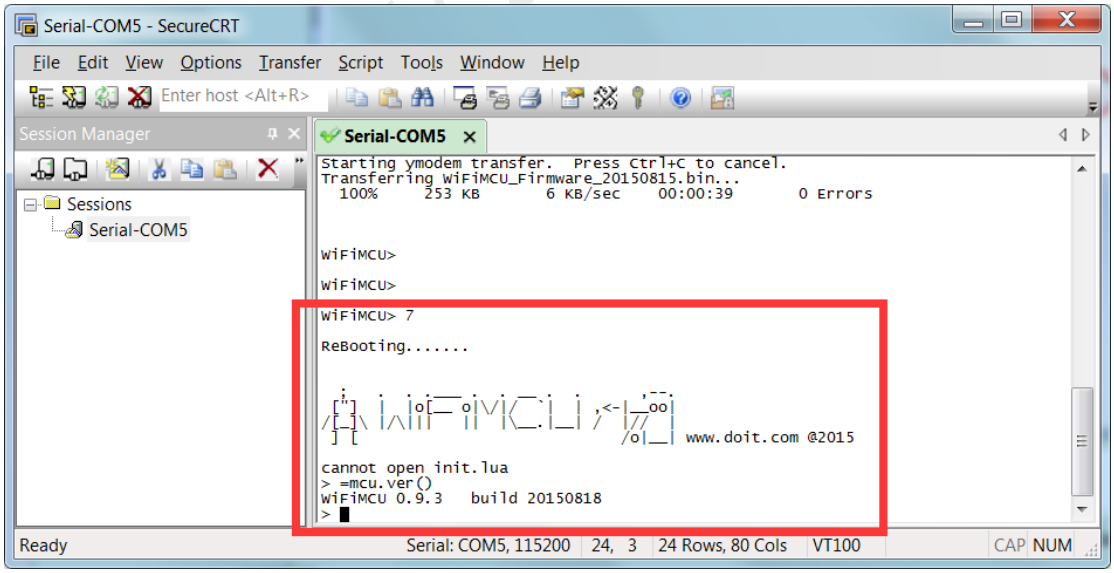


图 3.18 升级完成

4.4 使用 SWD 工具升级固件

WiFiMCU 的 SWD 接口如下图所示。D5，D6 分别是 SWCLK 和 SWDIO。开发者可以利用 MDK、IAR 等开发环境，编译源码后直接在 IDE 中下载固件。也可使用 SEGGER J-Flash 等第三方编程工具，填写对应的 Flash 烧写地址对 WiFiMCU 进行编程，实现固件升级。



图 3.19 SWD 接口

附录 资源链接

- CP210X USB 驱动下载
  http://www.silabs.com/products/mcu/Pages/USBtoUARTBridgeVCPDrivers.aspx
- WiFiMCU 固件源代码及固件下载
  https://github.com/SmartArduino/WiFiMCU/releases
- WiFiMCU STUDIO 下载
  https://github.com/SmartArduino/WiFiMCU-STUDIO
  WiFiMCU 的函数手册以及电路原理图：
  https://github.com/SmartArduino/WiFiMCU/tree/master/Document/
- eLua 开源项目
  https://github.com/elua/elua
- NodeMCU 开源项目
  https://github.com/nodemcu/nodemcu-firmware
- Lua 5.1.4 手册
  http://www.lua.org/manual/5.1/
- 网络社区
  www.emw3165.com http://bbs.smartarduino.com http://bbs.doit.am