



# Luat\_IOT\_SDK\_C 语言用户开发指导手册

# 目录

1 简介.....	3
2 开发环境搭建.....	3
2.1 下载驱动.....	3
2.2 安装驱动.....	3
2.3 驱动测试.....	3
3 用户程序编译.....	4
4 下载运行.....	6
4.1 下载工具介绍.....	6
4.2 固件选择.....	6
4.3 下载固件.....	6
5 新建工程.....	7
5.1 建立新项目.....	7
5.2 添加代码.....	8
5.3 建立编译 bat 文件.....	9
6 如何调试.....	10
6.1 调试工具介绍.....	10
6.2 打开调试工具.....	10
6.3 查看 trace.....	12
6.4 常见问题.....	14

## 1 简介

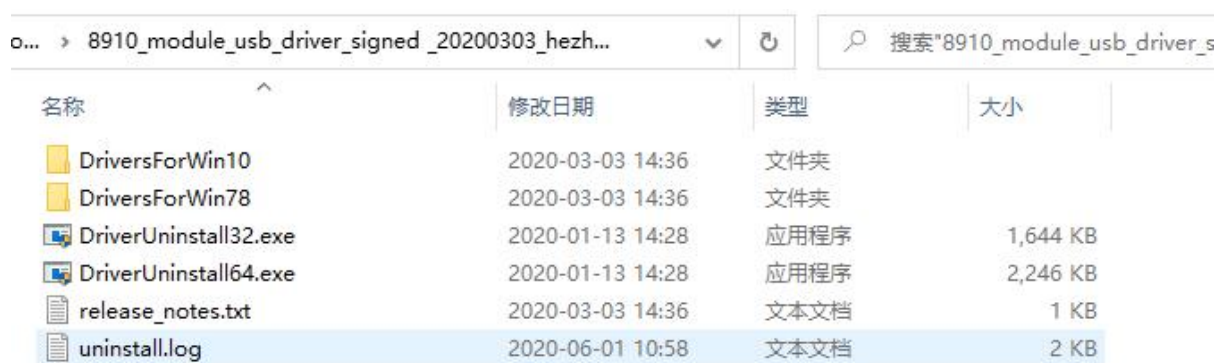
本文档主要对 720U 系列模块 CSDK 版本用户程序编程进行指导。此 CSDK 版本针对合宙模块提供的一套 C 语言的软件开发环境，让客户像开发单片机一样，使用合宙的无线通信模块。

## 2 开发环境搭建

### 2.1 下载驱动

从官网下载最新的驱动程序，驱动下载地址：[8910 module usb driver](#)

下载完成后解压安装包，进入安装包文件\8910\_module\_usb\_driver\_signed\_20200303\_hezhou

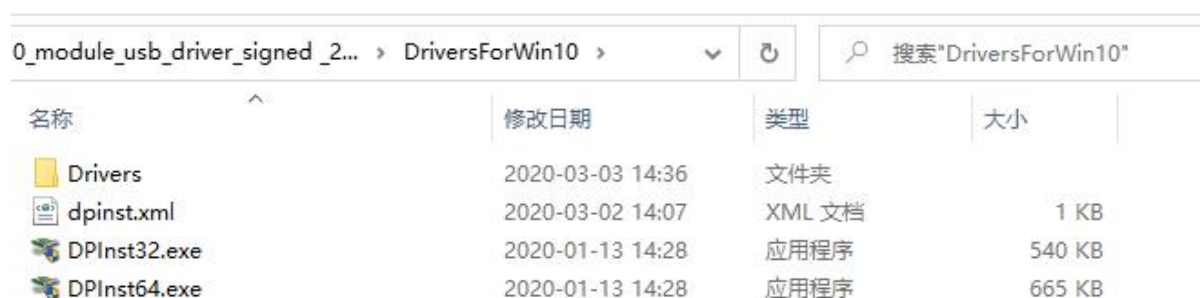


名称	修改日期	类型	大小
DriversForWin10	2020-03-03 14:36	文件夹	
DriversForWin78	2020-03-03 14:36	文件夹	
DriverUninstall32.exe	2020-01-13 14:28	应用程序	1,644 KB
DriverUninstall64.exe	2020-01-13 14:28	应用程序	2,246 KB
release_notes.txt	2020-03-03 14:36	文本文档	1 KB
uninstall.log	2020-06-01 10:58	文本文档	2 KB

### 2.2 安装驱动

根据自己的操作系统选择进入对应的文件夹。

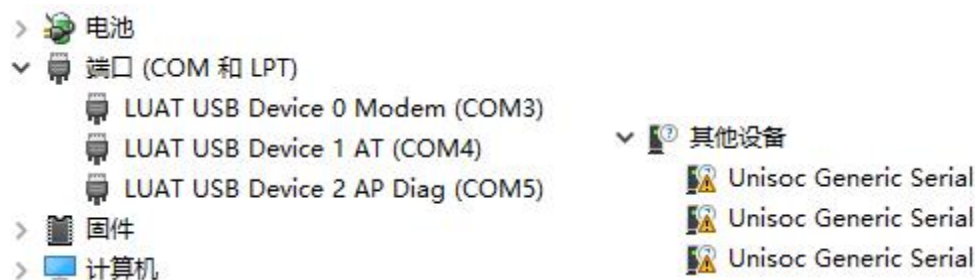
例如：win10 基于 X64 系统，选择 DriversForWin10/DPIInst64.exe 进行安装



名称	修改日期	类型	大小
Drivers	2020-03-03 14:36	文件夹	
dpinst.xml	2020-03-02 14:07	XML 文档	1 KB
DPIInst32.exe	2020-01-13 14:28	应用程序	540 KB
DPIInst64.exe	2020-01-13 14:28	应用程序	665 KB

### 2.3 驱动测试

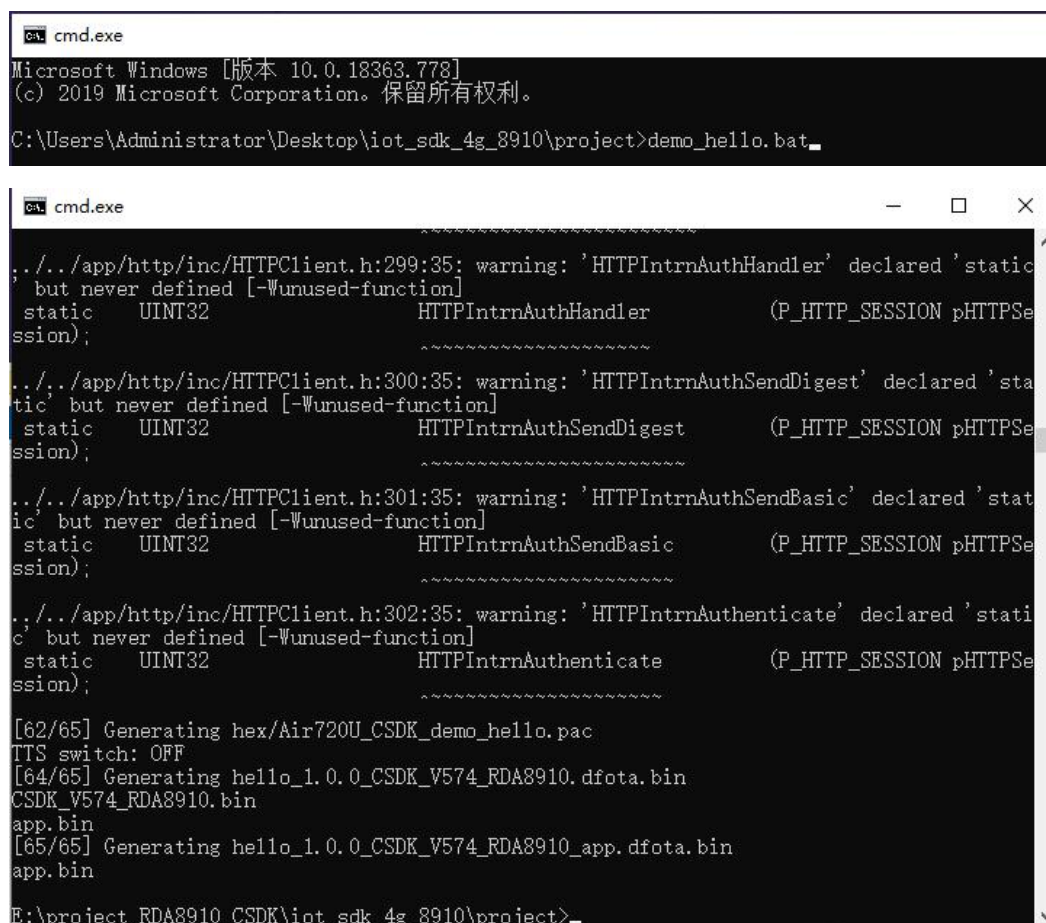
安装成功后将开发板插上电脑，将开关拨到 On，并且长按开机键，等待模块正常运行后查看设备管理器/端口，是否有如下左图三个设备。如果没有，查看设备管理器中其他设备有无驱动未安装如下右图，如果有说明驱动安装失败，需要重新安装驱动；否则建议更换 USB 线束后重新安装驱动。



### 3 用户程序编译

进入 `iot_sdk_4g_8910/project` 文件夹，双击运行 `cmd.exe`，在命令行选择对应的 demo 文件进行编译。

例如：编译 `demo_hello` 文件，命令行输入 `demo_hello.bat`，回车运行 bat 文件，此时开始编译文件。



```
cmd.exe
Microsoft Windows [版本 10.0.18363.778]
(c) 2019 Microsoft Corporation。保留所有权利。

C:\Users\Administrator\Desktop\iot_sdk_4g_8910\project>demo_hello.bat

..\..\app/http/inc/HTTPClient.h:299:35: warning: 'HTTPIntrnAuthHandler' declared 'static'
but never defined [-Wunused-function]
static UINT32 HTTPIntrnAuthHandler (P_HTTP_SESSION pHTTPSe
ssion);
..\..\app/http/inc/HTTPClient.h:300:35: warning: 'HTTPIntrnAuthSendDigest' declared 'sta
tic' but never defined [-Wunused-function]
static UINT32 HTTPIntrnAuthSendDigest (P_HTTP_SESSION pHTTPSe
ssion);
..\..\app/http/inc/HTTPClient.h:301:35: warning: 'HTTPIntrnAuthSendBasic' declared 'stat
ic' but never defined [-Wunused-function]
static UINT32 HTTPIntrnAuthSendBasic (P_HTTP_SESSION pHTTPSe
ssion);
..\..\app/http/inc/HTTPClient.h:302:35: warning: 'HTTPIntrnAuthenticate' declared 'stati
c' but never defined [-Wunused-function]
static UINT32 HTTPIntrnAuthenticate (P_HTTP_SESSION pHTTPSe
ssion);
[62/65] Generating hex/Air720U_CSDK_demo_hello.pac
TTS switch: OFF
[64/65] Generating hello_1.0.0_CSDK_V574_RDA8910.dfota.bin
CSDK_V574_RDA8910.bin
app.bin
[65/65] Generating hello_1.0.0_CSDK_V574_RDA8910_app.dfota.bin
app.bin

E:\project_RDA8910_CSDK\iot_sdk_4g_8910\project>
```

编译结束后，在 `iot_sdk_4g_8910` 目录下出现 `hex` 文件夹，对应的目录下面包含编译项目的文件；

project\_RDA8910\_CSDK &gt; iot\_sdk\_4g\_8910 &gt; hex

搜索"hex"

名称	修改日期	类型	大小
Air720U_CSDK_demo_hello	2020-06-01 11:31	文件夹	
Air720U_CSDK_demo_hello_map	2020-06-01 11:31	文件夹	

Air720U\_CSDK\_demo\_hello 目录下

包含 csdk 层的下载固件: Air720U\_CSDK\_demo\_hello\_APP.pac

包含底层和 csdk 层的下载固件: Air720U\_CSDK\_demo\_hello.pac

包含 csdk 层的 OTA 升级固件: hello\_1.0.0\_CSDK\_V574\_RDA8910\_app.dfota.bin

包含底层和 csdk 层的 OTA 升级固件: hello\_1.0.0\_CSDK\_V574\_RDA8910.dfota.bin

t\_sdk\_4g\_8910 &gt; hex &gt; Air720U\_CSDK\_demo\_hello

搜索"Air720U\_CSDK\_der

名称	修改日期	类型	大小
Air720U_CSDK_demo_hello.pac	2020-06-01 11:31	PAC 文件	4,627 KB
Air720U_CSDK_demo_hello_APP.pac	2020-06-01 11:31	PAC 文件	71 KB
appimg_flash_delete.pac	2020-06-01 11:31	PAC 文件	71 KB
hello_1.0.0_CSDK_V574_RDA8910.dfot...	2020-06-01 11:31	BIN 文件	3,854 KB
hello_1.0.0_CSDK_V574_RDA8910_app...	2020-06-01 11:31	BIN 文件	1 KB

Air720U\_CSDK\_demo\_hello\_map 目录下 包含编译生成的 map 和 elf 文件。

&lt; &gt; Air720U\_CSDK\_demo\_hello\_map

搜索"Air720U\_CSDK\_der

名称	修改日期	类型	大小
CSDK_V574_RDA8910.elf	2020-06-01 11:31	ELF 文件	8,933 KB
CSDK_V574_RDA8910.map	2020-06-01 11:31	MAP 文件	1,748 KB

如果编译出现错误, 可以在 cmd 命令行中搜索 "error" 来定位错误位置。

```

选择cmd.exe
E:/project_RDA8910_CSDK/iot_sdk_4g_8910/out/hello_debug/hex/Air720U_CSDK_demo_hello.pac
-- Configuring done
-- Generating done
-- Build files have been written to: E:/project_RDA8910_CSDK/iot_sdk_4g_8910/out/hello_debug
[1/8] Building C object demo/hello/CMakeFiles/r720U_CSDK_demo_hello.dir/demo_hello.c.obj
FAILED: demo/hello/CMakeFiles/Air720U_CSDK_demo_hello.dir/demo_hello.c.obj
E:/project_RDA8910_CSDK/iot_sdk_4g_8910/prebuilts/win32/gcc-arm-none-eabi/bin/arm-none-eabi-gcc.exe -I../components/include -I../components/newlib/include -I../components/openat_inc -I../api/include -I../app/ssl/..../api/include -I../app/ssl/inc -I../app/ftp/..../api/include -I../app/ftp/inc -I../app/http/..../api/include -I../app/http/inc -std=gnu11 -mcpu=cortex-a5 -mtune=generic-armv7-a -mthumb -mfpv4 -mfloat-abi=hard -mno-unaligned-access -g -O0 -Wall -fno-strict-aliasing -fcommon-sections -fdata-sections -MD -MT demo/hello/CMakeFiles/Air720U_CSDK_demo_hello.dir/demo_hello.c.obj -MF demo/hello/CMakeFiles/Air720U_CSDK_demo_hello.dir/demo_hello.c.obj -o demo/hello/CMakeFiles/Air720U_CSDK_demo_hello.dir/demo_hello.c.obj -c ../demo/hello/demo_hello.c
../demo/hello/demo_hello.c: In function 'appimg_enter':
../demo/hello/demo_hello.c:32:5: error: expected ';' before 'return'
    return 0;
    ^
../demo/hello/demo_hello.c:33:1: warning: control reaches end of non-void function [-Wreturn-type]
}
[2/8] Generating hex/appimg_flash_delete.pac
ninja: build stopped: subcommand failed.
E:/project_RDA8910_CSDK/iot_sdk_4g_8910/project>

```


## 4 下载运行

### 4.1 下载工具介绍

下载工具使用的是合宙官方工具 luatools。下载地址：[luatools v2](#)



### 4.2 固件选择

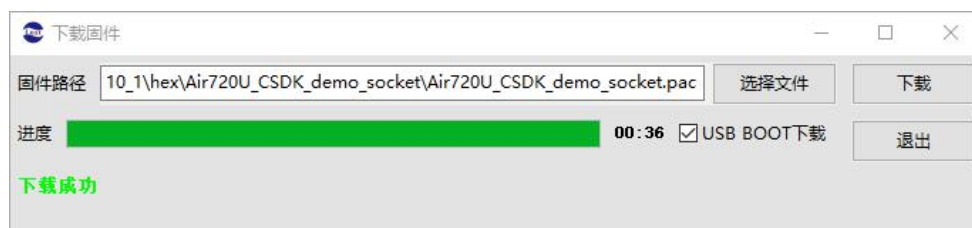
点击右上角下载固件图标，在下载固件弹窗中选择需要下载的文件，例如 socket\_demo 固件：

hex/Air720U\_CSDK\_demo\_socket/Air720U\_CSDK\_demo\_socket.pac



### 4.3 下载固件

勾选 USB BOOT 下载，点击下载。长按模块上的 U\_BOOT 键，再按重启键进入下载模式进行下载，下载过程中会有进度条提示下载进度，完成时会显示下载成功。



下载工具 LuaTools 的使用指南：<https://ask.openluat.com/article/1058>

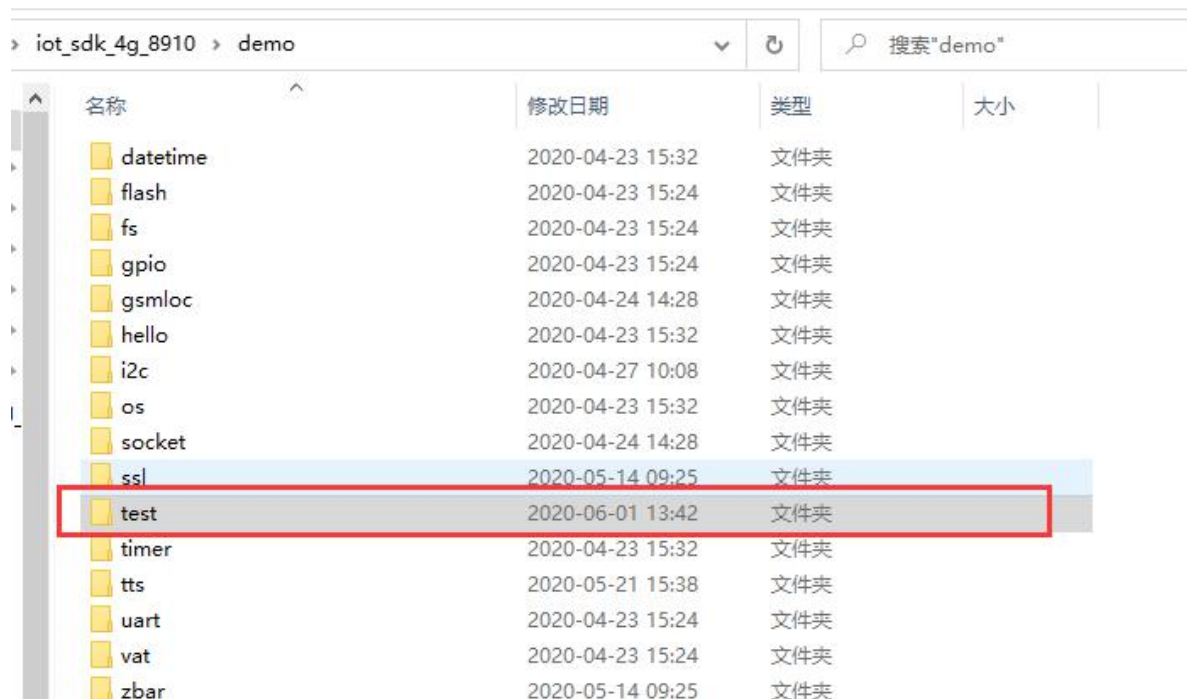


## 5 新建工程

### 5.1 建立新项目

首先下载并解压 `iot_sdk_4g_8910.zip` 文件，在 `iot_sdk_4g_8910/demo` 目录下新建项目的文件夹：

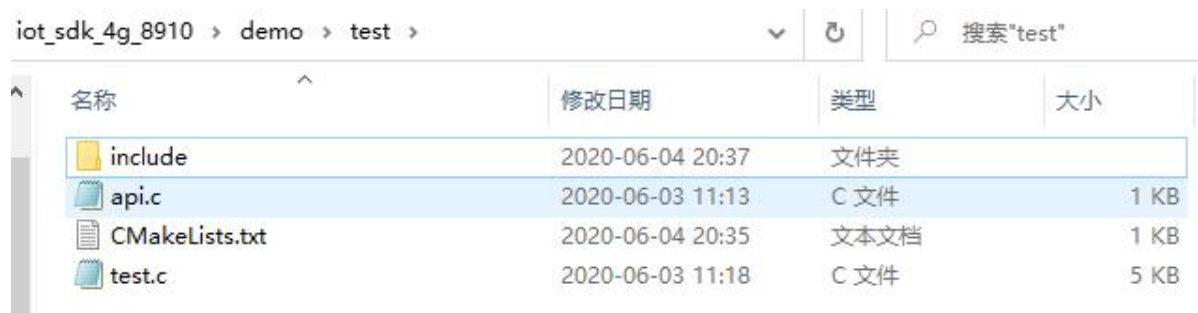
例如：test



名称	修改日期	类型	大小
datetime	2020-04-23 15:32	文件夹	
flash	2020-04-23 15:24	文件夹	
fs	2020-04-23 15:24	文件夹	
gpio	2020-04-23 15:24	文件夹	
gsmloc	2020-04-24 14:28	文件夹	
hello	2020-04-23 15:32	文件夹	
i2c	2020-04-27 10:08	文件夹	
os	2020-04-23 15:32	文件夹	
socket	2020-04-24 14:28	文件夹	
ssl	2020-05-14 09:25	文件夹	
test	2020-06-01 13:42	文件夹	
timer	2020-04-23 15:32	文件夹	
tts	2020-05-21 15:38	文件夹	
uart	2020-04-23 15:24	文件夹	
vat	2020-04-23 15:24	文件夹	
zbar	2020-05-14 09:25	文件夹	

在新建的文件夹下面新建两个文件：`CMakeLists.txt` 和 `test.c`；

可以从 `demo` 文件中拷贝一份过来再进行修改使用。



名称	修改日期	类型	大小
include	2020-06-04 20:37	文件夹	
api.c	2020-06-03 11:13	C 文件	1 KB
CMakeLists.txt	2020-06-04 20:35	文本文档	1 KB
test.c	2020-06-03 11:18	C 文件	5 KB

在 `test.c` 中建立入口函数，并加入自己开始执行的程序代码：

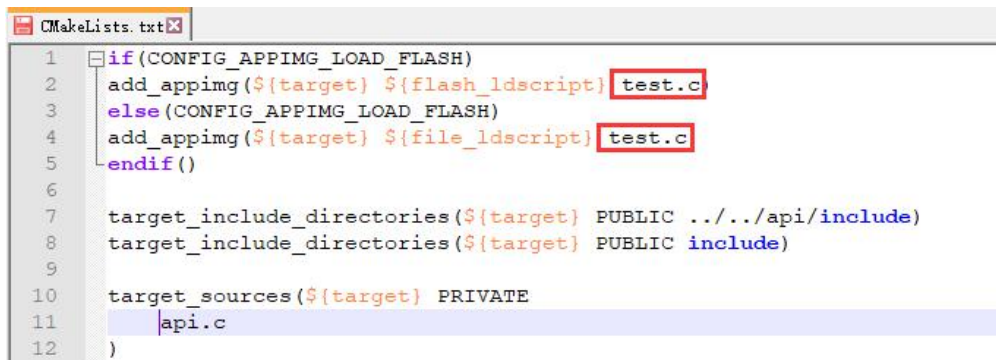
```
#include "iot_debug.h"

static void test(void)
{
    iot_debug_print("[hello]hello world %d", n);
}

int appimg_enter(void *param)
{
    iot_debug_print("[hello]appimg_enter");
    test();
    return 0;
}

void appimg_exit(void)
{
    iot_debug_print("[hello]application image exit");
}
```

将 CMakeLists.txt 里面如下图方框位置改成包含入口函数的\*.c 文件名称  
例如: test.c



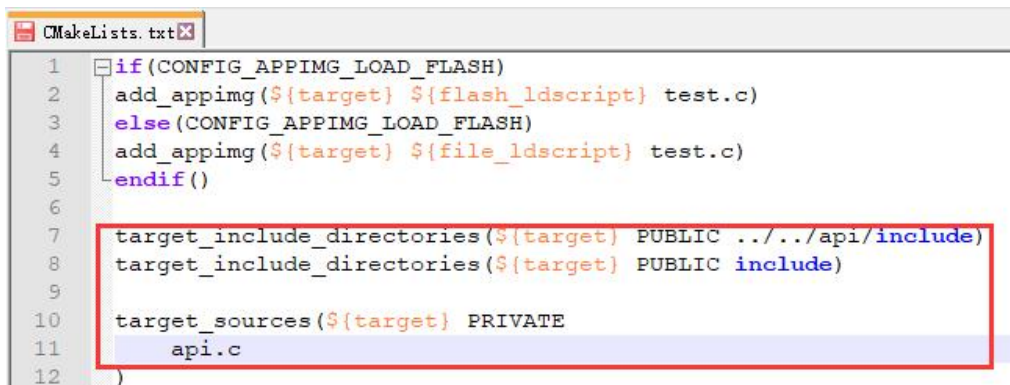
```
CMakeLists.txt
1  if(CONFIG_APPIMG_LOAD_FLASH)
2      add_appimg(${target} ${flash_ldscript} test.c)
3  else(CONFIG_APPIMG_LOAD_FLASH)
4      add_appimg(${target} ${file_ldscript} test.c)
5  endif()
6
7  target_include_directories(${target} PUBLIC ../../api/include)
8  target_include_directories(${target} PUBLIC include)
9
10 target_sources(${target} PRIVATE
11     api.c
12 )
```

## 5.2 添加代码

自己的程序代码按照如下格式加入 CMakeLists.txt 文件中进行编译使用。

头文件放在 `target_include_directories(${target} PUBLIC include)`

源文件放在 `target_sources(${target} PRIVATE api.c)`



```
CMakeLists.txt
1  if(CONFIG_APPIMG_LOAD_FLASH)
2      add_appimg(${target} ${flash_ldscript} test.c)
3  else(CONFIG_APPIMG_LOAD_FLASH)
4      add_appimg(${target} ${file_ldscript} test.c)
5  endif()
6
7  target_include_directories(${target} PUBLIC ../../api/include)
8  target_include_directories(${target} PUBLIC include)
9
10 target_sources(${target} PRIVATE
11     api.c
12 )
```



需要使用 CSDK 提供的 API 可以参考 `iot_sdk_4g_8910/demo` 目录下包含各个功能的例子，选择需要的功能参考 demo 中项目代码进行添加，实现自己想要的功能。

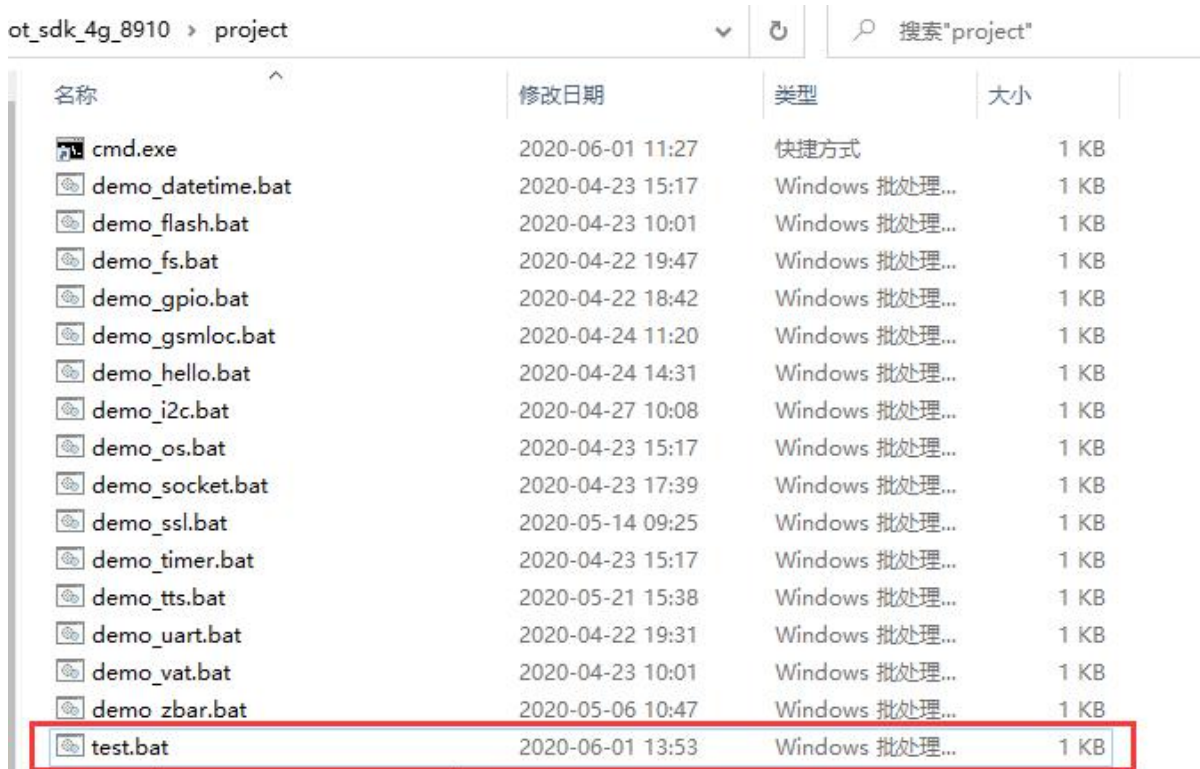
用户可使用的 API 所在文件：`iot_sdk_4g_8910/api` 目录下所包含的相关头文件，API 使用方法请参考《Luat\_IoT\_LTE\_SDK\_C 语言编程手册.chm》

注意：除了 `iot_sdk_4g_8910/api` 目录下所包含的相关头文件提供的 API 可使用外，其它\*.h 中的 API 不要使用，否则会出现无法预知的错误，请切记。

其它标准库函数调用，只要包含对应的标准头文件即可，当前是支持的。比如要使用 `strcpy` 函数，则需要包含头文件：`#include "string.h"`

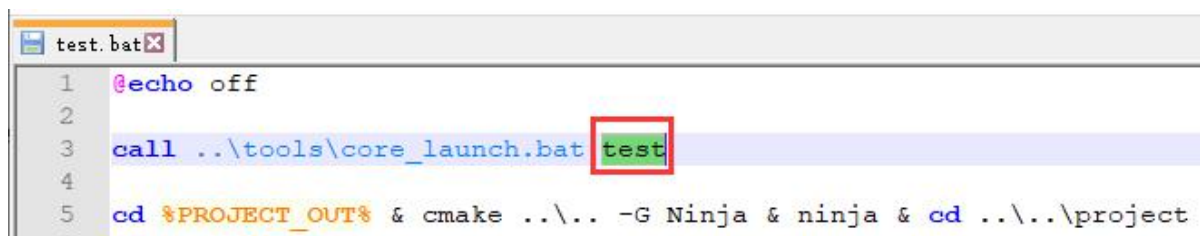
### 5.3 建立编译 bat 文件

在 `iot_sdk_4g_8910/demo` 目录下建立自己的编译 bat 文件



名称	修改日期	类型	大小
cmd.exe	2020-06-01 11:27	快捷方式	1 KB
demo_datetime.bat	2020-04-23 15:17	Windows 批处理...	1 KB
demo_flash.bat	2020-04-23 10:01	Windows 批处理...	1 KB
demo_fs.bat	2020-04-22 19:47	Windows 批处理...	1 KB
demo_gpio.bat	2020-04-22 18:42	Windows 批处理...	1 KB
demo_gsmloc.bat	2020-04-24 11:20	Windows 批处理...	1 KB
demo_hello.bat	2020-04-24 14:31	Windows 批处理...	1 KB
demo_i2c.bat	2020-04-27 10:08	Windows 批处理...	1 KB
demo_os.bat	2020-04-23 15:17	Windows 批处理...	1 KB
demo_socket.bat	2020-04-23 17:39	Windows 批处理...	1 KB
demo_ssl.bat	2020-05-14 09:25	Windows 批处理...	1 KB
demo_timer.bat	2020-04-23 15:17	Windows 批处理...	1 KB
demo_tts.bat	2020-05-21 15:38	Windows 批处理...	1 KB
demo_uart.bat	2020-04-22 19:31	Windows 批处理...	1 KB
demo_vat.bat	2020-04-23 10:01	Windows 批处理...	1 KB
demo_zbar.bat	2020-05-06 10:47	Windows 批处理...	1 KB
test.bat	2020-06-01 13:53	Windows 批处理...	1 KB

修改 bat 文件内容，修改内容为 demo 中对应文件夹名称



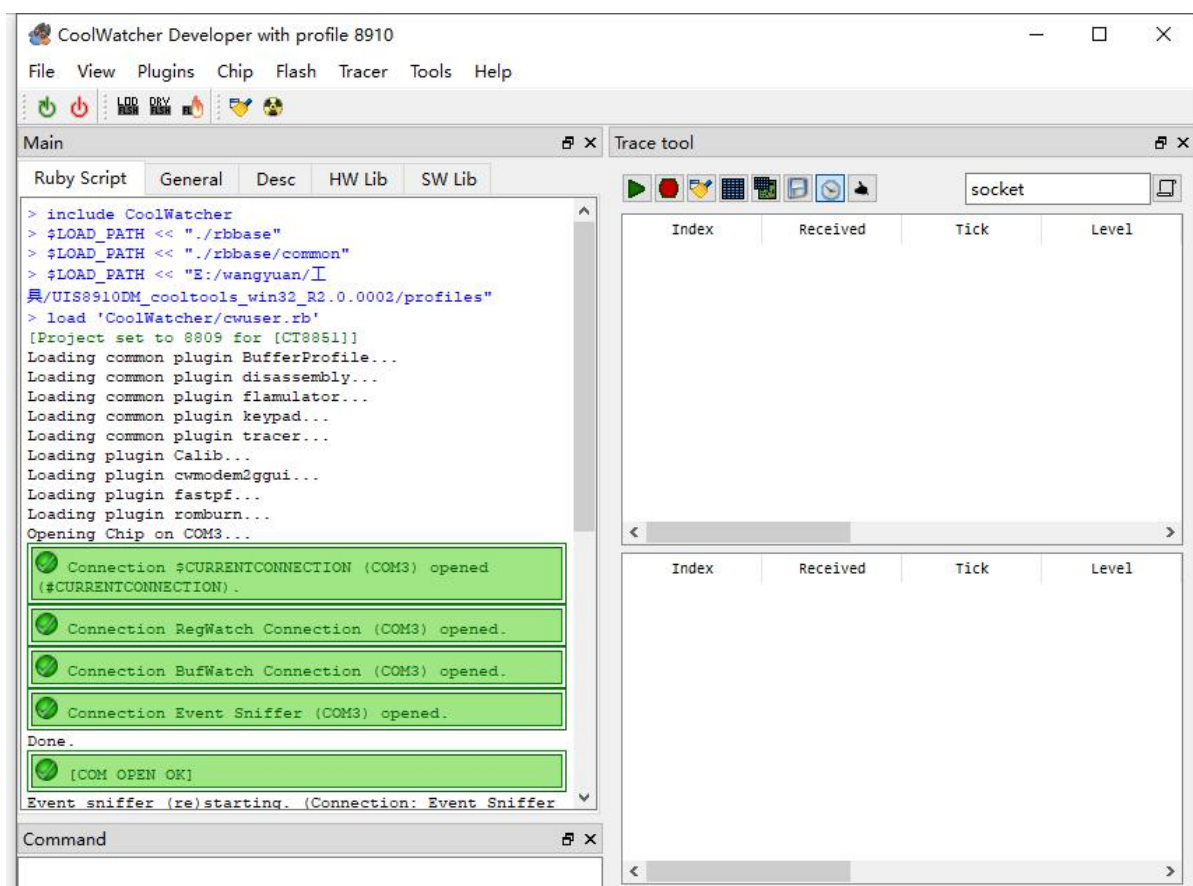
```
1 @echo off
2
3 call ..\tools\core_launch.bat test
4
5 cd %PROJECT_OUT% & cmake ..\.. -G Ninja & ninja & cd ..\..\project
```

## 6 如何调试

### 6.1 调试工具介绍

Coolwatcher 工具是通过 USB Diag 口或者设备的 host 口进行通信。使用 USB Diag 口进行通信使用 USB 线直接连接模块的 USB 管脚到 PC 的 USB 接口；host 口进行通信波特率为 921600，由于速率较高，所以建议使用 FTDI 等高速 USB 转串口线，其他 USB 转串口线可能会导致通信中断，从而影响调试。

Coolwatcher 工具下载地址：[Coolwatcher 下载](#)



### 6.2 打开调试工具

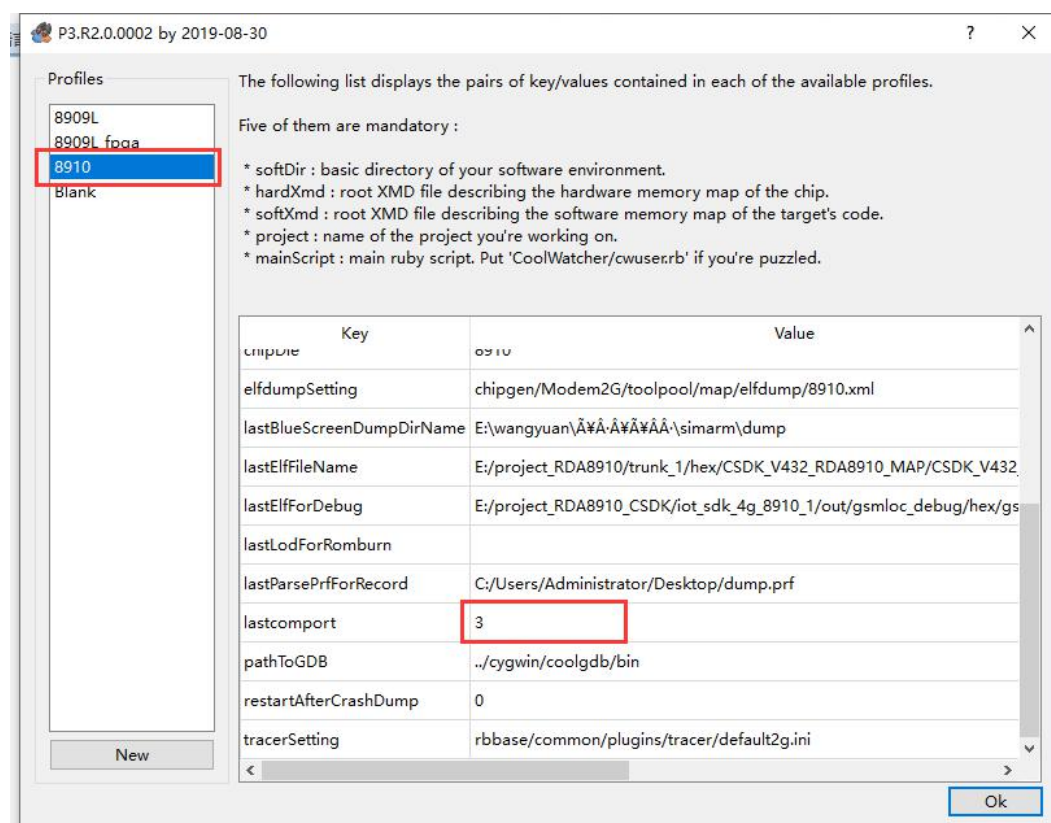
#### 1 确认通信串口号

电脑设备管理器中查看串口号，如下图



## 2 配置工具

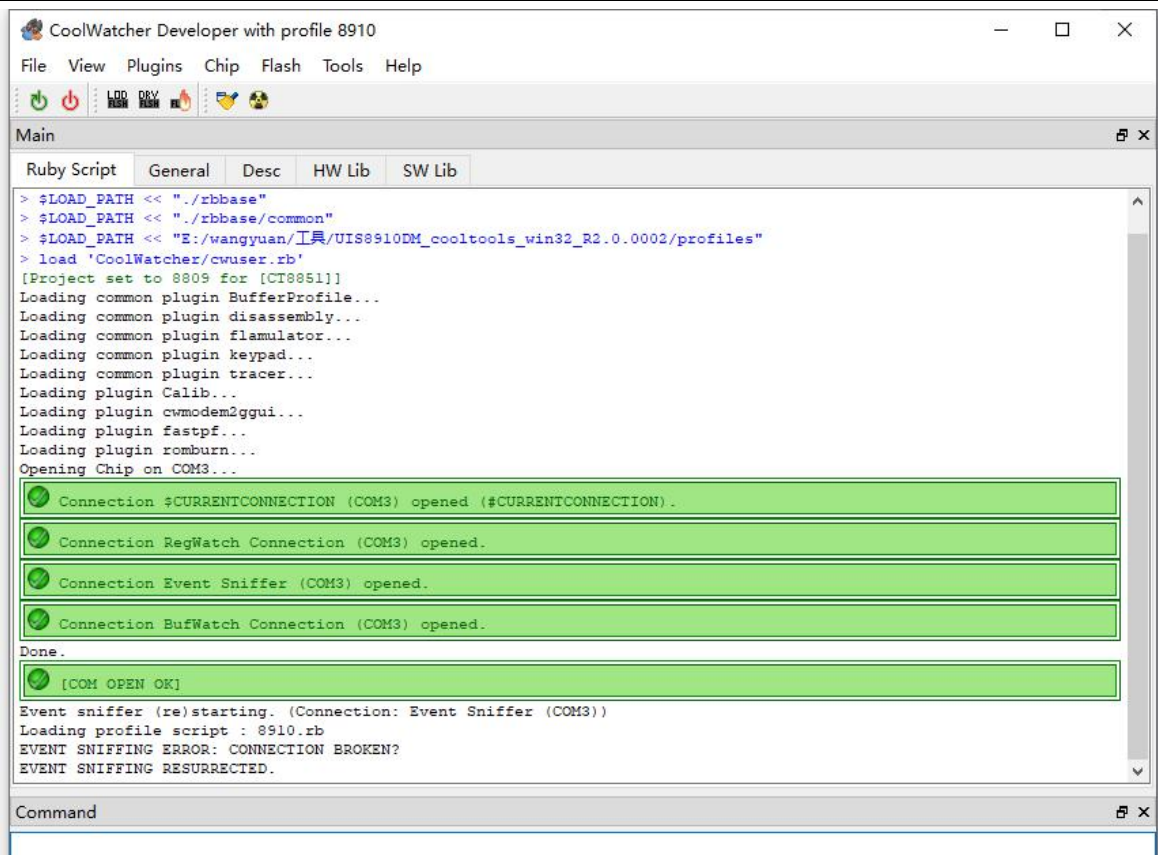
选择【8910】和 lastcomport，确认串口后点击【OK】



Lastcomport 和设备管理器中的串口号对应，例如【COM3】就填写 3。

## 3 确认打开是否正常

如果串口正常打开，工具左侧会有如下提示



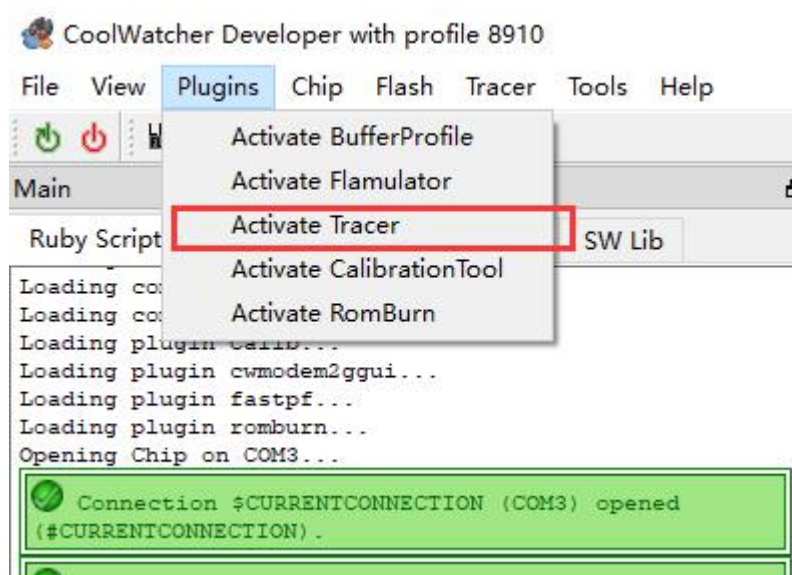
## 6.3 查看 trace

### 1 打开 trace tool

选择【plugins->Active Tracer】,工具右侧会出现【Trace tool】栏,如果没出现就右击一下工具空白处,选择【Trace tool】。

模块开机后,通过串口工具发 AT^TRACECTRL=0,1,3 这个指令,将日志打开

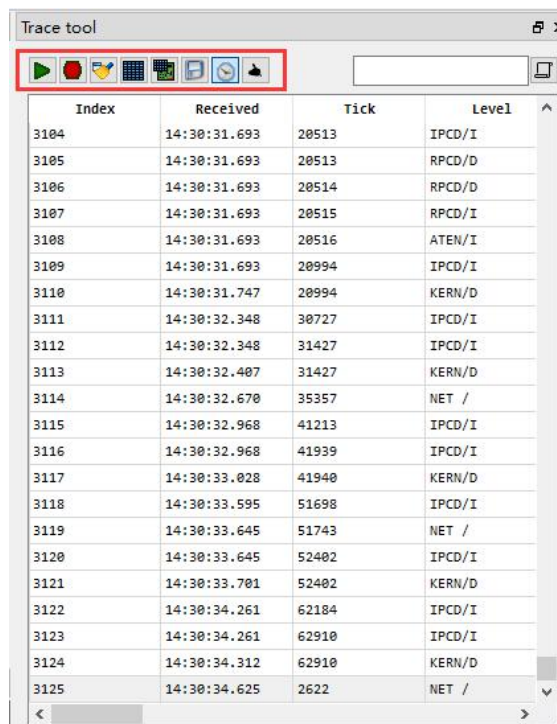
**注意: AT^TRACECTRL=0,1,3 是永久生效的,可以断电保存**





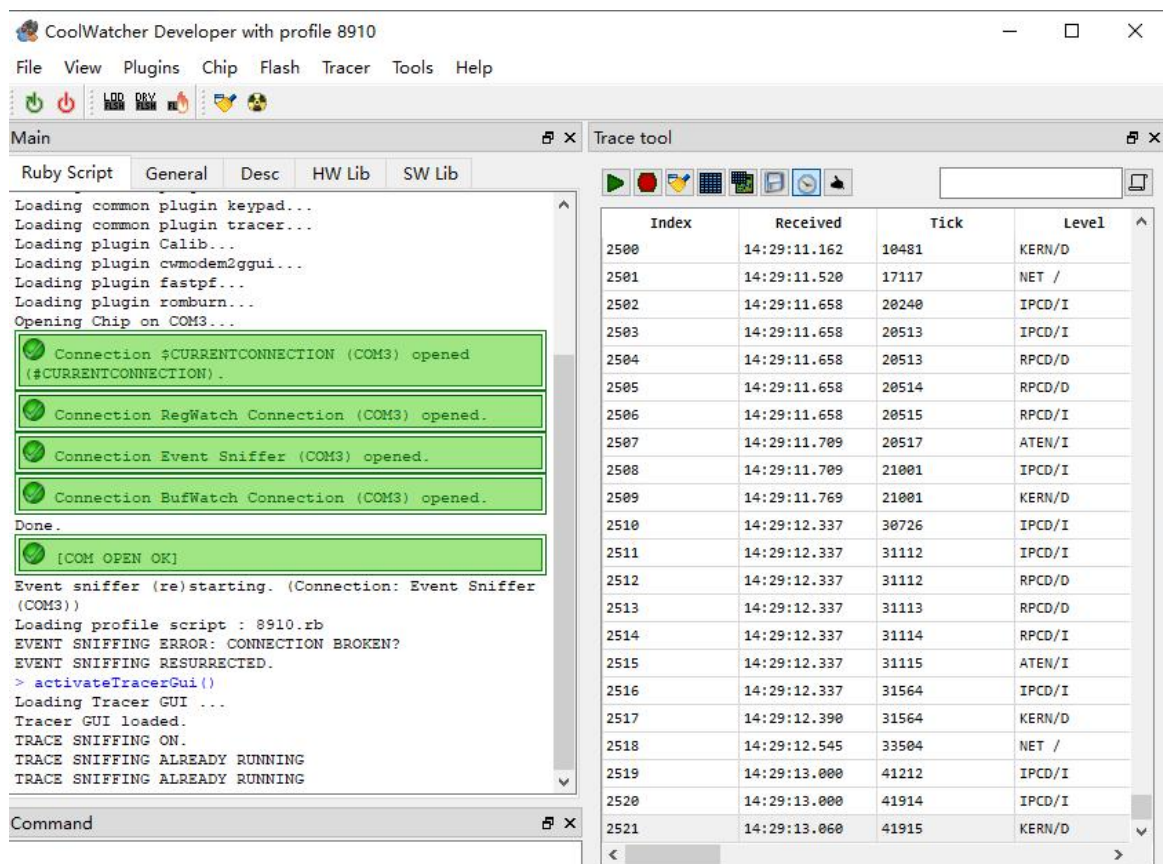
## 2 trace tool 工具说明

下图红框从左到右分别为【开始 trace】，【停止 trace】，【清空 trace】，【过滤 trace】，【应用过滤】，【保存 trace】，【使能 trace 时间】【设置 comment on/off】



The screenshot shows the 'Trace tool' window with a toolbar at the top. A red box highlights the toolbar icons: a green play button (start), a red stop button (stop), a yellow eraser (clear), a blue filter icon (filter), a green checkmark (apply), a blue save icon (save), and a clock icon (enable time). Below the toolbar is a table with the following data:

Index	Received	Tick	Level
3104	14:30:31.693	20513	IPCD/I
3105	14:30:31.693	20513	RPCD/D
3106	14:30:31.693	20514	RPCD/D
3107	14:30:31.693	20515	RPCD/I
3108	14:30:31.693	20516	ATEN/I
3109	14:30:31.693	20994	IPCD/I
3110	14:30:31.747	20994	KERN/D
3111	14:30:32.348	30727	IPCD/I
3112	14:30:32.348	31427	IPCD/I
3113	14:30:32.407	31427	KERN/D
3114	14:30:32.670	35357	NET /
3115	14:30:32.968	41213	IPCD/I
3116	14:30:32.968	41939	IPCD/I
3117	14:30:33.028	41940	KERN/D
3118	14:30:33.595	51698	IPCD/I
3119	14:30:33.645	51743	NET /
3120	14:30:33.645	52402	IPCD/I
3121	14:30:33.701	52402	KERN/D
3122	14:30:34.261	62184	IPCD/I
3123	14:30:34.261	62910	IPCD/I
3124	14:30:34.312	62910	KERN/D
3125	14:30:34.625	2622	NET /



The screenshot shows the 'CoolWatcher Developer' interface with profile 8910. The 'Main' window displays the 'Ruby Script' tab with the following log output:

```

Loading common plugin keypad...
Loading common plugin tracer...
Loading plugin Calib...
Loading plugin cwmodem2ggui...
Loading plugin fastpf...
Loading plugin romburn...
Opening Chip on COM3...
[Connection $CURRENTCONNECTION (COM3) opened (#CURRENTCONNECTION)].
[Connection RegWatch Connection (COM3) opened.].
[Connection Event Sniffer (COM3) opened.].
[Connection BufWatch Connection (COM3) opened.].
Done.
[COM OPEN OK]
Event sniffer (re)starting. (Connection: Event Sniffer (COM3))
Loading profile script : 8910.rb
EVENT SNIFFING ERROR: CONNECTION BROKEN?
EVENT SNIFFING RESURRECTED.
> activateTracerGui()
Loading Tracer GUI ...
Tracer GUI loaded.
TRACE SNIFFING ON.
TRACE SNIFFING ALREADY RUNNING
TRACE SNIFFING ALREADY RUNNING

```

The 'Trace tool' window is also open, showing a table of trace data:

Index	Received	Tick	Level
2500	14:29:11.162	10481	KERN/D
2501	14:29:11.520	17117	NET /
2502	14:29:11.658	20240	IPCD/I
2503	14:29:11.658	20513	IPCD/I
2504	14:29:11.658	20513	RPCD/D
2505	14:29:11.658	20514	RPCD/D
2506	14:29:11.658	20515	RPCD/I
2507	14:29:11.709	20517	ATEN/I
2508	14:29:11.709	21001	IPCD/I
2509	14:29:11.769	21001	KERN/D
2510	14:29:12.337	30726	IPCD/I
2511	14:29:12.337	31112	IPCD/I
2512	14:29:12.337	31112	RPCD/D
2513	14:29:12.337	31113	RPCD/D
2514	14:29:12.337	31114	RPCD/I
2515	14:29:12.337	31115	ATEN/I
2516	14:29:12.337	31564	IPCD/I
2517	14:29:12.390	31564	KERN/D
2518	14:29:12.545	33504	NET /
2519	14:29:13.000	41212	IPCD/I
2520	14:29:13.000	41914	IPCD/I
2521	14:29:13.060	41915	KERN/D

## 6.4 常见问题

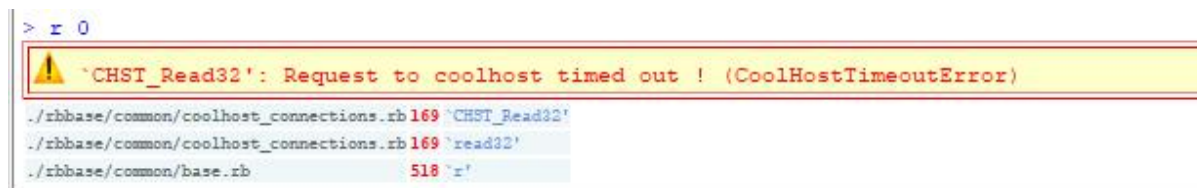
### 1 无法正常通信

一旦串口无法正常通信，就无法进行调试、输出 trace。

无法通信分为以下两种情况：

#### 1) 电脑串口异常

这种情况下，coolwatcher 工具都会无法正常打开对应串口，可以重新插拔串口尝试恢复



```
> r 0
! 'CHST_Read32': Request to coolhost timed out ! (CoolHostTimeoutError)
./rbbase/common/coolhost_connections.rb 169 'CHST_Read32'
./rbbase/common/coolhost_connections.rb 169 'read32'
./rbbase/common/base.rb 518 'r'
```

#### 2) 工具异常

在以上信息都确认没有问题的情况下，如果依然无法正常调试，可以通过【coolwatcher 命令行窗口】输入【reop】或者重启 coolwatcher 软件来恢复