



A76xx Series Open SDK_Compiled Download & Debug Guide

LTE Module

SIMCom Wireless Solutions Limited

SIMCom Headquarters Building, Building 3, No. 289 Linhong
Road, Changning District, Shanghai P.R. China

Tel: 86-21-31575100

support@simcom.com

www.simcom.com

Document Title:	A76xx Series Open SDK_Compile Download & Debug Guide
Version:	1.00
Category:	Application documentation
Status:	Released

GENERAL NOTES

SIMCOM OFFERS THIS INFORMATION AS A SERVICE TO ITS CUSTOMERS, TO SUPPORT APPLICATION AND ENGINEERING EFFORTS THAT USE THE PRODUCTS DESIGNED BY SIMCOM. THE INFORMATION PROVIDED IS BASED UPON REQUIREMENTS SPECIFICALLY PROVIDED TO SIMCOM BY THE CUSTOMERS. SIMCOM HAS NOT UNDERTAKEN ANY INDEPENDENT SEARCH FOR ADDITIONAL RELEVANT INFORMATION, INCLUDING ANY INFORMATION THAT MAY BE IN THE CUSTOMER'S POSSESSION. FURTHERMORE, SYSTEM VALIDATION OF THIS PRODUCT DESIGNED BY SIMCOM WITHIN A LARGER ELECTRONIC SYSTEM REMAINS THE RESPONSIBILITY OF THE CUSTOMER OR THE CUSTOMER'S SYSTEM INTEGRATOR. ALL SPECIFICATIONS SUPPLIED HEREIN ARE SUBJECT TO CHANGE.

COPYRIGHT

THIS DOCUMENT CONTAINS PROPRIETARY TECHNICAL INFORMATION WHICH IS THE PROPERTY OF SIMCOM WIRELESS SOLUTIONS LIMITED. COPYING, TO OTHERS AND USING THIS DOCUMENT, ARE FORBIDDEN WITHOUT EXPRESS AUTHORITY BY SIMCOM. OFFENDERS ARE LIABLE TO THE PAYMENT OF INDEMNIFICATIONS. ALL RIGHTS RESERVED BY SIMCOM IN THE PROPRIETARY TECHNICAL INFORMATION, INCLUDING BUT NOT LIMITED TO REGISTRATION GRANTING OF A PATENT, A UTILITY MODEL OR DESIGN. ALL SPECIFICATION SUPPLIED HEREIN ARE SUBJECT TO CHANGE WITHOUT NOTICE AT ANY TIME.

SIMCom Wireless Solutions Limited

SIMCom Headquarters Building, Building 3, No. 289 Linhong Road, Changning District, Shanghai P.R. China

Tel: +86 21 31575100

Email: simcom@simcom.com

For more information, please visit:

<https://www.simcom.com/download/list-863-en.html>

For technical support, or to report documentation errors, please visit:

<https://www.simcom.com/ask/> or email to: support@simcom.com

Copyright © 2023 SIMCom Wireless Solutions Limited All Rights Reserved.

Version History

Version	Date	Owner	What is new
V1.00	2022-11-21		First Edition

About this Document

This document applies to the A1803S open series, the A1603 open series and the A1606 open series.














Contents

Version History.....	3
About this Document.....	4
Contents.....	5
1 Compilation.....	6
1.1 16XX Compilation.....	6
1.1.1 Catalogue structure.....	6
1.1.2 Compilation tools.....	7
1.1.3 Compiler environment configuration.....	7
1.1.4 Compilation instructions.....	7
1.1.5 Compile generation.....	8
1.2 18XX Compilation.....	8
1.2.1 Catalogue structure.....	8
1.2.2 Compilation tools.....	8
1.2.3 Compiler environment configuration.....	9
1.2.4 Compilation instructions.....	9
1.2.5 Compile generation.....	10
2 Flashing.....	12
2.1 Installation of relevant drivers.....	12
2.2 16XX Flashing.....	13
2.3 18XX Flashing.....	14
3 Application guidance.....	17
3.1 Add Task.....	17
4 Debugging.....	20
4.1 General debugging.....	20
4.2 Log filtering.....	25
5DUMP.....	27

1 Compilation

1.1 16XX Compilation

1.1.1 Catalogue structure

	config	2022/8/9 10:55	文件夹	
	examples	2022/8/9 10:55	文件夹	
	kernel	2022/8/9 10:55	文件夹	
	out	2022/8/9 11:07	文件夹	
	sc_demo	2022/8/9 10:55	文件夹	
	sc_lib	2022/8/9 10:55	文件夹	
	script	2022/8/9 10:55	文件夹	
	tools	2022/8/9 10:55	文件夹	
	app_build_doc.md	2022/8/8 18:25	Markdown 源文件	2 KB
	CMakeLists.txt	2022/8/9 10:55	文本文档	7 KB
	makeDepend.mak	2022/8/8 17:15	MAK 文件	1 KB
	Makefile	2022/8/9 10:57	文件	18 KB
	sc_application.c	2022/7/27 19:54	C 源文件	2 KB

- **config**: some compilation options for each module regarding APP compilation, app link script, cmake compiletool configuration;
- **kernel**: the file generated by the compilation of the kernel of each module (to be relied on when packaging);
- **out**: the location of the target file generated by the compilation;
- **sc_demo**: the use case of api;
- **sc_lib**: static library and header files for the api;
- **script**: Temporarily used for packing script files;
- **tools**: compilation and packaging tools;
- **app_build_doc.md**: a brief introduction to app compilation;
- **cmakelists.txt**: cmake entry;
- **makeDepend.mak**: Makefile dependency file, see app_build_doc.md description for details;
- **makefile**: make entry;

1.1.2 Compilation tools

Tools	Role
gcc-arm-none-eabi	Cross-compilation tool-chain for compiling Apps
crc_set	Apps validation tool
7z	Compression tools
about	ASR Programming Image File Packaging Tool
cmake	cmake assembly tool for window
Ninja (optional)	Make compilation system for windows
GNUmake (default)	The make tool for Windows

1.1.3 Compiler environment configuration

Configuration items	Methods
kernel build environment configuration	Release the bin file directly without compiling
App compilation environment configuration (win32)	The compiler tool is in the tools/win32 directory; then use cmd "make+enter" to find the target
App compilation environment configuration (Linux)	For Linux, you only need to install cmake with a version number greater than 3.10 and compile it using a terminal

1.1.4 Compilation instructions

Performing actions	Instructions
Compile to generate the target file	Execute make A7670C_LANS in the root directory
Clear the compilation of a module	Execute make clean_A7670C_LANS in the root directory
Clear all modules for compilation	Execute make clean in the root directory

You can check the supported models and compile parameters by typing make or gnumake directly in the root directory. Type make all or gnumake all to list all modules and their corresponding options.

1.1.5 Compile generation

The user launches a cmd terminal in the root directory of the SDK and, depending on the model of the module, enters the corresponding model number to compile. In this case, the A7670C_LANS is used as an example.

```
S:\>gnumake A7630C_LANS
```

After execution, the A7630C_LANS will be generated in the S:\out directory

名称	修改日期	类型	大小
A7630C_LANS	2022/11/17 14:44	文件夹	

NOTE

Please do not include Chinese characters or spaces in the path.

1.2 18XX Compilation

1.2.1 Catalogue structure

名称	修改日期	类型	大小
sc_app	2022/10/26 10:05	文件夹	
sc_demo	2022/11/7 14:22	文件夹	
sc_lib	2022/10/26 10:05	文件夹	
script	2022/10/26 10:23	文件夹	
build.bat	2022/11/7 14:22	Windows 批处理...	6 KB
Makefile	2022/10/26 10:05	文件	1 KB

1.2.2 Compilation tools

Tools	Role
gcc-arm-none-eabi	Cross-compilation toolchain for compiling Apps
gnumake	makefile management tools
crc_set	Target File Validation Tool
SWD	ASR Programming Image File Packaging Tool
Python 3.8.5	Compilation environment

1.2.3 Compiler environment configuration

Configuration items	Methods
kernel build environment configuration	Release the bin file directly without compiling
app compilation environment configuration (win32)	Release the bin file directly without compiling
app compilation environment configuration (Linux)	For Linux, you only need to install cmake with a version number greater than 3.10 and compile it using a terminal

1.2.4 Compilation instructions

Performing actions	Instructions
Compile to generate the target file	Execute make A7600C_LABE_WIFI in the root directory
Clear the compilation of a module	Execute make clean_A7600C_LABE_WIFI in the root directory
Clear all modules for compilation	Execute make clean in the root directory

The user can view the supported models and compile parameters by typing make or gnumake directly into the root directory. Type make all or gnumake all to list all modules and their corresponding options.

```
X:\>make

-----
-
- build method: gnumake [target] (GIT_ID=[work id])
- target:[module](_[custom])(_app/_kernel/_force/_remake),[clean option],[install option]
-
- GIT_ID is used to dowload userspace code and tools automatically.
- show customs of the modules: gnumake [module]-list
- show all modules and customs at onece: gnumake all/info-all
-
- module list:
-   A5360G
-   A7100CE_GW
-   A7100CE_GW_B
-   A7600C_M2
-   A7600C_LABD
-   A7600C_LABE
-   A7600C_MABE
-   A7600C_TABE
-   A7602E_H_LABD
-   A7602E_H_LBBE
-   A7602E_H_TABE
-   A7602E_H_TBBE
-   A7605C_LCBE_R2
-   A7605C_TABE
-   A7605C_TCBE_R2
-   A7605E_H_TABE
-   A7605E_H_TCBE_R3
-   A7605SA_H_TABE
-   A7608E_H_LABE
-   A7608E_H_LBBE
-   A7608E_H_TABE
-   A7608E_H_TBBE
-   A7608SA_H_LABD
-   A7608SA_H_TABE
-   E9730C_LCBE
-   E9730C_MCBE
-   E9730C_TCBE
-   SIM7100CE_GW
-   SIM7100CE_GW_B
-   SIM7100CE_GW_B_SJ
-
- clean option list:
-   clean          [clean all modules.]
-   clean_app      [clean all app, target and object files.]
-   clean_kernel   [clean all kernel, just target files.]
-   clean_[module] [clean app and kernel, just kernel target files.]
-   clean_[module]_app [target and object files.]
-   clean_[module]_kernel [just target files.]
-
- install option list:
-   install_[module] [release one module to a SDK package. it will compile automatically when it not been compiled].
-   install_set      [release all modules in one SDK package, which have been compiled.]
-   uninstall_[module] [delete the SDK package for one module.]
-   uninstall_set    [delete the SDK package, which is mix some different modules.]
```

1.2.5 Compile generation

The user launches the cmd terminal in the root directory of the SDK and, depending on the model of the module, enters the corresponding model number to compile. This time uses A7600C_LABE_WIFI as an example.

```
X:\>make A7600C_LABE_WIFI
```

After execution is complete, the

```

column 32): Warning: L6314W: No section matches pattern SPI_Handle(SPI0Cmd).
\ccsw\platform\dev_plat\build\Falcon_12MB_GB15_MIFI_NezhaC_DM.sct", line 388 (c
lumn 32): Warning: L6314W: No section matches pattern SPI0(SSPCmd).
inished: 0 information, 55 warning and 0 error messages.
umake[1]: Leaving directory `Z:/tavor/Arbel/build'

*****          PASS          *****
*****          PASS          *****
*****  GNUmake ended successfully *****
*****          PASS          *****
*****          PASS          *****

*****
!!!!!!!!!!!!!!!!!!!! TAVOR build PASSED !!!!!!!!!!!!!!!!!!!!!
*****

c:\tavor\Arbel\bin>c:\tavor\Arbel\build\tavor_ddr_bin2init.exe Arbel_LWG_FALCON_M
FT_CUST_THIRCADX_DEV2.bin Arbel_LWG_FALCON_MIFI_CUST_THIRCADX_DEV2.init 32
one
*****          TRANSLATE          *****

```

A folder named A7630C_LABE_WIFI_SWD will be generated.

名称	修改日期	类型	大小
 A7600C_LABE_WIFI_SWD	2022/9/14 17:58	文件夹	

NOTE

Please do not include Chinese characters or spaces in the path.

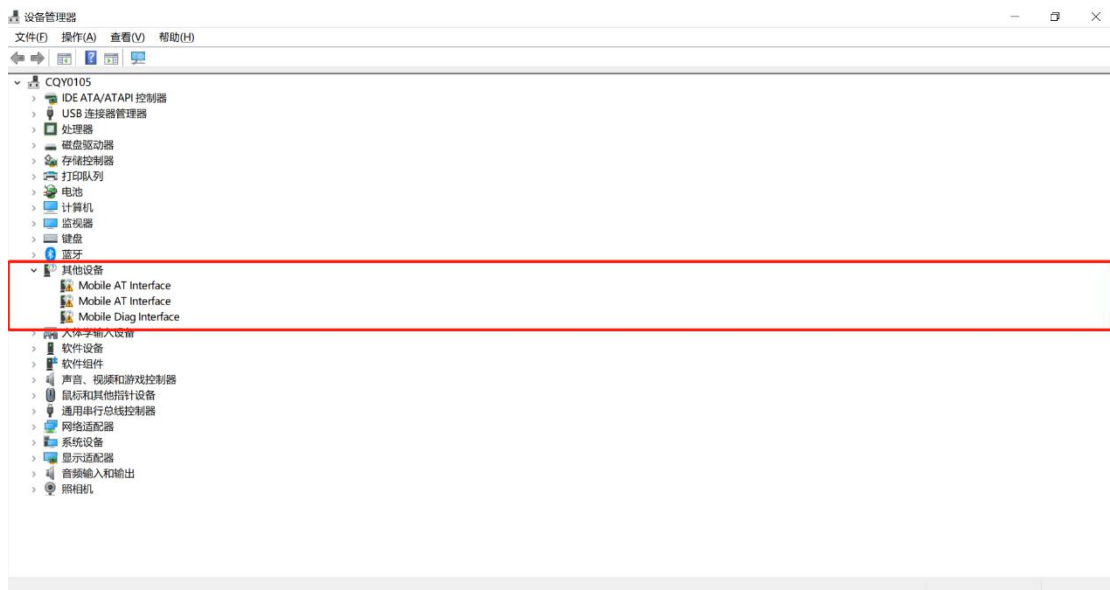
2 Flashing

2.1 Installation of relevant drivers

SIMCom provides the relevant drivers for USB as follows:

名称	修改日期	类型	大小
Windows7	2019/7/9 14:47	文件夹	
Windows8	2019/7/9 14:47	文件夹	
Windows10	2019/7/9 14:47	文件夹	
XP-Vista	2019/7/9 14:47	文件夹	

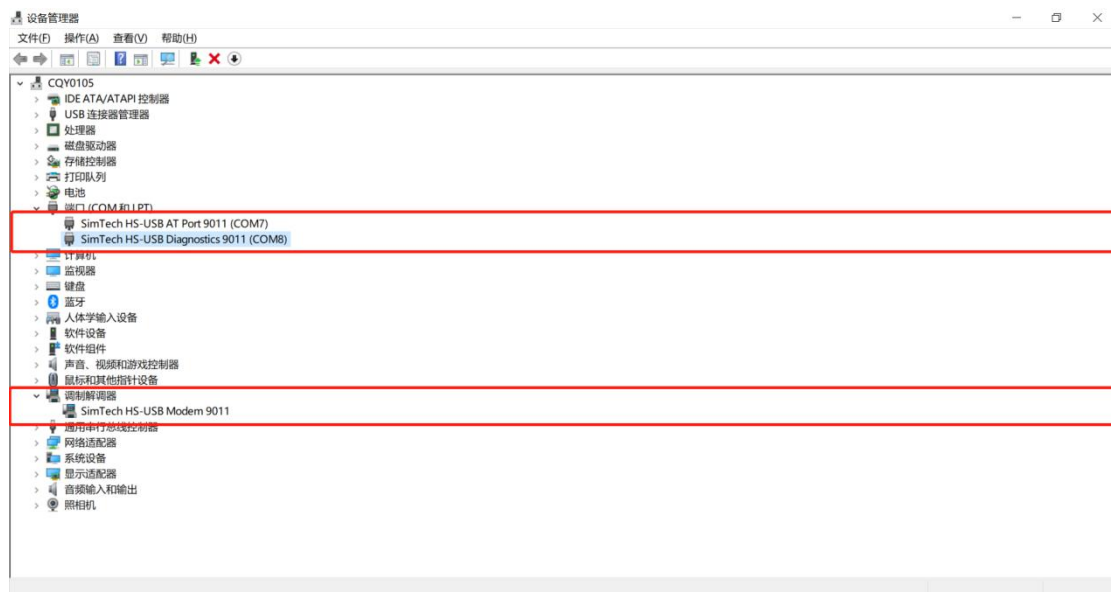
If the PC does not have a driver installed, the following devices will be prompted to install a driver when plugged into the USB



Step 1: Scan your computer for updated drivers with our supported drivers.



Step 2: Install the drivers in the following order until they are all successfully installed.



2.2 16XX Flashing

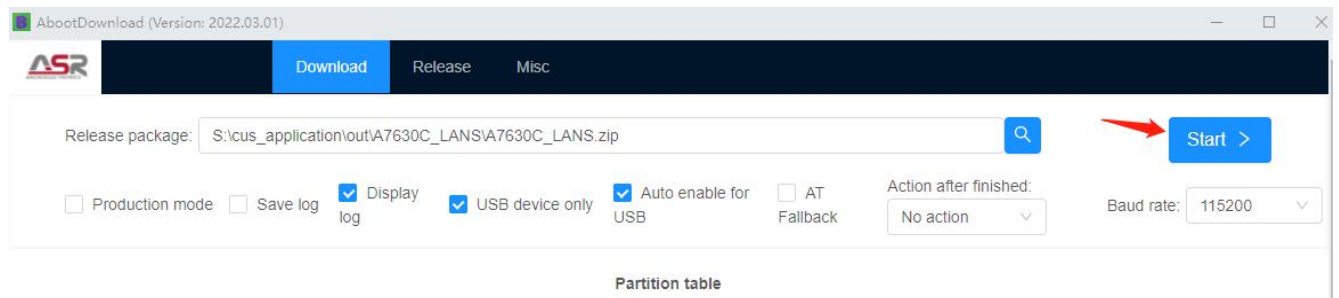
Step 1: Open about.exe

Step 2: Selecting the Flashing file

名称	修改日期	类型	大小
CMakeFiles	2022/11/17 14:44	文件夹	
lib	2022/11/17 14:44	文件夹	
sc_demo	2022/11/17 14:44	文件夹	
sc_lib	2022/11/17 14:44	文件夹	
A7630C_LANS.zip	2022/11/17 14:44	WinRAR ZIP 压缩...	11,675 KB
cmake_install.cmake	2022/11/17 14:44	CMAKE 文件	5 KB
CMakeCache.txt	2022/11/17 14:44	文本文档	16 KB
customer_app.bin	2022/11/17 14:44	BIN 文件	157 KB
customer_app.elf	2022/11/17 14:44	ELF 文件	283 KB
customer_app.elf.map	2022/11/17 14:44	MAP 文件	378 KB
Makefile	2022/11/17 14:44	文件	9 KB

选取此文件

Step 3: Click Start to start Flashing



2.3 18XX Flashing

Please ensure that the firmware package has been compiled before flashing. Currently, a separate flashing APP and a complete flashing firmware package are available.

Please ensure that each step is completed successfully, otherwise an error may occur. Detailed steps are shown below.

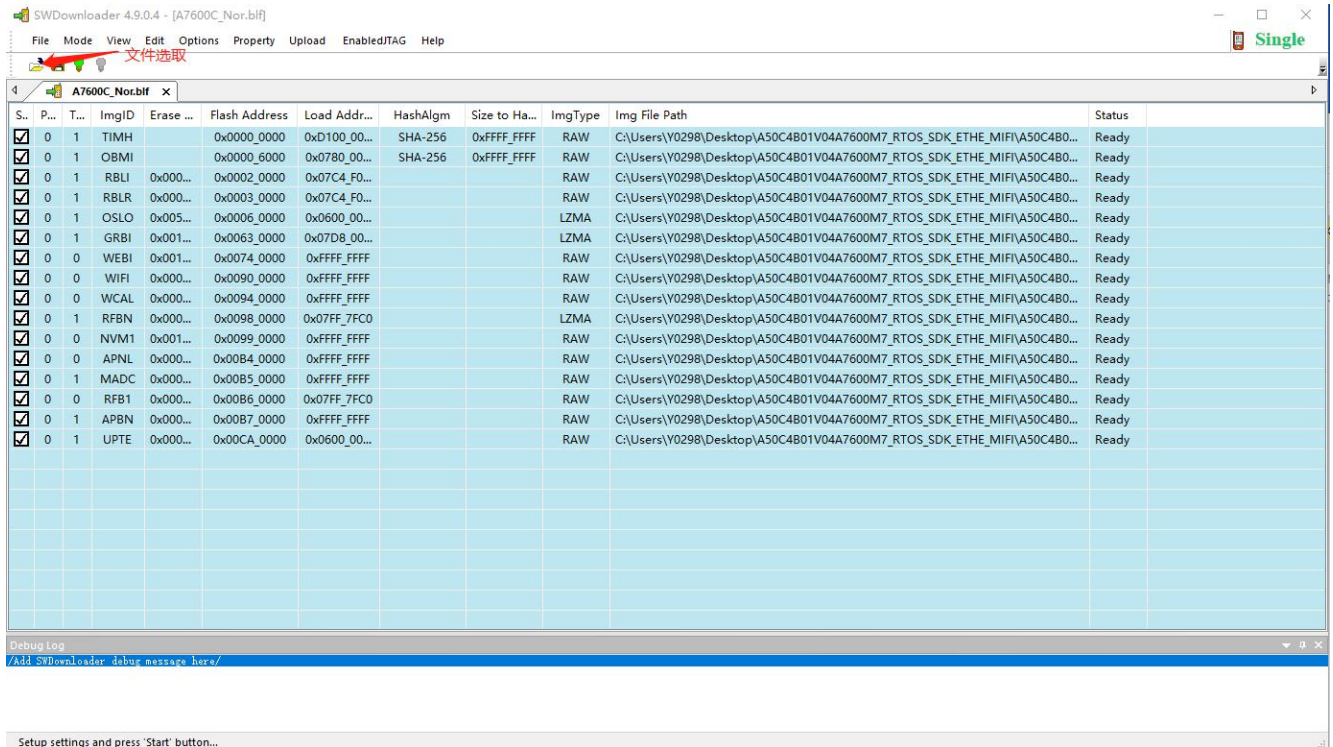
Step 1: Check the firmware package.

The compiled A7630C_LABE_WIFI_SWD file contains all the firmware packages needed for flashing.

名称	修改日期	类型	大小
A7600C_Nor.blf	2022/9/14 17:42	BLF 文件	17 KB
A7600C_Nor_Factory_only_Erase_All.blf	2022/9/14 17:42	BLF 文件	17 KB
AdditionalAPN.bin	2022/5/20 18:19	BIN 文件	64 KB
cp_lzma.bin	2022/9/14 17:56	BIN 文件	5,892 KB
customer_app.bin	2022/9/14 17:56	BIN 文件	111 KB
dsp_ADC.bin	2022/5/20 18:19	BIN 文件	64 KB
Falcon_loader_EVB_QSPI_Nor_PM803....	2022/6/14 14:38	BIN 文件	89 KB
FALCON_LWG_M13_A0_Flash_lzma_as...	2022/9/14 17:42	BIN 文件	1,046 KB
HERON_CALIFW_A0.bin	2022/5/20 18:19	BIN 文件	85 KB
HERON_FMACFW_A0.bin	2022/5/20 18:19	BIN 文件	152 KB
ntim_ddr.bin	2022/5/20 18:19	BIN 文件	113 KB
nvm.bin	2022/6/14 14:38	BIN 文件	0 KB
ReliableData.bin	2022/6/14 14:38	BIN 文件	64 KB
rf_lzma_asr.bin	2022/6/14 14:38	BIN 文件	3 KB
updater.bin	2022/5/20 18:19	BIN 文件	65 KB
WebData.bin	2022/5/20 18:19	BIN 文件	1,112 KB

Step 2: Open SWDownloader.exe.

Step 3: Select the firmware package.



NOTE

A76xxx_Nor.blf

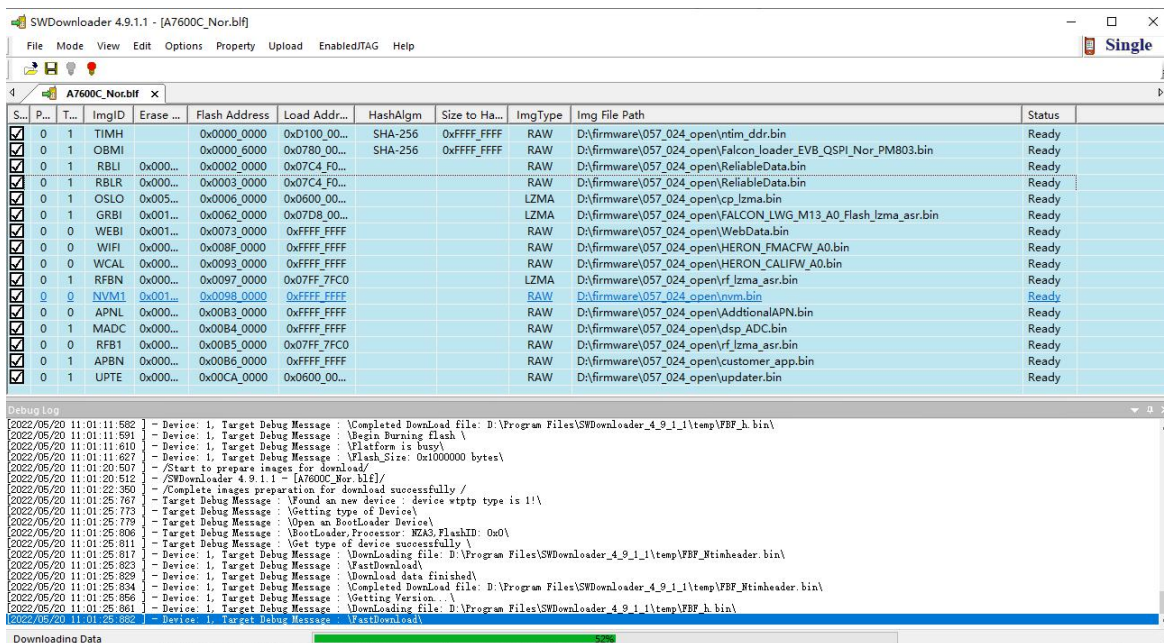
- - Delete only the selected partition and upgrade

A76xxx_Nor_Factory_only_Erase_All.blf

- - Delete all flashes and upgrade

A76xxx_Nor_Factory_only_Erase_All.blf will erase all flash, including RD data and factory calibration data, such as IMEI. **Do not perform this operation in a production environment!** A76xxx_nor.blf is recommended for software upgrades.

Step 4: Click on the "Start" button to begin the download, then press POWER_KEY to download.



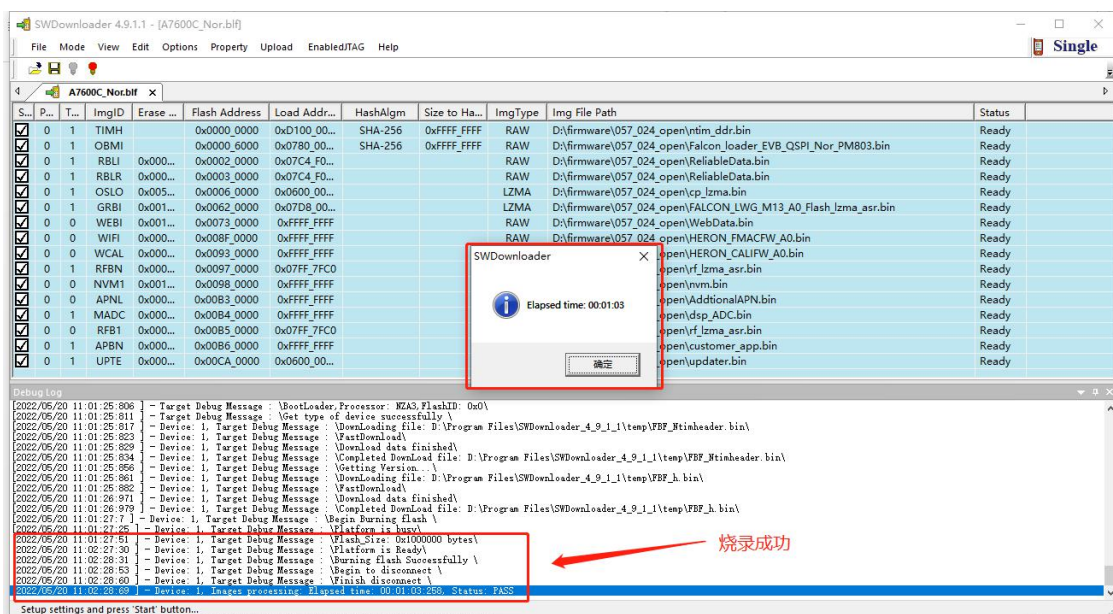
The screenshot shows the SWDownloader 4.9.1.1 interface. The main window displays a list of files to be downloaded, including TIMH, OBMI, RBLI, RBLR, OSLO, GRBI, WEBI, WIFR, WICAL, RFBN, NVMI, APNL, MADC, RFB1, APBN, and UPTE. Each file has a status of 'Ready'. The debug log at the bottom shows the progress of the download, including messages like 'Completed Download file', 'Begin Burning flash', and 'Platform is busy'.

NOTE

If the progress bar does not appear, try repeating step 4.

Step 5: If the download is successful, the following will be displayed.

Please remember to click on the "Back" button, otherwise the module may trigger the download mode again when it is re-powered.



The screenshot shows the SWDownloader 4.9.1.1 interface after a successful download. A dialog box is displayed in the center, indicating 'Elapsed time: 00:01:03'. The debug log at the bottom shows the progress of the download, including messages like 'Completed Download file', 'Begin Burning flash', and 'Platform is busy'. A red arrow points to the '烧录成功' (Burn Success) message in the log.

3 Application guidance

3.1 Add Task

Step 1: Add a task to the `sAPP_HelloWorldDemo()` function in the subdirectory `demo_helloworld.c`. To add a task you need to use `sAPI_TaskCreate()`, in addition, the structure registered by the entry function allows you to set the priority, stack and other information of the entry function, for more detailed operation of the task Please refer to the document "A76xx Series Open SDK_RTOS Development_Application Guide" on how to use task.

Reference `demo_helloworld.c` to create the task.

```
16 #include "simcom_os.h"
17 #include "simcom_debug.h"
18 #include "stdio.h"
19
20
21 sTaskRef helloWorldProcesser;
22 static UINT8 helloWorldProcesserStack[1024];
23 #if 0
24 /**
25  * @brief simcom_printf
26  * @param pointer *format
27  * @note The message will be output by sAPI_Debug and captured by CATSTUDIO tool.
28  * @retval void
29  */
30 void simcom_printf(const char *format,...){
31     char tmpstr[200];
32
33     va_list args;
34
35     memset(tmpstr,0,sizeof(tmpstr));
36
37     va_start(args,format);
38     sAPI_Vsnprintf(tmpstr,sizeof(tmpstr),format,args);
39     va_end(args);
40
41     sAPI_Debug("simcom_printf [%s]",tmpstr);
42 }
43 #endif
44
```

```

51 void sTask_HelloWorldProcesser(void* argv)
52 {
53     ...sAPI_Debug("Task runs successfully");
54     ...//simcom_printf("%s-%d-%f", "simcom", 2020, 10.23);
55 }
56
57
58
59 /**
60  * @brief helloworld task initial
61  * @param void
62  * @note
63  * @retval void
64  */
65 void sAPP_HelloWorldDemo(void)
66 {
67     ...SC_STATUS status = SC_SUCCESS;
68
69     ...status = sAPI_TaskCreate(&helloWorldProcesser, helloWorldProcesserStack, 1024, 150, "helloWorldProcesser", sTask_HelloWorldProcesser, (void *)0);
70     if(SC_SUCCESS != status) 利用sAPI_TaskCreate创建task
71     {
72         ...sAPI_Debug("Task create fail, status = [%d]", status);
73     }
74 }

```

task优先级 task主函数

Step 2: Add the task creation function to the userSpace_Main() function in the sc_application.c in the root directory.

Reference sc_application.c

```

49 void userSpace_Main(void* arg)
50 {
51     .../*simcom api init. Do not modify!*/
52     ...ApiMapInit(arg);
53     ...//sAPI_enableDUMP();
54     .../*UI demo task for customer with CLI method for all demo running,
55     ...customer need to define SIMCOM_UI_DEMO_TO_USB_AT_PORT or
56     ...SIMCOM_UI_DEMO_TO_UART1_PORT to select hardware interface.
57     ...*/
58     ...sleep(5); //Wait for USB initialization to complete and print catstudio log.
59
60     ...sAPP_SimcomUIDemo();
61
62     ...sAPP_UartTask();
63     ...sAPP_UrcTask();
64     ...sAPP_UsbVcomTask();
65     ...sAPP_HelloWorldDemo();
66     ...//sAPI_FotaCallback(fotacb);
67     ...printf("user app is running...");
68 }
69 }

```

Step 3: Add the directory name and add the link library to CMakeLists.txt in the root directory.

Refer to CMakeLists.txt in the root directory.

```

38  # 编译子目录
39  if(NOT DEFINED SIMCOM_SDK)
40  if(DEFINED QL)
41  add_subdirectory(./ql_lib ql_lib)
42  else()
43  add_subdirectory(./sc_lib sc_lib)
44  endif()
45  endif()
46
47  if(DEFINED QL)
48  add_subdirectory(./ql_demo ql_demo)
49  else()
50  add_subdirectory(./sc_demo sc_demo)
51  endif()

```

```

117  if(DEFINED QL)
118  if(NOT DEFINED SIMCOM_SDK)
119  target_link_libraries(userspace PRIVATE ql_lib)
120  endif()
121  target_link_libraries(userspace PRIVATE ql_demo)
122  else()
123  if(NOT DEFINED SIMCOM_SDK)
124  target_link_libraries(userspace PRIVATE sc_lib)
125  endif()
126  target_link_libraries(userspace PRIVATE sc_demo)
127  endif()

```

Step 4: Create a new CMakeLists.txt in the subdirectory and add the corresponding compiled source files and dependent header files paths.

Refer to CMakeLists.txt in the subdirectory.

```

1  cmake_minimum_required(VERSION 3.10)
2
3  AUX_SOURCE_DIRECTORY(./src sc_demo_src)
4  AUX_SOURCE_DIRECTORY(./src/token sc_demo_src)
5  AUX_SOURCE_DIRECTORY(./src/utils sc_demo_src)
6
7
8  # Add the static library
9  add_library(sc_demo STATIC ${sc_demo_src})
10 target_include_directories(sc_demo PUBLIC
11  ./inc
12  ${CMAKE_SOURCE_DIR}/sc_lib/inc
13  ${CMAKE_SOURCE_DIR}/sc_lib/${SCMODULE}/inc
14  ${CMAKE_SOURCE_DIR}/sc_lib/inc/GPIO
15  ${CMAKE_SOURCE_DIR}/sc_lib/inc/token
16  ${CMAKE_SOURCE_DIR}/sc_lib/inc/utils
17  ${CMAKE_SOURCE_DIR}/sc_demo/inc
18 )
19
20 SET(LIBRARY_OUTPUT_PATH ${PROJECT_BINARY_DIR}/lib)

```

4 Debugging

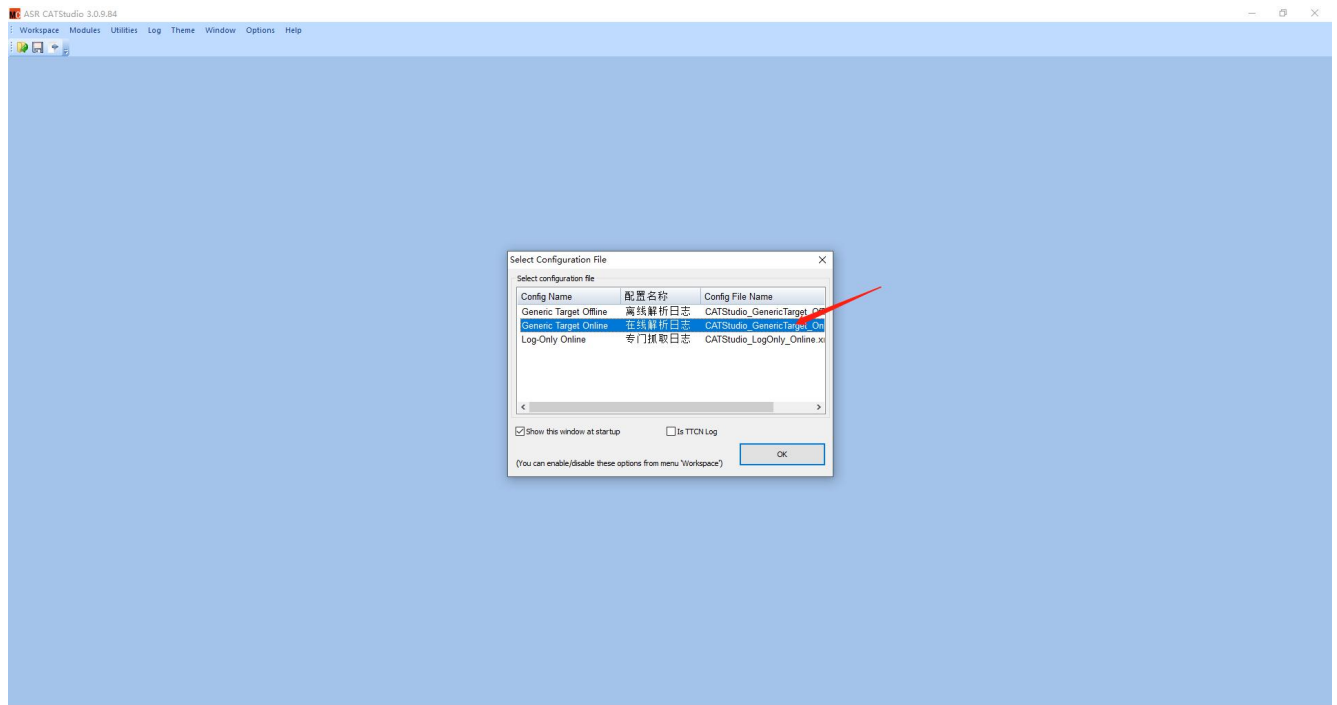
We support the use of CATStudio for debugging.

NOTE

ACAT logs are divided into online and offline logs.

4.1 General debugging

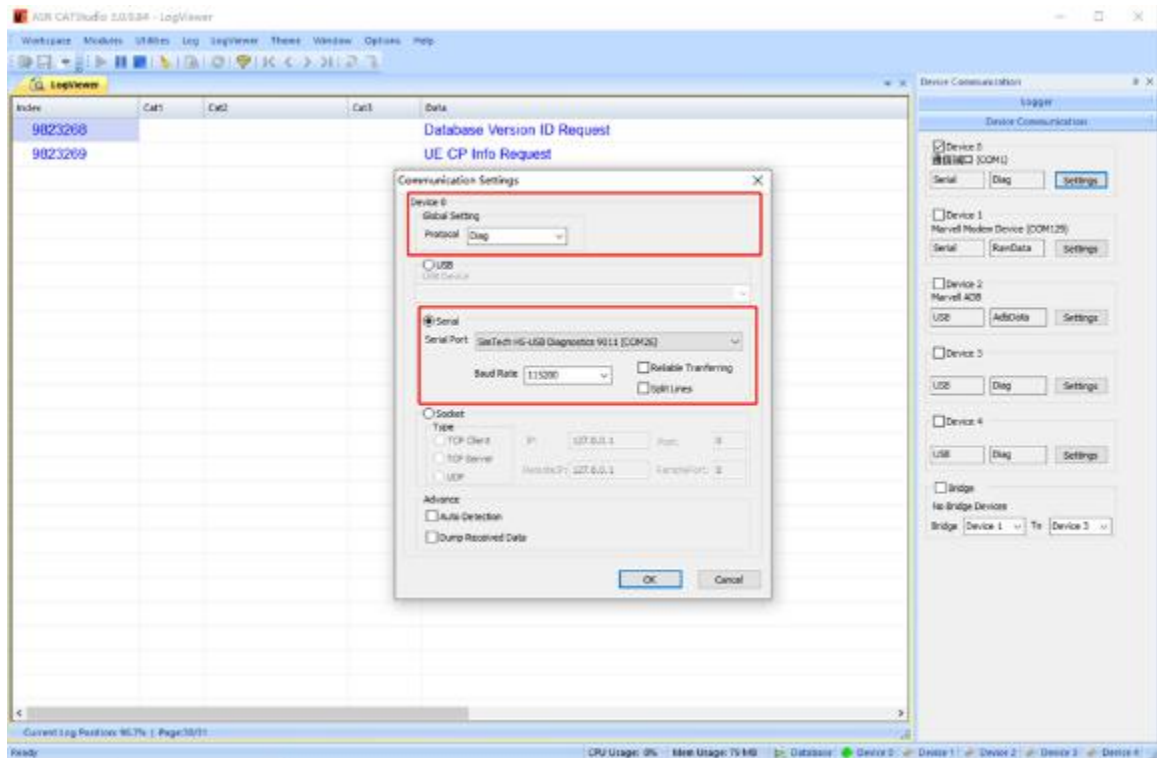
Step 1: Run CATStudio, if CATStudio runs successfully, the configuration file will be required, select the required file and click "OK".



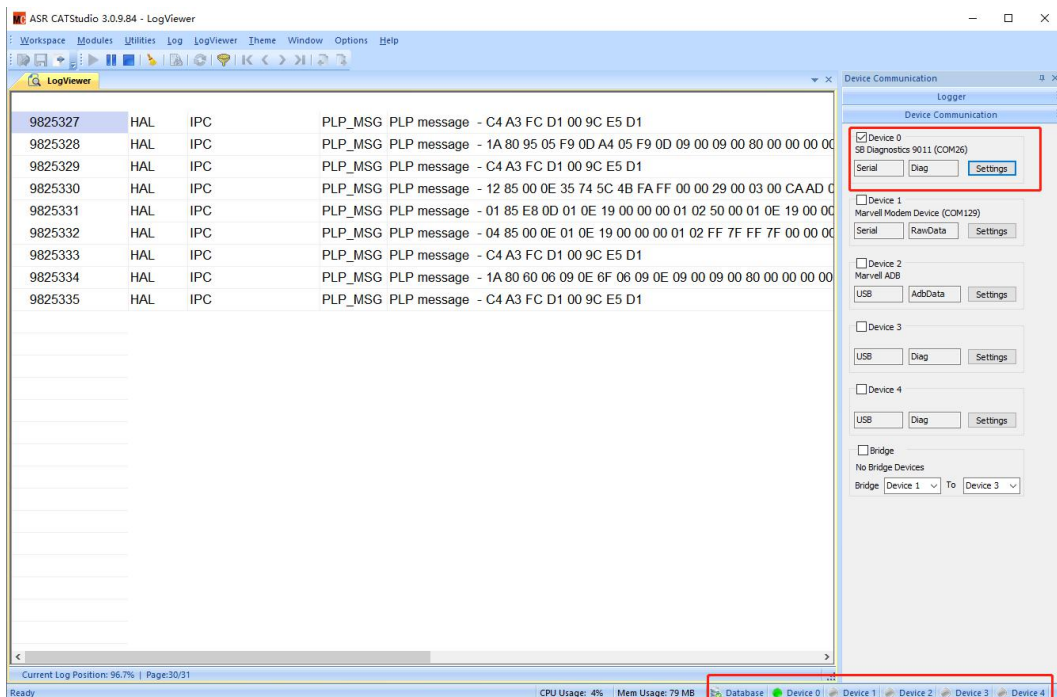
NOTE

To parse the log online, please select CATStudio_LWG_Online.xml.
To parse the log offline, please select CATStudio_LWG_Offline.xml.

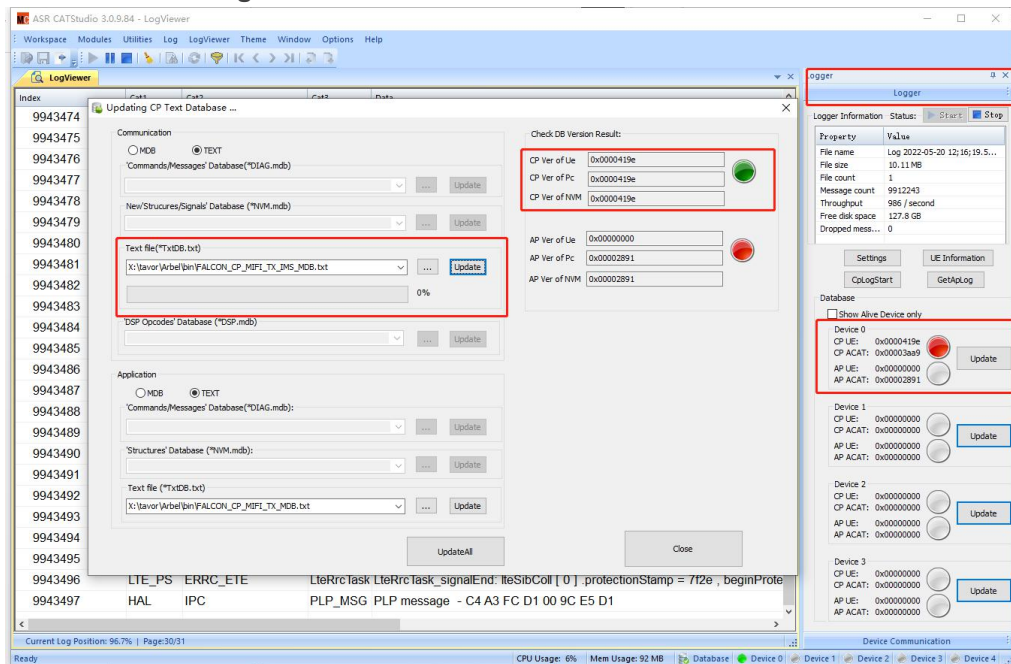
Step 2: As shown in the figure, click on the setting of a device number in "Device Communication" and select the connected device, e.g. you can configure it as Device0 output log.



Step 3: Check the device selected in the previous step, showing green when the connection is successful and red when the connection fails.



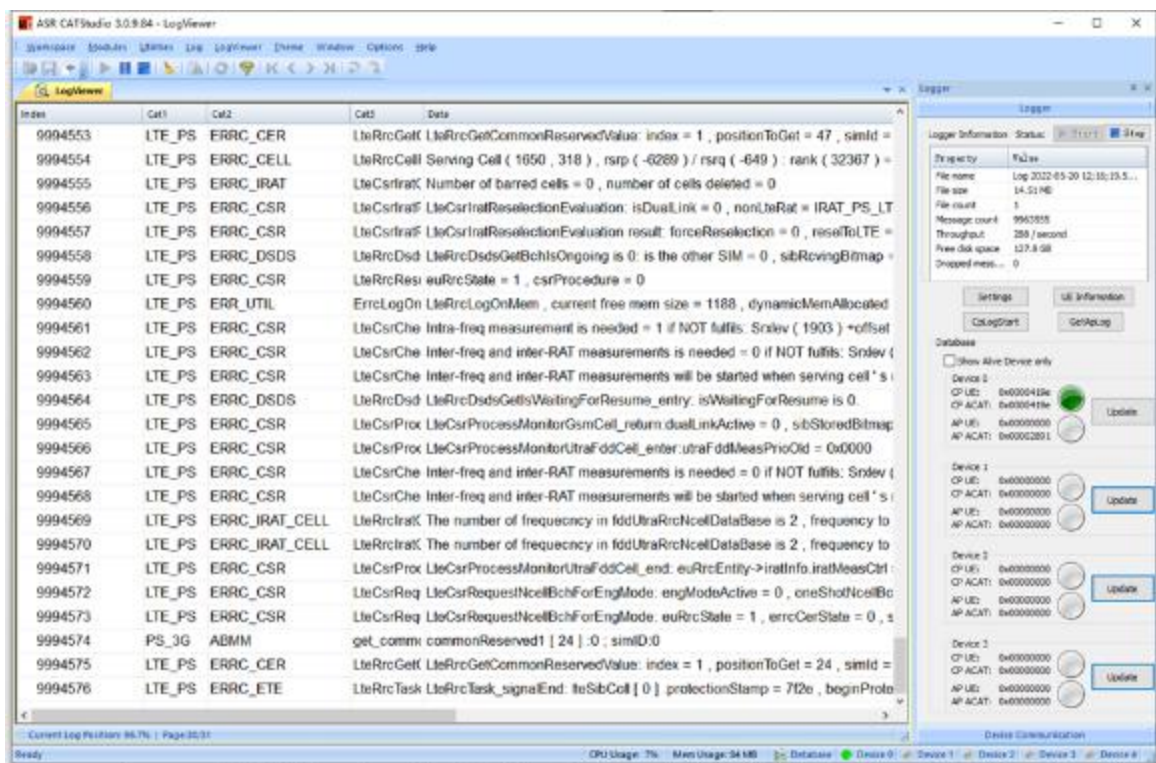
Step 4: Open the Logger panel and update the DB (the cp_MDB.txt file is stored in the \tavor\Arbel\bin directory). If the .txt is an exact match to the cp downloaded in the module, the DB version result will be shown in green.



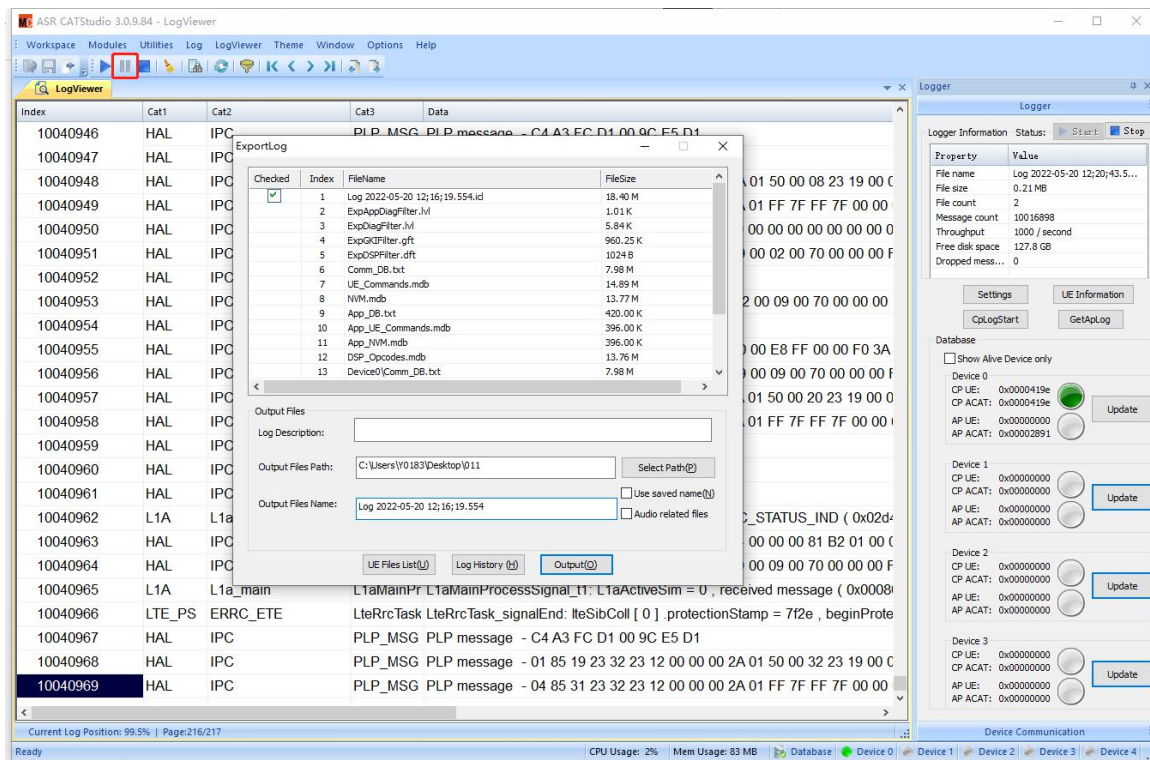
NOTE

Please ensure that the DB and CP.bin match exactly, otherwise some logs cannot be analysed.

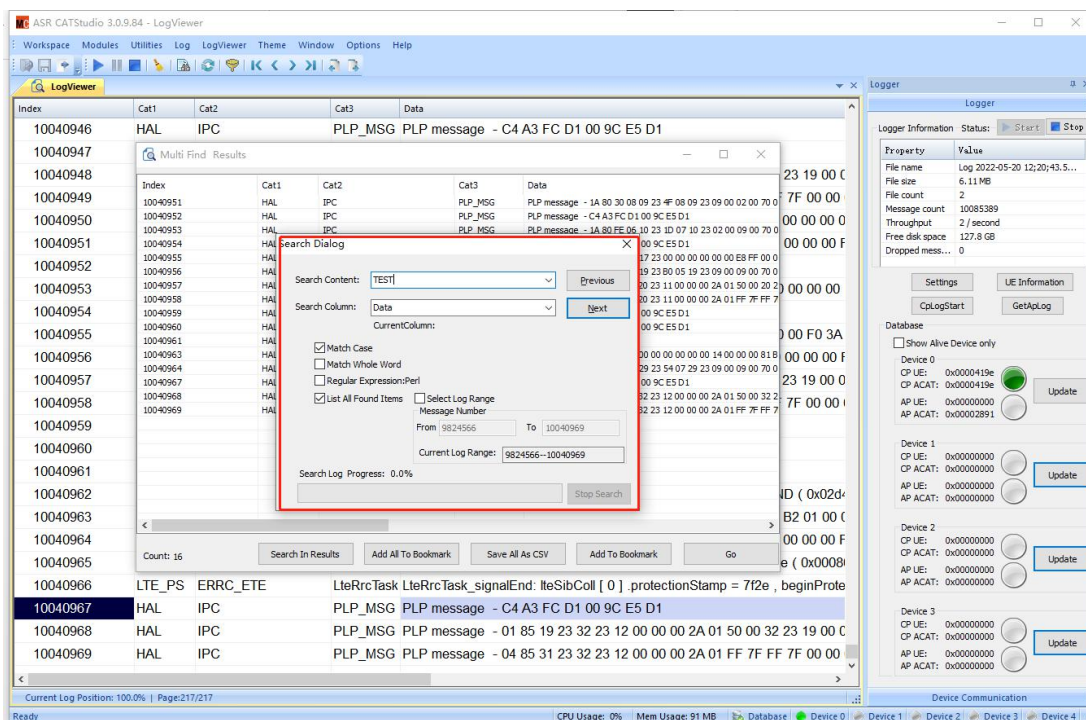
Step 5: Click on CATStudio Menu: Modules->LogViewer to open the LogViewer panel, where the logs are displayed.



Step 6: To save the logs by stopping the export, click on Menu:LOG->Log-> Export LOG- file in CATStudio and click on the "output" button to save the logs in zip format to the specified path.



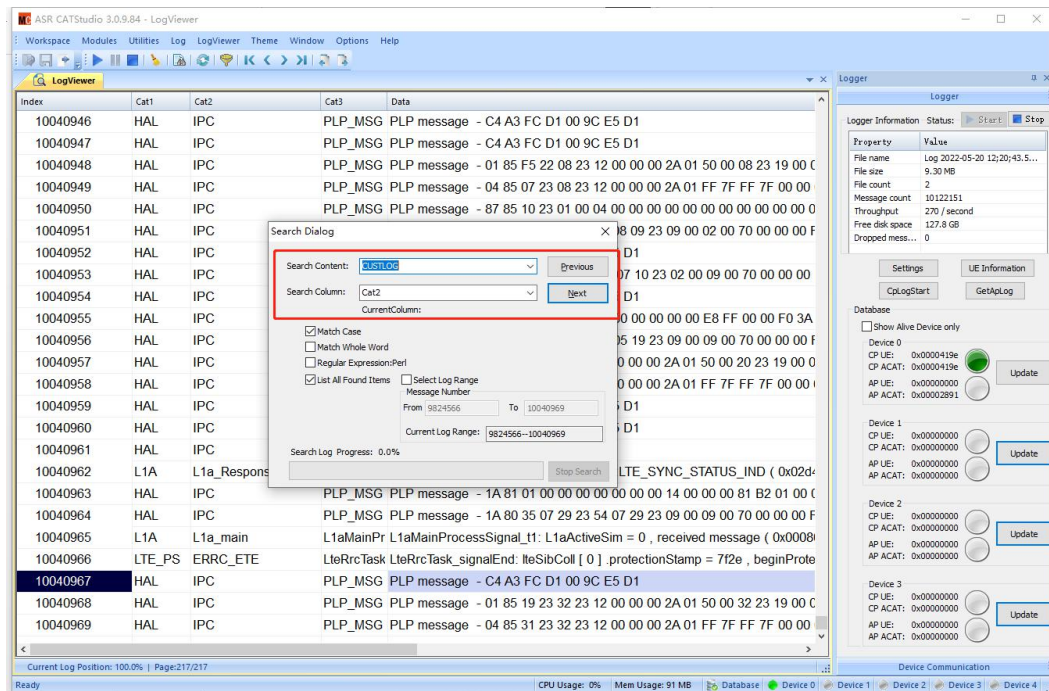
Step 7: When the output stops, you can search for the specified information by using CTRL+F. By selecting "Previous" or "Next" you can search up or down.



NOTE

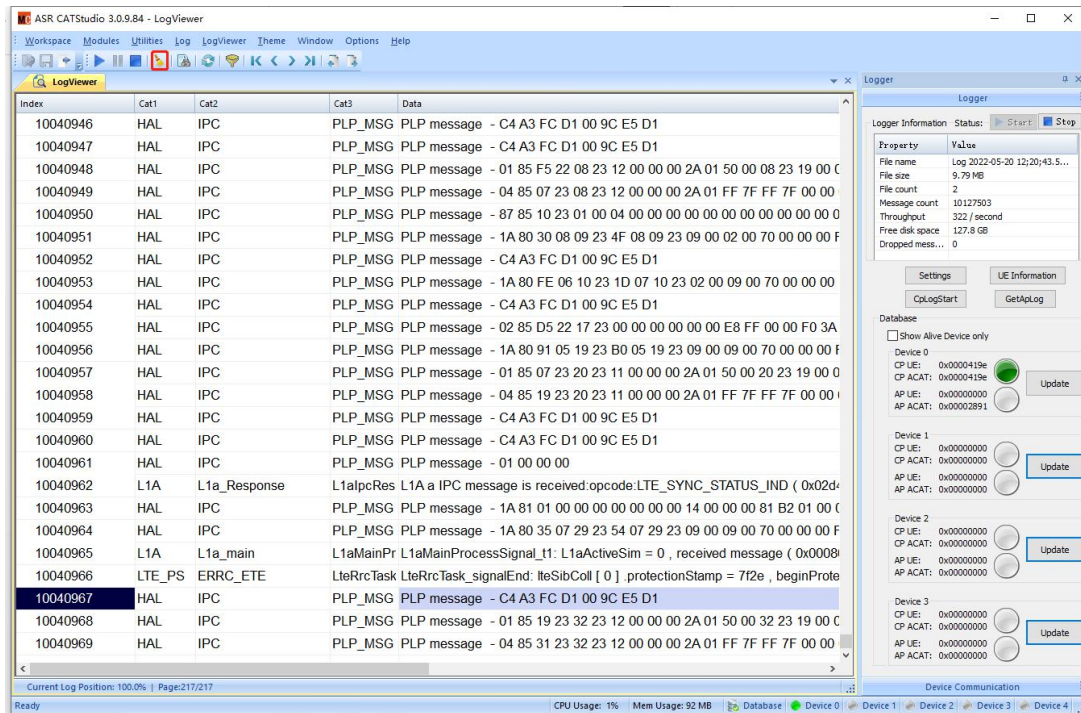
CATStudio supports filtering search results and searching within the results in the "Find Results" panel.

Step 8: Use CTRL+F to get the api sAPI_Debug output log and set the conditions as shown below.



Step 9: If there are no problems with the test results over a period of time, the logs so far are probably useless, so click the Clear button to clear the logs.

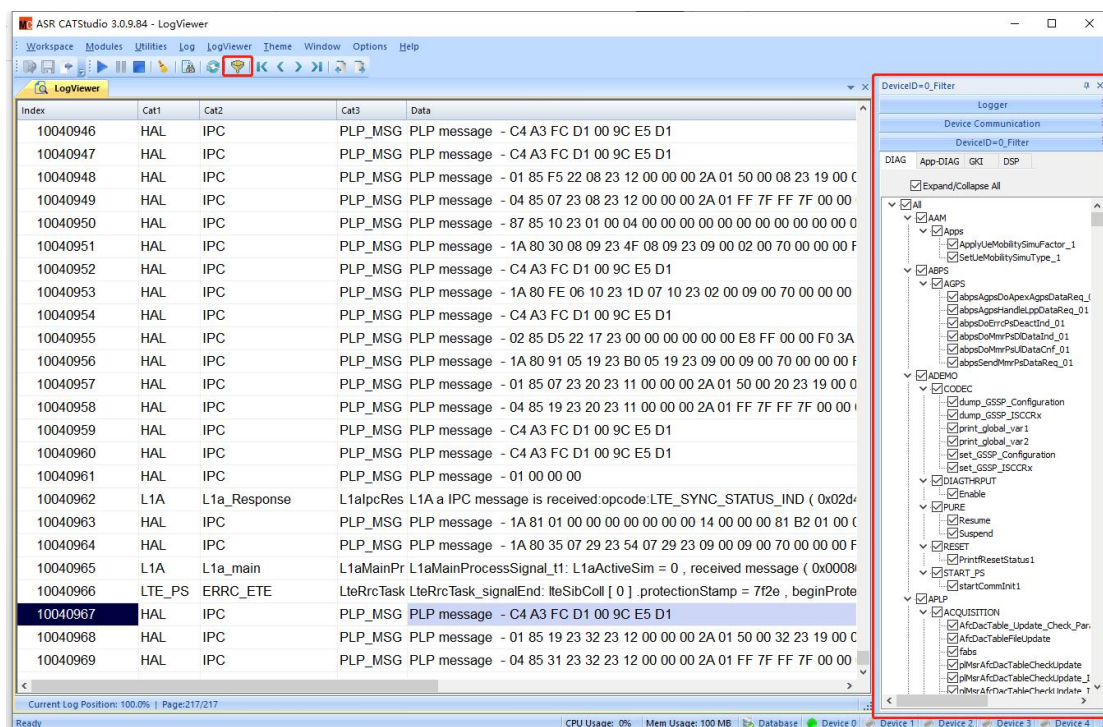
This operation facilitates analysis as the log file is not too large and useless information is not displayed in the log.



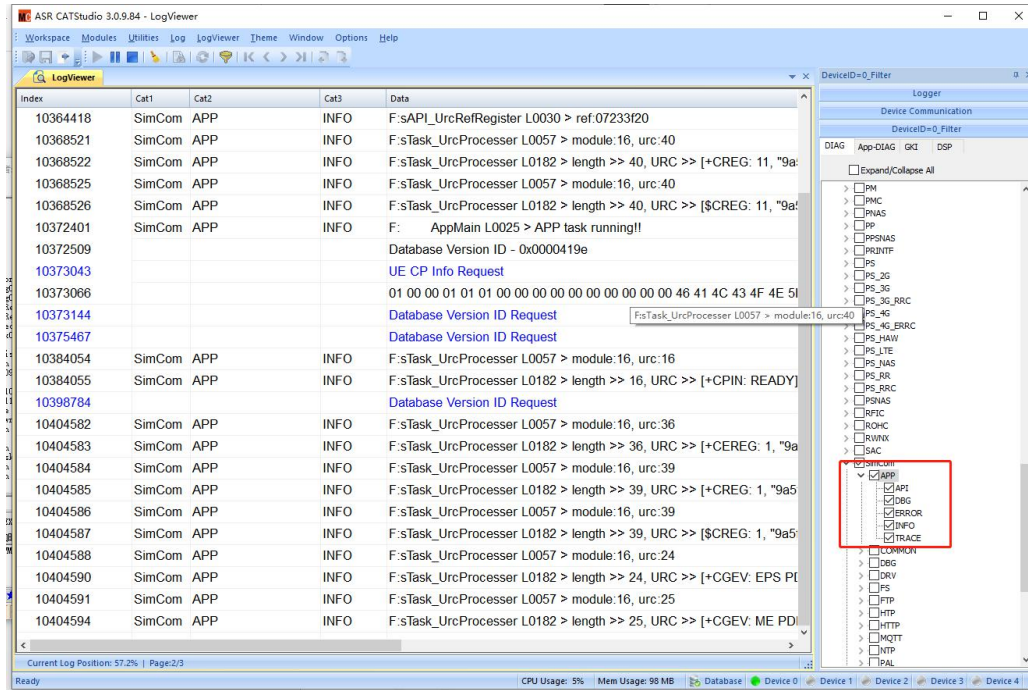
4.2 Log filtering

CATStudio supports log filtering, CATStudio prints all logs by default, too many logs will affect debugging efficiency, please refer to the following configuration to print only the sAPI_Debug interface. The procedure is as follows.

Step 1: Click on the Show Filter button as shown below.



Step 2: Click on the DIAG button to check CUST only and remove the option for App-DIAG GKT DSP, at this point you will only see the sAPI_Debug interface print



NOTE

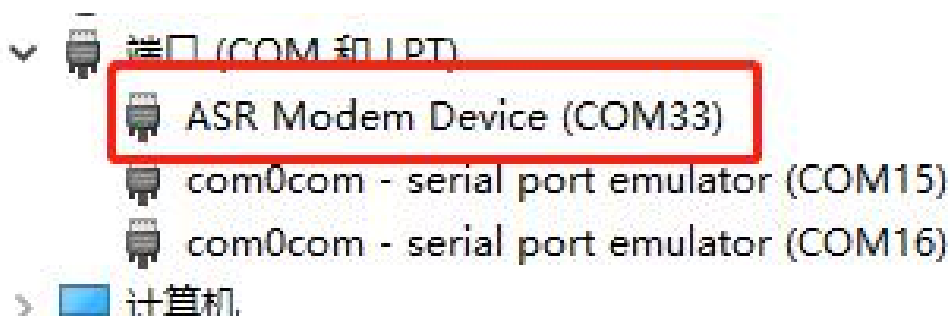
1. The logs will be automatically saved in the Exec\Bin Logs directory of the installation path.
2. Please clean up useless logs so as not to run out of disk space. Also, the .icl under the path can be opened by CATStudio in offline mode and has perfectly matching DB files.

5DUMP

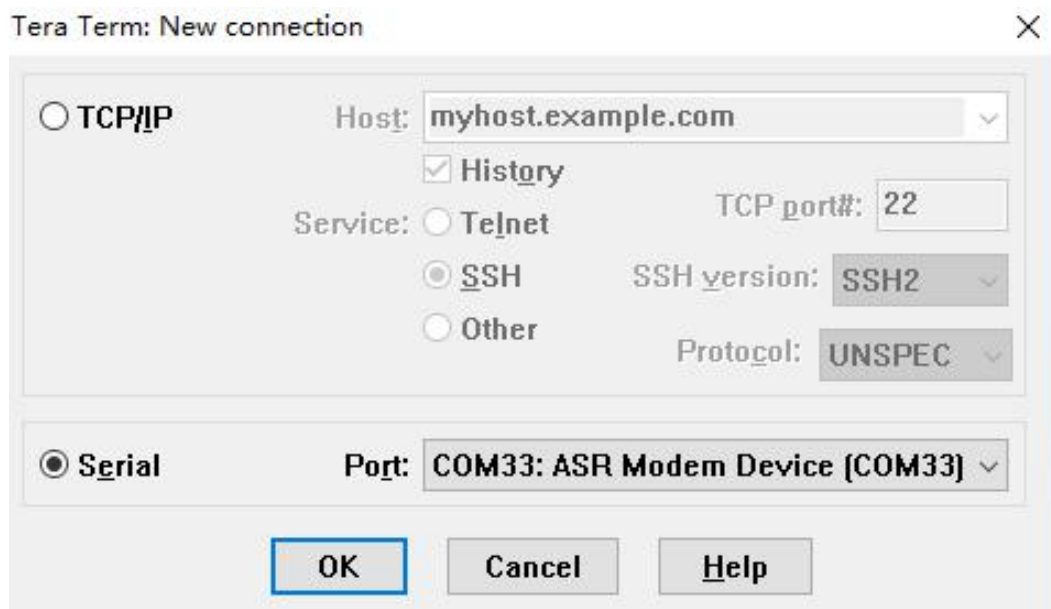
When the module is running, for some reason (memory application failure, illegal pointer access, stack overflow, etc.) it goes into dump mode. In this case the dump information needs to be exported for analysis.

Step 1: Open ttermpro.exe.

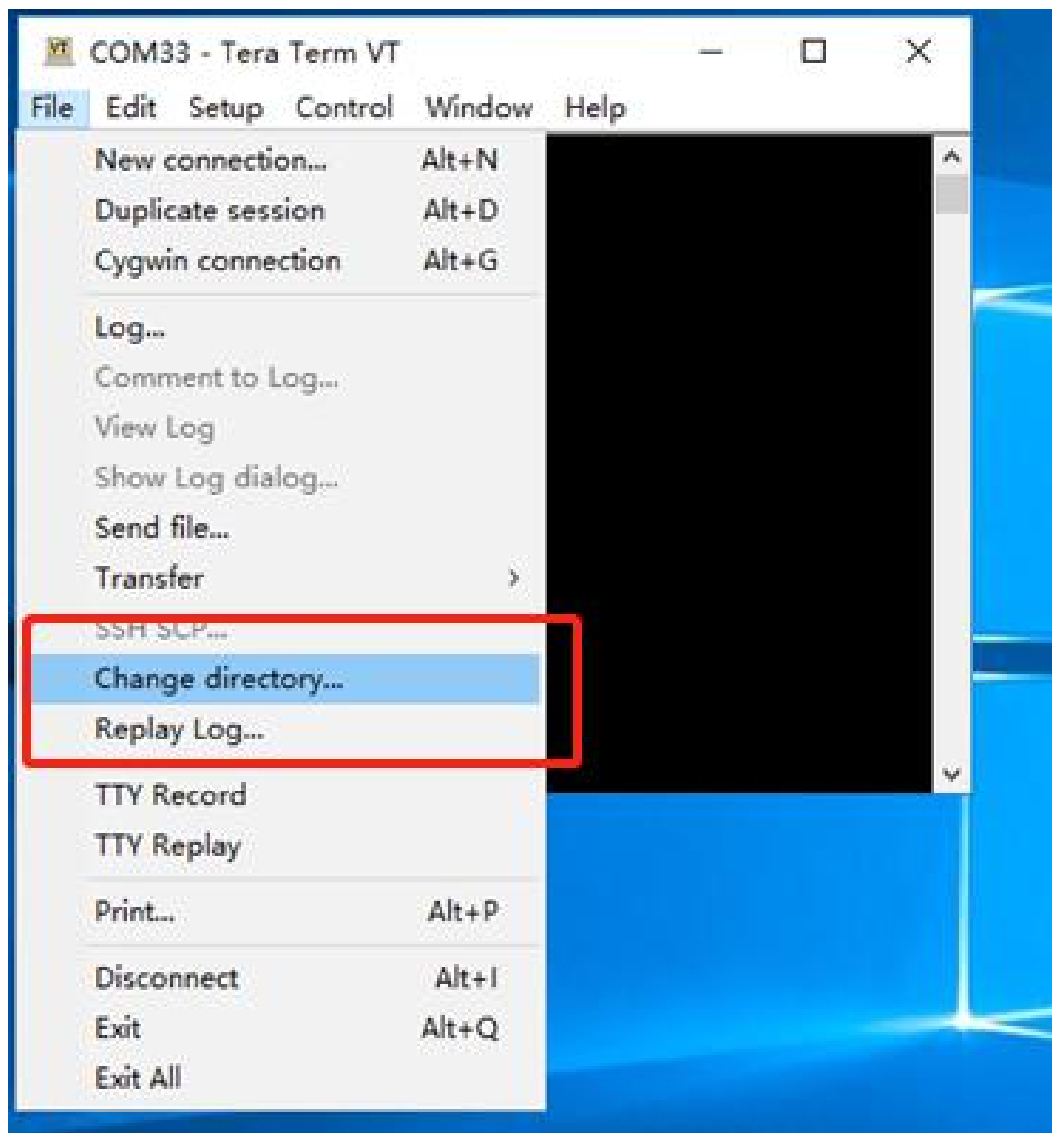
Step 2: Confirm the com number of the ASR Modem device port number in the Device Manager.



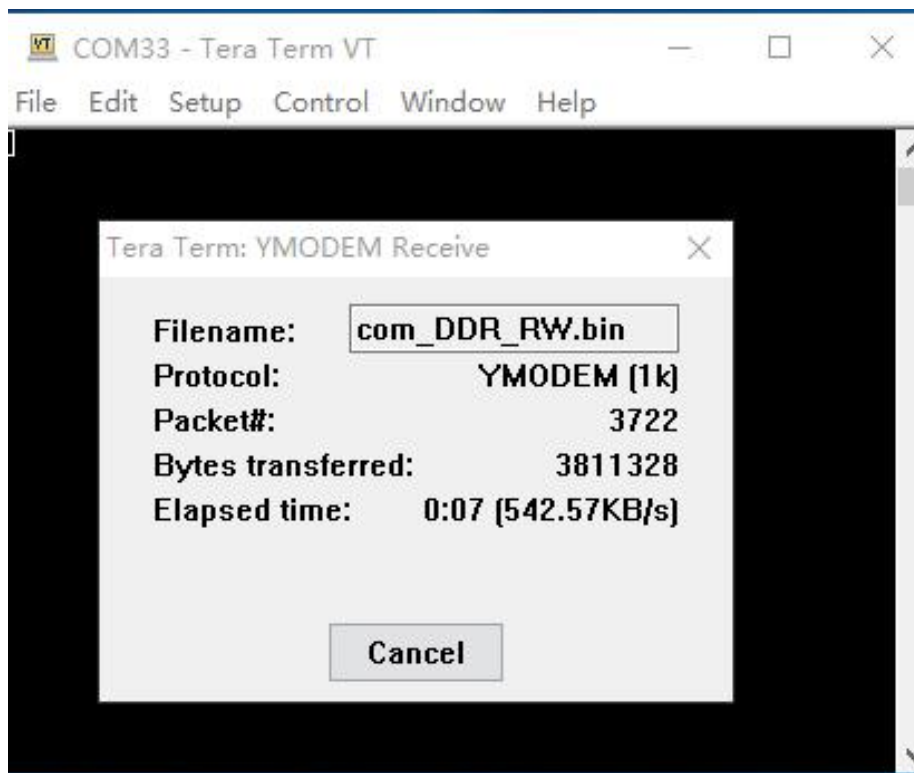
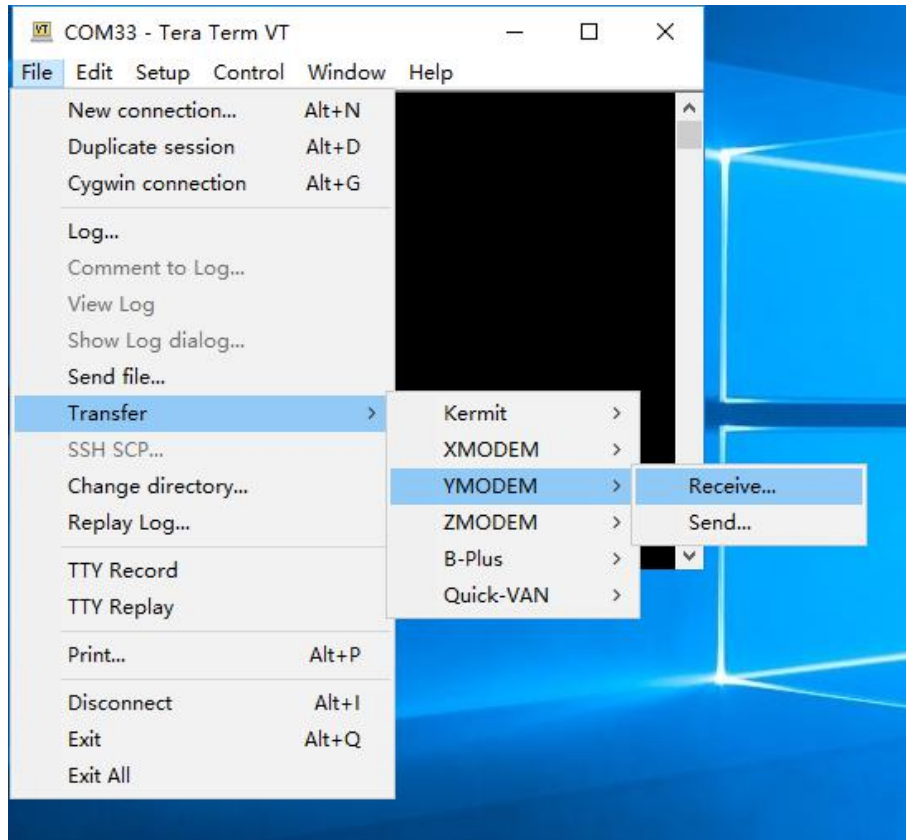
Step 3: Open ttermpro and select the com port to grab the dump.



Step 4: "Change directory" is used to select the location where the dump file is to be saved.



Step 5: Select the ymodem protocol for receiving the dump file.



Step 6: Check the received dump file.

com_compile.bin	2020/7/28 16:44	BIN 文件	1 KB
com_CustVer.bin	2020/7/28 16:44	BIN 文件	1 KB
com_DDR_RW.bin	2020/7/28 16:44	BIN 文件	14,272 KB
com_DSP_DDR.bin	2020/7/28 16:44	BIN 文件	2,112 KB
com_DSPdram.bin	2020/7/28 16:44	BIN 文件	256 KB
com_DTCM.bin	2020/7/28 16:44	BIN 文件	64 KB
com_dts5py.bin	2020/7/28 16:44	BIN 文件	2 KB
com_dtfm5py.bin	2020/7/28 16:44	BIN 文件	1 KB
com_EE_Itbui.bin	2020/7/28 16:44	BIN 文件	1 KB
com_Gpio5py.bin	2020/7/28 16:44	BIN 文件	26 KB
com_ITCM.bin	2020/7/28 16:44	BIN 文件	64 KB
com_I1x5py.bin	2020/7/28 16:44	BIN 文件	6 KB
com_L2_sram.bin	2020/7/28 16:44	BIN 文件	128 KB
com_PS_DAT.bin	2020/7/28 16:44	BIN 文件	448 KB
com_rj_reln.bin	2020/7/28 16:44	BIN 文件	1 KB
com_SQU.bin	2020/7/28 16:44	BIN 文件	64 KB
com_usbintD.bin	2020/7/28 16:44	BIN 文件	1 KB
com_usbintg.bin	2020/7/28 16:44	BIN 文件	32 KB
com_usbRse.bin	2020/7/28 16:44	BIN 文件	1 KB
com_wdtKICK.bin	2020/7/28 16:44	BIN 文件	1 KB
com_wifi5py.bin	2020/7/28 16:44	BIN 文件	2 KB