Build Environment Setup – IAR

This document illustrates how to build Realtek Wi-Fi SoC software under IAR SDK environment.

Table of Contents

1 How to get IAR

IAR provides an IDE environment for code building, downloading, and debugging. Please check "IAR Embedded Workbench" on http://www.iar.com/, and trail version is available.

2 Ameba-Z Hardware Configuration

Hardware block diagram is shown in **Figure 2-1 Hardware block diagram**. The block **USB UART** is used to supply power and catch logs, UART baud is 115200. **SWD** means J-Link SWD interface, when it is connected to J-Link Adapter correctly, you can download images to Ameba-Z from IAR flash downloader. **Reset button** is used to reset Ameba-Z to run firmware after IAR completes downloading.

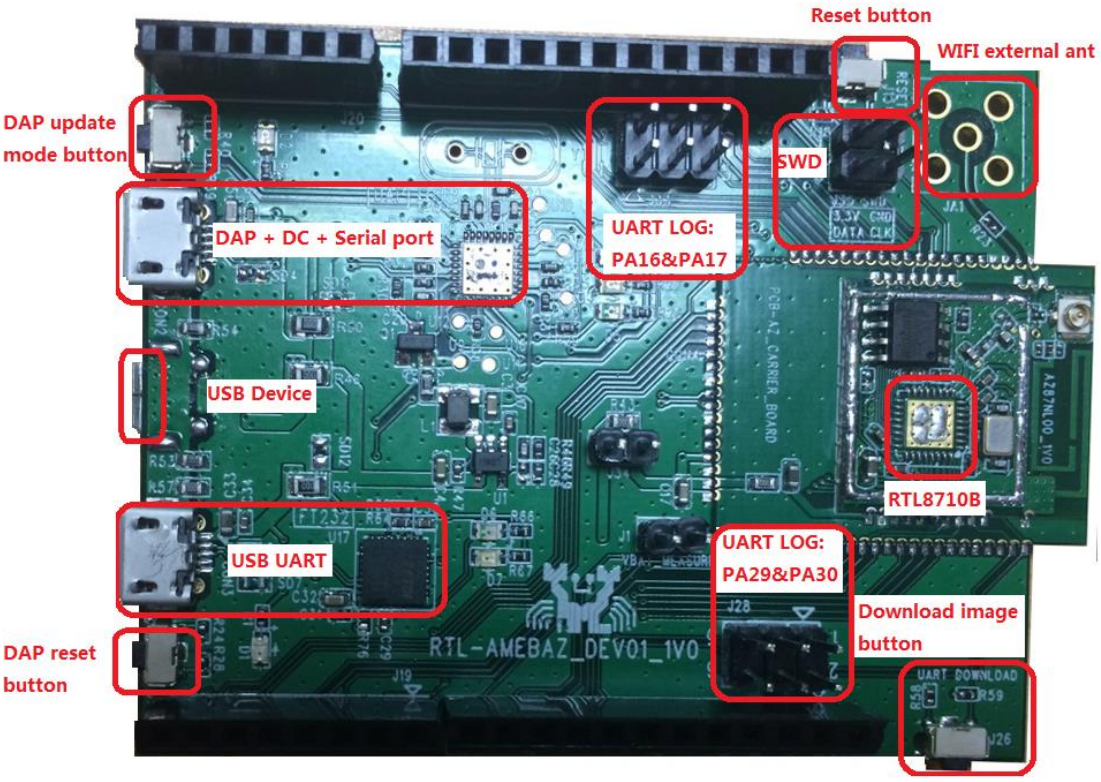


Figure 2-1 Hardware block diagram

The Dupont Line diagram between J-Link Adapter and Ameba-Z SWD is as follows:

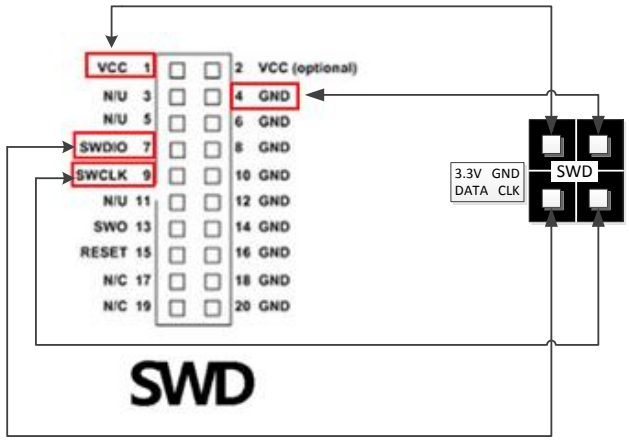


Figure 2-2 J-Link SWD connection

Please notice that DAP chip is not welding on the Ameba-Z demo board and so DAP function is not enabled which means MBED u disk copy is disabled.

Physical connection as shown below:

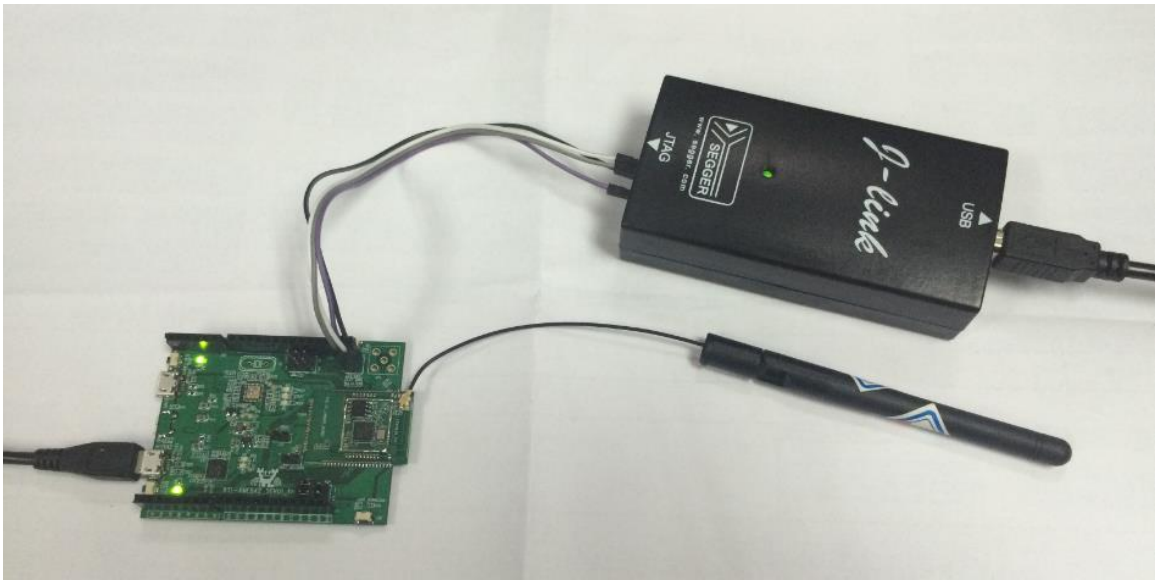


Figure 2-3 Physical connection

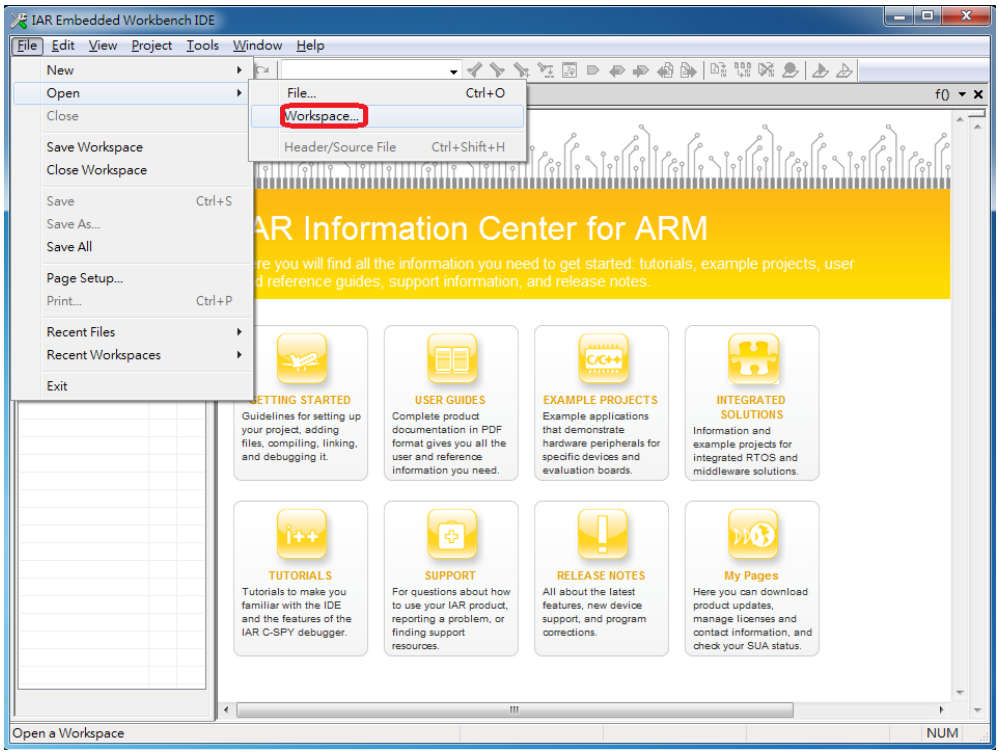
3 How to build and download code

Please make sure USB UART is connected to PC with USB line and SWD is connect to J-Link Adapter with Dupont Line before download code.

3.1 IAR build

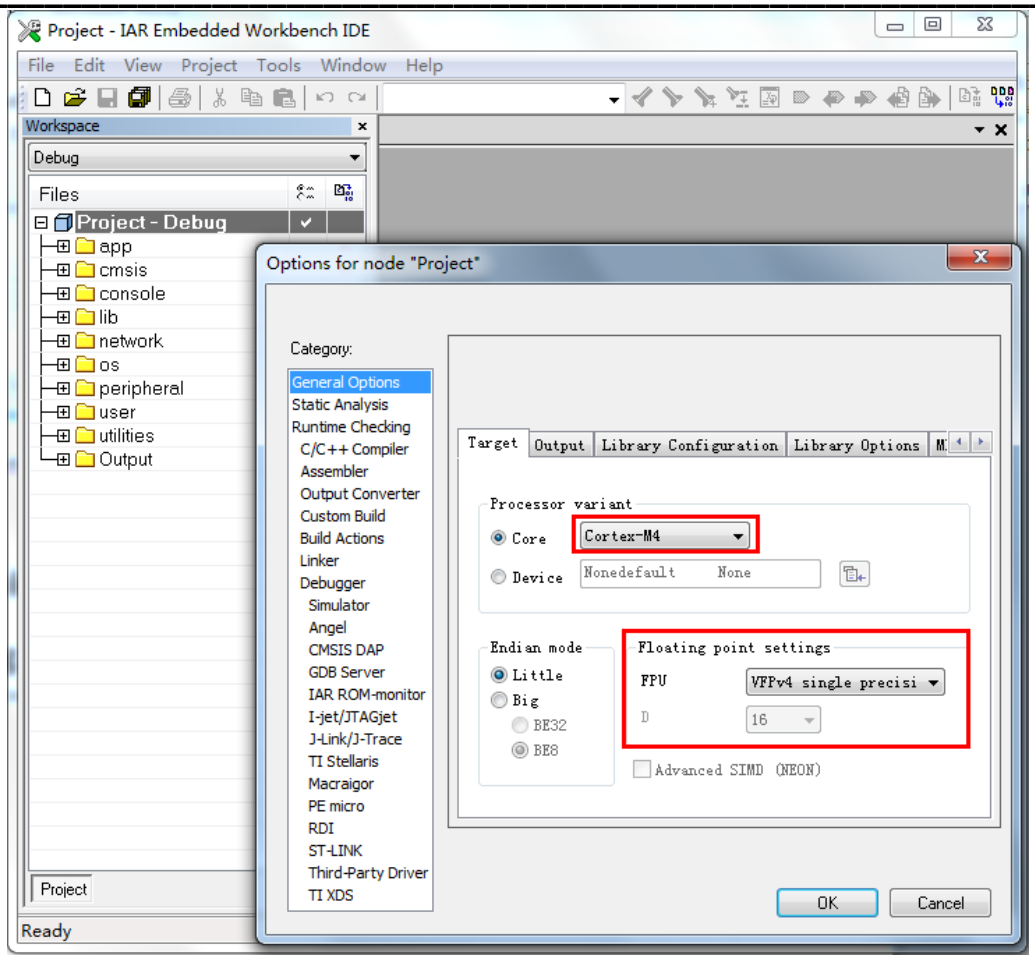
Step 1: Open IAR Workbench

Step 2: To open project, click File → Open → Workspace



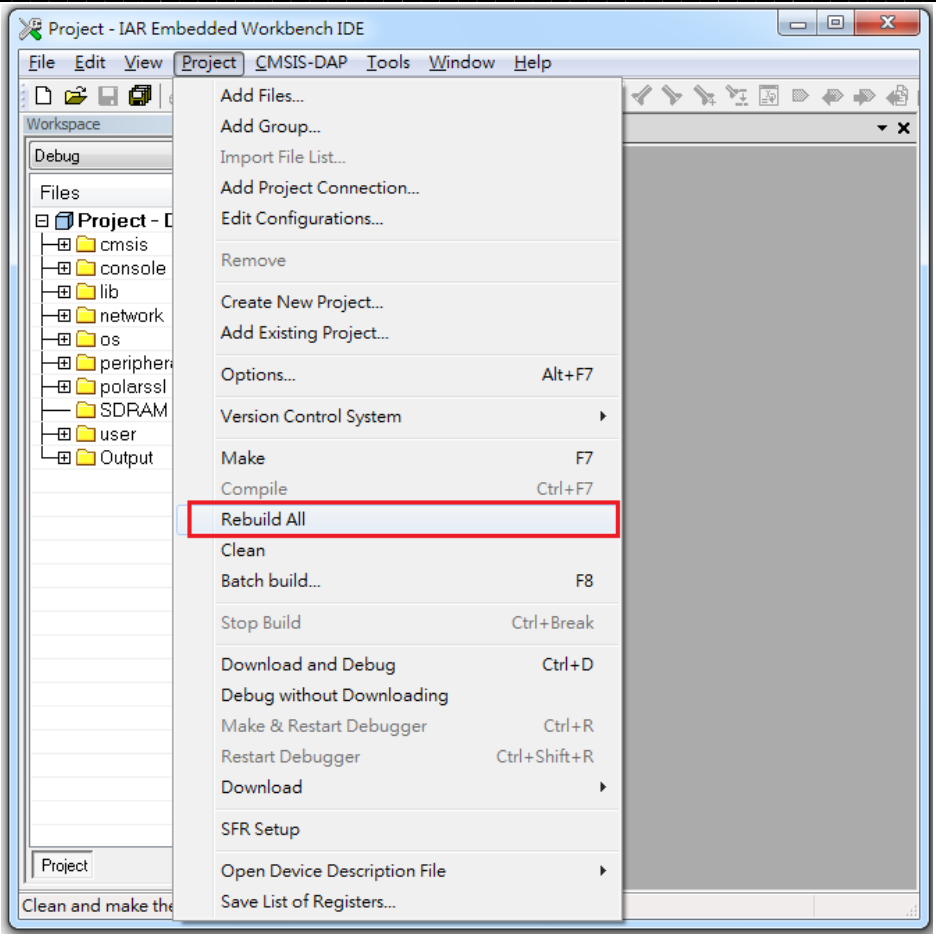
Step 3: Select Project.eww in project\realtek_amebaz_va0_example\EWARM-RELEASE

Click Project → Options, General Options->Target->Processor Variant->Core, make sure you have chosen **Cortex-M4f**. If your IAR not support this core, Please choose Cortex-M4 and Floating point settings->FPU is set to “VFPv4 single precision”.



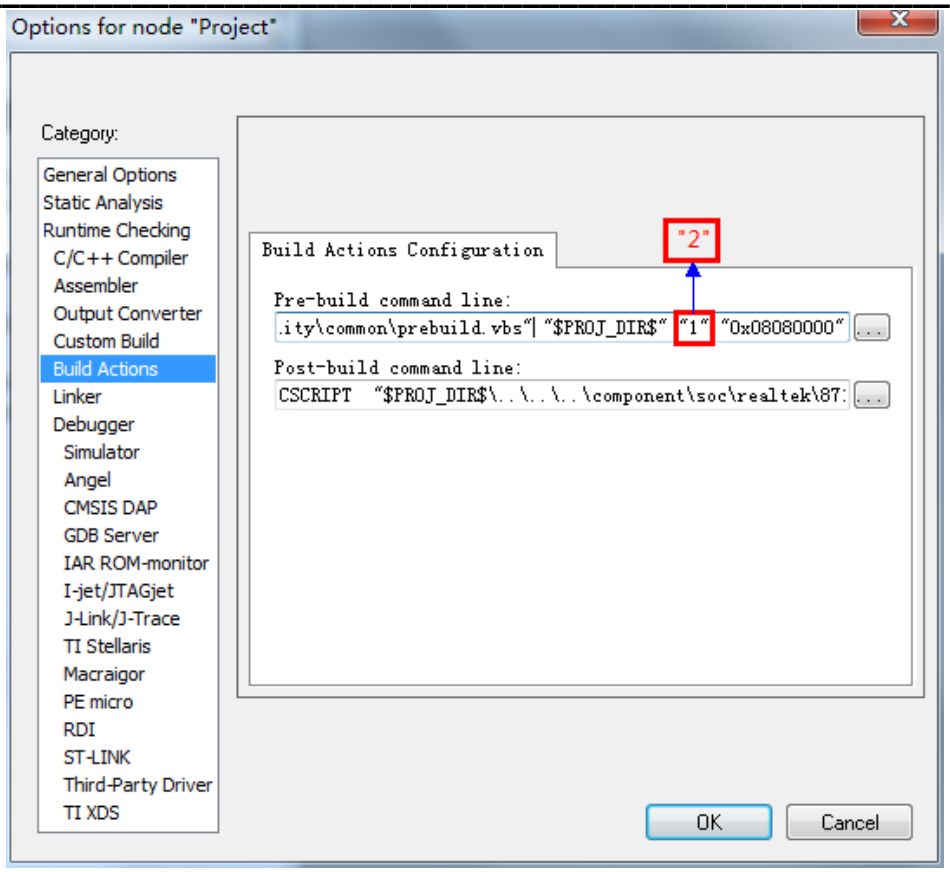
Step 4: To build project, click Project → Rebuild All

Then you will get boot_all.bin and image2_all_ota1.bin in project\realtek_amebaz_va0_example\EWARM-RELEASE \Debug\Exe.

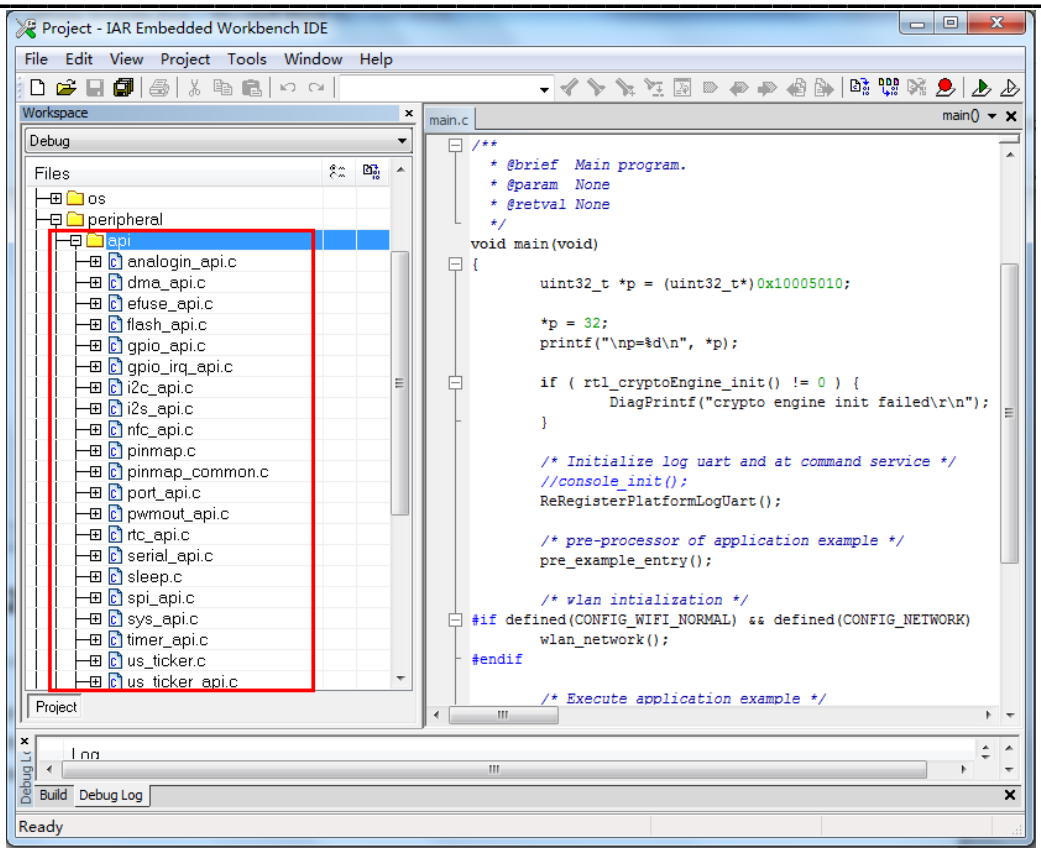


Step 5: To build image2_all_ota2.bin for OTA, click Project->Options->Build Actions->Build Actions Configuration->Pre-build command line, change the secondary parameter “1” to “2” shown as follows, then Click OK and make project “**Rebuild All**”.

Then you will get **image2_all_ota2.bin** in project\realtek_amebaz_va0_example\EWARM-RELEASE \Debug\Exe.



By the way, the mbed APIs include C files and Header files used by Ameba-Z is located in component\soc\realtek\8711b\mbed\. It will also be merged to component\common\mbed\ in next version. Now if you use Ameba-1, please choose files from component\common\mbed\ and if you use Ameba-Z, please choose files from component\soc\realtek\8711b\mbed\.

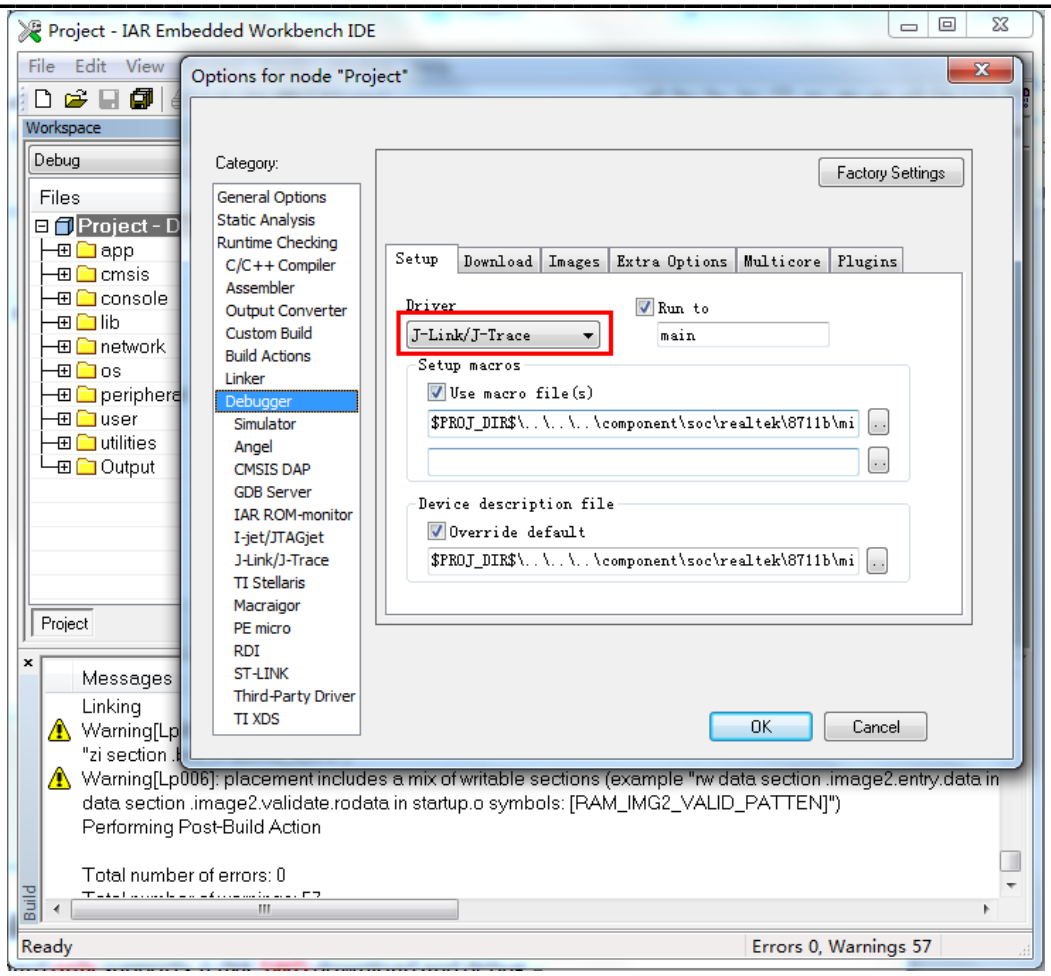


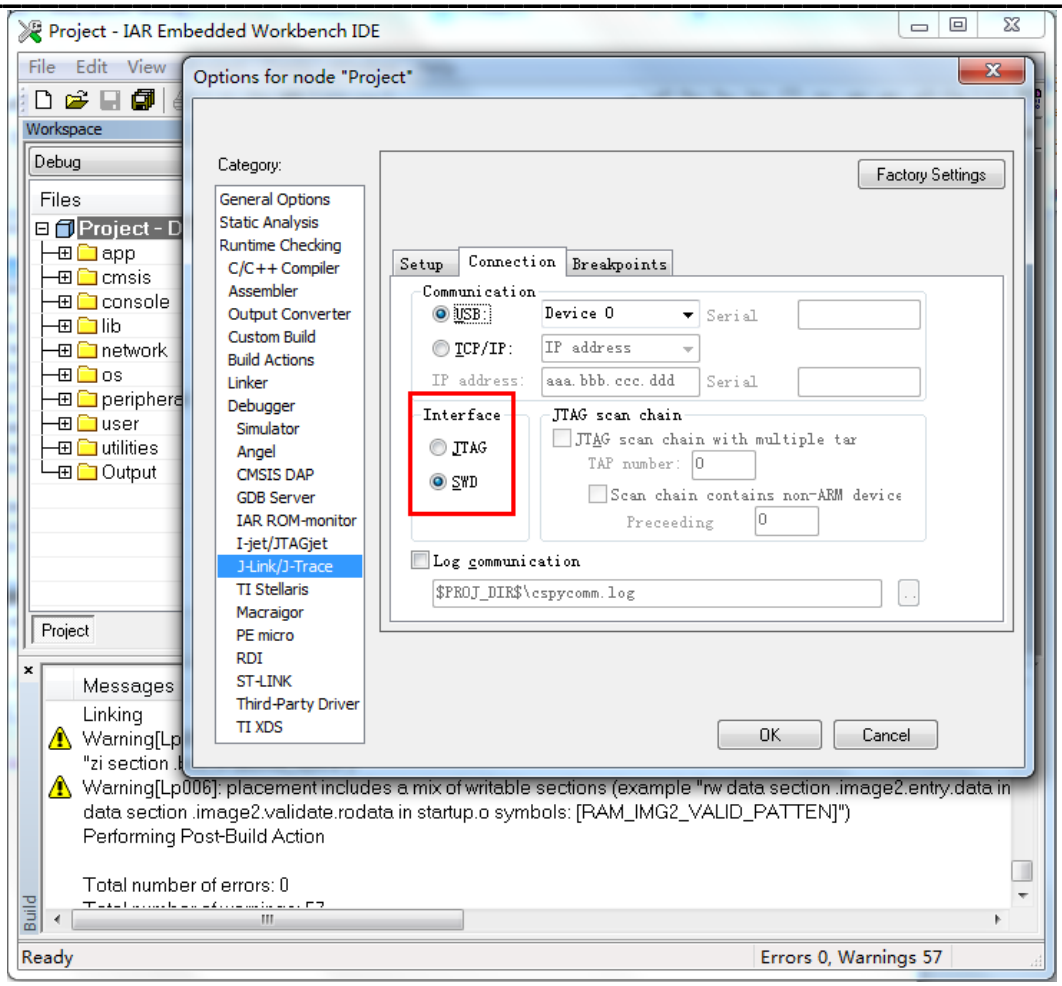
3.2 IAR download

The Ameba-Z demo board **only** supports JLINK **SWD** download and debug.

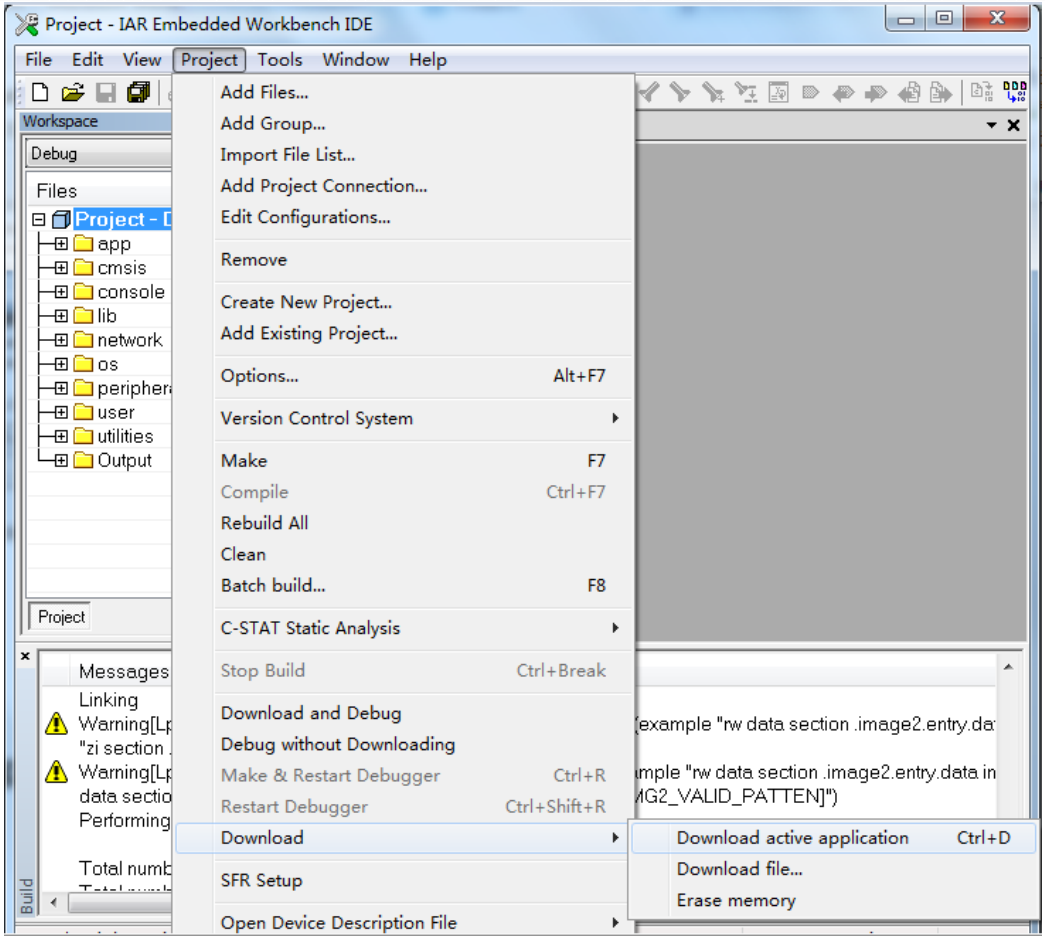
Please note that if you want use IAR download image to debug, please set the secondary parameter of prebuild.bat to "**1**" and build.

Step 1: Please check J-link debugger is setting correct. Click Project->Options->Debugger->Setup->Driver, and choose "J-Link/J-Trace". Then click Debugger-> J-Link/J-Trace->Connection->Interface and choose "SWD".


Project - IAR Embedded Workbench IDE
File Edit View
Workspace
Debug
Files
Project - D
app
cmsis
console
lib
network
os
periphera
user
utilities
Output
Project
Options for node "Project"
Category:
General Options
Static Analysis
Runtime Checking
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
CMSIS DAP
GDB Server
IAR ROM-monitor
I-jet/JTAGjet
J-Link/J-Trace
TI Stellaris
Macraigor
PE micro
RDI
ST-LINK
Third-Party Driver
TI XDS
Factory Settings
Setup
Download
Images
Extra Options
Multicore
Plugins
Driver
J-Link/J-Trace
Run to
main
Setup macros
Use macro file(s)
$PROJ_DIR$\..\..\..\component\soc\realtek\8711b\mi
Device description file
Override default
$PROJ_DIR$\..\..\..\component\soc\realtek\8711b\mi
OK
Cancel
Messages
Linking
Warning[Lp
"zi section .i
Warning[Lp006]: placement includes a mix of writable sections (example "rw data section .image2.entry.data in
data section .image2.validate.rodata in startup.o symbols: [RAM_IMG2_VALID_PATTEN]")
Performing Post-Build Action
Total number of errors: 0
Build
Ready
Errors 0, Warnings 57



Step 2: To download code, click Project->Download->Download active application.

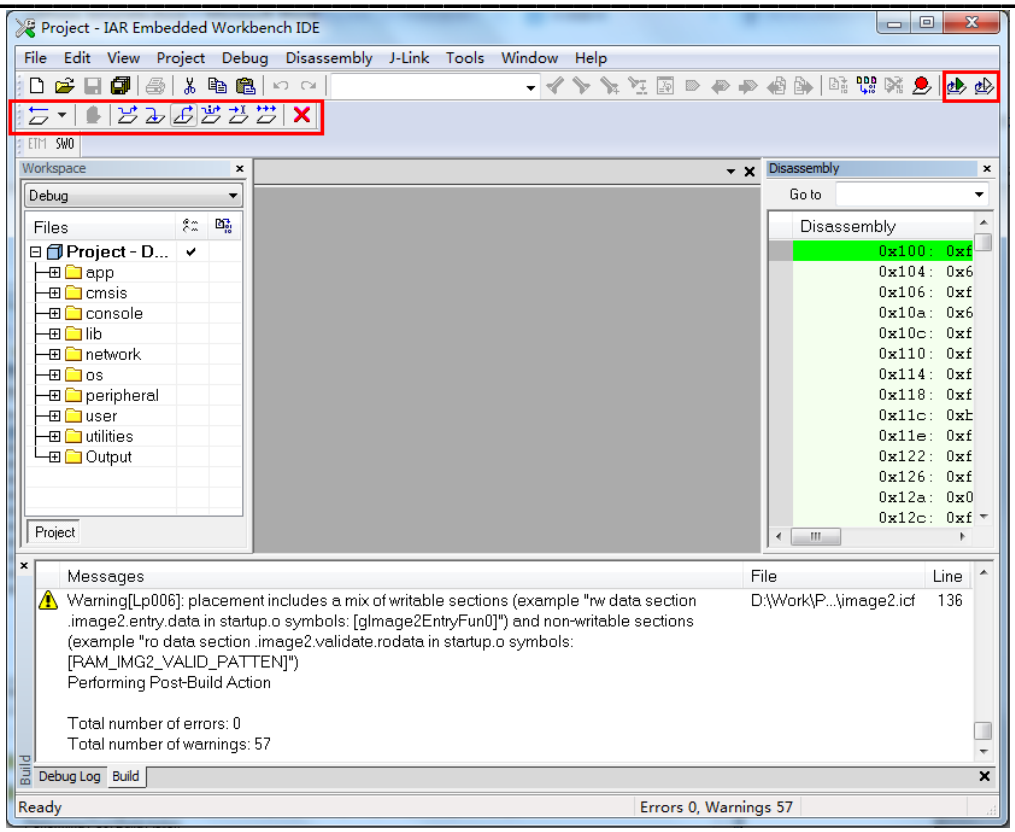


After firmware download, click Reset button to reboot the system.

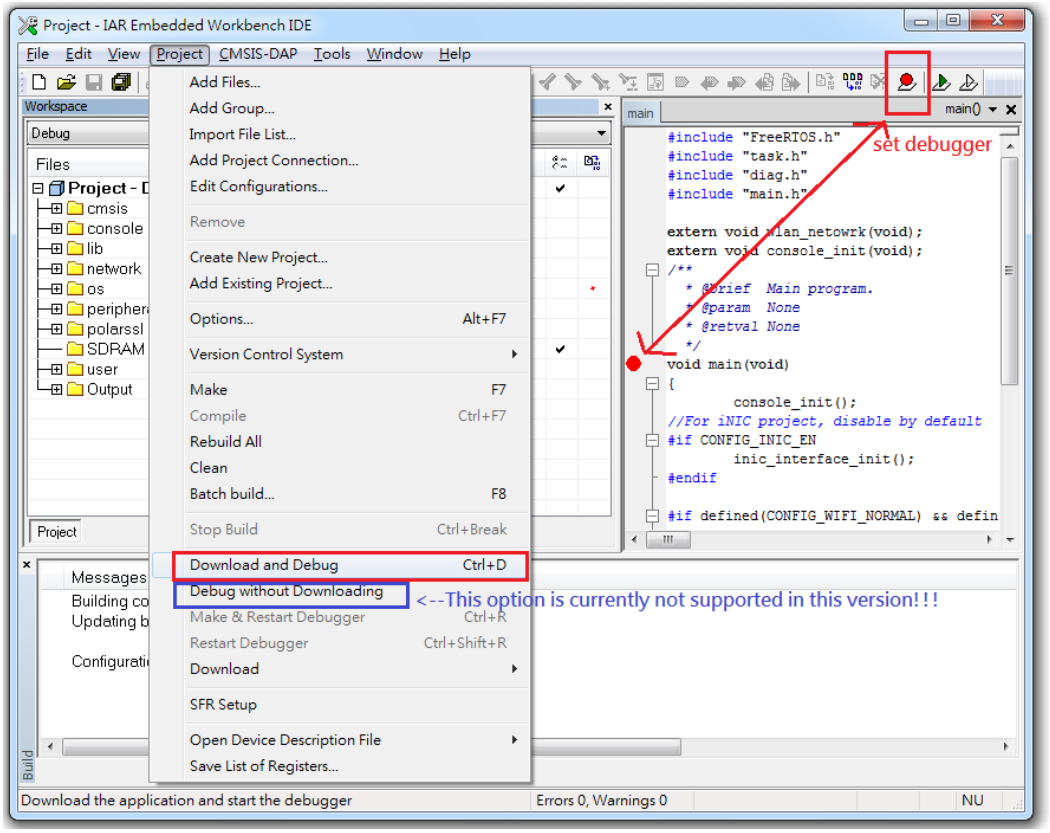
3.3 IAR debug

To debug or trace code step by step, click Project → Download and Debug or press one of the two buttons  in the IAR menu.

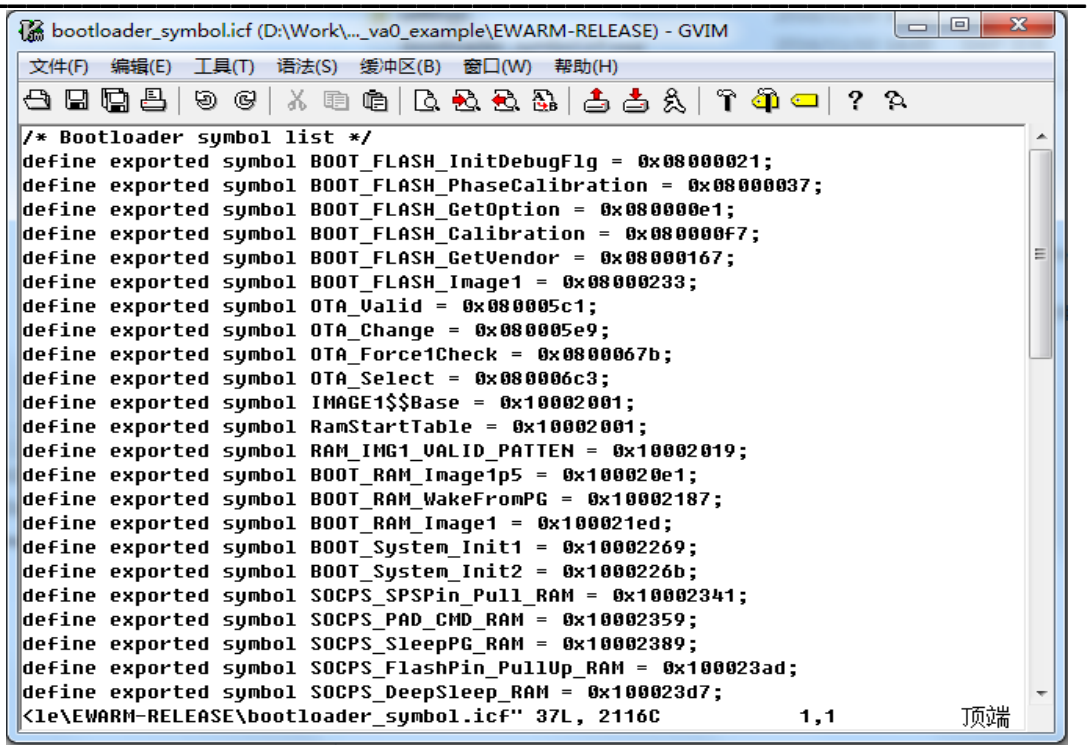
Upper left corner of the toolbar shows step by step tracking tools.

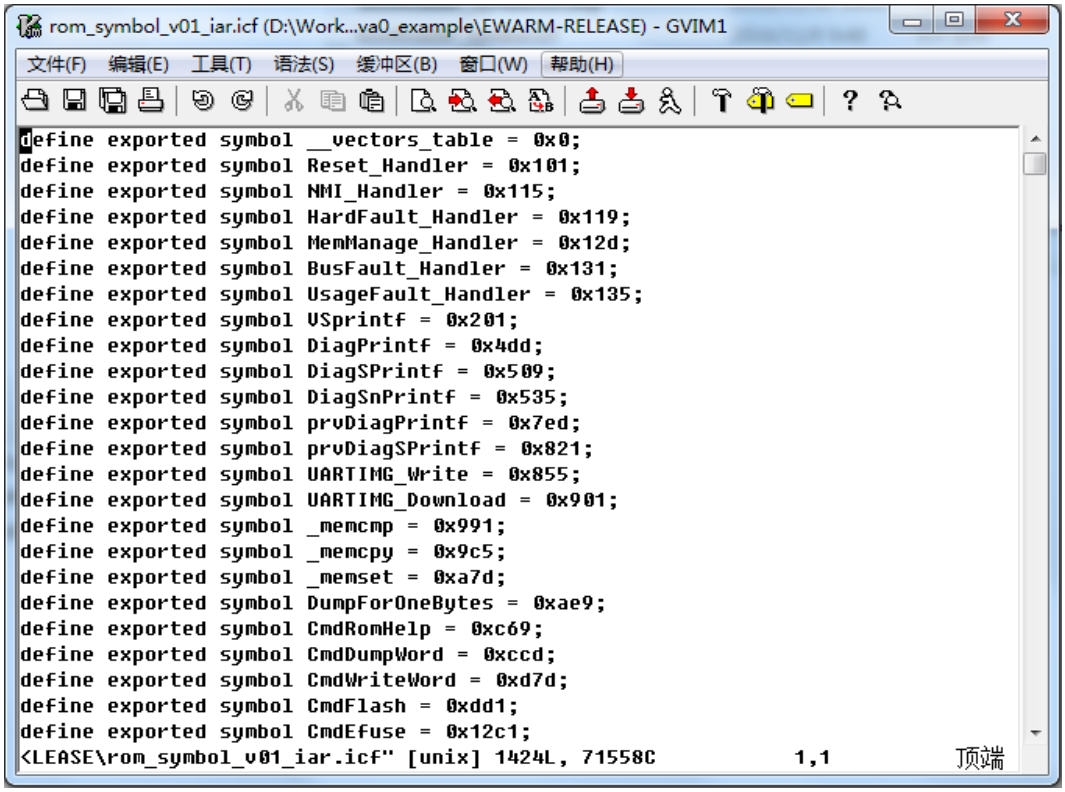


How to set Break point is shown as follows:



By the way, ROM code and Boot loader code is provided by means of the symbol list tables, rom_symbol_v01_iar.icf and bootloader_symbol.icf. You can find it in project\realtek_amebaz_va0_example\EWARM-RELEASE. So if you are tracking to the function which is running in the ROM or boot loader area, the source code can't be located.


bootloader_symbol.icf (D:\Work\..._va0_example\EWARM-RELEASE) - GVIM
文件(F) 编辑(E) 工具(T) 语法(S) 缓冲区(B) 窗口(W) 帮助(H)
/* Bootloader symbol list */
define exported symbol BOOT_FLASH_InitDebugFlg = 0x08000021;
define exported symbol BOOT_FLASH_PhaseCalibration = 0x08000037;
define exported symbol BOOT_FLASH_GetOption = 0x080000e1;
define exported symbol BOOT_FLASH_Calibration = 0x080000f7;
define exported symbol BOOT_FLASH_GetVendor = 0x08000167;
define exported symbol BOOT_FLASH_Image1 = 0x08000233;
define exported symbol OTA_Valid = 0x080005c1;
define exported symbol OTA_Change = 0x080005e9;
define exported symbol OTA_Force1Check = 0x0800067b;
define exported symbol OTA_Select = 0x080006c3;
define exported symbol IMAGE1$$Base = 0x10002001;
define exported symbol RamStartTable = 0x10002001;
define exported symbol RAM_IMG1_VALID_PATTEN = 0x10002019;
define exported symbol BOOT_RAM_Image1p5 = 0x100020e1;
define exported symbol BOOT_RAM_WakeFromPG = 0x10002187;
define exported symbol BOOT_RAM_Image1 = 0x100021ed;
define exported symbol BOOT_System_Init1 = 0x10002269;
define exported symbol BOOT_System_Init2 = 0x1000226b;
define exported symbol SOCPS_SPSPin_Pull_RAM = 0x10002341;
define exported symbol SOCPS_PAD_CMD_RAM = 0x10002359;
define exported symbol SOCPS_SleepPG_RAM = 0x10002389;
define exported symbol SOCPS_FlashPin_PullUp_RAM = 0x100023ad;
define exported symbol SOCPS_DeepSleep_RAM = 0x100023d7;
<le\EWARM-RELEASE\bootloader_symbol.icf" 37L, 2116C 1,1 顶端


rom_symbol_v01_iar.icf (D:\Work...va0_example\EWARM-RELEASE) - GVIM1
文件(F) 编辑(E) 工具(T) 语法(S) 缓冲区(B) 窗口(W) 帮助(H)
define exported symbol __vectors_table = 0x0;
define exported symbol Reset_Handler = 0x101;
define exported symbol NMI_Handler = 0x115;
define exported symbol HardFault_Handler = 0x119;
define exported symbol MemManage_Handler = 0x12d;
define exported symbol BusFault_Handler = 0x131;
define exported symbol UsageFault_Handler = 0x135;
define exported symbol VSprintf = 0x201;
define exported symbol DiagPrintf = 0x4dd;
define exported symbol DiagSPrintf = 0x509;
define exported symbol DiagSnPrintf = 0x535;
define exported symbol prvDiagPrintf = 0x7ed;
define exported symbol prvDiagSPrintf = 0x821;
define exported symbol UARTIMG_Write = 0x855;
define exported symbol UARTIMG_Download = 0x901;
define exported symbol _memcmp = 0x991;
define exported symbol _memcpy = 0x9c5;
define exported symbol _memset = 0xa7d;
define exported symbol DumpForOneBytes = 0xae9;
define exported symbol CmdRomHelp = 0xc69;
define exported symbol CmdDumpWord = 0xccd;
define exported symbol CmdWriteWord = 0xd7d;
define exported symbol CmdFlash = 0xdd1;
define exported symbol CmdEfuse = 0x12c1;
<LEASE\rom_symbol_v01_iar.icf" [unix] 1424L, 71558C 1,1 顶端

4 Image tool download

Please reference AN0112 Realtek Ameba-Z Image Tool user manual.doc.pdf for details.

Assuming that the Image Tool on PC is Server, which sends images files to Ameba-Z (Client) through UART. Client or Server, whichever starts first will be ok.

Steps on Ameba-Z

Step 1: For QFN32, pins marked with “1”on the figure should be connected by jumper cap.

For QFN48 & QFN68, pins marked with “2”shouldbe connected by jumper cap.

Step 2: Push the Image Download Button and keep it pressed.

Step 3: Power on the board or press the Reset Button. Now Ameba-Z goes into UART DOWNLOAD mode.

Step 4: Finally release the Image Download Button. Now the client is ready for receiving data.

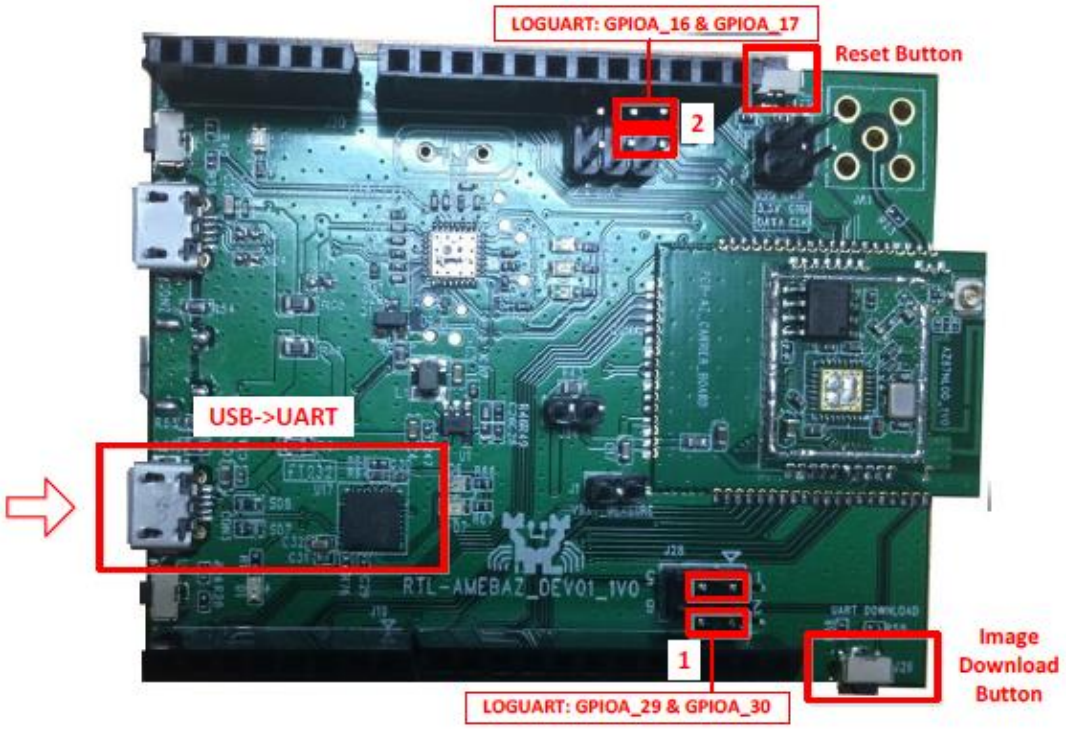


Figure 4-1 AmebaZ DEV

Steps on image tool

Step 1: Select serial port. The default baud rate is 1.5Mbps.

Step 2: Select transmission baud rate

Step 3: Open serial port

Step 4: Check the images to be transferred

Step 5: Select and load images files by browsing which are generated by IAR project.

Step 6: Input image addresses. The address which starts with 0x08 is for flash, and 0x10 for RAM.

Step 7: Push Download button to start. Now the server is ready to send data.

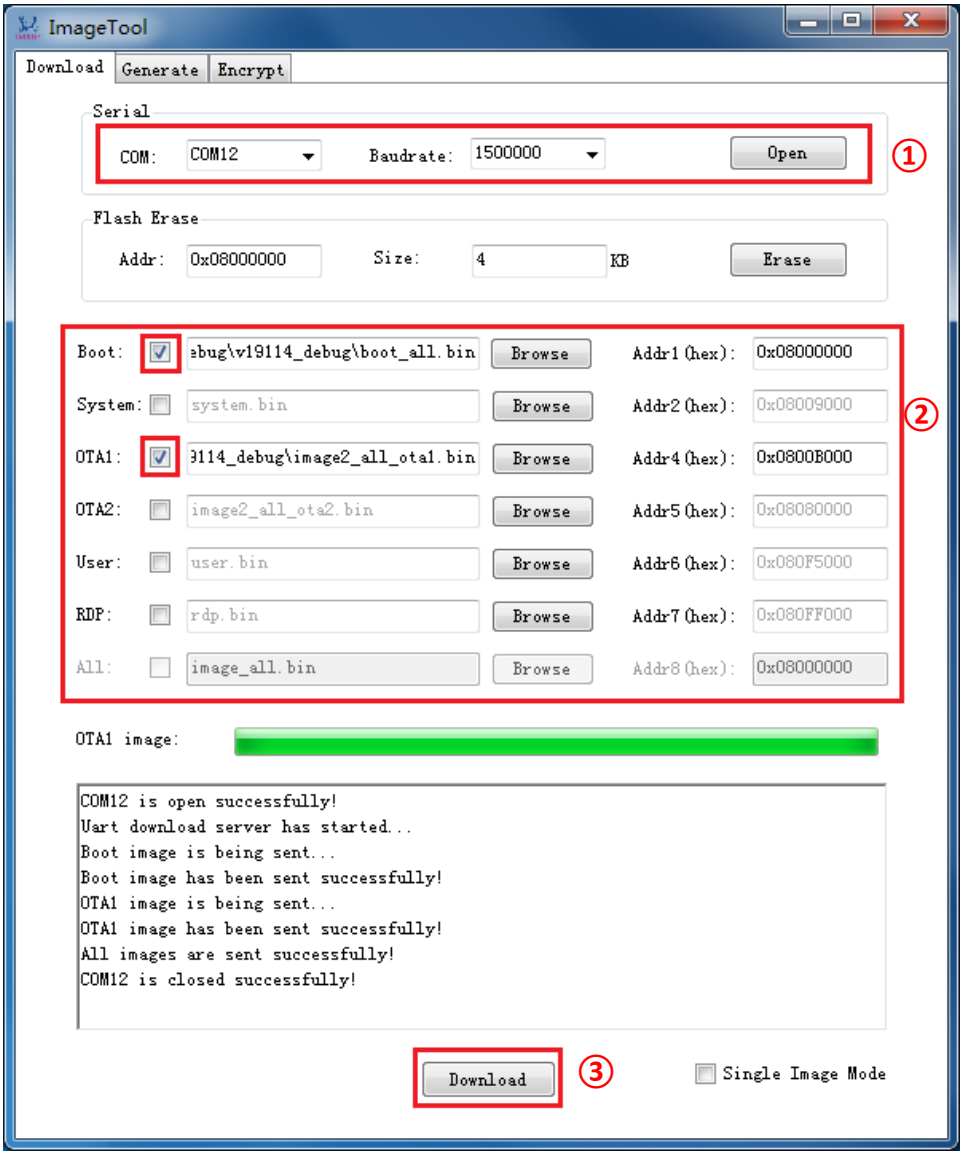


Figure 4-2 Image Download Tabpage

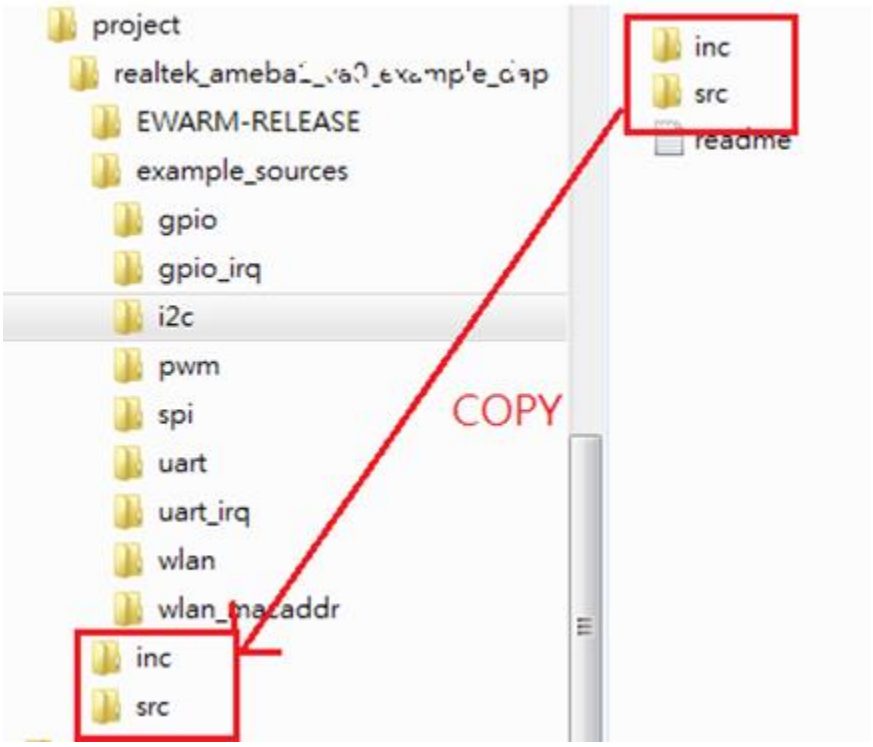
When the client and server are both ready, data transmission begins.

The progress bar will show the transmit progress of each image. You can also get the message of operation successful or errors occur from log window.

5 How to use peripheral sample code

There are several peripheral example code under folder “project\realtek_amebaz_va0_example \example_sources \”, you can copy & paste the example’s “inc” and “src” to project folder.

Ex. To use i2c example code, you can copy “src” and “inc” from “project\ realtek_amebaz_va0_example\example_sources\i2c\”.



Ameba API follows MBED API. User can check MBED website for peripheral API.

6 How to debug Hardfault

The Hardfault log is Similar to the following figure:

```
RTL8195A[HAL]: R0 = 0x0
RTL8195A[HAL]: R1 = 0x1005a37a
RTL8195A[HAL]: R2 = 0x6
RTL8195A[HAL]: R3 = 0x0
RTL8195A[HAL]: R12 = 0x1005f4a5
RTL8195A[HAL]: LR = 0x1004d0bf
RTL8195A[HAL]: PC = 0x1001ce00
RTL8195A[HAL]: PSR = 0x60000000
RTL8195A[HAL]: BFAR = 0x8
RTL8195A[HAL]: CFSR = 0x20000
RTL8195A[HAL]: HFSR = 0x40000000
RTL8195A[HAL]: DFSR = 0x0
RTL8195A[HAL]: AFSR = 0x0
RTL8195A[HAL]: PriMask 0x0
RTL8195A[HAL]: BasePri 0x0
RTL8195A[HAL]: SVC priority: 0x00
RTL8195A[HAL]: PendSVC priority: 0xf0
```

Firstly, you should record PC and LR register values.

Then open the application.asm in project\realtek_amebaz_va0_example\EWARM-RELEASE\Debug\Exe and find PC and LR in which functions. The instruction pointed by PC is that which caused system crash. And LR location shows who the caller of the crash function is.

If the value in PC register is in ROM (0x00000000~0x0007FFFF) or Boot loader area (0x10002000 ~ 0x10004FFF), you can check rom_symbol_v01_iar.icf or bootloader_symbol.icf to locate the function by the symbol address.

For example, if PC = 0x38478, then hardfault is happened in the function “ctr_drbg_random”.

```
define exported symbol ctr_drbg_random_with_add = 0x383ad;
define exported symbol ctr_drbg_random = 0x38469;
define exported symbol des_init = 0x388a5;
define exported symbol des_free = 0x388b1;
define exported symbol des3_init = 0x388c5;
define exported symbol des3_free = 0x388d5;
```

If PC is not a valid memory address, may be memory overflow or get flash instruction error.

If you need more information, please contact us with UART LOG, EWARM-RELEASE\Exe\application.asm and EWARM-RELEASE\Debug\application.map.

Please reference to Cortex-M3 Troubleshooting for more debug details.