AN0050
Application Note

AT32 MCU Compile by MDK5+GCC

Introduction

This application note introduces how to compile AT32 MCU standard library by MDK5+GCC.

Applicable products:

Part number	All AT32 series

Contents

List of tables

List of figures

1 Environment setup

1.1 GCC package

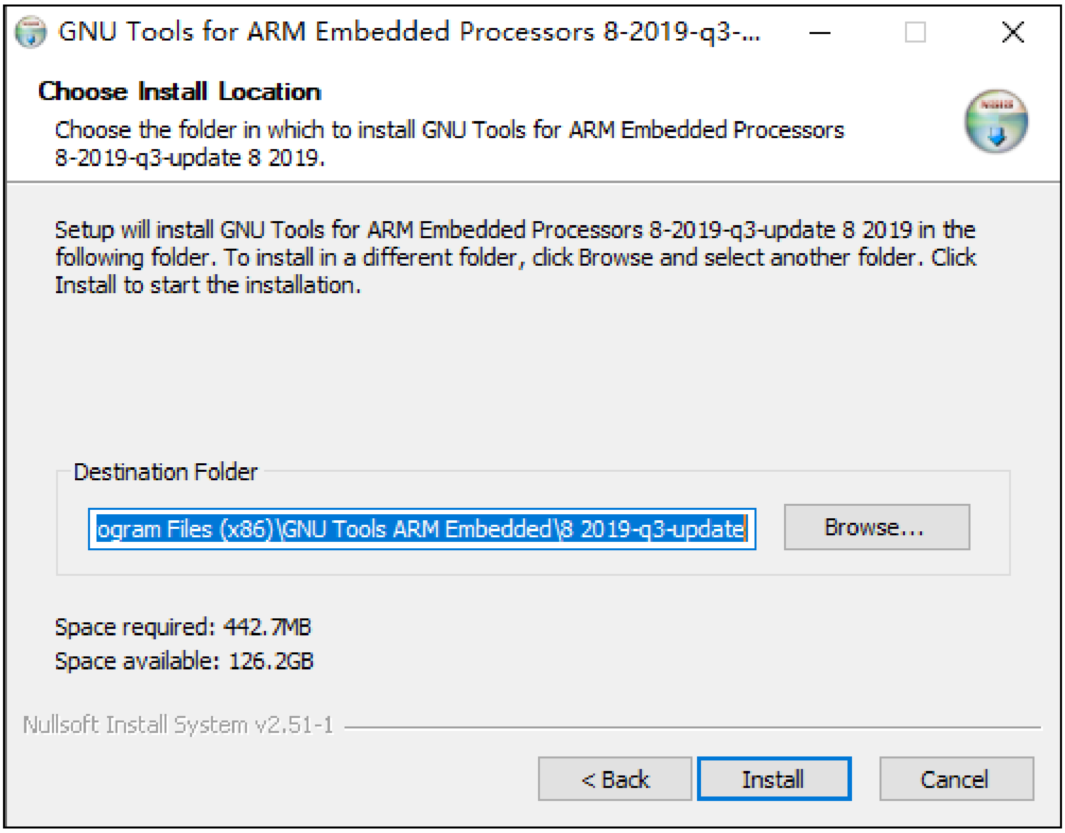
Download gcc-arm-none-eabi-8-2019-q3-update-win32-sha2.exe from ARM official website or update the version as needed.

For the convenience of demonstration, a GCC package is downloaded in advance and can be used directly.

1.2 GCC installation path

Select the default installation location “C:\Program Files (x86)\GNU Tools ARM Embedded\8 2019-q3-update”, as shown in Figure 1. Then click “Next” and record the installation path.

Figure 1. GCC installation location



GNU Tools for ARM Embedded Processors 8-2019-q3-...

Choose Install Location

Choose the folder in which to install GNU Tools for ARM Embedded Processors 8-2019-q3-update 8 2019.

Setup will install GNU Tools for ARM Embedded Processors 8-2019-q3-update 8 2019 in the following folder. To install in a different folder, click Browse and select another folder. Click Install to start the installation.

Destination Folder: ogram Files (x86)\GNU Tools ARM Embedded\8 2019-q3-update

Space required: 442.7MB

Space available: 126.2GB

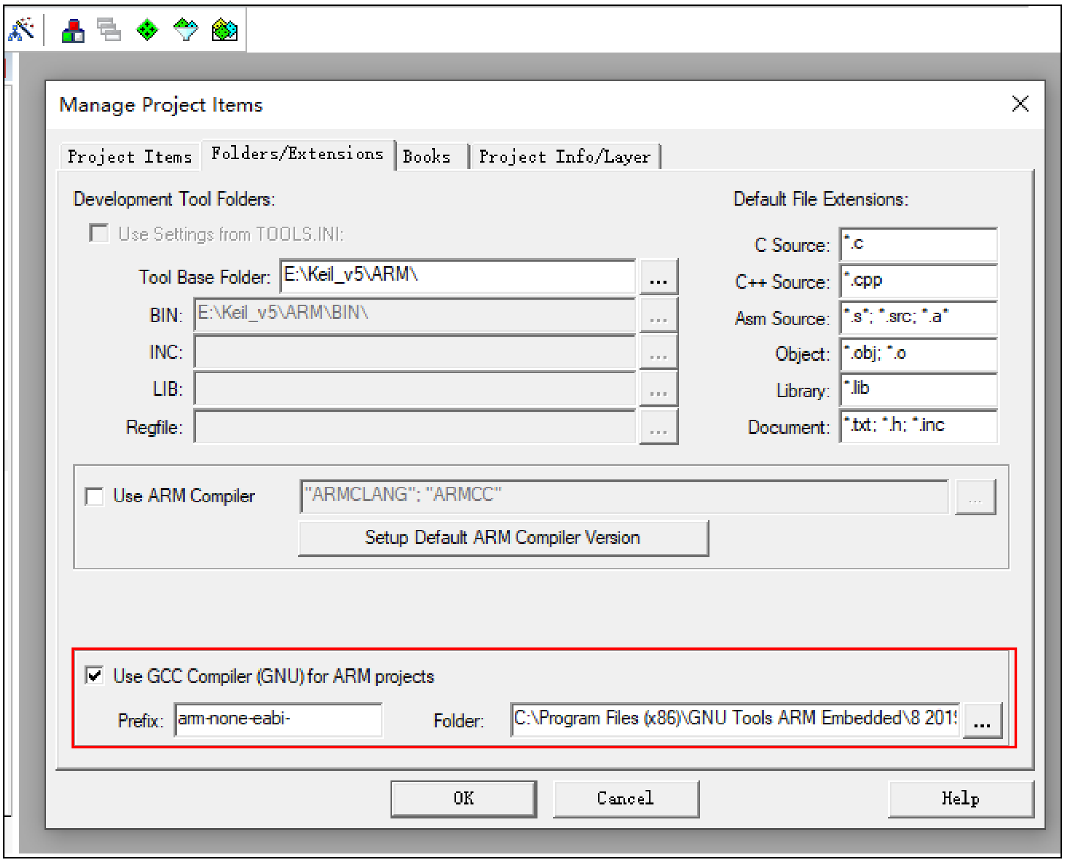
Nullsoft Install System v2.51-1

2 MDK project settings

2.1 GCC compiler

Tick “GCC Compiler” and select the installation path as mentioned in Section 1.2.

Figure 2. Use GCC compiler



Manage Project Items

Development Tool Folders:

□ Use Settings from TOOLS.INI:

Tool Base Folder: E:\Keil_v5\ARM\

BIN: E:\Keil_v5\ARM\BIN\

INC:

LIB:

Regfile:

Default File Extensions:

C Source: *.c

C++ Source: *.cpp

Asm Source: *.s*; *.src; *.a*

Object: *.obj; *.o

Library: *.lib

Document: *.txt; *.h; *.inc

□ Use ARM Compiler "ARMCLANG"; "ARMCC"

☑ Use GCC Compiler (GNU) for ARM projects

Prefix: arm-none-eabi-

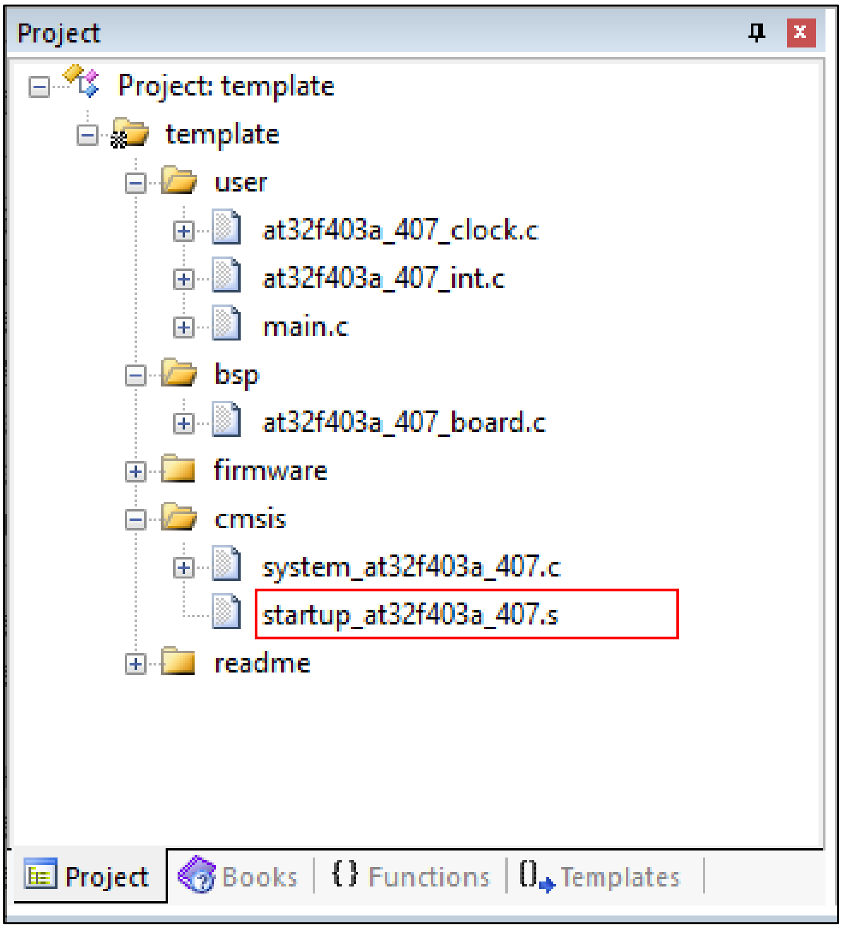
Folder: C:\Program Files (x86)\GNU Tools ARM Embedded\8 201

2.2 Add GCC startup file

The location of *startup.s* file in AT32 MCU standard library is as below:
AT32xxx_Firmware_Library_V2.x.x\AT32F403A_407_Firmware_Library_V2.0.0\libraries\cmsis\cm4\device_support\startup\gcc

Add GCC startup file to the project directory, as shown in Figure 3.

Figure 3. Add GCC startup file

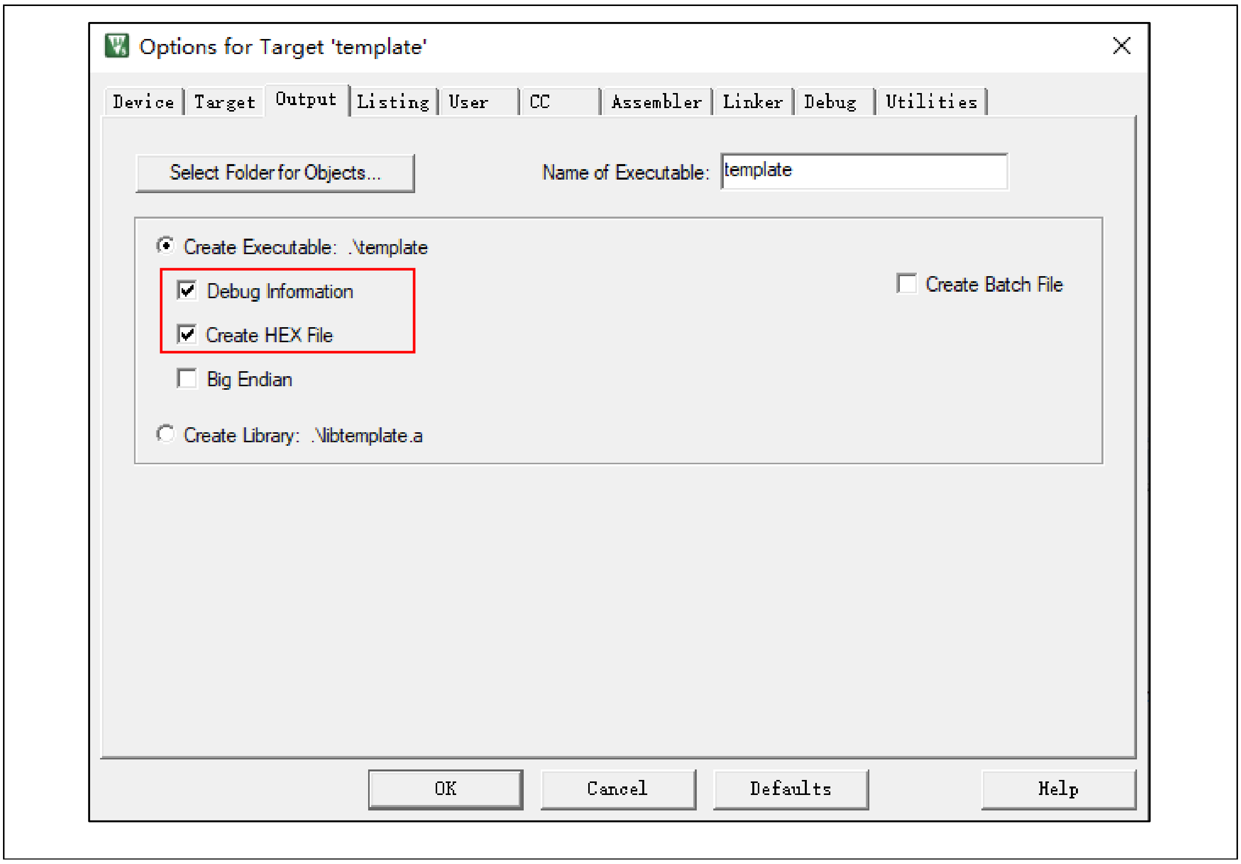


```
Project: template
  template
    user
      at32f403a_407_clock.c
      at32f403a_407_int.c
      main.c
    bsp
      at32f403a_407_board.c
    firmware
    cmsis
      system_at32f403a_407.c
      startup_at32f403a_407.s
    readme
```

2.3 Output configuration

Tick “Debug Information” for MCU debugging.
Tick “Create HEX File” to generate a HEX file.

Figure 4. Output configuration



Options for Target 'template'

Name of Executable: template

● Create Executable: .\template

☑ Debug Information

☑ Create HEX File

□ Big Endian

○ Create Library: .\libtemplate.a

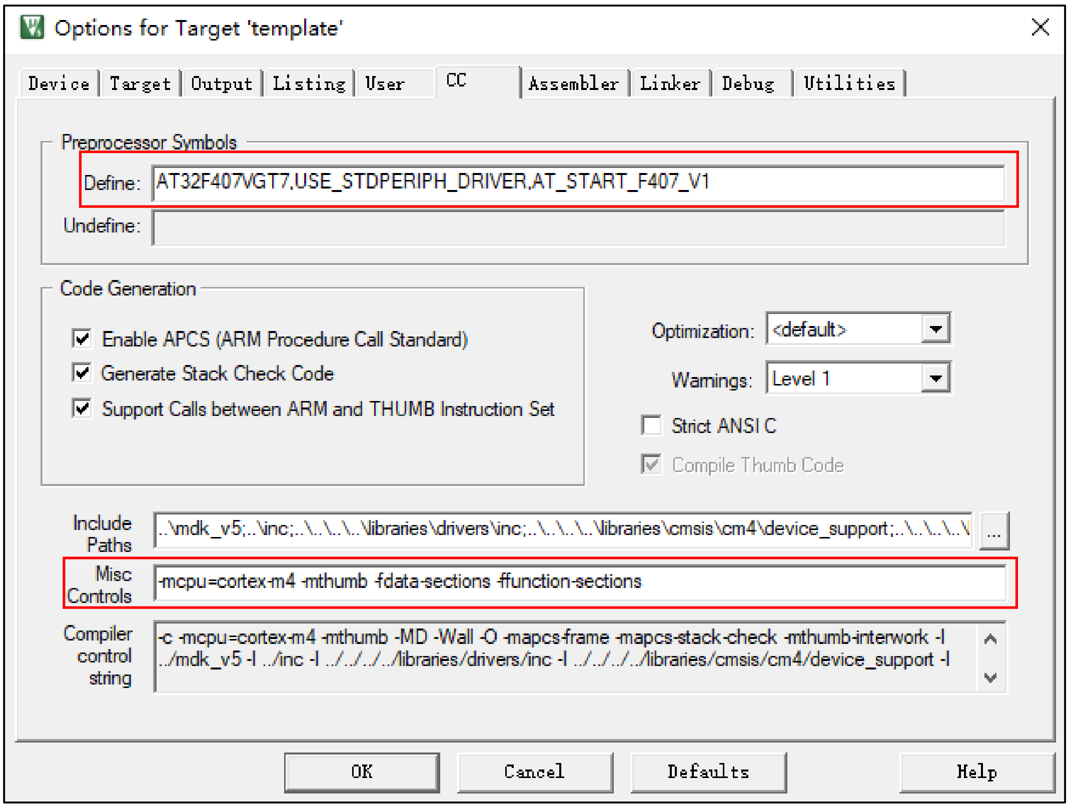
□ Create Batch File

2.4 CC configuration

Select “CC” field, and add “-mcpu=cortex-m4 -mthumb -fdata-sections -ffunction-sections” to “Misc Controls”.

Add device-related macro definitions in “Define” field.

Figure 5. CC configuration



Options for Target 'template'
Device | Target | Output | Listing | User | CC | Assembler | Linker | Debug | Utilities
OK
Cancel
Defaults
Help

Preprocessor Symbols

Define: AT32F407VGT7,USE_STDPERIPH_DRIVER,AT_START_F407_V1

Undefine:

Code Generation

☑ Enable APCS (ARM Procedure Call Standard)

☑ Generate Stack Check Code

☑ Support Calls between ARM and THUMB Instruction Set

Optimization: <default>

Warnings: Level 1

☐ Strict ANSI C

☑ Compile Thumb Code

Include Paths: ..\mdk_v5;..\inc;..\..\..\..\libraries\drivers\inc;..\..\..\..\libraries\cmsis\cm4\device_support;..\..\..\..\

Misc Controls: -mcpu=cortex-m4 -mthumb -fdata-sections -ffunction-sections

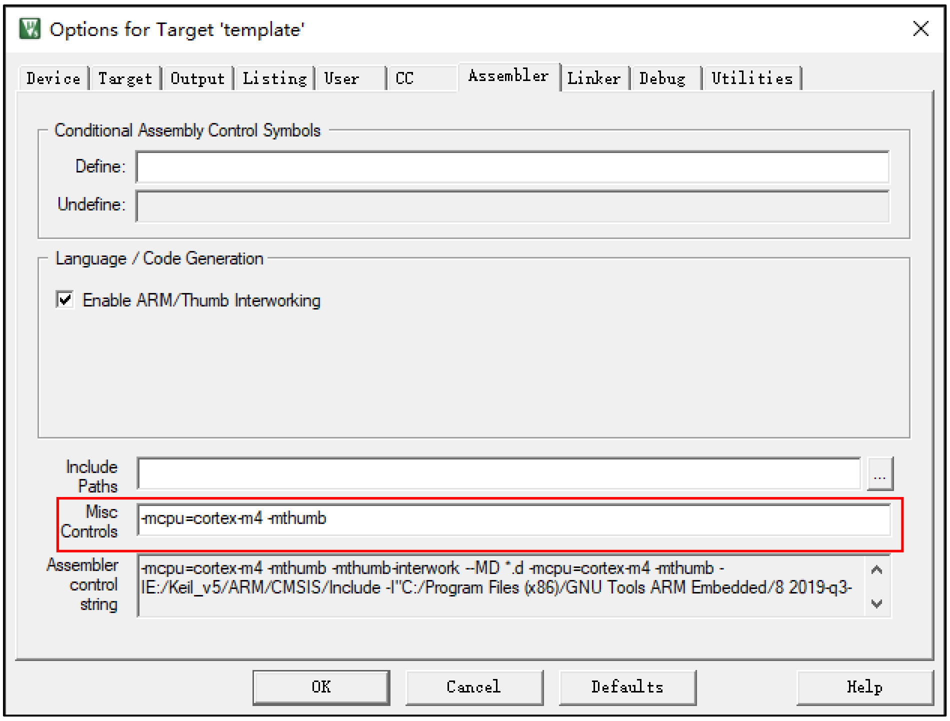
Compiler control string:

```
-c -mcpu=cortex-m4 -mthumb -MD -Wall -O -mapcs-frame -mapcs-stack-check -mthumb-interwork -I
../mdk_v5 -I ../inc -I ../../../../libraries/drivers/inc -I ../../../../libraries/cmsis/cm4/device_support -I
```

2.5 Assembler configuration

Select “Assembler” field and add “-mcpu=cortex-m4 -mthumb” to “Misc Controls”:

Figure 6. Assembler configuration



Conditional Assembly Control Symbols

Define:

Undefine:

Language / Code Generation

☑ Enable ARM/Thumb Interworking

Include Paths

Misc Controls: -mcpu=cortex-m4 -mthumb

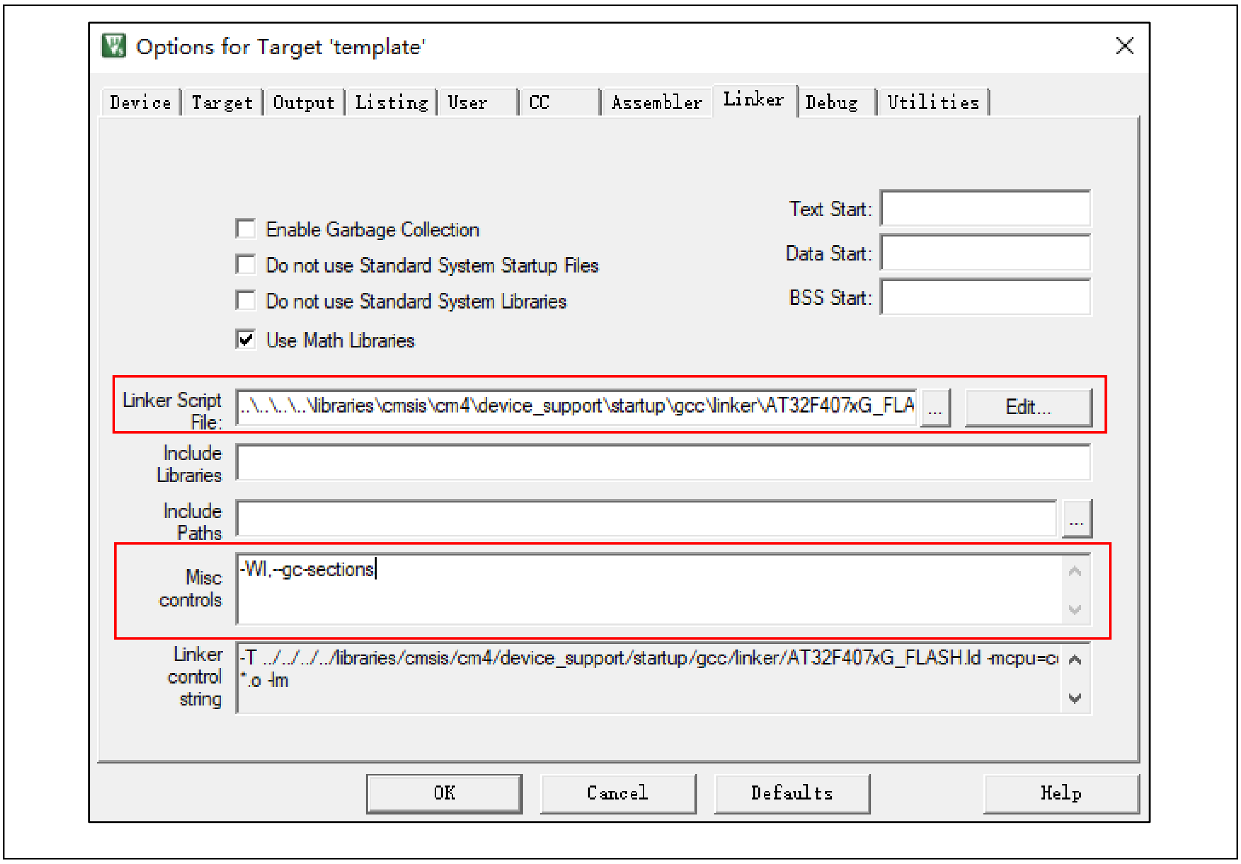
Assembler control string: -mcpu=cortex-m4 -mthumb -mthumb-interwork --MD *.d -mcpu=cortex-m4 -mthumb -IE:/Keil_v5/ARM/CMSIS/Include -I"C:/Program Files (x86)/GNU Tools ARM Embedded/8 2019-q3-

2.6 Linker configuration

Select “Linker” field and add “-Wl,--gc-sections” to “Misc Controls”.

Ld file path:
..\..\..\..\libraries\cmsis\cm4\device_support\startup\gcc\linker\AT32F407xG_FLASH.ld

Figure 7. Linker field configuration



□ Enable Garbage Collection
□ Do not use Standard System Startup Files
□ Do not use Standard System Libraries
☑ Use Math Libraries

Text Start:
Data Start:
BSS Start:

Linker Script File: ..\..\..\..\libraries\cmsis\cm4\device_support\startup\gcc\linker\AT32F407xG_FLA

Include Libraries:

Include Paths:

Misc controls: -Wl,--gc-sections

Linker control string: -T ../../../../libraries/cmsis/cm4/device_support/startup/gcc/linker/AT32F407xG_FLASH.ld -mcpu=c
*.o -lm

3 Project debugging

3.1 Compile

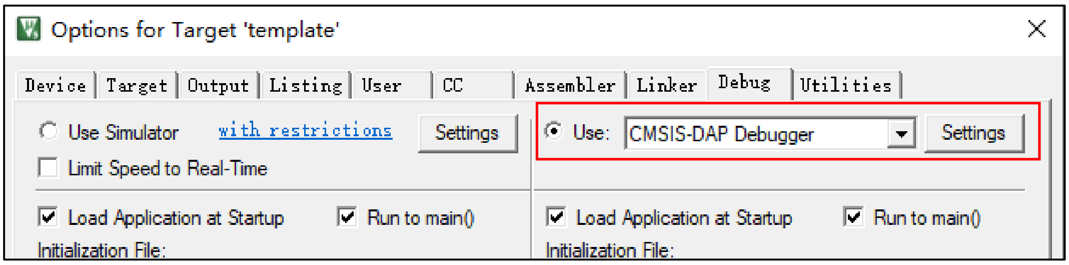
Click on Compile, and the generated .hex/.bin/.elf file can be found in the corresponding path.

3.2 Debug and download

Select AT-Link for debugging and download.

Open “Debug” field and select “CMSIS-DAP Debugger”, as shown in Figure 8.

Figure 8. Debug configuration



Options for Target 'template'

○ Use Simulator

□ Limit Speed to Real-Time

☑ Load Application at Startup ☑ Run to main()

Initialization File:

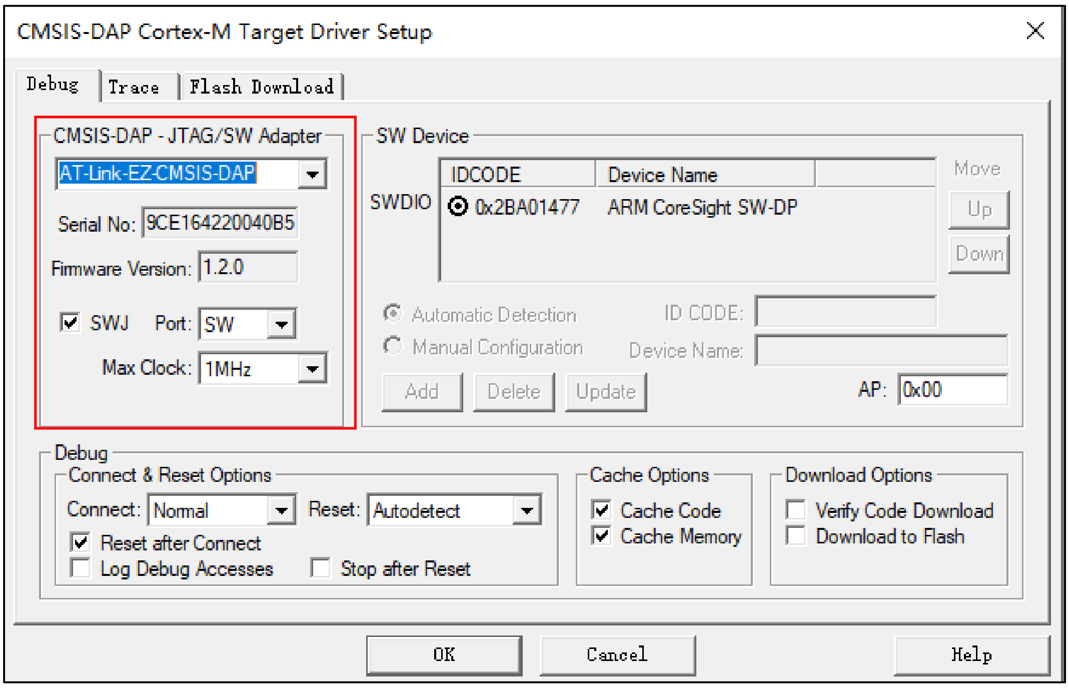
● Use: CMSIS-DAP Debugger

☑ Load Application at Startup ☑ Run to main()

Initialization File:

Click on “Setting” and select “AT-Link”, as shown in Figure 9.

Figure 9. Debug setting



CMSIS-DAP Cortex-M Target Driver Setup

CMSIS-DAP - JTAG/SW Adapter: AT-Link-EZ-CMSIS-DAP

Serial No: 9CE164220040B5

Firmware Version: 1.2.0

☑ SWJ Port: SW

Max Clock: 1MHz

SW Device

	IDCODE	Device Name
SWDIO	⊙ 0x2BA01477	ARM CoreSight SW-DP

● Automatic Detection ID CODE:

○ Manual Configuration Device Name:

AP: 0x00

Debug

Connect & Reset Options

Connect: Normal Reset: Autodetect

☑ Reset after Connect

□ Log Debug Accesses □ Stop after Reset

Cache Options

☑ Cache Code

☑ Cache Memory

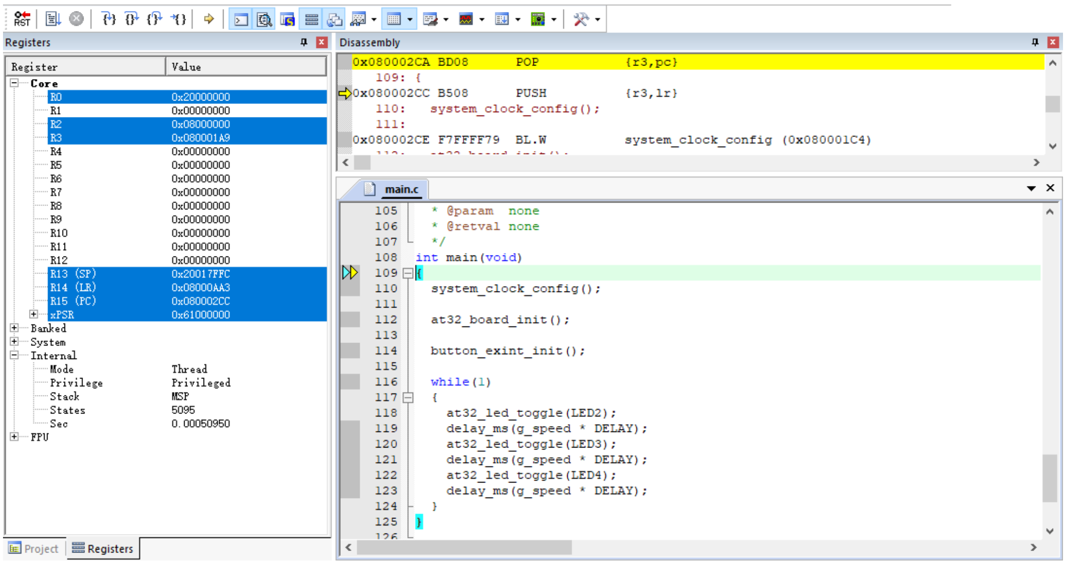
Download Options

□ Verify Code Download

□ Download to Flash

Finally, enter the debug interface, as shown in Figure 10.

Figure 10. Debug interface



Register	Value
Core	
R0	0x20000000
R1	0x00000000
R2	0x08000000
R3	0x080001A9
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x20017FFC
R14 (LR)	0x08000AA3
R15 (PC)	0x080002CC
xPSR	0x61000000
Banked	
System	
Internal	
Mode	Thread
Privilege	Privileged
Stack	MSP
States	5095
Sec	0.00050950
FPU	

```
0x080002CA BD08      POP           {r3,pc}
   109: {
0x080002CC B508      PUSH          {r3,lr}
   110:    system_clock_config();
   111:
0x080002CE F7FFFF79  BL.W          system_clock_config (0x080001C4)
```

```
  * @param  none
  * @retval none
  */
int main(void)
{
  system_clock_config();

  at32_board_init();

  button_exint_init();

  while(1)
  {
    at32_led_toggle(LED2);
    delay_ms(g_speed * DELAY);
    at32_led_toggle(LED3);
    delay_ms(g_speed * DELAY);
    at32_led_toggle(LED4);
    delay_ms(g_speed * DELAY);
  }
}
```

4 Revision history

Table 1. Document revision history

Date	Version	Revision note
2021.09.18	2.0.0	Initial release.

IMPORTANT NOTICE – PLEASE READ CAREFULLY

Purchasers are solely responsible for the selection and use of ARTERY's products and services; ARTERY assumes no liability for purchasers' selection or use of the products and the relevant services.

No license, express or implied, to any intellectual property right is granted by ARTERY herein regardless of the existence of any previous representation in any forms. If any part of this document involves third party's products or services, it does NOT imply that ARTERY authorizes the use of the third party's products or services, or permits any of the intellectual property, or guarantees any uses of the third party's products or services or intellectual property in any way.

Except as provided in ARTERY's terms and conditions of sale for such products, ARTERY disclaims any express or implied warranty, relating to use and/or sale of the products, including but not restricted to liability or warranties relating to merchantability, fitness for a particular purpose (based on the corresponding legal situation in any unjudicial districts), or infringement of any patent, copyright, or other intellectual property right.

ARTERY's products are not designed for the following purposes, and thus not intended for the following uses: (A) Applications that have specific requirements on safety, for example: life-support applications, active implant devices, or systems that have specific requirements on product function safety; (B) Aviation applications; (C) Aerospace applications or environment; (D) Weapons, and/or (E) Other applications that may cause injuries, deaths or property damages. Since ARTERY products are not intended for the above-mentioned purposes, if purchasers apply ARTERY products to these purposes, purchasers are solely responsible for any consequences or risks caused, even if any written notice is sent to ARTERY by purchasers; in addition, purchasers are solely responsible for the compliance with all statutory and regulatory requirements regarding these uses.

Any inconsistency of the sold ARTERY products with the statement and/or technical features specification described in this document will immediately cause the invalidity of any warranty granted by ARTERY products or services stated in this document by ARTERY, and ARTERY disclaims any responsibility in any form.