IDE Project Porting Guide

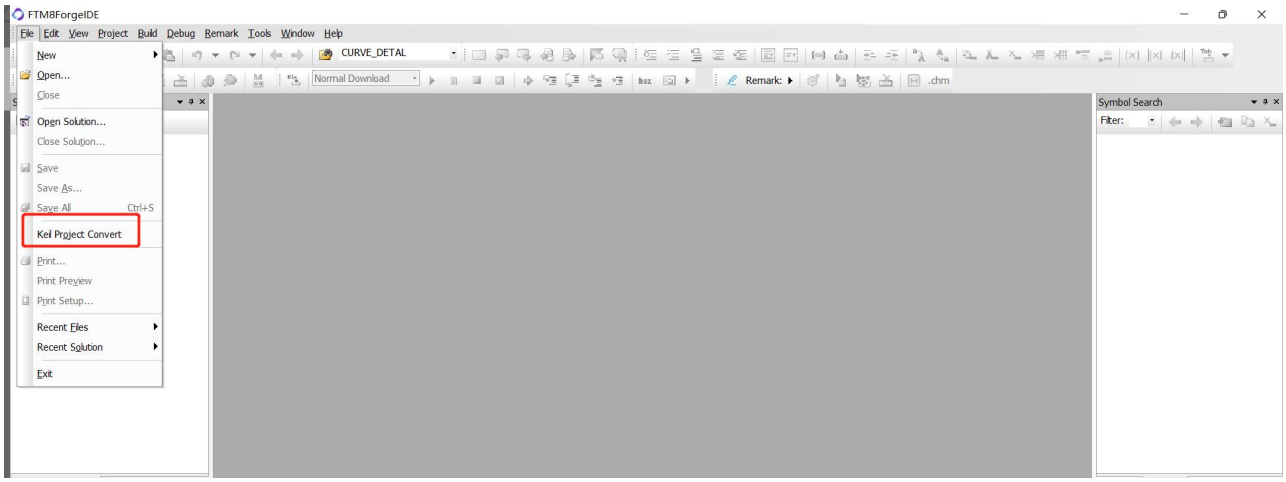
Fortior Technology (Shenzhen)Co., Ltd

Contents

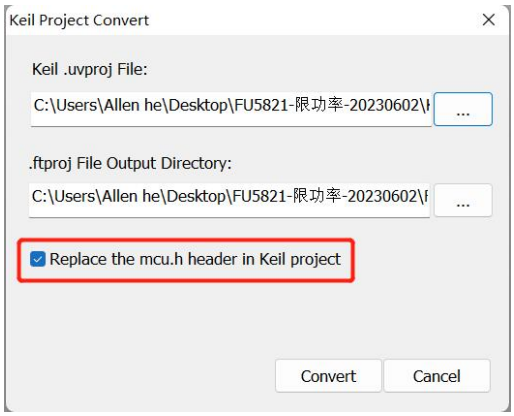
1 Convert Keil Project to IDE Project

1.1 Automatic Convert

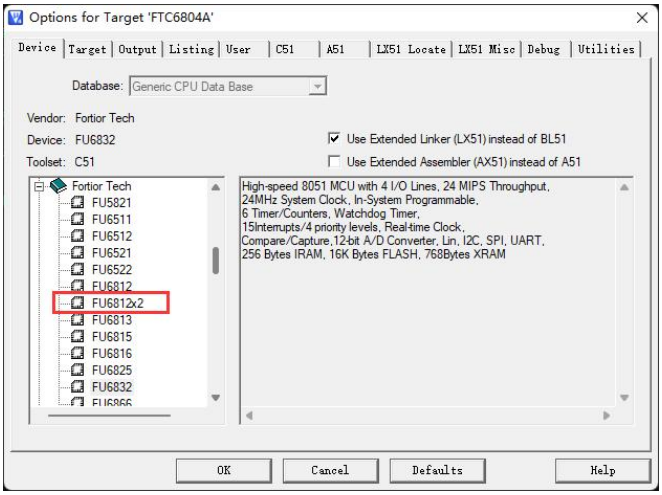
Start IDE Project and then select File → Keil Project Convert in the top left.



Select Keil Project to be converted, check Replace the mcu.h header in Keil project, and then click Convert. If the register name in the program is different from that defined in the header file after conversion, the latter prevails.



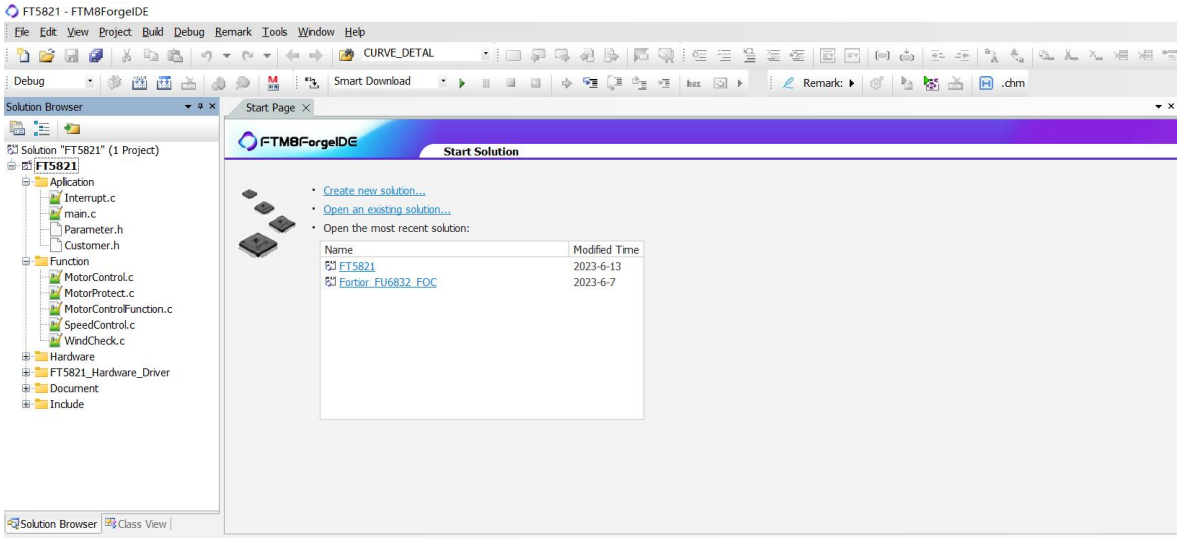
Before converting 6812x2 Project, you must check whether 6812x2 in Keil is selected. Otherwise, IDE access the header file 6812.



Note: IDE contains both 6812 and 6812x2 header files. Thus, after the convert, if 6812x2 or 6522_72

header file is not found, you shall revise local header file's name or the header name referenced in IDE.

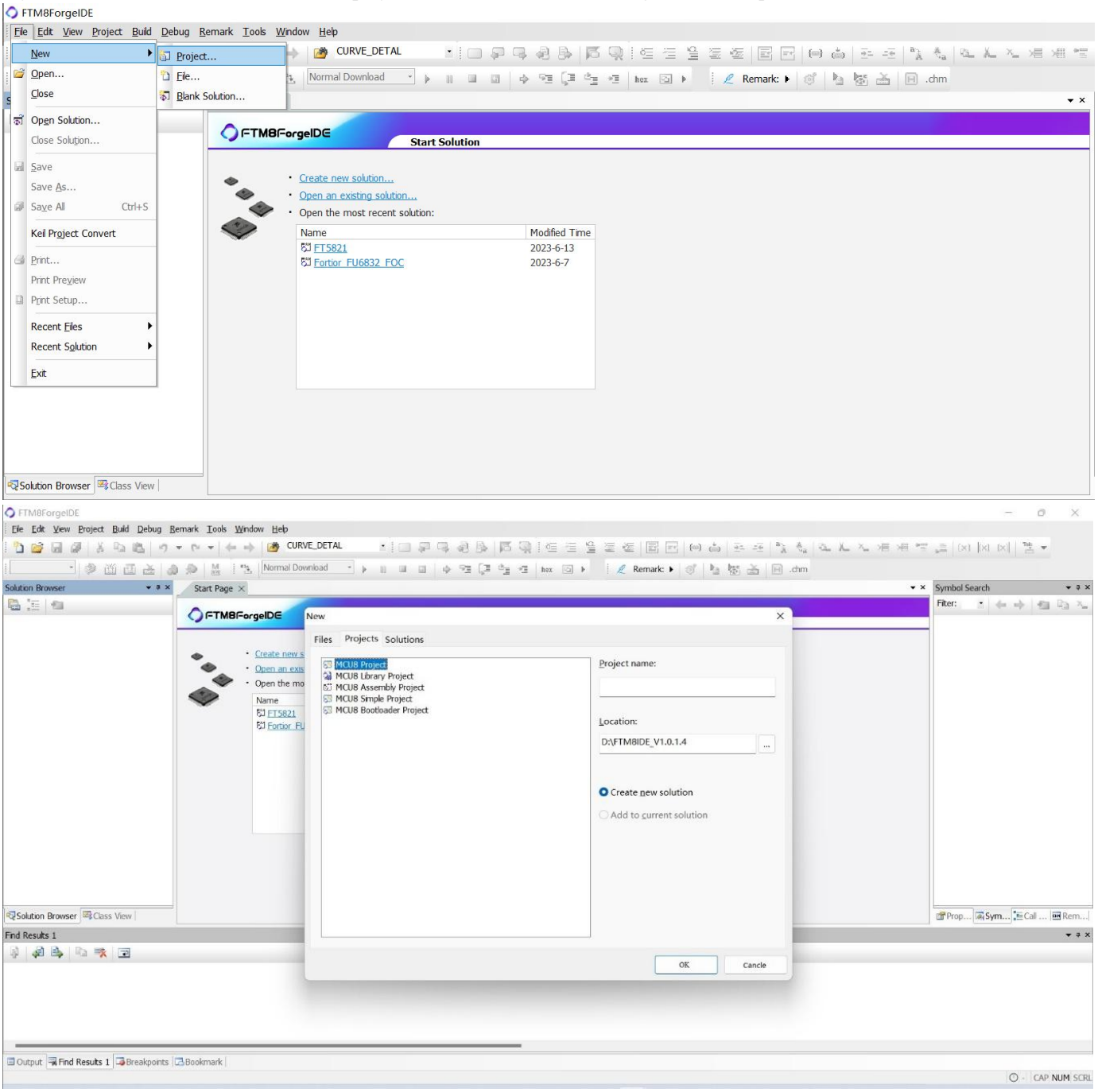
After converting, the folder structure of IDE is completely the same as that of Keil.



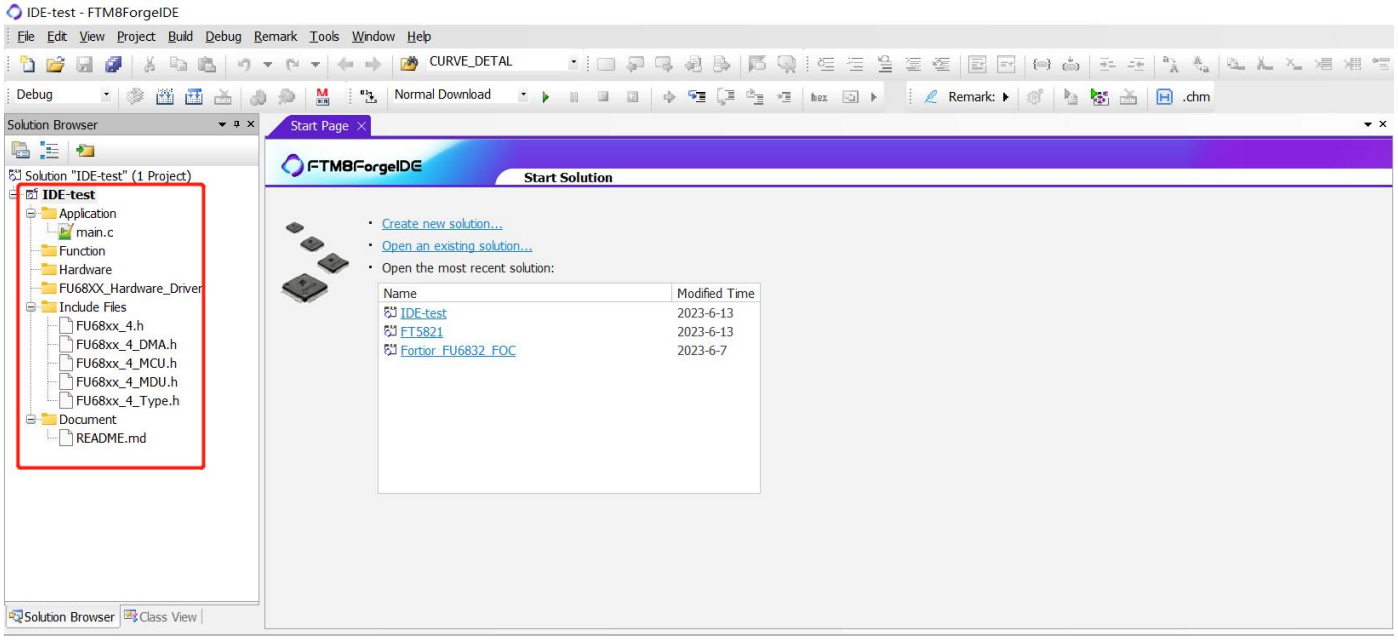
2 Create IDE Project

2.1 Create Common Project

Start IDE Project, select File → NEW → Project , and then select MCU8 Project. Project name is the name of selected project, and Location is the address of the project to be stored. You can change them as required.

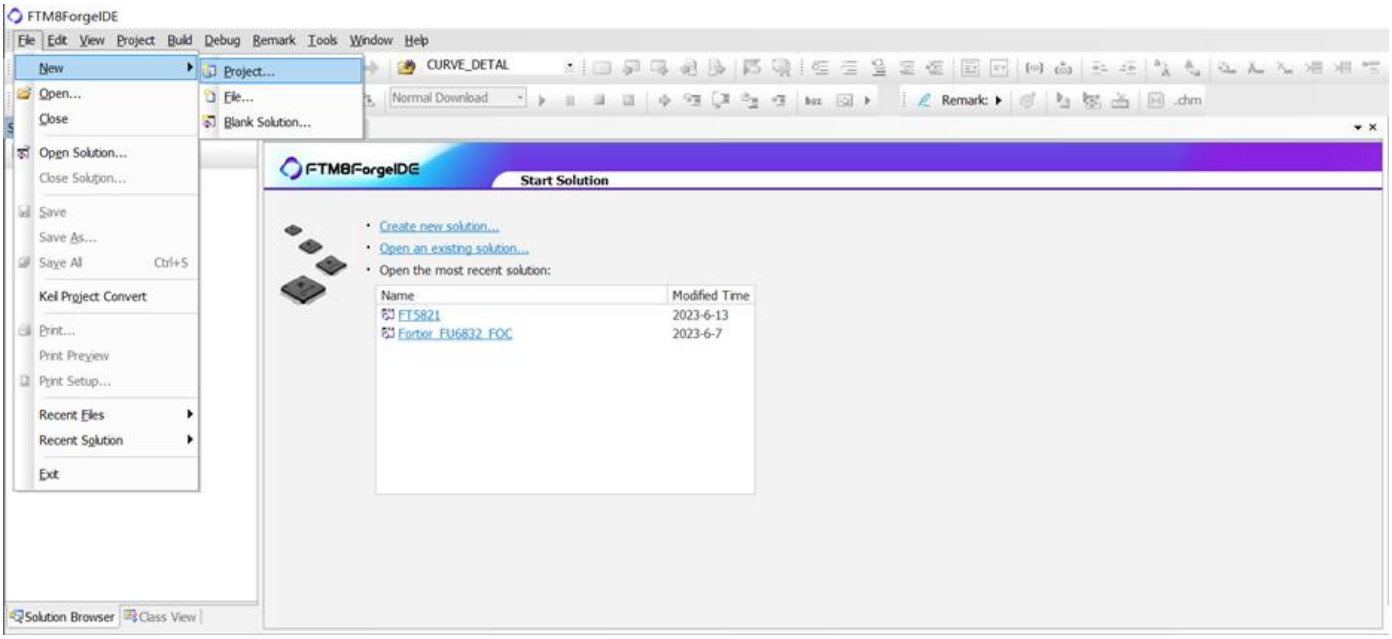


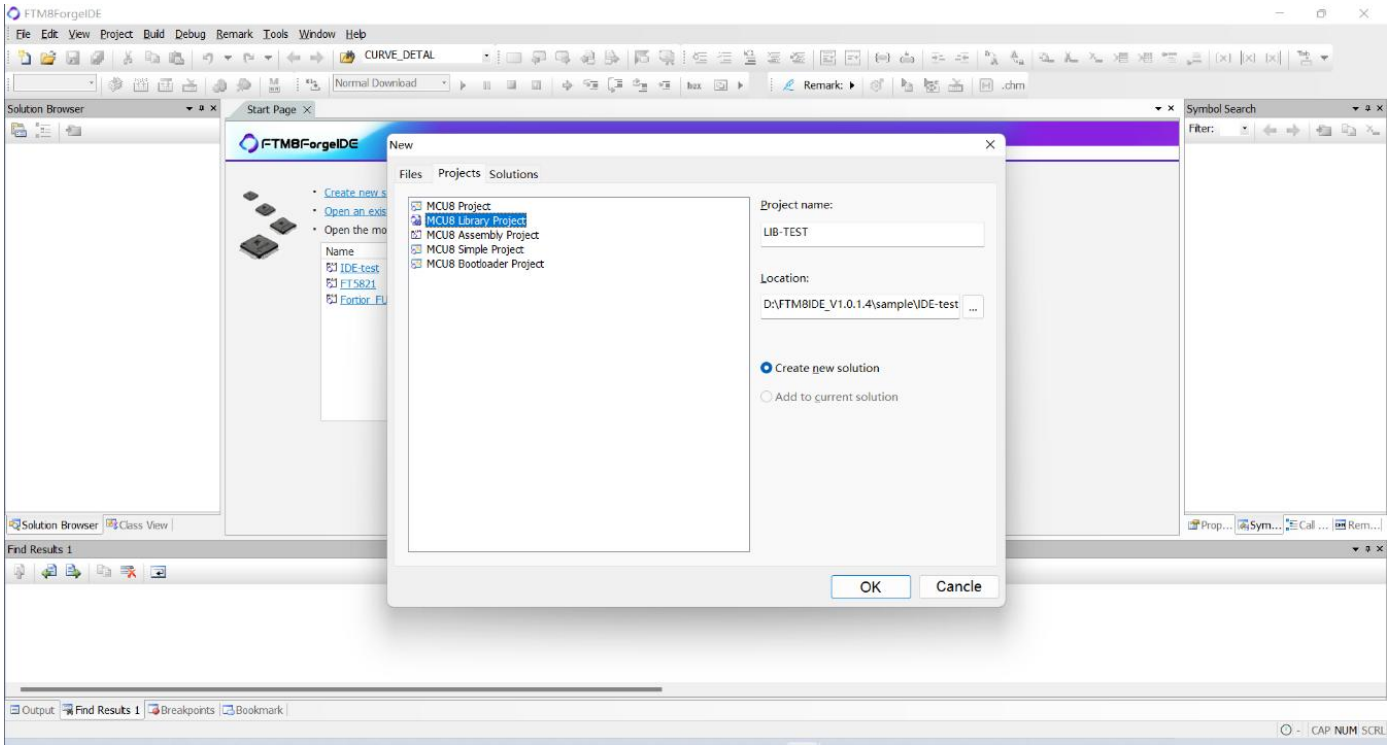
Click OK. After selecting corresponding chip model IDE generates the same directory structure as Keil. It is convenient for engineers to port and distinguish project files.



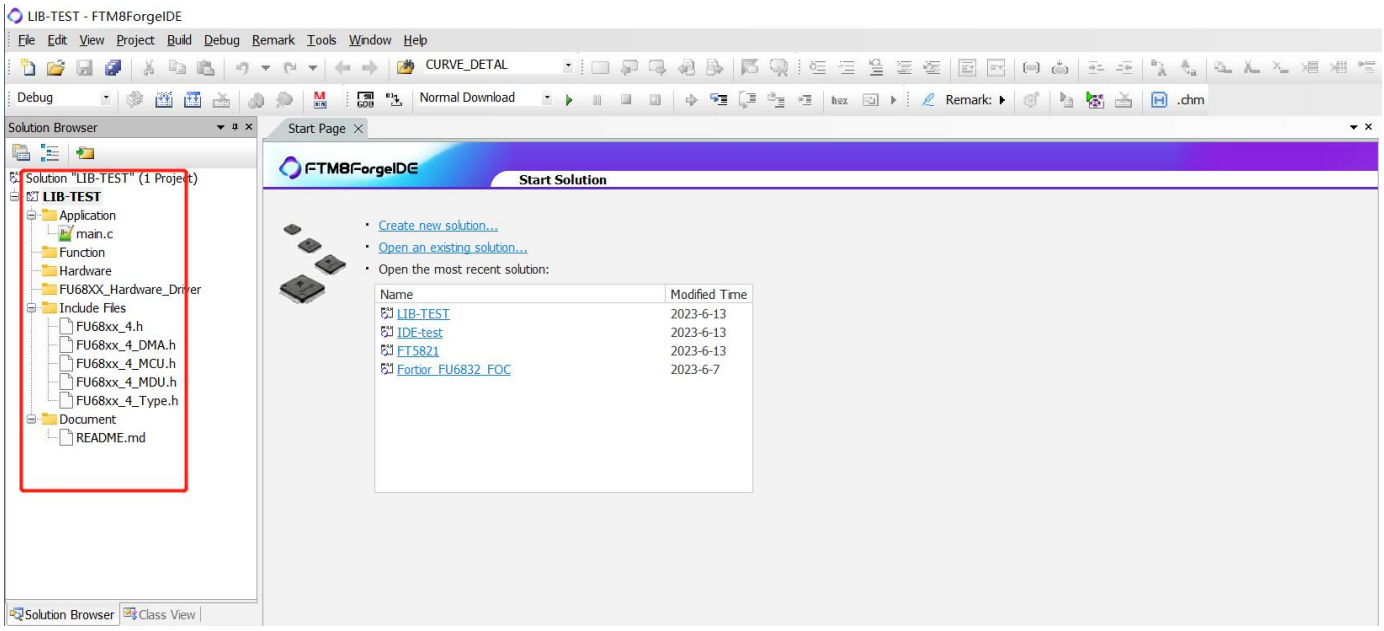
2.2 Create Lib Project

Start IDE, select File → NEW → Project, and then select MCU8 Library Project. Project name is the name of selected project, and Location is the address of the project to be stored. You can change them as required.



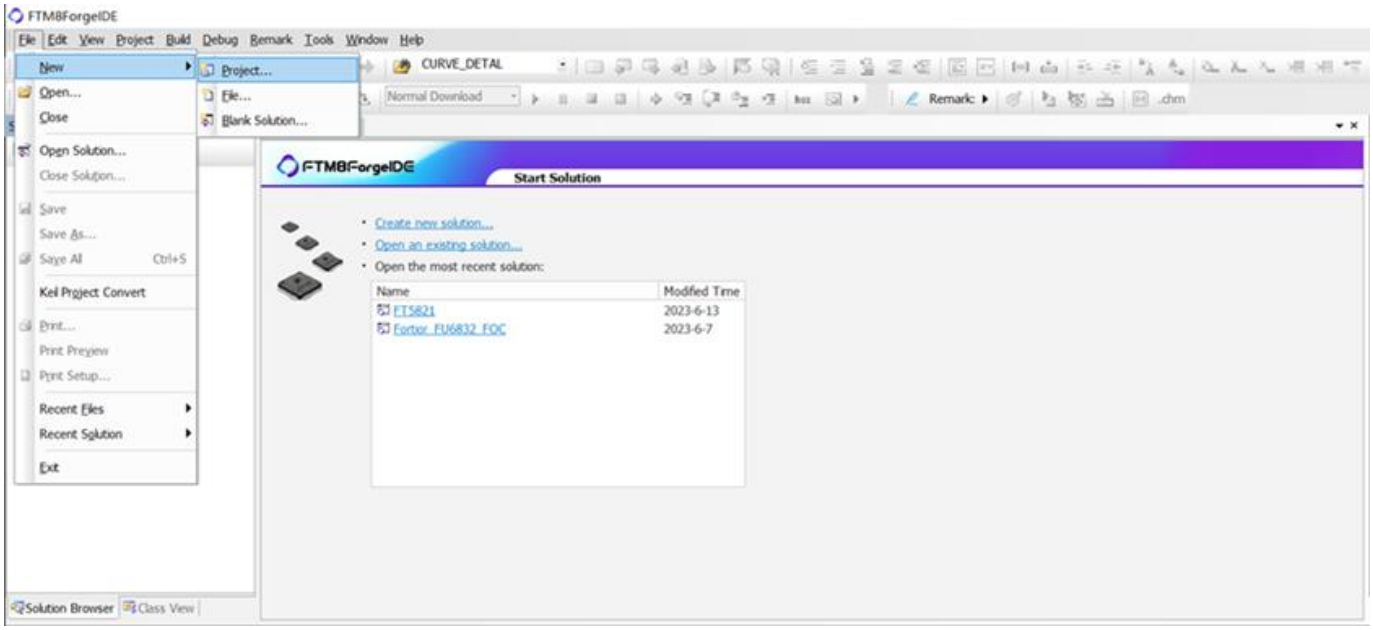


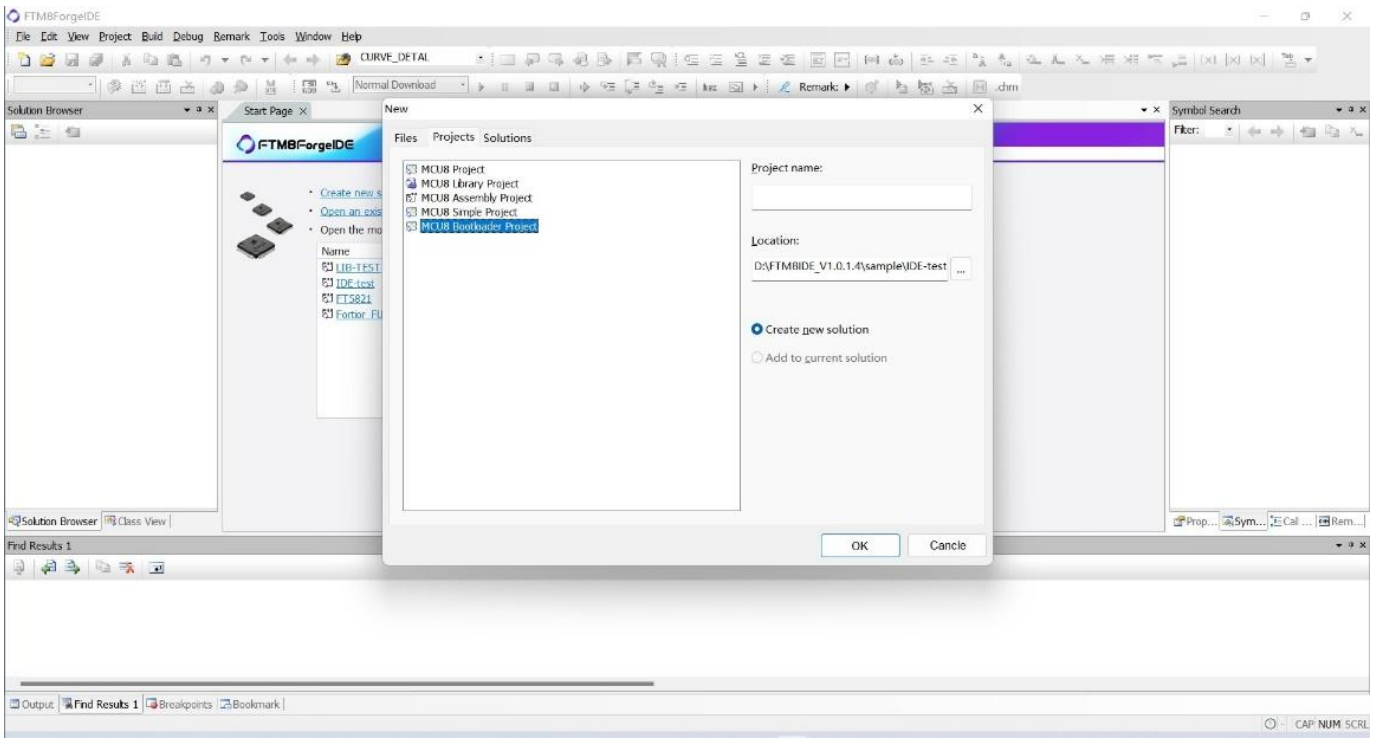
After selecting chip model, IDE creates a new project directory that is the same as common project directory. You can revise directory structure and involved files.



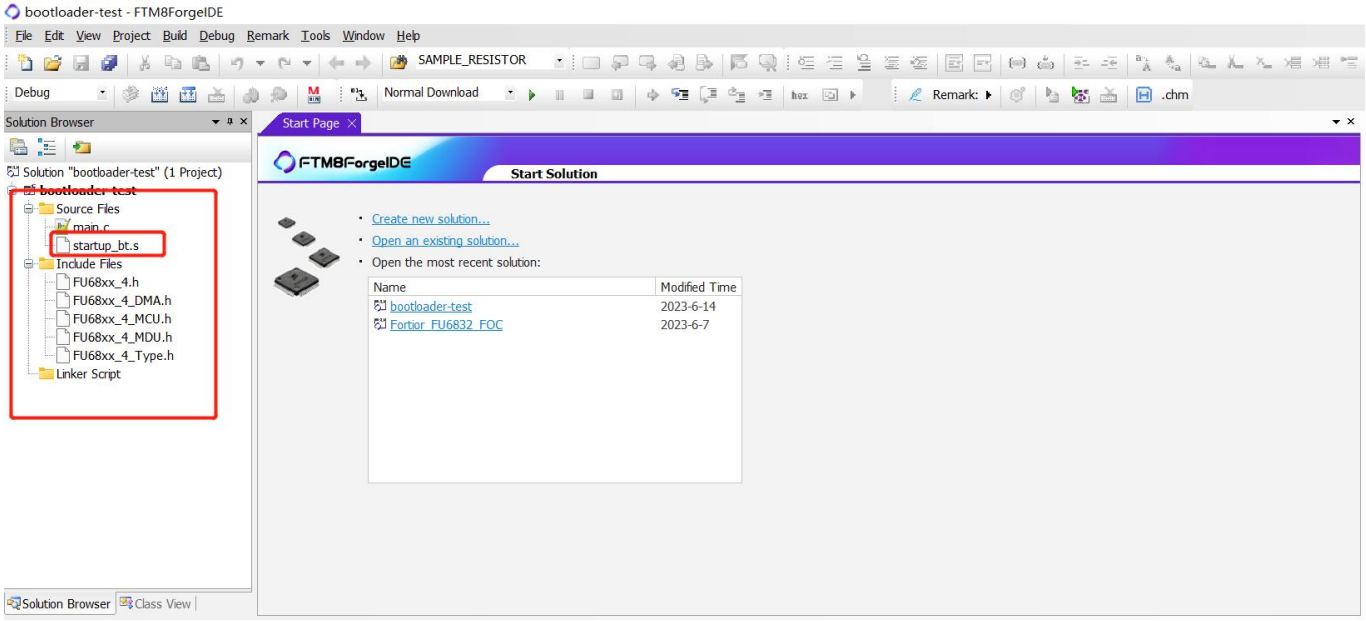
2.3 Create Bootloader Project

Start IDE, select File → NEW → Project, and then select MCU8 Bootloader Project. Project name is the name of selected project, and Location is the address of the project to be stored. You can change them as required.

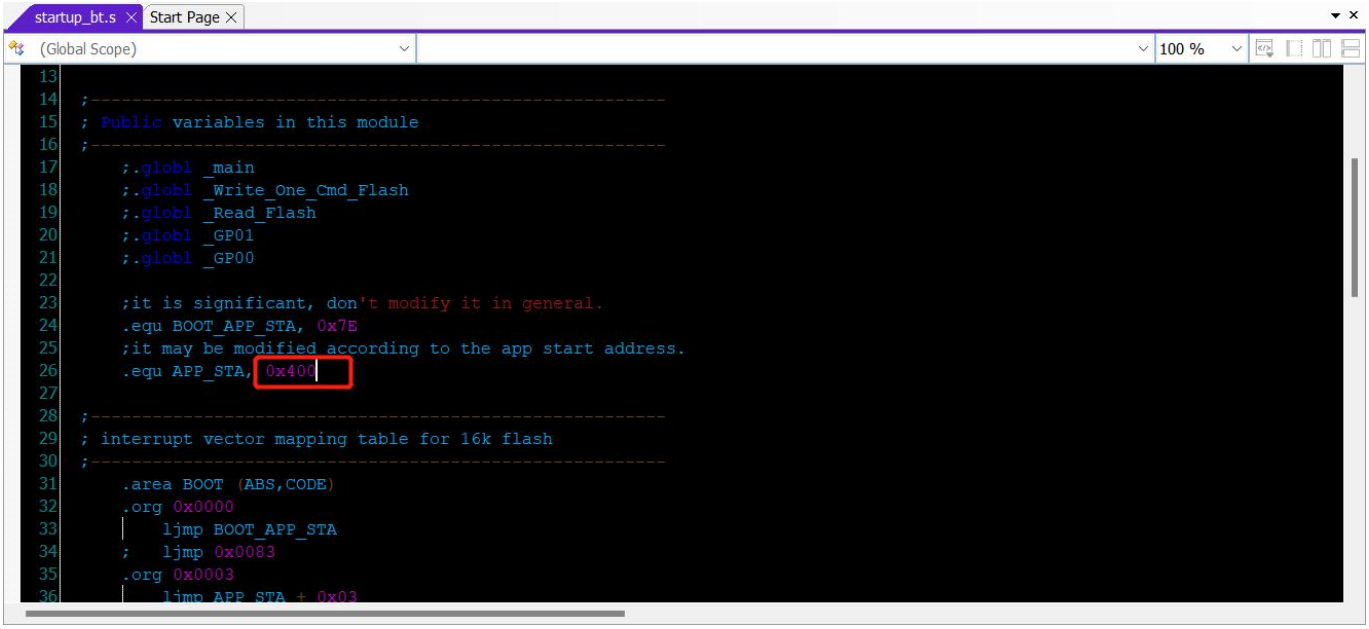

FTM8ForgeIDE
File Edit View Project Build Debug Remark Tools Window Help
New
Open...
Close
Open Solution...
Close Solution...
Save
Save As...
Save All Ctrl+S
Keil Project Convert
Print...
Print Preview
Print Setup...
Recent Files
Recent Solution
Exit
Project...
File...
Blank Solution...
CURVE_DETAL
Normal Download
Remark:
FTM8ForgeIDE
Start Solution
Create new solution...
Open an existing solution...
Open the most recent solution:
Name
Modfied Time
FT5821
2023-6-13
Fortior_FU6832_FOC
2023-6-7
Solution Browser
Class View


FTM8ForgeIDE
File Edit View Project Build Debug Remark Tools Window Help
CURVE_DETAL
Normal Download
Remark:
Solution Browser
Start Page
FTM8ForgeIDE
Symbol Search
Filter:
New
Files
Projects
Solutions
MCU8 Project
MCU8 Library Project
MCU8 Assembly Project
MCU8 Simple Project
MCU8 Bootloader Project
Project name:
Location:
D:\FTM8IDE_V1.0.1.4\sample\IDE-test
Create new solution
Add to current solution
OK
Cancle
Find Results 1
Output
Find Results 1
Breakpoints
Bookmark
CAP NUM SCRL

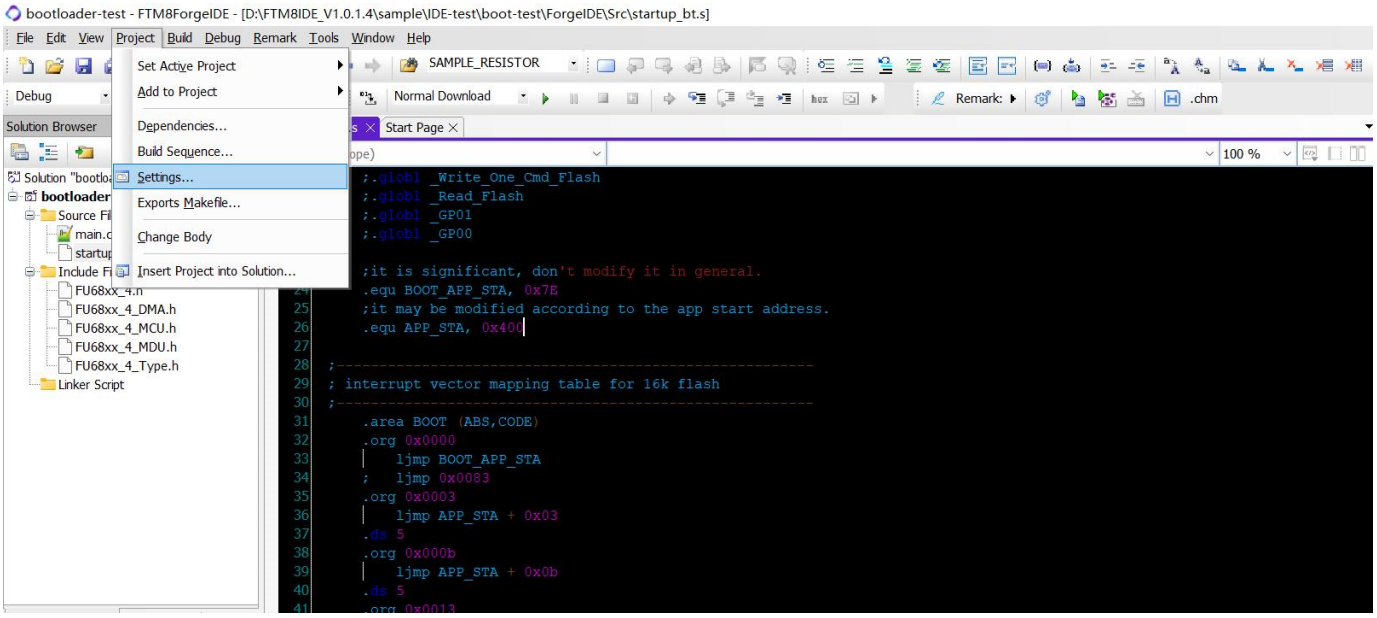
After selecting chip model, IDE creates a new bootloader project directory that is different from common project directory. You can revise directory structure and involved files. Meanwhile, bootloader generates startup_bt.s to revise address mapping.

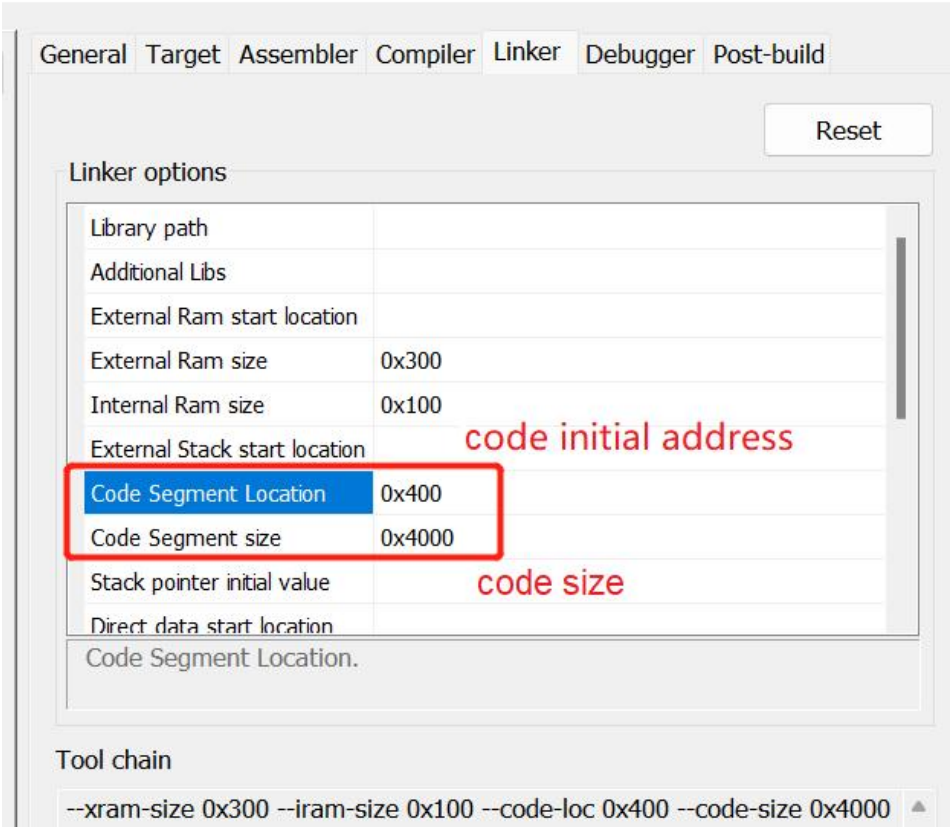


In Startup file, revise the address in the red flame as start address after shifting. The program executes from this address.



You shall revise code start address of the main project of App. Select Project→Settings→Linker and set initial address of App in Code Segment Location and code size in Code Segment Size.

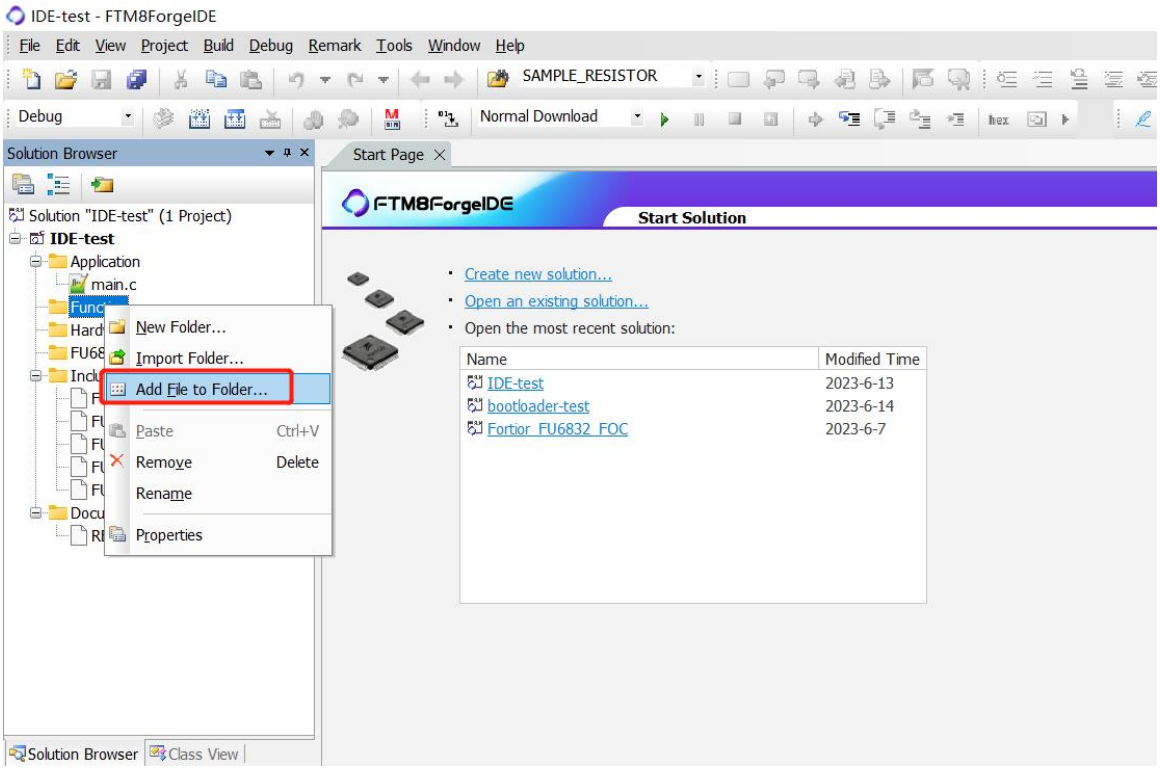

bootloader-test - FTM8ForgeIDE - [D:\FTM8IDE_V1.0.1.4\sample\IDE-test\boot-test\ForgeIDE\Src\startup_bt.s]
File Edit View Project Build Debug Remark Tools Window Help
Set Active Project
Add to Project
Dependencies...
Build Sequence...
Settings...
Exports Makefile...
Change Body
Insert Project into Solution...
SAMPLE_RESISTOR
Normal Download
Debug
Solution Browser
Start Page
100 %
Solution "bootloa
bootloader
Source Fi
main.c
startup
Include Fi
FU68xx_4.h
FU68xx_4_DMA.h
FU68xx_4_MCU.h
FU68xx_4_MDU.h
FU68xx_4_Type.h
Linker Script
;.globl _Write_One_Cmd_Flash
;.globl _Read_Flash
;.globl _GP01
;.globl _GP00
;it is significant, don't modify it in general.
.equ BOOT_APP_STA, 0x7E
;it may be modified according to the app start address.
.equ APP_STA, 0x400
; interrupt vector mapping table for 16k flash
.area BOOT (ABS,CODE)
.org 0x0000
ljmp BOOT_APP_STA
; ljmp 0x0083
.org 0x0003
ljmp APP_STA + 0x03
.ds 5
.org 0x000b
ljmp APP_STA + 0x0b
.ds 5


General
Target
Assembler
Compiler
Linker
Debugger
Post-build
Reset
Linker options
Library path
Additional Libs
External Ram start location
External Ram size
0x300
Internal Ram size
0x100
External Stack start location
code initial address
Code Segment Location
0x400
Code Segment size
0x4000
Stack pointer initial value
code size
Direct data start location
Code Segment Location.
Tool chain
--xram-size 0x300 --iram-size 0x100 --code-loc 0x400 --code-size 0x4000

3 Add Files

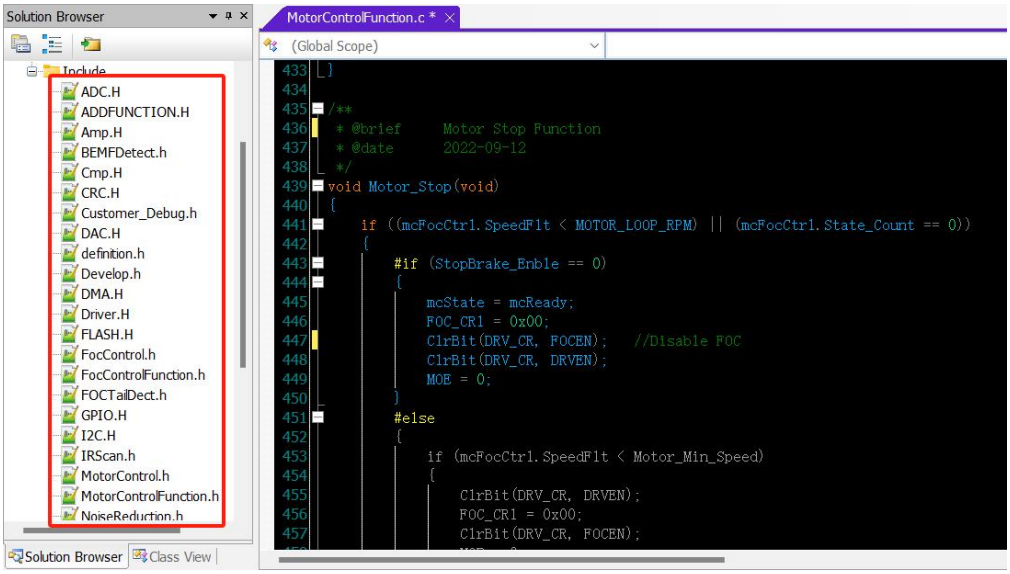
3.1 Add .c and .h Files

1. Select the folder and right click Add File to Project to add certain .c or .h files into present project



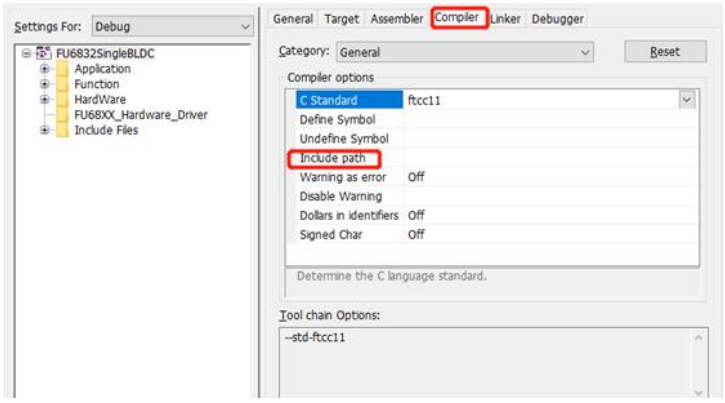
Note: It is suggested that the file to be added is included in local project directory, otherwise you shall add its path while selecting.

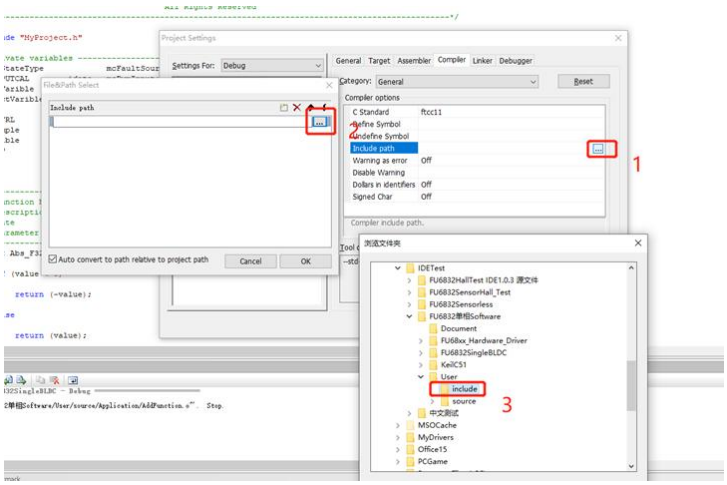
2. Include Files folder stored all .h files, including chip header files.

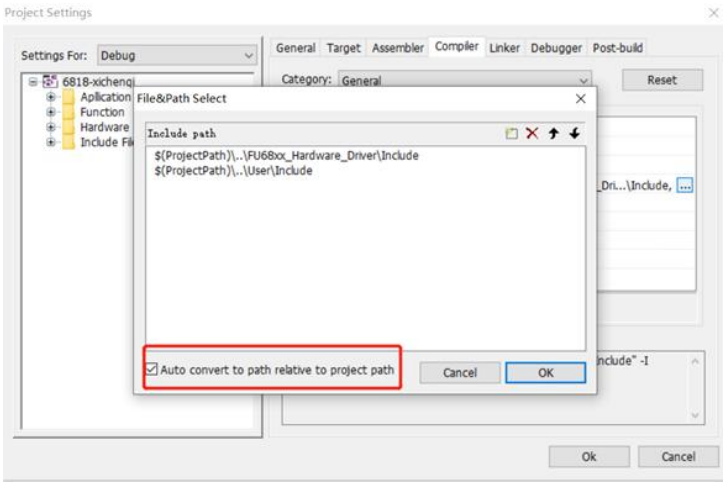


Note: Files to be referenced shall be added only once. You cannot add customer.h to include Files if it already exists in Application folder.

3. Header file(.h) reference path setting: Add all paths that include .h files into Include path under Project→Settings→Compiler (Normally, the paths that contain Include files in both FU68xx_Hardware_Driver and User shall be added.)





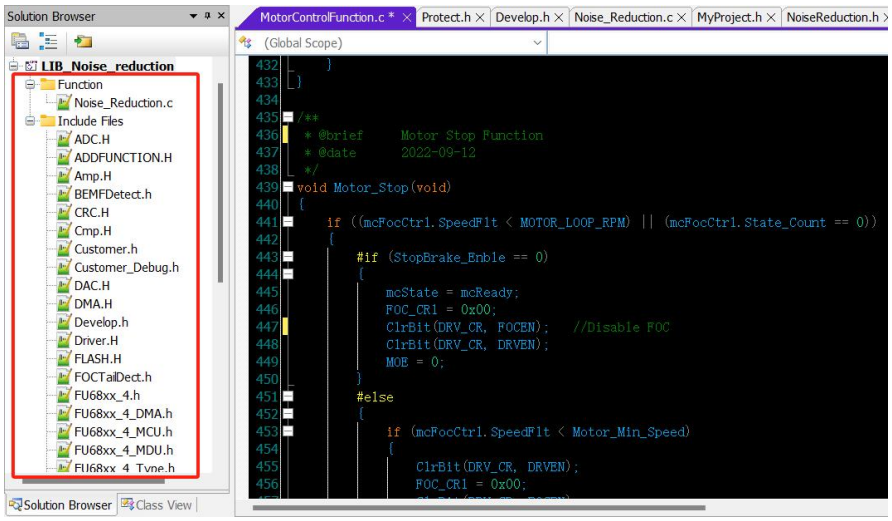


Note: Auto convert to path relative to project path (or relative path) must be selected so that the path changes along with the project path. Otherwise, you cannot access the path on other computer.

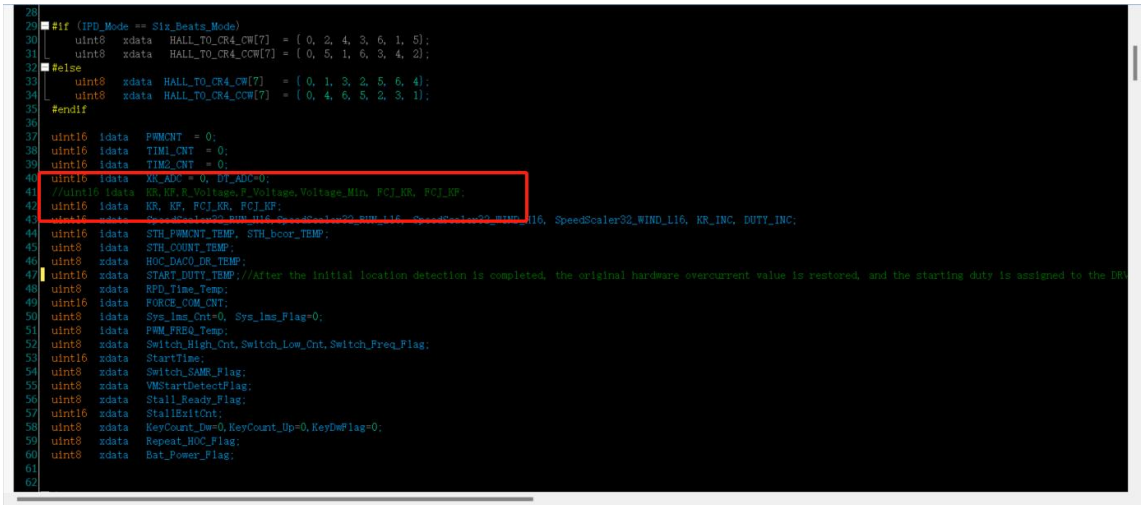
3.2 Add Lib Files

Since lib files in IDE are not compatible with those in Keil, they shall be generated in IDE with source code when lib files in Keil applied. **The lib project and the main project contain the same .h file.**

1. Add .h files and .c files into newly created Lib Project.

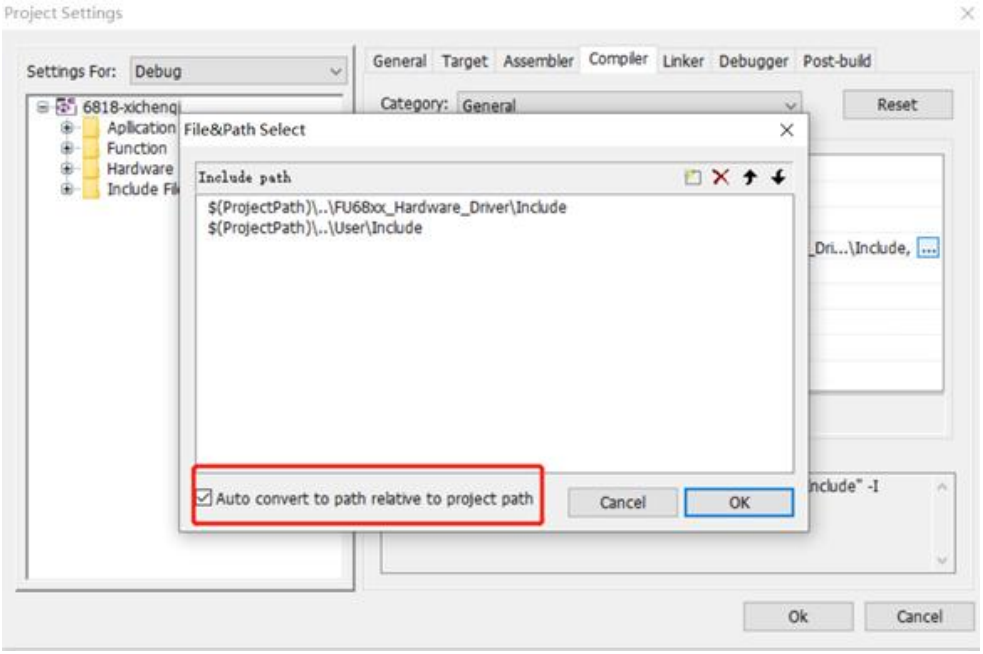


2. When protecting the librabry, if the variable is only used in the lib project and it is not used in the main project, then the variables shall be defined separately.



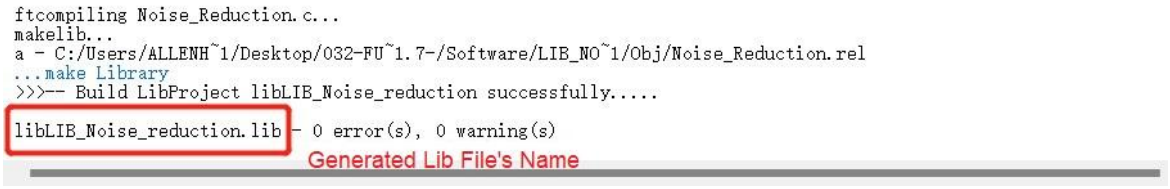
Example: If the variables in the red circle, R_Voltage, F_Voltage, and Voltage_Min are only used in the .c file of the lib project, but are not used in the main project, you shall define these three variables in the .c file of the lib project, otherwise the main project compilation shall report error that the three variables are not declared.

3. Set header files path: Add all paths that include .h files to Include path under Project→Settings→Compiler.

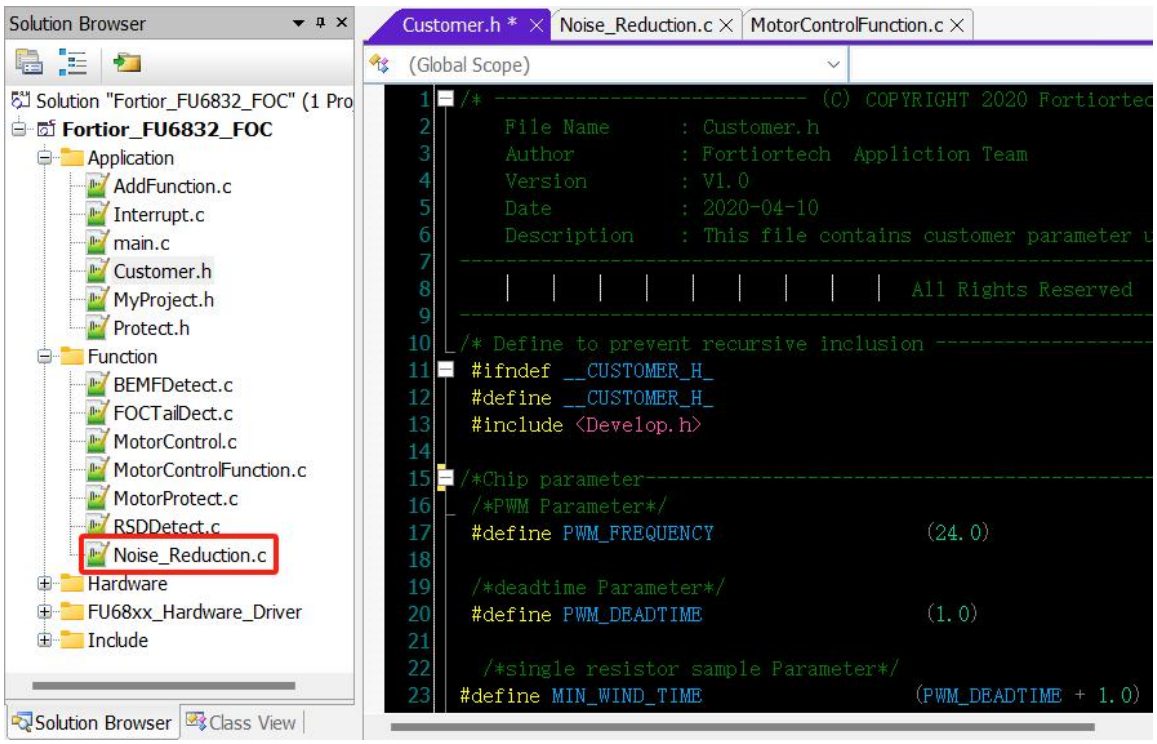


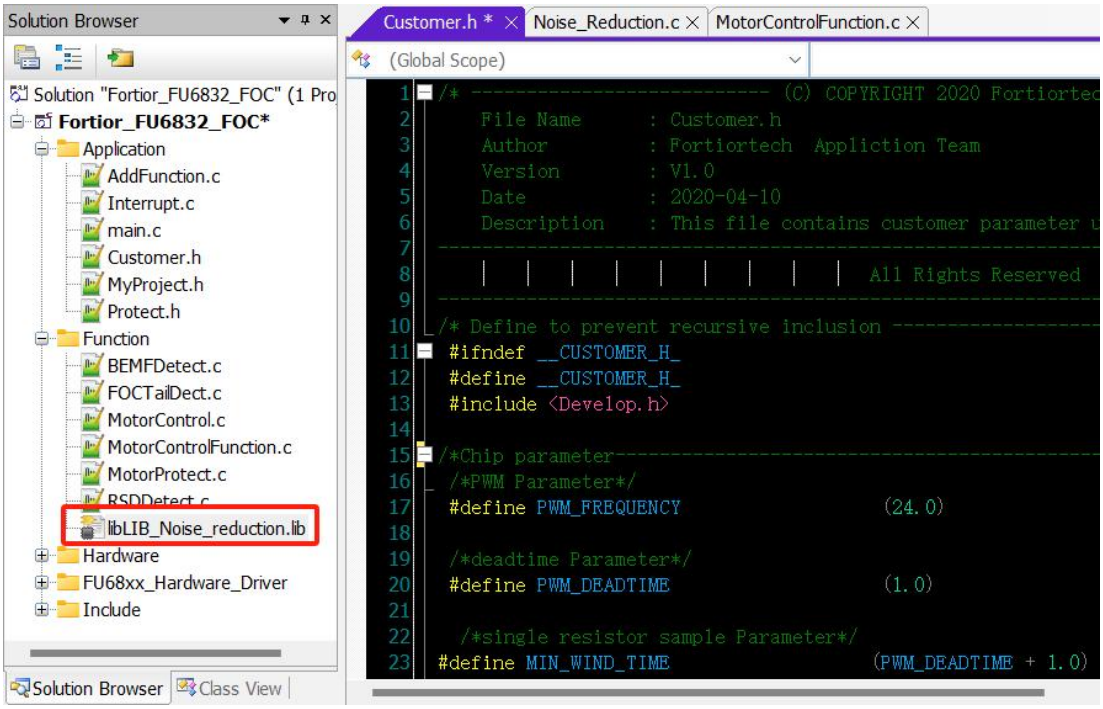
Note: Auto convert to path relative to project path (or relative path) must be selected.

4. Compile and generate lib files, whose paths are included in the debug file.

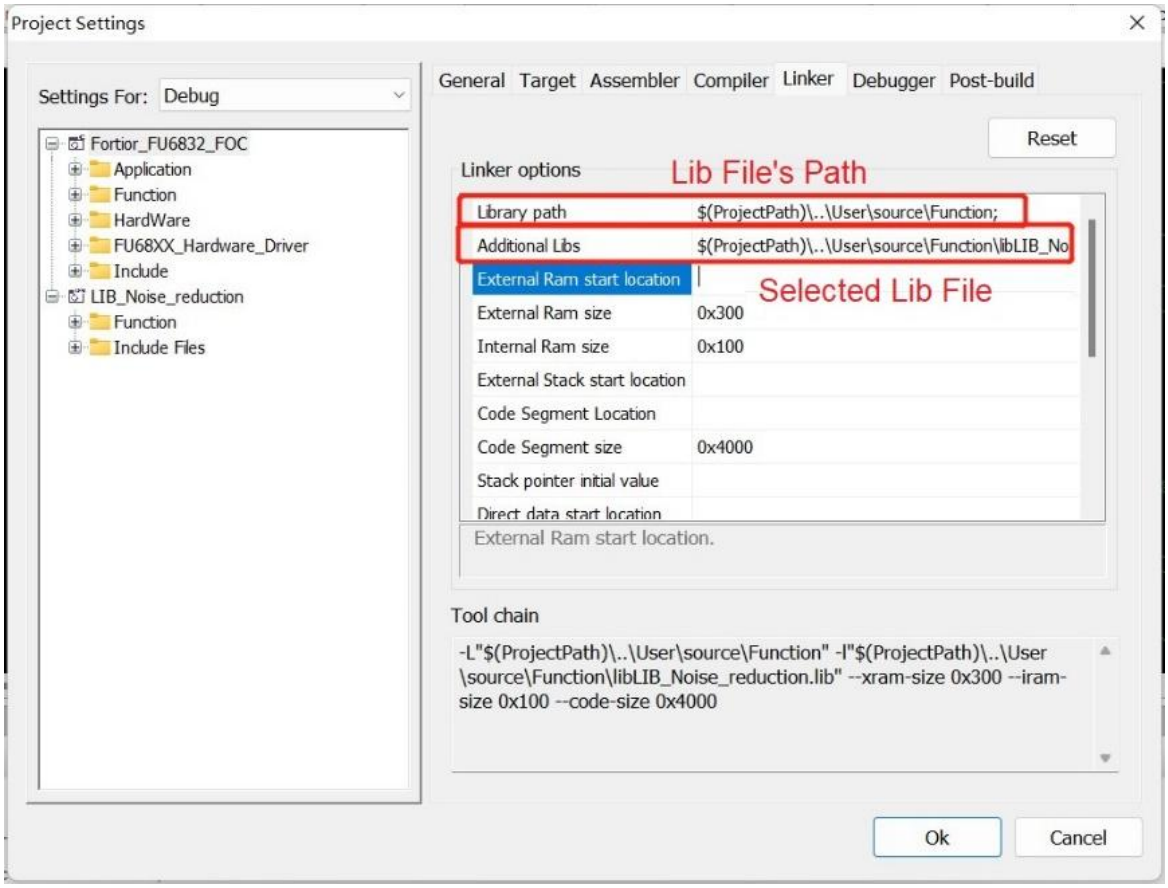


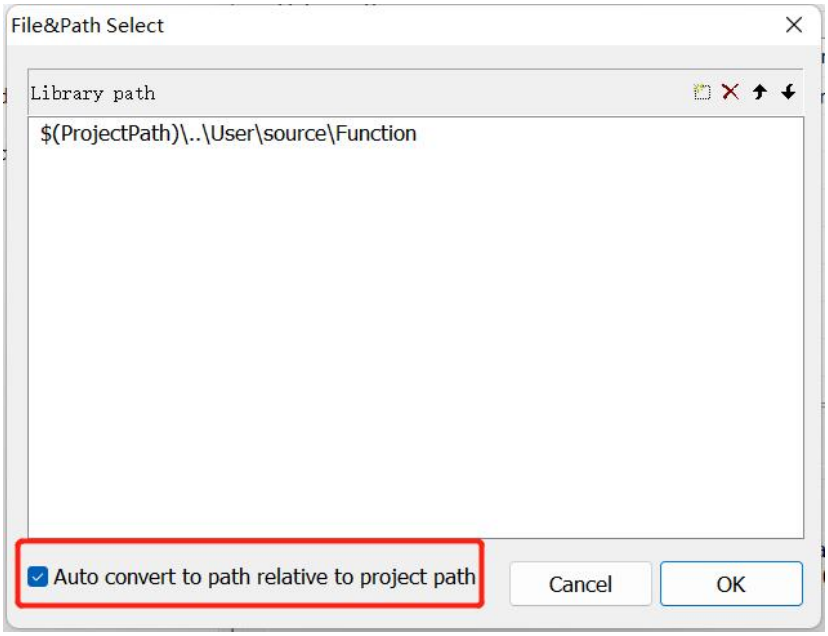
5. Add lib files into the source project, delete **.c** files to be wrapped, and add lib files generated by IDE.





6. Add link path of lib files.





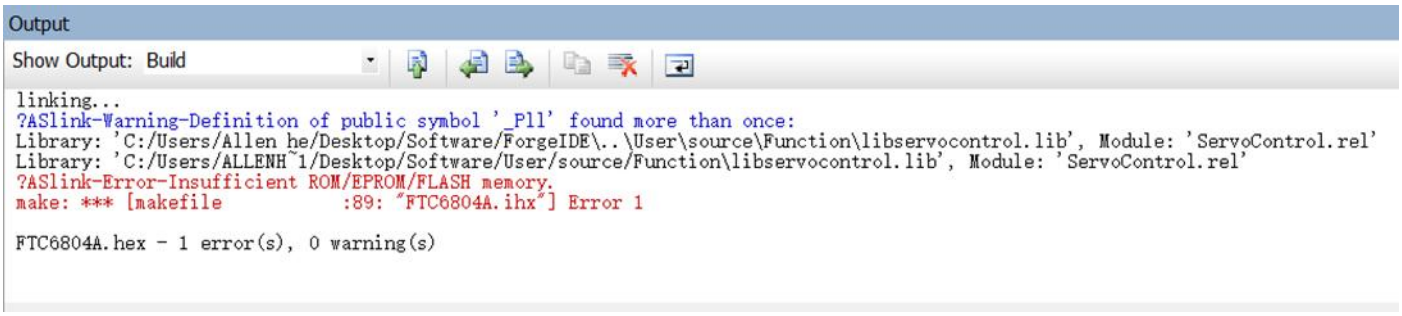
Note: Auto convert to path relative to project path (or relative path) must be selected. It is better to place lib files into the same project directory as other project files.

7. Wrapper library runs normally if code size after compilation is higher than or equal to the one before compilation. If not, it is high probability that the program is wrong.

```
Memory summary:
    Name             Start    End      Size     Max
    ---------------- -------- -------- -------- --------
    DATA RAM           0x08     0x0c        5      120
    IDATA RAM          0x21     0x5a       58      248
    EXTERNAL RAM     0x0001   0x0102      258      768
    ROM/EPROM/FLASH  0x0000   0x1bf1     7154    16384
```

3.2.1 Library Wrapping

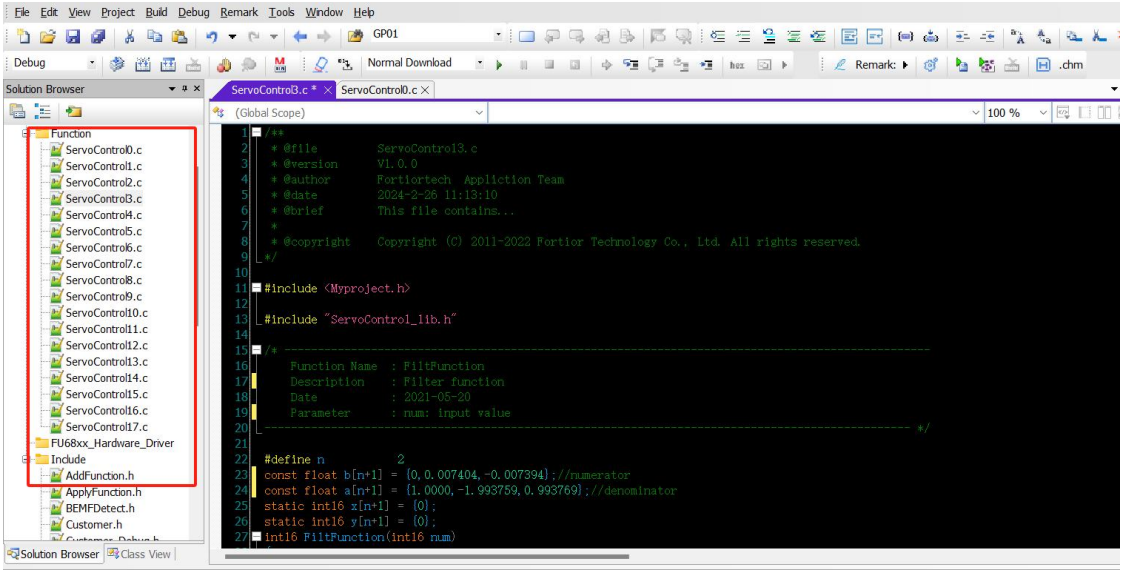
If the program appears to be over Flash or the code size is bigger than 0.5K or more than the size before coding, then you shall pay attention to the coding redundancy of the coding file, the compiler of IDE only segments the program by file, which means that when linking, as long as a certain function in the file is used, all the functions in the file shall be linked in, which leads to the introduction of some useless code, resulting in the increase of code size or over Flash. This leads to a code size increase or flash overload.



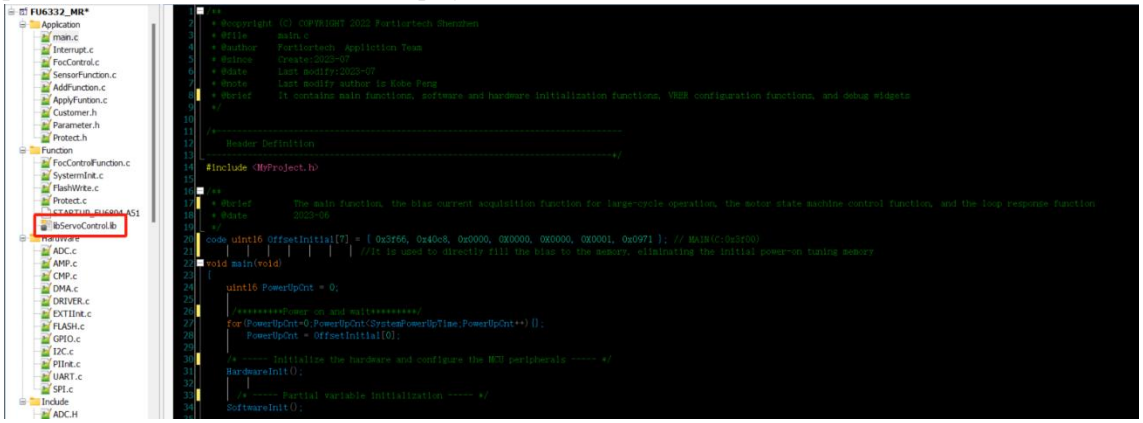
In order to solve this problem, you shall split the functions in the source file or functions of the same class into individual .c files to seal the library. IDE supports automatic splitting of functions without manual decomposition.

Steps:

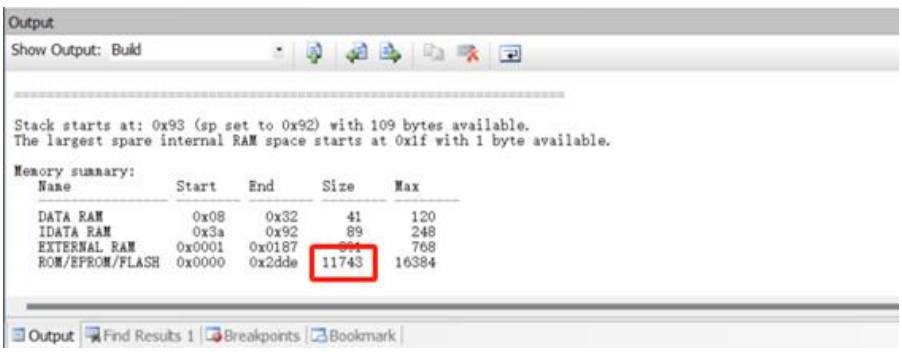
1. Click “File” → “Keil Project Convert” after IDE startup.
2. Select Keil Project to be converted and .h file and then click “Convert”. If the register name is different from the .h file defined after the .h file is converted, the register name shall be in accordance with .h file defined. The lib project and main project shall contain the same .h file.
3. After the conversion is complete, you shall see in the left browser window that the source library file has been split into separate .c files.



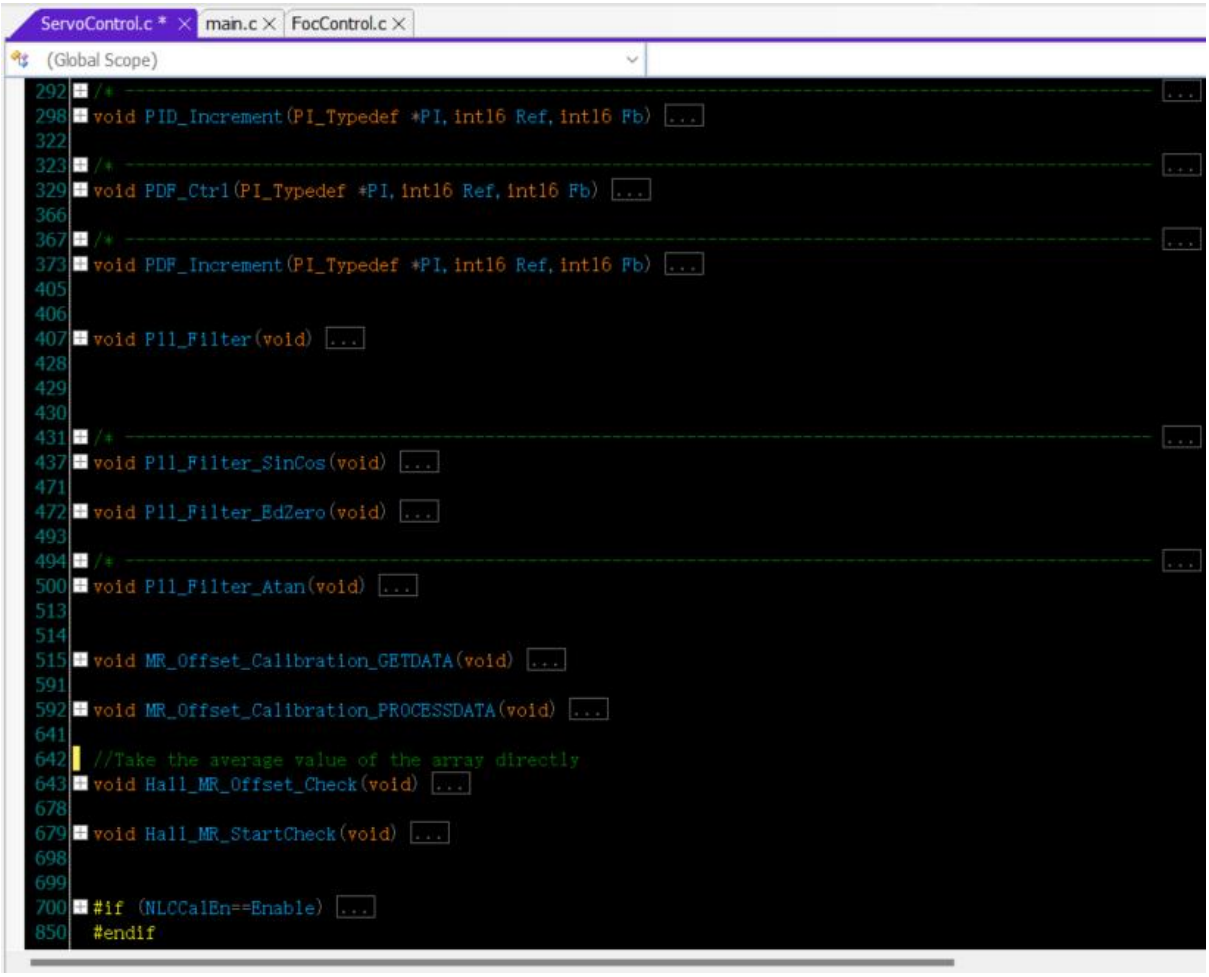
4. Compile the lib file and add it to the source project.



5. Compile to confirm that the code size after library wrapping is basically the same as before wrapping it.

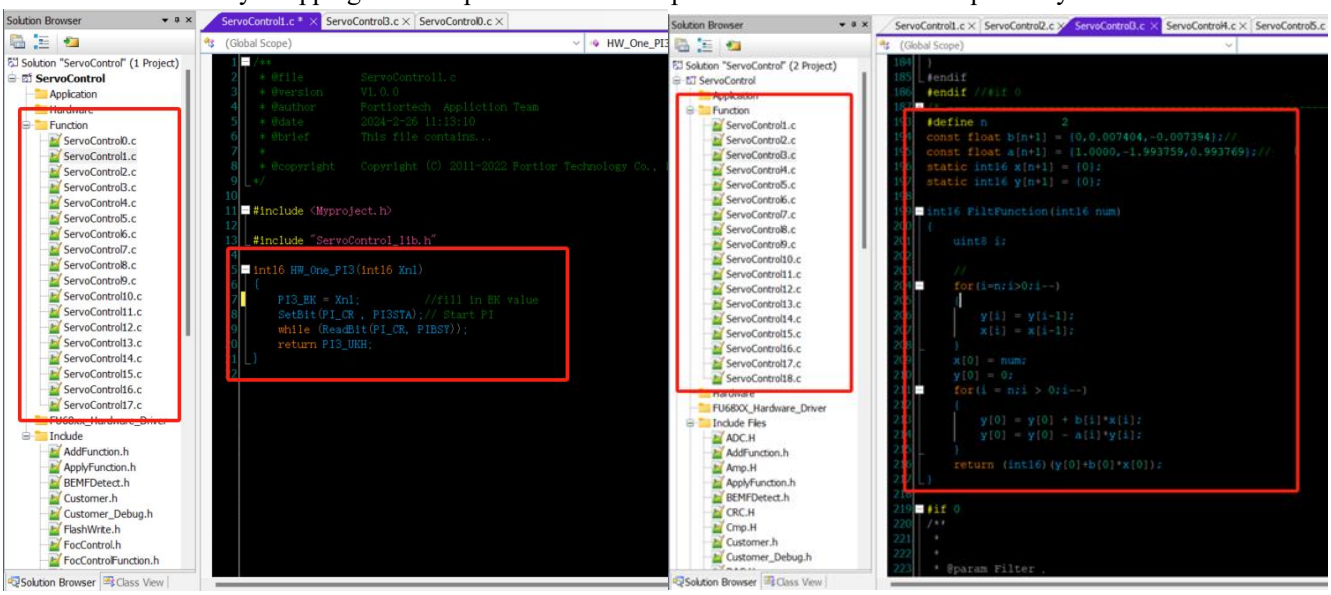


Note: If an exception occurs in the auto-split function, you shall manually split the function.

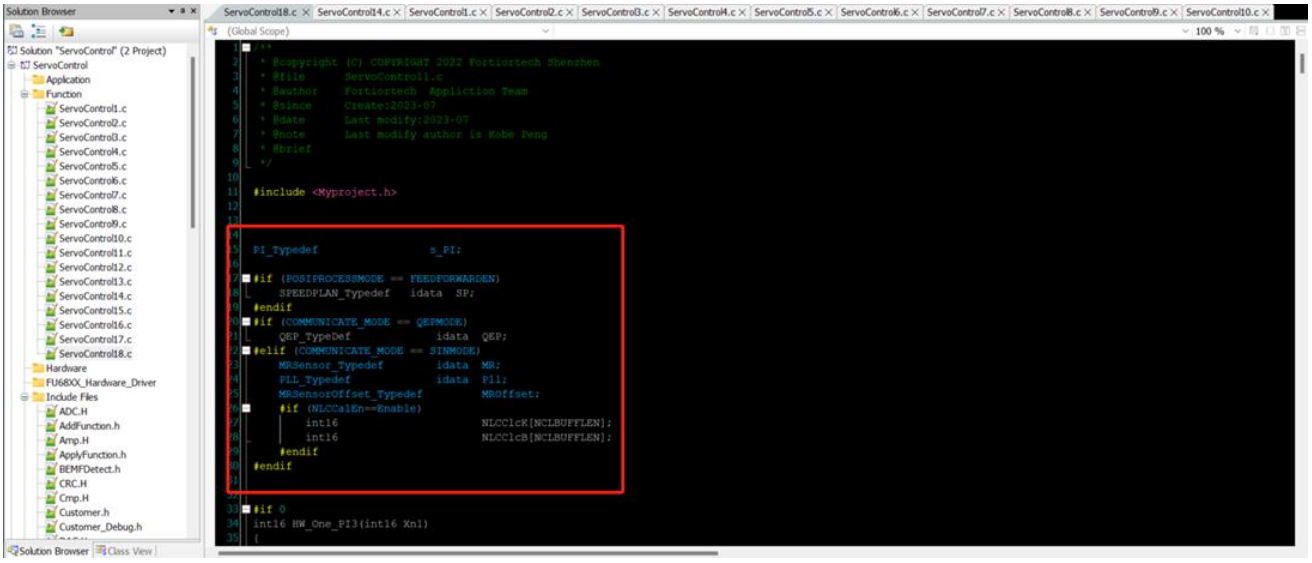


1） Set up new Lib project, see sector 2.2 for more details.

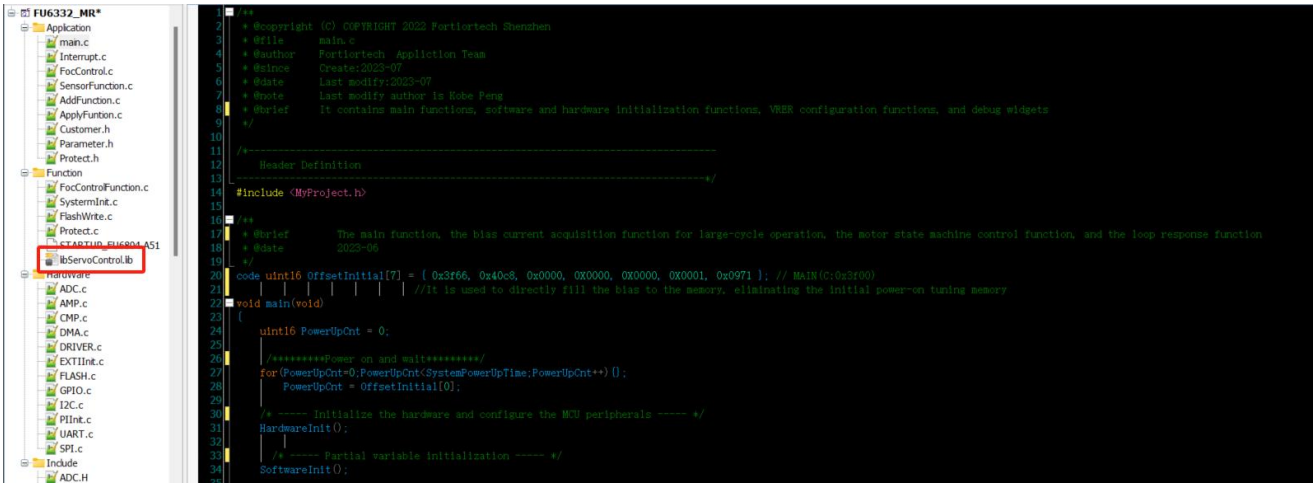
2） Add library wrapping file and split function into separate .c file and name it sequentially.



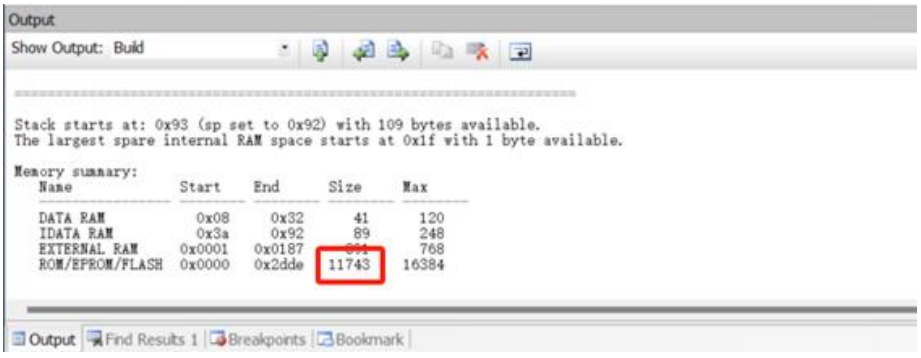
Note: Global variables in the source file shall also be split into a separate .c file.



3） Compile an generate lib project and add it into source project.



4） Compile and confirm that the code size after protecting the library is basically the same as before protecting.



4 Revision History

Rev.	Description	Date	Prepared by
V1.7	First release, translated from Chinese version 1.7	2024/03/25	Freya Fu

Copyright Notice

Fortior Technology (Shenzhen) Co., Ltd.
Room203, 2/F, Building No.11, Keji Central Road2,
Software Park, High-Tech Industrial Park, Shenzhen, P.R. China 518057
Tel: 0755-26867710
Fax: 0755-26867715
URL: http://www.fortiortech.com