ASSEMBLY LANGUAGE PROGRAMMING WITH KEIL U VISION 5

The fundamental concepts associated with assembly language programming using the LPC2378 microcontroller are discussed. Topics include an introduction to Keil u Vision and assembly language programming.

4.1 Introduction to KEIL u VISION

It is very similar to the AVR Studio except that it has an additional feature, as explained below

Keil u Vision is an IDE directed towards code development for multiple platforms like AVR, ARM, CORTEX-M, C166, C251, C51 and 8051 based MCU architectures manufactured by various companies. In comparison, Atmel Studio is a Visual Basic and .NET Framework-based IDE which only supports AVR and ARM architecture based MCU's only by Atmel.

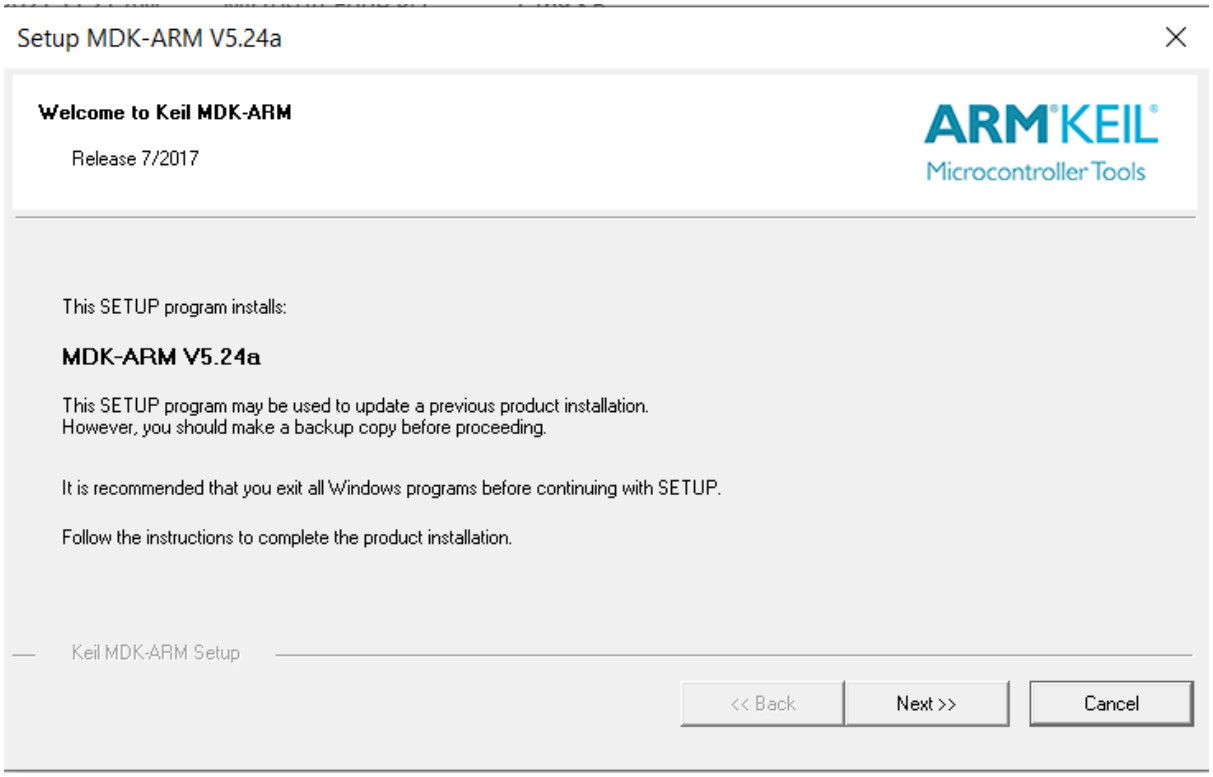
4.1.1 Downloading and installing KEIL micro vision

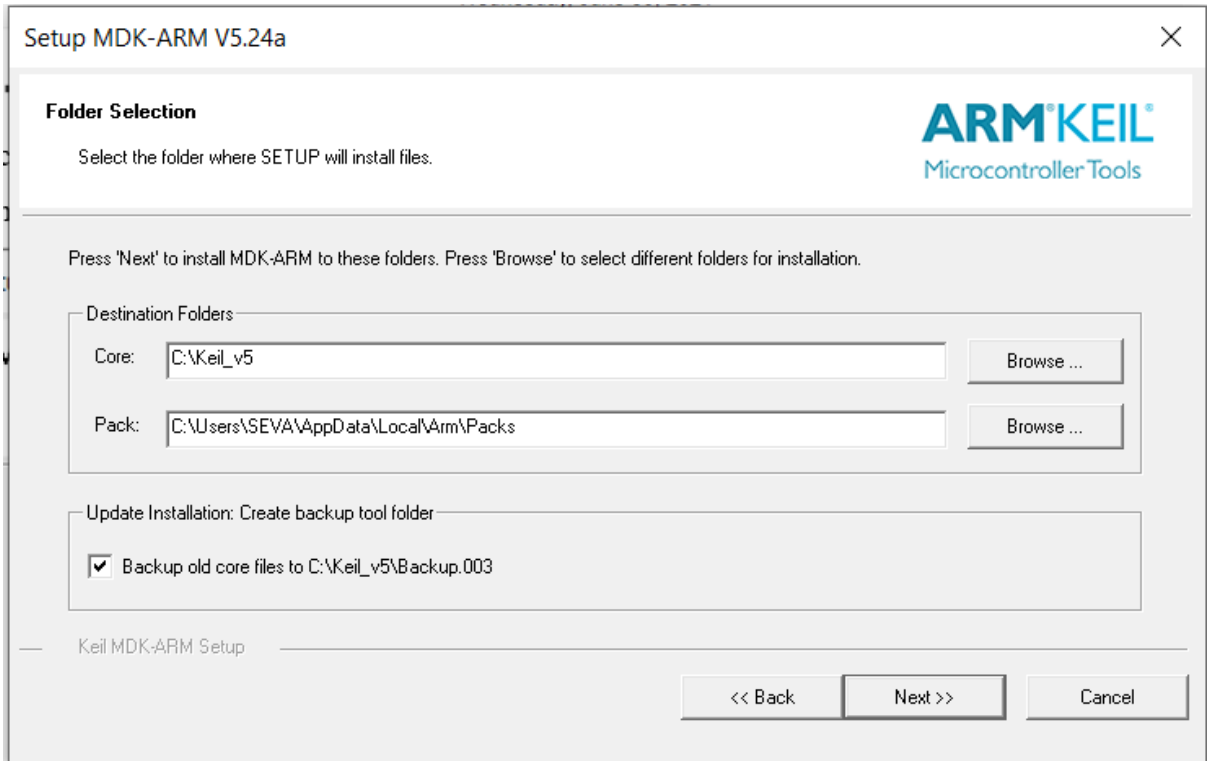
https://www2.keil.com/mdk5

The latest Keil IDE version downloaded from the Keil website is lighter and does not support ARM 7 devices. Hence, I am sharing with you the installation file for Keil, which supports ARM7 devices. Please install these files rather than downloading them from the Keil website

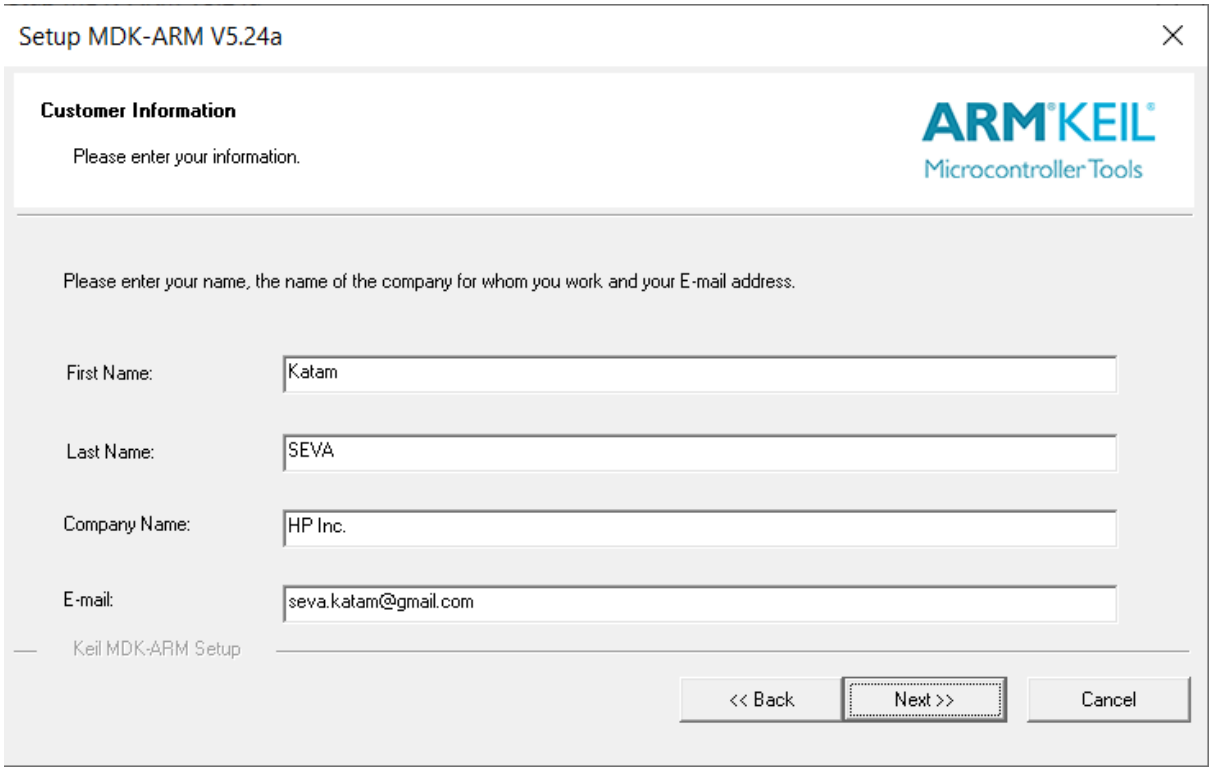
https://drive.google.com/file/d/1lMIAzuX5I9E-X9eDi3MFcOpvA0dA5cQS/view?usp=sharing

Run the downloaded program to install the Keil u vision IDE.

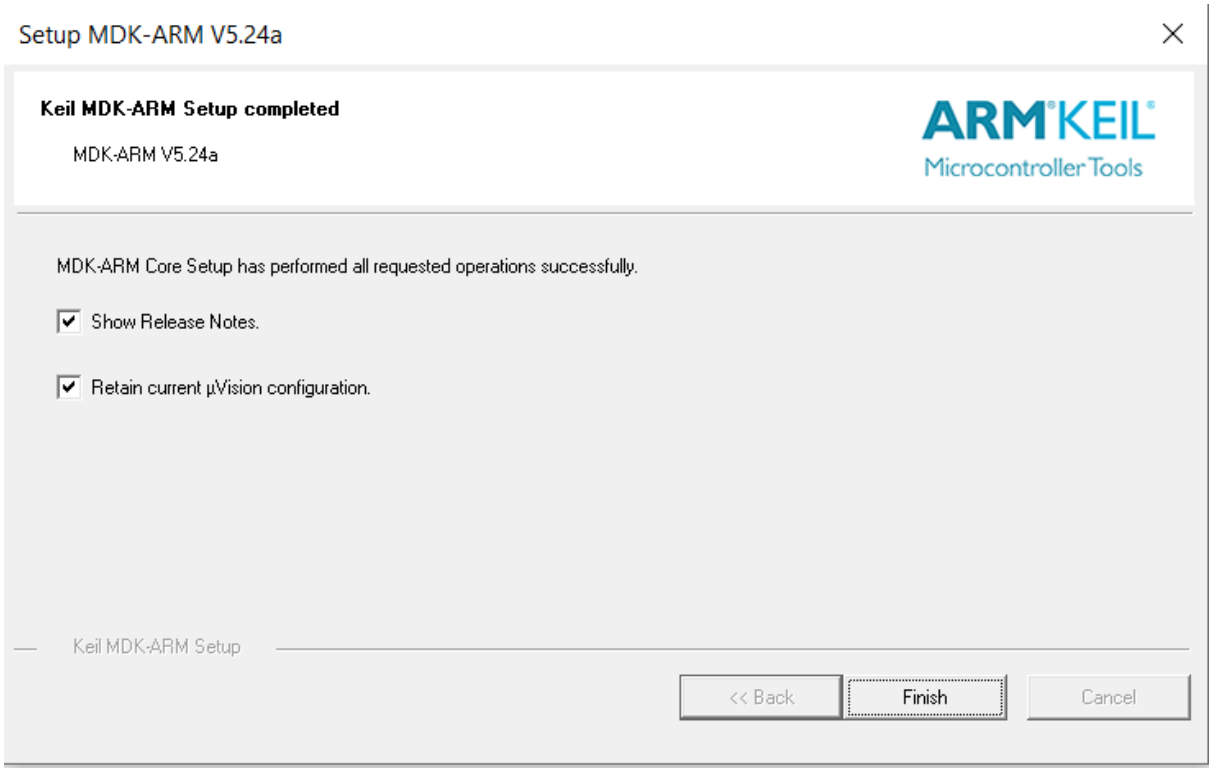




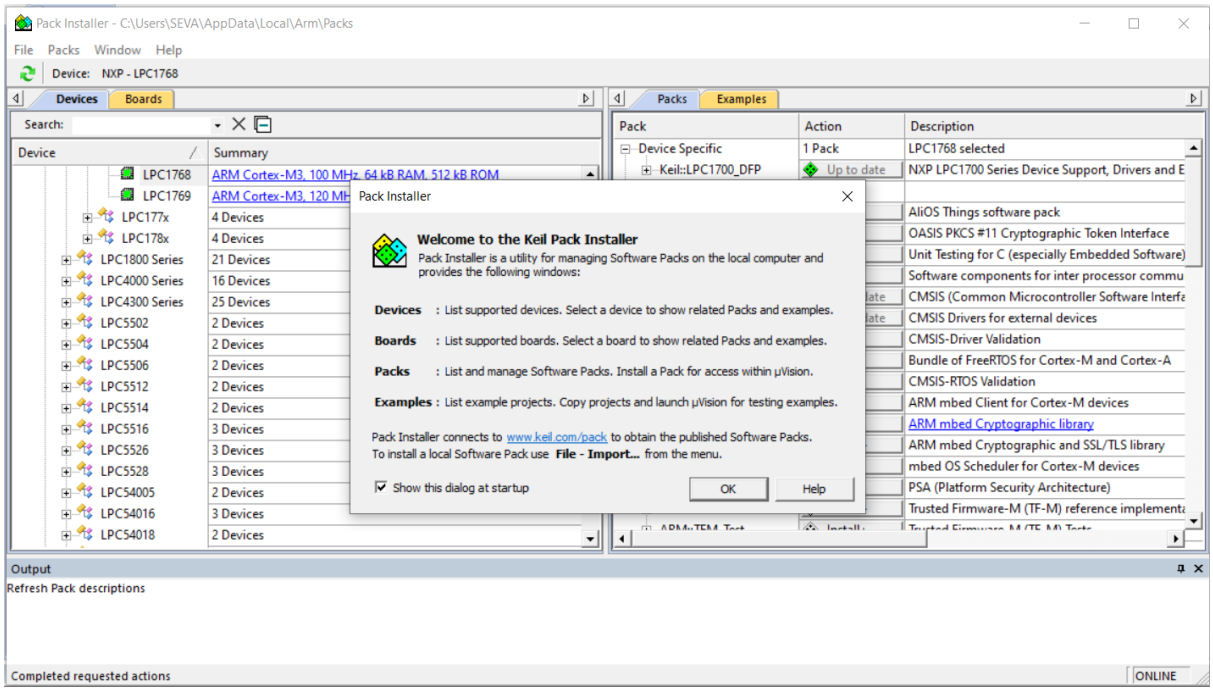
Store in the default destination folder.



Enter details and Click Next



Click Finish to complete installation



Click OK and close the tab

4.1.2. Part 2 of Installation:

ARM9 & ARM7 based Microcontrollers like LPC2378, other LPC3000 series MCUs, etc. are now "legacy stuff" in Keil uVision5 and are not available in the default installation. To enable support for **ARM7 LPC2378 and similar devices,** you need to install the **Legacy Support pack for ARM7, ARM9 & Cortex-R,** after which you can create projects for the legacy MCU without any fuzz. Legacy files will also enable backwards compatibility for projects made using **Keil MDK v4**.

https://www2.keil.com/mdk5/legacy

download MDK79523.EXE

https://drive.google.com/file/d/1UbOFM-myqkXl-ri0UjI7MgRPcCX5lzm_/view?usp=sharing

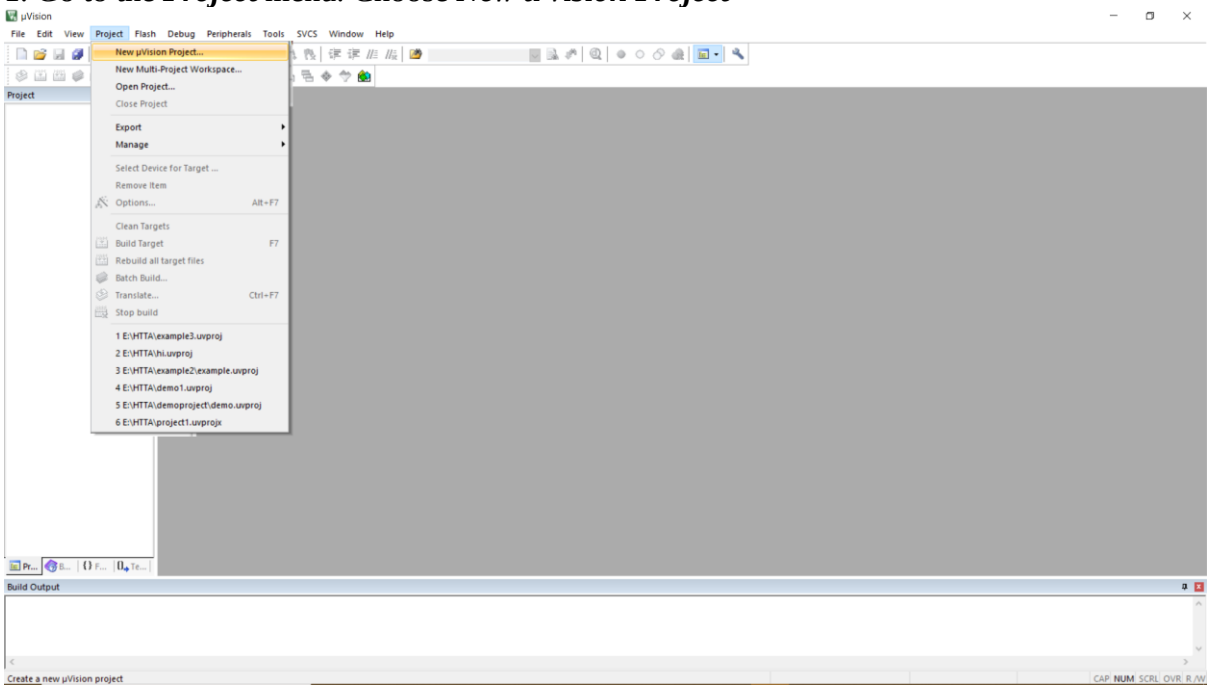
Proceed through similar steps and install this legacy pack for Keil

4.1.3. Opening Keil uvision5

Go to the ***Start*** menu and open Atmel Studio.

4.1.4 Creating the first project

1. Go to the ***Project*** menu. Choose ***New u vision Project***



2. In the opened dialog,

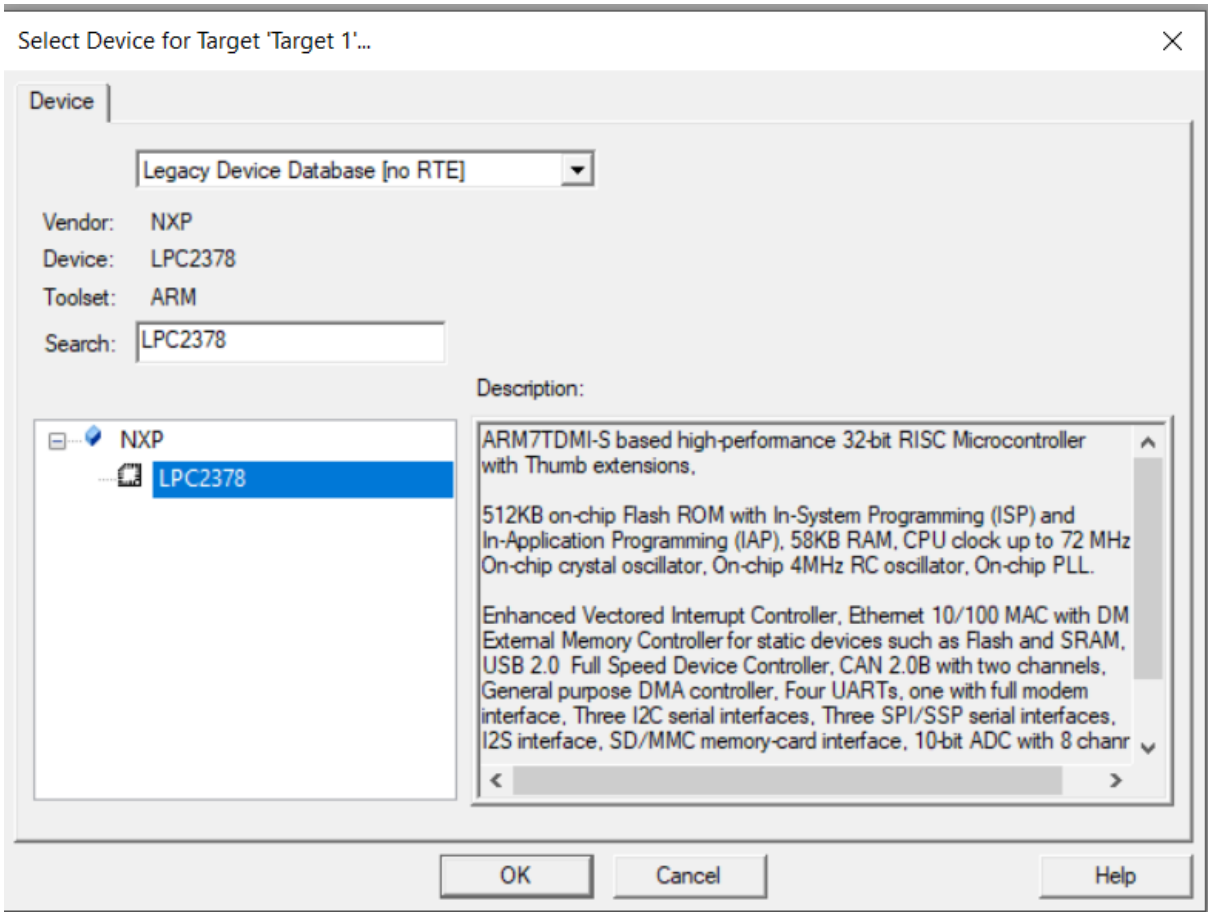
a. Name the project as ***demo1***.

Don't name randomly,

b. Each project will create listings and objects folders. So, creating a separate folder (folder ***project1***) for each project is recommended.

c. Choose the path where you like to save the project by clicking on the ***Browse*** button.

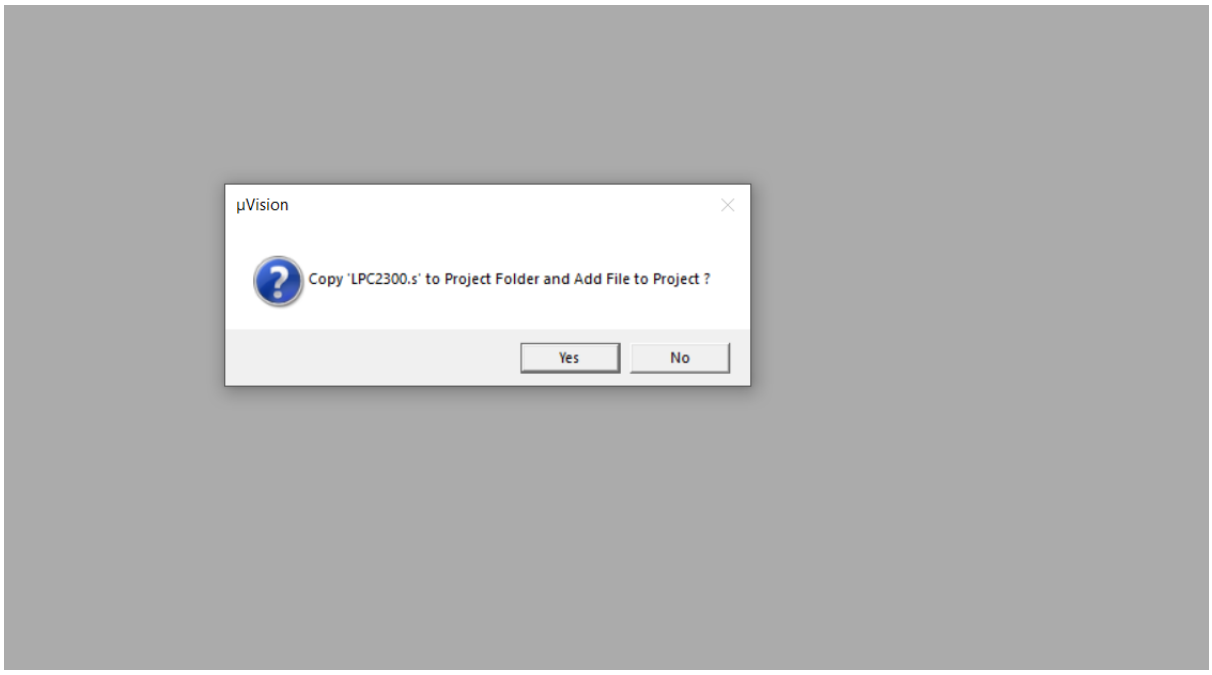
d. Press OK.



3. In the Select ***Device for target 1*** dialog

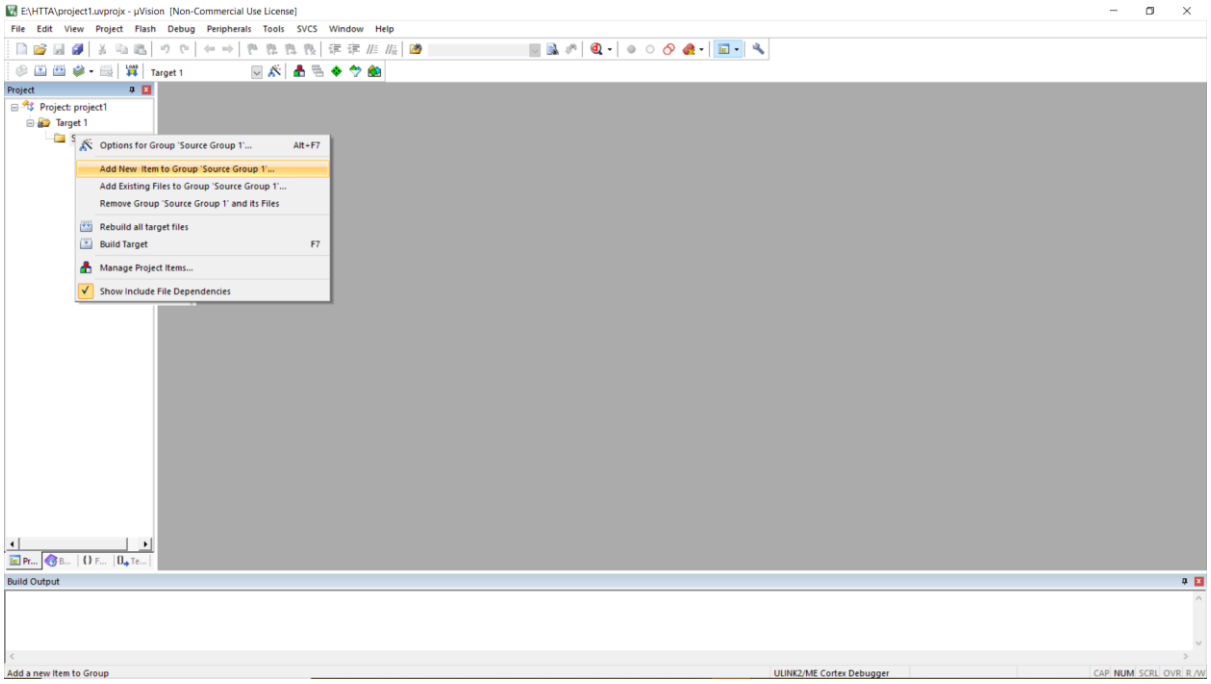
a. Select the device LPC2378 UNDER NXP (or any other Chips you want to use)

b. Select OK.

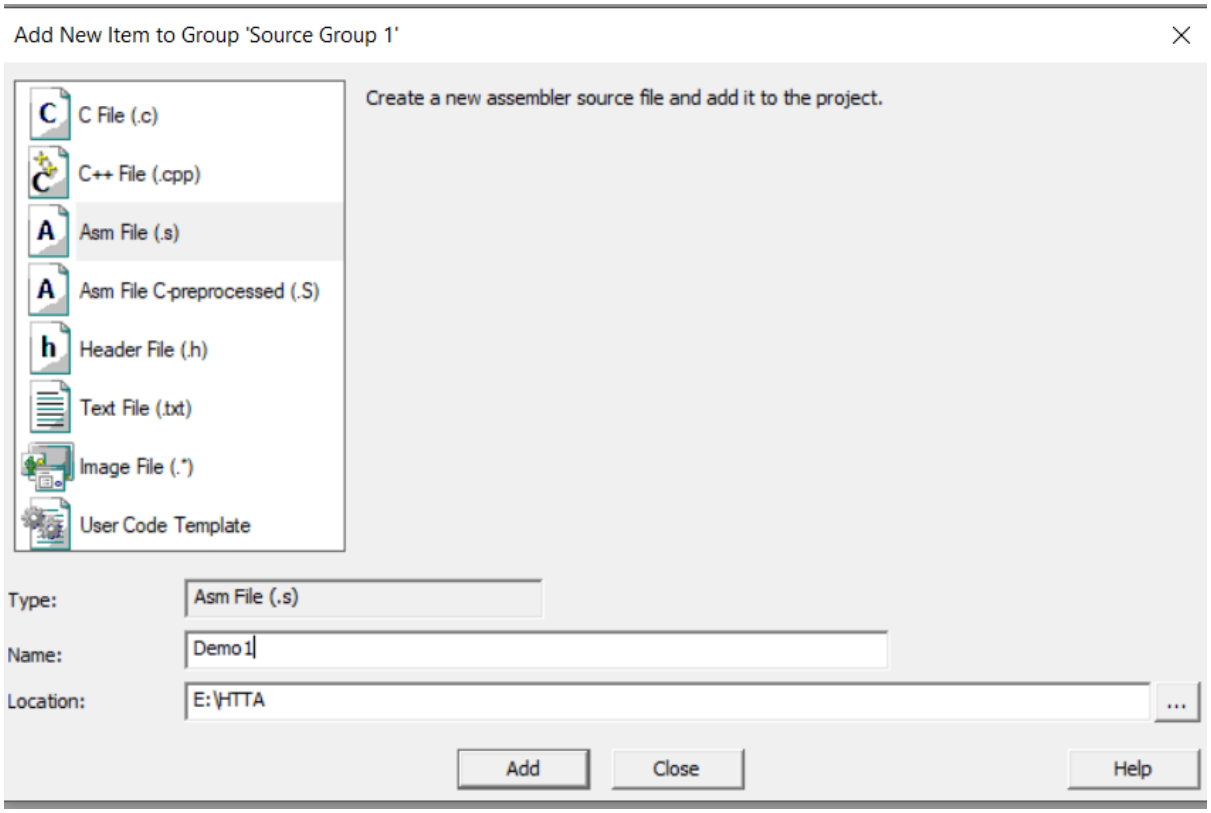


Click NO and proceed. It is needed and mandatory when we are writing in C

Right-click Source Group1 under Target on the left side of the Keil window. Select "Add new Item to group".



Select asm (. s) file in the window prompted, give a filename and add it to 'source group 1'.

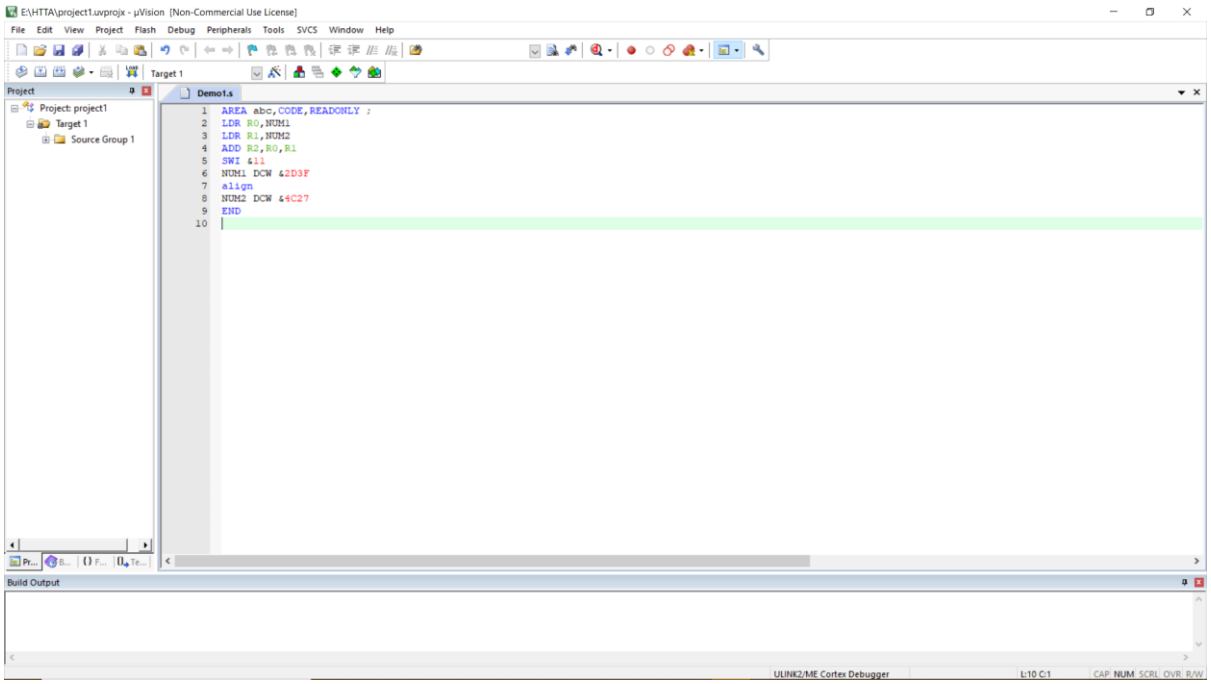


4. Writing the first assembly program

Write your program or copy the code from the existing program, and save the file

The following program adds two hexadecimal numbers.

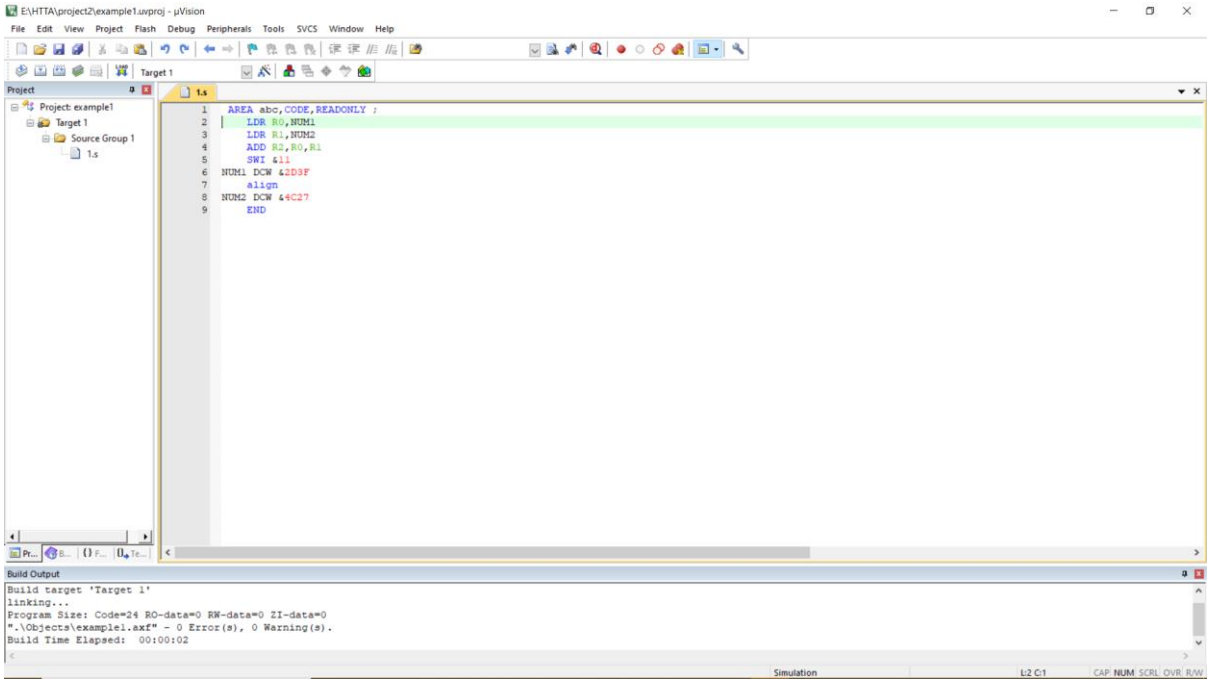
```
AREA abc,CODE,READONLY ;
LDR R0,NUM1
LDR R1,NUM2
ADD R2,R0,R1
SWI &11
NUM1 DCW &2D3F
align
NUM2 DCW &4C27
END
```



Building

Use Ctrl+ F7 to translate the code.

Press *F7* to B*uild*, or choose *Build Target* from the *Project* menu. The results of building the program are shown in the *Output* window.

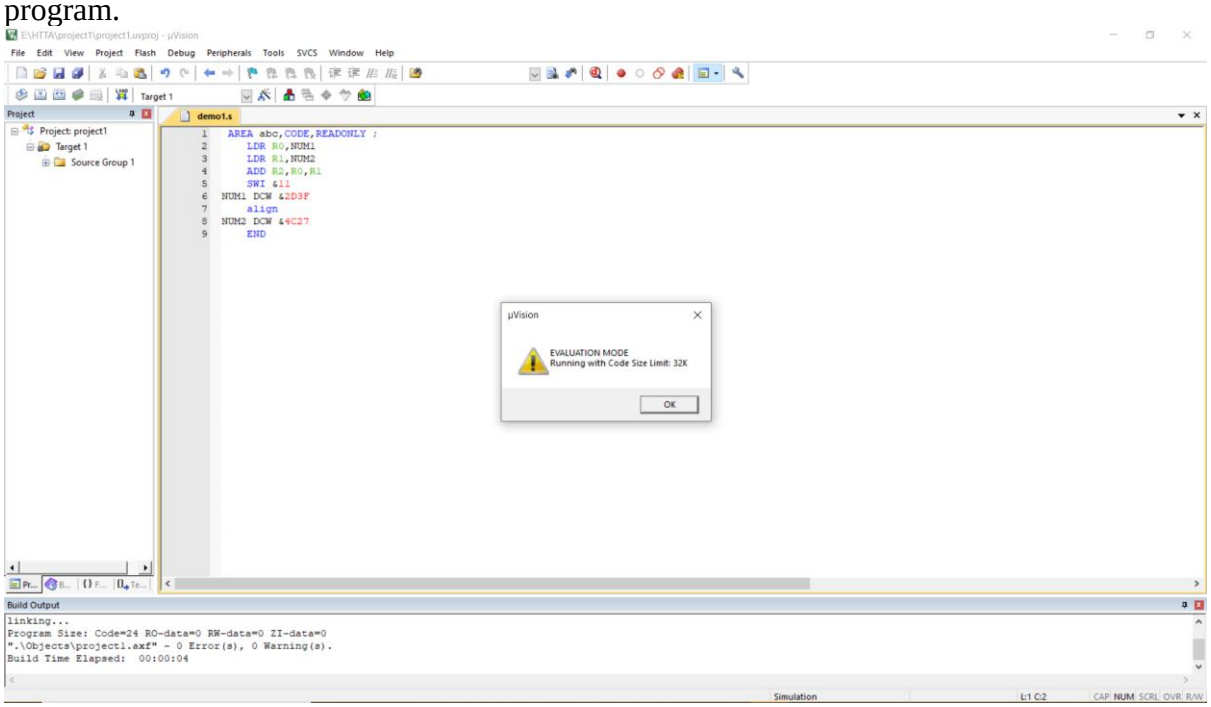


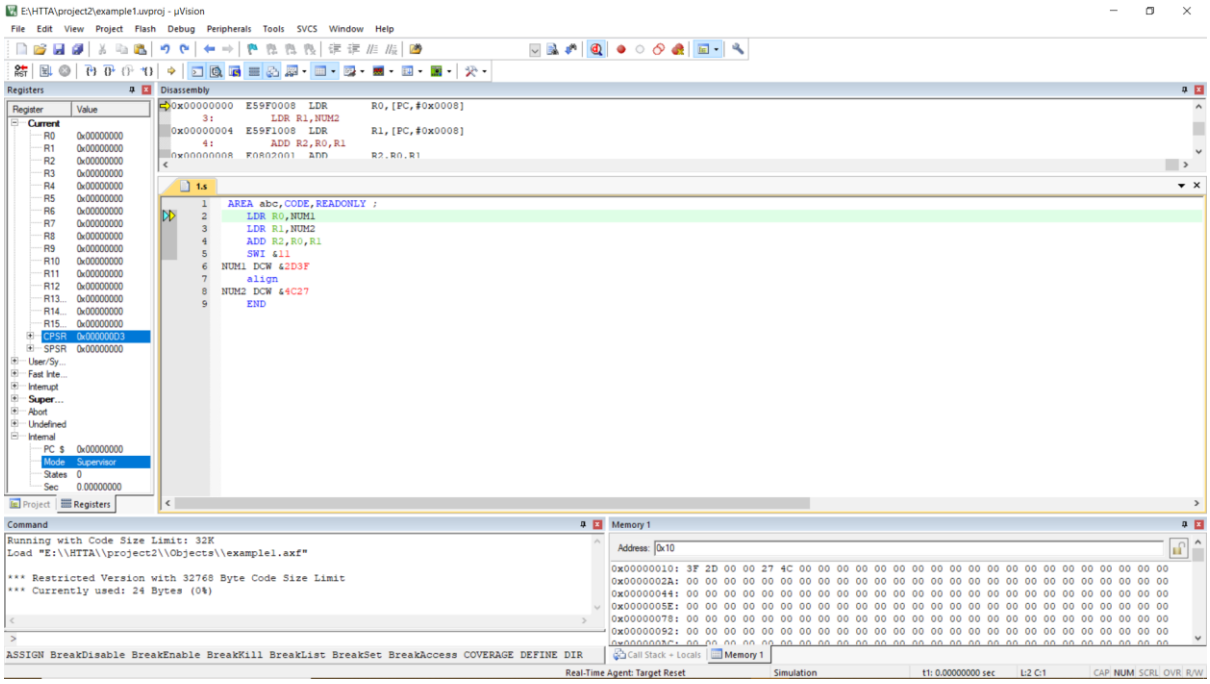
Debugging

1. To start debugging, press ***Ctrl+F5*** or choose ***Start/Stop Debug Session*** from the ***Debug*** menu.

2. The following Dialog appears with a warning
EVALUATION MODE: Running with Code Size Limit:32k. Press ***OK***.

3. Now, a yellow cursor is on the main program's first line, and the IDE is ready to debug the program.





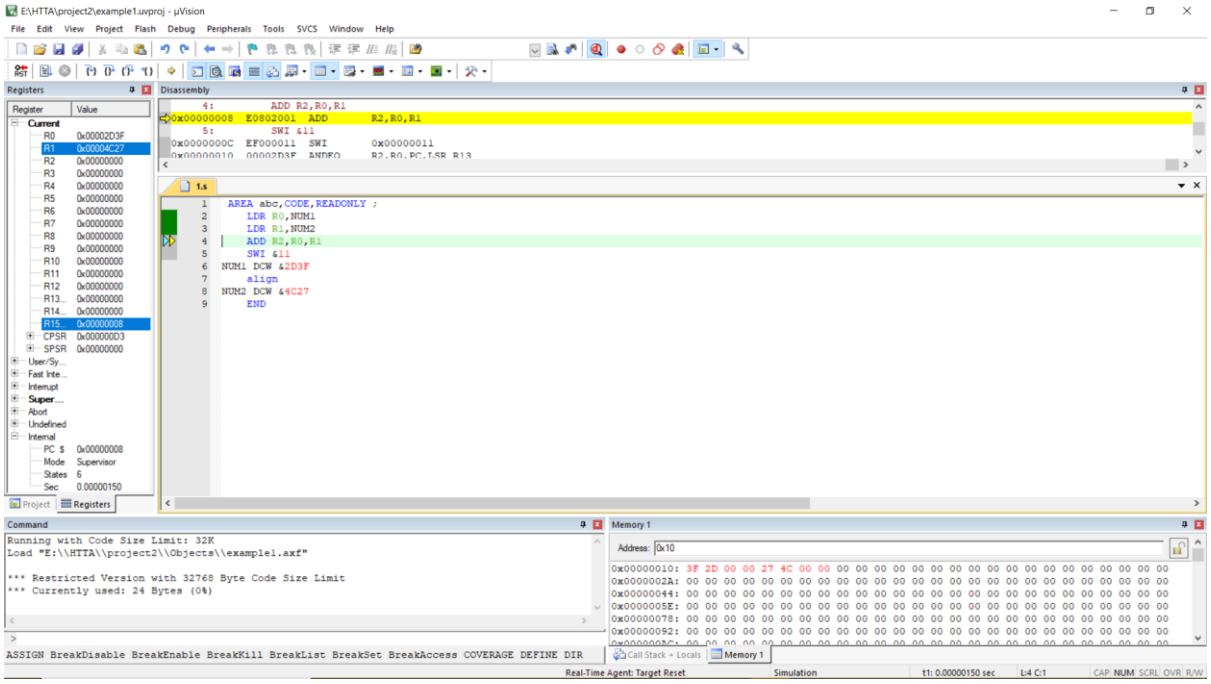
There are R00 to R15 general-purpose registers, all of them are initialized to ZERO.

After execution of the program, the corresponding registers are configured to values as indicated in the program.

5. To execute the instructions line by line, press ***F11***or click on the ***Step over*** icon

Step Into vs. Step Over
Both ***F10 (Step over)*** and ***F11 (Step into)*** execute one instruction and go to the next instruction. But they work differently when the cursor is on a function call. If the cursor is on the function call, ***Step into*** goes into the first instruction of the function, but ***Step Over*** executes the whole function and goes to the next instruction.

Step Out
If the execution is in a function, you can execute the function to the end by pressing the ***Step Out***.



Run to Cursor

You can put the cursor on an instruction and then press the Run to Cursor button. In the case, the program runs until it reaches the instruction which the cursor is on it.

Processor Tab

The Processor tab shows the current values of the CPU registers including R0-R31, SP (Stack Pointer) and PC (Program Counter). You can also change the values of registers by double clicking on their values and typing a new value.

Using Breakpoints

If you want to debug a portion of a program, add a breakpoint to the beginning of this part of the code and press the run button. The IDE runs the program, and when it reaches the breakpoint, it stops running, and the yellow cursor is shown on the breakpoint line.

Below, you see the steps in detail.

1. Right-click on the "LDI R17,0x0A" instruction. A pop-up menu appears. Choose ***Breakpoint*** and then ***Insert Breakpoint***. A red bullet appears on the left side of the " LDI R17,0x0A " instruction.
2. Press ***F5*** or the ***Continue*** button. The IDE runs the program until it reaches the Breakpoint. Now, you can continue debugging from the breakpoint using the ***Step into*** and ***Step over*** buttons.
3. Using ***Stop Debugging***, you can stop debugging whenever you want.