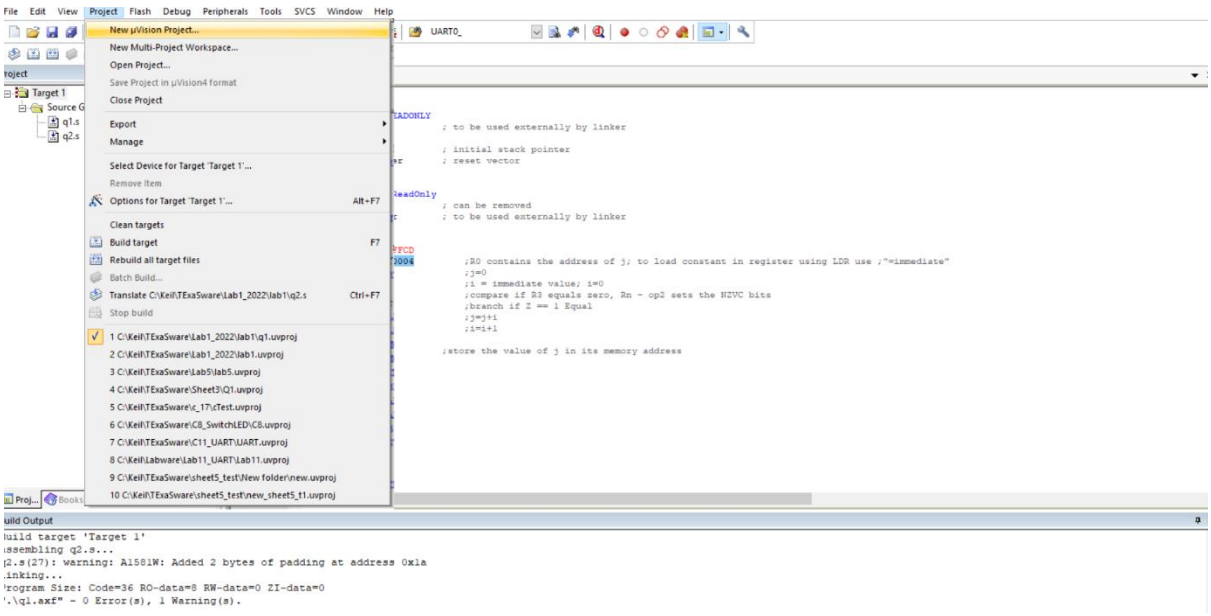
Lab 2

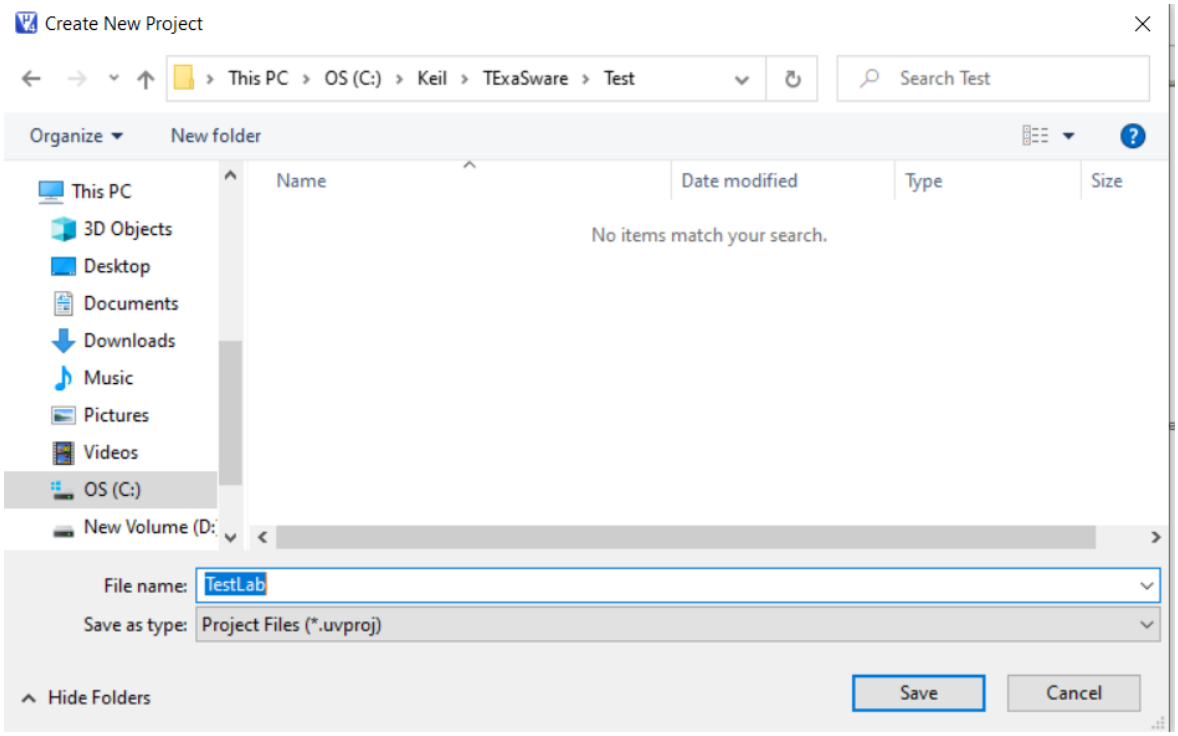
Introduction

To use the simulated behavior of Keil:

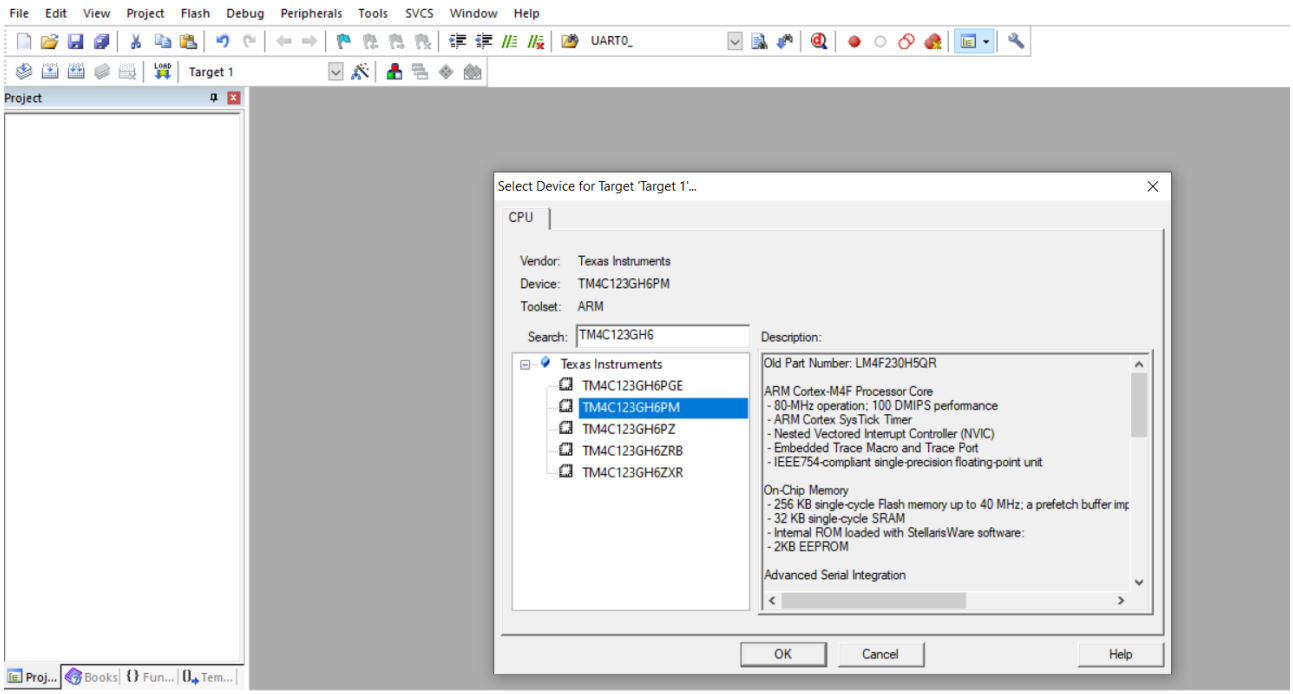
1. Create new project.
    a. By selecting “New uVision project” from project tab in the tool bar of the IDE.



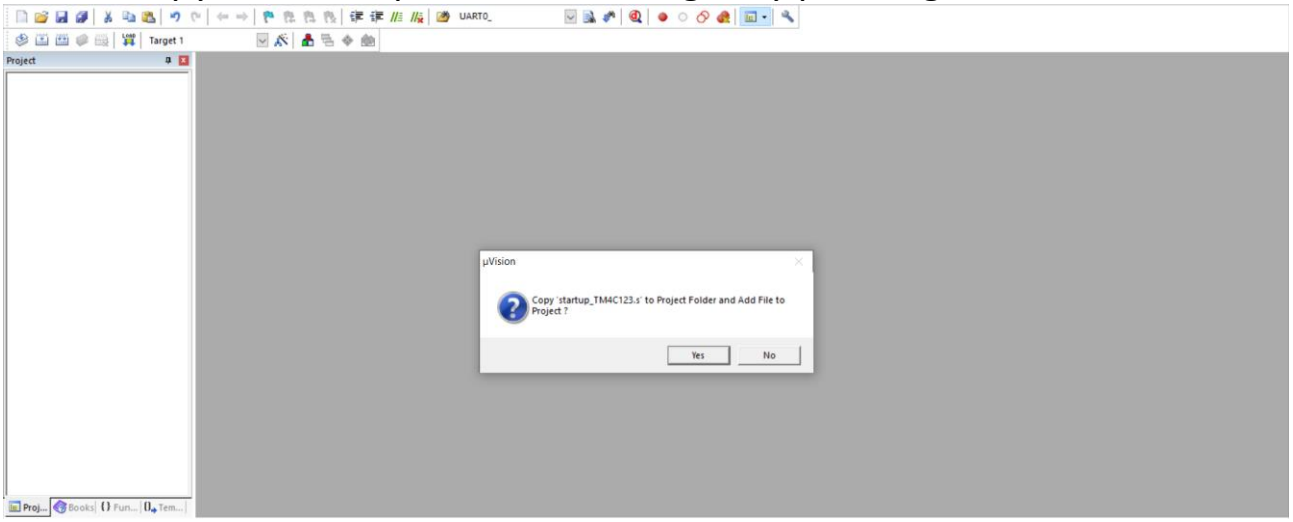
b. Choose the location of the project and rename it.



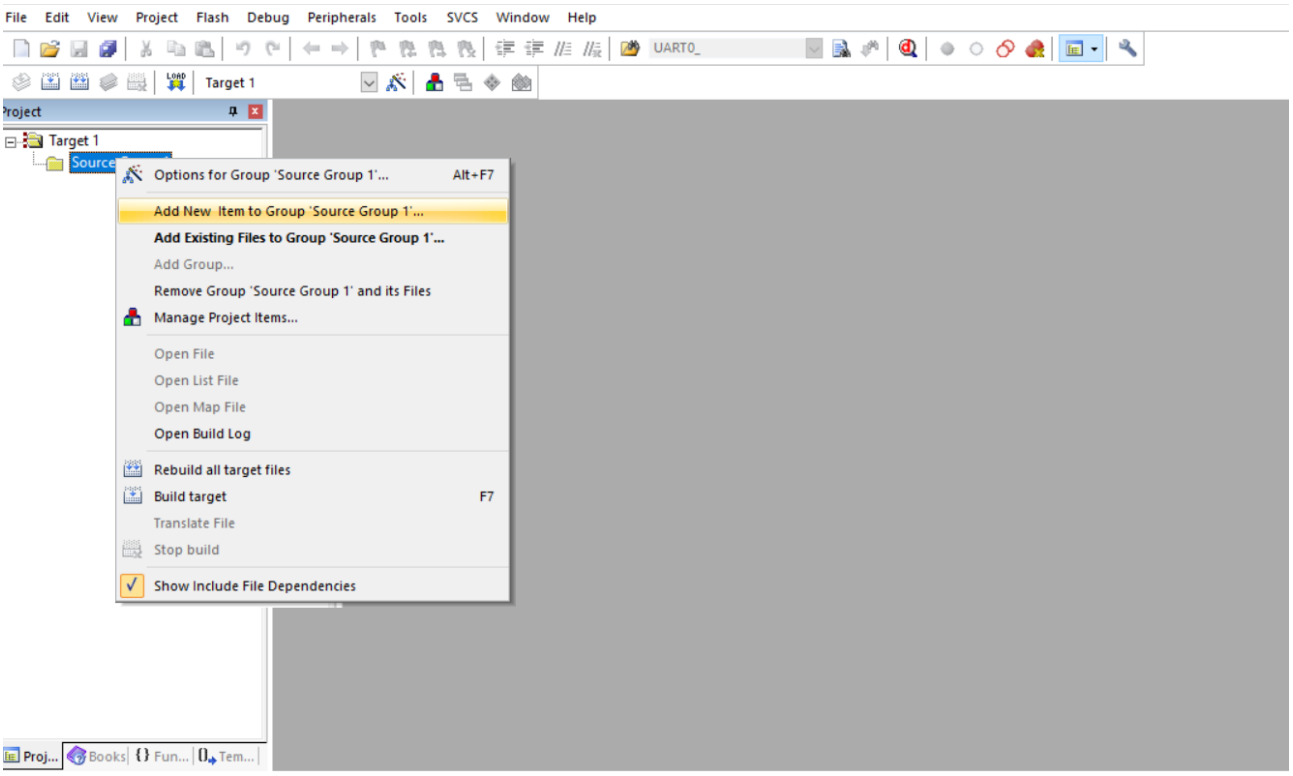
2. Choose the target TM4C123GH6PM device.



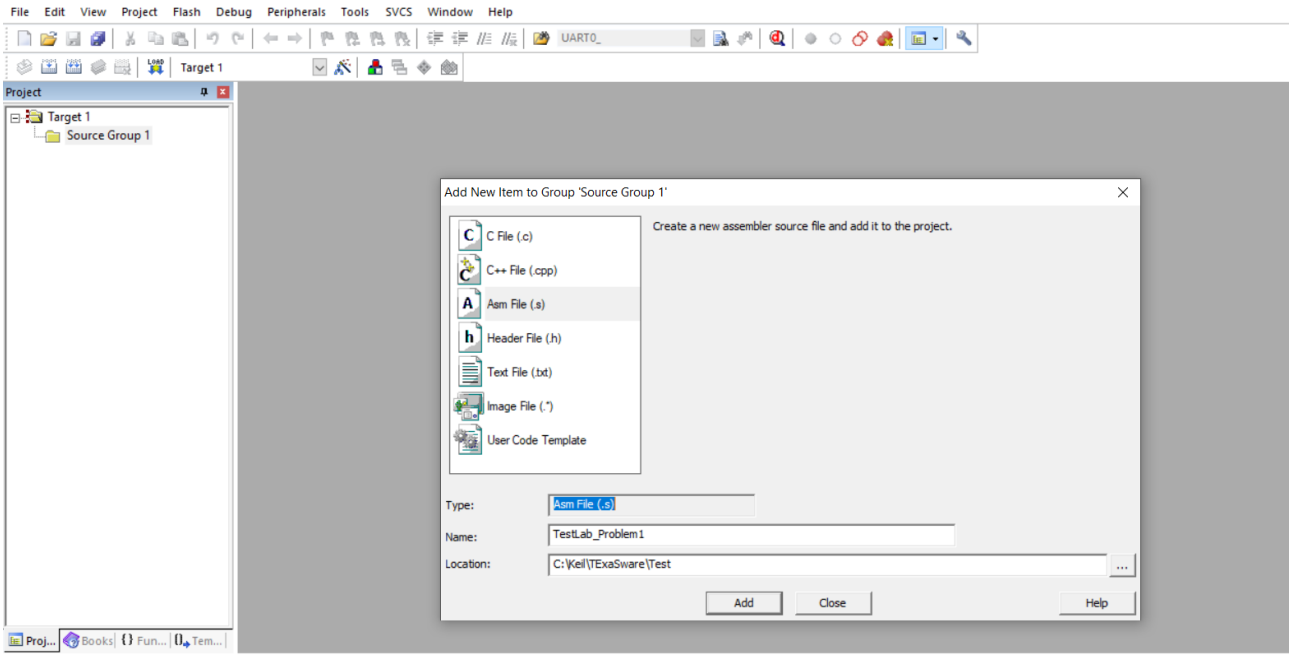
3. Do not copy the startup code of the target by pressing “No”.



4. Create a new assembly file by right click on “Source Group 1” and choose “Add New Item to Group…. ”.



5. Choose assembly file and rename your assembly file.



6. Paste the below startup code in the created assembly file.

```
                AREA RESET, DATA, READONLY
                EXPORT  __Vectors

__Vectors

                DCD    0x20008000
                DCD    Reset_Handler
                ALIGN

                AREA myCode, CODE, ReadOnly
                ENTRY
                EXPORT Reset_Handler

        Reset_Handler
```

Assembler Directives

AREA directive tells the assembler to define a new section of memory (SRAM or ROM).

CODE: contains machine instructions/const. data (R)

DATA: no instructions allowed here

READONLY: placed in ROM, for CODE by default

READWRITE: placed in SRAM for variables

ENTRY indicates to the assembler the beginning of the executable code

END indicates to the assembler the end of the source (asm) file

ALIGN {2} ensures the next instruction is 32-bit {16-bit} aligned

EQU defines a constant value or a fixed address. It does not set aside storage for a data item, but associates a constant number with a data or an address label

DCB, DCW, DCD allocate **aligned** byte, half-word (16-bit), word (32-bit) memory locations

SPACE is used for uninitialized data

Lab Exercise

Q1. Assume A is a label for 4x4 matrix and Z, and X are labels for arrays with 4 items (each item is 4 bytes) in the program. Write arm assembly for the following snippet code.

```
for (int row = 0; row < 4; row++)
    for (int column = 0; column < 4; column++)
        Z[row] += A [row, column] * X[column]
```

Q2. Design and write an ARM Assembly Language subroutine that calculates the value of xx^{yy}. The initialization part of the assembly code is given as below.

```
MOV R0,#7  ; R0 is xx

MOV R1,#3  ; R1 is yy
```

In your code, pass the parameters through registers. Explain the usage of the registers in this problem.

Lab Submission

Q3. Write ARM assembly code to sum the array items of size 10 and store it in the memory. The array contains the following values:1, 2, 3, 4, 5, 6, 7,8, 9, 10.

For the lab submission, you should submit a pdf document contains the following.

Q4. Design and write an ARM Assembly Language subroutine that retrieves the minimum and maximum numbers between 6 given numbers. The initialization part of the assembly code is given as below.

```
MOV R0,#6 ; R0 elem count
MOV R1,#-14 ; from R1 to R6 are the elements that we should calculate the minimum and maximum among them
MOV R2,#5
MOV R3,#32
MOV R4,#-7
MOV R5,#0
MOV R6,#-5
```

In your code, use a mixture between passing parameters through registers and through the stack. Explain the usage of the stack in this problem.

1. Cover page that contains
    a. your name,
    b. your ID,
    c. your department.

2. **For each question**, place 2 snapshots to show general register values and the special register values from the debugging window.
3. Place 1 snapshot for the final contents of the memory after the execution of the program.
4. Place your code in the document.
5. Your document will be submitted on LMS.