

CMPE 443 PRINCIPLES OF EMBEDDED SYSTEMS DESIGN

LAB #001 “Setup LPC4088”

Motivation

In this experiment, you will get introduced with the Keil development environment¹ and the development board which utilizes LPC4088 microcontroller from NXP Semiconductors². Within the scope of this experiment, you will learn:

- setting up and working in a typical embedded software development environment
- downloading an embedded code to the development board
- reading and modifying data in memory
- using a simulator to extract the execution time
- investigating the linker map file to extract the size of memory for specific memory components, code and data
- observing the effect of different compiler options on the size and execution time of a code segment
- observing the effect of different coding styles on the size and execution time of a code segment setting up a design exploration space

1) Setup Environment

1pt

From <https://www.keil.com/download/product/> download and install Keil uVision5 application (MDK-ARM).

2) Setup Device

1pt

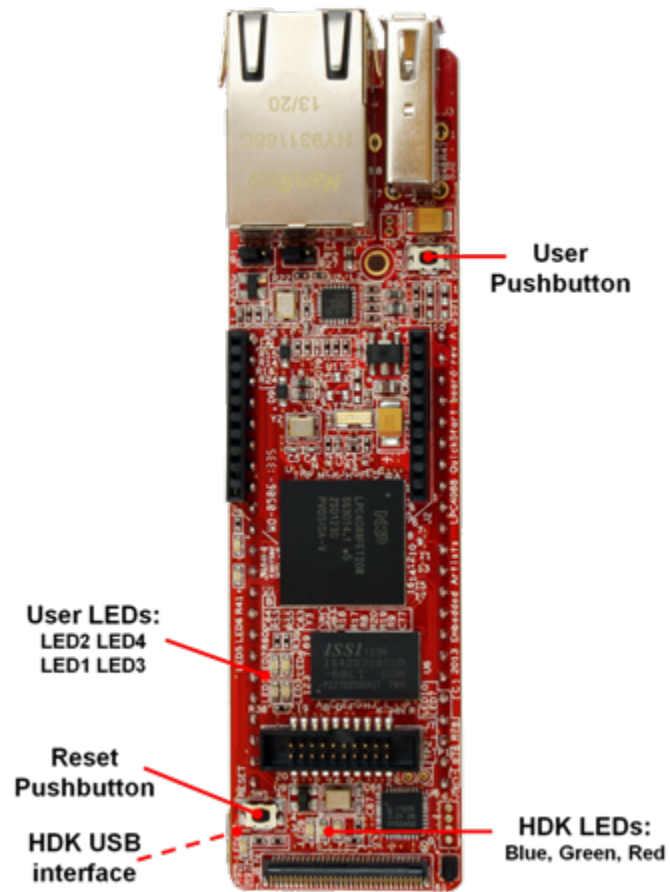
From <https://developer.mbed.org/handbook/Windows-serial-configuration> download mbed Windows Serial.

- Take the “micro-B to A” USB cable.
- Connect the “micro-B” part to the HDK USB port of the LPC4088 QuickStart Board. The HDK USB interface is found on the bottom side of the board under the Reset pushbutton (same end of the board where the FPC display connector is located) as shown in the below figure.

¹ "Keil Embedded Development Tools for ARM, Cortex-M ..." <<http://www.keil.com/>>


² "Quickstart Board for LPC4088 :: NXP Semiconductors." <<http://www.nxp.com/board/OM13063.html>>

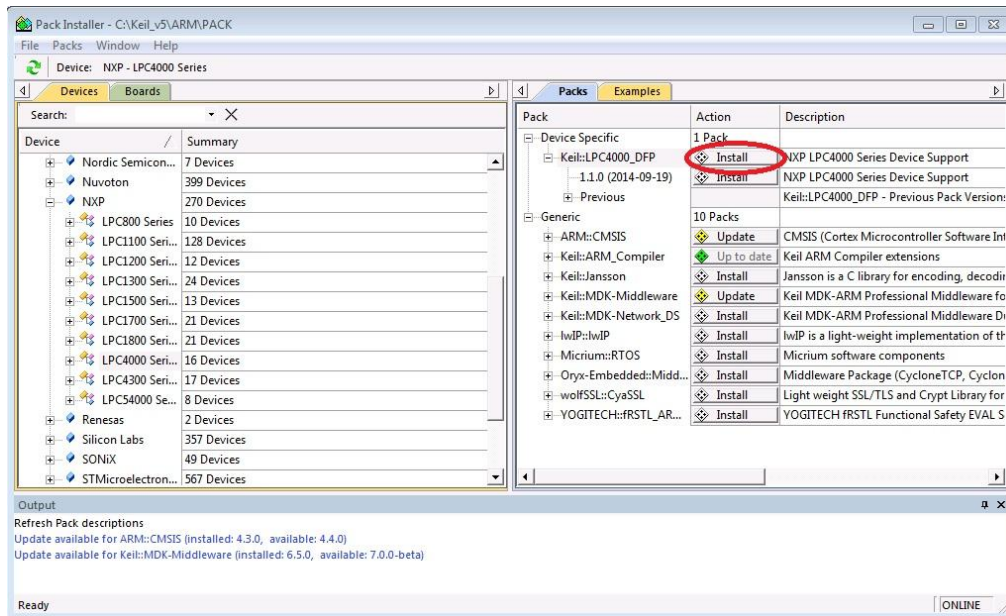
- Install mbed Windows Serial software.



3) Setup Project

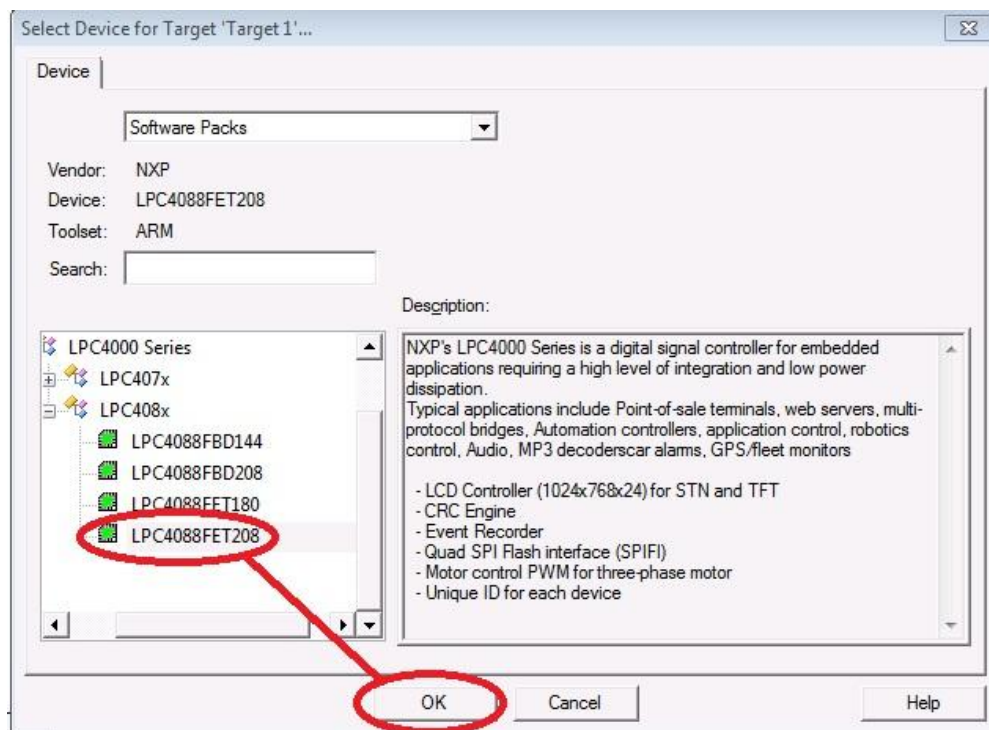
lpt

Enter Keil uVision5 and open Pack Installer (). Select NXP Devices and select LPC4000_DFP and install that pack. Also you can import Keil.LPC4000_DFP.2.0.0 pack file for installing.



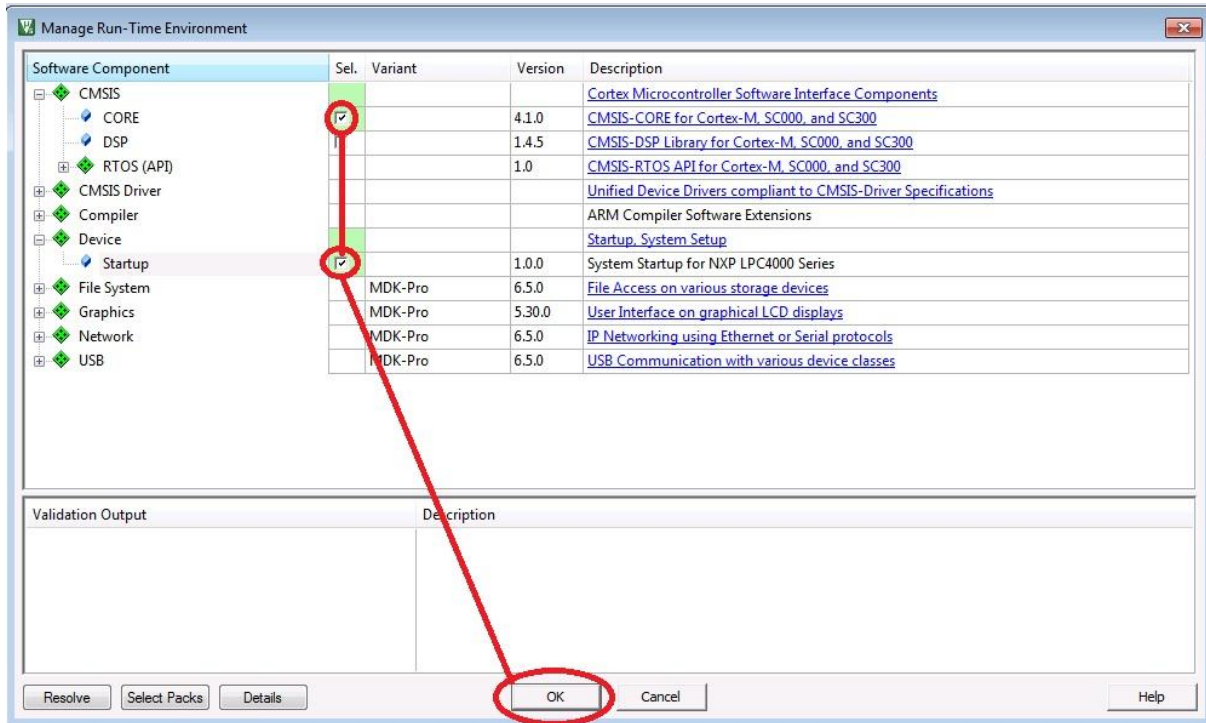
After installation, reload Software Packs using the menu: **Project - Manage - Reload Software Packs**.

For creating project select: **Project - New uVision Project**.



Select **LPC4088FET208** for Device.

After clicking **OK** button, select **CMSIS - CORE** and **Device - Startup**.



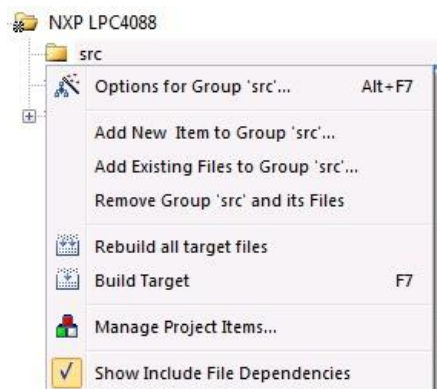
You can change target name “Target 1” to any name such as “NXP LPC4088”.

You can change source group name “Source Group 1” to any name such as “src”.

4) Add Source Files to Project

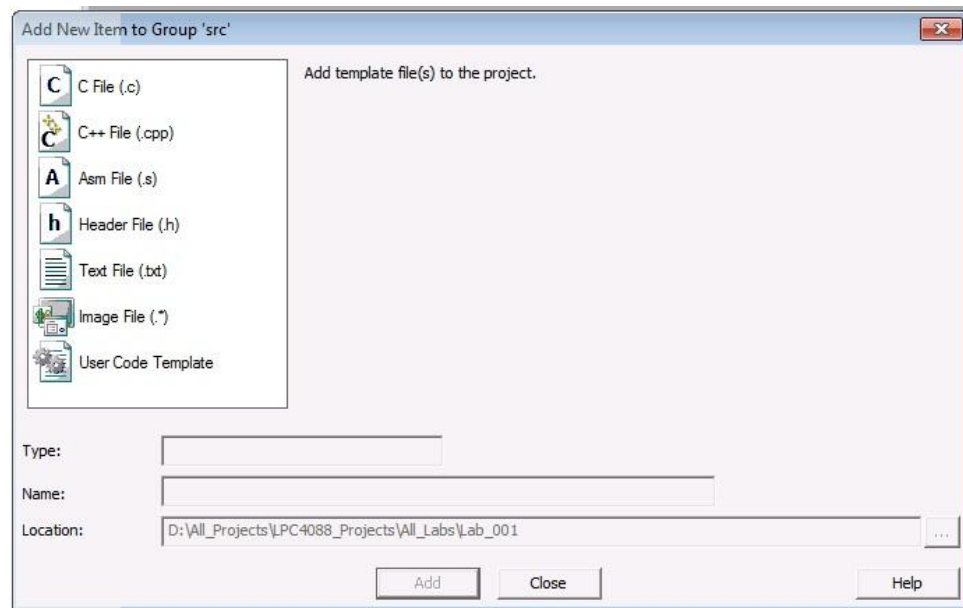
lpt

You can add files in several ways to a project. The most common way is to click on a file group in the window **Project** and use the context menu **Add New Item to Group** or **Add Existing Files to Group**.



Add New Item to Group

You can add new items: Right Click the **Source Group** - **Add New Item** – **Select File**.



For adding **main.c**: Right Click the **Source Group** - **Add New Item** – **Select C File** and name it as main.

In **main.c** file you can copy the following code:

```
#include "LPC407x_8x_177x_8x.h"

int millisecond = 1000;

void init() {
    LPC_GPIO1->CLR = 1 << 18;
    LPC_GPIO1->DIR |= 1 << 18;
    LPC_IOCON->P1_18 &= ~(0x3 << 3);
    LPC_IOCON->P1_18 |= (0 & 0x3) << 3;
}

void update() {
    int i;

    LPC_GPIO1->SET = 1 << 18;
```

```

        for(i=0;i<millisecond*12000;i++);
        LPC_GPIO1->CLR = 1 << 18;
        for(i=0;i<millisecond*12000;i++);
    }
int main() {
    init();
    while(1) {
        update();
    }
}

```

Add Existing Files to Group

You can add existing items: Right Click the **Source Group - Add Existing Item – Select Files from File Dialog**. Yet, this item is not related with this week's experiment. So, you can skip.

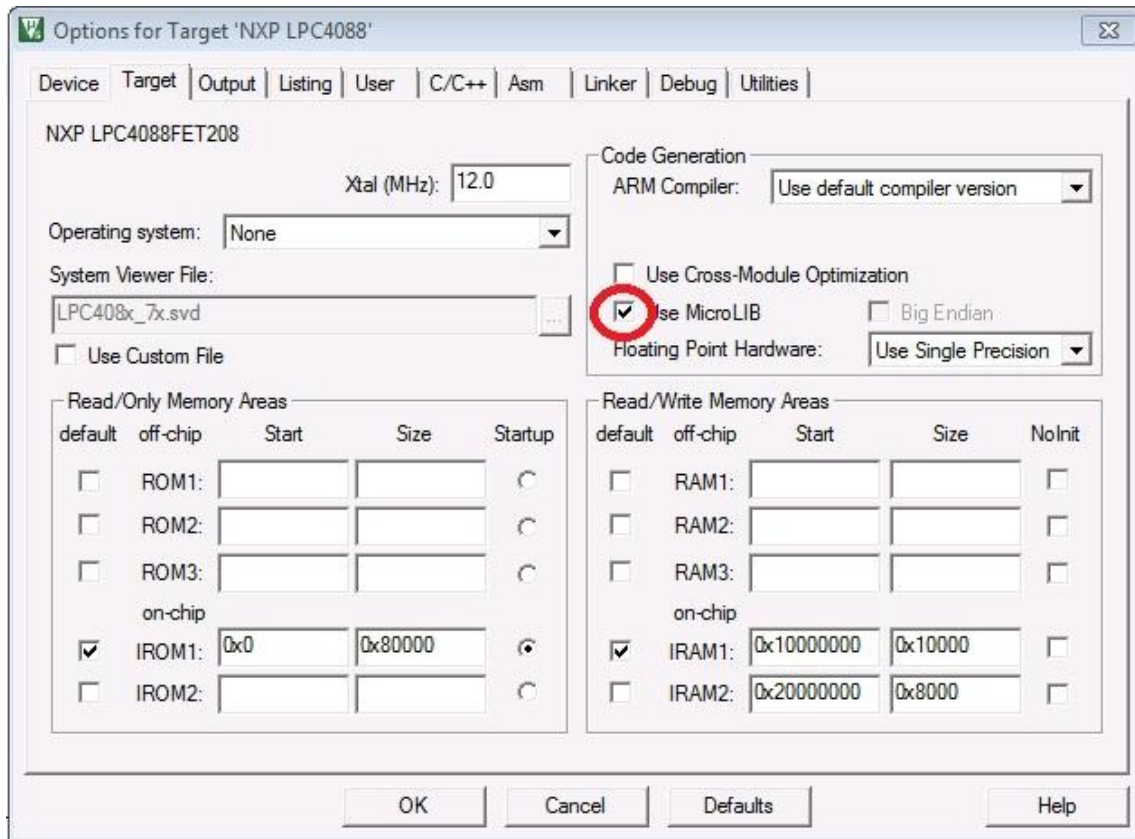
5) Open Existing Project

Select **Project - Open Project**. Yet, this item is not related with this week's experiment. So, you can skip.

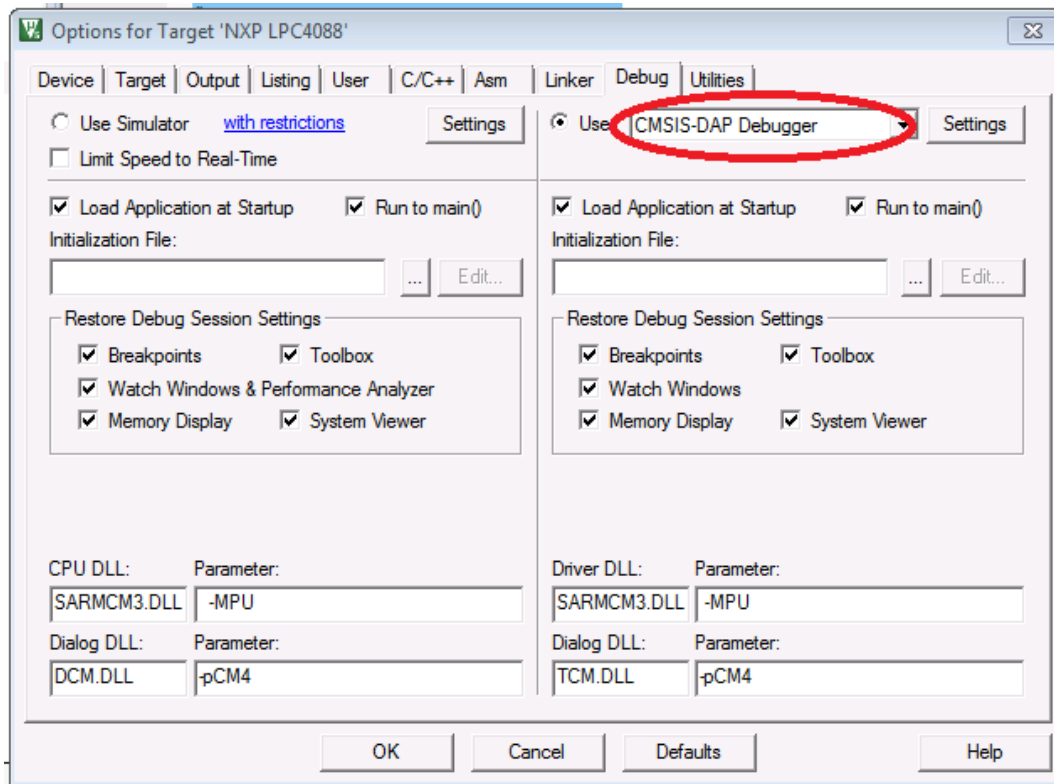
6) Debug Settings

3pts

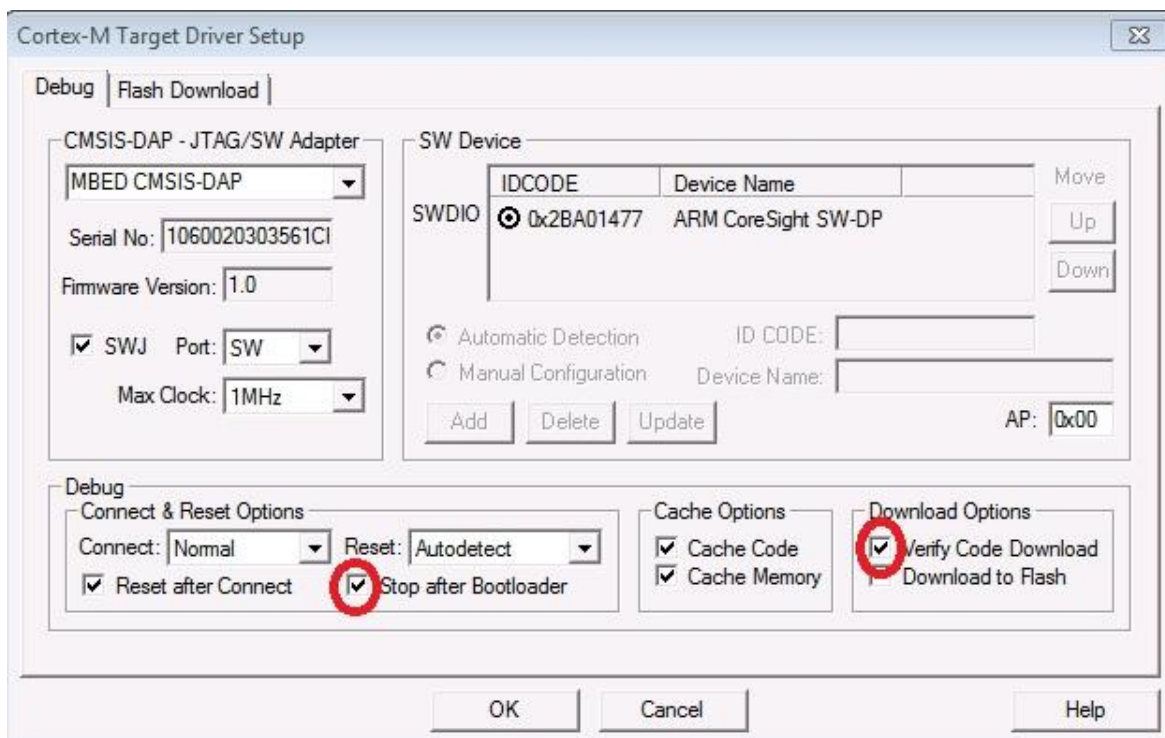
For debugging, select **Target Name - Options for target** 
In **Target** tab, click **use MicroLIB**



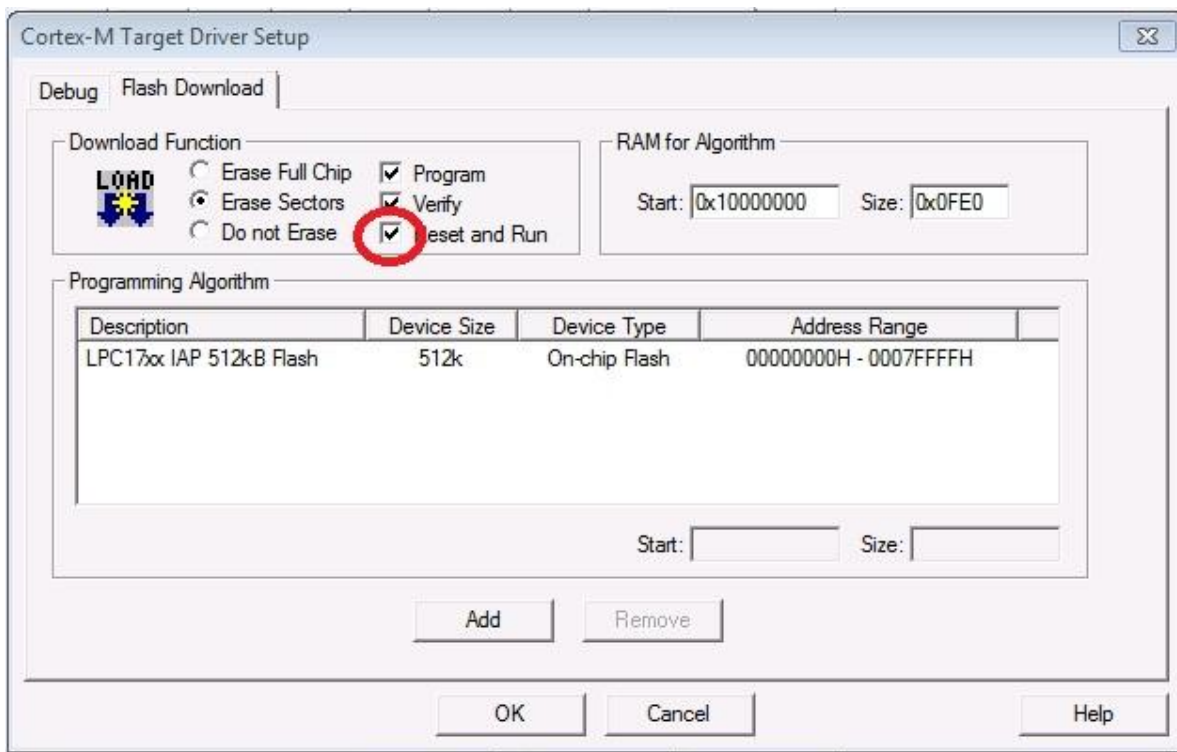
In **Debug** tab, select **CMSIS-DAP Debugger**.



Click **Settings** for CMSIS-DAP Debugger. In **Debug** tab, click **Stop after Bootloader** and **Verify Code Download**. (Make sure all the checkboxes and drop down menus are the same as the figure)













In **Flash Download** tab, click **Reset and Run**.



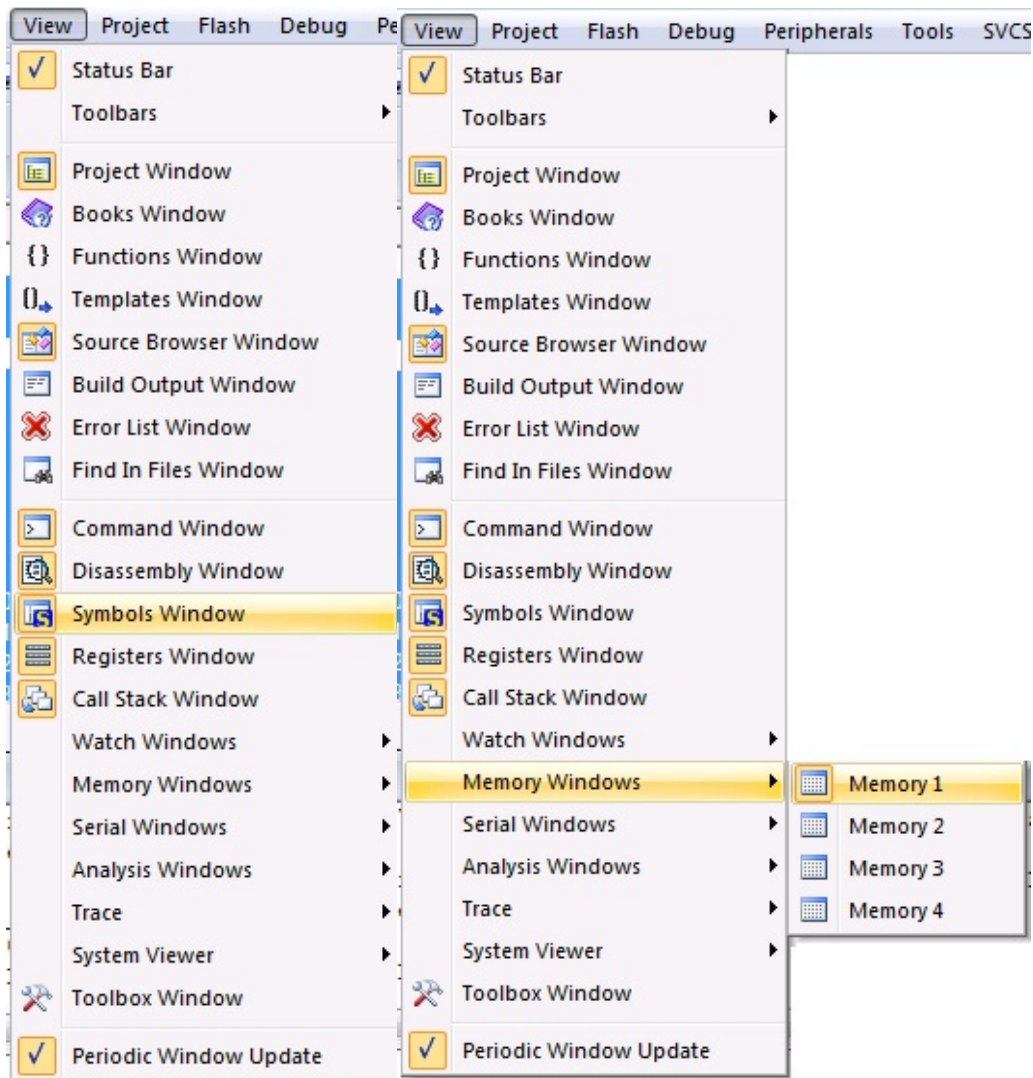
7) Debug Program

Open your **main.c** file from your Project Window. Now you can add **Breakpoints** by clicking next to the line.

Add breakpoint next to: `LPC_GPIO1->SET = 1 << 18;` and `LPC_GPIO1->CLR = 1 << 18;`

You can build your code by    . Before debugging, build and by using , run the code and observe what happens. After that, you can start debugging by . You can navigate by     in debug mode. (Every time build your code before loading)

In Debug Mode, there will be a view which is **Register View** and you can open **Symbols Window** from View - Symbols Window and also you can open **Memory1** from View - Memory Windows - Memory1.




In Register View, you can see the register values.

In Symbols View, you can see the address of the variables and functions.

In Memory1 view, you can change number representation to **Right Click - Signed - Int** and also you can modify data by **Right Click - Modify Memory** at ...


- What is the address of the variable "millisecond"? _____ *1pt*
- Which register has the value of "i" variable? _____ *1pt*

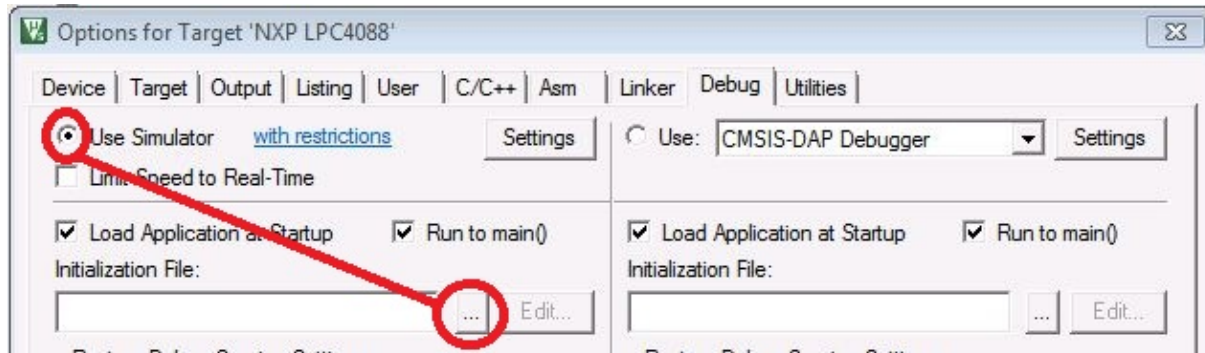
Modify Memory Content of "millisecond", disable breakpoints and press **Run** button  :

- If you write 0x1000 what changes on the LED? _____ *1pt*
- If you write 0x100 what changes on the LED? _____ *1pt*

8) Using Simulator

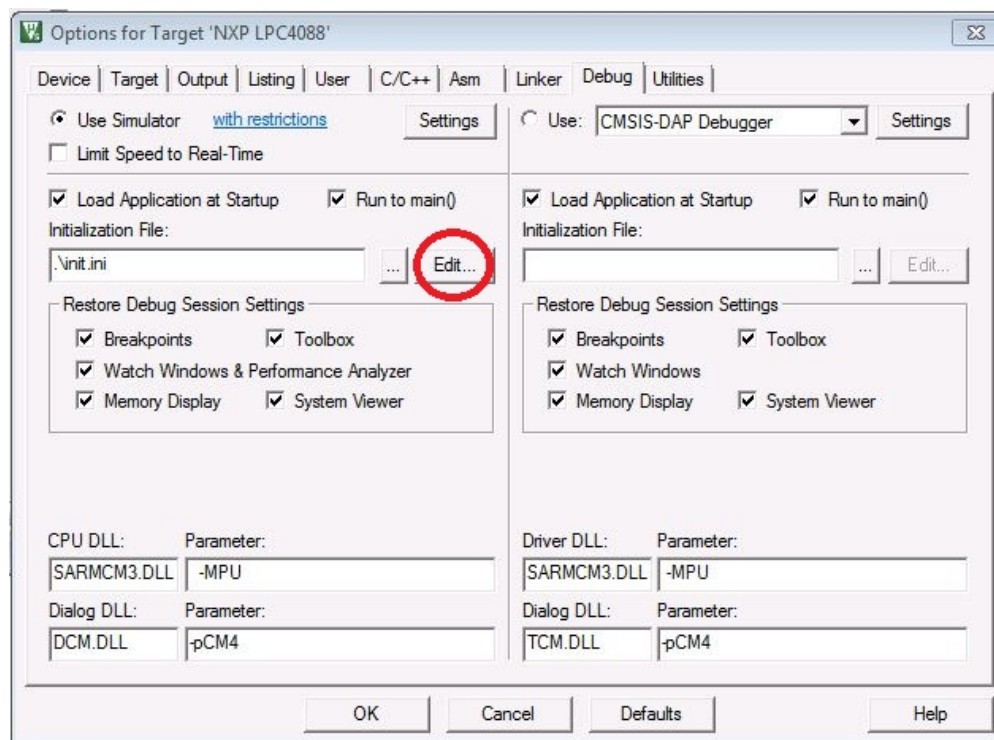
2pts

For using simulator, select Target Name - Options for target . In **Debug** tab, select **Use Simulator** and click browse **Initialization File** and then write a name and click to **Open** button.



If file is not exist, Keil will create for you if you want. After that, click **Edit**. Then write in the file:

MAP 0x40000000, 0x400FFFFF Read Write



After that, open **system_lpc407x_8x_177x_8x.c** file and use comment ("`//`") at 528th, 538th and 546th line (1706th, 1716th and 1724th line in new version).

```
//while ((LPC_SC->SCS & (1<<6)) == 0);/* Wait for Oscillator to be ready */
//while (!(LPC_SC->PLL0STAT & (1<<10)));/* Wait for PLOCK0 */
//while (!(LPC_SC->PLL1STAT & (1<<10)));/* Wait for PLOCK1 */
```

9) Cycle Count

Cycle count for a block of code can be found by using two different breakpoints. First breakpoint is at the beginning of the block and second breakpoint is at the end of the block.

After placing breakpoints, debug the code and when first breakpoint is reached, look at **Register Window - Interval - States**. Record the state number and continue the code and when second breakpoint is reached, look at **Register Window - Interval - States** again. The difference of the state numbers will be the cycle count for that block of code defined by two breakpoints.
Cycle Count = Difference of States.

By using the below code, find the number of states: _____ 2pts

(Place one breakpoint next to the update)

```
#include "LPC407x_8x_177x_8x.h"
```

```
void RGBtoGrayscale_C (uint8_t * __restrict dest, uint8_t * __restrict src, int size) {
```

```
    int i;
```

```
    for (i=0; i<size; i++) {
```

```
        int r = *src++;
```

```
        int g = *src++;
```

```
        int b = *src++;
```

```
        int y = (r*77)+(g*151)+(b*28);
```

```
        *dest++ = (y>>8);
```

```
    }
```

```
}
```

```
uint8_t rgb[3072] = {255,100,150};
```

```
uint8_t gray[1024];
```

```

void init() {}

void update() {
    RGBtoGrayscale_C(gray,rgb,1024);
}

int main() {
    init();

    while(1) {
        update();
    }
}

```

10) Linker Map

When code is built, Keil shows at the **Build Output Window**: Program Size: Code=... RO-data=... RW-data=... ZI-data=... (ZI-data : Zero Initialized) However you can get the more information about the code from Linker Map such as Code Size, RAM Size, ROM Size, Method Size. The location of that file is **Project Folder - Listings - *.map** At the end of that file you can get these:

- **Code Size:** It is the same as shown Code Size (byte)
- **RAM Size:** It is calculated as $RAM\ Size\ (RW\ Size) = RW\text{-}data + ZI\text{-}data$ (byte)
- **ROM Size:** It is calculated as $ROM\ Size = Code + RO\text{-}data + RW\text{-}data$ (byte)

For the previous code, what are these (read from map file):


- Code	_____	0,5pt
- RO Data	_____	0,5pt
- RW Data	_____	0,5pt
- ZI Data	_____	0,5pt
- RAM Size	_____	0,5pt
- ROM Size	_____	0,5pt


11) Optimization

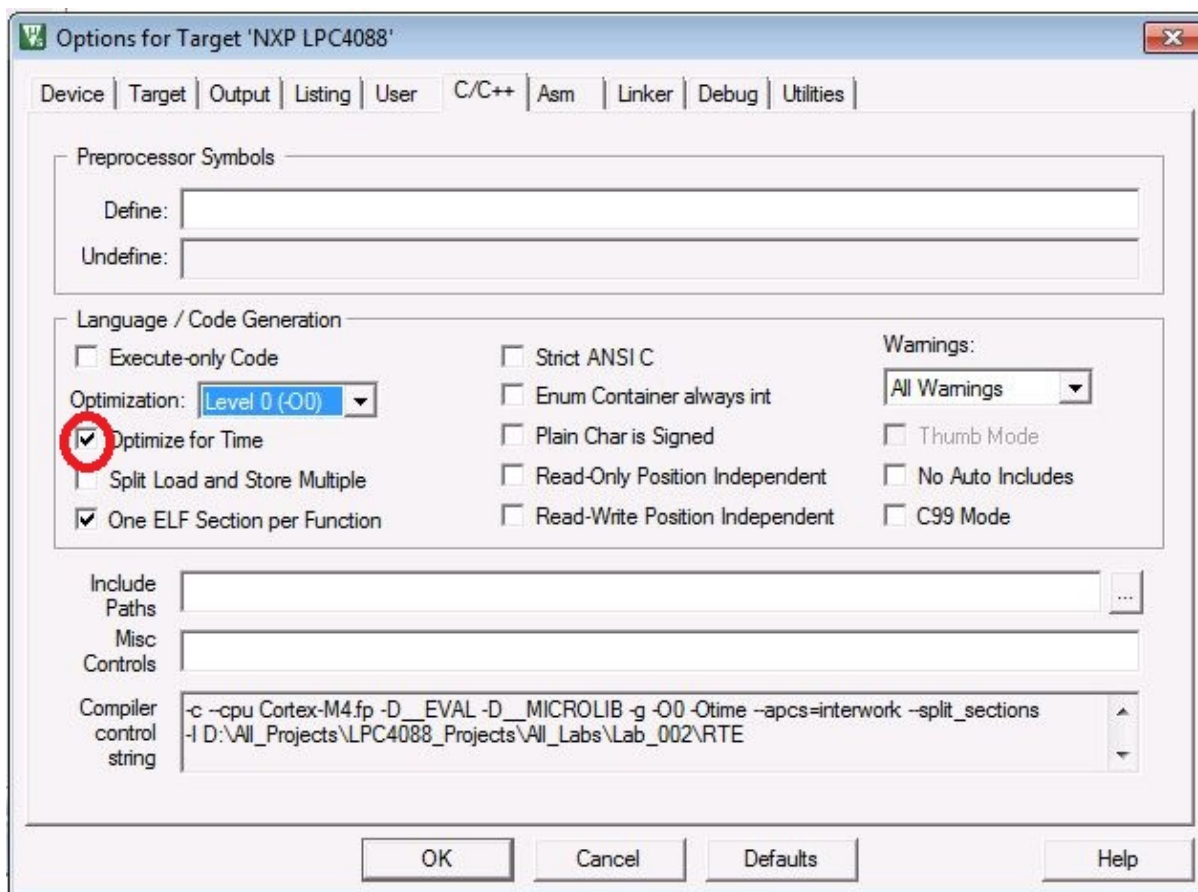
Compiler

2pts

In Keil, there are several compiler optimization ways which are explained following address: <http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.dui0472j/chr1359124221739.html>. In this lab, only space and time optimization with different levels will be used. For Time

Optimization, select Target Name - Options for target . In C/C++ tab, select **Optimize For Time**. Also you can select optimization level from **Optimization**.

For Space Optimization, select Target Name - Options for target . In C/C++ tab, deselect **Optimize For Time**. (Default optimization type is **Optimization For Space**)



Optimize for Space (RGBtoGrayscale_C_Optimized)					
O0					
Code	RO-Data	RW-Data	ZI-Data	ROM	State
O1					
Code	RO-Data	RW-Data	ZI-Data	ROM	State

O2					
Code	RO-Data	RW-Data	ZI-Data	ROM	State
O3					
Code	RO-Data	RW-Data	ZI-Data	ROM	State

Optimize for Space (RGBtoGrayscale_C)					
O0					
Code	RO-Data	RW-Data	ZI-Data	ROM	State
O1					
Code	RO-Data	RW-Data	ZI-Data	ROM	State
O2					
Code	RO-Data	RW-Data	ZI-Data	ROM	State
O3					
Code	RO-Data	RW-Data	ZI-Data	ROM	State

