CMPE 443 PRINCIPLES OF EMBEDDED SYSTEMS DESIGN

LAB #002 “Simulation”

Motivation

In this experiment, we will practice coding in C for Cortex M4. While executing your code on the LPC4088 experimental board, you will also study several features of the code: execution time, occupied memory space, average power consumption and energy.

Within the scope of this experiment, you will learn:

- 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

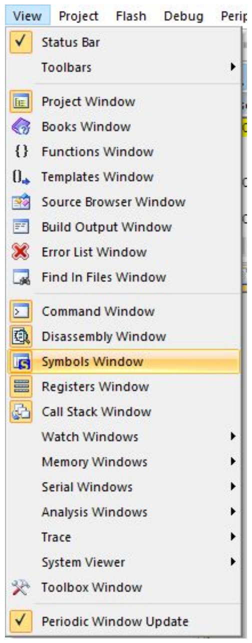
1) Using Simulator

In LAB 1, you configured the project as a simulation. Download **PRELAB2_Template.zip** file from moodle. You will use this project.

You will answer the questions on the **PRELAB2 Quiz** on the moodle.

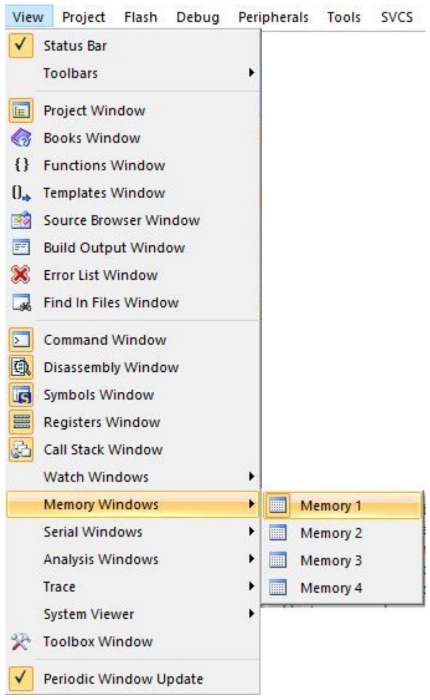
During debug operation, there are some tools that you can get the value of the variable or register. You can open **Symbols Windows** from the View tab. You can get the memory location of the variable and its type.

In the SensorData.h file, there are some variables. These variables are stored on the memory.



- What is location of the sensorData variable: (Answer in Moodle) ____________

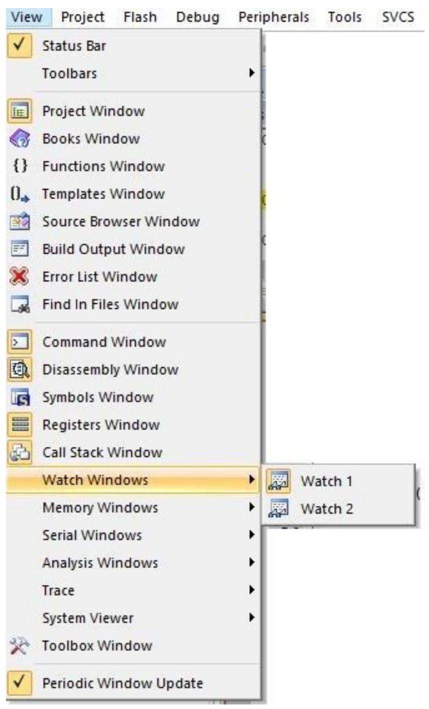
You can also look at the memory from the **Memory Windows**:



- View the Sensor data variable on the Memory View and Right Click on the Memory view and view the memory as **unsigned char (Non Decimal)**. Paste only the Memory ScreenShot here: (Answer in Moodle) ____________

- View the Sensor data variable on the Memory View and Right Click on the Memory view and view the memory as **Signed Integer (Non Decimal)**. Paste only the Memory ScreenShot here: (Answer in Moodle) ____________
- Is data stored in little endian or big endian? (Answer in Moodle) ____________

You can also watch the variable values from the watch window.



- Add the filteredSensorData variable to the Watch View and paste the screenshot of the Watch View here: (Answer in Moodle) ____________

2) Median Filter

The **Median Filter**[1] is a filtering technique which replaces each entry with the median of neighboring entries. Also, the number of neighbors are determined with window size.

For example, for a window of size 5; median(3, 80, 6, 5, 4) = median(3, 4, 6, 5, 80) = 5

The elements are sorted (3, 4, 5, 6, 80) and the middle one is taken (5).

In this prelab, you will filter the sensor data via median filter. WINDOW_SIZE can be changeable in the LAB section. Thus, your code should work for different WINDOW_SIZE.

[1] "Median filter - Wikipedia." https://en.wikipedia.org/wiki/Median_filter. Accessed 22 Nov. 2020.

The sensor data is stored in an array (sensorData array) and you will filter that data.

Consider data will be available at every second, so you should make an empty for-loop (for(i=0;i<millisecond*2000;i++);) for waiting to access the new data on the sensor data array.

The filteredSensorData will be the same as the sensor value for the first WINDOW_SIZE - 1 element. (For example if WINDOW_SIZE is 5, the first 4 elements of the filtered array is the same as the sensor data)

(Test your filter with different values)

3) QEMU Connection

You can generate axf file of the project and you can use this file without any virtualization. You can write on terminal:

```
qemu-system-arm -machine LPC4088 -kernel <Your axf file> -monitor stdio
```

After you run the QEMU, paste the screenshot of the QEMU: (Answer in Moodle)

____

4) Cycle Count

Cycle count for a block of code can be found by state numbers. When you debug the code, **Register Window - Interval - States** shows the state number. By placing breakpoints and taking the difference of the states, you can find the required cycle amount for a block of code. **Cycle Count = Difference of States.**

Place one breakpoint next to the **update** method and find the number of states: (Answer in Moodle)

____

3) Linker Map

When code is built, Keil shows at the **Build Output Window:** Program Size: Code=... RO-data=... RW-data=... ZI-data=... (RO Data is Read Only Data, RW is Read Write Data and ZI Data is Zero Initialized Data) However you can get 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-data + ZI-data* (byte)
- **ROM Size:** It is calculated as *ROM Size = Code + RO-data + RW-data* (byte)

For the previous code, what are these (read from map file (look **Grand Totals**, **Total RW Size** and **Total ROM Size**)): (Answer in Moodle)

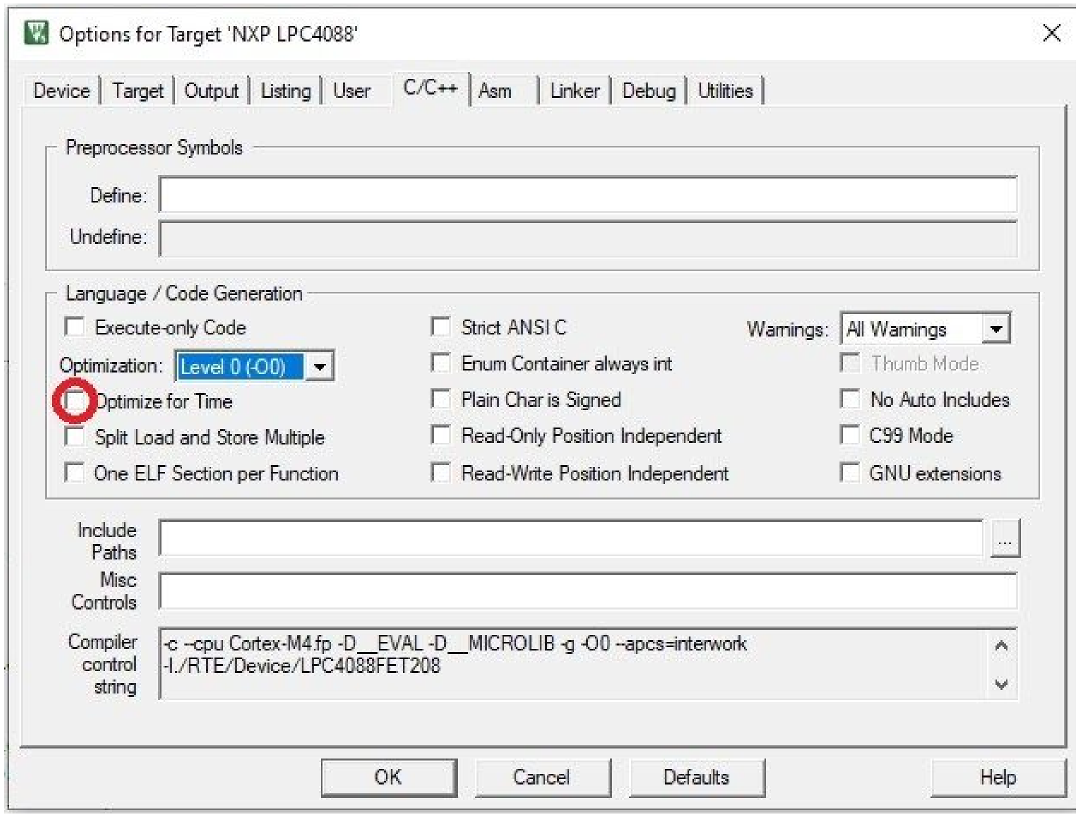
- Code ____________
- RO Data ____________
- RW Data ____________
- ZI Data ____________
- RAM Size ____________
- ROM Size ____________

4) Compiler Optimization

In Keil, there are several compiler optimization ways which are explained in the following address: http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.dui0472j/chr1359124221739.html. In this lab, only space and time optimizations 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**)

Note: If you do not reach the breakpoint next to the update method, just write the reason instead of the state number.


Options for Target 'NXP LPC4088'

Device | Target | Output | Listing | User | C/C++ | Asm | Linker | Debug | Utilities

Preprocessor Symbols

Define:

Undefine:

Language / Code Generation

☐ Execute-only Code
Optimization: Level 0 (-O0)
☐ Optimize for Time
☐ Split Load and Store Multiple
☐ One ELF Section per Function

☐ Strict ANSI C
☐ Enum Container always int
☐ Plain Char is Signed
☐ Read-Only Position Independent
☐ Read-Write Position Independent

Warnings: All Warnings
☐ Thumb Mode
☐ No Auto Includes
☐ C99 Mode
☐ GNU extensions

Include Paths

Misc Controls

Compiler control string:
```
-c --cpu Cortex-M4.fp -D__EVAL -D__MICROLIB -g -O0 --apcs=interwork
-I./RTE/Device/LPC4088FET208
```

OK | Cancel | Defaults | Help

(Answer in Moodle) :

Optimize for Space					
O0					
Code	RO-Data	RW-Data	ZI-Data	ROM	State

Optimize for Time					
O0					
Code	RO-Data	RW-Data	ZI-Data	ROM	State

5) Submission

The file name for the submissions should be (ex: PRE002_2020000000):

PRE<exp num>_<StudentID1>.axf (This will be generated **.axf** file)

PRE<exp num>_<StudentID1>.c (This will be **main.c**)