CMPE 443 PRINCIPLES OF EMBEDDED SYSTEMS DESIGN

LAB #002 "Debug, Memory and Optimization"

1) Debug

In this prelab, create a project as the PRELAB1 configuration (empty project) and change the main.c file with the file that is available on moodle. You can debug the problem via . There are some debug related buttons, you can use . Add breakpoints near the *"wait_counter = wait_counter + 1;"* lines.

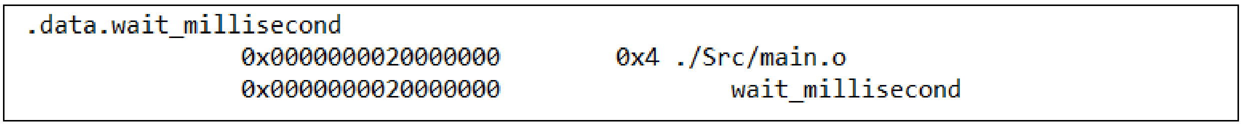
- When you run the code, what happens on the board?

- LD4 COM starts blinking red/off very rapidly as a result of the communication between PC and ST-LINK/V2-1 (in-circuit debugger). LD2 blinks blue/off in each iteration between the debug breakpoints. (Apparently the first expression makes it turn blue and the other one makes it turn off.)

2) Memory Addresses and Registers

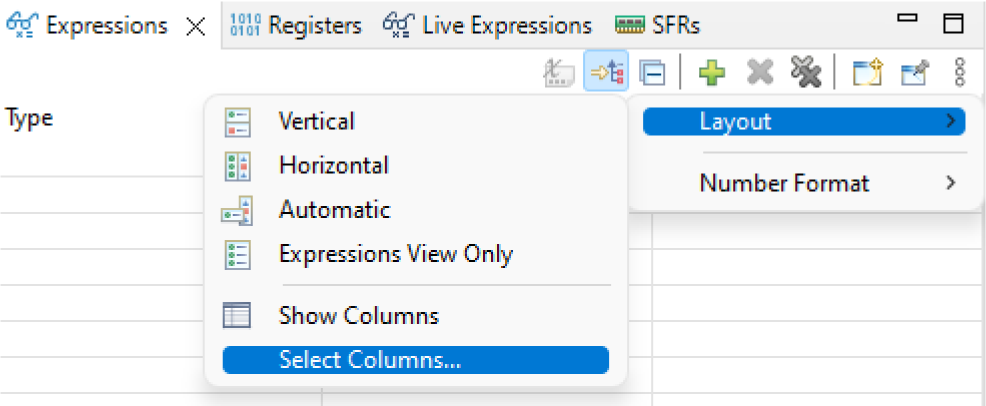
While debugging, you can access the memory of the board and can make some changes. In order to look at the variable value on the memory, you need to know the memory address of the variable. Build the project and look at the *{ProjectName}.map* file.

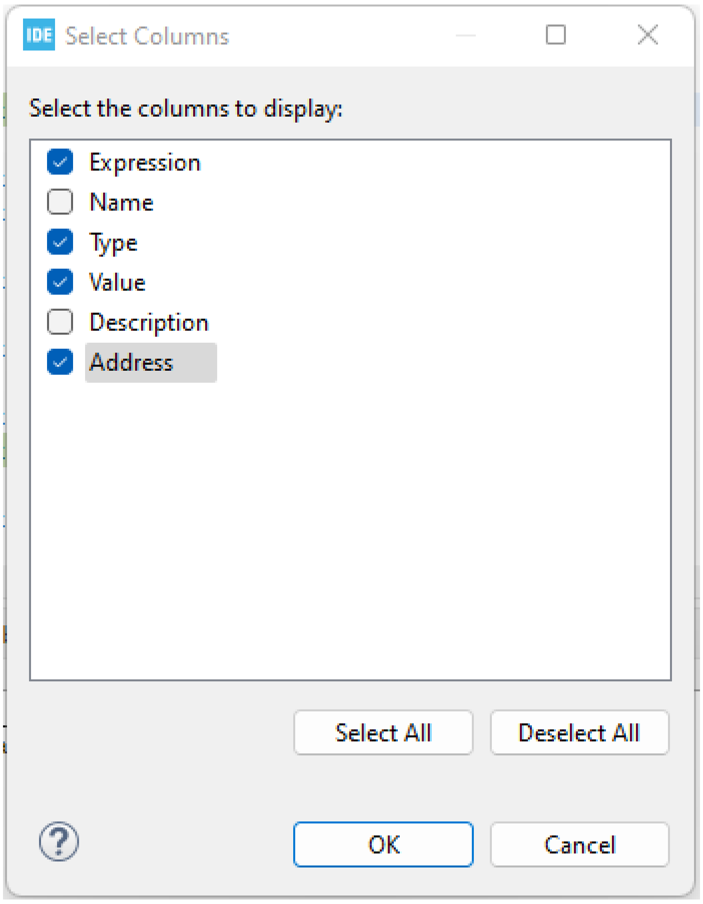
- Find the "wait_millisecond" variable at map file and paste screenshot (only that part):



```
 .data.wait_millisecond
                0x0000000020000000        0x4 ./Src/main.o
                0x0000000020000000                wait_millisecond
```

Also you can see the address of the variables at the IDE. *Show View - Expression View*.





Select the columns to display:

☑ Expression
□ Name
☑ Type
☑ Value
□ Description
☑ Address

Add the “wait_millisecond” variable to *Expression*.

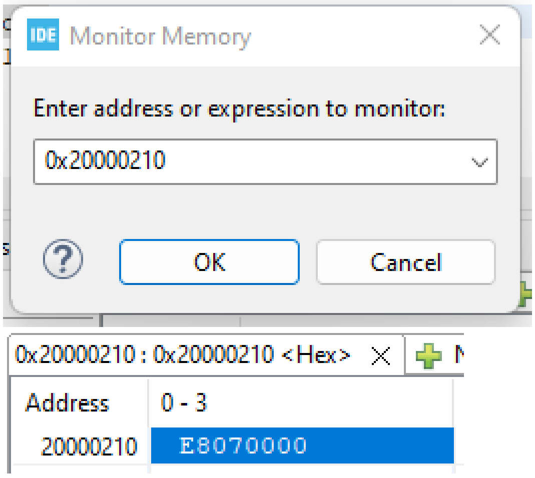
- What is the address of the “wait_millisecond” variable: 0x20000000

Not all the expressions are stored in the memory. *Show View - Registers*:

- Which register stored the wait_millisecond*333 value: r2

3) Memory

You can change the variable values on the memory during debugging. *Show View - Memory*, Add Memory Monitor and write the address of the variable.



Enter address or expression to monitor:

0x20000210

0x20000210 : 0x20000210 <Hex>

Address	0 - 3
20000210	E8070000

- Variable type is uint32 and if you see E8010000 in the memory, what is the value of the variable:

488

- Change the “wait_millisecond” value on the memory with E8070000, what changes on the board?

Time between the consecutive blinks (blue and off) increases – which is quite expected since we increased the waiting time caused by the loops.

4) Compiler Optimization

When you build your code, on the console (text data bss dec hex) will be written.

- What are these values for your projects?

Text	Data	Bss	Dec	Hex
980	12	1572	2564	a04

Open the properties of the project and change the Optimization to -Ofast.

- What are these (text data bss dec hex) values for this new setting?

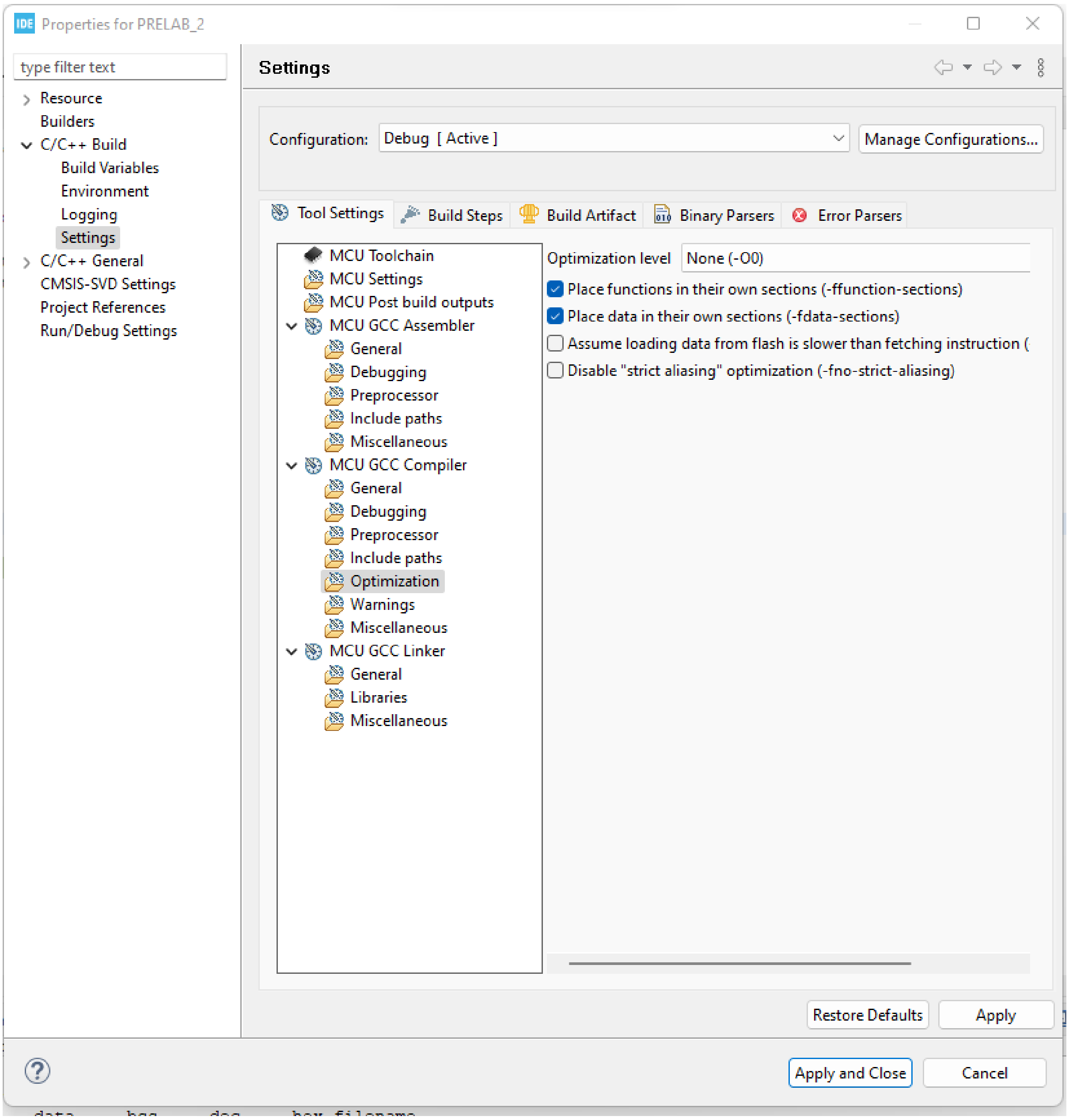
Text	Data	Bss	Dec	Hex
872	8	1568	2448	990

- Run your optimized code on the board, what changes on the board?

LD2 stops blinking as previously and stays blue throughout the run.

- Change *int index;* with *volatile int index;* what changes on the board?

Well, this time LD2 starts blinking blue/off again but faster than the previous runs (compared to the non-optimized version with *int index*).



Configuration: Debug [Active]

Optimization level: None (-O0)

☑ Place functions in their own sections (-ffunction-sections)

☑ Place data in their own sections (-fdata-sections)

□ Assume loading data from flash is slower than fetching instruction (

□ Disable "strict aliasing" optimization (-fno-strict-aliasing)

5) Submission

You will submit one zip file which contains this document and your project (all the files with the last configuration)

The naming of the zip file should be:

PRELAB<exp num>_<StudentID>.zip

6) Related Videos and Links

For *text data bss dec hex* meaning:

https://mirzafahad.github.io/2021-05-08-text-data-bss/

For debugging:

https://www.youtube.com/watch?v=BVC7KaUNCS8

Volatile Keyword:

https://www.youtube.com/watch?v=W3pFxSBkeJ8