**LAB: Assembly Programming**

**I. Overview**

In this tutorial, you will learn assembly programming to control an MUC. You will learn how to turn on LED assembly coding.

The objectives of this lab are learning

* How data flows between CPU register and memory
* How to program in assembly

**II. Pre-Lab**

1. **ARM-Cortex M4 Core Registers**

- Read the reference manual(CPU register) and programming manual for assembly programming.

|  |  |
| --- | --- |
|  |  |
| (a) ARM Cortex M4 Processor Registers | (b) Debugger Regsiter Window |

[출처] <https://developer.arm.com/docs/dui0553/a/the-cortex-m4-processor/programmers-model/core-registers>

R0 ~ R15 is already defined, so you can use these variables in keil compiler without definition. With assembly programming, you can load the address or the value of variables on each register. This process is necessary when you set peripheral register in assembly.

**B. Some Assembly Instructions**

* LDR

|  |
| --- |
| **LDR r0, addr\_var1** ; load the memory address of var1 via label addr\_var1 into R0 |
|  |
| **LDR r1, addr\_var2** ; load the memory address of var2 via label adr\_var2 into R1 |
|  |
| **LDR r2, [r0]** ; load the value (0x03) at memory address found in R0 to register R2 |
|  |

* STR

|  |
| --- |
| **STR r2, [r1]** ; store the value found in R2 (0x03) to the memory address found in R1 |
|  |

* With offset

|  |
| --- |
| **str r2, [r1, #2]** *; offset mode. Store the value found in R2 (0x03) to the memory address found in R1 plus 2. Base register (R1) unmodified.* |
|  |
| **str r2, [r1, #4]!** *;* ***pre****-indexed mode. Store the value found in R2 (0x03) to the memory address found in R1 plus 4. Base register (R1) modified: R1 = R1+4* |
|  |
| **ldr r3, [r1], #4** *;* ***post****-indexed mode. Load the value at memory address found in R1 to register R3. Base register (R1) modified: R1 = R1+4* |
|  |

[출처] <https://azeria-labs.com/memory-instructions-load-and-store-part-4/>

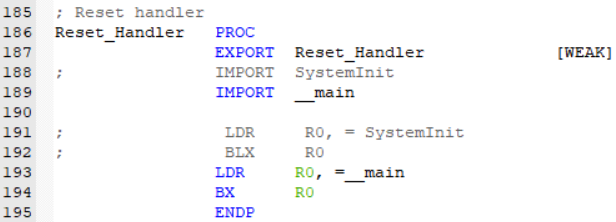
* Bit operation

**III. Tutorial**

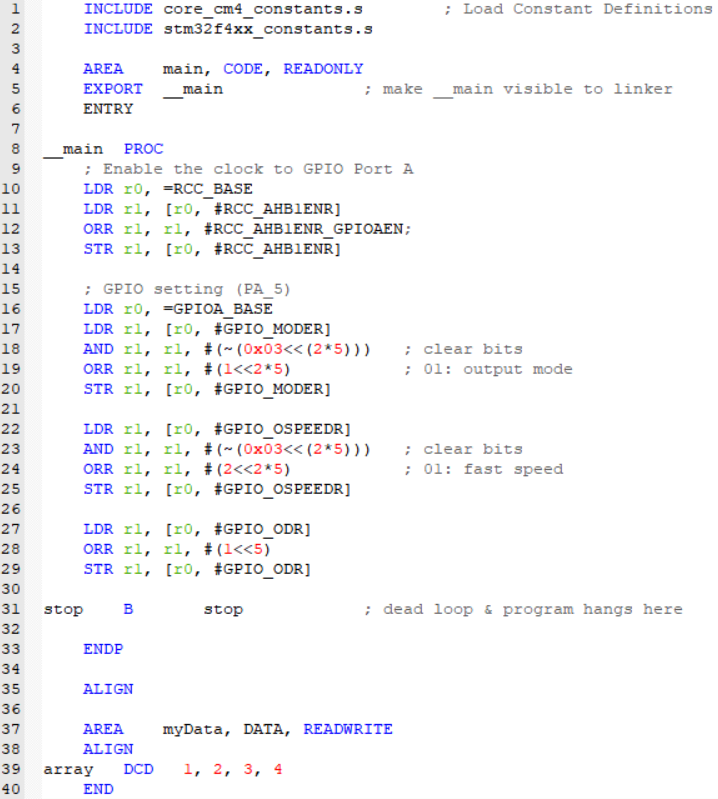
**A. Assembly programming for LED ON**

**1. Creating projects**

* Create a new project ‘**Tutorial6**’. Include ‘**startup\_stm32f411xe.s**’ and ‘**stm32f4xx\_constants.s**’ files.
* You have to annotate codes related to ‘SystemInit’ function in ‘**startup\_stm32f411xe.s**’. You can use ‘;’ character for annotation (or comment) in assembly programming.



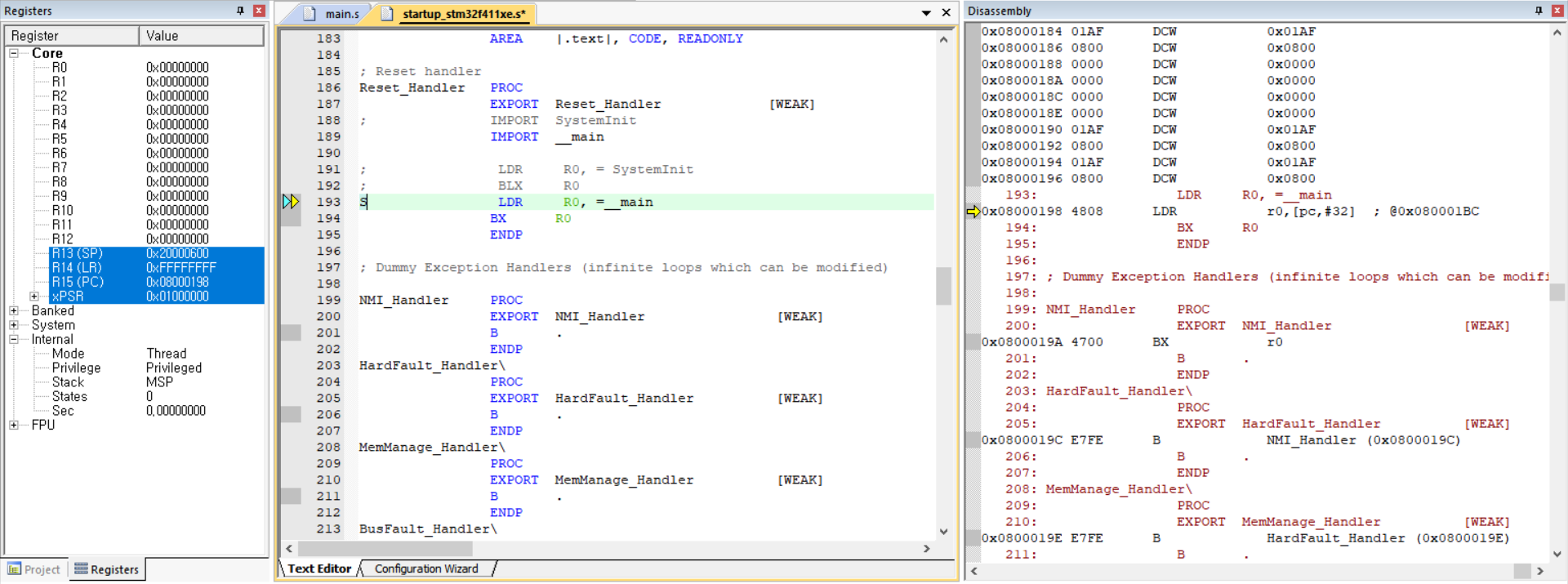
* Create the main source named as ‘**main.s**’ and copy following codes..



* Compile(F7) and flash(F8) the source code on board.
* Push the reset button(black) and verify the performance. LED should be turned on.

**2. Debugging**

* Click  button or use push ‘Ctrl + F5’ to enter debugging session.
* You can watch core registers and disassembly windows.



* Click the Run  button or push ‘F11’ for code execution.
* Watch how the values of R0 and R1 changes as code executes 1 by 1.

**IV. Exercise/Demo**

1. **Create an assembly code for calculating the dot product of two integer vectors x, y (5x1)**
   * *Pg 57 of Zhu🡪 give a template.*
2. **Mix C and Assembly** 
   * Create an assembly function that adds four numbers
   * *Pg 90 of Zhu🡪 give a template*

|  |
| --- |
|  |

* + Check the output pin with oscilloscope and observe how the signals change with input button

**Appendix**