Mark Lucernas
Tamara Sagakova

Unit 6 Lab (Group Lab Assignment)

The Code & Description

```
AREA myData, DATA


b    EQU 7
c    EQU 2
d    EQU 5
e    EQU 10
f    EQU 11

   AREA MYCODE, Code
        ENTRY
        EXPORT __main

__main

   LDR R0, =b    ; 7
   LDR R1, =c    ; 2
   LDR R2, =d    ; 5
   LDR R3, =e    ; A
   LDR R4, =f    ; B

   ADD R6, R0, R1     ; R0 (0x7) + R1 (0x2) = R6 (0x9)
   SUB R5, R6, R2     ; R6 (0x9) - R2 (0x5) = R5 (0x4)
   MUL R7, R4, R1     ; R4 (0x11) * R1 (0x2) = R7(0x16)
   ADD R8, R6, R3     ; R6 (0x9) + R3 (0xA) = R8 (0x13)
   SUB R9, R7, R1     ; R7 (0x16) - R1 (0x2) = R9 (0x14)
   MOV R10, R1        ; R10 = R1 (0x2)
   SUB R11, R10, R1   ; R10 (0x2) - R1 (0x2) = R11 (0x0)
   MUL R12, R5, R6              ; R5 (0x4) * R6 (0x9) = R12 (0x24)
   SUB R6, R0, R1     ; R1 (0x7) - R0 (0x2) = R6 (0x5)
   ADD R9, R3, R4     ; R3 (0xA) + R4 (0xB) = R9 (0x15)

stop    B    stop

   END
```

What the code does:

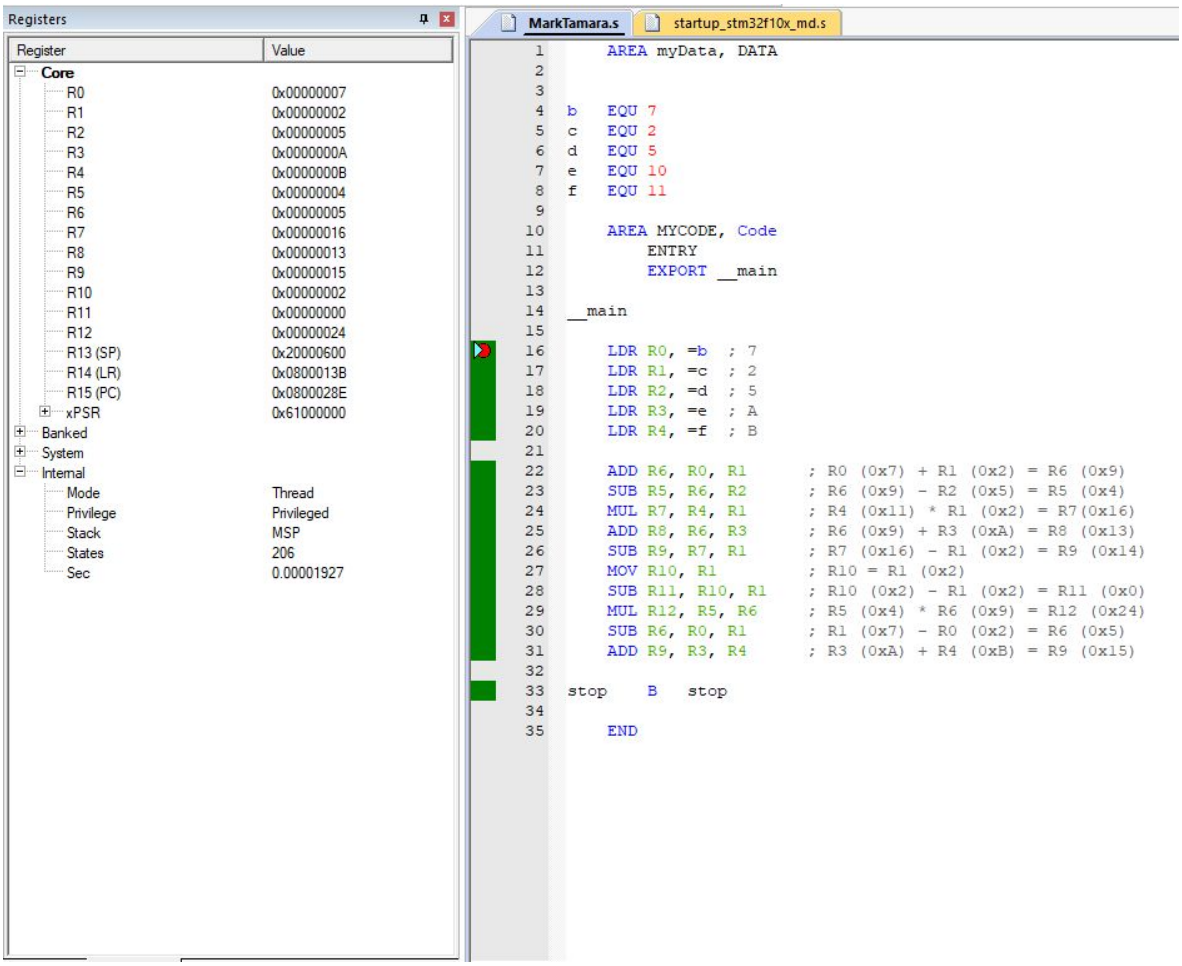
Our program utilizes the different basic instructions in assembly language which executes operation with registers. Our motive for this program is to store our favorite numbers being the most significant starting from register 0 (R0) to the least significant into register 12 (R12).

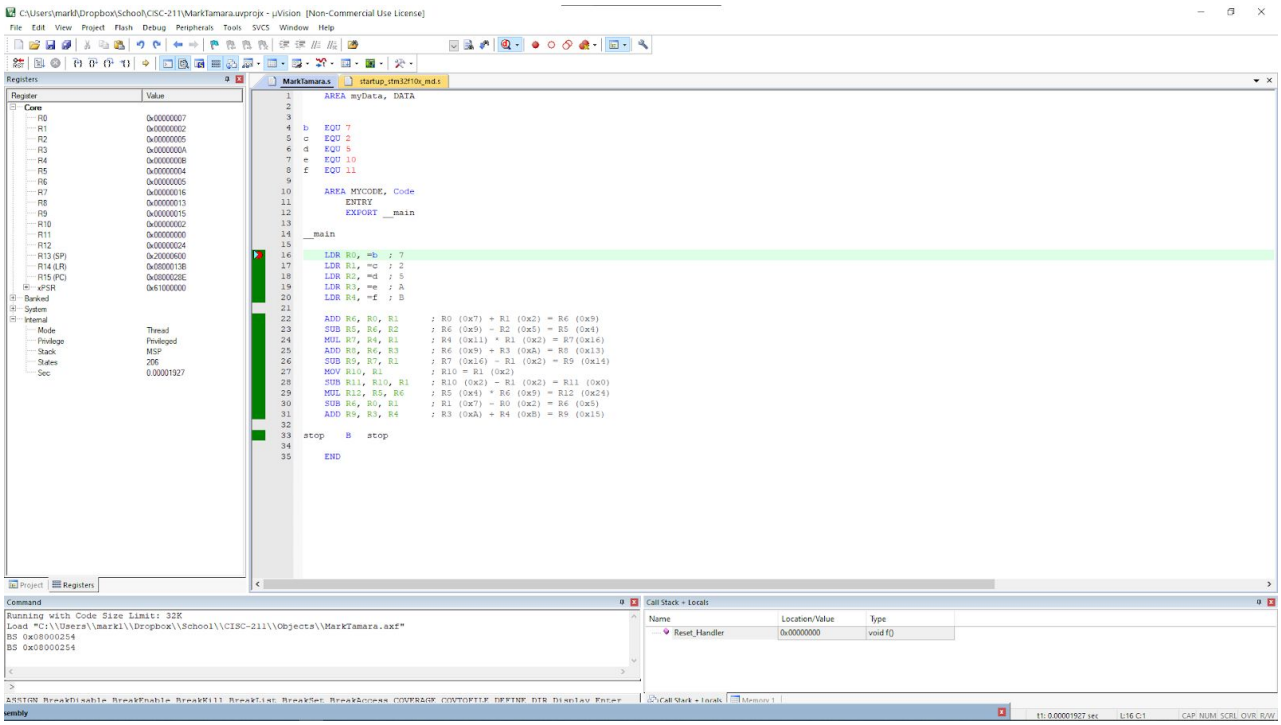
Lines 4 to 8 pre-defines data into five different variables (b, c, d, e, f) which would later be loaded into the registers for arithmetic operations. Lines 16 to 20 loads all the values from the variables into registers R0 to R4 accordingly. Then, lines 22 to 31 reads from the loaded registers to perform some operations, storing the results into registers R5 to R12. Finally, line 33 calls the *Branching* instruction that branches to the label *stop*, that is itself, creating a forever loop for debugging purposes.

Zoom Recording:

https://sdccd.us-west-2.instructuremedia.com/embed/424b6874-8174-45e1-a839-6fee2672484b

High resolution screen captures:




C:\Users\markl\Dropbox\School\CISC-211\MarkTamara.uvprojx - µVision [Non-Commercial Use License]
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Register
Value
Core
R0 0x00000007
R1 0x00000002
R2 0x00000005
R3 0x0000000A
R4 0x0000000B
R5 0x00000004
R6 0x00000009
R7 0x00000016
R8 0x00000013
R9 0x00000015
R10 0x00000002
R11 0x00000000
R12 0x00000024
R13 (SP) 0x20000600
R14 (LR) 0x0800013B
R15 (PC) 0x0800029E
xPSR 0x61000000
Banked
System
Internal
Mode Thread
Privilege Privileged
Stack MSP
States 206
Sec 0.00001927
Project
Registers
MarkTamara.s
startup_stm32f10x_md.s
1 AREA myData, DATA
2
3
4 b EQU 7
5 c EQU 2
6 d EQU 5
7 e EQU 10
8 f EQU 11
9
10 AREA MYCODE, Code
11 ENTRY
12 EXPORT __main
13
14 __main
15
16 LDR R0, =b ; 7
17 LDR R1, =c ; 2
18 LDR R2, =d ; 5
19 LDR R3, =e ; A
20 LDR R4, =f ; B
21
22 ADD R6, R0, R1 ; R0 (0x7) + R1 (0x2) = R6 (0x9)
23 SUB R5, R6, R2 ; R6 (0x9) - R2 (0x5) = R5 (0x4)
24 MUL R7, R4, R1 ; R4 (0x11) * R1 (0x2) = R7(0x16)
25 ADD R8, R6, R3 ; R6 (0x9) + R3 (0xA) = R8 (0x13)
26 SUB R9, R7, R1 ; R7 (0x16) - R1 (0x2) = R9 (0x14)
27 MOV R10, R1 ; R10 = R1 (0x2)
28 SUB R11, R10, R1 ; R10 (0x2) - R1 (0x2) = R11 (0x0)
29 MUL R12, R5, R6 ; R5 (0x4) * R6 (0x9) = R12 (0x24)
30 SUB R6, R0, R1 ; R1 (0x7) - R0 (0x2) = R6 (0x5)
31 ADD R9, R3, R4 ; R3 (0xA) + R4 (0xB) = R9 (0x15)
32
33 stop B stop
34
35 END
Command
Running with Code Size Limit: 32K
Load "C:\\Users\\markl\\Dropbox\\School\\CISC-211\\Objects\\MarkTamara.axf"
BS 0x08000254
BS 0x08000254
Call Stack + Locals
Name
Location/Value
Type
Reset_Handler
0x00000000
void f()
t1: 0.00001927 sec
L:16 C:1
CAP NUM SCRL OVR R/W