

CMPE-250 Laboratory Exercise 3
Memory, Conditional Branching and Debugging Tools

By submitting this report, I attest that its contents are wholly my individual writing about this exercise and that they reflect the submitted code. I further acknowledge that permitted collaboration for this exercise consists only of discussions of concepts with course staff and fellow students; however, other than code provided by the instructor for this exercise, all code was developed by me.

Chris Larson

Performed 11 September 2018

Submitted 18 September 2018

Lab Section 2

Instructor: Muhammad Shaaban

TA: Sebastian Echeverria

Anthony Bacchetta

Sahil Gogna

Lecture Section 01

Professor: Alessandro Sarra

SCREENSHOTS

Memory 1							
Address: 0x1FFFE100							
0x1FFFE100:	0000000023	-0000000006	0000000000	0000000000	00000000115	0000000000	0000000000
0x1FFFE11C:	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000
0x1FFFE138:	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000
0x1FFFE154:	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000
0x1FFFE170:	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000
0x1FFFE18C:	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000
0x1FFFE1A8:	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000
0x1FFFE1C4:	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000

Memory 1							
Address: 0x1FFFE100							
0x1FFFE100:	-0000000003	-0000000004	0000000000	0000000000	0000000005	0000000000	0000000000
0x1FFFE11C:	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000
0x1FFFE138:	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000
0x1FFFE154:	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000
0x1FFFE170:	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000
0x1FFFE18C:	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000
0x1FFFE1A8:	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000
0x1FFFE1C4:	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000	0000000000

Registers	
Register	Value
Core	
R0	0x00000041
R1	0xFFFFFFFF
R2	0xFFFFFFFF
R3	0x00000039
R4	0x00000008
R5	0x08000000
R6	0x1FFFE110
R7	0x00000080
R8	0x88888888
R9	0x99999999
R10	0xAAAAAAAA
R11	0BBBBBBBB
R12	0CCCCCCCC
R13 (SP)	0x1FFFE100
R14 (LR)	0xEEEEEEEE
R15 (PC)	0x00000158
xPSR	0x01000000

Registers	
Register	Value
Core	
R0	0x00000073
R1	0x00000017
R2	0xFFFFFFFF
R3	0x00000073
R4	0x00000000
R5	0x8B000000
R6	0x1FFFE110
R7	0x00000080
R8	0x88888888
R9	0x99999999
R10	0xAAAAAAAA
R11	0BBBBBBBB
R12	0CCCCCCCC
R13 (SP)	0x1FFFE100
R14 (LR)	0xEEEEEEEE
R15 (PC)	0x00000158
xPSR	0x01000000

CALCULATIONS FOR PRELAB

$$F = 2P - 3Q + 51$$

$$G = 5P - 4Q + 7$$

$$\text{Result} = F + G$$

Input Set 1 P=23, Q= -6

$$F = 0002*0017 - 0003*FFFA + 0033 = 0073$$

$G = 0005 * 0017 - 0004 * \text{FFFA} + 0007 = 0000$ **OVERFLOW**
Result = 0073

Input Set 2 P= -3, Q= -4

$F = 0002 * \text{FFFD} - 0003 * \text{FFFC} + 0033 = 0039$

$G = 0005 * \text{FFFD} - 0004 * \text{FFFC} + 0007 = 0008$

Result = 0039 + 0008 = 0041

QUESTION

Could you reduce or eliminate overflow by changing the order of operation within the expressions (F, G, and/or Result)? Explain why or why not.

- No because the result of input set 1 for the equation of G equals a number that is always going to be higher than 128.