Computer Architecture — Course: IT089IU
International University – VNU HCM — Date: March 2021
Dr. Le Hai Duong & Dr. Ly Tu Nga — Time: 6 hours

Laboratory Session 3

I. Bitwise Logic and Intro. to Procedure (70pts)

In these exercises, you can only use the following instructions:

```
and  andi nor  or   ori  sll  srl  xor  xori
```

1. **Exercise 1: (35pts) Write a program that**
   1.1 Put the number 0xDEADBEEF into register $t1 without using pseudoinstruction **li**. (**lab3_1_1.s**)
   1.2 Redo 1.1 as follows: use **ori** to load each letter into register. (**lab3_1_2.s**)
   1.3 Suppose that $t1 = 0xDEADBEEF. Using only register-to-register logic and shift instructions, Reverse the order of the bytes in $t1 so that register $t2 get the bit pattern 0xFEEBDAED. (**lab3_1_3.s**)
   1.4 Redo 1.3 using only **and**, **or**, and rotate instructions. (**lab3_1_4.s**)

2. **Exercise 2: (15pts) Write a program that**
   2.1 Set the corresponding bit in register $t1 through $t8. That is, in register $t1 set bit 1, register $t2 set bit 2, and so on. (**lab3_2_1.s**)
   2.2 By using **ONLY** shift instructions and register to register logic instructions (no **li** pseudoinstruction or **addi**), put the pattern 0xFFFFFFFF into register $t1. (**lab3_2_2.s**)

3. **Exercise 3: (20pts) Write a program that**
   3.1 Read in **ONE** unsigned integer in the range 0 to 15. Print out that number in hexadecimal. For example, given the input 13, print out `0xD`. (**lab3_3_1.s**)
   3.2 Modify the previous assembly, create a procedure `printHex(int num)`. This procedure takes in a number and print it out in hexadecimal. (**lab3_3_2.s**)
   3.3 Modify the previous assembly so that it can print out hexadecimal of any 32-bit integer input. For example, read in number 546263, print out `0x855D7`. (**lab3_3_3.s**)

II. MSP430 (30pts)

Given a sample code to control LED via pushing button in MSP430 as follows

No.	Sample codes	Comments/Results/ Functions
1. 2	`#include <msp430.h>` `#define Red BIT0`	

```
3   #define Green BIT6
4   #define Button BIT3
5
6   void main(void) {
7
8      WDTCTL = WDTPW | WDTHOLD;
9      P1OUT |= Red;
10     P1OUT &= ~Green;
11     P1DIR |= Red +Green;
12
13     P1DIR &= ~Button;
14
15     P1REN |= Button;
16     P1OUT |= Button;
17
18     while(1)
19     {
20          if ((P1IN & Button)!= Button)
21          {
22                  while ((P1IN & Button)!= Button)
23                  {
24
25                  }
26                  P1OUT ^= Red + Green;
27          }
28     }
29  }
```

Step 1: build the sample code in CCS, check the errors.

Step 2: **Not run**, the values of these registers (PORT_1_2):

P1OUT:
P1IN:
P1DIR:
P1REN:
P1IFG:

Step 3: Run, observe and collect the values of these registers in case of

	Red LED On	**Green LED On**
P1OUT		
P1IN	0x0F=0000111	
P1DIR		
P1REN		
P1IFG		

Comment and explain the Table above:

_____

_____

When running and pausing, click View and open ***Disassenbly*** window, write down **these instructions of sample code above:**

No.	C code	MIPS code
1.		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		
17		
18		
19		
20		
21		
22		
23		
24		
25		
26		
27		
28		
29		

Based on the Table above, **please explain the process of sample code under if-then else conditional statements:**

_____

_____

_____

_____

_____

_____

Problem 1: modify the sample code in order to when pressing the button two LEDs turn on and vice versa.

Your code:

Reference:

1. https://en.wikibooks.org/wiki/MIPS_Assembly/Pseudoinstructions
2. https://courses.missouristate.edu/KenVollmar/MARS/Help/SyscallHelp.html
3. https://www.assemblylanguagetuts.com/mips-assembly-programming-tutorials/#MIPS_Data_Types
4. https://en.wikibooks.org/wiki/MIPS_Assembly/Arithmetic_Instructions
5. https://gab.wallawalla.edu/~curt.nelson/cptr280/lecture/mips%20arithmetic%20instructions.pdf