Lab assignment 1

Q1. Run the demo to see if the output is same with the sample picture below ?If not please find the reason and modify it.

```
please input an integer : 3
it is an odd number (0: false,1:true) : 1
```

```
.include "macro_print_str.asm"
.text
main:
    print_string("please input an integer : ")
    li $v0,5
    syscall
    move    $t0,    $v0
    nor     $t1,    $zero,    $zero
    sra     $t2,    $t1,        31
    and     $a0, $t2,         $t0
```

```
print_string("it is an odd number (0: false,1:true) : ")
    li $v0,1
    syscall

    end
```

please submit the modified asm file(**LabA1Q1_*yourID*.asm**) and "macro_print_str.asm" in Sakai site, write down the reasons which mentioned in the Q1 to the report (**LabA1_*yourID*.pdf**).

Q2. We have already introduced three addressing methods of memory before winter holidays, they're direct addressing, indirect addressing and baseline addressing, and we have also known how to move an immediate value to general purpose registers (we can call this as immediate value addressing, it is a kind of register addressing method, but not a kind of memory addressing method, please note this), so four kinds of addressing method are represented.

In this homework, please accomplish addition, subtraction, multiplication and division operations of **two integers** at least once, and the operands sent to registers are fetched by **different addressing method** for each operation, and the **results are expected to store in memory** (when storing the results, you can choose any kind of addressing method). After storing the results, please **print the equations** in screen. And **detailed comments** are also expected.

Please run the codes in QtSpim or Mars, and capture the **screenshot of data segment** both before and after the code running, and **point out where the operands and results are** in data segment as Fig. 1 and Fig. 2 showed.

At last, please submit two files in Sakai site, one is source code file(**LabA1Q2_*yourID*.asm**), and the other is screenshot of data segment(**LabA1_*yourID*.pdf**).

1) Tips1: You can design your code as the comments showed

#.data

#distribute two integers for immediate value addressing method

#define all the labels of address in memory in which operands and results are stored

#.text

#immediate value addressing

#fetch the operands

#calculate

#store the result at address 0x********

#print the equation on screen in the form of A + B = C

#direct addressing

#.....

#indirect addressing

#.....

#baseline addressing

#.....

2) Tips2: screenshots of data segment

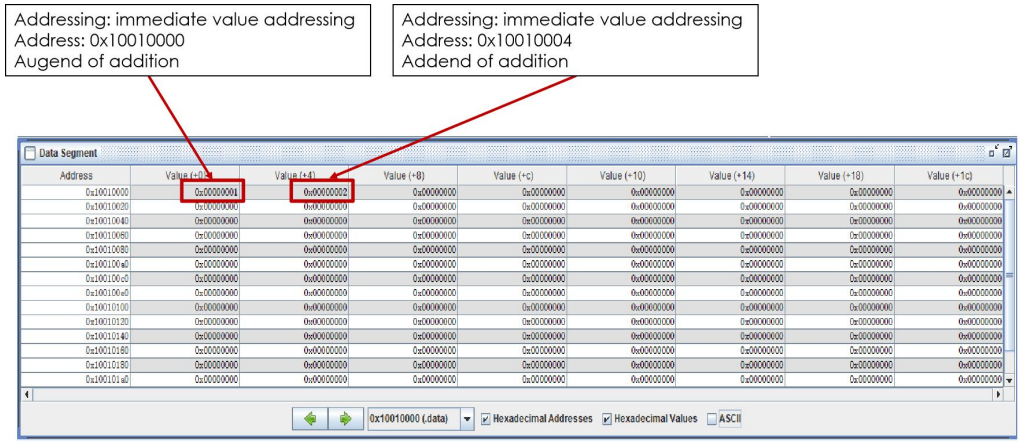

Fig.1. screenshot of data segment before running

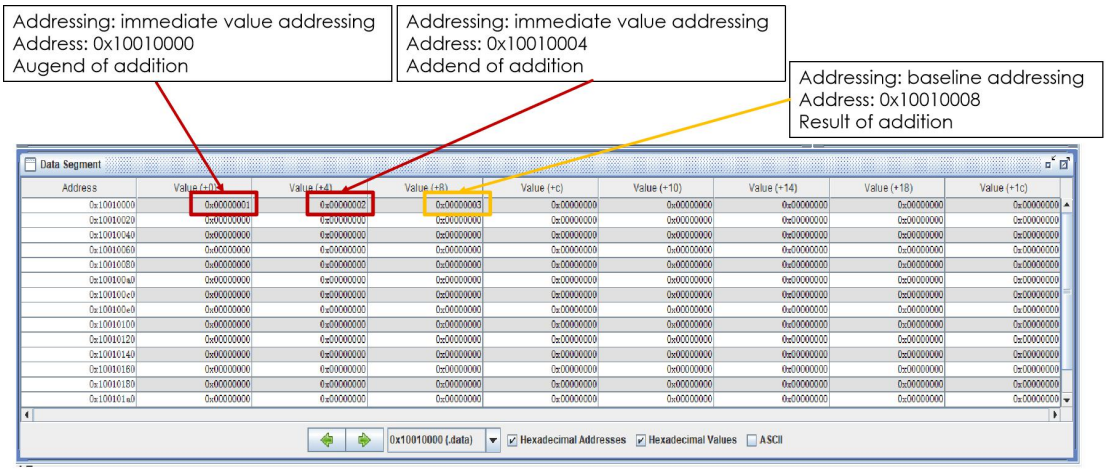

Fig.2. screenshot of data segment after running