CS2610 Computer Organization Laboratory
Lab - 6

Objective:

In this lab you will familiarize the basics of MIPS assembly programming using the QtSpim simulator.

Part-A

Problem:

Illustrate the following using QtSpim:

(1) Data transfer operations:

(a) Load all the temporary registers t0 to t9 with 4-bit hex value 0xF and saved registers s0 to s7 with the 4 bit hex value 0xA. Move data in t6 to s4 and s7 to t5

(b) Load all the temporary registers t0 to t9 with 16-bit hex value 0xFFFF and saved registers s0 to s7 with the 16 bit hex value 0xAAAA.

(c) Load all the temporary registers t0 to t9 with 32-bit hex value 0xFFFFFFFF and saved registers s0 to s7 with the 32 bit hex value 0xAAAAAAAA.

(d) Load all the temporary registers t0 to t9 and saved registers s0 to s7 with the 36 bit hex value 0x123456789.

Explain your observations in each of the above 4 cases.

(2) Storage of data in the data segment using ascii/asciiz:

Store 4 strings as indicated below. Load the address of each string into registers t0, t1, t2, t3 respectively. Run step by step and note how each string is getting stored and explain the changes occuring in the data segment:

```
str1:    .ascii "123456789abcedef"
str2:    .ascii "123456789ABCDEF"
str3:    .asciiz "123456789abcedef"
str4:    .asciiz "123456789ABCDEF"
```

(3) Load and store a word, a half word and a byte:

Store the strings as indicated below:

str1:   .ascii "123456789abcedef"
str2:   .ascii "123456789ABCDEF"

(a) Load the first 3 bytes of str1 into t0, t1, t2 and store the values in registers t3, t4, t5 into last 3 bytes of str2.
(b) Load the first 2 half words of str2 into t6, t7 and store the value in the register t8 into last halfword of str1.
(c) Load the last word of str2 into t9 and store the value in the register s1 into the last word of str1.

(4) Illustrate the use of syscall codes for the following operations :

(a) Exiting a program

(b) Printing an integer and a string:
Store your roll number as memory word. Then display it on the Console. The display should be like:

My roll number is : <Your roll number>

(c) Reading and printing an integer and string:
Read your name and roll number during run time and then display the same on the Console. There should be prompts for entering the name and roll number. ie. The display should be like:

Please enter your name: <Enter your name>
Please enter your roll no.: <Enter your roll number>
Your name is : <Your name should be displayed here>
Your roll number is : < Your roll number should be displayed here>

Indicate the changes that you see in in data segment.

(d) Reading and printing a floating point number
Read a floating point number during run time and then display the same on the Console. ie. The display should be like:

Please enter a floating point number : <Enter the number>
You have entered : <The number should be displayed here>

Part-B

(5) Arithmetic operations:

Display the sum, difference, product and quotient and remainder of two numbers in the console.

(a) When both the numbers are positive.
(b) When both the numbers are negative.
(c) When one number is positive and one is negative.
(d) Suppose your inputs are two hex numbers 0x08 and 0x04. Find the product (8*4) and quotient (8/4) without using the 'mul' and 'div' instruction.

For (a), (b), (c), the input numbers should be allowed to be given during run time. Use display statements wherever needed.

Remember that the arithmetic operations inside a computer is in 2's complement.

(6) Illustration of logical operations-1:

Common data Questions:

The RGBA(Red, Green, Blue, Alpha) codes of the background colour of an image are given as {0xee, 0xff, 0xdd, 0x0}. They are stored in register t0 as $t0 = 0x0eeffdd0).

Use minimal number of instructions for all the operations.

(a) Extract the 'red' and 'blue' colour codes to registers t1 and t2 respectively using register t0 **alone** (ie. $t1=0xee and $t2= 0xdd). You are not allowed to directly load 0xee and 0xdd to the registers $t1 and $t2.

(b) Modify your code to extract the 'green' colour code to register t2. You should use only registers t0 and t2 for this operation. All the operations should be performed through values loaded in registers only.

(c) The colour code of 'red' needs to be changed from 0xee to 0xbb that of 'green' from 0xff to 0xcc. Extract the modified set of of colour codes into register t3 by using the previous value already present in register t0 (ie. 0x0eeffdd0).

(d) Set the Alpha code to 0x1 and store the modified set of colour codes in register t4. Other colour codes remain same as in t0.

(7) Illustration of logical operations-2:

You are provided with a LED array of 32 LEDs. Create a two way running led using MIPS assembly. It means that the LED initially get turned on in the one way and then in reverse direction.

It should work as follows. The leds are given numbers 0 to 31. Initially LED 2 should be ON, then 18, then 0 and then in the reverse order. The process need not be repeated.

(Hint: Use a 32 bit register. Make use of rotate instructions instead of shift and loop instructions)

All the programs should include a header section with the program title(what the program does), comments wherever required.

Post-lab:

Submit a post-lab report with your verified outputs. The report should include your **well commented code snippets, snapshots of the changes observed in the registers, data segments with proper explanation.** Your submissions should be clear and concise.