Lab 2 Report

Amber Raymer

Amberraymer0562@floridapoly.edu

EEL4768C: Computer Architecture and Organization

Due Date: 31 January, 2019

NOTE: Professor Navid accidentally deleted my code and told me that I wouldn't need to rewrite the exercises. I've already gotten credit for showing him the code in class.

1.0 Project Description

Exercise 1: The objective of exercise one is to find the maximum of two numbers using conditional statements and then output the largest number.

Exercise 2a: The objective of exercise two a is to implement a program that increments from 0 to 15 and displays the results in decimal on the console.

Exercise 2b: Program 2b requires you to modify the code written in exercise 2a to display the results in hexadecimal on the console.

2.0 Program Design

Exercise 1: First, we have the user input two numbers that are stored in $t0 and $t1 respectively. Next, we use the conditional statement, bgt to find the larger number. If $t0 is larger than $t1, the program branches to t0_greater and stores the value in $t0 into $t2 and then prints $t2. If $t1 is larger than $t0, the program branches to t1_greater which stores $t1 in $t2 and then prints $t2. Both t0_greater and t1_greater jump to the "exit" label which uses "li $v0, 10" ("10" being the argument used to terminate) and a "syscall" to exit our program.

Exercise 2a: We begin in out "main" label and use an "add $t0, $0, $0" to store 0 in $t0. After that, we enter the "increment" label where we have a branch if equal instruction that checks if the value in $t0 is equal to 16. After that line, we use an "li" instruction to print the value in $t0 as a decimal. Next, we have an "addi" instruction that adds our immediate value 1 to the number in $t0. We then jump back to the beginning of "increment" until $t0 is equal to 16, where we then branch to "exit" to terminate the program.

Exercise 2b: We begin in out "main" label and use an "add $t0, $0, $0" to store 0 in $t0. After that, we enter the "increment" label where we have a branch if equal instruction that checks if the value in $t0 is equal to 16. After that line, we use an "li" instruction to print the value in $t0 as a hexadecimal value. Next, we have an "addi" instruction that adds our immediate value 1 to the number in $t0. We then jump back to the beginning of "increment" until $t0 is equal to 16, where we then branch to "exit" to terminate the program.

3.0 Symbol Table

The registers used in the MIPS code and their purpose are listed below. The labels used in the MIPS code are also identified and described. The data type is also identified where applicable. Table 1 is a symbol table for the registers used and Table 2 is a symbol table for the labels used.

Table 1a: Registers Used (Exercise 1)

Register	Use
$v0	Tells MARS what to do when a syscall is called and used to return values.
$a0	The argument to be used during some syscalls. Also used to pass arguments depending on the parameter used.
$t0	Used to hold the first number.
$t1	Used to hold the second number.

$t2	Used to store the larger number of $t1 and $t0.

Table 1b: Registers Used (Exercise 2a)

Register Use	
$v0	Tells MARS what to do when a syscall is called and used to return values.
$a0	The argument to be used during some syscalls. Also used to pass arguments depending on the parameter used.
$t0	Used to hold the number we are incrementing.
$0	Hard-wired to 0.

Table 2a: Labels Used (Exercise 1)

Register Use	
main	Executed first at runtime. Gets user inputs and uses bgt to find the largest number.
exit	Implements the syscalls necessary to terminate the program.
t0_greater	If $t0 is greater, this label stores $t0 in $t2 and then prints $t2.
t1_greater	If $t1 is greater, this label stores $t1 in $t2 and then prints $t2.

Table 2b: Labels Used (Exercise 2a and 2b)

Register Use	
main	Executed first at runtime. Gets user inputs and uses bgt to find the largest number.
exit	Implements the syscalls necessary to terminate the program.
increment	If $t0 is greater, this label stores $t0 in $t2 and then prints $t2.

4.0 Learning Coverage

The learning outcomes from Lab 2 are as follows:

- Use of conditional statements (beq, bgt)
- Printing decimal and hexadecimal values to the console
- Use of labels
- Terminating programs

5.0 Test Plan

Table 3a: Test Plan (Exercise 1)

This exercise has no external input values, so there isn't much to test for other than the standard case.

Test Case	Pixel Data	Expected Output
Standard/random	$t0 = 0	$t0 = 15

Table 3b: Test Plan (Exercise 2a)

Test Case	Pixel Data	Expected Output

Standard/random	$t0 = 3 $t1 = 5 $t0 = 5 $t1 = 3	$t2 = 5 $t2 = 5
Maximum values	$t0 = 2,147,483,647 $t1 = 2,147,483,646	$t2 = 2,147,483,646
Minimum values	$t0 = 0 $t1 = 1	$t2 = 1
Most bits different	$t0 = 131,071 $t1 = 0	$t2 = 131,071

Table 3c: Test Plan (Exercise 2b)

Test Case	Pixel Data	Expected Output
Standard/random	$t0 = 3 $t1 = 5 $t0 = 5 $t1 = 3	$t2 = 0x00000005 $t2 = 0x00000005
Maximum values	$t0 = 2,147,483,647 $t1 = 2,147,483,646	$t2 = 0x7FFFFFFF
Minimum values	$t0 = 0 $t1 = 1	$t2 = 0x00000001
Most bits different	$t0 = 131,071 $t1 = 0	$t2 = 0x0001FFFF

6.0 Test Results

Table 4a: Test Plan (Exercise 1)

Test Case	Pixel Data	Expected Output
Standard/random	$t0 = 0	$t0 = 15

Table 4b: Test Plan (Exercise 2a)

Test Case	Pixel Data	Expected Output
Standard/random	$t0 = 3 $t1 = 5 $t0 = 5 $t1 = 3	$t2 = 5 $t2 = 5
Maximum values	$t0 = 2,147,483,647 $t1 = 2,147,483,646	$t2 = 2,147,483,646
Minimum values	$t0 = 0 $t1 = 1	$t2 = 1
Most bits different	$t0 = 131,071 $t1 = 0	$t2 = 131,071

Table 3c: Test Plan (Exercise 2b)

Test Case	Pixel Data	Expected Output
Standard/random	$t0 = 3 $t1 = 5	$t2 = 0x00000005
	$t0 = 5 $t1 = 3	$t2 = 0x00000005
Maximum values	$t0 = 2,147,483,647 $t1 = 2,147,483,646	$t2 = 0x7FFFFFFF
Minimum values	$t0 = 0 $t1 = 1	$t2 = 0x00000001
Most bits different	$t0 = 131,071 $t1 = 0	$t2 = 0x0001FFFF

7.0 References

7.1 Lecture Notes

EEL 4768C Module-01, Module-02, Module-03: Slides adapted from Dr. DeMara (UCF), Dr. Sahawneh (FPU), Dr. Mutlu (CMU). They are available on Canvas.

7.2 MARS Simulator

The MARS Simulator for MIPS processors, available at:
http://courses.missouristate.edu/kenvollmar/mars/

7.3 MARS Syscall Reference Guide

http://courses.missouristate.edu/kenvollmar/mars/help/syscallhelp.html