Introduction to Computer Engineering

Winter 2014

Assignment 5: ADVANCED Assembly Language and computer architecture

Due Date: Friday April 11th, 2014 at 5:00PM

Please submit this assignment by 5:00 pm on the due date via *MyCourses*. Your assignment must be packaged as follows: one compressed file (.zip) containing the following files:

* Answers to all questions as well as discussions (including schematic circuit diagrams and timing printouts) in **one single** Microsoft Word (doc/docx) or portable document format (pdf) file. Note that scanned handwritten solutions are accepted.
* All files related to your SPIM programs. All files should be named according to the following convention:

LAST\_FIRST\_QX\_DESC.xxx

where “LAST” and “FIRST” are you last and first name respectively (no spaces), “QX” should be replaced by Q1, Q2 or Q3 depending on the question in the assignment to which your file belongs, “DESC” is an optional short description and “xxx” is the file extension.

A penalty of 10% will be applied if you do not follow these guidelines.

This assignment will be marked out of 40 points. **Late submissions will NOT be accepted.**

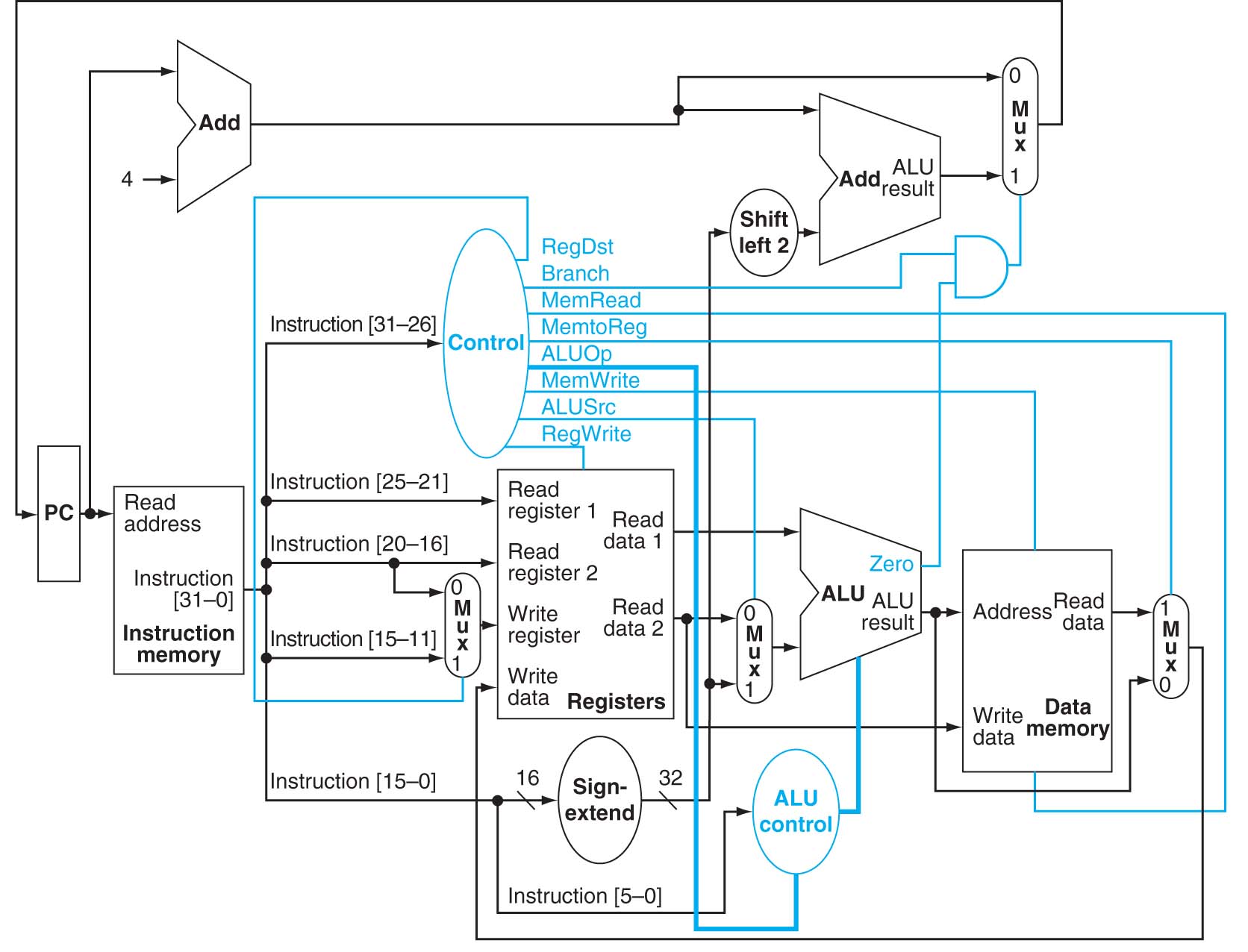
# Question 1 (20 points)

The figure below illustrates a simplified single-cycle data path for the MIPS architecture.

For each of the following instructions, write in the table the corresponding value of the control variables mentioned. Some of these values might be “don’t care”. (Note that you can simply copy and paste the table below into your own submission and fill it out)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Instructions | MemtoReg | ALUSrc | Branch | MemRead | RegDst | ALUOp | ALUOp4 |
| lw $4, 12($3) |  |  |  |  |  |  |  |
| addi $5, $4, -124 |  |  |  |  |  |  |  |
| bne $5, $4, exit |  |  |  |  |  |  |  |
| slt $4, $5, $4 |  |  |  |  |  |  |  |
| exit: sub $3, $4, $5 |  |  |  |  |  |  |  |

If, at the beginning of these 5 lines of code, the value of register $3 was 0x45000000, what is the value of this register at the end of their execution? You can assume that no overflow occurs.



ALUOp4

# Question 2 (20 points)

The following assembly code computes the greatest common divisor (gcd) of two positive integer numbers through the following algorithm:

The assembly code contains 3 errors that prevent the code from achieving its intended function. Indicate these three errors and how they should be corrected. Explain in detail what the impact of these errors is, and show through SPIM simulation that your corrected code works as expected.

.globl main

.text

## main function -- uses $v0, $a0 and $a1, $t0 as scratch, not preserved

main:

ori $v0, $0, 4

la $a0, prompt

syscall

ori $v0, $0, 5

syscall

move $t0, $v0

ori $v0, $0, 4

la $a0, prompt

syscall

ori $v0, $0, 5

syscall

move $a0, $t0 # first argument to gcd in $a0

move $a1, $v0 # second argument to gcd in $a1

jal gcd

move $t0, $v0

ori $v0, $0, 4

la $a0, outext

syscall

add $a0, $0, $t0 # print result

ori $v0, $0, 1

syscall

ori $v0, $0, 4

la $a0, crlf

syscall

ori $v0, $0, 10 # clean exit

syscall

########################################

# gcd function

# Arguments a and b in $a0 and $a1

# return value in $v0

# Scratch register: $t0 (not preserved)

########################################

gcd: addi $sp, $sp, -12

sw $ra, 0($sp)

sw $a0, 4($sp)

sw $a1, 8($sp)

bne $a1, $0, not0

add $v0, $0, $a0

j exgcd

not0: sltu $t0, $a1, $a0 # b<a?

beq $t0, $0, bgta

sub $t0, $a0, $a1

add $a0, $0, $a1

add $a1, $a0, $0

jal gcd

bgta: sub $a1, $a1, $a0

jal gcd

j exgcd

exgcd: addi $sp, $sp, 12

lw $a1, 8($sp) # restore

lw $a0, 4($sp)

lw $ra, 0($sp)

jr $ra

.data

prompt: .asciiz "Enter argument:"

crlf: .asciiz "\n"

outext:.asciiz "Result:"