Name: Id#

COE 301/ICS 233, Term 172

Computer Architecture & Assembly Language

Quiz# 4 Solution

 Date: Sunday, March 11, 2018

# Implement the following two procedures using MIPS assembly language. Use MIPS programing convention in saving registers.

## A procedure, RSum, that computes the sum of a given row. Assume that the procedure receives as parameters the address of the array in register $a0, the number of columns in register $a1, and the index of the row to be summed in register $a2. The procedure should return the sum of the row in register $v0.

RSum:

 xor $v0, $v0, $v0 # sum=0

 xor $t0, $t0, $t0 # i=0

# Computing starting address of row

 mul $t1, $a1, $a2

 sll $t1, $t1, 2

 add $t1, $t1, $a0 #t2 = starting address of row $a2

Loop:

lw $t2, ($t1)

 add $v0, $v0, $t2 # adding column elements

 addi $t1, $t1,4 # incrementing to next column element

 addi $t0, $t0, 1 # i = i + 1

 bne $t0, $a1, Loop

 jr $ra

## A procedure, ArrayRowSum, that displays the sums of all rows in the array based on using RSum procedure. Assume that the procedure receives as parameters the address of the array in register $a0, the number of rows in register $a1, and the number of columns in register $a2. Each row sum should be printed in a new line. To print an integer in register $a0, use syscall with $v0=1. To print a character in $a0, use syscall with $v0=11.

ArrayRowSum:

 addi $sp, $sp, -20 # allocate memory on stack

 sw $ra, ($sp) # save $ra

 sw $s0, 4($sp) # saving needed registers

 sw $s1, 8($sp)

 sw $s2, 12($sp)

 sw $s3, 16($sp)

 move $s0, $a0 # Array address

 move $s1, $a1 # Number of rows

 move $s2, $a2 # Number of colums

 xor $s3, $s3, $s3 # i = 0

Loop2:

move $a0, $s0

 move $a1, $s2

 move $a2, $s3

 jal RSum

 move $a0, $v0

 li $v0, 1 # print row sum

 syscall

 li $a0, '\n' # print new line

 li $v0, 11

 syscall

 addi $s3, $s3, 1

 bne $s3 $s1, Loop2

 lw $ra, ($sp) # restore $ra

 lw $s0, 4($sp) # restore saved registers

 lw $s1, 8($sp)

 lw $s2, 12($sp)

 lw $s3, 16($sp)

 addi $sp, $sp, 20 #free memory from stack

 jr $ra