 **California State University Long Beach**

**College of Engineering**

**Computer Engineering Computer Science Department**

**Thursday October 20, 2016**

**Computer Architecture**

**Lab Project -1**

**Name:Mark Levie Gumban Mendoza**

Table of Contents

[Problem Definition: 3](#_Toc465616526)

[1) Program it in MARS 3](#_Toc465616527)

[2) Explain its behavior 3](#_Toc465616528)

[3) Write the corresponding C statement? 3](#_Toc465616529)

[MARS Commented Source Code: 4](#_Toc465616530)

[Snapshot Outputs: 5](#_Toc465616531)

[1) Register Picture 5](#_Toc465616532)

[2) Memory Picture 6](#_Toc465616533)

[3) Console Picture: 7](#_Toc465616534)

[Problem Explanation and behavior. 8](#_Toc465616535)

Problem Definition:

# Problem Definition:

For the MIPS assembly instructions below,

## Program it in MARS

## Explain its behavior

## Write the corresponding C statement?

Assume that the variables f, g, h, i, and j are assigned to registers $s0, $s1, $s2, $s3, and $s4, respectively. Assume that the base address of the arrays A and B are in registers $s6 and $s7, respectively.

sll $t0, $s0, 2 #$t0 = f \* 4

add $t0, $s6, $t0 #$t0 = &A[f]

sll $t1, $s1, 2 #t1 = g \* 4

add $t1, $s7, $t1 #t1 = &B[g]

lw $s0, 0($t0) #f = A[f]

addi $t2, $t0, 4

lw $t0, 0($t2)

add $t0, $t0, $s0

sw $t0, 0($t1)

Note that you need to initialize the memory locations for A and B in the data fields as array variables. (You may want to research how to do that.)

Problem Solution:

# MARS Commented Source Code:

.data

A: .word 11,12,13,14 **#A[] =[10,20,30,40]**;

B: .word 15,16,17,18 **#B[] =[11,12,13,14]**;

.text

la $s6, A **#$s6 = A;**

la $s7, B **#$s7 = B;**

sll $t0, $s0, 2 **#$t0 = f \* 4**

add $t0, $s6, $t0 **#$t0 = &A[f];**

sll $t1, $s1, 2 **#t1 = g \* 4**

add $t1, $s7, $t1 **#t1 = &B[g];**

lw $s0, 0($0) **#f = A[f];**

addi $t2, $t0, 4 **#$t2 = &A[f] + 4;**

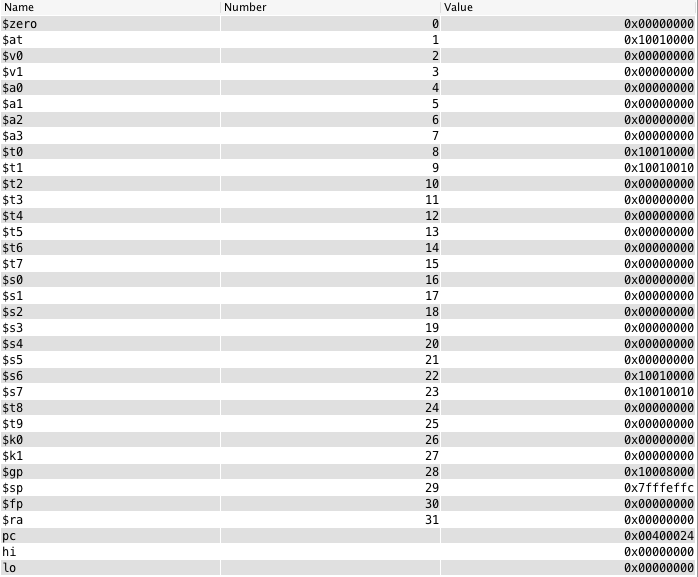
lw $t0, 0($t2) **#$t0=A[f+1];**

add $t0, $t0, $s0 **#t0 = A[f] + A[f+1]**;

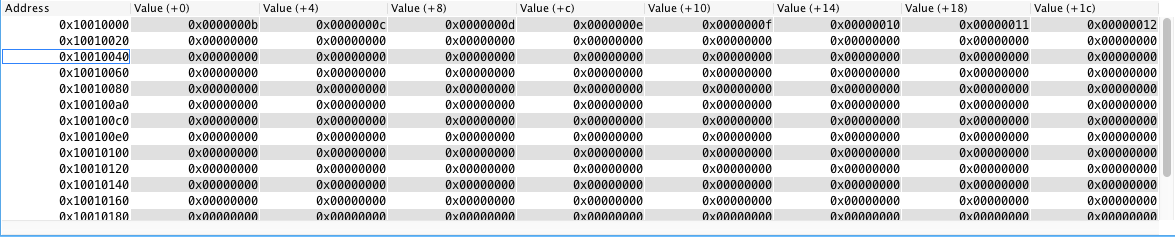
sw $t0, 0($t1) **#B[g] = A[f] + A[f + 1]**;

# Snapshot Outputs:

## Register Picture



## Memory Picture



## Console Picture:



# 

# Problem Explanation and behavior.

This problem involves 2 collections, of 4 words stored in memory, and accessing them and mutating them and storing them in the end. In order to do this, we do the following:

So first we must initialize the collections in memory, to do this we must do the following syntax:

.data

A: .word 11,12,13,14  
B: .word 15.16.17.18

Next we have to assign the addresses of these 2 collection into $s6 and $s7 with the following syntax.  
  
la $s6, A

la $s7, B

In MIPS, to access indices in a collection, you must know their address, each index in a collection, to calculate the indices of for example to 2nd element one must do the [index] \* [word size] which is 4, so the following syntax is what will be used to access the second element

sll $t0, $s0, 2

In order to access the element of A that was calculated above without overwriting $s6 we must add the address of A plus the offset which in this case would be 4 \* index however the calculation above resulted in 0 so currently we are doing &A + 0 with the following syntax:

add $t0, $s6, $t0

In MIPS, to access indices in a collection, you must know their address, each index in a collection, to calculate the indices of for example to 2nd element one must do the [index] \* [word size] which is 4, so the following syntax is what will be used to access the second element

sll $t1, $s1, 2

We do the same thing we did for A in “add $t0, $s6, $t0” but now applying to B with the offset being $t1 in this syntax.

add $t1, $s7, $t1

So by having the addresses of the elements in A and B storing them accordingly in $t0 and $t1 we can load the value of the element and save it in $s0

lw $s0, 0($t0)

to access element at index 1 we can add 4 to the address stored in $t0 using addi.

addi $t2, $t0, 4

then load that element and store it in t0

lw $t0, 0($t2)

then add the results from the load words we stored in $s0 and $t0

add $t0, $t0, $s0

and store that in the address of $t1 which is currently the base address of B replacing B’s first element

sw $t0, 0($t1)