

DATA TRANSFER IN MIPS

LAB # 04



Fall 2023

CSE-304L Computer Organization and Architecture Lab

Submitted by: **Ali Asghar**

Registration No.: **21PWCSE2059**

Class Section: **C**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”

Submitted to:

Dr. Bilal Habib

Date:

27th October 2023

Department of Computer Systems Engineering
University of Engineering and Technology, Peshawar

ASSESSMENT RUBRICS COA LABS

LAB REPORT ASSESSMENT				
Criteria	Excellent	Average	Nil	Marks Obtained
1. Objectives of Lab	All objectives of lab are properly covered [Marks 10]	Objectives of lab are partially covered [Marks 5]	Objectives of lab are not shown [Marks 0]	
2. MIPS instructions with Comments and proper indentations.	All the instructions are well written with comments explaining the code and properly indented [Marks 20]	Some instructions are missing are poorly commented code [Marks 10]	The instructions are not properly written [Marks 0]	
3. Simulation run without error and warnings	The code is running in the simulator without any error and warnings [Marks 10]	The code is running but with some warnings or errors. [Marks 5]	The code is written but not running due to errors [Marks 0]	
4. Procedure	All the instructions are written with proper procedure [Marks 20]	Some steps are missing [Marks 10]	steps are totally missing [Marks 0]	
5. OUTPUT	Proper output of the code written in assembly [Marks 20]	Some of the outputs are missing [Marks 10]	No or wrong output [Marks 0]	
6. Conclusion	Conclusion about the lab is shown and written [Marks 20]	Conclusion about the lab is partially shown [Marks 10]	Conclusion about the lab is not shown[Marks0]	
7. Cheating			Any kind of cheating will lead to 0 Marks	
<p style="text-align: center;">Total Marks Obtained: _____</p> <p style="text-align: center;">Instructor Signature: _____</p>				

Task 1:

Load a value from memory and add 10 to it. Store the result back in memory and show the result on console. (*hint: use MIPS instructions lw and sw*)

Code:

```
Task1.asm x Task2.asm x Task3.asm x Task4.asm x
1  .data
2      iword : .word 4
3  .text
4  .globl main
5  main:
6
7      lw $t1, iword
8      addi $t1, $t1, 10
9      sw $t1, iword
10
11     #output
12     li $v0, 1
13     lw $a0, iword
14     syscall
15
16     program_end:
17
18     #exit the process
19     li $v0, 10
20     syscall
21
```

Output:

```
Console
14
```

Task 2:

Load a value from memory and double it. Store the result back in memory also show on the console. (use *sll*, *sw* and *lw*)

Code:

```
Task1.asm x Task2.asm x Task3.asm x Task4.asm x
1  .data
2      iword : .word 4
3  .text
4  .globl main
5  main:
6
7      lw $t1, iword
8      sll $t1, $t1, 1
9      sw $t1, iword
10
11     #output
12     li $v0, 1
13     lw $a0, iword
14     #move $a0, $t2
15     syscall
16
17     program_end:
18
19     #exit the process
20     li $v0, 10
21     syscall
22
```

Output:

```
Console
8
```

Task 3:

Load an address of a label into a register and jump to that address and perform addition in that address. (use jr(jump register))

Code:

```
Task1.asm x Task2.asm x Task3.asm x Task4.asm x
1      .data
2          iword : .word 4
3      .text
4      .globl main
5      main:
6          la $t1, addition
7          li $t2, 8
8          jr $t1 #jump to addition
9
10     addition:
11         add $t3, $t2, $t2
12         #output
13         li $v0, 1
14         move $a0, $t3
15         syscall
16
17     program_end:
18
19         #exit the process
20         li $v0, 10
21         syscall
22
```

Output:

```
Console
16
```

Task 4:

Write assembly program to find the Fibonacci series.

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, ...

Users will be asked to enter a number, for instance 9. Then assembly will print the first 9 numbers of Fibonacci series.

Code:

```
Task1.asm Task2.asm Task3.asm Task4.asm
1  .data
2      #iword : .word 4
3      msg : .asciiz "Enter number of terms \n"
4      empty_space : .asciiz " "
5  .text
6  .globl main
7  main:
8
9      #output
10     li $v0, 4
11     la $a0, msg
12     #move $a0, $t2
13     syscall
14
15     #output
16     li $v0, 5
17     syscall
18     move $t0, $v0
19
20     li $t1, 0
21     li $t2, 1
22     li $t4, 2
23
24     j first_term
25
26 first_term:
27
28     #output
29     li $v0, 1
30     move $a0, $t1
31     syscall
```

```
Task1.asm x Task2.asm x Task3.asm x Task4.asm x
34      #output
35      li $v0,4
36      la $a0, empty_space
37      syscall
38
39
40      beq $t0, 1, program_end
41      bgt $t0, 2 second_term
42
43
44  second_term:
45
46      #output
47      li $v0,1
48      move $a0, $t2
49      syscall
50
51      #output
52      li $v0,4
53      la $a0, empty_space
54      syscall
55
56      beq $t0, 2, program_end
```

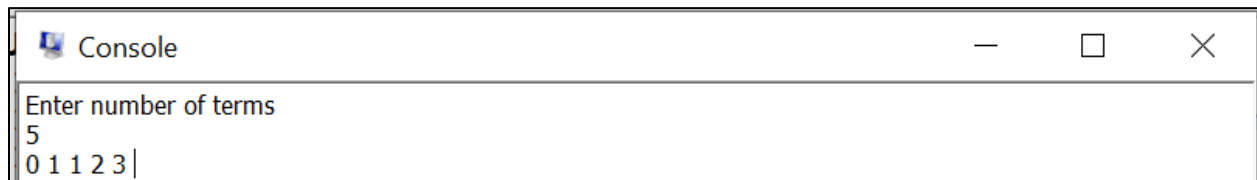
```
57      bgt $t0, 2 loop1
58
59
60  loop1:
61      #t4 is iterating variable
62      addi $t4, 1
63
64      add $t3, $t1, $t2
65
66      #output
67      li $v0,1
68      move $a0, $t3
69      syscall
```

```

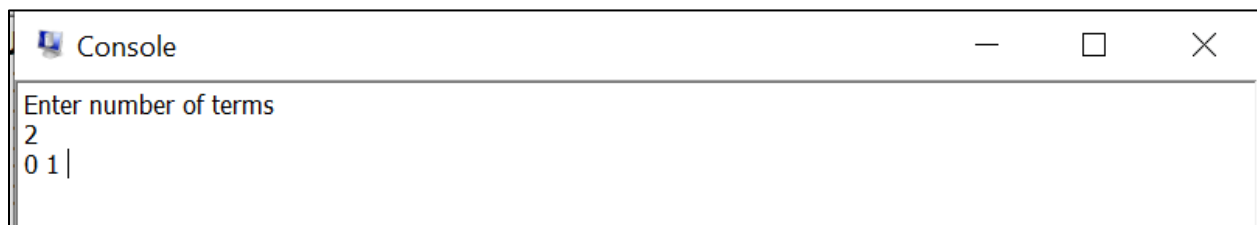
69      syscall
70
71      move $t1, $t2
72      move $t2, $t3
73
74      #output
75      li $v0, 4
76      la $a0, empty_space
77      syscall
78
79      beq $t0, $t4, program_end
80      j loop1
81
82  program_end:
83
84      #exit the process
85      li $v0, 10
86      syscall

```

Output:



A screenshot of a console window titled "Console". It displays the prompt "Enter number of terms" followed by the input "5". Below the input, the output "0 1 1 2 3|" is shown, indicating the sequence of numbers generated by the program.



A screenshot of a console window titled "Console". It displays the prompt "Enter number of terms" followed by the input "2". Below the input, the output "0 1|" is shown, indicating the sequence of numbers generated by the program.

Conclusion:

In this lab, I learned about the storing and loading instructions(lw and sw) in MIPS Assembly.