

## Assignment 1

EG 212/Computer architecture- MIPS processor design

Student Name: Sarthak Raj

Student ID: IMT2023569

Student Name: Harjot Singh

Student ID: IMT2023064

Student Name: Archit Jaju

Student ID: IMT2023128

26 February 2024

### Abstract

In this section, we will examine the inner workings of the MIPS assembly code. focusing on the design and implementation of a non-pipelined MIPS processor. Our objective is to understand the intricacies of processor architecture by translating C programs into MIPS assembly language and executing them on the MARS assembler platform.

## 1 Introduction

In this report, we have chosen to perform the calculation of Calculating the nth fibonacci sequence entry for a given number n. Calculating the frequency of a given element in an array of given length.and The sum of first N natural numbers by providing instructions in mars code. With the help of the assembler, it should read the assembly code and generate instruction code of 32bit for non-pipelined MIPS processor for to decode them, and then execute them.

## 2 C codes

### 2.1 (1)Fibonacci sequence

```
#include <iostream>
using namespace std;

int main() {
    int a = 0, b = 1, c, n;

    cout << "Enter the number of terms: ";
    cin >> n;

    cout << "Fibonacci Series:" << std::endl;

    if (n >= 1) {
        cout << a << " ";
    }
    if (n >= 2) {
        cout << b << " ";
    }

    for (int i = 3; i <= n; ++i) {
        c = a + b;
        std::cout << c << " ";
        a = b;
        b = c;
    }
}
```

## 2.2 (2) Sum of n natural numbers

```
#include <iostream>
using namespace std;
int main() {
    int n;
    cout << "Enter a positive integer: ";
    cin >> n;

    if (n < 0) {
        cout << "Please enter a positive integer." << std::endl;
        return 1;
    }

    int sum = 0;
    for (int i = 1; i <= n; ++i) {
        sum += i;
    }

    cout << "The sum of the first " << n << " natural numbers is: " << sum <<
    endl;
    return 0;
}
```

## 2.3 (3) Total number of occurrences in the list

```
#include <iostream>
using namespace std;
#include <vector>
int countOccurrences(const std::vector<int>& lst, int num) {
    int count = 0;
    for (int element : lst) {
        if (element == num) {
            count++;
        }
    }
    return count;
}
int main() {
    std::vector<int> lst;
    int n, num;

    cout << "Enter the number of integers in the list: ";
    cin >> n;
    cout << "Enter " << n << " integers separated by spaces: ";
    for (int i = 0; i < n; ++i) {
        int val;
        cin >> val;
        lst.push_back(val);
    }

    cout << "Enter the integer you want to find: ";
    cin >> num;
    int count = countOccurrences(lst, num);
    cout << "Total number of occurrences: " << count << endl;
    return 0;
}
```

## 3 MIPS instructions

### 3.1 Fibonacci sequence

```

Edit Execute
fib.asm*
17
18 main:
19     lui $s0, 0x1001
20
21     li $v0, 5           # System call code for reading integer
22     syscall
23     sw $v0, 0($s0)      # Store the user input in location input_num
24
25 loop:
26
27     lw $s1, 16($s0)
28     lw $s2, 0($s0)
29     beq $s1, $s2, exit  # if iterator==input_num -> branch to exit
30
31     lw $s3, 4($s0)
32     lw $s4, 8($s0)
33     lw $s5, 12($s0)
34     sw $s4, 4($s0)      # a = b
35     sw $s5, 8($s0)      # b = c
36     add $s6, $s4, $s5
37     sw $s6, 12($s0)     # c = a+b
38     addi $s1, $s1, 1    # iterator++
39     sw $s1, 16($s0)
40
41     j loop              # go to loop again
42 exit:
43     lw $s7, 4($s0)
44     sw $s7, 20($s0)     # storing the returned value(a) in ans
45
46     li $v0, 1           # System call code for printing integer
47     add $a0, $s7, $zero
48     syscall
49
50     li $v0, 10          # System call to exit
51     syscall

```

### 3.2 Sum of n natural numbers

```

Edit Execute
sum of n.asm*
9 main:
10     # load base address
11     lui $s0, 0x1001
12
13     # read integer input
14     li $v0, 5
15     syscall
16     sw $v0, n           # store the user input in location n
17
18 loop:
19     # load n and i
20     lw $t0, n
21     lw $t1, i
22
23     # compute n+1
24     addi $t0, $t0, 1
25
26     # check if i == n+1, if yes then exit loop
27     beq $t1, $t0, exit
28
29     # load sum, compute sum + i, and store sum
30     lw $t3, sum
31     add $t3, $t3, $t1
32     sw $t3, sum
33
34     # increment i and store it
35     addi $t1, $t1, 1
36     sw $t1, i
37
38     # jump to loop
39     j loop
40
41 exit:
42     # load sum, print integer, and exit

```

### 3.3 Total number of occurrences in the list

```

Edit Execute
sum of n.asm find in the list.asm*
.data
array: .word 1, 2, 3, 4, 5, 6, 7, 8, 9, 5, 6, 7, 8, 5, 5
array_size: .word 16
input_prompt: .asciiz "Enter a number: "
output_prompt: .asciiz "The number of occurrences of the input in the array is: "

.text
.globl main
main:
    la $t0, array
    li $t1, 0 # counter for number of occurrences
    li $t2, 0 # counter for array index
    # read input
    li $v0, 5
    syscall
    move $t3, $v0
    # load address of array_size
    la $t5, array_size
    lw $t6, 0($t5)

loop:
    bge $t2, $t6, print_result
    lw $t4, 0($t0)
    beq $t3, $t4, inc_counter
    addiu $t0, $t0, 4
    addiu $t2, $t2, 1
    j loop
inc_counter:
    addiu $t1, $t1, 1
    addiu $t0, $t0, 4
    addiu $t2, $t2, 1
    j loop

print_result:

```

Line: 26 Column: 11 ☐ Show Line Numbers

## 4 MIPS Result

### 4.1 Fibonacci sequence

Edit Execute

Text Segment

| Bkpt | Address | Code       | Basic              | Source  |
|------|---------|------------|--------------------|---|
|      | 4194304 | 0x3c101001 | lui \$16,4097      | 19: lui \$s0, 0x1001  |
|      | 4194308 | 0x24020005 | addiu \$2,\$0,5    | 21: li \$v0, 5 # System call code for reading integer               |
|      | 4194312 | 0x0000000c | syscall            | 22: syscall   |
|      | 4194316 | 0xae020000 | sw \$2,0(\$16)     | 23: sw \$v0, 0(\$s0) # Store the user input in location input num   |
|      | 4194320 | 0x8e110010 | lw \$17,16(\$16)   | 27: lw \$s1, 16(\$s0)   |
|      | 4194324 | 0x8e120000 | lw \$18,0(\$16)    | 28: lw \$s2, 0(\$s0)  |
|      | 4194328 | 0x1232000a | beq \$17,\$18,10   | 29: beq \$s1, \$s2, exit # if iterator==input num -> branch to exit |
|      | 4194332 | 0x8e130004 | lw \$19,4(\$16)    | 31: lw \$s3, 4(\$s0)  |
|      | 4194336 | 0x8e140008 | lw \$20,8(\$16)    | 32: lw \$s4, 8(\$s0)  |
|      | 4194340 | 0x8e15000c | lw \$21,12(\$16)   | 33: lw \$s5, 12(\$s0)   |
|      | 4194344 | 0xae140004 | sw \$20,4(\$16)    | 34: sw \$s4, 4(\$s0) # a = b  |
|      | 4194348 | 0xae150008 | sw \$21,8(\$16)    | 35: sw \$s5, 8(\$s0) # b = c  |
|      | 4194352 | 0x0295b020 | add \$22,\$20,\$21 | 36: add \$s6, \$s4, \$s5  |
|      | 4194356 | 0xae16000c | sw \$22,12(\$16)   | 37: sw \$s6, 12(\$s0) # c = a+b                                     |
|      | 4194360 | 0x22310001 | addi \$17,\$17,1   | 39: addi \$s1, \$s1, 1 # iterator++                                 |
|      | 4194364 | 0xae110010 | sw \$17,16(\$16)   | 40: sw \$s1, 16(\$s0)   |
|      | 4194368 | 0x08100004 | j 4194320          | 42: j loop # go to loop again                                       |
|      | 4194372 | 0x8e170004 | lw \$23,4(\$16)    | 45: lw \$s7, 4(\$s0)  |
|      | 4194376 | 0xae170014 | sw \$23,20(\$16)   | 46: sw \$s7, 20(\$s0) # storing the returned value(a) in ans        |
|      | 4194380 | 0x24020001 | addiu \$2,\$0,1    | 48: li \$v0, 1 # System call code for printing integer              |
|      | 4194384 | 0x02e02020 | add \$4,\$23,\$0   | 49: add \$a0, \$s7, \$zero  |
|      | 4194388 | 0x0000000c | syscall            | 50: syscall   |
|      | 4194392 | 0x2402000a | addiu \$2,\$0,10   | 52: li \$v0, 10 # System call to exit                               |
|      | 4194396 | 0x0000000c | syscall            | 53: syscall   |

Data Segment

| Address            | Value (+0) | Value (+4) | Value (+8) | Value (+12) | Value (+16) | Value (+20) | Value (+24) | Value (+28) |
|--------------------|------------|------------|------------|-------------|-------------|-------------|-------------|-------------|
| 0x10010000 (.data) |            |            |            |             |             |             |             |             |

☐ Hexadecimal Addresses
 ☐ Hexadecimal Values
 ☐ ASCII

Mars Messages

Run I/O

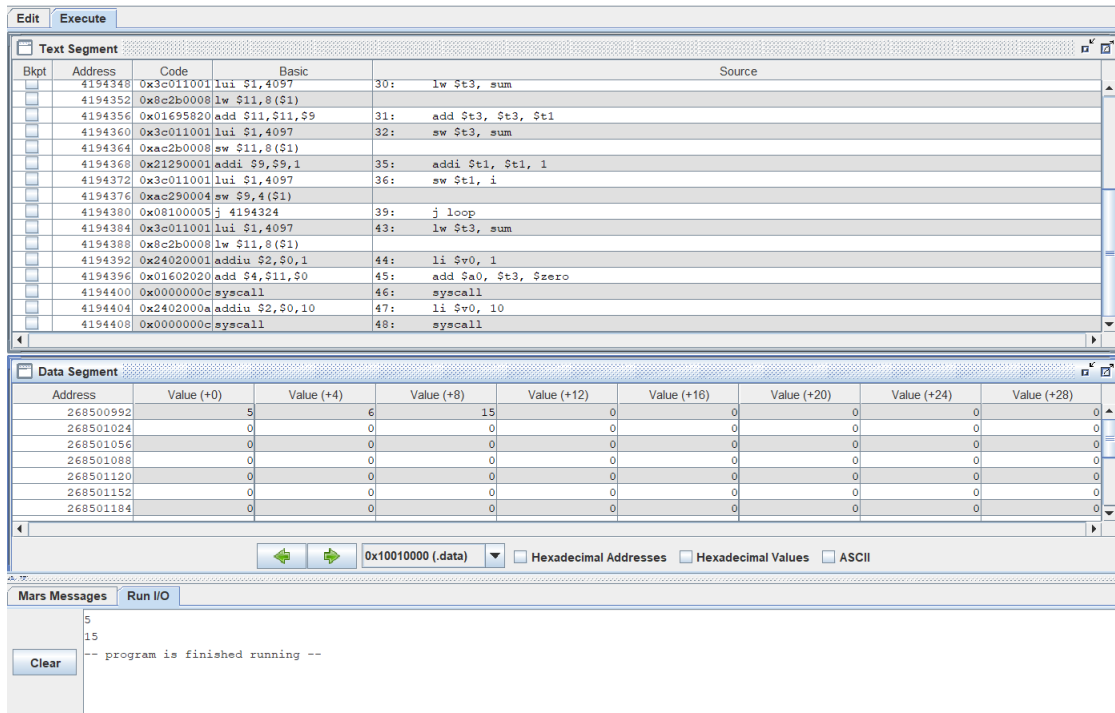
Clear

```

10
55
-- program is finished running --

```

## 4.2 Sum of n natural numbers



**Text Segment**

| Bkpt | Address | Code       | Basic             | Source                     |
|------|---------|------------|-------------------|----------------------------|
|      | 4194348 | 0x3c011001 | lui \$1,4097      | 30: lw \$t3, sum           |
|      | 4194352 | 0x8c2b0008 | lw \$11,8(\$1)    |                            |
|      | 4194356 | 0x01695820 | add \$11,\$11,\$9 | 31: add \$t3, \$t3, \$t1   |
|      | 4194360 | 0x3c011001 | lui \$1,4097      | 32: sw \$t3, sum           |
|      | 4194364 | 0x8c2b0008 | sw \$11,8(\$1)    |                            |
|      | 4194368 | 0x21290001 | addi \$9,\$9,1    | 35: addi \$t1, \$t1, 1     |
|      | 4194372 | 0x3c011001 | lui \$1,4097      | 36: sw \$t1, i             |
|      | 4194376 | 0x8c290004 | sw \$9,4(\$1)     |                            |
|      | 4194380 | 0x08100005 | j 4194324         | 39: j loop                 |
|      | 4194384 | 0x3c011001 | lui \$1,4097      | 43: lw \$t3, sum           |
|      | 4194388 | 0x8c2b0008 | lw \$11,8(\$1)    |                            |
|      | 4194392 | 0x24020001 | addiu \$2,\$0,1   | 44: li \$v0, 1             |
|      | 4194396 | 0x01602020 | add \$4,\$11,\$0  | 45: add \$a0, \$t3, \$zero |
|      | 4194400 | 0x0000000c | syscall           | 46: syscall                |
|      | 4194404 | 0x2402000a | addiu \$2,\$0,10  | 47: li \$v0, 10            |
|      | 4194408 | 0x0000000c | syscall           | 48: syscall                |

**Data Segment**

| Address   | Value (+0) | Value (+4) | Value (+8) | Value (+12) | Value (+16) | Value (+20) | Value (+24) | Value (+28) |
|-----------|------------|------------|------------|-------------|-------------|-------------|-------------|-------------|
| 268500992 | 5          | 6          | 15         | 0           | 0           | 0           | 0           | 0           |
| 268501024 | 0          | 0          | 0          | 0           | 0           | 0           | 0           | 0           |
| 268501056 | 0          | 0          | 0          | 0           | 0           | 0           | 0           | 0           |
| 268501088 | 0          | 0          | 0          | 0           | 0           | 0           | 0           | 0           |
| 268501120 | 0          | 0          | 0          | 0           | 0           | 0           | 0           | 0           |
| 268501152 | 0          | 0          | 0          | 0           | 0           | 0           | 0           | 0           |
| 268501184 | 0          | 0          | 0          | 0           | 0           | 0           | 0           | 0           |

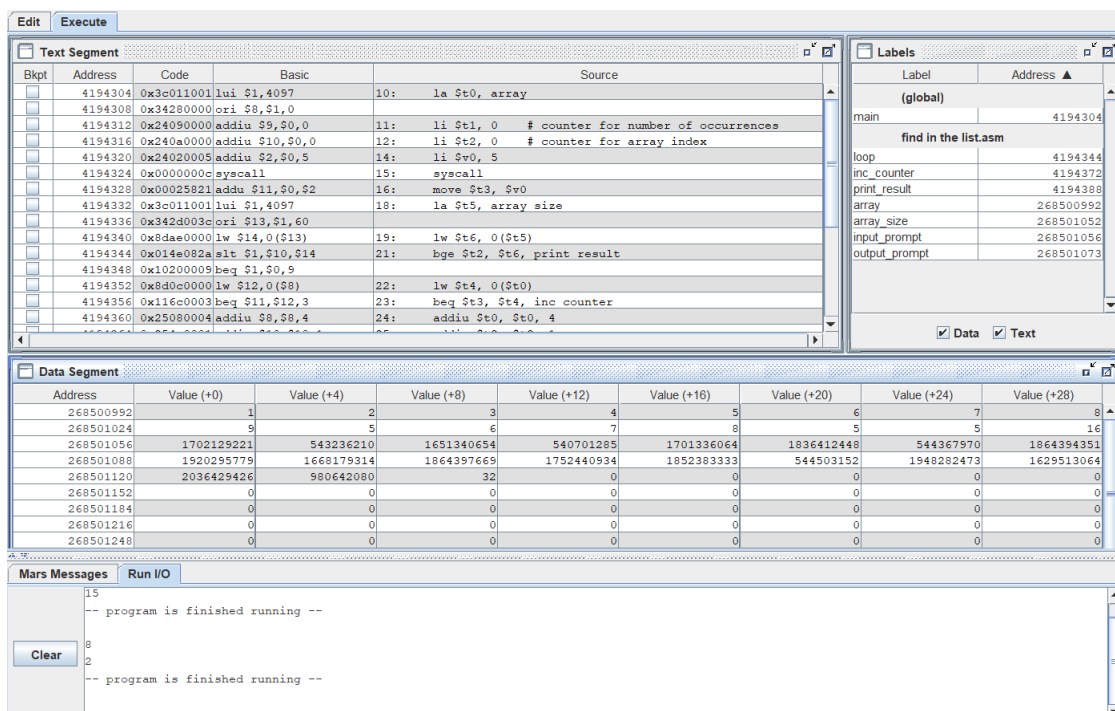
**Mars Messages**

```

5
15
-- program is finished running --

```

## 4.3 Total number of occurrences in the list



**Text Segment**

| Bkpt | Address | Code       | Basic              | Source   |
|------|---------|------------|--------------------|--|
|      | 4194304 | 0x3c011001 | lui \$1,4097       | 10: la \$t0, array                                 |
|      | 4194308 | 0x34280000 | ori \$8,\$1,0      |  |
|      | 4194312 | 0x24090000 | addiu \$9,\$0,0    | 11: li \$t1, 0 # counter for number of occurrences |
|      | 4194316 | 0x240a0000 | addiu \$10,\$0,0   | 12: li \$t2, 0 # counter for array index           |
|      | 4194320 | 0x24020005 | addiu \$2,\$0,5    | 14: li \$v0, 5                                     |
|      | 4194324 | 0x0000000c | syscall            | 15: syscall  |
|      | 4194328 | 0x00025821 | addu \$11,\$0,\$2  | 16: move \$t3, \$v0                                |
|      | 4194332 | 0x3c011001 | lui \$1,4097       | 18: la \$t5, array size                            |
|      | 4194336 | 0x342d003c | ori \$13,\$1,60    |  |
|      | 4194340 | 0x8dae0000 | lw \$14,0(\$13)    | 19: lw \$t6, 0(\$t5)                               |
|      | 4194344 | 0x014e082a | sllt \$1,\$10,\$14 | 21: bge \$t2, \$t6, print result                   |
|      | 4194348 | 0x10200009 | beq \$1,\$0,9      |  |
|      | 4194352 | 0x8d0c0000 | lw \$12,0(\$8)     | 22: lw \$t4, 0(\$t0)                               |
|      | 4194356 | 0x116c0003 | beq \$11,\$12,3    | 23: beq \$t3, \$t4, inc counter                    |
|      | 4194360 | 0x25080004 | addiu \$8,\$8,4    | 24: addiu \$t0, \$t0, 4                            |

**Data Segment**

| Address   | Value (+0) | Value (+4) | Value (+8) | Value (+12) | Value (+16) | Value (+20) | Value (+24) | Value (+28) |
|-----------|------------|------------|------------|-------------|-------------|-------------|-------------|-------------|
| 268500992 | 1          | 2          | 3          | 4           | 5           | 6           | 7           | 8           |
| 268501024 | 9          | 5          | 6          | 7           | 8           | 5           | 5           | 16          |
| 268501056 | 1702129221 | 543236210  | 1651340654 | 540701285   | 1701336064  | 1836412448  | 544367970   | 1864394351  |
| 268501088 | 1920295779 | 1668179314 | 1864397669 | 1752440934  | 1852383333  | 544503152   | 1948282473  | 1629513064  |
| 268501120 | 2036429426 | 980642080  | 32         | 0           | 0           | 0           | 0           | 0           |
| 268501152 | 0          | 0          | 0          | 0           | 0           | 0           | 0           | 0           |
| 268501184 | 0          | 0          | 0          | 0           | 0           | 0           | 0           | 0           |
| 268501216 | 0          | 0          | 0          | 0           | 0           | 0           | 0           | 0           |
| 268501248 | 0          | 0          | 0          | 0           | 0           | 0           | 0           | 0           |

**Labels**

| Label                | Address   |
|----------------------|-----------|
| (global)             |           |
| main                 | 4194304   |
| find in the list.asm |           |
| loop                 | 4194344   |
| inc_counter          | 4194372   |
| print_result         | 4194388   |
| array                | 268500992 |
| array_size           | 268501052 |
| input_prompt         | 268501056 |
| output_prompt        | 268501073 |

**Mars Messages**

```

15
-- program is finished running --
8
2
-- program is finished running --

```

## 5 Conclusion

Through this assignment, we delved into computer architecture and assembly language programming, focusing on developing a plague checker program. By translating C code into MIPS assembly and crafting a non-pipelined MIPS processor, we gained insights into computing systems for epidemiological analysis. Adhering to guidelines,

we included essential instructions for plague checking algorithms, facilitating testing with the MARS assembler. Designing the MIPS processor provided hands-on understanding of architecture, memory management, and instruction execution for epidemiological computation. This experience broadened our knowledge and equipped us with practical skills for real-world challenges in public health. We are confident these lessons will support our future work in epidemiology and computational health