 **Northwestern Polytechnic University**

**EE488 - Computer Architecture**

**Homework Assignment #3**

**Due day: 10/31/2021**

**Instruction:**

1. **Push the answer sheet to GitHub in word file**
2. **Overdue homework submission could not be accepted.**
3. **Takes academic honesty and integrity seriously (Zero Tolerance of Cheating & Plagiarism)**
4. Implement a program (MIPS Assembly) which multiplies a user input by 10 using only bit shift operations and addition. Check to see if your program is correct by using the *mult* and *mflo* operators. Your program should include a proper and useful prompt for input, and print the results in a meaningful manner.

.data

prompt: .asciiz "\nEnter an integer: "

prompt1: .asciiz "\nResult by multiplying 10 is : "

.text

li $v0,4

la $a0,prompt #it will print prompt

syscall

li $v0,5

syscall #ask user input

move $t1,$v0 #save a to t1

sll $s0,$t1,3

sll $s1,$t1,1

add $s2,$s0,$s1

li $v0,4

la $a0,prompt1 #it will print prompt

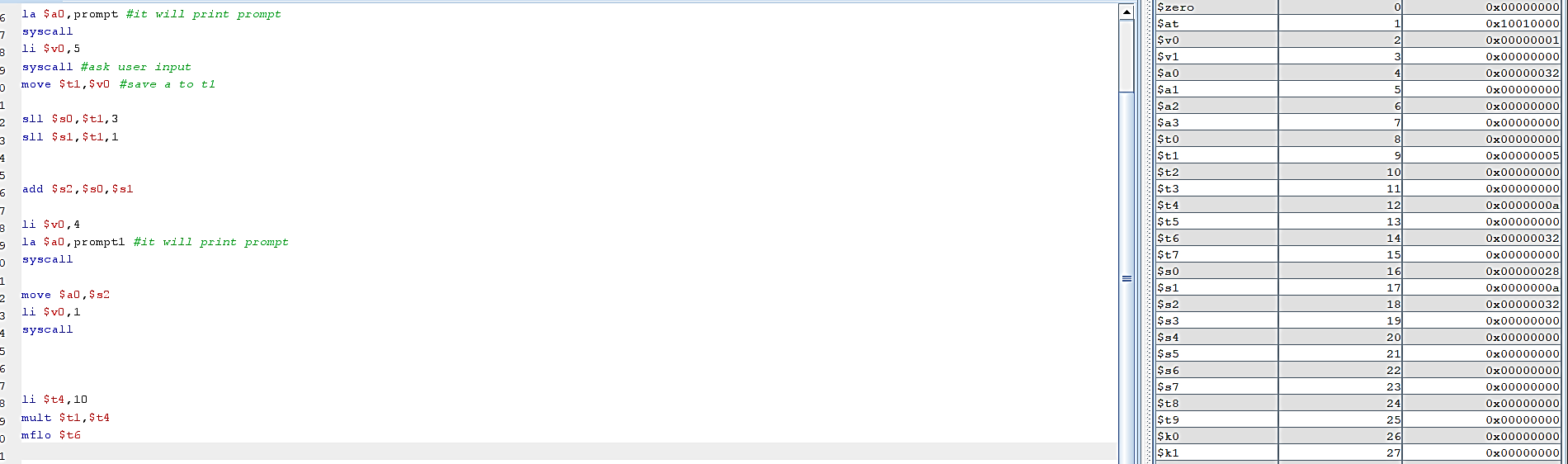
syscall

move $a0,$s2

li $v0,1

syscall

After verifying with mult and mflo instructions -



1. Write programs (MIPS Assembly) to evaluate the following expressions. The user should enter the variables, and the program should print back an answer. Prompt the user for all variables in the expression, and print the results in a meaningful manner. **The results should be as accurate as possible.**
2. Write a program (MIPS Assembly) to retrieve two numbers from a user, and swap those numbers using only the *XOR* operation. You should not use a temporary variable to store the numbers while swapping them. Your program should include a proper and useful prompt for input, and print the results in a meaningful manner.

.data

prompt:.asciiz"Enter first number ="

secnum:.asciiz"Enter second number ="

resultText:.asciiz"Value of first number after swapping ="

result:.asciiz "\nValue of second number after swapping ="

.text

li $v0, 4

la $a0, prompt #loading the string to print into the argument to enable printing

syscall

#the next block of code is for reading the first number provided by the user

li $v0, 5

syscall

move $t0, $v0

#asking the user to provide the second number

li $v0, 4

la $a0, secnum #loading the string into the argument to enable printing

syscall

#reading the second number to be provided to the user

li $v0, 5 #command to read the number provided by the user

syscall #executing the command for reading an integer

move $t1, $v0

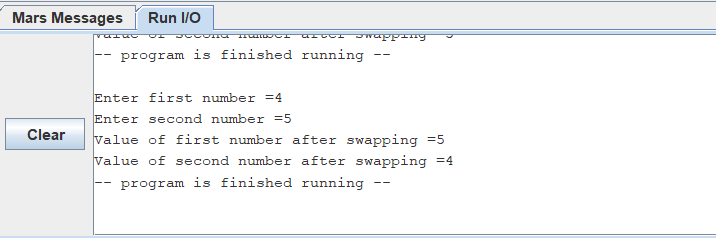
main:

add $s0, $0 , $t0

add $s1 , $0 , $t1

move $a0, $s0

move $a1, $s1

 jal swap #swap function call

jr $ra # return from main

swap:

# working of swap

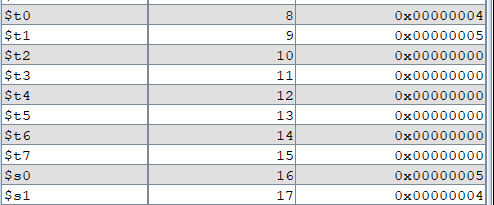
xor $s0,$s0,$s1

xor $s1,$s0,$s1

xor $s0,$s0,$s1

addi $v0, $zero, 4 # Print result string

la $a0, resultText

 syscall

addi $v0, $zero, 1 # Print result

move $a0, $s0

syscall

addi $v0, $zero, 4 # Print result string

la $a0, result

syscall

addi $v0, $zero, 1 # Print result

move $a0, $s1

syscall

#Exit program

addi $v0, $zero, 10

syscall

jr $ra #return from swap

1. Using only *sll* and *srl*, implement a program to check if a user input value is even or odd. The result should print out *0* if the number is even or *1* if the number is odd. Your program (MIPS Assembly) should include a proper and useful prompt for input, and print the results in a meaningful manner.

.data

prompt:.asciiz "Enter a num : "

result: .asciiz "\nResult"

promptE: .asciiz "\n0\nNum is even"

promptO: .asciiz "\n1\nNum is odd"

.text

main:

#prompt for input

li $v0,4

la $a0,prompt

syscall

#number input

li $v0,5

syscall

move $t0,$v0

srl $s0,$t0,1#Shift right by 1 bit

sll $t1,$s0,1#shift left by 1 bit

beq $t0,$t1, Even#if pos then jump to even

bne $t0,$t1, Odd #if pos then jump to odd

#op prog

li $v0,10

la $a0,result

syscall

Odd:

li $v0,4

la $a0,promptO #for odd

syscall

li $v0,10

la $a0,result

syscall

Even:

li $v0,4

la $a0, promptE #for even

syscall

li $v0,10

la $a0,result

syscall

1. Implement a program (MIPS Assembly) to prompt the user for two numbers, the first being any number and the second a prime number. Return to the user a *0* if the second number is a prime factor for the first one, otherwise any number if it is not. For example, if the user enters *60* and *5*, the program returns *0*. If the user enters *62* and *5*, the program returns *2*.

.data

prompt:.asciiz "enter first number = "

secnum:.asciiz"enter second number =" # This is the number we'll test to see

valid:.asciiz"valid input"

invalid:.asciiz"invalid input" # if it's prime.

# It will be located by the label `num`

.text

main:

li $v0, 4

la $a0, prompt

syscall

li $v0, 5

syscall

move $t0, $v0

#asking the user to provide the second number

li $v0, 4

la $a0,secnum

syscall

#reading the second number to be provided to the user

li $v0, 5

syscall

move $t1, $v0

move $a0, $t1

jal is\_prime # Send the number to the procedure!

add $a0, $zero, $v0 # Send the result as an argument to...

li $v0, 1 # ...print integer onscreen

syscall

## Tells if a number is prime

# $a0 The number to check if it's prime

# $v0 1 if the number is prime, 0 if it's not

is\_prime:

addi $t0, $zero, 2 # int x = 2

is\_prime\_test:

slt $t1, $t0, $a0 # if (x > num)

bne $t1, $zero, is\_prime\_loop

div $t1, $t0 #Divides $t0/$t1

mfhi $t3

bnez $t3, exit# It's prime!

li $v0 10

syscall

is\_prime\_loop: # else

div $a0, $t0

mfhi $t3 # c = (num % x)

slti $t4, $t3, 1

beq $t4, $zero, is\_prime\_loop\_continue # if (c == 0)

addi $v0, $zero, 4 # Print result string

la $a0, invalid

syscall

addi $v0, $zero, 1 # Print result

move $a0, $s1

# its not a prime

li $v0 10

syscall

is\_prime\_loop\_continue:

addi $t0, $t0, 1 # x++

j is\_prime\_test # continue the loop .data

# continue the loop

exit:

add $v0, $zero, $zero # its not a prime

jr $ra

li $v0 10

syscall

random:

addi $v0, $zero, 2 # It's prime!

jr $ra

li $v0 10

syscall