Practical 3

**Aim**: Working with MIPS assembly language using Mars simulator.

1. **. Write an ALP to perform 8-bit and 16 bit multiplication.**

# Program/Procedure (8-bit multiplication):

**Code:-**

**.data**

msg1: .asciiz " First Number:" msg2: .asciiz " Number:" msg3: .asciiz "Multiplication is:”

**.text**

la $a0, msg1

li $v0, 4 syscall

li $v0, 5 syscall

move $t0, $v0

la $a0, msg2

li $v0, 4 syscall

li $v0, 5

syscall

move $t1, $v0

la $a0, msg3

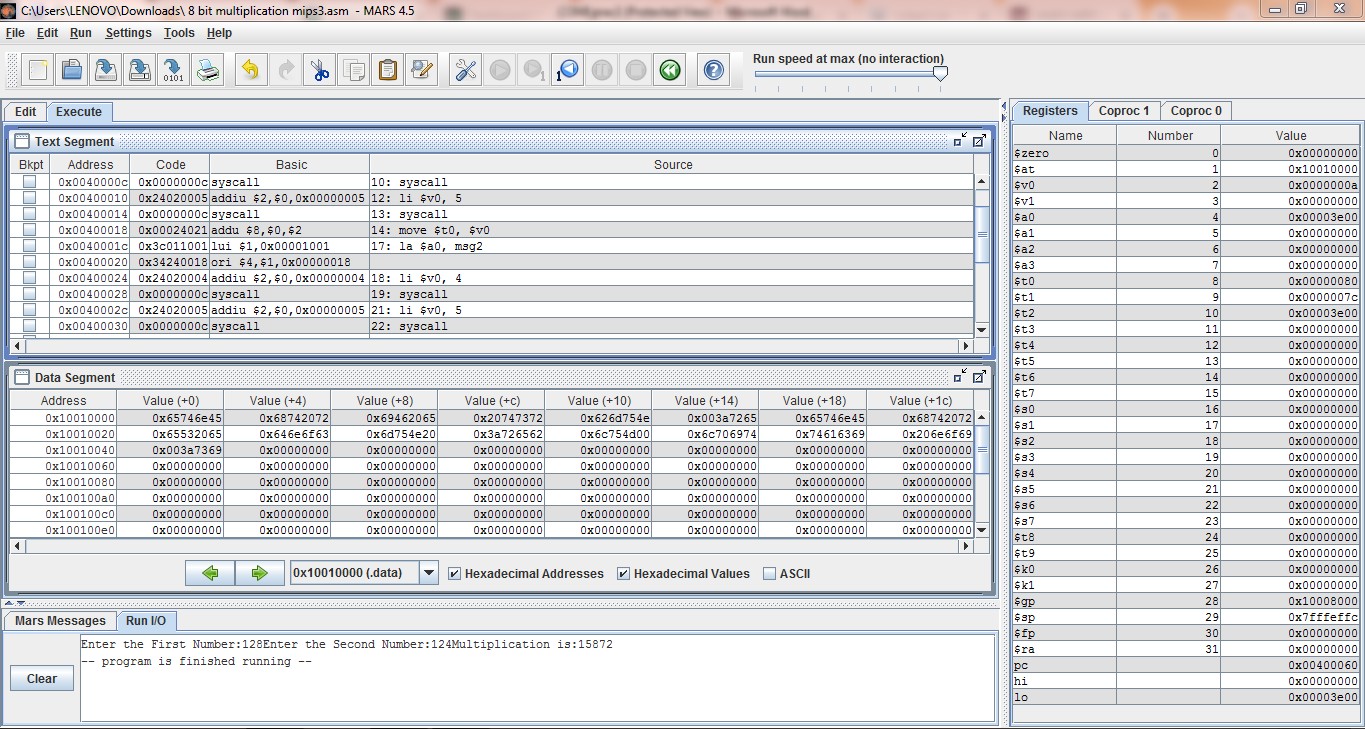
li $v0, 4 syscall

mul $t2,$t0,$t1

move $a0,$t2 li $v0,1 syscall

# Results:

li $v0, 10 syscall



# Program/Procedure (16 bit multiplication):

**Code:-**

**.data**

msg1: .asciiz "Enter the First Number:" msg2: .asciiz "Enter the Second Number:" msg3: .asciiz "Multiplication is:"

**.text**

la $a0, msg1

li $v0, 4 syscall

li $v0, 5 syscall

move $t0, $v0

la $a0, msg2

li $v0, 4 syscall

li $v0, 5 syscall

move $t1, $v0

la $a0, msg3

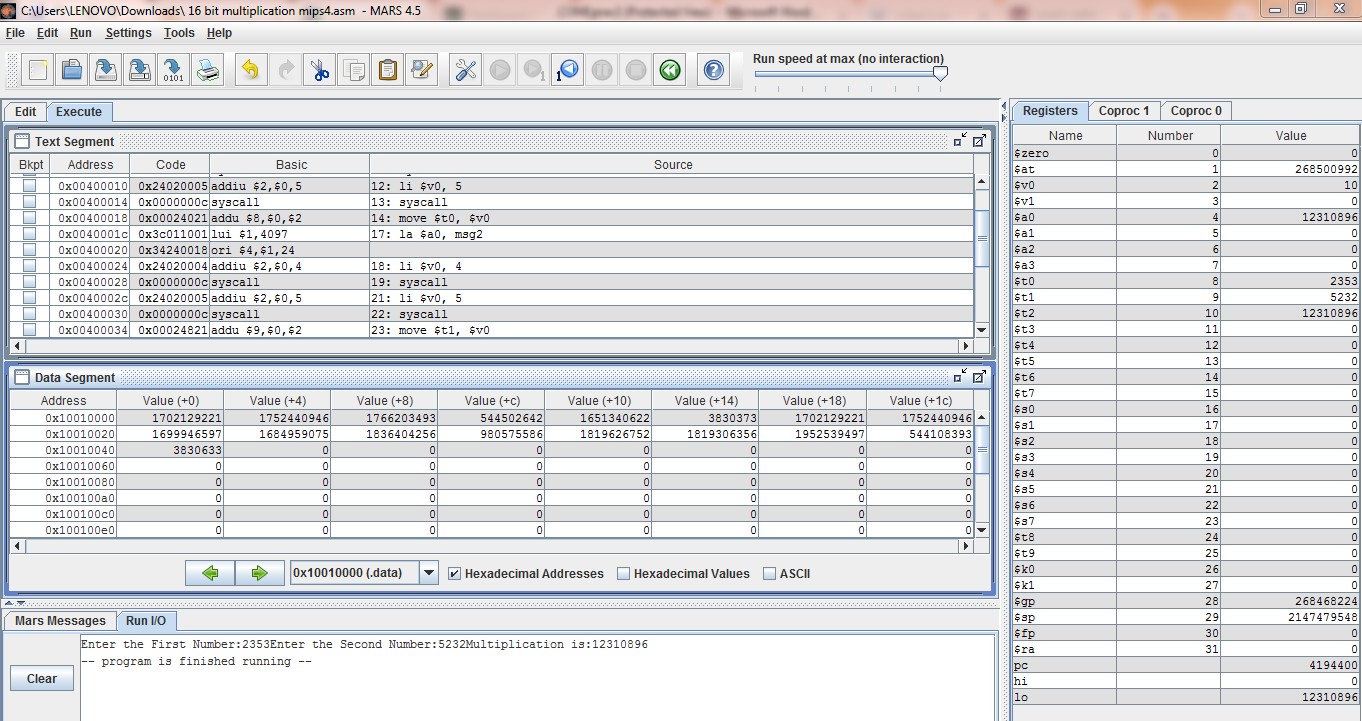
li $v0, 4 syscall

mul $t2,$t0,$t1

move $a0,$t2 li $v0,1 syscall

li $v0, 10 syscall

# Results:



1. Write an ALP to perform 8-bit and 16-bit division.

# Program/Procedure (8 bit Division):

**Code:-**

.data

m: .word 4

n: .word 2

.text

lw $t0, m

lw $t1, n

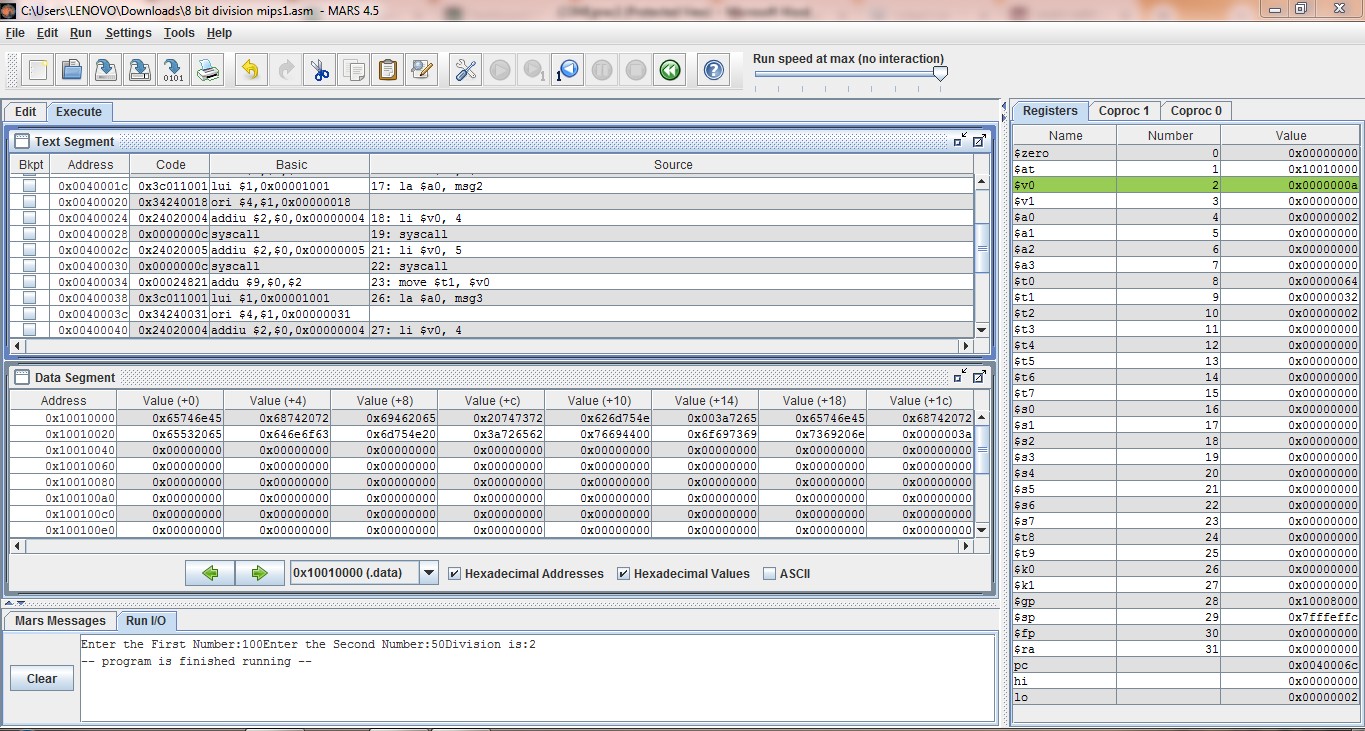
div $t2, $t0, $t1

li $v0, 1

move $a0, $t2

syscall

# Results:



* + **Program/Procedure (16 bit Division):**

.data

msg1: .asciiz "Enter the First Number:" msg2: .asciiz "Enter the Second Number:" msg3: .asciiz "Division is:"

.text

la $a0, msg1

li $v0, 4 syscall li $v0, 5 syscall

move $t0, $v0

la $a0, msg2

li $v0, 4 syscall li $v0, 5 syscall

move $t1, $v0

la $a0, msg3

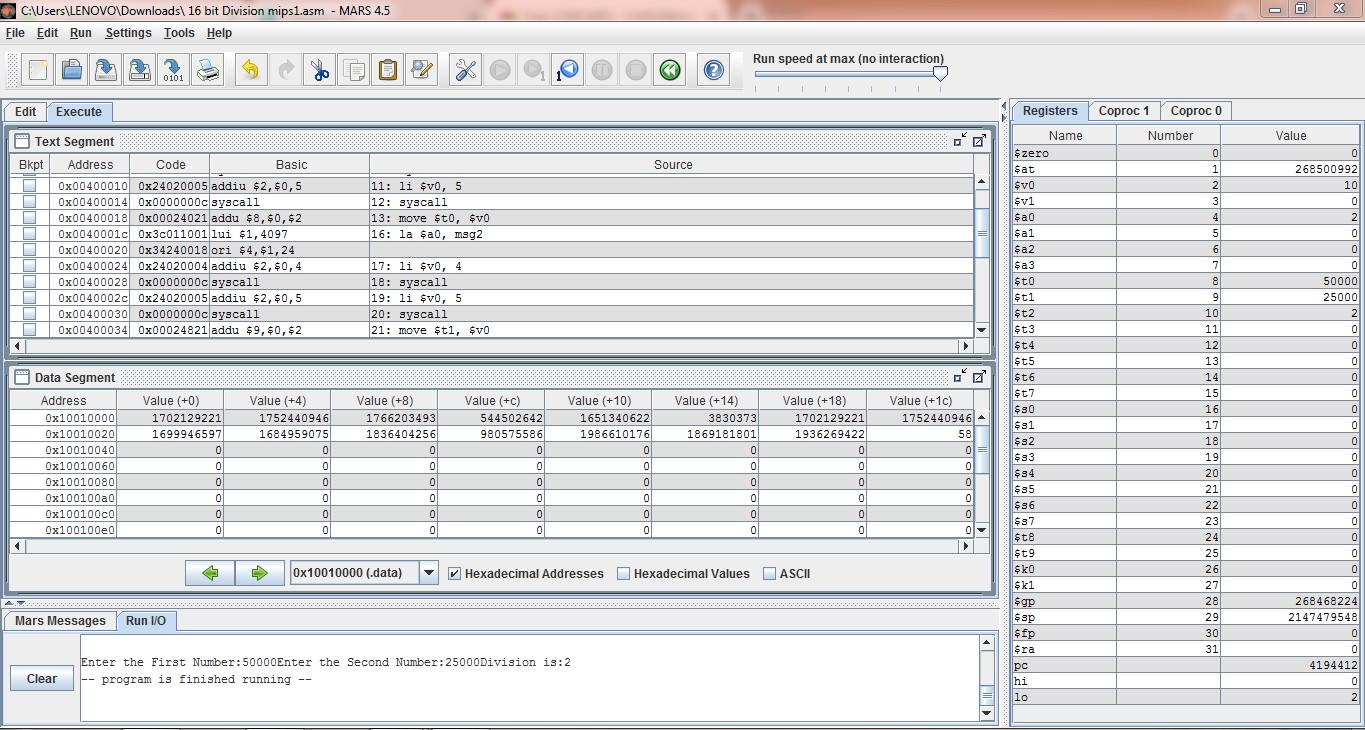
li $v0, 4

syscall

div $t2,$t0,$t1 move $a0,$t2 li $v0,1 syscall

li $v0, 10 syscall

# Results:



**Conclusion:**

From this practical, We know the MIPS programming concept like how to take input and how to multiply them and divide them and also know about the register files and store register file and how to use them.

**Sign: Date:**