1. $0: 0x00000000

$t1: 0x00000061

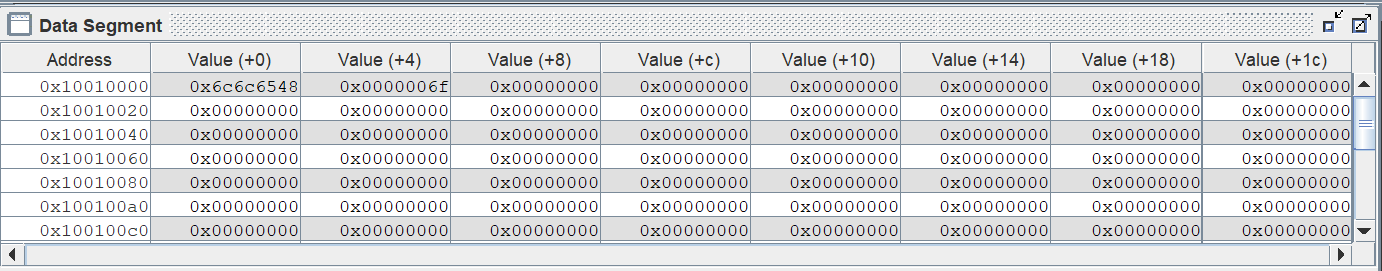


1. H: 0x48

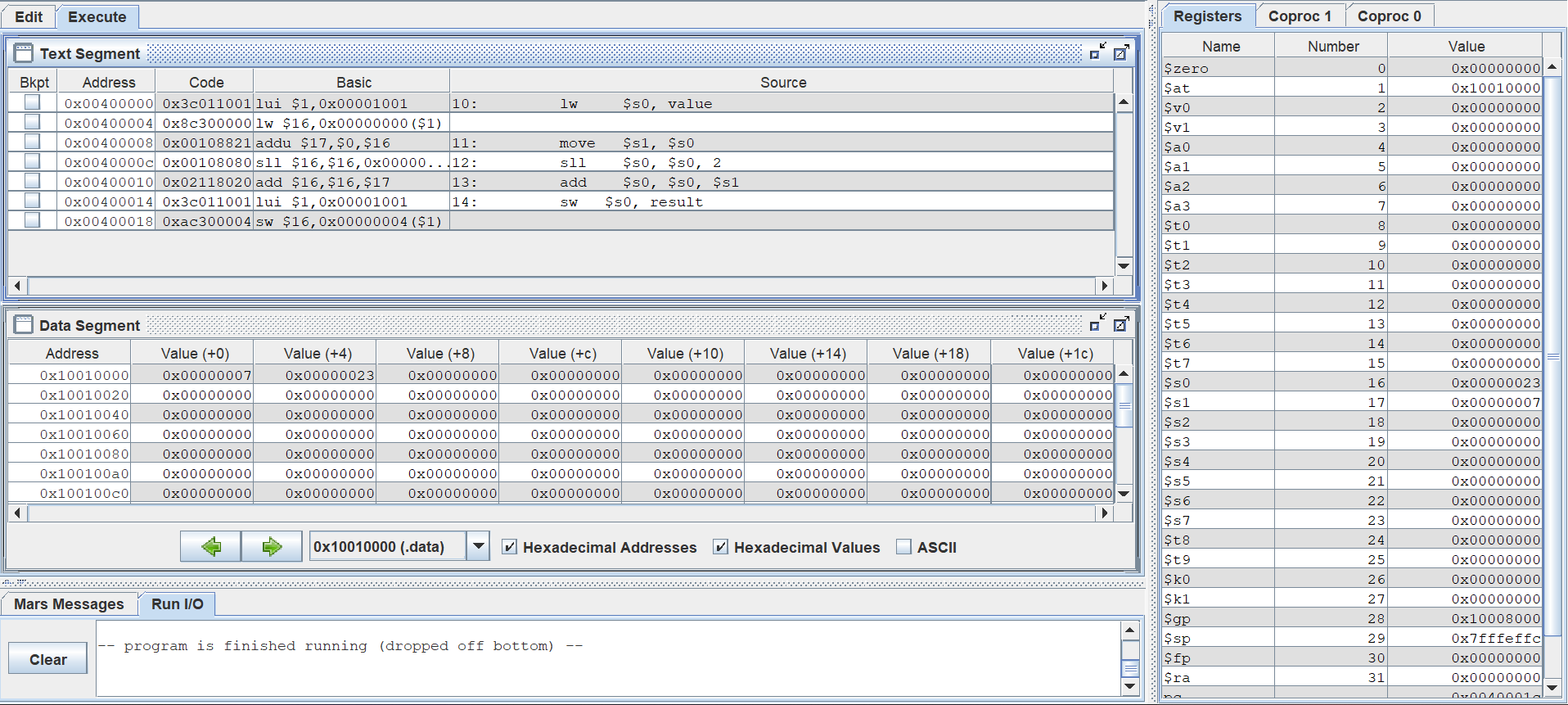
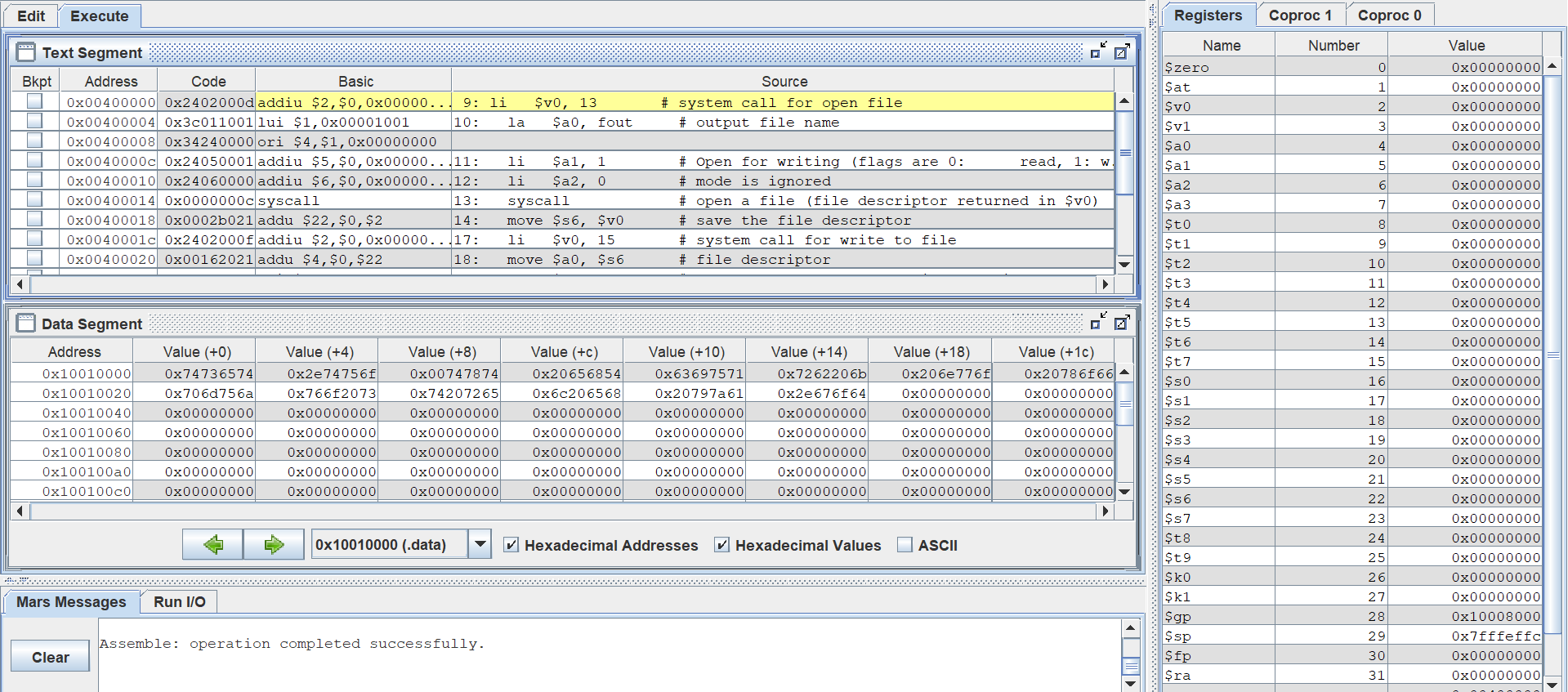
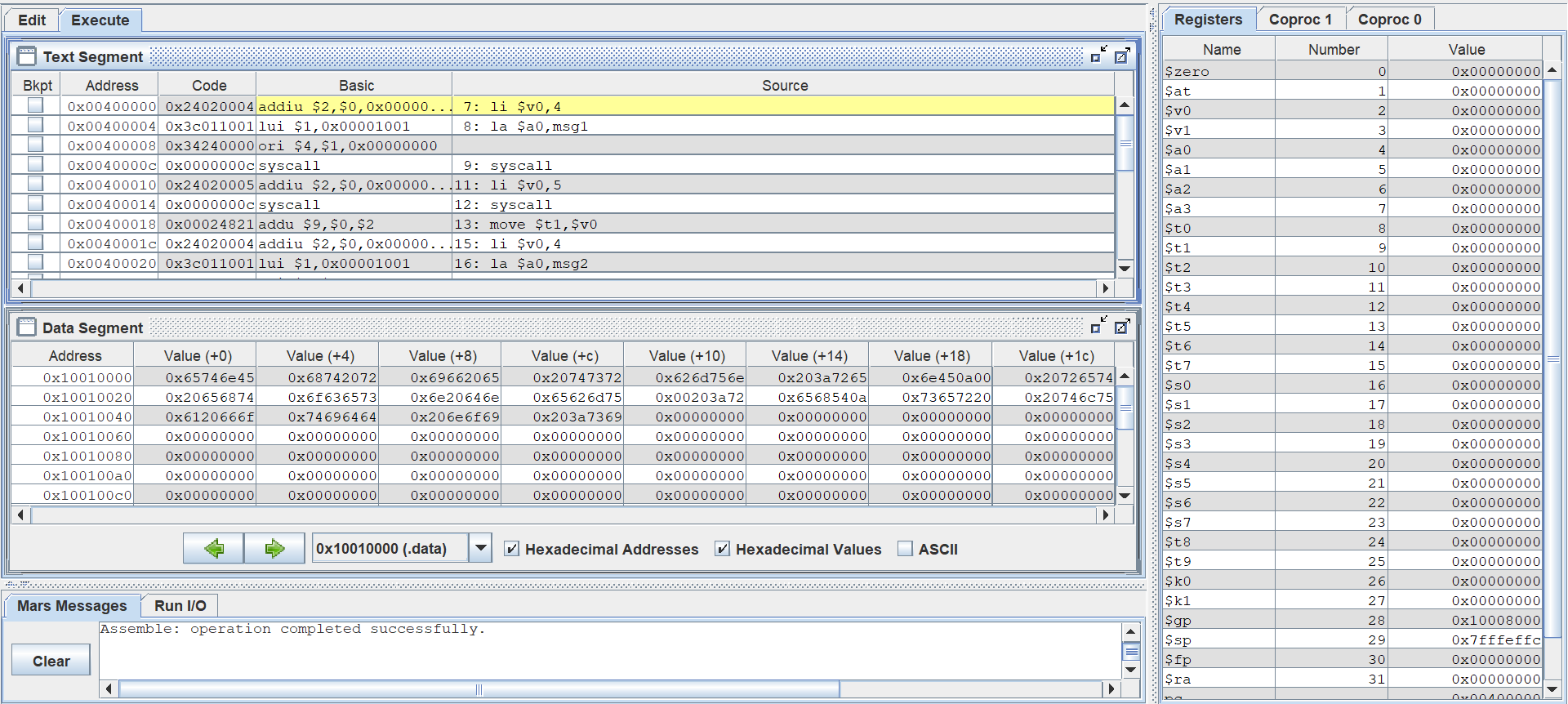
l:0x6c

e:0x65

o:0x6f



H e l l stored in value 0 and ‘o’ stored in value(+4) of 0x10010000

1. 
2. 
3. 

Q4)

1)#1\_4\_5\_10

.data

msg1: .asciiz "Enter the number: "

msg2: .asciiz "\nThe number you entered is : "

.text

li $v0,4

la $a0,msg1

syscall

li $v0,5

syscall

move $t1,$v0

li $v0,4

la $a0,msg2

syscall

li $v0,1

move $a0,$t1

syscall

li $v0,10

syscall

2)#4\_8\_11\_12

.data

buffer: .space 20

str1: .asciiz "Enter Name(max 20 chars): "

str2: .asciiz "Enter(Y/N) :\n"

str3: .asciiz "\nYour String is:\n"

str4: .asciiz "Your answer is:\n"

.text

li $v0,4

la $a0,str1

syscall

li $v0,8 #take in input

la $a0, buffer

li $a1, 20

syscall

move $t0, $a0

li $v0,4

la $a0,str2

syscall

li $v0 ,12

syscall

move $t1 , $v0

li $v0,4

la $a0,str3

syscall

li $v0 ,4

move $a0 , $t0

syscall

li $v0,4

la $a0,str4

syscall

li $v0 ,11

move $a0 , $t1

syscall

Q 5)

Code1:

.data

msg1: .asciiz "Provide two integers: "

addnum: .asciiz "\nSum of numbers: "

subnum: .asciiz "\nDifference of numbers: "

multinum: .asciiz "\nMultiplication of numbers: "

.text

main:

# print welcome message

li $v0, 4 #4-->sysytem call code for printing string

la $a0, msg1

syscall #system call

# read integers

li $v0, 5

syscall

move $t0, $v0

li $v0, 5

syscall

move $t1, $v0

# add and display

li $v0, 4

la $a0, addnum

syscall

add $a0, $t0, $t1

li $v0, 1

syscall

# subtract and display

li $v0, 4

la $a0, subnum

syscall

sub $a0, $t0, $t1

li $v0, 1

syscall

# multiply and display

mul $t3, $t0,$t1 #t3 stores the lower 32 but

li $v0,4

la, $a0, multinum # overflow

syscall

li $v0,1

add $a0, $zero,$t3

syscall

# exit

li $v0, 10

syscall

Code 2:

# check if integer is even or odd

.data

msg1: .asciiz "enter integer:"

even: .asciiz "\nNumber is even"

odd: .asciiz "\nNumber is odd"

.text

main:

li $v0, 4

la $a0, msg1

syscall

li $v0, 5 # reading the number

syscall

andi $t0, $v0, 1 # bitwise and to find odd even

li $v0, 4

beq $t0, $zero, valid #if equal even\_label

la $a0, odd # if false

j end\_label

valid:

la $a0, even #if true

end\_label:

syscall

li $v0, 10

syscall

Code 3:

.data

msg: .asciiz "Enter a number"

answer: .asciiz "\nFactorial is: "

.text

# welcome message

li $v0, 4

la $a0, msg

syscall

# read integer

li $v0, 5

syscall

# print the integer

move $a0, $v0

li $v0, 1

syscall

jal calculate\_factorial

move $a1, $v0

li $v0, 4

la $a0, answer

syscall

move $a0, $a1

li $v0, 1

syscall

li $v0, 10

syscall

calculate\_factorial:

addi $sp, $sp-4

sw $ra, ($sp)

li $v0, 1

multiply:

beq $a0, $zero, return

mul $v0, $v0, $a0

addi $a0, $a0, -1

j multiply

return:

lw $ra, ($sp)

jr $ra

Code 4:

# find min

.data

num\_a: .asciiz "\nEnter the number A: "

num\_b: .asciiz "\nEnter the number B: "

stat1: .asciiz "\nA is max"

stat2: .asciiz "\nB is max"

.text

main:

li $v0, 4

la $a0, num\_a

syscall

li $v0, 5 # reading the number

syscall

addi $t0,$v0,0

li $v0, 4

la $a0, num\_b

syscall

li $v0, 5 # reading the number

syscall

addi $t1,$v0,0

ble $t1,$t0,valid #if equal even\_label

li $v0, 4

la $a0, stat2 # if false

syscall

j end\_label

valid:

li $v0, 4

la $a0, stat1 #if true

syscall

end\_label:

Code 5:

# divide numbers

.data

msg: .asciiz "Enter two integer numbers: "

quotient: .asciiz "\nQuotient: "

remainder: .asciiz "\nRemainder: "

decimal: .asciiz "\nQuotient in decimal: "

.text

main:

# welcome message

li $v0, 4

la $a0, msg

syscall

# read two integers

li $v0, 5

syscall

move $t0, $v0

li $v0, 5

syscall

move $t1, $v0

# divide integers (quotient, remainder)

div $t0, $t1

li $v0, 4

la $a0, quotient

syscall

li $v0, 1

mflo $a0

syscall

li $v0, 4

la $a0, remainder

syscall

li $v0, 1

mfhi $a0

syscall

# divide integers (FP answer)

mtc1 $t0, $f12

cvt.s.w $f3, $f12

mtc1 $t1, $f12

cvt.s.w $f4, $f12

div.s $f12, $f3, $f4

li $v0, 4

la $a0, decimal

syscall

li $v0, 2

syscall

# exit

li $v0, 10

syscall