Lab Report 10.1

## Assignment 1

**Code:**

.eqv SEVENSEG\_LEFT 0xFFFF0011

.eqv SEVENSEG\_RIGHT 0xFFFF0010

.data

NUMS: .word 0x3F, 0x06, 0x5B, 0x4F, 0x66, 0x6D, 0x7D, 0x07, 0x7F, 0x6F

.text

main: li $a0, 6

li $a1, SEVENSEG\_LEFT

jal draw

li $a0, 4

li $a1, SEVENSEG\_RIGHT

jal draw

j exit

draw: la $t0, NUMS

sll $t1, $a0, 2 # i \* 4

add $t0, $t0, $t1 # Address(NUMS[i])

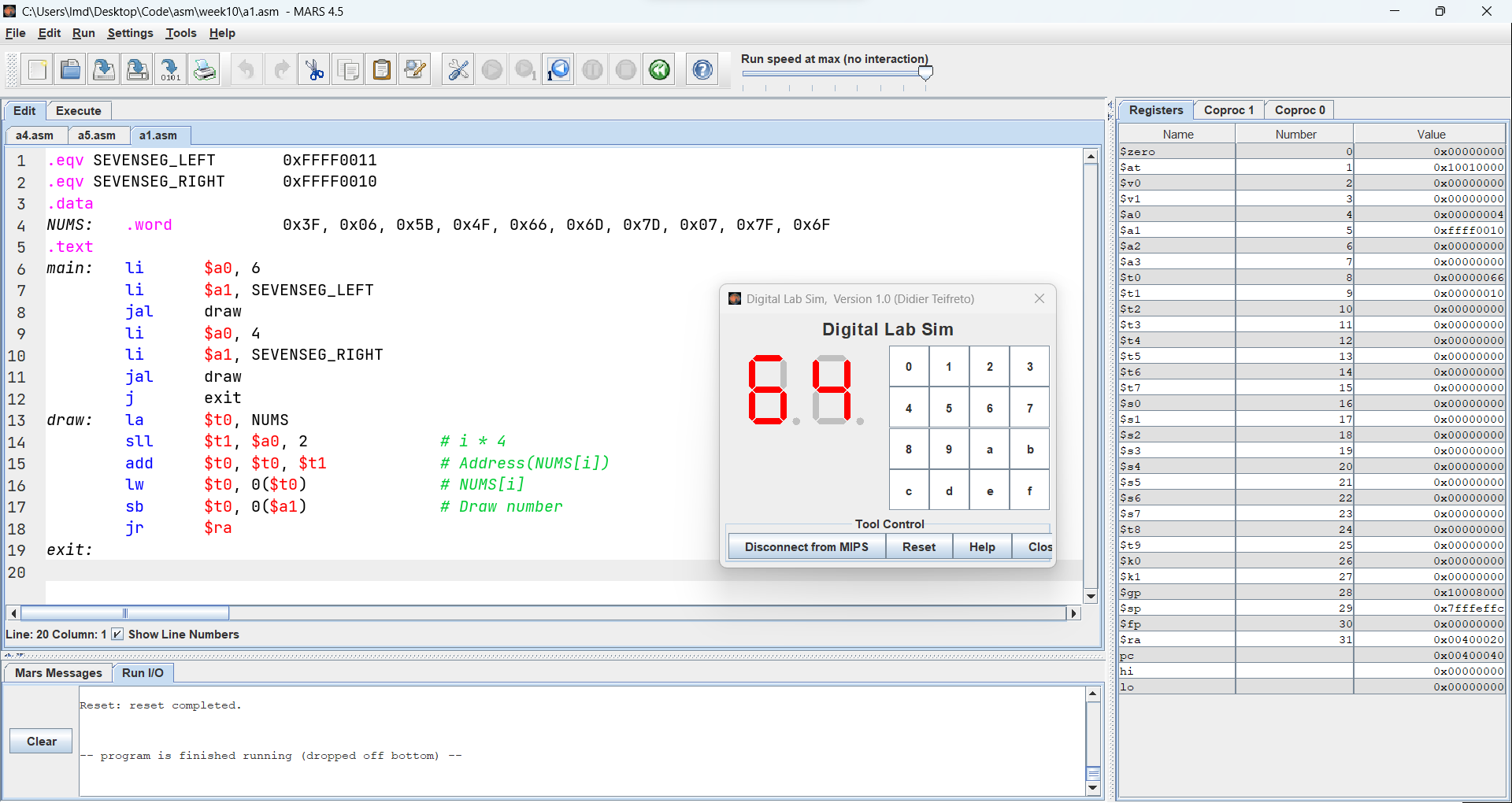
lw $t0, 0($t0) # NUMS[i]

sb $t0, 0($a1) # Draw number

jr $ra

exit:

**Results:**



## Assignment 2

**Code:**

.eqv SEVENSEG\_LEFT 0xFFFF0011

.eqv SEVENSEG\_RIGHT 0xFFFF0010

.data

NUMS: .word 0x3F, 0x06, 0x5B, 0x4F, 0x66, 0x6D, 0x7D, 0x07, 0x7F, 0x6F

.text

main: li $s0, 10

li $v0, 5

syscall

div $v0, $s0

mfhi $a0

li $a1, SEVENSEG\_RIGHT

jal draw

mflo $v0

div $v0, $s0

mfhi $a0

li $a1, SEVENSEG\_LEFT

jal draw

j exit

draw: la $t0, NUMS # Address(NUMS[0])

sll $t1, $a0, 2 # i \* 4

add $t0, $t0, $t1 # Address(NUMS[i])

lw $t0, 0($t0) # NUMS[i]

sb $t0, 0($a1) # Draw number

jr $ra

exit:

**Results:**

**Graphical user interface, application

Description automatically generated**

## Assignment 3

**Code:**

.eqv SEVENSEG\_LEFT 0xFFFF0011

.eqv SEVENSEG\_RIGHT 0xFFFF0010

.data

NUMS: .word 0x3F, 0x06, 0x5B, 0x4F, 0x66, 0x6D, 0x7D, 0x07, 0x7F, 0x6F

.text

main: li $s0, 10

li $v0, 12

syscall

div $v0, $s0

mfhi $a0

li $a1, SEVENSEG\_RIGHT

jal draw

mflo $v0

div $v0, $s0

mfhi $a0

li $a1, SEVENSEG\_LEFT

jal draw

j exit

draw: la $t0, NUMS

sll $t1, $a0, 2 # i \* 4

add $t0, $t0, $t1 # Address(NUMS[i])

lw $t0, 0($t0) # NUMS[i]

sb $t0, 0($a1) # Draw number

jr $ra

exit:

**Results:**

**A screenshot of a computer

Description automatically generated with medium confidence**

## Assignment 4

**Code:**

.eqv SCREEN 0x10010000

.eqv COLOUR0 0x00FFFFFF

.eqv COLOUR1 0x000000FF

.text

init: li $k0, SCREEN

li $s0, 0 # i = 0

li $s1, 0 # j = 0

li $s2, 0 # Current colour

li $s3, 0 # Current square

loop1: beq $s0, 8, exitL # Break if i == 8

loop2: beq $s1, 8, incL1 # Break if j == 8

sll $t0, $s0, 3 # t0 = 8 \* i

add $t0, $t0, $s1 # t0 = 8 \* i + j

color: # Get colour

add $t1, $t0, $s0 # t1 = 8 \* i + j + i

li $t2, 2

div $t1, $t2

mfhi $t2

beq $t2, 0, setC0

beq $t2, 1, setC1

setC0: li $s2, COLOUR0

j square

setC1: li $s2, COLOUR1

j square

square: # Get square

sll $t0, $t0, 2 # Offset bytes

add $s3, $k0, $t0 # Update square

sw $s2, 0($s3)

incL2: add $s1, $s1, 1

j loop2

incL1: add $s0, $s0, 1 # i++

li $s1, 0 # Reset j

j loop1

exitL:

**Results:**

**Graphical user interface, application

Description automatically generated**

## Assignment 5

**Code:**

.eqv SCREEN 0x10010000

.eqv RED 0x00FF0000

.eqv GREEN 0x0000FF00

.text

init: li $k0, SCREEN

li $s2, 0 # Current color

li $s3, 0 # Current square

li $s4, 64 # Number of rows

li $s5, 64 # Number of cols

li $t6, 6 # i1

li $t7, 4 # j1

li $t8, 32 # i2

li $t9, 60 # j2

add $s0, $zero, $t6 # i = i1

add $s1, $zero, $t7 # j = j1

loop1: bgt $s0, $t8, exitL # Break if i == i2

loop2: bgt $s1, $t9, incL1 # Break if j == j2

mul $t0, $s0, $s5 # t0 = numRols \* i

add $t0, $t0, $s1 # t0 = numRols \* i + j

color: # Get colour

beq $s0, $t6, border # Diff color for border

beq $s0, $t8, border # Diff color for border

beq $s1, $t7, border # Diff color for border

beq $s1, $t9, border # Diff color for border

li $s2, GREEN

j square

border: li $s2, RED

j square

square: # Get square

sll $t0, $t0, 2 # Offset bytes

add $s3, $k0, $t0 # Update square

sw $s2, 0($s3)

incL2: add $s1, $s1, 1

j loop2

incL1: add $s0, $s0, 1 # i++

add $s1, $zero, $t7 # Reset j

j loop1

exitL:

**Results:**

**Graphical user interface

Description automatically generated**

**Comments:**

* The code can draw rectangle from (i1, j1) to (i2, j2). No validations are implemented.
* The code assumes i1 < i2 && j1 < j2.