**Laboratory Exercise 10**

Họ và Tên : Vũ Nguyễn Khánh

MSSV : 20176795

**Assignment 1:**

.eqv SEVENSEG\_LEFT 0xFFFF0011 # Dia chi cua den led 7 doan trai.

# Bit 0 = doan a;

# Bit 1 = doan b; ...

# Bit 7 = dau .

.eqv SEVENSEG\_RIGHT 0xFFFF0010 # Dia chi cua den led 7 doan phai

.text

main:

li $a0, 0x6F # set value for segments

jal SHOW\_7SEG\_LEFT # show

li $a0, 0x6D # set value for segments

jal SHOW\_7SEG\_RIGHT # show

exit: li $v0, 10

syscall

endmain:

#---------------------------------------------------------------

# Function SHOW\_7SEG\_LEFT : turn on/off the 7seg

# param[in] $a0 value to shown

# remark $t0 changed

#---------------------------------------------------------------

SHOW\_7SEG\_LEFT: li $t0, SEVENSEG\_LEFT # assign port's address

sb $a0, 0($t0) # assign new value

jr $ra

#---------------------------------------------------------------

# Function SHOW\_7SEG\_RIGHT : turn on/off the 7seg

# param[in] $a0 value to shown

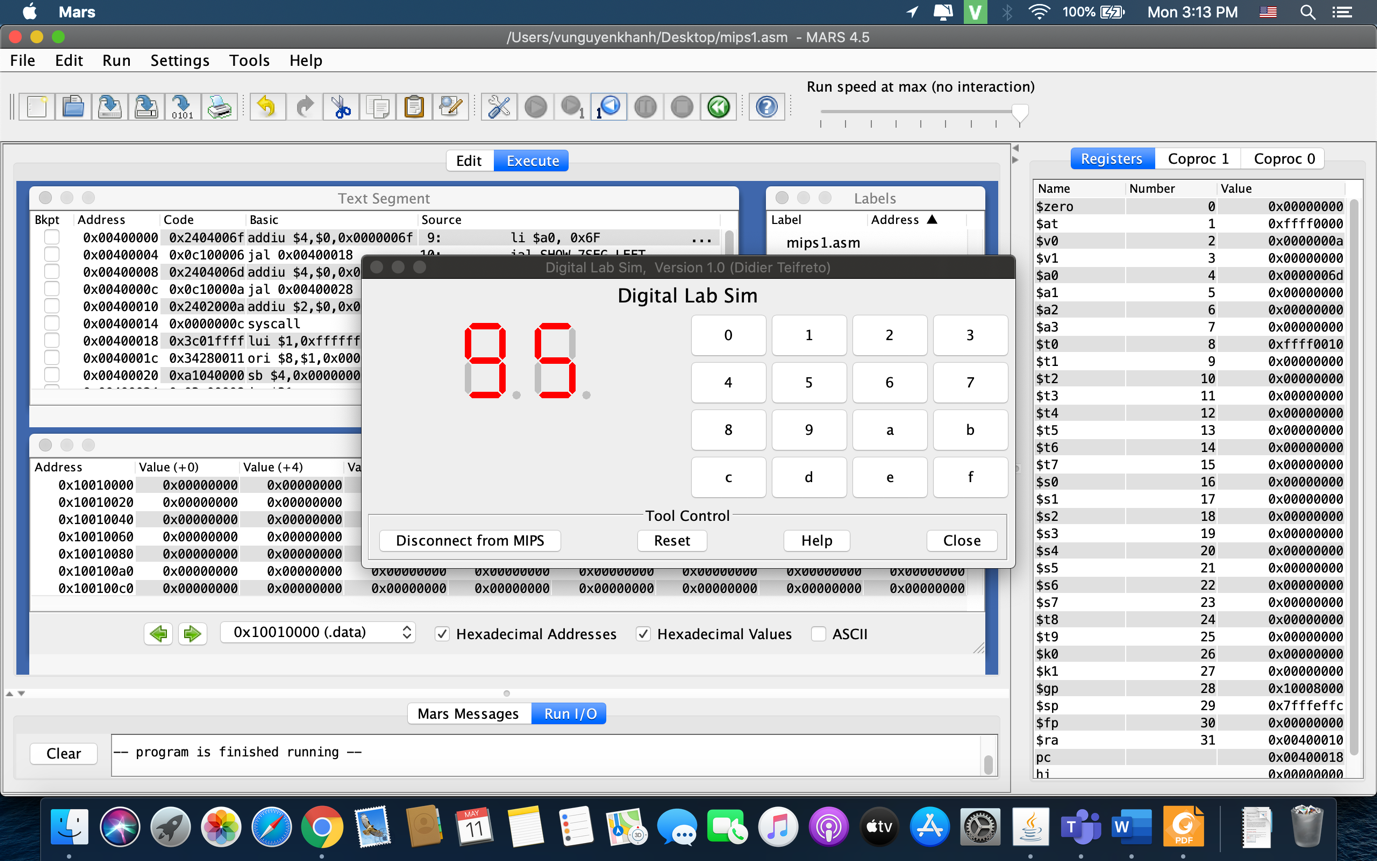
# remark $t0 changed

#---------------------------------------------------------------

SHOW\_7SEG\_RIGHT: li $t0, SEVENSEG\_RIGHT # assign port's address

sb $a0, 0($t0) # assign new value

jr $ra



**9** : các bit sáng là a, b, c, d, g, f ứng với các bit 0, 1, 2, 3, 5, 6 có giá trị bằng 1

01101111 = 6F

**5** : các bit sáng là a, c, d, g, f ứng với các bit 0, 2, 3, 5, 6 có giá trị bằng 1

01101101 = 6D

**Assignment 2:**

.eqv MONITOR\_SCREEN 0x10010000

.eqv RED 0x00FF0000

.text

li $k0, MONITOR\_SCREEN

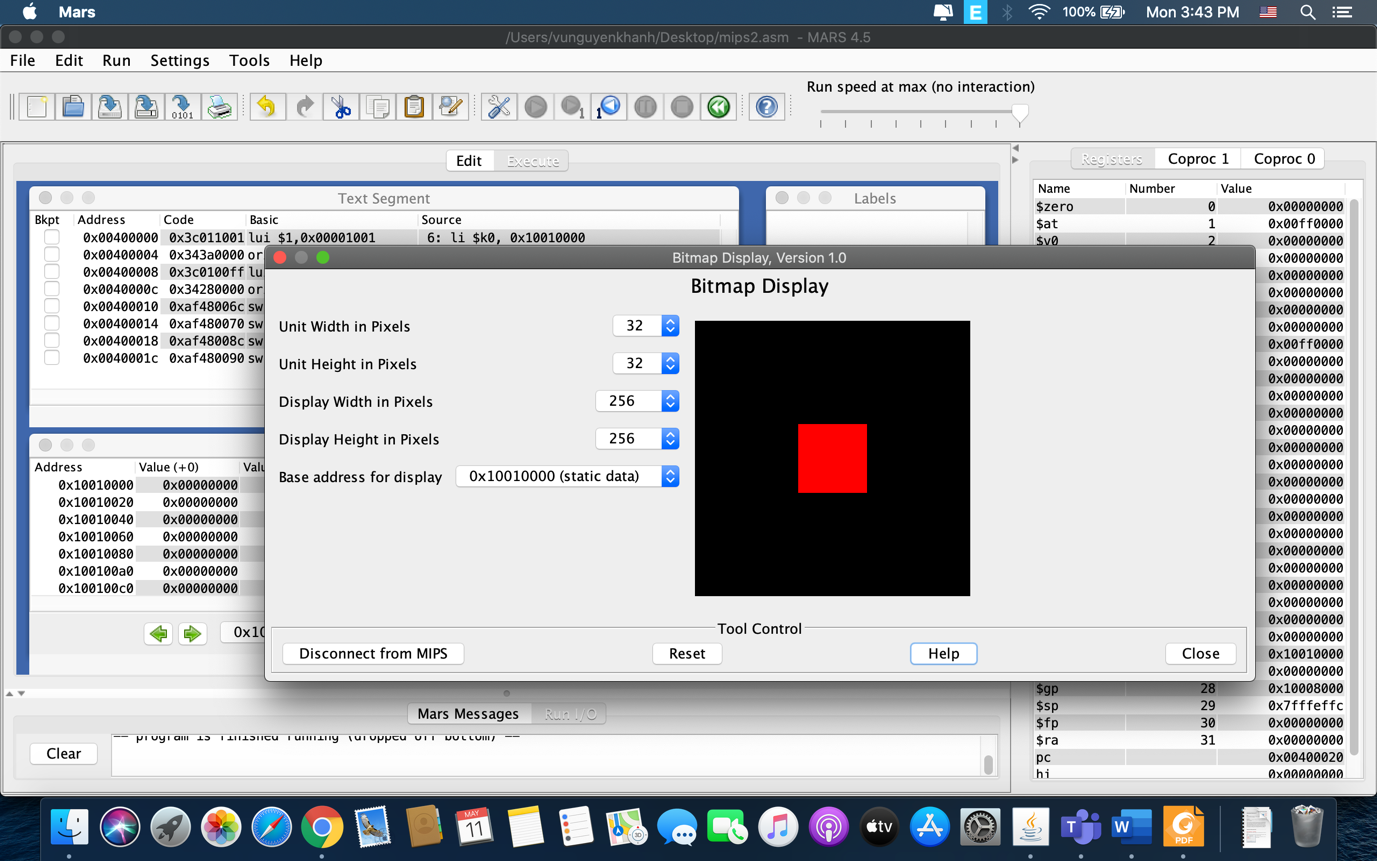
li $t0, RED

sw $t0, 108($k0)

sw $t0, 112($k0)

sw $t0, 140($k0)

sw $t0, 144($k0)



**Assignment 3:**

.eqv HEADING 0xffff8010 # Integer: An angle between 0 and 359

# 0 : North (up)

# 90: East (right)

# 180: South (down)

# 270: West (left)

.eqv MOVING 0xffff8050 # Boolean: whether or not to move

.eqv LEAVETRACK 0xffff8020 # Boolean (0 or non-0):

# whether or not to leave a track

.eqv WHEREX 0xffff8030 # Integer: Current x-location of MarsBot

.eqv WHEREY 0xffff8040 # Integer: Current y-location of MarsBot

.text

main: jal TRACK # draw track line

addi $a0, $zero, 90 # Marsbot rotates 90\* and start running

jal ROTATE

jal GO

sleep1: addi $v0,$zero,32 # Keep running by sleeping in 5000 ms

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

go1: addi $a0, $zero, 210 # Marsbot rotates 210\*

jal ROTATE

sleep2: addi $v0,$zero,32 # Keep running by sleeping in 5000 ms

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

go2: addi $a0, $zero, 330 # Marsbot rotates 330\*

jal ROTATE

sleep3: addi $v0,$zero,32 # Keep running by sleeping in 5000 ms

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

end\_main:

#-----------------------------------------------------------

# GO procedure, to start running

# param[in] none

#-----------------------------------------------------------

GO: li $at, MOVING # change MOVING port

addi $k0, $zero,1 # to logic 1,

sb $k0, 0($at) # to start running

jr $ra

#-----------------------------------------------------------

# STOP procedure, to stop running

# param[in] none

#-----------------------------------------------------------

STOP: li $at, MOVING # change MOVING port to 0

sb $zero, 0($at) # to stop

jr $ra

#-----------------------------------------------------------

# TRACK procedure, to start drawing line

# param[in] none

#-----------------------------------------------------------

TRACK: li $at, LEAVETRACK # change LEAVETRACK port

addi $k0, $zero,1 # to logic 1,

sb $k0, 0($at) # to start tracking

jr $ra

#-----------------------------------------------------------

# UNTRACK procedure, to stop drawing line

# param[in] none

#-----------------------------------------------------------

UNTRACK:li $at, LEAVETRACK # change LEAVETRACK port to 0

sb $zero, 0($at) # to stop drawing tail

jr $ra

#-----------------------------------------------------------

# ROTATE procedure, to rotate the robot

# param[in] $a0, An angle between 0 and 359

# 0 : North (up)

# 90: East (right)

# 180: South (down)

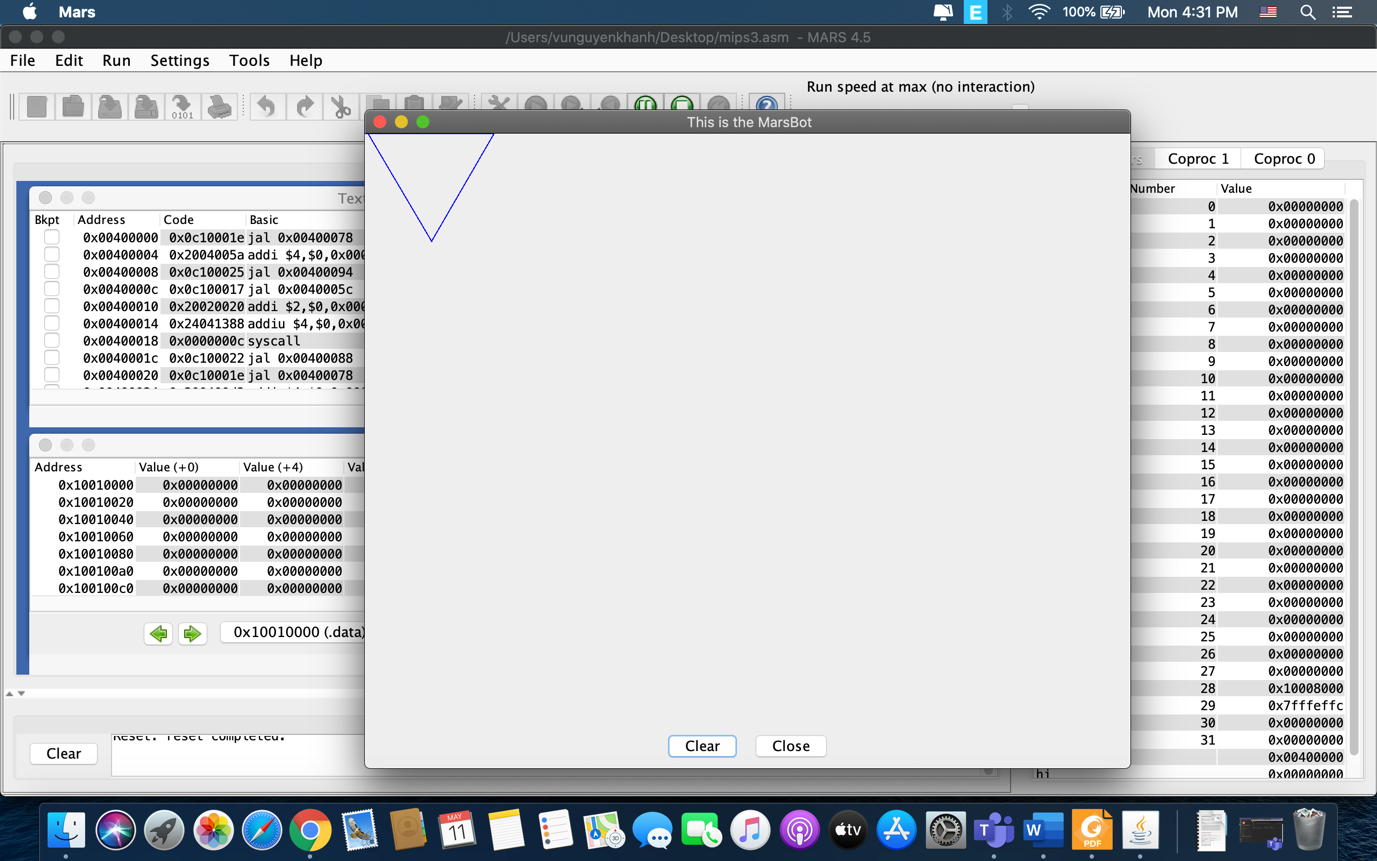
# 270: West (left)

#-----------------------------------------------------------

ROTATE: li $at, HEADING # change HEADING port

sw $a0, 0($at) # to rotate robot

jr $ra



**Assignment 4:**

.eqv KEY\_CODE 0xFFFF0004 # ASCII code from keyboard, 1 byte

.eqv KEY\_READY 0xFFFF0000 # =1 if has a new keycode ?

# Auto clear after lw

.eqv DISPLAY\_CODE 0xFFFF000C # ASCII code to show, 1 byte

.eqv DISPLAY\_READY 0xFFFF0008 # =1 if the display has already to do

# Auto clear after sw

.eqv e\_Char 0x65

.eqv x\_Char 0x78

.eqv i\_Char 0x69

.eqv t\_Char 0x74

.text

li $k0, KEY\_CODE

li $k1, KEY\_READY

li $s0, DISPLAY\_CODE

li $s1, DISPLAY\_READY

loop: nop

WaitForKey: lw $t1, 0($k1) # $t1 = [$k1] = KEY\_READY

beq $t1, $zero, WaitForKey # if $t1 == 0 then Polling

ReadKey: lw $t0, 0($k0) # $t0 = [$k0] = KEY\_CODE

j checkE

WaitForDis: lw $t2, 0($s1) # $t2 = [$s1] = DISPLAY\_READY

beq $t2, $zero, WaitForDis # if $t2 == 0 then Polling

ShowKey: sw $t0, 0($s0) # show key

nop

j loop

checkE: beq $t3, e\_Char, checkX # check if exist e in queue, checkX

bne $t0, e\_Char, WaitForDis # if current char is not e, continue

add $t3, $t0, $zero # save 'e' to $t3

j WaitForDis

checkX: beq $t4, x\_Char, checkI # check if exist x in queue, checkI

bne $t0, x\_Char, reset # if current char is not x, reset then continue

add $t4, $t0, $zero # save 'x' to $t4

j WaitForDis

checkI: beq $t5, i\_Char, checkT # check if exist i in queue, checkT

bne $t0, i\_Char, reset # if current char is not i, reset then continue

add $t5, $t0, $zero # save 'i' to $t5

j WaitForDis

checkT: beq $t0, t\_Char, terminate # check if meet t, terminate (exit word complete)

j reset # if current char is not t, reset then continue

reset: li $t3, 0 # set 'e' to unspecified

li $t4, 0 # set 'x' to unspecified

li $t5, 0 # set 'i' to unspecified

j WaitForDis

terminate: li $v0, 10

syscall

