Assignment Lab 10

Họ tên: Hồ Tuấn Huy

MSSV: 20225856

Assignment 1:

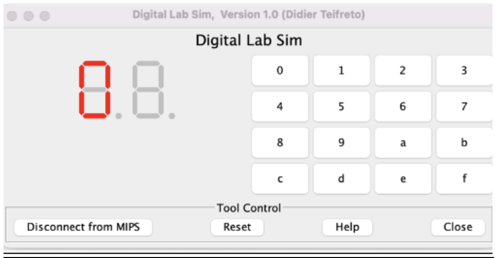
Assignment 1

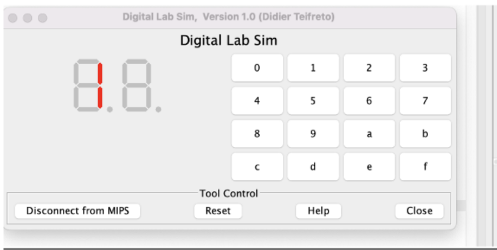
Create a new project, type in, and build the program of Home Assignment 1.
Show different values on LED . Write a program to make the led count from 0->9->0

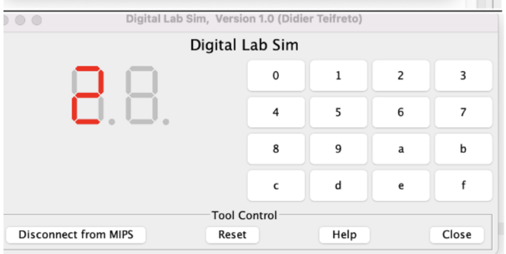
CODE:

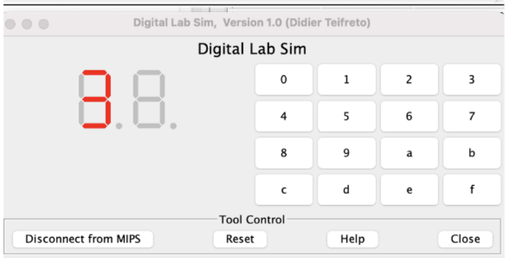
```
.eqv SEVENSEG_LEFT 0xFFFF0011 # Dia chi cua den led 7 doan trai.
 # Bit 0 = doan a;
 # Bit 1 = doan b; ...
# Bit 7 = dau .
.data
a: .byte 0x3f, 0x06, 0x5b, 0x4f, 0x66, 0x6d, 0x7d, 0x07, 0x7f ,0x6f
.text
la $t1, a
li $s1, 0
main:
loop:
 beq $s1,10,loop1
 lb $a0,($t1)
 j SHOW_7SEG_LEFT
 nop

SHOW_7SEG_LEFT:
 li $t0, SEVENSEG_LEFT
 sb $a0, 0($t0)
 nop
 addi $s1,$s1,1
 addi $t1,$t1,1
 li $v0,32
 li $a0,1000
 syscall
 j loop

loop1:
 li $s1,0
 la $t1,a
 j loop
```

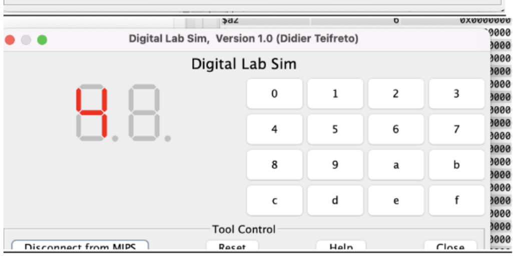
Kết quả:

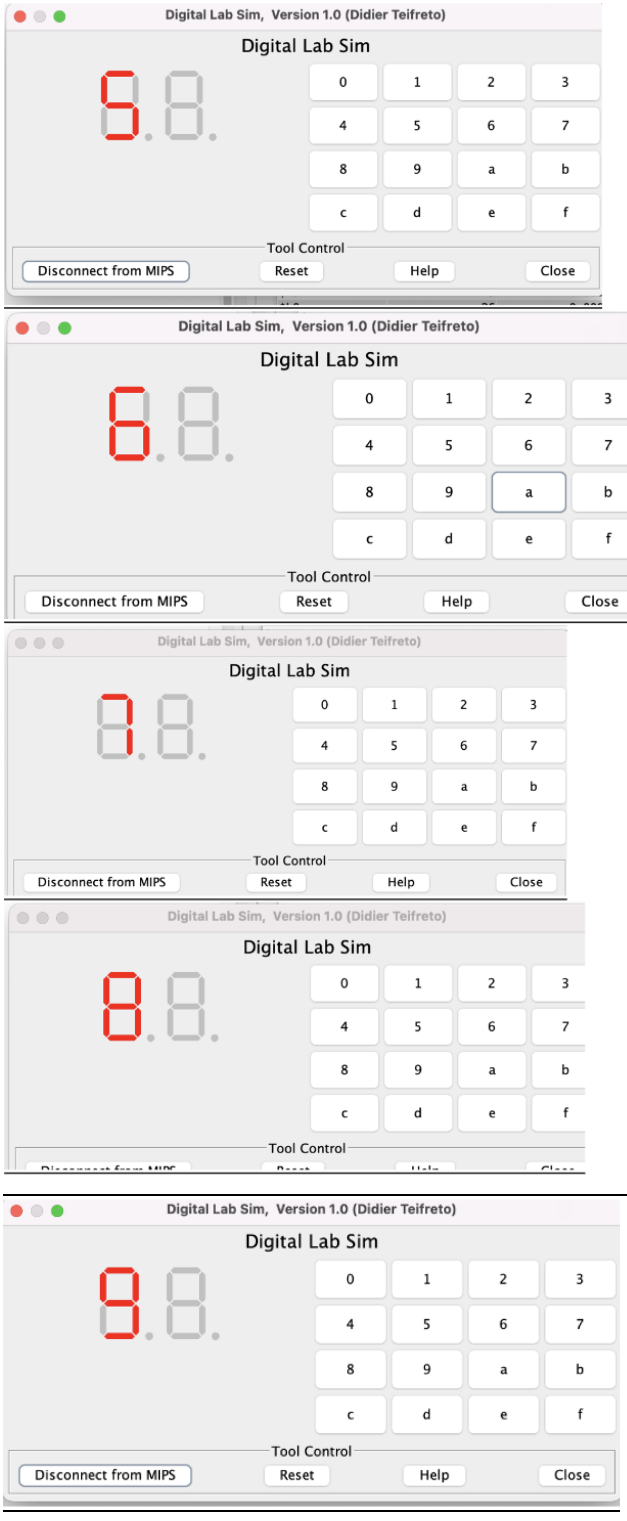

Digital Lab Sim, Version 1.0 (Didier Teifreto)
Digital Lab Sim
0 1 2 3
4 5 6 7
8 9 a b
c d e f
Tool Control
Disconnect from MIPS
Reset
Help
Close


Digital Lab Sim, Version 1.0 (Didier Teifreto)
Digital Lab Sim
0 1 2 3
4 5 6 7
8 9 a b
c d e f
Tool Control
Disconnect from MIPS
Reset
Help
Close


Digital Lab Sim, Version 1.0 (Didier Teifreto)
Digital Lab Sim
0 1 2 3
4 5 6 7
8 9 a b
c d e f
Tool Control
Disconnect from MIPS
Reset
Help
Close


Digital Lab Sim, Version 1.0 (Didier Teifreto)
Digital Lab Sim
0 1 2 3
4 5 6 7
8 9 a b
c d e f
Tool Control
Disconnect from MIPS
Reset
Help
Close


Digital Lab Sim, Version 1.0 (Didier Teifreto)
Digital Lab Sim
0 1 2 3
4 5 6 7
8 9 a b
c d e f
Tool Control



Giải thích:

- Trước hết khởi tạo mảng a chứa các giá trị byte tương ứng với các đoạn LED để hiển thị các số từ 0 đến 9.

- Khởi tạo các thanh ghi:
  + Load địa chỉ mảng a vào thanh ghi $t1.
  + Khởi tạo giá trị 0 cho thanh ghi $s1.
- Vòng lặp chính:
  + Nếu $s1=10 thì nhảy đến hàm loop1 (hàm lặp lại quá trình từ đầu).
  + Load byte tại địa chỉ $t1 vào thanh ghi $a0.
  + Nhảy đến hàm SHOW_7SEG_LEFT.
- Hiển thị số trên màn hình LED 7 đoạn:
  + Load địa chỉ của LED 7 đoạn vào thanh ghi $t0.
  + Lưu byte trong $a0 vào địa chỉ $t0 (hiển thị số trên LED 7 đoạn).
  + Tăng giá trị $s1 lên 1.
  + Tăng địa chỉ $t1 lên 1 để trỏ đến phần tử tiếp theo trong mảng.

Assignment 2:

Assignment 2

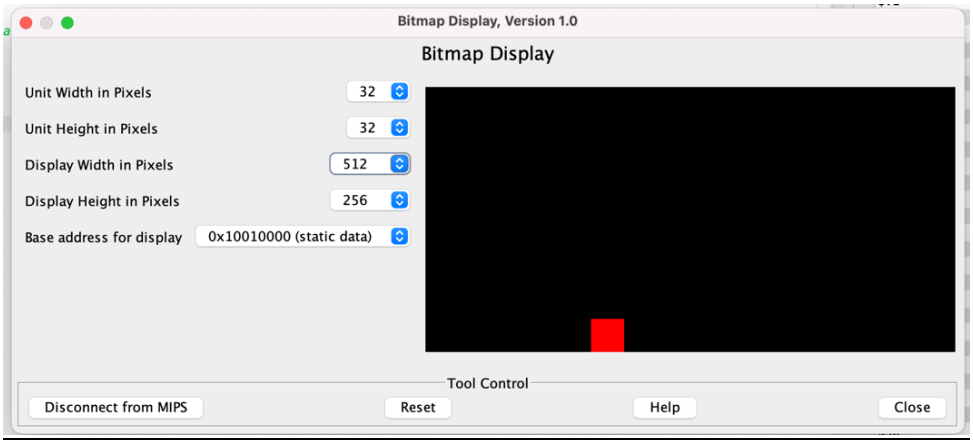
Create a new project, type in, and build the program of Home Assignment 2.
Draw something. Write a program to draw a red square, and moving the square from right to left

---

CODE:

```
.eqv MONITOR_SCREEN 0x10010000 #Dia chi bat dau cua bo nho man hinh
.eqv RED 0x00FF0000 #Cac gia tri mau thuong su dung
.eqv BLACK 0x00000000
.text
li $k0, MONITOR_SCREEN #Nap dia chi bat dau cua man hinh
 loop:
 li $t0, RED
 sw $t0, 0($k0)
 nop
 li $v0,32
 li $a0,1000
 syscall
 li $t0, BLACK
 sw $t0, 0($k0)
 nop
 addi $k0,$k0,4
 j loop
```

Kết quả:



Giải thích:

- Chương trình liên tục thay đổi màu của các điểm ảnh trên màn hình từ đỏ sang đen và ngược lại.
- Mỗi điểm ảnh được vẽ bằng cách lưu giá trị màu tương ứng vào địa chỉ bộ nhớ màn hình.
- Vòng lặp loop điều khiển quá trình này và có delay giữa các lần thay đổi màu.

Assignment 3:

Assignment 3

Create a new project, type in, and build the program of Home Assignment 3.
Make the Bot run and draw a triangle by tracking and redraw the 1st triangle

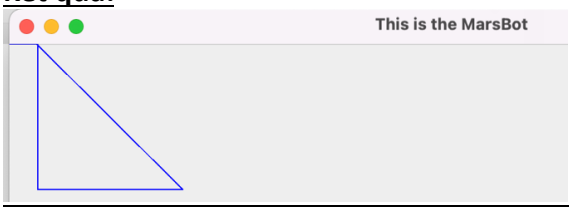
CODE:

```
.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
        nop
        addi $a0, $zero, 90      # Marsbot rotates 90* and start
#running
        jal ROTATE
        nop
        jal GO
        nop
```

```
sleep1:
        addi $v0,$zero,32               # Keep running by sleeping in 1000 ms
        li $a0,1000
        syscall
        jal UNTRACK # keep old track
        nop
        jal TRACK # and draw new track line
        nop
goDOWN:
        addi $a0, $zero, 180 # Marsbot rotates 180*
        jal ROTATE
        nop
sleep2:
        addi $v0,$zero,32 # Keep running by sleeping in 2000 ms
        li $a0,5000
        syscall
        jal UNTRACK # keep old track
        nop
        jal TRACK # and draw new track line
        nop
goRIGHT:
        addi $a0, $zero, 90 # Marsbot rotates 90*
        jal ROTATE
        nop
sleep3:
        addi $v0,$zero,32 # Keep running by sleeping in 1000 ms
        li $a0,5000
        syscall
        jal UNTRACK # keep old track
        nop
        jal TRACK # and draw new track line
        nop
goASKEW:
        addi $a0, $zero, -45 # Marsbot rotates -45*
        jal ROTATE
        nop
sleep4:
        addi $v0,$zero,32 # Keep running by sleeping in 2000 ms
        li $a0,7071
        syscall
        jal UNTRACK # keep old track
        nop
end_main:
        jal STOP
```

```
        nop
GO:
        li $at, MOVING # change MOVING port
        addi $k0, $zero,1 # to logic 1,
        sb $k0, 0($at) # to start running
        nop
        jr $ra
        nop

STOP:
        li $at, MOVING # change MOVING port to 0
        sb $zero, 0($at) # to stop
        nop
        jr $ra
        nop

TRACK:
        li $at, LEAVETRACK # change LEAVETRACK port
        addi $k0, $zero,1 # to logic 1,
        sb $k0, 0($at) # to start tracking
        nop
        jr $ra
        nop

UNTRACK:
        li $at, LEAVETRACK # change LEAVETRACK port to 0
        sb $zero, 0($at) # to stop drawing tail
        nop
        jr $ra
        nop

ROTATE:
        li $at, HEADING # change HEADING port
        sw $a0, 0($at) # to rotate robot
        nop
        jr $ra
        nop
```

Kết quả:



Giải thích:

- Hàm ROTATE: chỉnh góc của con trỏ để đi tiếp.
- Hàm STOP: set giá trị MOVING về 0 để dừng con trỏ.
- Hàm GO: set giá trị MOVING về 1 và con trỏ di chuyển theo góc được chỉ định qua hàm ROTATE.
- Hàm TRACK: set LEAVETRACK về 1 và bắt đầu vẽ.
- Hàm UNTRACK: set LEAVETRACK về 0 và dừng vẽ.
- Các nhãn loop để set thời gian con trỏ di chuyển theo hướng vừa được set bằng hàm ROTATE để vẽ các đoạn theo ý muốn.
- Thời gian sleep hay chiều dài của các nét đã được tính toán và điều chỉnh để chữ vẽ ra không bị lệch quá nhiều.

Assignment 4:

Assignment 4

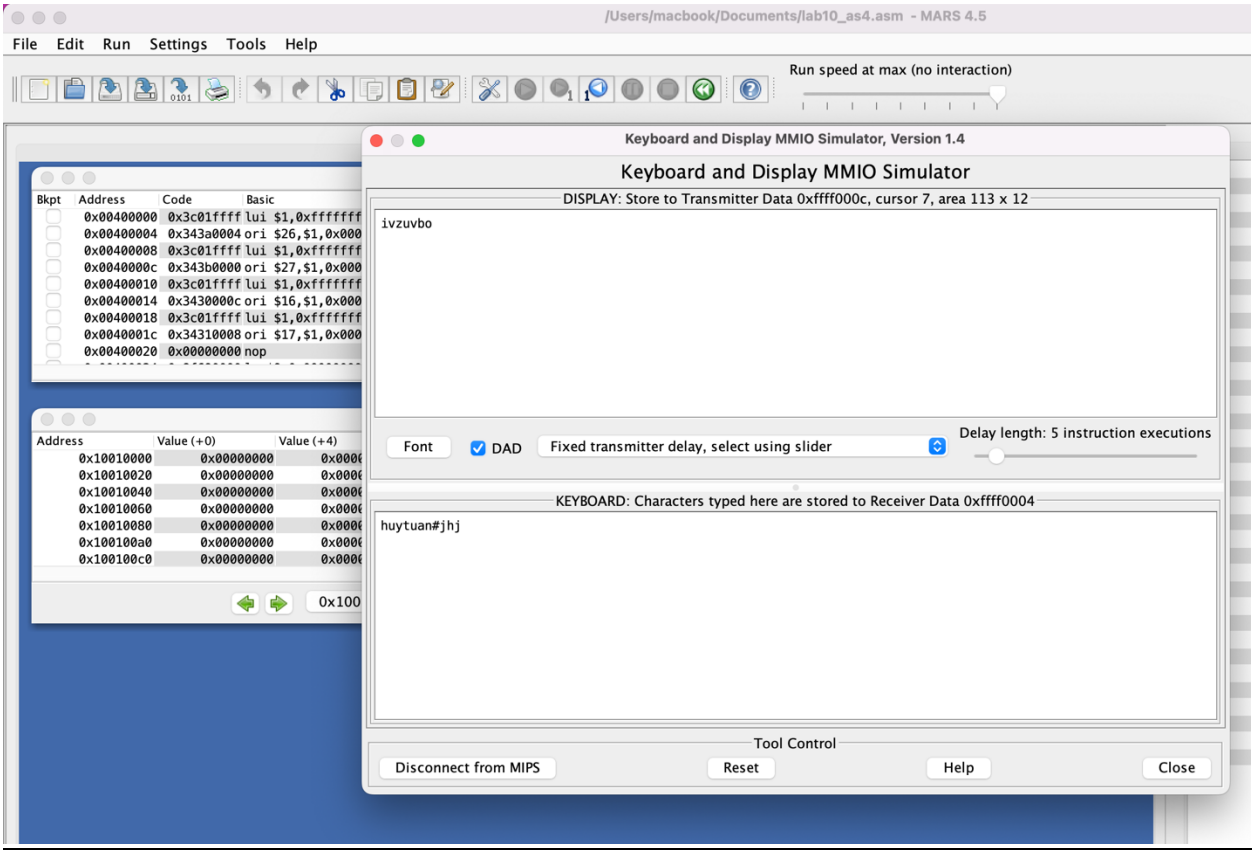
Create a new project, type in, and build the program of Home Assignment 4.
Read key char and terminate the application when receiving "exit" command

CODE:

```
.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
.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
nop
beq $t1, $zero, WaitForKey # if $t1 == 0 then Polling
nop
#-----------------------------------------------------
ReadKey: lw $t0, 0($k0) # $t0 = [$k0] = KEY_CODE
nop
beq $t0, '#', exit
#-----------------------------------------------------
WaitForDis: lw $t2, 0($s1) # $t2 = [$s1] = DISPLAY_READY
nop
beq $t2, $zero, WaitForDis # if $t2 == 0 then Polling
nop
#-----------------------------------------------------
```

```
Encrypt: addi $t0, $t0, 1 # change input key
#---------------------------------------------------
ShowKey: sw $t0, 0($s0) # show key
nop
#---------------------------------------------------
j loop
nop
exit:
li $v0, 10
syscall
```

Kết quả:



Giải thích:

- WaitForKey là hàm kiểm tra đã nhập được gì chưa.
- ReadKey để đọc ký tự vừa nhập, nếu phát hiện '#' thì nhảy đến exit thoát chương trình.
- WaitForDis để kiểm tra ký tự đã có thể hiển thị chưa.
- Encrypt để tăng giá trị vừa nhập vào 1 rồi in ra màn hình.