# **BÁO CÁO THỰC HÀNH KIẾN TRÚC MÁY TÍNH TUẦN 10**

**Họ và tên: Hoàng Văn Thắng**

**MSSV: 20235828**

### Assignment 1

Tạo project để thực hiện Home Assignment 1. Thay đổi các giá trị hiển thị trên LED 7 đoạn để hiển thị 2 chữ số cuối của MSSV.

**Nhập chương trình**

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

# Bit 0 = a

# Bit 1 = b

# ...

# Bit 7 = dau .

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

.text

main:

li a0, 0x5B # set value for segments

jal SHOW\_7SEG\_LEFT # show

li a0, 0x7F # set value for segments

jal SHOW\_7SEG\_RIGHT # show

exit:

li a7, 10

ecall

end\_main:

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

# 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

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

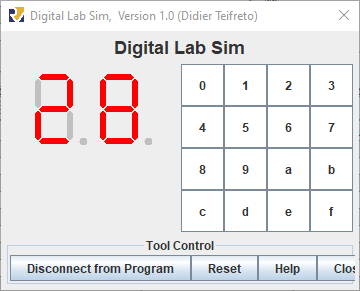
SHOW\_7SEG\_RIGHT:

li t0, SEVENSEG\_RIGHT # assign port's address

sb a0, 0(t0) # assign new value

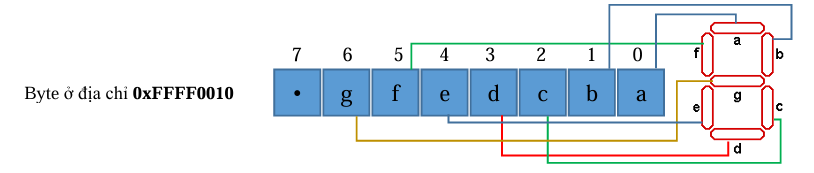
jr ra

**Kết quả chạy**



Khi ta lưu trữ byte vào địa chỉ của LED 7 thanh thì đèn sẽ sáng theo quy luật sau:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| nth bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| segment | . | g | f | e | d | c | b | a |
| 1 | bật | | | | | | | |
| 0 | tắt | | | | | | | |



Để hiển thị 2 chữ số cuối của MSSV (20235828 – tức số 28):

* Hiển thị số 2 thanh a, b, g, e, d sáng giá trị truyền vào là 0101 1011 = 0x5B
* Hiển thị số 8 thanh a, b, c, d, e, f, g sáng giá trị truyền vào là 0111 1111 = 0x7F

### Assignment 2

Tạo project để hiển thị trên LED 7 đoạn 2 chữ số cuối của mã ASCII (ở hệ cơ số 10) của ký tự được nhập từ bàn phím.

**Nhập chương trình**

.eqv SEVENSEG\_LEFT 0xFFFF0011

.eqv SEVENSEG\_RIGHT 0xFFFF0010

.data

message: .asciz "Nhap ky tu: "

newline: .asciz "\n"

SEG\_TABLE:

.byte 0x3F, 0x06, 0x5B, 0x4F, 0x66, 0x6D, 0x7D, 0x07, 0x7F, 0x6F

.text

main:

li a7, 4

la a0, message

ecall

li a7, 12

ecall

mv t0, a0

li a7, 4

la a0, newline

ecall

li t3, 100

remu t0, t0, t3

li t4, 10

divu t1, t0, t4

remu t2, t0, t4

la t5, SEG\_TABLE

add t6, t5, t1

lbu a0, 0(t6)

jal SHOW\_7SEG\_LEFT

add t6, t5, t2

lbu a0, 0(t6)

jal SHOW\_7SEG\_RIGHT

exit:

li a7, 10

ecall

end\_main:

SHOW\_7SEG\_LEFT:

li t0, SEVENSEG\_LEFT

sb a0, 0(t0)

jr ra

SHOW\_7SEG\_RIGHT:

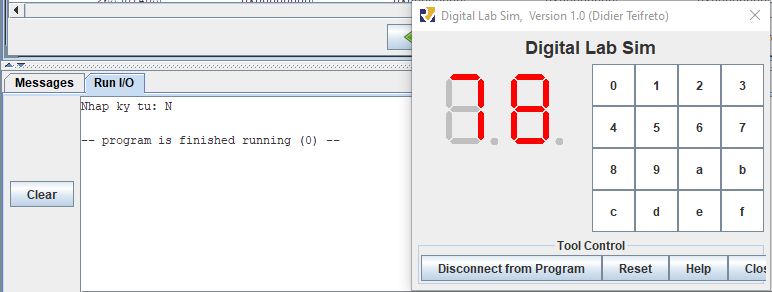
li t0, SEVENSEG\_RIGHT

sb a0, 0(t0)

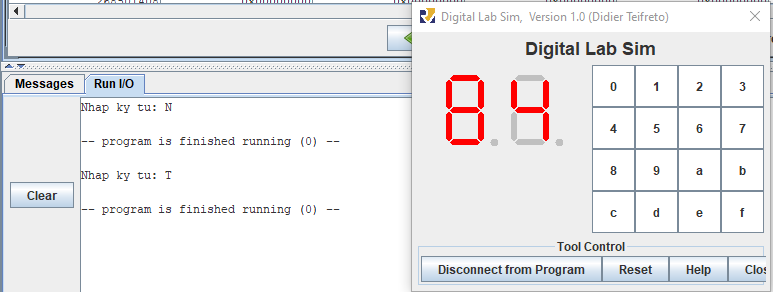
jr ra

**Kết quả chạy chương trình**

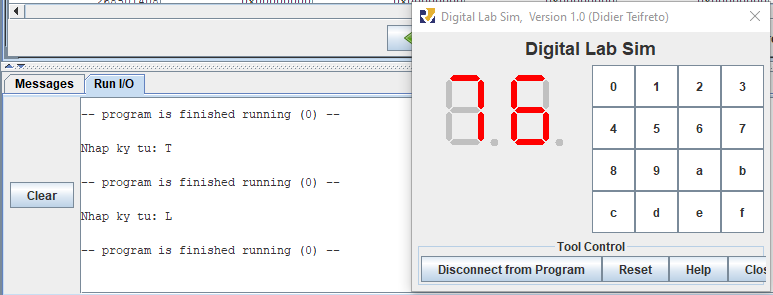
* Trường hợp ký tự đầu vào là N (ASCII: 78)



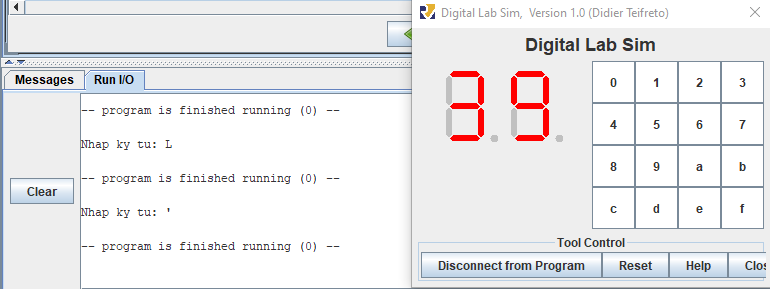
* Trường hợp ký tự đầu vào là T (ASCII: 84)



* Trường hợp ký tự đầu vào là L (ASCII: 76)



Trường hợp ký tự đầu vào là ký tự “’” (ASCII: 39)



**Giải thích**

Chương trình ban đầu sẽ đọc ký tự đầu vào từ bàn phím (syscall: 12), sau đó lấy 2 chữ số cuối của mã ASCII bằng cách chia lấy phần dư của mã ASCII của ký tự đầu vào cho 100. Tách 2 chữ số hàng chục và hàng đơn vị của 2 chữ số cuối của mã bằng cách (chữ số hàng chục thì chia lấy phần nguyên của 2 chữ số cho 10, chữ số hàng đơn vị thì chia lấy phần dư của 2 chữ số cho 10). Sau khi có nội dung 2 chữ số cuối thì ta sẽ sử dụng bảng mã SEG\_TABLE để tra xem giá trị đầu vào để truyền cho LED bên trái và LED bên phải. Sau đó kết thúc chương trình.

### Assignment 3

Tạo project để thực hiện Home Assignment 2. Cập nhật mã nguồn để vẽ bàn cờ vua trên màn hình với 2 màu bất kỳ (khác màu đen).

**Nhập chương trình**

.eqv MONITOR\_SCREEN 0x10010000

.eqv WHITE 0x00FFFFFF

.eqv YELLOW 0x00FFFF00

.text

main:

li t0, 0

outer:

li t1, 0

inner:

slli t3, t0, 3

add t3, t3, t1

slli t3, t3, 2

li t4, MONITOR\_SCREEN

add t3, t3, t4

add t5, t0, t1 # t5 = i + j

andi t5, t5, 1 # (i+j)%2

beqz t5, choose\_white

li t6, YELLOW

j store

choose\_white:

li t6, WHITE

store:

sw t6, 0(t3)

addi t1, t1, 1

li t6, 8

blt t1, t6, inner

addi t0, t0, 1

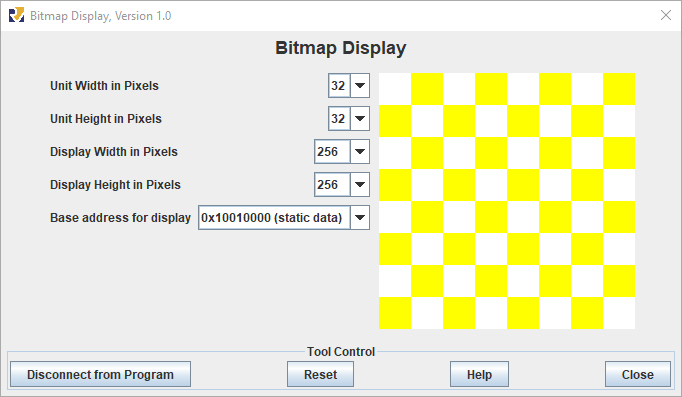
blt t0, t6, outer

exit:

li a7, 10

ecall

**Kết quả chạy**



**Giải thích kết quả chạy**

Tại chạy vòng lặp lồng hàng (thanh ghi t0) và cột (thanh ghi t1) bắt đầu từ 0 cho đến 7. Trong vòng lặp bên trong, ta sẽ tính địa chỉ của ô cần được tô màu. Sau đó ta sẽ tính (t0 + t1) chia lấy phần dư cho 2 để xác định màu cần tô. Nếu (t0 + t1) là số chẵn thì tô màu trắng, còn (t0 + t1) là số lẻ thì tô màu vàng. Sau đó ta sẽ ghi màu vào địa chỉ tương ứng trong bộ nhớ màn hình. Sau khi ta vẽ đủ 64 ô thì sẽ thoát chương trình (syscall: 10).

### Assignment 4

Tạo project để thực hiện Home Assignment 3. Cập nhật mã nguồn để hoàn thành yêu cầu sau: Nhập ký tự thường hiển thị ký tự hoa tương ứng, nhập ký tự hoa hiển thị ký tự thường tương ứng, nhập ký tự số thì giữ nguyên, nhập ký tự khác hiển thị ký tự \*. Khi nhập chuỗi ký tự "exit" thì kết thúc chương trình.

**Nhập chương trình**

.eqv KEY\_CODE 0xFFFF0004

.eqv KEY\_READY 0xFFFF0000

.eqv DISPLAY\_CODE 0xFFFF000C

.eqv DISPLAY\_READY 0xFFFF0008

.text

main:

li a0, KEY\_CODE

li a1, KEY\_READY

li s0, DISPLAY\_CODE

li s1, DISPLAY\_READY

li s2, 'a'

li s3, 'z'

li s4, 'A'

li s5, 'Z'

li s6, '0'

li s7, '9'

li t5, '\*'

li s8, 0

li s9, 0

li s10, 0

li s11, 0

loop:

WaitForKey:

lw t1, 0(a1)

beq t1, zero, WaitForKey

ReadKey:

lw t0, 0(a0)

CheckExit:

mv s11, s10

mv s10, s9

mv s9, s8

mv s8, t0

li t1, 'e'

bne s11, t1, ProcessChar

li t1, 'x'

bne s10, t1, ProcessChar

li t1, 'i'

bne s9, t1, ProcessChar

li t1, 't'

bne s8, t1, ProcessChar

j exit

ProcessChar:

blt t0, s2, CheckUpper

blt s3, t0, CheckUpper

addi t0, t0, -32

j display

CheckUpper:

blt t0, s4, CheckDigit

blt s5, t0, CheckDigit

addi t0, t0, 32

j display

CheckDigit:

blt t0, s6, star

blt s7, t0, star

j display

star:

mv t0, t5

display:

WaitForDis:

lw t2, 0(s1)

beq t2, zero, WaitForDis

ShowKey:

sw t0, 0(s0)

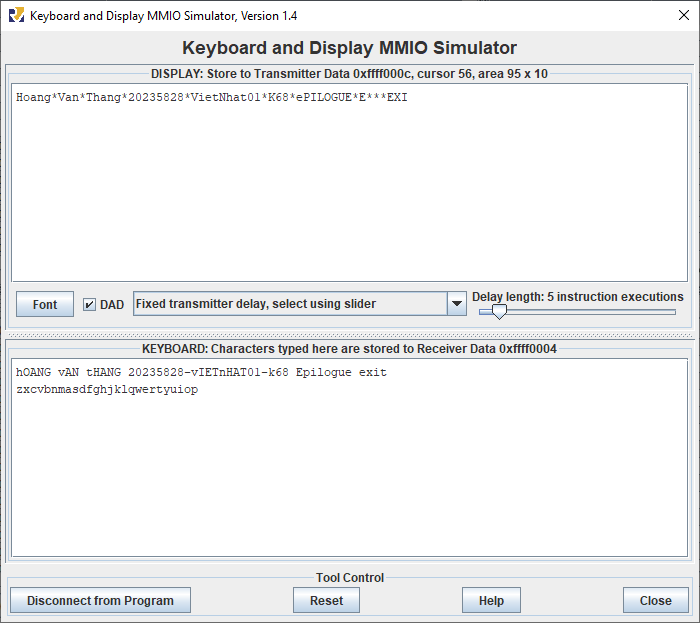
j loop

exit:

li a7, 10

ecall

**Kết quả chạy chương trình**



**Giải thích**

Ta sẽ bắt đầu đọc ký tự đầu vào từ bàn phím khi KEY\_READY = 1 (khi có ký tự được nhập vào). Sau đó ta sẽ cập nhật buffer của 4 ký tự mới nhất được nhập (được lưu trong các thanh ghi từ s8 – s11) để có thể kiểm tra thoát chương trình. Nếu buffer của 4 ký tự mới nhất là “exit” thì ta sẽ thoát chương trình. Nếu ký tự được nhập vào là một chữ cái in thường (a char z) thì ta cập nhật char = char – 32 (tương đương với việc chuyển ký tự từ chữ in thường thành chữ cái in hoa), nếu ký tự được nhập vào là một chữ cái in hoa (A char Z) thì ta cho cập nhật char = char + 32 (tương đương việc chuyển ký tự từ chữ cái in hoa thành chữ cái in thường). Nếu ký tự được nhập vào là một chữ số (0 char 9) thì ta giữ nguyên ký tự số đó. Nếu ký tự được nhập vào là một ký tự khác với các ký tự trên thì ta sẽ chuyển ký tự đó thành ký tự **\***. Cuối cùng, ta sẽ bắt đầu hiển thị ký tự ra màn hình khi DISPLAY\_READY = 1.