**BÁO CÁO THỰC HÀNH KIẾN TRÚC MÁY TÍNH**

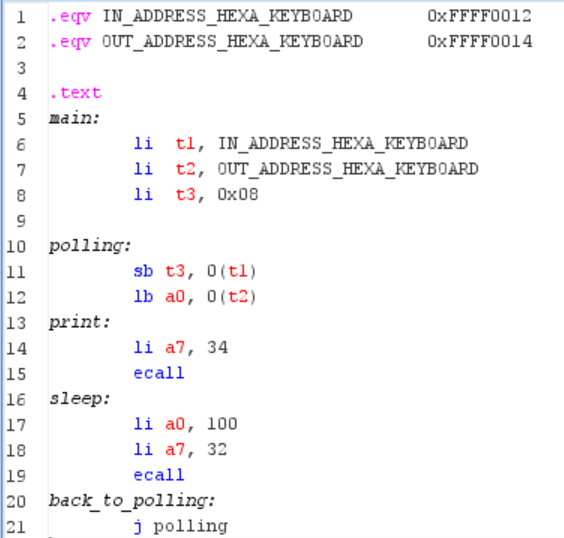
**Lab 11 : Lập trình xử lý ngắt**

|  |  |
| --- | --- |
| **Họ tên** | **MSSV** |
| Phạm Minh Hiển | 20235705 |

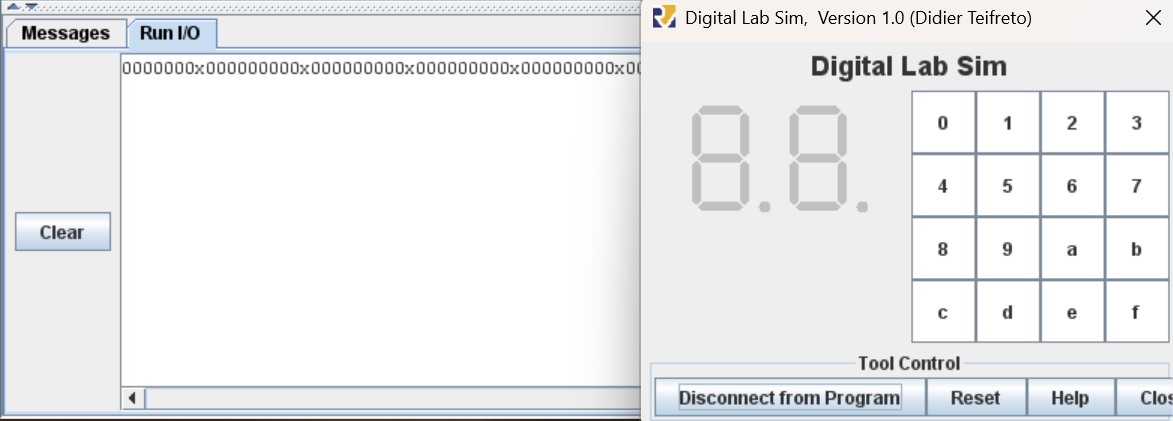
**Assignment 1:**

Tạo project thực hiện Home Assignment 1. Cập nhật mã nguồn để chương trình có thể in ra mã của tất cả 16 nút bấm trên keypad.

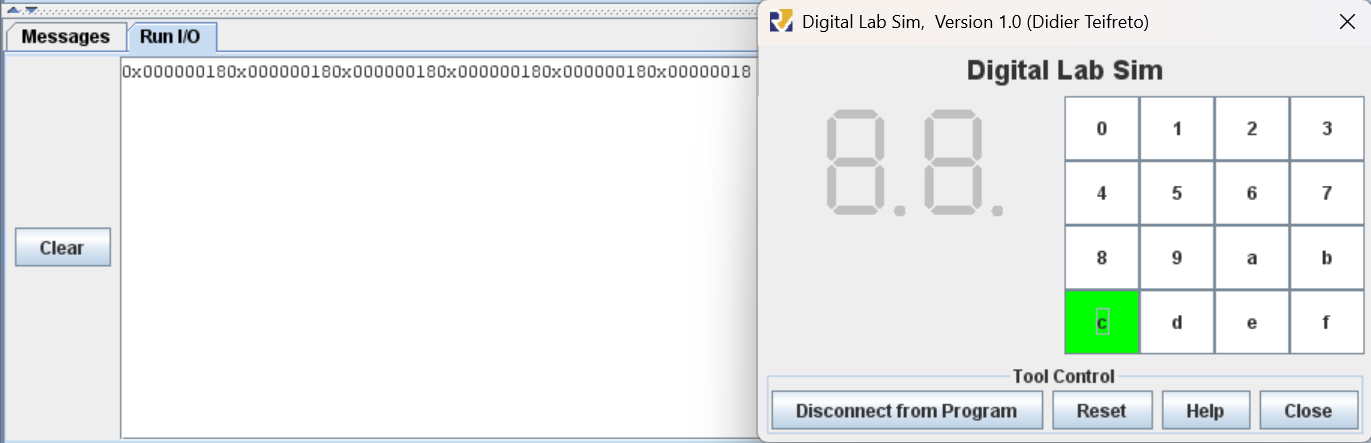
* Home Assignment 1:

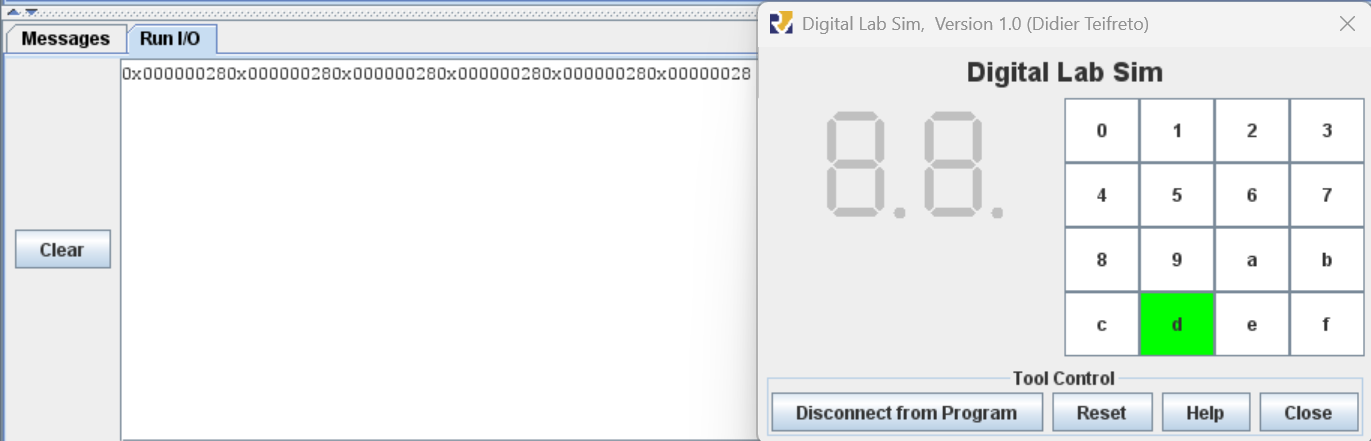


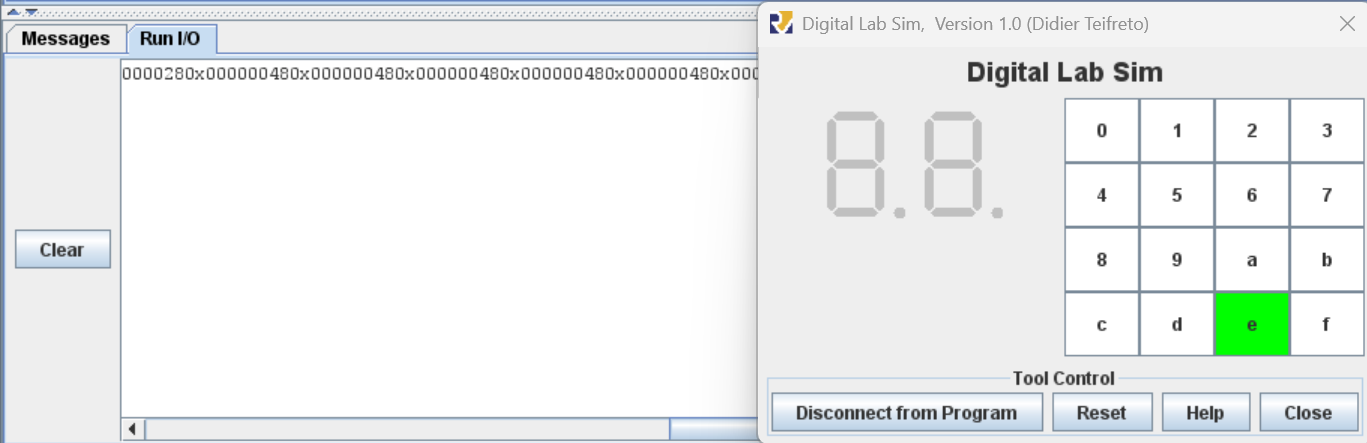
* Kết quả :

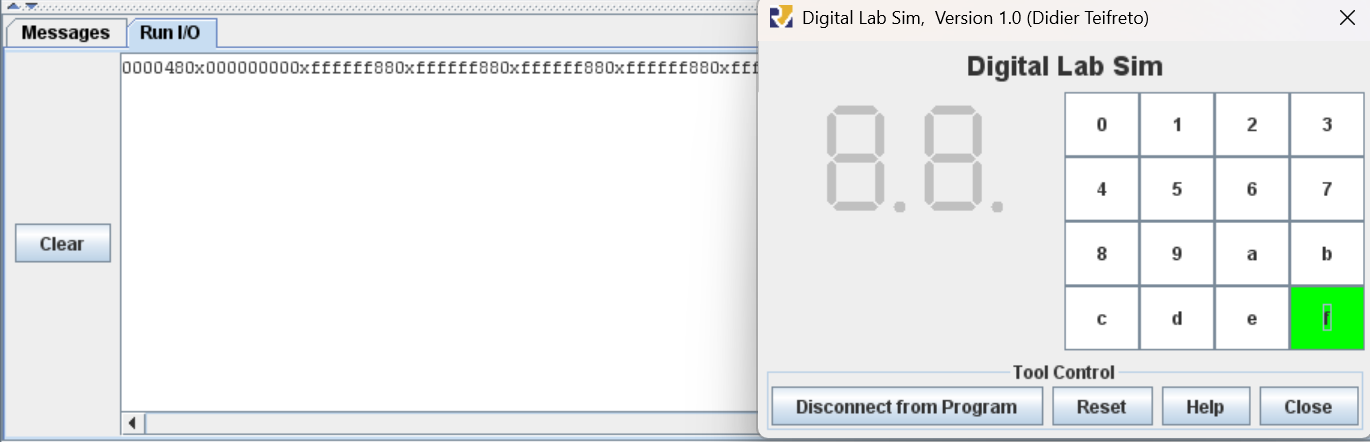


+ Khi chưa nhấn (nút trắng) thì chương trình liên tục in 0x00000000 ra **Run I/O**.









+ Sau khi nhấn 1 trong 4 nút c, d, e, f (nút xanh), thì sẽ in liên tiếp ra màn hình **Run I/O** mã hexa tương ứng với vị trí của nút được nhấn lần lượt là 0x00000018, 0x00000028, 0x00000048, 0xffffff88.

* Cập nhật mã nguồn để hiển thị với tất cả 16 nút:

**Source Code:**

.eqv IN\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0012

.eqv OUT\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0014

.text

main:

li t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li t2, OUT\_ADDRESS\_HEXA\_KEYBOARD

polling:

li t3, 0x01 # row 0

scan\_rows:

sb t3, 0(t1) # Ghi dong hien tai vao input

lb a0, 0(t2) # Doc gia tri hien tai vao dia chi input

beq a0, zero, skip\_print # Neu khong co phim nhan, bo qua in

li a7, 34 # in

ecall

li a0, 100 # sleep

li a7, 32

ecall

skip\_print:

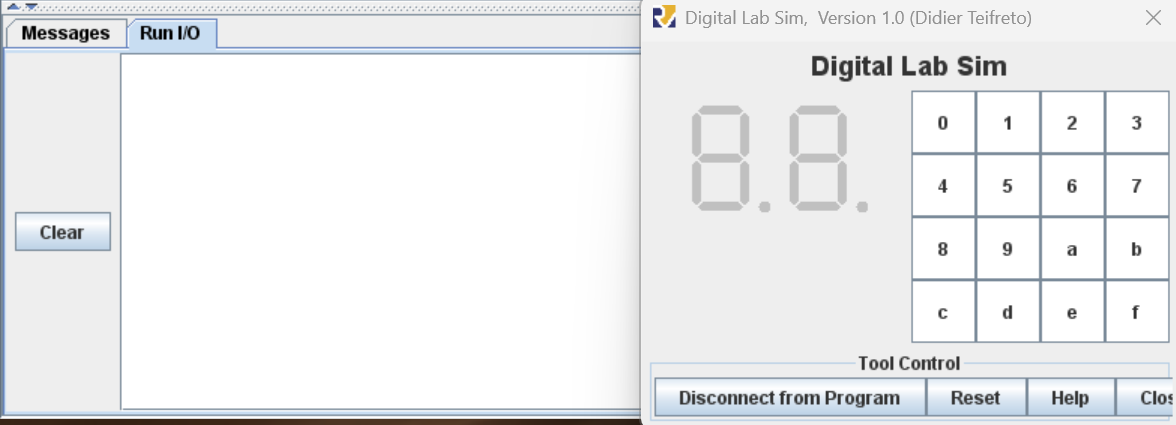
slli t3, t3, 1 # Dich sang dong tiep theo

li t4, 0x10

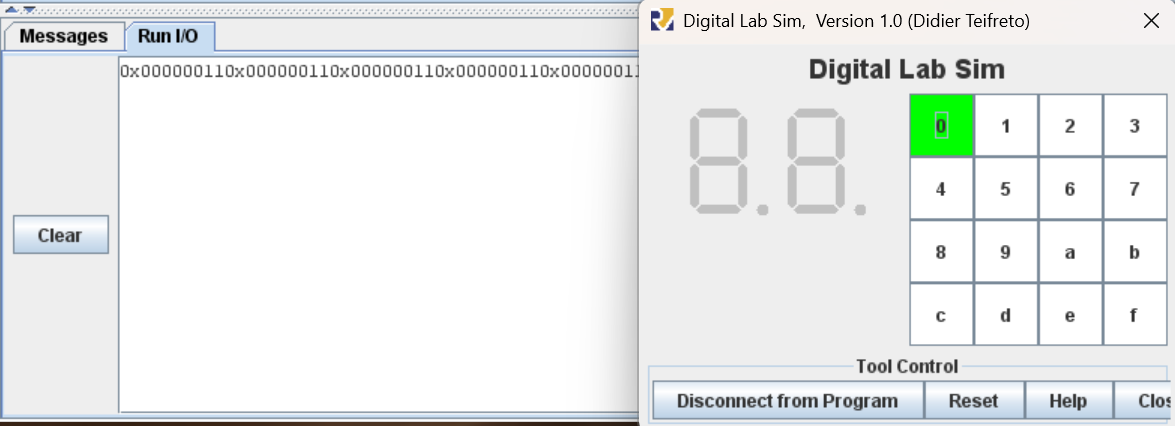
blt t3, t4, scan\_rows # t3 < 0x10, con dong can quet

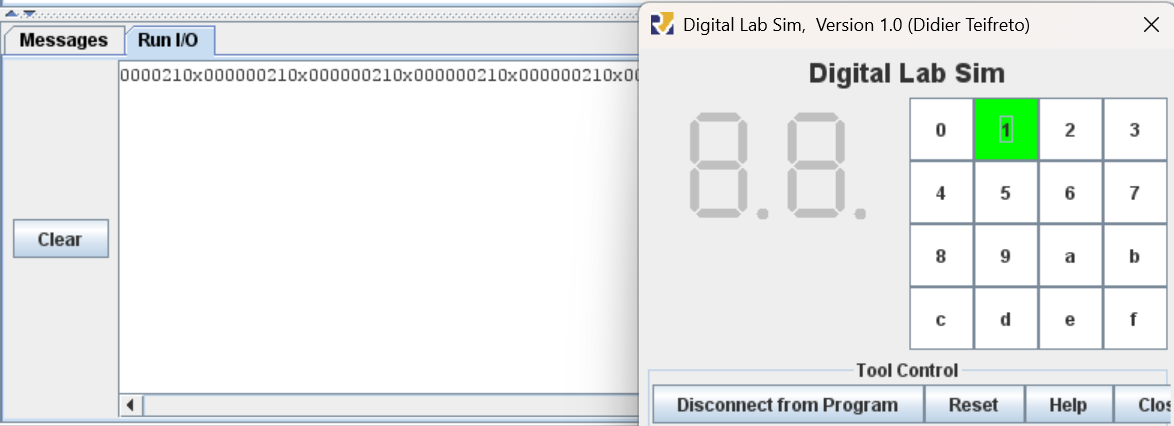
j polling # quet het 4 dong thi quay lai polling

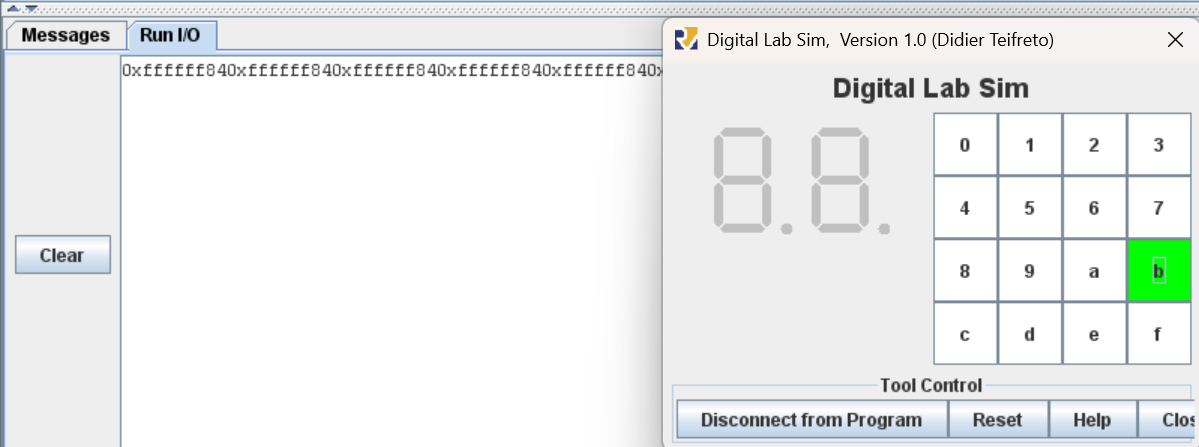
* Kết quả:



+ Khi chưa nhấn nút nào (nút trắng) thì chưa in gì ra **Run I/O.**







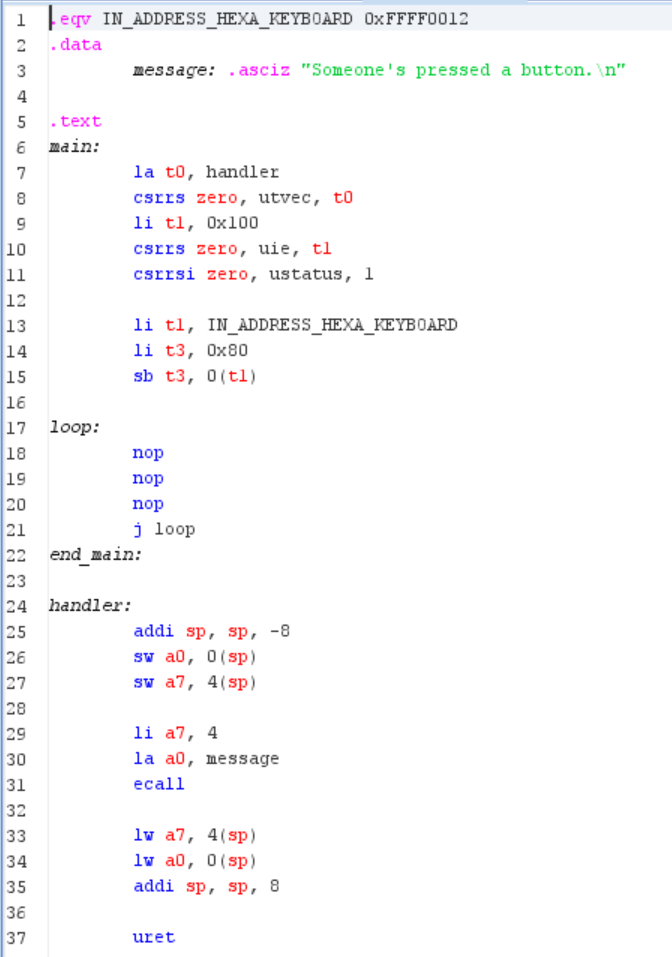
+ Khi nhấn các nút trên keypad(nút xanh) thì chương trình sẽ in ra màn hình **Run I/O** mã hexa tương ứng. Ví dụ: 0 thì in ra 0x00000011, 1 thì in ra 0x00000021, 2 thì in ra 0x00000041, ….. Trong đó 2 chữ số cuối trong mã hexa thể hiện vị trí lần lượt là cột và hàng từ trái sang phải với 1, 2, 4, 8 tương ứng cột/hàng 0, 1, 2, 3.

+ Khi nhấn 1 trong 4 nút ở cột cuối thì do chữ số bên trái là số 8 có mã nhị phân là 1000 nên sẽ in ra 0xffffff8x.

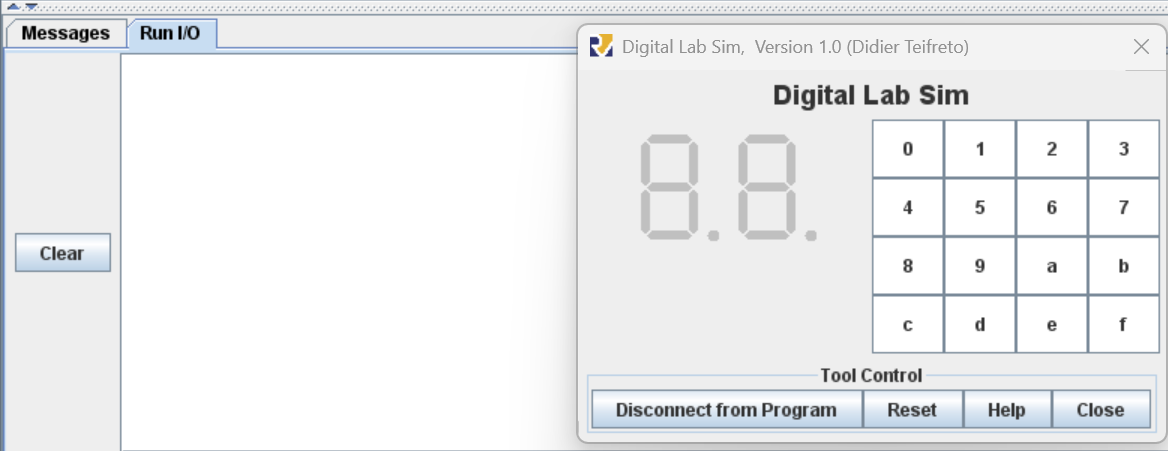
* Chương trình có vẻ hoạt động chính xác.

**Assignment 2:**

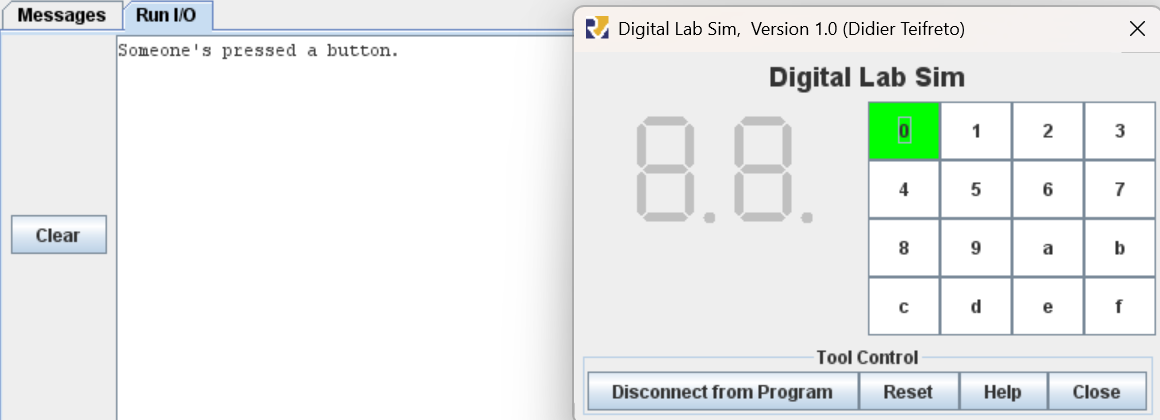
Tạo project để thực hiện và thử nghiệm Home Assignment 2. Chạy ở chế độ từng dòng lệnh, quan sát giá trị của các thanh ghi để hiểu cách chương trình hoạt động.

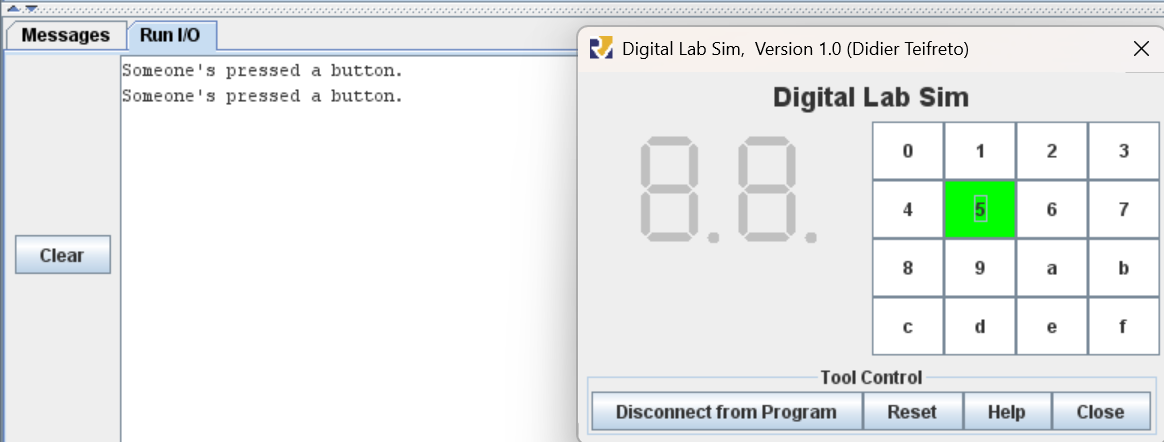


* Kết quả:



+ Ban đầu chưa nhấn nút (nút trắng), chương trình chưa in ra màn hình **Run I/O.**





+ Sau khi nhấn 1 nút bất kỳ (nút xanh) thì chương trình sẽ in ra màn hình chuỗi “Someone’s pressed a button.”

* Chạy ở chế độ từng dòng lệnh:

+ *main*: khởi tạo ngắt

* **la**: t0 chứa địa chỉ hàm xử lý ngắt *handler* 0x00400038.
* **csrrs:** gán **utvec** = **t0** để CPU biết khi có ngắt sẽ nhảy đến handler.
* **li** + **csrrs** uie : gán **uie** = **t1** = 0x100 (bit số 8 = 1) để bật phép nhận external interrupts.
* **csrrs** ustatus : bật bit 0 của **ustatus** cho phép CPU nhận interrupts tổng quát.
* t1 = 0xFFFF0012 (địa chỉ đầu vào keypad), t3 = 0x80 là bit thứ 7 = 1 cấu hình keypad cho phép gửi interrupt khi có phím nhấn.

+ *loop*: vòng lặp vô hạn chờ interrupt từ keypad.

+ Khi có phím được nhấn, keypad gửi External Interrupt, CPU tự động nhảy tới *handler*.

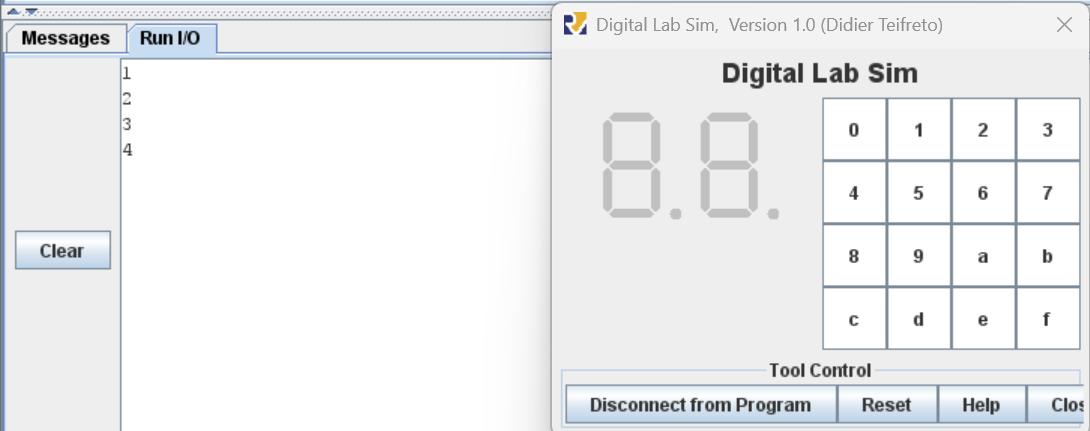
+ *handler*: xử lý ngắt

* **addi, sw :** Cấp phát stack và lưu a0, a7 vào.
* In chuỗi “Someone’s pressed a button.”
* **lw, addi :** Khôi phục a0, a7 từ stack.
* **uret :** quay lại chương trình trở lại *loop.*

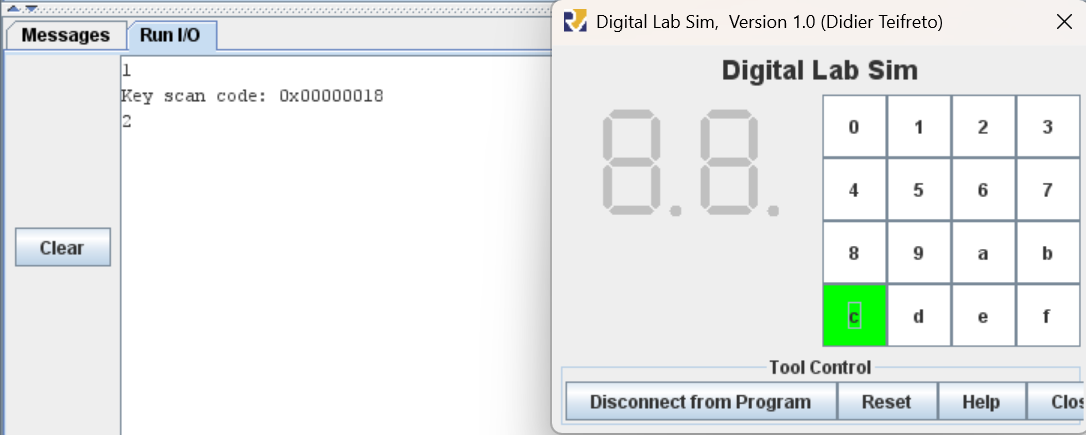
**Assignment 3:**

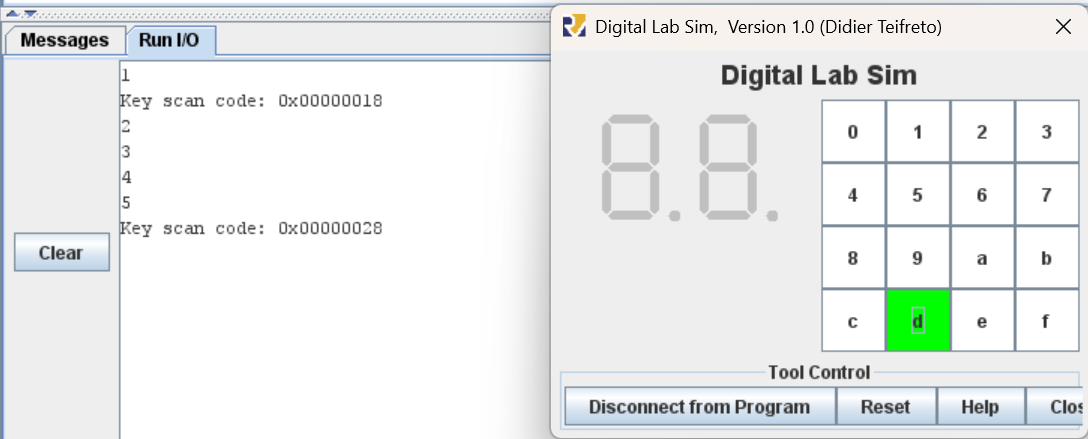
Tạo project để thực hiện và thử nghiệm Home Assignment 3. Chạy ở chế độ từng dòng lệnh, quan sát giá trị của các thanh ghi để hiểu cách chương trình hoạt động. Cập nhật mã nguồn để chương trình có thể in ra mã của tất cả 16 nút bấm trên keypad.

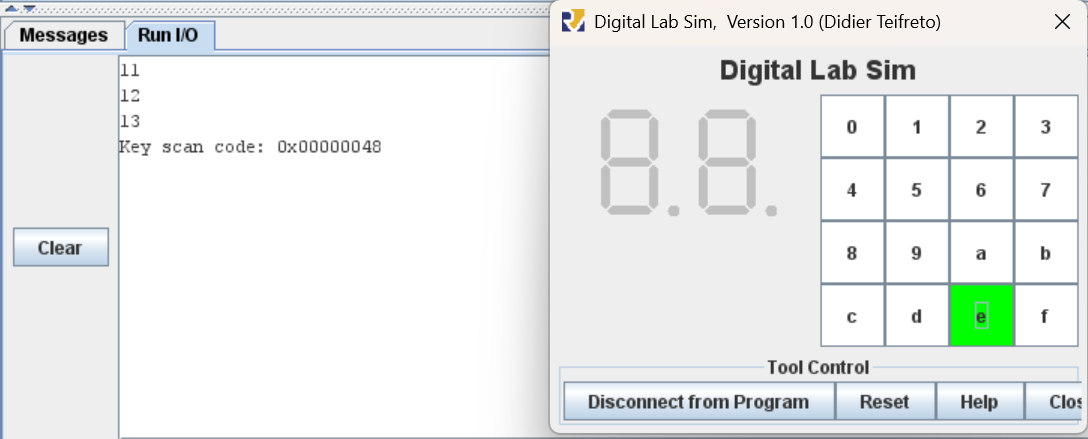
* Home Assignment 3:

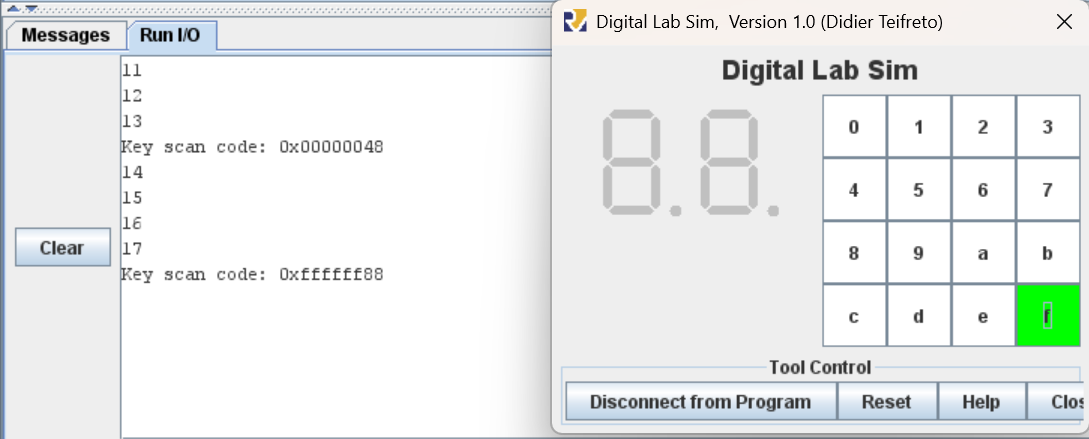


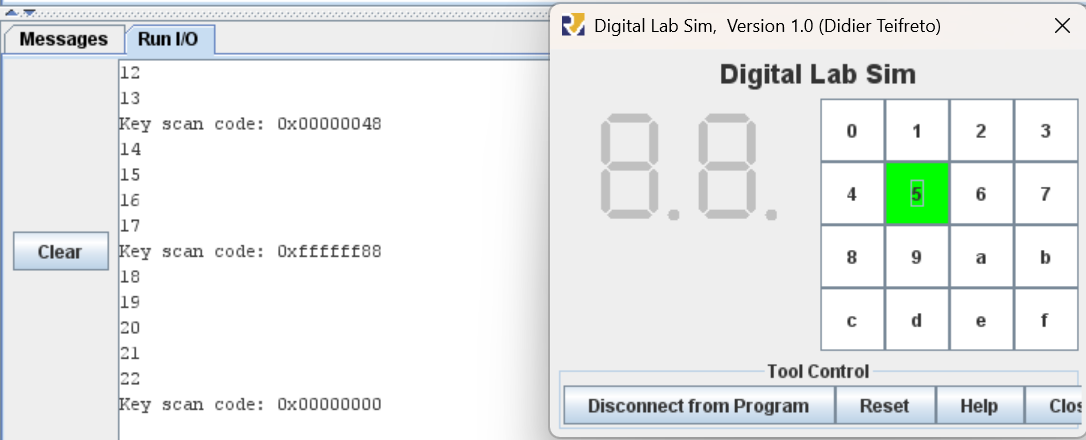
+ Ban đầu khi chưa nhấn nút (nút trắng) thì chương trình in ra chuỗi số nguyên liên tiếp từ 1 trên **Run I/O**.











+ Sau khi chương trình khởi động nếu ta nhấn nút (nút xanh) thì chương trình sẽ in ra chuỗi “Key scan code: “ cùng với mã hexa tương ứng vị trí nút được ấn nếu là 1 trong 4 nút c, d, e, f và mã 0x00000000 nếu là các nút còn lại.

+ Sau khi in chuỗi, nếu không có động tác khác thì chương trình tiếp túc in ra chuỗi số nguyên liên tiếp.

* Chạy từng dòng lệnh:

+ Nạp địa chỉ handler 0x00400058 vào CSR utvec — tức là đặt **hàm xử lý ngắt**.

+ Bật ngắt thiết bị (bit thứ 8 của uie CSR = 1 thông qua t1 = 0x100) — cho phép ngắt từ bên ngoài.

+ Set bit UIE (User Interrupt Enable) trong ustatus, cho phép nhận ngắt khi đang ở user mode, lúc này ustatus = 0x00000001.

+ Ghi 0x80 vào IN\_ADDRESS\_HEXA\_KEYBOARD: Yêu cầu bàn phím bật chế độ gửi interrupt khi bấm phím.

+ Reset thanh ghi s0 về 0 — dùng đếm số lần lặp.

+ *loop:* Tăng biến đếm s0 và in, xuống dòng, nghỉ 300ms rồi lặp lại.

+ *handler:*

* Cấp phát stack và backup 4 thanh ghi a0, a7, t1, t2.
* In ra “Key scan code: “
* Gửi 0x88 vào IN\_ADDRESS\_HEXA\_KEYBOARD: Yêu cầu đọc mã phím đang bấm.
* Đọc mã phím vào a0.
* In mã phím
* Khôi phục các thanh ghi trong stack.
* uret quay lại chương trình chính.
* Cập nhật mã nguồn để hiển thị với tất cả 16 nút:

**Source Code:**

.eqv IN\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0012

.eqv OUT\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0014

.data

message: .asciz "Key scan code: "

.text

main:

la t0, handler

csrrs zero, utvec, t0

li t1, 0x100

csrrs zero, uie, t1

csrrsi zero, ustatus, 1

li t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li t3, 0x80

sb t3, 0(t1)

xor s0, s0, s0

loop:

addi s0, s0, 1

prn\_seg:

add a0, zero, s0

li a7, 1

ecall

li a0, '\n'

li a7, 11

ecall

sleep:

li a0, 300

li a7, 32

ecall

j loop

end\_main:

handler:

addi sp, sp, -16

sw a0, 0(sp)

sw a7, 4(sp)

sw t1, 8(sp)

sw t2, 12(sp)

li a7, 4

la a0, message

ecall

li t2, 0x80 # Dong dau tien, bit 7 = 1

scan\_rows:

li t1, IN\_ADDRESS\_HEXA\_KEYBOARD

sb t2, 0(t1) # Gui dong chon

li t1, OUT\_ADDRESS\_HEXA\_KEYBOARD

lb t3, 0(t1) # Doc du lieu tu cot

beq t3, zero, next\_row # Khong co phim thi quet dong khac

add a0, zero, t3

li a7, 34

ecall

li a7, 11

li a0, '\n'

ecall

next\_row:

srli t2, t2, 1 # Sang dong tiep theo

bnez t2, scan\_rows # Neu con dong thi tiep tuc quet

li t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li t3, 0x80

sb t3, 0(t1)

lw t2, 12(sp)

lw t1, 8(sp)

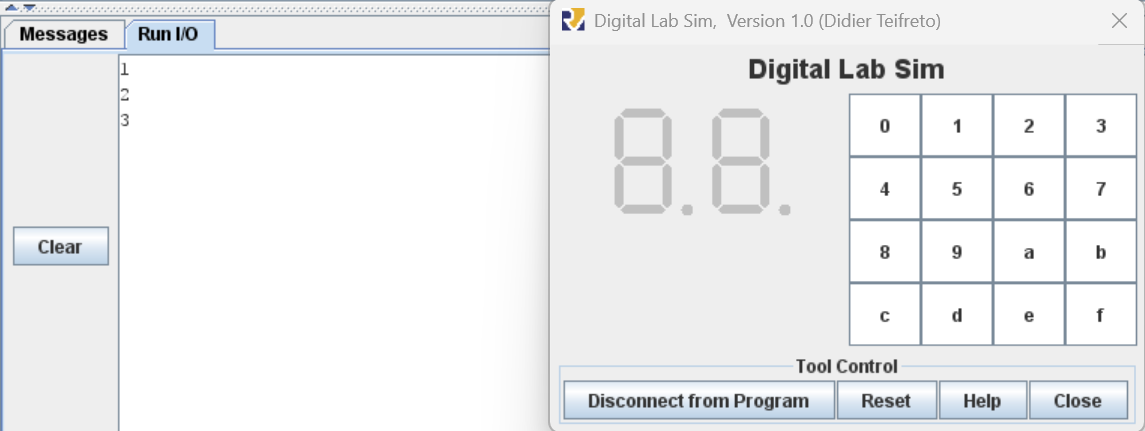
lw a7, 4(sp)

lw a0, 0(sp)

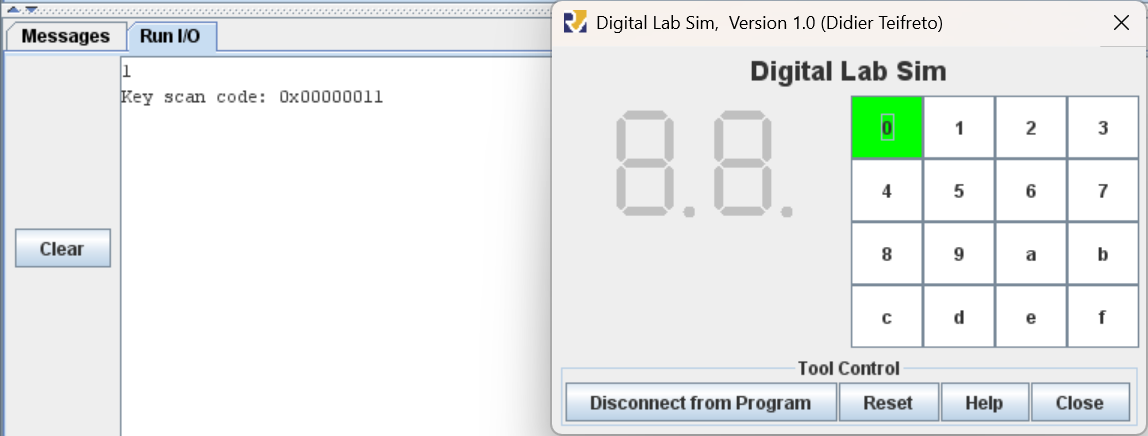
addi sp, sp, 16

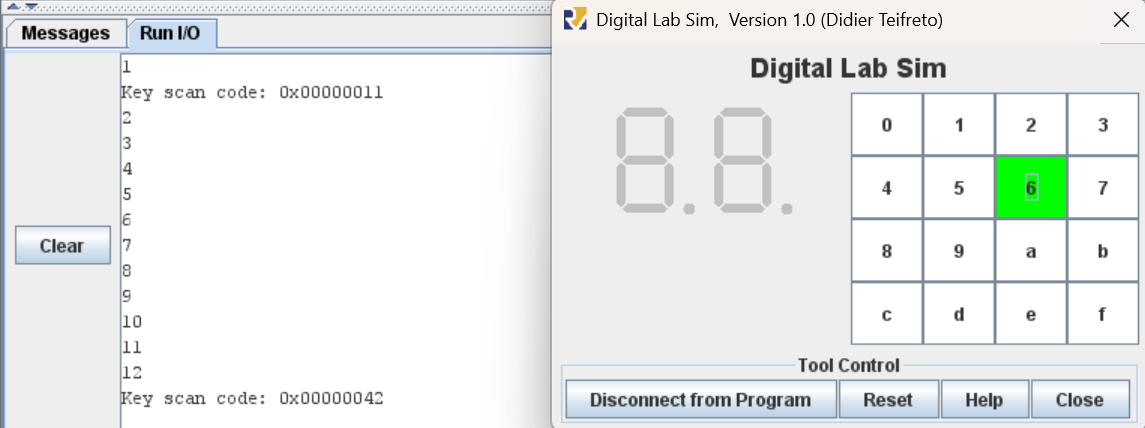
uret

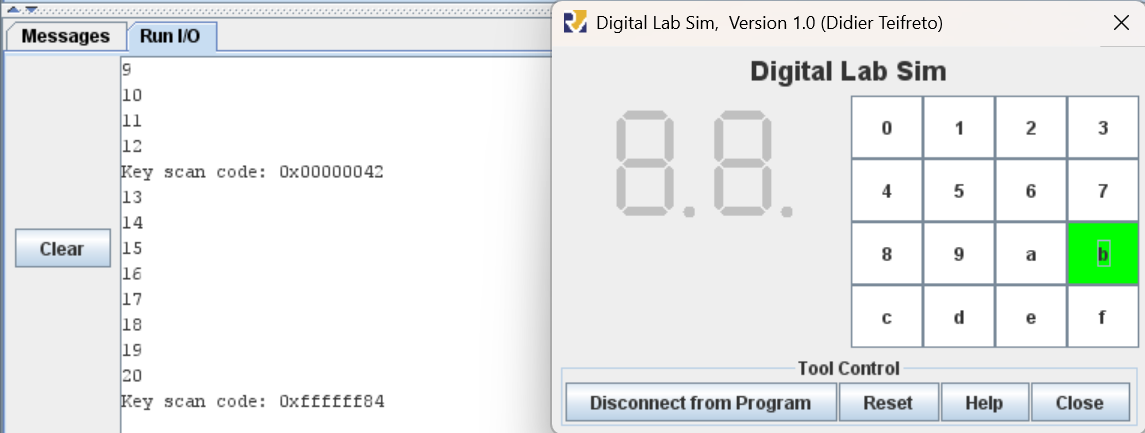
* Kết quả:



+ Ban đầu chưa nhấn nút nào (nút trắng) thì chương trình in ra chuỗi số nguyên liên tiếp bắt đầu từ 1.







+ Sau khi nhấn 1 nút bất kỳ (nút xanh) thì chương trình sẽ in ra chuỗi “Key scan code: “ cùng với mã hexa tương ứng vị trí của nút bấm ấy.

+ Sau khi in chuỗi thông báo, nếu không có động tác khác thì chương trình tiếp túc in ra chuỗi số nguyên liên tiếp.

**Additional Assignment:**

- Bắt sự kiện nhấn nút bàn phím Keypad 4x4 bằng ngắt.

- Chia Bitmap Display thành lưới 4x4.

- Khi nhấn nút nào trên keypad thì ô tương ứng với nút đó có màu đỏ, các ô khác có màu đen.

**Source Code:**

.eqv MONITOR\_SCREEN 0x10010000

.eqv RED 0x00FF0000

.eqv BLACK 0xFF000000

.eqv IN\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0012

.eqv OUT\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0014

.text

main:

addi sp, sp, -16

addi t0, x0, 1

addi t1, x0, 2

addi t2, x0, 4

addi t3, x0, 8

sw t0, 0(sp)

sw t1, 4(sp)

sw t2, 8(sp)

sw t3, 12(sp)

li s10, MONITOR\_SCREEN # Luu dia chi pixel dc to

la t0, handler # Lay dia chi ham handler

csrrs zero, utvec, t0

li t1, 0x100 # Bit 8 = 1 (0x100)

csrrs zero, uie, t1 # Thiet lap bit UEIE trong thanh ghi uie

csrrsi zero, ustatus, 1 # Bit 0 trong ustatus

li t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li t3, 0x80 # Bit 7 = 1

sb t3, 0(t1) # Kich hoat ngat

loop:

j loop

end\_main:

handler:

li s9, BLACK

sw s9, 0(s10) # To den o cu

get\_key\_code:

li t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li t2, 0x80 # Bat lai bit 7 de doc hang

li t3, 0

li t4, 0x01 # Hang dau tien (bit 0)

li t5, 2

read:

add t6, t2, t4 # Ket hop dong can doc voi bat ngat

sb t6, 0(t1) # Doc hang moi

mul t4, t4, t5 # Hang tiep theo

li s0, OUT\_ADDRESS\_HEXA\_KEYBOARD

lbu a0, 0(s0) # Doc byte 8 bit

addi s9, a0, 0 # Luu ma

beq a0, zero, read # Neu chua bam phim nao, doc lai

li t6, MONITOR\_SCREEN

li t5, 4 # 4 o moi hang

addi a2, sp, 0 # Tro den gia tri hang

jal row\_col # Tim chi so hang

mul t1, t1, t5 # row \* 4

add t6, t6, t1 # + dia chi man hinh

srli a0, a0, 4 # Gia tri cot

jal row\_col

add t6, t6, t1 # Hang + Cot

li s2, RED

sw s2, 0(t6) # To do

addi s10, t6, 0 # Cap nhat vi tri o mau hien tai

uret # Tra quyen dieu khien

row\_col:

andi t0, a0, 0xf # Giu lai 4 bit cuoi

addi t1, x0, 0 # i = 0

for:

add a3, a2, t1 # Tro toi phan tu mang i

lw t3, 0(a3) # Gia tri tai i

beq t3, t0, end\_rc # Neu ma phim dang xu ly thi dung

addi t1, t1, 4 # Tang i

j for

end\_rc:

jr ra

* Kết quả:

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.