Thực hành kiến trúc máy tính

Báo cáo thực hành

Bài 11. Lập trình xử lý ngắt

|  |  |
| --- | --- |
| Họ Tên | Lê Thành An |
| MSSV | 20235631 |

**ASSIGNMENT 1**

ĐOẠN MÃ :

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

# col 0x1 col 0x2 col 0x4 col 0x8

# row 0x1 0 1 2 3

# 0x11 0x21 0x41 0x81

# row 0x2 4 5 6 7

# 0x12 0x22 0x42 0x82

# row 0x4 8 9 a b

# 0x14 0x24 0x44 0x84

# row 0x8 c d e f

# 0x18 0x28 0x48 0x88

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

# Command row number of hexadecimal keyboard (bit 0 to 3)

# Eg. assign 0x1, to get key button 0,1,2,3

# assign 0x2, to get key button 4,5,6,7

# NOTE must reassign value for this address before reading,

# eventhough you only want to scan 1 row

.eqv IN\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0012

# Receive row and column of the key pressed, 0 if not key pressed

# Eg. equal 0x11, means that key button 0 pressed.

# Eg. equal 0x28, means that key button D pressed.

.eqv OUT\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0014

.data

A: .asciz "\n"

.text

main:

li t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li t2, OUT\_ADDRESS\_HEXA\_KEYBOARD

li t3, 0x01 # start with row 1 (0x01)

polling:

print:

# Check current row

sb t3, 0(t1)

lb a0, 0(t2)

# set the row to scan

# read scan code of key button

# Only print if a key is pressed (a0 != 0)

beqz a0, slow\_down # if no key pressed, skip print

li a7, 34

ecall

# print integer (hexa)

# Print space for readability

li a0, 32

li a7, 11

ecall

slow\_down:

li a7, 4

la a0, A

ecall

next\_row:

# sleep 300ms - longer delay to keep execution speed under control

# Rotate through rows in sequence: 0x01 -> 0x02 -> 0x04 -> 0x08 -> 0x01

li t4, 0x01

beq t3, t4, set\_row\_2

li t4, 0x02

beq t3, t4, set\_row\_4

li t4, 0x04

beq t3, t4, set\_row\_8

li t4, 0x08

beq t3, t4, set\_row\_1

# Fallback (shouldn't reach here)

li t3, 0x01

j polling

set\_row\_1:

li t3, 0x01

j polling

set\_row\_2:

li t3, 0x02

j polling

set\_row\_4:

li t3, 0x04

j polling

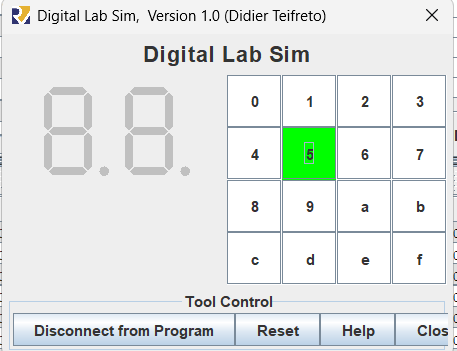
set\_row\_8:

li t3, 0x08

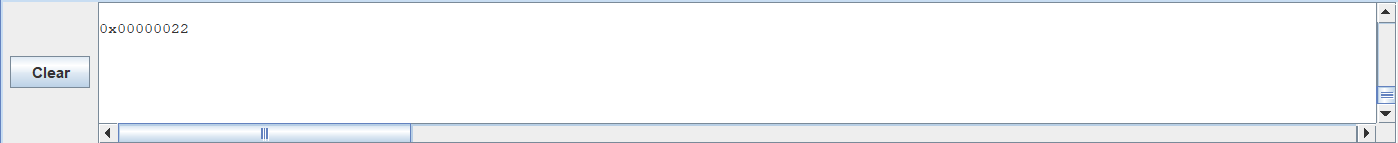
j polling

Kết quả:

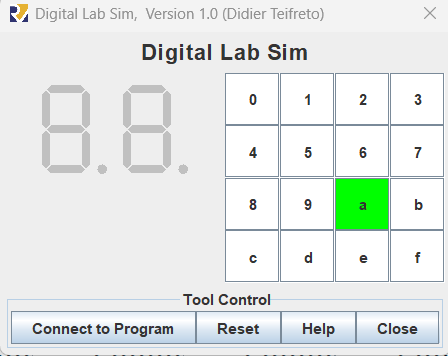
Ấn 5



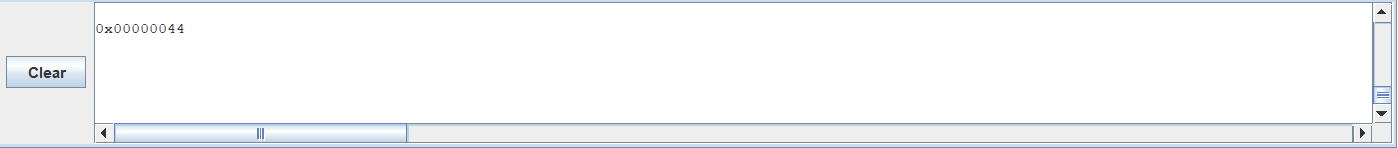
Kết quả là:



Ấn a:



Kết quả là:



**ASSIGNMENT 2**

ĐOẠN MÃ :

.eqv IN\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0012

.data

message: .asciz "Someone's presed a button.\n"

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

# MAIN Procedure

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

.text

main:

# Load the interrupt service routine address to the UTVEC register

la t0, handler

csrrs zero, utvec, t0

# Set the UEIE (User External Interrupt Enable) bit in UIE register

li t1, 0x100

csrrs zero, uie, t1 # uie - ueie bit (bit 8)

# Set the UIE (User Interrupt Enable) bit in USTATUS register

csrrsi zero, ustatus, 1 # ustatus - enable uie (bit 0)

# Enable the interrupt of keypad of Digital Lab Sim

li t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li t3, 0x80 # bit 7 = 1 to enable interrupt

sb t3, 0(t1)

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

# No-end loop, main program, to demo the effective of interrupt

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

loop:

nop

nop

nop

j loop

end\_main:

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

# Interrupt service routine

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

handler:

# ebreak # Can pause the execution to observe registers

# Saves the context

addi sp, sp, -8

sw a0, 0(sp)

sw a7, 4(sp)

# Handles the interrupt

# Shows message in Run I/O

li a7, 4

la a0, message

ecall

# Restores the context

lw a7, 4(sp)

lw a0, 0(sp)

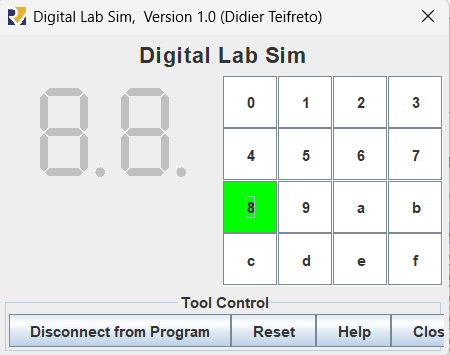
addi sp, sp, 8

# Back to the main procedure

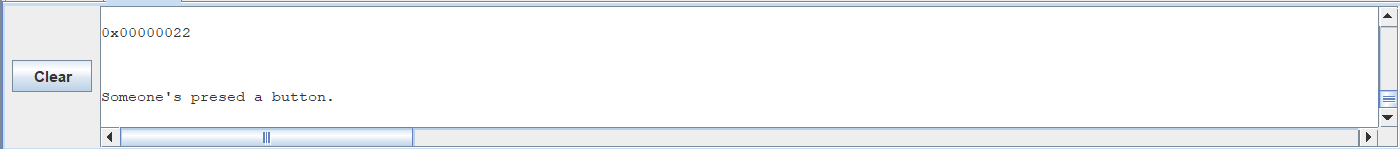
uret

Kết quả:

Ấn 1 nút bất kì:



Kết quả là :



Thanh ghi:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Thời điểm** | **utvec** | **uie** | **ustatus** | **PC (giả định)** | **Ghi chú** |
| Trước cấu hình ngắt | Không xác định | 0 | 0 | main | Bắt đầu chương trình |
| Sau csrrs utvec, t0 | handler addr | 0 | 0 | tiếp tục dòng tiếp theo | Gán địa chỉ trình xử lý ngắt |
| Sau csrrs uie, t1 | Không đổi | 0x100 | 0 | tiếp tục | Mở ngắt ngoài người dùng |
| Sau csrrsi ustatus, 1 | Không đổi | 0x100 | 0x1 | tiếp tục | Cho phép tiếp nhận ngắt |
| Trước khi có ngắt | Không đổi | 0x100 | 0x1 | loop (vị trí j loop) | Đang thực hiện vòng lặp vô tận |
| Khi có ngắt xảy ra | Không đổi | 0x100 | 0x1 | handler addr | PC nhảy đến handler |
| Sau uret | Không đổi | 0x100 | 0x1 | trở về loop | PC quay lại tiếp tục thực thi |

ĐOẠN MÃ :

.eqv IN\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0012

.eqv OUT\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0014

.data

message: .asciz "Key scan code: "

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

# MAIN Procedure

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

.text

main:

# Load the interrupt service routine address to the UTVEC register

la t0, handler

csrrs zero, utvec, t0

# Set the UEIE (User External Interrupt Enable) bit in UIE register

li t1, 0x100

csrrs zero, uie, t1 # uie - ueie bit (bit

# Set the UIE (User Interrupt Enable) bit in USTATUS register

csrrsi zero, ustatus, 1 # ustatus - enable uie (bit 0)

# Enable the interrupt of keypad of Digital Lab Sim

li t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li t3, 0x80 # bit 7 = 1 to enable interrupt

sb t3, 0(t1)

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

# Loop to print a sequence numbers

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

xor s0, s0, s0 # count = s0 = 0

loop:

addi s0, s0, 1 # count = count + 1

prn\_seq:

addi a7, zero, 1

add a0, s0, zero # Print auto sequence number

ecall

addi a7, zero, 11

li a0, '\n' # Print EOL

ecall

sleep:

addi a7, zero, 32

li a0, 300 # Sleep 300 ms

ecall

j loop

end\_main:

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

# Interrupt service routine

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

handler:# Saves the context

addi sp, sp, -16

sw a0, 0(sp)

sw a7, 4(sp)

sw t1, 8(sp)

sw t2, 12(sp)

# Handles the interrupt

prn\_msg:

addi a7, zero, 4

la a0, message

ecall

get\_key\_code:

li t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li a0, 0

check\_row\_1:

li t2, 0x81 # Check row 4 and re-enable bit 7

sb t2, 0(t1) # Must reassign expected row

li t1, OUT\_ADDRESS\_HEXA\_KEYBOARD

lb a0, 0(t1)

bne a0, zero, prn\_key\_code

check\_row\_2:

li t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li t2, 0x82

sb t2, 0(t1)

li t1, OUT\_ADDRESS\_HEXA\_KEYBOARD

lb a0, 0(t1)

bne a0, zero, prn\_key\_code

check\_row\_3:

li t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li t2, 0x84

sb t2, 0(t1)

li t1, OUT\_ADDRESS\_HEXA\_KEYBOARD

lb a0, 0(t1)

bne a0, zero, prn\_key\_code

check\_row\_4:

li t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li t2, 0x88

sb t2, 0(t1)

li t1, OUT\_ADDRESS\_HEXA\_KEYBOARD

lb a0, 0(t1)

bne a0, zero, prn\_key\_code

prn\_key\_code:

li a7, 34

ecall

li a7, 11

li a0, '\n' # Print EOL

ecall

# Restores the context

lw t2, 12(sp)

lw t1, 8(sp)

lw a7, 4(sp)

lw a0, 0(sp)

addi sp, sp, 16

# Back to the main procedure

uret

Kết quả:

ấn 5

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, số

Nội dung do AI tạo ra có thể không chính xác.

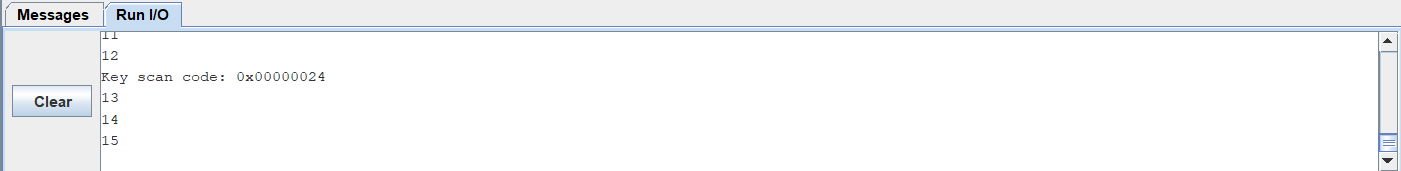
Ảnh có chứa ảnh chụp màn hình, văn bản, hàng

Nội dung do AI tạo ra có thể không chính xác.

ấn 9

Ảnh có chứa văn bản, ảnh chụp màn hình, số, biểu đồ

Nội dung do AI tạo ra có thể không chính xác.



ấn 8

Ảnh có chứa văn bản, ảnh chụp màn hình, số, biểu đồ

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa ảnh chụp màn hình, văn bản, hàng

Nội dung do AI tạo ra có thể không chính xác.

Thanh ghi:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Thời điểm** | **utvec** | **uie** | **ustatus** | **PC** | **Ghi chú** |
| Trước khi chạy main | Chưa xác định | 0 | 0 | entry point | Các ngắt chưa được bật |
| Sau csrrs utvec, t0 | handler addr | 0 | 0 | sau dòng đó | Gán địa chỉ trình phục vụ ngắt |
| Sau csrrs uie, t1 | Giữ nguyên | 0x100 | 0 | tiếp tục | Bật ngắt người dùng ngoài |
| Sau csrrsi ustatus,1 | Giữ nguyên | 0x100 | 0x1 | tiếp tục | Cho phép nhận ngắt trong u-mode |
| Khi xảy ra ngắt | không đổi | không đổi | không đổi | nhảy tới handler | PC nhảy tới địa chỉ handler |
| Sau uret | không đổi | không đổi | không đổi | trở về PC cũ | Tiếp tục vòng lặp từ vị trí bị gián đoạn |

**ASSIGNMENT Bổ sung**

ĐOẠN MÃ :

.eqv IN\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0012 # Address of keyboard

.eqv OUT\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0014 # Output address of keyboard

.eqv MONITOR\_SCREEN 0x10010000 # Start address of display memory

# Bitmap display settings: 32x32, 128x128 pixels

# Each cell will be 8x8 pixels (2x2 words)

# Color constants

.eqv RED 0x00FF0000

.data

message: .asciz "Key scan code: \n"

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

# MAIN Procedure

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

.text

li s10, MONITOR\_SCREEN

main:

# Load the interrupt service routine address to the UTVEC register

la t0, handler

csrrs zero, utvec, t0

# Set the UEIE (User External Interrupt Enable) bit in UIE register

li t1, 0x100

csrrs zero, uie, t1 # uie - ueie bit (bit 8)

# Set the UIE (User Interrupt Enable) bit in USTATUS register

csrrsi zero, ustatus, 1 # ustatus - enable uie (bit 0)

# Enable the interrupt of keypad of Digital Lab Sim

li t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li t3, 0x80 # bit 7 = 1 to enable interrupt

sb t3, 0(t1)

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

# No-end loop, main program

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

loop:

nop

nop

nop

j loop

end\_main:

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

# Draw a colored cell on the bitmap display based on the key code

# a0 - contains the key scan code

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

draw\_cell:

# Save the context

addi sp, sp, -16

sw ra, 0(sp)

sw s0, 4(sp)

sw s1, 8(sp)

sw s2, 12(sp)

# Map key scan code to grid position based on provided data:

# Row 1: 0x11(0), 0x21(1), 0x41(2), 0x81(3)

# Row 2: 0x12(4), 0x22(5), 0x42(6), 0x82(7)

# Row 3: 0x14(8), 0x24(9), 0x44(10), 0x84(11)

# Row 4: 0x18(12), 0x28(13), 0x48(14), 0x88(15)

# Calculate position on grid

li s0, -1 # Default cell number

# Row 1

li t0, 0x11

beq a0, t0, cell\_0

li t0, 0x21

beq a0, t0, cell\_1

li t0, 0x41

beq a0, t0, cell\_2

li t0, 0x81

beq a0, t0, cell\_3

# Row 2

li t0, 0x12

beq a0, t0, cell\_4

li t0, 0x22

beq a0, t0, cell\_5

li t0, 0x42

beq a0, t0, cell\_6

li t0, 0x82

beq a0, t0, cell\_7

# Row 3

li t0, 0x14

beq a0, t0, cell\_8

li t0, 0x24

beq a0, t0, cell\_9

li t0, 0x44

beq a0, t0, cell\_10

li t0, 0x84

beq a0, t0, cell\_11

# Row 4

li t0, 0x18

beq a0, t0, cell\_12

li t0, 0x28

beq a0, t0, cell\_13

li t0, 0x48

beq a0, t0, cell\_14

li t0, 0x88

beq a0, t0, cell\_15

j draw\_exit # Invalid key code

cell\_0:

li s9, RED

sw s9, 0(s10)

j draw\_exit # Return after drawing the cell

cell\_1:

li s9, RED

sw s9, 4(s10)

j draw\_exit # Return after drawing the cell

cell\_2:

li s9, RED

sw s9, 8(s10)

j draw\_exit # Return after drawing the cell

cell\_3:

li s9, RED

sw s9, 12(s10)

j draw\_exit # Return after drawing the cell

cell\_4:

li s9, RED

sw s9, 16(s10)

j draw\_exit # Return after drawing the cell

cell\_5:

li s9, RED

sw s9, 20(s10)

j draw\_exit # Return after drawing the cell

cell\_6:

li s9, RED

sw s9, 24(s10)

j draw\_exit # Return after drawing the cell

cell\_7:

li s9, RED

sw s9, 28(s10)

j draw\_exit # Return after drawing the cell

cell\_8:

li s9, RED

sw s9, 32(s10)

j draw\_exit # Return after drawing the cell

cell\_9:

li s9, RED

sw s9, 36(s10)

j draw\_exit # Return after drawing the cell

cell\_10:

li s9, RED

sw s9, 40(s10)

j draw\_exit # Return after drawing the cell

cell\_11:

li s9, RED

sw s9, 44(s10)

j draw\_exit # Return after drawing the cell

cell\_12:

li s9, RED

sw s9, 48(s10)

j draw\_exit # Return after drawing the cell

cell\_13:

li s9, RED

sw s9, 52(s10)

j draw\_exit # Return after drawing the cell

cell\_14:

li s9, RED

sw s9, 56(s10)

j draw\_exit # Return after drawing the cell

cell\_15:

li s9, RED

sw s9, 60(s10)

j draw\_exit # Return after drawing the cell

draw\_exit:

# Restore the context

lw ra, 0(sp)

lw s0, 4(sp)

lw s1, 8(sp)

lw s2, 12(sp)

addi sp, sp, 16

jr ra

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

# Interrupt service routine

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

handler:

# Save the context

addi sp, sp, -24

sw ra, 0(sp)

sw a0, 4(sp)

sw a7, 8(sp)

sw t0, 12(sp)

sw t1, 16(sp)

sw t2, 20(sp)

# Handle the interrupt

# Print message

li a7, 4

la a0, message

ecall

# Get key scan code

li t0, IN\_ADDRESS\_HEXA\_KEYBOARD

li t2, 0x81 # Check row 1 (0x1) and re-enable interrupt (0x80)

sb t2, 0(t0)

li t0, OUT\_ADDRESS\_HEXA\_KEYBOARD

lb t1, 0(t0)

bnez t1, key\_found

li t0, IN\_ADDRESS\_HEXA\_KEYBOARD

li t2, 0x82 # Check row 2 (0x2) and re-enable interrupt (0x80)

sb t2, 0(t0)

li t0, OUT\_ADDRESS\_HEXA\_KEYBOARD

lb t1, 0(t0)

bnez t1, key\_found

li t0, IN\_ADDRESS\_HEXA\_KEYBOARD

li t2, 0x84 # Check row 3 (0x4) and re-enable interrupt (0x80)

sb t2, 0(t0)

li t0, OUT\_ADDRESS\_HEXA\_KEYBOARD

lb t1, 0(t0)

bnez t1, key\_found

li t0, IN\_ADDRESS\_HEXA\_KEYBOARD

li t2, 0x88 # Check row 4 (0x8) and re-enable interrupt (0x80)

sb t2, 0(t0)

li t0, OUT\_ADDRESS\_HEXA\_KEYBOARD

lb t1, 0(t0)

key\_found:

# Print key scan code in hex

mv a0, t1

li a7, 34 # Print in hex

ecall

# Print newline

li a7, 11

li a0, '\n'

ecall

# Draw the cell based on the key scan code

mv a0, t1

jal draw\_cell

# Restore the context

lw ra, 0(sp)

lw a0, 4(sp)

lw a7, 8(sp)

lw t0, 12(sp)

lw t1, 16(sp)

lw t2, 20(sp)

addi sp, sp, 24

# Return from the interrupt routine

uret

Kết quả:

Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, biểu đồ

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, hàng

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, biểu đồ

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa ảnh chụp màn hình, văn bản, biểu đồ, phần mềm

Nội dung do AI tạo ra có thể không chính xác.