**BÀI THỰC HÀNH TUẦN 12**

**KIẾN TRÚC MÁY TÍNH**

Họ và tên: Đinh Huy Dương

MSSV: 20215020

**Bài 1:**

.eqv IN\_ADRESS\_HEXA\_KEYBOARD 0xFFFF0012

.eqv OUT\_ADRESS\_HEXA\_KEYBOARD 0xFFFF0014

.text

main: li $t1, IN\_ADRESS\_HEXA\_KEYBOARD

li $t2, OUT\_ADRESS\_HEXA\_KEYBOARD

li $t3, 0x01 # check row 4 with key C, D,E, F

polling: sb $t3, 0($t1 ) # must reassign expected row

nop

lb $a0, 0($t2) # read scan code of key button

print: li $v0, 34 # print integer (hexa)

syscall

sleep: li $a0, 100 # sleep 1100ms

li $v0, 32

syscall

check\_key:

lb $t4,0($t2)

beq $t4,$0,update\_row

hold\_key:

j polling

update\_row:

sll $t3,$t3,1 # shift left logical by one bit to update the row value

beq $t3,16,reset\_row

nop

j polling

reset\_row:

li $t3,1

j polling

A screenshot of a computer

Description automatically generated with medium confidence

Phím 9 được bấm và được lặp lại nhờ hàm check\_key và hold\_key

**Bài 2:**

.eqv IN\_ADRESS\_HEXA\_KEYBOARD 0xFFFF0012

.data

Message: .asciiz "Oh my god. Someone's presed a button.\n"

temp: .asciiz "Chillin\n\n"

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

# MAIN Procedure

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

.text

main:

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

# Enable interrupts you expect

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

# Enable the interrupt of Keyboard matrix 4x4 of Digital Lab Sim

li $t1, IN\_ADRESS\_HEXA\_KEYBOARD

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

sb $t3, 0($t1)

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

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

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

Loop: nop

nop

li $v0,4

la $a0, temp

syscall

sleep: li $a0, 1100 # sleep 1100ms

li $v0, 32

syscall

nop

nop

b Loop # Wait for interrupt

end\_main:

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

# GENERAL INTERRUPT SERVED ROUTINE for all interrupts

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

.ktext 0x80000180

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

# Processing

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

IntSR: addi $v0, $zero, 4 # show message

la $a0, Message

syscall

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

# Evaluate the return address of main routine

# epc <= epc + 4

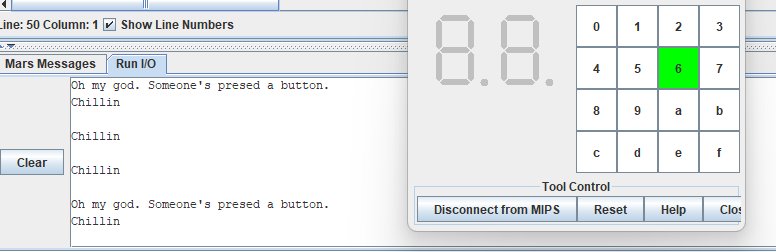
#--------------------------------------------------------

next\_pc: mfc0 $at, $14 # $at <= Coproc0.$14 = Coproc0.epc

addi $at, $at, 4 # $at = $at + 4 (next instruction)

mtc0 $at, $14 # Coproc0.$14 = Coproc0.epc <= $at

return: eret # Return from exception



Chèn thêm lệnh in ra màn hình 1 xâu để biểu diễn sự quay trở về main sau Interrupt (in ra xâu “Chillin”), và khi ấn 1 phím, ta hiện ra thông báo và quay lại main.

Thanh $pc sẽ mang vùng địa chỉ của lệnh ngắt:



Thanh $epc ở Coproc0 sẽ mang lại giá trị địa chỉ câu lệnh ở vùng main trước khi ngắt để quay trở về:

A screenshot of a computer

Description automatically generated with low confidence

**Bài 3:**

Tương tự như bài 2, thanh $pc sẽ mang địa chỉ của vùng lệnh ngắt tại 0x80000180

và $epc sẽ mang địa chỉ của câu lệnh trước khi ngắt:

A screenshot of a computer

Description automatically generated with low confidence

.eqv IN\_ADRESS\_HEXA\_KEYBOARD 0xFFFF0012

.eqv OUT\_ADRESS\_HEXA\_KEYBOARD 0xFFFF0014

.data

Message: .asciiz "\nKey scan code "

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

# MAIN Procedure

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

.text

main:

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

# Enable interrupts you expect

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

# Enable the interrupt of Keyboard matrix 4x4 of Digital Lab Sim

li $t1, IN\_ADRESS\_HEXA\_KEYBOARD

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

sb $t3, 0($t1)

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

# Loop an print sequence numbers

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

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

Loop: addi $s0, $s0, 1 # count = count + 1

prn\_seq: addi $v0,$zero,1

add $a0,$s0,$zero # print auto sequence number

syscall

prn\_eol: addi $v0,$zero,11

li $a0,'\n' # print endofline

syscall

sleep: addi $v0,$zero,32

li $a0,300 # sleep 300 ms

syscall

nop # WARNING: nop is mandatory here.

b Loop # Loop

end\_main:

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

# GENERAL INTERRUPT SERVED ROUTINE for all interrupts

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

.ktext 0x80000180

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

# SAVE the current REG FILE to stack

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

IntSR: addi $sp,$sp,4 # Save $ra because we may change it later

sw $ra,0($sp)

addi $sp,$sp,4 # Save $ra because we may change it later

sw $at,0($sp)

addi $sp,$sp,4 # Save $ra because we may change it later

sw $v0,0($sp)

addi $sp,$sp,4 # Save $a0, because we may change it later

sw $a0,0($sp)

addi $sp,$sp,4 # Save $t1, because we may change it later

sw $t1,0($sp)

addi $sp,$sp,4 # Save $t3, because we may change it later

sw $t3,0($sp)

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

# Processing

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

prn\_msg: addi $v0, $zero, 4

la $a0, Message

syscall

li $t3, 0x01 # check row 4 and re-enable bit 7

ori $t5,$t3,0x80

li $t1, IN\_ADRESS\_HEXA\_KEYBOARD

li $t2, OUT\_ADRESS\_HEXA\_KEYBOARD

get\_cod: sb $t5, 0($t1) # must reassign expected row

lb $a0, 0($t2)

update\_row:

sll $t3,$t3,1 # shift left logical by one bit to update the row value

ori $t5,$t3,0x80

beq $t3,32,reset\_row

nop

beq $a0,$0,get\_cod

j prn\_cod

reset\_row:

li $t3,1

j get\_cod

prn\_cod: li $v0,34

syscall

li $v0,11

li $a0,'\n' # print endofline

syscall

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

# Evaluate the return address of main routine

# epc <= epc + 4

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

next\_pc: mfc0 $at, $14 # $at <= Coproc0.$14 = Coproc0.epc

addi $at, $at, 4 # $at = $at + 4 (next instruction)

mtc0 $at, $14 # Coproc0.$14 = Coproc0.epc <= $at

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

# RESTORE the REG FILE from STACK

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

restore: lw $t3, 0($sp) # Restore the registers from stack

addi $sp,$sp,-4

lw $t1, 0($sp) # Restore the registers from stack

addi $sp,$sp,-4

lw $a0, 0($sp) # Restore the registers from stack

addi $sp,$sp,-4

lw $v0, 0($sp) # Restore the registers from stack

addi $sp,$sp,-4

lw $ra, 0($sp) # Restore the registers from stack

addi $sp,$sp,-4

return: eret # Return from exception

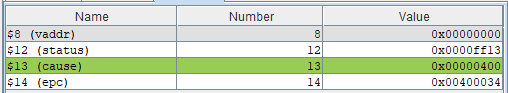
Ta sẽ lặp các hàng trong khi đang Interrupt để có thể lấy được giá trị đã có

A screenshot of a computer

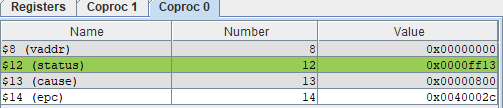
Description automatically generated

**Bài 4:**

Interrupt khi Time Interval:

****

Interrupt khi nhấn 1 phím:



**Bài 5:**

Khi bị ngắt mềm teq = 1:

A screenshot of a number

Description automatically generated with low confidence

Máy sẽ khởi động Interrupt mềm và thực hiện Decode hiển thị như sau:

**A screenshot of a computer

Description automatically generated**