Thực hành Kiến trúc máy tính

Week 11

Vũ Ngọc Đức – 20225816

Assignment 1

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

# 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

nextline: .asciiz "\n"

.text

main:

li $t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li $t2, OUT\_ADDRESS\_HEXA\_KEYBOARD

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

li $t4, 0x02 # check row 4 with key C, D, E, F

li $t5, 0x04 # check row 4 with key C, D, E, F

li $t6, 0x08 # check row 4 with key C, D, E, F

li $t0, 0

polling:

beq $t0, 100, end

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

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

bne $a0, $zero, print

sb $t4, 0($t1 ) # must reassign expected row

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

bne $a0, $zero, print

sb $t5, 0($t1 ) # must reassign expected row

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

bne $a0, $zero, print

sb $t6, 0($t1 ) # must reassign expected row

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

bne $a0, $zero, print

j next

print:

li $v0, 34 # print integer (hexa)

syscall

la $a0, nextline

li $v0, 4

syscall

next:

addi $t0, $t0, 1

sleep:

li $a0, 2000 # sleep 2s

li $v0, 32

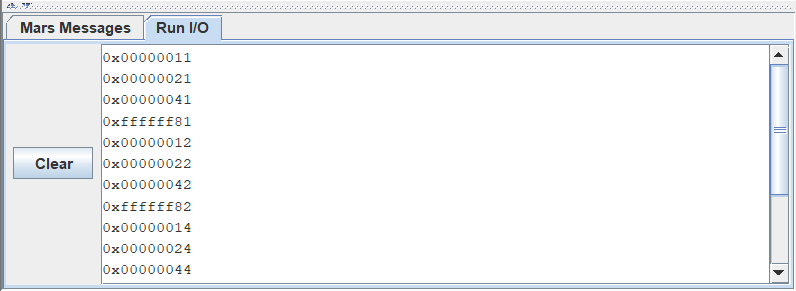
syscall

back\_to\_polling:

j polling # continue polling

end:

Nhập lần lượt từ 0 đến f trong Digital Lab Sim:



Ảnh có chứa văn bản, phần mềm, ảnh chụp màn hình, Biểu tượng máy tính

Mô tả được tạo tự động

Assignment 2

.eqv IN\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0012

.data

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

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

# MAIN Procedure

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

.text

main:

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

# Enable interrupts you expect

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

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

li $t1, IN\_ADDRESS\_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

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:

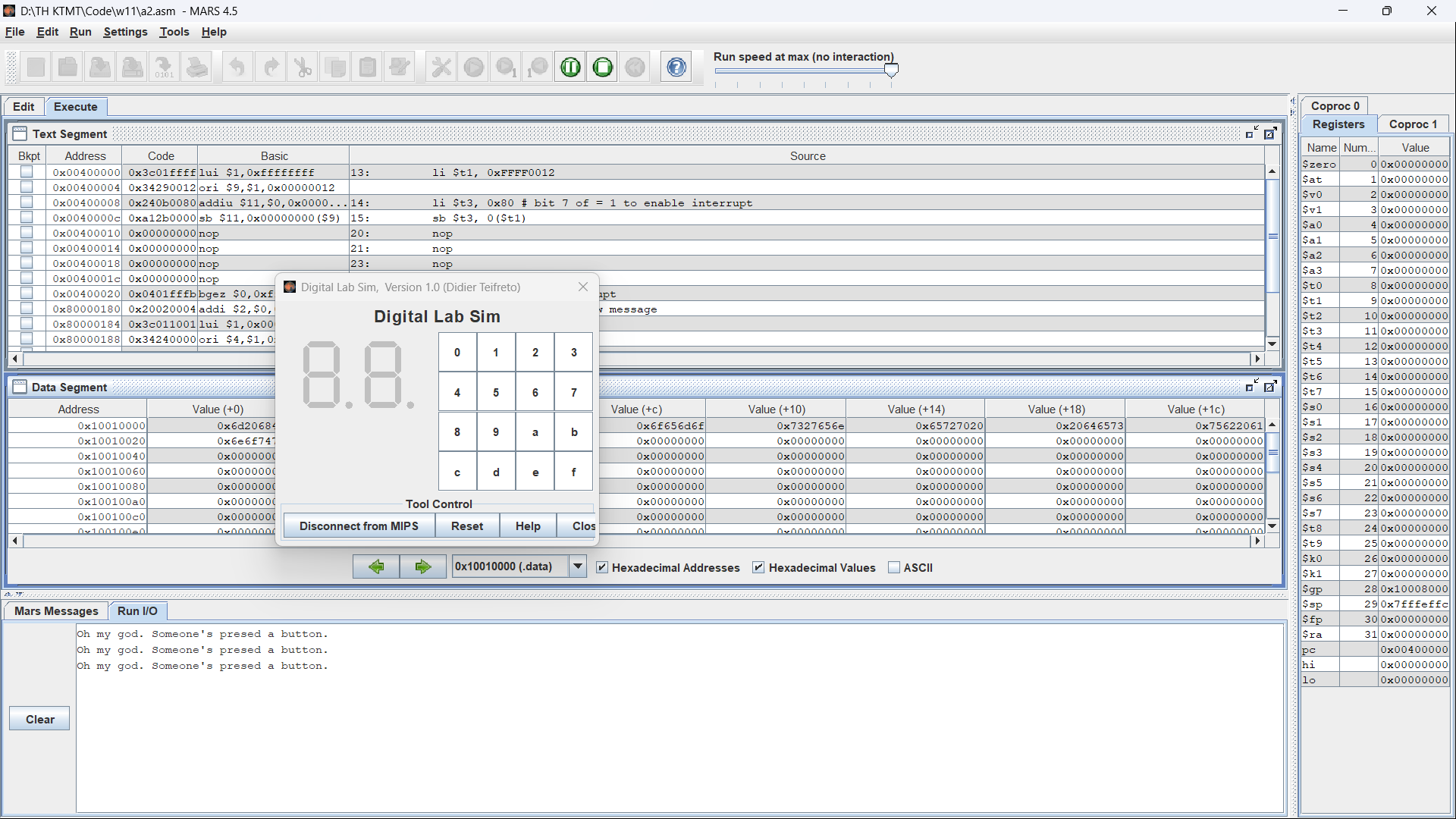
mfc0 $at, $14 # $at <= Coproc0.$14 = Coproc0.epc

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

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

return:

eret # Return from exception



Assignment 3

.eqv IN\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0012

.eqv OUT\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0014

.data

Message: .asciiz "Key scan code "

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

# MAIN Procedure

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

.text

main:

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

# Enable interrupts you expect

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

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

li $t1, IN\_ADDRESS\_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 $at because we may change it later

sw $at,0($sp)

addi $sp,$sp,4 # Save $sp 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

get\_cod:

li $t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li $t3, 0x81 # check row 4 and re-enable bit 7

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

li $t1, OUT\_ADDRESS\_HEXA\_KEYBOARD

lb $a0, 0($t1)

bne $a0, $zero, prn\_cod

li $t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li $t3, 0x82 # check row 4 and re-enable bit 7

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

li $t1, OUT\_ADDRESS\_HEXA\_KEYBOARD

lb $a0, 0($t1)

bne $a0, $zero, prn\_cod

li $t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li $t3, 0x84 # check row 4 and re-enable bit 7

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

li $t1, OUT\_ADDRESS\_HEXA\_KEYBOARD

lb $a0, 0($t1)

bne $a0, $zero, prn\_cod

li $t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li $t3, 0x88 # check row 4 and re-enable bit 7

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

li $t1, OUT\_ADDRESS\_HEXA\_KEYBOARD

lb $a0, 0($t1)

bne $a0, $zero, prn\_cod

prn\_cod:

li $v0, 34

syscall

li $v0, 11

li $a0, '\n' # print end of line

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 $at, 0($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

Nhập lần lượt 1, 5, 9, d được kết quả:

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

Mô tả được tạo tự động

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

Mô tả được tạo tự động

Assignment 4

.eqv IN\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0012

.eqv COUNTER 0xFFFF0013 # Time Counter

.eqv MASK\_CAUSE\_COUNTER 0x00000400 # Bit 10: Counter interrupt

.eqv MASK\_CAUSE\_KEYMATRIX 0x00000800 # Bit 11: Key matrix interrupt

.data

msg\_keypress: .asciiz "Someone has pressed a key!\n"

msg\_counter: .asciiz "Time inteval!\n"

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

# MAIN Procedure

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

.text

main:

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

# Enable interrupts you expect

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

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

li $t1, IN\_ADDRESS\_HEXA\_KEYBOARD

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

sb $t3, 0($t1)

# Enable the interrupt of TimeCounter of Digital Lab Sim

li $t1, COUNTER

sb $t1, 0($t1)

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

# Loop a print sequence numbers

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

Loop:

nop

nop

nop

sleep:

addi $v0,$zero,32 # BUG: must sleep to wait for Time Counter

li $a0,500 # sleep 500 ms

syscall

nop # WARNING: nop is mandatory here.

b Loop

end\_main:

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

# GENERAL INTERRUPT SERVED ROUTINE for all interrupts

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

.ktext 0x80000180

IntSR: #--------------------------------------------------------

# Temporary disable interrupt

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

dis\_int:

li $t1, COUNTER # BUG: must disable with Time Counter

sb $zero, 0($t1)

# no need to disable keyboard matrix interrupt

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

# Processing

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

get\_caus:

mfc0 $t1, $13 # $t1 = Coproc0.cause

IsCount:

li $t2, MASK\_CAUSE\_COUNTER# if Cause value confirm Counter..

and $at, $t1,$t2

beq $at,$t2, Counter\_Intr

IsKeyMa:

li $t2, MASK\_CAUSE\_KEYMATRIX # if Cause value confirm Key..

and $at, $t1,$t2

beq $at,$t2, Keymatrix\_Intr

others:

j end\_process # other cases

Keymatrix\_Intr:

li $v0, 4 # Processing Key Matrix Interrupt

la $a0, msg\_keypress

syscall

j end\_process

Counter\_Intr:

li $v0, 4 # Processing Counter Interrupt

la $a0, msg\_counter

syscall

j end\_process

end\_process:

mtc0 $zero, $13 # Must clear cause reg

en\_int: #--------------------------------------------------------

# Re-enable interrupt

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

li $t1, COUNTER

sb $t1, 0($t1)

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

# 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

Nhập lần lượt 1, 9 ,7 được kết quả:

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

Mô tả được tạo tự động

Assignment 5

.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

.eqv MASK\_CAUSE\_KEYBOARD 0x0000034 # Keyboard Cause

.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

beq $t1, $zero, WaitForKey # if $t1 == 0 then Polling

MakeIntR:

teqi $t1, 1 # if $t0 = 1 then raise an Interrupt

j loop

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

# Interrupt subroutine

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

.ktext 0x80000180

get\_caus:

mfc0 $t1, $13 # $t1 = Coproc0.cause

IsCount:

li $t2, MASK\_CAUSE\_KEYBOARD # if Cause value confirm Keyboard..

and $at, $t1,$t2

beq $at,$t2, Counter\_Keyboard

j end\_process

Counter\_Keyboard:

ReadKey:

lw $t0, 0($k0) # $t0 = [$k0] = KEY\_CODE

WaitForDis:

lw $t2, 0($s1) # $t2 = [$s1] = DISPLAY\_READY

beq $t2, $zero, WaitForDis # if $t2 == 0 then Polling

Encrypt:

addi $t0, $t0, 1 # change input key

ShowKey:

sw $t0, 0($s0) # show key

nop

end\_process:

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

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

Mô tả được tạo tự động