BÀI THỰC HÀNH TUẦN 12
KIẾN TRÚC MÁY TÍNH

Họ và tên: Đinh Huy Dương

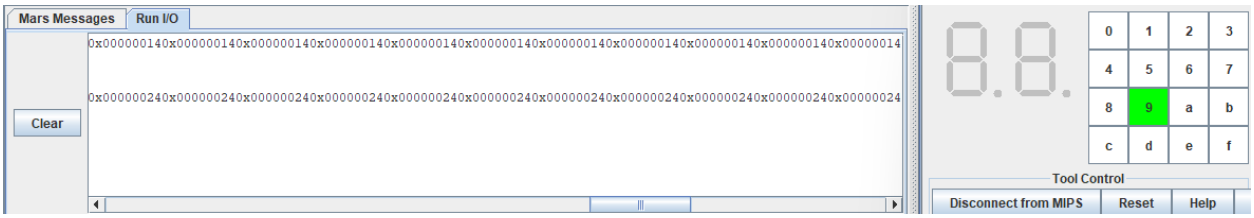
MSSV: 20215020

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
```



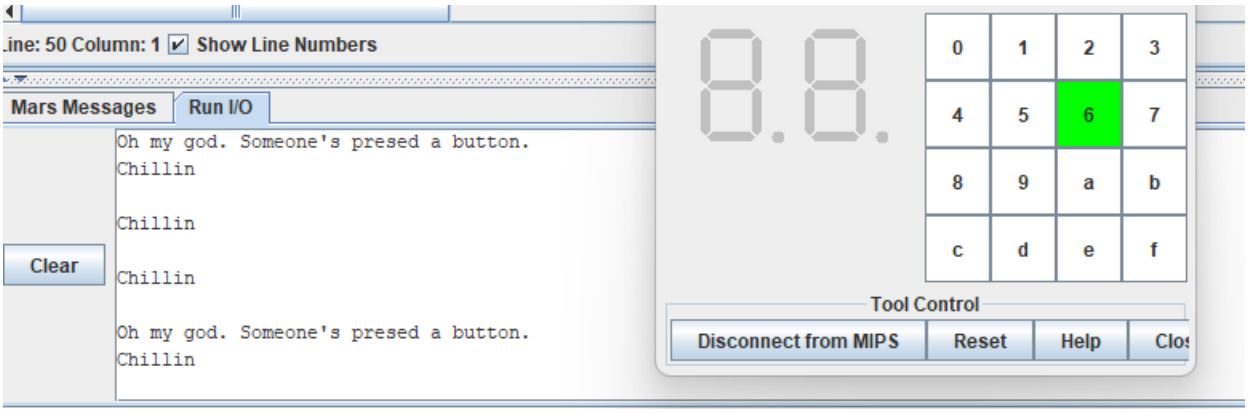
Phím 9 được bấm và được lặp lại nhờ hàm check_key và hold_key

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:

pc		0x80000180

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

Name	Number	Value
$8 (vaddr)	8	0x00000000
$12 (status)	12	0x0000ff13
$13 (cause)	13	0x00000800
$14 (epc)	14	0x00400034

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:

Name	Number	Value
$8 (vaddr)	8	0x00000000
$12 (status)	12	0x0000ff13
$13 (cause)	13	0x00000800
$14 (epc)	14	0x0040001c

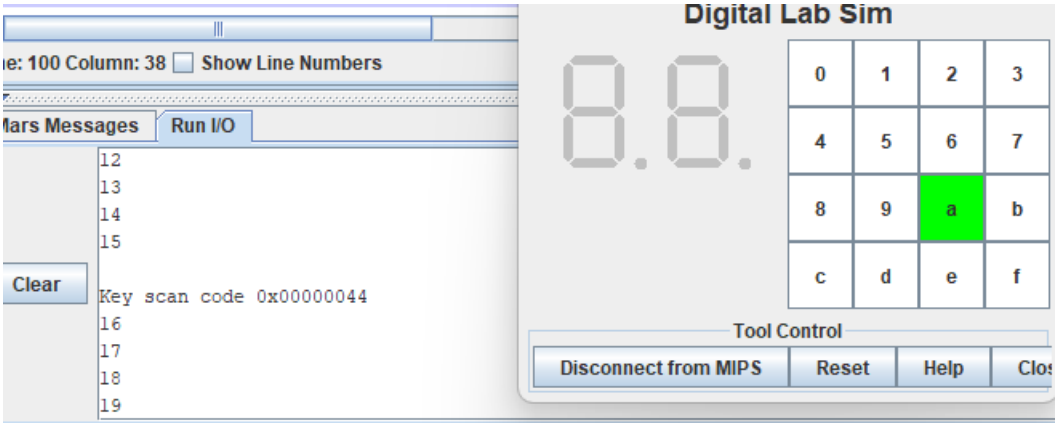
```
.eqv IN_ADRESS_HEXA_KEYBOARD 0xFFFF0012
.eqv OUT_ADRESS_HEXA_KEYBOARD 0xFFFF0014
.data
Message: .asciiz "\nKey scan code "
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# MAIN Procedure
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
.text
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



Bài 4:

Interrupt khi Time Interval:

Name	Number	Value
$8 (vaddr)	8	0x00000000
$12 (status)	12	0x0000ff13
$13 (cause)	13	0x00000400
$14 (epc)	14	0x00400034

Interrupt khi nhấn 1 phím:

Registers | Coproc 1 | Coproc 0

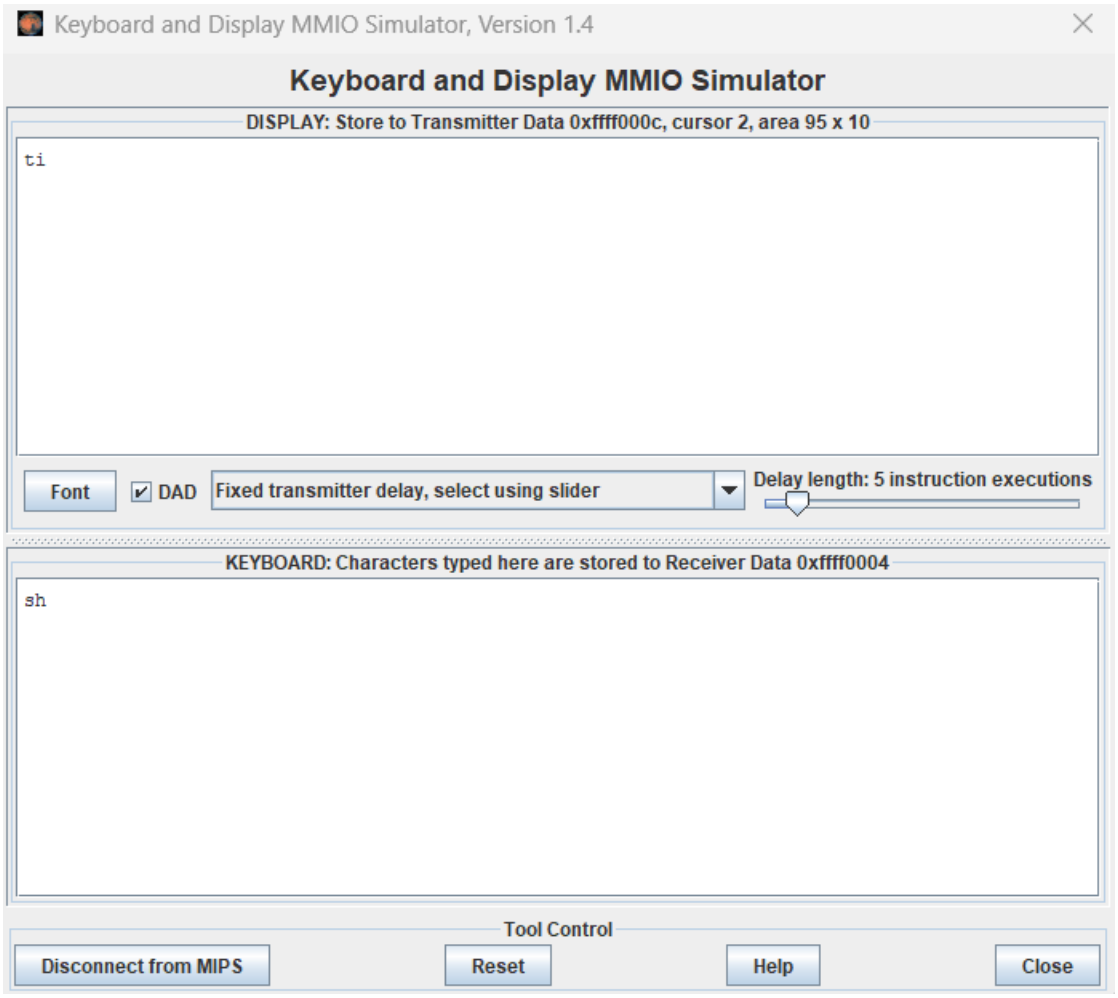
Name	Number	Value
$8 (vaddr)	8	0x00000000
$12 (status)	12	0x0000ff13
$13 (cause)	13	0x00000800
$14 (epc)	14	0x0040002c

Bài 5:

Khi bị ngắt mềm teq = 1:

Name	Number	Value
$8 (vaddr)	8	0x00000000
$12 (status)	12	0x0000ff13
$13 (cause)	13	0x00000034
$14 (epc)	14	0x0040002c

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


Keyboard and Display MMIO Simulator, Version 1.4
Keyboard and Display MMIO Simulator
DISPLAY: Store to Transmitter Data 0xffff000c, cursor 2, area 95 x 10
ti
Font
DAD
Fixed transmitter delay, select using slider
Delay length: 5 instruction executions
KEYBOARD: Characters typed here are stored to Receiver Data 0xffff0004
sh
Tool Control
Disconnect from MIPS
Reset
Help
Close