BÁO CÁO THỰC HÀNH KIẾN TRÚC MÁY TÍNH (IT3280) WEEK 11

Họ và tên: Trịnh Minh Đức

MSSV: 20225813

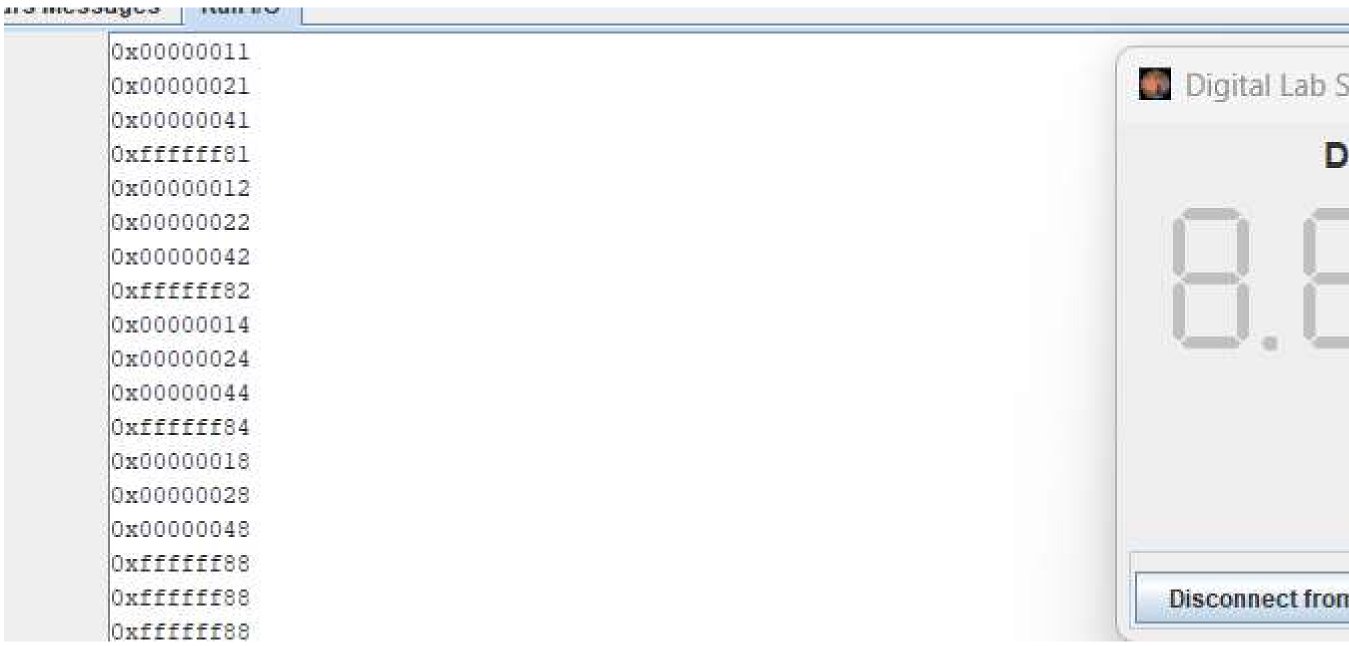
Assignment 1:

- Code:

```
.eqv IN_ADRESS_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_ADRESS_HEXA_KEYBOARD 0xFFFF0014
.data
nl: .asciiz "\n"
.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
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, exit
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  continue

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

 la $a0, nl
 li $v0, 4
 syscall
continue:
 addi  $t0, $t0, 1

sleep:
 li  $a0, 3000  # sleep 3s
 li  $v0, 32
 syscall
back_to_polling:
j  polling # continue polling
exit:
```

➔ Kết quả chạy : khi nhấn từ nút 0-> f kết quả được in ra màn hình như trên



```
0x00000011
0x00000021
0x00000041
0xffffff81
0x00000012
0x00000022
0x00000042
0xffffff82
0x00000014
0x00000024
0x00000044
0xffffff84
0x00000018
0x00000028
0x00000048
0xffffff88
0xffffff88
0xffffff88
```

Assignment 2:

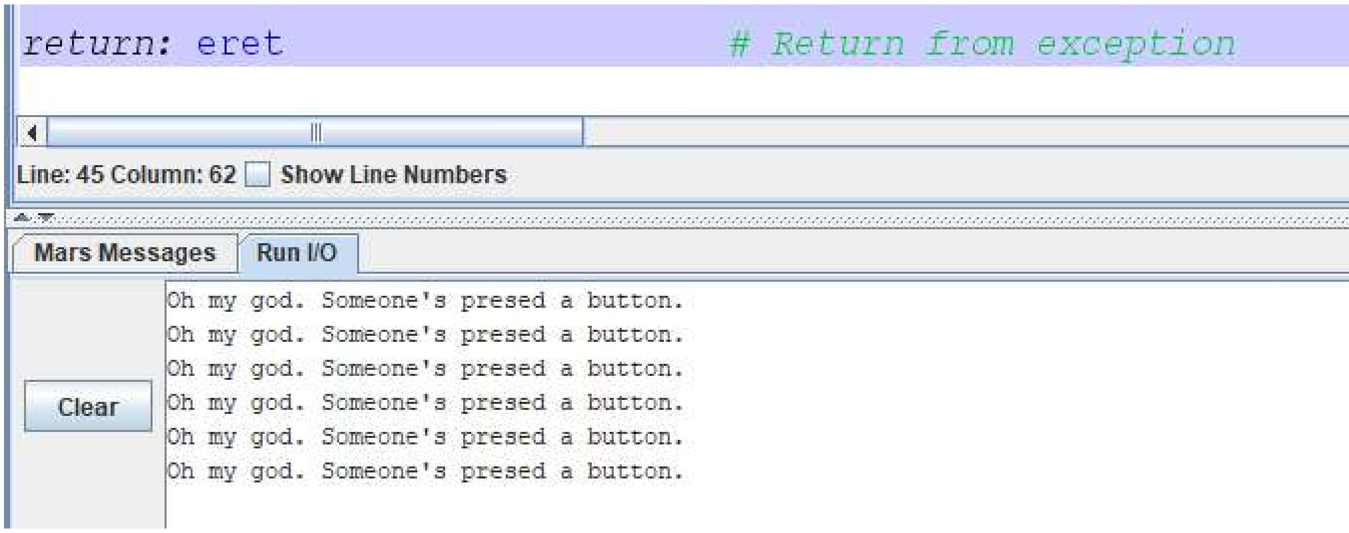
- Code:

```
.eqv IN_ADRESS_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_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
      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
```

➔ Kết quả chương trình đã bị ngắt khi nhấn vào các phím từ 0-> f



```
return: eret                    # Return from exception
```

Line: 45 Column: 62 Show Line Numbers

```
Oh my god. Someone's presed a button.
Oh my god. Someone's presed a button.
Oh my god. Someone's presed a button.
Oh my god. Someone's presed a button.
Oh my god. Someone's presed a button.
Oh my god. Someone's presed a button.
```

Assignment 3:

- Code:

```
.eqv IN_ADRESS_HEXA_KEYBOARD 0xFFFF0012
.eqv OUT_ADRESS_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_ADRESS_HEXA_KEYBOARD

li  $t3, 0x80

sb  $t3, 0($t1)


# bit 7 = 1 to enable

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

# Loop an print sequence numbers

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

xor  $s0, $s0, $s0

Loop:

addi  $s0, $s0, 1

prn_seq:
```

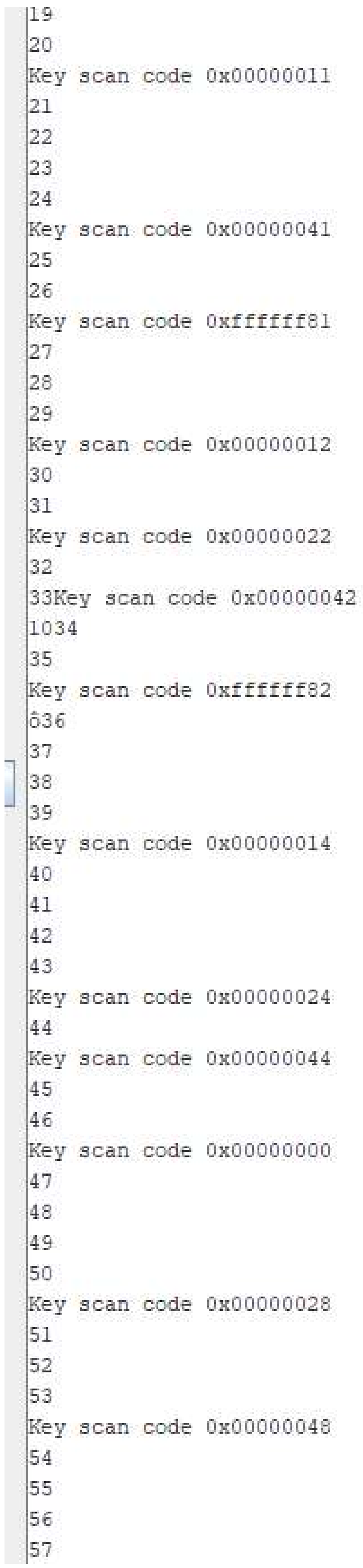
```
addi  $v0,$zero,1

add  $a0,$s0,$zero

syscall

prn_eol:

addi  $v0,$zero,11

li  $a0,'\n'

# count = $s0 = 0

# count = count + 1

# print auto sequence number

# print endofline

syscall

sleep:

addi  $v0,$zero,32

li  $a0,500

syscall

nop

b Loop

end_main:

# sleep 0,5 s

# WARNING: nop is mandatory here.

# Loop

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

# 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_ADRESS_HEXA_KEYBOARD

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

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

li $t1, OUT_ADRESS_HEXA_KEYBOARD

lb $a0, 0($t1)

bne  $a0, $zero, prn_cod

li  $t1, IN_ADRESS_HEXA_KEYBOARD

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

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

li  $t1, OUT_ADRESS_HEXA_KEYBOARD

lb  $a0, 0($t1)

bne  $a0, $zero, prn_cod

li $t1, IN_ADRESS_HEXA_KEYBOARD

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

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

li $t1, OUT_ADRESS_HEXA_KEYBOARD

lb $a0, 0($t1)

bne  $a0, $zero, prn_cod

li  $t1, IN_ADRESS_HEXA_KEYBOARD
li $t3, 0x88  # check row 4 and re-enable bit 7

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

li $t1, OUT_ADRESS_HEXA_KEYBOARD

lb $a0, 0($t1)

bne  $a0, $zero, prn_cod

prn_cod:

li  $v0,34

syscall


li $v0,11

li $a0,'\n'

syscall

# print endofline

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

# 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
lw  $ra, 0($sp)  # Restore the registers from stack
addi  $sp,$sp,-4
return:
eret    # Return from exception
```

➔ Kết quả khi nhấn lần lượt các phím chương trình ngắt quãng và tự tăng các số



```
19
20
Key scan code 0x00000011
21
22
23
24
Key scan code 0x00000041
25
26
Key scan code 0xffffff81
27
28
29
Key scan code 0x00000012
30
31
Key scan code 0x00000022
32
33Key scan code 0x00000042
1034
35
Key scan code 0xffffff82
ô36
37
38
39
Key scan code 0x00000014
40
41
42
43
Key scan code 0x00000024
44
Key scan code 0x00000044
45
46
Key scan code 0x00000000
47
48
49
50
Key scan code 0x00000028
51
52
53
Key scan code 0x00000048
54
55
56
57
```