Báo cáo thực hành kiến trúc máy tính tuần 11

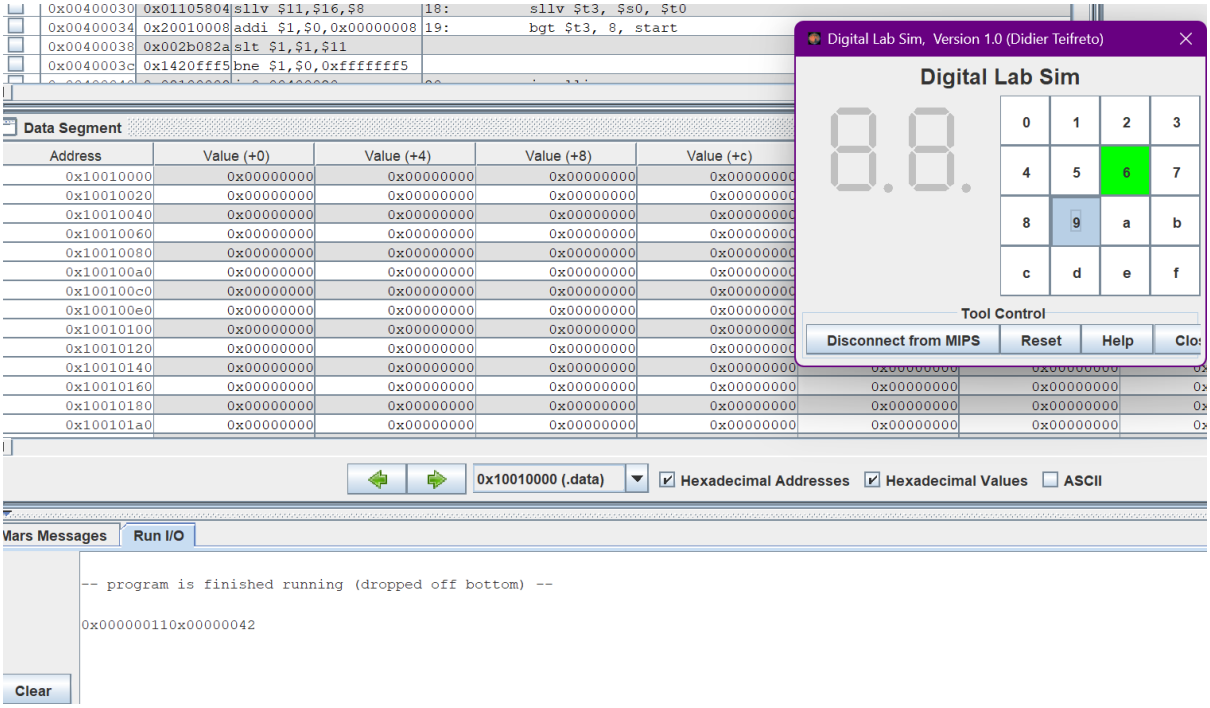
Phạm Thành Lập 20215076

Assign 1:

- Code

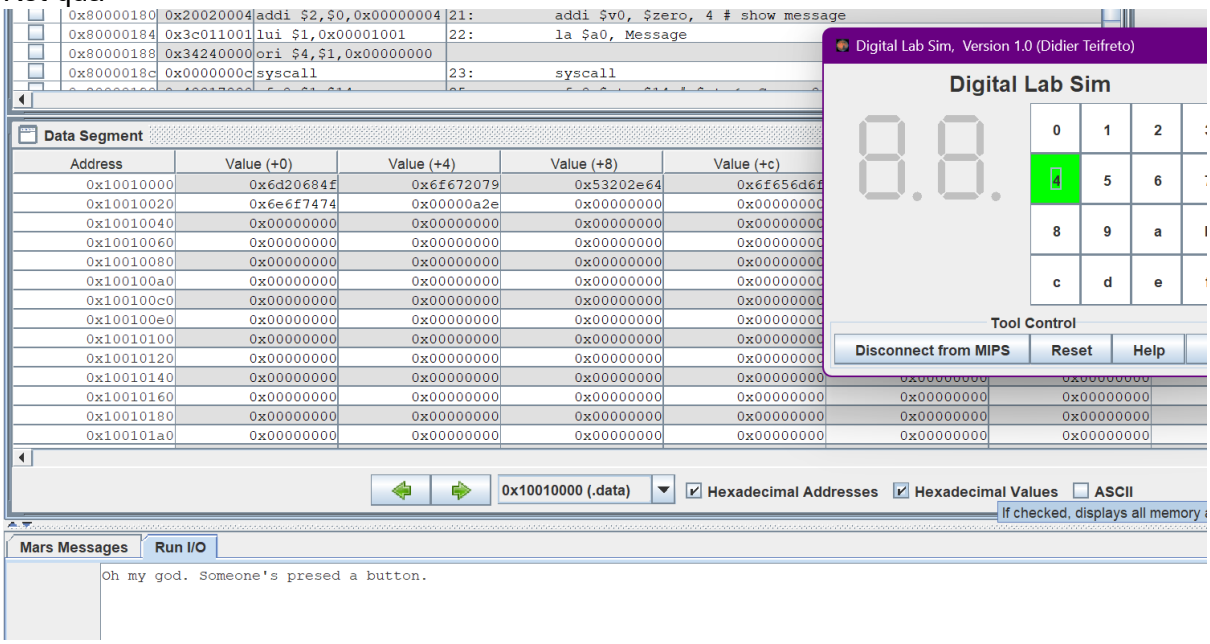
```
.eqv IN_ADDRESS_HEXA_KEYBOARD 0xFFFF0012
.eqv OUT_ADDRESS_HEXA_KEYBOARD 0xFFFF0014

.text
main:
        li $t1, IN_ADDRESS_HEXA_KEYBOARD
        li $t2, OUT_ADDRESS_HEXA_KEYBOARD
        addi $s1, $zero, 0
start:
        li $t3, 0x01 # check row 1
        addi $t0, $zero, 0
        addi $s0, $zero, 1
polling:
        sb $t3, 0($t1 ) # must reassign expected row
        lb $a0, 0($t2) # read scan code of key button
        bnez $a0, print
        addi $t0, $t0, 1
        sllv $t3, $s0, $t0
        bgt $t3, 8, start
        j polling
print:
        beq $a0, $s1, sleep
        li $v0, 34 # print integer (hexa)
        syscall
        add $s1, $zero, $a0
sleep:
        li $a0, 100 # sleep 100ms
        li $v0, 32
        syscall
back_to_polling:
        j start # continue polling
end:
        li $v0, 10
        syscall
```

- Result



Assign 2:

- Kết quả



Trạng thái	pc	epc
Trước khi ngắt	0x00400000	0x00000000
Khi ngắt xảy ra	0x80000180	0x00400000
Trước khi kết thúc chương trình con	0x8000019c	0x004000004

- Giải thích:

- Trước khi có Exception xảy ra, $pc bắt đầu với giá trị 0x00400000
  do chưa có Exception xảy ra nên giá trị của $14 epc vẫn là 0
- Khi có Exception xảy ra $pc nhảy đến địa chỉ cố định 0x80000180 (địa chỉ lỗi), và thanh ghi $14 (epc) được khởi tạo và bắt đầu với giá trị 0x00400000 bắt đầu thực hiện chương trình con khi Exception xảy ra
  trong chương trình con. Trong quá trình xảy ra, cả giá trị $pc và $14 epc đều đồng thời được tăng lên có nghĩa cả chương trình chính và chương trình Exception đều được thực hiện
- Trước khi kết thúc chương trình, $14 epc tăng thêm 4, và sau khi kết thúc thì $14 trở lại giá trị trước khi Exception xảy ra là 0
- Khi không có Exception xảy ra, trình biên dịch sẽ chỉ thực hiện chương trình chính như bình thường.

Assign 3:

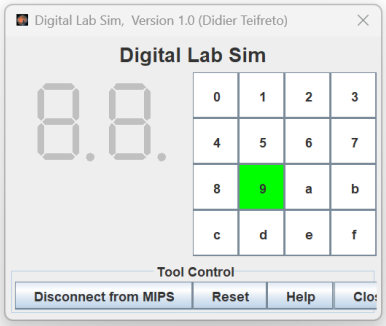
- Code

```
.eqv IN_ADDRESS_HEXA_KEYBOARD 0xFFFF0012
.eqv OUT_ADDRESS_HEXA_KEYBOARD 0xFFFF0014

.data
        Message: .asciiz "Key scan code "
.text
main:
        li $t1, IN_ADDRESS_HEXA_KEYBOARD
        li $t3, 0x80 # bit 7 = 1 to enable
        sb $t3, 0($t1)
        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:
.ktext 0x80000180
IntSR:
        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)
prn_msg:
        addi $v0, $zero, 4
        la $a0, Message
        syscall
start:
        li $t5, 0x80
        addi $t4, $zero, 1
get_cod:
        add $t3, $t5, $t4
        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)
        bnez $a0, prn_cod
        sll $t4, $t4, 1
        bgt $t4, 8, start
        j get_cod
prn_cod:
        li $v0,34
        syscall
        li $v0,11
        li $a0,'\n' # print end of line
        syscall
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:
        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
return:
        eret # Return from exception
```

- Result

```
1
2
3
4
5
6
7
8
Key scan code 0x00000014
9
10
11
12
13
14
15
16
17
18
19
Key scan code 0x00000018
20
21
22
23
24
25
26
27
28
Key scan code 0x00000024
29
30
31
32
33
34
```



Assign 4:

- Kết quả:

```
Time inteval!
Time inteval!
Time inteval!
Someone has pressed a key!
Time inteval!
Time inteval!
Time inteval!
Time inteval!
Time inteval!
```

1) Trường hợp 1: Khi ta có kết nối với Digital Lab Sim và có ấn nút:

-	Trước khi việc ngắt xảy ra: Thanh ghi $pc có giá trị 0x00400000, thanh ghi $14 (epc) có giá trị 0x00000000

-	Khi việc ngắt xảy ra: Thanh ghi $pc có giá trị 0x80000180, thanh ghi $14 (epc) có giá trị 0x00400000

-	Trước khi kết thúc chương trình con: Thanh ghi $pc có giá trị 0x800001f0, thanh ghi $14 (epc) có giá trị 0x00400004

2) Trường hợp 2: Khi ta có kết nối với Digital Lab Sim nhưng không ấn nút:

-	Trước khi việc ngắt xảy ra: Thanh ghi $pc có giá trị 0x0040002c, thanh ghi $14 (epc) có giá trị 0x00000000

-	Khi việc ngắt xảy ra: Thanh ghi $pc có giá trị 0x80000180, thanh ghi $14 (epc) có giá trị 0x0040002c

-	Trước khi kết thúc chương trình con: Thanh ghi $pc có giá trị 0x800001f0, thanh ghi $14 (epc) có giá trị 0x00400030

3) Trường hợp 3: Khi không kết nối với Digital Lab Sim: Sẽ bị kẹt trong vòng lặp vĩnh viễn, không có
Exception nào xảy ra
• Giải thích chương trình: Tương tự bài 2, khi ta có ấn nút từ bàn phím, ta sẽ thực hiện nhảy lệnh
ngay đến chương trình báo Exception đã ấn nút. Nếu không, chương trình sẽ thực hiện Exception
thời gian. Tuy vậy, trường hợp 1 và 2 có sự khác nhau về giá trị thanh ghi $14 (epc) do khi ta ấn
nút, Exception 1 xảy ra và sẽ nhảy ngay sau khi bắt đầu chương trình, lúc dấy thanh ghi $pc có giá
trị 0x0040000. Còn khi ta không ấn nút, chương trình chính sẽ chạy được 1 phần lệnh nhất định,
đến vòng lặp loop, rồi mới có Exception 2 xảy ra, và trước khi bắt đầu xảy ra, thanh ghi $pc có giá
trị 0x00400002c. Thanh ghi $14 (epc) lúc đó mới ghi chép giá trị của thanh ghi $pc và bắt đầu thực
hiện chương trình con.

Assign 5:

Kết quả của chương trình:

-	Trước khi việc ngắt xảy ra: Thanh ghi $pc có giá trị 0x0040002c, thanh ghi $14 (epc) có giá trị 0x00000000

-	Khi việc ngắt xảy ra: Thanh ghi $pc có giá trị 0x80000180, thanh ghi $14 (epc) có giá trị 0x0040002c

-	Trước khi kết thúc chương trình con: Thanh ghi $pc có giá trị 0x800001b8, thanh ghi $14 (epc) có giá trị 0x00400030