Báo cáo Thực Hành Kiến Trúc Máy Tính

Bài 4:

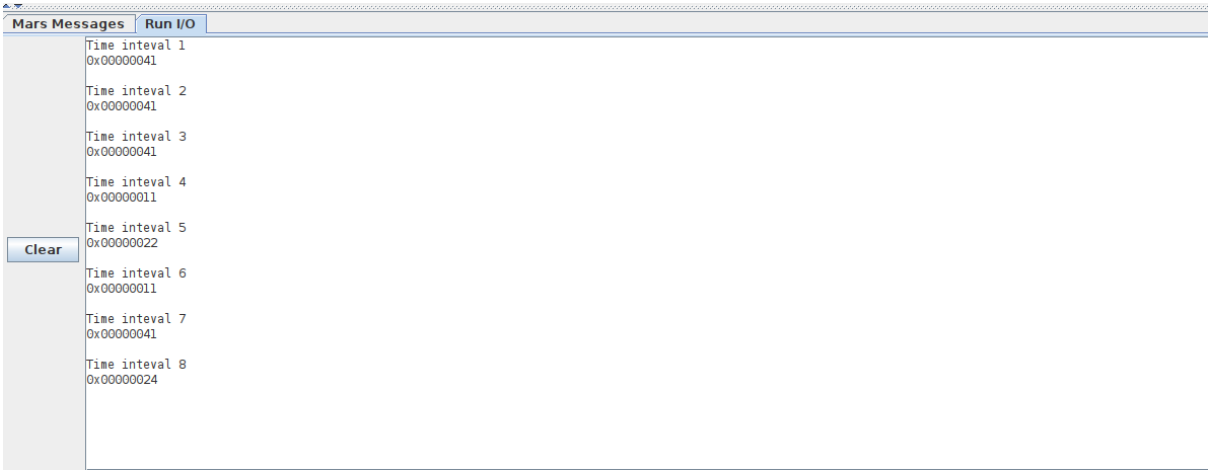
lab11_5.asm | mips11_4.asm | lab11_3.asm | testMars.asm

```
1  .eqv IN_ADRESS_HEXA_KEYBOARD 0xFFFF0012
2  .eqv OUT_ADRESS_HEXA_KEYBOARD 0xFFFF0014
3  .eqv COUNTER 0xFFFF0013 # Time Counter
4  .eqv MASK_CAUSE_COUNTER 0x00000400 # Bit 10: Counter interrupt
5  .eqv MASK_CAUSE_KEYMATRIX 0x00000800 # Bit 11: Key matrix interrupt
6
7  .data
8         msg_keypress: .asciiz "Someone has pressed a key!\n"
9         msg_counter: .asciiz "Time inteval!\n"
10 #~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
11 # MAIN Procedure
12 #~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
13 .text
14 main:
15  #-------------------------------------------------------
16  # Enable interrupts you expect
17  #-------------------------------------------------------
18  # Enable the interrupt of Keyboard matrix 4x4 of Digital Lab Sim
19         li $t1, IN_ADRESS_HEXA_KEYBOARD
20         li $t3, 0x80 # bit 7 = 1 to enable
21         sb $t3, 0($t1)
22 # Enable the interrupt of TimeCounter of Digital Lab Sim
23         li $t1, COUNTER
24         sb $t1, 0($t1)
25
26  #-------------------------------------------------------
27  # Loop an print sequence numbers
28  #-------------------------------------------------------
29 Loop:
30         nop
31         nop
32         nop
33 sleep:
34         addi $v0,$zero,32 # BUG: must sleep to wait for Time Counter
35         li $a0,200 # sleep 300 ms
36         syscall
37         nop # WARNING: nop is mandatory here.
38         b Loop
39 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 $t1, IN_ADRESS_HEXA_KEYBOARD
        li $t2, OUT_ADRESS_HEXA_KEYBOARD

inter_1:
        li $t3, 0x81 # check row 1 with key 0, 1, 2, 4
        sb $t3, 0($t1) # must reassign expected row
        jal inter

inter_2:
        li $t3, 0x82 # check row 2 with key 4, 5, 6, 7
        sb $t3, 0($t1) # must reassign expected row
        jal inter

inter_3:
        li $t3, 0x84 # check row 3 with key 8, 9, A, B
        sb $t3, 0($t1) # must reassign expected row
        jal inter

inter_4:
        li $t3, 0x88 # check row 4 with key C, D, E, F
        sb $t3, 0($t1) # must reassign expected row
        jal inter

after_inter_4:
        beq $a0, 0x0, prn_cod
        j next_pc

inter:
        lb $a0, 0($t2) # read scan code of key button
        bne $a0, 0x0, prn_cod
        jr $ra

prn_cod:
        li $v0,34
        syscall

j end_process
```

```
113 Counter_Intr:
114         li $v0, 4 # Processing Counter Interrupt
115         la $a0, msg_counter
116         syscall
117
118         li $v0,1
119         add $a0, $s7, $zero
120         syscall
121
122         add $s7, $s7, 1
123
124         li $v0, 4 # Processing Counter Interrupt
125         la $a0, msg_counter_endl
126         syscall
127
128         j end_process
129 end_process:
130         mtc0 $zero, $13 # Must clear cause reg
```

```
112         mtc0 $zero, $13 # Must clear cause reg
113 en_int: #-----------------------------------------------------
114  # Re-enable interrupt
115  #-----------------------------------------------------
116         li $t1, COUNTER
117         sb $t1, 0($t1)
118  #-----------------------------------------------------
119  # Evaluate the return address of main routine
120  # epc <= epc + 4
121  #-----------------------------------------------------
122 next_pc:
123         mfc0 $at, $14 # $at <= Coproc0.$14 = Coproc0.epc
124         addi $at, $at, 4 # $at = $at + 4 (next instruction)
125         mtc0 $at, $14 # Coproc0.$14 = Coproc0.epc <= $at
126  return:
127         eret # Return from exception
128
```



Khi đang thực hiện vòng lặp mà có tín hiệu nhấn từ ma trận Lab Sim hoặc Khoảng thời gian lập tới giới hạn thì Chương trình sẽ thực hiện ngắt.

Đầu tiên chương trình thực hiện so sánh và tìm ra nguyên nhân ngắt, nếu là vượt quá thời gian lặp thì sẽ in ra message còn nếu là tín hiệu nhấn từ Lab Sim thì sẽ in ra ký tự được nhấn (hệ cơ số 16)

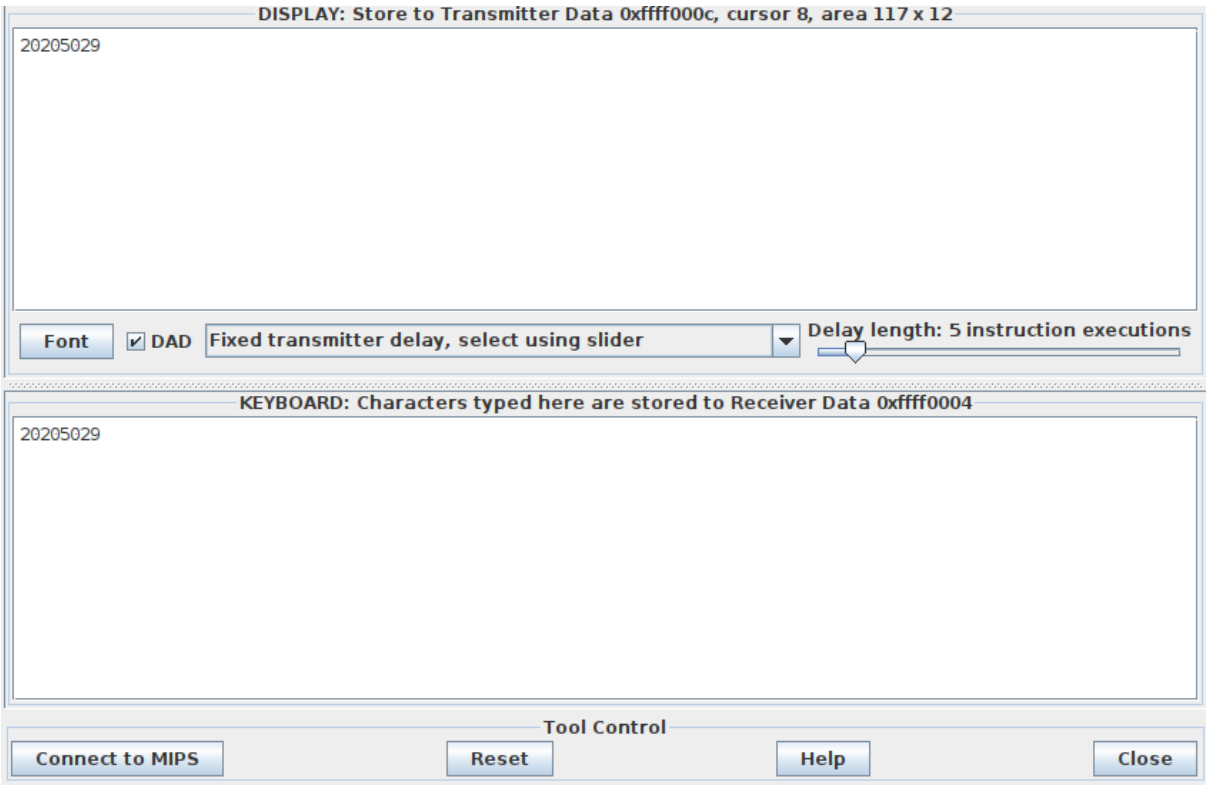
MSSV của em là: 20205029

Trong đó:

- 0x41 là 2
- 0x11 là 0
- 0x22 là 5
- 0x24 là 9

Bài 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


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

```



Ngắt mềm được thực thi khi có một phím bất kỳ được nhấn.

Giải thích:

- Dòng lệnh 17 -> 21 thực hiện điều này
- Khi mà có một phím được nhập vào từ KEYBOARD thì biến t1 sẽ bằng 1 (KEY_REDRY đã sẵn sàng)
- Mà câu lệnh dòng 21 “ teqi $t1, 1 ”: tức là khi biến t1 = 1 thì ngắt mềm sẽ được thực thi