Laboratory Exercise 11

Interrupts & IO programming

Đỗ Hải Dương – 20194528

Assignment 4

Code:

```
.eqv IN_ADRESS_HEXA_KEYBOARD 0xFFFF0012
.eqv OUT_ADRESS_HEXA_KEYBOARD 0xFFFF0014
.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! "
#-----------------------------------------------------------
# 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)
 # Enable the interrupt of TimeCounter of Digital Lab Sim
        addi $s5, $0, 0
        li $t1, COUNTER
        sb $t1, 0($t1)

 #-----------------------------------------------------
 # Loop an print sequence numbers
```

```
 #-----------------------------------------------------
Loop:
        nop
        nop
        nop

sleep:
        addi $v0,$zero,32       # BUG: must sleep to wait for Time Counter
        li $a0,1000             # sleep 300 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
 #-----------------------------------------------------
```

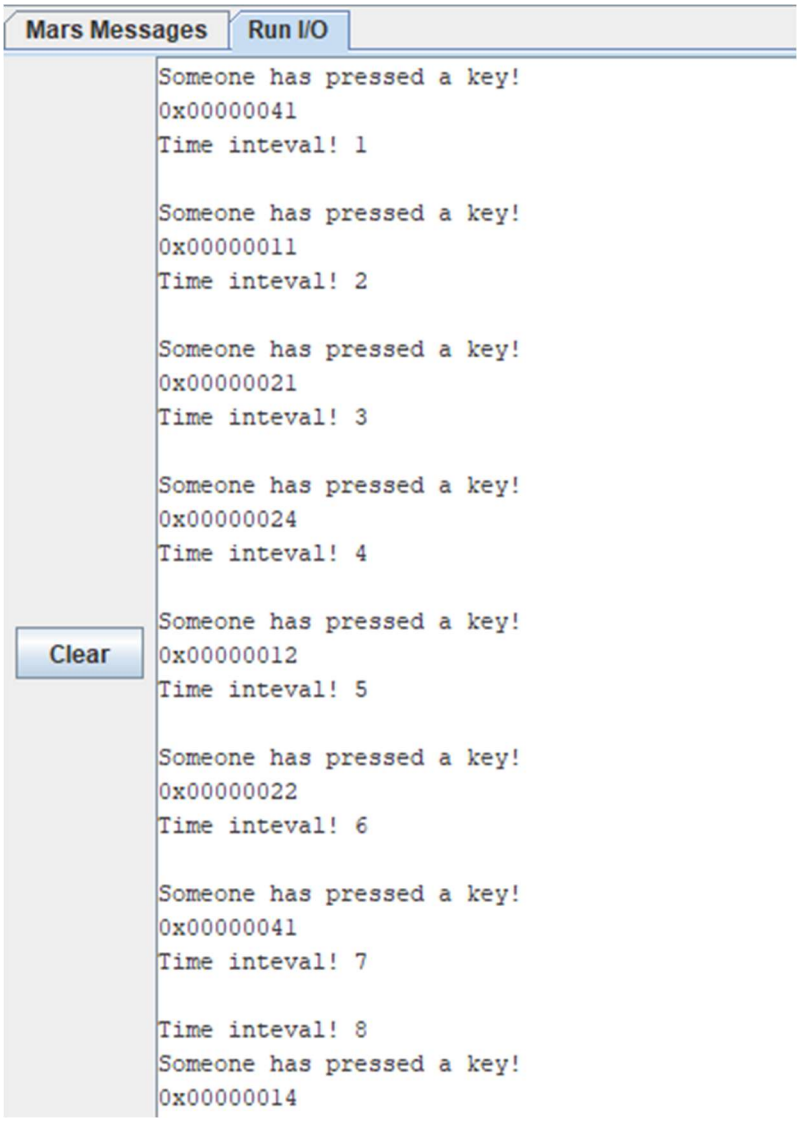
```
 # Processing
 #-----------------------------------------------------
get_caus:
        mfc0 $t1, $13            # $t1 = Coproc0.cause

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

        li $t1, IN_ADRESS_HEXA_KEYBOARD
        li $t2, OUT_ADRESS_HEXA_KEYBOARD

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

```
82          jal check
83
84          li $t3, 0x82      # check row 4 and re-enable bit 7
85          jal check
86
87          li $t3, 0x84      # check row 4 and re-enable bit 7
88          jal check
89
90          li $t3, 0x88      # check row 4 and re-enable bit 7
91
92  check:
93          sb $t3, 0($t1)    # must reassign expected row
94          lb $a0, 0($t2)
95          bne $a0,0x0,prn_cod
96          jr $ra
97
98  prn_cod:
99          li $v0,34
100         syscall
101         li $v0,11
102         li $a0,'¥n'               # print endofline
103         syscall
104         j end_process
105
106 Counter_Intr:
107         li $v0, 4                 # Processing Counter Interrupt
108         la $a0, msg_counter
```

```
109         syscall
110
111         addi $s5, $s5, 1          # count = count + 1
112         addi $v0,$zero,1
113         add $a0,$s5,$zero         # print auto sequence number
114         syscall
115
116         li $v0, 11
117         li $a0,'¥n'               # print endofline
118         syscall
119         j end_process
120
121 end_process:
122         mtc0 $zero, $13           # Must clear cause reg
123
124 en_int:
125  #-----------------------------------------------------
126  # Re-enable interrupt
127  #-----------------------------------------------------
128         li $t1, COUNTER
129         sb $t1, 0($t1)
130  #-----------------------------------------------------
131  # Evaluate the return address of main routine
132  # epc <= epc + 4
133  #-----------------------------------------------------
134 next_pc:
135         mfc0 $at, $14             # $at <= Coproc0.$14 = Coproc0.epc
```

```
136           addi $at, $at, 4          # $at = $at + 4 (next instruction)
137           mtc0 $at, $14             # Coproc0.$14 = Coproc0.epc <= $at
138
139 return:  eret
```

Kết quả : 20194528



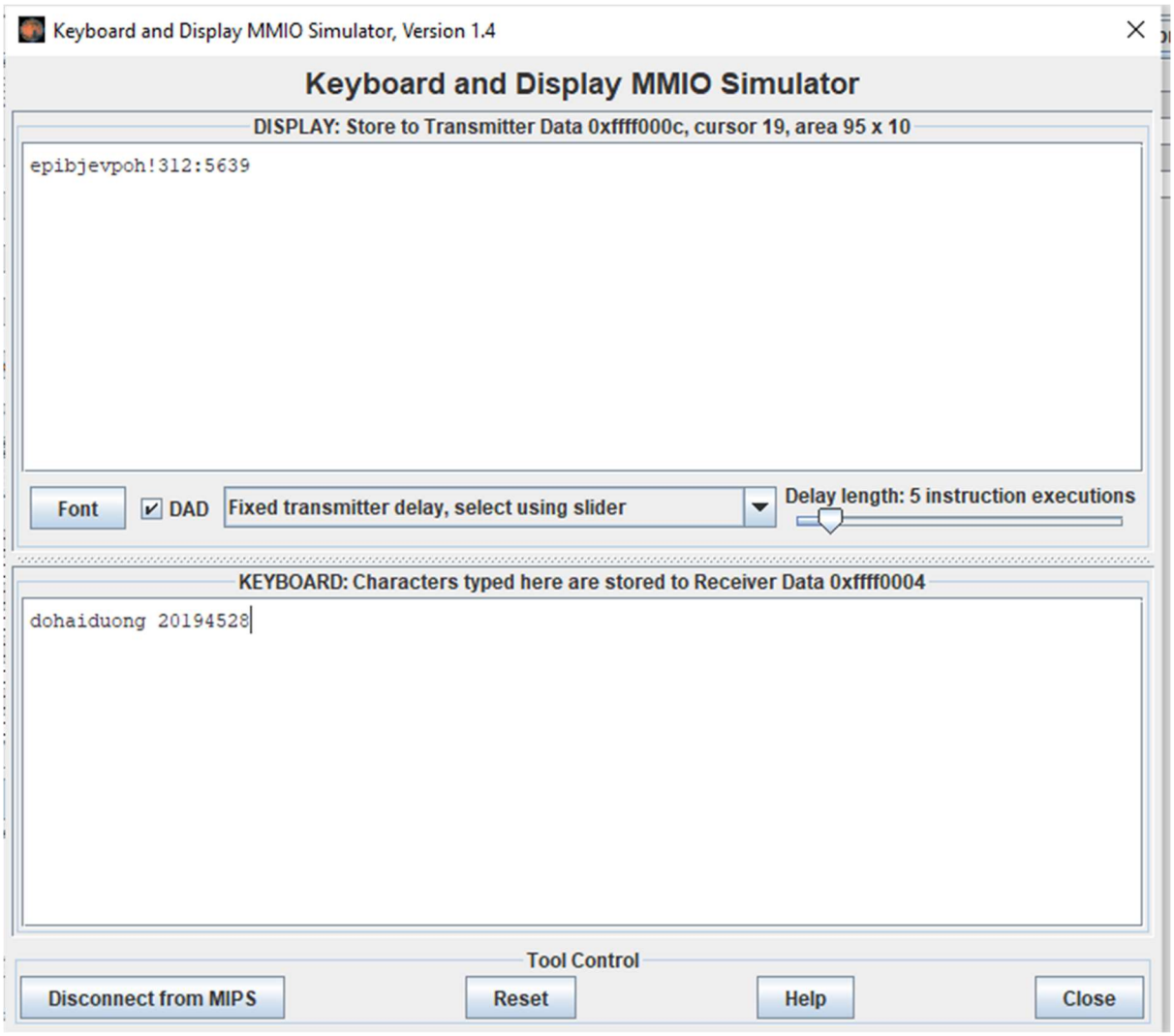
Assignment 5

Code:

```
.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 $t1 = 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
```

Kết quả:


Keyboard and Display MMIO Simulator, Version 1.4
Keyboard and Display MMIO Simulator
DISPLAY: Store to Transmitter Data 0xffff000c, cursor 19, area 95 x 10
epibjevpoh!312:5639
Font
DAD
Fixed transmitter delay, select using slider
Delay length: 5 instruction executions
KEYBOARD: Characters typed here are stored to Receiver Data 0xffff0004
dohaiduong 20194528
Tool Control
Disconnect from MIPS
Reset
Help
Close