Báo cáo Thực hành KTMT buổi 12

Họ và tên: Nguyễn Đức Phú

MSSV: 20215116

Assignment 1:

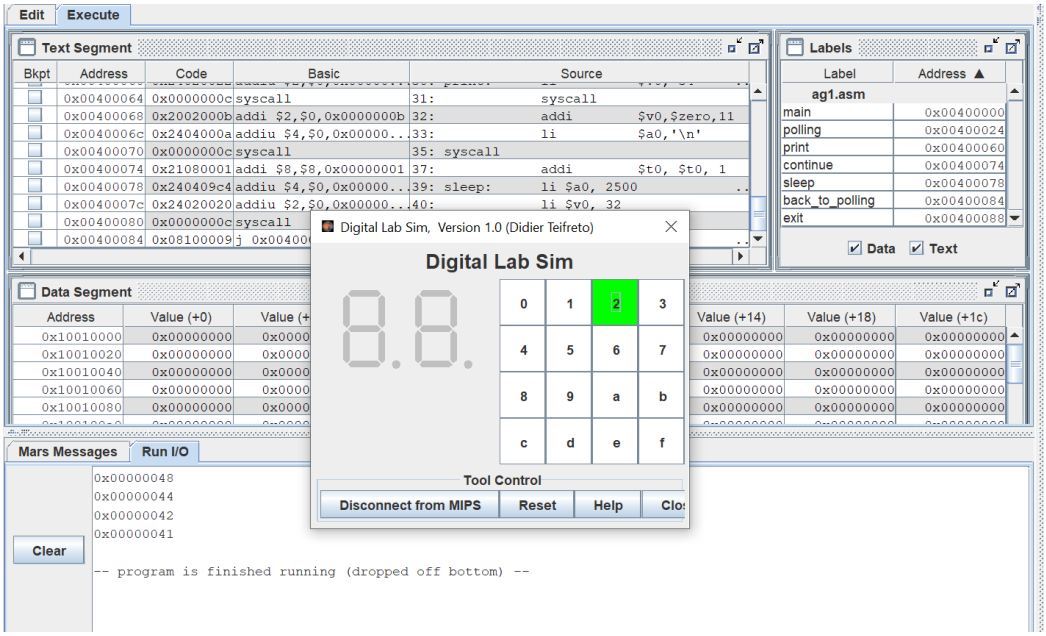
- Code:

```
.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
     li $t3, 0x01 # check row 1 with key C, D,E, F
     li $t4, 0x02 # check row 2 with key C, D,E, F
     li $t5, 0x04 # check row 3 with key C, D,E, F
     li $t6, 0x08 # check row 4 with key C, D,E, F
     li $t0, 0
polling:
     beq  $t0, 4, exit       #exit after print 4 times
     sb   $t3, 0($t1 )
     lb   $a0, 0($t2)
     bne  $a0, $zero, print

     sb   $t4, 0($t1 )
     lb   $a0, 0($t2)
     bne  $a0, $zero, print

     sb   $t5, 0($t1 )
     lb   $a0, 0($t2)
     bne  $a0, $zero, print

     sb   $t6, 0($t1 )
     lb   $a0, 0($t2)
     bne  $a0, $zero, print
     j    continue
print:    li   $v0, 34        # print integer (hexa)
     syscall
     addi      $v0,$zero,11
     li        $a0,'\n'

syscall
```

```
continue:
      addi       $t0, $t0, 1

sleep:
      li         $a0, 2500       # sleep 2500ms
      li         $v0, 32
      syscall
back_to_polling:
      j          polling         # continue polling
exit:
```

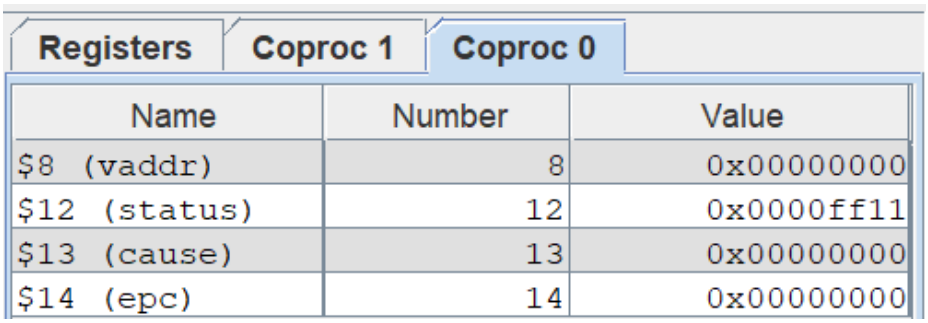
- Kết quả chạy thử:

  Nhấn lần lượt các phím e, a, 6,2



Assignment 2:

- Nhận xét hoạt động:
  - Trước khi xảy ra ngắt:



- Sau khi xảy ra ngắt:

pc		0x80000180

⇨ Thanh ghi PC nhảy tới giá trị của của .ktext

Registers | Coproc 1 | Coproc 0

Name	Number	Value
$8 (vaddr)	8	0x00000000
$12 (status)	12	0x0000ff13
$13 (cause)	13	0x00000800
$14 (epc)	14	0x00400010

⇨ Giá trị của thanh ghi 12 thay đổi báo hiệu xảy ra ngắt

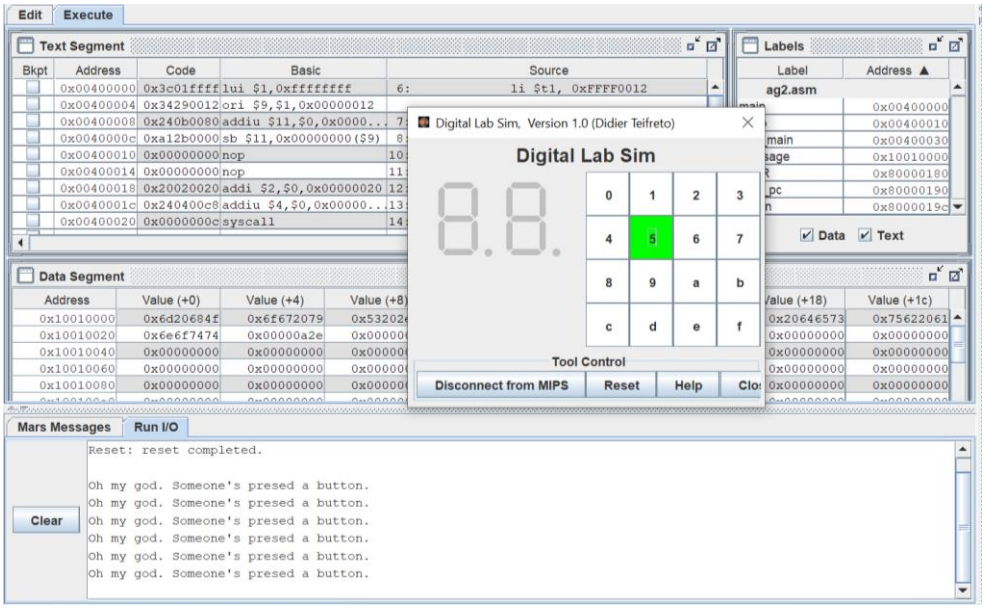
⇨ Thanh ghi 13 set giá trị 0x00000800 cho biết nguyên nhân là KeyMatrix

⇨ Thanh ghi 14 chứa địa chỉ của lệnh vừa thực hiện xong
Giá trị này được tăng lên 4 bằng các câu lệnh trong chương trình con để sau đó trả về cho thanh ghi PC địa chỉ lệnh kế tiếp

Registers | Coproc 1 | Coproc 0

Name	Number	Value
$8 (vaddr)	8	0x00000000
$12 (status)	12	0x0000ff13
$13 (cause)	13	0x00000800
$14 (epc)	14	0x00400014

- Câu lệnh in ra `Message` được thực hiện tại chương trình con phục vụ ngắt



Assignment 3:

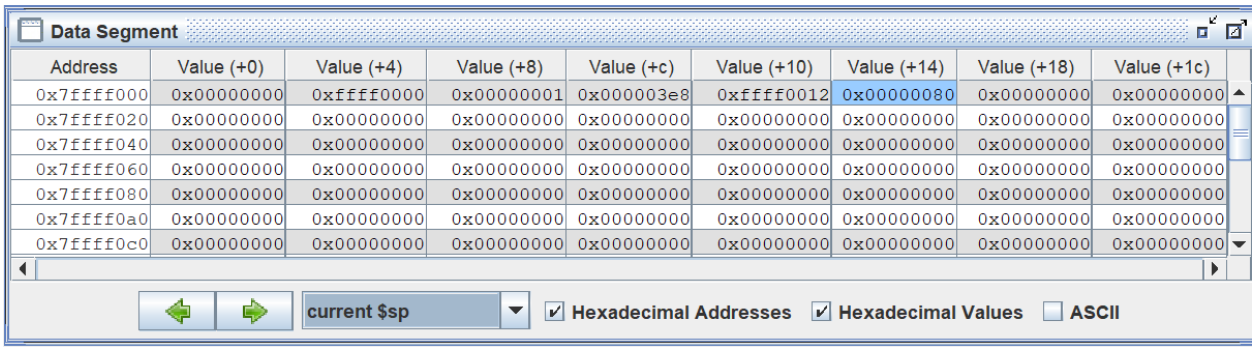
- Code:

```
.eqv IN_ADRESS_HEXA_KEYBOARD 0xFFFF0012
.eqv OUT_ADRESS_HEXA_KEYBOARD 0xFFFF0014
.data
Message: .asciiz "Key scan code "
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# MAIN Procedure
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
.text
main:
     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   $a0,1000            # sleep 1000 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
     sw        $ra,0($sp)
     addi      $sp,$sp,4      # Save $at
     sw        $at,0($sp)
     addi      $sp,$sp,4      # Save $sp
     sw        $v0,0($sp)
     addi      $sp,$sp,4      # Save $a0
     sw        $a0,0($sp)
     addi      $sp,$sp,4      # Save $t1
     sw        $t1,0($sp)
     addi      $sp,$sp,4      # Save $t3
     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 1 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 2 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 3 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
```

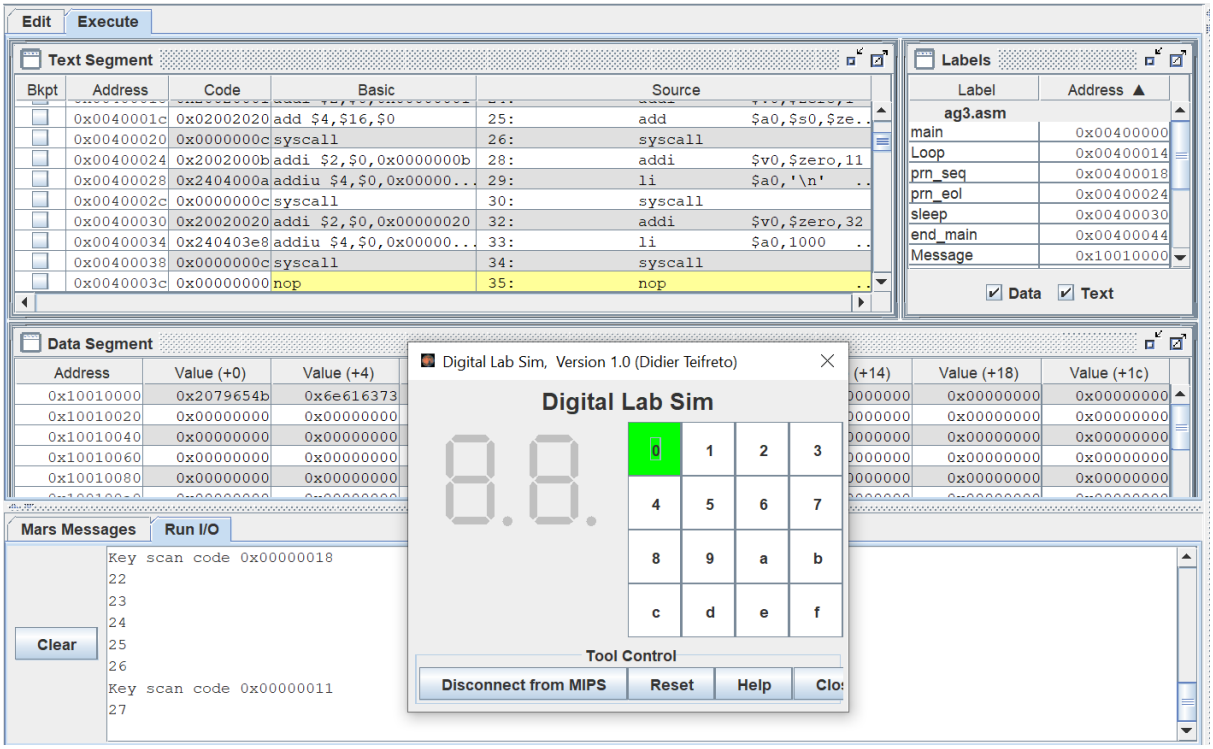
```
     li   $t1, IN_ADRESS_HEXA_KEYBOARD
     li   $t3, 0x88      # check row 4 and re-enable bit 7
     sb   $t3, 0($t1)    # must reassign expected row
     li   $t1, OUT_ADRESS_HEXA_KEYBOARD
     lb   $a0, 0($t1)
     bne  $a0, $zero, prn_cod

prn_cod:
     li   $v0,34
     syscall
     li   $v0,11
     li   $a0,'\n'  # 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 from stack
     addi      $sp,$sp,-4
     lw        $t1, 0($sp)    # Restore from stack
     addi      $sp,$sp,-4
     lw        $a0, 0($sp)    # Restore from stack
     addi      $sp,$sp,-4
     lw        $v0, 0($sp)    # Restore from stack
     addi      $sp,$sp,-4
     lw        $ra, 0($sp)    # Restore from stack
     addi      $sp,$sp,-4
     lw        $ra, 0($sp)    # Restore from stack
     addi      $sp,$sp,-4
return:
     eret                     # Return from exception
```

- Kết quả:
  - Khi nhấn nút tại Digital Lab Sim: ngắt xảy ra và có các cơ chế tương tự như đã trình bày ở Assignment 2
  - Khi ngắt xảy ra giá trị các thanh ghi `$ra $at $sp $a0 $t1 $t3` được lưu vào stack:



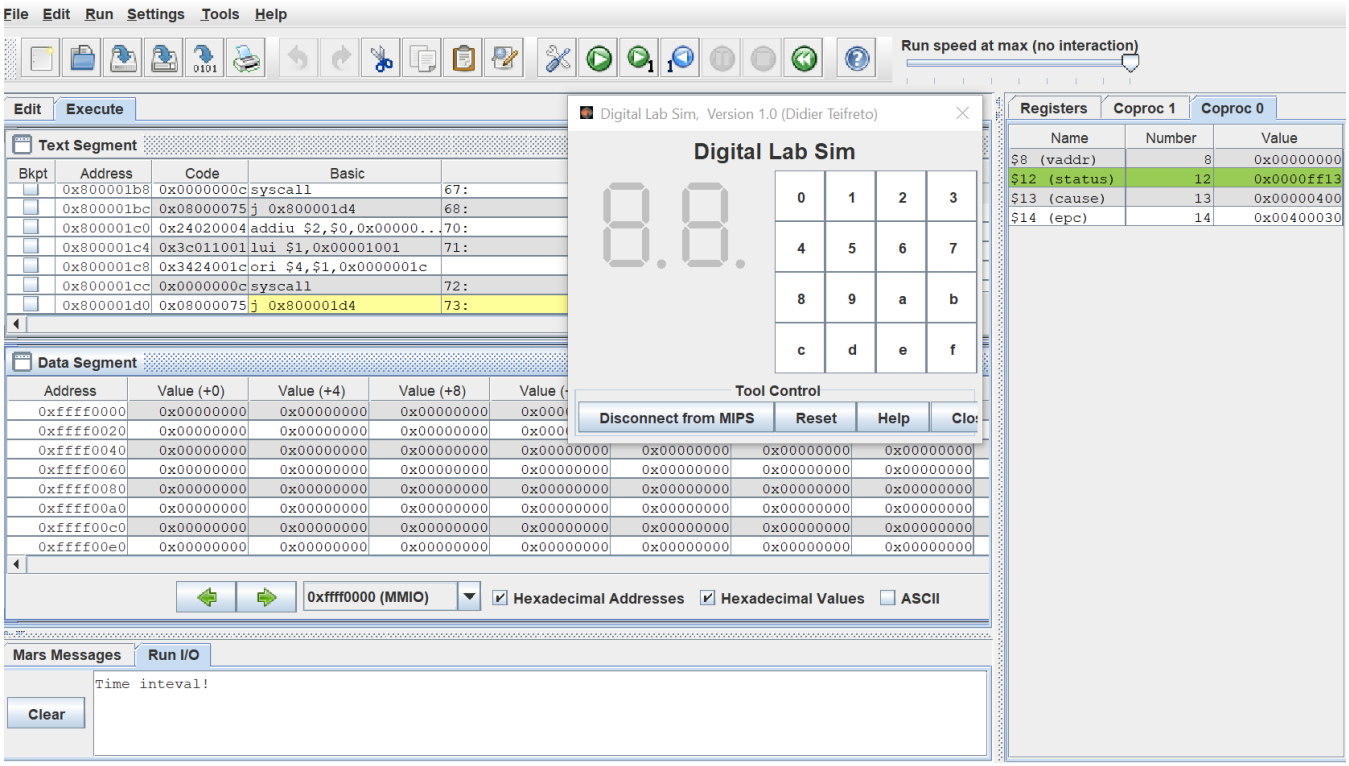
⇨ Sau đó được restore lại khi kết thúc chương trình con ngắt

  - Kết quả chạy:



Assignment 4:

- Với trường hợp ngắt do Time counter:



⇨ Giá trị tại thanh ghi 13 là 0x00000400 cho biết nguyên nhân ngắt do time counter

⇨ Giá trị này được and với *MASK_CAUSE_COUNTER* và trả về chính nó
Còn khi and với giá trị *MASK_CAUSE_KEYMATRIX* thì trả về 0x00000000

⇨ Chương trình nhảy tới nhãn Counter_Intr và in ra "Time interval!" tại cửa sổ Run I/O

⇨ Sau đó giá trị tại $14 được tăng lên 4 qua các câu lệnh tại cuối chương trình con. Giá trị này được trả về cho thanh ghi PC và tiếp tục chương trình chính

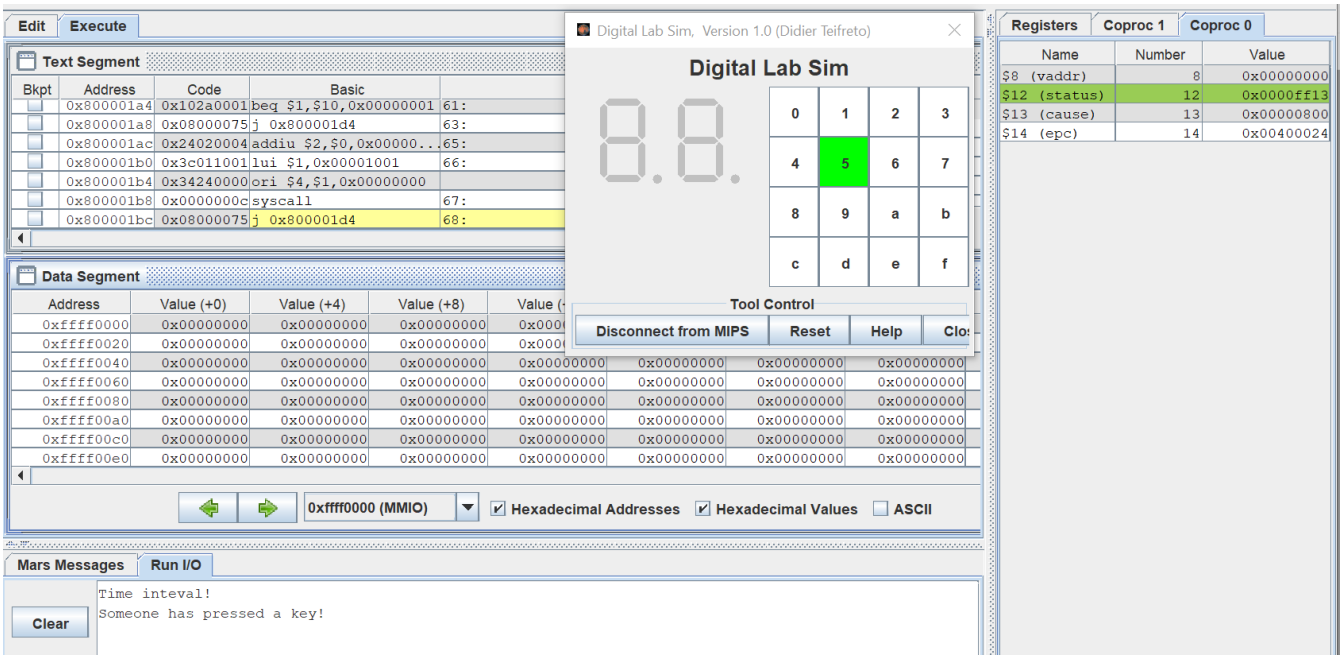
- Với trường hợp ngắt do KeyMatrix:

⇨ Các cơ chế ngắt xảy ra tương tự

⇨ Khác so với trường hợp trên ở giá trị thanh ghi 13
$13 = 0x00000800 cho biết nguyên nhân ngắt do KeyMatrix

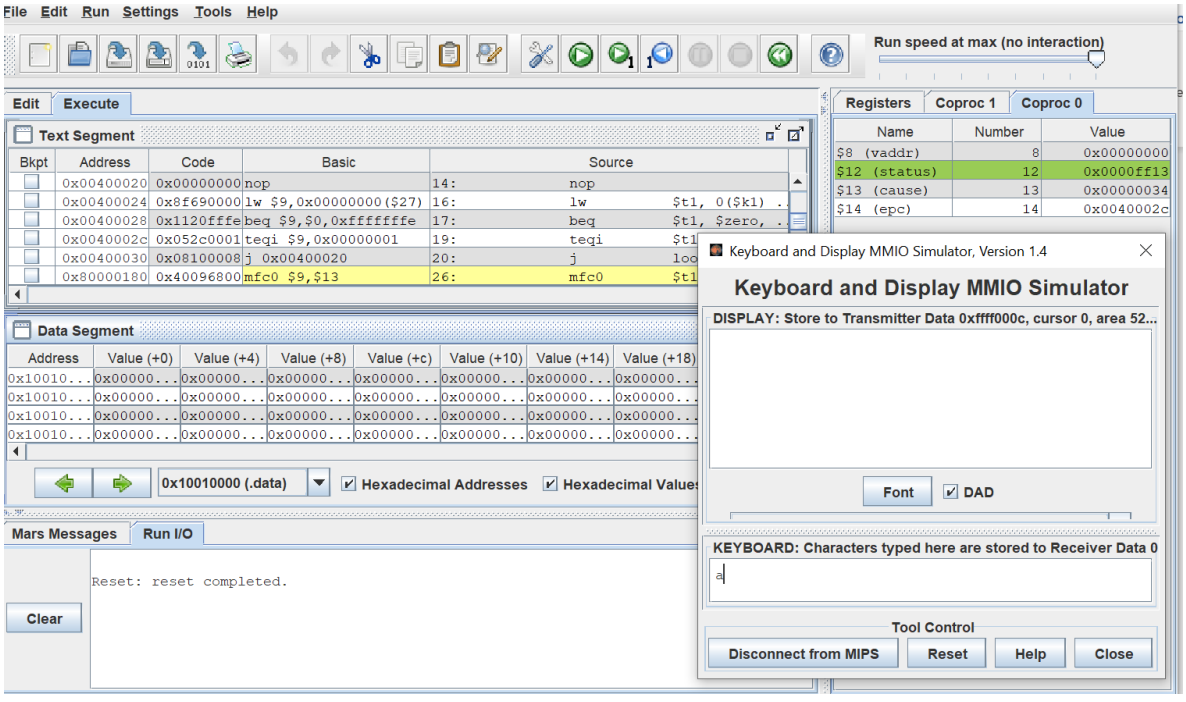
⇨ Giá trị này được and với *MASK_CAUSE_KEYMATRIX* và trả về chính nó
Còn khi and với giá trị *MASK_CAUSE_COUNTER* thì trả về 0x00000000

⇨ Chương trình nhảy tới nhãn Keymatrix_Intr và in ra “Someone has pressed a key” tại cửa sổ Run I/O

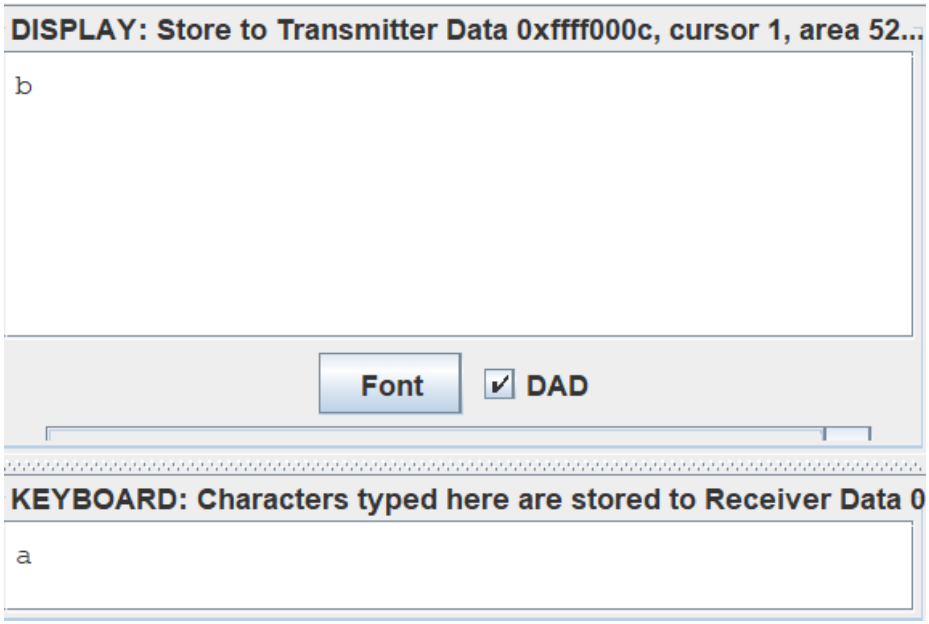


⇨ Sau đó giá trị tại $14 được tăng lên 4 qua các câu lệnh tại cuối chương trình con. Giá trị này được trả về cho thanh ghi PC và tiếp tục chương trình chính

Assignment 5:



- ⇨ Khi nhập ký tự bất kỳ tại Keyboard chương trình thoát khỏi vòng lặp WairForKey và sử dụng lệnh `teqi` tạo ra ngắt mềm
- ⇨ Thanh ghi 12 thay đổi trạng thái báo hiệu ngắt
- ⇨ Thanh ghi 13 thay đổi thành 0x00000034 cho biết nguyên nhân ngắt
- ⇨ Tại chương trình con phục vụ ngắt: key được đọc vào và tăng giá trị mã ASCII thêm 1 rồi hiển thị ra Display



- ⇨ Thanh ghi 14 được tăng lên 4 bằng các câu lệnh ở cuối chương trình con

Name	Number	Value
$8 (vaddr)	8	0x00000000
$12 (status)	12	0x0000ff13
$13 (cause)	13	0x00000034
$14 (epc)	14	0x00400030

- ⇨ Trả về cho thanh ghi PC và tiếp tục chương trình chính

pc		0x00400030