

# Laboratory Exercise 11 – Report:

Nguyễn Hải Dương- 20194530

## 1. Assignment 1

- Mã nguồn:

```
1  .eqv IN_ADRESS_HEXА_KEYBOARD 0xFFFF0012
2  .eqv OUT_ADRESS_HEXА_KEYBOARD 0xFFFF0014
3
4
5  .text
6  main:
7      li $t1, IN_ADRESS_HEXА_KEYBOARD
8      li $t2, OUT_ADRESS_HEXА_KEYBOARD
9
10
11  start_polling_1:
12      li $t3, 0x01 # check row 1 with key 0, 1, 2, 4
13      sb $t3, 0($t1) # must reassign expected row
14      jal polling
15
16  start_polling_2:
17      li $t3, 0x02 # check row 2 with key 4, 5, 6, 7
18      sb $t3, 0($t1) # must reassign expected row
19      jal polling
20
21  start_polling_3:
22      li $t3, 0x04 # check row 3 with key 8, 9, A, B
23      sb $t3, 0($t1) # must reassign expected row
24      jal polling
25
26  start_polling_4:
27      li $t3, 0x08 # check row 4 with key C, D, E, F
28      sb $t3, 0($t1) # must reassign expected row
29      jal polling
30
31  check_after_polling_4:
32      beq $a0, 0x0, print
33      j start_polling_1
34
```

```

35 polling:
36     lb $a0, 0($t2) # read scan code of key button
37     bne $a0, 0x0, print
38     jr $ra
39
40 print:
41     li $v0, 34 # print integer (hexa)
42     syscall
43
44 sleep:
45     li $a0, 3000 # sleep 100ms
46     li $v0, 32
47     syscall
48
49 back_to_start_polling:
50     j start_polling_1 # back to check row 1
51

```

## - Kết quả chạy:

The screenshot displays a debugger interface with three main panels:

- Text Segment:** Shows assembly instructions with their addresses, codes, and comments. For example, at address 0x00400000, there is a `lui $t1, 0xffff0012` instruction.
- Labels:** Lists program labels and their corresponding addresses. Labels include `bail1.txt`, `main`, `start_polling_1`, `start_polling_2`, `start_polling_3`, `start_polling_4`, `check_after_polling`, `polling`, `print`, `sleep`, and `back_to_start_polling`.
- Data Segment:** Displays memory values at various offsets from address 0x10010000. The values are mostly zeros, indicating uninitialized memory.

The Mars Messages window shows the following output:

```

0x00000041
Reset: reset completed.

0x000000410x000000110x000000210x000000240x000000120x000000220xfffff810x00000011

```

A "Clear" button is visible at the bottom left of the message area.

MSSV: 20194530

## - Giải thích:

**Yêu cầu: Check toàn bộ các ký tự từ 0 -> F**

**In ra kết quả khi nhập mã số sinh viên từ bàn phím**

0x41 là số 2;

0x11 là số 0;

0x21 là số 1;

0x24 là số 9;

0x12 là số 4;

0x22 là số 5;

0x41 là số 3;

0x11 là số 0;

MSSV: 20194530 → Kết quả như ảnh trên

## **2. Assignment 2**

**- Mã nguồn:**

```

1  #home assignment 2
2
3  .eqv IN_ADRESS_HEXА_KEYBOARD 0xFFFF0012
4  .data
5  Message: .asciiz "Nguyen Hai Duong\n"
6  #~~~~~
7  # MAIN Procedure
8  #~~~~~
9  .text
10 main:
11  #-----
12  # Enable interrupts you expect
13  #-----
14  # Enable the interrupt of Keyboard matrix 4x4 of Digital Lab Sim
15  li $t1, IN_ADRESS_HEXА_KEYBOARD
16  li $t3, 0x80 # bit 7 of = 1 to enable interrupt
17  sb $t3, 0($t1)
18  #-----
19  # No-end loop, main program, to demo the effective of interrupt
20  #-----
21  Loop: nop
22  nop
23  nop
24  nop
25  b Loop # Wait for interrupt
26  end_main:
27  #~~~~~
28  # GENERAL INTERRUPT SERVED ROUTINE for all interrupts
29  #~~~~~
30  .ktext 0x80000180
31  #-----
32  # Processing
33  #-----
34  IntSR: addi $v0, $zero, 4 # show message

```

```

34 IntSR: addi $v0, $zero, 4 # show message
35 la $a0, Message
36 syscall
37 #-----
38 # Evaluate the return address of main routine
39 # epc <= epc + 4
40 #-----
41 next_pc: mfc0 $at, $14 # $at <= Coproc0.$14 = Coproc0.epc
42 addi $at, $at, 4 # $at = $at + 4 (next instruction)
43 mtc0 $at, $14 # Coproc0.$14 = Coproc0.epc <= $at
44 return: eret # Return from exception
45

```

## - Kết quả chạy mô phỏng:

Reset: reset completed.

Clear

Reset: reset completed.

Nguyen Hai Duong

The screenshot shows a MIPS simulator interface with four main panels:

- Text Segment:** Displays assembly code with addresses, codes, and comments. Key instructions include `addi $v0, $zero, 4`, `la $a0, Message`, `syscall`, and `eret`.
- Labels:** Lists labels and their corresponding addresses, such as `main` at `0x00400000` and `return` at `0x0000019c`.
- Data Segment:** Shows memory addresses and their values in hexadecimal. The address `0x10010000` is highlighted.
- Registers:** A table on the right lists registers (e.g., `$zero`, `$at`, `$v0`) and their current values.

## - Giải thích:

Khi nhấn phím bất kì từ 0 -> F thì sẽ hiện ra tên và mssv

### 3. Assignment 3

- Mã nguồn:

---

```
1  # home assignment 3
2
3  .eqv IN_ADDRESS_HEX_KEYBOARD 0xFFFF0012
4  .eqv OUT_ADDRESS_HEX_KEYBOARD 0xFFFF0014
5  .data
6  Message: .asciiz "Key scan code "
7  #~~~~~
8  # MAIN Procedure
9  #~~~~~
10 .text
11 main:
12  #-----
13  # Enable interrupts you expect
14  #-----
15  # Enable the interrupt of Keyboard matrix 4x4 of Digital Lab Sim
16  li $t1, IN_ADDRESS_HEX_KEYBOARD
17  li $t3, 0x80 # bit 7 = 1 to enable
18  sb $t3, 0($t1)
19  #-----
20  # Loop an print sequence numbers
21  #-----
22  xor $s0, $s0, $s0 # count = $s0 = 0
23  Loop: addi $s0, $s0, 1 # count = count + 1
24  prn_seq: addi $v0, $zero, 1
25  add $a0, $s0, $zero # print auto sequence number
26  syscall
27  prn_eol: addi $v0, $zero, 11
28  li $a0, '\n' # print endofline
29  syscall
30  sleep: addi $v0, $zero, 32
31  li $a0, 300 # sleep 300 ms
32  syscall
33  nop # WARNING: nop is mandatory here.
34  b Loop # Loop
```

---

```

34  b Loop # Loop
35  end_main:
36  ~~~~~
37  # GENERAL INTERRUPT SERVED ROUTINE for all interrupts
38  ~~~~~
39  .ktext 0x80000180
40  #-----
41  # SAVE the current REG FILE to stack
42  #-----
43  IntSR: addi $sp,$sp,4 # Save $ra because we may change it later
44  sw $ra,0($sp)
45  addi $sp,$sp,4 # Save $ra because we may change it later
46  sw $at,0($sp)
47  addi $sp,$sp,4 # Save $ra because we may change it later
48  sw $v0,0($sp)
49  addi $sp,$sp,4 # Save $a0, because we may change it later
50  sw $a0,0($sp)
51  addi $sp,$sp,4 # Save $t1, because we may change it later
52  sw $t1,0($sp)
53  addi $sp,$sp,4 # Save $t3, because we may change it later
54  sw $t3,0($sp)
55  #-----
56  # Processing
57  #-----
58  prn_msg:addi $v0, $zero, 4
59  la $a0, Message
60  syscall
61  get_cod:

```

---

```

61 get_cod:
62     li $t1, IN_ADRESS_HEX_A_KEYBOARD
63     li $t2, OUT_ADRESS_HEX_A_KEYBOARD
64 start_interrupt_1:
65     li $t3, 0x81 # check row 1 with key 0, 1, 2, 3
66     sb $t3, 0($t1) # must reassign expected row
67     jal interrupt
68
69 start_interrupt_2:
70     li $t3, 0x82 # check row 2 with key 4, 5, 6, 7
71     sb $t3, 0($t1) # must reassign expected row
72     jal interrupt
73
74 start_interrupt_3:
75     li $t3, 0x84 # check row 3 with key 8, 9, A, B
76     sb $t3, 0($t1) # must reassign expected row
77     jal interrupt
78
79 start_interrupt_4:
80     li $t3, 0x88 # check row 4 with key C, D, E, F
81     sb $t3, 0($t1) # must reassign expected row
82     jal interrupt
83
84 check_after_interrupt_4:
85     beq $a0, 0x0, prn_cod
86     j next_pd
87
88 interrupt:
89     lb $a0, 0($t2) # read scan code of key button
90     bne $a0, 0x0, prn_cod
91     jr $ra
92 prn_cod: li $v0, 34

```



```

92  prn_cod:li $v0,34
93  syscall
94  li $v0,11
95  li $a0,'\n' # print endofline
96  syscall
97  #-----
98  # Evaluate the return address of main routine
99  # epc <= epc + 4
100 #-----
101 next_pc:mfc0 $at, $14 # $at <= Coproc0.$14 = Coproc0.epc
102 addi $at, $at, 4 # $at = $at + 4 (next instruction)
103 mtc0 $at, $14 # Coproc0.$14 = Coproc0.epc <= $at
104 #-----
105 # RESTORE the REG FILE from STACK
106 #-----
107 restore:lw $t3, 0($sp) # Restore the registers from stack
108 addi $sp,$sp,-4
109 lw $t1, 0($sp) # Restore the registers from stack
110 addi $sp,$sp,-4
111 lw $a0, 0($sp) # Restore the registers from stack
112 addi $sp,$sp,-4
113 lw $v0, 0($sp) # Restore the registers from stack
114 addi $sp,$sp,-4
115 lw $ra, 0($sp) # Restore the registers from stack
116 addi $sp,$sp,-4
117 return:eret # Return from exception
118

```

- Kết quả chạy:



0x12 là số 4;

0x22 là số 5;

0x41 là số 3;

0x11 là số 0;

MSSV: 20194530 → Kết quả như ảnh trên