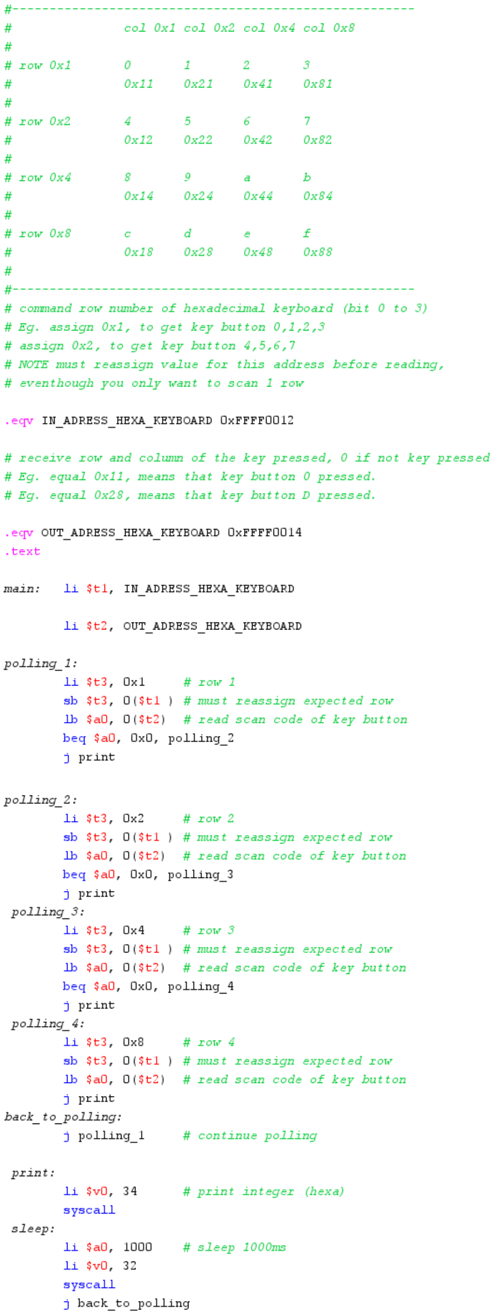
BÀI TẬP TUẦN 11

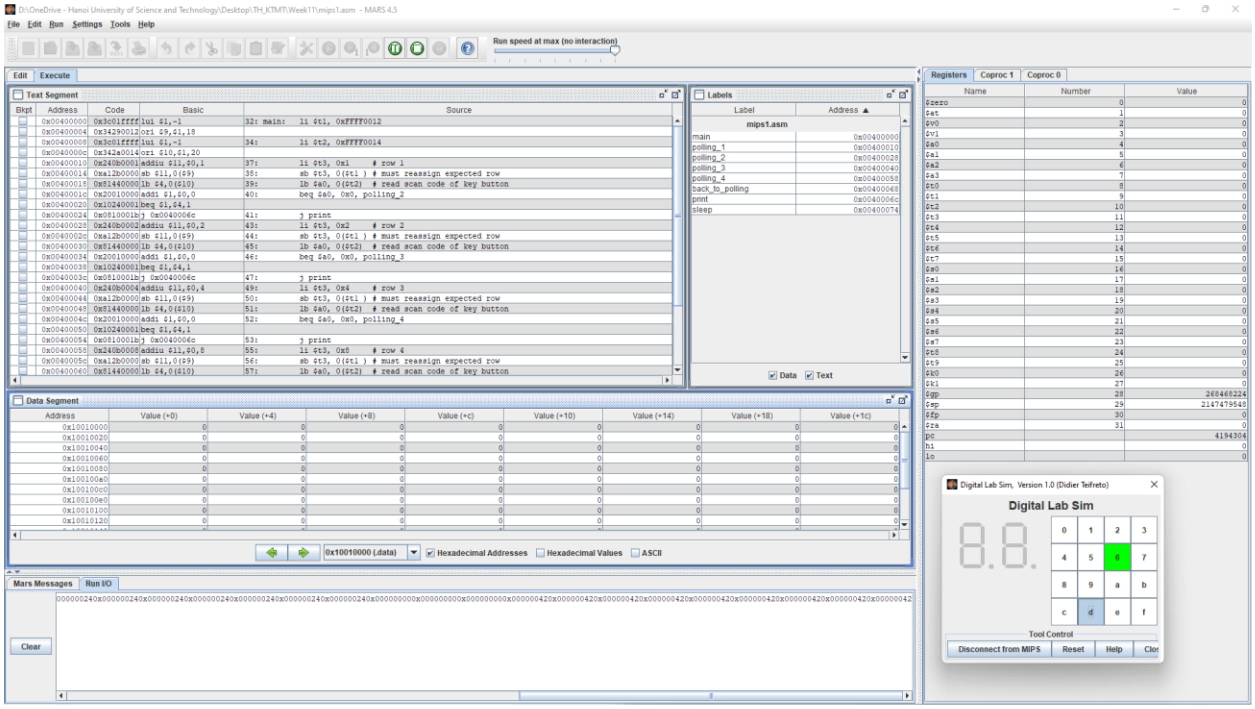
Assignment 1:

- *Chương trình:*



```
#-----------------------------------------------------
#               col 0x1 col 0x2 col 0x4 col 0x8
#
# row 0x1       0       1       2       3
#               0x11    0x21    0x41    0x81
#
# row 0x2       4       5       6       7
#               0x12    0x22    0x42    0x82
#
# row 0x4       8       9       a       b
#               0x14    0x24    0x44    0x84
#
# row 0x8       c       d       e       f
#               0x18    0x28    0x48    0x88
#
#-----------------------------------------------------
# command row number of hexadecimal keyboard (bit 0 to 3)
# Eg. assign 0x1, to get key button 0,1,2,3
# assign 0x2, to get key button 4,5,6,7
# NOTE must reassign value for this address before reading,
# eventhough you only want to scan 1 row

.eqv IN_ADRESS_HEXA_KEYBOARD 0xFFFF0012

# receive row and column of the key pressed, 0 if not key pressed
# Eg. equal 0x11, means that key button 0 pressed.
# Eg. equal 0x28, means that key button D pressed.

.eqv OUT_ADRESS_HEXA_KEYBOARD 0xFFFF0014
.text

main:   li $t1, IN_ADRESS_HEXA_KEYBOARD

        li $t2, OUT_ADRESS_HEXA_KEYBOARD

polling_1:
        li $t3, 0x1     # row 1
        sb $t3, 0($t1 ) # must reassign expected row
        lb $a0, 0($t2)  # read scan code of key button
        beq $a0, 0x0, polling_2
        j print


polling_2:
        li $t3, 0x2     # row 2
        sb $t3, 0($t1 ) # must reassign expected row
        lb $a0, 0($t2)  # read scan code of key button
        beq $a0, 0x0, polling_3
        j print
 polling_3:
        li $t3, 0x4     # row 3
        sb $t3, 0($t1 ) # must reassign expected row
        lb $a0, 0($t2)  # read scan code of key button
        beq $a0, 0x0, polling_4
        j print
 polling_4:
        li $t3, 0x8     # row 4
        sb $t3, 0($t1 ) # must reassign expected row
        lb $a0, 0($t2)  # read scan code of key button
        j print
back_to_polling:
        j polling_1     # continue polling

 print:
        li $v0, 34      # print integer (hexa)
        syscall
 sleep:
        li $a0, 1000    # sleep 1000ms
        li $v0, 32
        syscall
        j back_to_polling
```

- *Kết quả:*



D:\OneDrive - Hanoi University of Science and Technology\Desktop\TH_KTMT\Week11\mips1.asm - MARS 4.5
File Edit Run Settings Tools Help
Run speed at max (no interaction)
Edit Execute
Text Segment
Labels
Registers Coproc 1 Coproc 0
Data Segment
Mars Messages Run I/O
Clear
Digital Lab Sim, Version 1.0 (Didier Teifreto)
Digital Lab Sim
Tool Control
Disconnect from MIPS Reset Help Clo

```
Bkpt Address     Code       Basic               Source
     0x00400000 0x3c01ffff lui $1,-1           main:   li $t1, 0xFFFF0012
     0x00400004 0x34290012 ori $9,$1,18
     0x00400008 0x3c01ffff lui $1,-1                   li $t2, 0xFFFF0014
     0x0040000c 0x342a0014 ori $10,$1,20
     0x00400010 0x240b0001 addiu $11,$0,1              li $t3, 0x1      # row 1
     0x00400014 0xa12b0000 sb $11,0($9)                sb $t3, 0($t1 ) # must reassign expected row
     0x00400018 0x81440000 lb $4,0($10)                lb $a0, 0($t2)  # read scan code of key button
     0x0040001c 0x20010000 addi $1,$0,0                beq $a0, 0x0, polling_2
     0x00400020 0x10240001 beq $1,$4,1
     0x00400024 0x0810001b j 0x0040006c                j print
     0x00400028 0x240b0002 addiu $11,$0,2              li $t3, 0x2      # row 2
     0x0040002c 0xa12b0000 sb $11,0($9)                sb $t3, 0($t1 ) # must reassign expected row
     0x00400030 0x81440000 lb $4,0($10)                lb $a0, 0($t2)  # read scan code of key button
     0x00400034 0x20010000 addi $1,$0,0                beq $a0, 0x0, polling_3
     0x00400038 0x10240001 beq $1,$4,1
     0x0040003c 0x0810001b j 0x0040006c                j print
     0x00400040 0x240b0004 addiu $11,$0,4              li $t3, 0x4      # row 3
     0x00400044 0xa12b0000 sb $11,0($9)                sb $t3, 0($t1 ) # must reassign expected row
     0x00400048 0x81440000 lb $4,0($10)                lb $a0, 0($t2)  # read scan code of key button
     0x0040004c 0x20010000 addi $1,$0,0                beq $a0, 0x0, polling_4
     0x00400050 0x10240001 beq $1,$4,1
     0x00400054 0x0810001b j 0x0040006c                j print
     0x00400058 0x240b0008 addiu $11,$0,8              li $t3, 0x8      # row 4
     0x0040005c 0xa12b0000 sb $11,0($9)                sb $t3, 0($t1 ) # must reassign expected row
     0x00400060 0x81440000 lb $4,0($10)                lb $a0, 0($t2)  # read scan code of key button
```

Label	Address ▲
mips1.asm	
main	0x00400000
polling_1	0x00400010
polling_2	0x00400028
polling_3	0x00400040
polling_4	0x00400058
back_to_polling	0x00400068
print	0x0040006c
sleep	0x00400074

☑ Data ☑ Text

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0	0	0	0	0	0	0	0
0x10010020	0	0	0	0	0	0	0	0
0x10010040	0	0	0	0	0	0	0	0
0x10010060	0	0	0	0	0	0	0	0
0x10010080	0	0	0	0	0	0	0	0
0x100100a0	0	0	0	0	0	0	0	0
0x100100c0	0	0	0	0	0	0	0	0
0x100100e0	0	0	0	0	0	0	0	0
0x10010100	0	0	0	0	0	0	0	0
0x10010120	0	0	0	0	0	0	0	0

0x10010000 (.data) ☑ Hexadecimal Addresses ☐ Hexadecimal Values ☐ ASCII

```
000000240x000000240x000000240x000000240x000000240x000000240x000000240x000000000x000000000x000000000x000000420x000000420x000000420x000000420x000000420x000000420x000000420x000000420x00000042
```

Name	Number	Value
$zero	0	0
$at	1	0
$v0	2	0
$v1	3	0
$a0	4	0
$a1	5	0
$a2	6	0
$a3	7	0
$t0	8	0
$t1	9	0
$t2	10	0
$t3	11	0
$t4	12	0
$t5	13	0
$t6	14	0
$t7	15	0
$s0	16	0
$s1	17	0
$s2	18	0
$s3	19	0
$s4	20	0
$s5	21	0
$s6	22	0
$s7	23	0
$t8	24	0
$t9	25	0
$k0	26	0
$k1	27	0
$gp	28	268468224
$sp	29	2147479548
$fp	30	0
$ra	31	0
pc		4194304
hi		0
lo		0

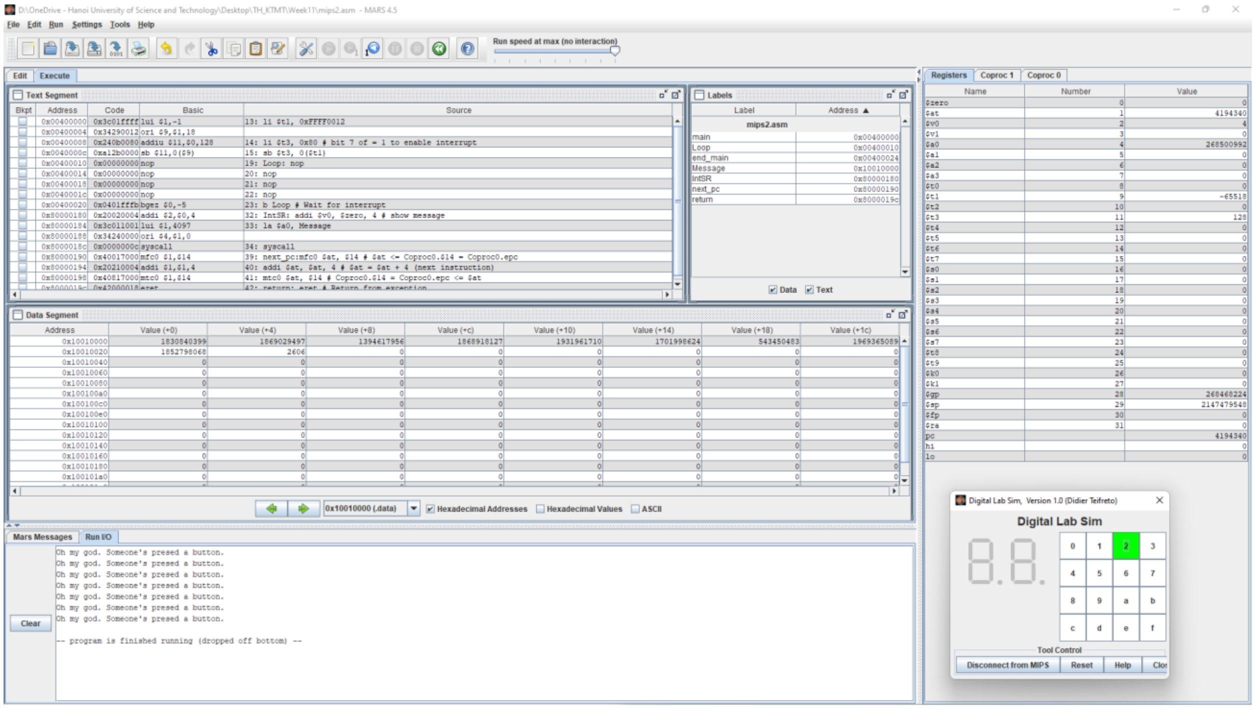
0	1	2	3
4	5	🟩 6	7
8	9	a	b
c	d	e	f

Assignment 2:

- *Chương trình:*

```
.eqv IN_ADDRESS_HEXA_KEYBOARD 0xFFFF0012
.data
Message: .asciiz "Oh my god. Someone's presed a button.\n"
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# MAIN Procedure
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
.text
main:
#---------------------------------------------------------
# Enable interrupts you expect
#---------------------------------------------------------
# Enable the interrupt of Keyboard matrix 4x4 of Digital Lab Sim
li $t1, IN_ADDRESS_HEXA_KEYBOARD
li $t3, 0x80 # bit 7 of = 1 to enable interrupt
sb $t3, 0($t1)
#---------------------------------------------------------
# No-end loop, main program, to demo the effective of interrupt
#---------------------------------------------------------
Loop: nop
nop
nop
nop
b Loop # Wait for interrupt
end_main:
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# GENERAL INTERRUPT SERVED ROUTINE for all interrupts
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
.ktext 0x80000180
#--------------------------------------------------------
# Processing
#--------------------------------------------------------
IntSR: addi $v0, $zero, 4 # show message
la $a0, Message
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
return: eret # Return from exception
```

- *Kết quả:*


D:\OneDrive - Hanoi University of Science and Technology\Desktop\TH_KTMT\Week11\mips2.asm - MARS 4.5
File Edit Run Settings Tools Help
Run speed at max (no interaction)
Edit
Execute
Text Segment
Labels
☑ Data ☑ Text
Data Segment
0x10010000 (.data)
☑ Hexadecimal Addresses ☐ Hexadecimal Values ☐ ASCII
Mars Messages
Run I/O
Clear
Registers
Coproc 1
Coproc 0
Digital Lab Sim, Version 1.0 (Didier Teifreto)
Digital Lab Sim
8.8.
0 1 2 3
4 5 6 7
8 9 a b
c d e f
Tool Control
Disconnect from MIPS
Reset
Help
Clo

Text Segment

Bkpt	Address	Code	Basic	Source
☐	0x00400000	0x3c01ffff	lui $1,-1	13: li $t1, 0xFFFF0012
☐	0x00400004	0x34290012	ori $9,$1,18	
☐	0x00400008	0x240b0080	addiu $11,$0,128	14: li $t3, 0x80 # bit 7 of = 1 to enable interrupt
☐	0x0040000c	0xa12b0000	sb $11,0($9)	15: sb $t3, 0($t1)
☐	0x00400010	0x00000000	nop	19: Loop: nop
☐	0x00400014	0x00000000	nop	20: nop
☐	0x00400018	0x00000000	nop	21: nop
☐	0x0040001c	0x00000000	nop	22: nop
☐	0x00400020	0x0401fffb	bgez $0,-5	23: b Loop # Wait for interrupt
☐	0x80000180	0x20020004	addi $2,$0,4	32: IntSR: addi $v0, $zero, 4 # show message
☐	0x80000184	0x3c011001	lui $1,4097	33: la $a0, Message
☐	0x80000188	0x34240000	ori $4,$1,0	
☐	0x8000018c	0x0000000c	syscall	34: syscall
☐	0x80000190	0x40017000	mfc0 $1,$14	39: next_pc:mfc0 $at, $14 # $at <= Coproc0.$14 = Coproc0.epc
☐	0x80000194	0x20210004	addi $1,$1,4	40: addi $at, $at, 4 # $at = $at + 4 (next instruction)
☐	0x80000198	0x40817000	mtc0 $1,$14	41: mtc0 $at, $14 # Coproc0.$14 = Coproc0.epc <= $at
☐	0x8000019c	0x42000018	eret	42: return: eret # Return from exception

Labels

Label	Address ▲
mips2.asm	
main	0x00400000
Loop	0x00400010
end_main	0x00400024
Message	0x10010000
IntSR	0x80000180
next_pc	0x80000190
return	0x8000019c

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	1830840399	1869029497	1394617956	1868918127	1931961710	1701998624	543450483	1969365089
0x10010020	1852798068	2606	0	0	0	0	0	0
0x10010040	0	0	0	0	0	0	0	0
0x10010060	0	0	0	0	0	0	0	0
0x10010080	0	0	0	0	0	0	0	0
0x100100a0	0	0	0	0	0	0	0	0
0x100100c0	0	0	0	0	0	0	0	0
0x100100e0	0	0	0	0	0	0	0	0
0x10010100	0	0	0	0	0	0	0	0
0x10010120	0	0	0	0	0	0	0	0
0x10010140	0	0	0	0	0	0	0	0
0x10010160	0	0	0	0	0	0	0	0
0x10010180	0	0	0	0	0	0	0	0
0x100101a0	0	0	0	0	0	0	0	0

Run I/O

```
Oh my god. Someone's presed a button.
Oh my god. Someone's presed a button.
Oh my god. Someone's presed a button.
Oh my god. Someone's presed a button.
Oh my god. Someone's presed a button.
Oh my god. Someone's presed a button.
Oh my god. Someone's presed a button.

-- program is finished running (dropped off bottom) --
```

Registers

Name	Number	Value
$zero	0	0
$at	1	4194340
$v0	2	4
$v1	3	0
$a0	4	268500992
$a1	5	0
$a2	6	0
$a3	7	0
$t0	8	0
$t1	9	-65518
$t2	10	0
$t3	11	128
$t4	12	0
$t5	13	0
$t6	14	0
$t7	15	0
$s0	16	0
$s1	17	0
$s2	18	0
$s3	19	0
$s4	20	0
$s5	21	0
$s6	22	0
$s7	23	0
$t8	24	0
$t9	25	0
$k0	26	0
$k1	27	0
$gp	28	268468224
$sp	29	2147479548
$fp	30	0
$ra	31	0
pc		4194340
hi		0
lo		0

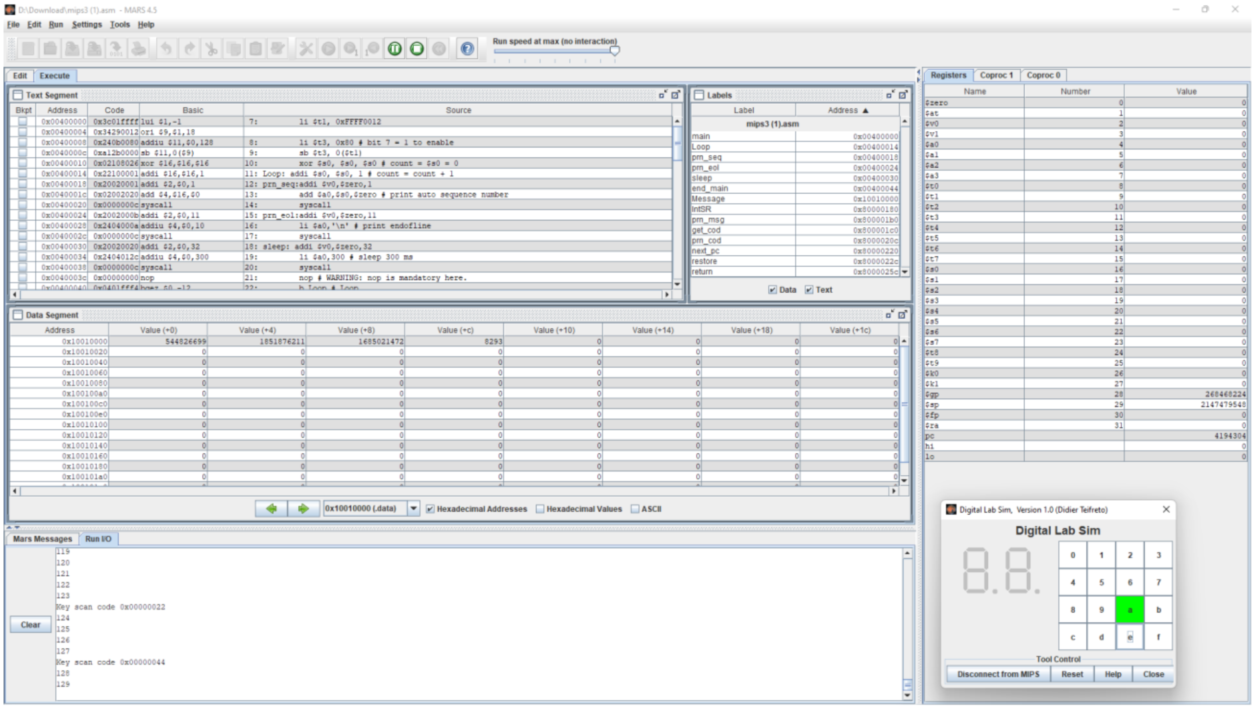
Assignment 3:

- *Chương trình:*

```
.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 $ra because we may change it later
        sw $ra,0($sp)
        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
get_cod:li $t2, IN_ADDRESS_HEXA_KEYBOARD
        li $t3, 0x81             # check row 4 and re-enable bit 7
        sb $t3, 0($t2)           # must reassign expected row
        li $t1, OUT_ADDRESS_HEXA_KEYBOARD
        lb $a0, 0($t1)
        bne     $a0, $0, prn_cod
        li $t3, 0x82             # check row 4 and re-enable bit 7
        sb $t3, 0($t2)           # must reassign expected row
        lb $a0, 0($t1)
        bne $a0, $0, prn_cod
        li $t3, 0x84             # check row 4 and re-enable bit 7
        sb $t3, 0($t2)           # must reassign expected row
        lb $a0, 0($t1)
        bne $a0, $0, prn_cod
        li $t3, 0x88             # check row 4 and re-enable bit 7
        sb $t3, 0($t2)           # must reassign expected row
        lb $a0, 0($t1)
prn_cod:li $v0,34
        syscall
        li $v0,11
        li $a0,'\n'              # print endofline
        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 $ra, 0($sp)           # Restore the registers from stack
        addi $sp,$sp,-4
        lw $ra, 0($sp)           # Restore the registers from stack
        addi $sp,$sp,-4
return: eret                     # Return from exception
```

- *Kết quả:*



D:\Download\mips3 (1).asm - MARS 4.5
File Edit Run Settings Tools Help
Run speed at max (no interaction)
Edit Execute
Text Segment
Labels
Registers Coproc 1 Coproc 0
Data Segment
Mars Messages Run I/O
Clear

Bkpt	Address	Code	Basic	Source
□	0x00400000	0x3c01ffff	lui $1,-1	7: li $t1, 0xFFFF0012
□	0x00400004	0x34290012	ori $9,$1,18	
□	0x00400008	0x240b0080	addiu $11,$0,128	8: li $t3, 0x80 # bit 7 = 1 to enable
□	0x0040000c	0xa12b0000	sb $11,0($9)	9: sb $t3, 0($t1)
□	0x00400010	0x02108026	xor $16,$16,$16	10: xor $s0, $s0, $s0 # count = $s0 = 0
□	0x00400014	0x22100001	addi $16,$16,1	11: Loop: addi $s0, $s0, 1 # count = count + 1
□	0x00400018	0x20020001	addi $2,$0,1	12: prn_seq:addi $v0,$zero,1
□	0x0040001c	0x02002020	add $4,$16,$0	13: add $a0,$s0,$zero # print auto sequence number
□	0x00400020	0x0000000c	syscall	14: syscall
□	0x00400024	0x2002000b	addi $2,$0,11	15: prn_eol:addi $v0,$zero,11
□	0x00400028	0x2404000a	addiu $4,$0,10	16: li $a0,'\n' # print endofline
□	0x0040002c	0x0000000c	syscall	17: syscall
□	0x00400030	0x20020020	addi $2,$0,32	18: sleep: addi $v0,$zero,32
□	0x00400034	0x2404012c	addiu $4,$0,300	19: li $a0,300 # sleep 300 ms
□	0x00400038	0x0000000c	syscall	20: syscall
□	0x0040003c	0x00000000	nop	21: nop # WARNING: nop is mandatory here.
□	0x00400040	0x0401fff4	bgez $0,-12	22: b Loop # Loop

Label	Address ▲
mips3 (1).asm	
main	0x00400000
Loop	0x00400014
prn_seq	0x00400018
prn_eol	0x00400024
sleep	0x00400030
end_main	0x00400044
Message	0x10010000
IntSR	0x80000180
prn_msg	0x800001b0
get_cod	0x800001c0
prn_cod	0x8000020c
next_pc	0x80000220
restore	0x8000022c
return	0x8000025c

☑ Data ☑ Text

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	544826699	1851876211	1685021472	8293	0	0	0	0
0x10010020	0	0	0	0	0	0	0	0
0x10010040	0	0	0	0	0	0	0	0
0x10010060	0	0	0	0	0	0	0	0
0x10010080	0	0	0	0	0	0	0	0
0x100100a0	0	0	0	0	0	0	0	0
0x100100c0	0	0	0	0	0	0	0	0
0x100100e0	0	0	0	0	0	0	0	0
0x10010100	0	0	0	0	0	0	0	0
0x10010120	0	0	0	0	0	0	0	0
0x10010140	0	0	0	0	0	0	0	0
0x10010160	0	0	0	0	0	0	0	0
0x10010180	0	0	0	0	0	0	0	0
0x100101a0	0	0	0	0	0	0	0	0

0x10010000 (.data) ☑ Hexadecimal Addresses □ Hexadecimal Values □ ASCII

Name	Number	Value
$zero	0	0
$at	1	0
$v0	2	0
$v1	3	0
$a0	4	0
$a1	5	0
$a2	6	0
$a3	7	0
$t0	8	0
$t1	9	0
$t2	10	0
$t3	11	0
$t4	12	0
$t5	13	0
$t6	14	0
$t7	15	0
$s0	16	0
$s1	17	0
$s2	18	0
$s3	19	0
$s4	20	0
$s5	21	0
$s6	22	0
$s7	23	0
$t8	24	0
$t9	25	0
$k0	26	0
$k1	27	0
$gp	28	268468224
$sp	29	2147479548
$fp	30	0
$ra	31	0
pc		4194304
hi		0
lo		0

```
119
120
121
122
123
Key scan code 0x00000022
124
125
126
127
Key scan code 0x00000044
128
129
```

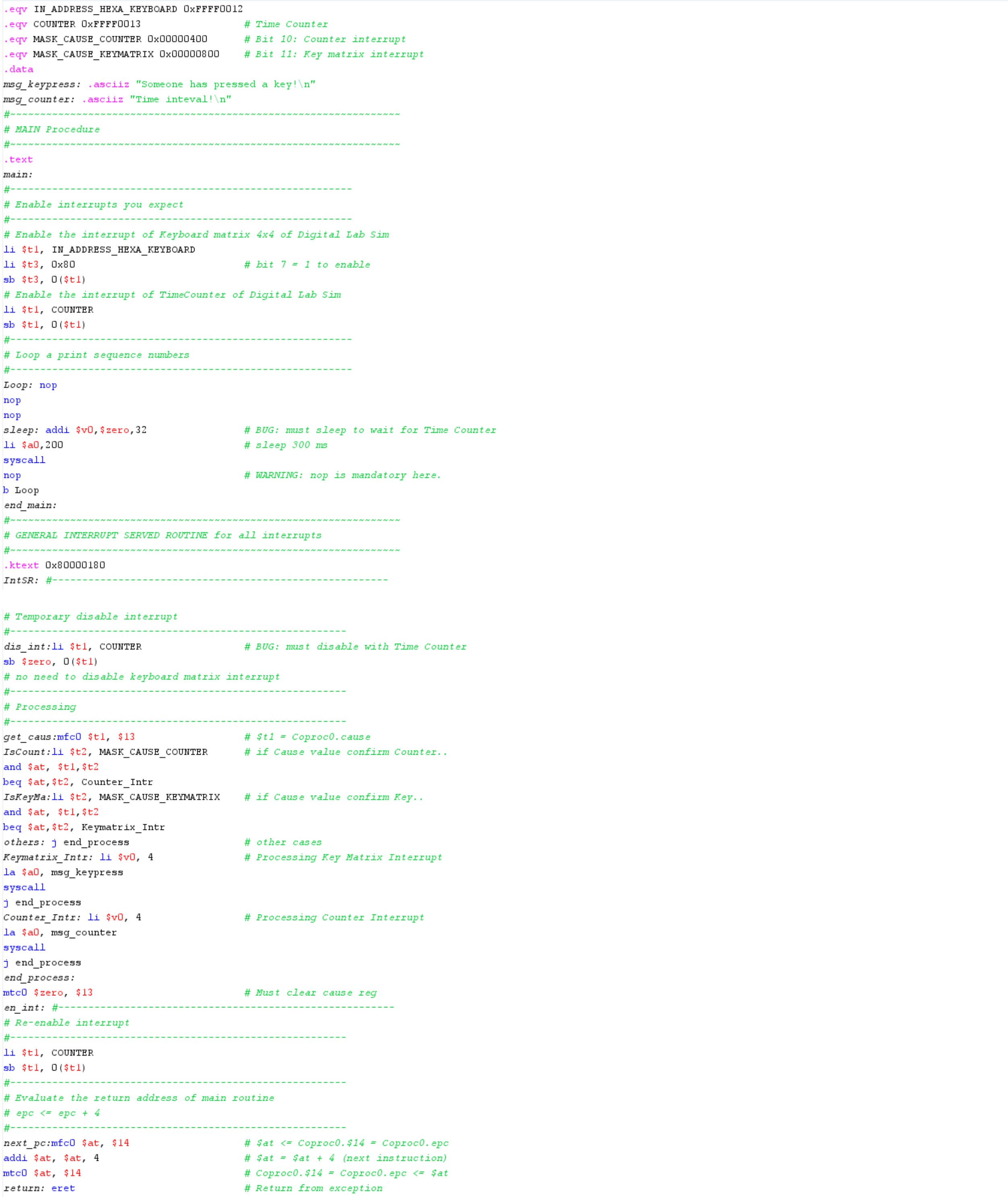
Digital Lab Sim, Version 1.0 (Didier Teifreto)

Digital Lab Sim

0	1	2	3
4	5	6	7
8	9	🟩 a	b
c	d	e	f

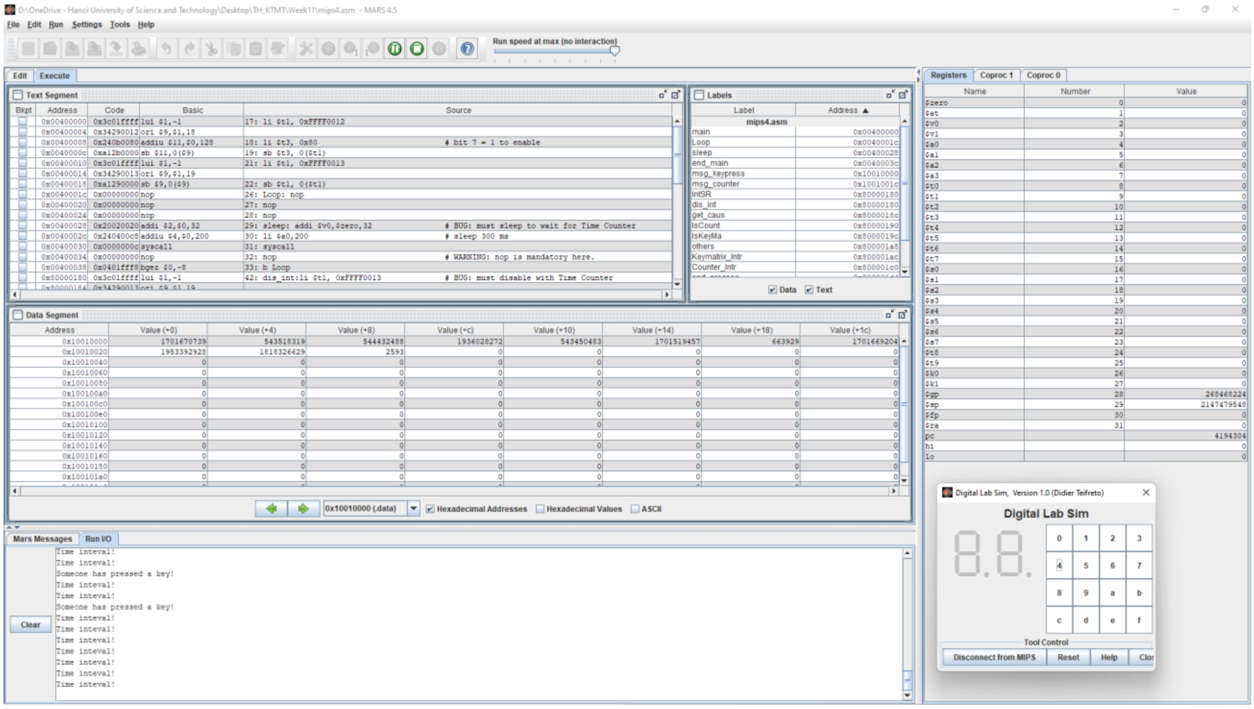
Assignment 4:

- *Chương trình:*



```
.eqv IN_ADDRESS_HEXA_KEYBOARD 0xFFFF0012
.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!\n"
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# MAIN Procedure
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
.text
main:
#---------------------------------------------------------
# Enable interrupts you expect
#---------------------------------------------------------
# Enable the interrupt of Keyboard matrix 4x4 of Digital Lab Sim
li $t1, IN_ADDRESS_HEXA_KEYBOARD
li $t3, 0x80                            # bit 7 = 1 to enable
sb $t3, 0($t1)
# Enable the interrupt of TimeCounter of Digital Lab Sim
li $t1, COUNTER
sb $t1, 0($t1)
#---------------------------------------------------------
# Loop a print sequence numbers
#---------------------------------------------------------
Loop: nop
nop
nop
sleep: addi $v0,$zero,32                # BUG: must sleep to wait for Time Counter
li $a0,200                              # 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
j end_process
Counter_Intr: li $v0, 4                 # Processing Counter Interrupt
la $a0, msg_counter
syscall
j end_process
end_process:
mtc0 $zero, $13                         # Must clear cause reg
en_int: #--------------------------------------------------------
# Re-enable interrupt
#--------------------------------------------------------
li $t1, COUNTER
sb $t1, 0($t1)
#--------------------------------------------------------
# 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
return: eret                            # Return from exception
```

- *Kết quả:*


D:\OneDrive - Hanoi University of Science and Technology\Desktop\TH_KTMT\Week11\mips4.asm - MARS 4.5
File
Edit
Run
Settings
Tools
Help
Run speed at max (no interaction)
Edit
Execute
Registers
Coproc 1
Coproc 0
Mars Messages
Run I/O
Clear
Digital Lab Sim, Version 1.0 (Didier Teifreto)
Digital Lab Sim
0
1
2
3
4
5
6
7
8
9
a
b
c
d
e
f
Tool Control
Disconnect from MIPS
Reset
Help
Clo

Text Segment

Bkpt	Address	Code	Basic	Source
□	0x00400000	0x3c01ffff	lui $1,-1	17: li $t1, 0xFFFF0012
□	0x00400004	0x34290012	ori $9,$1,18	
□	0x00400008	0x240b0080	addiu $11,$0,128	18: li $t3, 0x80 # bit 7 = 1 to enable
□	0x0040000c	0xa12b0000	sb $11,0($9)	19: sb $t3, 0($t1)
□	0x00400010	0x3c01ffff	lui $1,-1	21: li $t1, 0xFFFF0013
□	0x00400014	0x34290013	ori $9,$1,19	
□	0x00400018	0xa1290000	sb $9,0($9)	22: sb $t1, 0($t1)
□	0x0040001c	0x00000000	nop	26: Loop: nop
□	0x00400020	0x00000000	nop	27: nop
□	0x00400024	0x00000000	nop	28: nop
□	0x00400028	0x20020020	addi $2,$0,32	29: sleep: addi $v0,$zero,32 # BUG: must sleep to wait for Time Counter
□	0x0040002c	0x240400c8	addiu $4,$0,200	30: li $a0,200 # sleep 300 ms
□	0x00400030	0x0000000c	syscall	31: syscall
□	0x00400034	0x00000000	nop	32: nop # WARNING: nop is mandatory here.
□	0x00400038	0x0401fff8	bgez $0,-8	33: b Loop
□	0x80000180	0x3c01ffff	lui $1,-1	42: dis_int:li $t1, 0xFFFF0013 # BUG: must disable with Time Counter
□	0x80000184	0x34290013	ori $9,$1,19	

Labels

Label	Address ▲
mips4.asm	
main	0x00400000
Loop	0x0040001c
sleep	0x00400028
end_main	0x0040003c
msg_keypress	0x10010000
msg_counter	0x1001001c
IntSR	0x80000180
dis_int	0x80000180
get_caus	0x8000018c
IsCount	0x80000190
IsKeyMa	0x8000019c
others	0x800001a8
Keymatrix_Intr	0x800001ac
Counter_Intr	0x800001c0

☑ Data ☑ Text

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	1701670739	543518319	544432488	1936028272	543450483	1701519457	663929	1701669204
0x10010020	1953392928	1818326629	2593	0	0	0	0	0
0x10010040	0	0	0	0	0	0	0	0
0x10010060	0	0	0	0	0	0	0	0
0x10010080	0	0	0	0	0	0	0	0
0x100100a0	0	0	0	0	0	0	0	0
0x100100c0	0	0	0	0	0	0	0	0
0x100100e0	0	0	0	0	0	0	0	0
0x10010100	0	0	0	0	0	0	0	0
0x10010120	0	0	0	0	0	0	0	0
0x10010140	0	0	0	0	0	0	0	0
0x10010160	0	0	0	0	0	0	0	0
0x10010180	0	0	0	0	0	0	0	0
0x100101a0	0	0	0	0	0	0	0	0

0x10010000 (.data) ☑ Hexadecimal Addresses □ Hexadecimal Values □ ASCII

Name	Number	Value
$zero	0	0
$at	1	0
$v0	2	0
$v1	3	0
$a0	4	0
$a1	5	0
$a2	6	0
$a3	7	0
$t0	8	0
$t1	9	0
$t2	10	0
$t3	11	0
$t4	12	0
$t5	13	0
$t6	14	0
$t7	15	0
$s0	16	0
$s1	17	0
$s2	18	0
$s3	19	0
$s4	20	0
$s5	21	0
$s6	22	0
$s7	23	0
$t8	24	0
$t9	25	0
$k0	26	0
$k1	27	0
$gp	28	268468224
$sp	29	2147479548
$fp	30	0
$ra	31	0
pc		4194304
hi		0
lo		0

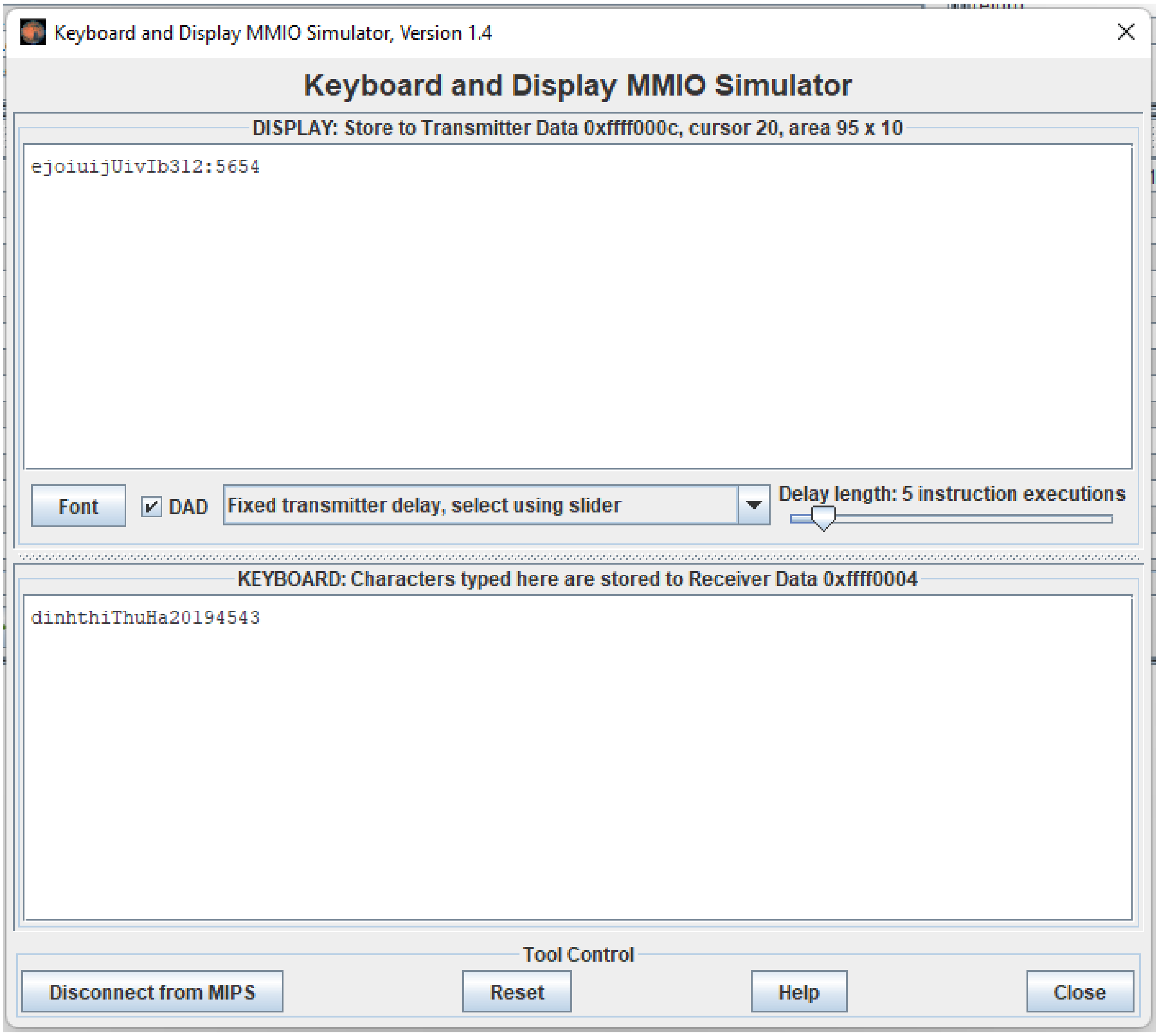
```
Time inteval!
Time inteval!
Someone has pressed a key!
Time inteval!
Time inteval!
Someone has pressed a key!
Time inteval!
Time inteval!
Time inteval!
Time inteval!
Time inteval!
Time inteval!
Time inteval!
```

Assignment 5:

- *Chương trình:*

```
.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 20, area 95 x 10

```
ejoiuijUivIb312:5654
```

☑ DAD

Fixed transmitter delay, select using slider

Delay length: 5 instruction executions

KEYBOARD: Characters typed here are stored to Receiver Data 0xffff0004

```
dinhthiThuHa20194543
```

Font
Tool Control
Disconnect from MIPS
Reset
Help
Close