Báo cáo tuần 11

Phùng Ngọc Vinh – 20194719

Bài 1:

Create a new project, type in, and build the program of Home Assignment 1. Upgrade the source code so that it could defect all 16 key buttons, from 0 to F.

*Mã nguồn:

```
#-----------------------------------------------
#               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
	li $t3, 0x08   # check row 4 with key C, D, E, F
	li $t4, 0x1
	li $t5, 0x2
	li $t6, 0x4
polling:
	sb $t3, 0($t1)		# must reassign expected row
	lb $a0,	0($t2)		# read scan code of key button
	bnez $a0, print

	sb $t4, 0($t1)		# must reassign expected row
	lb $a0,	0($t2)		# read scan code of key button
	bnez $a0, print

	sb $t5, 0($t1)		# must reassign expected row
```

```
        lb $a0,     0($t2)      # read scan code of key button
        bnez $a0, print

        sb $t6, 0($t1)          # must reassign expected row
        lb $a0,     0($t2)      # read scan code of key button
        bnez $a0, print

print:
        li $v0, 34              # print integer (hexa)
        syscall
sleep:
        li $a0, 100             # sleep 100ms
        li $v0, 32
        syscall

back_to_polling:
        j polling               # continue polling
```

*Giải thích:

- Các giá trị 0x1, 0x2, 0x4, 0x8 tương ứng với các row 1, 2, 3, 4 trong Digital lab slim.
- Trong mỗi row, $a0 nhận được pressed (tức là giá trị khác 0) thì sẽ được in ra màn hình console.

Bài 2:

Create a new project, type in, and build the program of Home Assignment 2.

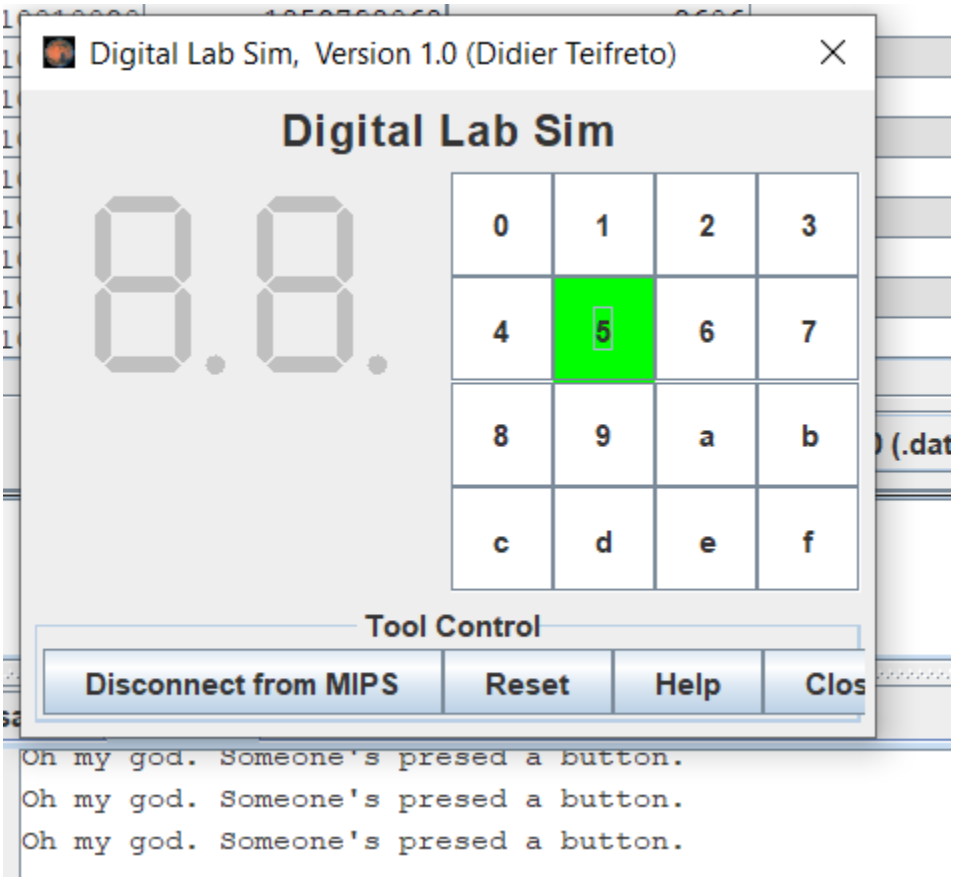
*Mã nguồn:

```
.eqv IN_ADRESS_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_ADRESS_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:



Bài 3:

Create a new project, type in, and build the program of Home Assignment 3. Upgrade the source code so that it could defect all 16 key buttons, from 0 to F.

*Mã nguồn:

```
.eqv IN_ADRESS_HEXA_KEYBOARD 0xFFFF0012
.eqv OUT_ADRESS_HEXA_KEYBOARD 0xFFFF0014
.data
        Message: .asciiz "Key scan code "
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

```
# 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)
        #-----------------------------------------------------
        # 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               # print endofline
        li $a0,'\n'
        syscall
```

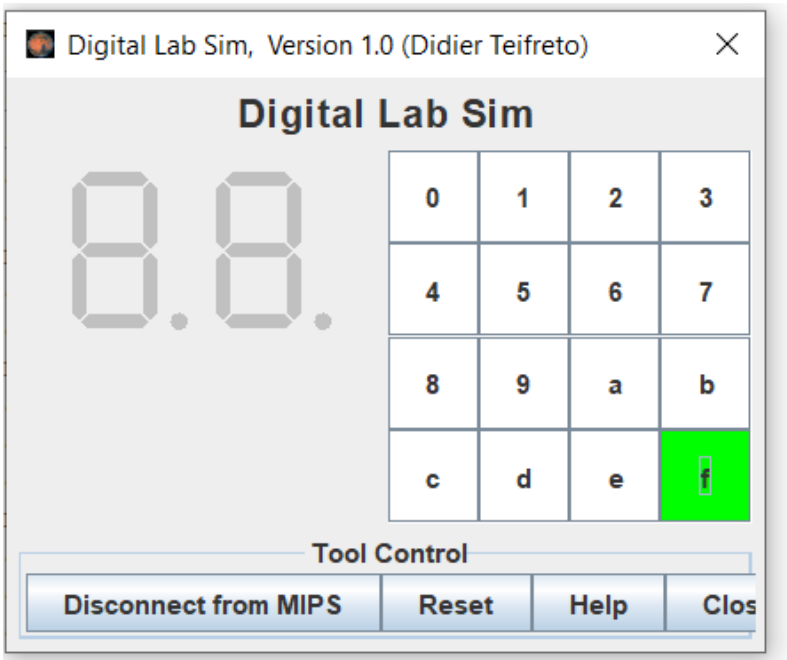
```
sleep:
        addi $v0,$zero,32
        li $a0,300              # sleep 300 ms
        syscall
        nop                     # WARNING: nop is mandatory here.
        b Loop                          # Loop
end_main:


#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# GENERAL INTERRUPT SERVED ROUTINE for all interrupts
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
.ktext 0x80000180
        #---------------------------------------------------
        # SAVE the current REG FILE to stack
        #---------------------------------------------------
IntSR:
        addi $sp,$sp,4                  # Save $ra 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)
        #---------------------------------------------------
        # Processing
        #---------------------------------------------------
prn_msg:
        addi $v0, $zero, 4
        la $a0, Message
        syscall
get_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)
        bnez $a0, prn_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)
        bnez $a0, prn_cod

        li $t1, IN_ADRESS_HEXA_KEYBOARD
        li $t3, 0x82                    # 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)
        bnez $a0, prn_cod

        li $t1, IN_ADRESS_HEXA_KEYBOARD
        li $t3, 0x84                # 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)
        bnez $a0, 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 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:




```
Key scan code 0xffffff81
70
71
72
Key scan code 0xffffff82
73
74
75
Key scan code 0xffffff84
76
77
Key scan code 0xffffff88
78
79
80
81
82
```

*Giải thích:

- Gán $t3 lần lượt là 0x81, 0x82, 0x84, 0x88 tương ứng với row 1, 2, 3, 4
- Trong mỗi row, $a0 nhận được pressed (tức là giá trị khác 0) thì sẽ được in ra màn hình console.

Bài 4:

Create a new project, type in, and build the program of Home Assignment 4.
*Mã nguồn:

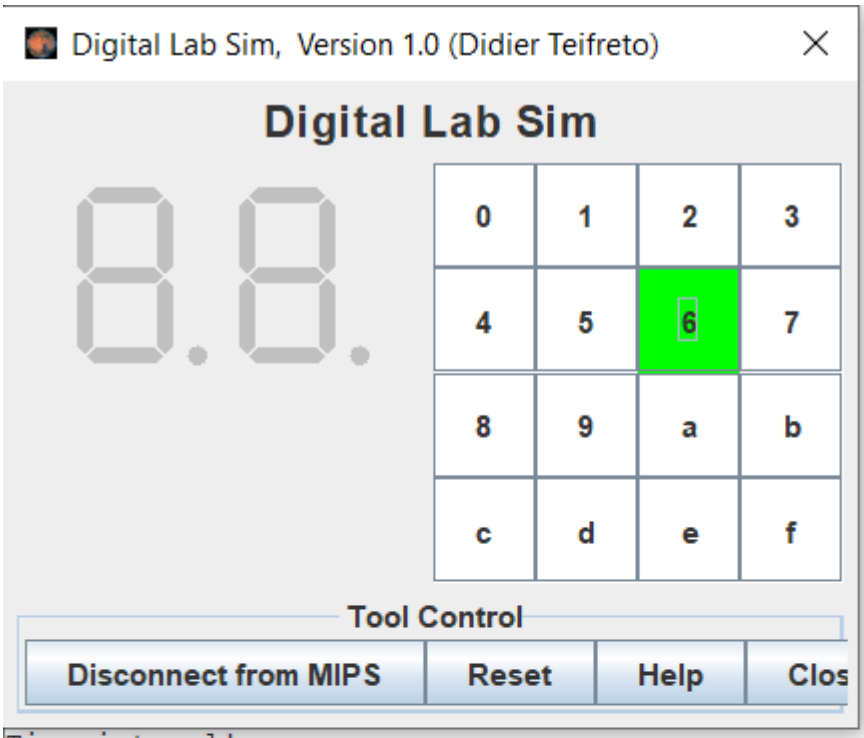
```
.eqv IN_ADRESS_HEXA_KEYBOARD 0xFFFF0012
.eqv COUNTER 0xFFFF0013                         # Time Counter

.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_ADRESS_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 an 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:




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

Bài 5:

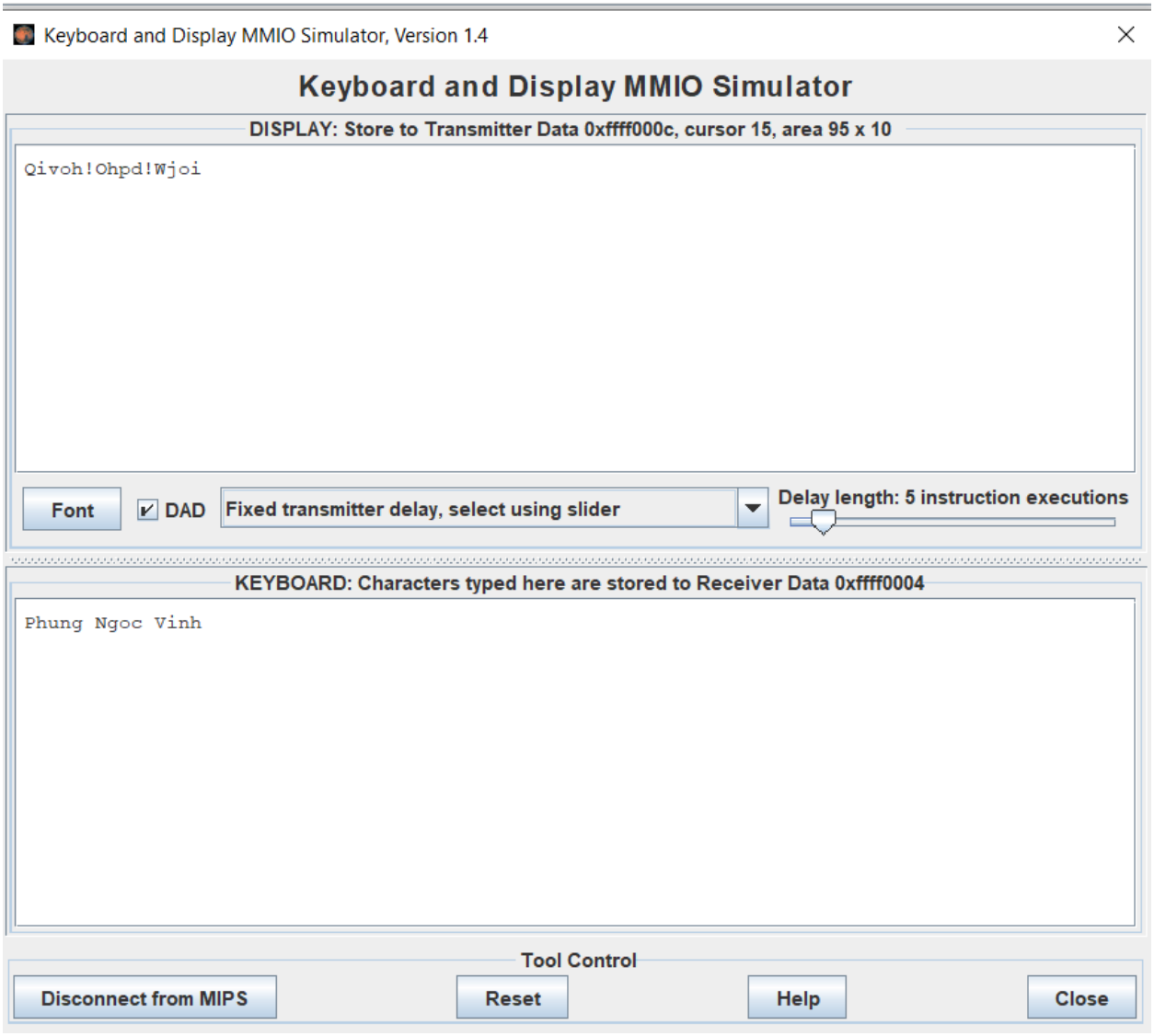
Create a new project, type in, and build the program of Home Assignment 5.

*Mã nguồn:

```
.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
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 15, area 95 x 10
Qivoh!Ohpd!Wjoi
Font
DAD
Fixed transmitter delay, select using slider
Delay length: 5 instruction executions
KEYBOARD: Characters typed here are stored to Receiver Data 0xffff0004
Phung Ngoc Vinh
Tool Control
Disconnect from MIPS
Reset
Help
Close