Lab Report 11

## Assignment 1

**Code:**

.eqv IN\_ADDRESS\_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\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0014

.text

main:

li $t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li $t2, OUT\_ADDRESS\_HEXA\_KEYBOARD

polling:

check01:

li $t3, 0x01 # check 0, 1, 2, 3

sb $t3, 0($t1) # must reassign expected row

lbu $a0, 0($t2) # read scan code of key button

bne $a0, 0, print

check02:

li $t3, 0x02 # check 4, 5, 6, 7

sb $t3, 0($t1) # must reassign expected row

lbu $a0, 0($t2) # read scan code of key button

bne $a0, 0, print

check03:

li $t3, 0x04 # check 8, 9, a, b

sb $t3, 0($t1) # must reassign expected row

lbu $a0, 0($t2) # read scan code of key button

bne $a0, 0, print

check04:

li $t3, 0x08 # check c, d, e, f

sb $t3, 0($t1) # must reassign expected row

lbu $a0, 0($t2) # read scan code of key button

print:

li $v0, 34 # print integer (hexa)

syscall

sleep:

li $a0, 1000 # sleep 1000ms

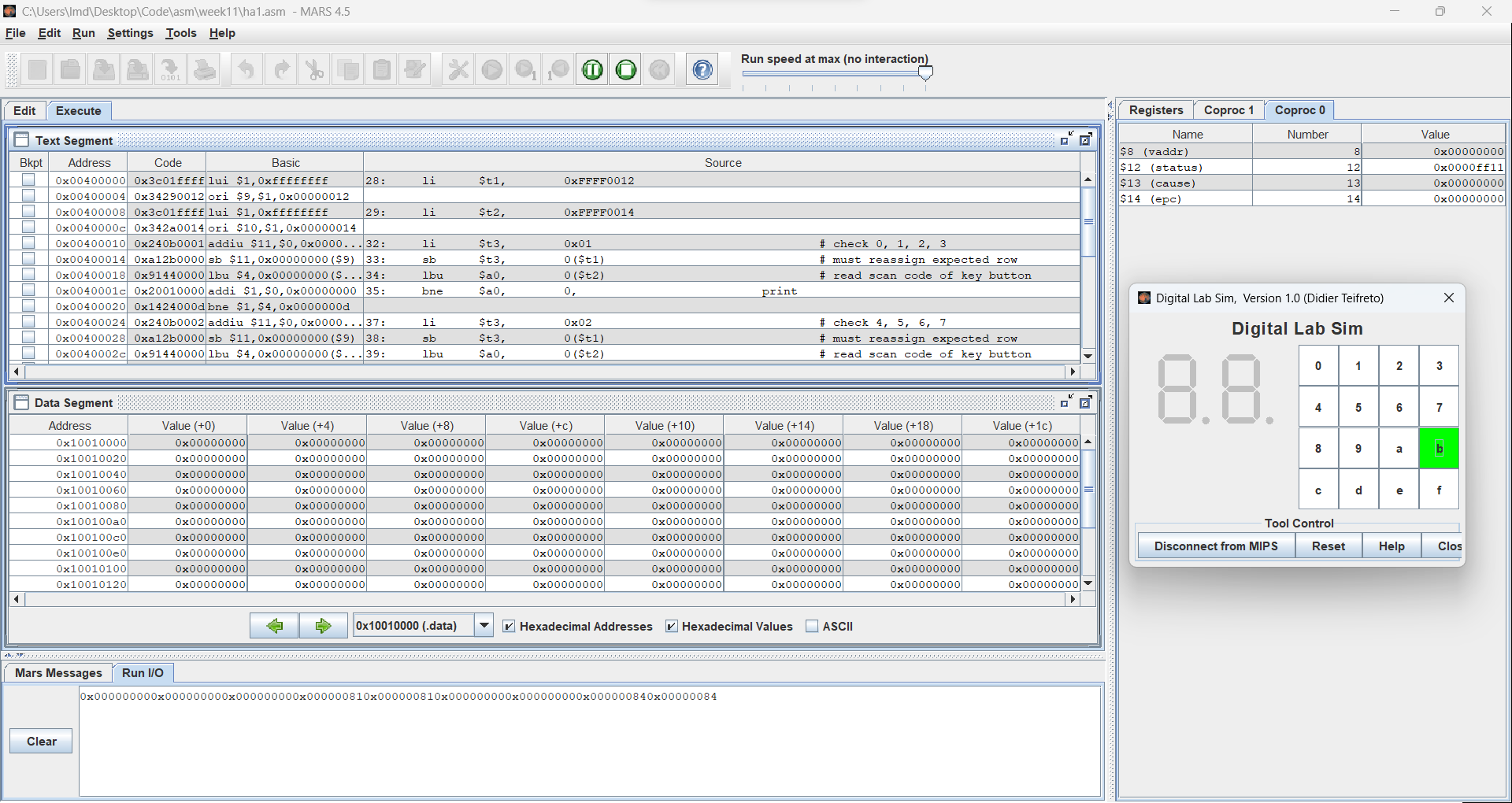
li $v0, 32

syscall

back\_to\_polling:

j polling # continue polling

**Results:**

****

**Comments:**

* Because of the polling technique, there are no activities recorded in Coproc 0.

## Assignment 2

**Comments:**

* Set a breakpoint for the interruption.
* After enabling interruption, the code will keep sleeping until an interruption happens.
* If an interruption happens, the execution will pause at the specified breakpoint. **Graphical user interface, application

  Description automatically generated**
* Here, $12 = 0x0000ff13, $13 = 0x00000800, $14 = 0x00400024.
* After this, the program prints the message and updates $14 to 0x00400028 and jump to code at the address 0x00400028.

**Graphical user interface, application

Description automatically generated**

## Assignment 3

**Code:**

.eqv IN\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0012

.eqv OUT\_ADDRESS\_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\_ADDRESS\_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, 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 $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

check01:

li $t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li $t3, 0x81 # check row 1 and re-enable bit 7

sb $t3, 0($t1) # must reassign expected row

li $t1, OUT\_ADDRESS\_HEXA\_KEYBOARD

lbu $a0, 0($t1)

bne $a0, 0, prn\_cod

check02:

li $t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li $t3, 0x82

sb $t3, 0($t1) # must reassign expected row

li $t1, OUT\_ADDRESS\_HEXA\_KEYBOARD

lbu $a0, 0($t1)

bne $a0, 0, prn\_cod

check03:

li $t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li $t3, 0x84

sb $t3, 0($t1) # must reassign expected row

li $t1, OUT\_ADDRESS\_HEXA\_KEYBOARD

lbu $a0, 0($t1)

bne $a0, 0, prn\_cod

check04:

li $t1, IN\_ADDRESS\_HEXA\_KEYBOARD

li $t3, 0x88

sb $t3, 0($t1) # must reassign expected row

li $t1, OUT\_ADDRESS\_HEXA\_KEYBOARD

lbu $a0, 0($t1)

prn\_cod:

li $v0, 34

syscall

li $v0, 11

li $a0, '\n' # print end of line

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

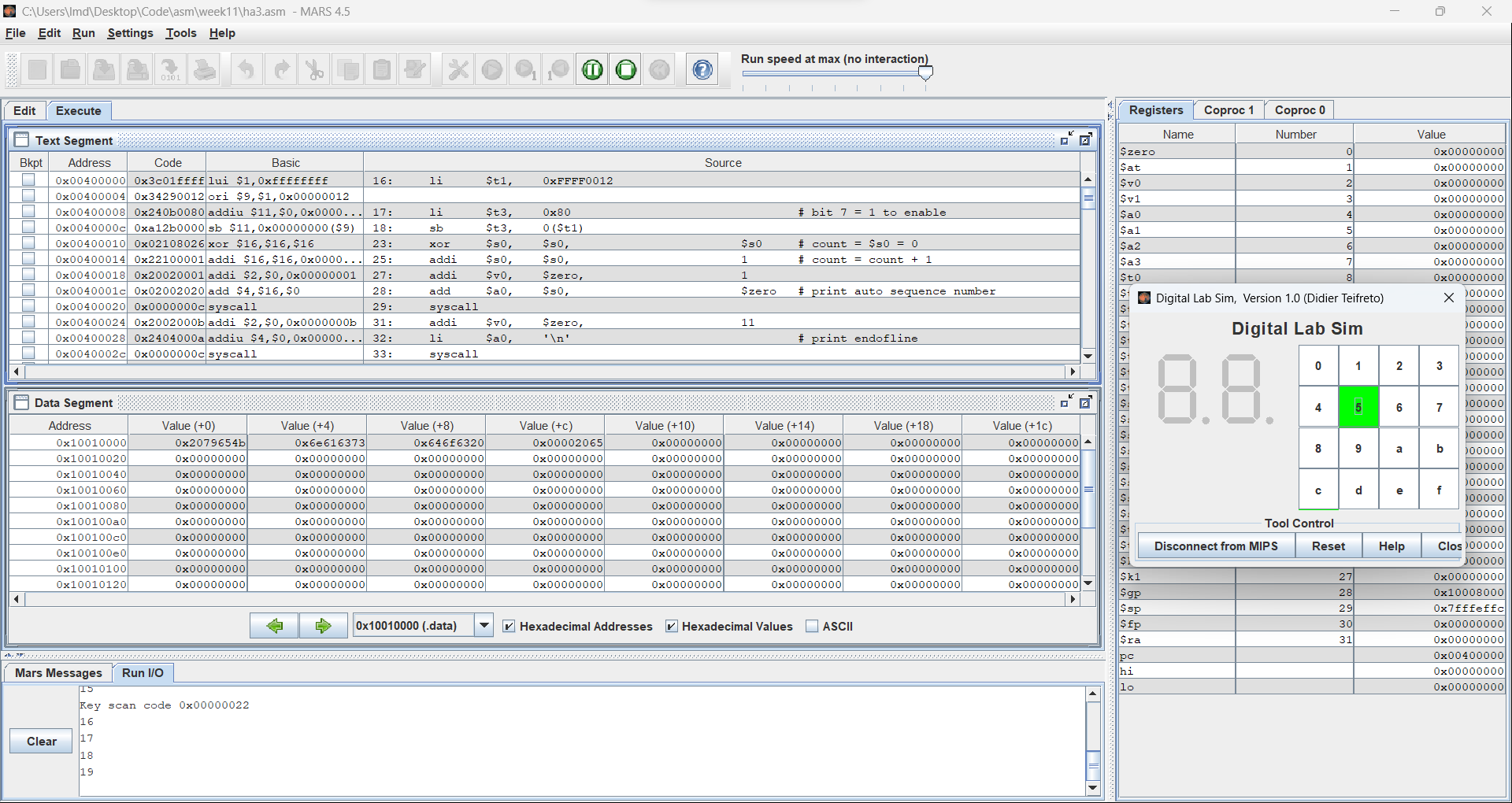
lw $at, 0($sp) # Restore the registers from stack

addi $sp, $sp, -4

return:

eret # Return from exception

**Results:**

****

**Comments:**

* Clicking a button will interrupt the program and print the button code to the console.
* Set a breakpoint to see changes in Coproc 0.

A screenshot of a computer

Description automatically generated with medium confidence

* Here, clicking 5 will make the program output “Key scan code 0x00000022”.
* In Coproc 0, $12 = 0x0000ff13, $13 = 0x00000800, $14 = 0040003c.

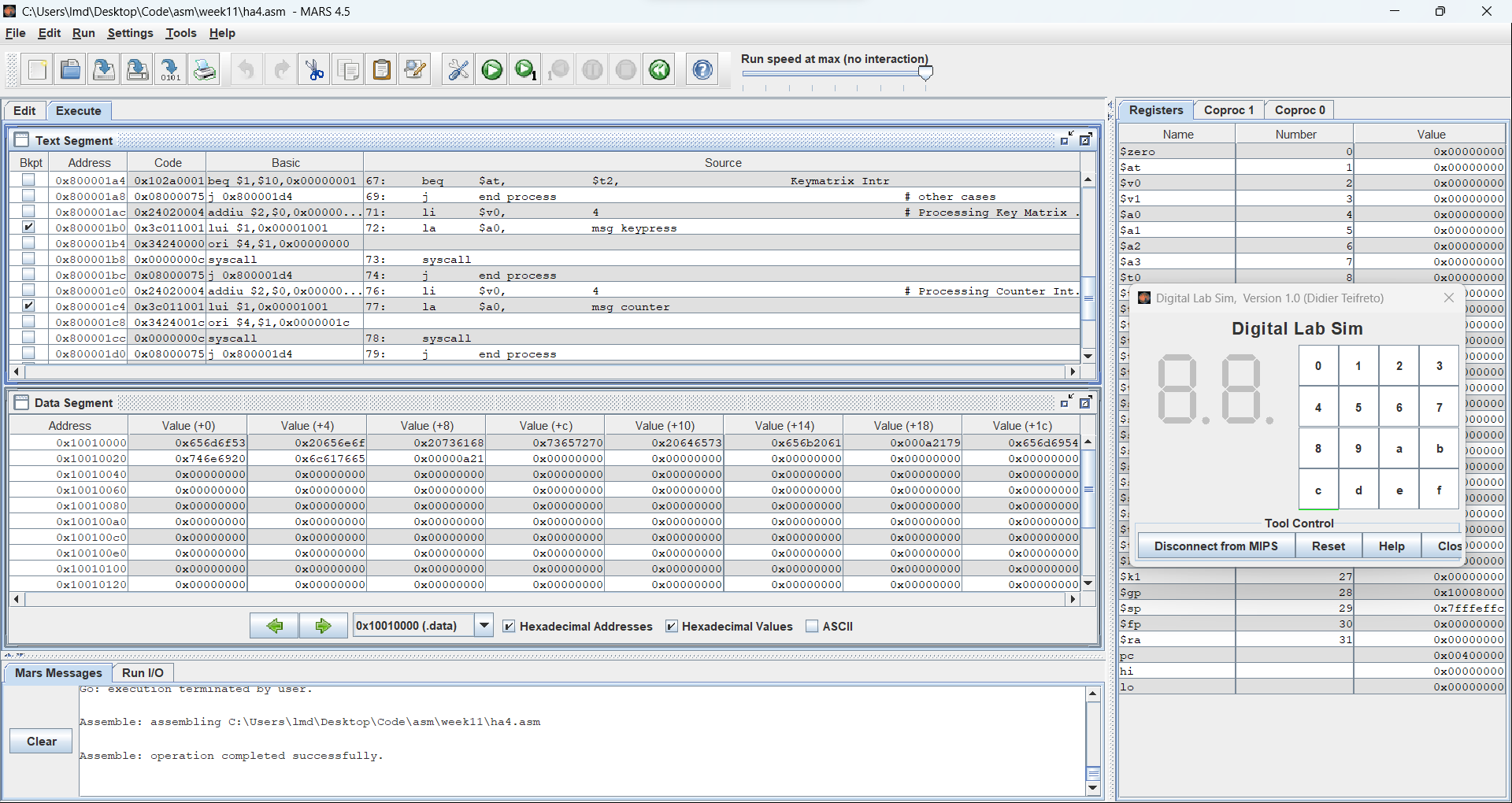
A screenshot of a computer

Description automatically generated

## Assignment 4

**Comments:**

* Set a breakpoint for each interruption.

****

* If time counter interruption happens, $13 = 0x00000400 and the program outputs “Time interval!”.

**Graphical user interface, application

Description automatically generated**

* If key matrix interruption happens, $13 = 0x00000800 and the program outputs “Someone has pressed a key!”.
* **Graphical user interface, application, table, Excel

  Description automatically generated**

## Assignment 5

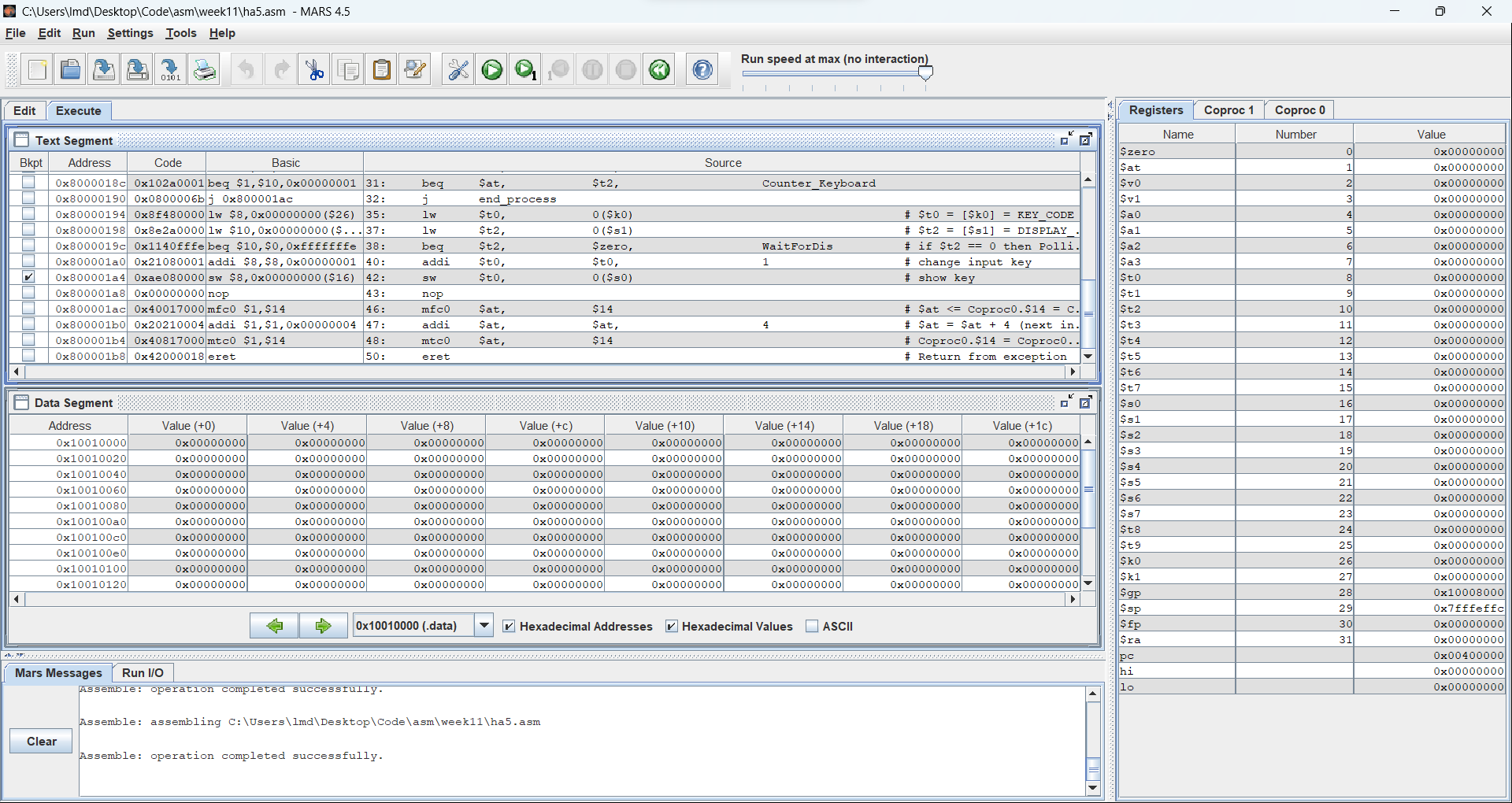
**Results:**

**A screenshot of a computer

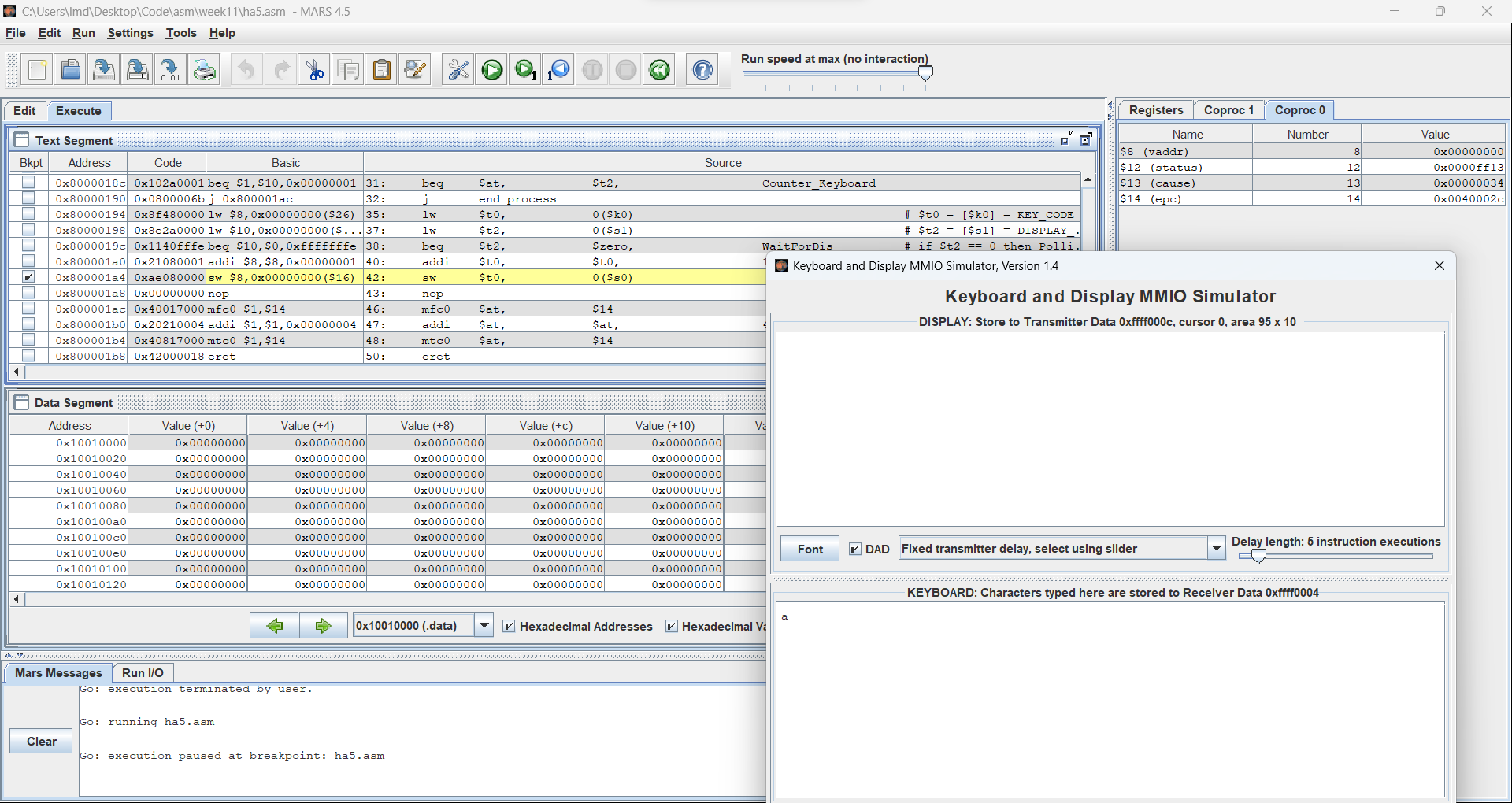
Description automatically generated**

**Comments:**

* Set breakpoint for the interruption.

****

* If the interruption happens, $13 = 0x00000034 and the program outputs the encoded character.

****