HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

SCHOOL OF INFORMATION AND TECHNOLOGY

**COMPUTER ARCHITECTURE LAB**

**20192 FINAL REPORT**



Supervisor**: M.S NGUYEN THI THANH NGA**

GROUP 10

|  |  |
| --- | --- |
| Tran Manh Hung | 20176775 |
| Nguyen Trung Hau | 20176749 |
| Le Huy Van | 20176906 |

**MIPS REPORT – FINAL PROJECT 20192**

**1. Thông tin nhóm**

Tran Manh Hung – ICT.01–K62 – 20176775 – Project 3

Nguyen Trung Hau – ICT.01-K62 – 20176749 – Project 5

Le Huy Van – ICT.01-K62 – 20176906 – Project 8

**2. Báo cáo**

**2.1. Project 3**

* **Phân tích:**

Chương trình sẽ đo tốc độ gõ bàn phím và hiển thị kết quả bằng 2 đèn led 7 đoạn. Nguyên tắc:

- Cho một đoạn văn bản mẫu, cố định sẵn trong mã nguồn. Ví dụ *“bo mon ky thuat may tinh”*

- Đo thời gian gõ phím

- Trong thời khoảng đó, người dùng nhập các kí tự từ bàn phím. Ví dụ nhập “*bo m****O****n ky* ***5****huat may tinh*”. Chương trình cần phải đếm số kí tự đúng (trong ví dụ trên thì người dùng gõ sai chữ ***O*** và ***5***) mà người dùng đã gõ và hiển thị lên các đèn led.

* **Cách làm:**

**Bước 1:** Đo độ dài chuỗi nguồn.

**Bước 2:** Hiển thị chuỗi nguồn lên màn hình công cụ Keyboard and Display MMIO Simulator

**Bước 3:** Khi bắt đầu nhập phím từ công cụ keyboard thì lưu lại giờ bắt đầu.

**Bước 4:** Mỗi phím được nhập vào giả lập sẽ được lưu vào một chuỗi. Kết thúc khi nhấn enter hoặc đạt độ dài của chuỗi nguồn.

**Bước 5:** So sánh chuỗi đã lưu khi nhập từ giả lập với chuỗi nguồn. Tính số kí tự đúng, kết thúc thời gian để in ra LED và hộp thoại số kí tự đã nhập.

* **Mã nguồn:**

.eqv SEVENSEG\_LEFT 0xFFFF0011 # Address of left 7 led

.eqv SEVENSEG\_RIGHT 0xFFFF0010 # Address of right 7 led

.eqv KEY\_READY 0xFFFF0000 # =1 if has a new keycode

.eqv KEY\_CODE 0xFFFF0004 # ASCII code from keyboard, 1 byte

.eqv DISPLAY\_READY 0xFFFF0008 # =1 if the display has already to do

.eqv DISPLAY\_CODE 0xFFFF000C # ASCII code to show, 1 byte

.data

source: .asciiz "bo mon ky thuat may tinh"

input: .space 1000

bytehex: .byte 63,6,91,79,102,109,125,7,127,111 # value for SEVENSEG LED

message1: .asciiz "Total time of typing (ms): "

message2: .asciiz "The number of character you have typed:"

message3: .asciiz "Typing speed (char/ms): "

.text

addi $t5,$t5,1 # condition to start time when press the first key

################# Keyboard input

li $k0, KEY\_CODE

li $k1, KEY\_READY

li $s3, DISPLAY\_CODE

li $s4, DISPLAY\_READY

################# Get length of the source string

get\_source\_length:

la $a0, source # a0 = source[0]

la $a1, input # a1 = input[0]

li $t0, 0 # t0 index i

li $v0, 0 # string initial length = 0

check\_null:

add $t1, $a0, $t0 # t1 = address of (source[0] + i)

lb $t2, 0($t1) # t2 = source[i]

beqz $t2, end\_check\_null # If t2 = 0 ; t2 is a null char

addi $t0, $t0, 1 # index++

addi $v0, $v0, 1 # length++

j check\_null

end\_check\_null:

add $s5, $v0, $zero # s5 = source string length

li $s6, 0 # current number of char in input string

################# Initialize value

li $v0, 0 # reset length to 0

li $t0, 0 # reset index i to 0

li $s7, 0 # number of correct characters

################# Show the source string user have to input

li $s0, 0 # reset i = 0

display\_string: # display string in upper MMIO window

add $s1, $a0, $s0 # s1 = address if input[0] + i

lb $s2, 0($s1) # Load the word of the source

beqz $s2, end\_of\_display # Check null char

sw $s2, 0($s3) # display to screen

nop

addi $s0, $s0, 1 # index i++

j display\_string

nop

end\_of\_display:

li $s0, 0 # Reset: i = 0

################# Inputed key from keyboard

input\_loop:

nop

WaitForKey:

lw $t1, 0($k1) # $t1 = $k1 = KEY\_READY

nop

beqz $t1, WaitForKey # waiting input from users

nop

ReadKey:

lw $t0, 0($k0) # $t0 = $k0 = KEY\_CODE

nop

beqz $t5,WaitForDis # Start time first tiem after press first key

li $v0, 30 # Start time value in t8

syscall

addi $t8, $a0, 0

la $a1, input # reset a1 = input[0]

la $a0, source # reset a0 = source[0]

li $t5, 0

nop

WaitForDis:

lw $t2, 0($s4) # $t2 = $s4 = DISPLAY\_READY

nop

beq $t2, $zero, WaitForDis # if $t2 == 0 then Polling

nop

StoreKey:

add $s1, $a1, $s0 # s1 = address of input[0] + i

beq $t0, 10, LED # incase the input key is ENTER

beq $t0, 8, back\_space # incase the input key is BACKSPACE

sb $t0, 0($s1) # Store the input to the source String input[]

addi $s0, $s0, 1 # index i++

j continue

nop

back\_space:

beq $s0,0,input\_loop # if index 0 and backspace, jump to loop again

add $s0,$s0,-1 #i -= 1

add $s1,$a1,$s0 # Address

sb $zero, 0($s1) # Set previous character value to 0

addi $s6,$s6,-1 # Decrease index i--

j input\_loop

continue:

addi $s6,$s6,1 # number of char in input string ++

beq $s6,$s5,LED # When input String length reaches source string length

j input\_loop

nop

exit:

li $t9, 0

li $v0,30

syscall

add $t9,$a0,$zero # t9 = Stopping time

li $v0,56

sub $a1,$t9,$t8 # Calculate time interval

la $a0,message1 # Dialog shows typing time

syscall

li $v0, 56

li $a1, 0

add $a1, $s6, $a1

la $a0,message2 # Dialog shows number of typed characters

syscall

li $v0, 10 #terminate program

syscall

####################### Compare string and display LED section

LED:

addi $s0,$zero,0 # i = 0

# Comparing source string and input string

stringCompare:

add $s1, $a1, $s0 # Address of input[0] + i

add $s2, $a0, $s0 # Address of source[0] + i

lb $t1, 0($s1)

lb $t2, 0($s2)

beq $t1,$t2,correct

j nextStep

correct:

addi $s7,$s7,1 # Correct char++

nextStep:

addi $s0,$s0,1 # index i++

bne $s0,$s5, stringCompare # if i != strlen then continue compare

div $s7, $s7, 10 #divide the number of correct char by 10

mflo $t6 # t6 = quotient, displayed on Left LED

mfhi $t7 # t7 = remainder, displayed on Right LED

# load value to SEVENSEG\_LEFT

fetch\_left:

la $t9, bytehex # t9 = bytehex[0]

add $t9, $t9, $t6 # t9 = bytehex[i] ( index i= value of t6)

lb $a0, 0($t9) # load bytehex[i] into $a0

jal SHOW\_7SEG\_LEFT # display

nop

# load value to SEVENSEG\_RIGHT

fetch\_right:

la $t9, bytehex

add $t9, $t9, $t7

lb $a0, 0($t9)

jal SHOW\_7SEG\_RIGHT # dislay

nop

endLED:

j exit

SHOW\_7SEG\_LEFT:

li $t0, SEVENSEG\_LEFT

sb $a0, 0($t0)

nop

jr $ra

nop

SHOW\_7SEG\_RIGHT:

li $t0, SEVENSEG\_RIGHT

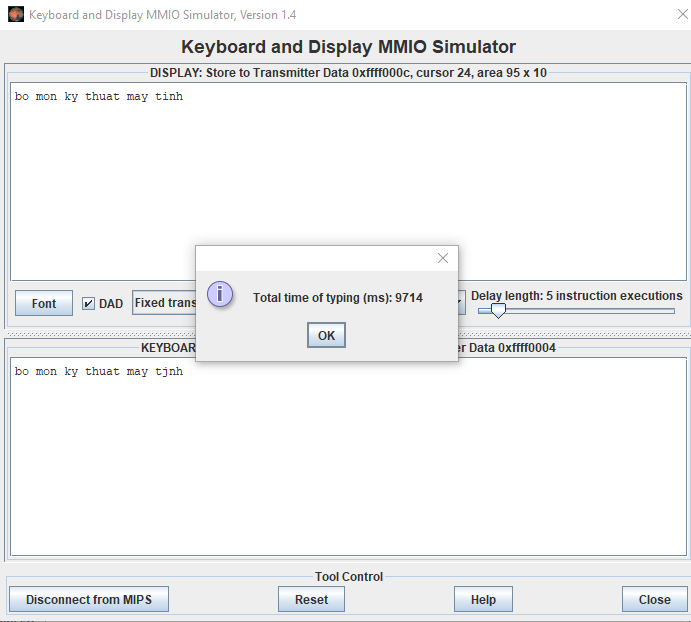
sb $a0, 0($t0)

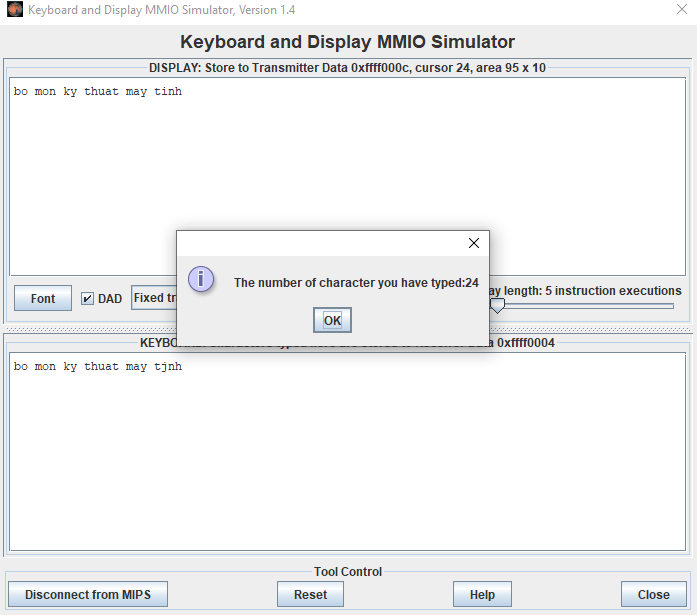
nop

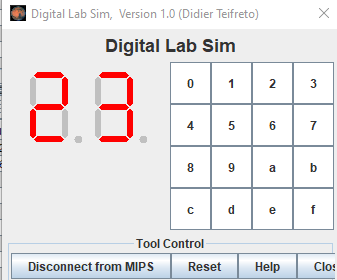
jr $ra

nop

* **Hình ảnh minh họa:**

****

****

****