ĐẠI HỌC BÁCH KHOA HÀ NỘI

TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

****

BÁO CÁO

FINAL PROJECT 03

Học phần: Thực hành kiến trúc máy tính

Mã học phần: IT3280

Giảng viên hướng dẫn: TS. Hoàng Văn Hiệp

Sinh viên thực hiện: Nguyễn Khánh Nam

Mã số sinh viên: 20225749

Mã lớp: 147798

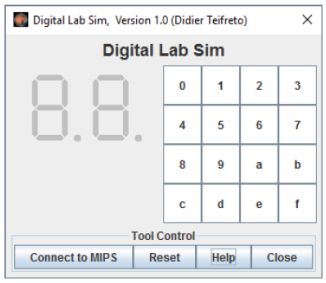
Hà Nội, tháng 6 năm 2024

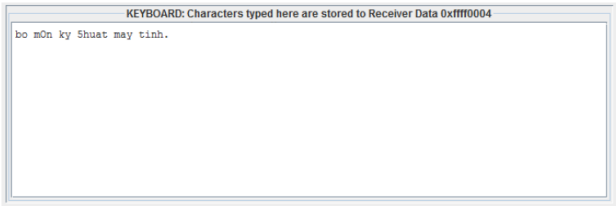
I. Đề bài

3. Kiểm tra tốc độ và độ chính xác khi gõ văn bản

Chương trình sau 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"*
- Sử dụng bộ định thời Timer (trong bộ giả lập Digi Lab Sim) để tạo ra khoảng thời gian để đo. Đây là thời gian giữa 2 lần ngắt, chu kì ngắt.
- 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.





II. Giải quyết bài toán

- Sử dụng bộ đếm thời gian Timer trong Tool Digital Lab Sim và Tool Keyboard and Display MMIO Simulator.
- Thiết lập địa chỉ kiểm tra input KEY CODE và trạng thái KEY READY của Tool Keyboard and Display MMIO Simulator và địa chỉ kích hoạt bộ đếm thời gian COUNTER trong Tool Digital Lab Sim cùng địa chỉ hiển thị LED 7 thanh:

```
.eqv LEFT_SEGS 0xFFFF0011
.eqv RIGHT_SEGS 0xFFFF0010

.eqv COUNTER 0xFFFF0013
.eqv MASK_CAUSE_COUNTER 0x00000400

.eqv KEY_CODE 0xFFFF0004
.eqv KEY_READY 0xFFFF0000
```

- Thiết lập mảng segsdisplay_number_arr chứa mã HEX của các số 0-9 để hiển thị ra LED 7 thanh.
- String chứa chuỗi kiểm tra. Và thiết lập sẵn các chuỗi in kết quả.

```
.data
segsdisplay_number_arr: .byte 63, 6, 91, 79, 102, 109, 125, 7, 127, 111 #mảng
chứa số 0-9 để hiển thị ra 7segs LED (mỗi số 1 byte)

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

msg1: .asciiz "Toc do go phim: "
msg2: .asciiz " words/minute\n"
msg3: .asciiz "Tong thoi gian chay chuong trinh: "
msg4: .asciiz " (s)\n"
```

- Thanh ghi $k0, $k1 chứa địa KEY_CODE và KEY_READY.
- Kích hoạt bộ đếm Timer để tạo khoảng thời gian đo tốc độ gõ.
- Thanh ghi $a1 chứa địa chỉ string test, $a2 chứa địa chỉ mảng số hiển thị LED 7 thanh.
- Thanh ghi $t8 và $t9 chứa địa chỉ LED 7 thanh trái và phải.
- Các thanh ghi:
    - $s1: đếm số ký tự đúng
    - $s2: tổng số ký tự được nhập
    - $s3: tổng thời gian chạy chương trình (s)
    - $s4: tổng số lần ngắt do bộ đếm Timer
    - $s5: ký tự đứng trước ký tự được kiểm tra đúng
    - $s6: đếm số từ nhập vào

```
.text

li $k0, KEY_CODE
li $k1, KEY_READY

# Kích hoạt ngắt COUNTER
li $t0, COUNTER
sb $t0, 0($t0)  #Kích hoạt

la $a1, string  #Lưu add của string test case
la $a2, segsdisplay_number_arr # Lưu add của mảng chứa số 7-segs Display

li $t8, LEFT_SEGS
li $t9, RIGHT_SEGS
```

```
li $s1, 0   #Số ký tự đúng
li $s2, 0   #Tổng số ký tự nhập vào
li $s3, 0   #Tổng thời gian chạy chương trình
li $s4, 0   #Tổng số lần ngắt Counter
li $s5, 0   #Chứa ký tự đứng trước ký tự đang được check
li $s6, 0   #Số từ
```

- Vòng lặp vô hạn đợi kiểm tra ký tự được nhập vào và nhận ngắt cho bộ đếm Timer:

```
Wait4keyboard:
    lb $v1, 0($k1)      #UPDATE: sử dụng thanh ghi khác để check KEY READY
    nop
    li $v0, 32  #Sleep
    li $a0, 5            # Ngủ CPU đợi Counter 5ms
    syscall
    nop
    bne $v1, 0, keyboard_intr
    nop
    b Wait4keyboard
    nop
    beq $v1, 0, Wait4keyboard
    nop
keyboard_intr: jal check_string
        b Wait4keyboard
        nop
```

- Chương trình con kiểm tra ký tự nhập vào:
    - Ngắt bộ đếm Timer.
    - Kiểm tra ký tự nhập vào đồng thời với ký tự trong string địa chỉ $a1 và tăng biến đếm ký tự đúng $s1.
    - Địa chỉ $a1 được tăng 1 byte sau mỗi lần kiểm tra.
    - Đếm số từ nhập vào $s6.
    - Đồng thời chạy chương trình con hiển thị LED 7 thanh số ký tự đúng nhập vào show_7segsLed.

```
check_string:
# Dừng ngắt COUNTER để không ảnh hưởng khi chạy ctrinh ngắt
li $t0, COUNTER
sb $0, 0($t0)   #STOP
    lb $t3, 0($a1)      #Get char string test case
    beq $t3, 0, end_prog    #Check ở string test case tới NULL thì dừng
```

```
    nop
    lb $t4, 0($k0)      #Get Key code

    bne $t3, $t4, continue1 #Kiểm tra ký tự đúng hay không
    nop
    addi $s1, $s1, 1        # Số ký tự đúng ++
continue1:
#---------------------------------------------------------------
# Kiểm tra để đếm số TỪ nhập vào:
# Nếu nhập vào space cần kiểm tra 2 TH
# TH1: trước space vừa nhập là 1 char -> Tính +1 từ
# TH2: trước space vừa nhập cũng là 1 space -> Không tính từ -> về main
# Nếu KHÔNG nhập vào space thì tiếp tục đọc ký tự nhập vào tiếp (về main)
#---------------------------------------------------------------
    bne $t4, ' ', continue2
    nop
    beq $s5, ' ', continue2
    nop
    addi $s6 , $s6, 1       #Tăng số từ +1

continue2:
    addi $s5, $t4, 0        # Save lại char vừa check
    addi $a1, $a1, 1        # Tăng con trỏ string test case thêm 1 byte

    addi $sp, $sp, -4   #Lưu lại địa chỉ return của $ra trước đó
    sw $ra, 0($sp)

    jal show_7segsLed
    nop

    lw $ra, 0($sp)      # Lấy lại địa chỉ return của $ra trước đó
    addi $sp, $sp, 4
# Kích hoạt ngắt COUNTER

li $t0, COUNTER
sb $t0, 0($t0)  #Kích hoạt

    jr $ra              #RETURNS
    nop
```

- Chương trình con hiển thị LED 7 thanh:

```
#----------------------SHOW LEDS-------------
show_7segsLed:

    addi $t5, $s1, 0
    addi $t6, $0, 10
    div $t5, $t6

    mflo $t5        # Lấy số hàng chục
    mfhi $t6        # Lấy số hàng đơn vị

#SHOW LEFT LED
    add $t5, $a2, $t5   # $t5 chứa address của số hàng chục trong mảng 7-segs
display
    lb $t5, 0($t5)
    sb $t5, 0($t8)      # Show số hàng chục ra LED

#SHOW RIGHT LED
    add $t6, $a2, $t6   # $t6 chứa address của số hàng đơn vị trong mảng
    lb $t6, 0($t6)
    sb $t6, 0($t9)      # Show số hàng đơn vị ra LED

    jr $ra          #Return về ngắt
    nop
#---------------------------------------------------------------
```

- Chương trình ngắt tại địa chỉ .ktext 0x80000180:
    - Ngắt hoạt động bộ đếm Timer khi bắt đầu chương trình ngắt và kích hoạt lại trước khi kết thúc nhảy vào chương trình chính.
    - Tăng biến đếm thời gian $s3 sau mỗi lần đủ độ trễ bộ đếm Timer 40 lần ngắt.

```
.ktext 0x80000180
# Dừng ngắt COUNTER để không ảnh hưởng khi chạy ctrinh ngắt
li $t0, COUNTER
sb $0, 0($t0)   #STOP

counterIntr:
    blt $s4, 40, continue3  # Chưa đủ số lần ngắt Counter thì chưa đếm giây
    nop

    addi $s4, $0, 0     # reset số lần ngắt
```

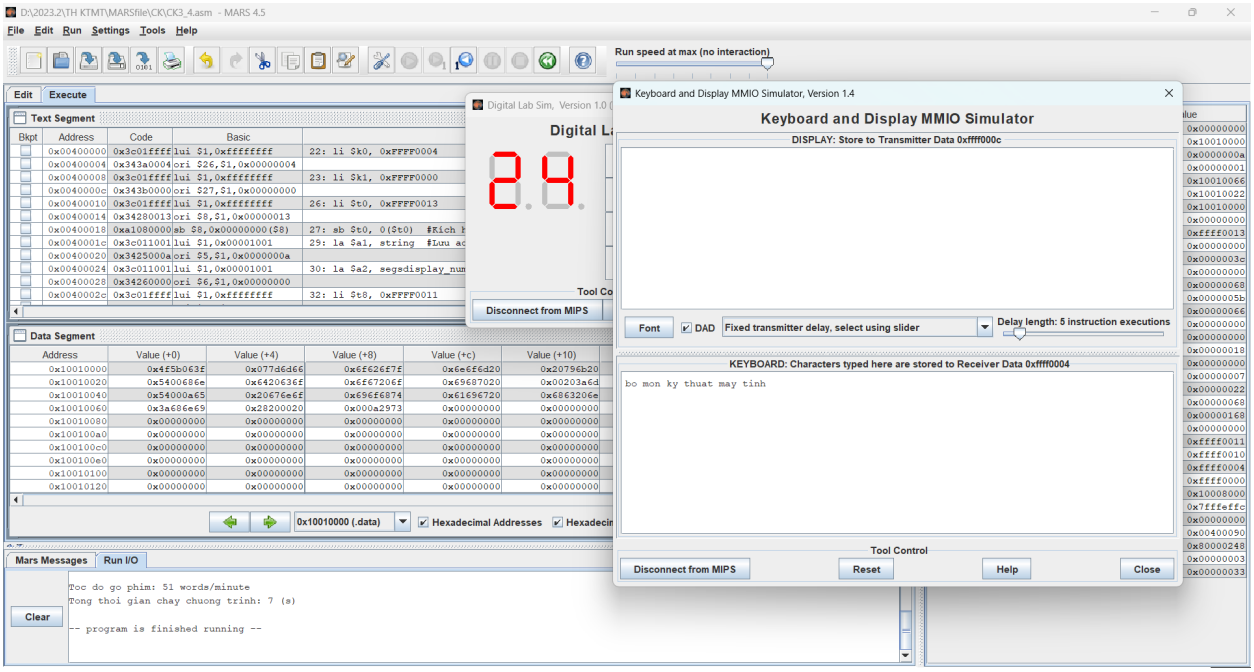
```
    addi $s3, $s3, 1        # Tăng thời gian chạy
    j end_interrupt
    nop

    continue3:
    addi $s4, $s4, 1        # Tăng số lần ngắt

    j end_interrupt
    nop


end_interrupt:  #Kết thúc chương trình ngắt quay về main cần tăng epc cho đúng về
ctrinh main
# VÀ kích hoạt trở lại ngắt COUNTER

# Kích hoạt lại ngắt COUNTER
li $t0, COUNTER
sb $t0, 0($t0)  #Kích hoạt

next_pc:
    mfc0 $at, $14
    addi $at, $at, 4
    mtc0 $at, $14
return_to_main: eret
```

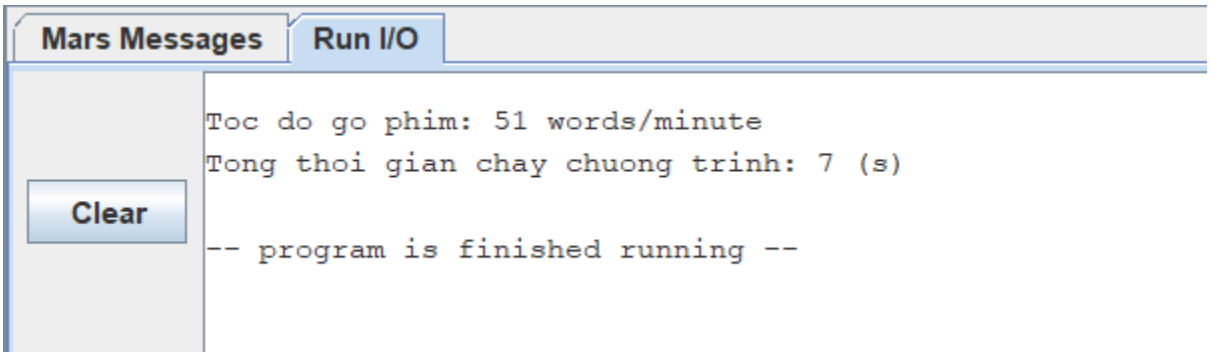
- Kết thúc chương trình:
    - Tính tốc độ gõ phím theo đơn vị wpm (từ/phút) và hiển thị kết quả

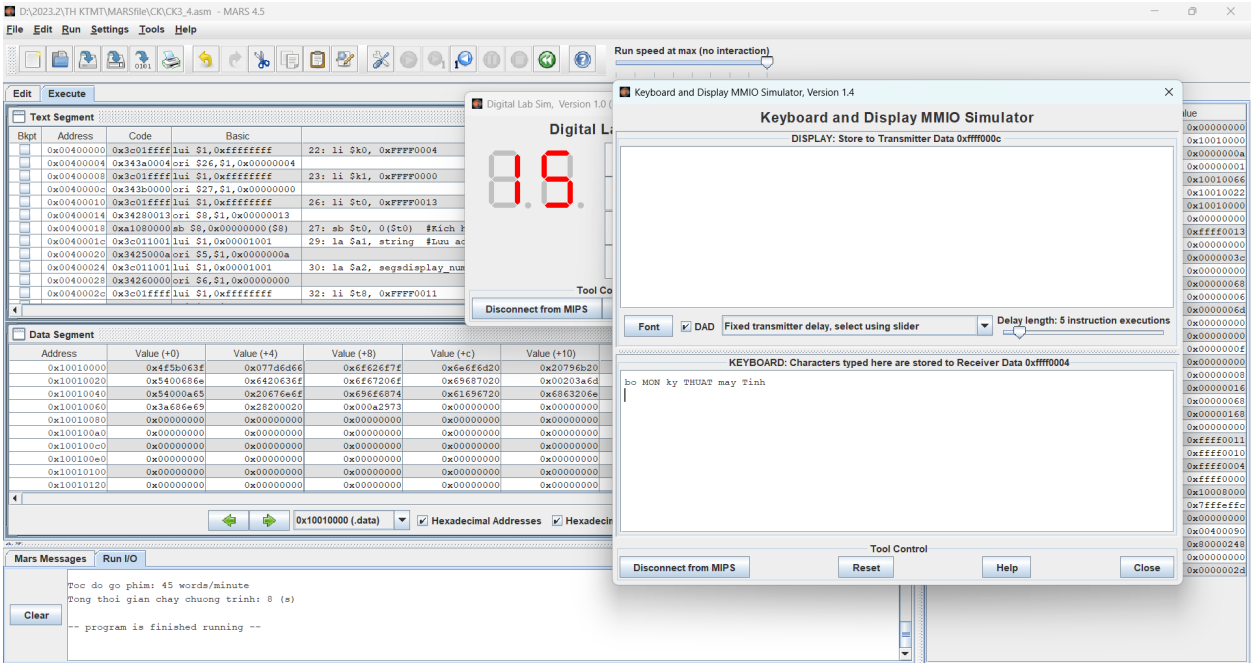
```
#------------------------------------------------------------------------------
end_prog:


    li $v0, 4
    la $a0, msg1
    syscall

    li $v0, 1
    addi $s6, $s6, 1
    addi $t2, $0, 60
    mult $s6, $t2       #Số từ x 60
    mflo $s6
    div $s6, $s3        # (số từ x 60)/tổng số thời gian (s)
```

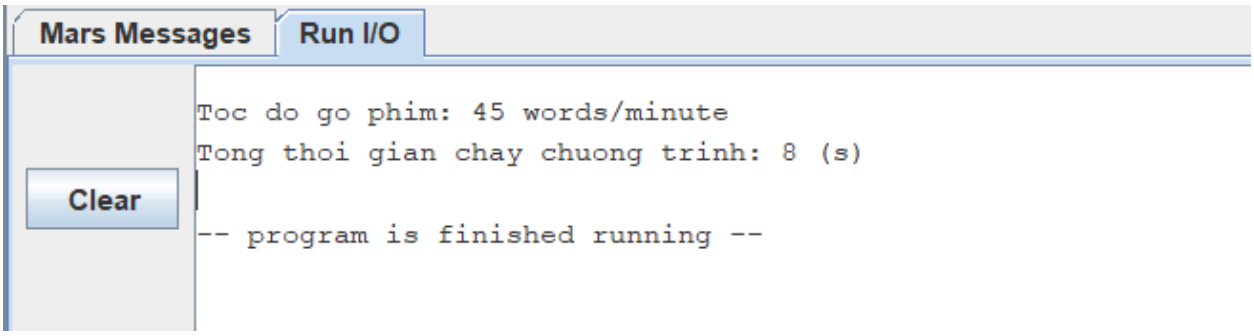
```
    mflo $a0            # wpm
    syscall

    li $v0, 4
    la $a0, msg2
    syscall

    li $v0, 4
    la $a0, msg3
    syscall

    li $v0, 1
    addi $a0, $s3, 0
    syscall

    li $v0, 4
    la $a0, msg4
    syscall

    li $v0, 10
    syscall
end:
#-------------------------------------------------------------------------------
```

III. Kết quả


Keyboard and Display MMIO Simulator
DISPLAY: Store to Transmitter Data 0xffff000c
KEYBOARD: Characters typed here are stored to Receiver Data 0xffff0004
bo mon ky thuat may tinh
Tool Control
Disconnect from MIPS
Reset
Help
Close
Mars Messages
Run I/O
Toc do go phim: 51 words/minute
Tong thoi gian chay chuong trinh: 7 (s)
-- program is finished running --
Clear


Mars Messages
Run I/O
Toc do go phim: 51 words/minute
Tong thoi gian chay chuong trinh: 7 (s)
-- program is finished running --
Clear


Keyboard and Display MMIO Simulator
DISPLAY: Store to Transmitter Data 0xffff000c
KEYBOARD: Characters typed here are stored to Receiver Data 0xffff0004
bo MON ky THUAT may Tinh
Tool Control
Disconnect from MIPS
Reset
Help
Close
Mars Messages
Run I/O
Toc do go phim: 45 words/minute
Tong thoi gian chay chuong trinh: 8 (s)
-- program is finished running --
Clear


Mars Messages
Run I/O
Toc do go phim: 45 words/minute
Tong thoi gian chay chuong trinh: 8 (s)
-- program is finished running --
Clear

IV. Code

```
.eqv LEFT_SEGS 0xFFFF0011
.eqv RIGHT_SEGS 0xFFFF0010

.eqv COUNTER 0xFFFF0013
.eqv MASK_CAUSE_COUNTER 0x00000400

.eqv KEY_CODE 0xFFFF0004
.eqv KEY_READY 0xFFFF0000

.data
segsdisplay_number_arr: .byte 63, 6, 91, 79, 102, 109, 125, 7, 127, 111 #mảng
chứa số 0-9 để hiển thị ra 7segs LED (mỗi số 1 byte)

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

msg1: .asciiz "Toc do go phim: "
msg2: .asciiz " words/minute\n"
msg3: .asciiz "Tong thoi gian chay chuong trinh: "
msg4: .asciiz " (s)\n"

.text

li $k0, KEY_CODE
li $k1, KEY_READY

# Kích hoạt ngắt COUNTER
li $t0, COUNTER
sb $t0, 0($t0)  #Kích hoạt

la $a1, string  #Lưu add của string test case
la $a2, segsdisplay_number_arr # Lưu add của mảng chứa số 7-segs Display

li $t8, LEFT_SEGS
li $t9, RIGHT_SEGS


li $s1, 0   #Số ký tự đúng
li $s2, 0   #Tổng số ký tự nhập vào
li $s3, 0   #Tổng thời gian chạy chương trình
li $s4, 0   #Tổng số lần ngắt Counter
li $s5, 0   #Chứa ký tự đứng trước ký tự đang được check
li $s6, 0   #Số từ
```

```
Wait4keyboard:
    lb $v1, 0($k1)      #UPDATE: sử dụng thanh ghi khác để check KEY READY
    nop
    li $v0, 32  #Sleep
    li $a0, 5
    syscall
    nop
    bne $v1, 0, keyboard_intr
    nop
    b Wait4keyboard
    nop
    beq $v1, 0, Wait4keyboard
    nop
keyboard_intr: jal check_string
        b Wait4keyboard
        nop

check_string:
# Dừng ngắt COUNTER để không ảnh hưởng khi chạy ctrinh ngắt
li $t0, COUNTER
sb $0, 0($t0)   #STOP
    lb $t3, 0($a1)      #Get char string test case
    beq $t3, 0, end_prog    #Check ở string test case tới NULL thì dừng
    nop
    lb $t4, 0($k0)      #Get Key code

    bne $t3, $t4, continue1 #Kiểm tra ký tự đúng hay không
    nop
    addi $s1, $s1, 1        # Số ký tự đúng ++
continue1:
#---------------------------------------------------------------
# Kiểm tra để đếm số TỪ nhập vào:
# Nếu nhập vào space cần kiểm tra 2 TH
# TH1: trước space vừa nhập là 1 char -> Tính +1 từ
# TH2: trước space vừa nhập cũng là 1 space -> Không tính từ -> về main
# Nếu KHÔNG nhập vào space thì tiếp tục đọc ký tự nhập vào tiếp (về main)
#---------------------------------------------------------------
    bne $t4, ' ', continue2
    nop
    beq $s5, ' ', continue2
    nop
    addi $s6 , $s6, 1       #Tăng số từ +1

continue2:
    addi $s5, $t4, 0        # Save lại char vừa check
```

```
    addi $a1, $a1, 1        # Tăng con trỏ string test case thêm 1 byte

    addi $sp, $sp, -4   #Lưu lại địa chỉ return của $ra trước đó
    sw $ra, 0($sp)

    jal show_7segsLed
    nop

    lw $ra, 0($sp)      # Lấy lại địa chỉ return của $ra trước đó
    addi $sp, $sp, 4
# Kích hoạt ngắt COUNTER

li $t0, COUNTER
sb $t0, 0($t0)  #Kích hoạt

    jr $ra              #RETURNS
    nop
#----------------------SHOW LEDS-------------
show_7segsLed:

    addi $t5, $s1, 0
    addi $t6, $0, 10
    div $t5, $t6

    mflo $t5        # Lấy số hàng chục
    mfhi $t6        # Lấy số hàng đơn vị

#SHOW LEFT LED
    add $t5, $a2, $t5   # $t5 chứa address của số hàng chục trong mảng 7-segs
display
    lb $t5, 0($t5)
    sb $t5, 0($t8)      # Show số hàng chục ra LED

#SHOW RIGHT LED
    add $t6, $a2, $t6   # $t6 chứa address của số hàng đơn vị trong mảng
    lb $t6, 0($t6)
    sb $t6, 0($t9)      # Show số hàng đơn vị ra LED

    jr $ra          #Return về ngắt
    nop
#---------------------------------------------------------------

.ktext 0x80000180
# Dừng ngắt COUNTER để không ảnh hưởng khi chạy ctrinh ngắt
li $t0, COUNTER
```

```
sb $0, 0($t0)   #STOP

counterIntr:
    blt $s4, 40, continue3  # Chưa đủ số lần ngắt Counter thì chưa đếm giây
    nop

    addi $s4, $0, 0     # reset số lần ngắt
    addi $s3, $s3, 1        # Tăng thời gian chạy
    j end_interrupt
    nop

    continue3:
    addi $s4, $s4, 1        # Tăng số lần ngắt

    j end_interrupt
    nop


end_interrupt:  #Kết thúc chương trình ngắt quay về main cần tăng epc cho đúng về
ctrinh main
# VÀ kích hoạt trở lại ngắt COUNTER

# Kích hoạt lại ngắt COUNTER
li $t0, COUNTER
sb $t0, 0($t0)  #Kích hoạt

    mtc0 $0, $13        #RESET cause reg

next_pc:
    mfc0 $at, $14
    addi $at, $at, 4
    mtc0 $at, $14
return_to_main: eret

#-----------------------------------------------------------


#-----------------------------------------------------------------------------
------
end_prog:


    li $v0, 4
    la $a0, msg1
    syscall
```

```
    li $v0, 1
    addi $s6, $s6, 1
    addi $t2, $0, 60
    mult $s6, $t2       #Số từ x 60
    mflo $s6
    div $s6, $s3        # (số từ x 60)/tổng số thời gian (s)
    mflo $a0            # wpm
    syscall

    li $v0, 4
    la $a0, msg2
    syscall

    li $v0, 4
    la $a0, msg3
    syscall

    li $v0, 1
    addi $a0, $s3, 0
    syscall

    li $v0, 4
    la $a0, msg4
    syscall

    li $v0, 10
    syscall
end:
#-------------------------------------------------------------------------------
---------------
```