TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

-----OO📖OO-----

BÁO CÁO THỰC HÀNH
KIẾN TRÚC MÁY TÍNH

Giáo viên hướng dẫn: Ths. Lê Bá Vui

Sinh Viên: Lê Thị Khánh Hoà -20194565

Đỗ Quốc Huy - 20205084

Hà Nội, Ngày 21 Tháng 07 Năm 2022

MỤC LỤC

Nội Dung **Trang**

Bài 8: Mô phỏng ổ đĩa RAID 5

Sinh Viên Thực Hiện: Lê Thị Khánh Hòa – 20194565

I. Đề bài: Hệ thống ổ đĩa RAID5 cần tối thiểu 3 ổ đĩa cứng, trong đó phần dữ liệu parity sẽ được chứa lần lượt lên 3 ổ đĩa như trong hình bên. Hãy viết chương trình mô phỏng hoạt động của RAID 5 với 3 ổ đĩa, với giả định rằng, mỗi block dữ liệu có 4 kí tự. Giao diện như trong minh họa dưới. *Giới hạn chuỗi kí tự nhập vào có độ dài là bội của 8*.

Trong ví dụ sau, chuỗi kí tự nhập vào từ bàn phím *(DCE.****ABCD1234HUSTHUST)* sẽ được chia thành các block 4 byte. Block 4 byte đầu tiên "DCE." sẽ được lưu trên Disk 1, Block 4 byte tiếp theo "****" sẽ lưu trên Disk 2, dữ liệu trên Disk 3 sẽ là 4 byte parity được tính từ 2 block đầu tiên với mã ASCII là 6e='D' xor '*' ; 69='C' xor '*'; 6f='E' xor '*'; 04='.' xor '*'

```
Nhap chuoi ki tu : DCE.****ABCD1234HUSTHUST
     Disk 1                Disk 2                Disk 3
 --------------        --------------        --------------
|      DCE.      |      |      ****      |      [[ 6e,69,6f,04]]
|      ABCD      |      [[ 70,70,70,70]]      |      1234      |
[[ 00,00,00,00]]      |      HUST      |      |      HUST      |
 --------------        --------------        --------------
```

II. Cách Làm:

Các bước làm trong bài bao gồm:
- Cho người dùng nhập vào từ bàn phím chuỗi block
- Xử lí đếm độ dài chuỗi nhập vào
- Kiểm tra độ dài chuỗi nhập vào chia hết cho 8 hay không (bằng cách kiểm tra byte 4 bit cuối có bằng 0000 hoặc 1000 hay không)
- Lưu dữ liệu Raid5: Đọc mỗi lần 8 kí tự, 4 kí tự đầu lưu trong block1, 4 kí tự sau lưu trong block2, block3 lưu kết quả block1 xor block2.
- In ra màn hình: Kiểm tra parity cần được lưu ở disk nào, xử lí việc in ra kết quả của parity theo hệ 16 và thứ tự in ra của dữ liệu.

- Trình bày các disk theo mẫu.

III. Mã Nguồn:

```
.data
        title1: .asciiz  "Nhap chuoi ki tu : "
        title2: .asciiz      "     Disk 1              Disk 2              Disk 3\n"
        title3: .asciiz      " --------------        --------------        --------------\n"
        title4: .asciiz  "|     "
        title5: .asciiz  "     |"
        title6: .asciiz  "[[ "
        title7: .asciiz  "]]"
        title8: .asciiz  "      "
        title9: .asciiz  "\n"
        title10: .asciiz ","
        error_length: .asciiz "Do dai chuoi khong hop le! Nhap lai.\n"

        str: .space 100             # bien luu chuoi nhap tu ban phim
        block1: .space 100          # block 1
        block2: .space 100          # block 2
        block3: .space 100          # block 3 (ket qua phep xor)

# ------------------------------------------------------------------------------------------
# TAC DUNG CUA CAC THANH GHI BIEN

# s0 : dia chi chua ky tu trong chuoi str
# s1 : dia chi block 1
# s2 : dia chi block 2
# s3 : dia chi block 3 (chua su dung)
# s4 : check dieu kien chuoi la boi cua 8
# s5 : ky tu lay ra tu trong chuoi str
```

```
# s6 : bien chi dinh disk save parity

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

.text
input:
        # title nhap chuoi
        li $v0, 4
        la $a0, title1
        syscall

        # doc chuoi tu ban phim
        li $v0, 8
        la $a0, str
        li $a1, 100
        syscall

main:
        la $s0, str         # s0 = address(str) -> $s0 chỉ đến kí tự đầu tiên trong chuỗi nhập
length:
        addi $t3, $zero, 0          # t3 = length
        addi $t0, $zero, 0          # t0 = index -> chỉ số


check_char:
        add $t1, $s0, $t0           # t1 = address of string[i]
        lb $t2, 0($t1)              # t2 = string[i]
```

```
        nop
        beq $t2, 10, test_length  # t2 = '\n' ket thuc xau
        nop
        addi $t3, $t3, 1  # length++
        addi $t0, $t0, 1  # index++
        j check_char
        nop
test_length:
        and $t1, $t3, 0x0000000f                # xoa het cac byte cua $t3 ve 0, chi giu lai byte cuoi
        bne $t1, 0, test1                       # byte cuoi bang 0 hoac 8 thi so chia het cho 8
        j raid5
test1:
        beq $t1, 8, raid5
error1:
        li $v0, 4
        la $a0, error_length
        syscall
        j input
raid5:
        # tieu de output
        li $v0, 4
        la $a0, title2
        syscall

        # ky tu ngan cach
        li $v0, 4
        la $a0, title3
        syscall
```

```
        la $s1, block1    # s1 = address(block1)    -> block1, block2 luu du lieu; block3 luu ket qua xor
        la $s2, block2    # s2 = address(block2)

        li $s6, 0
        # quy tac luu parity: 0 -> disk 3; 1 -> disk 2; 2 -> disk 1

start:
        li $s4, 0         # reset lai sau 8 byte duoc doc tu chuoi str

check_block_full:
        beq $s4, 8, block_3
load_from_str:
        lb $s5, 0($s0)            # lay ky tu tai dia chi s0 -> $s5 = string[0]
        addi $s0, $s0, 1 # tang dia chi s0 len 1
        beq $s5, '\n', exit_main  # neu ket thuc chuoi thi thoat main

        addi $s4, $s4, 1 # so thu tu ky tu vua lay ra tu str
        nop
        slti $t1, $s4, 5    # neu so thu tu nho hon 5 thi luu vao block 1
        beq $t1, 1, block_1
        j block_2


block_1:
        sb $s5, 0($s1)            # luu ky tu vao block 1 [i]
        addi $s1, $s1, 1 # tang len block 1 [i+1]
        j load_from_str           # quay lai doc ky tu tiep theo trong str
```

```
block_2:
        sb $s5, 0($s2)          # luu ky tu vao block 2 [i]
        addi $s2, $s2, 1 # tang len block 2 [i+1]
        j check_block_full      # quay lai check so ky tu da luu

block_3:
        addi $s1, $s1, -4       # quay ve dia chi ky tu dau tien trong block 1
        addi $s2, $s2, -4       # quay ve dia chi ky tu dau tien trong block 2

        add $t8, $s1, $zero     # t8 = s1
        add $t9, $s2, $zero     # t9 = s2

        li $t4, 0           # bien dem so lan xor

save_to_disk:
        beq $s6, 0, disk_3_parity
        beq $s6, 1, disk_2_parity
        beq $s6, 2, disk_1_parity

disk_1_parity:
        nop
        # ki tu mo block 3
        li $v0, 4
        la $a0, title6
        syscall

        nop
        jal save_block_3        # save block 3 vao disk 1
```

```
        nop

        # ki tu dong block 3
        li $v0, 4
        la $a0, title7
        syscall

        # ki tu cach giua cac disk
        li $v0, 4
        la $a0, title8
        syscall

        nop
        jal save_block_1        # save block 1 vao disk 2

        # ki tu cach giua cac disk
        li $v0, 4
        la $a0, title8
        syscall

        nop
        jal save_block_2        # save block 2 vao disk 3
        nop

        j refresh_disk_parity

disk_2_parity:
        nop
```

```
jal save_block_1          # save block 1 vao disk 1
nop

# ki tu cach giua cac disk
li $v0, 4
la $a0, title8
syscall

nop
# ki tu mo block 3
li $v0, 4
la $a0, title6
syscall

nop
jal save_block_3          # save block 3 vao disk 2
nop

# ki tu dong block 3
li $v0, 4
la $a0, title7
syscall

# ki tu cach giua cac disk
li $v0, 4
la $a0, title8
syscall
```

```
        nop
        jal save_block_2        # save block 2 vao disk 3
        nop

        j refresh_disk_parity

disk_3_parity:
        nop
        jal save_block_1        # save block 1 vao disk 1
        nop

        # ki tu cach giua cac disk
        li $v0, 4
        la $a0, title8
        syscall

        nop
        jal save_block_2        # save block 2 vao disk 2
        nop

        # ki tu cach giua cac disk
        li $v0, 4
        la $a0, title8
        syscall

        nop
        # ki tu mo block 3
        li $v0, 4
```

```
        la $a0, title6
        syscall

        nop
        jal save_block_3        # # save block 3 vao disk 3
        nop

        # ki tu dong block 3
        li $v0, 4
        la $a0, title7
        syscall


        j refresh_disk_parity


save_block_1:
        # ki tu mo
        li $v0, 4
        la $a0, title4
        syscall

        # noi dung disk
        li $v0, 4
        la $a0, block1
        syscall

        # ki tu dong
```

```
        li $v0, 4
        la $a0, title5
        syscall

        nop
        jr $ra

save_block_2:
        # ki tu mo
        li $v0, 4
        la $a0, title4
        syscall

        # noi dung disk
        li $v0, 4
        la $a0, block2
        syscall

        # ki tu dong
        li $v0, 4
        la $a0, title5
        syscall

        nop
        jr $ra

save_block_3:
        lb $t1, 0($t8)          # lay ky tu block 1 [i]
```

```
        lb $t2, 0($t9)          # lay ky tu block 2 [i]
        xor $t3, $t1, $t2 # xor 2 ky tu
        addi $t4, $t4, 1  # so lan xor tang len 1

        # chuyen ve he co so 16
        div $a0, $t3, 16 # lay thuong khi chia cho 16
        li $t6, 0         # thong bao dang lay thuong
        j check_quotient_remainder

save_block_3_back_1:
        mfhi $a0                # lay du khi chia cho 16
        li $t6, 1         # thong bao dang lay du
        j check_quotient_remainder

save_block_3_back_2:
        addi $t8, $t8, 1  # tang len block 1 [i+1]
        addi $t9, $t9, 1  # tang len block 2 [i+1]
        j save_block_3          # tiep tuc xor 2 ki tu tiep theo trong block 1 va 2

# kiem tra thuong hoac du co < 10 khong
check_quotient_remainder:
        slti $t5, $a0, 10
        beq $t5, 1, print_int

# neu >= 10 thi chuyen sang ky tu A,B,C,D,E,F
convert_char_16:
        beq $a0, 10, print_A
        beq $a0, 11, print_B
```

```
        beq $a0, 12, print_C
        beq $a0, 13, print_D
        beq $a0, 14, print_E
        beq $a0, 15, print_F



print_A:
        li $a0, 'A'
        j print_char

print_B:
        li $a0, 'B'
        j print_char

print_C:
        li $a0, 'C'
        j print_char

print_D:
        li $a0, 'D'
        j print_char

print_E:
        li $a0, 'E'
        j print_char

print_F:
        li $a0, 'F'
```

```
        j print_char

print_char:
        li $v0, 11
        syscall
        nop
        j exit_or_back

print_int:
        li $v0, 1
        syscall
        nop
        j exit_or_back

exit_or_back:
        beq $t6, 0, save_block_3_back_1        # da lay thuong
        beq $t6, 1, print_comma        # da lay du

print_comma:
        beq $t4, 4, complete_block_3    # neu t4 = 4 thi khong in dau phay va hoan thanh block 3
        # in dau phay neu t4 < 4
        li $v0, 4
        la $a0, title10
        syscall
        j save_block_3_back_2

# hoan thanh block 3
complete_block_3:
```

```
        jr $ra


refresh_disk_parity:
        # xuong dong
        li $v0, 4
        la $a0, title9
        syscall
        #chuyen doi disk parity
        addi $s6, $s6, 1 # tang them 1 vao bien de xac dinh disk save parity tiep theo
        div $s6, $s6, 3          # chia 3 lay du
        mfhi $s6                 # lay du
        j start



exit_main:
        # ky tu ngan cach
        li $v0, 4
        la $a0, title3
        syscall

        # ket thuc chuong trinh
        li $v0, 10
        syscall
```

IV. Kết Quả Chạy Mô Phỏng Chương Trình:

- Xâu nhập vào là: *lethikhanhhoa123*

```
Nhap chuoi ki tu : lethikhanhhoa123
     Disk 1                Disk 2                Disk 3
 --------------        --------------        --------------
|     leth     |      |     ikha     |      [[ 05,0E,1C,09]]
|     nhho     |      [[ 0F,59,5A,5C]]      |     a123     |
 --------------        --------------        --------------
```

- Xâu nhập vào là: *abcxyz123*

```
Nhap chuoi ky tu : abcxyz123
      Disk 1                 Disk 2                Disk 3
----------------       -----------------      ----------------
Do dai chuoi khong hop le! Nhap lai.
Nhap chuoi ky tu :
```

- Xâu nhập vào là: *abcxyz123456!@#$*

```
----------------       -----------------      ----------------
Do dai chuoi khong hop le! Nhap lai.
Nhap chuoi ky tu : abcxyz123456!@#$
      Disk 1                 Disk 2                Disk 3
----------------       -----------------      ----------------
|     abcx     |       |     yz12     |       [[ 18,18,52,4a]]
|     3456     |       [[ 12,74,16,12]]       |     !@#$     |
----------------       -----------------      ----------------
```

Bài 3: KIỂM TRA TỐC ĐỘ VÀ ĐỘ CHÍNH XÁC KHI GÕ VĂN BẢN

Sinh Viên Thực Hiện: Đỗ Quốc Huy – 20205084

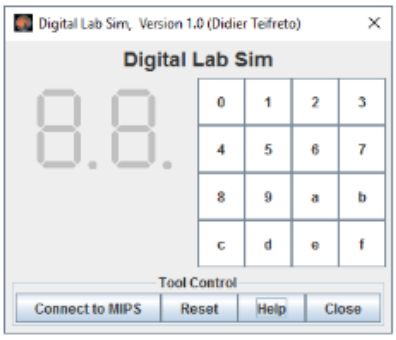
I. Đề bài: Thực hiện chương trình đo tốc độ gõ bàn phím và hiển thị kết quả bằng 2 đèn led 7 đoạn. Nguyên tắc:

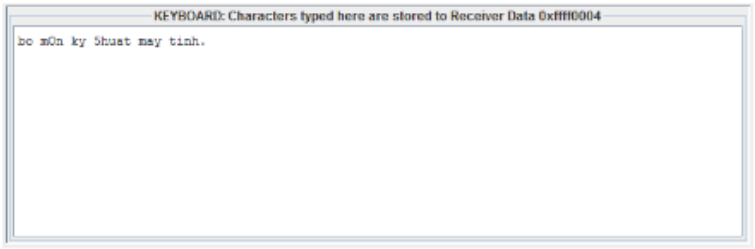
- Cho một đoạn văn bản mẫu, cố định sẵn trong mã nguồn. Ví dụ *"bo mon ky thuat may tinh"*

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

- Người dùng nhập các kí tự từ bàn phím. Ví dụ nhập *"bo mOn ky 5huat 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.

- Chương trình đồng thời cần tính được tốc độ gõ: thời gian hoàn thành và số từ trên một đơn vị thời gian.





II. Cách Làm:

1. Ý tưởng:

Sử dụng số vòng lặp để tính thời gian (cứ 100 vòng lặp là 1 giây, 1 vòng lặp = 10ms). Số ký tự nhập vào mỗi giây là số ký tự nhập vào mỗi 100 vòng lặp. Sau khi hoàn thành nhập (người dùng nhấn ENTER) thì *thời gian nhập = số vòng lặp đếm được x10(ms)*. Xâu nhập vào được so sánh từng ký tự với xâu nguồn (được lưu trữ sẵn trong mã nguồn) để tìm ra số ký tự nhập đúng.

2. Cách làm:

- Thanh ghi $s3 đếm số vòng lặp đã thực hiện.
- Trong mỗi vòng lặp, nếu KEY_READY = 0 thì thực hiện sleep 10ms rồi chuyển sang vòng lặp kế tiếp; nếu KEY_READY = 1 thì tăng số ký tự được nhập trong 1 giây thêm 1 và thực hiện chương trình con phục vụ ngắt.
- Tại đây, kiểm tra nguyên nhân gây ra ngắt có phải từ bàn phím không. Nếu không phải, kết thúc chương trình con và thực hiện lệnh tiếp theo của chương trình chính. Nếu nguyên nhân gây ra ngắt là từ bàn phím, hiển thị ký tự đó lên màn hình MMIO, lưu trữ vào địa chỉ dùng để lưu trữ xâu nhập vào từ bàn phím và tăng ký tự nhập vào từ bàn phím thêm 1, sau đó kết thúc chương trình con và thực hiện lệnh tiếp theo của chương trình chính.
- Kiểm tra số vòng lặp có chia hết cho 100 không (được 1 giây hay chưa). Nếu không chia hết cho 100, tiếp tục thực hiện sleep rồi chuyển sang vòng lặp tiếp theo. Nếu chia hết cho 100, in ra số ký tự nhập được trong 1 giây rồi thực hiện sleep và chuyển sang vòng lặp tiếp theo.
- Vòng lặp tiếp tục cho đến khi người dùng nhập vào ký tự "\n" (nhấn phím ENTER, tức kết thúc xâu nhập vào).
- Khởi tạo giá trị số ký tự đã được kiểm tra và số ký tự nhập đúng = 0.
- So sánh độ dài 2 xâu, lưu độ dài xâu nhỏ hơn vào thanh ghi $t8.
- Kiểm tra lần lượt từng ký tự ở vị trí tương ứng của 2 xâu. Mỗi lần kiểm tra, số ký tự đã được kiểm tra cộng thêm 1, nếu ký tự ở cùng vị trí của 2 xâu giống nhau thì số ký tự nhập đúng cộng thêm 1, khác nhau thì giữ nguyên.
- Lặp đi lặp lại việc kiểm tra này đến khi duyệt hết các ký tự của xâu có độ dài ngắn hơn.
- In ra số ký tự nhập đúng, thời gian hoàn thành nhập và hiển thị trên Digital Lab Sim.
- Sau khi kết thúc chương trình, hiển thị thông báo xác nhận người dùng có muốn trở lại chương trình hay không. Nếu người dùng nhấn YES, chạy chương trình 1 lần nữa. Trong các trường hợp còn lại, thoát chương trình.

III. Mã Nguồn:

```
.eqv SEVENSEG_LEFT    0xFFFF0011 # Dia chi cua den led 7 doan trai
                                 #Bit 0 = doan a
                                 #Bit 1 = doan b
                                 #Bit 7 = dau .
.eqv SEVENSEG_RIGHT   0xFFFF0010 # Dia chi cua den led 7 doan phai

.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

.data
LED7SEG     : .byte 0x3F,0x06,0x5B,0x4F,0x66,0x6D,0x7D,0x07,0x7F,0x6F         #biểu diễn 0,1,2,3,4,5,6,7,8,9
string : .space 1000                    #khoang trong de luu cac ky tu nhap tu ban phim.
stringsource : .asciiz "bo mon ky thuat may tinh"
keyinls: .asciiz "\n So ky tu nhap trong ls :  "
numright: .asciiz  "\n So ky tu nhap dung la: "
time:   .asciiz "\n Thoi gian nhap la: "
ms:     .asciiz "ms\n"
notification: .asciiz "\n Chuong trinh da ket thuc. Ban muon tro lai chuong trinh?"

.text
        li   $k0,  KEY_CODE
        li   $k1,  KEY_READY
        li   $s0, DISPLAY_CODE
        li   $s1, DISPLAY_READY
main:
        li $s2,0                            #dung de dem toan bo so ky tu nhap vao
        li $s3,0                            #dung de dem so vong lap
        li $t5,100                          #so vong lap trong 1 giay
        li $t6,0                            #bien dem so ky tu nhap duoc trong 1s
        li $v1,0                            # =1 khi chuong trinh ket thuc
LOOP:
```

```
LOOP:
WaitForKey:
        lw   $t1, 0($k1)
        beq  $t1, $zero, next             # if $t1 == 0 (khong co ky tu duoc nhap), chuyen sang vong lap ke tiep
CountIn1s:
        addi $t6,$t6,1                  #so ky tu nhap duoc trong 1s +1
        teqi $t1, 1                       # if $t1 = 1 then raise an Interrupt
next:
        addi    $s3, $s3, 1             #so vong lap +1
        div     $s3,$t5                 #lay so vong lap chia cho 100 de xac dinh da duoc 1s hay chua
        mfhi    $t7                     #luu phan du cua phep chia tren
        bne     $t7,0,sleep             #neu chua duoc 1s (so vong lap != x00), nhay den sleep
                                        #neu da duoc 1s thi nhay den printKeyin1s de in ra man hinh
printKeyin1s:
        li $v0,4                        #bat dau chuoi lenh in ra so ky tu nhap duoc trong 1s
        la $a0,keyin1s
        syscall
        li    $v0,1                     #in ra so ky tu trong 1s
        add   $a0,$0,$t6
        syscall

Display_7SEG:
        li $t4, 10
        div $t6,$t4                     #lay so ky tu nhap duoc trong 1s chia cho 10
        mflo $t7                        #luu gia tri phan nguyen, gia tri nay se duoc hien thi o den LED ben trai (hang chuc)
        la $a2,LED7SEG                  #lay dia chi cua danh sach luu gia tri den LED
        add $a2,$a2,$t7                 #xac dinh dia chi cua gia tri den LED tuong ung
        lb $a0,0($a2)                   #lay noi dung cho vao $a0
        jal   SHOW_7SEG_LEFT            # show

        mfhi $t7                        #luu gia tri phan du, gia tri nay se duoc hien thi o den LED ben phai (hang don vi)
        la $a2,LED7SEG
        add $a2,$a2,$t7
        lb $a0,0($a2)                   # set value for segments
        jal  SHOW_7SEG_RIGHT            # show

```

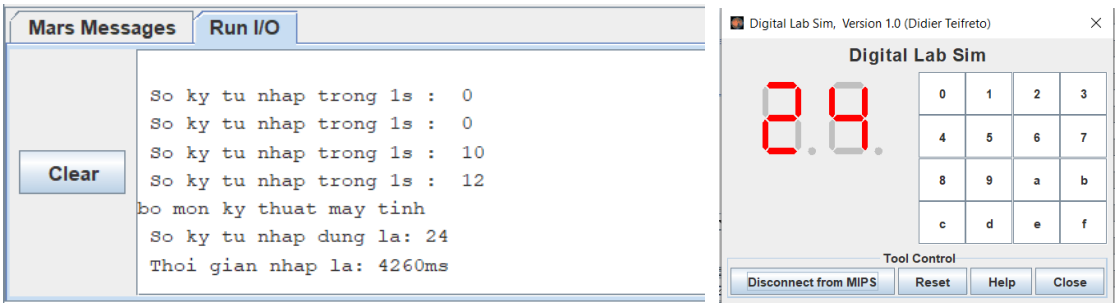
```
71
72          li    $t6,0                     #sau khi da hoan thanh, reset bien dem so ky tu trong 1s de bat dau chu ky moi
73          beq $v1,1,Notify
74
75      sleep:
76          addi    $v0,$zero,32
77          li      $a0,10                  # sleep 10 ms
78          syscall
79          nop
80          b       LOOP
81      END_main:                           #ket thuc chuong trinh
82          li $v0,10
83          syscall
84
85      SHOW_7SEG_LEFT:
86          li   $t0,  SEVENSEG_LEFT        # assign port's address
87          sb   $a0,  0($t0)               # assign new value
88          jr   $ra
89
90      SHOW_7SEG_RIGHT:
91          li   $t0,  SEVENSEG_RIGHT       # assign port's address
92          sb   $a0,  0($t0)               # assign new value
93          jr   $ra
94
95      .ktext    0x80000180                #interupt
96          mfc0  $t1, $13                   #luu nguyen nhan xay ra ngat
97          li    $t2, MASK_CAUSE_KEYBOARD
98          and   $at, $t1,$t2
99          beq   $at,$t2, Counter_Keyboar          #neu nguyen nhan xay ra ngat tu Keyboard, thuc hien Counter_Keyboard
100         j     EndProcess                        #neu khong phai tu Keyboard, EndProcess
101
102     Counter_Keyboar:
103     ReadKey:      lb   $t0, 0($k0)                # $t0 = [$k0] = KEY_CODE
104     WaitForDis:
105                   lw   $t2, 0($s1)                # $t2 = [$s1] = DISPLAY_READY
106                   beq  $t2, $zero, WaitForDis     # if $t2 == 0 then Polling
```

```
107     ShowKey:
108                 sb $t0, 0($s0)                  # hien thi ky tu vua nhap tu ban phim tren man hinh MMIO
109                 nop
110
111                 la  $t7,string                  # $t7 luu dia chi cua chuoi nhap vao
112                 add $t7,$t7,$s2                 #dia chi byte cuoi cung duoc nhap tu keyboard
113                 sb $t0,0($t7)
114                 addi $s2,$s2,1                  #so ky tu duoc nhap +1
115                 beq $t0,10,EndLOOP                      #neu gap ky tu "\n" (ENTER) (ket thuc xau) thi nhay den EndLOOP
116
117     EndProcess:
118     next_pc:    mfc0    $at, $14            # $at <= Coproc0.$14 = Coproc0.epc
119                 addi    $at, $at, 4         # $at = $at + 4 (next instruction)
120                 mtc0    $at, $14            # Coproc0.$14 = Coproc0.epc <= $at
121     RETURN:     eret                        # return from exception
122     EndLOOP:
123             li $v0,11
124             li $a0,'\n'                     # xuong dong
125             syscall
126             li $t1,0                        # $t1 luu so ky tu da duoc kiem tra
127             li $t3,0                        # dem so ky tu nhap dung
128             li $t8,24                       #$t8 luu do dai xau goc stringsource trong ma nguon
129             slt $t7,$s2,$t8                 #so sanh xem do dai xau nhap tu ban phim va do dai cua xau co dinh trong ma nguon
130                                             #xau nao nho hon thi duyet theo do dai cua xau do
131             bne $t7,1, Check
132             add $t8,$0,$s2
133             addi $t8,$t8,-1                 #tru 1 vi ky tu cuoi cung la dau enter thi khong can xet.
134     Check:
135             la $t2,string
136             add $t2,$t2,$t1
137             li $v0,11                       #in ra cac ky tu da nhap tu ban phim.
138             lb $t5,0($t2)                   #lay ky tu thu $t1 trong string luu vao $t5 de so sanh voi ky tu thu $t1 o stringsource
139             move $a0,$t5
140             syscall
141             la $t4,stringsource
142             add $t4,$t4,$t1
```

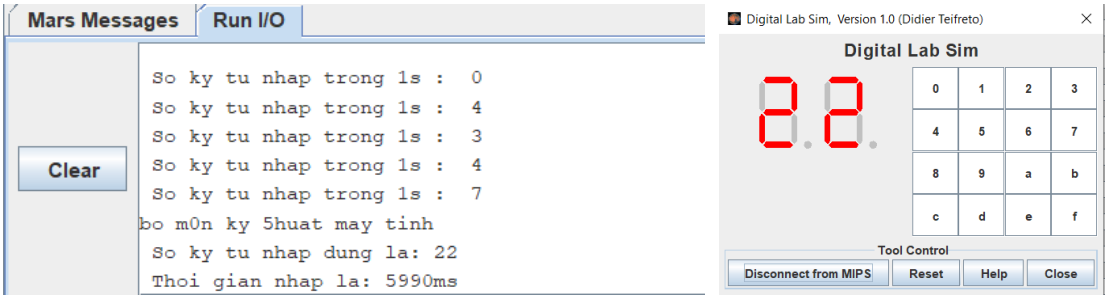
```
143         lb $t6,0($t4)                   #lay ky tu thu $t1 trong stringsource luu vao $t6
144         bne $t6,$t5,CONTINUE            #neu 2 ky tu thu $t1 khac nhau thi xet ky tu tiep theo
145         addi $t3,$t3,1                  #giong nhau thi tang bien dem so ky tu dung len 1
146 CONTINUE:
147         addi $t1,$t1,1                  #sau khi so sanh 1 ky tu, tang bien dem len
148         beq $t1,$t8,Print_NumRight      #neu da duyet het so ky tu can xet thi in ra man hinh so ky tu nhap dung
149         j Check                         #neu chua duyet het thi tiep tuc xet tiep cac ky tu
150 Print_NumRight:
151         li $v0,4                        #in xau numright
152         la $a0,numright
153         syscall
154         li $v0,1                        #in so ky tu dung
155         add $a0,$0,$t3
156         syscall
157         mul $t4,$s3, 10                 #tinh thoi gian nhap (tinh theo ms)
158         li $v0,4                        #in chuoi time
159         la $a0, time
160         syscall
161         add $a0,$0,$t4
162         li $v0,1
163         syscall
164         li $v0,4
165         la $a0, ms
166         syscall
167         li $v1,1
168         li $t6,0                        #sau khi ket thuc chuong trinh, so ky tu dung duoc luu vao $t6 roi quay tro ve phan hien thi.
169         add $t6,$0,$t3
170         b Display_7SEG
171 Notify:
172         li $v0, 50
173         la $a0, notification
174         syscall
175         beq $a0,0,main
176         b exit
177 exit:
178
```

IV. Kết Quả Chạy Mô Phỏng Chương Trình:

- Xâu nhập vào là: *bo mon ky thuat may tinh*



- Xâu nhập vào là: *bo m0n ky 5huat may tinh*



- Xâu nhập vào là: *bo mon ki thaut*