FINAL PROJECT

-Họ tên: Hồ Tuấn Huy
-MSSV: 20225856

8. Mô phỏng ổ đĩa RAID 5

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      |
 --------------        --------------        --------------
```

1. **Phân tích đề bài:**
   - Đầu vào là một dãy ký tự sao cho độ dài của dãy là bội của 8.
   - Đầu ra là in ra màn hình console kết quả chạy giả lập ổ đĩa RAID 5.
2. **Thuật toán được sử dụng:**

   Chương trình được chia ra 3 hàm chính như sau:
   - Hàm nhập chuỗi ký tự và kiểm tra chuỗi đó có chiều dài có phải là bội của 8 hay không, hoặc là chuỗi rỗng.
   - Hàm RAID 5 được chia làm 3 phần:

     + split1: Tính 4 byte từ 2 block đầu tiên và lưu vào Disk 3. Nếu còn chuỗi cần lưu, hàm sẽ tiếp tục split2.

     + split2: Tính 4 byte parity từ 2 block tiếp theo và lưu vào Disk 2. Nếu còn chuỗi cần lưu, hàm sẽ tiếp tục split3.

     + split3: Tính 4 byte parity từ 2 block tiếp theo và lưu vào Disk 1. Nếu còn chuỗi cần lưu, hàm sẽ quay trở về split1.

     Kết thúc 3 phần trên mà còn chuỗi ký tự và chuỗi party thì lặp lại từ phần 1, cho đến khi hết ký tự để xét thì dừng.
   - Hàm hex để chuyển 4 byte parity từ chuẩn ASCII sang hexa.

     ***Lưu ý:**

     + Trong quá trình xử lý, các giá trị parity được tính toán bằng cách XOR các giá trị của các block dữ liệu.

     + Chương trình đảm bảo rằng các giá trị parity được phân phối đều trên các đĩa, giúp bảo vệ dữ liệu theo nguyên tắc RAID 5.

3. Cách thực hiện:

Duyệt toàn bộ chuỗi ký tự, tính mã chẵn lẻ (parity) của 2 ký tự cách nhau 4 ký tự (đối với mỗi cặp ký tự, chương trình sẽ tính toán giá trị parity bằng cách XOR giá trị ASCII của 2 ký tự này). Lưu trữ ký tự đầu tiên vào một vùng nhớ trống (tương ứng một ổ đĩa), lưu trữ ký tự thứ 2 vào một vùng nhớ trống khác (tương ứng ổ đĩa khác), lưu giá trị mã ASCII của byte parity vào một vùng nhớ trống thứ 3 (tương ứng với ổ đĩa lưu trữ parity). In ra màn hình theo cấu hình thiết lập để được giả lập hệ thống ổ đĩa. Chương trình có thể lặp lại nhiều lần.

4. Ý nghĩa các thanh ghi trong chương trình:

- $s1: địa chỉ của Disk1.
- $s2: địa chỉ của Disk2.
- $s3: địa chỉ của Disk3.
- $t3: độ dài của chuỗi nhập vào.
- $t0: index.
- $t1: địa chỉ của ký tự hiện tại trong chuỗi.
- $t2: string[i].
- $t4: gán giá trị bằng 7 (cho lặp tới 0 để đủ 8 bit).
- $t7: địa chỉ của hex.
- $a0: chỉ số của mảng hex.

5. Source Code:

```
.data
        start: .asciiz "Nhap chuoi ky tu : "
        hex: .byte '0','1','2','3','4','5','6','7','8','9','a','b','c','d','e','f'
        d1: .space 4
        d2: .space 4
        d3: .space 4
        array: .space 32
        string: .space 5000
        enter: .asciiz "\n"
        error_length: .asciiz "Do dai chuoi khong hop le! Nhap lai.\n"
        m: .asciiz "     Disk 1              Disk 2             Disk 3\n"
        m2: .asciiz "----------------      ----------------      ----------------\n"
        m3: .asciiz "|     "
        m4: .asciiz "     |      "
        m5: .asciiz "[[ "
        m6: .asciiz "]]      "
        comma: .asciiz ","
        ms: .asciiz "Try again?"

.text
        la      $s1, d1                 # Tuong ung disk 1
        la      $s2, d2                 # Tuong ung disk 2
        la      $s3, d3                 # Tuong ung disk 3
```

```
        la      $a2, array                  # dia chi mang chua parity

input:  li      $v0, 4                      # nhap ten (chuoi)
        la      $a0, start
        syscall
        li      $v0, 8
        la      $a0, string
        li      $a1, 1000
        syscall
        move    $s0, $a0                    # s0 chua dia chi xau moi nhap
        li      $v0, 4
        la      $a0, m
        syscall
        li      $v0, 4
        la      $a0, m2
        syscall

#----------------------kiem tra do dai co chia het cho 8 khong--------------------------
length: addi    $t3, $zero, 0           # t3 = length
        addi    $t0, $zero, 0           # t0 = index

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: move $t5, $t3
        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       split1
test1:  beq     $t1, 8, split1
        j       error1
error1:li       $v0, 4
        la      $a0, error_length
        syscall
        j       input
#------------------------------ket thuc kiem tra do dai---------------------------------
```

```
#------------------------lay parity----------------------------
HEX:    li      $t4, 7
loopH:  blt     $t4, $0, endloopH
        sll     $s6, $t4, 2                     # s6 = t4*4
        srlv    $a0, $t8, $s6                   # a0 = t8>>s6
        andi    $a0, $a0, 0x0000000f # a0 = a0 & 0000 0000 0000 0000 0000 0000 0000
1111 => lay byte cuoi cung cua a0
        la      $t7, hex
        add     $t7, $t7, $a0
        bgt     $t4, 1, nextc
        lb      $a0, 0($t7)                     # print hex[a0]
        li      $v0, 11
        syscall
        # lb $t6, 0($t7)
        # beq $t6, 48, in
nextc:  addi    $t4,$t4,-1
        j       loopH
# in:   bgt $t4, 1, loopH
        # move $a0, $t6
        # li $v0, 11
        # syscall
endloopH: jr    $ra
#-------------------------------------------------------------

#------------------------------Mo phong RAID 5-------------------------------
# xet 6 khoi dau
# lan 1: luu vao 2 khoi 1,2; xor vao 3-------
split1:addi     $t0, $zero, 0                   # so byte duoc in ra (4 byte)
        addi    $t9, $zero, 0
        addi    $t8, $zero, 0
        la      $s1, d1
        la      $s2, d2
        la      $a2, array
print11:li      $v0, 4
        la      $a0, m3
        syscall
b11:    lb      $t1, ($s0)
        addi    $t3, $t3, -1
        sb      $t1, ($s1)
b21:    add     $s5, $s0, 4
        lb      $t2, ($s5)                      # t2 chua dia chi tung byte cua dick 2
        addi    $t3, $t3, -1
        sb      $t2, ($s2)
b31:    xor     $a3, $t1, $t2
```

```
        sw      $a3, ($a2)
        addi    $a2, $a2, 4
        addi    $t0, $t0, 1
        addi    $s0, $s0, 1
        addi    $s1, $s1, 1
        addi    $s2, $s2, 1
        bgt     $t0, 3, reset
        j       b11
reset:  la      $s1, d1
        la      $s2, d2
print12:lb      $a0, ($s1)
        li      $v0, 11
        syscall
        addi    $t9, $t9, 1
        addi    $s1, $s1, 1
        bgt     $t9, 3, next11
        j       print12
next11:li       $v0, 4
        la      $a0, m4
        syscall
        li      $v0, 4
        la      $a0, m3
        syscall
print13:lb      $a0, ($s2)
        li      $v0, 11
        syscall
        addi    $t8, $t8, 1
        addi    $s2, $s2, 1
        bgt     $t8, 3, next12
        j       print13
next12:li       $v0, 4
        la      $a0, m4
        syscall
        li      $v0, 4
        la      $a0, m5
        syscall

        la      $a2, array
        addi    $t9, $zero, 0
print14:lb      $t8, ($a2)
        jal     HEX
        li      $v0, 4
        la      $a0, comma
        syscall
```

```
        addi    $t9, $t9, 1
        addi    $a2, $a2, 4
        bgt     $t9, 2, end1                    # in ra 3 parity dau co dau ",", parity cuoi
cung k co
        j       print14
end1:   lb      $t8, ($a2)
        jal     HEX
        li      $v0, 4
        la      $a0, m6
        syscall
        li      $v0, 4
        la      $a0, enter
        syscall
        beq     $t3, 0, exit1
#-------------------------------------------------------------
split2:la       $a2, array
        la      $s1, d1
        la      $s3, d3
        addi    $s0, $s0, 4
        addi    $t0, $zero, 0
print21:li      $v0, 4
        la      $a0, m3
        syscall
b12:    lb      $t1, ($s0)
        addi    $t3, $t3, -1
        sb      $t1, ($s1)
b32:    add     $s5, $s0, 4
        lb      $t2, ($s5)
        addi    $t3, $t3, -1
        sb      $t2, ($s3)
b22:    xor     $a3, $t1, $t2
        sw      $a3, ($a2)
        addi    $a2, $a2, 4
        addi    $t0, $t0, 1
        addi    $s0, $s0, 1
        addi    $s1, $s1, 1
        addi    $s3, $s3, 1
        bgt     $t0, 3, reset2
        j       b12
reset2:la       $s1, d1
        la      $s3, d3
        addi    $t9, $zero, 0
print22:lb      $a0, ($s1)
        li      $v0, 11
```

```
        syscall
        addi    $t9, $t9, 1
        addi    $s1, $s1, 1
        bgt     $t9, 3, next21
        j       print22
next21:li       $v0, 4
        la      $a0, m4
        syscall
        la      $a2, array
        addi    $t9, $zero, 0
        li      $v0, 4
        la      $a0, m5
        syscall
print23:lb      $t8, ($a2)
        jal     HEX
        li      $v0, 4
        la      $a0, comma
        syscall
        addi    $t9, $t9, 1
        addi    $a2, $a2, 4
        bgt     $t9, 2, next22
        j       print23
next22:lb       $t8, ($a2)
        jal     HEX
        li      $v0, 4
        la      $a0, m6
        syscall
        li      $v0, 4
        la      $a0, m3
        syscall
        addi    $t8, $zero, 0
print24:lb      $a0, ($s3)
        li      $v0, 11
        syscall
        addi    $t8, $t8, 1
        addi    $s3, $s3, 1
        bgt     $t8, 3, end2
        j       print24

end2:   li      $v0, 4
        la      $a0, m4
        syscall
        li      $v0, 4
        la      $a0, enter
```

```
        syscall
        beq     $t3, 0, exit1
#-------------------------------------------------------------
split3:la       $a2, array
        la      $s2, d2
        la      $s3, d3
        addi    $s0, $s0, 4
        addi    $t0, $zero, 0
print31:li      $v0, 4
        la      $a0, m5
        syscall
b23:    lb      $t1, ($s0)
        addi    $t3, $t3, -1
        sb      $t1, ($s1)
b33:    add     $s5, $s0, 4
        lb      $t2, ($s5)
        addi    $t3, $t3, -1
        sb      $t2, ($s3)
b13:    xor     $a3, $t1, $t2
        sw      $a3, ($a2)
        addi    $a2, $a2, 4
        addi    $t0, $t0, 1
        addi    $s0, $s0, 1
        addi    $s1, $s1, 1
        addi    $s3, $s3, 1
        bgt     $t0, 3, reset3
        j       b23
reset3:la       $s2, d2
        la      $s3, d3
        la      $a2, array
        addi    $t9, $zero, 0
print32:lb      $t8, ($a2)
        jal     HEX
        li      $v0, 4
        la      $a0, comma
        syscall
        addi    $t9, $t9, 1
        addi    $a2, $a2, 4
        bgt     $t9, 2, next31
        j       print32
next31:lb       $t8, ($a2)
        jal HEX
        li      $v0, 4
        la      $a0, m6
```

```
        syscall
        li      $v0, 4
        la      $a0, m3
        syscall
        addi    $t9, $zero, 0
print33:lb      $a0, ($s2)
        li      $v0, 11
        syscall
        addi    $t9, $t9, 1
        addi    $s2, $s2, 1
        bgt     $t9, 3, next32
        j       print33
next32:addi     $t9, $zero, 0
        addi    $t8, $zero, 0
        li      $v0, 4
        la      $a0, m4
        syscall
        li      $v0, 4
        la      $a0, m3
        syscall
print34:lb      $a0, ($s3)
        li      $v0, 11
        syscall
        addi    $t8, $t8, 1
        addi    $s3, $s3, 1
        bgt     $t8, 3, end3
        j       print34

end3:   li      $v0, 4
        la      $a0, m4
        syscall
        li      $v0, 4
        la      $a0, enter
        syscall
        beq     $t3, 0, exit1
#-------------------------------end 6 khoi dau---------------------------------------
# chuyen sang 6 khoi tiep theo
nextloop: addi $s0, $s0, 4
        j split1

exit1:  li      $v0, 4
        la      $a0, m2
        syscall
        j       ask
```

```
#-------------------------------ket thuc mo phong RAID 5---------------------------------

#-----------------------------------try again---------------------------------------
ask:    li      $v0, 50
        la      $a0, ms
        syscall
        beq     $a0, 0, clear
        nop
        j       exit
        nop
# clear: dua string ve trang thai ban dau de thuc hien lai qua trinh
clear:  la      $s0, string
        add     $s3, $s0, $t5           # s3: dia chi byte cuoi cung duoc su dung trong string
        li      $t1, 0
goAgain: sb     $t1, ($s0)              # set byte o dia chi s0 thanh 0
        nop
        addi    $s0, $s0, 1
        bge     $s0, $s3, input
        nop
        j       goAgain
        nop
#-----------------------------------end try again-------------------------------------

exit:   li      $v0, 10
        syscall
```

6. Thực thi:

a. *Cách chạy chương trình:*

- Nhập một chuỗi ký tự từ bàn phím sao cho độ dài của chuỗi là bội của 8 và khác rỗng. Nếu không nhập đúng thì chương trình sẽ báo lỗi và yêu cầu người dùng nhập lại cho đến khi nhập đúng.
- Khi nhập đúng thì chương trình sẽ chạy và in ra giả lập ổ đĩa RAID 5.
- Khi chương trình chạy và in xong sẽ có lựa chọn thực hiện lại hoặc kết thúc chương trình.

b. *Kết quả thực thi:*

- Với trường hợp ví dụ đề bài:

```
Nhap chuoi ky 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     |
________________      ________________      ________________
```

- Với trường hợp nhập xâu có độ dài không phải là bội của 8:

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

- Với trường hợp nhập xâu có độ dài là bội của 8:

```
Nhap chuoi ky tu : hohuyhuy
      Disk 1                Disk 2               Disk 3
________________       ________________      ________________
|     hohu     |       |     yhuy     |      [[ 11,07,1d,0c]]
________________       ________________      ________________
```