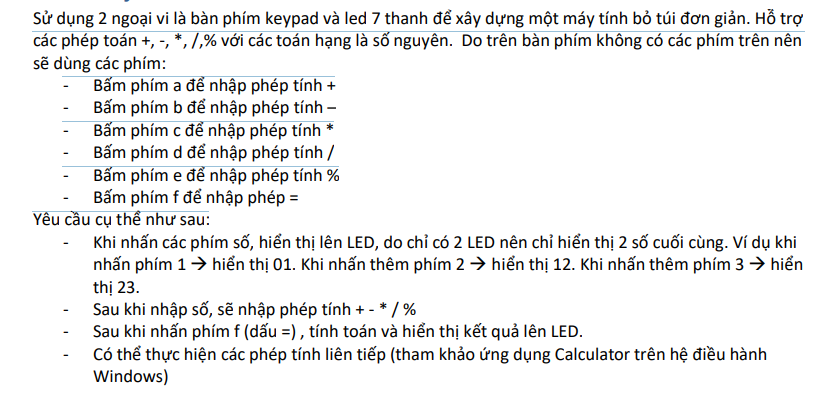
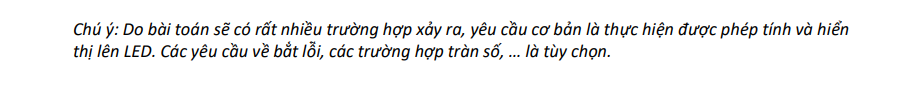
1. ***ĐỀ 1***0
2. **Đề bài : Máy tính bỏ túi**





1. **Phân tích cách làm**

* Khởi tạo chương trình chính là 1 vòng lặp vô hạn . Khi có một Exception xảy ra, MIPS sẽ luôn nhảy tới địa chỉ cố định 0x80000180 để thực hiện chương trình con phục vụ ngắt. Để viết chương trình con phục vụ ngắt, ta sẽ sử dụng chỉ thị .ktext để viết code ở địa chỉ 0x80000180 nói trên.
* Trong .ktext sẽ thực hiện các công việc:
* Quét bàn phim trên Digital Lab Sim và lấy địa chỉ vị trí của phím đã được nhập.
* Xác định cụ thể phim đó thuộc loại gì:

Nếu phím đó là số hạng thứ 1:

+ Update lại số thứ 1 hiện tại ( thông qua update)

+ Hiển thị ra LED

Nếu phím đó là các toán tử +,-,\*,/,% :

+ Thực hiện lưu lại số hạng thứ 1 ( thông qua hàm savefirstnum)

+ Reset lại thanh LED bên trái về giá trị 0 ( thông qua reset\_left\_7seg)

Nếu phím đó là số hạng thứ 2:

+ Update lại số hạng thứ 2 hiện tại

+ Hiển thị ra LED

Nếu phím đó là toán tử = :

+ Thực hiện lưu lại số hạng thứ 2 ( thông qua hàm savesecondnum)

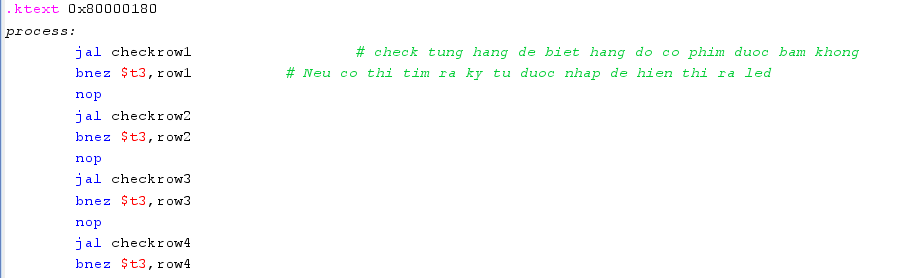
+ Gán thanh ghi $k0 =1 để có thể tai sử dụng kết quả phép tính

+ Kiểm tra xem phép toán này thuộc loại nào ( thông qua thanh ghi $s1)

+ Thực hiện phép toán và in kết quả ra cửa sổ RunI/O cũng như hiển thị ra LED

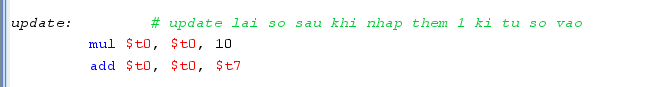
1. **Thuật toán**

* Xử lý khi có phím được nhập vào



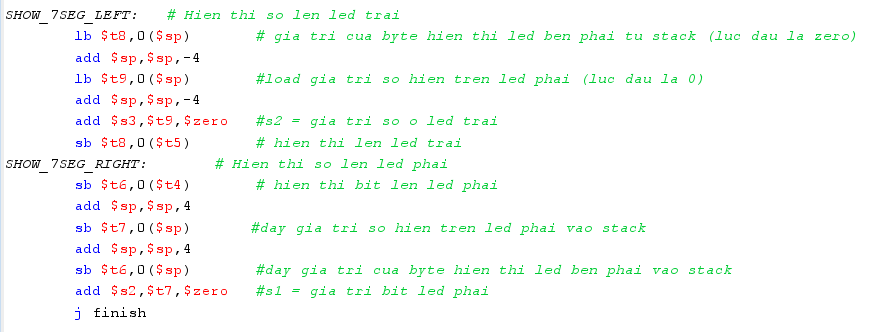
Ta sử dụng thủ tục checkrow để kiểm tra xem phím đó có thuộc trong hàng đang kiểm tra hay không. Nếu thuộc ( $t3 != 0 ) thì sẽ nhảy sang thủ tục row để kiểm tra cụ thể là phím gì.

* Cập nhất lại số khi nhập phím mới vào



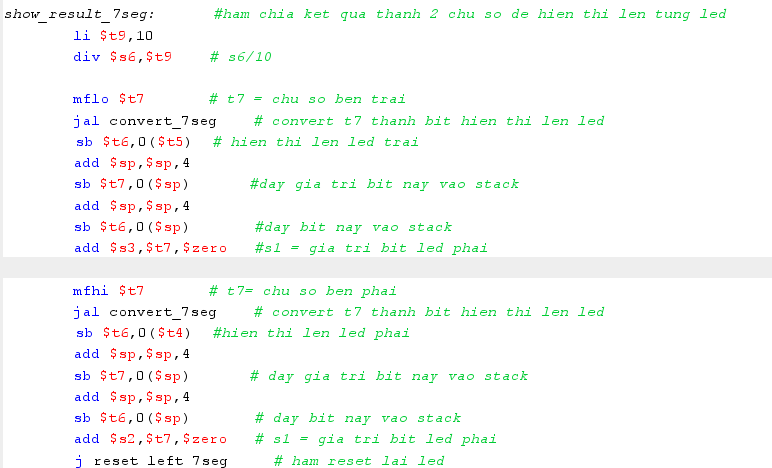
Biến tạm thời $t0 = 10 \* $t0 + $ t7 ( $t7 là giá trị của số hiện trên LED phải )

* Hiển thị chữ số bên trái và phải ra LED 7segment



Giải thuật của quá trình này là sử dụng ngăn xếp ( stack ). Mỗi khi có số mới được nhập vào từ bàn phim , ta **lấy lần lượt 2 phần tử đầu tiên trên ngăn xếp** ( tương đương với giá trị và byte )  **gán cho các thanh ghi hiển thị bên LED trái** rồi sau đó lại **gán giá trị các thanh ghi liên quan đến phím vừa nhập cho các thanh ghi hiển thị bên LED phải** và **đẩy 2 phần tử mới đó vào ngăn xếp**. Quá trình này sẽ liên tục lặp lại .

* Hiển thị kết quả phép tinh ra LED 7seg



Để hiển thị kết quả phép tinh ra LED 7seg sao cho chỉ hiện ra 2 chữ số cuối của kết quả, ta chia kết quả phép tinh cho 100 rồi lấy phần dư là 2 chữ số cuối. Sau đó tiếp tục chia phần dư này cho 10 . Thương của phép tính trên chinh là chữ số bên trai, còn phần dư chính là chữ số bên phải.

1. **Mã nguồn**

# n10\_g15\_NguyenTuanNam

.data

.eqv IN\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0012

.eqv OUT\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0014

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

# col 0x1 col 0x2 col 0x4 col 0x8

#

# row 0x1 0 1 2 3

# 0x11 0x21 0x41 0x81

#

# row 0x2 4 5 6 7

# 0x12 0x22 0x42 0x82

#

# row 0x4 8 9 a b

# 0x14 0x24 0x44 0x84

#

# row 0x8 c d e f

# 0x18 0x28 0x48 0x88

#

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

# command row number of hexadecimal keyboard (bit 0 to 3)

# Eg. assign 0x1, to get key button 0,1,2,3

# assign 0x2, to get key button 4,5,6,7

# NOTE must reassign value for this address before reading,

# eventhough you only want to scan 1 row

# 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 SEVENSEG\_LEFT 0xFFFF0011 # Dia chi cua den led 7 doan trai.

.eqv SEVENSEG\_RIGHT 0xFFFF0010 # Dia chi cua den led 7 doan phai

zero: .byte 0x3f #gia tri bit tuong ung

one: .byte 0x6

two: .byte 0x5b

three: .byte 0x4f

four: .byte 0x66

five: .byte 0x6d

six: .byte 0x7d

seven: .byte 0x7

eight: .byte 0x7f

nine: .byte 0x6f

warning: .asciiz "khong the thuc hien phep chia cho 0 \n"

.text

main:

declare:

li $t5,SEVENSEG\_LEFT # $t5: Bien gia tri so cua den LED trai

li $t4,SEVENSEG\_RIGHT # $t1: Bien gia tri so cua den LED phai

li $s0,0 # bien kiem tra loai bien nhap vao, 0: so, 1 :toan tu

li $s1,0 # bien kiem tra loai toan tu, 1:cong, 2:tru, 3:nhan, 4:chia, 5 : %

li $s2,0 # so dang hien thi o led phai

li $s3,0 # so dang hien thi o led trai

li $s4,0 # so thu nhat

li $s5,0 # so thu hai

li $s6,0 # ket qua 2 so, cong ,tru, nhan, chia

li $t0,0 # gia tri so tam thoi

li $t1, IN\_ADDRESS\_HEXA\_KEYBOARD #bien dieu khien hang keyboard va enable keyboard interrupt

li $t2, OUT\_ADDRESS\_HEXA\_KEYBOARD #bien chua vi tri key nhap vao the hang va cot

li $t3, 0x80 # bit 7 of = 1 to enable interrupt

sb $t3, 0($t1)

li $t6,0 #byte hien thi len led ben phai

li $t7,0 #gia tri cua so hien tren led ben phai

li $t8,0 #byte hien thi len led ben trai

li $t9,0 #gia tri cua so hien tren led ben trai

storefirstvalue:

li $t9,0 #gia tri cua bit can hien thi ban dau :0

addi $sp,$sp,4 #day vao stack

sb $t9,0($sp)

lb $t8,zero #bit dau tien can hien thi :0

addi $sp,$sp,4 #day vao stack

sb $t8 ,0($sp)

loop: #loop de doi nhap phim tu digital lab sim

nop

b loop

endloop:

end\_main:

li $v0,10

syscall

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

# GENERAL INTERRUPT SERVED ROUTINE for all interrupts

# Xu ly khi co interupt va hien thi so vua bam len den led 7 doan

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

.ktext 0x80000180

process:

jal checkrow1 # check tung hang de biet hang do co phim duoc bam khong

bnez $t3,row1 # Neu co thi tim ra ky tu duoc nhap de hien thi ra led

nop

jal checkrow2

bnez $t3,row2

nop

jal checkrow3

bnez $t3,row3

nop

jal checkrow4

bnez $t3,row4

checkrow1:

addi $sp,$sp,4

sw $ra,0($sp) # luu ra lai vi ve sau co the doi

li $t3,0x81 # Kich hoat interrupt, cho phep bam phim o hang 1

sb $t3,0($t1)

li $t2,OUT\_ADDRESS\_HEXA\_KEYBOARD # gan dia chi chua vi tri cua phim duoc nhap cho t2

lb $t3,0($t2) # load byte cua vi tri phim duoc nhap

lw $ra,0($sp)

addi $sp,$sp,-4

jr $ra

checkrow2:

addi $sp,$sp,4

sw $ra,0($sp)

li $t3,0x82 # kich hoat interrupt, cho phep bam phim o hang 2

sb $t3,0($t1)

li $t2,OUT\_ADDRESS\_HEXA\_KEYBOARD # gan dia chi chua vi tri cua phim duoc nhap cho t2

lb $t3,0($t2) # load byte cua vi tri phim duoc nhap

lw $ra,0($sp)

addi $sp,$sp,-4

jr $ra

checkrow3:

addi $sp,$sp,4

sw $ra,0($sp)

li $t3,0x84 # Kich hoat interrupt, cho phep bam phim o hang 3

sb $t3,0($t1)

li $t2,OUT\_ADDRESS\_HEXA\_KEYBOARD # gan dia chi chua vi tri cua phim duoc nhap cho t2

lb $t3,0($t2) # load byte cua vi tri phim duoc nhap

lw $ra,0($sp)

addi $sp,$sp,-4

jr $ra

checkrow4:

addi $sp,$sp,4

sw $ra,0($sp)

li $t3,0x88 # Kich hoat interrupt, cho phep bam phim o hang 4

sb $t3,0($t1)

li $t2,OUT\_ADDRESS\_HEXA\_KEYBOARD # gan dia chi chua vi tri cua phim duoc nhap cho t2

lb $t3,0($t2) # load byte cua vi tri phim duoc nhap

lw $ra,0($sp)

addi $sp,$sp,-4

jr $ra

row1: # check tung ky tu trong row

beq $t3,0x11,key\_0

beq $t3,0x21,key\_1

beq $t3,0x41,key\_2

beq $t3,0xffffff81,key\_3

key\_0:

li $k0,0 # nhap hep tinh moi ( khong su dung lai ket qu phep tinh cu )

lb $t6,zero #t4=zero (tuc = 0x3f, tong cac bit thanh ghi de tao thanh so 0 tren led)

li $t7,0 #t7= 0

j update

key\_1:

li $k0,0

lb $t6,one

li $t7,1

j update

key\_2:

li $k0,0

lb $t6,two

li $t7,2

j update

key\_3:

li $k0,0

lb $t6,three

li $t7,3

j update

row2:

beq $t3,0x12,key\_4

beq $t3,0x22,key\_5

beq $t3,0x42,key\_6

beq $t3,0xffffff82,key\_7

key\_4:

li $k0,0

lb $t6,four

li $t7,4

j update

key\_5:

li $k0,0

lb $t6,five

li $t7,5

j update

key\_6:

li $k0,0

lb $t6,six

li $t7,6

j update

key\_7:

li $k0,0

lb $t6,seven

li $t7,7

j update

row3:

beq $t3,0x14,key\_8

beq $t3,0x24,key\_9

beq $t3 0x44,key\_a

beq $t3 0xffffff84,key\_b

key\_8:

li $k0,0

lb $t6,eight

li $t7,8

j update

key\_9:

li $k0,0

lb $t6,nine

li $t7,9

j update

key\_a: #bam phim cong

addi $a3,$zero,1

addi $s0,$s0,1 # bien check phim nhap vao la 1 toan tu

addi $s1,$zero,1 # bien check the loai toan tu

j savefirstnum # Luu lai so thu 1

key\_b: #bam phim tru

addi $a3,$zero,2

addi $s0,$s0,1

addi $s1,$zero,2

j savefirstnum

row4:

beq $t3,0x18,key\_c

beq $t3,0x28,key\_d

beq $t3,0x48,key\_e

beq $t3 0xffffff88,key\_f

key\_c: #bam phim nhan

addi $a3,$zero,3

addi $s0,$s0,1

addi $s1,$zero,3

j savefirstnum

key\_d: #bam phim chia

addi $a3,$zero,4

addi $s0,$s0,1

addi $s1,$zero,4

j savefirstnum

key\_e: #bam phim lay so du

addi $a3,$zero,5

addi $s0,$s0,1

addi $s1,$zero,5

j savefirstnum

savefirstnum: # Luu lai so thu 1 sau khi co phim nhan vao la 1 toan tu

bne $k0,$zero,savefirstnumtoRecalculate

addi $s4, $t0, 0

li $t0, 0

j checkoperator

savefirstnumtoRecalculate : # check neu nhu co bien su dung lai kq phep tinh

add $t0,$zero,$s6

addi $s4, $t0, 0

li $t0, 0

j checkoperator

key\_f: # bam phim =

li $k0,1 # co the su dung ket qua phep tinh nay o lan sau

jal savesecondnumber

beq $s1,1,sum # s3=1--> cong

beq $s1,2,subtract

beq $s1,3,multiply

beq $s1,4,divide

beq $s1,5,surplus

savesecondnumber: #ham tinh so thu 2 dang hien thi tren led trong 2 so

addi $s5, $t0, 0 # Luu lai so thu 2 khi co toan tu nhap vao la =

jr $ra

sum:

add $s6,$s5,$s4

li $s1,0

li $t0, 0

j printsum

nop # s6=s5+s4

printsum:

li $v0, 1

move $a0, $s4

syscall

li $s4, 0

li $v0, 11

li $a0, ' '

syscall

li $v0, 11

li $a0, '+'

syscall

li $v0, 11

li $a0, ' '

syscall

li $v0, 1

move $a0, $s5

syscall

li $s5, 0 #reset $s5

li $v0, 11

li $a0, ' '

syscall

li $v0, 11

li $a0, '='

syscall

li $v0, 11

li $a0, ' '

syscall

li $v0, 1

move $a0, $s6

syscall

nop

li $v0, 11

li $a0, '\n'

syscall

li $s7,100

div $s6,$s7

mfhi $s6 # chi lay 2 chu so cuoi cua ket qua de in ra led

j show\_result\_7seg # chuyen den ham chia ket qua thanh 2 chu so de hien thi len tung led

nop

subtract:

sub $s6,$s4,$s5 # s6=s4-s5

li $s1,0

li $t0, 0

j printsubtract

nop

printsubtract:

li $v0, 1

move $a0, $s4

syscall

li $v0, 11

li $a0, ' '

syscall

li $v0, 11

li $a0, '-'

syscall

li $v0, 11

li $a0, ' '

syscall

li $v0, 1

move $a0, $s5

syscall

li $v0, 11

li $a0, ' '

syscall

li $v0, 11

li $a0, '='

syscall

li $v0, 11

li $a0, ' '

syscall

li $v0, 1

move $a0, $s6

syscall

li $v0, 11

li $a0, '\n'

syscall

j show\_result\_7seg # chuyen den ham chia ket qua thanh 2 chu so de hien thi len tung led

nop

multiply:

mul $s6,$s4,$s5 # s6=s4\*s5

li $s1,0

li $t0, 0

j printmultiply

nop

printmultiply:

li $v0, 1

move $a0, $s4

syscall

li $v0, 11

li $a0, ' '

syscall

li $v0, 11

li $a0, '\*'

syscall

li $v0, 11

li $a0, ' '

syscall

li $v0, 1

move $a0, $s5

syscall

li $v0, 11

li $a0, ' '

syscall

li $v0, 11

li $a0, '='

syscall

li $v0, 11

li $a0, ' '

syscall

li $v0, 1

move $a0, $s6

syscall

li $v0, 11

li $a0, '\n'

syscall

li $s7,100

div $s6,$s7

mfhi $s6 # chi lay phan du la 2 chu so sau cung cua ket qua de in ra

j show\_result\_7seg # chuyen den ham chia ket qua thanh 2 chu so de hien thi len tung led

nop

divide:

beq $s5,0,divide0

li $s1,0

div $s4,$s5 # s6=s4/s5

mflo $s6

mfhi $s7

li $t0, 0

j printdivide

nop

printdivide:

li $v0, 1

move $a0, $s4

syscall

li $v0, 11

li $a0, ' '

syscall

li $v0, 11

li $a0, '/'

syscall

li $v0, 11

li $a0, ' '

syscall

li $v0, 1

move $a0, $s5

syscall

li $v0, 11

li $a0, ' '

syscall

li $v0, 11

li $a0, '='

syscall

li $v0, 11

li $a0, ' '

syscall

li $v0, 1

move $a0, $s6

syscall

li $v0, 11

li $a0, ' '

syscall

li $v0, 11

li $a0, 'r'

syscall

li $v0, 11

li $a0, '='

syscall

li $v0, 1

move $a0, $s7

syscall

li $v0, 11

li $a0, '\n'

syscall

j show\_result\_7seg # chuyen den ham chia ket qua thanh 2 chu so de hien thi len tung led

nop

divide0:

li $v0, 55

la $a0, warning

li $a1, 0

syscall

j reset\_left\_7seg

surplus:

beq $s5,0,divide0

li $s1,0

div $s4,$s5 # s6=s4/s5

mfhi $s6

li $t0, 0

j printsurplus

nop

printsurplus:

li $v0, 1

move $a0, $s4

syscall

li $v0, 11

li $a0, ' '

syscall

li $v0, 11

li $a0, '%'

syscall

li $v0, 11

li $a0, ' '

syscall

li $v0, 1

move $a0, $s5

syscall

li $v0, 11

li $a0, ' '

syscall

li $v0, 11

li $a0, '='

syscall

li $v0, 11

li $a0, ' '

syscall

li $v0, 1

move $a0, $s6

syscall

li $v0, 11

li $a0, '\n'

syscall

j show\_result\_7seg # chuyen den ham chia ket qua thanh 2 chu so de hien thi len tung led

nop

show\_result\_7seg: #ham chia ket qua thanh 2 chu so de hien thi len tung led

li $t9,10

div $s6,$t9 # s6/10

mflo $t7 # t7 = chu so ben trai

jal convert\_7seg # convert t7 thanh bit hien thi len led

sb $t6,0($t5) # hien thi len led trai

add $sp,$sp,4

sb $t7,0($sp) #day gia tri bit nay vao stack

add $sp,$sp,4

sb $t6,0($sp) #day bit nay vao stack

add $s3,$t7,$zero #s1 = gia tri bit led phai

mfhi $t7 # t7= chu so ben phai

jal convert\_7seg # convert t7 thanh bit hien thi len led

sb $t6,0($t4) #hien thi len led phai

add $sp,$sp,4

sb $t7,0($sp) # day gia tri bit nay vao stack

add $sp,$sp,4

sb $t6,0($sp) # day bit nay vao stack

add $s2,$t7,$zero # s1 = gia tri bit led phai

j reset\_left\_7seg # ham reset lai led

convert\_7seg:

addi $sp,$sp,4

sw $ra,0($sp)

beq $t7,0,case\_0 # t7=0 -->ham chuyen 0 thanh bit zero hien thi len led

beq $t7,1,case\_1

beq $t7,2,case\_2

beq $t7,3,case\_3

beq $t7,4,case\_4

beq $t7,5,case\_5

beq $t7,6,case\_6

beq $t7,7,case\_7

beq $t7,8,case\_8

beq $t7,9,case\_9

case\_0: #ham chuyen 0 thanh bit zero hien thi len led

lb $t6,zero #t4=zero

lw $ra,0($sp)

addi $sp,$sp,-4

jr $ra

case\_1:

lb $t6,one

lw $ra,0($sp)

addi $sp,$sp,-4

jr $ra

case\_2:

lb $t6,two

lw $ra,0($sp)

addi $sp,$sp,-4

jr $ra

case\_3:

lb $t6,three

lw $ra,0($sp)

addi $sp,$sp,-4

jr $ra

case\_4:

lb $t6,four

lw $ra,0($sp)

addi $sp,$sp,-4

jr $ra

case\_5:

lb $t6,five

lw $ra,0($sp)

addi $sp,$sp,-4

jr $ra

case\_6:

lb $t6,six

lw $ra,0($sp)

addi $sp,$sp,-4

jr $ra

case\_7:

lb $t6,seven

lw $ra,0($sp)

addi $sp,$sp,-4

jr $ra

case\_8:

lb $t6,eight

lw $ra,0($sp)

addi $sp,$sp,-4

jr $ra

case\_9:

lb $t6,nine

lw $ra,0($sp)

addi $sp,$sp,-4

jr $ra

update: # update lai so sau khi nhap them 1 ki tu so vao

mul $t0, $t0, 10

add $t0, $t0, $t7

checkoperator:

beq $s0,1,reset\_left\_7seg # reset lai so o ben leg ben trai ve 0 khi phim vua nhap vao la 1 toan tu

nop

SHOW\_7SEG\_LEFT: # Hien thi so len led trai

lb $t8,0($sp) # gia tri cua byte hien thi led ben phai tu stack (luc dau la zero)

add $sp,$sp,-4

lb $t9,0($sp) #load gia tri so hien tren led phai (luc dau la 0)

add $sp,$sp,-4

add $s3,$t9,$zero #s2 = gia tri so o led trai

sb $t8,0($t5) # hien thi len led trai

SHOW\_7SEG\_RIGHT: # Hien thi so len led phai

sb $t6,0($t4) # hien thi bit len led phai

add $sp,$sp,4

sb $t7,0($sp) #day gia tri so hien tren led phai vao stack

add $sp,$sp,4

sb $t6,0($sp) #day gia tri cua byte hien thi led ben phai vao stack

add $s2,$t7,$zero #s1 = gia tri bit led phai

j finish

reset\_left\_7seg: # reset lai so o ben tahi leg ben trai ve 0 khi phim vua nhap vao la 1 toan tu

li $s0,0 #s0=0--> doi nhap so tiep theo trong 2 so

li $t9,0

addi $sp,$sp,4

sb $t9,0($sp)

lb $t8,zero # day bit zero vao stack

addi $sp,$sp,4

sb $t8,0($sp)

finish:

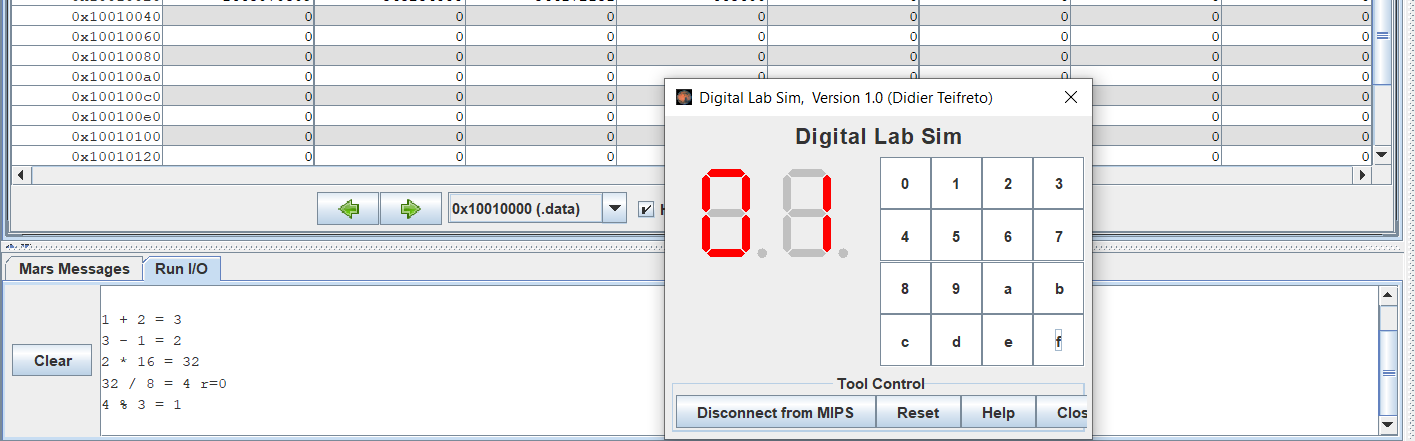
la $a3, loop

mtc0 $a3, $14

eret

1. **Kết quả chạy mô phỏng**

* **Các phép toán thường**

****

* **Phép toán chia cho 0**

