**PROJECT REPORT**

**Assembly Language Lab Final Project**

**NGUYEN MANH KHANG – 20176792**

SIMPLE CALCULATOR

**Problem:**

Sử dụng 2 ngoại vi là bàn phím và led 7 thanh để xây dựng một máy tính bỏ túi đơn giản. Hỗ trợ các phép toán +, -, \*, /. Do trên bàn phím không có các phím trên nên sẽ dùng các phím

* Bấm phím a để nhập phép tính +
* Bấm phím b để nhập phép tính –
* Bấm phím c để nhập phép tính \*
* Bấm phím d để nhập phép tính /
* Bấm phím f để nhập phép =

Yêu cầu cụ thể như sau:

* Khi nhấn các phím số, hiển thị lên LED, do chỉ có 2 LED nên chỉ hiện thị 2 số cuối cùng. Ví dụ khi nhấn phím 1 à hiện thị 01. Khi nhấn thêm phím 2 à hiển thị 12. Khi nhấn thêm phím 3 à hiển thị 23.
* Sau khi nhập số, sẽ nhập phép tính + - \* /
* Sau khi nhấn phím f (dấu =) , tính toán và hiển thị kết quả lên LED.

Chú ý: Do bài toán sẽ có rất nhiều trường hợp xảy ra, yêu cầu cơ bản là thực hiện được phép tính và hiển thị lên LED. Các yêu cầu về bắt lỗi, các trường hợp tràn số, … là mở rộng, không bắt buộc.

**How to use:**

* Open digital lab sim tool and run the program
* Click the numbers in hexa keyboard and the corresponding number will appear on the seven segment display
* Choose an operator (see the problem description for what key is equal to what operator)
* Choose the number again, if the user doesn’t click a number key, then the number will be default to zero
* Click f to compute the expression, the last 2 digit of the result will be display on the seven segment display. The whole equation is logged to the console

**Solution:**

The main program is simply an infinite loop. A value, 0x80, is stored to the IN\_ADDRESS\_HEXA\_KEYBOARD to enable interrupt on each key press. When a key is pressed, the program jump to the .ktext section, which is the process of the calculator. In .ktext we have:

* Scan the keyboard and get the code of the pressed keyboard
* Convert the keyboard code to number and seven segment code
* Check whether it is a number or an operator
  + If it is a number:
* Push the number to the memory
* Display it at the seven segments display
* Exit the handler
  + If it is an operator:
    - If it is not an “equal”, then put it in the memory, change $s0 to 1 (indicate now the next number go in will be the second operand), then end the exception
    - If it is an “equal”, proceed to next part
* After press equal, the program will display the answer to the seven segments display

**Parameters:**

$t0 Bien gia tri so cua den LED trai

$t5 Bien gia tri so cua den LED phai

s0 Bien kiem tra loai bien nhap vao, 0: so, 1 :toan tu, 2: terminate key

$s1 So dang hien thi o led phai

$s2 So dang hien thi o led trai

$s3 Bien kiem tra loai toan tu, 1:cong, 2:tru, 3:nhan, 4:chia

$s4 So thu nhat

$s5 So thu 2

$s6 Ket qua 2 so, cong ,tru, nhan, chia

$t1 IN\_ADDRESS\_HEXA\_KEYBOARD bien dieu khien hang keyboard va

enable keyboard interrupt

$t2 OUT\_ADDRESS\_HEXA\_KEYBOARD #bien chua vi tri key nhap vao

theo hang va cot

$t3 Bit dung enable keyboard interrupt va enable

kiem tra tung hang keyboard

$t7 Gia tri nguyen cua so hien tren led

$t4 Byte cua so hien thi len led ,zero->nine

**Source Code:**

.data

zero: .byte 0x3f

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

mess1: .asciiz "khong the tinh duoc so am \n"

mess2: .asciiz "khong the chia cho so 0 \n"

.eqv IN\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0012

# 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 OUT\_ADDRESS\_HEXA\_KEYBOARD 0xFFFF0014

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

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

.text

main:

li $t0,SEVENSEG\_LEFT # $t0: Bien gia tri so cua den LED trai

li $t5,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, 2: terminate key

li $s1,0 # so dang hien thi o led phai

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

li $s3,0 # bien kiem tra loai toan tu, 1:cong, 2:tru, 3:nhan, 4:chia

li $s4,0 # so thu nhat

li $s5,0 # so thu 2

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

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

# Enable the interrupt of Keyboard matrix 4x4 of Digital Lab Sim

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 dung enable keyboard interrupt va enable kiem tra tung hang keyboard

sb $t3, 0($t1)

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

li $t4,0 #byte hien thi len led ,zero->nine

storefirstvalue:

li $t7,0 #gia tri bit can hien thi ban dau :0

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

sb $t7,0($sp)

lb $t4,zero #bit dau tien can hien thi :0

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

sb $t4,0($sp)

loop1: #loop de doi nhap phim tu digital lab sim

beq $s0,2,endloop1 #neu phim terminate(phim e) duoc bam ,thoat loop

nop

nop

nop

nop

b loop1

nop

nop

nop

b loop1

nop

nop

b loop1

endloop1:

end\_main:

li $v0,10

syscall

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

# GENERAL INTERRUPT SERVED ROUTINE for all interrupts

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

.ktext 0x80000180

process:

jal checkrow1 #check hang 1 xem co phim nao duoc nhap ko

bnez $t3,convertrow1 #t3 != 0 --> co phim duoc nhap, convert phim do thanh bit hien ra led

nop

jal checkrow2

bnez $t3,convertrow2

nop

jal checkrow3

bnez $t3,convertrow3

nop

jal checkrow4

bnez $t3,convertrow4

checkrow1:

addi $sp,$sp,4

sw $ra,0($sp)

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

sb $t3,0($t1)

jal getvalue # get vi tri ( hang va cot ) cua phim duoc nhao neu co

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)

jal getvalue

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)

jal getvalue

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)

jal getvalue

lw $ra,0($sp)

addi $sp,$sp,-4

jr $ra

getvalue:

addi $sp,$sp,4

sw $ra,0($sp)

li $t2,OUT\_ADDRESS\_HEXA\_KEYBOARD #dia chi chua ci tri phim duoc nhap

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

lw $ra,0($sp)

addi $sp,$sp,-4

jr $ra

convertrow1: #ham convert tu vi tri sang bit de chuyen den led

beq $t3,0x11,case\_zero #0x11 -->phim o hang 1 cot 1--> 0

beq $t3,0x21,case\_one

beq $t3,0x41,case\_two

beq $t3,0xffffff81,case\_three

case\_zero:

lb $t4,zero #t4=zero (tuc = 63, tong cac bit thanh ghi de tao thanh so 0 tren led)

li $t7,0 #t7= gia tri cua t4

j done

case\_one:

lb $t4,one

li $t7,1

j done

case\_two:

lb $t4,two

li $t7,2

j done

case\_three:

lb $t4,three

li $t7,3

j done

convertrow2:

beq $t3,0x12,case\_four

beq $t3,0x22,case\_five

beq $t3,0x42,case\_six

beq $t3,0xffffff82,case\_seven

case\_four:

lb $t4,four

li $t7,4

j done

case\_five:

lb $t4,five

li $t7,5

j done

case\_six:

lb $t4,six

li $t7,6

j done

case\_seven:

lb $t4,seven

li $t7,7

j done

convertrow3:

beq $t3,0x14,case\_eight

beq $t3,0x24,case\_nine

beq $t3 0x44,case\_a

beq $t3 0xffffff84,case\_b

case\_eight:

lb $t4,eight

li $t7,8

j done

case\_nine:

lb $t4,nine

li $t7,9

j done

case\_a: #truong hop phim cong

addi $s0,$s0,1 #bien check phim nhap vao chuyen thanh 1(chung to nhap vao 1 toan tu)

addi $s3,$zero,1 #bien check loai toan tu chuyen thanh 1(tuc phep cong)

j setfirstnumber #chuyen den ham chuyen 2 byte dang hien tren 2 led thanh so de tinh toan

case\_b: #truong hop phim tru

addi $s0,$s0,1

addi $s3,$zero,2

j setfirstnumber

convertrow4:

beq $t3,0x18,case\_c

beq $t3,0x28,case\_d

beq $t3,0x48,case\_e

beq $t3 0xffffff88,case\_f

case\_c: #truong hop phim nhan

addi $s0,$s0,1

addi $s3,$zero,3

j setfirstnumber

case\_d: #truong hop phim chia

addi $s0,$s0,1

addi $s3,$zero,4

j setfirstnumber

setfirstnumber: #ham tinh so dau tien hien thi tren led trong 2 so

mul $s4,$s2,10 #s4=s2\*10+s1

add $s4,$s4,$s1

j done

case\_e: #truong hop terminate key

addi $s0,$s0,2

j finish

case\_f: #truong hop bam =

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

mul $s5,$s2,10 #s5=s2\*10+s1

add $s5,$s5,$s1

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

beq $s3,2,tru

beq $s3,3,nhan

beq $s3,4,chia

cong:

add $s6,$s5,$s4

j incong

nop #s6=s5+s4

incong:

li $v0, 1

move $a0, $s4

syscall

li $v0, 11

li $a0, '+'

syscall

li $v0, 1

move $a0, $s5

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 2 chu so cuoi cua ket qua de in ra led

j splitnumber # chuyen den ham chia ket qua thanh 2 chu so de hien thi len tung led

nop

tru:

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

blt $s6,0,truam

j intru

nop

intru:

li $v0, 1

move $a0, $s4

syscall

li $v0, 11

li $a0, '-'

syscall

li $v0, 1

move $a0, $s5

syscall

li $v0, 11

li $a0, '='

syscall

li $v0, 1

move $a0, $s6

syscall

li $v0, 11

li $a0, '\n'

syscall

j splitnumber # chuyen den ham chia ket qua thanh 2 chu so de hien thi len tung led

nop

nhan:

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

j innhan

nop

innhan:

li $v0, 1

move $a0, $s4

syscall

li $v0, 11

li $a0, '\*'

syscall

li $v0, 1

move $a0, $s5

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 #ch? l?y 2 ch? s? sau cùng c?a k?t qu? ?? in ra

j splitnumber # chuyen den ham chia ket qua thanh 2 chu so de hien thi len tung led

nop

chia:

beq $s5,0,chia0

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

mflo $s6

mfhi $s7

j inchia

nop

inchia:

li $v0, 1

move $a0, $s4

syscall

li $v0, 11

li $a0, '/'

syscall

li $v0, 1

move $a0, $s5

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 splitnumber # chuyen den ham chia ket qua thanh 2 chu so de hien thi len tung led

nop

chia0:

li $v0, 55

la $a0, mess2

li $a1, 0

syscall

j resetled

truam:

li $v0, 55

la $a0, mess1

li $a1, 0

syscall

j resetled

splitnumber: #ham chia ket qua thanh 2 chu so de hien thi len tung led

li $t8,10

div $s6,$t8 #s6/10

mflo $t7 #t7 = result

jal convert #chuyen den ham chuyen t7 thanh bit hien thi len led

sb $t4,0($t0) # hien thi len led trai

mfhi $t7 #t7= remainder

jal convert #convert t7 thanh bit hien thi len led

#addi $sp,$sp,4

sb $t4,0($t5) #hien thi len led phai

j resetled #ham reset lai led

convert:

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 $t4,zero #t4=zero

j finishconvert #ket thuc

case\_1:

lb $t4,one

j finishconvert

case\_2:

lb $t4,two

j finishconvert

case\_3:

lb $t4,three

j finishconvert

case\_4:

lb $t4,four

j finishconvert

case\_5:

lb $t4,five

j finishconvert

case\_6:

lb $t4,six

j finishconvert

case\_7:

lb $t4,seven

j finishconvert

case\_8:

lb $t4,eight

j finishconvert

case\_9:

lb $t4,nine

j finishconvert

finishconvert:

lw $ra,0($sp)

addi $sp,$sp,-4

jr $ra

done:

beq $s0,1,resetled #s0=1-->toan tu-->chuyen den ham reset led

loadtoleftled: # ham hien thi bit len led trai

lb $t6,0($sp) #load bit hien thi led tu stack

add $sp,$sp,-4

lb $t8,0($sp) #load gia tri cua bit nay

add $sp,$sp,-4

add $s2,$t8,$zero #s2 = gia tri bit led trai

sb $t6,0($t0) # hien thi len led trai

loadtorightled: # ham hien thi bit len led phai

sb $t4,0($t5) # hien thi bit len led phai

add $sp,$sp,4

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

add $sp,$sp,4

sb $t4,0($sp) #day bit nay vao stack

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

j finish

resetled:

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

li $t8,0

addi $sp,$sp,4

sb $t8,0($sp)

lb $t6,zero # day bit zero vao stack

addi $sp,$sp,4

sb $t6,0($sp)

finish:

j end\_exception

nop

end\_exception:

# return to start of the loop instead of where the interrupt occur, since the loop doesn't do meaningful thing

la $a3, loop1

mtc0 $a3, $14

eret