ĐẠI HỌC BÁCH KHOA HÀ NỘI

TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

****

BÁO CÁO

FINAL-PROJECT CUỐI KÌ

Học phần: Thực hành kiến trúc máy tính

Giảng viên hướng dẫn: Phạm Ngọc Hưng

Sinh viên thực hiện:

Dương Văn Giới- 20215041

Mã lớp: 139365

Contents

1.Đề tài thực hiện 3

Project 10 – Máy tính bỏ túi 3

2.Thực hiện: 4

2.1 Công cụ thực hiện : 4

2.2 Cách thực hiện : 4

2.2.1. Yêu cầu cho chương trình: 4

2.2.2. Giải quyết: 4

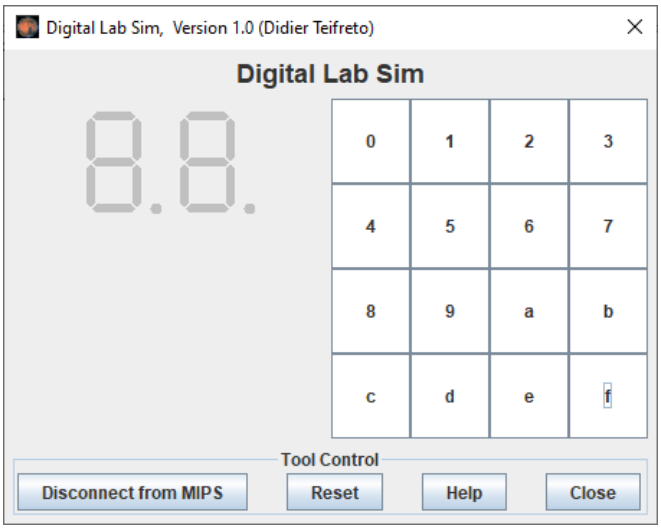
2.3 Thực hiện chương trình: 6

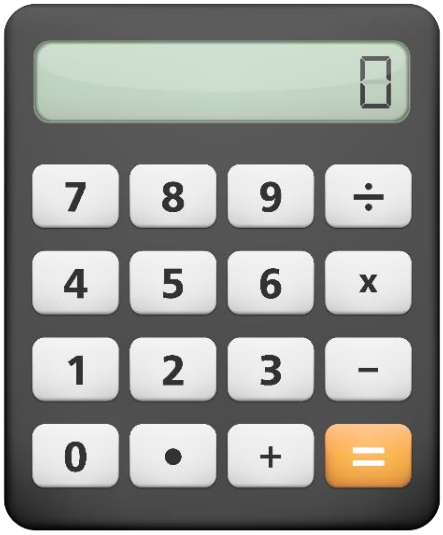
A. Các thanh ghi và nhãn trong chương trình 6

B.Các hàm chính 8

C.Chạy chương trình: 13

D.Mã Nguồn: 14





1.Đề tài thực hiện

Project 10 – Máy tính bỏ túi

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

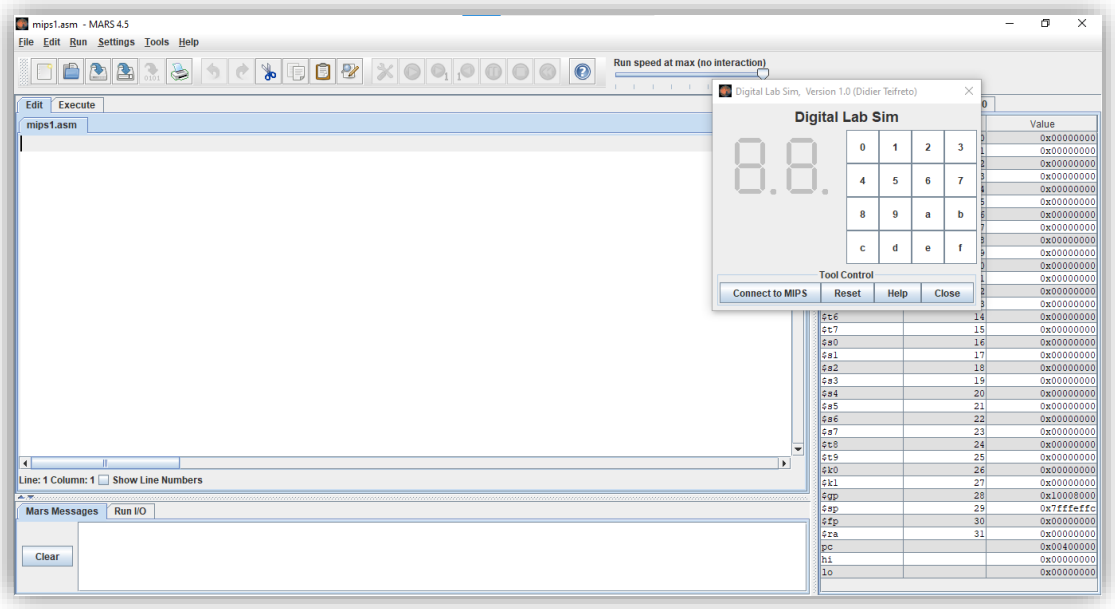
- Bấm phím a để nhập phép tính +
- Bấm phím b để nhập phép tính –
- Bấm phím c để nhập phép tính *
- Bấm phím d để nhập phép tính /
- Bấm phím f để nhập phép = Yêu cầu cụ thể như sau:
- Khi nhấn các phím số, hiển thị lên LED, do chỉ có 2 LED nên chỉ hiện thị 2 số cuối cùng. Ví dụ khi nhấn phím 1 à hiện 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.

2.Thực hiện:

2.1 Công cụ thực hiện :

Mars 4.5



2.2 Cách thực hiện :

2.2.1. Yêu cầu cho chương trình:

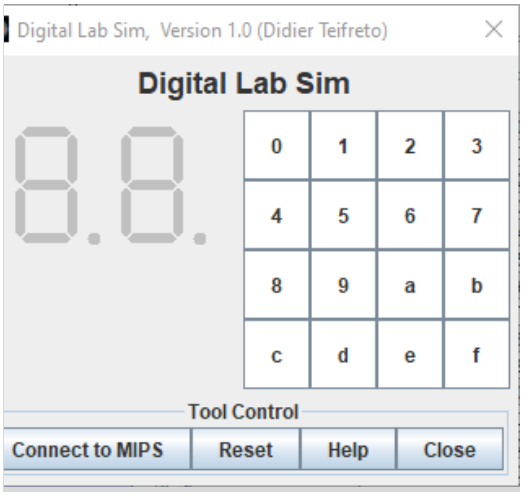
- Sử dụng Digital Lab Sim để tương tác với chương trình.
- Hỗ trợ các toán tử +-*/= (abcdf) (e có thể sử dụng làm phí xóa tất cả tương tự như phím AC).
- Các số và kết quả sẽ được hiển thị trên 2 Led 7 đoạn.
- Chi tiết về biểu thức sẽ được in ra phần RUN I/O.

2.2.2. Giải quyết:

- Làm sao để nhập được phím từ Digital lab sim?:

Chương trình chính sẽ có một vòng lặp loop để thực hiện chờ phím được nhập, khi phím được nhập thì chương trình sẽ nhảy đến chương trình con xử lý ngắt. Trong chương trình con xử lý ngắt sẽ có một vòng lặp để quét các hàng phím, kiểm tra xem vị trí của ký tự nhập vào. Kết quả sẽ trả về mã của phím14, nếu mã phím là 0x48 thì cho kết thúc chương trình vì đây là phím e(end) ta đã quy định là phím tắt .

- Sau khi đã có mã của phím rồi thì làm gì tiếp theo?:
  -Sau khi có mã phím nó sẽ nhảy đến vùng làm việc trong hàm main. Chương trình sẽ không thể thực hiện việc tính toán và hiển thị trên led bằng mã phím kia được, lúc đó ta cần tiền xử lý trước khi thực hiện tính toán và in lên trên led.
  -Tiền xử lý: thực hiện chuyển sang hai dạng số và mã led (đối với a,b,c,d,e,f) thì không cần. Chuyển sang dạng số từ 0-14 (10,11,12,13,14 ứng với +-*/=), đồng thời lưu lại giá trị mã led $t9.



	0x1 0	0x2 0	0x4 0	0x8 0
0x01	0	1	2	3
0x02	4	5	6	7
0x04	8	9	a	b
0x08	c	d	e	f

zero	0x11=0111111
one	0X21=0000110
two	0x41=1011011
three	0x81=1001111
four	0X12=1100110
five	0x22=1101100
six	0x42=1111101
seven	0x82=0000111
eight	0x14=1111111
nine	0x24=1101111
add	0x44
sub	0xffffff84
mul	0x18
div	0x28
equal	0xffffff88

1. Bảng giá chuyển đổi mã phím

- Sau khi chuyển đổi xong thì làm gì? : (Thực hiện kiểm tra và nhập vào phép toán)
  Sau khi đã chuyển đổi các phím rồi thì ta cần kiểm tra xem vị trí nhập hiện tại là ở đâu:
  -Chưa nhập gì:
  +) Nếu là số thì ta thực hiện nhập vào số thứ nhất.
  +) Nếu là dấu = thì ta thực hiện tính toán và kết quả sẽ là 0+0=0 và nhảy sang phép toán sau.
  +) Nếu là toán tử thì ta cho báo lỗi syntax error! Nhảy sang phép toán tiếp theo.
  -Đang ở số thứ nhất($t2=1):
  +) Nếu là số thì ta thực hiện nhập tiếp vào số thứ nhất.
  +) Nếu là dấu = thì ta thực hiện tính toán và kết quả sẽ là s1+0=0 và nhảy sang phép toán sau.

+) Nếu là toán tử thì ta cập nhật lại vị trí nhập vào giờ sẽ là ở số thứ 2 đồng thời lưu lại giá trị của toán tử vào $s4 để thực hiện phép tính ở phía sau.

-Đang ở số thứ hai($t2=2):

+) Nếu là số thì ta thực hiện nhập tiếp vào số thứ hai.
+) Nếu là dấu = thì ta thực hiện kiểm tra xem kí tự trước đó có phải là toán tử hay không, nếu phải thì ta sẽ báo lỗi syntax error! Còn không thì sẽ nhảy đến tính toán và in ra.
+) Nếu là toán tử thì ta sẽ báo lỗi syntax error! luôn vì vị trí nhập đang là 2 nên không nhập thêm toán tử nữa (1+1+ error)

- Nhập vào số thứ, nhập vào số thứ hai: số mới = số cũ *10+số mới nhập

- Tính toán:

  Check xem toán tử đã được lưu ở $s4 là gì, sau đó thực hiện phép toán tương ứng và lưu kết quả vào $s5. Riêng đối với phép chia thì sẽ có thêm thanh ghi $t7 để lưu lại giá trị của phần dư.

- Hiển thị kết quả: kết quả sẽ được hiển thị ở trên led và mục Run I/O của mars45

2.3 Thực hiện chương trình:

A. Các thanh ghi và nhãn trong chương trình

```
.eqv IN_ADRESS_HEXA_KEYBOARD    0xFFFF0012
.eqv OUT_ADRESS_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
.data
error1:     .asciiz     "Syntax ERROR \n \[OK]: Cancel "
error2:     .asciiz     "Math ERROR \n \[OK]: Cancel "
daucong:    .asciiz     "+"
dautru:     .asciiz     "-"
daunhan:    .asciiz     "*"
dauchia:    .asciiz     "/"
daubang:    .asciiz     "="
mess1:      .asciiz     " , Du:"
enter:      .asciiz     "\n"


#Chuyen doi cod qua chuoi 7 bit hien thi tren led
```

```
zero:  .byte 0x3f   #0x11=0=0111111
one:   .byte 0x6    #0X21=1=0000110
two:   .byte 0x5b   #0x41=2=1011011
three: .byte 0x4f   #0x81=3=1001111
four:  .byte 0x66   #0X12=4=1100110
five:  .byte 0x6d   #0x22=5=1101100
six:   .byte 0x7d   #0x42=6=1111101
seven: .byte 0x7    #0x82=7=0000111
eight: .byte 0x7f   #0x14=8=1111111
nine:  .byte 0x6f   #0x24=9=1101111
```

Các thanh ghi được sử dụng :

$s0	Mã led trái
$s1	Mã led phải
$s2	Số thứ nhất
$s3	Số thứ hai
$s4	Toán tử
$s5	Kết quả
$s7	Mã led phải cũ
$t0	Phân loại phím số (0) ,tóan tử (1), end (2)
$t1	Lưu loại phím trước là số (0), toán tử (1)
$t2	Vị trí đang nhập vào hiện tại
$t3	Giá trị số ở led trái
$t4	Giá trị số ở led phải
$t5	Thanh ghi dùng để chứa các giá trị tức thời
$t6	Thanh ghi dùng để làm biến thực hiện trong hàm convert2,3
$s6	Thanh ghi lưu giá trị trả về trong hàm convert3
$t7	Lưu giá trị phần dư trong phép chia
$t8	Giá trị của phím vừa nhập vào 0-14
$t9	Mã phím

Ngoài ra còn một số thanh ghi khác nữa...

B.Các hàm chính

MAIN:

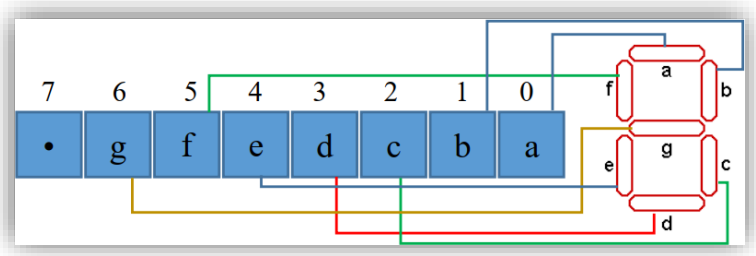
```
MAIN:   jal     SET                 #khai bao cac thanh ghi
        s2:     li  $s1,0x3f        #khi nhap s2 thi bat dau reset led trai ve 0 (tuc
                                      led phai cu ve 0)
        loop:   nop
        nop
        b   loop
  begin:
    #nhap du lieu va xu ly
        beq     $t0,2,endbegin      #neu phim e duoc an thi ket thuc chuong trinh
        nop
    f1: jal     CONVERT1            #chuyen ki tu vua nhap ve dang so 0,1,2,3,4,5
                                     6,7,8,9,10,11,12,13,14
        nop
    sfx:jal     CHECK               #Kiem tra du lieu va nhay den mot trong nhung ham
                                      f2,f3,f4,f5,f6
        nop
    f2: jal     Math_ERROR          #Quay Lai ham MAIN tinh bieu thuc moi
        nop
    f3: jal     Syntax_ERROR        #Quay Lai ham MAIN tinh bieu thuc moi
        nop
    f4: jal     ThemVaoS1           #Them so vua nhap vao so thu nhat s2=s2*10+$t8
        nop
        jal     SHOWLED             #In ra Led 7 thanh
        nop
        b       loop
    f5: jal     ThemVaoS2           #Them so vua nhap vao so thu hai s2=s3*10+$t8
        nop
        jal     SHOWLED             #In ra Led 7 thanh
        nop
        b       loop
    f6:     jal     TinhToan        #Thuc hien viec tinh +-*/
        nop

    #thuc hien xu ly ket qua
    f7: jal     CONVERT2        #chuyen doi so thanh 2 so hang chuc va don vi
        nop
        add $t6,$t3,$zero       #chuyen doi so thanh ma hien thi tren led 7 thanh
        jal     CONVERT3        #input $t6 1-9; output $s6 ma hien thi
        add $s0,$s6,$zero
        nop
        add $t6,$t4,$zero       #chuyen doi so thanh ma hien thi tren led 7 thanh
        jal     CONVERT3        #input $t6 1-9; output $s6 ma hien thi
        add $s1,$s6,$zero
```

```
        nop
    f8: jal SHOWLED         #In kq ra Led 7 thanh
        nop
        jal     PRINTCALC       #In ra chi tiet phep tinh
        nop
        b   MAIN            #nhay den phep toan tiep theo
   endbegin:
END:
        li  $v0, 10
        syscall
```

Thông tin các hàm:

Hàm convert 1:

Chuyển đổi mã phím về dạng số từ 0-14 (10,11,12,13,14 ứng với +-*/ =) và về dạng mã led



Hàm convert 2:

Trả về hàng chục và đơn vị của một số,lưu vào hai giá trị

Hàm convert3:

Trả về $s6 chứa mã led của một số được cho vào $t6.

Hàm Check:

Hàm kiểm tra và điều khiển con trỏ Pc nhảy đến hàm tiếp theo dựa vào giá trị của phím, vị trí nhập, phím trước đó...

```
   GOTO_F2:#Math_ERROR
        la  $ra,f2
        jr  $ra
    GOTO_F3:#Syntax_ERROR
        la  $ra,f3
        jr  $ra
    GOTO_F4:#ThemVaoS1
        li  $t2,1
        la  $ra,f4
```

```
        jr  $ra
    GOTO_F5:#ThemVaoS2
        la  $ra,f5
        jr  $ra
    GOTO_F6:#TinhToan
        la  $ra,f6
        jr  $ra
    GOTO_S2:#chuyển đến nhập só thứ hai
        la  $ra,s2
        jr  $ra
```

Hàm Math ERROR và Hàm Syntax ERROR:

Đưa ra thông báo lỗi đồng thời nhảy đến phép toán tiếp theo

```
      la    $ra,MAIN
      jr     $ra
```

Hàm Themvaos1:

Thêm vào số thứ nhất

```
ThemVaoS1:
    add     $s0,$s7,$zero   #dich chuyen led phai sang trai
    add $t5,$s2,$s2 #t5=2s2
    sll $s2,$s2,3   #s2(+)=8s2
    add $s2,$s2,$t5 #s2(+)=10s2
    add $s2,$s2,$t8 #s2(+)=10s2+t8(so moi them vao)
    add $t6,$s2,$zero   #t6=s2  convert2(t6)
    jr  $ra
```

Hàm Themvaos2:

Thêm vào số thứ hai

```
ThemVaoS2:
    add     $s0,$s7,$zero   #dich chuyen led phai sang trai
    add $t5,$s3,$s3 #t5=2s3
    sll $s3,$s3,3   #s3(+)=8s3
    add $s3,$s3,$t5 #s3(+)=10s3
    add $s3,$s3,$t8 #s3(+)=10s3+t8(so moi them vao)
    add $t6,$s3,$zero   #t6=s3  convert2(t6)
    jr  $ra
```

Hàm TinhToan:

Thực hiện kiểm tra $s4 và tính toán

```
TinhToan:
    beq     $s4,10,cong
    beq     $s4,11,tru
    beq     $s4,12,nhan
    beq     $s4,13,chia
    cong:
        add     $s5,$s2,$s3     #s5=so1 + so 2
        j   finish
    tru:
        sub $s5,$s2,$s3 #s5=s01 - so2
        j   finish
        nop
    nhan:
        mul $s5,$s2,$s3
        j   finish
        nop
    chia:   beq $s3,$zero,div_error
        div     $s2,$s3
        mflo    $s5
        mfhi    $t7
        j   finish
        nop
        div_error:
        la  $ra,f2          #Math error
        jr  $ra
        nop
    finish:
        add $t6,$s5,$zero   #t6=s2  convert2(t6)
        jr  $ra
```

Hàm ShowLed:

Chỉ đơn giản là hiển thị led thôi, mã led trái $s0, phải $s1

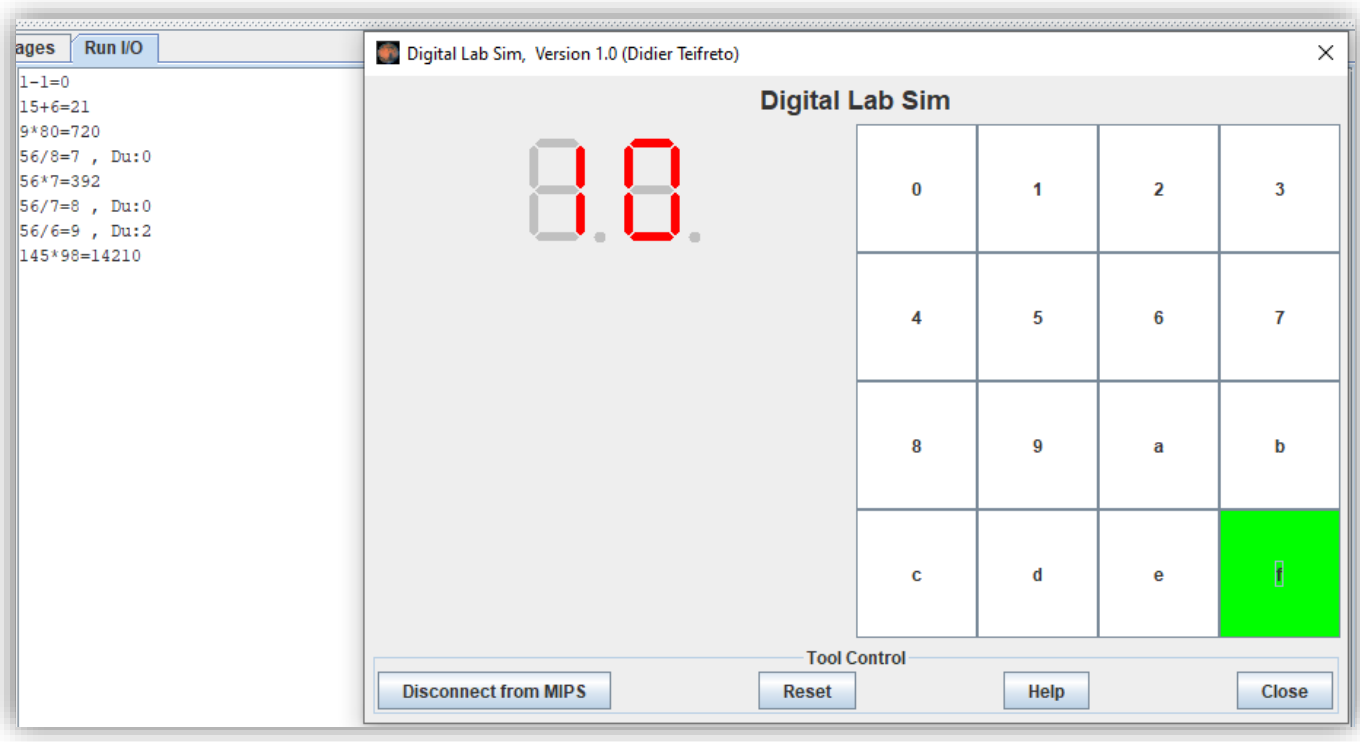
```
SHOWLED:
    show_7seg_left:
        li  $k0, SEVENSEG_LEFT  # assign port's address
        sb  $s0, 0($k0)         # assign new value
        nop
    show_7seg_right:
        li  $k0, SEVENSEG_RIGHT     # assign port's address
        sb  $s1, 0($k0)         # assign new value
        nop
        jr  $ra
        nop
```

Hàm PrintCalc:

In chi tiết thông tin của phép toán.

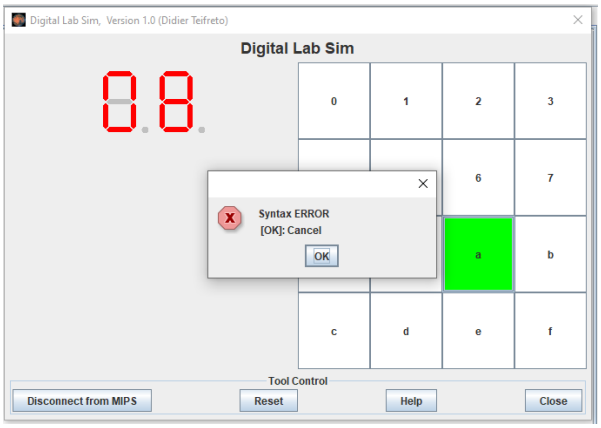
C.Chạy chương trình:

Với các phép toán hợp lệ:

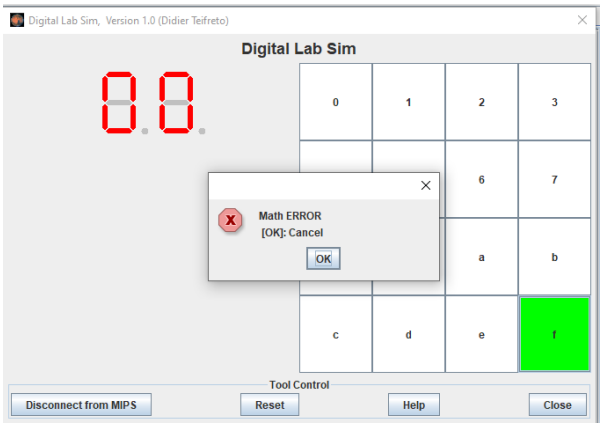


Khi lỗi:

8++ (Syntax ERROR)



1/0 (Math ERROR)



D.Mã Nguồn:

```
.eqv IN_ADRESS_HEXA_KEYBOARD    0xFFFF0012
.eqv OUT_ADRESS_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
# Bit 0 = doan a;
# Bit 1 = doan b;
# Bit 2 = doan c;
# Bit 3 = doan d;
# Bit 4 = doan e;
# Bit 5 = doan f;
# Bit 6 = doan g;
# Bit 7 = dau .
.data
error1:     .asciiz     "Syntax ERROR \n \[OK]: Cancel "
error2:     .asciiz     "Math ERROR \n \[OK]: Cancel "
daucong:    .asciiz     "+"
dautru:     .asciiz     "-"
daunhan:    .asciiz     "*"
dauchia:    .asciiz     "/"
daubang:    .asciiz     "="
mess1:      .asciiz     " , Du:"
enter:      .asciiz     "\n"
#Chuyen doi cod qua chuoi 7 bit hien thi tren led
zero:  .byte 0x3f   #0x11=0=0111111
one:   .byte 0x6    #0X21=1=0000110
two:   .byte 0x5b   #0x41=2=1011011
three: .byte 0x4f   #0x81=3=1001111
four:  .byte 0x66   #0X12=4=1100110
five:  .byte 0x6d   #0x22=5=1101100
six:   .byte 0x7d   #0x42=6=1111101
seven: .byte 0x7    #0x82=7=0000111
eight: .byte 0x7f   #0x14=8=1111111
nine:  .byte 0x6f   #0x24=9=1101111
.text
#=============================================================================
MAIN:       jal     SET             #khai bao cac thanh ghi
            s2: li  $s1,0x3f        #khi nhap s2 thi bat dau reset led trai ve 0
                                    (tuc led phai cu ve 0)
    loop:   nop
            nop
        b   loop
    begin:
    #nhap du lieu va xu ly
        beq     $t0,2,endbegin      #neu phim e duoc an thi ket thuc chuong trinh
        nop
```

```
    f1: jal CONVERT1                #chuyen ki tu vua nhap ve dang so
                                    0,1,2,3,4,5,6,7,8,9,10,11,12,13,14
        nop
    sfx:jal    CHECK                #Kiem tra du lieu va nhay den mot trong nhung
        nop                         ham f2,f3,f4,f5,f6

    f2: jal Math_ERROR              #Quay Lai ham MAIN tinh bieu thuc moi
        nop
    f3: jal Syntax_ERROR            #Quay Lai ham MAIN tinh bieu thuc moi
        nop
    f4: jal ThemVaoS1               #Them so vua nhap vao so thu nhat
                                    s2=s2*10+$t8
        nop
        jal SHOWLED                 #In ra Led 7 thanh
        nop
        b   loop
    f5: jal ThemVaoS2               #Them so vua nhap vao so thu hai s2=s3*10+$t8
        nop
        jal SHOWLED                 #In ra Led 7 thanh
        nop
        b   loop
    f6:     jal     TinhToan        #Thuc hien viec tinh +-*/
        nop

    #thuc hien xu ly ket qua
    f7: jal     CONVERT2        #chuyen doi so thanh 2 so hang chuc va don vi
        nop
        add $t6,$t3,$zero       #chuyen doi so thanh ma hien thi tren led 7 thanh
        jal     CONVERT3        #input $t6 1-9; output $s6 ma hien thi
        add $s0,$s6,$zero
        nop
        add $t6,$t4,$zero       #chuyen doi so thanh ma hien thi tren led 7 thanh
        jal     CONVERT3        #input $t6 1-9; output $s6 ma hien thi
        add $s1,$s6,$zero
        nop
    f8: jal SHOWLED         #In kq ra Led 7 thanh
        nop
        jal     PRINTCALC       #In ra chi tiet phep tinh
        nop
        b   MAIN            #nhay den phep toan tiep theo
   endbegin:
END:
        li  $v0, 10
        syscall
#================================================================================
#KHAI BAO CAC THANH GHI SU DUNG TRONG CHUONG TRINH
SET:
    # Enable the interrupt of Keyboard matrix 4x4 of Digital LabSim
```

```
    li  $t1, IN_ADRESS_HEXA_KEYBOARD
    li  $t3, 0x80
    sb  $t3, 0($t1)
    #khai bao cac thanh ghi
    li  $s0,0x3f#so hien thi tren led trai
    li  $s1,0x3f#so hien thi tren led phai
    li  $s2,0   #so thu nhat
    li  $s3,0   #so thu hai
    li  $s4,10  #toan tu
    li  $s5,0   #ket qua
    li  $s7,0   #gia tri led phai cu
    li  $t0,0   #kiem tra phim nhap vao la so (0) ,toan tu (1), end (2)
    li  $t1,-1  #kiem tra phim truoc la so (0), toan tu (1)
    li  $t2,0   #vi tri dang nhap lieu hien tai la chua nhap gi (0) ,so thu nhat
(1), so thu hai (2)
    li  $t3,0   #gia tri so o led trai
    li  $t4,0   #gia tri so o led phai
    li  $t5,0   #thanh ghi chua cac gia tri tuc thoi
    li  $t6,0   #thanh ghi luu gia tri bien x trong ham convert2,convert3
    li  $s6,0   #thanh ghi luu gia tra ve trong ham convert3
    li  $t7,0   #luu gia tri phan du trong phep chia
    li  $t8,0   #gia tri cua phim vua nhap vao
0,1,2,3,4,5,6,7,8,9,10(+),11(-),12(*),13(/),14(=)
    li  $t9,0   #gia tri cua code nhap tu ban phim
    jr  $ra
#---------------------------------------------------------------
#---------------------------------------------------------------
#FUNCTION 1     #FUNCTION 1         #FUNCTION 1
CONVERT1:   add $t1,$t0,$zero       #luu lai loai phim cua phim truoc
        add $s7,$s1,$zero       #luu lai gia tri led phai cu
        li  $t0,0           #neu la toan hang
        add $a0,$t9,$zero       #lay code tu $t9
        ori $t5,$zero,0x11
        beq     $a0,$t5,case_zero
        ori $t5,$zero,0x21
        beq     $a0,$t5,case_one
        ori $t5,$zero,0x41
        beq     $a0,$t5,case_two
        li  $t5,0xffffff81
        beq     $a0,$t5,case_three
        ori $t5,$zero,0x12
        beq     $a0,$t5,case_four
        ori $t5,$zero,0x22
        beq     $a0,$t5,case_five
        ori $t5,$zero,0x42
        beq     $a0,$t5,case_six
        li  $t5,0xffffff82
        beq     $a0,$t5,case_seven
```

```
        ori $t5,$zero,0x14
        beq     $a0,$t5,case_eight
        ori $t5,$zero,0x24
        beq     $a0,$t5,case_nine
        li  $t0,1           #neu la toan tu
        ori $t5,$zero,0x44
        beq     $a0,$t5,case_add
        li  $t5,0xffffff84
        beq     $a0,$t5,case_sub
        ori $t5,$zero,0x18
        beq     $a0,$t5,case_mul
        ori $t5,$zero,0x28
        beq     $a0,$t5,case_div
        li  $t5,0xffffff88
        beq     $a0,$t5,case_equ
        li  $t0,2           #neu la phim tat
        ori $t5,$zero,0x11
        beq     $a0,0x00000048,return

    case_zero:
        lb  $s1,zero        #cap nhat led phai
        li  $t8,0           #ki tu vua nhap
        j   update1
    case_one:
        lb  $s1,one         #cap nhat led phai
        li  $t8,1           #ki tu vua nhap
        j   update1
    case_two:
        lb  $s1,two         #cap nhat led phai
        li  $t8,2           #ki tu vua nhap
        j   update1
    case_three:
        lb  $s1,three       #cap nhat led phai
        li  $t8,3           #ki tu vua nhap
        j   update1
    case_four:
        lb  $s1,four        #cap nhat led phai
        li  $t8,4           #ki tu vua nhap
        j   update1
    case_five:
        lb  $s1,five        #cap nhat led phai
        li  $t8,5           #ki tu vua nhap
        j   update1
    case_six:
        lb  $s1,six         #cap nhat led phai
        li  $t8,6           #ki tu vua nhap
        j   update1
    case_seven:
```

```
        lb  $s1,seven       #cap nhat led phai
        li  $t8,7           #ki tu vua nhap
        j   update1
    case_eight:
        lb  $s1,eight       #cap nhat led phai
        li  $t8,8           #ki tu vua nhap
        j   update1
    case_nine:
        lb  $s1,nine        #cap nhat led phai
        li  $t8,9           #ki tu vua nhap
        j   update1
    case_add:
        li  $t8,10          #ki tu vua nhap
        nop
        jr  $ra
        nop
    case_sub:
        li  $t8,11          #ki tu vua nhap
        nop
        jr  $ra
        nop
    case_mul:
        li  $t8,12          #ki tu vua nhap
        nop
        jr  $ra
        nop
    case_div:
        li  $t8,13          #ki tu vua nhap
        nop
        jr  $ra
        nop
    case_equ:
        li  $t8,14          #ki tu vua nhap
        nop
        jr  $ra
        nop
    end_cv:
    update1:
        add $s0,$s7,0
        nop
        jr  $ra
        nop
#---------------------------------------------------------------
#FUNCTION 7     #FUNCTION 7     #FUNCTION 7
CONVERT2:
    li  $t3,0   #gia tri so o led trai
    li  $t4,0   #gia tri so o led phai
    li  $t5,10
```

```
    div $t6,$t5
    mflo    $t6
    mfhi    $t4 #led ben trai o chu so hang don vi
    div $t6,$t5
    mflo    $t6
    mfhi    $t3 #led ben phai o chu so hang chuc
    jr  $ra
CONVERT3:
        bne     $t6,0,case_1
        lb  $s6,zero
        jr  $ra
    case_1: bne     $t6,1,case_2
        lb  $s6,one
        jr  $ra
    case_2: bne     $t6,2,case_3
        lb  $s6,two
        jr  $ra
    case_3: bne     $t6,3,case_4
        lb  $s6,three
        jr  $ra
    case_4: bne     $t6,4,case_5
        lb  $s6,four
        jr  $ra
    case_5: bne     $t6,5,case_6
        lb  $s6,five
        jr  $ra
    case_6: bne     $t6,6,case_7
        lb  $s6,six
        jr  $ra
    case_7: bne     $t6,7,case_8
        lb  $s6,seven
        jr  $ra
    case_8: bne $t6,8,case_9
        lb  $s6,eight
        jr  $ra
    case_9: lb  $s6,nine
        jr  $ra
#-----------------------------------------------------------------
#GOTO FUNCTION x    #GOTO FUNCTION x    #GOTO FUNCTION x
CHECK:
    beq $t2,1,khu1
    beq $t2,2,khu2
    khu0: #kiem tra vi tri nhap 0
        beq $t0,$zero,GOTO_F4
        beq $t8,14,GOTO_F6
        j   GOTO_F3
    khu1: #kiem tra vi tri nhap 1
        beq $t0,$zero,GOTO_F4   #kiem tra loai phim nhap vao la so
```

```
        beq $t8,14,GOTO_F6      #phim nhap vao la dau =
        add $s4,$t8,0       #neu la toan tu thi luu lai
        li  $t2,2
        j   GOTO_S2

    khu2:#kiem tra vi tri nhap 2
        beq $t0,$zero,GOTO_F5   #kiem tra loai phim nhap vao la so,neu co thi nhap
vao s2
        nop
        ori $t5,$zero,14
        bne $t8,$t5,khu2_1      #kiem tra co phai dau bang hay khong
        beq $t1,$zero,GOTO_F6   #neu la dau bang thi kiem tra ki tu truoc do la toan
tu hay khong,neu co thi tinh toa
    khu2_1:
        nop
        ori     $t5,$zero,1
        beq $t1,$t5,GOTO_F3     #phim hien tai la toan tu, neu phim truoc do la toan
tu-> error
        j   GOTO_F6
    GOTO_F2:#Math_ERROR
        la  $ra,f2
        jr  $ra
    GOTO_F3:#Syntax_ERROR
        la  $ra,f3
        jr  $ra
    GOTO_F4:#ThemVaoS1
        li  $t2,1
        la  $ra,f4
        jr  $ra
    GOTO_F5:#ThemVaoS2
        la  $ra,f5
        jr  $ra
    GOTO_F6:#TinhToan
        la  $ra,f6
        jr  $ra
    GOTO_S2:
        la  $ra,s2
        jr  $ra


#-----------------------------------------------------------------
#FUNCTION 2,3       #FUNCTION 2,3       #FUNCTION 2,3
#BAO LOI !!!
Syntax_ERROR:
        li  $v0, 55
        la  $a0, error1
        li  $a1, 0
        syscall
```

```
        la  $ra,MAIN
        jr  $ra

Math_ERROR: li  $v0, 55
        la  $a0, error2
        li  $a1, 0
        syscall
        la  $ra,MAIN
        jr  $ra
#---------------------------------------------------------------
#---------------------------------------------------------------

ThemVaoS1:
    add     $s0,$s7,$zero   #dich chuyen led phai sang trai
    add $t5,$s2,$s2 #t5=2s2
    sll $s2,$s2,3   #s2(+)=8s2
    add $s2,$s2,$t5 #s2(+)=10s2
    add $s2,$s2,$t8 #s2(+)=10s2+t8(so moi them vao)
    add $t6,$s2,$zero   #t6=s2  convert2(t6)
    jr  $ra
ThemVaoS2:
    add     $s0,$s7,$zero   #dich chuyen led phai sang trai
    add $t5,$s3,$s3 #t5=2s3
    sll $s3,$s3,3   #s3(+)=8s3
    add $s3,$s3,$t5 #s3(+)=10s3
    add $s3,$s3,$t8 #s3(+)=10s3+t8(so moi them vao)
    add $t6,$s3,$zero   #t6=s3  convert2(t6)
    jr  $ra
TinhToan:
    beq     $s4,10,cong
    beq $s4,11,tru
    beq $s4,12,nhan
    beq $s4,13,chia
    cong:
        add     $s5,$s2,$s3     #s5=so1 + so 2
        j   finish
    tru:
        sub $s5,$s2,$s3 #s5=s01 - so2
        j   finish
        nop
    nhan:
        mul $s5,$s2,$s3
        j   finish
        nop
    chia:   beq $s3,$zero,div_error
        div     $s2,$s3
        mflo    $s5
        mfhi    $t7
```

```
        j   finish
        nop
        div_error:
        la  $ra,f2          #Math error
        jr  $ra
        nop
    finish:
        add $t6,$s5,$zero   #t6=s2  convert2(t6)
        jr  $ra
SHOWLED:
    show_7seg_left:
        li  $k0, SEVENSEG_LEFT  # assign port's address
        sb  $s0, 0($k0)         # assign new value
        nop
    show_7seg_right:
        li  $k0, SEVENSEG_RIGHT     # assign port's address
        sb  $s1, 0($k0)         # assign new value
        nop
        jr  $ra
        nop
PRINTCALC:
    beq     $s4,10,incong
    beq $s4,11,intru
    beq $s4,12,innhan
    beq $s4,13,inchia
   incong:
    li  $v0, 1
    move    $a0, $s2
    syscall
    li  $v0, 11
    li  $a0, '+'
    syscall
    li  $v0, 1
    move    $a0,$s3
    syscall
    li  $v0, 11
    li  $a0, '='
    syscall
    li  $v0, 1
    move    $a0, $s5
    syscall
    li  $v0, 11
    li  $a0, '\n'
    syscall
    jr  $ra
    intru:
    li  $v0, 1
    move    $a0, $s2
```

```
    syscall
    li  $v0, 11
    li  $a0, '-'
    syscall
    li  $v0, 1
    move    $a0,$s3
    syscall
    li  $v0, 11
    li  $a0, '='
    syscall
    li  $v0, 1
    move    $a0, $s5
    syscall
    nop
    li  $v0, 11
    li  $a0, '\n'
    syscall
    jr  $ra
    innhan:
    li  $v0, 1
    move    $a0, $s2
    syscall
    li  $v0, 11
    li  $a0, '*'
    syscall
    li  $v0, 1
    move    $a0,$s3
    syscall
    li  $v0, 11
    li  $a0, '='
    syscall
    li  $v0, 1
    move    $a0, $s5
    syscall
    nop
    li  $v0, 11
    li  $a0, '\n'
    syscall
    jr  $ra
   inchia:
    li  $v0, 1
    move    $a0, $s2
    syscall
    li  $v0, 11
    li  $a0, '/'
    syscall
    li  $v0, 1
    move    $a0,$s3
```

```
    syscall
    li  $v0, 11
    li  $a0, '='
    syscall
    li  $v0, 1
    move    $a0, $s5
    syscall
    nop
    la  $a0,mess1
        li  $v0, 4
        syscall
        li  $v0, 1
    move    $a0, $t7
    syscall
    li  $v0, 11
    li  $a0, '\n'
    syscall
    jr  $ra
#================================================================================
========
# XU LY NGAT
.ktext 0x80000180
    # SAVE the current REG FILE to stack
    IntSR:  addi    $sp,$sp,4       # Save $ra because we may change it later
        sw  $ra,0($sp)
        addi    $sp,$sp,4       # Save $ra because we may change it later
        sw  $at,0($sp)
    # Processing
    get_cod:li  $k0, IN_ADRESS_HEXA_KEYBOARD
        li  $k1, OUT_ADRESS_HEXA_KEYBOARD
        li  $a2, 0x01
    get_cod_loop:               #lap va kiem tra ma cua phim
        addi    $a1,$a2,0x80
        sll     $a2,$a2,1
        sb  $a1,0($k0)      # must reassign expected row
        lb  $a0,0($k1)
        beq     $a0,$zero,get_cod_loop
        add $t9,$a0,0       #luu code vao $t9

return:     la  $a3,begin
        mtc0    $a3, $14
        eret # Return from exception
```