BÁO CÁO CUỐI KÌ

MÔN HỌC : THỰC HÀNH KIẾN TRÚC MÁY TÍNH

MÃ LỚP : 139165

Giáo viên hướng dẫn: Phạm Ngọc Hưng

Họ và tên sinh viên: Nguyễn Văn Thái MSSV: 20215135

Chủ đề 10: Máy tính bỏ túi

Nội dung Project:

1. Demo chương trình.
2. Source code.
3. Thực hiện chạy chương trình với MARS.
4. Giải thích.

1. Demo chương trình

- 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 bđể nhập phép tính –
- Bấm phím c để nhập phép tính *
- Bấm phím d để nhập phép tính /
- Bấm phím f để nhập phép =
- Bấm phím e để thực hiện kết thúc chương trình đang chạy.
  Yêu cầu cụ thểnhư 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 →hiện thị 01. Khi nhấn thêm phím 2 →hiển thị 12. Khi nhấn thêm phím 3 →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.

2. 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 chia cho so 0 \n"

.eqv IN_ADDRESS_HEXA_KEYBOARD  0xFFFF0012
.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:
start:
        li $t0,SEVENSEG_LEFT           # $t0: Bien gia tri so cua den LED trai
        li $t5,SEVENSEG_RIGHT          # $t5: Bien gia tri so cua den LED phai
        li $s0,0                       # bien kiem tra loai bien nhap vao, 0: so, 1 :toan tu, 2: AC 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 hai
        li $s6,0                        # ket qua 2 so, cong ,tru, nhan, chia
        li $t9,0                        # gia tri 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 dung enable keyboard interrupt va enable kiem tra tung hang keyboar
        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 cua 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
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Xu ly khi xay ra interupt
# Hien thi so vua bam len den led 7 doan
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
.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)           # 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)           # luu tru 8 bit bac thap t1 vao t3
        jal getvalue            # get vi tri ( hang va cot ) cua phim duoc nhap neu co
        lw $ra,0($sp)
        addi $sp,$sp,-4
        jr $ra                  #chuyen den cau lenh co dia chi thanh ghi $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 vi 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:                    # 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 = 0x3f, tong cac bit thanh ghi de tao thanh so 0 tren led)
        li $t7,0                #t7= 0
        j updatetmp
case_one:
        lb $t4,one
        li $t7,1
        j updatetmp
case_two:
        lb $t4,two
        li $t7,2
        j updatetmp
case_three:
        lb $t4,three
        li $t7,3
        j updatetmp
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 updatetmp
case_five:
        lb $t4,five
        li $t7,5
        j updatetmp
```

```
case_six:
        lb $t4,six
        li $t7,6
        j updatetmp
case_seven:
        lb $t4,seven
        li $t7,7
        j updatetmp
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 updatetmp
case_nine:
        lb $t4,nine
        li $t7,9
        j updatetmp
case_a: #truong hop phim cong
        addi $a3,$zero,1
        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 $a3,$zero,2
        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 $a3,$zero,3
        addi $s0,$s0,1
        addi $s3,$zero,3
        j setfirstnumber
case_d: #truong hop phim chia
        addi $a3,$zero,4
        addi $s0,$s0,1
        addi $s3,$zero,4
        j setfirstnumber

case_e:  #truong hop terminate key
        addi $s0,$s0,2
        j finish
setfirstnumber:                 # ham tinh so dau tien hien thi tren led trong 2 so
        addi $s4, $t9, 0
        li $t9, 0
        j done
case_f:  #truong hop bam =
        addi $s5, $t9, 0

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
        li $s3,0
        li $t9, 0
        j incong
        nop                     # s6=s5+s4

incong:
        li $v0, 1
        move $a0, $s4
        syscall
        li $s4, 0


        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, 1
        move $a0, $s6
        syscall
        nop
        #li $s4, $s6

        li $v0, 11
        li $a0, '\n'
        syscall
        li $s7,100
        div $s6,$s7         # chia $a6 cho $s7
        mfhi $s6            # chi lay 2 chu so cuoi cua ket qua de in ra led
        j show_result_in_led       # 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
        li $s3,0
        li $t9, 0
        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
```

```
292
293         li $v0, 11
294         li $a0, '\n'
295         syscall
296         j show_result_in_led       # chuyen den ham chia ket qua thanh 2 chu so de hien thi len tung led
297         nop
298 nhan:
299         mul $s6,$s4,$s5     # s6=s4*s5
300         li $s3,0
301         li $t9, 0
302         j innhan
303         nop
304 innhan:
305         li $v0, 1
306         move $a0, $s4
307         syscall
308
309         li $v0, 11
310         li $a0, '*'
311         syscall
312
313         li $v0, 1
314         move $a0, $s5
315         syscall
316
317         li $v0, 11
318         li $a0, '='
319         syscall
320
321         li $v0, 1
322         move $a0, $s6
323         syscall
324
325         li $v0, 11
326         li $a0, '\n'
327         syscall
328         li $s7,100
329         div $s6,$s7
330         mfhi $s6            # chi lay 2 chu so sau cung cua ket qua in ra
331         j show_result_in_led       # chuyen den ham chia ket qua thanh 2 chu so de hien thi len tung led
332         nop
333 chia:
334         beq $s5,0,chia0
335         li $s3,0
336         div $s4,$s5         # s6=s4/s5
337         mflo $s6
338         mfhi $s7
339         li $t9, 0
340         j inchia
341         nop
342 inchia:
343         li $v0, 1
344         move $a0, $s4
345         syscall
346
347         li $v0, 11
348         li $a0, '/'
349         syscall
350
351         li $v0, 1
352         move $a0, $s5
353         syscall
354
355         li $v0, 11
356         li $a0, '='
357         syscall
358
```

```
        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_in_led       # chuyen den ham chia ket qua thanh 2 chu so de hien thi len tung led
        nop
chia0:
        li $v0, 55
        la $a0, mess1
        li $a1, 0
        syscall
        j resetled

show_result_in_led:     #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 doi t7 thanh bit hien thi len led
        #---------
        sb $t4,0($t0)  # 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 $t4,0($sp)        #day bit nay vao stack
        add $s2,$t7,$zero   #s2 = gia tri bit led phai
        #----------
        mfhi $t7        #t7= remainder
        jal convert     #convert t7 thanh bit hien thi len led
        sb $t4,0($t5)  #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 $t4,0($sp)        # day bit nay vao stack
        add $s1,$t7,$zero   # s1 = gia tri bit 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
```

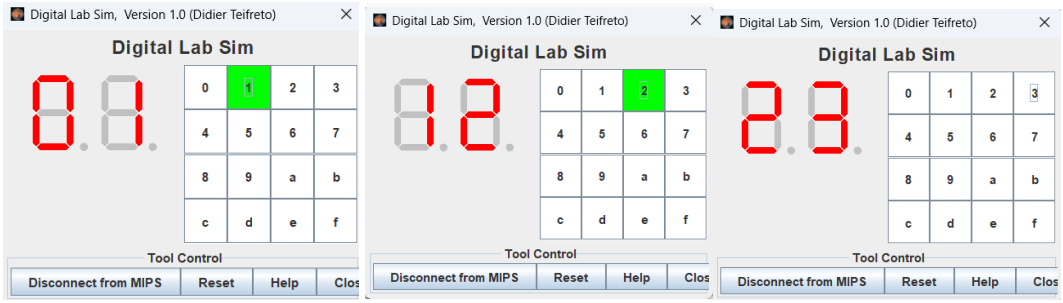
```
        beq $t7,9,case_9
case_0: #ham chuyen 0 thanh bit zero hien thi len led
        lb $t4,zero    #t4=zero
        j finishconvert #ket thuc
case_1:
        lb $t4,one
        j finishconvert
case_2:
        lb $t4,two
        j finishconvert
case_3:
        lb $t4,three
        j finishconvert
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
updatetmp:
         mul $t9, $t9, 10       #st9 thanh bit bac thap của St9 va 10
         add $t9, $t9, $t7
done:
        beq $s0,1,resetled   # s0=1-->toan tu-->chuyen den ham reset led
        nop
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
```

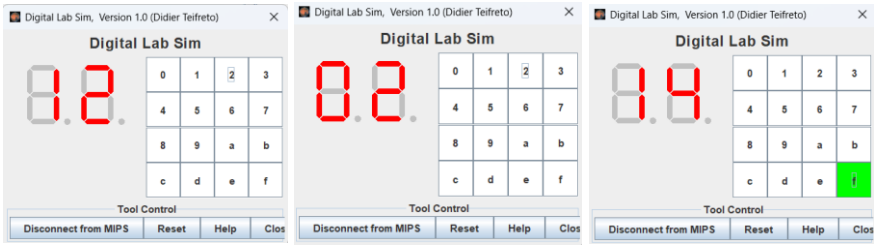
```
492         nop
493 end_exception:
494         la $a3, loop1
495         mtc0 $a3, $14          #dat thanh ghi $18 thanh gia tri duoc luu tru trong $a3
496         eret
```

3. Thực hiện chạy chương trình với MARS

3.1. Test chức năng xử lí số hơn 2 chữ số:



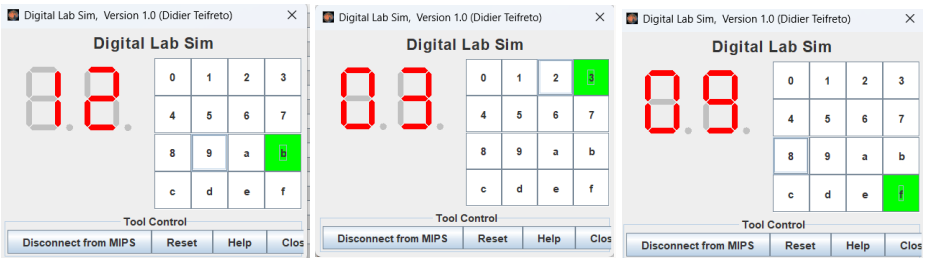
3.2. Chức năng phép cộng:



Kết quả :

```
12+2=14
```

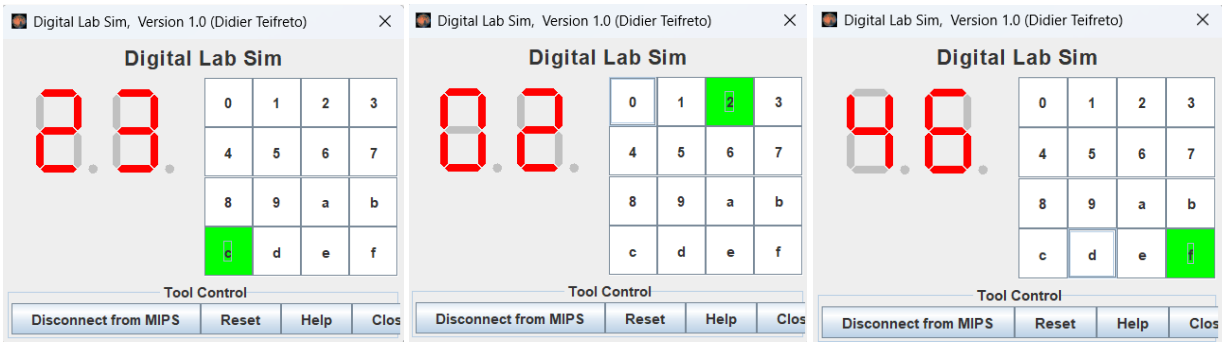
3.3 Phép trừ



Kết quả:

```
12-3=9
```

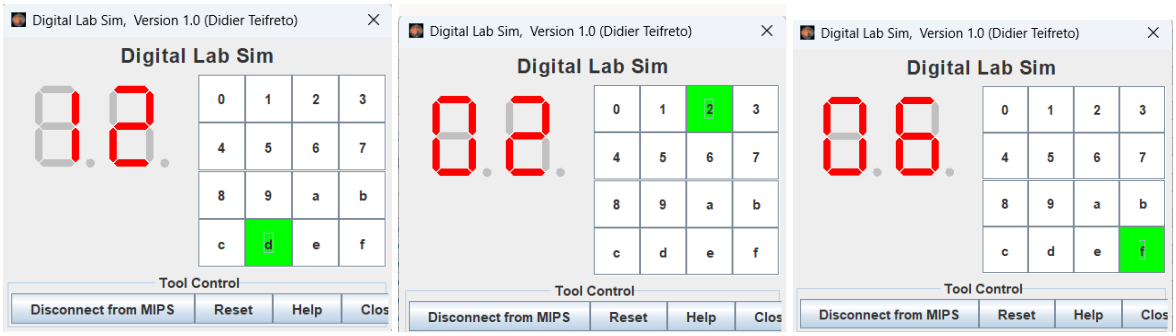
3.4 Phép nhân



Kết quả:

```
123*2=246
```

3.5 Phép chia



Kết quả:

```
12/2=6
```

4. Giải thích code

- Thực hiện khai báo các địa chỉ của đèn led, địa chỉ của các ô phím khi thực hiện ấn,…
- Start: thực hiện khai báo các giá trị cần sử dụng.
- Storefirstvalue: thực hiện gán các giá trị ban đầu bằng không khi chưa ấn gì .
- loop1: có vai trò là vòng lặp vô hạn để đợi cho đến khi có phím được ấn .
- phần end_main: kết thúc chương trình.
- Trong phần loop1 khi mà có phím được ấn thì sẽ thực hiện đi đến dòng lệnh có địa chỉ `68 .ktext 0x80000180`
- process(dòng 69-80): thực hiện đi check từng hàng 1 và kiểm tra xem phím được ấn có ở đó không

- checkrow1, checkrow2,… checkrow4(dòng 81-116): thực hiện kiểm tra các dòng từ địa chỉ của các dòng 0x81, 0x82, 0x84, 0x88. Sau đó thực hiện đi đến getvalue để thực hiện lấy được địa chỉ chính xác của phím vừa nhập. Sau đó thực hiện quay trở lại thực hiện dòng lệnh tiếp theo của ở process convertrow
- các hàm convertrow: có vai trò lấy được địa chỉ để hiển thị ở led ứng với số nhập từ bàn phím và giá trị của số đó. Với các số nhập vào từ 0->9 thì thực hiện tính giá trị của số ở hàm updatetmp. Còn với các phần không phải số là các toán tử +, -, *, / thì thực hiện lưu vào thanh ghi với quy định bản thân chọn. Và đi đến hàm setfirstnumber
- trường hợp là các dấu +, -, *, / thì

+ setfirstnumber để thực hiện tính giá trị số thứ nhất sau đó đi đến hàm done để kiểm tra xem vừa nhập vào là toán tử hay là số, nếu là toán tử thì tiếp tục đi đến hàm resetled để khởi tạo lại các thanh ghi kiểm tra và lưu giá trị địa chỉ của zero vào trong stack để nếu tiếp theo ấn = thì kết quả số thứ 2 mặc định sẽ là 0 mà không phải giá trị vô định.

+ sau khi thực hiện xong thì quay trở lại vòng lặp chờ cho đến khi số thứ 2 được nhập vào

- trường hợp là các số 0->9 thì tính giá trị của số ở hàm updatetmp sau đó thực hiện tính toán các giá trị, địa chỉ để thực hiện in ra led ở các hàm tiếp theo loadtoleftled và loadtorightled sau đó đi đến finish để thực hiện quay trở lại vòng lặp loop ở tiếp theo chờ cho đến khi có phím được ấn.
- trường hợp là chữ e thì thực hiện đi đến finish luôn chương trình đang chạy.
- trường hợp là chứ f (dấu =) thì thực hiện kiểm tra và in ra các biểu thức tính toán và in ra kết quả phép tính dòng (217-373).
- Hiển thị ra kết quả ở led ở hàm show_result_in_led: thực hiện lấy ra 2 số cuối và tách ra từng số 1. Thực hiện convert các số đó thành các địa chỉ của ô tương ứng và quay trở lại hiển thị qua đèn led lần lượt 2 đèn. Sau đó thực hiện đi đến resetled làm và quay trở lại hàm loop để chờ cho phép tính tiếp theo.

** giải thích thêm 1 số phần:

- Thực hiện lấy ra 2 chữ số cuối của kết quả phép tính bằng cách chia kết quả cho 100 và lấy phần phần dư chính là 2 chữ số cuối của kết quả để in ra thanh led

- Thực hiện in số của phép tính bằng cách thực hiện lấy ra bit hiển thị led, giá trị của bit từ trong stack và lưu vào địa chỉ của các thanh led ở thanh ghi $t0, $t5.