**BÀI BÁO CÁO KẾT QUẢ GIỮA KỲ**

**KIẾN TRÚC MÁY TÍNH**

Họ và tên: Đinh Huy Dương

MSSV: 20215020

**Bài 1 (A-11):**

Ý tưởng: Lấy các chữ số trong N bằng việc chia cho 10. Phần nguyên sẽ được lưu trong thanh ghi $hi, phần dư được lưu trong thanh $lo.

Sử dụng hàm “cal” để lặp lại quá trình chia 10 để lấy các chữ số và gán lại N bằng phần dư cho đến khi N=0.

Trong quá trình lặp ta gọi hàm “sum” tính tổng để kiểm tra chữ số có lẻ hay chẵn. Để kiểm tra, ta sẽ thực hiện phép AND số đó với số 1. Nếu kết quả trả về =1 thì số là số lẻ và ngược lại là số chẵn.

Kết quả:

A screenshot of a computer code

Description automatically generated with low confidence

Mã nguồn:

.data

message: .asciiz "Insert the number N: "

result1: .asciiz "\nSum of odd digits: "

result2: .asciiz "\nSum of even digits: "

.text

main:

li $v0,4

la $a0,message

syscall

li $v0,5

syscall

addi $t0,$0,10 # Decimal based number divider

addi $t1,$0,0

li $t5,0 #init $t5, $t6

li $t6,0

add $s0,$v0,$0 # $s0 = N

jal cal

nop

li $v0,4

la $a0,result2 #print the even result

syscall

li $v0,1

add $a0,$0,$t5

syscall

li $v0,4

la $a0,result1 #print the odd result

syscall

li $v0,1

add $a0,$0,$t6

syscall

li $v0,10

syscall #exit

end\_main:

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

#Procedure: cal

#var:

# $t2: Extracted digit

cal:

while:

div $s0,$t0 # Extract the digits

mfhi $t2 # $t2 = N /10

addi $sp,$sp,-4

sw $ra,0($sp) # Store the value of $ra to the Stack

jal sum # Call procedure sum

nop

lw $ra,0($sp)

addi $sp,$sp,4 # Pop $ra out of the Stack

mflo $s0 # N = N% 10

bne $s0,$0,while # Continue the loop if N > 0

jr $ra

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

#Procedure: sum

# var:

# $t4: check if $t2 is odd/even

# $t5: sum of even digit

# $t6: sum of odd digit

sum:

andi $t4,$t2,1 # if the least significant bit is set, the number is odd

bnez $t4,odd

add $t5,$t5,$t2 # add to sum of even digit

continue:

jr $ra

odd:

add $t6,$t6,$t2 # add to sum of odd digit

j continue

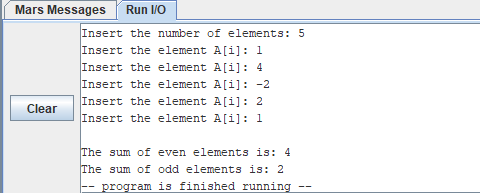
**Bài 2 (B-7):**

Chương trình duyệt qua mảng và kiểm tra từng phần tử xem nó lẻ hay chẵn và công dần vào mảng tương ứng. Tương tự bài 1, ta kiểm tra bằng việc AND phần tử với 1 và nếu kết quả ra 1, nó lẻ và cộng vào tổng lẻ.

Trong chương trình có hàm “input” sử dụng để nhập các phần tử vào trong mảng ở Data Segment. Duyệt đến khi nào chỉ số i bé hơn số các phần tử N.

Hàm “sum” sẽ kiểm tra xem phần tử là lẻ hay chẵn bằng việc gọi thêm một hàm “check” nữa

Kết quả:



Mã nguồn:

.data

message: .asciiz "Insert the number of elements: "

message1: .asciiz "Insert the element A[i]: "

resulto: .asciiz "\nThe sum of odd elements is: "

resulte: .asciiz "\nThe sum of even elements is: "

temp: .asciiz "" #add another data so A takes a new data block

A: .word

.text

main:

li $v0,4

la $a0,message

syscall

li $v0,5

syscall

add $s0,$0,$v0 # $s0 = N = Number of elements of array A

li $t0,0 # index = i =0

la $s1,A # $t1 = the address of A[0]

jal input

nop

li $t0,0 #reset i=0

jal sum

nop

li $v0,4

la $a0,resulte #print the even result

syscall

li $v0,1

add $a0,$0,$t5

syscall

li $v0,4

la $a0,resulto #print the odd result

syscall

li $v0,1

add $a0,$0,$t6

syscall

li $v0,10 #exit

syscall

end\_main:

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

# Procedure: input

# Input the value into the array in the Data Segment

# var:

# $s0: N

# $s1: address of A[0]

# $t0 = i

# $t1 = 4i

# $t2: address of A[i]

#

input:

li $v0,4

la $a0,message1

syscall

li $v0, 5

syscall

sll $t1,$t0,2 # $t1 =4\*i

add $t2,$s1,$t1 # the address of the current A[i]

sw $v0,0($t2) # store into the Data segment

addi $t0,$t0,1 # i++

bne $t0,$s0,input # if i<N, continue the loop

jr $ra

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

# Procedure: sum

# var:

# $t3 = A[i]

sum:

sll $t1,$t0,2 # $t1 =4\*i

add $t2,$s1,$t1 # the address of the current A[i]

lw $t3,0($t2) # $t3 = A[i]

addi $sp,$sp,-4

sw $ra,0($sp) # store the current $ra into Stack

jal check # check if A[i] is odd/even

nop

lw $ra,0($sp) # recover $ra

addi $t0,$t0,1 # i++

bne $t0,$s0,sum # if i<N, continue the loop

jr $ra

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

# Procedure: check

# var:

# $t4 = A[i] AND 1

# $t5: sum of even elements

# $t6: sum of odd elements

check:

andi $t4,$t3,1 # if A[i] AND 1 =1, then A[i] is an odd number

bnez $t4,odd

add $t5,$t5,$t3 # if A[i] is even then add to the even sum

continue:

jr $ra

odd:

add $t6,$t6,$t3 # # if A[i] is odd then add to the odd sum

j continue

**Bài 3 (C-12):**

Ta duyệt qua từng ký tự của xâu, so sánh chúng với các giá trị ASCII của các giá trị, kết hợp với các biểu thức logic được giải thích trong mã nguồn để đưa xét xem có phải thuộc khoảng ký tự nào. Tùy vào ký tự sẽ nhảy đến các hàm tương ứng và lưu tại rstr là xâu đích

Hàm “upper” để in hoa xâu, trừ -32 ký tự ASCII, ngược lại là “lower” cộng thêm 32, và “skip2” là để bỏ qua và lưu vào xâu đích với các ký tự khác

Kết quả:



Mã nguồn:

.data

message: .asciiz "Insert the string: "

result:. asciiz "The result: "

str: .space 100

rstr: .space 100

.text

main:

li $v0,4

la $a0,message

syscall

li $v0,8

la $a0,str

li $a1,100 # input the string with the maximum length of 100 byte

syscall

la $s0, str # address of the string

la $s1,rstr # address of the result string

li $s2,0 # index =0

li $s5,1 # mask so it can only be 0/1

li $t8,122 # $t8 = z

li $t9,90 # $t9 = Z

jal traverse

nop

li $v0,4

la $a0,message

syscall

li $v0,4

la $a0,rstr

syscall

li $v0,10 # exit

syscall

end\_main:

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

# Procedure: traverse

# var:

# $s0: address of str[0]

# $s2: index

# $s3: address of str[i]

# $t0 = str[i]

# $t5 = a <= str[i] <= z ? 1:0

# $t6 = A <= str[i] <= Z ? 1:0

# $t7: check if str[i] is an alphabetical character

traverse:

add $s3,$s0,$s2

lb $t0,0($s3) # load str[i] into $t0

slti $t1,$t0,97 # check str[i] < a

sgt $t2,$t0,$t8 # check str[i] > z

slti $t3,$t0,65 # check str[i] < A

sgt $t4,$t0,$t9 # check str[i] > Z

nor $t5,$t1,$t2 # (str[i] < a) NOR (str[i] > z) = a <= str[i] <= z

and $t5,$s5,$t5

nor $t6,$t3,$t4 # (str[i] < A) NOR (str[i] > Z) = A <= str[i] <= Z

and $t6,$s5,$t6

or $t7,$t5,$t6

beq $t7,$0,skip2

beq $t5,$s5,upper # if a< str[i] <z, go to upper

beq $t6,$s5,lower # if A< str[i] <Z, go to lower

skip:

addi $s2,$s2,1 # i++

bne $t0,0,traverse

jr $ra

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

# Procedure: Upper

upper:

addi $t0,$t0,-32 # Upper(str[i])

add $s4,$s1,$s2 # address of the current result string

sb $t0,0($s4) # store the result

j skip

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

# Procedure: Lower

lower:

addi $t0,$t0,32 # Lower(str[i])

add $s4,$s1,$s2 # address of the current result string

sb $t0,0($s4) # store the result

j skip

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

skip2:

add $s4,$s1,$s2 # address of the current result string

sb $t0,0($s4) # store the result

j skip