**Assignment number: 4**

**Subject: MICROPROCESSOR LAB**

Name: ***RIA MITTAL***

Class: ***SECOND YEAR ENGINEERING***

Division: ***B***

Roll no: ***222008***

Batch: ***B1***

**PROBLEM STATEMENT:**

Write X86/64 ALP to perform non-overlapped and overlapped block transfer (with and without string specific instructions). Block containing data can be defined in the data segment.

**Code:**

**Non-overlap:**

section .data

menu1 db 10,"----MENU----"

db 10,"1)Non Overlapping withOUT String"

db 10,"2)Non Overlapping with String"

db 10,"3)Exit"

menu1\_len equ $-menu1

m1 db 10,"Block contents before transfer are="

m1\_l1 equ $-m1

m2 db 10,"Block contents after transfer are="

m2\_l2 equ $-m2

m3 db 10,"Source Block contents are="

m3\_l3 equ $-m3

m4 db 10,"Destination Block contents are="

m4\_l4 equ $-m4

m5 db 10,"Do you want to continue="

m5\_l5 equ $-m5

m6 db 10,"Enter your choice=",10

m6\_l6 equ $-m6

srcblk db 11h,12h,13h,14h,15h ;declare source array of 5 elements

dstblk times 5 db 0

cnt equ 05

spacechar db 20h ;spaces between no.s

lfmsg db 10 ;new line

;--------------------------------------------------------------------------------

section .bss

choice resb 02 ;to accept choice

dispbuff resb 02 ;

;----------------------macro for accept and display-----------------------

%macro scall 4

mov rax,%1

mov rdi,%2

mov rsi,%3

mov rdx,%4

syscall

%endmacro

;-------------------macro for exit---------------------------------

%macro exit 0

mov rax,60

xor rdi,rdi

syscall

%endmacro

;--------------------------------------------------------------------

section .text

global \_start

\_start:

scall 01,01,m1,m1\_l1

call showblks ;call showblk procedure before transfer

;--------------------ACCEPT MENU FROM USER-----------------------------------

menu:

scall 01,01,menu1,menu1\_len

scall 01,01,m6,m6\_l6

scall 0,0,choice,02

mov al,byte[choice]

cmp al,'1'

jne case2

call nwstr ;NONOVERLAPPING WITHOUT STRING

jmp cont ;continue label

case2: cmp al,'2'

jne case3

call nstr ;NONOVERLAPPING WITH STRING

jmp cont

case3:

cmp al,'3'

je exit

jmp cont

cont: ;DO you want to continue

scall 01,01,m5,m5\_l5

scall 0,0,choice,02

mov al,byte[choice]

cmp al,'y'

je menu

jmp exit

;--------------DISPLAY Source and Destination BLOCKS-----------------

showblks:

scall 01,01,m3,m3\_l3

mov rbx,srcblk

call dispblk\_proc

scall 01,01,m4,m4\_l4

mov rbx,dstblk

call dispblk\_proc

ret

;

dispblk\_proc:

mov rcx,05

rdisp:

push rcx

mov dl,[rbx]

call disp8\_proc

inc rbx

scall 01,01,spacechar,01

pop rcx

loop rdisp

ret

;-----------DISPLAY 2 digit Number-----------------------------

disp8\_proc:

mov rcx,02

mov rdi,dispbuff

dup1:

rol dl,04

mov al,dl

and al,0fh

cmp al,09h

jbe L1

add al,07

L1: add al,30h

mov [rdi],al

inc rdi

loop dup1

scall 01,01,dispbuff,02

ret

;------------------NONOVERLAP WITHOUT STRING-----------------------

nwstr:

mov rsi,srcblk

mov rdi,dstblk

mov rcx,05

UP1:

mov al,[rsi]

mov [rdi],al

inc rsi

inc rdi

loop UP1

scall 01,01,m2,m2\_l2

call showblks

ret

;------------------NONOVERLAP WITH STRING INSTRUCTIONS----------------------- ;------------------

nstr:

mov rsi,srcblk

mov rdi,dstblk

mov rcx,05

cld

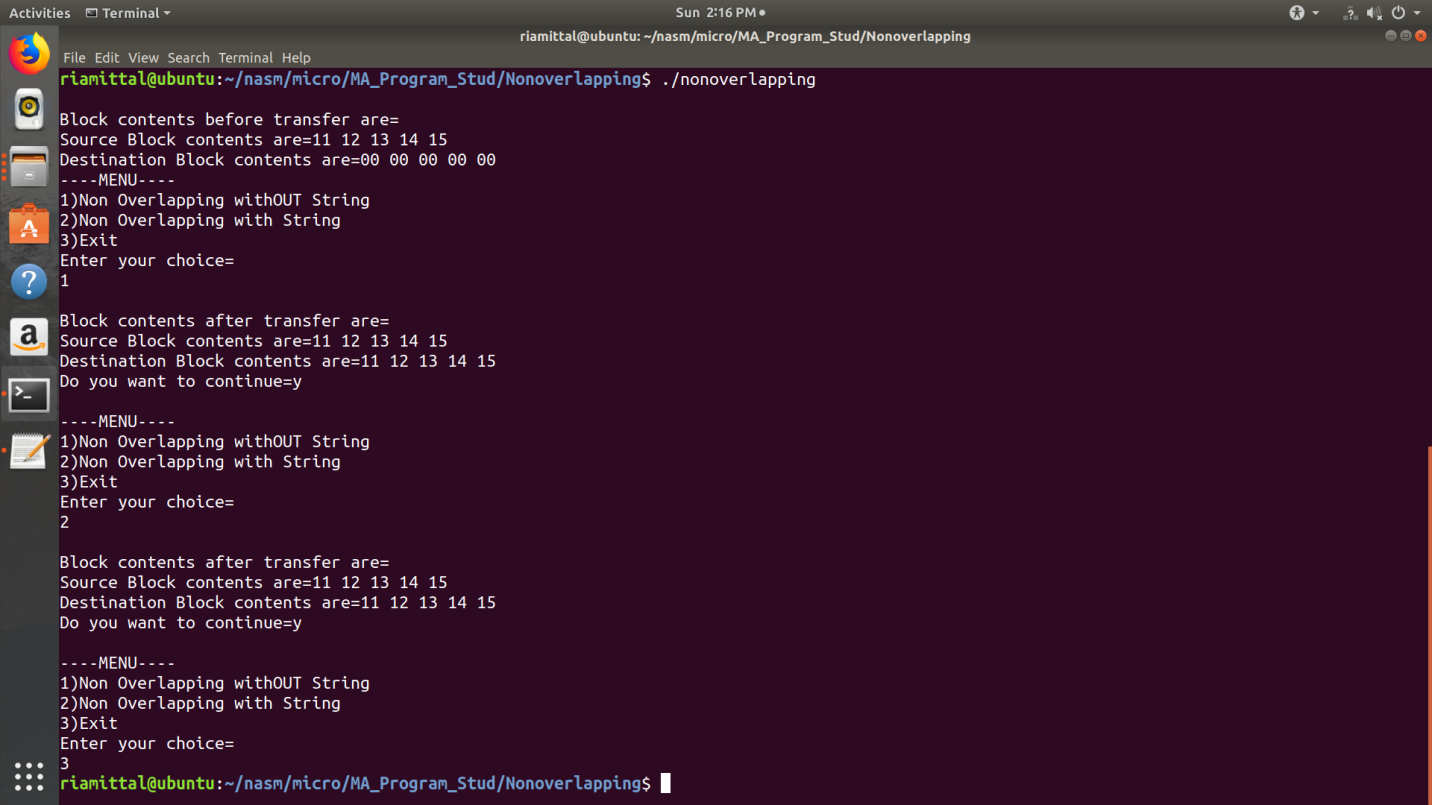
rep movsb

scall 01,01,m2,m2\_l2

call showblks

ret

;-------------------------------------------------------------------------



**Overlap:**

section .data

menumsg db 10,'\*\*\*Overlap block transfer\*\*\*',10

db 10,'1.Block transfer without string '

db 10,'2.Block transfer with string '

db 10,'3.exit '

menumsg\_len equ $-menumsg

wrmsg db 10,'Wrong choice entered',10

wrmsg\_len equ $-wrmsg

bfrmsg db 10,'Block contents before transfer: '

bfrmsg\_len equ $-bfrmsg

afrmsg db 10,'Block contents after transfer:'

afrmsg\_len equ $-afrmsg

srcmsg db 10,'Source block contents '

srcmsg\_len equ $-srcmsg

dstmsg db 10,'Destination block contents '

dstmsg\_len equ $-dstmsg

srcblk db 01h,02h,03h,04h,05h

dstblk times 3 db 0

cnt equ 05

spacechar db 20h

lfmsg db 10,10

section .bss

optionbuff resb 02

dispbuff resb 02

%macro dispmsg 2

mov rax,04

mov rbx,01

mov rcx,%1

mov rdx,%2

int 80h

%endmacro

%macro accept 2

mov rax,03

mov rbx,00

mov rcx,%1

mov rdx,%2

int 80h

%endmacro

section .text

global \_start

\_start:

dispmsg bfrmsg,bfrmsg\_len

call show

menu:

dispmsg menumsg,menumsg\_len

accept optionbuff,02

cmp byte [optionbuff],'1'

jne case2

call wos

jmp exit1

case2:

cmp byte [optionbuff],'2'

jne case3

call ws ;ws=with string

jmp exit1

case3:

cmp byte [optionbuff],'3'

je exit

dispmsg wrmsg,wrmsg\_len

jmp menu

exit1:

dispmsg afrmsg,afrmsg\_len

call show

dispmsg lfmsg,2

jmp menu

exit:

mov eax,01

mov ebx,00

int 80h

dispblk:

mov rcx,cnt

rdisp:

push rcx

mov bl,[esi]

call disp8

inc esi

dispmsg spacechar,1

pop rcx

loop rdisp

ret

wos:

mov esi,srcblk + 04h

mov edi,dstblk + 02h

mov ecx,cnt

x:

mov al,[esi]

mov [edi],al

dec esi

dec edi

loop x

ret

ws:

mov esi,srcblk + 04h

mov edi,dstblk + 02h

mov ecx,cnt

std

rep movsb

show:

dispmsg srcmsg,srcmsg\_len

mov esi,srcblk

call dispblk

dispmsg dstmsg,dstmsg\_len

mov esi,dstblk-02h

call dispblk

ret

disp8:

mov ecx,02

mov edi,dispbuff

dub1:

rol bl,4

mov al,bl

and al,0fh

cmp al,09h

jbe x1

add al,07

x1:

add al,30h

mov [edi],al

inc edi

loop dub1

dispmsg dispbuff,3

ret

