Name: Prabodh Wankhede

Roll no. 57

Class: SE(B)

Batch – B3

Assignment No. 4

Problem Statement: Write x86/64 ALP to perform multiplication of two 8 bit hexadecimal nos. Use successive addition and add and shift method (use of 64 bit registers is expected)

**Code:**

**Program: Successive Addition**

section .data

welmsg db 10,'Multiplication using successive addition',10

welmsg\_len equ $-welmsg

nummsg db 10,'Enter two digits of number::'

nummsg\_len equ $-nummsg

resmsg db 10,'Multiplication of elements::'

resmsg\_len equ $-resmsg

blankmsg db 10,' ',10

blank\_len equ $-blankmsg

section .bss

numascii resb 03

multi1 resb 01

multi2 resb 01

resl resb 01

resh resb 01

dispbuff resb 04

%macro display 2

mov rax,01

mov rdi,01

mov rsi,%1

mov rdx,%2

syscall

%endmacro

%macro accept 2

mov rax,00

mov rdi,00

mov rsi,%1

mov rdx,%2

syscall

%endmacro

section .text

global \_start

\_start:

display welmsg,welmsg\_len

display nummsg,nummsg\_len

accept numascii,3

call packnum

mov byte[multi1],bl

display nummsg,nummsg\_len

accept numascii,3

call packnum

mov byte[multi2],bl

mov ecx,00h

mov eax,[multi1]

mov bl,[multi2]

addup:

add ecx,eax

dec bl

jnz addup

mov [resl],ecx

display resmsg,resmsg\_len

mov ebx,[resl]

mov bh,00h

call disp16\_proc

display blankmsg,blank\_len

exit:mov rax,60

mov ebx,00

syscall

packnum:

mov bl,0

mov ecx,02

mov esi,numascii

up1:rol bl,04

mov al,[esi]

cmp al,39h

jbe skip1

sub al,07h

skip1: sub al,30h

add bl,al

inc esi

loop up1

ret

disp16\_proc:

mov ecx,4

mov edi,dispbuff

dup1: rol bx,4

mov al,bl

and al,0fh

cmp al,09

jbe dskip

add al,07h

dskip: add al,30h

mov[edi],al

inc edi

loop dup1

display dispbuff,4

ret

OUTPUT::

[HL@localhost ~]$ nasm -f elf64 mul.asm

[HL@localhost ~]$ ld -o mul mul.o

[HL@localhost ~]$ ./mul

Multiplication using successive addition

Enter two digits of number::02

Enter two digits of number::05

Multiplication of elements::000A

[HL@localhost ~]$

PROGRAM :: ADD AND SHIFT

section .data

welmsg db 10,'Multiplication using Add and shift method',10

welmsg\_len equ $-welmsg

nummsg db 10,'enter two digits of no',10

nummsg\_len equ $-nummsg

resmsg db 10,'Multiplication of elements',10

resmsg\_len equ $-resmsg

blankmsg db 10,'',10

blankmsg\_len equ $-blankmsg

section .bss

numascii resb 03

multi1 resb 02

resl resb 02

multi2 resb 02

dispbuff resb 04

%macro display 2

mov rax,01

mov rdi,01

mov rsi,%1

mov rdx,%2

syscall

%endmacro

%macro accept 2

mov rax,00

mov rdi,00

mov rsi,%1

mov rdx,%2

syscall

%endmacro

section .text

global \_start

\_start:

display welmsg,welmsg\_len

display nummsg,nummsg\_len

accept numascii,3

call packnum

mov byte[multi1],bl

display nummsg,nummsg\_len

accept numascii,3

call packnum

mov byte[multi2],bl

mov al,[multi1]

mov cl,0

mov edx,08h

addup:

rcr al,01

jnc next1

mov bh,00h

shl bx,cl

add [resl],bx

mov bl,[multi2]

next1:

inc cl

dec edx

jnz addup

display resmsg,resmsg\_len

mov ebx, [resl]

call disp16\_proc

display blankmsg,blankmsg\_len

exit:

mov rax,60

mov rbx,00

syscall

packnum:

mov bl,0

mov ecx,02

mov esi,numascii

up1:

rol bl,04

mov al,[esi]

cmp al,39h

jbe skip1

sub al,07h

skip1:

sub al,30h

add bl,al

inc esi

loop up1

ret

disp16\_proc:

mov ecx,4

mov edi,dispbuff

dup1:

rol bx,4

mov al,bl

and al,0fh

cmp al,09

jbe dskip

add al,07h

dskip:

add al,30h

mov [edi],al

inc edi

loop dup1

display dispbuff,4

ret

**OUTPUT:**

hl-2@hl2-Veriton-Series:~$ nasm -f elf64 shift.asm

hl-2@hl2-Veriton-Series:~$ ld -o shift shift.o

hl-2@hl2-Veriton-Series:~$ ./shift

Multiplication using Add and shift method

enter two digits of no

02

enter two digits of no

04

Multiplication of elements

0008

hl-2@hl2-Veriton-Series:~$

**Program: Add and Shift Method**

section .data

welmsg db 10,'Multiplication using add and shift method',10

welmsg\_len equ $-welmsg

nummsg db 10,'Enter two digit number::'

nummsg\_len equ $-nummsg

resmsg db 10,'Multiplication of elements::'

resmsg\_len equ $-resmsg

blankmsg db 10,' ',10

blankmsg\_len equ $-blankmsg

section .bss

numascii resb 03

multi1 resb 01

multi2 resb 01

resl resb 01

resh resb 01

dispbuff resb 04

%macro display 2

mov rax,01

mov rdi,01

mov rsi,%1

mov rdx,%2

syscall

%endmacro

%macro accept 2

mov rax,00

mov rdi,00

mov rsi,%1

mov rdx,%2

syscall

%endmacro

section .text

global \_start:

\_start:

display welmsg, welmsg\_len

display nummsg, nummsg\_len

accept numascii,3

call packnum

mov byte[multi1],bl

display nummsg, nummsg\_len

accept numascii,3

call packnum

mov byte[multi2],bl

mov al, [multi1]

mov cl,0

mov edx,08h

addup:

rcr al, 01

jnc next1

mov bh, 00h

shl bx, cl

add [resl],bx

mov bl,[multi2]

next1:

inc cl

dec edx

jnz addup

display resmsg,resmsg\_len

mov ebx,[resl]

call disp16\_proc

display blankmsg, blankmsg\_len

exit:

mov rax,60

mov rbx,00

syscall

packnum:

mov bl,0

mov ecx,2

mov esi,numascii

up1:

rol bl,4

mov al,[esi]

sub al, 30H

cmp al, 09H

jbe skip1

sub al, 07h

skip1:

add bl,al

inc esi

loop up1

ret

disp16\_proc:

mov edi, dispbuff

mov ecx,4

up2:

rol bx,4

mov dl,bl

and dl,0Fh

cmp dl,09h

jbe skip2

add dl,07h

skip2:

add dl, 30h

mov[edi], dl

inc edi

loop up2

display dispbuff,4

ret

**\*\*\*\*OUTPUT\*\*\*\***  
[student@localhost ~]$ nasm -f elf64 addnshift.asm  
[student@localhost ~]$ ld -o addnshift addnshift.o  
[student@localhost ~]$ ./addnshift  
  
Multiplication using add and shift method  
  
Enter two digit number::03  
  
Enter two digit number::02  
  
Multiplication of elements::0006