23I0544  
MUHAMMAD HAMMAD

LAB 09

TASK 01:

;Task#1

;Write ASM instructions that calculate EAX \* 21 using binary multiplication.

;Hint: 21 = 2^4 + 2^2 + 2^0

INCLUDE Irvine32.inc

.data

.code

main PROC

mov eax, 1

mov ebx, eax

shl eax, 4

mov ecx, eax

mov eax, ebx

shl eax, 2

add ebx, eax

add ebx, ecx

mov eax, ebx

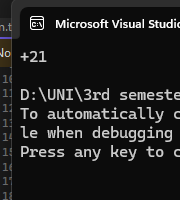
call writeint

call crlf

EXIT

main ENDP

END main



TASK 02:

;Task#2

;Give an assembly language program to move -128 in ax and expand eax. Using shift and rotate

;instruction.

INCLUDE Irvine32.inc

.data

.code

main PROC

mov eax, 0

call dumpregs

mov ax, -128

call dumpregs

shl eax, 16

call dumpregs

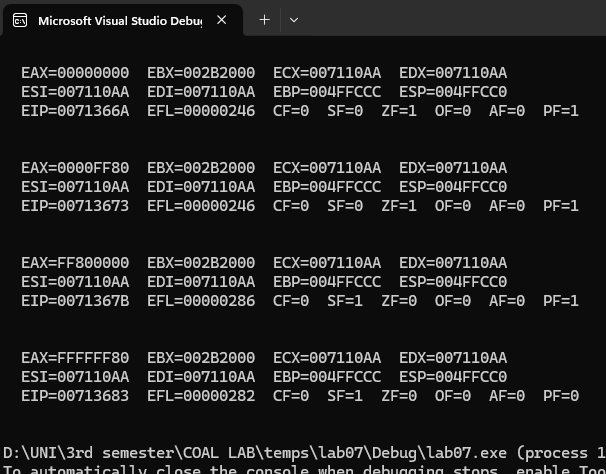
sar eax, 16

call dumpregs

EXIT

main ENDP

END main



TASK: 03

;Task#3

;Write a series of instructions that shift the lowest bit of AX into the highest bit of BX without using

;the SHRD instruction. Next, perform the same operation using SHRD.

INCLUDE Irvine32.inc

.data

.code

main PROC

mov eax, 0

mov ebx, 0

call writebin

mov ecx, eax

mov eax, ebx

call crlf

call writebin

mov eax, ecx

call crlf

call crlf

mov eax, 1111111111111111b

mov ebx, 0000000000000000b

call writebin

call crlf

mov ecx, eax

mov eax, ebx

call writebin

call crlf

mov eax, ecx

call crlf

ror ax, 1

rcr bx, 1

call writebin

call crlf

mov ecx, eax

mov eax, ebx

call writebin

call crlf

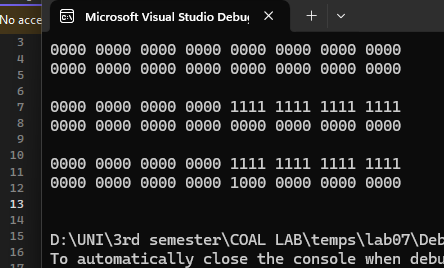
mov eax, ecx

call crlf

EXIT

main ENDP

END main



TASK 04:

;Task#4

;Implement the following C++ expression in assembly language, using 32-bit signed operands:

;val1 = (val2 / val3) \* (val1 / val2);

INCLUDE Irvine32.inc

.data

val1 WORD 12

val2 WORD 6

val3 WORD 3

result SDWORD ?

.code

main PROC

mov eax, 0

mov ebx, 0

mov ecx, 0

mov ax, val2

cwd

idiv val3

mov bx, ax

mov ax, val1

cwd

idiv val2

mov cx, ax

imul bx, cx

mov result, ebx

mov eax, result

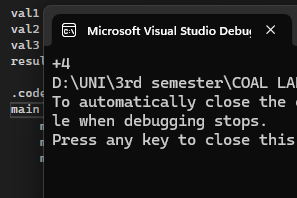
call WriteInt

EXIT

main ENDP

END main

END main

  
TASK 05:

;Task#5

;Create a procedure Extended\_Add procedure to add two 64-bit (8-byte) integers.

INCLUDE Irvine32.inc

.data

num1l DWORD 12345678h ; using hexadecimal values

num1h DWORD 00000001h

num2l DWORD 87654321h

num2h DWORD 00000002h

resl DWORD ?

resh DWORD ?

.code

Extended\_Add PROC

mov eax, num1l

add eax, num2l

mov resl, eax

mov eax, num1h

adc eax, num2h

mov resh, eax

ret

Extended\_Add ENDP

main PROC

call Extended\_Add

mov eax, resh

call WriteHex

mov eax, resl

call WriteHex

EXIT

main ENDP

END main

