A picture containing text

Description automatically generated

**Name: Saad Rehman  
Student ID: 21F-9640  
Section: 3A**  
**Degree: BS-CS  
Teacher Name: Sir Abdul Qadeer Bilal  
Course: EL-2003 (COAL)  
Lab Number: 07  
……………………………………………………………………………**

**Note: Perform each Question on AFD Debugger.**

**Qno.1**

Write a code to multiple 19 with 5 using shift operator.

**Answer**

[org 0x100]

jmp start

multiplicand: db 19 ; Its 4 bit multipicand, (8 bit space)

multiplier: db 5 ; 4 bit multiplier

result: db 0 ; result in 8 bit

start:

mov cl, 4 ; bit count

mov bl, [multiplicand] ; storing multiplicand in bl

mov dl, [multiplier] ; storing multiplier in dl

checkbit:

shr dl, 1 ; move right most bit in carry

jnc skip ; it will skip the addition, if bit is zero

add [result], bl ; result

skip:

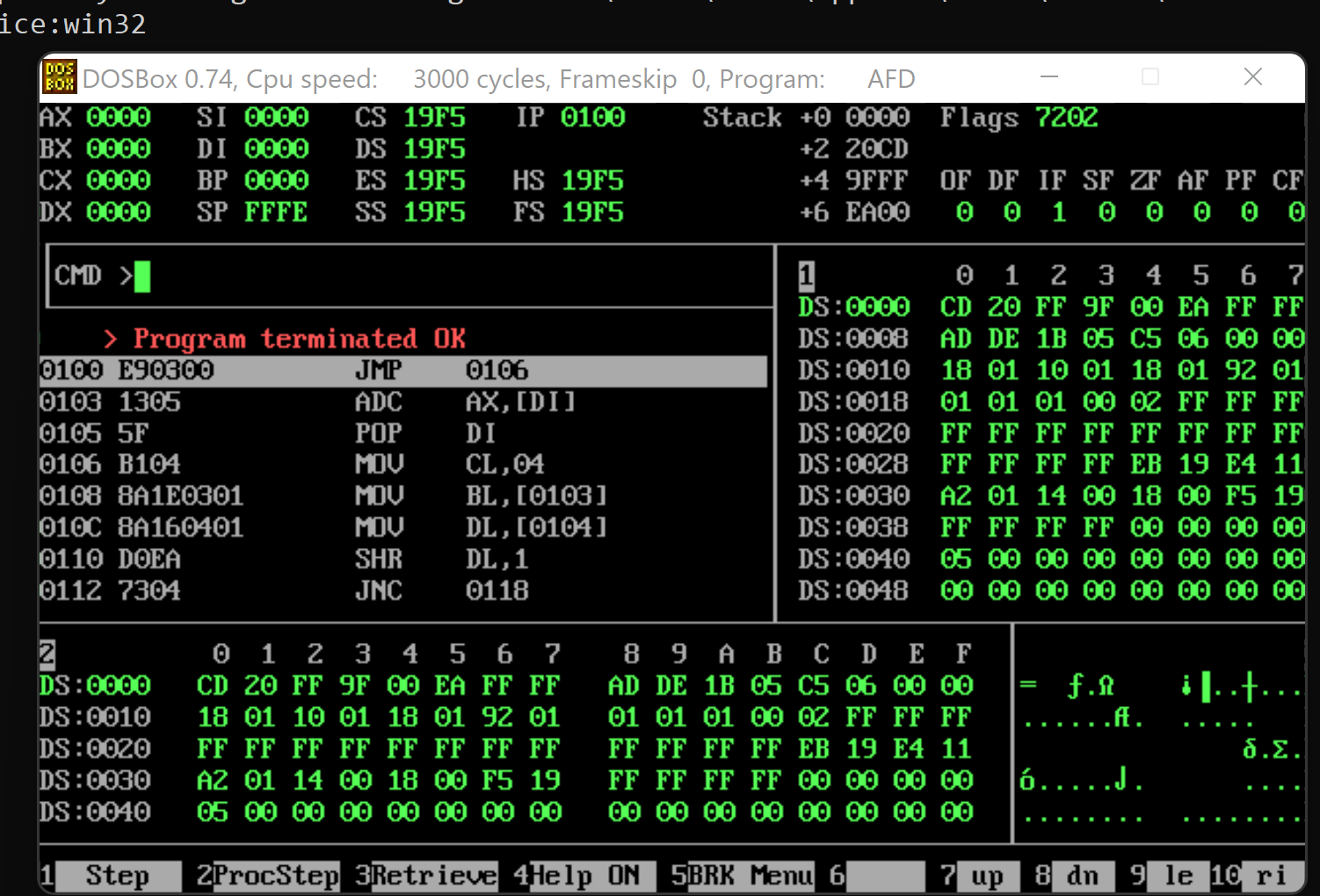
shl bl, 1 ; shift multiplicand left

dec cl ; decrement bit count

jnz checkbit ; stop if bit is zero

mov ax, 0x4c00 ;

int 0x21

****

**Qno.2**

Write a code to solve the following equation ((7-2)\*5+(8/4))\*2.

**Answer**

[org 0x100]

jmp start

num1: db 7,2,5,8,4,2 ;

result: db 0 ; result in 8 bit

result2: db 0

start:

mov cl, 4 ; bit count

mov al,[num1]

mov bl,[num1+1]

sub al,bl

mov bl,[num1+2]

mul al,bl

mov [result],al

xor al,bl

mov al,[num1+3]

mov bl,[num1+4]

div al,bl

add [result],al

mov bl,[num1+5]

mul [result],bl

mov [result2],[result]

mov ax, 0x4c00 ;

int 0x21

A screenshot of a computer

Description automatically generated with medium confidence

**Qno.3**

Write a code for Extended multiplication 1300 \* 500.

**Answer**

[org 0x100]

jmp start

multiplicand: dd 1300 ; it's 16 bit multiplicand (32bit space)

multiplier: dw 500 ; The 16 bit multiplier

result: dd 0 ; result in 32 bit

start:

mov cl, 16 ; counter

mov dx, [multiplier] ; store multiplier in dx

checkbit:

shr dx, 1 ; move right most bit in carry

jnc skip ; jump if not carry

mov ax, [multiplicand]

add [result], ax ;

mov ax, [multiplicand+2]

adc [result+2], ax ;

skip:

shl word [multiplicand], 1

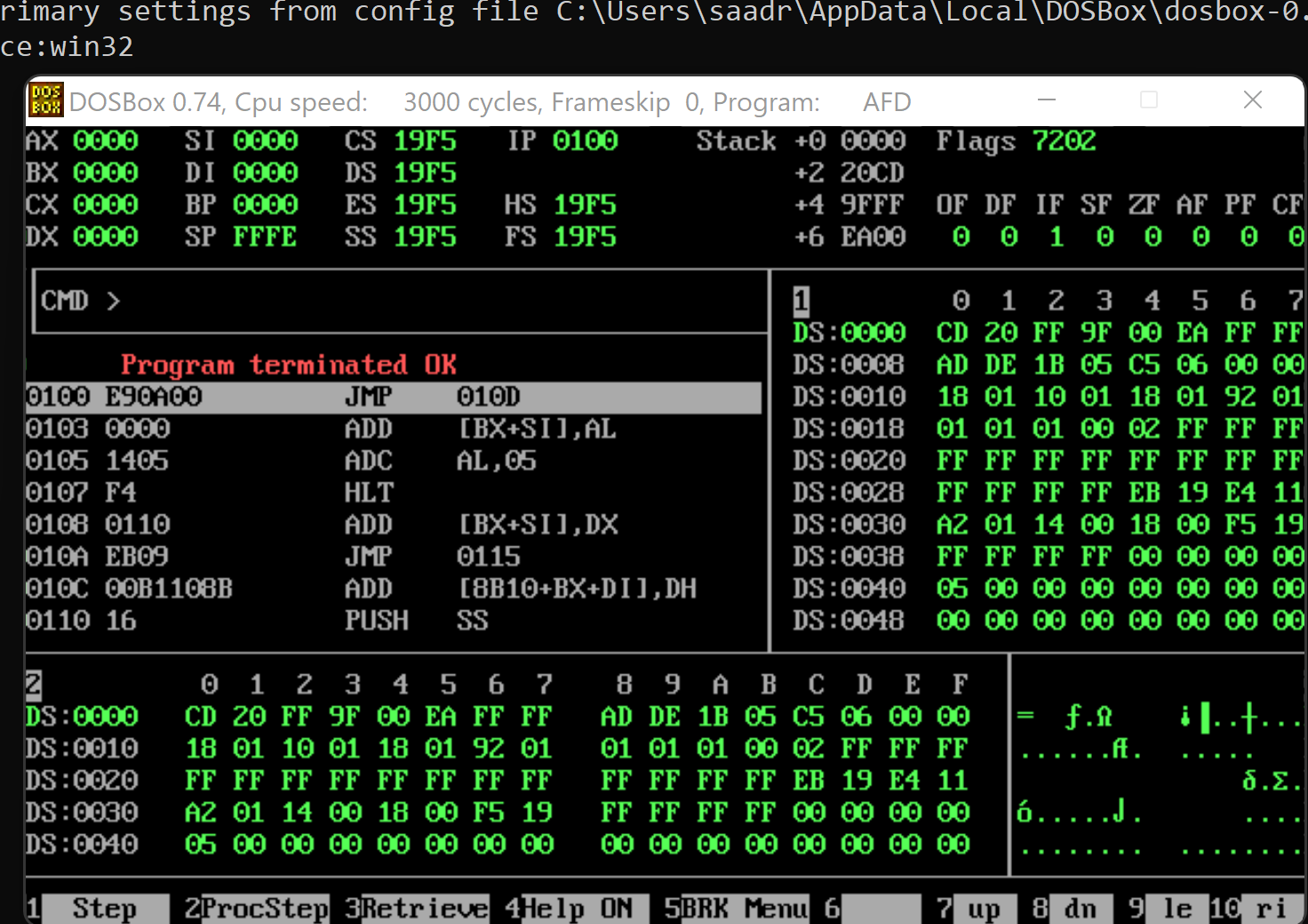
rcl word [multiplicand+2], 1 ;

dec cl ; decrement bit count

jnz checkbit ; repeat if bits left

mov ax, 0x4c00 ;

int 0x21



**Qno.4**

Write a detailed note on why we compare different flags for signed and unsigned

numbers. Give examples and add code with screenshot

**Answer**

Signed data types use a flag sign before the negative numbers they represent. Unsigned data types do not use a flag sign before numbers, as they only represent positive integers.

In the case of an unsigned int, shifting right just means that we remove the LSB, shift all of the remaining bits 1 position to the right and then fill the MSB with a 0.  
  
However, in the case of a signed int the required behavior is different. Steps and remain the same, but step may have a 1 or a 0 inserted - this depends on whether the number is negative. A negative number will have a 1 inserted, a positive one will have a 0 inserted.  
  
The instruction for an unsigned int is SHR (Shift Right), while the instruction for a signed int is SAR (Sign Adjust (shift)Right).

[org 0x100]

jmp start

signedmultiplicand: db -19 ; Its 4 bit multipicand, (8 bit space)

unsignedmultiplier: db 5 ; 4 bit multiplier

result: db 0 ; result in 8 bit

start:

mov cl, 4 ; bit count

mov bl, [signedmultiplicand] ; storing multiplicand in bl

mov dl, [unsignedmultiplier] ; storing multiplier in dl

checkbit:

shr dl, 1 ; move right most bit in carry

jnc skip ; it will skip the addition, if bit is zero

add [result], bl ; result

skip:

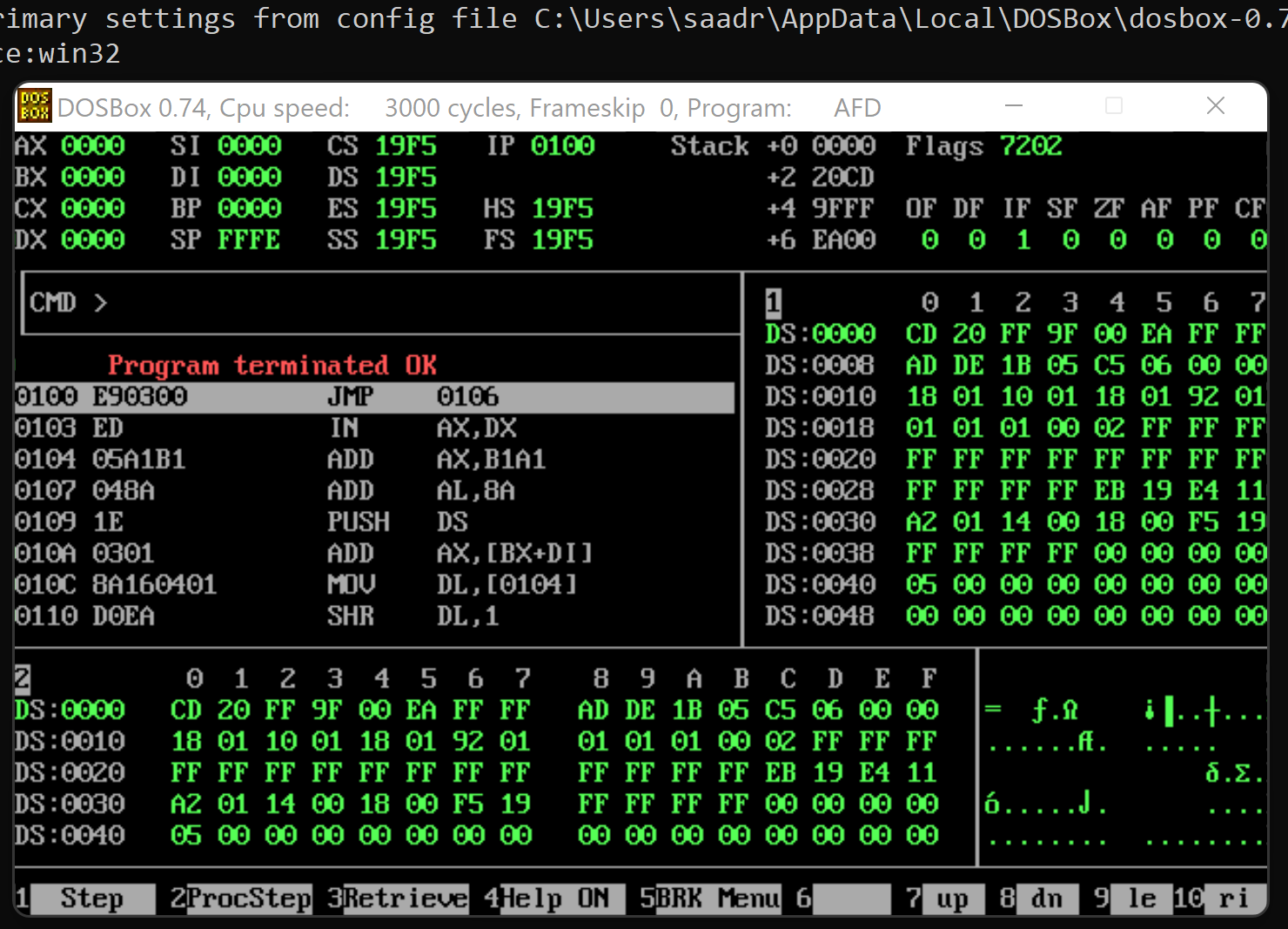
shl bl, 1 ; shift multiplicand left

dec cl ; decrement bit count

jnz checkbit ; stop if bit is zero

mov ax, 0x4c00 ;

int 0x21



**The End**