Lab 4 – Bitwise Operations

Bitwise operations are operations that are executed for each bit (bit-by-bit).

We have:

- logical bitwise operations
- shift operations
- rotate operations

Boolean op -> logical bitwise operations

Operand1 – op1	Operand2 – op2	And op1, op2	Or op1, op2	Xor op1, op2	Not op1
0	0	0	0	0	1
0	1	0	1	1	1
1	0	0	1	1	0
1	1	1	1	0	0

The IA-32 assembly instructions that perform these logical bitwise operations are detailed below:

and *operand1, operand2*

or *operand1, operand2*

xor *operand1, operand2*

not *operand1*

where *operand1* and *operand2* are either registers, memory references (i.e. variables) or constants (*operand1* can not be a constant!) both of the same size/type: byte, word, doubleword.

Each single bit of *operand2* is and/or/xor with each corresponding bit from *operand1*.

The result is in all cases saved in operand1

The *not* operation is a unary operation, so the bits of *operand1* are modified directly

General rules for OR and AND

X or 0 = X
X or 1 = 1
X and 0 = 0
X and 1 = X

Examples:
Data segment: a db 0000.1111b
Code segment:
and byte[a], 0000.0101b ; => 0000.1111b and
0000.0101b => a=0000.0101b
or byte[a], 0000.1101b ; => 0000.0101b or
0000.1101b =>a=0000.1101b
xor byte[a], 0000.0110b ; => 0000.1101b xor
0000.0110b =>a=0000.1011b

Xor allows us to save value 0 in a register (instead of using mov eax, 0):
Xor eax, eax => eax = 0

not byte[a] ; => not 0000.1011b =>a=1111.0100b

Shift instructions and operations

SHL *destination, count* *destination: reg, mem* *count: imm8, or CL reg*	The SHL **(shift logic left)** instruction performs a logical left shift on the destination operand, filling the lowest bit with 0. The highest bit is moved to the Carry flag, and the bit that was in the Carry flag is discarded.	mov dl, 0000.1111b shl dl, 3 ; dl = 0111.1000b

SHR *destination, count*
 	
destination: reg, mem
count: imm8, or CL reg
	The SHR **(shift logic right)** instruction performs a logical right shift on the destination operand, replacing the highest bit with a 0.
	The lowest bit is copied into the Carry flag, and the bit that was previously in the Carry flag is lost.
		mov bl, 1010.1010b
		shr bl, 4 ; bl = 0000.1010b
		
		mov eax, 12340000h
		shr eax, 16 ; eax = 00001234h

SAL *destination, count*
 	
destination: reg, mem
count: imm8, or CL reg
	The SAL **(shift arithmetic left)** instruction works the same as the SHL instruction. For each shift count, SAL shifts each bit in the destination operand to the next highest bit position. The lowest bit is assigned 0.
	The highest bit is moved to the Carry flag, and the bit that was in the Carry flag is discarded.
		mov al, 1110.0011b
		sal al, 1 ; al=1100.0110b, cf=1
		
		mov bl, 0000.1111b
		sal bl, 2 ; bl = 0011.1100b

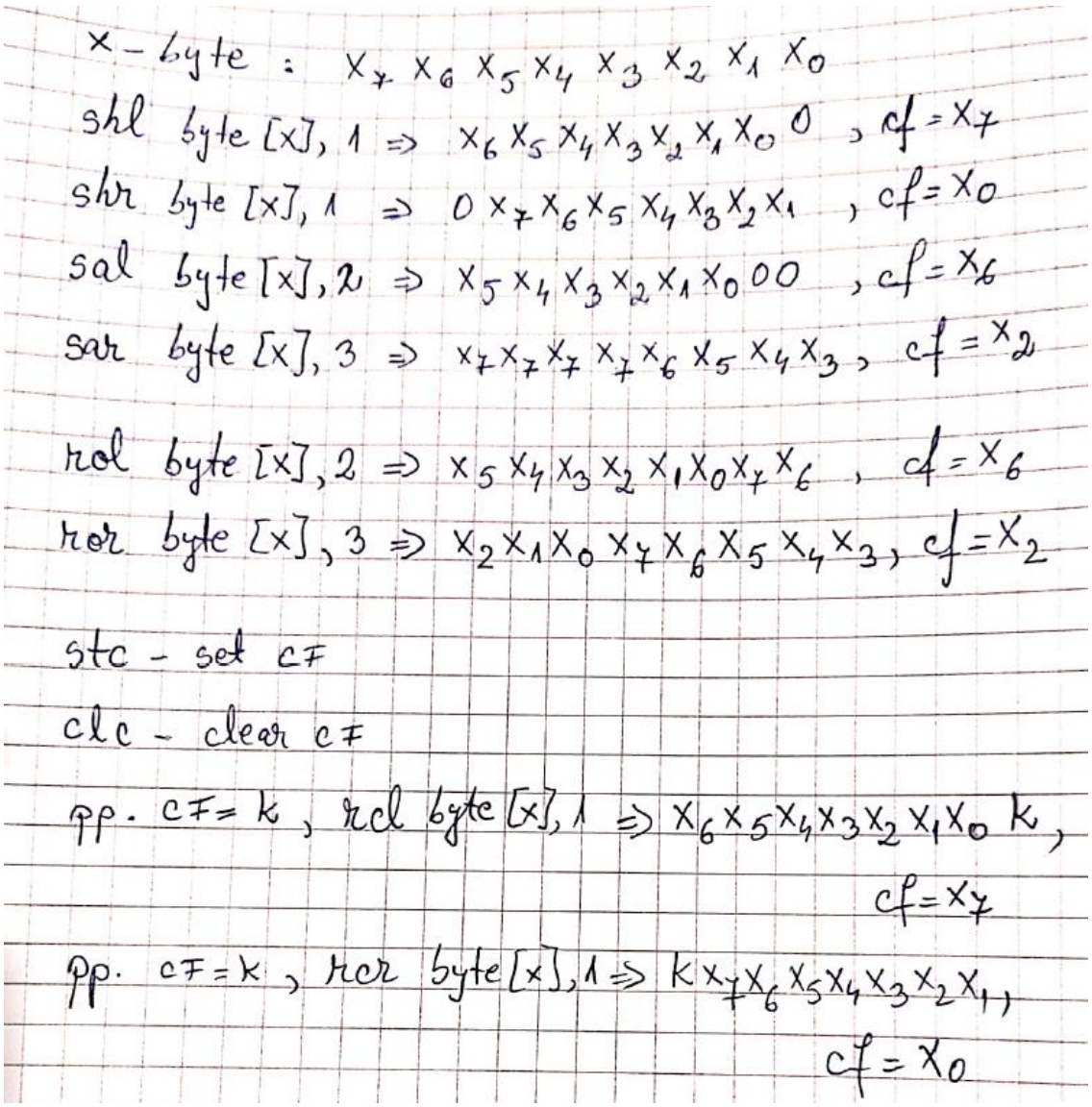
SAR *destination, count*
 	
destination: reg, mem
count: imm8, or CL reg
	The SAR **(shift arithmetic right)** instruction performs a logical right shift on the destination operand, filling all the positions with the most significant bit (bit sign).
	The lowest bit is copied into the Carry flag, and the bit that was previously in the Carry flag is lost.
		mov al, 1110.0011b
		sar al, 1 ; al=1111.0001b, cf =1
		
		mov bl, 10101010b
		shr bl, 4 ; bl = 11111010b

Rotate instructions and operations

ROL *destination, count*
 	
destination: reg, mem
count: imm8, or CL reg
	The ROL **(rotate left)** instruction shifts each bit to the left.
	The highest bit is copied into the Carry flag and the lowest bit position.
		mov bl, 0111.1000b
		rol bl, 4 ; bl = 1000.0111b, cf = 1
		
		mov bl, 1111.0000b
		rol bl, 6 ; bl = 0011.1100b

ROR *destination, count* *destination: reg, mem* *count: imm8, or CL reg*	The ROR **(rotate right)** instruction shifts each bit to the right and copies the lowest bit into the Carry flag and the highest bit position.	mov al, 1010.1110b ror al, 3 ; al=1101.0101b, cf=1

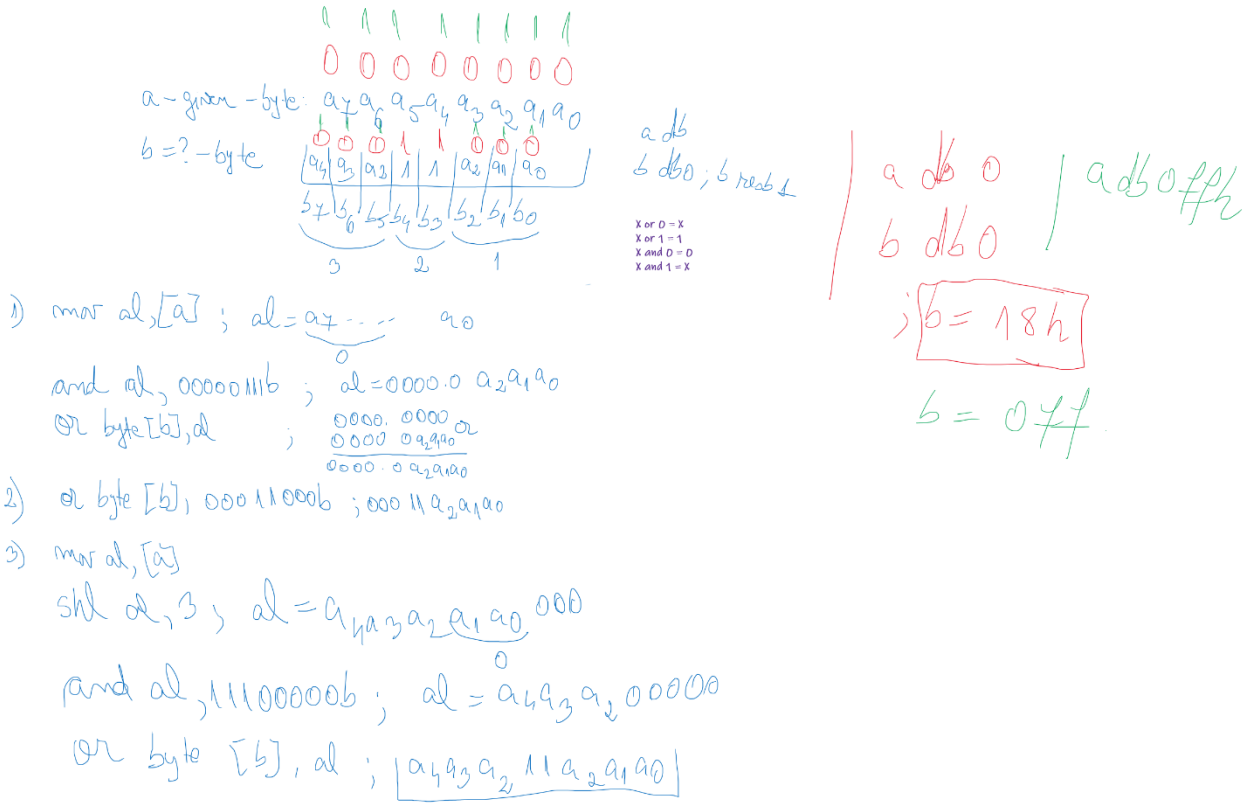
		mov al, 1111.0000b ror al, 5 ; al=1000.0111b
RCL *destination, count* *destination: reg, mem* *count: imm8, or CL reg*	The RCL **(rotate carry left)** instruction shifts each bit to the left, copies the Carry flag to the LSB, and copies the MSB into the Carry flag.	mov al, 0 stc rcl al, 1 ; al=0000.0001b, cf = 0 mov al, 0 stc rcl al, 2 ; al=0000.0010b, cf=0 mov al, 0 stc rcl al, 3 ;al = 0000.0100b, cf=0
RCR *destination, count* *destination: reg, mem* *count: imm8, or CL reg*	The RCR **(rotate carry right)** instruction shifts each bit to the right, copies the Carry flag into the MSB, and copies the LSB into the Carry flag.	mov bl, 0 stc rcr bl, 1 ; bl=1000.0000b, cf = 0 mov bl, 0 stc rcr bl, 2 ;bl=0100.0000b, cf=0 mov bl, 0 stc rcr bl, 3, bl =0010.0000b, cf=0



Prob1:

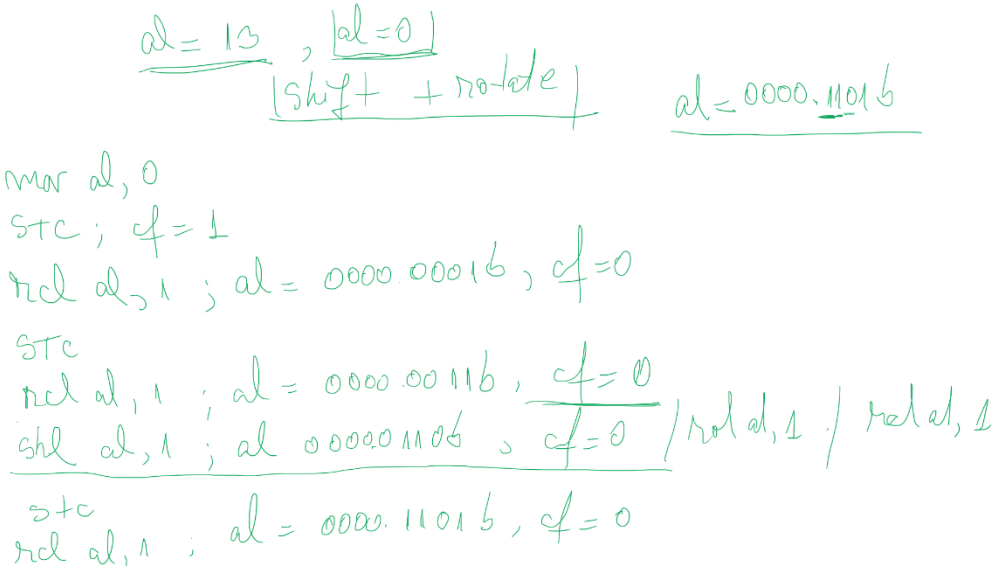
Being given a byte A, construct a new byte B in the following way:

- bits 0-2 of B should be equal to bits 0-2 of A
- bits 3-4 of B should be set to 1
- bits 5-7 of B should be equal to bits 2-4 of A



Prob 2:

Generate value 13 in al.



Prob 3.

Identify the numbers a and b from 2 doubleword X and Y:

a is represented of bits 11-16 of X

b is represented of bits 26-31 of Y

Compute then a*b.