Lab # 11

# OBJECTive

Learning Logic group of instructions (AND, OR and XOR).

**THEORY:**

The processor instruction set provides the instructions AND, OR, XOR, TEST, and NOT Boolean logic, which tests, sets, and clears the bits according to the need of the program.

The format for these instructions –

|  |  |  |
| --- | --- | --- |
| **Sr.No.** | **Instruction** | **Format** |
| 1 | AND | AND operand1, operand2 |
| 2 | OR | OR operand1, operand2 |
| 3 | XOR | XOR operand1, operand2 |
| 5 | NOT | NOT operand1 |

The first operand in all the cases could be either in register or in memory. The second operand could be either in register/memory or an immediate (constant) value. However, memory-to-memory operations are not possible. These instructions compare or match bits of the operands and set the CF, OF, PF, SF and ZF flags.

## The AND Instruction

The AND instruction is used for supporting logical expressions by performing bitwise AND operation. The bitwise AND operation returns 1, if the matching bits from both the operands are 1, otherwise it returns 0. For example −

**and destination, source**

movAL, 01

and **AL,FF**

## The OR Instruction

The OR instruction is used for supporting logical expression by performing bitwise OR operation. The bitwise OR operator returns 1, if the matching bits from either or both operands are one. It returns 0, if both the bits are zero.

**For example,**

**or destination, source**

mov AX, FA

or AX,11

## The XOR Instruction

The XOR instruction implements the bitwise XOR operation. The XOR operation sets the resultant bit to 1, if and only if the bits from the operands are different. If the bits from the operands are same (both 0 or both 1), the resultant bit is cleared to 0.

For example,

**xor destination, source**

mov AX,02

xor AX, DD

## The NOT Instruction

The NOT instruction implements the bitwise NOT operation. NOT operation reverses the bits in an operand. The operand could be either in a register or in the memory.

For example,

**not destination**

not EB

**Exercise:**

1. **Write a program to transfer following values in the specified registers**

AL=04, BL=05

Using emu 8086 trainer, do not display the result using service routine only verify your answer with manual calculation and single step button

1. ADD AL, BL
2. SUB AL,BL
3. AND AL,BL
4. OR AL,BL
5. XOR AL,BL
6. NOT AL
7. **Write a program to transfer following values in the specified registers using debug tool and verify your answer with manual calculation.**

AL=50, BL=A0

Perform XOR ,OR &AND operations