Practical: 8

Aim: Write an Assembly language program for 8-bit &16-bit logical operations in 8086.

Theory:

This program is written in x86 assembly language. It demonstrates the use of basic bitwise logical operations (AND, OR, XOR, NOT) between two 8-bit registers (AL and BL). The program processes the data and outputs the final result in binary format, then waits for a key press before terminating.

Step-by-Step Explanation:

1. **Loading Initial Values into Registers:**
   - The program starts by assigning values to the AL and BL registers.
   - In this case:
     - AL = 2 (binary: 00000010)
     - BL = 3 (binary: 00000011)
2. **Logical Operations:**
   - The program performs a series of logical operations on the values in AL and BL:
   - **AND Operation**:
     - BL = BL AND AL
     - This compares each bit of BL and AL, resulting in a 1 only if both bits are 1.
     - Result: 00000010 AND 00000011 = 00000010 (result in BL is 00000010)
   - **OR Operation**:
     - BL = BL OR AL
     - This compares each bit of BL and AL, resulting in 1 if either of the bits is 1.
     - Result: 00000010 OR 00000010 = 00000010 (no change)
   - **XOR Operation**:
     - BL = BL XOR AL
     - This compares each bit of BL and AL, resulting in 1 if the bits differ.
     - Result: 00000010 XOR 00000010 = 00000000
   - **NOT Operation**:
     - BL = NOT BL
     - This inverts each bit in BL (turns 0 into 1 and 1 into 0).
     - Result: NOT 00000000 = 11111111
3. **Printing the Result in Binary:**
   - After performing the logical operations, the final result in BL is printed in binary form.
   - The program uses a loop to print each bit, starting from the most significant bit (MSB).
   - The test instruction checks if the MSB is 1 or 0, and the corresponding value is printed.
   - The shl instruction shifts the bits of BL to the left to check the next bit.

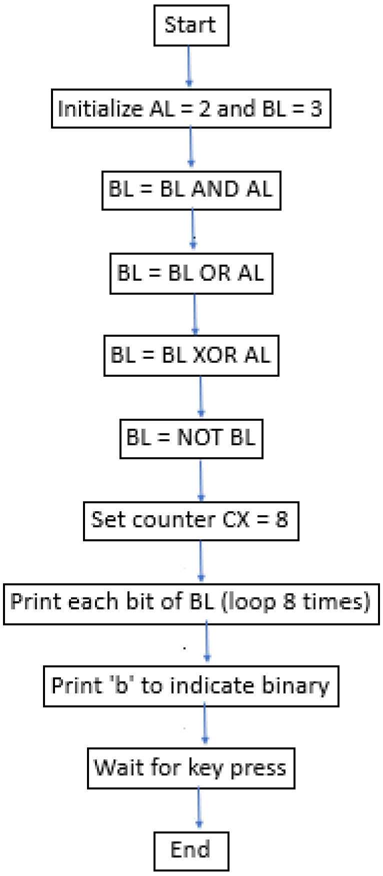
4. **Printing the Binary Suffix:**
   - After printing all 8 bits, the program appends the letter 'b' to indicate that the output is in binary format.

5. **Waiting for Key Press:**
    - Before exiting, the program waits for a key press to ensure that the output is visible until the user interacts.
6. **End of Program:**
    - The program ends with the ret instruction, returning control to the operating system.

Algorithm for Assembly Program: Logical Operations and Binary Output

1. **Start**
2. **Initialize Registers:**
    - Load the value 2 into register AL.
    - Load the value 3 into register BL.
3. **Perform Bitwise AND Operation:**
    - Apply the bitwise AND between BL and AL.
    - Store the result in BL.
4. **Perform Bitwise OR Operation:**
    - Apply the bitwise OR between BL and AL.
    - Store the result in BL.
5. **Perform Bitwise XOR Operation:**
    - Apply the bitwise XOR between BL and AL.
    - Store the result in BL.
6. **Perform Bitwise NOT Operation:**
    - Apply the bitwise NOT to BL.
    - Invert all bits in BL.
7. **Prepare for Binary Output:**
    - Set the loop counter CX to 8 (for 8 bits).
8. **Print Each Bit of BL:**
    - For each iteration (8 iterations):
        - Test the most significant bit (MSB) of BL.
        - If the MSB is 1, print 1; otherwise, print 0.
        - Shift BL left by one bit to prepare for the next bit.
9. **Print Binary Suffix:**
    - After printing all 8 bits, append the character 'b' to indicate binary format.
10. **Wait for Key Press:**
    - Wait for the user to press any key to proceed.
11. **End**

Flowchart for Algorithm:



Conclusion: Hence, we have implemented an Assembly language program for logical operations in 8086.