Microprocessor and Assembly Language CSC-321

Sheeza Zaheer

Lecturer

COMSATS UNIVERSITY ISLAMABAD

LAHORE CAMPUS

Logic Instructions

OUTLINE

- **Logic Instructions**
  - AND
  - OR
  - XOR
  - NOT
  - TEST
- **References**
  - **Chapter 7, Section 7.1,** Ytha Yu and Charles Marut, “Assembly Language Programming and Organization of IBM PC

Logic Instructions

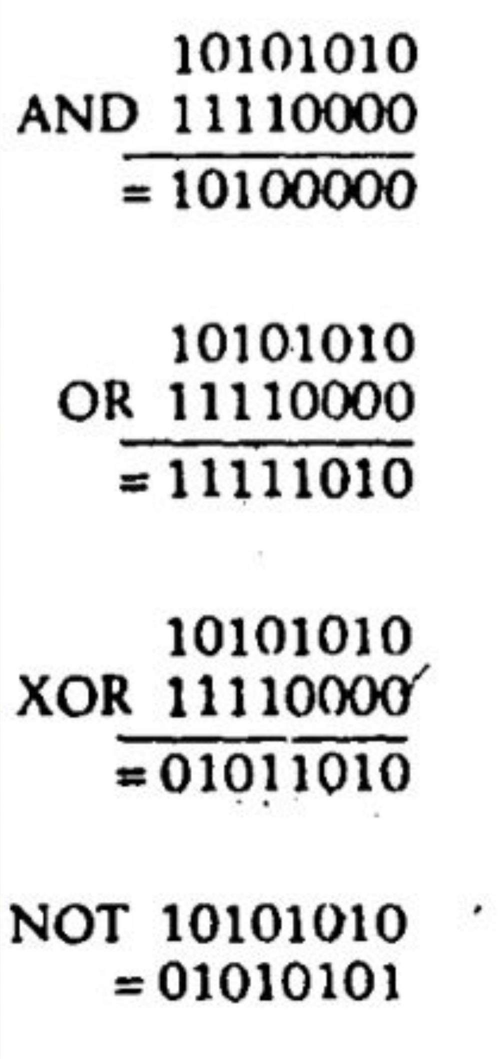
- To manipulate individual bits
- Binary Value 0 treated as false
- Binary Value 1 treated as true
- In Assembly Language:
  - AND
  - OR
  - XOR
  - NOT
  - TEST

Truth Tables

a	b	a AND b	a OR b	a XOR b
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

a	NOT a
0	1
1	0

Examples



```
          10101010
AND   11110000
      = 10100000

          10101010
  OR   11110000
      = 11111010

          10101010
XOR   11110000
      = 01011010

NOT  10101010
      = 01010101
```

Syntax

AND *destination, source*

OR *destination, source*

XOR *destination, source*

- Destination:
  - Stores result
  - Can be Register or Memory Location
- Source:
  - May be a Constant, Register or Memory Location
- Memory to memory operation not allowed

Effects on Flags

- SF, ZF, PF reflects the result
- AF is undefined
- CF, OF = 0

MASK

- To modify only selective bits in destination, we construct a source bit pattern known as **MASK**.
- To choose mask, use following properties:
  - b AND 1 = b (e.g. 0 AND 1 = 0 , 1 AND 1 = 1)
  - b AND 0 = 0 (e.g. 0 AND 0 = 0 , 1 AND 0 = 0)
  - b OR 1 = 1 (e.g. 0 OR 1 = 1 , 1 OR 1 = 1)
  - b OR 0 = b (e.g. 0 OR 0 = 0 , 1 OR 0 = 1)
  - b XOR 0 = b (e.g. 0 XOR 0 = 0 , 1 XOR 0 = 1)
  - b XOR 1 = ~b (complement of b) (e.g. 0 XOR 1 = 1 , 1 XOR 1 = 0)

Where b represents a bit (0 or 1)

AND Instruction

The AND instruction:

- May be used to **clear** specific destination bits while preventing the others.

- A 0 mask bit clears the corresponding destination bit.

- A 1 mask bit preserves the corresponding destination bit.

Example 1

- **Clear the sign bit of AL while leaving the other bits unchanged.**
- *Solution*:

  AND AL, 7Fh

Where 7Fh (0111 1111) is the mask.

Suppose,

AL = 1000 1010

AND 0111 1111

$\overline{0000\ 1010}$

OR Instruction

The OR instruction:

- May be used to **set** specific destination bits while preventing the others.

- A 1 mask bit sets the corresponding destination bit.

- A 0 mask bit preserves the corresponding destination bit.

Example 2

- **Set the MSB and LSB of AL while preserving the other bits.**
- *Solution*:

  OR AL, 81h

Where 81h (1000 0001) is the mask.

Suppose,

```
AL =    1000 1010
OR      1000 0001
        _________
        1000 1011
```

XOR Instruction

The XOR instruction:

- May be used to **complement** specific destination bits while preventing the others.

- A 1 mask bit complements the corresponding destination bit.

- A 0 mask bit preserves the corresponding destination bit.

Example 3

- **Change the sign bit of DX.**
- *Solution:*

XOR DX, 8000h

Where 80h (1000 0000) is the mask.

Suppose,

DX =   1000 1010 0001 1011

XOR   1000 0000 0000 0000

0000 1010 0001 1011

Converting an ASCII digit to a Number

- ASCII code for digit “0-9” is “30h-39h”

AND AL, CFh ;Clears the high nibble.

Suppose,

```
AL =      0011 0101
AND      1100 1111
         _________
          0000 0101
```

- *How to convert decimal digit to ASCII code?*

Converting a Lowercase letter to Uppercase

- Lower case: 61h to 7Ah
- Uppercase: 41h to 5Ah
- Lower to upper case, only clear bit 5. So, the mask is 1101 1111b (0DFh)

  AND DL, 0DFh

Suppose,

```
DL =     0110 0101
AND      1101 1111
         ---------
         0100 0101
```

Clearing a Register

```
MOV AX, 0     ;machine code 3 bytes
```

OR

```
SUB AX, AX    ;machine code 2 bytes
```

OR

```
XOR AX, AX    ;machine code 2 bytes
```

Testing a Register for zero

```
CMP CX, 0
```

Is same like:

```
OR CX, CX    ;sets ZF = 1 if CX is 0
```

NOT Instruction

- Performs the one’s complement operation on the destination.
- Syntax:

  NOT *destination*
- No effect on flags
- Example: Complement the bit in AX:

  NOT *AX*

TEST Instruction

- Performs an AND operation without changing destination i.e. only status flags updated.
- Syntax:

  TEST *destination, source*
- Effects on flags:
  - SF, ZF and PF reflects the results
  - AF is undefined
  - CF, OF = 0

Contd..

- Examining the individual bits:

  TEST *destination*, *mask*

- If destination have all zero, then ZF = 1
- The tested bit position is 1 if and only if corresponding source bit is 1
- **Example**: Jump to label BELOW if AL contains an even number.
- *Solution*: Even numbers have 0 at bit 0 so the mask is 0000 0001

  TEST *AL, 1*

  JZ BELOW

e.g.

2= 0010, 3 = 0011, 4 = 0100, 5 = 0101, 6 = 0110, 7 = 0111

Suppose,

```
AL =     0111 0100
AND     0000 0001
        0000 0000
```