CPNM Lecture 18 - Bit Wise Operators

Mridul Sankar Barik

Jadavpur University

2022-23

Bit Wise Operators I

- Logical Operators
  - Bitwise AND (&): The bits in the result are set to 1 if the corresponding bits in the two operands are both 1

    ```
    unsigned char a = 5, b = 1, c;
    c = a & b; /*c gets 1*/
    ```

  - Bitwise OR (|): The bits in the result are set to 1 if at least one of the corresponding bits in the two operands is 1

    ```
    unsigned char a = 5, b = 2, c;
    c = a | b; /*c gets 7*/
    ```

  - Bitwise Exclusive OR (∧): The bits in the result are set to 1 if exactly one of the corresponding bits in the two operands is 1

    ```
    unsigned char a = 5, b = 1, c;
    c = a ^ b; /*c gets 4*/
    ```

  - Bitwise NOT (˜): All 0 bits are set to 1 and all 1 bits are set to 0 (One's complement)

Bit Wise Operators II

```
unsigned char a = 1, b;
b = ~a; /*b gets 254*/
```

- Shift Operators
  - Right shift (>>): Shifts the bits of the first operand right by the number of bits specified by the second operand; the method of filling from the left is machine dependent.

    ```
    unsigned char a = 5, b;
    b = a >> 1; /*b gets 2*/
    ```

    Right shift by one bit divides a number by 2
  - Left shift (<<): Shifts the bits of the first operand left by the number of bits specified by the second operand; fill from right with 0 bits

    ```
    unsigned char a = 5, b;
    b = a << 1; /*b gets 10*/
    ```

    Left shift by one bit multiplies a number by 2

Bit Wise Operators III

- Example

```
/* print powers of 2*/
unsigned int a = 1, i;
for(i = 0; i < 32; i++)
    printf("2^%d = %u\n", i, a<<i);
```

- Example

```
/* print bits of an unsigned char*/
unsigned char uc=7;
unsigned char mask=128;
int i = 0;
for(i = 0; i<8; i++){
    if((uc&mask)>0)
        printf("1");
    else
        printf("0");
    mask=mask>>1;
}
```