**BITWISE OPERATOR**

Bitwise operator are operators which act on the individual bits of the data stored in the memory location. The operators that are used to manipulate the bits of given data are called bitwise operators.

There are 6 bitwise operators: as shown in the previous table

|  |  |  |
| --- | --- | --- |
| **BITWISE OPERATORS** | **SYMBOLS USED** | **USAGE** |
| BITWISE AND | & | a & b |
| BITWISE OR | | | a | b |
| BITWISE XOR | ^ | a ^ b |
| Left Shift Operator | << | a<<2 |
| Right Shift Operator | >> | a>>2 |
| One’s Complement | ~ | ~a |

The Bitwise operations of AND, OR and XOR is given below

|  |  |  |
| --- | --- | --- |
| AND | OR | XOR(^) |
| 0&0=0 | 0|0=0 | 0^0=0 |
| 0&1=0 | 0|1=1 | 0^1=1 |
| 1&0=0 | 1|0=1 | 1^0=1 |
| 1&1=1 | 1|1=1 | 1^1=0 |

These operators are used to perform logical operation (and, or, not) on individual bits of a binary number.

* Ex for ~ (bitwise complement) Ex for & (bitwise AND)

a=13 0000 1101 a=13 0000 1101

~a= 1111 0010 b=6 0000 0110

a & b 0000 0100

* Ex for || (bitwise OR) Ex for ^ (bitwise XOR)

a=13 0000 1101 a=13 0000 1101

b=6 0000 0110 b=6 0000 0110

a | b 0000 1111 a ^ b 0000 1011

The operator that is used to shift the data by a specified number of bit positions towards left or right is called 9shift operator.

* The syntax is shown below for<<

b=a << num;

where a is value to be shifted

num is number of bits to be shifted

Ex for << (left shift):

a=13 0000 1101

b=a<<1 0001 1010

* The syntax is shown below for>>

b=a>> num;

Ex for >>(right shift):

a=13 0000 1101

b=a<<1 0000 0110

**TERNARY OPERATOR**

Ternary operator which is also known as conditional operator, operates on three operands. Ternary operator return one value if condition is true and returns another value if condition is false.

The syntax is shown below:

(exp 1)? exp2:exp3;

Where exp 1 is an expression evaluated to true or false;

If exp 1 is evaluated to true, exp 2 is executed;

If exp 1 is evaluated to false, exp3 is executed.

Example: a=10; b=15;

X=(a>b)? a:b;