C++ Bitwise Operators - Concepts & Examples

Bitwise Operators in C++

1. AND (&)

- Description: Returns 1 if both bits are 1.
- Example: 5 & 3 -> 0101 & 0011 = 0001 -> Result = 1

2. OR (|)

- Description: Returns 1 if at least one of the bits is 1.
- Example: 5 | 3 -> 0101 | 0011 = 0111 -> Result = 7

3. XOR (^)

- Description: Returns 1 if bits are different.
- Example: 5 ^ 3 -> 0101 ^ 0011 = 0110 -> Result = 6

4. NOT (~)

- Description: Inverts the bits (1's complement).
- Example: ~5 -> In binary: 00000101 -> Inverted: 11111010 -> Result: -6

5. Left Shift (<<)

- Description: Shifts bits to the left by given number of positions. Adds 0s on the right.
- Example: 5 << 1 -> 0101 -> 1010 -> Result = 10
- Equivalent to: 5 * 2^1 = 10

6. Right Shift (>>)

- Description: Shifts bits to the right by given number of positions.
- Example: 10 >> 1 -> 1010 -> 0101 -> Result = 5
- Equivalent to: 10 / 2^1 = 5

7. Check if a number is Power of 2 (without loop):
- Formula: (n > 0) && ((n & (n - 1)) == 0)
- Example: 8 -> 1000 & 0111 = 0000 -> Power of 2 -> True

8. Reversing an Integer using %, /:
- Extract last digit using: digit = n % 10
- Build reversed number: reversed = reversed * 10 + digit
- Remove last digit: n = n / 10

These bitwise operations are very fast and useful for performance-critical or low-level coding.