Bitwise Operators in C

Diego Gonzalez Ayala

Bitwise AND (&) operator

- It takes two bits at a time and performs **AND** operation
- **AND (&)** is a binary operator
- Result of **AND (&)** is 1 when both bits are 1

A	B	A&B
0	0	0
0	1	0
1	0	0
1	1	1

Bitwise OR (|) operator

- It takes two bits at a time and performs **OR** operation
- **OR (|)** is a binary operator
- Result of **OR (|)** is 0 when both bits are 0

A	B	A|B
0	0	0
0	1	1
1	0	1
1	1	1

Bitwise NOT (~) operator

- It takes two bits at a time and performs **NOT** operation
- **NOT (&)** is a binary operator
- Result of **AND (&)** is 1 when both bits are 1

A	~A
1	0
1	0
1	0
1	0

Bitwise LEFT SHIFT (<<) operator

- It takes two operands at a time to perform **LEFT SHIFT (<<)** operation
- First operand --> Whose bits get **left shifted**
- Second operand --> Decides the number of places to shift the bits
- **LEFT SHIFT (<<)** is equivalent to multiply the first operand by **2^{RightOperand}**

|Example:

Operation: x<<1

- Let **x** = 3
- Binary representation of x: **0000 0011**

Operation result:

- **x<<1** = 6
- Binary representation of x<<1: **0000 1100**
- Mathematical representation: **3*2^1 = 6**

Bitwise RIGHT SHIFT (>>) operator

- It takes two operands at a time to perform **RIGHT SHIFT (>>)** operation
- First operand --> Whose bits get **RIGHT shifted**
- Second operand --> Decides the number of places to shift the bits
- **RIGHT SHIFT (>>)** is equivalent to divide the first operand by **2^{RightOperand}**

|Example:

Operation: x>>1

- Let **x** = 10
- Binary representation of x: **0000 1010**

Operation result:

- **x>>1** = 5
- Binary representation of x<<1: **0000 0101**
- Mathematical representation: **10/2^1 = 5**