Go Bitwise Operations Cheatsheet

Generated by smallnest

1 Basic Operations

```
// AND
0b0001 & 0b0010 = 0000

// OR
0b0001 | 0b0010 = 0011

// XOR
0b0001 ^ 0b0010 = 0011

// NOT (for int8)
^0b0010 = 11111101

// Bitclear (AND NOT)
0b0011 &^ 0b0010 = 0001

// Left Shift
1 << 2 = 4  // 1 * 2^2

// Right Shift
1 >> 2 = 0  // 1 / 2^2
```

2 Bit Manipulation Functions

```
import "math/bits"

// Count ones
bits.OnesCount8(0b00101110) = 4

// Count significant bits
bits.Len8(0b00101110) = 6
bits.Len8(0b00000000) = 0

// Count leading zeros
bits.LeadingZeros8(0b00101110) = 2
bits.LeadingZeros8(0b00000000) = 8

// Count trailing zeros
bits.TrailingZeros8(0b00101110) = 1
bits.TrailingZeros8(0b00000000) = 8

// Rotate left
bits.RotateLeft8(0b00101110, 3) = 01110001

// Rotate right
bits.RotateLeft8(0b00101110, -3) = 11000101

// Reverse bits
bits.Reverse8(0b00101110) = 01110100

// Reverse bytes
bits.ReverseBytes16(0x00ef) = 0xef00
```

3 Integer Arithmetic Tricks

```
// Multiply by 2^n
x = y << n // 1<<8 = 256

// Divide by 2^n
x = y >> n // 256>>8 = 1

// Check if x is even
(x & 1) == 0 // 256&1 = 0

// Check if x is a power of 2
x != 0 && x&(x-1) == 0 // 256&(256-1)= 0

// Check if a number is divisible by 8
((n >> 3) << 3) == n

// Check if x and y have opposite signs
(x ^ y) < 0 // 256^(-256) < 0
```

4 Single Bit Operations

```
    // Set the nth bit
    y = x | (1 << n) // 10000000 | (1<<3) = 10001000

    // Unset the nth bit
    y = x &^ (1 << n) // 01111111 &^ (1<<3) = 01110111

    // Toggle the nth bit
    y = x ^ (1 << n) // 10000000 ^ (1<<3) = 10001000

    // Toggle all bits except nth bit
    y = ^(x ^ (1 << n)) // ^(10000000 ^ (1<<3)) = 01110111

    // Toggle rightmost m bits
    y = x ^ ^(-1 << n) // 10000000 ^^ (-1<<3) = 10000111

    // Test if the nth bit is set
    (x & (1 << n)) != 0 // 10000000 & (1<<3) = 00000000

    // Turn off rightmost 1-bit
    y = x & (x - 1) // 01111111 & (128-1) = 01111110

    // Isolate rightmost 1-bit
    y = x & (-x) // 01111000 & (-120) = 00001000

    // Right propagate rightmost 1-bit
    y = x | (x - 1) // 01011000 | (88-1) = 01011111

    // Turn on rightmost 0-bit
    y = x | (x + 1) // 01011000 | (88+1) = 01011001

    // Isolate rightmost 0-bit
    y = ^x & (x + 1) // ^00001001 & (88+1) = 00000010
```

Follow me