**Bit Hack #1. Check if the integer is even or odd**

if ((x & 1) == 0) {

x is even

}

else {

x is odd

}

**Bit Hack #2. Test if the n-th bit is set to 1**

if (x & (1<<n)) {

n-th bit is set

}

else {

n-th bit is not set

}

**Bit Hack #3. Set the n-th bit to 1**

y = x | (1<<n)

**Bit Hack #4. Unset the n-th bit => set it to 0**

y = x & ~(1<<n)

**Bit Hack #5. Toggle the n-th bit. => flip; 1=>0; 0=>1**

y = x ^ (1<<n)

**Bit Hack #6. Turn off the rightmost 1-bit**

y = x & (x-1)

**Bit Hack #7. Isolate the rightmost 1-bit**

y = x & (-x)

**Bit Hack #8. Right propagate the rightmost 1-bit**

y = x | (x-1)

**Bit Hack #9. Isolate the rightmost 0-bit**

y = ~x & (x+1)

**Bit Hack #10. Turn on the rightmost 0-bit**

y = x | (x+1)

<http://www.catonmat.net/blog/low-level-bit-hacks-you-absolutely-must-know/>