Bitwise Not

~ is a bitwise inversion operator and it acts exactly as defined:

The bitwise inversion of x is defined as -(x+1).

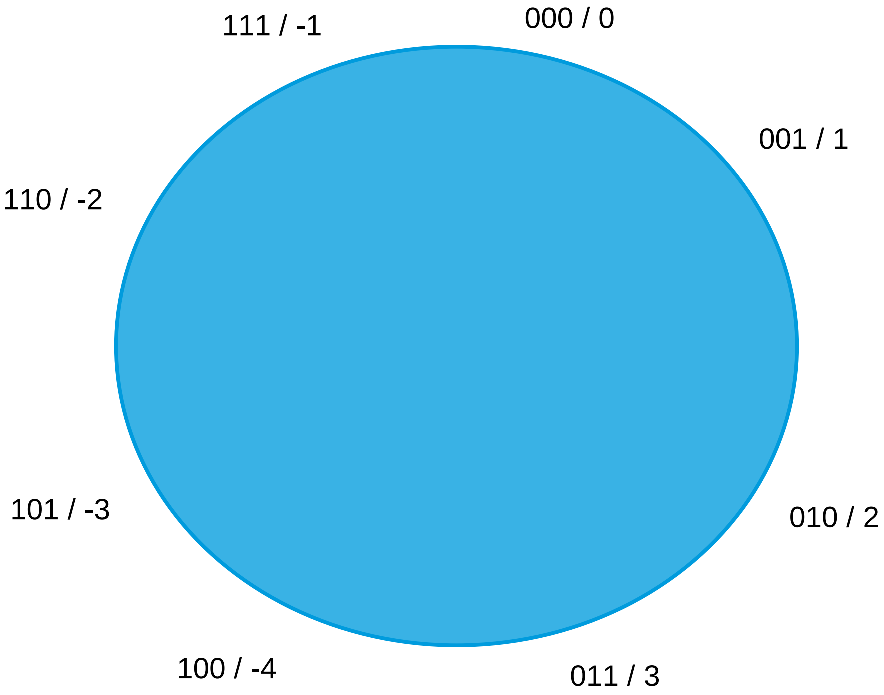
This is simply how the bitwise inversion of the two's complement representation of an integer works.

The two's complement wheel visualizes this pretty well: PTO

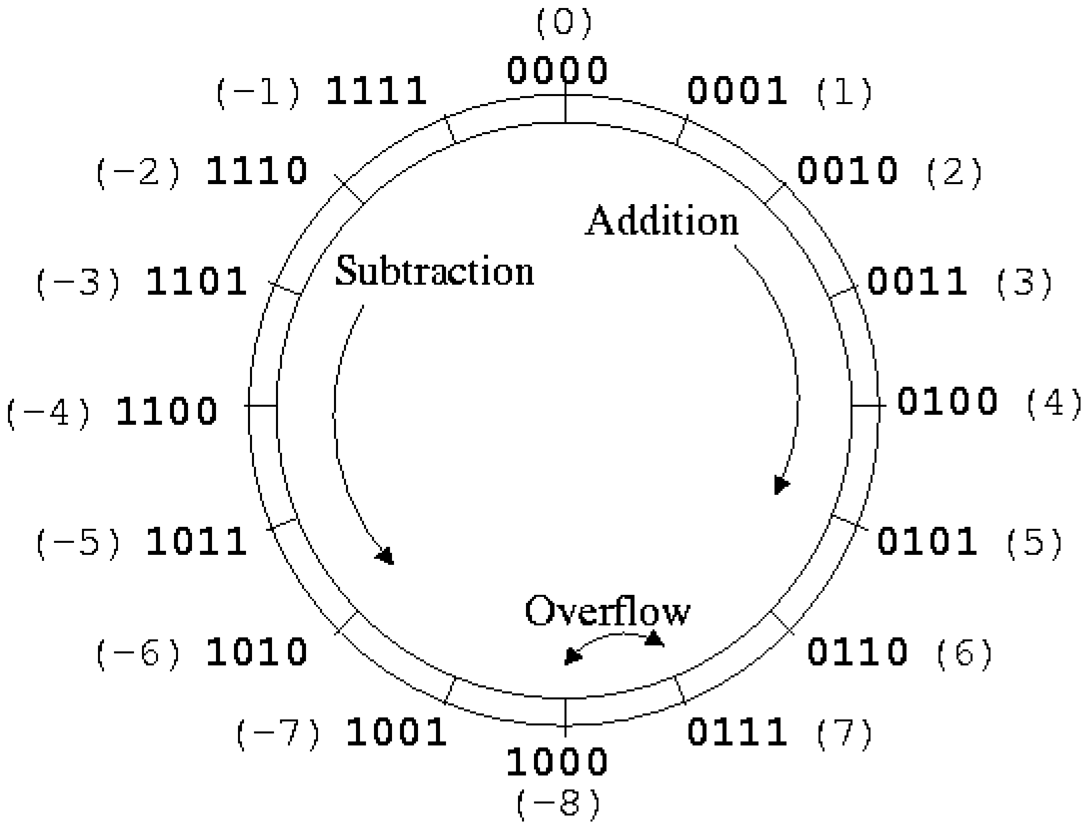
As you can see, the bitwise inversion of 1 is -2, the bitwise inversion of 2 is -3, ..., and the bitwise inversion of 60 will be -61.

Bitwise ~

When x is an integer:
~x = -(x+1)
Below we are assuming register size is of 3 bits.



When register size is of 4 bits:



Bitwise Not

A simple rule to remember the bitwise NOT operation on integers is -(x+1).

```
print(~60) # Outputs -61
print(~-60) # Outputs 59
```

How integers are stored in memory?

Binary equivalent of 65 is (1000001) 2 .

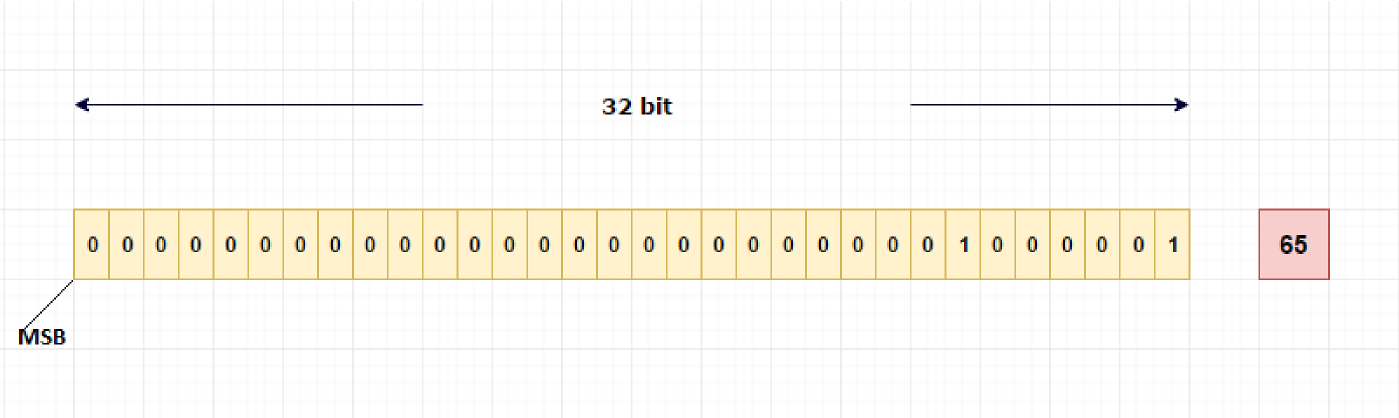
The MSB (most significant bit) bit is used to indicate whether the number is positive or negative.

For positive numbers MSB will be 0.

For negative numbers MSB will be 1.

In our case, 65 is positive so MSB will be 0.

This binary equivalent of 65 will be stored in 32-bit memory like below,



How negative integers are stored in memory?

Before going to that, let’s have a look on 1’s complement of a number.
1’s complement of a number

1’s compliment of number is just inverting binary bits of an actual number.

Example:

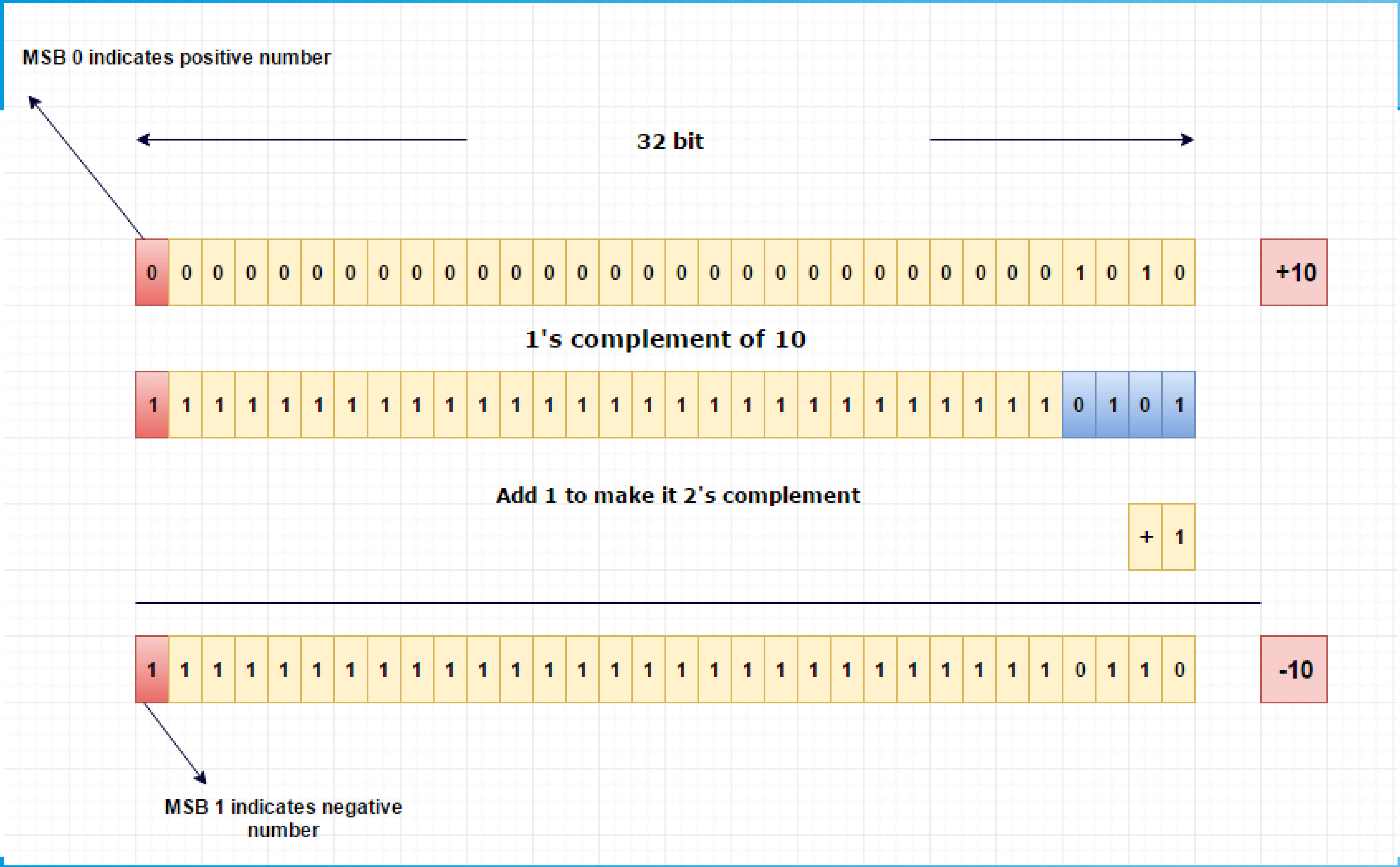
10

Binary representation will be (1010) 2

1’s compliment of 10 is, 0101 (switching 0 to 1 and 1 to 0).
2’s complement of a number

To get 2’s complement of a number, just add 1 to 1’s complement of an actual number.

Pictorial Explanation: PTO


MSB 0 indicates positive number
32 bit
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0
+10
1's complement of 10
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 1
Add 1 to make it 2's complement
+ 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0
-10
MSB 1 indicates negative number

Ques: If we print the value of ~4 using printf, it will print -5. How?

Ans: Let's take a look on how -5 will be stored in computer?

As we discussed earlier negative numbers are represented using 2’s complement form.

To know how to calculate 2’s complement of a number, kindly refer the link.How integers are stored in memory

So, -5 will be stored like below,

5 = (00000101)

1’s complement of 5 = (11111010)

Add 1 to get the 2’s complement,

-5 = (11111011)

Result of ~4 is equal to the representation of -5 in binary, hence ~4 is -5.

Example 2

Let’s take number 2,

2 = (00000010) 2

1’s complement of 2 = (11111101) 2

Add 1 to get the 2’s complement,

~2 = (11111101) 2

Result of ~2 is equal to the representation of -3 in binary, hence ~2 is -3.

In Code

```
>>> ~0
-1
>>> ~1
-2
>>> ~2
-3
>>> ~3
-4
>>> ~-1
0
>>> ~-2
1
>>> ~-3
2
>>> ~-4
3
```