Operations in Binary

Binary Addition

Now that we know binary numbers, we will learn how to add them. Binary addition is much like your normal everyday addition (decimal addition), except that it carries on a value of 2 instead of a value of 10.
For example: in decimal addition, if you add 8 + 2 you get ten, which you write as 10; in the sum this gives a digit 0 and a carry of 1. Something similar happens in binary addition when you add 1 and 1; the result is two (as always), but since two is written as 10 in binary, we get, after summing 1 + 1 in binary, a digit 0 and a carry of 1.

Therefore in binary:

No. of state	A + B	Carry	Sum
0	0 + 0	0	0
1	0 + 1	0	1
2	1 + 0	0	1
3	1 + 1	1	0

Example. Suppose we would like to add two binary numbers 10 and 11. We start from the last digit. Adding 0 and 1, we get 1 (no carry). That means the last digit of the answer will be one. Then we move one digit to the left: adding 1 and 1 we get 10. Hence, the answer is 101. Note that binary 10 and 11 correspond to 2 and 3 respectively. And the binary sum 101 corresponds to decimal 5: is the binary addition corresponds to our regular addition.

Problem: 100101 + 10101 = ?.
Answer: 100101 + 10101 = 111010.

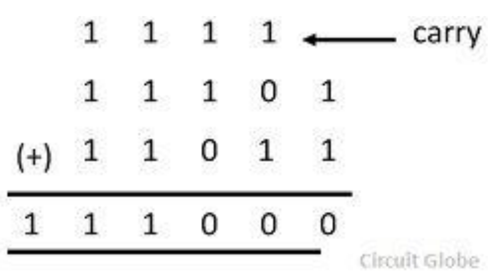
Explanation:

			1		1	
	1	0	0	1	0	1
+		1	0	1	0	1
	1	1	1	0	1	0

first column (from the right!) : 1 + 1 = 0 carry 1
second column : 0 + 0 + 1 (carried) = 1
third column : 1 + 1 + no carry = 0 carry 1
fourth column : 0 + 0 + 1 (carried) = 1
fifth column : 0 + 1 + no carry = 1
sixth column : 1 + 0 + no carry = 1

Therefore the answer is 1 1 1 0 1 0!

Example :



Example :

00011010 + 00001100 = 00100110

```
          1   1                       carries
    0  0  0  1  1  0  1  0   =   26(base 10)
  + 0  0  0  0  1  1  0  0   =   12(base 10)
  ------------------------
    0  0  1  0  0  1  1  0   =   38(base 10)
```

00010011 + 00111110 = 01010001

```
       1  1  1  1  1                  carries
    0  0  0  1  0  0  1  1   =   19(base 10)
  + 0  0  1  1  1  1  1  0   =   62(base 10)
  ------------------------
    0  1  0  1  0  0  0  1   =   81(base 10)
```

Example :

1011.01+11.011= 1110.101

```
      11   1
     1011.01
+      11.011
-------------------------
     1110.101
```

Subtraction in Binary

Subtraction and Borrow, these two words will be used very frequently for the binary subtraction. There are four rules of binary subtraction.

No. of state	A - B	Borrow	Subtract
0	0 - 0	0	0
1	0 - 1	1	0
2	1 - 0	0	1
3	1 - 1	0	0

Example :

0011010-001100=00001110

```
            0  10
               0  10              Borrow
  0   0  ~~1~~  ~~1~~  0  1  0    =26
  0   0  0  1  1  0  0            =12

-----------------------------------------
  0   0  0  1  1  1  0          = 14
```

Example :

```
            0  10         Borrow
        1  ~~1~~  ~~0~~   0     =12
   -  1  0  1    0        =10

-----------------------------------------
        0  0  1  0        =2
```

Example :

```
 0  0 1 10 10
 10101.101
-  1011.11
 ----------
   1001.111
```

Example:

```
              1  10  10
           0  10  0      0  10        borrow
   1   0   1   0  1  .  1  0  1
-      1   0   1  1  .  1  1
-------------------------------------
       1   0   0  1  .  1  1  1
```

Multiplications in Binary :

Binary multiplication is similar to decimal multiplication. It is simpler than decimal multiplication because only 0s and 1s are involved. There are four rules of binary multiplication.

Case	A x B	Multiplication
1	0 x 0	0
2	0 x 1	0
3	1 x 0	0
4	1 x 1	1

Example:

0011010 x 001100 = 100111000

```
      0011010   = 26₁₀
     x0001100   = 12₁₀
    ----------
      0000000
     0000000
    0011010
   0011010
   ----------
   0100111000   = 312₁₀
```

Binary Division

The binary division operation is similar to the base 10 decimal system, except the base 2. The division is probably one of the most challenging operations of the basic arithmetic operations. There are different ways to solve division problems using binary operations. Long division is one of them and the easiest and the most efficient way.

Binary Division Rules

The binary division is much easier than the decimal division when you remember the following division rules. The main rules of the binary division include:

- 1÷1 = 1
- 1÷0 = Meaningless
- 0÷1 = 0
- 0÷0 = Meaningless

Similar to the decimal number system, the binary division is similar, which follows the four-step process:

- Divide
- Multiply
- Subtract
- Bring down

Important Note: Binary division follows the long division method to find the resultant in an easy way.

Comparison with Decimal Value

$(01111100)_2 = (1111100)_2 = 124_{10}$

$(0010)_2 = (10)_2 = 2_{10}$

You will get the resultant value as 62 when you divide 124 by 2.

So the binary equivalent of 62 is $(111110)_2$

$(111110)_2 = 62_{10}$

Both the binary and the decimal system produce the same result.

Binary Division Examples

Example 1.

Question: Solve 01111100 ÷ 0010

Solution:

Given

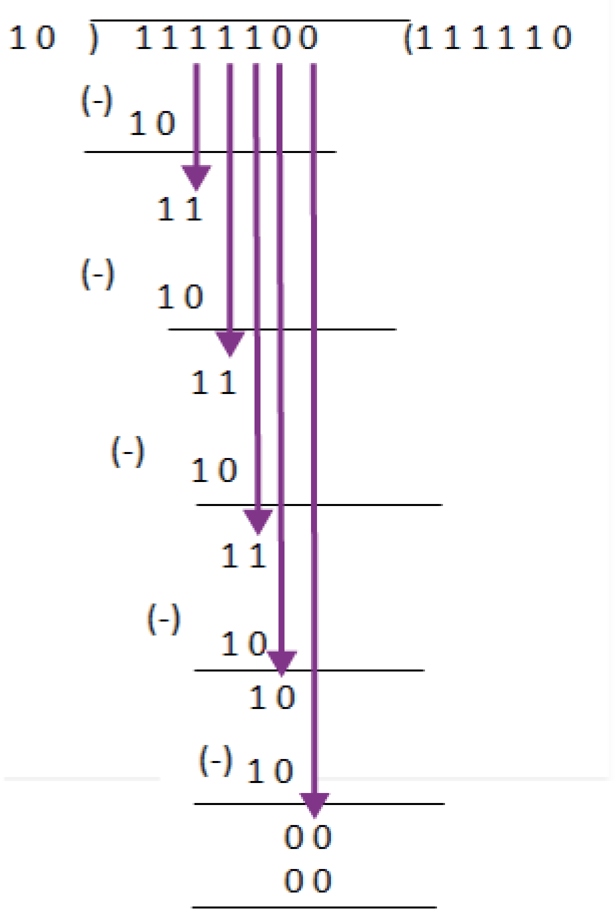
01111100 ÷ 0010

Here the dividend is 01111100, and the divisor is 0010

Remove the zero's in the **Most Significant Bit** in both the dividend and divisor, that doesn't change the value of the number.

So the dividend becomes 1111100, and the divisor becomes 10.

Now, use the long division method.



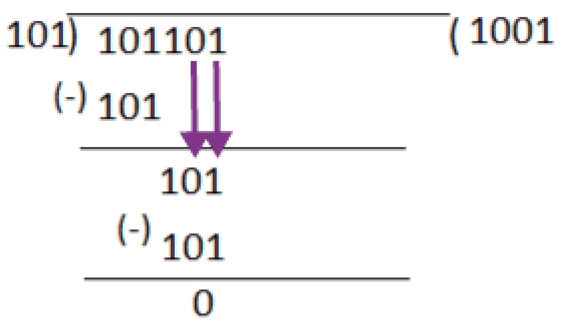
- **Step 1:** First, look at the first two numbers in the dividend and compare with the divisor. Add the number 1 in the quotient place. Then subtract the value, you get 1 as remainder.
- **Step 2:** Then bring down the next number from the dividend portion and do the step 1 process again
- **Step 3:** Repeat the process until the remainder becomes zero by comparing the dividend and the divisor value.
- **Step 4:** Now, in this case, after you get the remainder value as 0, you have zero left in the dividend portion, so bring that zero to the quotient portion.

Therefore, the resultant value is quotient value which is equal to 111110

So, 01111100 ÷ 0010 = 111110

Example 2: Solve using the long division method: 101101 ÷ 101

Solution:



Representation methods

- Signed Magnitude Representation

The signed magnitude (also referred to as sign and magnitude) representation is most familiar to us as the base 10 number system. A plus or minus sign to the left of a number indicates whether the number is positive or negative as in $+12_{10}$ or -12_{10}. In the binary signed magnitude representation, the leftmost bit is used for the sign, which takes on a value of 0 or 1 for '+' or '–', respectively. The remaining bits contain the absolute magnitude.

Consider representing $(+12)_{10}$ and $(-12)_{10}$ in an eight-bit format:

B7	B6	B5	B4	B3	B2	B1	B0

B7 B6 B5 B4 B3 B2 B1 B0

B7 : for the sign of number

If the number is (+) => B7=0

If the number is (-) => B7=1

B0-B6: Is for the magnitude

$(+12)_{10} = (00001100)_2$

B7	B6	B5	B4	B3	B2	B1	B0
0	0	0	0	1	1	0	0

$(-12)_{10} = (10001100)_2$

B7	B6	B5	B4	B3	B2	B1	B0
1	0	0	0	1	1	0	0

The negative number is formed by simply changing the sign bit in the positive number from 0 to 1. Notice that there are both positive and negative representations for zero: +0= 00000000 and -0= 10000000.

تمثيل الرقم ٠ باستخدام طريقة المقدار و الاشارة يكون كالتالي

B7	B6	B5	B4	B3	B2	B1	B0
٠	٠	٠	٠	٠	٠	٠	٠

B7	B6	B5	B4	B3	B2	B1	B0
١	٠	٠	٠	٠	٠	٠	٠

- One's Complement Representation

The one's complement operation is trivial to perform: convert all of the 1's in the number to 0's, and all of the 0's to 1's. We can observe that in the one's complement representation the leftmost bit is 0 for positive numbers and 1 for negative numbers, as it is for the signed magnitude representation. This negation, changing 1's to 0's and changing 0's to 1's, is known as complementing the bits. Consider again representing (+12)10 and (–12)10 in an eight-bit format, now using the one's complement representation:

(+12)10 = (00001100)2

(–12)10 = (11110011)2

Note again that there are representations for both +0 and –0, which are 00000000 and 11111111, respectively. As a result, there are only $2^8 - 1 = 255$ different numbers that can be represented even though there are 2^8 different bit patterns.

The one's complement representation is not commonly used. This is at least partly due to the difficulty in making comparisons when there are two representations for 0. There is also additional complexity involved in adding numbers.

- Two's Complement Representation

The two's complement is formed in a way similar to forming the one's complement: **complement all of the bits in the number, but then add 1, and if that addition results in a carry-out from the most significant bit of the number, discard the carry-out.**

Examination of the fifth column of Table above shows that in the two's complement representation, the leftmost bit is again 0 for positive numbers and is 1 for negative numbers. However, this number format does not have the unfortunate characteristic of signed-magnitude and one's complement representations: it has only one representation for zero. To see that this is true, consider forming the negative of $(+0)_{10}$, which has the bit pattern: $(+0)_{10} = (00000000)_2$

Forming the one's complement of $(00000000)_2$ produces $(11111111)_2$ and adding

1 to it yields $(00000000)_2$, thus $(-0)_{10} = (00000000)_2$. The carry out of the leftmost position is discarded in two's complement addition (except when detecting an overflow condition). Since there is only one representation for 0, and since all bit patterns are valid, there are 28 = 256 different numbers that can be represented.

Consider again representing $(+12)_{10}$ and $(-12)_{10}$ in an eight-bit format, this time using the two's complement representation.

Starting with $(+12)_{10} = (00001100)_2$,

١'s complement, or negate the number, producing $(11110011)_2$

Now add one

```
  1 1 1 1   0 0 1 1
+                 1
-------------------------------------
  1 1 1 1   0 1 0 0
```

producing $(11110100)_2$, and thus $(-12)_{10} = (11110100)_2$:

$(+12)_{10} = (00001100)_2$

$(-12)_{10} = (11110100)_2$

There is an equal number of positive and negative numbers provided zero is considered to be a positive number, which is reasonable because its sign bit is 0. The positive numbers start at 0, but the negative numbers start at −1, and so the magnitude of the most negative number is one greater than the magnitude of the most positive number. The positive number with the largest magnitude is +127, and the negative number with the largest magnitude is −128. There is thus no positive number that can be represented that corresponds to the negative of −128. If we try to form the two's complement negative of −128, then we will arrive at a negative number, as shown below:

$(-128)_{10} = (10000000)_2$
$(-128)_{10} = (01111111$
$(-128)_{10} + (+0000001)2$
$(-128)_{10}$ ————————)2
$(-128)10 = (10000000)_2$

(0) = (0 0 0 0 0 0 0 0)
1's complement = (1 1 1 1 1 1 1 1)
Add 1 = +(0 0 0 0 0 0 0 1)
The 2's complement = (0 0 0 0 0 0 0 0)

The two's complement representation is the representation most commonly used in conventional computers.

-Binary Subtraction Using 2's Complement

What is a 2 's Complement?
To implement this method for subtracting two binary numbers, the first step is to find the 2's complement of the number to be subtracted from another number. To get the 2's complement, first of all, 1's complement is found, and then 1 is added. The addition is the required 2's complement.
Suppose we need to find the 2's complement of the binary number 10010. First, find 1's complement. To find this, replace all 1 to 0 and all 0 to 1. Therefore, 1's complement of 10010 will be 01101. Add 1 to this, and we will get the 2's complement, i.e. 01110.

To learn how to subtract binary numbers using 2's complement, **which is the subtraction of a smaller number from a larger** number using 2's complement subtraction, the following steps are to be followed:

- **Step 1:** Determine the 2's complement of the smaller number
- **Step 2:** Add this to the larger number.
- **Step 3:** Omit the carry. Note that there is always a carry in this case.

The following example illustrates the above-mentioned steps:
Exampe: Subtract (1010)2 **from** (1111)2 **using 2's complement method.**
Ans:

- **Step 1:** 2's complement of (1010)2 is (0110)2.
- **Step 2:** Add (0110)2 to (1111)2.

This is shown below:

		1 1	
1 1 1 1	1 1 1 1	1 1 1 1	
- 1 0 1 0	+(-1 0 1 0)	+0 1 0 1	1's complement
		+ 1	1- bit
--------------	---------------	----------------	
0 1 0 1		1 0 1 0 1	

Omit (the leading 1)   The result (0 1 0 1)

Example: Subtract (1010)2 from(1000)2 using 2's complement.
Ans:

		1	
1 0 0 0	1 0 0 0	1 0 0 0	
- 1 0 1 0	+(-1 0 1 0)	+0 1 0 1	1's complement
		+ 1	1- bit
--------------	---------------	----------------	
0 1 0 1		1 1 1 0	

The result (wrong)

Subtraction Using r's Complement:

Let's say you want to subtract the number 01010100 from 11100011. We can do this using 2 's complement by simply doing the **subtraction using r's complement.**

Steps to Find r's Complement:

To find r's complement, add 1 to the calculated (r−1) 's complement.

Here is an example:

Q. Find the 7's and 8's complement of the number (563)8

- **Step 1:** Identify the base (or) radix. Here r=8.
- **Step 2:** Since 7 is the largest digit in the number system, subtract each digit of the given number from 7, i.e. if it's a three-digit number, subtract the number from 777.

∴(214)8 is the 7's complement of a given number

- **Step 3:** To find r's complement, i.e. 8's complement, then add ' 1 ' to the result of 7 's complement number.

∴(215)8 is the 8 's complement of the given number.

Solved Examples

Q 1. 10110 - 11010

Ans: 11010 has a 2s complement of (00101+1) or 00110.
Add the 2's complement to the minuend (10110+00110) or 11100.
Now taking its complement;
The solution is (00011+1)= - (00100)

Q 2. 10110-01111

Ans: 01111's 2s complement is 10001.
The minuend plus the complement of two (10110-10001) equals 100111.
The response is 00111.

Q 3. 0100-11101

Ans: 11101's 2s complement is 00011
The minuend plus the complement of two (10100- 00011) equals 10111.
Since there is no carry here, the response is 01001.

Q 4. 110101 - 101001

Ans: 101001's complement in 2 is 010111
(110101-010111) Add the minuend and the 2's complement to get 1001100.
Carry, the result's leftmost bit is a 1 and is ignored.
The response is 001100.

Practice Questions

Q 1. 1001 - 0100

Ans: 0101

Q 2. 0100 - 1011

Ans: 1011

Q 3. 0110 - 0100

Ans: 0010

Q 4. 10110- 11101

Ans: 00111

Q 5. 110-101

Ans: 001