Arithmetic Logical operation

chapter:3

Binary Arithmetic

1. Binary Addition
2. Binary Subtraction
3. Binary Multiplication
4. Binary Division
5. r's and r-1's complement method for decimal and binary number

Binary Addition

To make the binary math calculator perform the addition of binary numbers, follow the steps below:

1. **Input the first number** in the first field of the binary addition calculator. Remember to **use only zeros and ones**. You don't need to enter leading zeros, e.g., for "00001111," you can input just "1111".
2. **Enter the second binary number** in the second row.
3. The binary addition calculator will display the **result in the third field**.
4. In `advanced mode` (at the bottom), you can make the binary addition calculator **print the long addition** of the binary numbers.

What are the rules of binary Addition

There are four basic binary addition rules:

- `0 + 0 = 0`
- `0 + 1 = 1`
- `1 + 0 = 1`
- `1 + 1 = 10` (write "0" in the column and carry 1 to the next bit

The above equations work like in the decimal system, only here **you need to carry 1 when the sum exceeds 1** (in the decimal system, we do it when it exceeds 9).

Binary sum: Adding binary Number

How to add binary numbers? You could always convert binary numbers to decimal, add them as usual, and restore the result to the binary form.

The other way is to **use the above binary addition rules and perform a long addition**. Let's have a look at this binary sum:

```
carry:        1  1
            ________
              1001
            + 1101
            ________
            =10110
```

Binary Subtraction

In this part, we will describe the methods of dealing with the subtraction of binary numbers, the **Borrow Method** . There are several other tricks as well, but this is the most prevalent and help you understand the problem better.

1. **Borrow Method** – all you have to do is align the numbers as you would do with regular decimal subtraction. The procedure is almost the same! The only difference is that you operate with only two digits, not ten. You need to subtract digits in the same column, following these rules:
   - `1 - 0 = 1`;
   - `1 - 1 = 0`;
   - `0 - 0 = 0`; and
   - `0 - 1 = 1`, which actually comes from `10 - 1 = 1`, as you *borrow* `1` from the closest digit to the left, and after all that, the `1` you borrowed becomes `0`.

Example of Binary Subtraction

1 Example 1

```
      110
    ~~100~~1
  -  101
  ---------
  =  0010
```

2. Example 2

```
    10101
  -  001
  ---------
  =  10100
```

Binary Multiplication

While the decimal numeral system, which we are all familiar with, is based on the powers of 10, the **binary system has the base 2**. This means that **every digit of a binary number, a so-called bit, can only represent two logical values: `0` or `1`**. Therefore, binary numbers are commonly used in digital electronics and communications, representing the two states on and off.

We can convert binary numbers to the decimal system. They also allow the application of arithmetic operations, like addition, subtraction, division, and, as we will see in this binary calculator, multiplication.

Rules for Binary Multiplication

Binary multiplication has 4 basic rules:

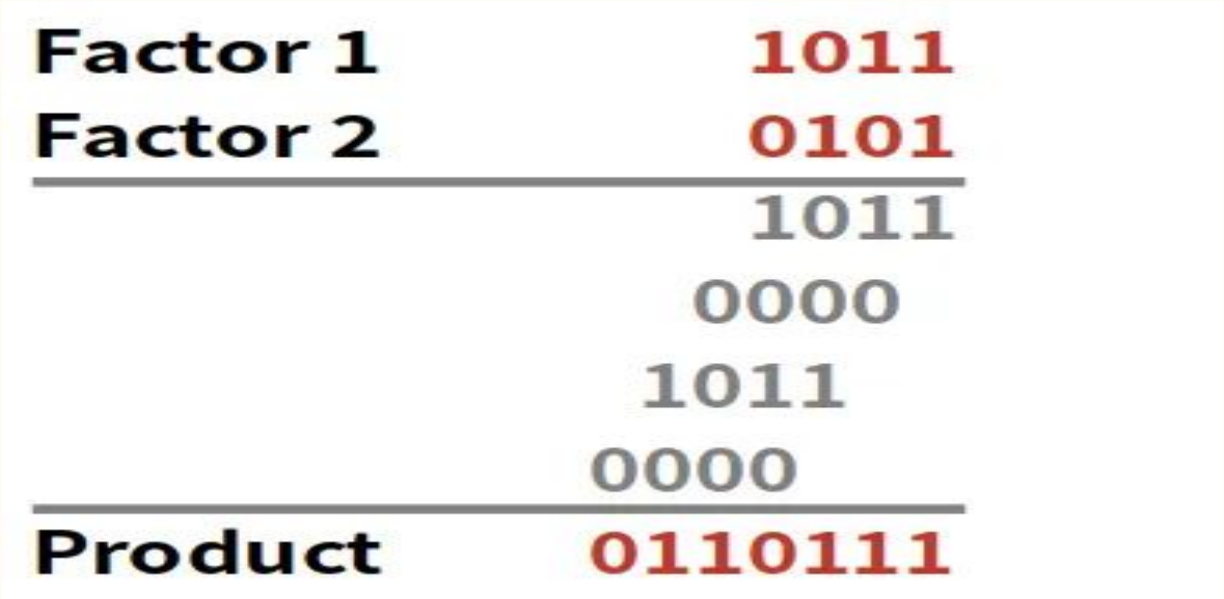
$$0 \times 0 = 0$$
$$0 \times 1 = 0$$
$$1 \times 0 = 0$$
$$1 \times 1 = 1$$

Based on those rules, binary multiplication is **very similar to decimal long multiplication**. We can even consider it slightly easier since we only have to deal with the digits `0` and `1`.

Multiply the multiplier by each digit of the multiplicand to achieve intermediate products, whose last digit is in the position of the corresponding multiplicand digit. The final product is the sum of those intermediate products.

Example of Binary Multiplication

As an example, let us look at the multiplication of `1011` and `0101` (13 and 5 in the decimal system):



The step-by-step procedure for the multiplication of those binary numbers is:

Binary Division

From a very early age, we are all taught to be familiar with the decimal system, numbers of the base. We know how to understand them and perform arithmetic operations with them. But all communication in our digital world is based on **binary numbers, whose digits can only have two values: `0` and `1`**. These binary numbers can be converted to the decimal system (both when integer and when they are binary fraction) but arithmetic operations like binary addition. binary multiplication, or binary subtraction can also be executed in the binary system.

Binary division strongly follows the decimal long division. The procedure consists of repeated binary subtraction steps.

Rules for Binary Division

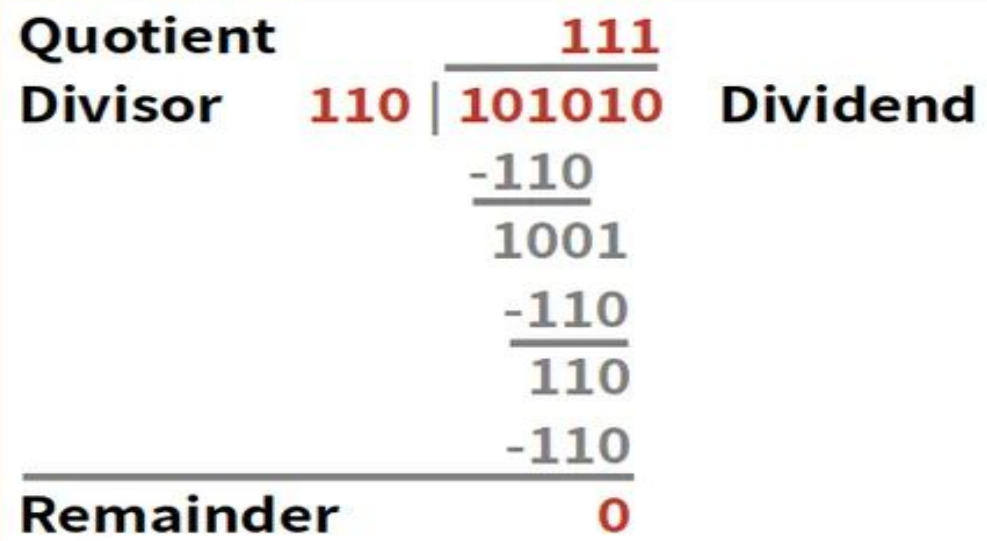
Beginning with the left (most significant) bit, it is inspected if **the divisor can be subtracted from the dividends' current digit**. If so, a `1` is noted in that bit of the quotient; if not, a `0`. The **remainder of the division process is carried, and the dividend's next digit is added** to it. You repeat this procedure is until the right (least significant) bit is reached.

The division remainder is the remainder after the last division process. It is smaller than the divisor and can therefore not be divided any further.

Let's take a look at this procedure with the example below. The binary number `101010` is divided by `110` (42 and 6 in the decimal system):

Example of Binary Division

The division remainder is the remainder after the last division process. It is smaller than the divisor and can therefore not be divided any further.
Let's take a look at this procedure with the example below. The binary number `101010` is divided by `110` (42 and 6 in the decimal system):



Signed and Unsigned Binary Number

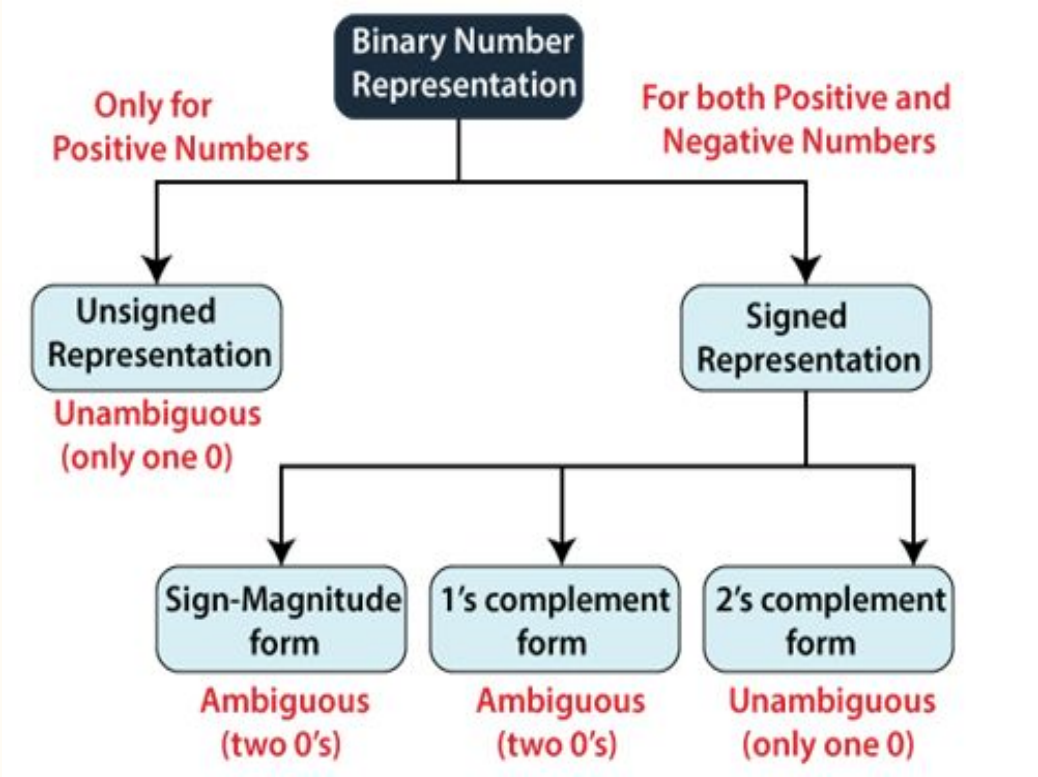
Integer variables can be signed or unsigned. Signed numbers use a sign flag to indicate positive or negative values, while unsigned numbers only store positive values without a sign flag.

In everyday life, we write positive numbers without a sign and negative numbers with a minus sign. However, in digital systems, binary numbers cannot have a minus sign. Instead, a special method is used to represent the sign in binary form.

Binary Number Representation

Computers understand only binary (0 and 1). Binary numbers can be signed or unsigned. Positive numbers can be represented in both signed and unsigned formats, but negative numbers can only be represented in signed format.

The main difference is that unsigned numbers do not use a sign bit, while signed numbers do.



Unsigned Numbers

Unsigned numbers do not use a sign to represent negative values, so they are always positive. By default, decimal numbers are assumed to be positive unless a negative sign is specified.

In unsigned binary numbers, there is no sign bit, so they only represent the magnitude of a number. For example, the number zero (0) is always positive in unsigned binary. Since each number has a unique binary form in unsigned representation, it is called an **unambiguous representation technique**. The range of unsigned binary numbers is from 0 to 2^n-1, where nnn is the number of bits.

Example:
To represent the decimal number 102 in unsigned binary:
Convert the decimal number to binary, focusing only on its magnitude.
The result is **1100110**.

Signed Numbers

Signed numbers use a **sign bit** to differentiate between positive and negative integers. This technique includes both the sign bit and the magnitude of the number. To represent a negative decimal number, a specific sign bit is added to the binary number.

Signed numbers can be represented in three ways. With a sign bit, zero can have two representations: positive (0) and negative (1), which makes it **ambiguous**. However, in **2's complement representation**, there is only one representation for zero, making it **unambiguous**.

Here are the three types of signed binary number representations:

1. **Sign-and-Magnitude Representation**
2. **1's Complement Representation**
3. **2's Complement Representation**

Sign magnitude form

In this form, a binary number has a bit for a sign symbol. If this bit is set to 1, the number will be negative else the number will be positive if it is set to 0. Apart from this sign-bit, the n-1 bits represent the magnitude of the number.

1's Complement form

By inverting each bit of a number, we can obtain the 1's complement of a number. The negative numbers can be represented in the form of 1's complement. In this form, the binary number also has an extra bit for sign representation as a sign-magnitude form.

Here are three examples of 1's complement representation for signed binary numbers:

1. **Positive Decimal Number (5):**
   - Binary representation: 0101 (4 bits)
   - In 1's complement, positive numbers remain unchanged.
   - **1's Complement: 0101**

1′s Complement form

- **Negative Decimal Number (-5):**
  i. Binary representation of 5: 0101(4 bits)
  ii. Take the complement (invert all bits): 1010
  iii. **1's Complement: 1010**
- **Negative Decimal Number (-12):**
  i. Binary representation of 12: 1100(4 bits)
  ii. Take the complement (invert all bits): 0011
  iii. **1's Complement: 0011**

Note: The leftmost bit in these examples represents the sign bit (0 for positive, 1 for negative).

2′s Complement form

By inverting each bit of a number and adding plus 1 to its least significant bit, we can obtain the 2's complement of a number. The negative numbers can also be represented in the form of 2's complement. In this form, the binary number also has an extra bit for sign representation as a sign-magnitude form.

Example: Represent +5 and -5 in 2's complement (using 4 bits)

1. **Positive Decimal Number (+5):**
   - Binary representation: 0101(4 bits).
   - In 2's complement, positive numbers remain unchanged.
   - **2's Complement: 0101**
2. **Negative Decimal Number (-5):**
   - Start with the binary representation of 5: 0101.
   - Take the **1's complement** (invert all bits): 1010.
   - Add 1 to the result: 1010+1=1011
   - **2's Complement: 1011**

9’s and 10’s Complement form

If the number is binary, then we use 1's complement and 2's complement. But in case, when the number is a decimal number, we will use the 9's and 10's complement. The 10's complement is obtained from the 9's complement of the number, and we can also find the 9's and 10's complement using the r's and (r-1)'s complement formula.

9′s Complement

The **9's complement** is a method of representing negative numbers, similar to the 1's complement but applied in base 10 (decimal system). To find the 9's complement of a number, you subtract each digit from 9.

Steps to find the 9's complement:

1. Subtract each digit of the number from 9.
2. If the number is negative, apply the 9's complement to the absolute value and add a negative sign.

9′s Complement example

Example 1: 9's complement of 5 (decimal number)

- The 9's complement of 5 is obtained by subtracting it from 9:
  9−5=4 => **9's complement: 4**

Example 2: 9's complement of 124 (decimal number)

- Subtract each digit from 9:
  9−1=8,
  9−2=7,
  9−4=5,. => **9's complement: 875**

Example 3: 9's complement of 56 (decimal number)

- Subtract each digit from 9:
  9−5=4,
  9−6=3. => **9's complement: 43**

10’s Complement

The **10's complement** is a method used to represent negative numbers in the decimal system, similar to the 2's complement used in binary. To find the 10's complement of a number, you follow these steps:

Steps to find the 10's complement:

1. **Find the 9's complement** of the number by subtracting each digit from 9.
2. **Add 1** to the result of the 9's complement.

10’s Complement example

Example 1: 10's complement of 5 (decimal number)

1. **Find the 9's complement of 5**:
   9−5=4
2. **Add 1 to the 9's complement**:
   4+1=5 => **10's complement of 5: 5**

Example 2: 10's complement of 124 (decimal number)

1. **Find the 9's complement of 124**:
   9−1=8,
   9−2=7,
   9−4=5.
   The 9's complement is **875**.
2. **Add 1 to the 9's complement**:
   875+1=876 =>**10's complement of 124: 876**

10’s Complement example

Example 3: 10's complement of 56 (decimal number)

1. **Find the 9's complement of 56**:
   9−5=4,
   9−6=3.
   The 9's complement is **43**.
2. **Add 1 to the 9's complement**:
   43+1=44 =>**10's complement of 56: 44**

Radix and Diminished Radix Complement

The most commonly used complements are 1's, 2's, 9's, and 10's complements, but there are many other types as well. Complements are used to simplify subtraction in number systems. If r is the base of the number system, two types of complements are possible:

1. **r's complement** (also known as Radix complement), and
2. **(r-1)'s complement** (also known as Diminished Radix complement).

For example, in binary (base 2), we use 1's and 2's complements. In octal (base 8), we use 7's and 8's complements.

The formulas for finding these complements are:

- **r's complement**: complement=$(r^n)_{10} - N$
- **(r-1)'s complement**: complement=$(r^n)_{10} - 1 - N$
  where N is the number and r is the base (radix).

Advantage of r's Complement

These are the following advantages of using r's complement:

- In r's complement, we can further use existing addition circuitry means there is no special circuitry.
- There is no need to determine whether the minuend and subtrahend are larger or not because the result has the right sign automatically.
- The negative zeros are eliminated by r's complement.

r's Complement example

Example 1: Binary System (Base 2)

Find the **2's complement** of 5 and subtract 5 from 12.

1. **Step 1: Represent 12 and 5 in binary:**
   - $12=1100_2$
   - $5=0101_2$.
2. **Step 2: Find the 2's complement of 5:**
   - First, find the 1's complement (invert the bits): 0101→1010
   - Add 1: 1010+1=1011.

     2's Complement of 5: 1011_2.
3. **Step 3: Add 121212 and 2′s2's2′s complement of 555:**
   1100+1011=10111(carry ignored).

   The result is 0111_2=7, which is 12−5.

r's Complement example

Example 1: Decimal System (Base 10)

Find the **10's complement** of 248 and subtract 248 from 543.

1. **Step 1: Find 10's Complement of 248:**
   - The formula for 10's complement is $(10^n)-N$, where n is the number of digits in N.
   - N=248, and n=3 (since 248 has 3 digits).
   - $10^3=1000$

     $10^3-248=1000-248=752$.
     10's Complement: 752.

2. **Step 2: Add 543543543 and 752752752:**
   543+752=1295
3. **Step 3: Ignore the extra digit (carry):** Drop the leading 111 (carry), leaving 295.
   The result is 295, which is 543−248.

1’s complement addition and subtraction

Addition using 1's complement

There are three different cases possible when we add two binary numbers which are as follows:

Case 1: Addition of the positive number with a negative number when the positive number has a greater magnitude.

Initially, calculate the 1's complement of the given negative number. Sum up with the given positive number. If we get the end-around carry 1, it gets added to the LSB.

1’s complement addition

Example: 1101 and -1001

1. First, find the 1's complement of the negative number 1001. So, for finding 1's complement, change all 0 to 1 and all 1 to 0. The 1's complement of the number 1001 is 0110.
2. Now, add both the numbers, i.e., 1101 and 0110; 1101+0110=1 0011
3. By adding both numbers, we get the end-around carry 1. We add this end around carry to the LSB of 0011. 0011+1=0100

Note: The resultant is a negative value

1's complement addition

Case 2: Adding a positive value with a negative value in case the negative number has a higher magnitude.

Initially, calculate the 1's complement of the negative value. Sum it with a positive number. In this case, we did not get the end-around carry. So, take the 1's complement of the result to get the final result.

Example: 1101 and -1110

1. First find the 1's complement of the negative number 1110. So, for finding 1's complement, we change all 0 to 1, and all 1 to 0. 1's complement of the number 1110 is 0001.
2. Now, add both the numbers, i.e., 1101 and 0001; 1101+0001= 1110
3. Now, find the 1's complement of the result 1110 that is the final result. So, the 1's complement of the result 1110 is 0001, and we add a negative sign before the number so that we can identify that it is a negative number.

1's complement addition

Case 3: Addition of two negative numbers

In this case, first find the 1's complement of both the negative numbers, and then we add both these complement numbers. In this case, we always get the end-around carry, which get added to the LSB, and for getting the final result, we take the 1's complement of the result.

Example: -1101 and -1110 in five-bit register

1. Firstly find the 1's complement of the negative numbers 01101 and 01110. So, for finding 1's complement, we change all 0 to 1, and all 1 to 0. 1's complement of the number 01110 is 10001, and 01101 is 10010.
2. Now, we add both the complement numbers, i.e., 10001 and 10010; 10001+10010= 1 00011

Note: The resultant is a negative value

1’s complement addition

3. By adding both numbers, we get the end-around carry 1. We add this end-around carry to the LSB of 00011.
   00011+1=00100
4. Now, find the 1's complement of the result 00100 that is the final answer. So, the 1's complement of the result 00100 is 110111, and add a negative sign before the number so that we can identify that it is a negative number.

Note: The resultant is a negative value

Subtraction using 1's complement

These are the following steps to subtract two binary numbers using 1's complement

- In the first step, find the 1's complement of the subtrahend.
- Next, add the complement number with the minuend.
- If got a carry, add the carry to its LSB. Else take 1's complement of the result which will be negative

***Note**: The subtrahend value always get subtracted from minuend.*

Subtraction using 1's complement

***Note**: The subtrahend value always get subtracted from minuend.*

Example 1: 10101 - 00111

We take 1's complement of subtrahend 00111, which comes out 11000. Now, sum them. So,

10101+11000 =1 01101.

In the above result, we get the carry bit 1, so add this to the LSB of a given result, i.e., 01101+1=01110, which is the answer.

Example 2: 10101 - 10111

We take 1's complement of subtrahend 10111, which comes out 01000. Now, add both of the numbers. So,

10101+01000 =11101.

In the above result, we didn't get the carry bit. So calculate the 1's complement of the result, i.e., 00010, which is the negative number and the final answer.

Addition and subtraction using 2's complement

There are three different cases possible when we add two binary numbers using 2's complement, which is as follows:

Case 1: Addition of the positive number with a negative number when the positive number has a greater magnitude.

Initially find the 2's complement of the given negative number. Sum up with the given positive number. If we get the end-around carry 1 then the number will be a positive number and the carry bit will be discarded and remaining bits are the final result.

Addition and subtraction using 2’s complement

Example: 1101 and -1001

1. First, find the 2's complement of the negative number 1001. So, for finding 2's complement, change all 0 to 1 and all 1 to 0 or find the 1's complement of the number 1001. The 1's complement of the number 1001 is 0110, and add 1 to the LSB of the result 0110. So the 2's complement of number 1001 is 0110+1=0111
2. Add both the numbers, i.e., 1101 and 0111; 1101+0111=1 0100
3. By adding both numbers, we get the end-around carry 1. We discard the end-around carry. So, the addition of both numbers is 0100.

Addition and subtraction using 2's complement

Note: The resultant is a negative value.

Case 2: Adding of the positive value with a negative value when the negative number has a higher magnitude.

Initially, add a positive value with the 2's complement value of the negative number. Here, no end-around carry is found. So, we take the 2's complement of the result to get the final result.

Example: 1101 and -1110

1. First, find the 2's complement of the negative number 1110. So, for finding 2's complement, add 1 to the LSB of its 1's complement value 0001. 0001+1=0010
2. Add both the numbers, i.e., 1101 and 0010; 1101+0010= 1111
3. Find the 2's complement of the result 1110 that is the final result. So, the 2's complement of the result 1110 is 0001, and add a negative sign before the number so that we can identify that it is a negative number.

Addition and subtraction using 2's complement

Note: The resultant is a negative value.

Case 3: Addition of two negative numbers

In this case, first, find the 2's complement of both the negative numbers, and then we will add both these complement numbers. In this case, we will always get the end-around carry, which will be added to the LSB, and forgetting the final result, we will take the2's complement of the result.

Example: -1101 and -1110 in five-bit register

1. Firstly find the 2's complement of the negative numbers 01101 and 01110. So, for finding 2's complement, we add 1 to the LSB of the 1's complement of these numbers. 2's complement of the number 01110 is 10010, and 01101 is 10011.
2. We add both the complement numbers, i.e., 10001 and 10010; 10010+10011= 1 00101
3. By adding both numbers, we get the end-around carry 1. This carry is discarded and the final result is the 2.s complement of the result 00101. So, the 2's complement of the result 00101 is 11011, and we add a negative sign before the number so that we can identify that it is a negative number.