颠倒二进制位

颠倒给定的 32 位无符号整数的二进制位。

提示：

- 请注意，在某些语言（如 Java）中，没有无符号整数类型。在这种情况下，输入和输出都将被指定为有符号整数类型，并且不应影响您的实现，因为无论整数是有符号的还是无符号的，其内部的二进制表示形式都是相同的。
- 在 Java 中，编译器使用二进制补码记法来表示有符号整数。因此，在上面的 **示例 2** 中，输入表示有符号整数 -3，输出表示有符号整数 -1073741825。

进阶:

如果多次调用这个函数，你将如何优化你的算法？

示例 1：

输入： 00000010100101000001111010011100

输出： 00111001011110000010100101000000

解释： 输入的二进制串 **00000010100101000001111010011100** 表示无符号整数 **43261596，**

因此返回 964176192，其二进制表示形式为 **00111001011110000010100101000000**。

示例 2：

输入： 11111111111111111111111111111101

输出： 10111111111111111111111111111111

解释： 输入的二进制串 **11111111111111111111111111111101** 表示无符号整数 4294967293，

因此返回 3221225471 其二进制表示形式为 **10111111111111111111111111111111** 。

示例 1：

输入： n = 00000010100101000001111010011100

输出： 964176192 (00111001011110000010100101000000)

解释： 输入的二进制串 **00000010100101000001111010011100** 表示无符号整数 **43261596，**

因此返回 964176192，其二进制表示形式为 **00111001011110000010100101000000**。

示例 2：

输入：n = 11111111111111111111111111111101

输出：3221225471 (10111111111111111111111111111111)

解释：输入的二进制串 **11111111111111111111111111111101** 表示无符号整数 4294967293，

因此返回 3221225471 其二进制表示形式为 **10111111111111111111111111111111** 。

```
class Solution {
public:
    uint32_t reverseBits(uint32_t n) {
        uint32_t res=0;
        for(int i=0;i<32;i++)
        {
            res=(res<<1)+((n>>i)&1);
        }
        return res;
    }
};
```