1. Reverse Bits

Reverse bits of a given 32 bits unsigned integer.

**Example 1:**

Input: 00000010100101000001111010011100  
Output: 00111001011110000010100101000000  
Explanation: The input binary string 00000010100101000001111010011100 represents the unsigned integer 43261596, so return 964176192 which its binary representation is 00111001011110000010100101000000.

**Example 2:**

Input: 11111111111111111111111111111101  
Output: 10111111111111111111111111111111  
Explanation: The input binary string 11111111111111111111111111111101 represents the unsigned integer 4294967293, so return 3221225471 which its binary representation is 10111111111111111111111111111111.

**Note:**

* Note that in some languages such as Java, there is no unsigned integer type. In this case, both input and output will be given as signed integer type and should not affect your implementation, as the internal binary representation of the integer is the same whether it is signed or unsigned.
* In Java, the compiler represents the signed integers using [2’s complement notation](https://en.wikipedia.org/wiki/Two's_complement). Therefore, in **Example 2** above the input represents the signed integer -3 and the output represents the signed integer -1073741825.

**Follow up**:

If this function is called many times, how would you optimize it?

**解**

class Solution {  
 public:  
 uint32\_t reverseBits(uint32\_t n) {  
 uint32\_t res = 0;  
 for(int i = 0; i < 32; ++i){  
 res <<= 1;  
 if(n & 1 == 1)res += 1;  
 n >>= 1;  
 }  
 return res;  
 }  
};

class Solution {  
 public:  
 uint32\_t reverseBits(uint32\_t n) {  
 // 前16位与后16位互换  
 n = ((n & 0xffff0000) >> 16) | ((n & 0x0000ffff) << 16);  
 // 前16位和后16位中，每8位互换位置  
 n = ((n & 0xff00ff00) >> 8) | ((n & 0x00ff00ff) << 8);  
 // 第 1、2、3、4个8位中，每4位互换位置  
 n = ((n & 0xf0f0f0f0) >> 4) | ((n & 0x0f0f0f0f) << 4);  
 // 第 1、2、3...8个4位中，每2位互换位置  
 n = ((n & 0xcccccccc) >> 2) | ((n & 0x33333333) << 2);  
 // 每2位为一组，互换位置  
 n = ((n & 0xaaaaaaaa) >> 1) | ((n & 0x55555555) << 1);  
 return n;  
 }  
};