Swap bits in a given number

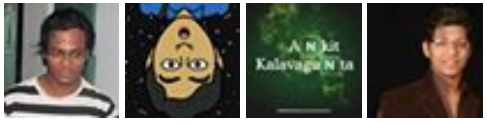
Given a number x and two positions (from right side) in binary representation of x, write a function that swaps n bits at given two positions and returns the result. It is also given that the two sets of bits do not overlap.

Examples:

```
Let p1 and p2 be the two given positions.

Example 1
Input:
x = 47 (00101111)
p1 = 1 (Start from second bit from right side)
p2 = 5 (Start from 6th bit from right side)
n = 3 (No of bits to be swapped)
Output:
227 (11100011)
The 3 bits starting from the second bit (from right side) are
swapped with 3 bits starting from 6th position (from right side)


Example 2
Input:
x = 28 (11100)
p1 = 0 (Start from first bit from right side)
p2 = 3 (Start from 4th bit from right side)
n = 2 (No of bits to be swapped)
```

Interview Experiences
Advanced Data Structures
Dynamic Programming
Greedy Algorithms
Backtracking
Pattern Searching
Divide & Conquer
Mathematical Algorithms
Recursion
Geometric Algorithms

```
Output:
7 (00111)
The 2 bits starting from 0th postion (from right side) are
swapped with 2 bits starting from 4th position (from right side)
```

Solution

We need to swap two sets of bits. XOR can be used in a similar way as it is used to swap 2 numbers. Following is the algorithm.

```
1) Move all bits of first set to rightmost side
   set1 =  (x >> p1) & ((1U << n) - 1)
Here the expression (1U << n) - 1 gives a number that
contains last n bits set and other bits as 0. We do &
with this expression so that bits other than the last
n bits become 0.
2) Move all bits of second set to rightmost side
   set2 =  (x >> p2) & ((1U << n) - 1)
3) XOR the two sets of bits
   xor = (set1 ^ set2)
4) Put the xor bits back to their original positions.
   xor = (xor << p1) | (xor << p2)
5) Finally, XOR the xor with original number so
   that the two sets are swapped.
   result = x ^ xor
```

Implementation:

```
#include<stdio.h>

int swapBits(unsigned int x, unsigned int p1, unsigned int p2, unsigne
{
    /* Move all bits of first set to rightmost side */
    unsigned int set1 =  (x >> p1) & ((1U << n) - 1);

    /* Moce all bits of second set to rightmost side */
    unsigned int set2 =  (x >> p2) & ((1U << n) - 1);

    /* XOR the two sets */
    unsigned int xor = (set1 ^ set2);
```


Popular Posts

All permutations of a given string
Memory Layout of C Programs
Understanding “extern” keyword in C
Median of two sorted arrays
Tree traversal without recursion and without stack!
Structure Member Alignment, Padding and Data Packing
Intersection point of two Linked Lists
Lowest Common Ancestor in a BST.
Check if a binary tree is BST or not
Sorted Linked List to Balanced BST

```
    /* Put the xor bits back to their original positions */
    xor = (xor << p1) | (xor << p2);

    /* XOR the 'xor' with the original number so that the
       two sets are swapped */
    unsigned int result = x ^ xor;

    return result;
}

/* Drier program to test above function*/
int main()
{
    int res =  swapBits(28, 0, 3, 2);
    printf("\nResult = %d ", res);
    return 0;
}
```

Output:

```
Result = 7
```

Following is a shorter implementation of the same logic

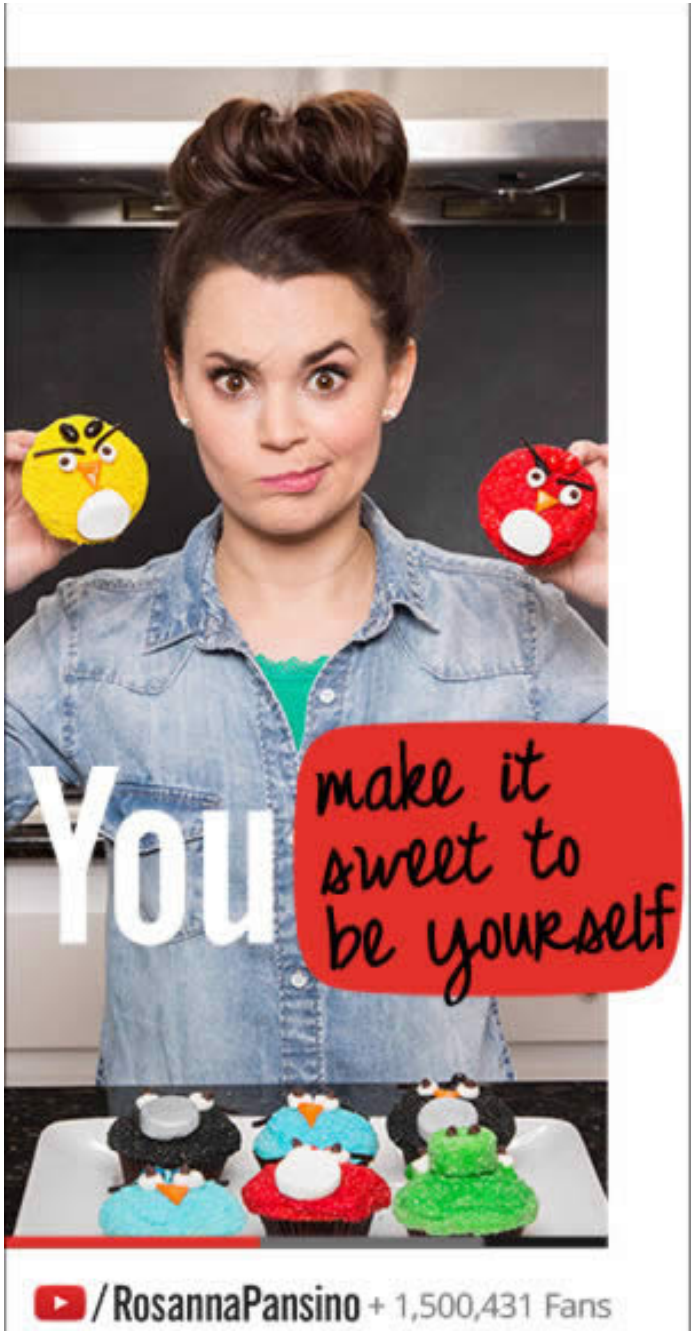
```
int swapBits(unsigned int x, unsigned int p1, unsigned int p2, unsigne
{
    /* xor contains xor of two sets */
    unsigned int xor = ((x >> p1) ^ (x >> p2)) & ((1U << n) - 1);

    /* To swap two sets, we need to again XOR the xor with original se
    return x ^ ((xor << p1) | (xor << p2));
}
```

References:

Swapping individual bits with XOR

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.



Related Tpoics:

- Check if a number is multiple of 9 using bitwise operators
- How to swap two numbers without using a temporary variable?
- Divide and Conquer | Set 4 (Karatsuba algorithm for fast multiplication)
- Find position of the only set bit
- Swap all odd and even bits
- Add two bit strings
- Write your own strcmp that ignores cases
- Binary representation of a given number

Writing code in comment? Please use **ideone.com** and share the link here.

9 Comments GeeksforGeeks

Recent Comments

Abhi You live US or India?

Google (Mountain View) interview · 23 minutes ago

Aman Hi, Why arent we checking for conditions...

Write a C program to Delete a Tree. · 1 hour ago

kzs please provide solution for the problem...

Backtracking | Set 2 (Rat in a Maze) · 1 hour ago

Sanjay Agarwal bool tree::Root_to_leaf_path_given_sum(tree...

Root to leaf path sum equal to a given number · 1 hour ago

GOPI GOPINATH @admin Highlight this sentence "We can easily...

Count trailing zeroes in factorial of a number · 1 hour ago

newCoder3006 If the array contains negative numbers also. We...

Find subarray with given sum · 1 hour ago

Sort by Newest

Join the discussion…

Rawat • 2 years ago

This solution works fine, I first copied the n-bits from p1 position to p2 and thar p2 to p1 position. Basic swap Algorithm !

```
 #include <iostream>
using namespace std;
int main()
{
    int val,p1,p2,n,org,orgN,newVal=0,ntNum=0;
    cin>>val>>p1>>p2>>n;
    org=val;orgN=n;
    while(n--)
        newVal|=(val&(1<<(p1+n)))>0?(1<<(p1+n)):0;
    newVal<<=(p2-p1);
    n=orgN;
    while(n--)
        ntNum|=1<<(p2+n);
    val&=~ntNum;
    val|=newVal;
```

see more

• Reply • Share ›

kavish • 2 years ago

could you please explain the role of (1U<<n)-1 in detail?

• Reply • Share ›

Ram ↗ kavish • 2 years ago

Kavish,

Here n signifies the number of bits to be swapped.
consider n=2
1U<<2 mean it gives 0100 (ignoring the other nibbles on MSB side)
now when you substract 1 from it, it will become 0011

Ideas is to have only those n bits rest of them are masked to zero.

akshayjohri89 • 2 years ago

How about just
Create a mask of all 1s for 1st set by

```
 /* Paste your code here (You may delete these lines if not writing c

unsigned int swapbits(unsigned int x,int p1,int p2, int n){
unsigned int mask1,mask2,temp;
mask1=1<<p1;

//Create mask1
for(i=0;i<n-1;i++)
    mask1=mask1|(mask1<<1);

//Keeping a copy of mask1 in temp
temp=mask1;

//Create mask2 by shifting mask1 by p2-p1
mask2=mask1<<(p2-p1);
```

• Reply • Share ›

Tushar_Pucsd • 2 years ago

If any one explain (1U << n) how it exucute?

• Reply • Share ›

ish7 • 2 years ago

Awesome :)

• Reply • Share ›

Venki • 2 years ago

I guess the positions should not overlap.

• Reply • Share ›

GeeksforGeeks ➦ Venki • 2 years ago

Thanks Venki. This point has been added to the problem statement.

• Reply • Share ›

Pavan ➦ GeeksforGeeks • a year ago

In the example given, if x =47, n=3, p1=1, p2=3, the positions o
works currently here as well. Please let me know why does this
does not overlap.

```
/* Paste your code here (You may delete these lines if
```

• Reply • Share ›

Subscribe Add Disqus to your site