Program to count number of set bits in an (big) array

Given an integer array of length N (an arbitrarily large number). How to count number of set bits in the array?

The simple approach would be, create an efficient method to count set bits in a word (most prominent size, usually equal to bit length of processor), and add bits from individual elements of array.

Various methods of counting set bits of an integer exists, see this for example. These methods run at best O(logN) where N is number of bits. Note that on a processor N is fixed, count can be done in O(1) time on 32 bit machine irrespective of total set bits. Overall, the bits in array can be computed in O(n) time, where 'n' is array size.

However, a table look up will be more efficient method when array size is large. Storing table look up that can handle 2^{32} integers will be impractical.

The following code illustrates simple program to count set bits in a randomly generated 64 K integer array. The idea is to generate a look up for first 256 numbers (one byte), and break every element of array at byte boundary. A meta program using C/C++ preprocessor generates the look up table for counting set bits in a byte.

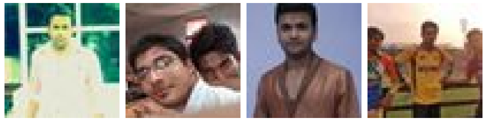
The mathematical derivation behind meta program is evident from the following table (Add the column and row indices to get the number, then look into the table to get set bits in that number. For example, to get set bits in 10, it can be extracted from row named as 8 and column named as 2),

```
   0, 1, 2, 3
 0 - 0, 1, 1, 2 -------- GROUP_A(0)
```

Google™ Custom Search

GeeksforGeeks

53,526 people like GeeksforGeeks.



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

```
 4 - 1, 2, 2, 3 -------- GROUP_A(1)
 8 - 1, 2, 2, 3 -------- GROUP_A(1)
12 - 2, 3, 3, 4 -------- GROUP_A(2)
16 - 1, 2, 2, 3 -------- GROUP_A(1)
20 - 2, 3, 3, 4 -------- GROUP_A(2)
24 - 2, 3, 3, 4 -------- GROUP_A(2)
28 - 3, 4, 4, 5 -------- GROUP_A(3) ... so on
```

From the table, there is a patten emerging in multiples of 4, both in the table as well as in the group parameter. The sequence can be generalized as shown in the code.

Complexity:

All the operations takes O(1) except iterating over the array. The time complexity is O(n) where ‘n’ is size of array. Space complexity depends on the meta program that generates look up.

Code:

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

/* Size of array 64 K */
#define SIZE (1 << 16)

/* Meta program that generates set bit count
   array of first 256 integers */

/* GROUP_A - When combined with META_LOOK_UP
   generates count for 4x4 elements */

#define GROUP_A(x) x, x + 1, x + 1, x + 2

/* GROUP_B - When combined with META_LOOK_UP
   generates count for 4x4x4 elements */

#define GROUP_B(x) GROUP_A(x), GROUP_A(x+1), GROUP_A(x+1), GROUP_A(x+2

/* GROUP_C - When combined with META_LOOK_UP
   generates count for 4x4x4x4 elements */

#define GROUP_C(x) GROUP_B(x), GROUP_B(x+1), GROUP_B(x+1), GROUP_B(x+2
```

Dynamic DevOps Suite
Fast, flexible and hands-free DevOps
from build through Deploy
OM software
Click to Learn More >

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

```
/* Provide appropriate letter to generate the table */

#define META_LOOK_UP(PARAMETER) \
   GROUP_##PARAMETER(0),  \
   GROUP_##PARAMETER(1),  \
   GROUP_##PARAMETER(1),  \
   GROUP_##PARAMETER(2)   \

int countSetBits(int array[], size_t array_size)
{
   int count = 0;

   /* META_LOOK_UP(C) - generates a table of 256 integers whose
      sequence will be number of bits in i-th position
      where 0 <= i < 256
   */

    /* A static table will be much faster to access */
       static unsigned char const look_up[] = { META_LOOK_UP(C) };

    /* No shifting funda (for better readability) */
    unsigned char *pData = NULL;

   for(size_t index = 0; index < array_size; index++)
   {
      /* It is fine, bypass the type system */
      pData = (unsigned char *)&array[index];

      /* Count set bits in individual bytes */
      count += look_up[pData[0]];
      count += look_up[pData[1]];
      count += look_up[pData[2]];
      count += look_up[pData[3]];
   }

   return count;
}

/* Driver program, generates table of random 64 K numbers */
int main()
{
   int index;
   int random[SIZE];

   /* Seed to the random-number generator */
   srand((unsigned)time(0));
```

```
    /* Generate random numbers. */
    for( index = 0; index < SIZE; index++ )
    {
        random[index] = rand();
    }

    printf("Total number of bits = %d\n", countSetBits(random, SIZE));
    return 0;
}
```

Contributed by **Venki**. 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

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

- 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

3 Tweet 0 0

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

6 Comments GeeksforGeeks

Sort by Newest ▾

Join the discussion…

Guest • 6 months ago

other way to count the number of set bit.

```
int countSetbit(int n)
{
int count=0;

while(n&n-1)
{
n=n&n-1;
count++;

}

return count;

}
```

take an example :

n=13

bits: 1101

see more

• Reply • Share ›

Bohemia • 8 months ago

```
# define SB2(n) n, n+1, n+1, n+2

# define SB4(n) SB2(n), SB2(n+1), SB2(n+1), SB2(n+2)

# define SB6(n) SB4(n), SB4(n+1), SB4(n+1), SB4(n+2)

//The macros basically generate the number of set bits in integers from 0 to 2

int main()

{

int Table256[256]={SB6(0),SB6(1),SB6(1),SB6(2)};//My LookUp Table

int arr[]={1,3,7,15};//My N- element Array

int n=sizeof(arr)/sizeof(arr[0]);//N

int i;//to iterate over the array

int sumSetBits=0;//contains # of set bit
```

see more

• Reply • Share ›

Hanish Bansal • 11 months ago

## concatenates what is before it with what is after it

• Reply • Share ›

Sindhu • 2 years ago

Hi venki,
I tried executing this code on a borland compiler.META_LOOK_UP(C)gives er
this defined as standard function. do u know its implementation.

Also groups should be defined the way u hav defined?

i still didnt get clear idea abt groups. how do u generalise defining groups

• Reply • Share ›

pranav • 3 years ago

please someone explain clearly GROUP_A, GROUP_B, etc. concept and mea

```
#define META_LOOK_UP(PARAMETER) \
GROUP_##PARAMETER(0), \
GROUP_##PARAMETER(1), \
GROUP_##PARAMETER(1), \
GROUP_##PARAMETER(2) \
```

• Reply • Share ›

Venki ➦ pranav • 3 years ago

@pranav, the idea is to create table of 256 entries. The table contains
index of table. The macro META_LOOK_UP(C) generates the table. At
of look_up will be something like

static unsigned char const look_up[] = { 0, 1, 1, 2, 1, 2, 2, 3, ... 6, 7, 7,

The code in main, simply splits the array into individual bytes and looks

• Reply • Share ›