std::countr_zero in C++

by grammer jtp

General metrics

4,620	746	87	2 min 59 sec	5 min 44 sec
characters	words	sentences	reading time	speaking time

Writing Issues

✓ No issues found

Plagiarism

3
%

0
source

3% of your text matches 0 sources on the web or in archives of academic publications

Unique Words

Measures vocabulary diversity by calculating the percentage of words used only once in your document

19%

unique words

Rare Words

Measures depth of vocabulary by identifying words that are not among the 5,000 most common English words.

30%

rare words

Word Length

Measures average word length

4.5

characters per word

Sentence Length

Measures average sentence length

8.6

words per sentence

std::countr_zero in C++

std::countr_zero in C++

This countr_zero function is introduced in C++20. This function is present in the <bit> header. This function is used to count the trailing zeros in a binary representation of a number. That means this function is used to get the count of a number of consecutive zeros in an unsigned integer from the least significant bit.

The syntax of the countr_zero function in Cpp

int result = countr_zero(unsigned int value);

This function will take an unsigned integer type value as input and return an integer value that represents the number of trailing zeros in the given number.

If the zero is given as input, then the function returns the width of the integer type.

Example 1:

Input: 8

Output: 3

Explanation:

The binary representation of the given is as follows

00000000000000000000000000001000

Then, the trailing zeros in the above number are 3.

Example 2:

Input: 0

Output: 32

Explanation:

The binary representation of the given is as follows

00000000000000000000000000000000

Then, the trailing zeros in the above representation are 32

Cpp program to illustrate the countr_zero function

```
#include <iostream>

#include <bitset>

#include <bit>

using namespace std;

int main() {

    unsigned int value;

    cout << "Enter an unsigned integer value: ";

    cin >> value;

    cout << "Binary representation of " << value << ": ";

    cout << bitset<sizeof(unsigned int) * 8>(value) << endl;

    int num_zeros = countr_zero(value);
```

```
   cout << "Number of trailing zeros in " << value << ": " <<
num_zeros << endl;
   return 0;
}
```

Output:

Explanation:

The above program illustrates the functionality of the countr_zero function. First, an unsigned integer variable is declared, which stores the input number given by the user. Then, a prompt is written asking the user to enter an unsigned integer value. A binary representation of the given number is printed using the bitset with a width equal to the number. Then, using the countr_zero function, the number of trailing zeros is printed.

Applications of countr_zero function

Cpp program to get the least significant set bit in given number

```
#include <iostream>
#include <bit>
#include <bitset>
using namespace std;
```

```
int findLeastSignificantSetBit(unsigned int n) {
    if (n == 0) {
        return -1;
    }
    return sizeof(n) * 8 - countr_zero(n) - 1;
}
int main() {
    unsigned int number;
    cout << "Enter an unsigned integer value: ";
    cin >> number;
    cout << "Binary representation of " << number << ": ";
    cout << bitset<sizeof(unsigned int) * 8>(number) << endl;
    int bitPosition = findLeastSignificantSetBit(number);
    if (bitPosition != -1) {
        cout << "The least significant set bit in " << number << " is at position: " << bitPosition << endl;
    } else {
        cout << number << "has no set bits." << endl;
    }

    return 0;
}
```

Output:

Explanation:

The above function is an application of the countr_zero function. This program is used to find the least significant set bit position in given number. This program has two main functions, which are findLeastSignigicantSetBit functions. In the main function, an unsigned integer variable is declared to store the user input. The user has to enter the number for which the user wants to know the LSB. Then, the binary representation of the given number is printed using the bitset library. Now, findLeastSignigicantSetBit is called by passing the number as a parameter. This function will return the position of the least significant bit in the given number. If else statement is used to check whether the given number has set bits or not. If the result given by the findLeastSignigicantSetBit function is not equal to -1, then there is a set bit in number and printed the position of the set bit. Otherwise, a print message saying that the number has no set bits is printed.

The findLeastSignigicantSetBit takes an unsigned integer as input. It checks whether the input is zero or not. If it is zero, it returns -1, which indicates there are no set bits in the given number. This function uses the countr_zero function. The trailing zeros are stored in a variable named "numTrailingZeros," which is used to find the final position. Then, the final result is returned to the main function.

Conclusion:

This article is all about the countr_zero function, which is introduced in c++20. This function is used to get the number of trailing zeros in the binary representation of the given unsigned integer value. This article also gave one application program that uses the countr_zero function to find the least significant set bit in the given unsigned integer value.

1.	*the number of trailing zeros in the binary representation of the*	std::builtins::u32 - Alumina Docs https://docs.alumina-lang.net/std/builtins/u32.html	Originality