Fastest way of finding if a number is a Power of 2

Ankur M. Satle

https://ankursatle.wordpress.com | /ankursatle

ankursatle@gmail.com | @ankursatle

How to Win at Technical Interviews

The Secret Protocol You're Expected to Follow

A Lightning Talk from yesterday!

No!
This problem does not need a hash_map
☺☺☺

Find out if a given unsigned is a power of 2

Problem Statement

There are 10 kinds of people…
Those that understand numbers
And those that don’t…

Powers of 2?

Power of 2	Value	Power of 2	Value, base 2
2^{00}	1	2^{00}	0x000000000001
2^{01}	2	2^{01}	0x000000000010
2^{02}	4	2^{02}	0x000000000100
2^{03}	8	2^{03}	0x000000001000
2^{04}	16	2^{04}	0x000000010000
2^{05}	32	2^{05}	0x000000100000
2^{06}	64	2^{06}	0x000001000000
2^{07}	128	2^{07}	0x000010000000
2^{08}	256	2^{08}	0x000100000000
2^{09}	1024	2^{09}	0x001000000000
2^{10}	2048	2^{10}	0x010000000000
2^{11}	4096	2^{11}	0x100000000000

Count bits

```
constexpr bool is_pow2_bit_counting_loop(unsigned n) {
    int no_bits = 0;
    for (auto i = 0u; i < sizeof(n); ++i) {
        if ((n & (1u << i)) == n) ++no_bits;
    }
    return no_bits == 1;
}
```

Count bits - generic

```cpp
#include <concepts>
constexpr bool is_pow2_bit_counting_loop(std::unsigned_integral auto n) {
    int no_bits = 0;
    for (auto i = 0u; i < sizeof(decltype(n)); ++i) {
        if ((n & (1u << i)) == n) ++no_bits;
    }
    return no_bits == 1;
}
```

Match with powers of 2

```
constexpr bool is_pow2_match_with_powers(std::unsigned_integral auto n)
{
    for (auto i = 0u; i < sizeof(decltype(n)); ++i) {
        if (n == (1u << i)) return true;
    }
    return false;
}
```

Kernighan & Ritchie from C Programming Language

```cpp
constexpr bool is_pow2_knr(std::unsigned_integral auto n) {
  return (n & (n - 1)) == 0;
}
```

```
4  → 0x0100 & 0x0011 = 0
5  → 0x0101 & 0x0100 = 0x0100
10 → 0x1011 & 0x1010 = 0x1010
```

Bitset

```cpp
#include <bitset>
constexpr bool is_pow2_bitset(std::unsigned_integral auto n) {
    std::bitset<sizeof(decltype(n))> bs(n);
    return bs.count() == 1;   //NOT constexpr! :(
}
```

Standard library header <bit> (C++20)

This header is part of the numeric library.

Types

endian (C++20)	indicates the endianness of scalar types (enum)

Functions

bit_cast (C++20)	reinterpret the object representation of one type as that of another (function template)
byteswap (C++23)	reverses the bytes in the given integer value (function template)
has_single_bit (C++20)	checks if a number is an integral power of two (function template)
bit_ceil (C++20)	finds the smallest integral power of two not less than the given value (function template)
bit_floor (C++20)	finds the largest integral power of two not greater than the given value (function template)
bit_width (C++20)	finds the smallest number of bits needed to represent the given value (function template)
rotl (C++20)	computes the result of bitwise left-rotation (function template)
rotr (C++20)	computes the result of bitwise right-rotation (function template)
countl_zero (C++20)	counts the number of consecutive 0 bits, starting from the most significant bit (function template)
countl_one (C++20)	counts the number of consecutive 1 bits, starting from the most significant bit (function template)
countr_zero (C++20)	counts the number of consecutive 0 bits, starting from the least significant bit (function template)
countr_one (C++20)	counts the number of consecutive 1 bits, starting from the least significant bit (function template)
popcount (C++20)	counts the number of 1 bits in an unsigned integer (function template)

Synopsis

Popcount

```cpp
#include <bit>
constexpr bool is_pow2_popcount(std::unsigned_integral auto n) {
    return std::popcount(n) == 1;
}
```

has_single_bit

```cpp
#include <bit>
constexpr bool is_pow2_has_single_bit(std::unsigned_integral auto n) {
    return std::has_single_bit(n);
}
```

Performance!?
`popcount` is just one instruction

Which of the 10 kinds are you?

Thank you!

Ankur M. Satle