

Битовые операции



Санкт-Петербургский
государственный университет

Типы операций

- Логические операции: $a \ \&\& \ b$, $a \ || \ b$, ...
- Битовые операции: $a \ \& \ b$, $a \ | \ b$, ...
- Таблицы истинности для них одинаковы

Инверсия

$\sim A$ (логическое «НЕ»; NOT)

<hr/>	
A	$\sim A$
<hr/>	
0	1
1	0

100010101110	
<hr/>	
011101010001	

Конъюнкция

A & B (Логическое «И»; AND)

A	B	A & B
0	0	0
0	1	0
1	0	0
1	1	1

```
100010101110
110101110100
-----
100000100100
```

Дизъюнкция

$A \mid B$ (Логическое «ИЛИ»; OR)

A	B	$A \mid B$
0	0	0
0	1	1
1	0	1
1	1	1

```
100010101110
110101110100
-----
110111111110
```

Строгая дизъюнкция

$A \wedge B$ (Исключающее «ИЛИ»; XOR)

A	B	$A \wedge B$
0	0	0
0	1	1
1	0	1
1	1	0

```
100010101110
110101110100
-----
010111011010
```

Логический сдвиг влево

« n

Пример: 11100101 « 2 = 10010100

1 « 8 = ?

Логический сдвиг вправо

» n

Пример: 01011011 » 3 = 00001011

Другие сдвиги

- Циклический
- Арифметический
 - влево: как логический
 - вправо: все биты сдвигаются вправо, а на освободившееся место размножается знаковый бит

Пример: $10010111 \gg 1 = 11001011$

А что, если ...

- ① Сдвинуть на 0 бит? Ничего
- ② Сдвинуть на отрицательное число бит?
- ③ Сдвинуть на число битов большее, чем есть в слове?

6.5.8 Bitwise shift operators

Syntax

- 1 *shift-expression*:
additive-expression
shift-expression << *additive-expression*
shift-expression >> *additive-expression*

Constraints

- 2 Each of the operands shall have integer type.

Semantics

- 3 The integer promotions are performed on each of the operands. The type of the result is that of the promoted left operand. If the value of the right operand is negative or is greater than or equal to the width of the promoted left operand, the behavior is undefined.
- 4 The result of $E1 \ll E2$ is $E1$ left-shifted $E2$ bit positions; vacated bits are filled with zeros. If $E1$ has an unsigned type, the value of the result is $E1 \times 2^{E2}$, wrapped around. If $E1$ has a signed type and nonnegative value, and $E1 \times 2^{E2}$ is representable in the result type, then that is the resulting value; otherwise, the behavior is undefined.
- 5 The result of $E1 \gg E2$ is $E1$ right-shifted $E2$ bit positions. If $E1$ has an unsigned type or if $E1$ has a signed type and a nonnegative value, the value of the result is the integral part of the quotient of $E1/2^{E2}$. If $E1$ has a signed type and a negative value, the resulting value is implementation-defined.

Свойства XOR

1 $A \oplus A = 0$

2 $A \oplus 0 = A$

Помехоустойчивое кодирование

$$X_1 \wedge X_2 \wedge X_3 = X_4$$

Дано: X_2, X_3, X_4
 $X_1 = ?$

Порядок вычислений

false && func1() && func2()

0 & func1() & func2()

false & func1() & func2()

- 1 Каковы условия корректности выражений?
- 2 Будут ли вычисляться значения функций в каждом из случаев?

Пример:

IP адреса: 192.168.1.42/24

Пример:

Выделение битов контрольной суммы:

```
bit_i = (a >> i) & 1;
```

```
bit_ij = ...;
```


Задача из практики

Эмуляция удаления из списка

Из $[1, 2, 3, 4, 5, 6]$ хотим получить $[1, 3, 5]$ не удаляя элементы из исходного списка

Задача на битовые хаки

Задача: Проверить, является ли значение переменной `x` типа `long long` степенью двойки.

Вариант 1

```
bool check(long long x)
{
    while (x != 1) {
        if (x % 2 == 1)
            return false;
        x = x / 2;
    }
    return true;
}
```

Вариант 2

```
bool check(long long x)
{
    double y = log2(x);
    return ((int)y == y);
}
```

Вариант 3

```
bool check(long long x)
{
    return x & (x - 1) == 0;
}
```

x =	0100110
x-1 =	0100101
<hr/>	
& =	0100100

Битовые операции

- Повышают производительность, и иногда значительно
- Лежат в основе многих стандартных методов
- Внедряются компиляторами при оптимизации
- Снижают читаемость кода

Потенциально рекомендуемая литература

Henry Warren «Hacker's delight»

- Пример про степень двойки взят оттуда
- Там же огромное количество других трюков с битами
- В книге около 500 страниц, поэтому её слегка страшно рекомендовать в 1 семестре