Основы алгебры логики

Содержание

russian

1. Логические функции
   - Основы
   - Важнейшие логические функции
   - Формулы вместо функций трёх и более агрументов
2. Формулы
   - Конструирование функций формулами
   - Запись формул
   - Алгебра логики
3. Базис
   - Основной логический базис
   - Логические базисы
   - Избыточность основного логического базиса

russian russian russian russian russian russian russian russian russian russian russian russian

Область определения и область значения

- Любое логическое выражение истинно либо ложно.
- Обозначим истину символом 1, а ложь — символом 0.

Логическими функциями называются функции вида

$$f(x_1, \ldots, x_n),$$

где, как аргументы x_i, так и функция принимают значение либо 0, либо 1.

Способы задания логических функций

Таблица истинности

x_1	$\cdots$	x_{n-1}	x_n	$f(x_1, \cdots, x_{n-1}, x_n)$
0	$\cdots$	0	0	$y_0 = f(0, \cdots, 0, 0)$
0	$\cdots$	0	1	$y_1 = f(0, \cdots, 0, 1)$
0	$\cdots$	1	0	$y_2 = f(0, \cdots, 1, 0)$
0	$\cdots$	1	1	$y_3 = f(0, \cdots, 1, 1)$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
1	$\cdots$	1	1	$y_{2^n-1} = f(1, \cdots, 1, 1)$

Каково количество всех возможных функций n аргументов?

Функции одного аргумента

x	f_0	f_1	f_2	f_3
0	0	0	1	1
1	0	1	0	1

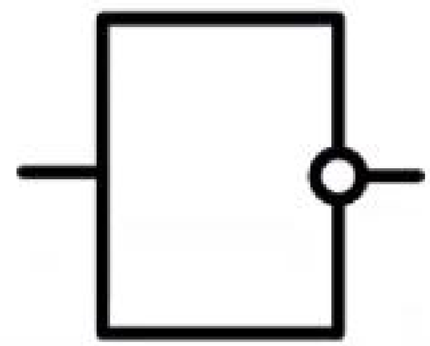
Функций одного аргумента $f(x)$ всего $2^{2^1} = 4$. Некоторые не представляют практического интереса, хотя и имеют название:

1. Константа нуля для любого аргумента вернёт 0: $f_0(x) = 0$.
2. Константа единицы для любого аргумента вернет 1: $f_3(x) = 1$,
3. Тождественная функция вернет значение аргумента: $f_1(x) = x$,
4. Инверсия вернет противоположное значение аргумента: $f_2(x) = \overline{x}$.

Функции одного аргумента

Отрицание, инверсия, *НЕ*, *NOT*

x_1	$не(x_1)$
0	1
1	0



НЕ

«$НЕ(x)$» также обозначается: «$\neg x$», «$\overline{x}$».

Функции двух аргументов

x_1	x_2	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Из $2^{2^2} = 16$ функций двух аргументов лишь 9 имеют название (см., например, [5]).

Функции двух аргументов I

1. $f_0(x_1, x_2) = 0$ – константа нуля
2. $f_1(x_1, x_2) = x_1 x_2$ – коньюнкция
3. $f_2(x_1, x_2) = x_1 \overline{x_2} = \overline{x_1 \to x_2} = x_1 \mapsto x_2$ – левая коимпликация
4. $f_3(x_1, x_2) = x_1 \overline{x_2} \vee x_1 x_2 = x_1$
5. $f_4(x_1, x_2) = \overline{x_1} x_2 = \overline{x_1 \vee \overline{x_2}} = \overline{x_1 \leftarrow x_2} = x_1 \Lsh x_2$ – правая коимпликация
6. $f_5(x_1, x_2) = \overline{x_1} x_2 \vee x_1 x_2 = x_2$
7. $f_6(x_1, x_2) = \overline{x_1} x_2 \vee x_1 \overline{x_2} = x_1 \oplus x_2$ – сложение по модулю 2
8. $f_7(x_1, x_2) = x_1 \vee x_2$ – дизъюнкция
9. $f_8(x_1, x_2) = \overline{x_1}\ \overline{x_2} = \overline{x_1 \vee x_2} = x_1 \uparrow x_2$ – стрелка Пирса (функция Вебба)
10. $f_9(x_1, x_2) = \overline{x_1}\ \overline{x_2} \vee x_1 x_2 = x_1 \sim x_2$ – эквивалентность
11. $f_{10}(x_1, x_2) = \overline{x_2}$ – отрицание

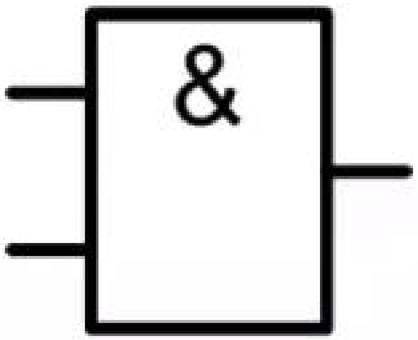
Функции двух аргументов II

12. $f_{11}(x_1, x_2) = \overline{x_2} \vee x_1 = x_1 \leftarrow x_2$ – правая импликация
13. $f_{12}(x_1, x_2) = \overline{x_1}$ – отрицание
14. $f_{13}(x_1, x_2) = \overline{x_1} \vee x_2 = x_1 \rightarrow x_2$ – левая импликация
15. $f_{14}(x_1, x_2) = \overline{x_1} \vee \overline{x_2} = \overline{x_1 x_2} = x_1 \mid x_2$ – функция Шеффера
16. $f_{15}(x_1, x_2) = 1$ – константа единицы

Функции двух аргументов

Конъюнкция, *И*, *AND*

x_1	x_2	$и(x_1, x_2)$
0	0	0
0	1	0
1	0	0
1	1	1



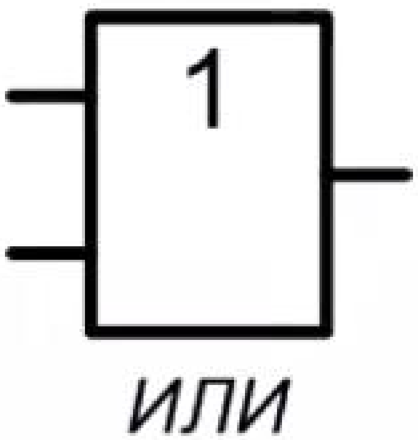
И

«$и(x, y)$» также обозначается: «$x \wedge y$», «$x \& y$», «$x \cdot y$» или «xy».

Функции двух аргументов

Дизъюнкция, *ИЛИ*, *OR*

x_1	x_2	$или(x_1, x_2)$
0	0	0
0	1	1
1	0	1
1	1	1

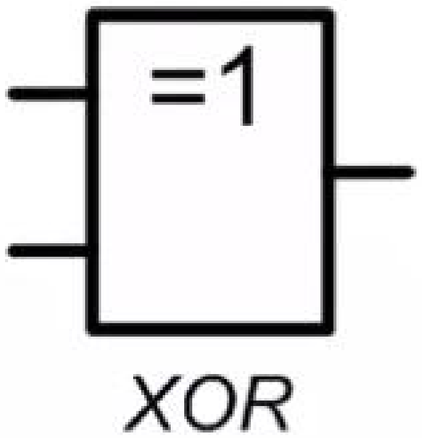


«$или(x, y)$» также обозначается: «$x \vee y$».

Функции двух аргументов

«Исключающее или», «сложение по модулю два», *XOR* — eXclusive OR

x_1	x_2	$xor(x_1, x_2)$
0	0	0
0	1	1
1	0	1
1	1	0



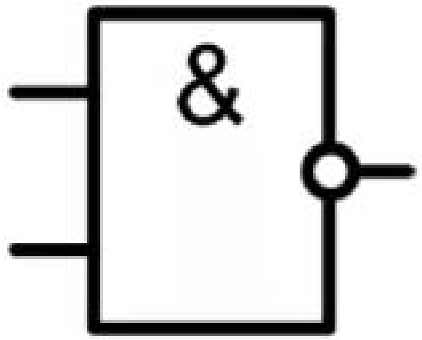
«$xor(x, y)$» также обозначается: «$x \oplus y$».
Это действительно *ИЛИ*, за исключением[1] того, что $xor(1, 1) = 0$.

[1] Гораздо проще запомнить эту функцию, как результат сложения в 2 СС двух бит с отброшенным переносом

x_1	x_2	*если-то*(x_1, x_2)
0	0	1
0	1	1
1	0	0
1	1	1

«*если-то*(x, y)» также обозначается: «$x \to y$».
Это аналог высказывания «если x_1, то x_2». Оно ложно лишь тогда, когда посылка x_1 истинна, а следствие x_2 ложно.

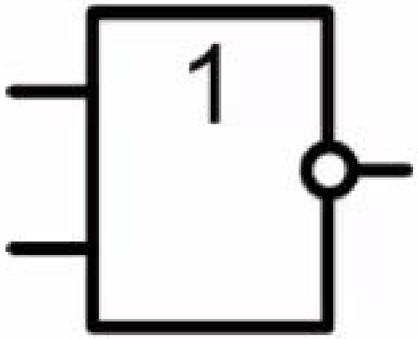
x_1	x_2	*и-не*(x_1, x_2)
0	0	1
0	1	1
1	0	1
1	1	0



И-НЕ

«*и-не*(x, y)» также обозначается: «$x \mid y$».

x_1	x_2	*или-не*(x_1, x_2)
0	0	1
0	1	0
1	0	0
1	1	0



ИЛИ-НЕ

«*или-не*(x, y)» также обозначается: «$x \uparrow y$».

Функции трёх и более аргументов

Не имеет смысла рассматривать функции трех и большего количества аргументов, в силу того, что их можно выразить формулой, сконструированной из функций одного и/или двух аргументов.

Формула

Формулой будем называть выражение

$$f(t_1, \ldots, t_n),$$

где t_i — **подформула**, т.е. либо аналогичного вида формула, либо переменная, принимающая одно из значений 0, либо 1.
Прежде чем найти значение формулы, нужно найти значения подформул, стоящих в аргументах.

Пример формулы

Example (Задача)

Найти значение формулы

$$или(0, и(не(1), 1)).$$

Решение.

$или(0, и(не(1), 1)) \Rightarrow или(0, и(0, 1)) \Rightarrow или(0, 0) \Rightarrow 0$ $\square$

Конструирование функций формулами

Функции произвольного количества аргументов можно конструировать на основе вышеназванных функций одного и двух аргументов.

Example

Функция трёх аргументов задана формулой:

$$g(x_1, x_2, x_3) = \textit{или}(x_1, \textit{и}(\textit{не}(x_2), x_3)).$$

Символы операций вместо функций

Вместо

1. «*не*(x)» пишут «$(\neg x)$» или «$(\bar{x})$»;
2. «*и*(x, y)» пишут «$(x \wedge y)$», «$(x \& y)$», «$(x \cdot y)$» или «(xy)»;
3. «*и-не*(x, y)» пишут «$(x \mid y)$»;
4. «*или*(x, y)» пишут «$(x \vee y)$»;
5. «*или-не*(x, y)» пишут «$(x \uparrow y)$».
6. «*xor*(x, y)» пишут «$(x \oplus y)$»;
7. «*если-то*(x, y)» пишут «$(x \to y)$»;

Задавая приоритет операций, лишние скобки опускают.

Example

«$\neg x \vee y \cdot z$» то же самое, что «$(\neg x) \vee (y \cdot z)$».

Операции вместо функций

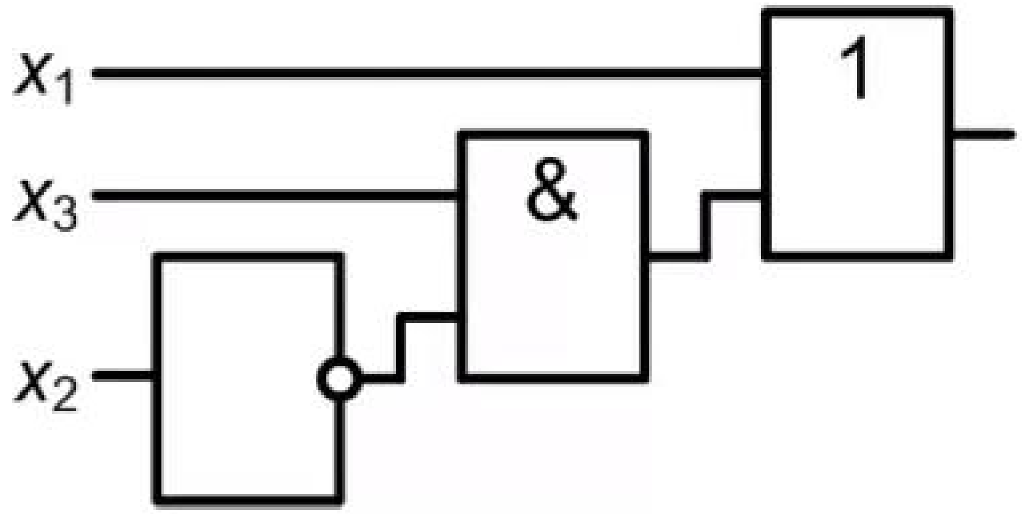
Example

Функция

$$g(x_1, x_2, x_3) = \textit{или}(x_1, \textit{и}(\textit{не}(x_2), x_3))$$

может быть записана с помощью обозначений операций:

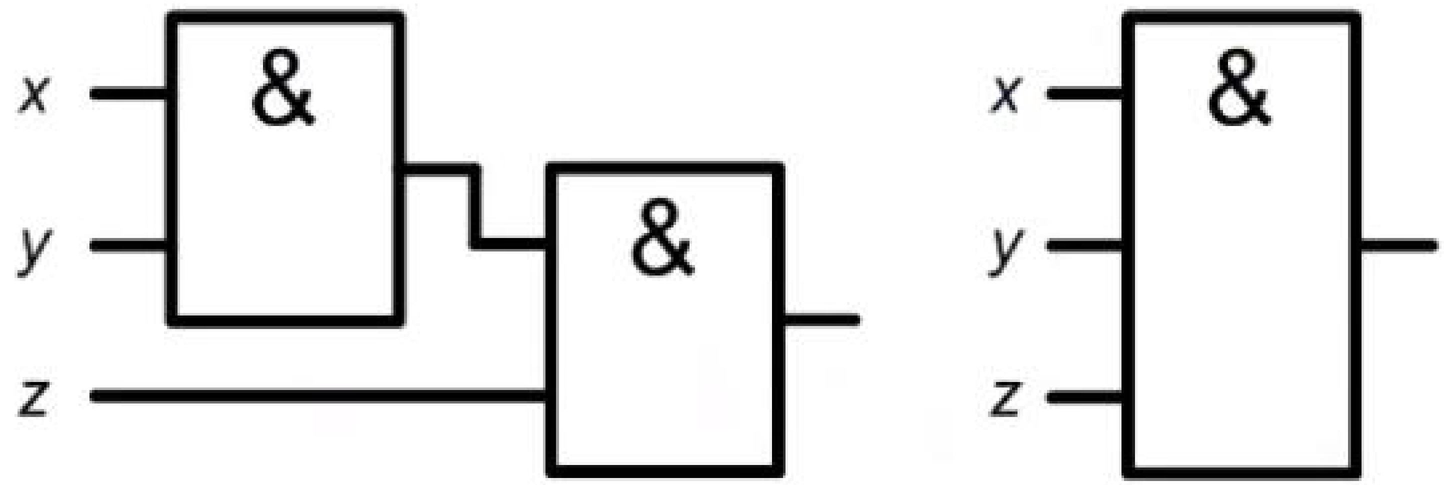
$$g(x_1, x_2, x_3) = x_1 \vee \neg x_2 \cdot x_3$$



Такие операции как *И*, *ИЛИ*, *XOR* ассоциативны. То есть, например,

$$(x \cdot y) \cdot z = x \cdot (y \cdot z) = x \cdot y \cdot z.$$

Поэтому на схемах допустимы трех и более-входовые элементы[2]:



[2]Которым соответствуют микросхемы, шаблоны ПЛИС и т.д.

1. Ассоциативность:

$$x_1 \cdot (x_2 \cdot x_3) = (x_1 \cdot x_2) \cdot x_3; \quad x_1 \vee (x_2 \vee x_3) = (x_1 \vee x_2) \vee x_3.$$

2. Коммутативность:

$$x_1 \cdot x_2 = x_2 \cdot x_1; \quad x_1 \vee x_2 = x_2 \vee x_1.$$

3. Дистрибутивность:

$$x_1 \cdot (x_2 \vee x_3) = (x_1 \cdot x_2) \vee (x_1 \cdot x_3); \quad x_1 \vee (x_2 \cdot x_3) = (x_1 \vee x_2) \cdot (x_1 \vee x_3).$$

4. Идемпотентность:

$$x \cdot x = x; \quad x \vee x = x.$$

5. Двойное отрицание:

$$\overline{\overline{x}} = x.$$

Алгебра логики II

Свойства операций

6. Свойства констант:

$$x \cdot 1 = x; \qquad x \cdot 0 = 0; \qquad x \vee 1 = 1$$
$$x \vee 0 = x; \qquad \overline{0} = 1; \qquad \overline{1} = 0$$

7. Закон де Моргана:

$$\overline{x_1 \cdot x_2} = \overline{x_1} \vee \overline{x_2}; \quad \overline{x_1 \vee x_2} = \overline{x_1} \cdot \overline{x_2}.$$

8. Закон противоречия:

$$x \cdot \overline{x} = 0.$$

9. Закон «исключённого третьего»:

$$x \vee \overline{x} = 1.$$

1. Поглощение:
$$x \vee (x \cdot y) = x; \quad x \cdot (x \vee y) = x.$$
2. Склеивание:
$$(x \cdot y) \vee (x \cdot \overline{y}) = x.$$
3. Обобщённое склеивание:
$$(x \cdot z) \vee (y \cdot \overline{z}) \vee (x \cdot y) = (x \cdot z) \vee (y \cdot \overline{z});$$
$$x \vee (\overline{x} \cdot y) = x \vee y;$$
$$x_1 \vee f(x_1, x_2, \ldots, x_n) = x_1 \vee (\overline{x_1} \cdot f(0, x_2, \ldots, x_n)) \vee$$
$$\vee (x_1 \cdot f(1, x_2, \ldots, x_n)) = x_1 \vee f(0, x_2, \ldots, x_n).$$

Основной логический базис

Функции *И*, *ИЛИ*, *НЕ* составляют основной логический базис. Любую функцию можно выразить формулой на их основе. Эти логические связки часто используются в обыденной жизни.

Example (Задача)

Представить функцию $f(x_1, x_2, x_3)$ в основном логическом базисе.

x_1	x_2	x_3	$f(x_1, x_2, x_3)$
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

Решение.

На наборе $x_1 = 0$, $x_2 = 1$, $x_3 = 0$ функция $f(x_1, x_2, x_3)$ должна равняться единице. При этом видно, что функция $f_2(x_1, x_2, x_3) = \overline{x_1} \cdot x_2 \cdot \overline{x_3}$ равняется единице только на этом наборе. $f(x_1, x_2, x_3)$ также должна равняться 1 на наборе $x_1 = 1$, $x_2 = 1$, $x_3 = 0$. $f_6(x_1, x_2, x_3) = x_1 \cdot x_2 \cdot \overline{x_3}$. Стало быть[a]
$f(x_1, x_2, x_3) = f_2(x_1, x_2, x_3) \vee f_6(x_1, x_2, x_3) = (\overline{x_1} \cdot x_2 \cdot \overline{x_3}) \vee (x_1 \cdot x_2 \cdot \overline{x_3})$. □

---

[a] Конечно, это решение можно оптимизировать. Вдумавшись, можно видеть, что от x_1 функция не зависит: $f(x_1, x_2, x_3) = x_2 \cdot \overline{x_3}$. Такое представление называется ДНФ — дизъюнктивная нормальная форма. Оптимизацию формул здесь не рассматриваем

Функционально полные системы булевых функций

Функционально полной системой булевых функций называется совокупность таких функций, что произвольная булева функция может быть записана в виде формулы через функции этой совокупности.

$f(x_1, ..., x_n)$

1. булевы функции, сохраняющие константу 0 $f(0, 0, ..., 0) = 0$;
2. булевы функции, сохраняющие константу 1 $f(1, 1, ..., 1) = 1$;
3. самодвойственные булевы функции $f(x_1, ..., x_n) = \overline{f(\overline{x_1}, ..., \overline{x_n})}$;
4. линейные булевы функции $f(x_1, ..., x_n) = c_0 \oplus c_1 x_1 \oplus ... \oplus c_n x_n$, где $c_i \in [0; 1]$;
5. монотонные булевы функции $f(x_1, ..., x_n) \geqslant f(y_1, ..., y_n)$, если $\langle x_1, ..., x_n \rangle \geqslant \langle y_1, ..., y_n \rangle$.

Критерий полноты Поста-Яблонского

Система булевых функций является полной тогда и только тогда, когда она:

- содержит функцию, не принадлежащую классу K_0;
- содержит функцию, не принадлежащую классу K_1;
- содержит функцию, не принадлежащую классу K_C;
- содержит функцию, не принадлежащую классу K_L;
- содержит функцию, не принадлежащую классу K_M.

Некоторые логические базисы

$\{\uparrow\}$ – ИЛИ-НЕ – базис Пирса (базис Вебба)
$\{|\}$ – И-НЕ – базис Шеффера
$\{\to, 0\}$, $\{\to, \neg\}$ – импликативные базисы
$\{\&, \neg\}$ – И, НЕ – коньюнктивный базис Буля
$\{\vee, \neg\}$ – ИЛИ, НЕ – дизьюнктивный базис Буля
$\{\oplus, \&, \neg\}$ – базис Жегалкина

Например, функцию *И* можно выразить через функции *ИЛИ* и *НЕ*:

$$x_1 \cdot x_2 = \overline{\overline{x_1} \vee \overline{x_1}},$$

в чём нетрудно убедиться непосредственной проверкой:

$x_1 \cdot x_2$	x_1	x_2	$\overline{\overline{x_1} \vee \overline{x_1}}$
0	0	0	0
0	0	1	0
0	1	0	0
1	1	1	1

Следовательно, базис образуют функции *ИЛИ* и *НЕ*!

Основной логический базис

Избыточность основного логического базиса

Например, функцию *И* можно выразить через функции *ИЛИ* и *НЕ*:

$$x_1 \cdot x_2 = \overline{\overline{x_1} \vee \overline{x_1}},$$

в чём нетрудно убедиться непосредственной проверкой:

$x_1 \cdot x_2$	x_1	x_2	$\overline{\overline{x_1} \vee \overline{x_1}}$
0	0	0	0
0	0	1	0
0	1	0	0
1	1	1	1

Следовательно, базис образуют функции *ИЛИ* и *НЕ*!

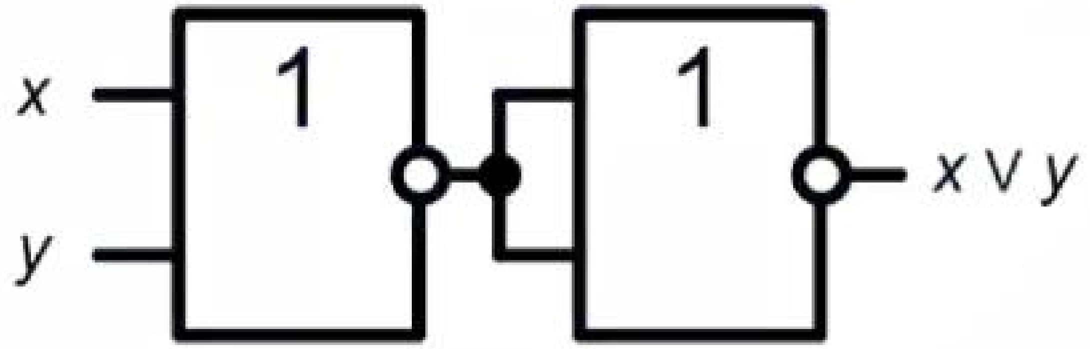
Базис из одной функции

Стрелка Пирса, *ИЛИ-НЕ*

x_1	x_2	$x_1 \uparrow x_2$
0	0	1
0	1	0
1	0	0
1	1	0

Достаточно выразить, например функции *НЕ* и *ИЛИ*:

- *НЕ*: $\neg x = x \uparrow x$;
- *ИЛИ*: $x \vee y = \neg(x \uparrow y) = (x \uparrow y) \uparrow (x \uparrow y)$;



Оператор языка C/C++	Действие оператора
`x=y`	Присвоить переменной `x` значение `y`
`x==y`	Сравнить значения переменных `x` и `y`
`~x`	Битовое *НЕ*
`x|y`	Битовое *ИЛИ*
`x&y`	Битовое *И*
`x^y`	Битовое *XOR*
`x<<n`	Сдвиг значения `x` на `n` бит влево
`x>>n`	Сдвиг значения `x` на `n` бит вправо

	7	6	5	4	3	2	1	0
x	0	0	1	1	1	1	0	0
y	0	1	0	1	1	0	1	0

z1 = ~x	1	1	0	0	0	0	1	1
z2 = x|y	0	1	1	1	1	1	1	0
z3 = x&y	0	0	0	1	1	0	0	0
z4 = x^y	0	1	1	0	0	1	1	0
z5 = x<<1	0	1	1	1	1	0	0	0
z6 = x>>2	0	0	0	0	1	1	1	1

В заключение

Основы алгебры логики даны практически во всех учебниках.
Фундаментальное изложение можно найти в [1, 6].
С точки зрения программиста в [5, 2].
Доступно и кратко в [3].
Работа над битами с помощью арифметико-логических команд процессора [4].

Библиография I

- В.А.Горбатов. Дискретная математика: учебник для студентов ВТУЗов / В.А.Горбатов, А.В.Горбатов, М.В.Горбатова. — М.: ООО «Издательство АСТ» and ООО «Издательство Астрель», 2003.
- Г.Хаггард. Дискретная математика для программистов: учебное пособие / Г.Хаггард, Дж.Шлипф, С.Уайтсайдс. — М.: Бином. Лаборатория знаний, 2010.
- С.В.Судоплатов. Дискретная математика: Учебник / С.В.Судоплатов, Е.В.Овчинникова. — М.: ИНФРА-М;Новосибирск;Изд-во НГТУ, 2005.
- Г.Уоррен-мл. Алгоритмические трюки для программистов / Г.Уоррен-мл. — 2 изд. — М.: Издательский дом «Вильямс», 2014.

Библиография II

- Ф.А.Новиков. Дискретная математика для программистов / Ф.А.Новиков. — СПб.: Питер, 2000.
- С.В.Яблонский. Введение в дискретную математику: учебное пособие для вузов / С.В.Яблонский; Под ред. В.А.Садовничего. — М.: Высшая школа, 2001.