Логические основы компьютеров

Логические основы компьютеров

Логические выражения и операции

Булева алгебра

Двоичное кодирование – все виды информации кодируются с помощью 0 и 1.

Задача – разработать оптимальные правила обработки таких данных.

Джордж Буль разработал основы алгебры, в которой используются только 0 и 1 (алгебра логики, булева алгебра).

Почему «логика»?

Результат выполнения операции можно представить как истинность (1) или ложность (0) некоторого высказывания.

Логические высказывания

Логическое высказывание – это повествовательное предложение, относительно которого можно однозначно сказать, истинно оно или ложно.

Высказывание или нет?

- Сейчас идет дождь.
- Жирафы летят на север.
- История – интересный предмет.
- У квадрата – 10 сторон и все разные.
- Красиво!
- В городе N живут 2 миллиона человек.
- Который час?

Обозначение высказываний

A – Сейчас идет дождь.
B – Форточка открыта.

простые высказывания (элементарные)

! **Любое высказывание может быть ложно (0) или истинно (1).**

Составные высказывания строятся из простых с помощью логических связок (операций) «**и**», «**или**», «**не**», «**если … то**», «**тогда и только тогда**» и др.

A и B	Сейчас идет дождь и открыта форточка.
A или не B	Сейчас идет дождь или форточка закрыта.
если A, то B	Если сейчас идет дождь, то форточка открыта.
не A и B	Сейчас нет дождя и форточка открыта.
A тогда и только тогда, когда B	Дождь идет тогда и только тогда, когда открыта форточка.

Операция НЕ (инверсия)

Если высказывание **A** истинно, то «**не A**» ложно, и наоборот.

A	не A
0	1
1	0

также: $\overline{\mathbf{A}}$,
not A (Паскаль),
! A (Си)

таблица истинности операции НЕ

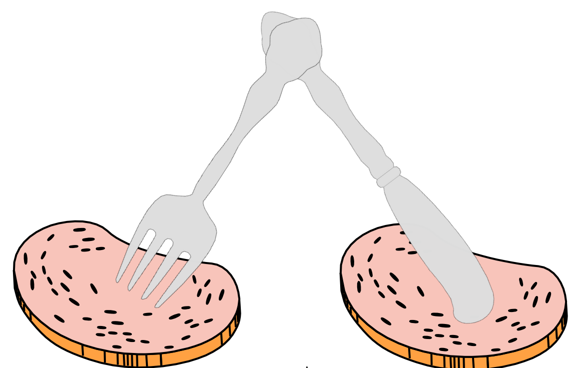
Таблица истинности логического выражения X – это таблица, где в левой части записываются все возможные комбинации значений исходных данных, а в правой – значение выражения X для каждой комбинации.

Операция И (логическое умножение, конъюнкция)

Высказывание «**А и B**» истинно тогда и только тогда, когда **А** и **B** истинны одновременно.

также: **A·B**, **A ∧ B**, **A and B** (Паскаль), **A && B** (Си)

	A	B	А и B
0	0	0	0
1	0	1	0
2	1	0	0
3	1	1	1



A ∧ B

конъюнкция – **от лат.** ***conjunctio*** **— соединение**

Операция ИЛИ (логическое сложение, дизъюнкция)

Высказывание «**A или B**» истинно тогда, когда истинно **A** или **B**, или оба вместе.

также: **A+B**, **A ∨ B**, **A or B** (Паскаль), **A || B** (Си)

A	B	A или B
0	0	0
0	1	1
1	0	1
1	1	1

дизъюнкция – от лат. *disjunctio* — разъединение

Импликация («если …, то …»)

Высказывание «**A → B**» истинно, если не исключено, что из **A** следует **B**.

A – «Работник хорошо работает».

B – «У работника хорошая зарплата».

A	B	A → B
0	0	1
0	1	1
1	0	0
1	1	1

$$A \to B = \overline{A} + B$$

Импликация («если …, то …»)

«Если Вася идет гулять, то Маша сидит дома».

A – «Вася идет гулять».

B – «Маша сидит дома».

$$A \to B = 1$$

A	B	$A \to B$
0	0	1
0	1	1
1	0	0
1	1	1

А если Вася не идет гулять?

Маша может пойти гулять (B=0), а может и не пойти (B=1)!

Эквиваленция («тогда и только тогда, …»)

Высказывание «**A** ↔ **B**» истинно тогда и только тогда, когда **A** и **B** равны.

A	B	A ↔ B
0	0	1
0	1	0
1	0	0
1	1	1

$$A \leftrightarrow B = \overline{A \oplus B} = A \cdot B + \overline{A} \cdot \overline{B}$$

Базовый набор операций

С помощью операций **И, ИЛИ** и **НЕ** можно реализовать любую логическую операцию.

И **ИЛИ** **НЕ**

базовый набор операций

Сколько всего существует логических операции с двумя переменными?

Логические формулы

Прибор имеет три датчика и может работать, если два из них исправны. Записать в виде формулы ситуацию «авария».

A – «Датчик № 1 неисправен».

B – «Датчик № 2 неисправен».

C – «Датчик № 3 неисправен».

Аварийный сигнал:

X – «Неисправны два датчика».

X – «Неисправны датчики № 1 **и** № 2» **или**
«Неисправны датчики № 1 **и** № 3» **или**
«Неисправны датчики № 2 **и** № 3».

$$X = A \cdot B + A \cdot C + B \cdot C$$

логическая формула

Составление таблиц истинности

$$X = A \cdot B + \overline{A} \cdot B + \overline{B}$$

	A	B	$A \cdot B$	$\overline{A} \cdot B$	$\overline{B}$	X
0	0	0	0	0	1	1
1	0	1	0	1	0	1
2	1	0	0	0	1	1
3	1	1	1	0	0	1

Логические выражения могут быть:

- **тождественно истинными** (всегда 1, тавтология)
- **тождественно ложными** (всегда 0, противоречие)
- **вычислимыми** (зависят от исходных данных)

Составление таблиц истинности

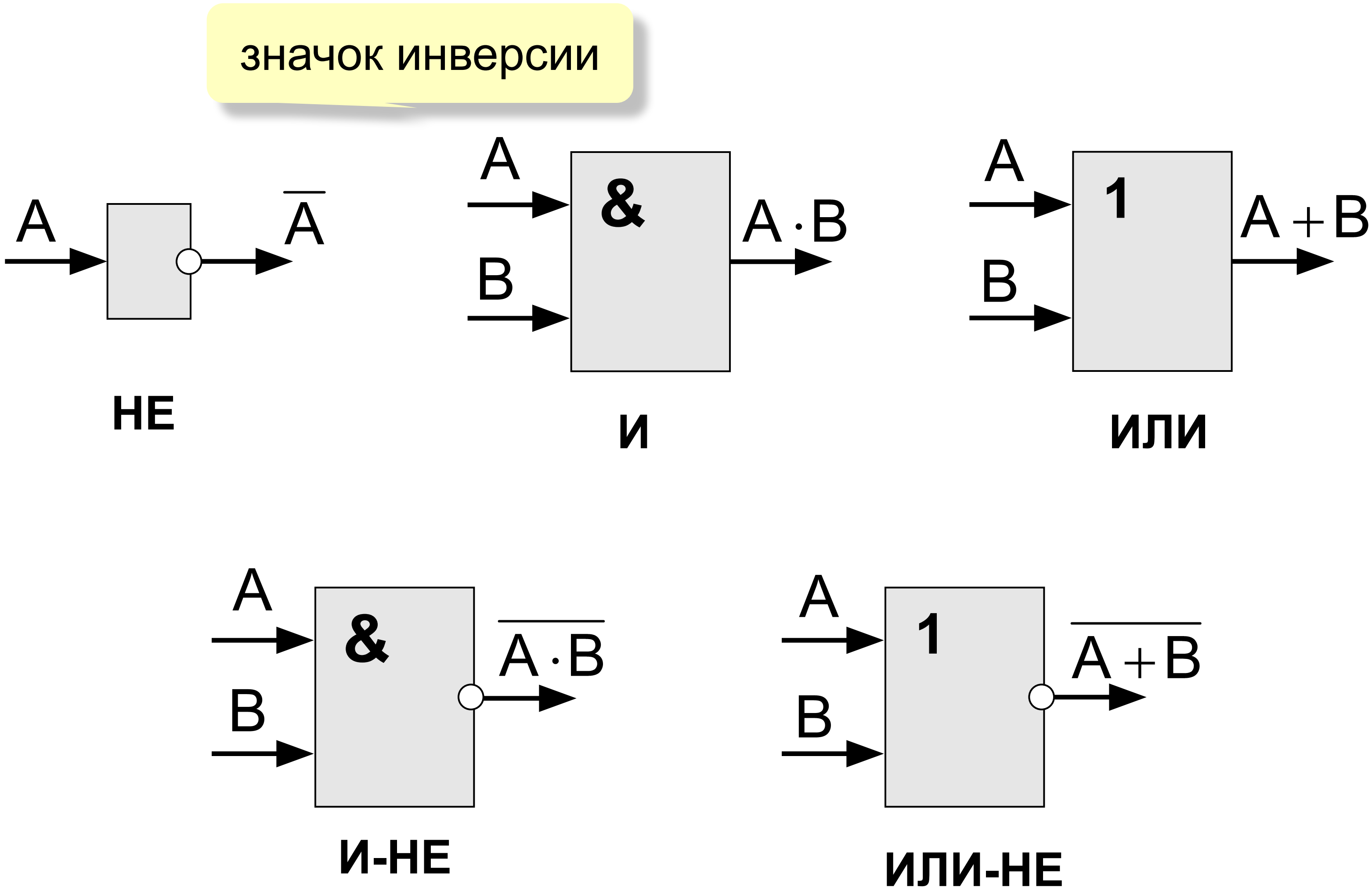
$$X = A \cdot B + A \cdot C + B \cdot C$$

	A	B	C	A·B	A·C	B·C	X
0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0
2	0	1	0	0	0	0	0
3	0	1	1	0	0	1	1
4	1	0	0	0	0	0	0
5	1	0	1	0	1	0	1
6	1	1	0	1	0	0	1
7	1	1	1	1	1	1	1

Логические основы компьютеров

Логические элементы компьютера

Логические элементы компьютера

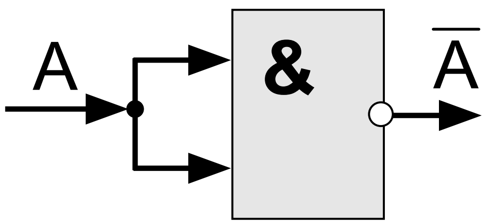


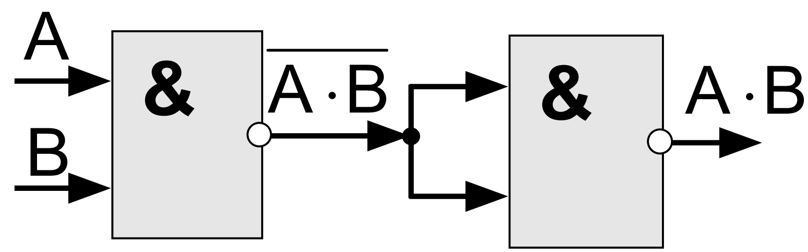
Логические элементы компьютера

Любое логическое выражение можно реализовать на элементах **И-НЕ** или **ИЛИ-НЕ.**

НЕ: $\overline{A} = \overline{A} + \overline{A} = \overline{A \cdot A}$

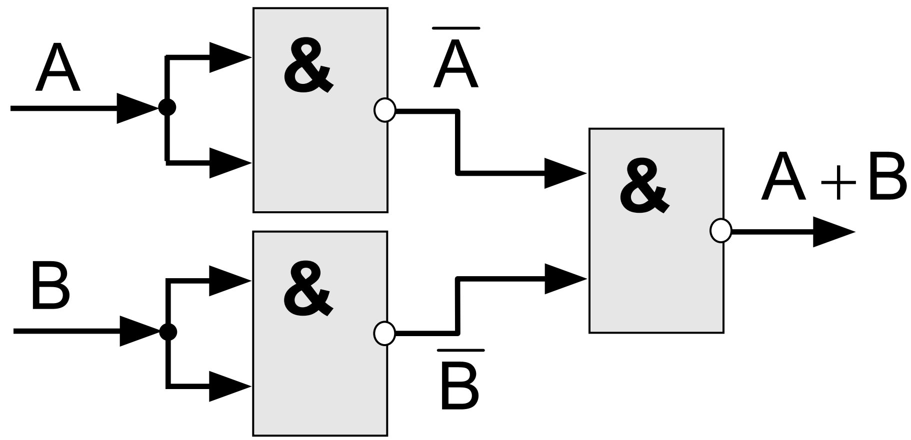
И: $A \cdot B = \overline{\overline{A \cdot B}}$





ИЛИ:

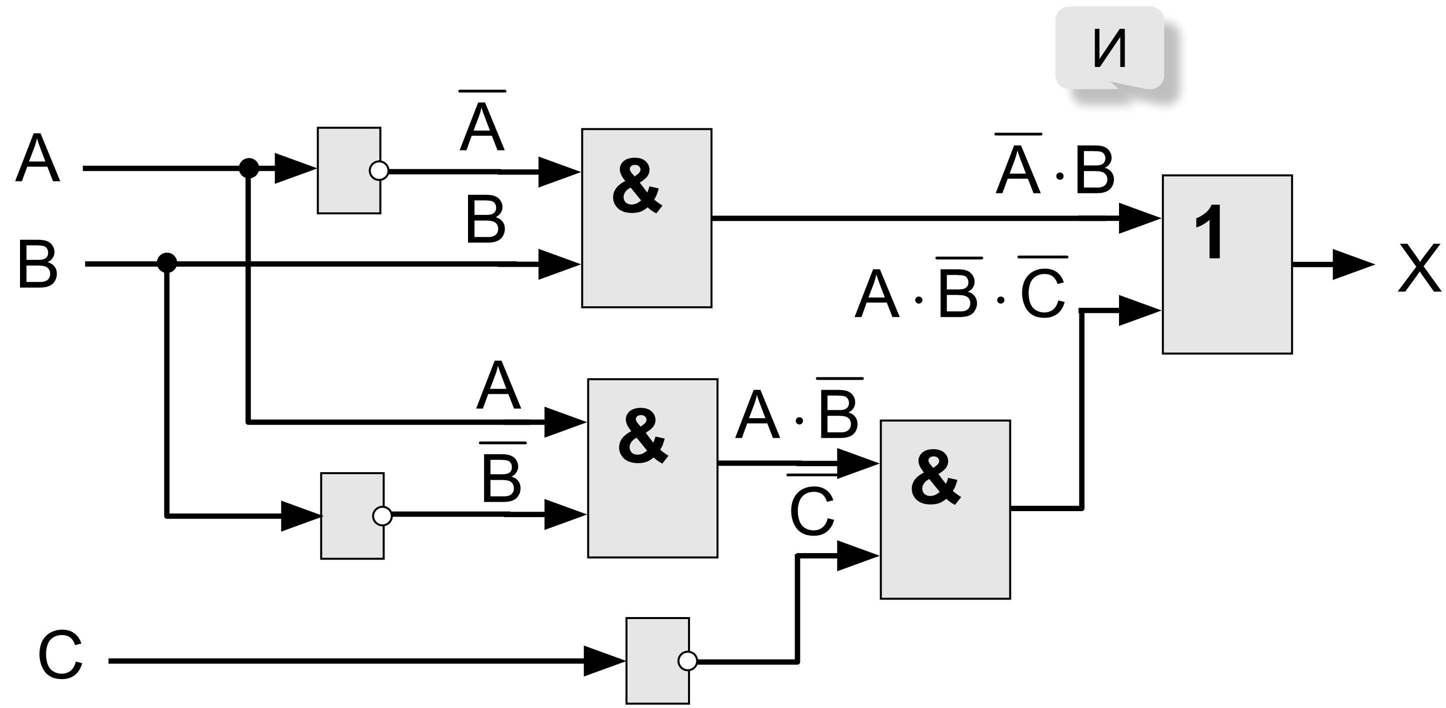
$A + B = \overline{\overline{A} \cdot \overline{B}}$



Составление схем

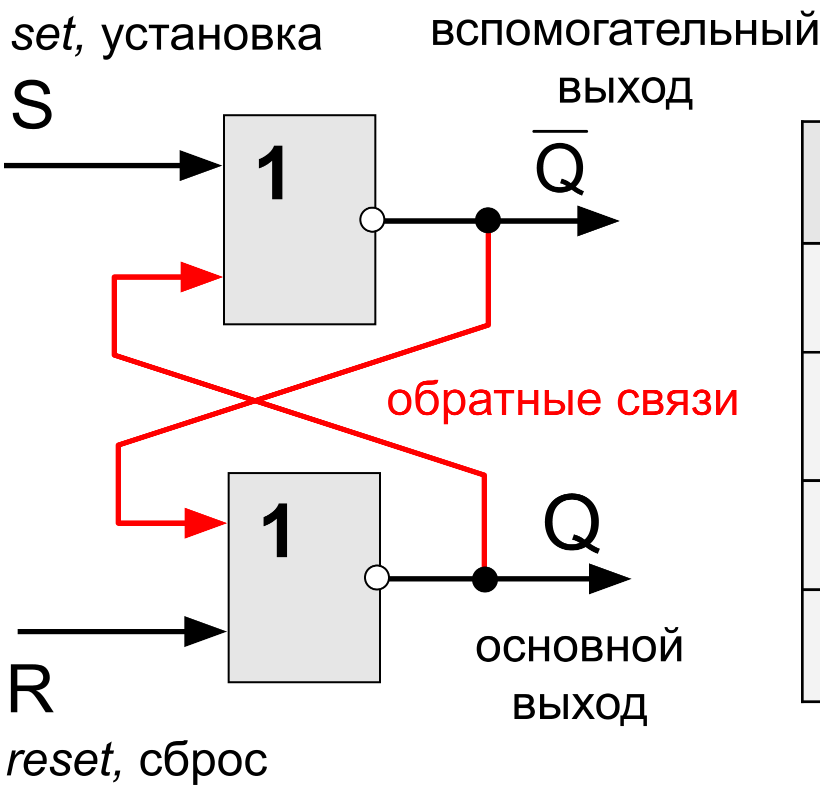
последняя операция - ИЛИ

$$X = \overline{A} \cdot B + A \cdot \overline{B} \cdot \overline{C}$$



Триггер (англ. *trigger* – защёлка)

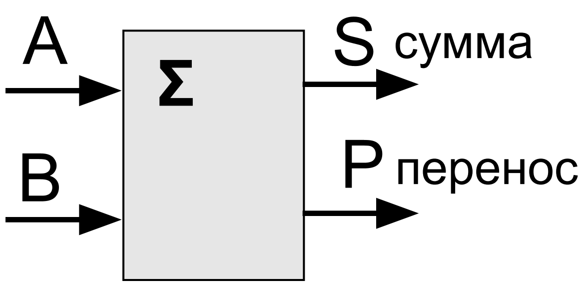
Триггер – это логическая схема, способная хранить 1 бит информации (1 или 0). Строится на 2-х элементах **ИЛИ-НЕ** или на 2-х элементах **И-НЕ.**



S	R	Q	$\overline{Q}$	режим
0	0	Q	$\overline{Q}$	хранение
0	1	0	1	сброс
1	0	1	0	установка 1
1	1	0	0	запрещен

Полусумматор

Полусумматор – это логическая схема, способная складывать два одноразрядных двоичных числа.



$$P = A \cdot B$$

$$S = A \oplus B = A \cdot \overline{B} + \overline{A} \cdot B$$

A	B	P	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

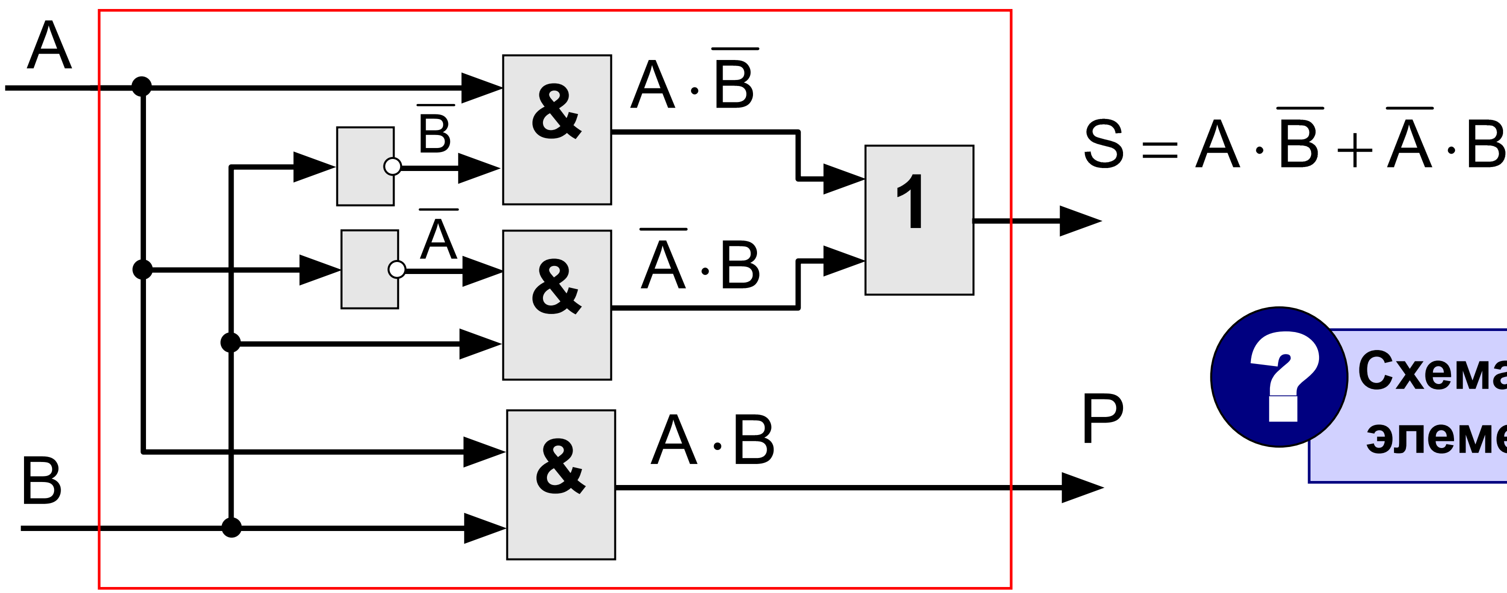
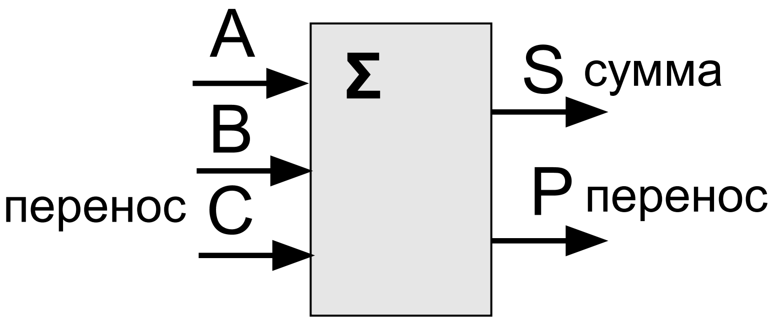


Схема на 4-х элементах?

Сумматор

Сумматор – это логическая схема, способная складывать два одноразрядных двоичных числа с переносом из предыдущего разряда.



A	B	C	P	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Многоразрядный сумматор

это логическая схема, способная складывать два n-разрядных двоичных числа.

$$
\begin{array}{rl}
 & A = \quad a_n \; a_{n-1} \; \ldots \; a_1 \\
+ & B = \quad b_n \; b_{n-1} \; \ldots \; b_1 \\
\hline
 & C = \boxed{p} \; c_n \; c_{n-1} \; \ldots \; c_1
\end{array}
$$

перенос