Белорусско-Российский университет
Кафедра «Программное обеспечение информационных технологий»

## Информатика

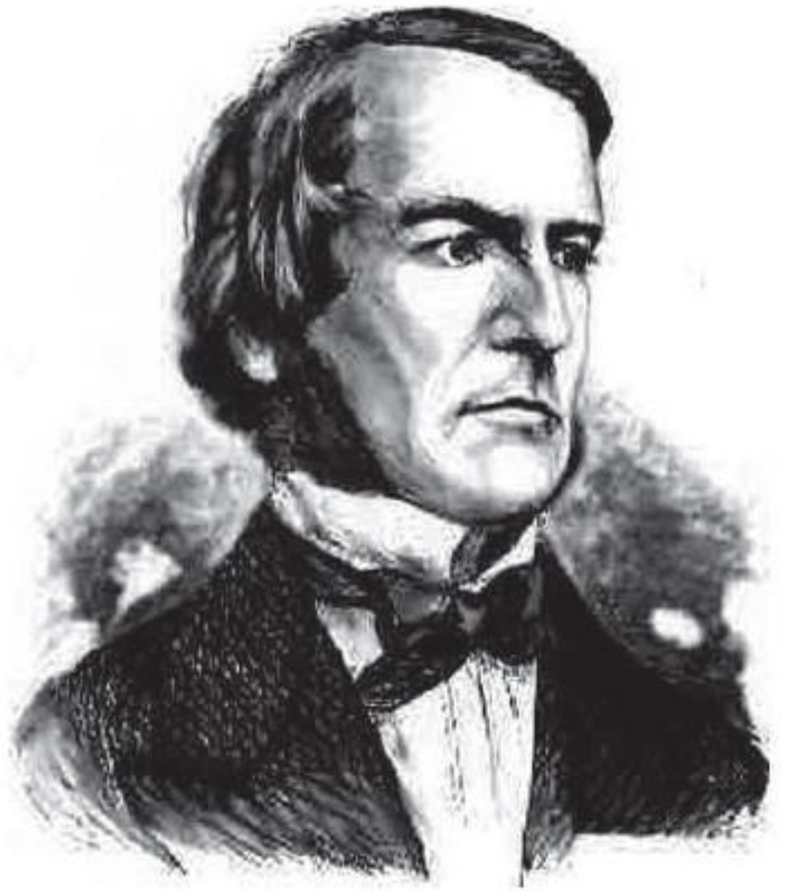
# Логические основы компьютерной техники

**КУТУЗОВ Виктор Владимирович**

Республика Беларусь, Могилев, 2023

# Элементы алгебры логики

# Логика и компьютер



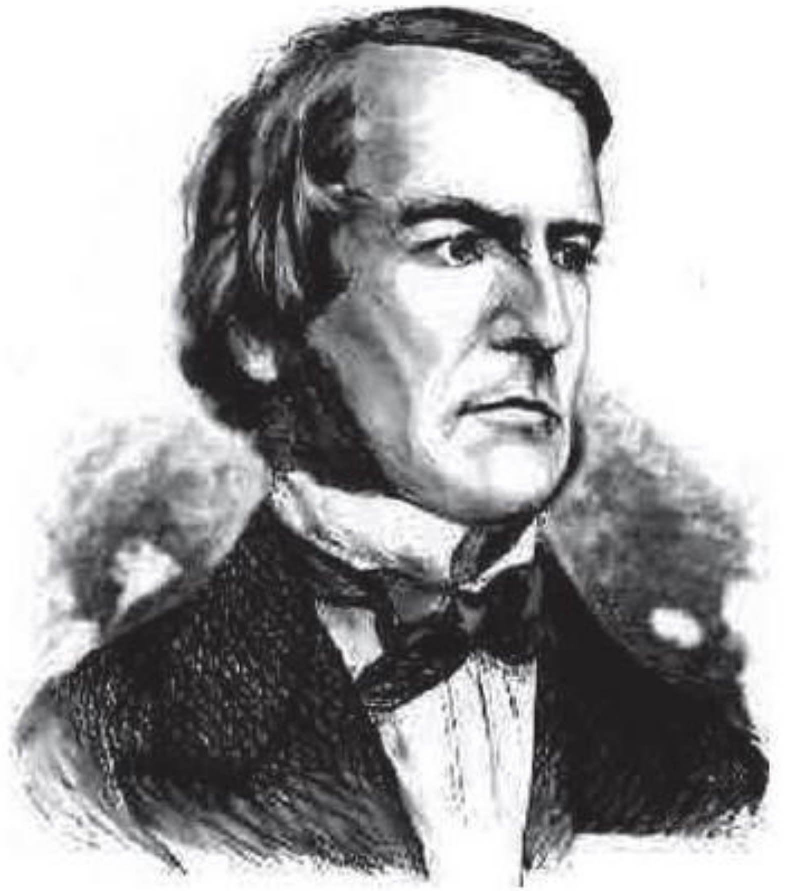
Джордж Буль
(1815–1864)

**Для описания алгоритмов работы цифровых устройств необходим соответствующий математический аппарат.**

Такой аппарат для решения задач формальной логики в середине XIX века разработал ирландский математик Джордж Буль.

По его имени математический аппарат и получил название **булевой алгебры, или алгебры логики**.

# Логика и компьютер



Джордж Буль
(1815–1864)

**Джордж Буль (1815–1864)** — английский математик, основоположник алгебры логики.

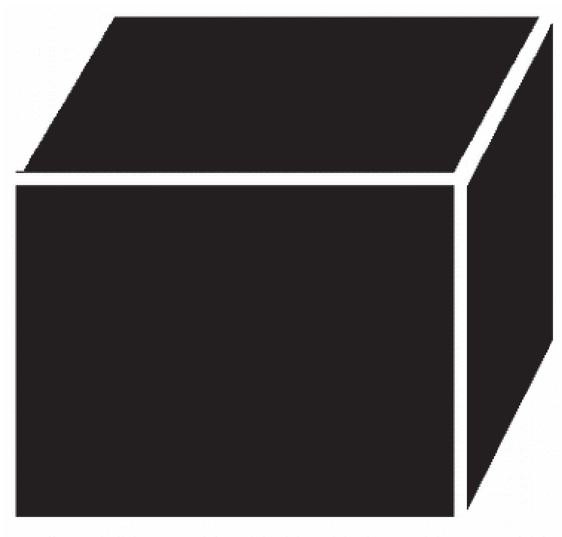
Дж. Буль изучал логику мышления математическими методами и разработал алгебраические методы решения традиционных логических задач.

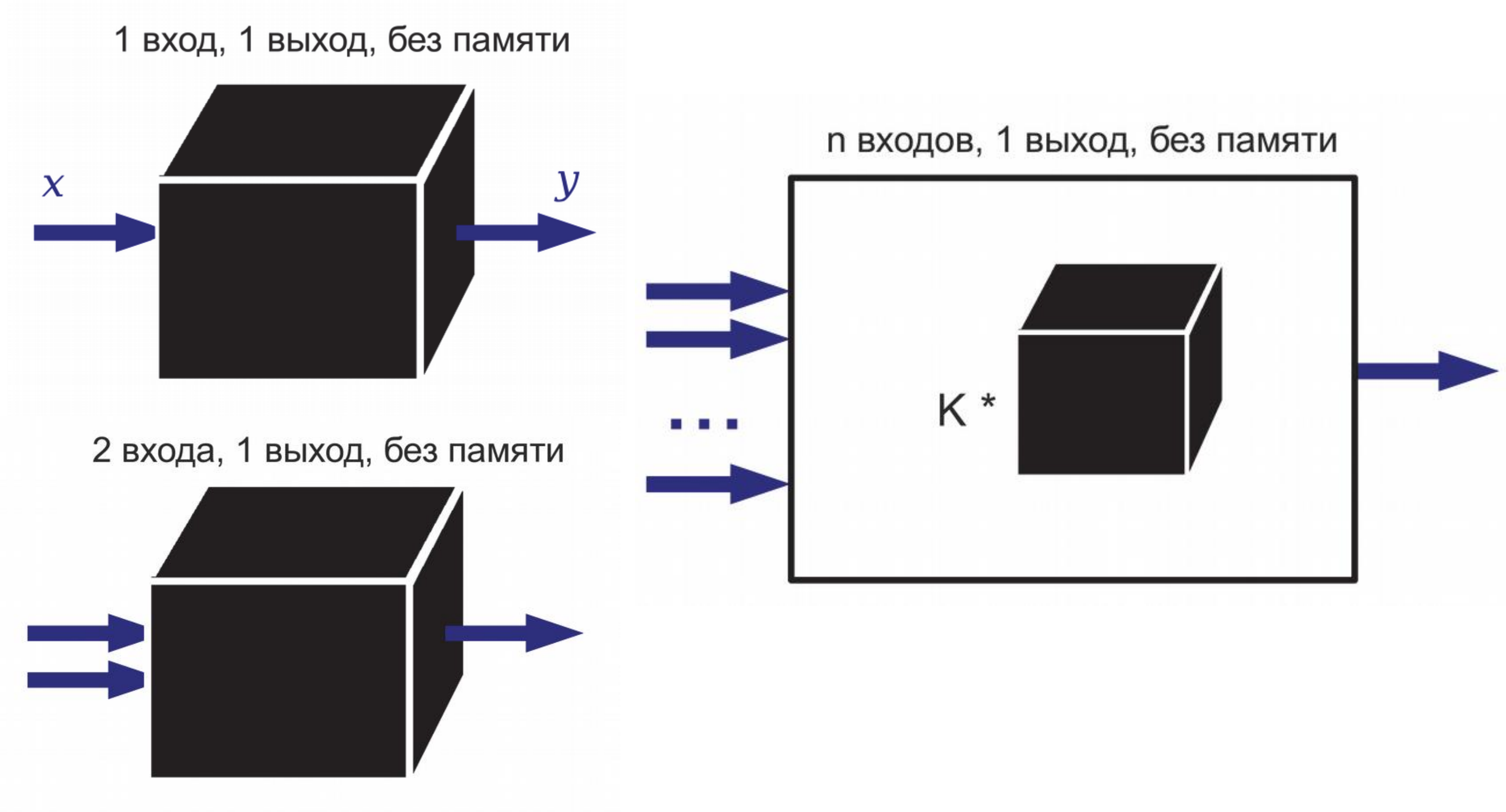
**В 1854 году он опубликовал работу, в которой изложил суть алгебры логики, основанной на трёх операциях: AND, OR, NOT.**

В 1938 году Клод Шеннон применил алгебру логики для описания процесса функционирования релейно-контактных и электронно-ламповых схем.

# Логика и компьютер

- Созданный Д. Булем аппарат математической логики позже стал носить название **булевой алгебры**.
- Фактически он позволил представить современный компьютер в виде «чёрного ящика».





# Логика и компьютер

- **Алгебра логики** — раздел математики, изучающий высказывания, рассматриваемые с точки зрения их логических значений (истинности или ложности), и логические операции над ними.

**Булева алгебра** — это математическая система, оперирующая двумя понятиями: «**событие истинно**» и «**событие ложно**».

- Естественно ассоциировать эти понятия с цифрами, используемыми в двоичной системе счисления.
- Будем их называть соответственно логическими единицей (**лог. 1**) и нулем (**лог. 0**).
- Любое высказывание будем обозначать как **ложно** (**0**), **истинно** (**1**).

# Логические высказывания и переменные

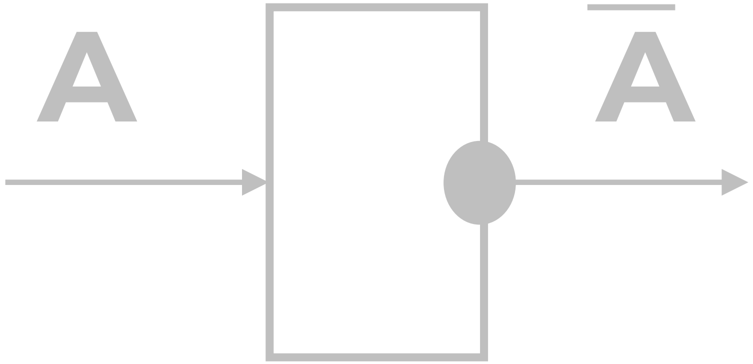
- **Высказывание** — это предложение, в отношении которого можно сказать, истинно оно или ложно.

- **Например**, высказывание «Джордж Буль — основоположник алгебры логики» истинно, а высказывание «2 + 2 = 5» ложно.

- Из имеющихся высказываний можно строить новые высказывания.
- Для этого используются логические связки — слова и словосочетания «не», «и», «или», «если ..., то», «тогда и только тогда» и др.

# Логические высказывания и переменные

- Обоснование истинности или ложности элементарных высказываний не является задачей алгебры логики. Эти вопросы решаются теми науками, к сфере которых относятся элементарные высказывания.
- Такое сужение интересов позволяет обозначать высказывания символическими именами (например, A, B, C).
- Так, если обозначить элементарное высказывание «Джордж Буль — основоположник алгебры логики» именем A, а элементарное высказывание «2 + 2 = 5» именем B, то составное высказывание «Джордж Буль — основоположник алгебры логики, и 2 + 2 = 5» можно записать как «A и B».
- **Здесь**
  **A, B — логические переменные,**
  **«и» — логическая связка.**




# Логические операции

# Логические высказывания и переменные

- **Для логических значений** «**истина**» и «**ложь**» могут использоваться следующие обозначения:

| Истина | Ложь |
|---|---|
| И | Л |
| true | false |
| да | нет |
| 1 | 0 |

- Истинность или ложность составных высказываний зависит от истинности или ложности образующих их высказываний и определённой трактовки связок (логических операций над высказываниями).

# Три основные логические операции

- Все возможные логические функции переменных можно образовать с помощью **трех основных операций:**
  - **Конъюнкция** (от лат. conjunctio союз, связь) — логическая операция, по своему применению максимально приближённая к союзу «и». Синонимы: **логическое «И»**, логическое умножение, или просто «И».
  - **Дизъюнкция** (лат. disjunctio — разобщение) — логическая операция, по своему применению максимально приближённая к союзу «или» в смысле «или то, или это, или оба сразу». Синонимы: **логическое «ИЛИ»**, включающее «ИЛИ», логическое сложение, иногда просто «ИЛИ».
  - **Отрицание в логике** — унарная операция над суждениями, результатом которой является суждение, «противоположное» исходному. Синоним: **логическое «НЕ»**, инверсия.

# «НЕ» - Инверсия - NOT

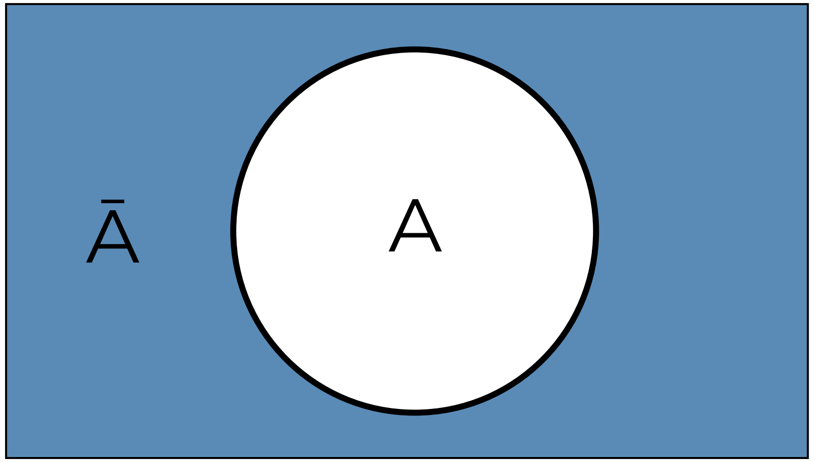
- **Операцию «НЕ»** называют отрицанием или инверсией (англ. inverse — обратный). В алгебре логики всего два возможных значения (0 и 1), поэтому логические отрицание — это переход от одного значения к другому, от 1 к 0 или наоборот. **Если высказывание А истинно, то НЕ А ложно, и наоборот.**

- **Обозначения: НЕ, not, ¬ , ‾ .**

**Таблица истинности**

| A | не A, (not A), $\bar{A}$ |
|---|---|
| 0 | **1** |
| 1 | **0** |

**Диаграмма Венна (круг Эйлера)**



# «И» - Конъюнкция - AND

- **Логическая схема «И»** называется также конъюнктором, выполняет операцию логического умножения (конъюнкции), может иметь от двух до восьми входов.
- **Обозначения: ∧ , ×, &, И, and.**

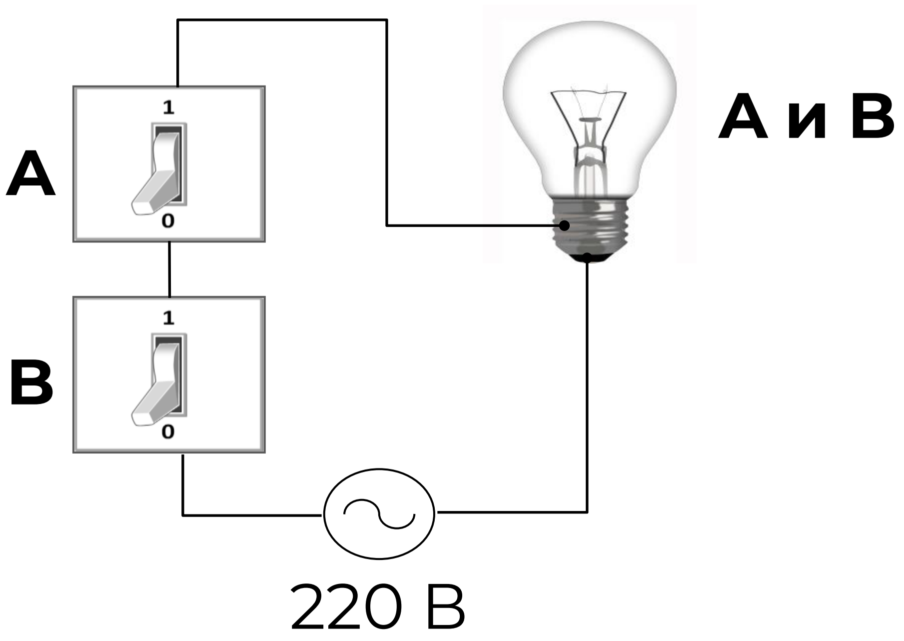
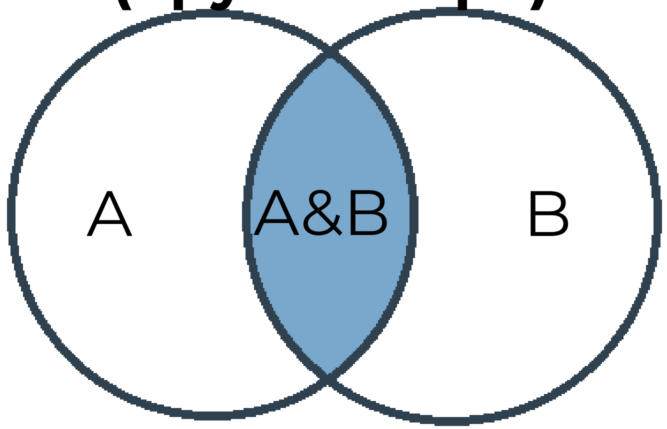



**Таблица истинности**

| A | B | F = A & B |
|---|---|---|
| 0 | 0 | **0** |
| 0 | 1 | **0** |
| 1 | 0 | **0** |
| 1 | 1 | **1** |

**Диаграмма Венна (круг Эйлера)**



# «ИЛИ» - Дизъюнкция - OR

- **Логическая схема «ИЛИ»** называется также дизъюнктором, выполняет операцию логического сложения (дизъюнкции), может иметь от двух до восьми входов.
- **Обозначения: V, |, ИЛИ, or, +.**

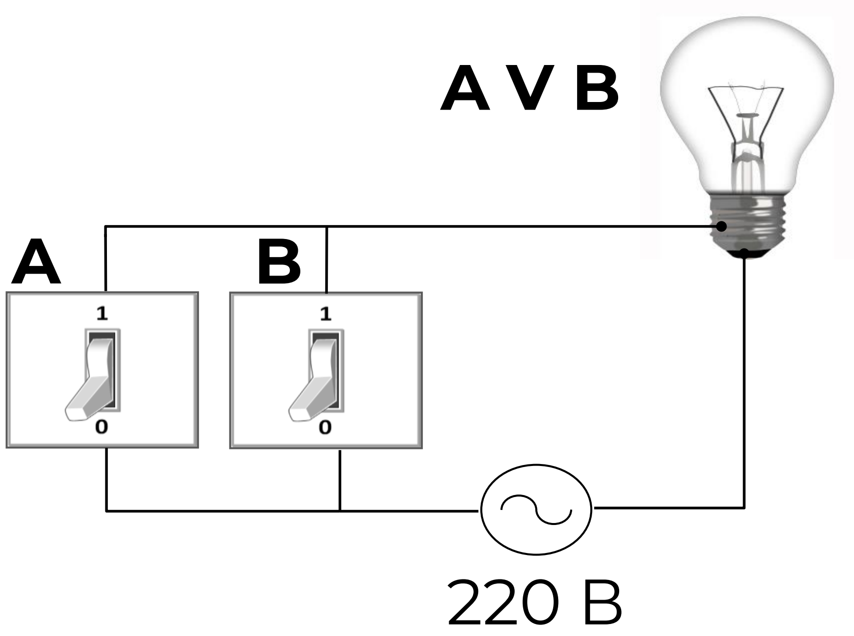
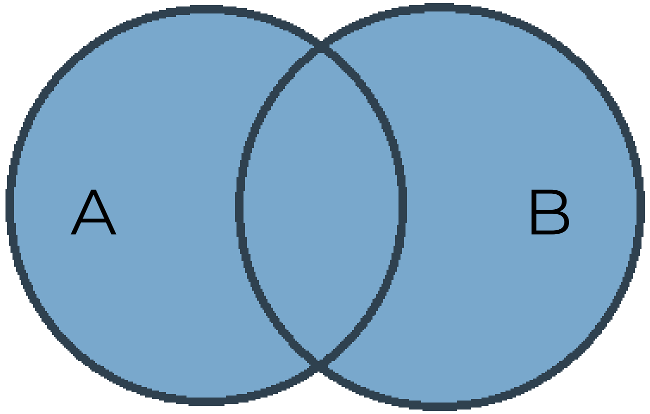



**Таблица истинности**

| A | B | F = A V B |
|---|---|---|
| 0 | 0 | **0** |
| 0 | 1 | **1** |
| 1 | 0 | **1** |
| 1 | 1 | **1** |

**Диаграмма Венна (круг Эйлера)**




**A V B**

# И, ИЛИ, НЕ - базовый набор логических операций

И | ИЛИ | НЕ

**Любая логическая операция может быть представлена в виде комбинации трех базовых (И, ИЛИ, НЕ)**, поэтому любые устройства компьютера, производящие обработку или хранение информации (сумматоры в процессоре, ячейки памяти в оперативной памяти и др.), могут быть собраны из базовых логических элементов, как из кирпичиков.

Логические элементы компьютера оперируют с сигналами, представляющими собой электрические импульсы. Есть импульс — логическое значение сигнала 1, нет импульса — значение 0. На вход логического элемента поступают сигналы-аргументы, на выходе появляется сигнал-функция.
Преобразование сигнала логическим элементом задается таблицей состояния, которая фактически является таблицей истинности, соответствующей логической функции.

# Логические операции

- **И** (Конъюнкция - AND ) ∧ , ×, &, И, and
- **ИЛИ** (Дизъюнкция - OR ) V, |, ИЛИ, or, +.
- **НЕ** (Инверсия - NOT ), НЕ, not, ¬ , ¯

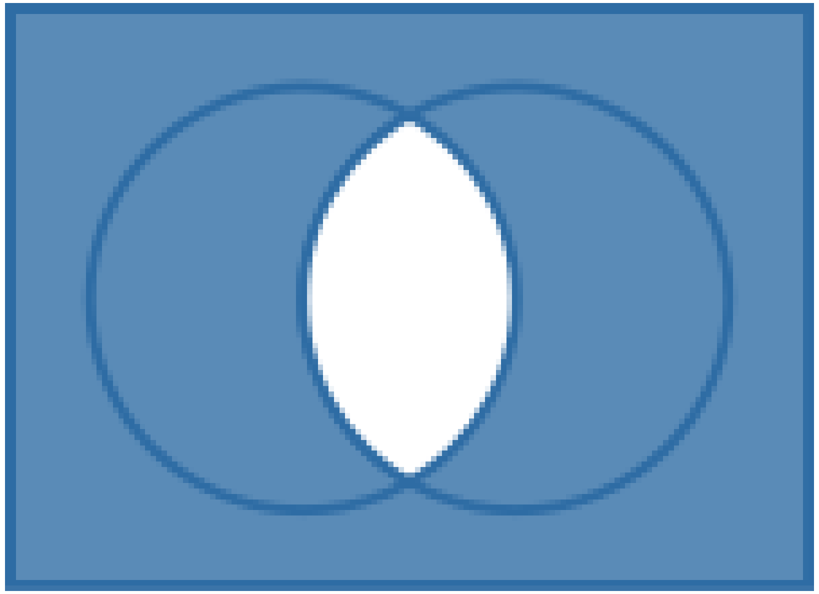
- **И-НЕ** (Штрих Шеффера) |
- **ИЛИ-НЕ** (Стрелка Пирса) ↓
- **Исключающее ИЛИ** (eXclusive OR, XOR) ⊕ , ^ , xor
- **Импликация** («если ..., то ...») →
- **Равнозначность (эквивалентность)** – ~ ; ⇔ ; ↔ ;
- **Исключающее ИЛИ-НЕ** (NOT eXclusive OR) ↔

# «И-НЕ», Штрих Шеффера

- **Инверсия функции конъюнкции. Операция «И-НЕ» (штрих Шеффера)**
- Мнемоническое правило для И-НЕ с любым количеством входов звучит так — на выходе будет:
- **«1»** тогда и только тогда, когда хотя бы на одном входе действует «0»,
- **«0»** тогда и только тогда, когда на всех входах действуют «1».
- **Обозначения: |**

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

| A | B | A \| B |
|---|---|---|
| 0 | 0 | **1** |
| 0 | 1 | **1** |
| 1 | 0 | **1** |
| 1 | 1 | **0** |



# «И-НЕ», Штрих Шеффера

- Базовые операции через «И-НЕ»:

$$\overline{A} = A \mid A$$

$$A \cdot B = \overline{A \mid B} = (A \mid B) \mid (A \mid B)$$

$$A + B = \overline{A} \mid \overline{B} = (A \mid A) \mid (B \mid B)$$

| **A** | **B** | **A \| B** |
|---|---|---|
| 0 | 0 | **1** |
| 0 | 1 | **1** |
| 1 | 0 | **1** |
| 1 | 1 | **0** |

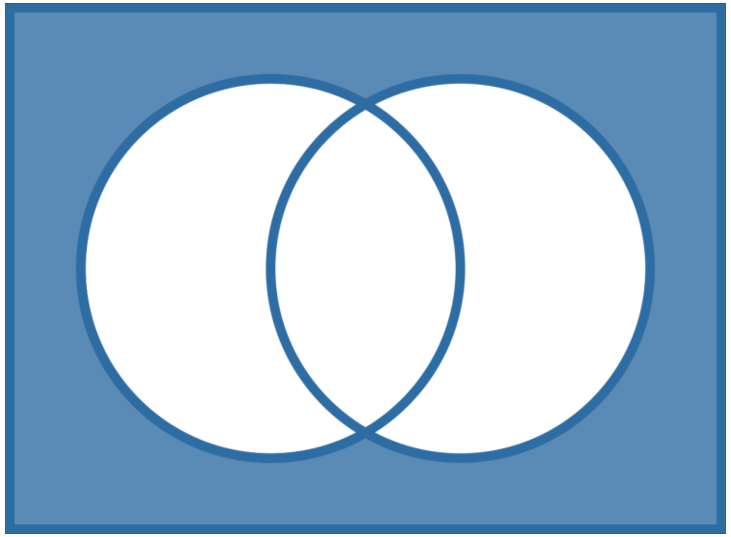
$$A \mid B = \overline{A \cdot B}$$

# «ИЛИ-НЕ» Стрелка Пирса

- **Инверсия функции дизъюнкции. Операция «ИЛИ-НЕ» (стрелка Пирса)**
- Мнемоническое правило для ИЛИ-НЕ с любым количеством входов звучит так — на выходе будет:
- **«1»** тогда и только тогда, когда на всех входах действуют «0»,
- **«0»** тогда и только тогда, когда хотя бы на одном входе действует «1».

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

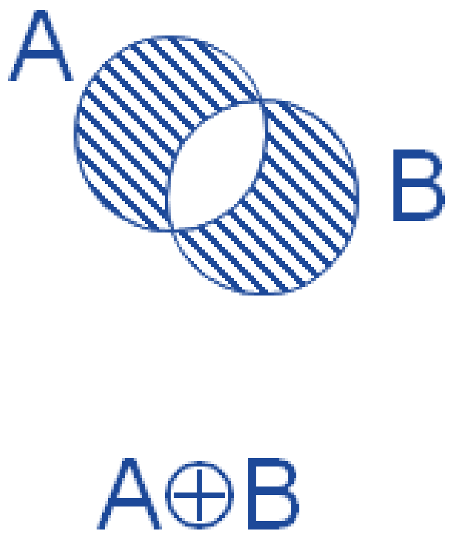
| A | B | $A \downarrow B$ |
|---|---|---|
| 0 | 0 | **1** |
| 0 | 1 | **0** |
| 1 | 0 | **0** |
| 1 | 1 | **0** |



# Исключающее ИЛИ (eXclusive OR, XOR)

- **Сложение (сумма) по модулю 2 (неравнозначность, инверсия равнозначности). Операция «исключающее ИЛИ»**
- Мнемоническое правило для суммы по модулю 2 с любым количеством входов звучит так — на выходе будет:
- **«1»** тогда и только тогда, когда на входе действует нечётное количество «1»,
- **«0»** тогда и только тогда, когда на входе действует чётное количество «1».
- Словесное описание: «истина на выходе — при истине только на входе 1, либо при истине только на входе 2».

| A | B | A ⊕ B |
|---|---|---|
| 0 | 0 | **0** |
| 0 | 1 | **1** |
| 1 | 0 | **1** |
| 1 | 1 | **0** |




**Обозначение: ⊕ , ^ , xor**

# Исключающее ИЛИ (eXclusive OR, XOR)

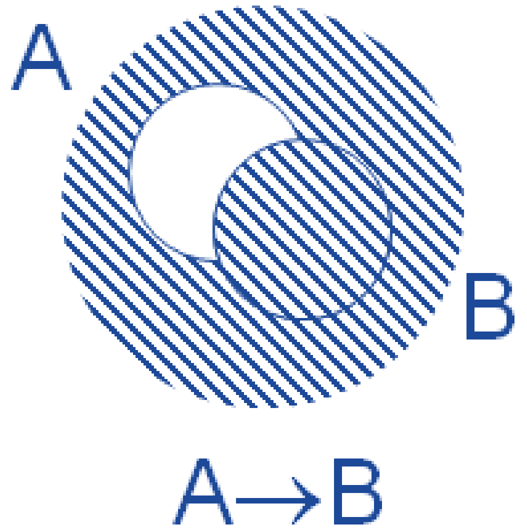
- **Строгая дизъюнкция обозначается символом $\oplus$ .**
- В русском языке строгой (разделительной) дизъюнкции соответствует связка «либо».
- В отличие от обычной дизъюнкции (связка «или») в высказывании, содержащем строгую дизъюнкцию, мы утверждаем, что произойдёт только одно событие.
- **Например,** высказывая утверждение «На сегодняшнем матче Петя сидит на трибуне А либо на трибуне Б», мы считаем, что Петя сидит либо только на трибуне А, либо только на трибуне Б, и что сидеть одновременно на двух трибунах Петя не может.

# Импликация («если ..., то ...»)

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

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



# Логическая операция эквивалентности (равнозначность)

- **Логическая операция эквивалентности (равнозначность)** - логическое равенство образуется соединением двух простых высказываний в одно с помощью оборота речи
- «... тогда и только тогда, когда ...».
- **Обозначение ~** ; ⇔ ; ↔ .
- Составное высказывание, образованное с помощью логической операции эквивалентности, истинно тогда и только тогда, когда оба высказывания одновременно либо ложны, либо истинны.

| A | B | A↔B |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

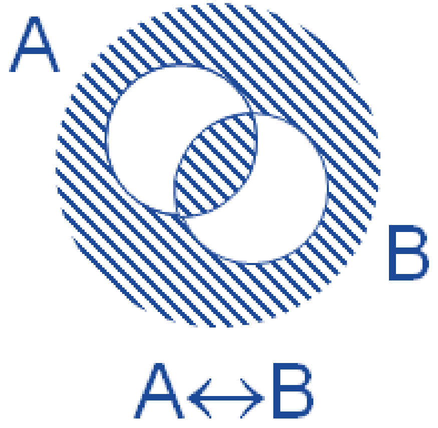
# Исключающее ИЛИ-НЕ (NOT eXclusive OR)

- **Эквивалентность (равнозначность, тождество). Операция «исключающее ИЛИ-НЕ»**
- Мнемоническое правило эквивалентности с любым количеством входов звучит так — на выходе будет:
- **«1»** тогда и только тогда, когда на входе действует чётное количество «1» или «0».
- **«0»** тогда и только тогда, когда на входе действует нечётное количество «1».
- Словесная запись: «истина на выходе при истине на входе 1 и входе 2 или при лжи на входе 1 и входе 2».

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

| A | B | $A \leftrightarrow B$ |
|---|---|---|
| 0 | 0 | **1** |
| 0 | 1 | **0** |
| 1 | 0 | **0** |
| 1 | 1 | **1** |

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



# Исключающее ИЛИ-НЕ (NOT eXclusive OR)

- **В логике эквиваленция обозначается символом ↔.**
- В разговорной речи для выражения взаимной обусловленности используется связка «тогда и только тогда, когда», а в математике — «необходимо и достаточно».

- Рассмотрим высказывание «Денис пойдёт в бассейн тогда и только тогда, когда он выучит уроки».
- Это высказывание истинно (договорённость соблюдается), если истинны оба элементарных высказывания («Денис пойдёт в бассейн», «Денис выучит уроки»).
- Высказывание истинно (договорённость не нарушается) и в том случае, если оба элементарных высказывания ложны («Денис не пойдёт в бассейн», «Денис не выучит уроки»).
- Если же одно из двух высказываний ложно («Денис пойдёт в бассейн, хотя и не выучит уроки», «Денис выучит уроки, но не пойдёт в бассейн»), то договорённость нарушается, и составное высказывание становится ложным.

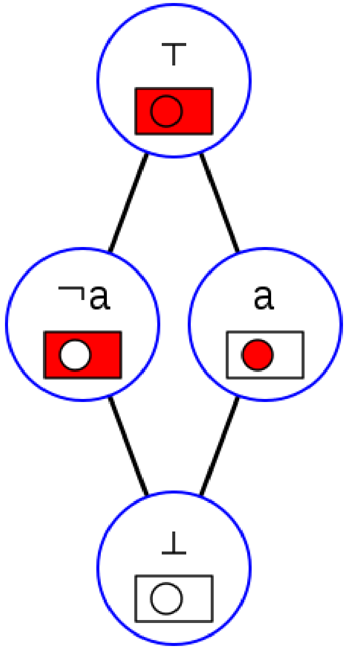
# Логические операции и их обозначения

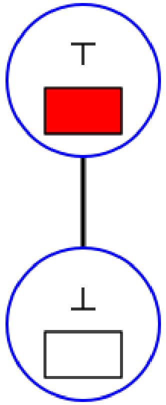
| Операция | Обозначение | Речевой оборот |
|---|---|---|
| **Отрицание** (инверсия, логическое НЕ) | $\neg A,\ \overline{A},\ \text{НЕ}\ A,$ $\text{not}\ A$ | «Не», «неверно, что» |
| **Конъюнкция** (логическое умножение, логическое И) | $A \wedge B,\ A\ \&\ B,$ $A \cdot B,\ AB,$ $A\ \text{И}\ B,\ A\ \text{and}\ B$ | «И», «как ..., так и», «вместе с», «но», «хотя», «а» |
| **Дизъюнкция** (логическое сложение, логическое ИЛИ) | $A \vee B,\ A + B,$ $A \mid B,\ A\ \text{ИЛИ}\ B,$ $A\ \text{or}\ B$ | «Или», «или ..., или ..., или оба вместе» |
| **Строгая дизъюнкция** (исключающая дизъюнкция, исключающее ИЛИ) | $A \oplus B,\ A\ \text{xor}\ B$ | «Либо ..., либо», «только ... или только» |
| **Импликация** (логическое следование) | $A \to B,\ A \Rightarrow B$ | «Если ..., то», «из ... следует», «влечёт» |
| **Эквиваленция** (эквивалентность, равнозначность) | $A \leftrightarrow B,\ A \Leftrightarrow B,$ $A \equiv B$ | «Эквивалентно», «равносильно», «необходимо и достаточно», «тогда и только тогда, когда» |

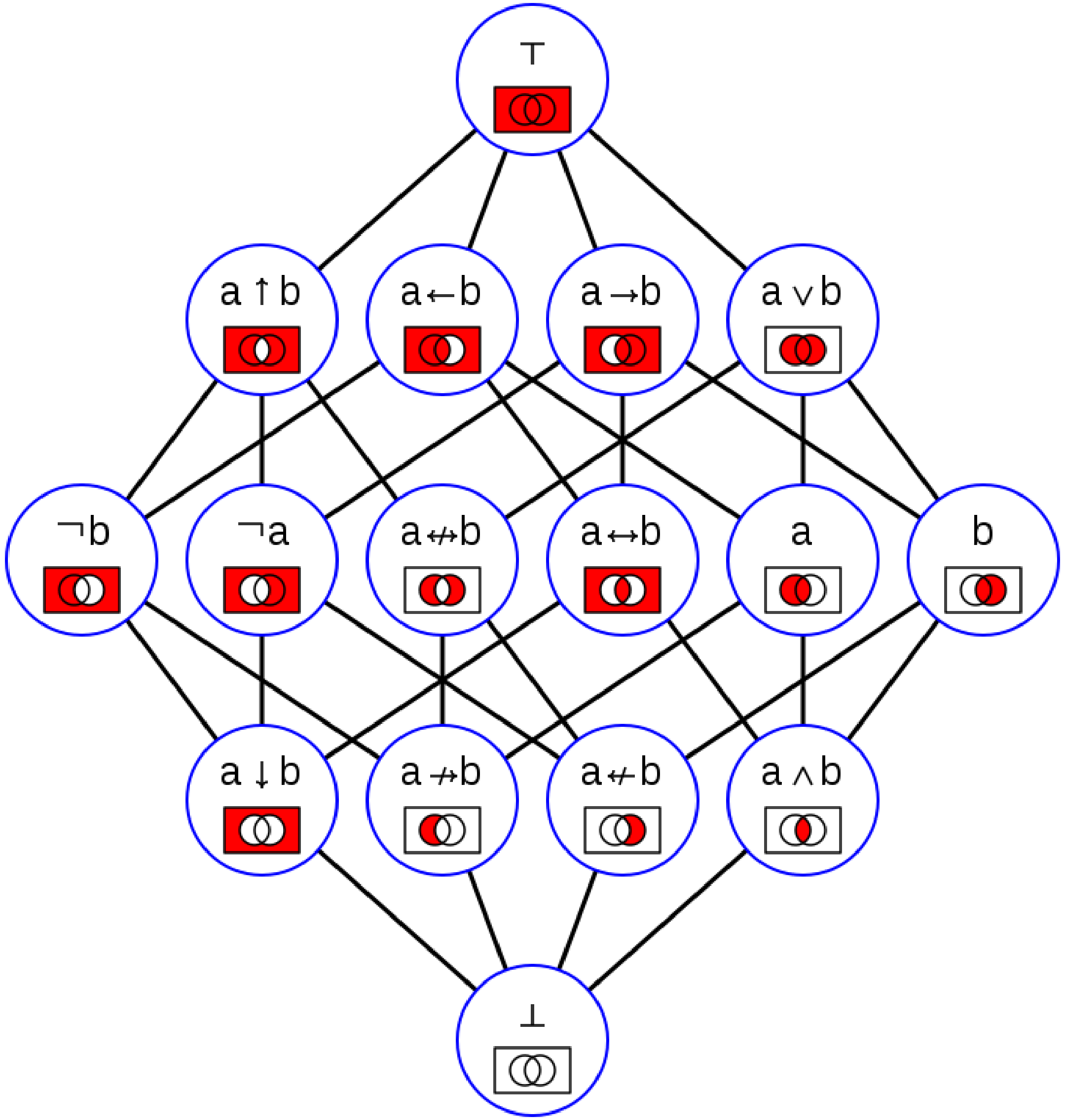
# Логические операции

| Логические переменные | | Логические операции | | | | | |
|---|---|---|---|---|---|---|---|
| | | Отрицание | Конъюнкция | Дизъюнкция | Импликация | Строгая дизъюнкция | Эквиваленция |
| $A$ | $B$ | $\overline{A}$ | $A \& B$ | $A \vee B$ | $A \rightarrow B$ | $A \oplus B$ | $A \leftrightarrow B$ |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 |

# Логические связки




Тавтология ⊤
противоположное соединение ↑
противоположное значение ←
материальный условный →
логическая дизъюнкция ∨
логическое отрицание ¬
исключительная дизъюнкция ↮
биусловный ↔
логическое утверждение
противоположная дизъюнкция ↓
логическое дополнение ↛
противоположное примыкание ↚
логическое соединение ∧
Противоречие ⊥

# Логические выражения

- **Составное логическое высказывание можно представить в виде логического выражения (формулы),** состоящего из логических констант (0, 1), логических переменных, знаков логических операций и скобок.
- **Для логического выражения справедливо:**
  - **1)** всякая логическая переменная, а также логические константы (0, 1) есть логическое выражение;
  - **2)** если A — логическое выражение, то и $\overline{A}$ — логическое выражение;
  - **3)** если A и B — выражения, то, связанные любой бинарной операцией, они также представляют собой логическое выражение.

# Приоритет выполнения логических операций

- При преобразовании или вычислении значения логического выражения **логические операции выполняются в соответствии с их приоритетом:**
  1. Логическое отрицание (инверсия) – «**не**»; ¬ ; ¯ .
  2. Логическое умножение (конъюнкция) – «**и**»; **&**; **^** ; • .
  3. Логическое сложение (дизъюнкция) – «**или**»; **+**; ∨ .
  4. Логическое следование (импликация) – →.
  5. Логическая операция эквивалентности – **~** ; ⇔ ; ↔ .

- Операции одного приоритета выполняются в порядке их следования, слева направо.
- Для изменения указанного порядка могут использоваться скобки.

# Пример №1 Логические выражения

- Выясним, какие из приведённых слов удовлетворяют логическому условию (первая буква согласная → вторая буква согласная) & (последняя буква гласная → предпоследняя буква гласная):

1) ОЗОН; 2) ИГРА; 3) МАФИЯ; 4) ТРЕНАЖ.

- Вычислим значение логического выражения для каждого из данных слов:
  - 1) (0 → 1) & (0 → 1) = 1 & 1 = 1;
  - 2) (0 → 1) & (1 → 0) = 1 & 0 = 0;
  - 3) (1 → 0) & (1 → 1) = 0 & 1 = 0;
  - 4) (1 → 1) & (0 → 1) = 1 & 1 = 1.
- Итак, заданному условию удовлетворяют первое и четвёртое слова.
- Решение логического уравнения — это один или несколько наборов значений логических переменных, при которых логическое уравнение становится истинным выражением.

# Пример №2 Логические выражения

Решим логическое уравнение

$$(A \to C) \vee ((\overline{B \vee C}) \,\&\, A) \vee D = 0.$$

- Дизъюнкция ложна в том и только в том случае, когда ложно каждое из образующих её высказываний. Иными словами, наше уравнение соответствует системе уравнений:

$$\begin{cases} A \to C = 0; \\ (\overline{B \vee C}) \,\&\, A = 0; \\ D = 0. \end{cases}$$

- Таким образом, значение переменной D уже найдено. Импликация равна нулю в единственном случае — когда из истины следует ложь.
- Иначе говоря, в нашем случае: A = 1 и C = 0.

# Пример №2 Логические выражения

- Подставим найденные значения переменных в уравнение $(\overline{B \vee C}) \,\&\, A = 0$.
- Получим: $(\overline{B \vee 0}) \,\&\, 1 = 0$ или $\overline{B} = 0$, т. е. $B = 1$.
- Ответ: A = 1, B = 1, C = 0, D = 0.

- Логические уравнения могут иметь не одно, а несколько и даже очень много решений.
- Зачастую требуется, не выписывая все решения уравнения, указать их количество.

# Пример №3 Логические выражения

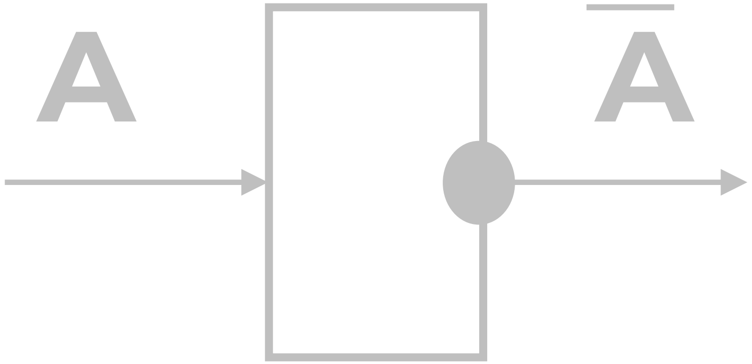
- Выясним, сколько различных решений имеет логическое уравнение

$$(A \;\&\; B \;\&\; \overline{C}) \vee (\overline{B} \;\&\; C \;\&\; D) = 1.$$

- Дизъюнкция истинна, если истинно хотя бы одно из образующих её высказываний. Решение данного логического уравнения равносильно совокупности, состоящей из двух уравнений:

$$\left[\begin{array}{l} A \,\&\, B \,\&\, \overline{C} = 1; \\ \overline{B} \,\&\, C \,\&\, D = 1. \end{array}\right.$$

- Первое равенство будет выполняться только при A = 1, B = 1 и C = 0. Поскольку D в этом уравнении не задействовано, оно может принимать любое из двух значений (0 или 1). Таким образом, всего первое уравнение имеет два решения.




# Таблицы истинности

# Таблицы истинности

- **Построение таблиц истинности**
- Таблицу значений, которые принимает логическое выражение при всех сочетаниях значений (наборах) входящих в него переменных, называют таблицей истинности логического выражения.

- **Таблица истинности** — таблица, описывающая логическую функцию.

# Три основные логические операции

**Конъюнкция**

| *A* | *B* | *A* и *B* |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

**Дизъюнкция**

| *A* | *A* | *A* или *B* |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

**Отрицание**

| *A* | $\overline{A}$ |
|---|---|
| 0 | 1 |
| 1 | 0 |

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

# Таблицы истинности

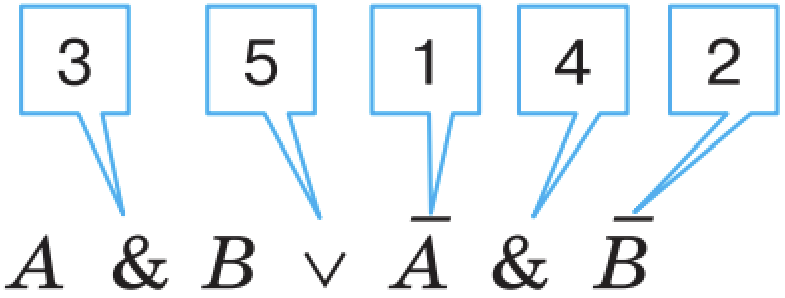
- Для того **чтобы построить таблицу истинности** логического выражения, **достаточно:**
  - **1)** определить число строк таблицы $m = 2^n$, где n — число переменных в логическом выражении;
  - **2)** определить число столбцов таблицы как сумму чисел логических переменных и логических операций в логическом выражении;
  - **3)** установить последовательность выполнения логических операций с учётом скобок и приоритетов операций;
  - **4)** заполнить строку с заголовками столбцов таблицы истинности, занеся в неё имена логических переменных и номера выполняемых логических операций;
  - **5)** выписать наборы входных переменных с учётом того, что они представляют собой ряд целых n-разрядных двоичных чисел от 0 до $2^n – 1$;
  - **6)** провести заполнение таблицы истинности по столбцам, выполняя логические операции.

# Пример №1 Таблицы истинности

- Построим таблицу истинности для логического выражения

$$A \;\&\; B \vee \overline{A} \;\&\; \overline{B}$$

- В этом выражении две логические переменные и пять логических операций. Всего в таблице истинности будет пять строк (22 плюс строка заголовков) и 7 столбцов.
- Начнём заполнять таблицу истинности с учётом следующего порядка выполнения логических операций: сначала выполняются операции отрицания (в порядке следования), затем операции конъюнкции (в порядке следования), последней выполняется дизъюнкция.



# Пример №1 Таблицы истинности

$$\underset{3}{A \& B} \underset{5}{\vee} \underset{1}{\bar{A}} \underset{4}{\&} \underset{2}{\bar{B}}$$

| *A* | *B* | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 0 | 1 |

# Пример № 2

Определить истинность формулы

$$F=((C \vee B) \rightarrow B) \wedge (A \wedge B) \rightarrow B$$

Формула является **тождественно истинной**, если все значения строк результирующего столбца будут равны **1**.

**1 шаг.** Определяем количество **строк** в таблице:

$$m=2^3=8$$

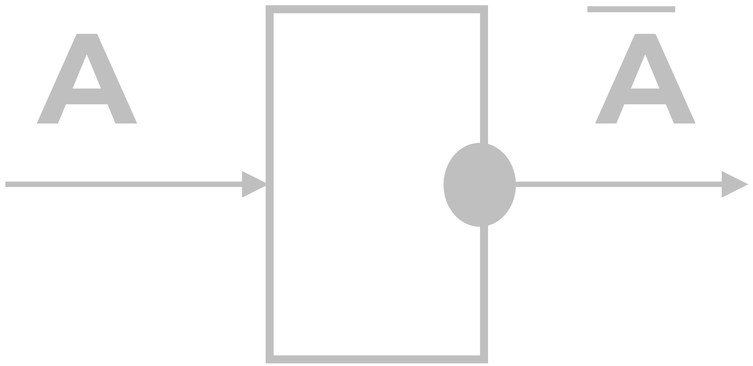
**2 шаг.** Определяем количество **столбцов** в таблице:

$$k=3+5=8$$

# Пример № 2

$$F=((C \vee B) \to B) \wedge (A \wedge B) \to B$$

| 1 | 2 | 3 | 4=3 ∨ 2 | 5=4→2 | 6=1∧2 | 7=5∧6 | 8=7→2 |
|---|---|---|---|---|---|---|---|
| A | B | C | C ∨ B | (C ∨ B) →B | A∧ B | ((C ∨ B) →B) ∧ (A∧B) | F |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |




# Преобразование логических выражений

# Преобразование логических выражений

- **Способ определения истинности логического выражения путём построения его таблицы истинности становится неудобным при увеличении количества логических переменных,** т. к. за счёт существенного увеличения числа строк таблицы становятся громоздкими.
- В таких случаях выполняются преобразования логических выражений в равносильные.
- Для этого используют свойства логических операций, которые иначе называют **законами алгебры логики**.

# Законы алгебры логики

| название | для И | для ИЛИ |
|---|---|---|
| двойного отрицания | $\overline{\overline{A}} = A$ | |
| исключения третьего | $A \cdot \overline{A} = 0$ | $A + \overline{A} = 1$ |
| операции с константами | $A \cdot 0 = 0, \quad A \cdot 1 = A$ | $A + 0 = A, \quad A + 1 = 1$ |
| повторения | $A \cdot A = A$ | $A + A = A$ |
| поглощения | $A \cdot (A + B) = A$ | $A + A \cdot B = A$ |
| переместительный | $A \cdot B = B \cdot A$ | $A + B = B + A$ |
| сочетательный | $A \cdot (B \cdot C) = (A \cdot B) \cdot C$ | $A + (B + C) = (A + B) + C$ |
| распределительный | $A + B \cdot C = (A + B) \cdot (A + C)$ | $A \cdot (B + C) = A \cdot B + A \cdot C$ |
| законы де Моргана | $\overline{A \cdot B} = \overline{A} + \overline{B}$ | $\overline{A + B} = \overline{A} \cdot \overline{B}$ |

# Пример №1

- **Упростим логическое выражение**

$$A \mathbin{\&} B \mathbin{\&} C \lor A \mathbin{\&} B \mathbin{\&} \overline{C}$$

- Последовательно применим дистрибутивный закон и закон исключённого третьего:

$$A \mathbin{\&} B \mathbin{\&} C \lor A \mathbin{\&} B \mathbin{\&} \overline{C} =$$

$$= A \mathbin{\&} B \mathbin{\&} (C \lor \overline{C}) =$$

$$= A \mathbin{\&} B \mathbin{\&} 1 = A \mathbin{\&} B$$

# Пример №2

- Упростим логическое выражение

$$(A \vee B) \,\&\, (A \vee B \vee C) \,\&\, (A \vee B \vee \overline{C}).$$

$$(A \vee B) \,\&\, (A \vee B \vee C) \,\&\, (A \vee B \vee \overline{C}) = (A \vee B) \,\&\, (0 \vee C \vee \overline{C}) =$$
$$= (A \vee B) \,\&\, 1 = A \vee B.$$

# Логические функции

- Значение любого логического выражения определяется значениями входящих в него логических переменных.
- Тем самым логическое выражение может рассматриваться как способ задания логической функции.
- Совокупность значений n аргументов удобно интерпретировать как строку нулей и единиц длины n.
- Существует ровно $2^n$ различных двоичных строк длины n.
- Так как на каждой такой строке некая функция может принимать значение 0 или 1, общее количество различных булевых функций от n аргументов равно $A = 2^{n^2}$.

- Для n = 2 существует 16 различных логических функций

# Логические функции

| $A$ | $B$ | $F_1$ | $F_2$ | $F_3$ | $F_4$ | $F_5$ | $F_6$ | $F_7$ | $F_8$ | $F_9$ | $F_{10}$ | $F_{11}$ | $F_{12}$ | $F_{13}$ | $F_{14}$ | $F_{15}$ | $F_{16}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 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 |

$F_1(A, B) = 0$ — константа «ложь»;
$F_2(A, B) = A \,\&\, B$ — конъюнкция;
$F_3(A, B) = \overline{A \to B}$ — отрицание импликации;
$F_4(A, B) = A$ — функция, равная первому аргументу;
$F_5(A, B) = \overline{B \to A}$ — отрицание обратной импликации;
$F_6(A, B) = B$ — функция, равная второму аргументу;
$F_7(A, B) = A \oplus B$ — строгая дизъюнкция;
$F_8(A, B) = A \vee B$ — дизъюнкция;

# Логические функции

| $A$ | $B$ | $F_1$ | $F_2$ | $F_3$ | $F_4$ | $F_5$ | $F_6$ | $F_7$ | $F_8$ | $F_9$ | $F_{10}$ | $F_{11}$ | $F_{12}$ | $F_{13}$ | $F_{14}$ | $F_{15}$ | $F_{16}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 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 |

$F_9(A, B) = A \downarrow B$ — стрелка Пирса (отрицание дизъюнкции, ИЛИ-НЕ);

$F_{10}(A, B) = A \leftrightarrow B$ — эквиваленция;
$F_{11}(A, B) = \overline{B}$ — отрицание второго аргумента;
$F_{12}(A, B) = B \rightarrow A$ — обратная импликация;
$F_{13}(A, B) = \overline{A}$ — отрицание первого аргумента;
$F_{14}(A, B) = A \rightarrow B$ — импликация;
$F_{15}(A, B) = A \mid B$ — штрих Шеффера (отрицание конъюнкции, И-НЕ);
$F_{16}(A, B) = 1$ — константа «истина».

# Логические функции

- С увеличением числа аргументов количество логических функций резко возрастает.
- Так, для трёх переменных существует 256 различных логических функций!
- Но изучать их все нет никакой необходимости.
- Дело в том, что путём преобразований функция любого количества переменных может быть выражена через функции только двух переменных.
- Более того, можно использовать не все, а лишь некоторые логические функции двух переменных.
- Например:
  - 1) $F_2$ и $F_{11}$ (конъюнкция и отрицание второго аргумента);
  - 2) $F_8$ и $F_{13}$ (дизъюнкция и отрицание первого аргумента);
  - 3) $F_9$ (стрелка Пирса, отрицание дизъюнкции);
  - 4) $F_{15}$ (штрих Шеффера, отрицание конъюнкции).

# Составление логического выражения по таблице истинности и его упрощение

- **Алгоритм составления логического выражения по таблице истинности** достаточно прост.
- Для этого надо:
  - **1)** отметить в таблице истинности наборы переменных, при которых значение логического выражения равно единице;
  - **2)** для каждого отмеченного набора записать конъюнкцию всех переменных следующим образом: если значение некоторой переменной в этом наборе равно 1, то в конъюнкцию включаем саму переменную, в противном случае — её отрицание;
  - **3)** все полученные конъюнкции связать операциями дизъюнкции.

# Пример

| A | B | C | F |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |

- Имеется следующая таблица истинности:
- После выполнения двух первых шагов алгоритма получим:

| A | B | C | F | |
|---|---|---|---|---|
| 0 | 1 | 0 | 1 | $\overline{A} \,\&\, B \,\&\, \overline{C}$ |
| 0 | 1 | 1 | 1 | $\overline{A} \,\&\, B \,\&\, C$ |
| 1 | 1 | 0 | 1 | $A \,\&\, B \,\&\, \overline{C}$ |

# Пример

После выполнения третьего шага получаем логическое выражение:

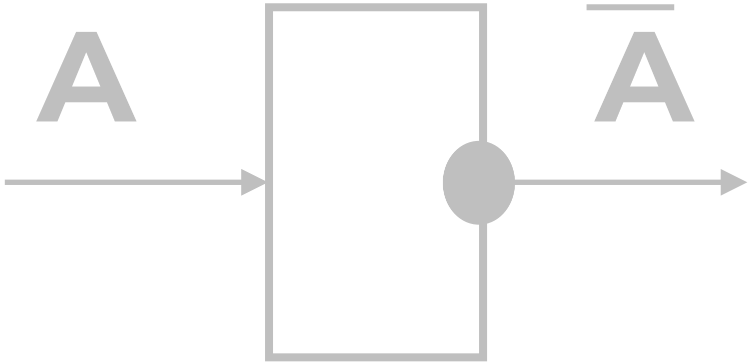
$$\overline{A} \;\&\; B \;\&\; \overline{C} \vee \overline{A} \;\&\; B \;\&\; C \vee A \;\&\; B \;\&\; \overline{C}.$$

Попробуем упростить полученное логическое выражение. Прежде всего, вынесем за скобки $B$ — общий сомножитель, имеющийся у всех трёх слагаемых, затем — сомножитель $\overline{A}$, а далее используем законы алгебры логики.

$$\begin{aligned}
&\overline{A} \;\&\; B \;\&\; \overline{C} \vee \overline{A} \;\&\; B \;\&\; C \vee A \;\&\; B \;\&\; \overline{C} = \\
&= B \;\&\; (\overline{A} \;\&\; \overline{C} \vee \overline{A} \;\&\; C \vee A \;\&\; \overline{C}) = \\
&= B \;\&\; (\overline{A} \;\&\; (\overline{C} \vee C) \vee A \;\&\; \overline{C}) = B \;\&\; (\overline{A} \;\&\; 1 \vee A \;\&\; \overline{C}) = \\
&= B \;\&\; (\overline{A} \vee A \;\&\; \overline{C}) = B \;\&\; (\overline{A} \vee A) \;\&\; (\overline{A} \vee \overline{C}) = \\
&= B \;\&\; 1 \;\&\; (\overline{A} \vee \overline{C}) = B \;\&\; (\overline{A} \vee \overline{C}) = B \;\&\; \overline{A \;\&\; C}.
\end{aligned}$$

# Таблицы истинности

- Способ определения истинности логического выражения путём построения его таблицы истинности становится неудобным при увеличении количества логических переменных, т. к. за счёт существенного увеличения числа строк таблицы становятся громоздкими. В таких случаях выполняются преобразования логических выражений в равносильные. Для этого используют свойства логических операций, которые иначе называют законами алгебры логики. Аналогичные законы имеют место и в алгебре множеств.
- Логическая функция может быть задана с помощью таблицы истинности или аналитически, т. е. с помощью логического выражения.
- Для всякой таблицы истинности можно составить соответствующее ей логическое выражение.




# Элементы схемотехники. Логические схемы

# Элементы схемотехники. Логические схемы

- **Любое устройство компьютера, выполняющее арифметические или логические операции, может рассматриваться как преобразователь двоичной информации**: значения входных переменных для него — последовательность нулей и единиц, а значение выходной функции — новая двоичная последовательность.
- **Необходимые преобразования информации в блоках компьютера производятся логическими устройствами** двух типов: комбинационными схемами и цифровыми автоматами с памятью.

- В комбинационной схеме набор выходных сигналов в любой момент времени полностью определяется набором входных сигналов.
- В цифровых автоматах с памятью набор выходных сигналов зависит не только от набора входных сигналов, но и от внутреннего состояния данного устройства. Такие устройства всегда имеют память.

# Логические элементы

- **Работу любого логического элемента математически удобно описать как логическую функцию**, которая упорядоченному набору из нулей и единиц ставит в соответствие значение, также равное нулю или единице.

- **В схемотехнике широко используются логические элементы.**

- **Логический элемент** — это устройство с $n$ входами и одним выходом, которое преобразует входные двоичные сигналы в двоичный сигнал на выходе.

# Стандарты

ФЕДЕРАЛЬНОЕ АГЕНТСТВО
ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСТ Р МЭК 60617-DB-12M— 2015

Графические символы для схем
(в формате базы данных)

IEC 60617-DB-12M:2012
Graphical symbols for diagrams
(IDT)

Издание официальное

Москва
Стандартинформ
2016

MILITARY STANDARD
GRAPHIC SYMBOLS FOR LOGIC DIAGRAMS

DEUTSCHE NORM August 1997
Graphische Symbole für Schaltpläne
DIN EN 60617-2
IEC 617-2

ГОСТ 2.743–91
МЕЖГОСУДАРСТВЕННЫЙ СТАНДАРТ
ЕДИНАЯ СИСТЕМА КОНСТРУКТОРСКОЙ ДОКУМЕНТАЦИИ
ОБОЗНАЧЕНИЯ УСЛОВНЫЕ ГРАФИЧЕСКИЕ В СХЕМАХ
ЭЛЕМЕНТЫ ЦИФРОВОЙ ТЕХНИКИ
Издание официальное

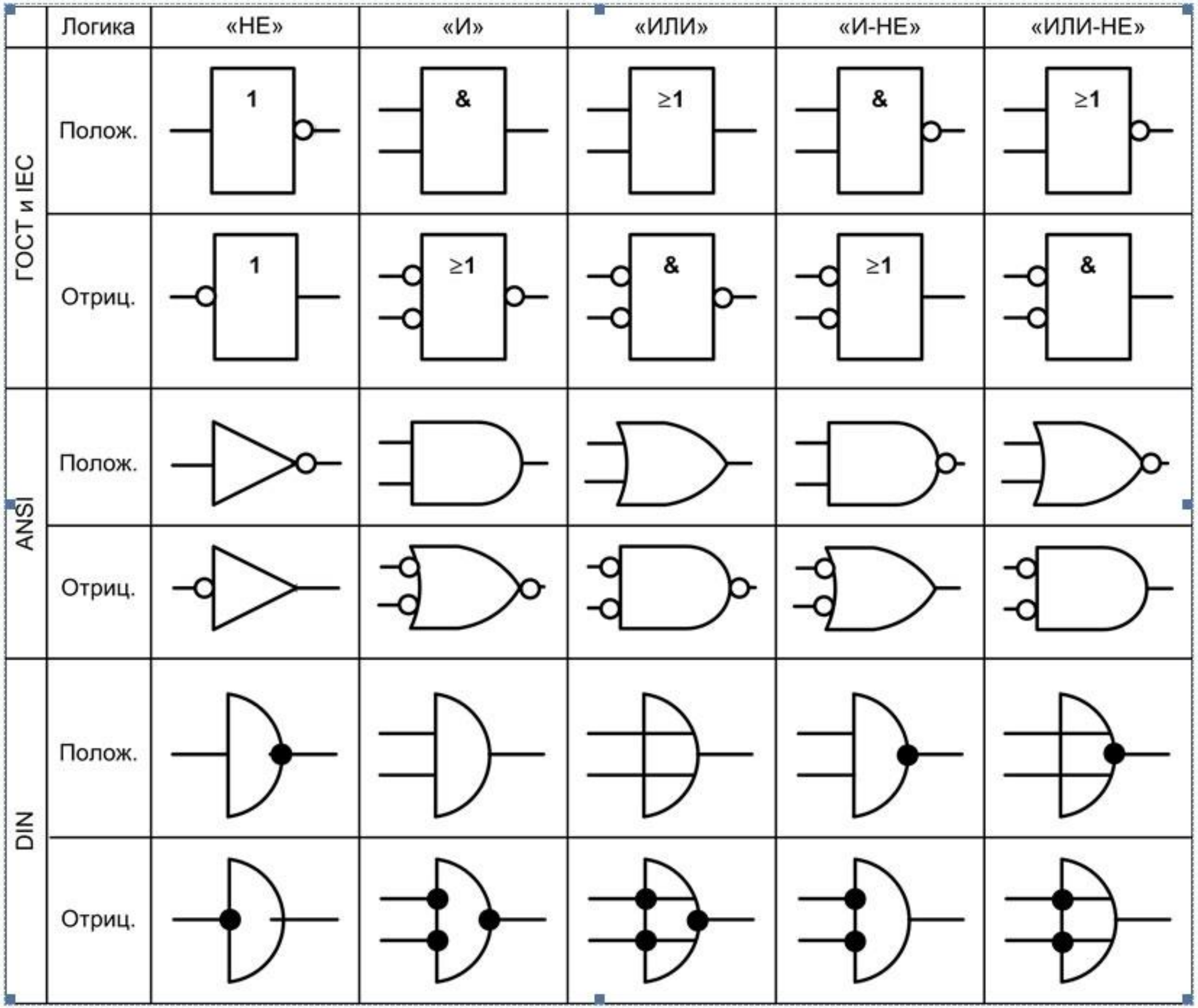
**В технической литературе используются несколько стандартов на условные обозначения элементов:**
- российский (ГОСТ 2.743–91);
- европейский (DIN EN 60617);
- американский (milspec 806B поддерживается в англоязычной и японской литературе).

Кроме этого, в русскоязычной технической литературе до появления ГОСТ активно использовался стандарт МЭК 117-15А, созданный Международной электротехнической комиссией (International Electrotechnical Comission, IEC) в которую СССР, а затем и Россия входят с 1922 г.

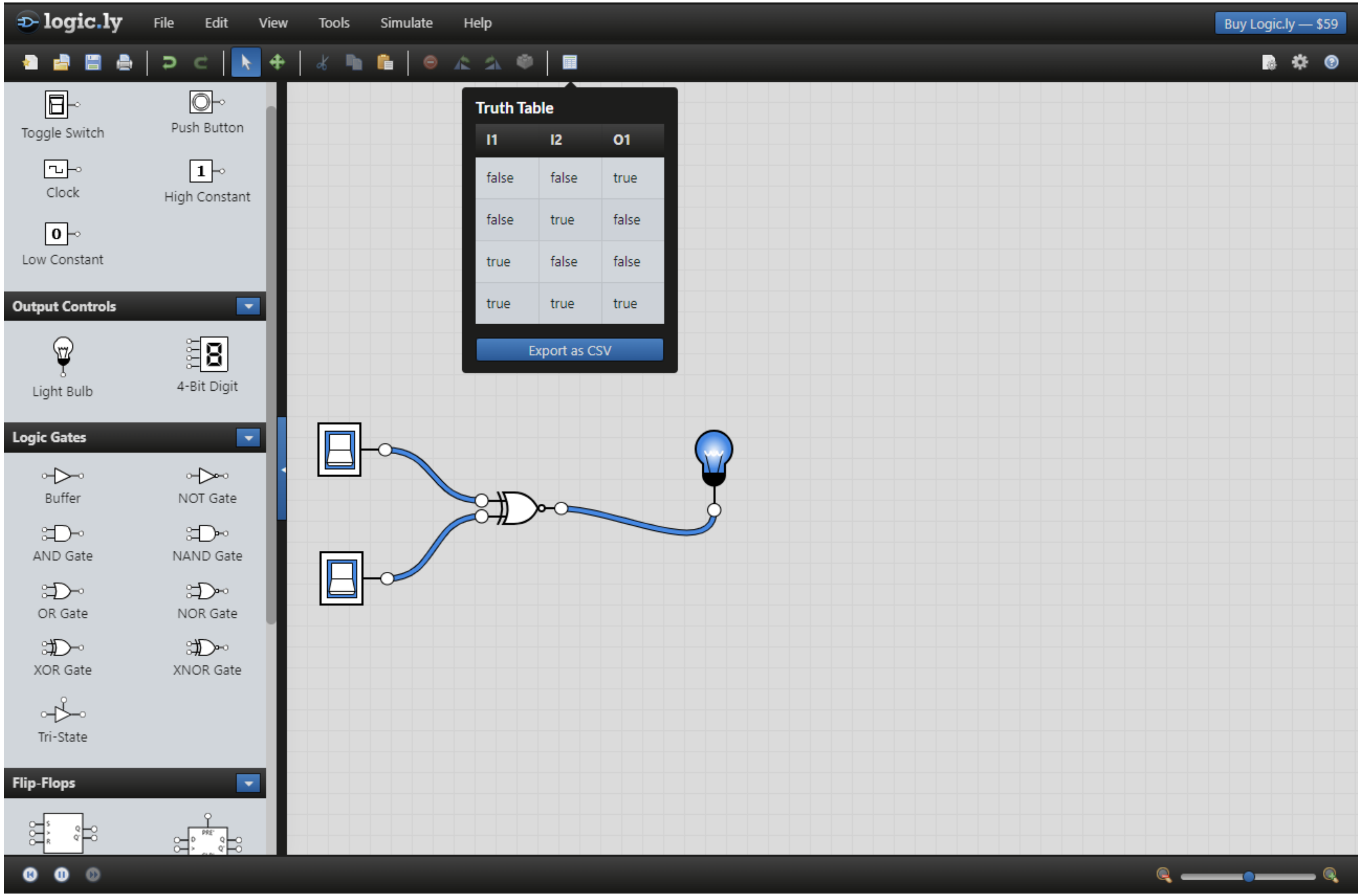
**В настоящее время действующим стандартом МЭК является стандарт IEC 60617.**

MI L-ST D-806B
https://bitsavers.org/pdf/mil-std/MIL-STD-806B_Graphical_Symbols_For_Logic_Diagrams_19620226.pdf
DIN EN 60617
http://moodle.jrobi.hu/BOCI/moodle/Technikus/DINEN60617(1999).pdf
ГОСТ 2.743–91
https://meganorm.ru/Data2/1/4294849/4294849507.pdf

**ГОСТ Р МЭК 60617-DB-12M -2015** (2315 стр.)
http://www.omegametall.ru/Data2/1/4293763/4293763109.pdf


Логика
«НЕ»
«И»
«ИЛИ»
«И-НЕ»
«ИЛИ-НЕ»
ГОСТ и IEC
Полож.
1
&
≥1
&
≥1
Отриц.
1
≥1
&
≥1
&
ANSI
Полож.
Отриц.
DIN
Полож.
Отриц.

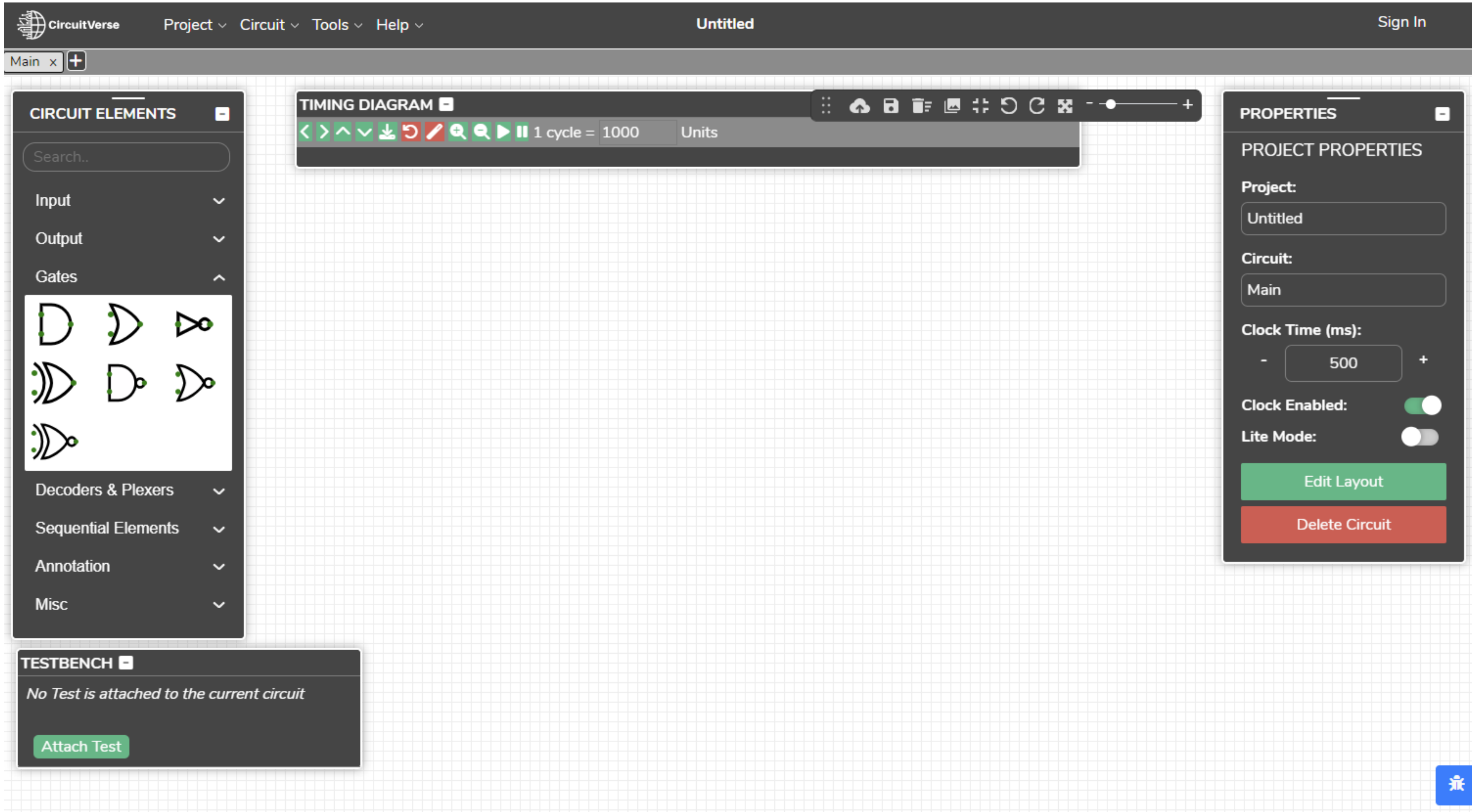
# https://logic.ly/demo/




**Сервис logic.ly** позволяет строить логические схемы, используя входные (переключатели) и выходные сигналы (лампочку), а также логические элементы (NOT, OR, AND, XOR). Пользователь может построить схему любой сложности.

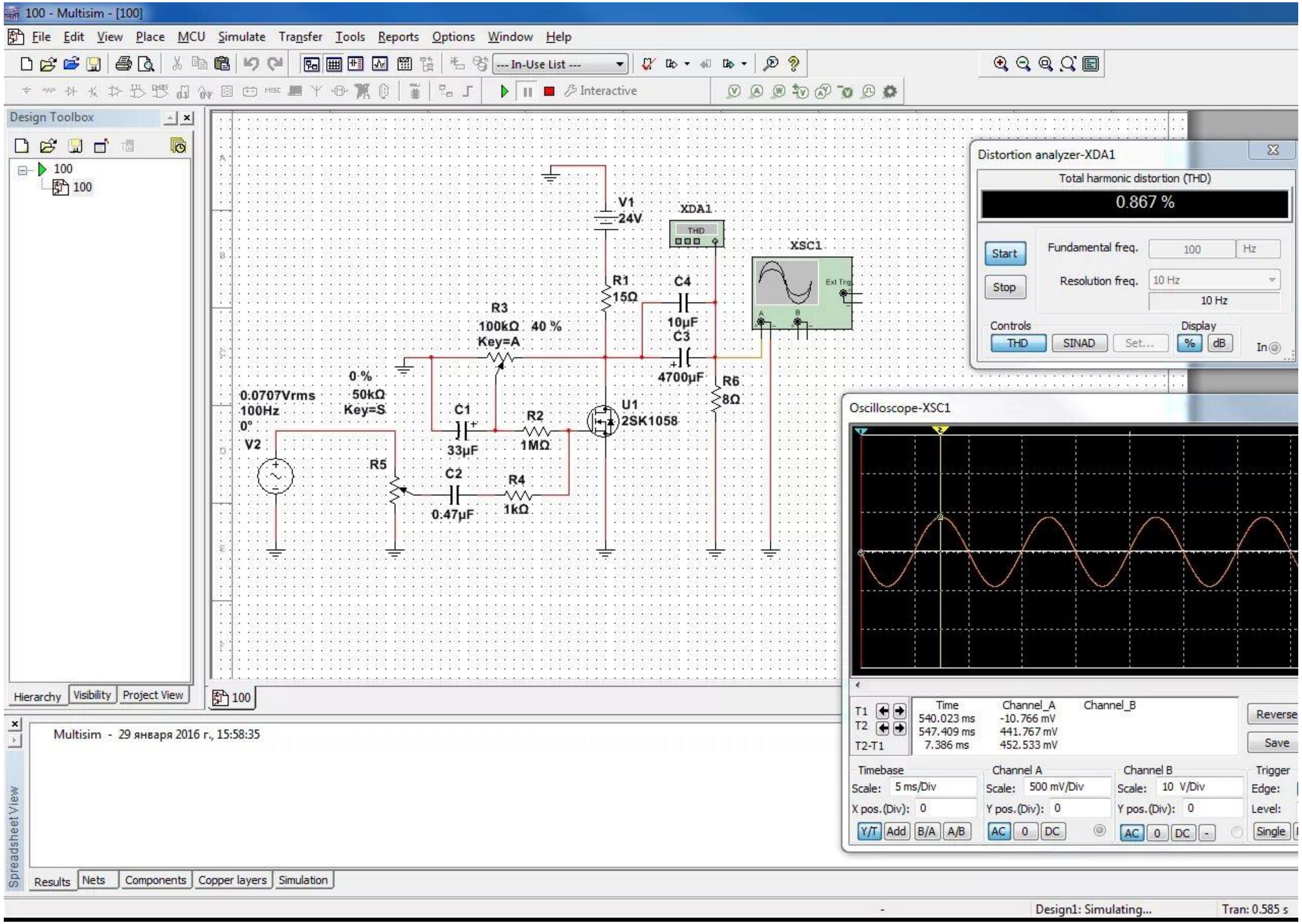
Логические схемы в Logic.ly позволяют:
- научиться использовать логические элементы, входные и выходные сигналы;
- получить навык построения логических схем;
- переключать входные данные в режиме реального времени;
- выполнять проверку заполненных таблиц истинности;
- понимать как работают логические схемы.

# https://circuitverse.org/simulator

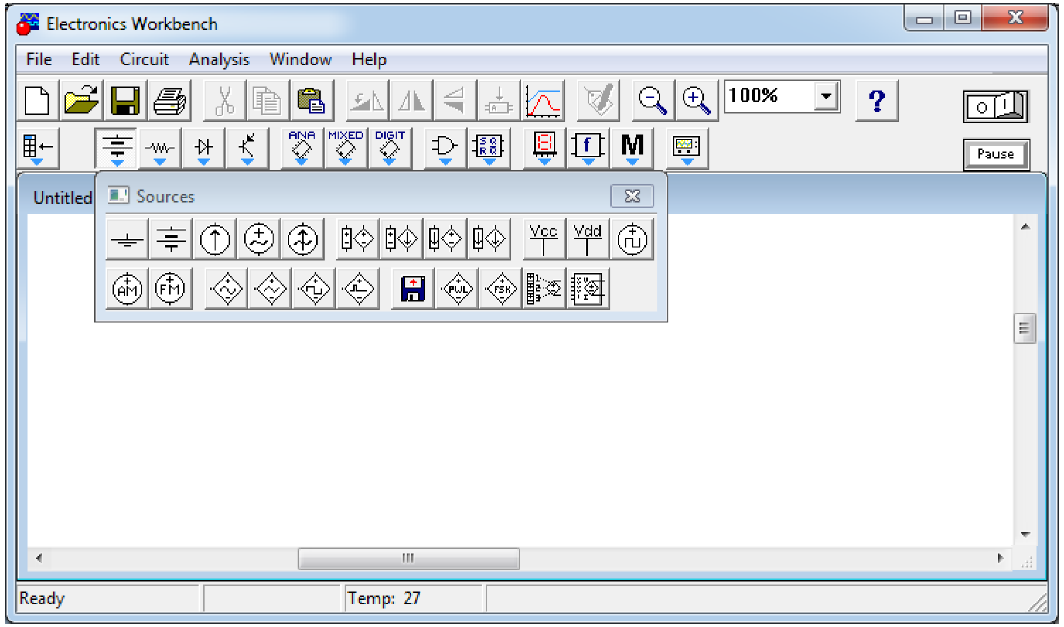



Симулятор **CircuitVerse** содержит большинство элементов первичной цепи как от комбинационной, так и от последовательной схемы. CircuitVerse позволяет использовать многобитовые провода (шины) и подсхемы. CircuitVerse очень похож на Logisim, который в настоящее время является самым популярным цифровым симулятором в академических кругах.

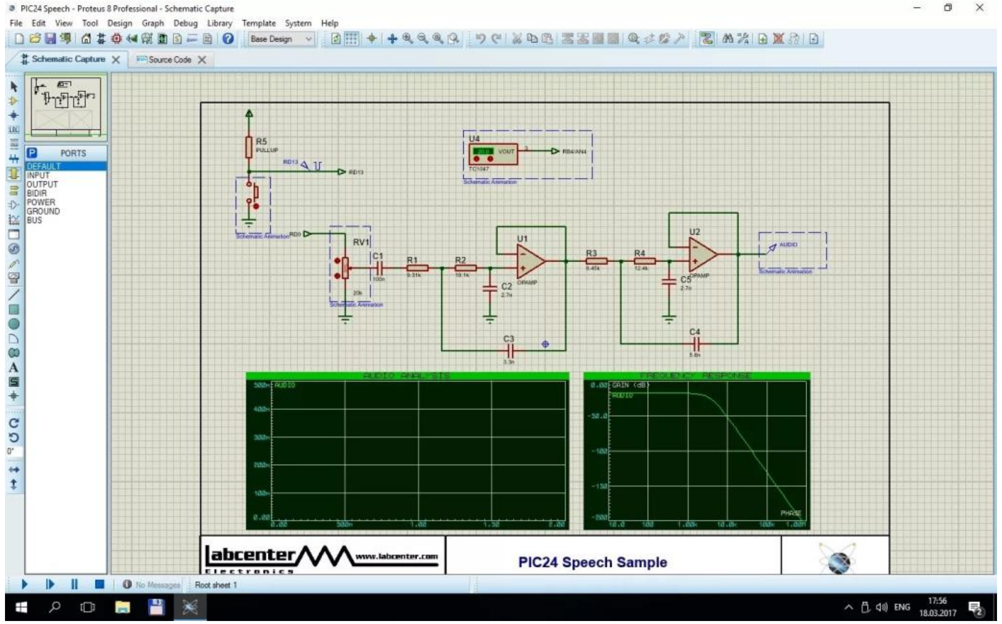
# Multisim



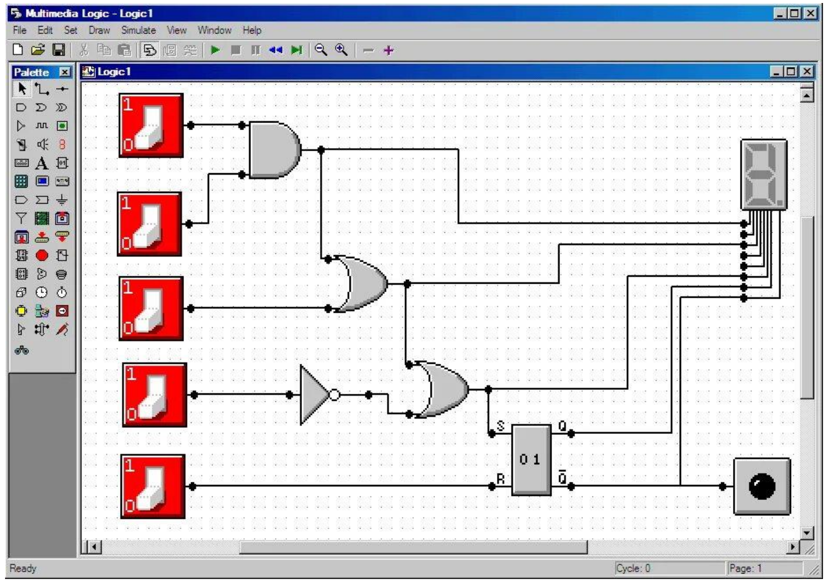

**Multisim** – приложение для создания и тестирования электрических цепей в схемотехнике.
Оно включает сотни виртуальных электронных компонентов и измерительных устройств, умеет моделировать их работу и взаимодействие, что необходимо для проектирования, анализа и отладки электрических схем. Программа поддерживает имитацию различных режимов работы цепей и применяется на всех этапах их разработки.



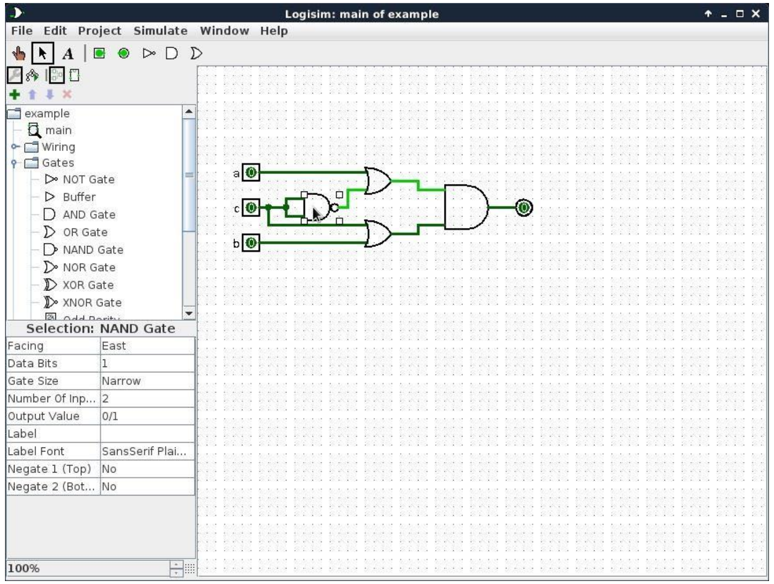

Electronics Workbench




Proteus Design Suite




Multimedia Logic (MMLogic)




Logisim

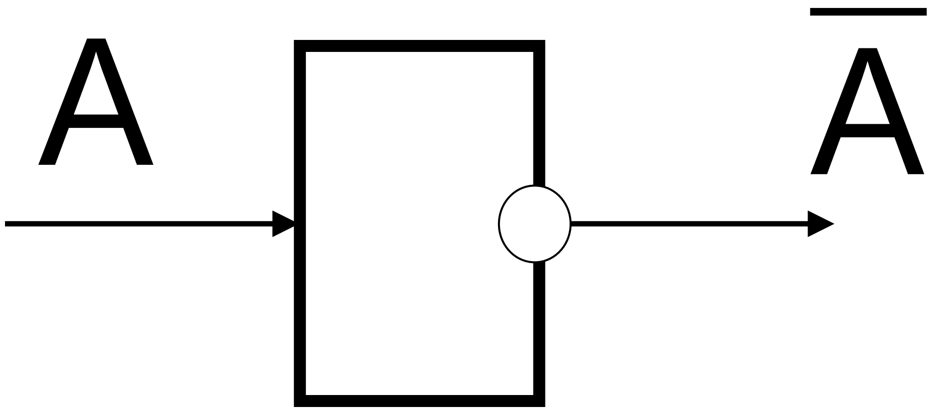
# Логическая схема «НЕ» - Инверсия - NOT

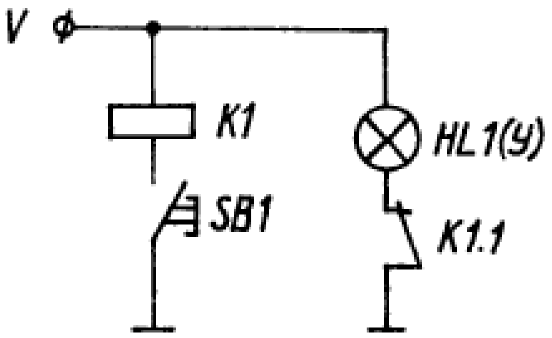
- **Логическая схема «НЕ»** называется также инвертором, выполняет логическую операцию отрицания (инверсии).
- **Обозначения: НЕ, ¬ , ¯ .**

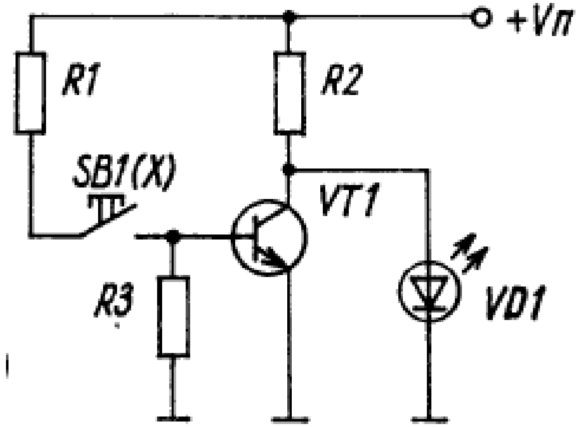
**Таблица истинности**

| A | не A (not A) |
|---|---|
| 0 | **1** |
| 1 | **0** |

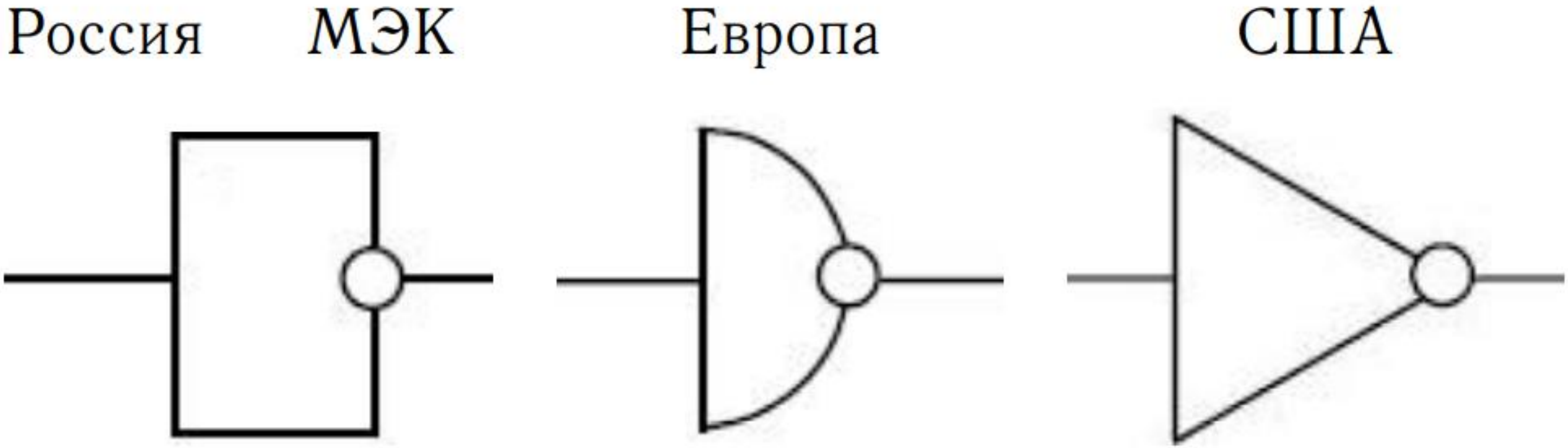
**Графическое обозначение**



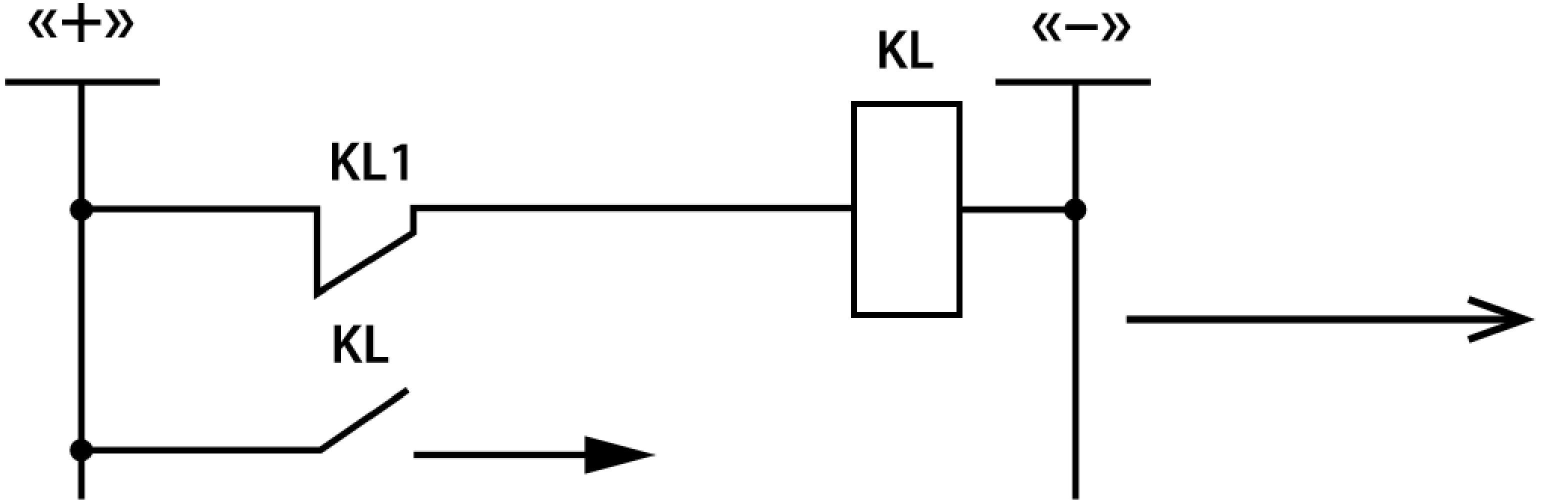



# Обозначение условное графическое логического элемента **НЕ** (NOT)

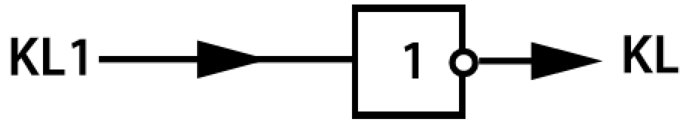


# Логическая схема «НЕ» - Инверсия - NOT

Схема «НЕ», выполненная с помощью реле

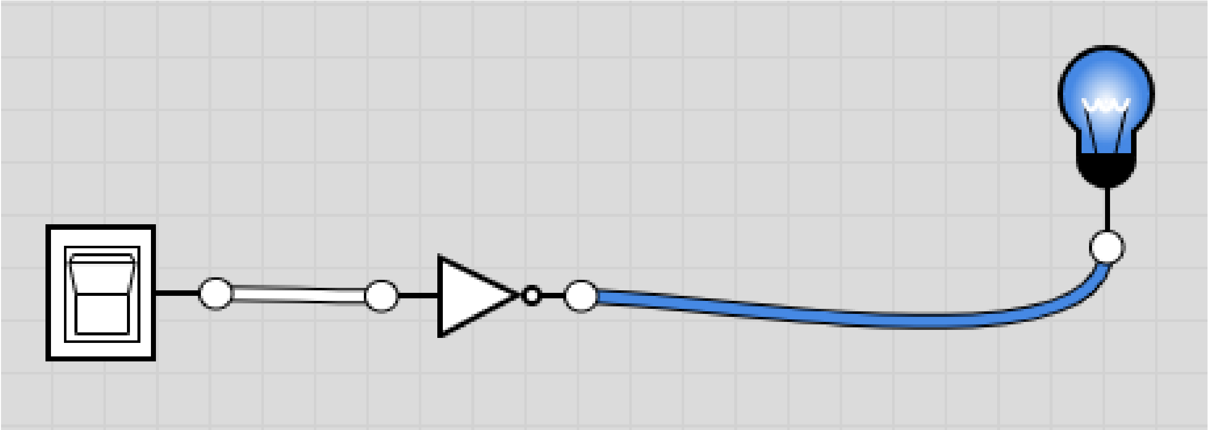


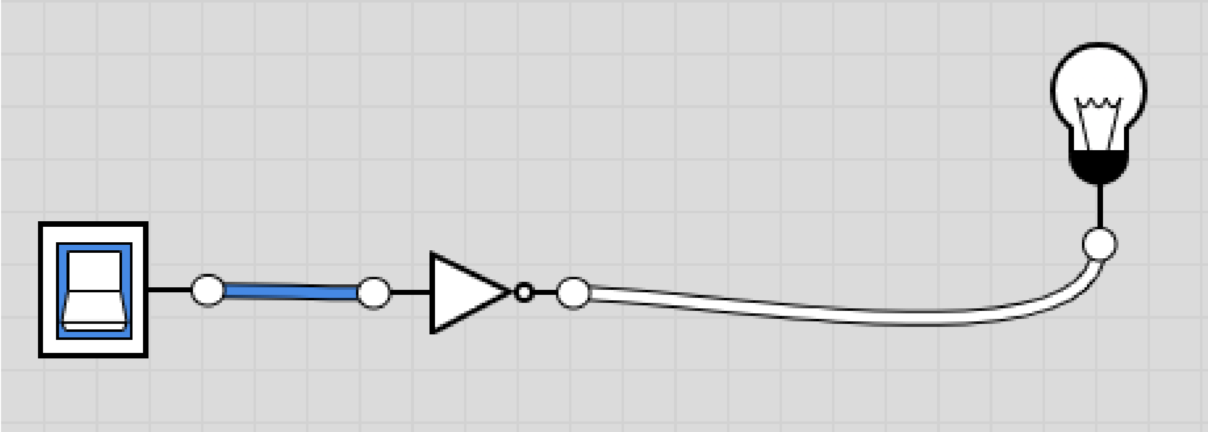

Логический элемент «НЕ»



# Пример в logic.ly

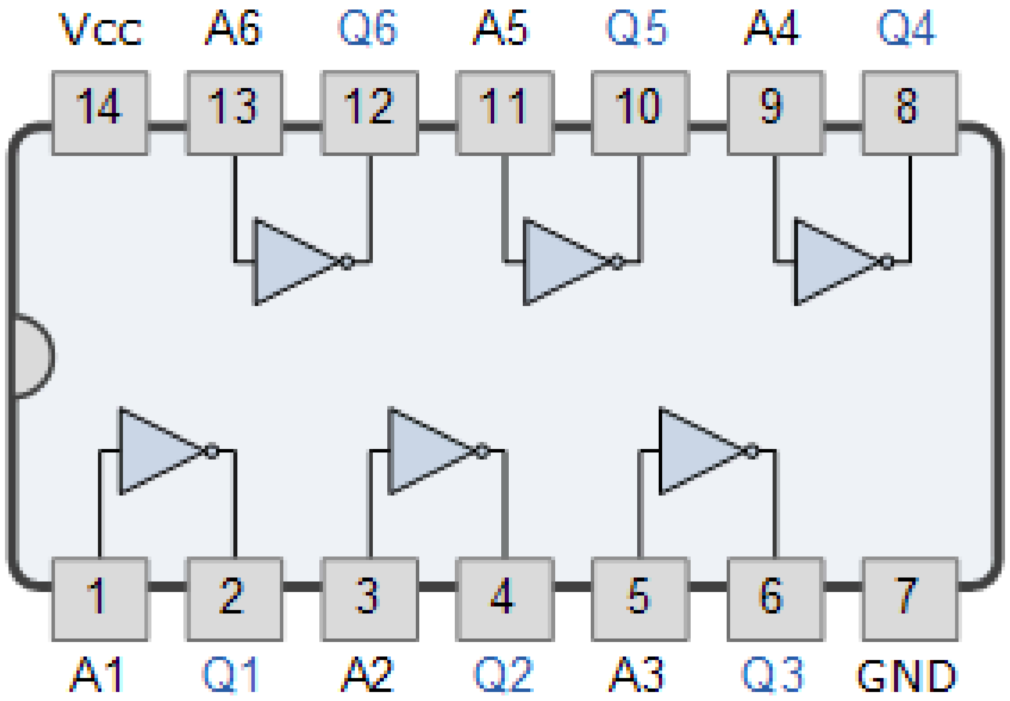
Логическая схема «**НЕ**» - Инверсия - NOT



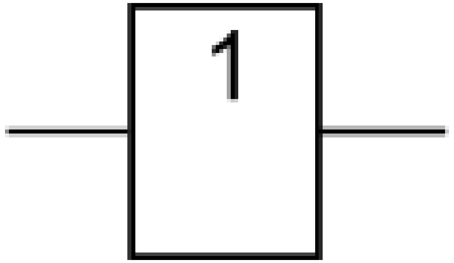
# Пример микросхемы с логическим элементом «НЕ»

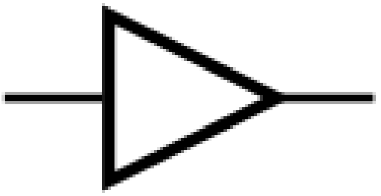
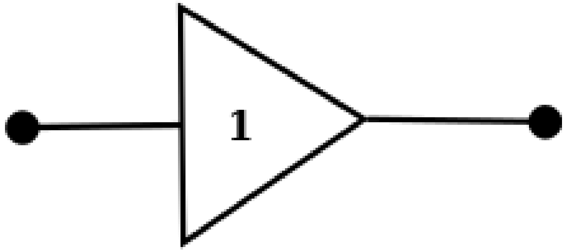
## IC 7404 (NOT gate Inverter)




# Повторитель (буфер)

- Повторение - Выходная логическая переменная равна входной.




| Ввод | Вывод |
|---|---|
| 0 | 0 |
| 1 | 1 |

**Цифровой буфер (или буфер напряжения)** - элемент электронной схемы, используемый для изоляции входа от выхода. Состояние выхода буфера отражает состояние входа. Входное сопротивление буфера высокое. Он потребляет небольшой ток, чтобы избежать помех во входной цепи. Также называемый буфером с единичным коэффициентом усиления, цифровой буфер намеренно не усиливает и не ослабляет входной сигнал.

Цифровой буфер важен при передаче данных, транслируя импульсы напряжения между подключенными системами. **Буферы используются в регистрах (устройстве хранения данных) и шинах (устройстве передачи данных).** Цифровой буфер с тремя состояниями может подключать устройство к цифровой шине. Выходной сигнал буфера с тремя состояниями либо высокий, либо низкий, либо отключенный.

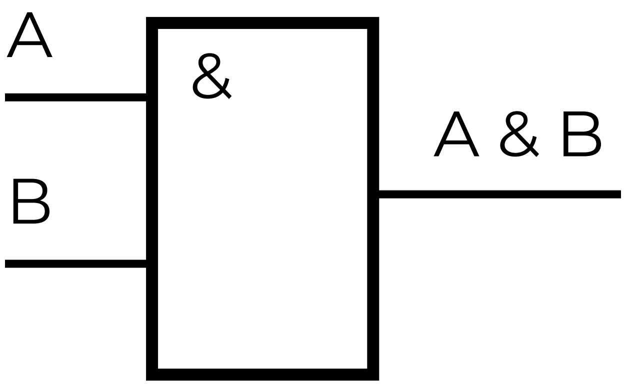
# Логическая схема «И» - Конъюкция - AND

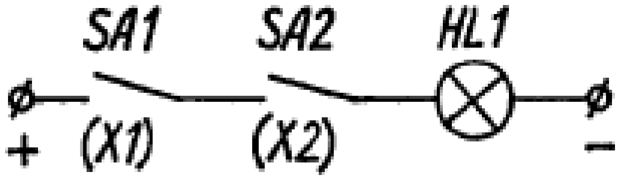
- **Логическая схема «И»** называется также конъюнктором, выполняет операцию логического умножения (конъюнкции), может иметь от двух до восьми входов.
- **Обозначения: ∧ , ×, &, И.**

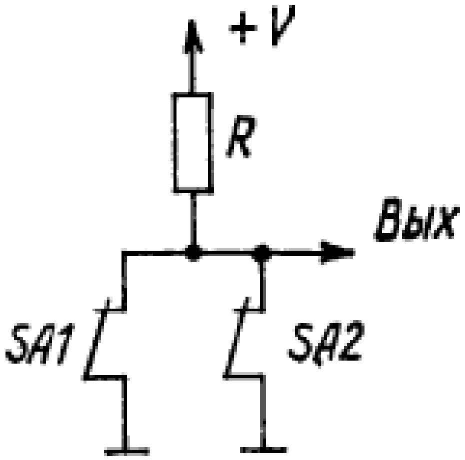
**Таблица истинности**

| A | B | F = A & B |
|---|---|---|
| 0 | 0 | **0** |
| 0 | 1 | **0** |
| 1 | 0 | **0** |
| 1 | 1 | **1** |

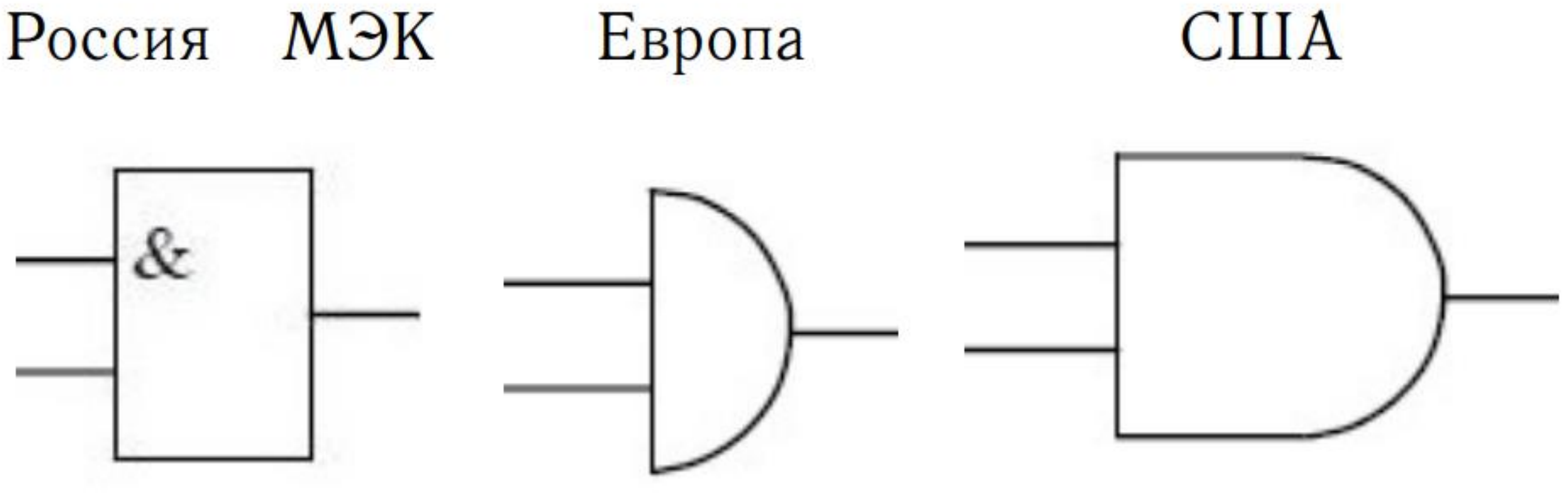
**Графическое обозначение**
двухвходовой схемы «И»



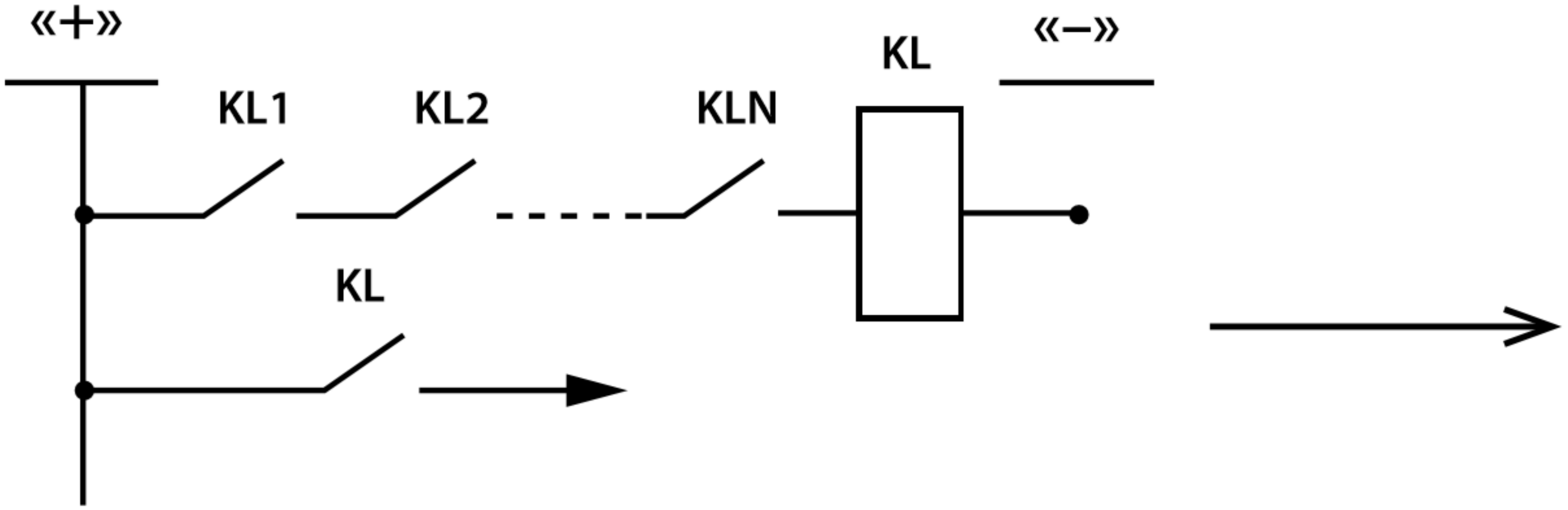



# Обозначение условное графическое логического элемента И (AND)

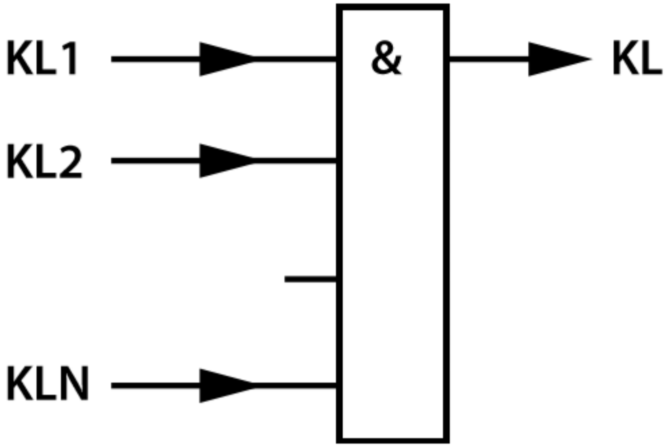


# Логическая схема «И» - Конъюкция - AND

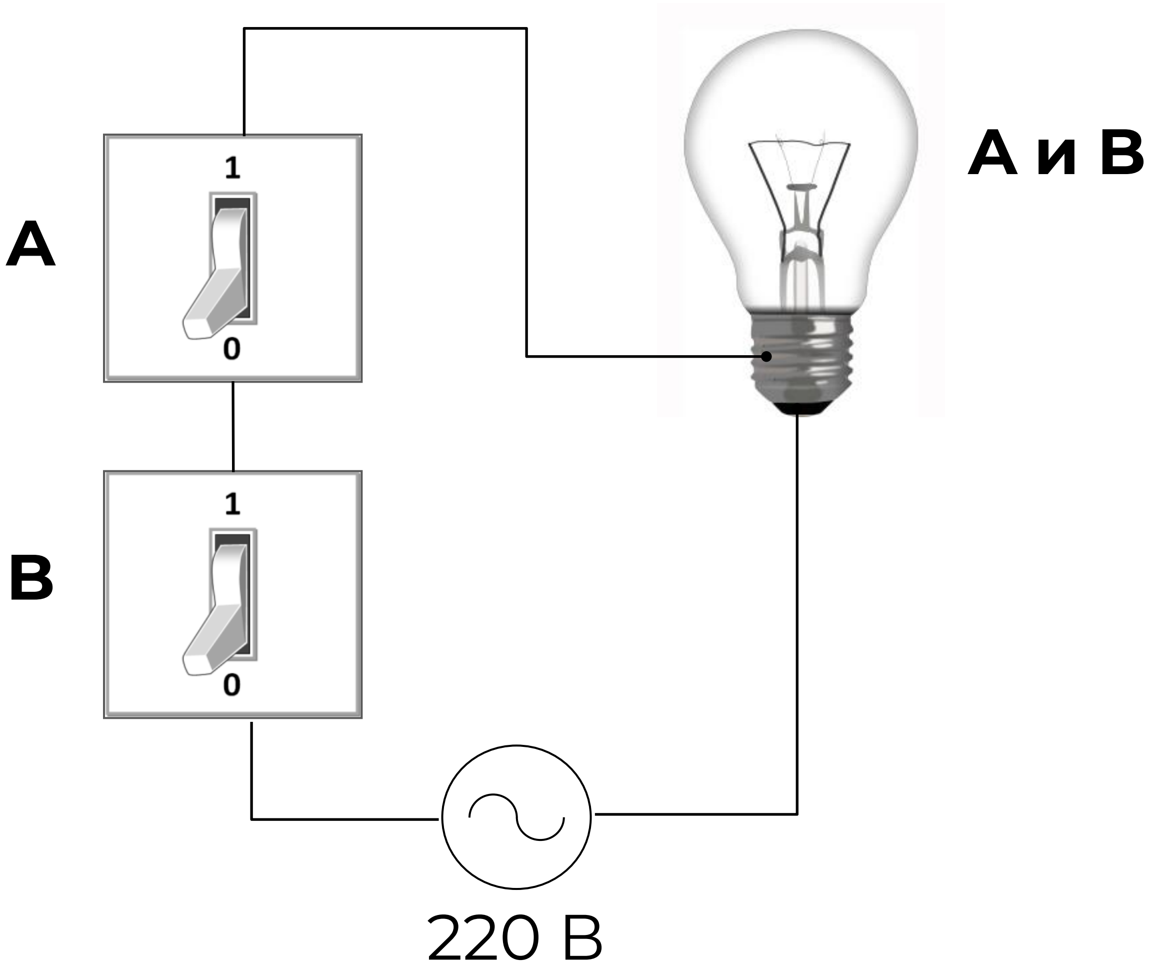
Схема «И», выполненная с помощью реле




Логический элемент «И»

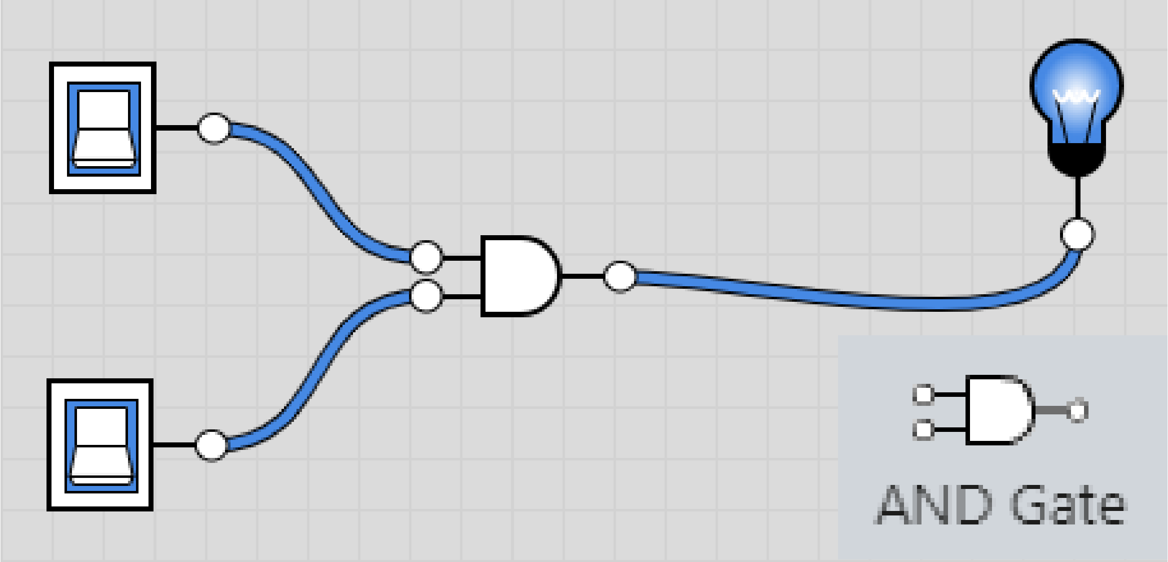


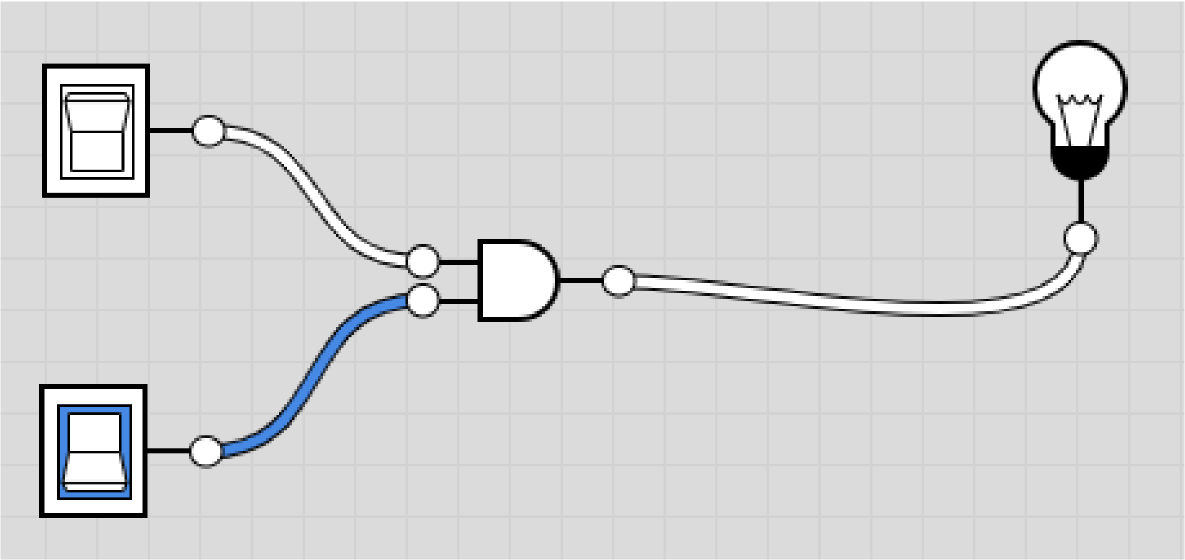
# Логическая схема «И» - Конъюкция - AND

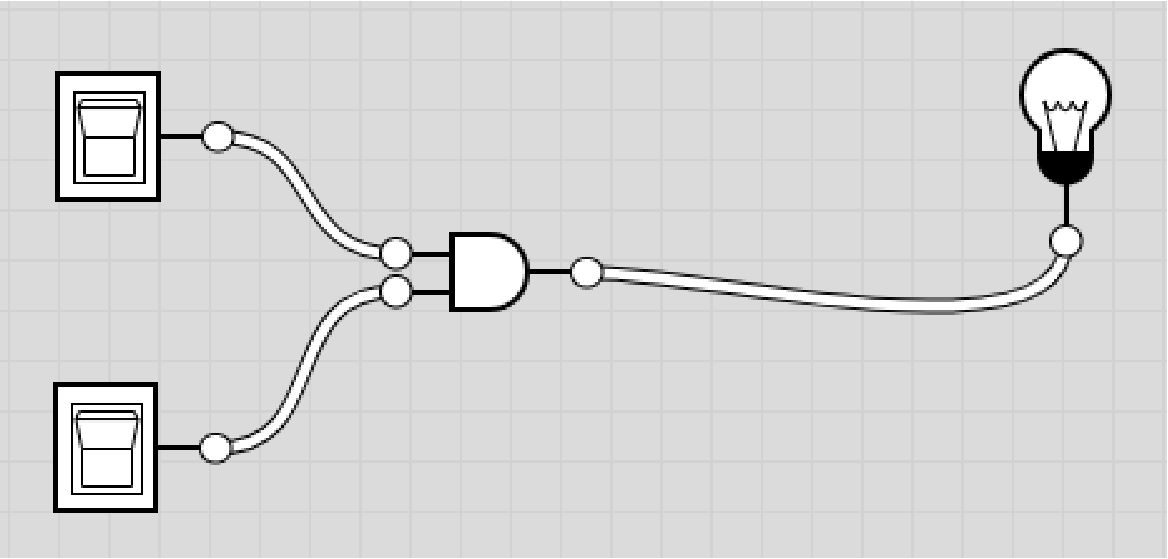


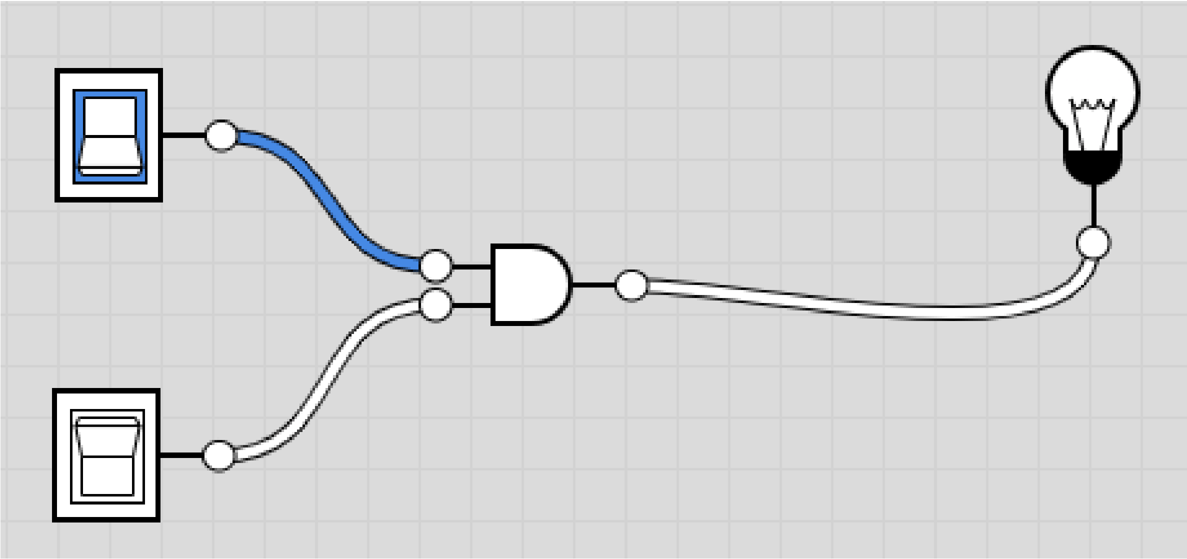

| A | B | A и B |
|---|---|---|
| 0 | 0 | **0** |
| 0 | 1 | **0** |
| 1 | 0 | **0** |
| 1 | 1 | **1** |

# Пример в logic.ly





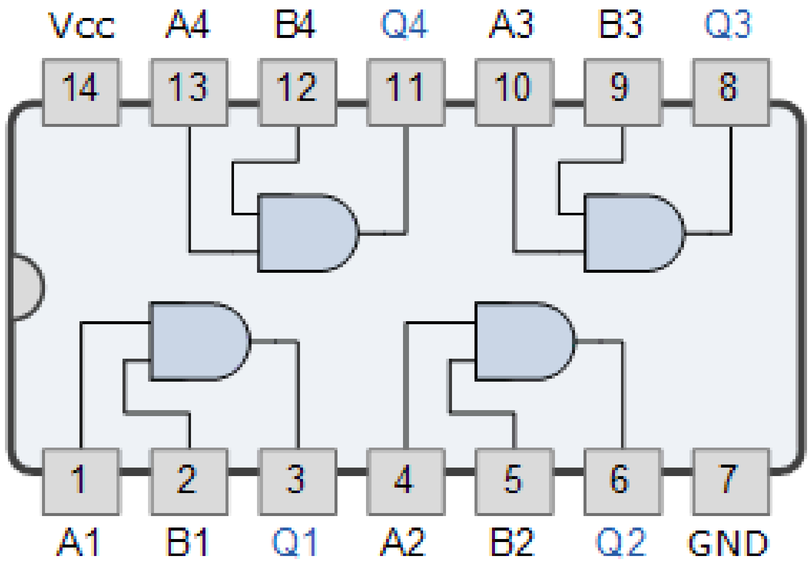


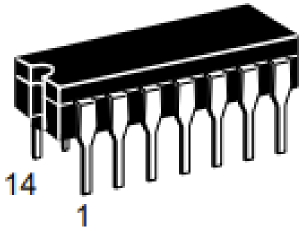
# Пример микросхемы с логическим элементом «И»

MOTOROLA

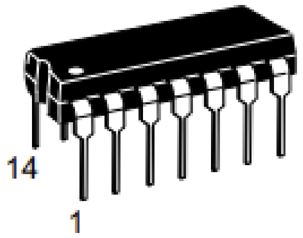
## QUAD 2-INPUT AND GATE

## SN54/74LS08

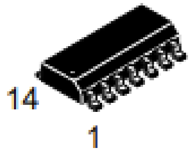



**J SUFFIX**
CERAMIC
CASE 632-08




**N SUFFIX**
PLASTIC
CASE 646-06




**D SUFFIX**
SOIC
CASE 751A-02

**ORDERING INFORMATION**

| | |
|---|---|
| SN54LSXXJ | Ceramic |
| SN74LSXXN | Plastic |
| SN74LSXXD | SOIC |

https://albertno-youtube.github.io/datasheets/7408_AND.pdf

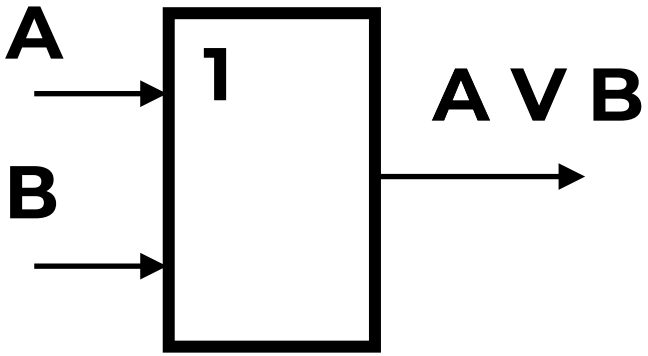
# Логическая схема «ИЛИ» - Дизъюнкция - OR

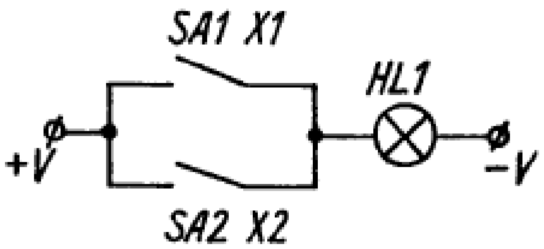
- **Логическая схема «ИЛИ»** называется также дизъюнктором, выполняет операцию логического сложения (дизъюнкции), может иметь от двух до восьми входов.
- **Обозначения: V, |, ИЛИ, +.**

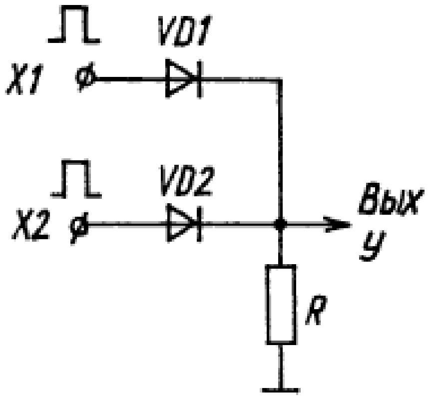
**Таблица истинности**

| A | B | F = A V B |
|---|---|---|
| 0 | 0 | **0** |
| 0 | 1 | **1** |
| 1 | 0 | **1** |
| 1 | 1 | **1** |

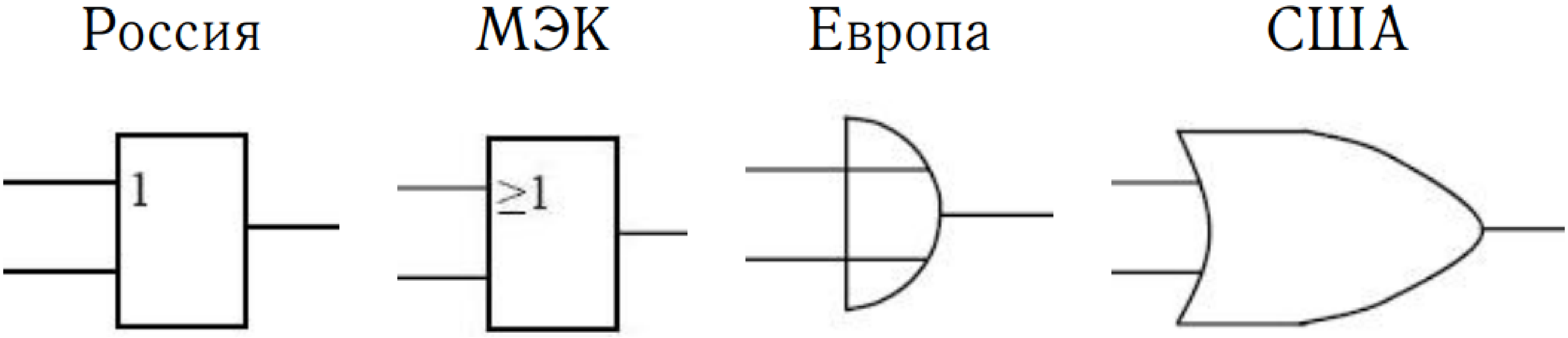
**Графическое обозначение** двухвходовой схемы «ИЛИ»



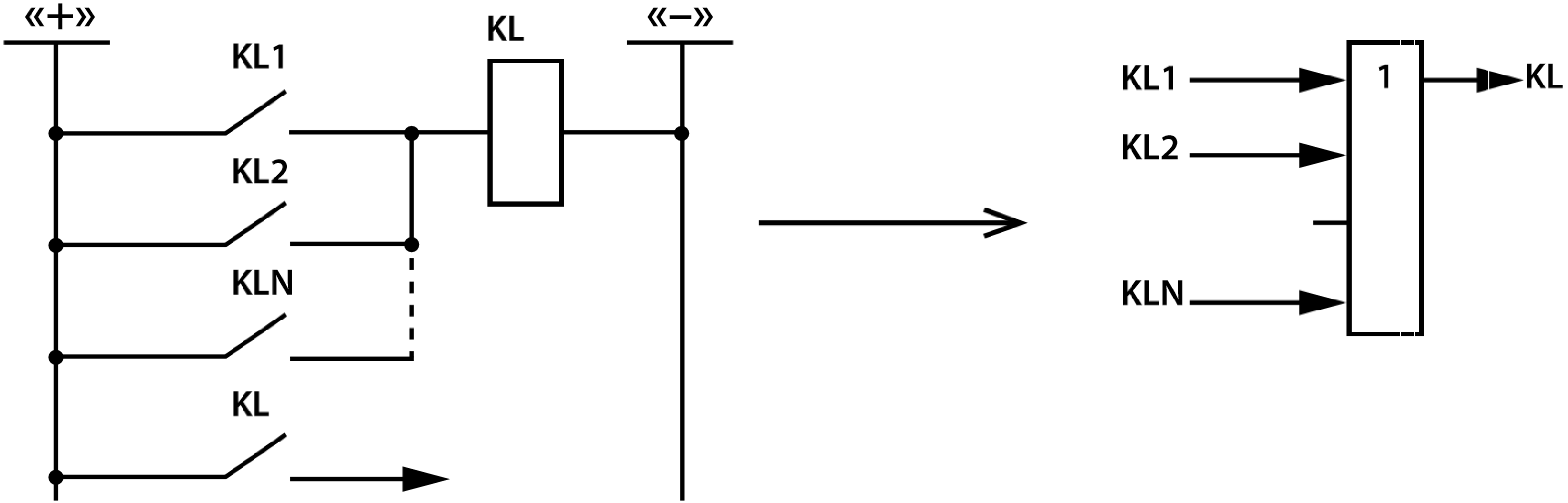
# Обозначение условное графическое логического элемента ИЛИ (OR)



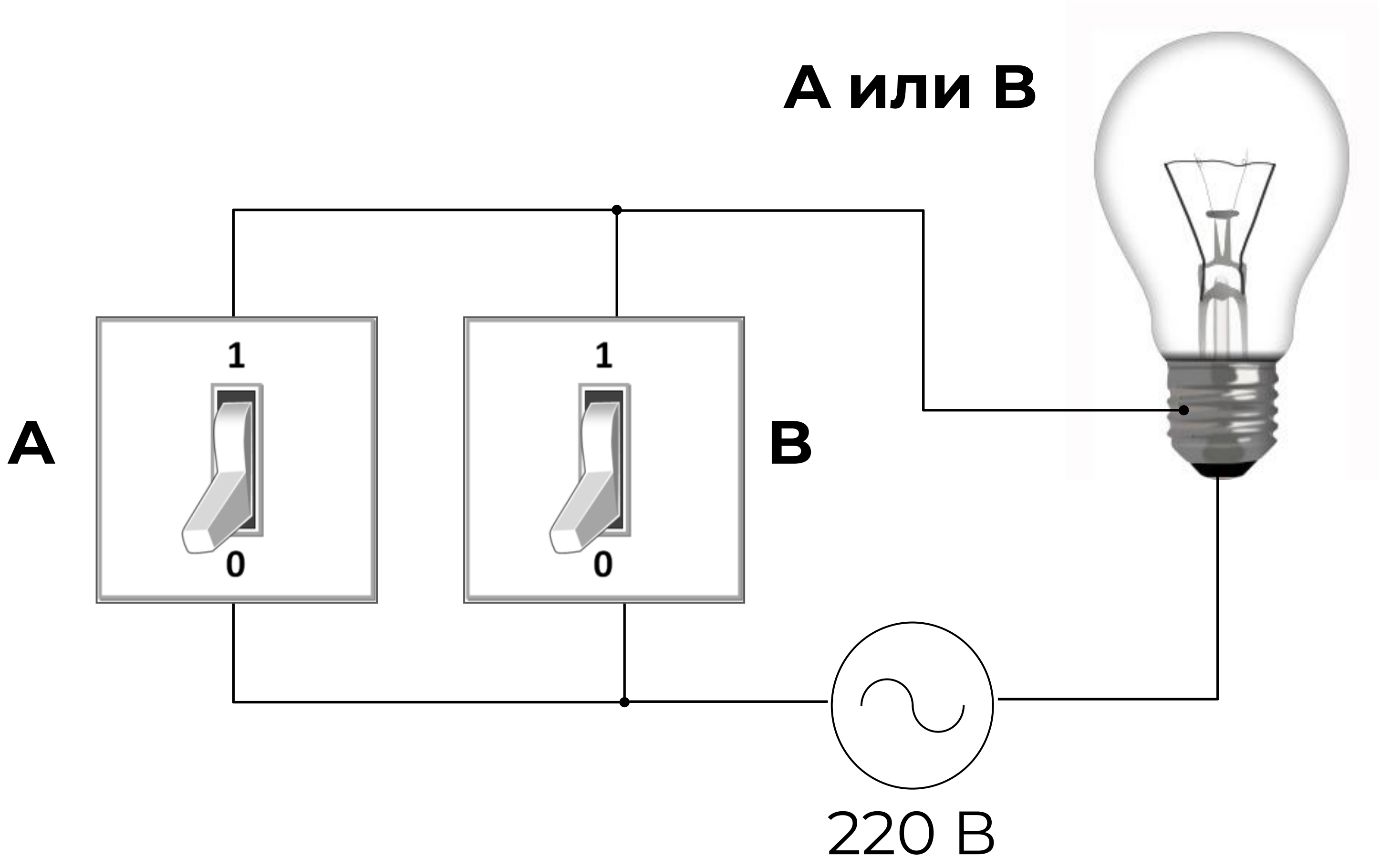
# Логическая схема «ИЛИ» - Дизъюнкция - OR

Схема «ИЛИ», выполненная с помощью реле

Логический элемент «ИЛИ»

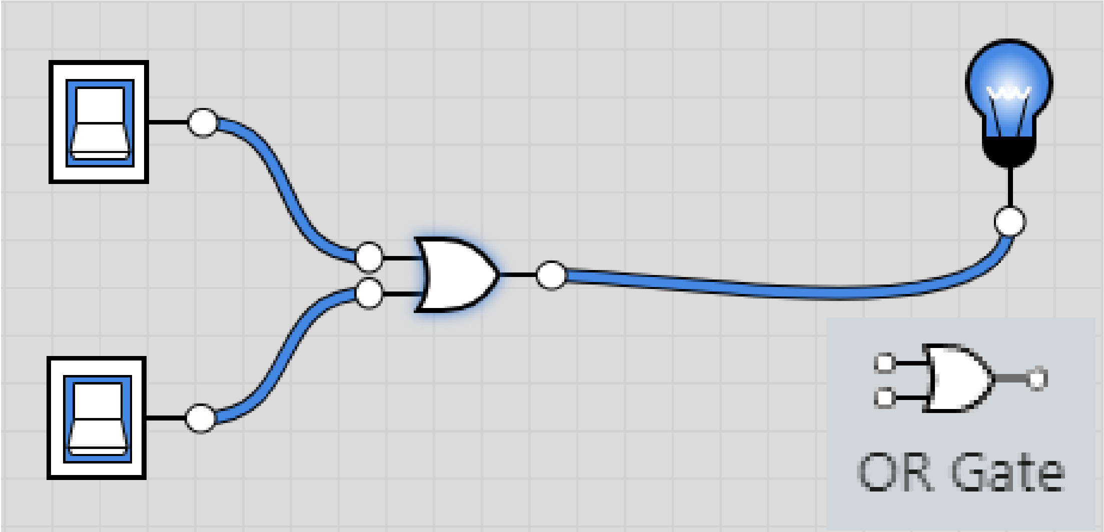


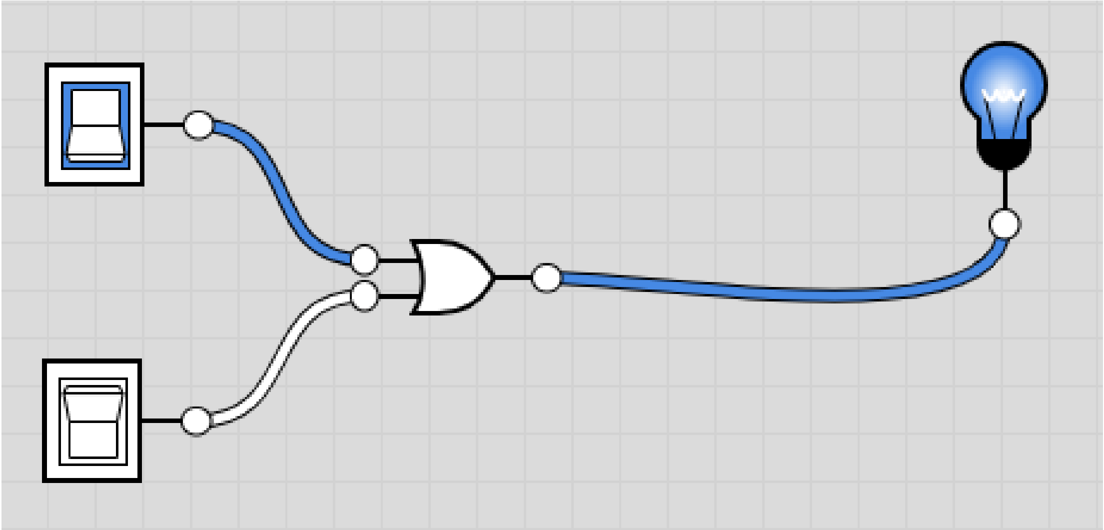
# Логическая схема «ИЛИ» - Дизъюнкция - OR

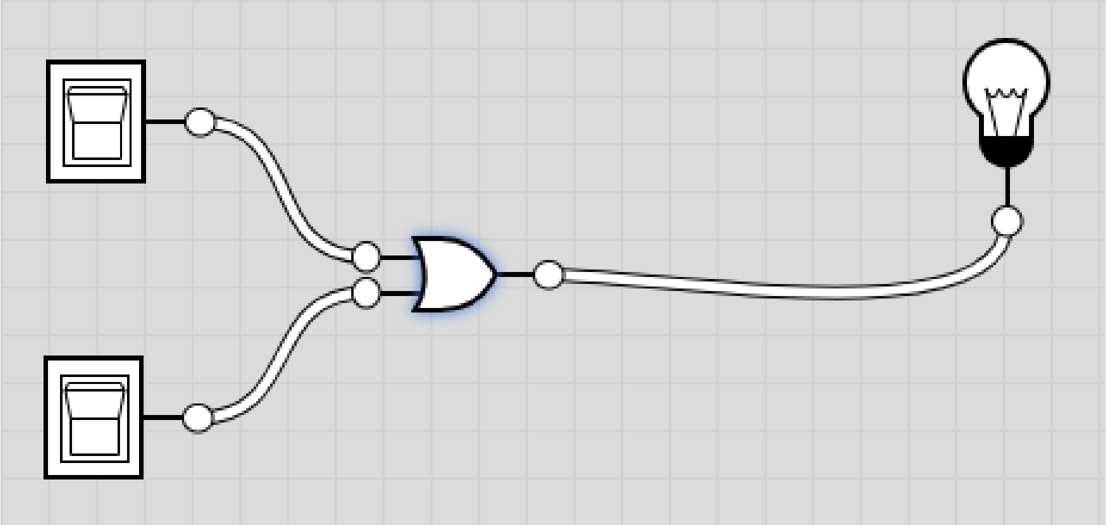


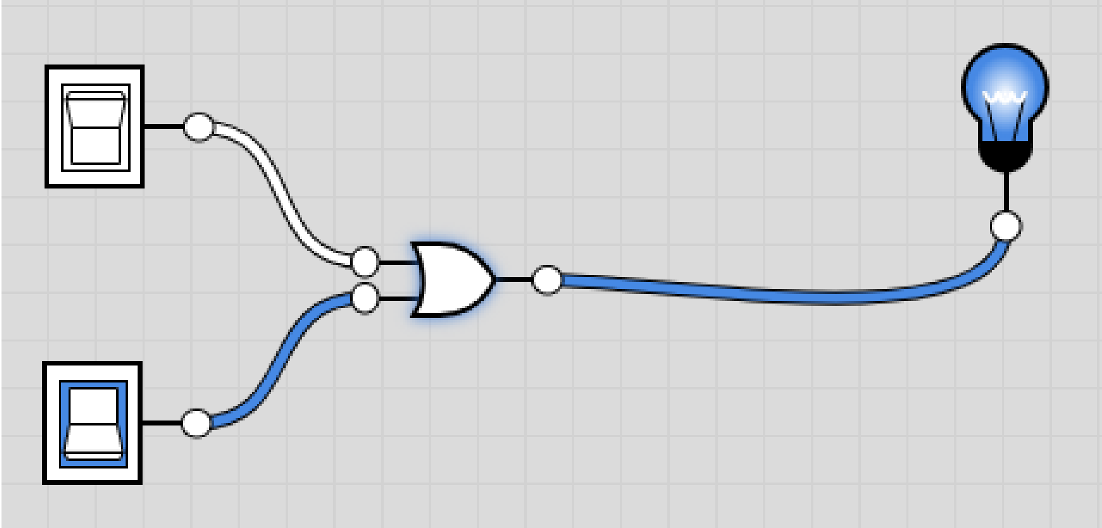

| A | B | A или B |
|---|---|---|
| 0 | 0 | **0** |
| 0 | 1 | **1** |
| 1 | 0 | **1** |
| 1 | 1 | **1** |

# Пример в logic.ly





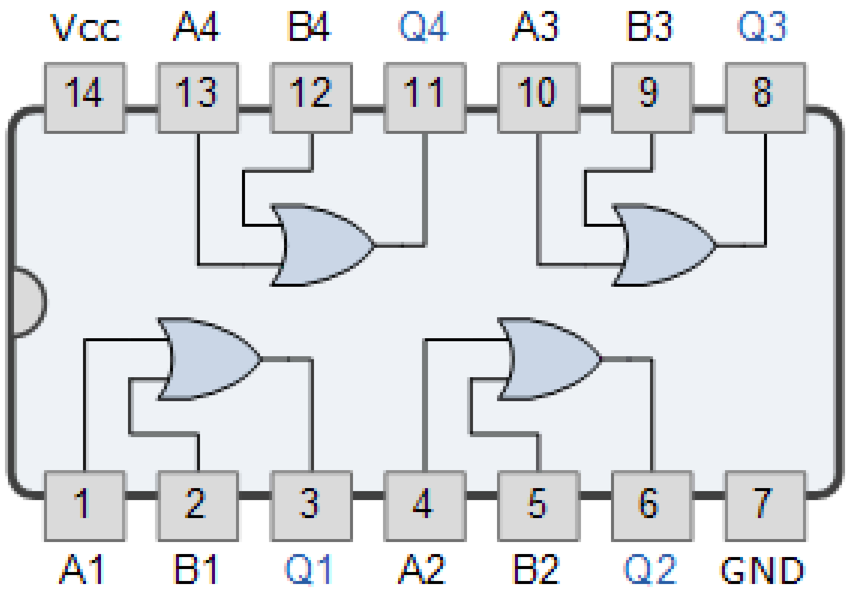


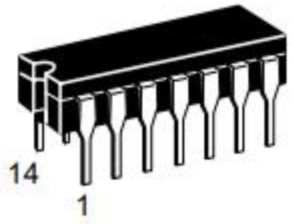
# Пример микросхемы с логическим элементом «ИЛИ»

MOTOROLA

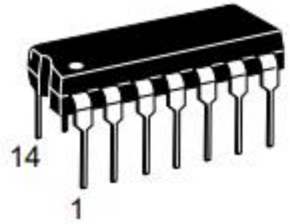
## QUAD 2-INPUT OR GATE

## SN54/74LS32




**J SUFFIX**
CERAMIC
CASE 632-08




**N SUFFIX**
PLASTIC
CASE 646-06

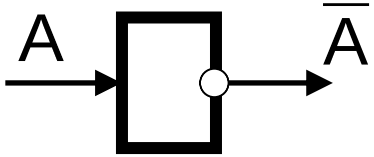

**D SUFFIX**
SOIC
CASE 751A-02

**ORDERING INFORMATION**

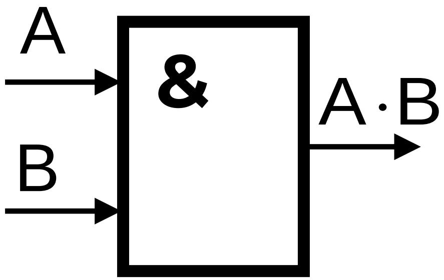
| | |
|---|---|
| SN54LSXXJ | Ceramic |
| SN74LSXXN | Plastic |
| SN74LSXXD | SOIC |

https://albertno-youtube.github.io/datasheets/7432_OR.pdf

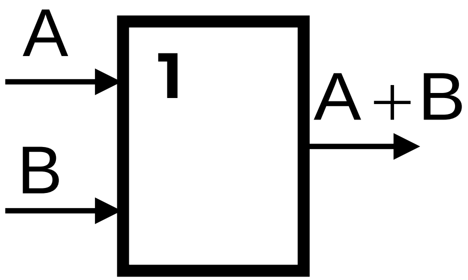
# Логические элементы компьютера



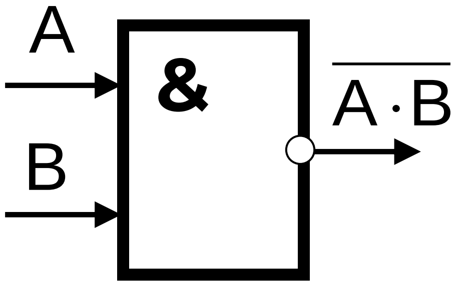

НЕ



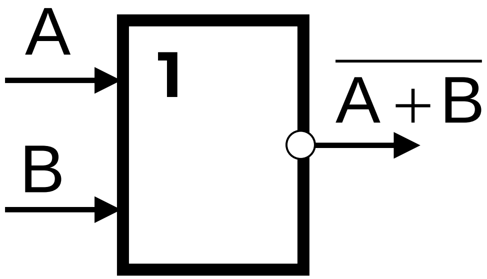

И




ИЛИ



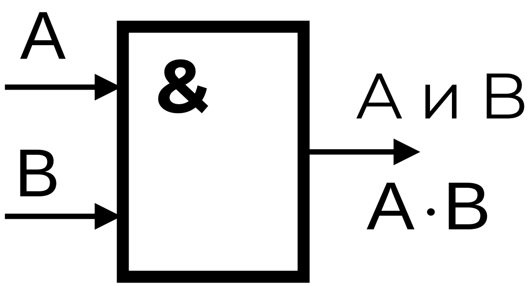

И-НЕ



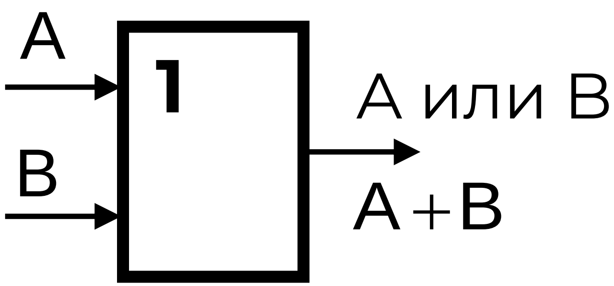

ИЛИ-НЕ

# Элементы «И» и «ИЛИ»

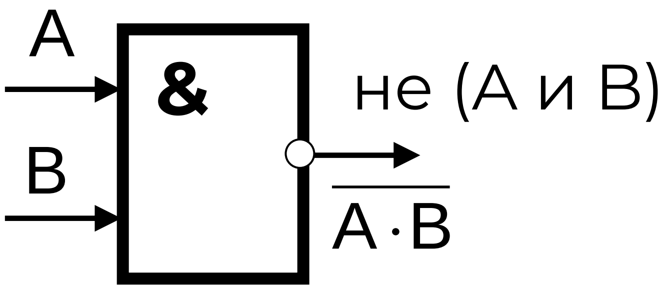
**«И»**

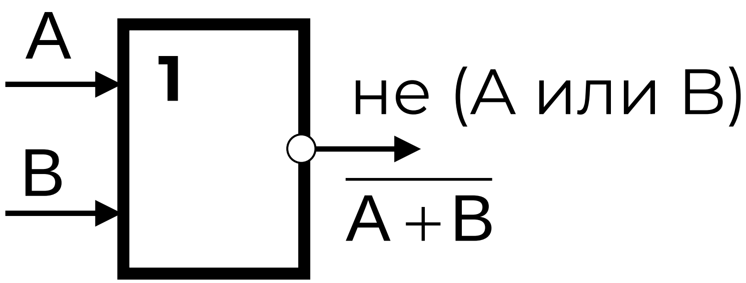



**«ИЛИ»**

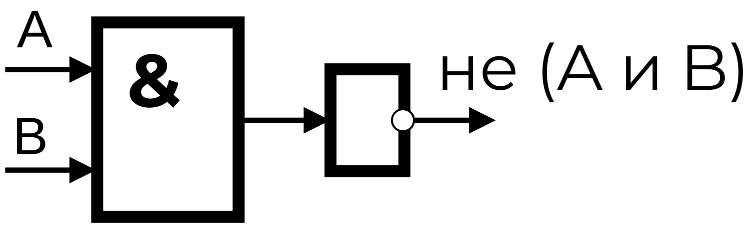



## Двойные элементы:

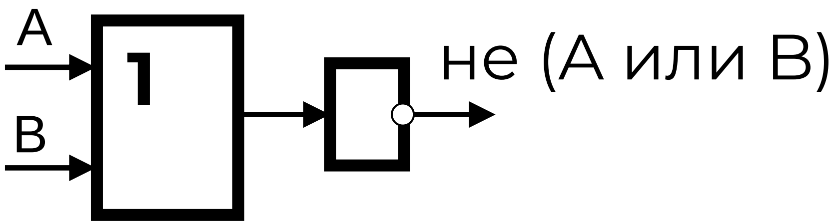



**«И-НЕ»**

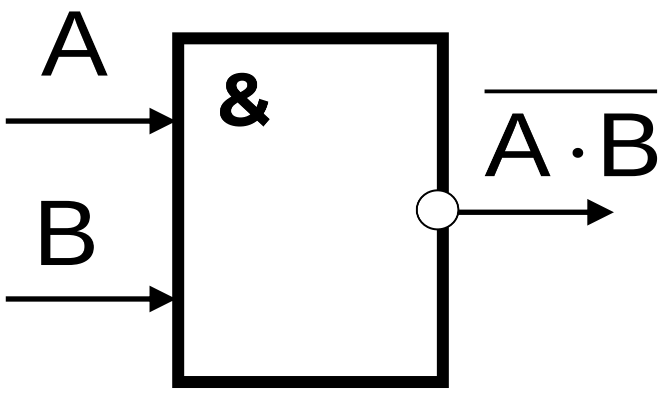



**«ИЛИ-НЕ»**



# Логическая схема «И-НЕ»

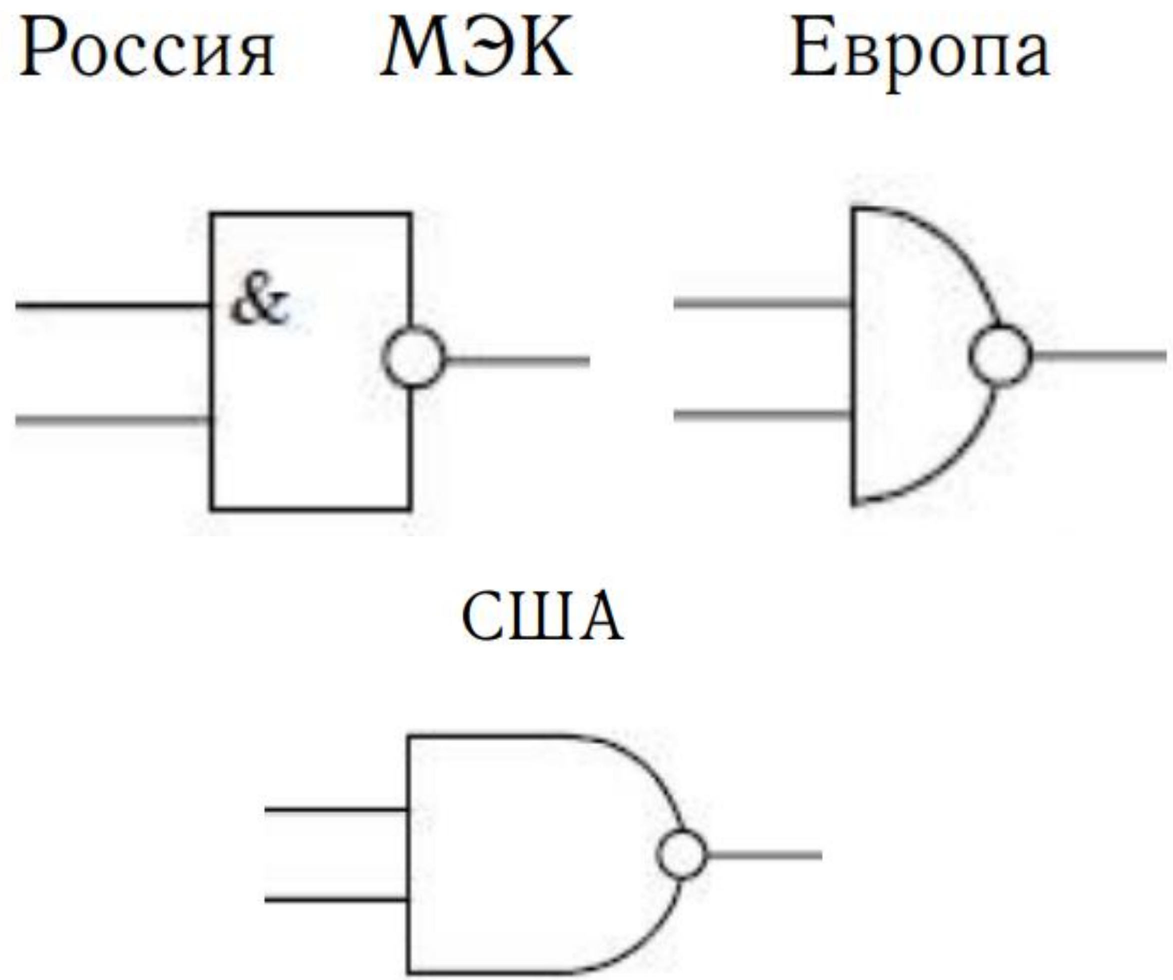
- **Инверсия функции конъюнкции.**
  **Операция «И-НЕ» (штрих Шеффера)**
- Мнемоническое правило для И-НЕ с любым количеством входов звучит так — на выходе будет:
- **«1»** тогда и только тогда, когда хотя бы на одном входе действует «0»,
- **«0»** тогда и только тогда, когда на всех входах действуют «1».

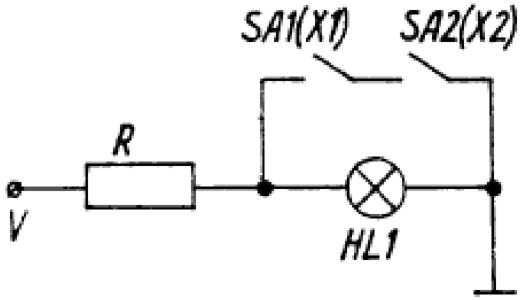


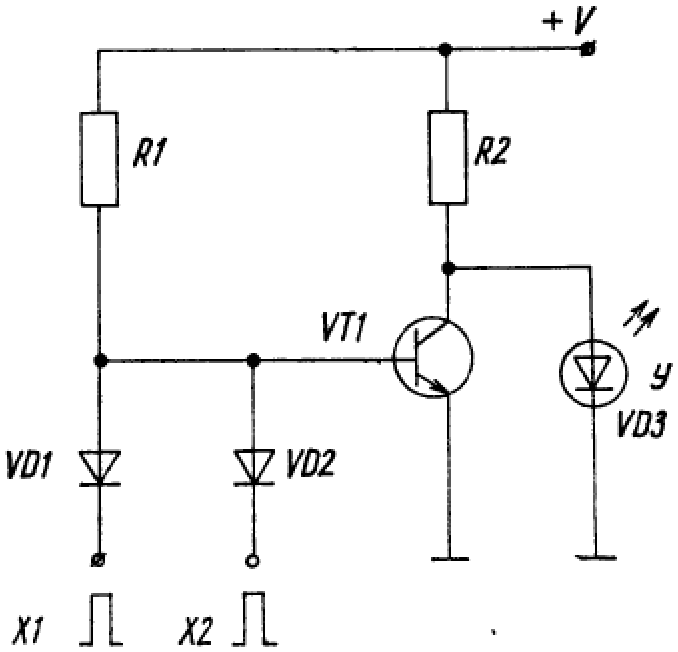

**И-НЕ**

| **A** | **B** | **A\|B** |
|---|---|---|
| 0 | 0 | **1** |
| 0 | 1 | **1** |
| 1 | 0 | **1** |
| 1 | 1 | **0** |

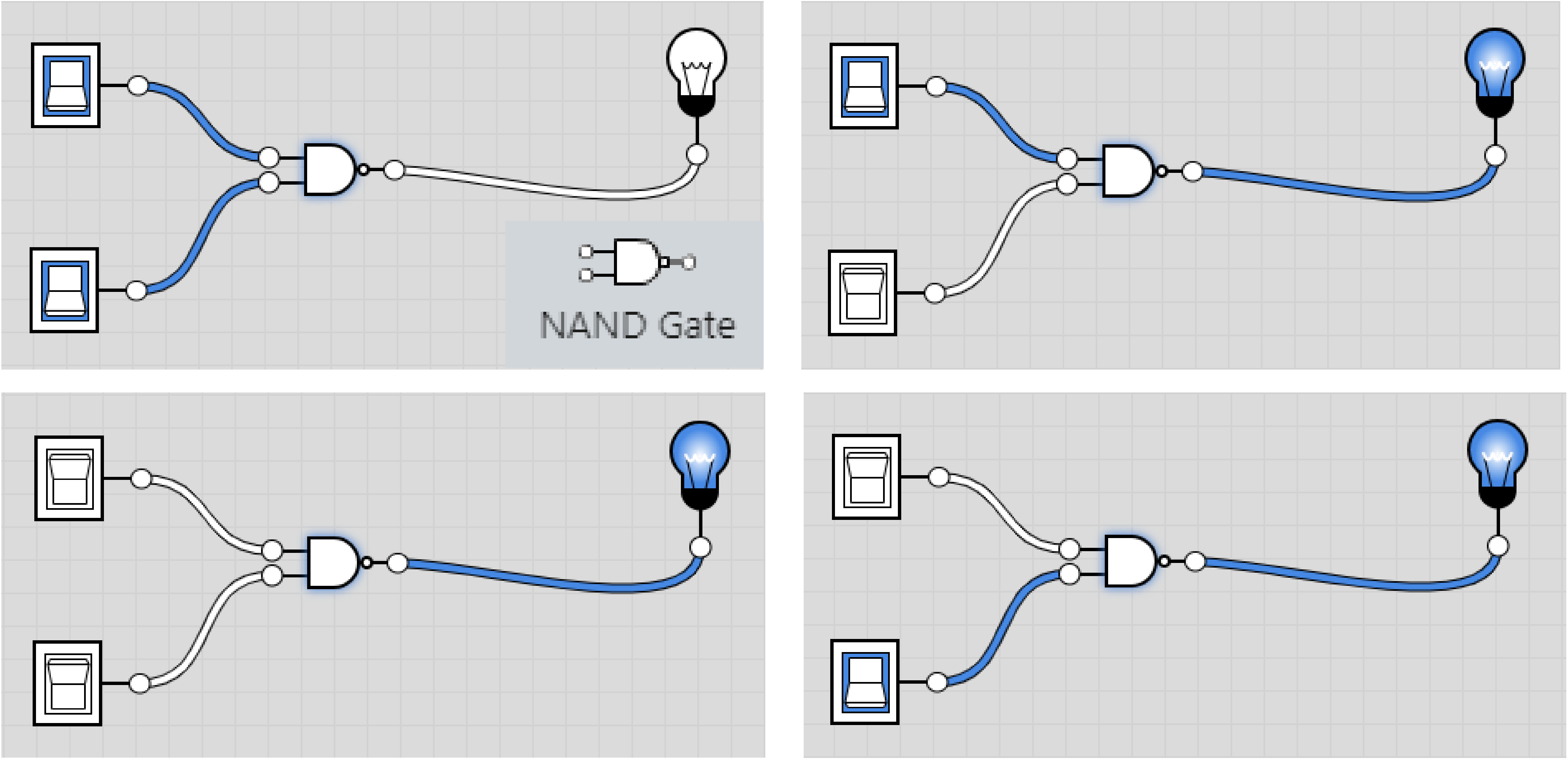
# Логическая операция **И-НЕ**, Штрих Шеффера (NOT AND, NAND)



# Пример в logic.ly



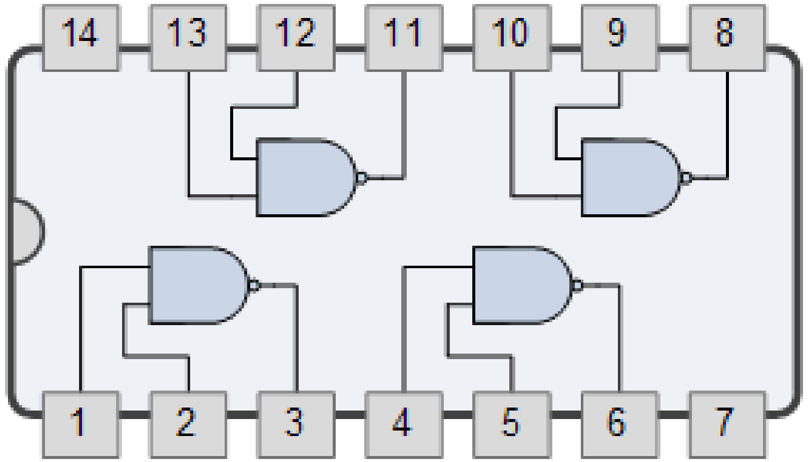
# Пример микросхемы с логическим элементом «И-НЕ»

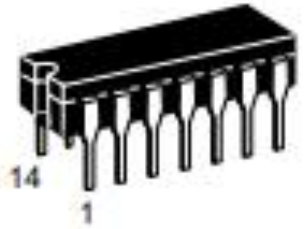
MOTOROLA

**QUAD 2-INPUT NAND GATE**

**SN54/74LS00**

7400 Quad 2-input Logic NAND Gate




**J SUFFIX**
CERAMIC
CASE 632-08


**N SUFFIX**
PLASTIC
CASE 646-06

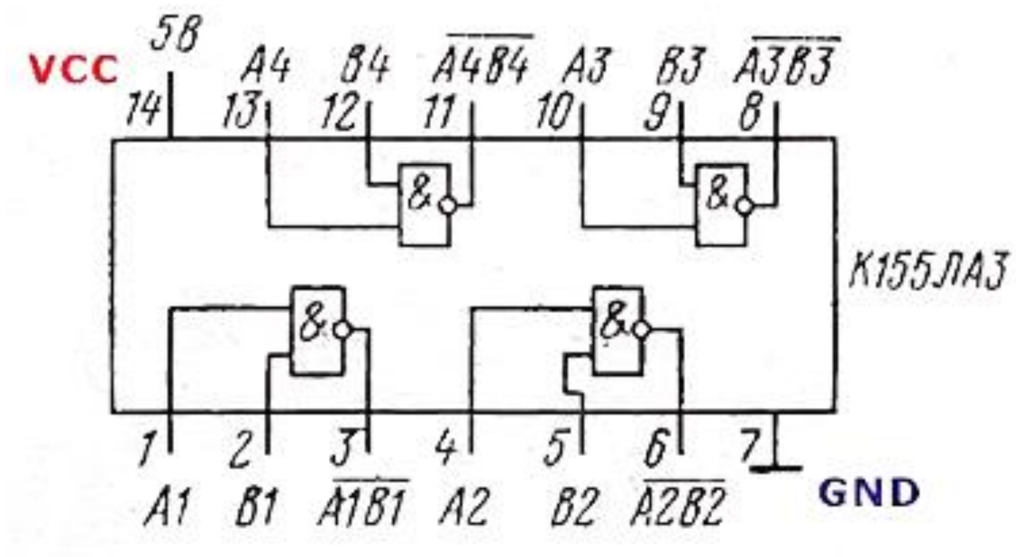

**D SUFFIX**
SOIC
CASE 751A-02

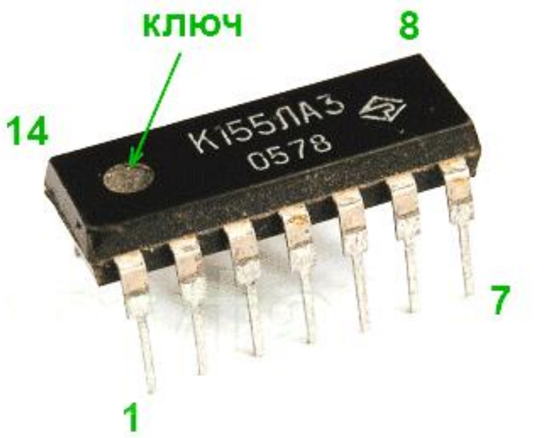
**ORDERING INFORMATION**

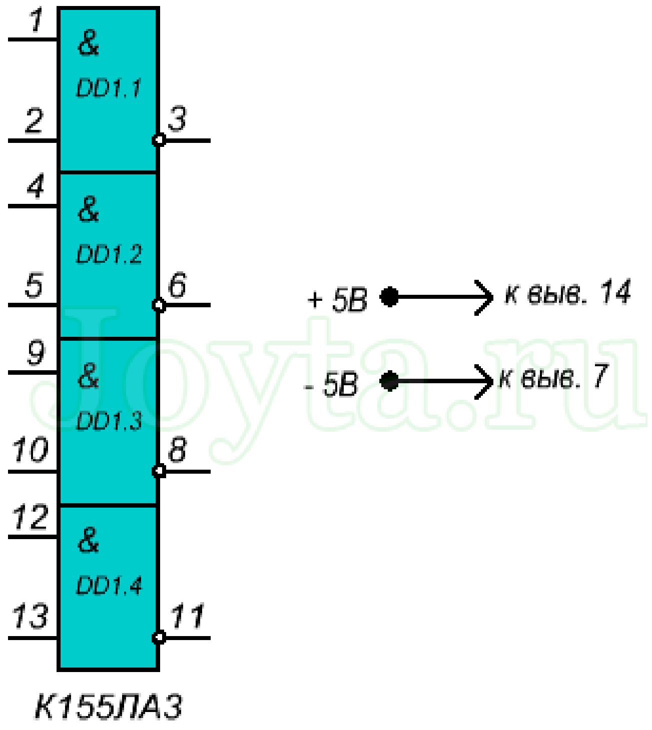
| | |
|---|---|
| SN54LSXXJ | Ceramic |
| SN74LSXXN | Plastic |
| SN74LSXXD | SOIC |

https://albertno-youtube.github.io/datasheets/7400_NAND.pdf

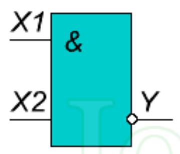
# Пример: микросхема к155ла3



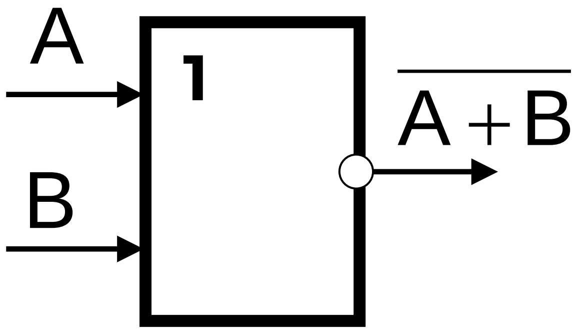

**Микросхема К155ЛА3: логическая, ТТЛ(транзисторно-транзисторная логика).**
**В функциональной структуре микросхемы К155ЛА3 имеется 4 самостоятельных логических элементов 2И-НЕ.** «Двойка» означает количество входов, а «И-НЕ» логическую функцию, как раз и известную как «инверсия конъюнкции». Одно лишь их объединяет, а это линии питания (общий вывод — 7, вывод 14 – положительный полюс питания) Как правило, контакты питания микросхем не изображаются на принципиальных схемах.
Каждый отдельный 2И-НЕ элемент микросхемы К155ЛА3 на схеме обозначают DD1.1, DD1.2, DD1.3, DD1.4. По правую сторону элементов находятся выходы, по левую сторону входы.




| X1 | X2 | Y |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

# Логическая схема «ИЛИ-НЕ»

- **Инверсия функции дизъюнкции.**
  **Операция «ИЛИ-НЕ» (стрелка Пирса)**
- Мнемоническое правило для ИЛИ-НЕ с любым количеством входов звучит так — на выходе будет:
- **«1»** тогда и только тогда, когда на всех входах действуют «0»,
- **«0»** тогда и только тогда, когда хотя бы на одном входе действует «1».

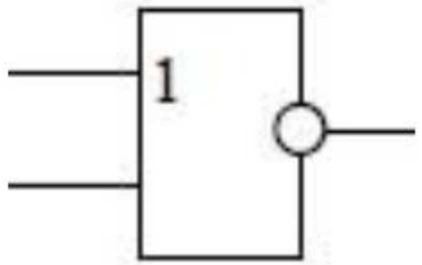



**ИЛИ-НЕ**

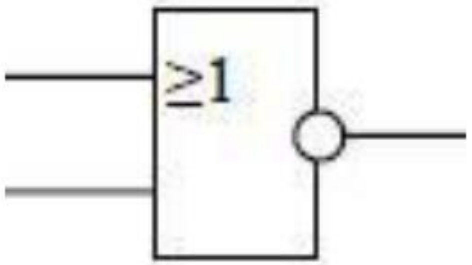
| A | B | A↓B |
|---|---|---|
| 0 | 0 | **1** |
| 0 | 1 | **0** |
| 1 | 0 | **0** |
| 1 | 1 | **0** |

# Логическая операция **ИЛИ-НЕ**, Стрелка Пирса (NOT OR)

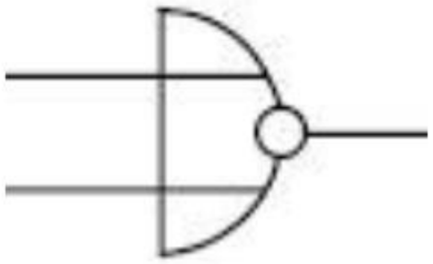
Россия



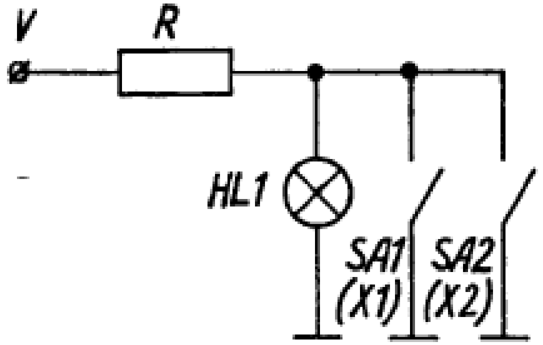

МЭК

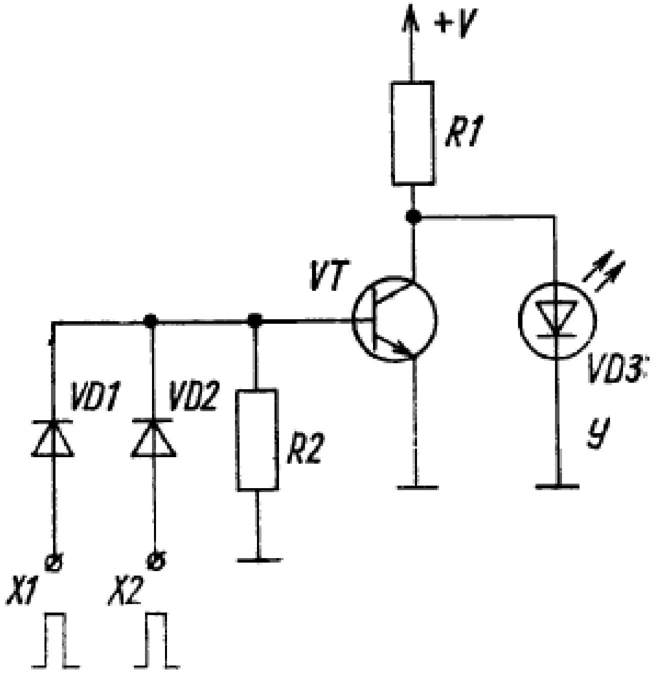



Европа

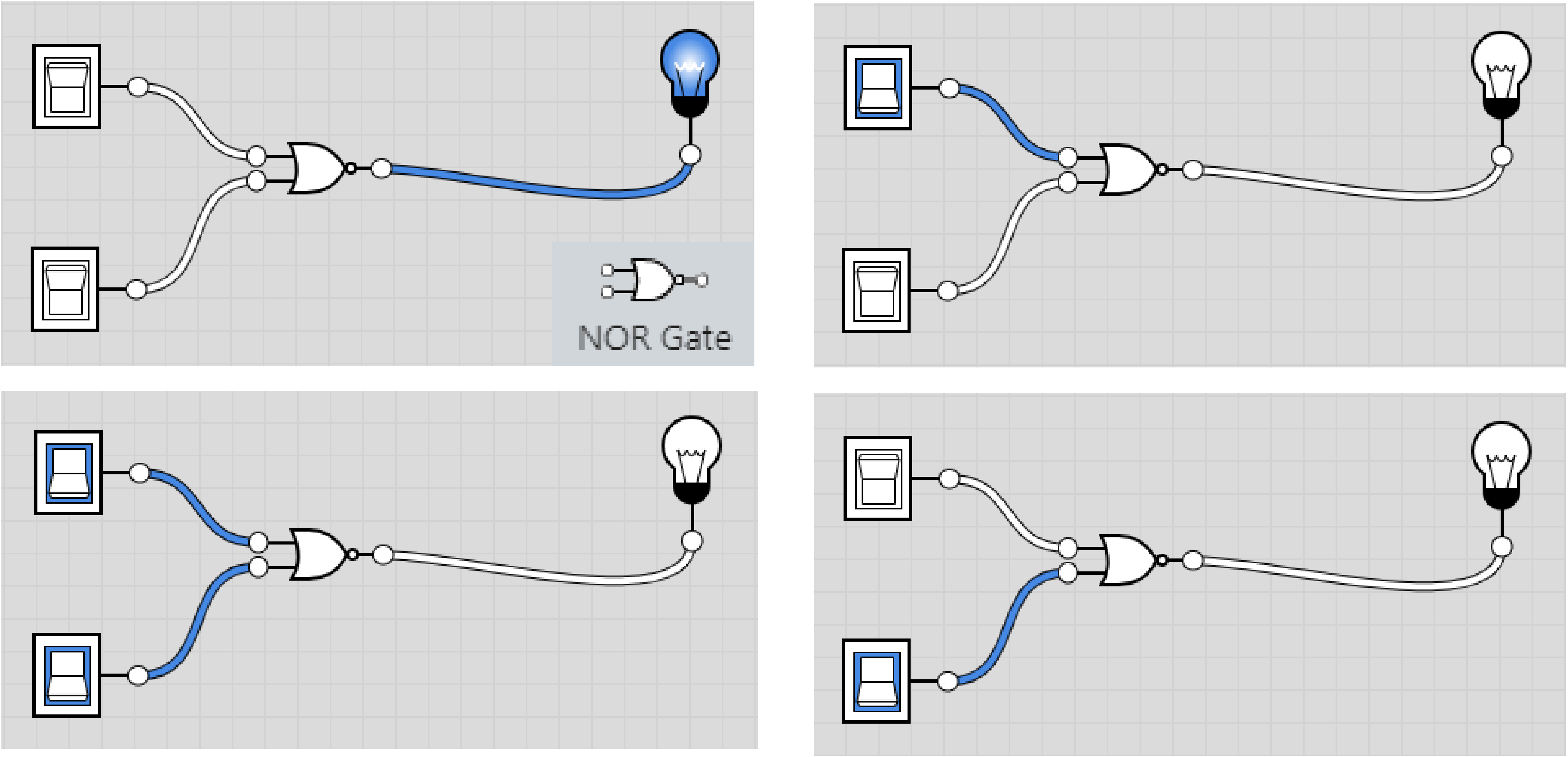


США



# Пример в logic.ly



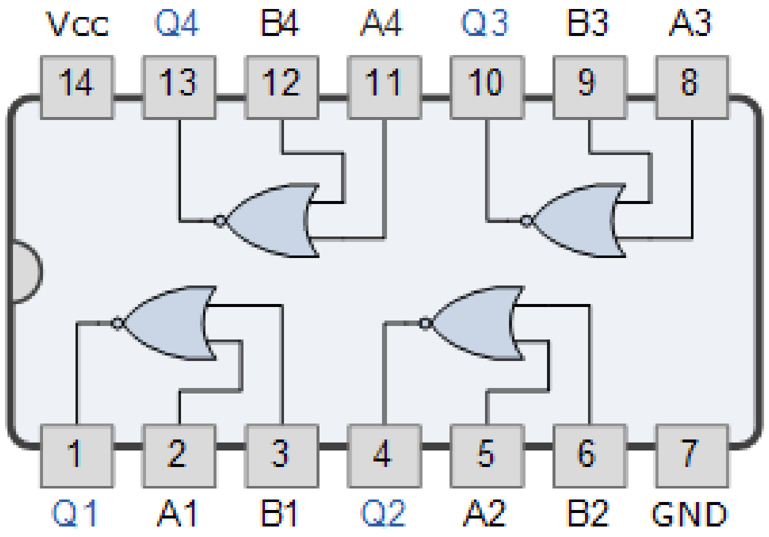
# Пример микросхемы с логическим элементом «ИЛИ-НЕ»

MOTOROLA

**QUAD 2-INPUT NOR GATE**

**SN54/74LS02**

7402 Quad 2-input Logic NOR Gate




https://albertno-youtube.github.io/datasheets/7402_NOR.pdf


**J SUFFIX**
CERAMIC
CASE 632-08

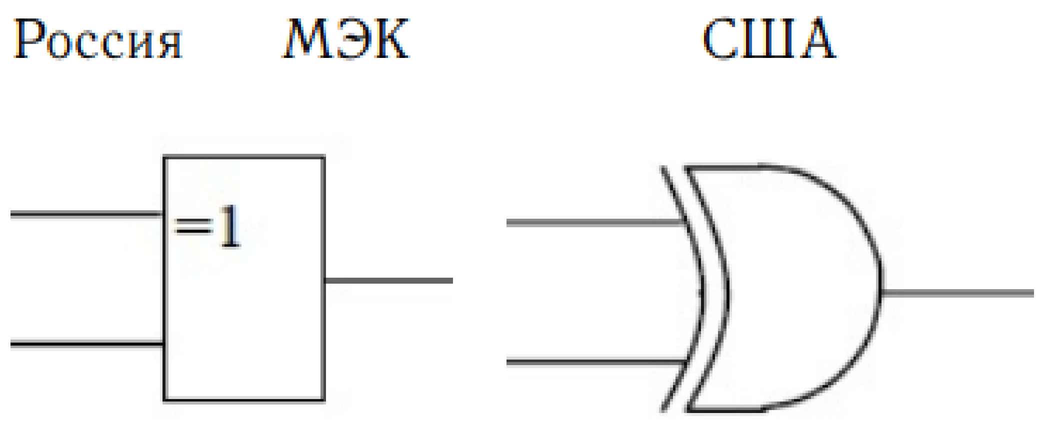

**N SUFFIX**
PLASTIC
CASE 646-06


**D SUFFIX**
SOIC
CASE 751A-02

**ORDERING INFORMATION**

| | |
|---|---|
| SN54LSXXJ | Ceramic |
| SN74LSXXN | Plastic |
| SN74LSXXD | SOIC |

# Логическая операция
# Исключающее ИЛИ (eXclusive OR, XOR)



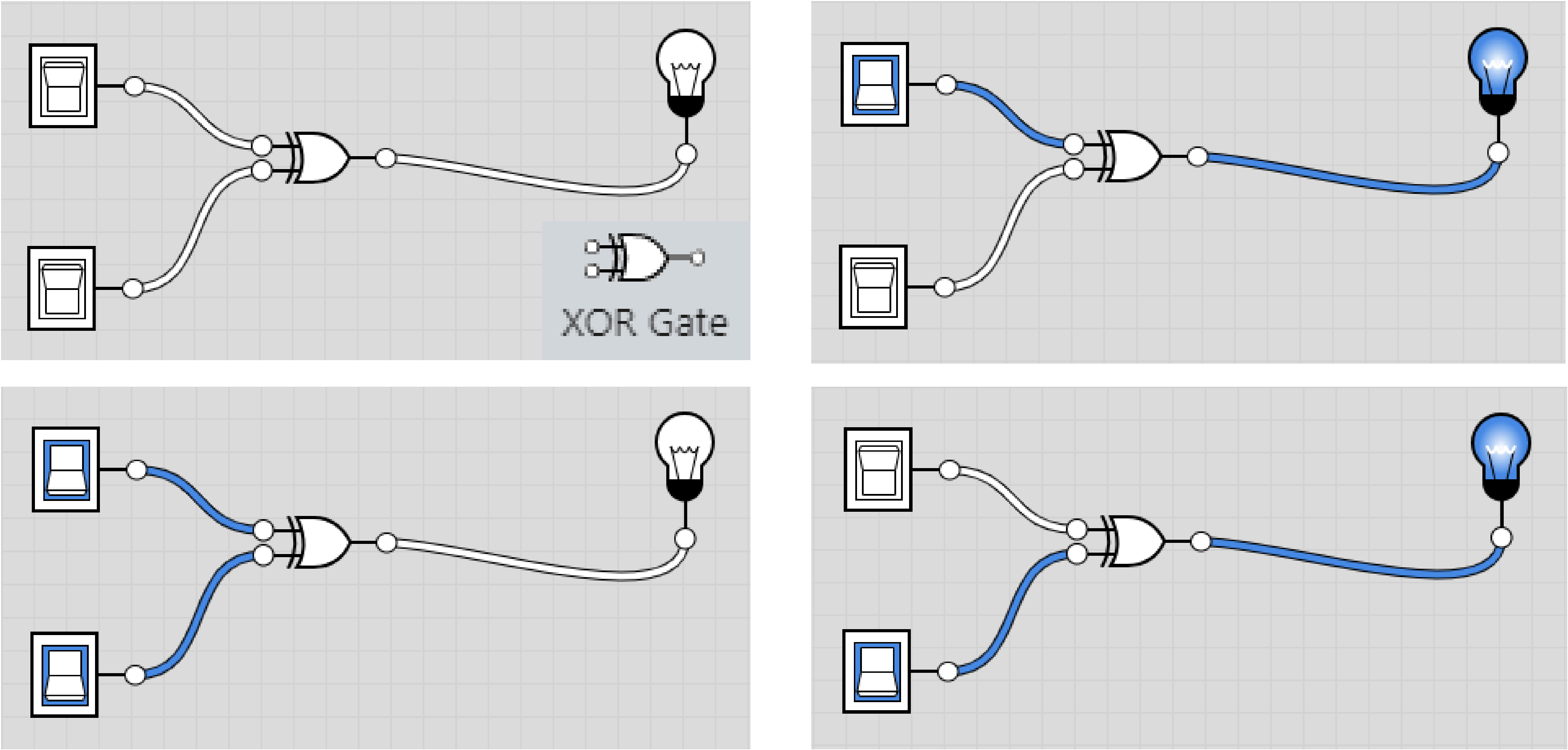

| A | B | A ⊕ B |
|---|---|---|
| 0 | 0 | **0** |
| 0 | 1 | **1** |
| 1 | 0 | **1** |
| 1 | 1 | **0** |

- **Сложение (сумма) по модулю 2 (неравнозначность, инверсия равнозначности). Операция «исключающее ИЛИ»**
- Мнемоническое правило для суммы по модулю 2 с любым количеством входов звучит так — на выходе будет:
- **«1»** тогда и только тогда, когда на входе действует нечётное количество «1»,
- **«0»** тогда и только тогда, когда на входе действует чётное количество «1».
- Словесное описание: «истина на выходе — при истине только на входе 1, либо при истине только на входе 2».

# Логическая операция **Исключающее ИЛИ** (eXclusive OR, XOR)

- **Строгая дизъюнкция обозначается символом $\oplus$.**
- В русском языке строгой (разделительной) дизъюнкции соответствует связка «либо».
- В отличие от обычной дизъюнкции (связка «или») в высказывании, содержащем строгую дизъюнкцию, мы утверждаем, что произойдёт только одно событие.
- **Например,** высказывая утверждение «На сегодняшнем матче Петя сидит на трибуне А либо на трибуне Б», мы считаем, что Петя сидит либо только на трибуне А, либо только на трибуне Б, и что сидеть одновременно на двух трибунах Петя не может.

# Пример в logic.ly

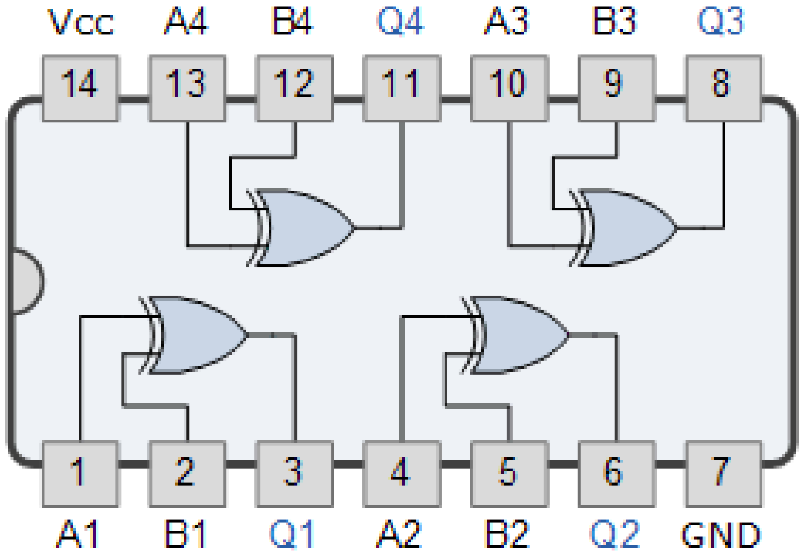


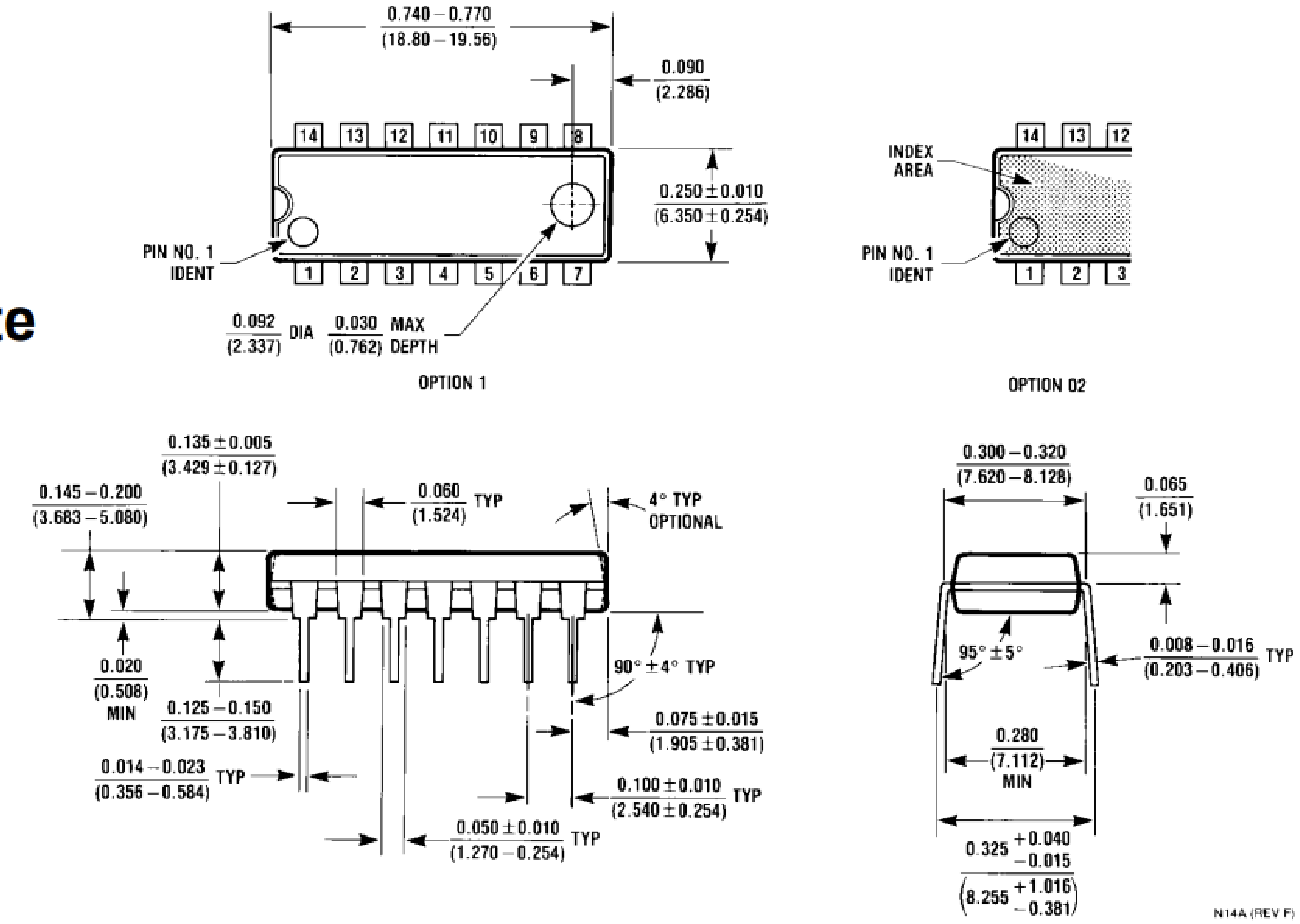
# Пример микросхемы с логическим элементом «XOR»

FAIRCHILD SEMICONDUCTOR™

**DM7486**
**Quad 2-Input Exclusive-OR Gate**

7486 Quad 2-input Exclusive-OR Gate

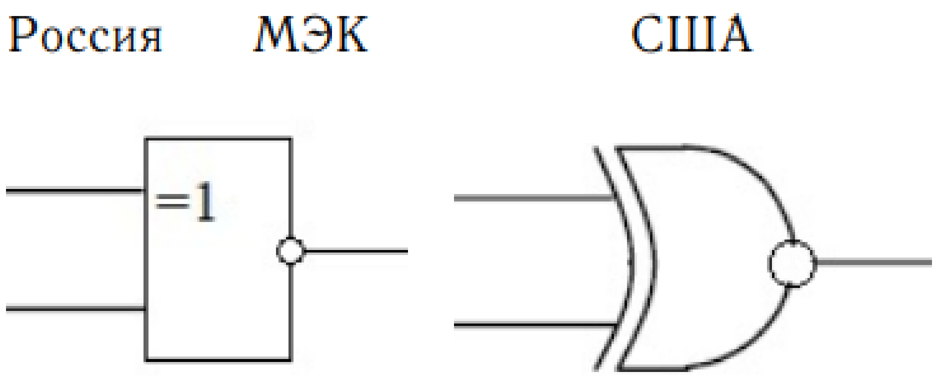



**14-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300 Wide Package Number N14A**

https://radio-hobby.org/uploads/datasheet/7/7486/7486.pdf

# Логическая операция
# Исключающее ИЛИ-НЕ (NOT eXclusive OR)




| A | B | A ↔ B |
|---|---|---|
| 0 | 0 | **1** |
| 0 | 1 | **0** |
| 1 | 0 | **0** |
| 1 | 1 | **1** |

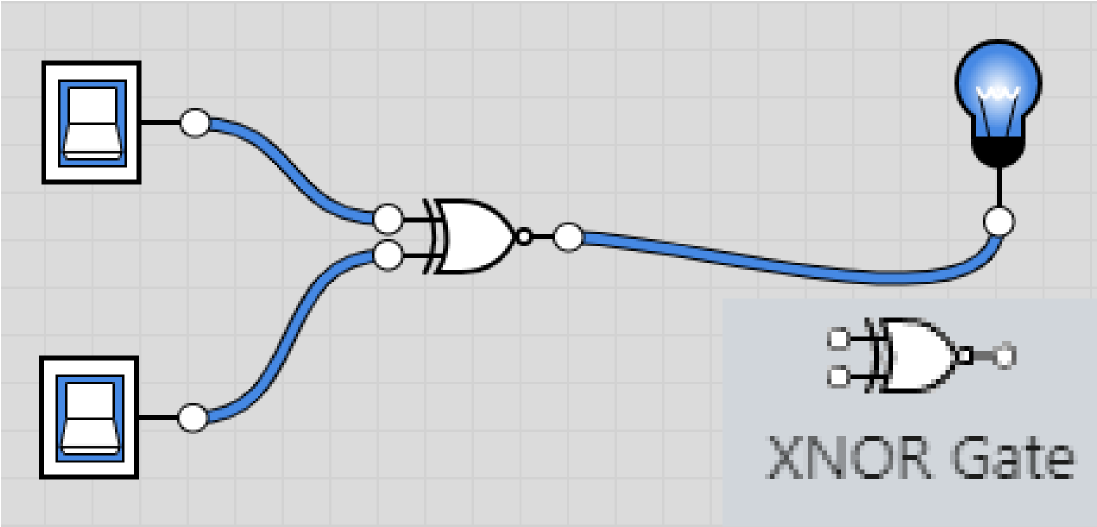
- **Эквивалентность (равнозначность, тождество). Операция «исключающее ИЛИ-НЕ»**
- Мнемоническое правило эквивалентности с любым количеством входов звучит так — на выходе будет:
- **«1»** тогда и только тогда, когда на входе действует чётное количество «1» или «0».
- **«0»** тогда и только тогда, когда на входе действует нечётное количество «1».
- Словесная запись: «истина на выходе при истине на входе 1 и входе 2 или при лжи на входе 1 и входе 2».

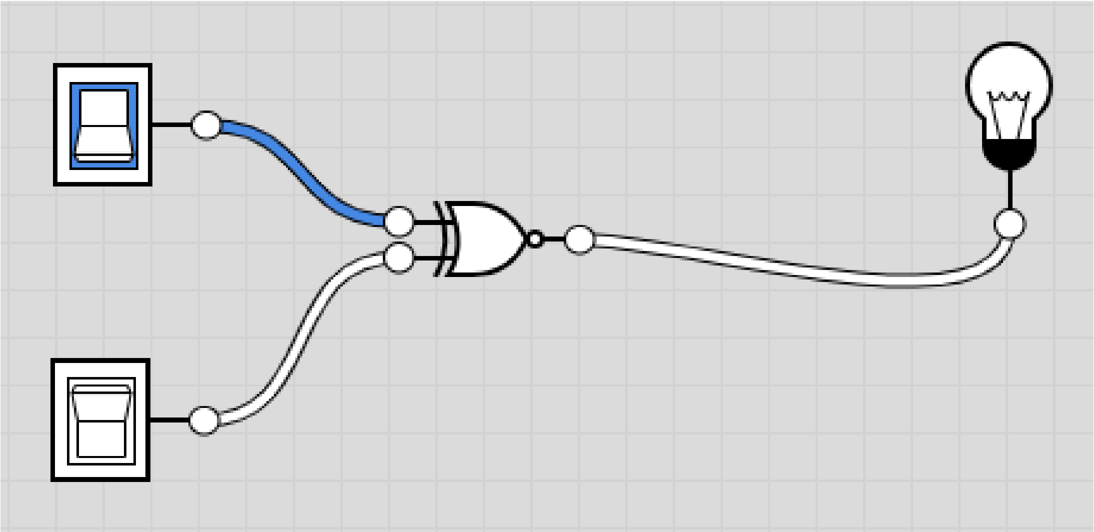
# Логическая операция **Исключающее ИЛИ-НЕ** (NOT eXclusive OR)

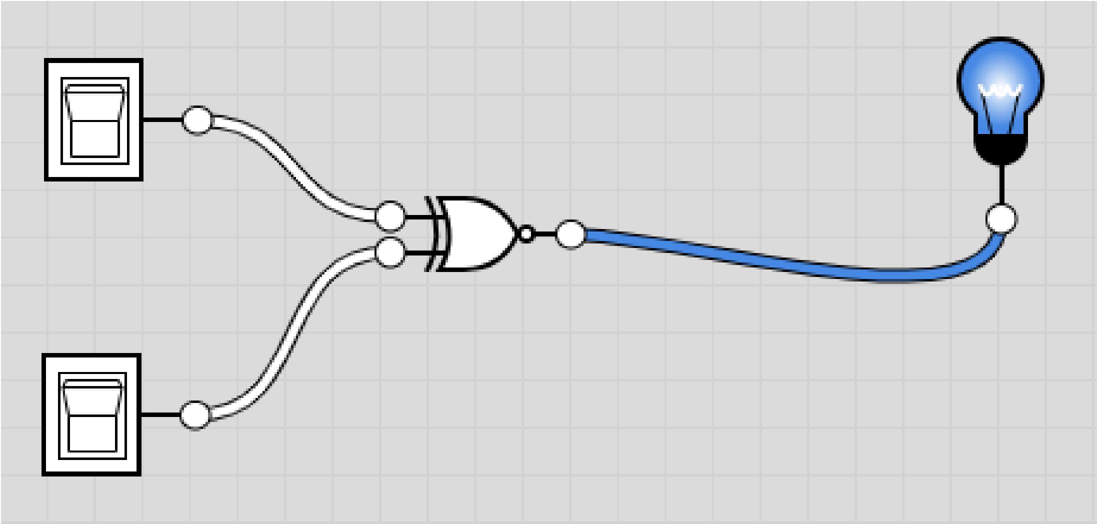
- **В логике эквиваленция обозначается символом ↔.**
- В разговорной речи для выражения взаимной обусловленности используется связка «тогда и только тогда, когда», а в математике — «необходимо и достаточно».

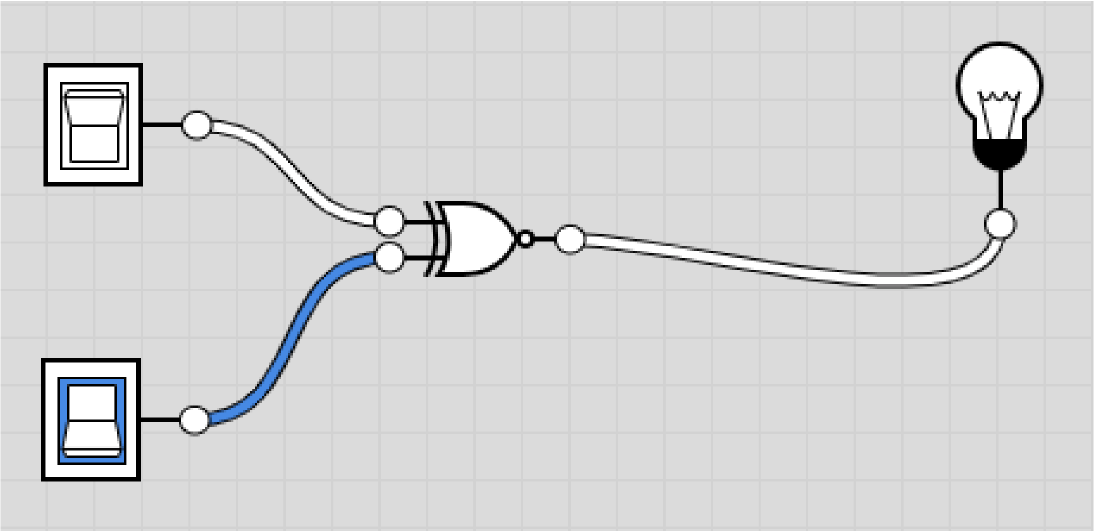
- Рассмотрим высказывание «Денис пойдёт в бассейн тогда и только тогда, когда он выучит уроки».
- Это высказывание истинно (договорённость соблюдается), если истинны оба элементарных высказывания («Денис пойдёт в бассейн», «Денис выучит уроки»).
- Высказывание истинно (договорённость не нарушается) и в том случае, если оба элементарных высказывания ложны («Денис не пойдёт в бассейн», «Денис не выучит уроки»).
- Если же одно из двух высказываний ложно («Денис пойдёт в бассейн, хотя и не выучит уроки», «Денис выучит уроки, но не пойдёт в бассейн»), то договорённость нарушается, и составное высказывание становится ложным.

# Пример в logic.ly







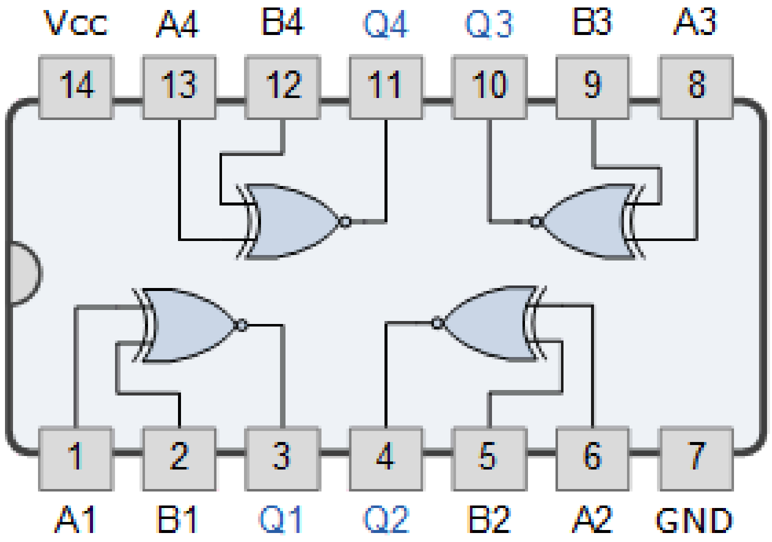
# Пример микросхемы с логическим элементом «ИЛИ-НЕ»

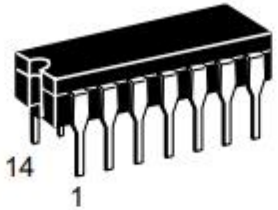
MOTOROLA

**QUAD 2-INPUT**
**EXCLUSIVE NOR GATE**

**SN54/74LS266**

74266 Quad 2-input Ex-NOR Gate




**J SUFFIX**
CERAMIC
CASE 632-08


**N SUFFIX**
PLASTIC
CASE 646-06

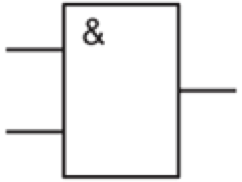
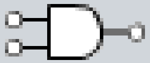
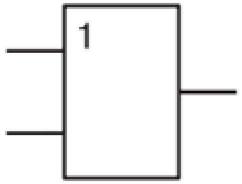
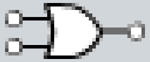
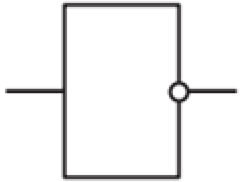
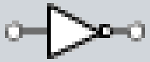
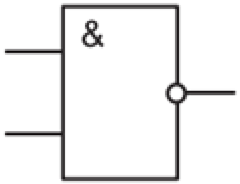
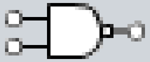
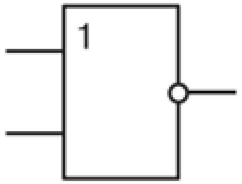
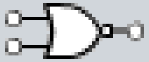

**D SUFFIX**
SOIC
CASE 751A-02

**ORDERING INFORMATION**

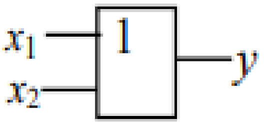
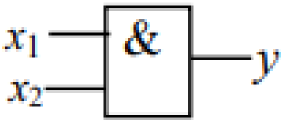
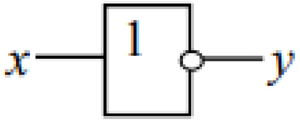
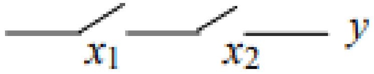
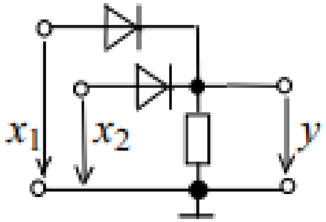
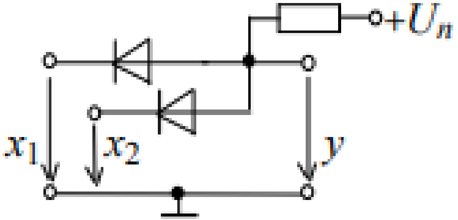
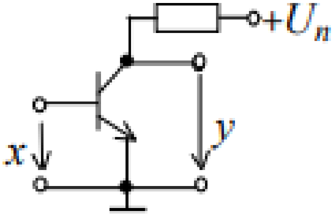
| | |
|---|---|
| SN54LSXXXJ | Ceramic |
| SN74LSXXXN | Plastic |
| SN74LSXXXD | SOIC |

https://datasheetspdf.com/pdf-file/620678/MotorolaSemiconductor/74266/1

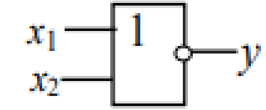
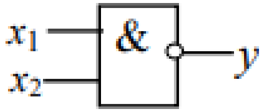
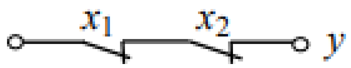
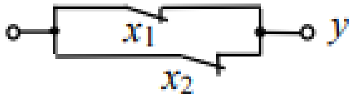
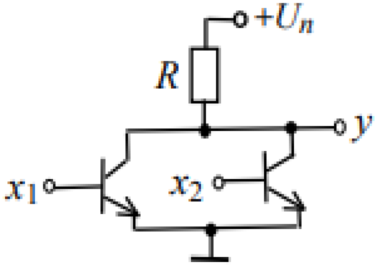
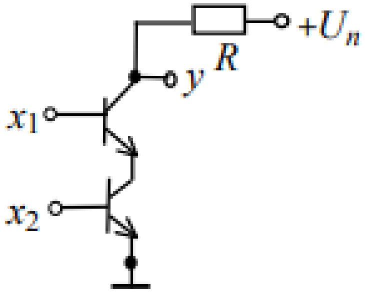
# Условные обозначения типовых логических элементов

| Наименование элемента | Условное обозначение | Обозначение в logic.ly | Название функции и её формула |
|---|---|---|---|
| **И**<br>AND |  | <br>AND Gate | Конъюнкция<br>$F = A \& B$ |
| **ИЛИ**<br>OR |  | <br>OR Gate | Дизъюнкция<br>$F = A \vee B$ |
| **НЕ**<br>NOT |  | <br>NOT Gate | Инверсия<br>$F = \overline{A}$ |
| **И-НЕ**<br>NOT AND (NAND) |  | <br>NAND Gate | Штрих Шеффера<br>$F = \overline{A \& B}$ |
| **ИЛИ-НЕ**<br>NOT OR |  | <br>NOR Gate | Стрелка Пирса<br>$F = \overline{A \vee B}$ |

# Формы отображения основных логических функций

<table>
<tr><td>Наименование</td><td>Дизъюнкция</td><td>Конъюнкция</td><td>Инверсия</td></tr>
<tr><td>Символическая</td><td>$\vee$ или +</td><td>$\wedge$ или $\cdot$</td><td>$\bar{x}$</td></tr>
<tr><td>Буквенная</td><td>ИЛИ</td><td>И</td><td>НЕ</td></tr>
<tr><td>Условная графическая</td><td></td><td></td><td></td></tr>
<tr><td>Аналитическая</td><td>$y = x_1 \vee x_2 = x_1 + x_2$</td><td>$y = x_1 \wedge x_2 = x_1 x_2$</td><td>$y = \bar{x}$</td></tr>
<tr><td>Табличная (истинности)</td><td><table><tr><td>$x_1$</td><td>$x_2$</td><td>$y$</td></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table></td><td><table><tr><td>$x_1$</td><td>$x_2$</td><td>$y$</td></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table></td><td><table><tr><td>$x$</td><td>$y$</td></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table></td></tr>
<tr><td>Контактная</td><td></td><td></td><td></td></tr>
<tr><td>Схемо-техническая</td><td></td><td></td><td></td></tr>
</table>

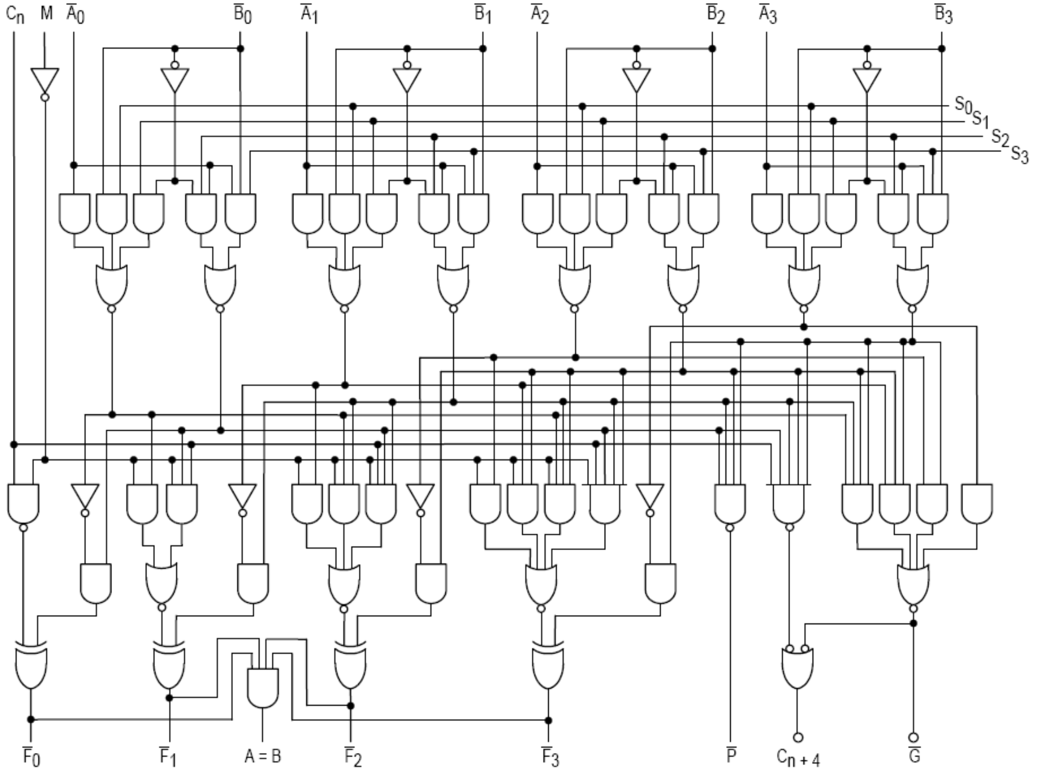
# Формы отображения основных логических функций

<table>
<tr><th>Наименование</th><th>Функция Пирса</th><th>Функция Шеффера</th></tr>
<tr><td>Символическая</td><td>$\downarrow$</td><td>$|$</td></tr>
<tr><td>Буквенная</td><td>ИЛИ-НЕ</td><td>И-НЕ</td></tr>
<tr><td>Условная графическая</td><td></td><td></td></tr>
<tr><td>Аналитическая</td><td>$y = x_1 \downarrow x_2$</td><td>$y = x_1 | x_2$</td></tr>
<tr><td>Табличная (истинности)</td><td><table>
<tr><th>$x_1$</th><th>$x_2$</th><th>$y$</th></tr>
<tr><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>0</td></tr>
</table></td><td><table>
<tr><th>$x_1$</th><th>$x_2$</th><th>$y$</th></tr>
<tr><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>0</td></tr>
</table></td></tr>
<tr><td>Контактная</td><td></td><td></td></tr>
<tr><td>Схемо-техническая</td><td></td><td></td></tr>
</table>

# Логическая реализация типовых устройств компьютера

- **Обработка любой информации на компьютере сводится к выполнению процессором различных арифметических и логических операций. Для этого в состав процессора входит так называемое арифметико-логическое устройство (АЛУ).** Оно состоит из ряда устройств, построенных на рассмотренных выше логических элементах.

- Важнейшими из таких устройств являются:
  - **триггеры,**
  - **полусумматоры,**
  - **сумматоры,**
  - **шифраторы,**
  - **дешифраторы,**
  - **счетчики,**
  - **регистры.**



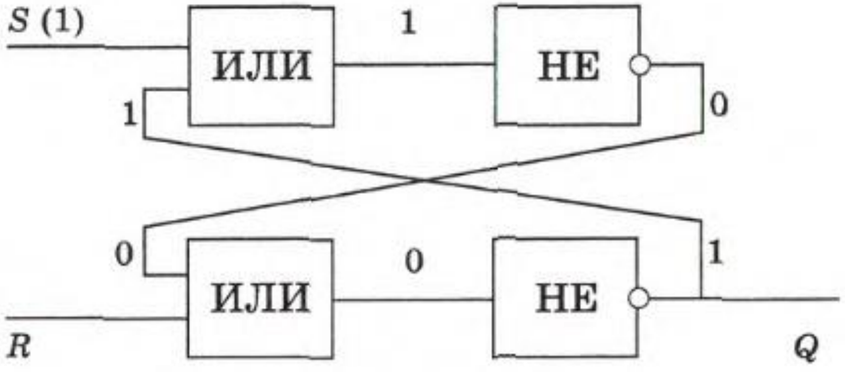

Комбинационная логическая схема 4-битного АЛУ, реализованная в 24-выводной микросхеме ТТЛ, модель 74181

# Этапы конструирования логического устройства

- **Конструирование логического устройства состоит из следующих этапов:**
- **1. Построение таблицы истинности** по заданным условиям работы проектируемого узла (т.е. по соответствию его входных и выходных сигналов).
- **2. Конструирование логической функции** данного узла по таблице истинности, ее преобразование (упрощение), если это возможно и необходимо.
- **3. Составление функциональной схемы** проектируемого узла по формуле логической функции.
- После этого остается только реализовать полученную схему.

# Триггер

- **Важнейшей структурной единицей оперативной памяти компьютера, а также внутренних регистров процессора является триггер.**
- **Триггер** - электронная схема, применяемая для хранения значения одноразрядного двоичного кода.
- **Триггер может находиться в одном из двух устойчивых состояний, что позволяет запоминать, хранить и считывать 1 бит информации.**
- Триггер можно построить из двух логических элементов «ИЛИ» и двух элементов «НЕ». При этом выход первого элемента «НЕ» соединен со входом второго элемента «ИЛИ», а выход второго элемента «НЕ» соединен со входом первого элемента «ИЛИ». Триггер имеет установочный вход S (от англ, set — установка) и вход сброса R (от англ, reset — сброс), а также выход Q.
- Для записи 1 бита на вход S (установочный) подается сигнал 1. Последовательно рассмотрев по логической схеме прохождение сигнала, видим, что на выходе триггера Q устанавливается 1. Триггер переходит в это состояние и будет находиться в нем и после того, как сигнал на входе S исчезнет. Таким образом, триггер будет устойчиво хранить 1 бит.




| Входы | | Выход Q |
|---|---|---|
| S | R | |
| 0 | 0 | Q |
| 1 | 0 | 1 |
| 0 | 1 | 0 |

Для того чтобы сбросить бит данных и подготовиться к приему нового бита, подается сигнал 1 на вход (сброс), после чего триггер возвратится к исходному «нулевому» состоянию.
Подача на оба входа S и Я логической единицы может привести к неоднозначному результату, поэтому такая комбинация входных сигналов запрещена.

# Триггер

- **Триггеры можно классифицировать**
    - по способу приема информации,
    - по принципу построения,
    - по функциональным возможностям.

- **По способу приема информации** триггеры подразделяются на асинхронные и синхронные.

- **Асинхронные** триггеры воспринимают информационные сигналы и реагируют на них в момент появления на входах триггера. **Синхронные** триггеры реагируют на информационные сигналы при наличии разрешающего сигнала на специальном управляющем входе C, называемом входом синхронизации (тактовым входом).

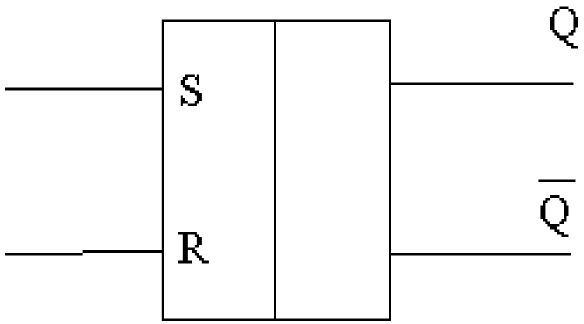
# Триггер

- По принципу построения триггеры со статическим управлением можно разделить на одноступенчатые триггеры и двухступенчатые триггеры.
- Одноступенчатые триггеры характеризуются наличием одной ступени запоминания информации. В двухступенчатых триггерах имеются две ступени запоминания информации: вначале информация записывается в первую ступень, а затем переписывается во вторую и появляется на выходе.

- **По функциональным возможностям различаются:**
  - триггер с раздельной установкой состояний 0 и 1 (**RS-триггер**);
  - триггер с приемом информации по одному входу D (**D-триггер** или триггер задержки);
  - триггер со счетным входом T (**T-триггер**);
  - универсальный триггер с информационными входами J и K (**JK-триггер**).

# Триггер (англ. trigger – защёлка)

- **Триггер** - электронная схема, применяемая для хранения значения одноразрядного двоичного кода.
- Воздействуя на входы триггера, его переводят в одно из двух возможных состояний (0 или 1). С поступлением сигналов на входы триггера в зависимости от его состояния либо происходит переключение, либо исходное состояние сохраняется. При отсутствии входных сигналов триггер сохраняет свое состояние сколь угодно долго.
- Термин триггер происходит от английского слова trigger - защёлка, спусковой крючок. Для обозначения этой схемы в английском языке чаще употребляется термин flip-flop, что в переводе означает "хлопанье". Это звукоподражательное название электронной схемы указывает на её способность почти мгновенно переходить ("перебрасываться") из одного электрического состояния в другое.
- Существуют разные варианты исполнения триггеров в зависимости от элементной базы (И-НЕ, ИЛИ-НЕ) и функциональных связей между сигналами на входах и выходах (RS, JK, T, D и другие).
- Самый распространённый тип триггера - это RS-триггер (S и R соответственно от английских set - установка, и reset - сброс).

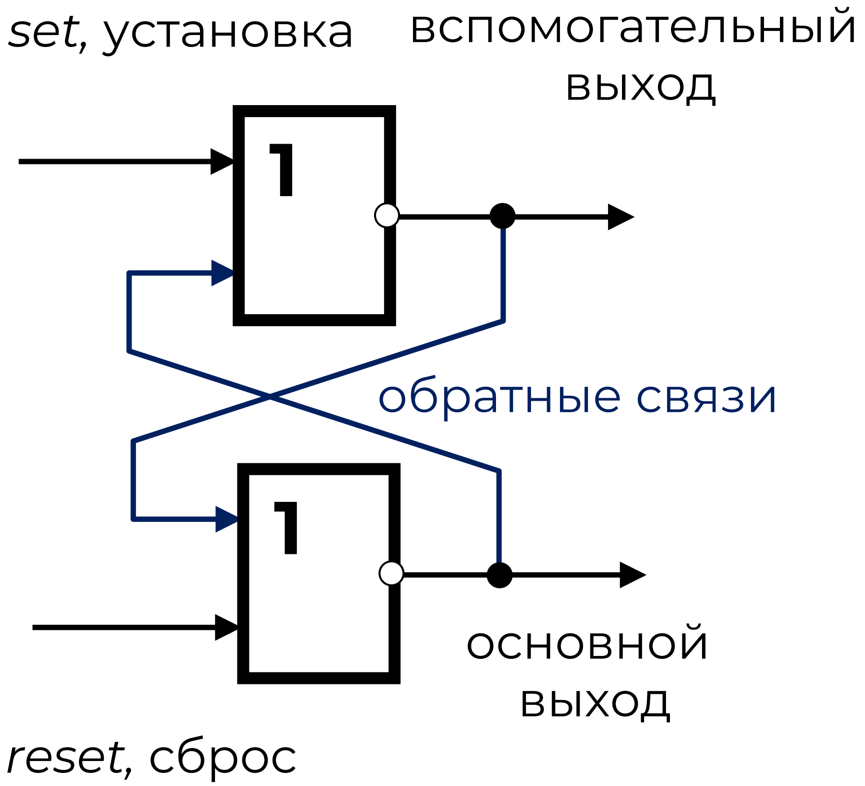



S(set) - установка триггера

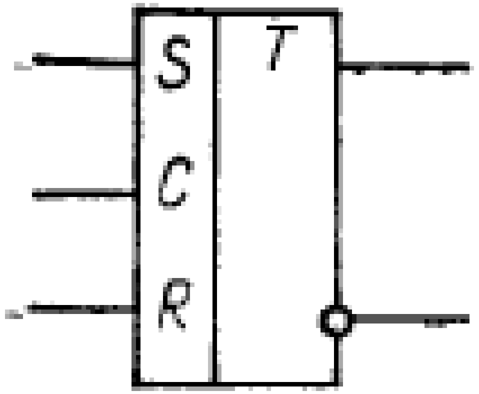
R(reset) - сброс

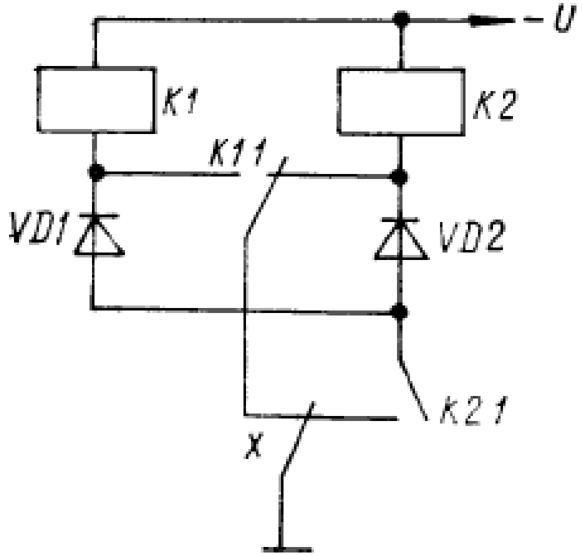
# Триггер

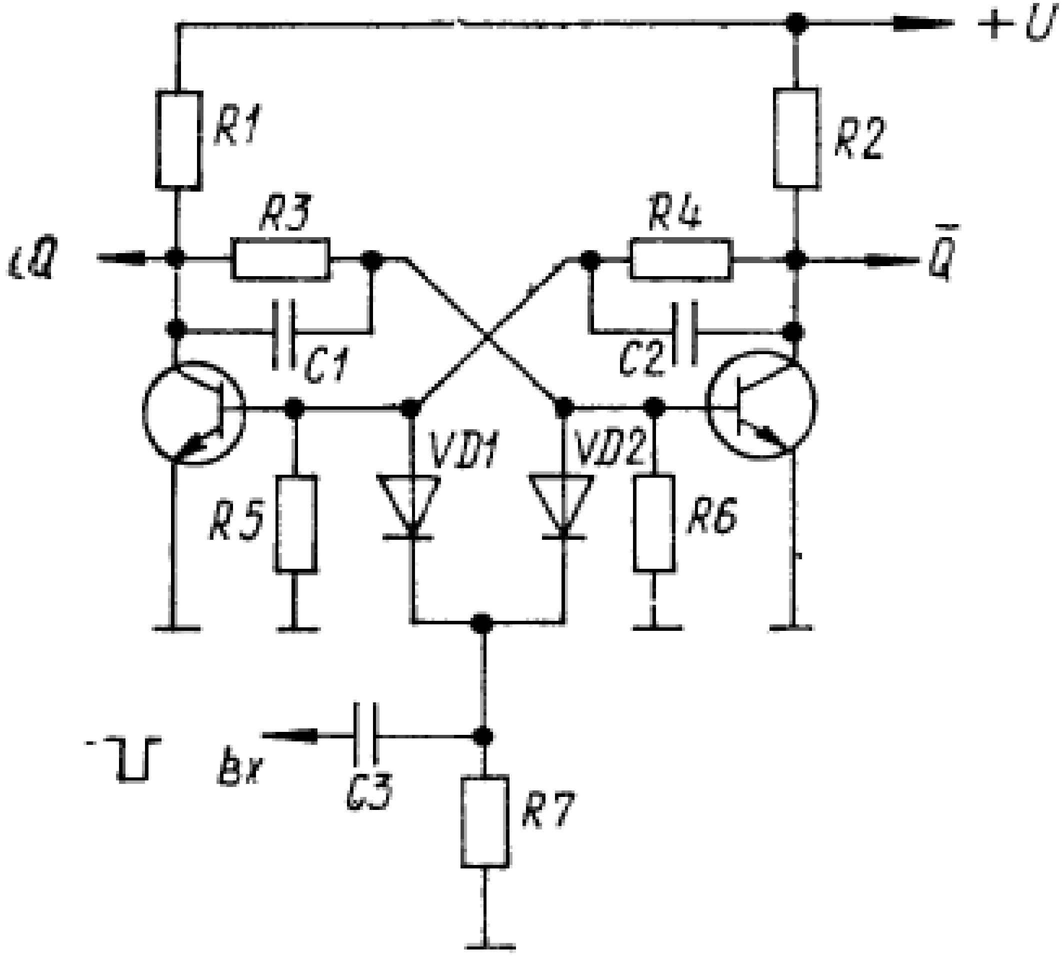
- **Триггер** – это логическая схема, способная хранить 1 бит информации (1 или 0). Строится на 2-х элементах ИЛИ-НЕ или на 2-х элементах И-НЕ.



# Триггер


S
C
R
T


-U
K1
K2
K11
VD1
VD2
K21
x


+U
R1
R2
R3
R4
Q
Q̄
C1
C2
VD1
VD2
R5
R6
Bx
C3
R7

# Сумматор двоичных чисел

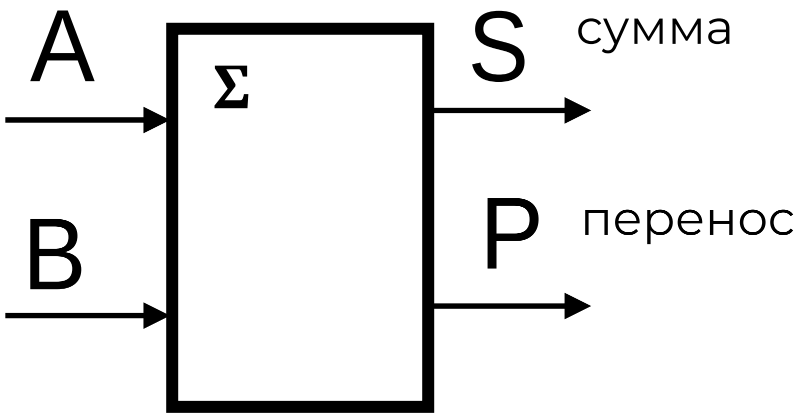
В целях максимального упрощения работы компьютера все многообразие математических операций в процессоре сводится к сложению двоичных чисел.

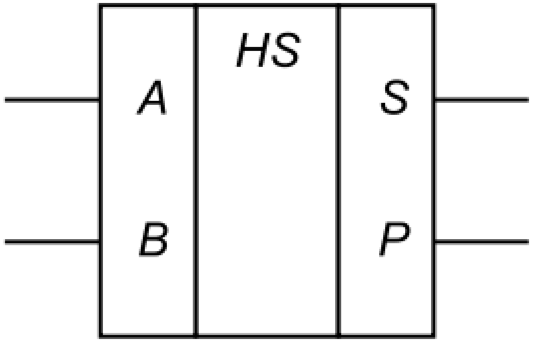
Поэтому главной частью процессора является сумматор, который как раз и обеспечивает такое сложение.

# Полусумматор

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

| A | B | P | S |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 |



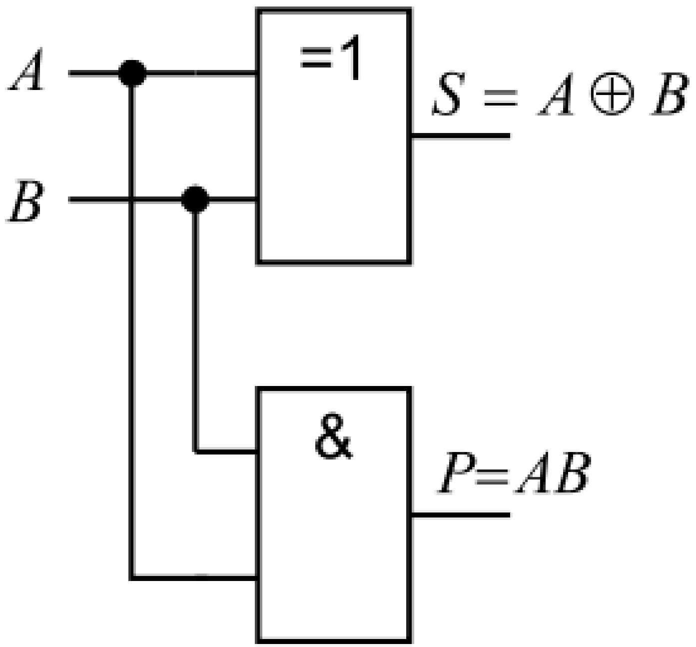



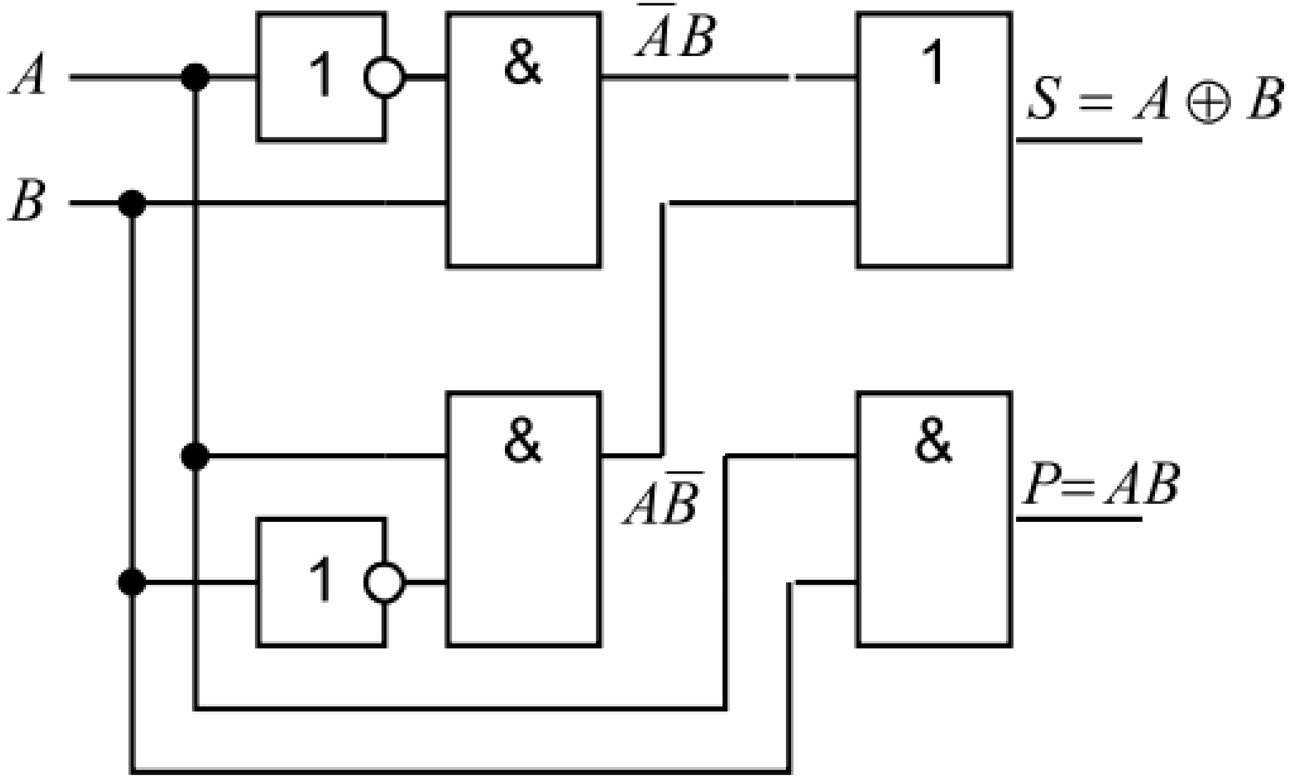

Булевы функции, описывающие работу полусумматора, имеют вид

$$S = \overline{A}B + A\overline{B} = A \oplus B \qquad P = AB.$$

# Полусумматор

- Логическая структура полусумматора в общем и развернутом видах

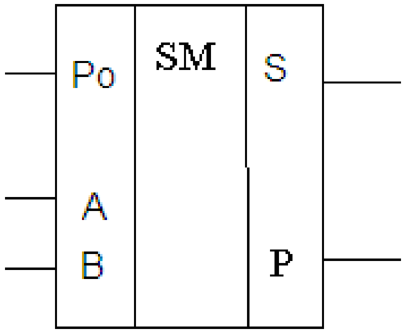



Полусумматор имеет два входа и поэтому пригоден для использования только в младшем разряде многоразрядных двоичных чисел. Начиная со второго разряда многоразрядных чисел, необходимо использовать полный одноразрядный сумматор, содержащий три входа, на один из которых подается сигнал переноса из предыдущего разряда.

# Сумматор

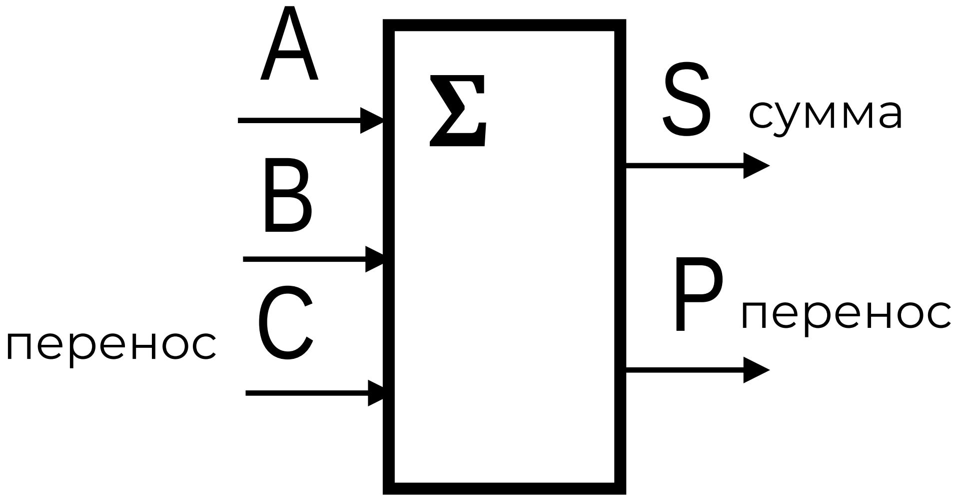
- **Сумматор - это электронная логическая схема, выполняющая суммирование двоичных чисел поразрядным сложением.** Сумматор является центральным узлом арифметико-логического устройства процессора. Находит он применение и в других устройствах компьютера.
- В реальных электронных схемах сумматор изображается так:




- Сумматор выполняет сложение многозначных двоичных чисел. Он представляет собой последовательное соединение одноразрядных двоичных сумматоров, каждый из которых осуществляет сложение в одном разряде. Если при этом возникает переполнение разряда, то перенос суммируется с содержимым старшего соседнего разряда.

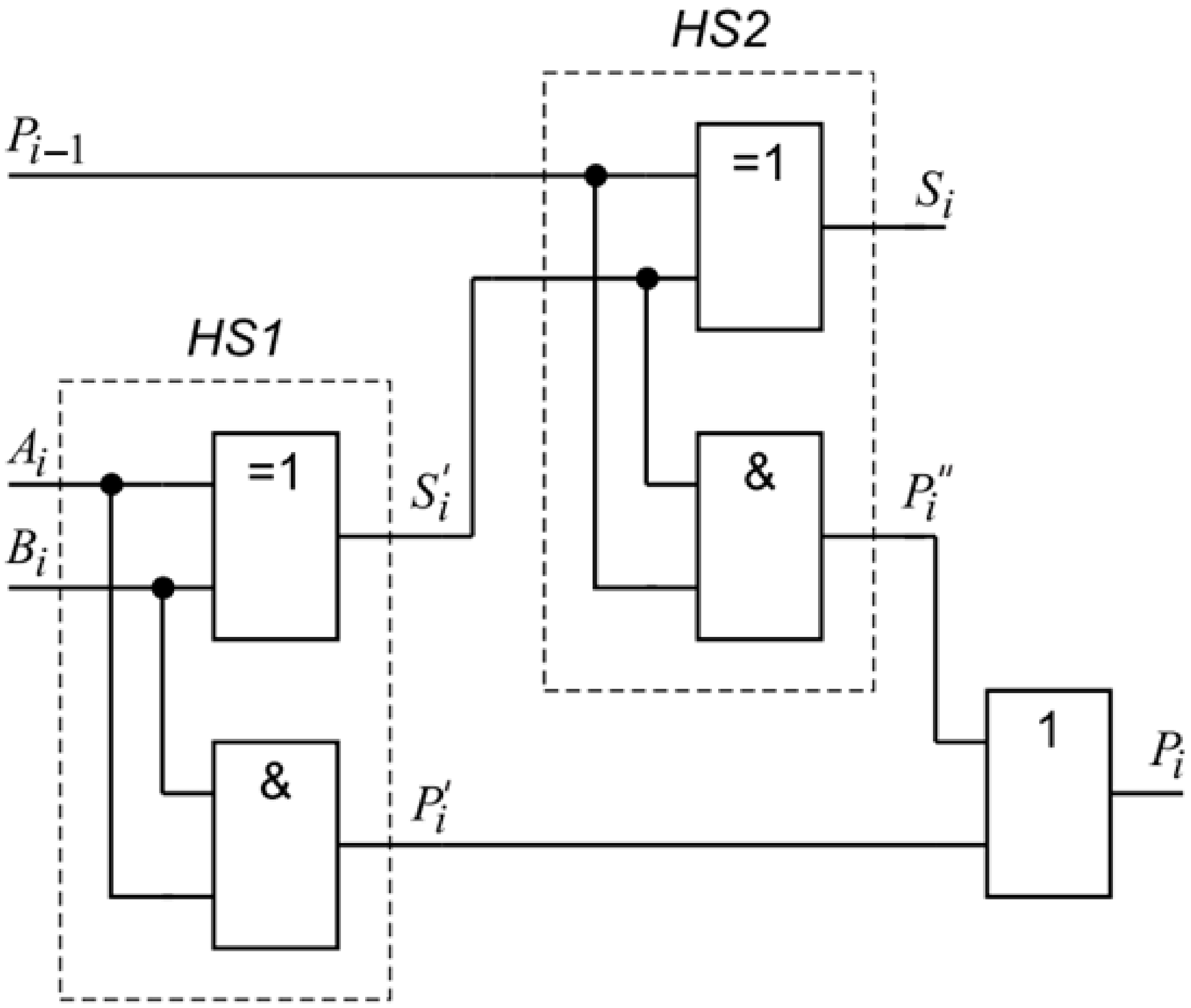
# Сумматор

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

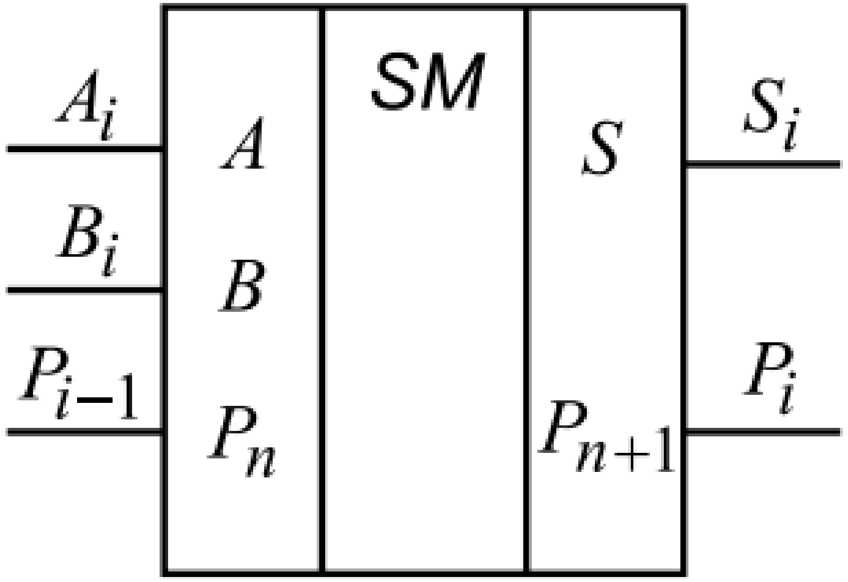



| A | B | C | P | S |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |

# Полный одноразрядный сумматор




реализация на полусумматорах




условное графическое обозначение

# Шифратор и дешифратор

- Шифратор и дешифратор являются типовыми узлами ЭВМ.

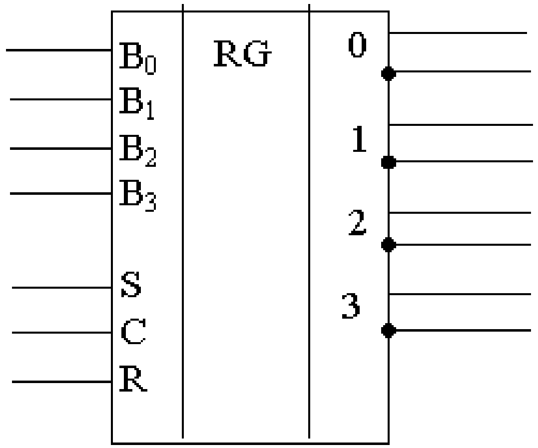
- **Шифратор (кодер)** - это логическое устройство, которое преобразует единичный сигнал на одном из входов в n-разрядный двоичный код. Наибольшее применение он находит в устройствах ввода информации (например в клавиатуре), для преобразования десятичных чисел в двоичную систему счисления.

- **Дешифратор (декодер)** - это логическое устройство, преобразующее двоичный код, поступающий на его входы, в сигнал только на одном из его выходов. Дешифраторы широко применяются в устройствах управления, в системах цифровой индикации с газоразрядными индикаторами, для построения распределителей импульсов по различным цепям и т.д. Схема используется для перевода двоичных цифр в десятичные. Дешифратор двоичного n-разрядного кода имеет 2n выходов, т.к. каждому из 2n значений входного кода должен соответствовать единичный сигнал на одном из выходов дешифратора.

# Регистры

- Функциональная схема компьютера, состоящая из триггеров, предназначенная для запоминания многоразрядных кодов и выполнения над ними некоторых логических преобразований называется регистром.
- Упрощенно регистр можно представить как совокупность ячеек, в каждой из которых может быть записано одно из двух значений: 0 или 1, то есть один разряд двоичного числа.

- С помощью регистров можно выполнять следующие операции: установку, сдвиг, преобразование. Основными типами регистров являются параллельные и последовательные (сдвигающие).
- Совокупность регистров, используемых ЭВМ для запоминания программы работы, исходных и промежуточных результатов называется оперативной памятью (ОП).
- Регистры содержатся в различных вычислительных узлах компьютера - процессоре, периферийных устройствах и т.д.



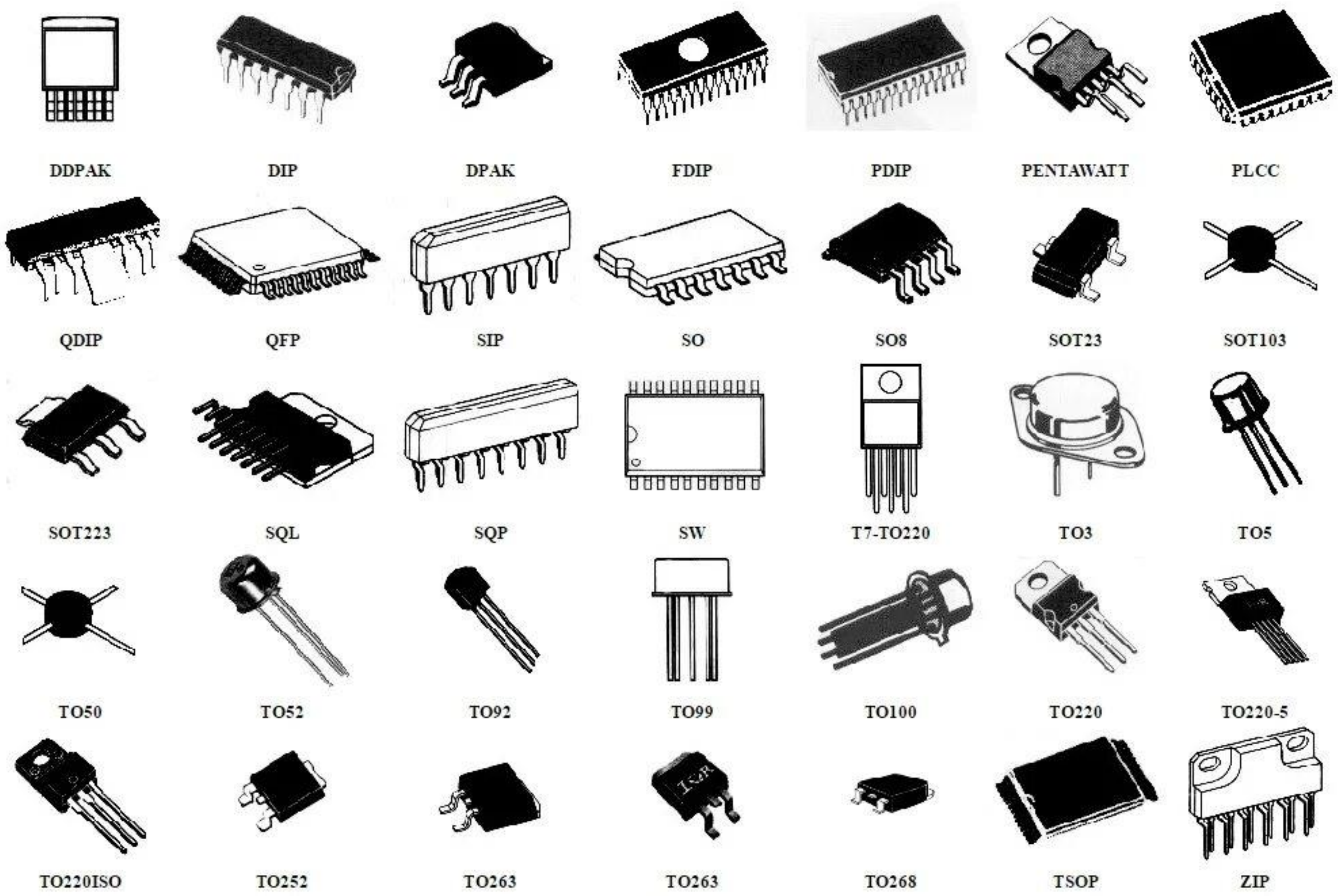

- **Регистр** - это устройство, предназначенное для хранения многоразрядного двоичного числового кода, которым можно представлять и адрес, и команду, и данные.

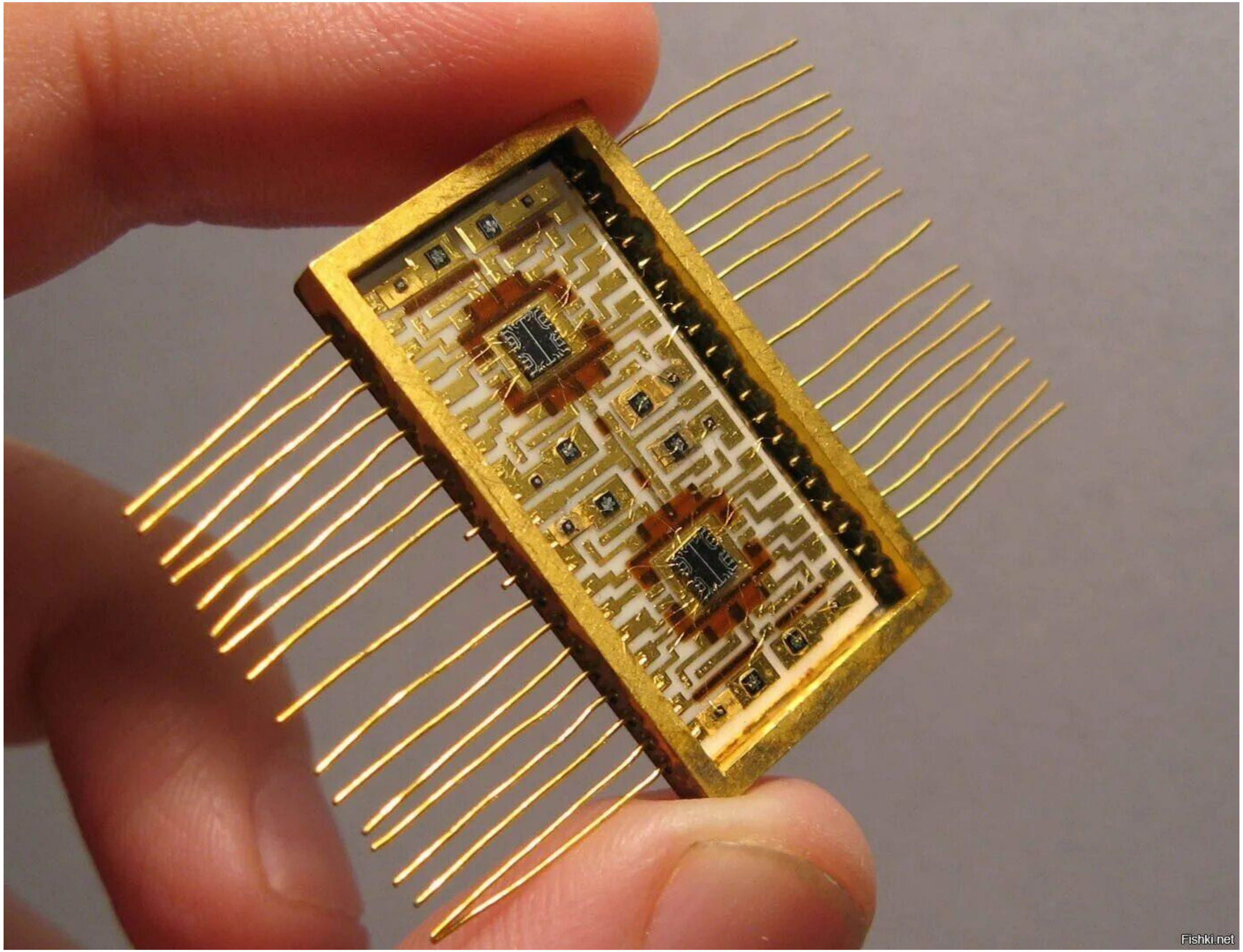
# Регистры, Счетчики

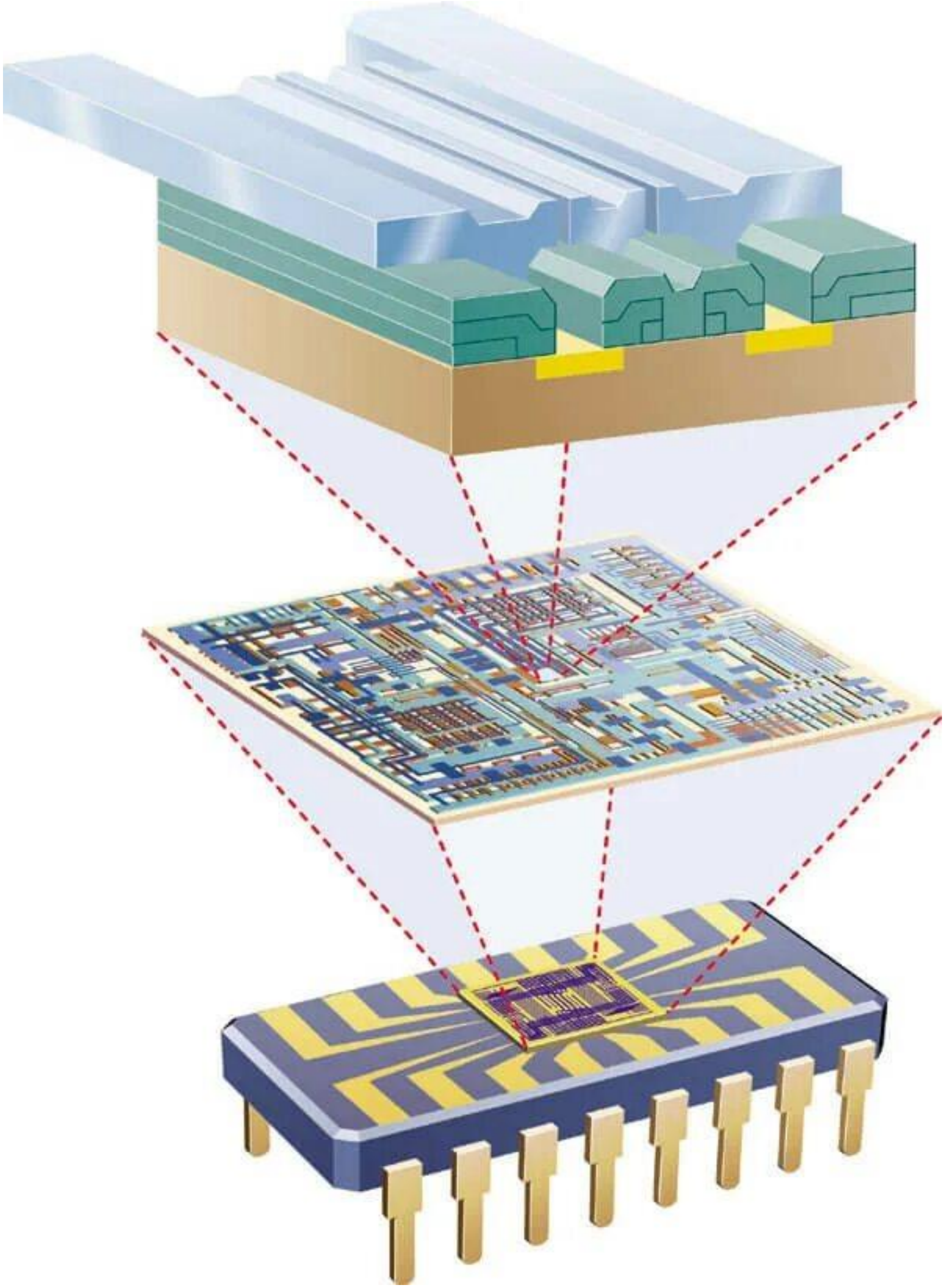
- Существует несколько типов регистров, отличающихся видом выполняемых операций.
- Некоторые важные регистры имеют свои названия, например:
  - **сдвиговый регистр** - предназначен для выполнения операции сдвига;
  - **счетчики** - **схемы, способные считать поступающие на вход импульсы**. К ним относятся Т-триггеры (название от англ. tumble - опрокидываться). Этот триггер имеет один счетный вход и два выхода. Под действием сигналов триггер меняет свое состояние с нулевого на единичное и наоборот. Число перебрасываний соответствует числу поступивших сигналов;
  - **счетчик команд** - регистр устройства управления процессора (УУ), содержимое которого соответствует адресу очередной выполняемой команды; служит для автоматической выборки программы из последовательных ячеек памяти;
  - **регистр команд** - регистр УУ для хранения кода команды на период времени, необходимый для ее выполнения. Часть его разрядов используется для хранения кода операции, остальные - для хранения кодов адресов операндов.
- В ЭВМ применяются регистры 8, 16, 32, 48 и 64 разрядов.

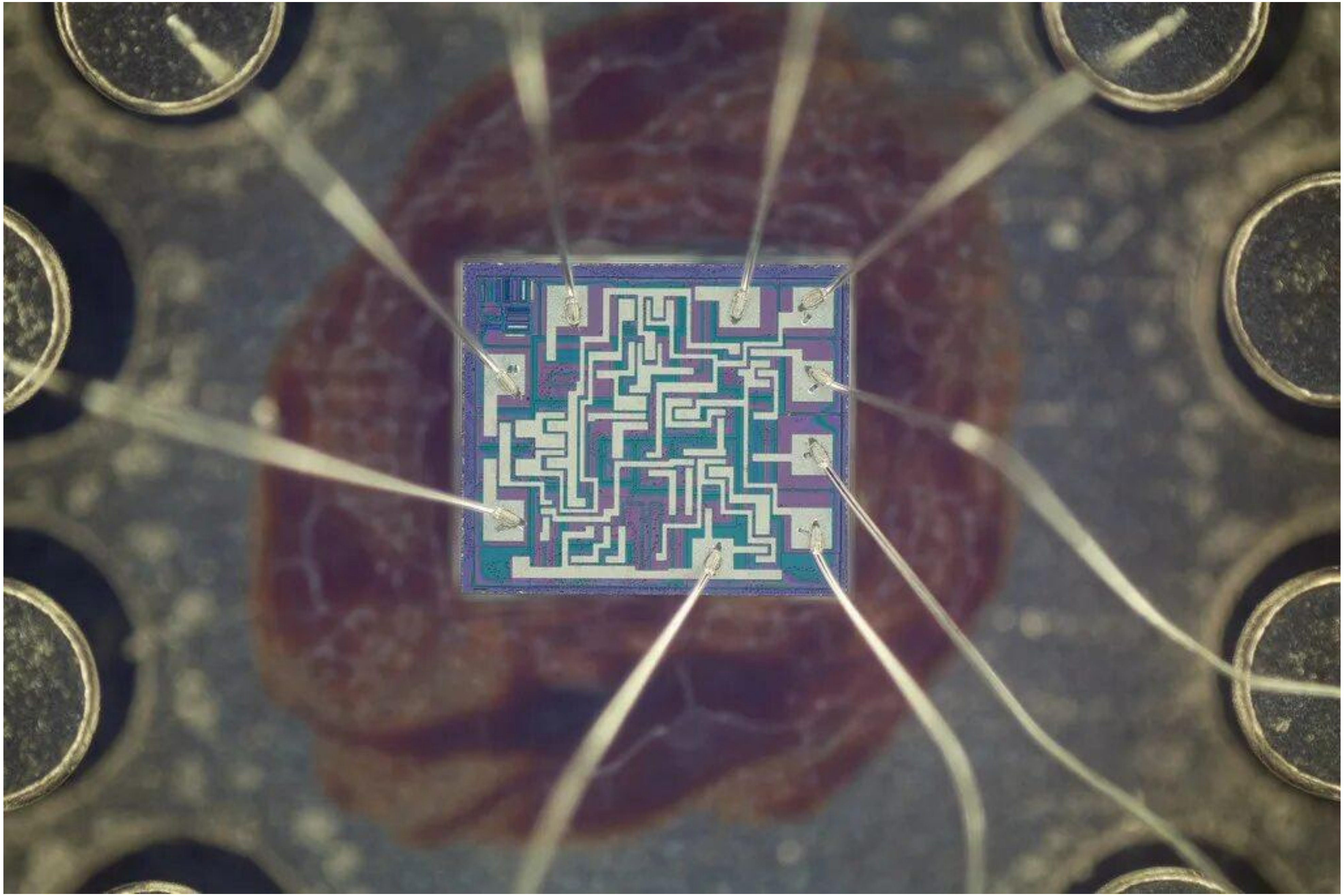

КР1008ВЖ7
9007
КР587ИК3
КР514КТ1
8708
К561ИЕ9
0483
Р537РУ2А
0683
К145ИК8П
9109
DV74HC14AN

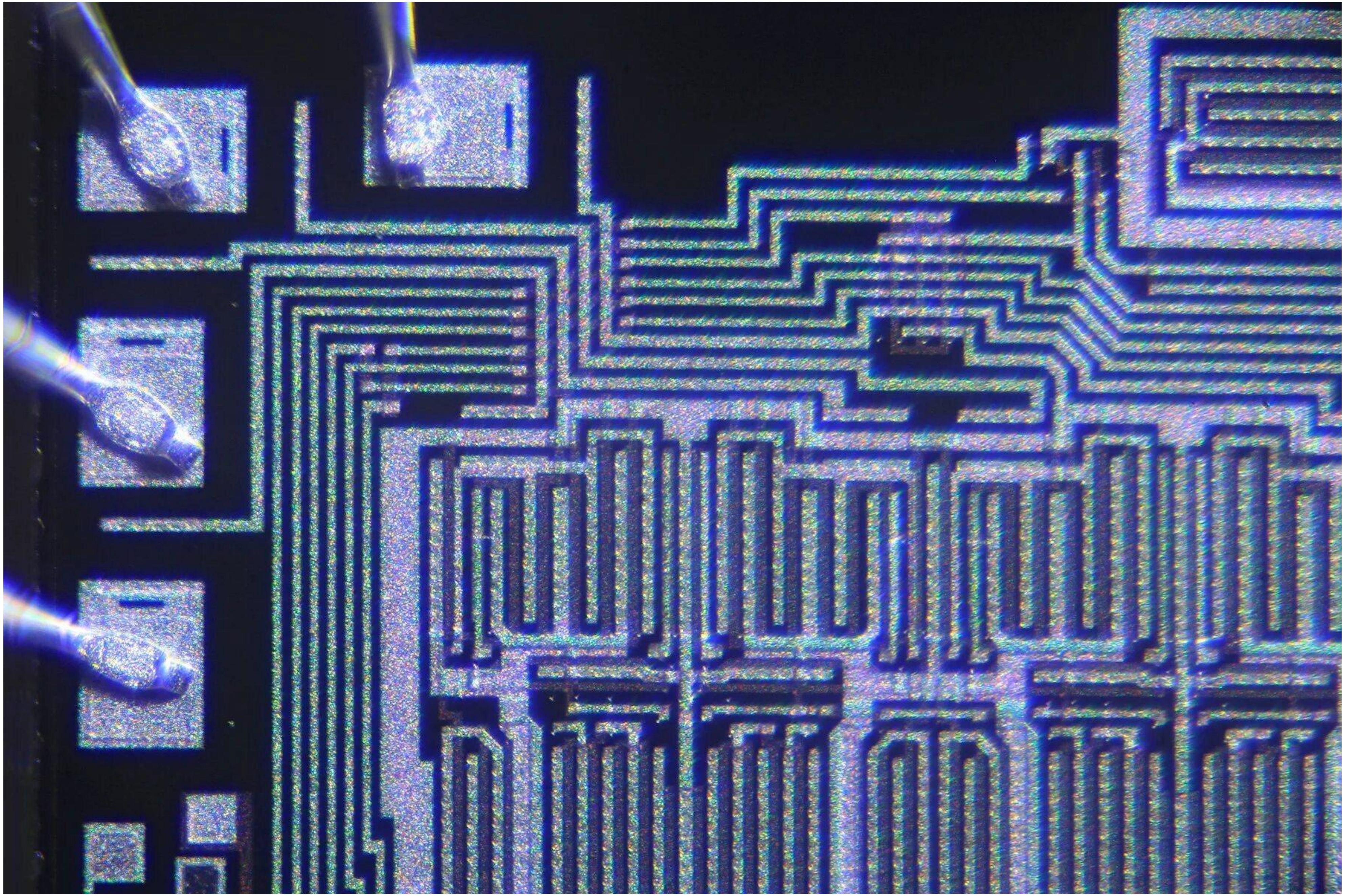
# Микросхемы - типы корпусов

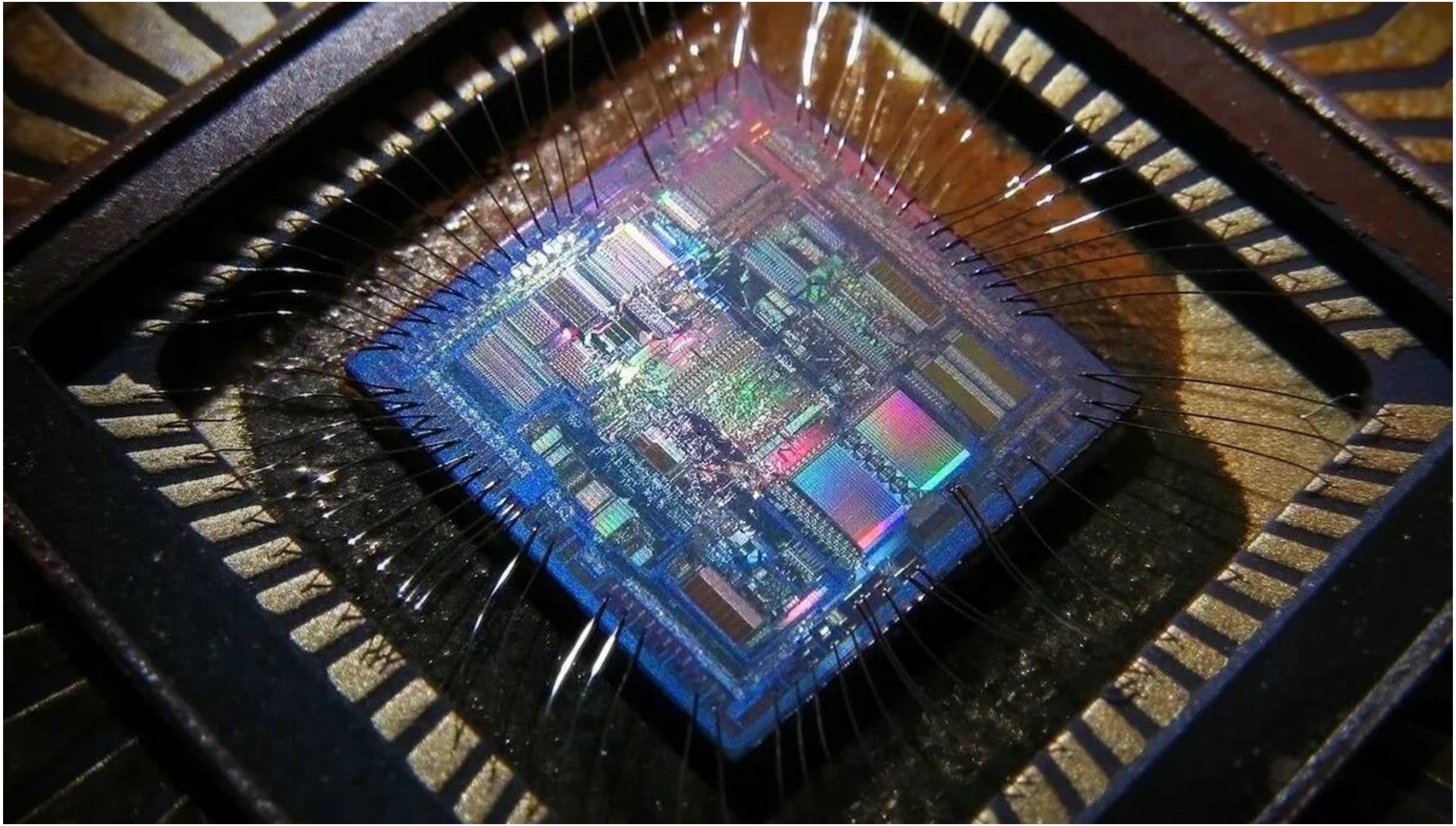


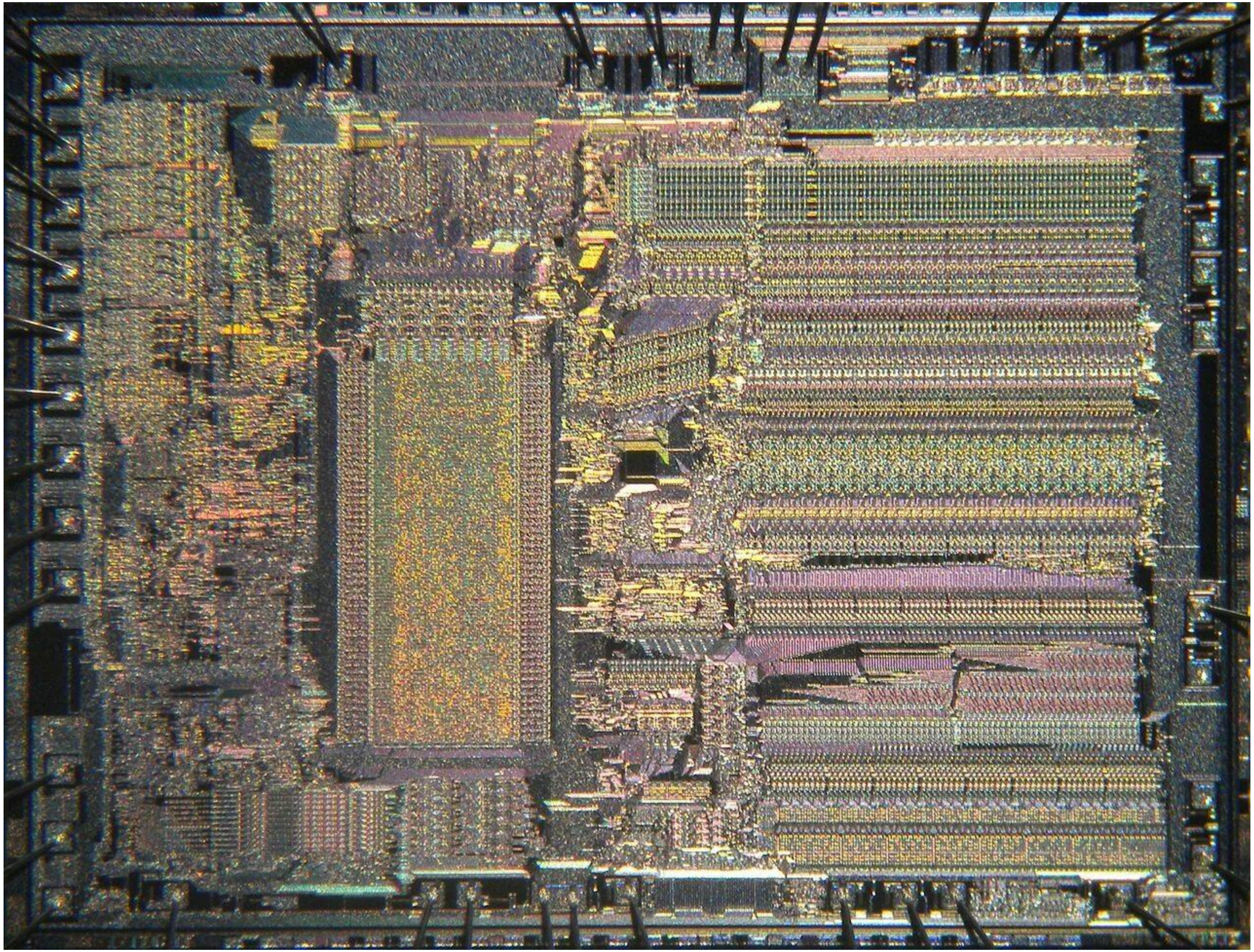

Fishki.net

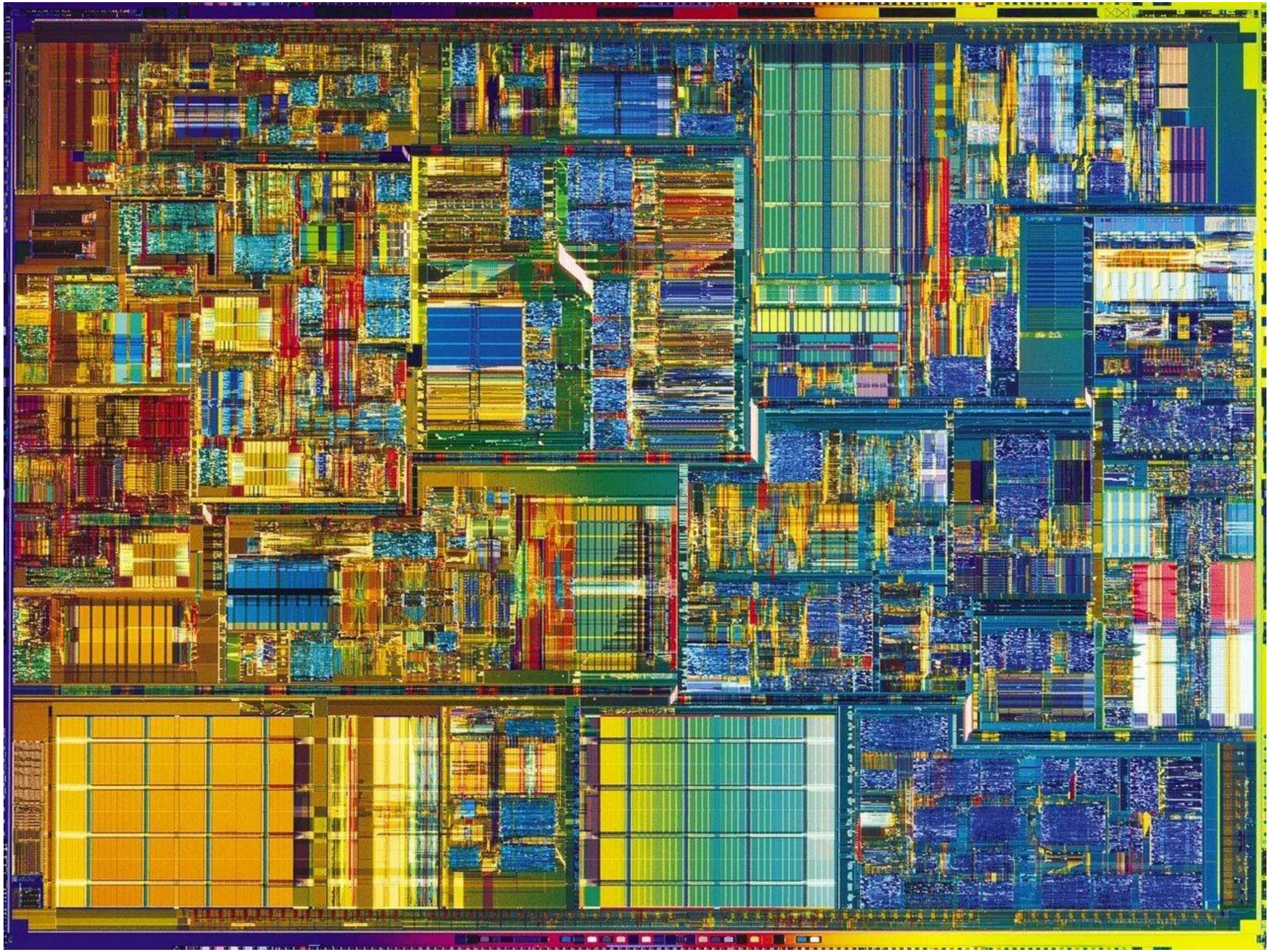


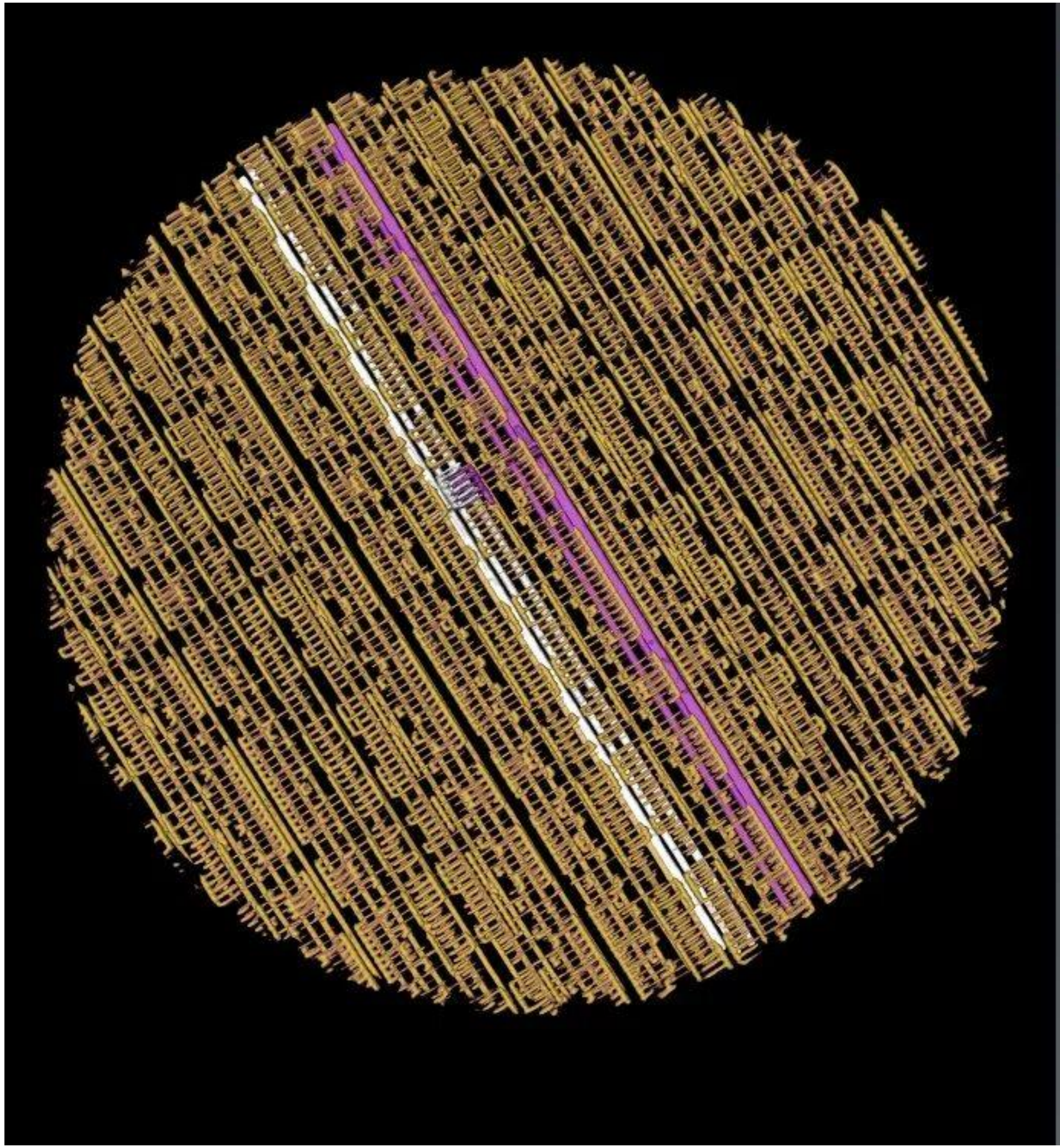


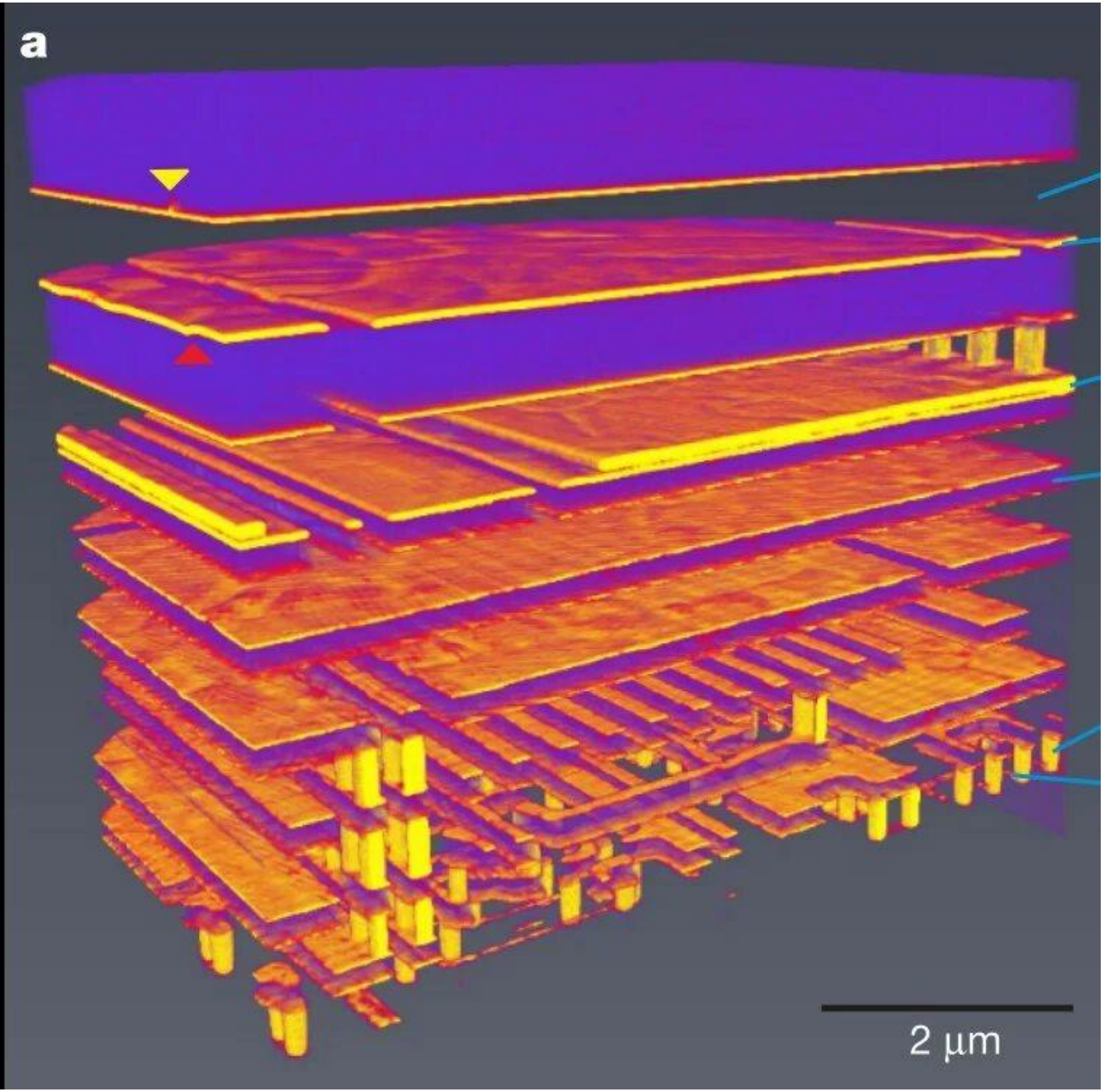


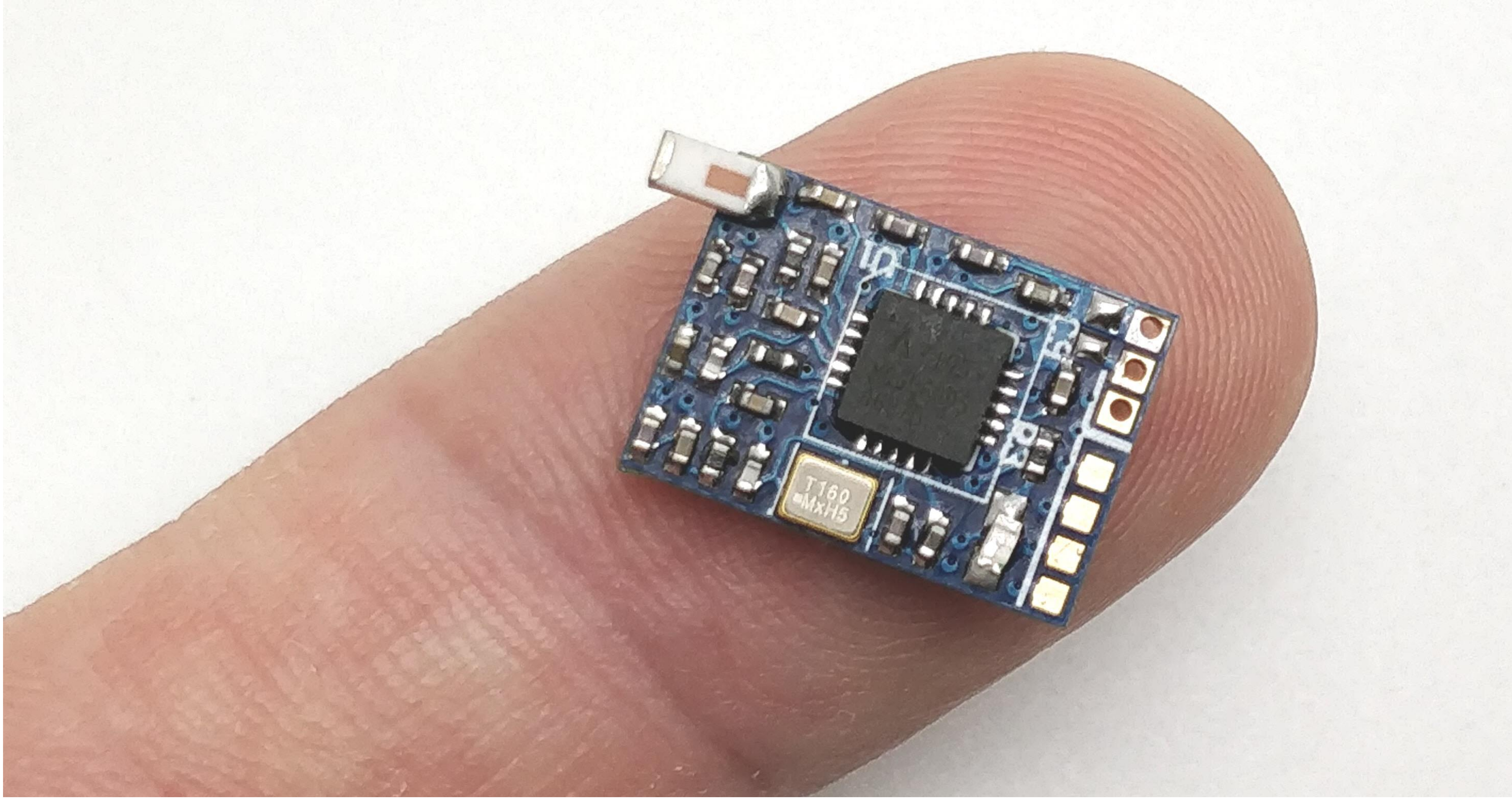


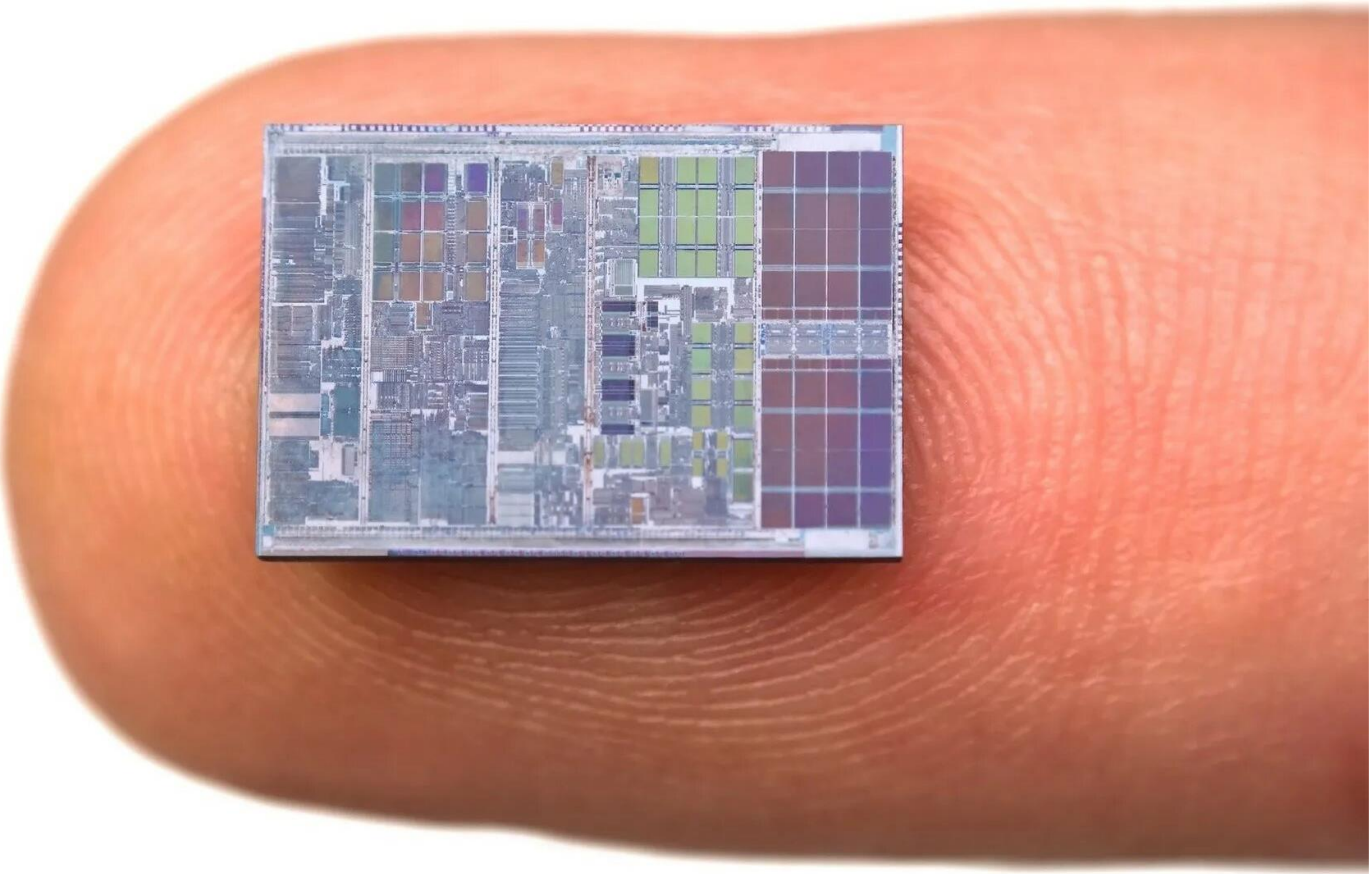







a
2 μm


T160
MxH5



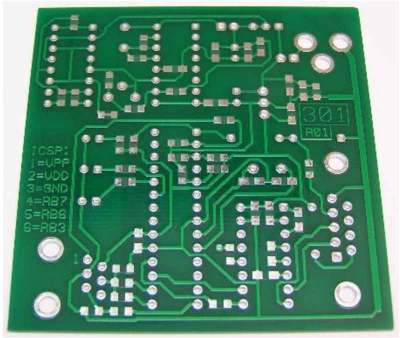
# Этапы развития вычислительной техники

## I поколение ЭВМ


электронно-вакуумные лампы

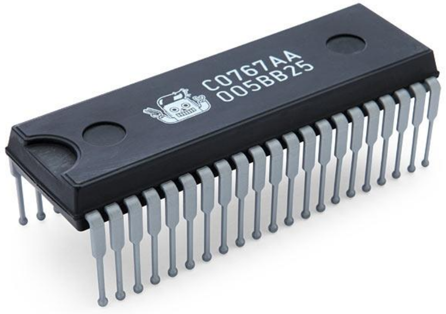
## II поколение ЭВМ

транзисторы, диоды, резисторы



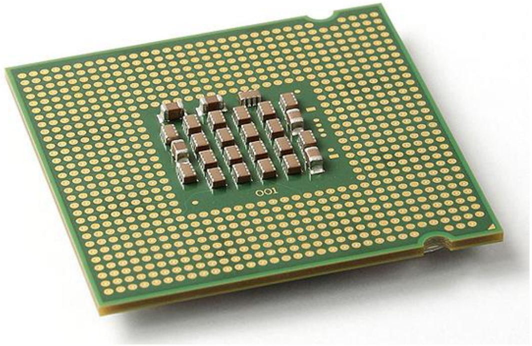
печатные платы

## III поколение ЭВМ

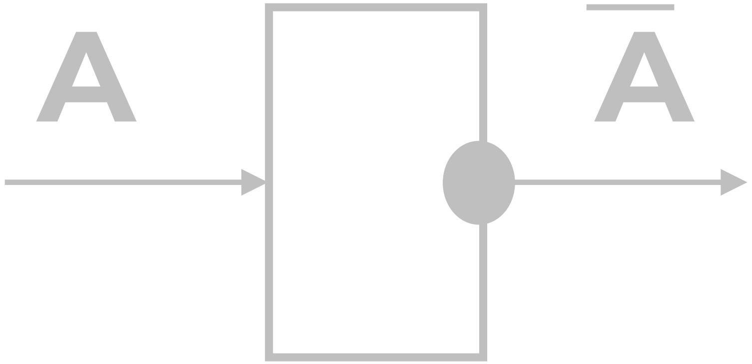



интегральные схемы

## IV поколение ЭВМ



Процессоры (БИС)




# Дополнительные примеры с решениями

# Пример

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

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

# Пример

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

равносильны

| | A | B | $A+B$ | $\overline{A}+\overline{B}$ | X |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | 1 | 1 |
| 2 | 1 | 0 | 1 | 1 | 1 |
| 3 | 1 | 1 | 1 | 0 | 0 |

| $A \cdot \overline{B} + \overline{A} \cdot B$ |
|---|
| 0 |
| 1 |
| 1 |
| 0 |

# Пример

Составить таблицу истинности для формулы $\overline{x} \vee \overline{y}$.

Таблица истинности будет иметь следующий вид:

| $x$ | $y$ | $\overline{x}$ | $\overline{y}$ | $\overline{x} \vee \overline{y}$ |
|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 |

# Пример

***Задача*** Найти формулу, которая определяет функцию $f(x,y)$ по следующей таблице истинности:

| $x$ | $y$ | $f(x,y)$ |
|---|---|---|
| 1 | 1 | 0 |
| 1 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 0 | 0 |

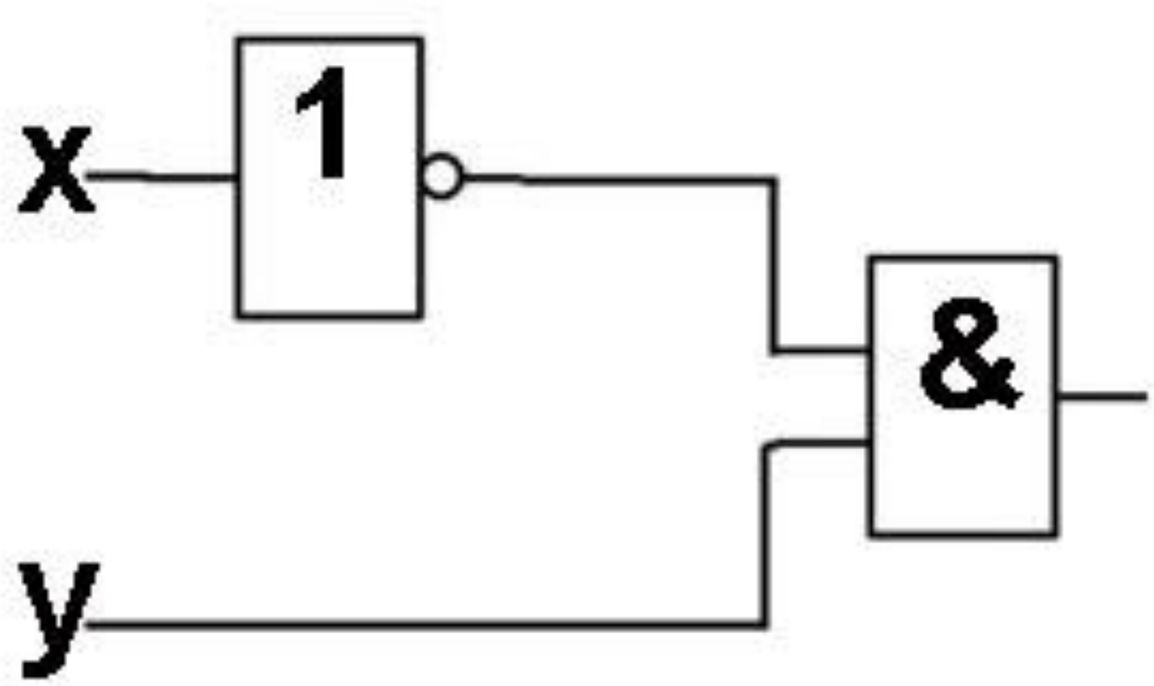
***Решение.*** Воспользуемся правилом получения формулы алгебры логики из таблицы истинности для функции $f(x,y)$. Получим:

$$f(x,y) \equiv x \,\&\, y \vee \bar{x} \,\&\, y .$$

Упростим полученную формулу:

$$x \,\&\, y \vee \bar{x} \,\&\, y \equiv (x \vee \bar{x}) \,\&\, y \equiv 1 \,\&\, y \equiv y .$$

# Пример № 1




Найдем булеву функцию логической схемы и составим таблицу истинности для логической схемы.

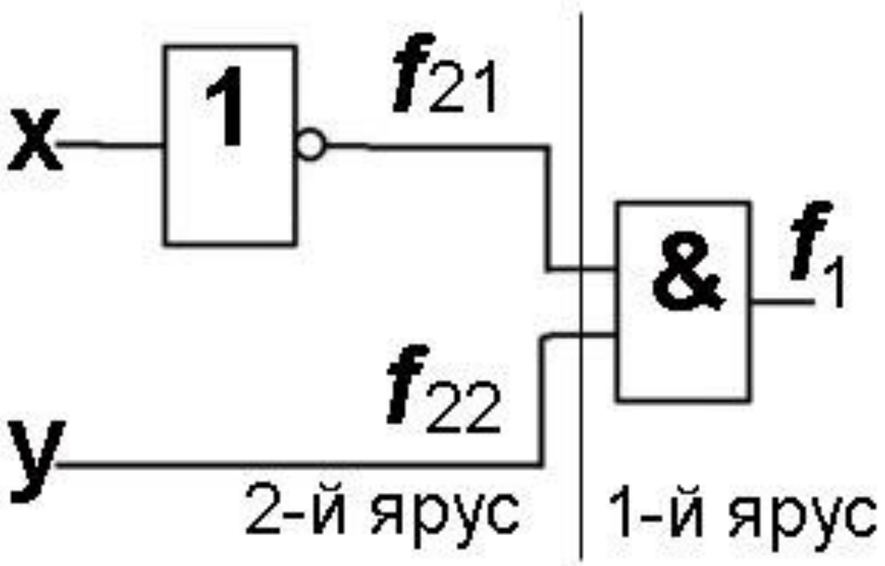
# Пример № 1. Решение

Решение. Разбиваем логическую схему на ярусы. Запишем все функции, начиная с 1-го яруса:

$$f_1 = f_{21} \wedge f_{22}$$

$$f_{21} = \bar{x}$$

$$f_{22} = y$$



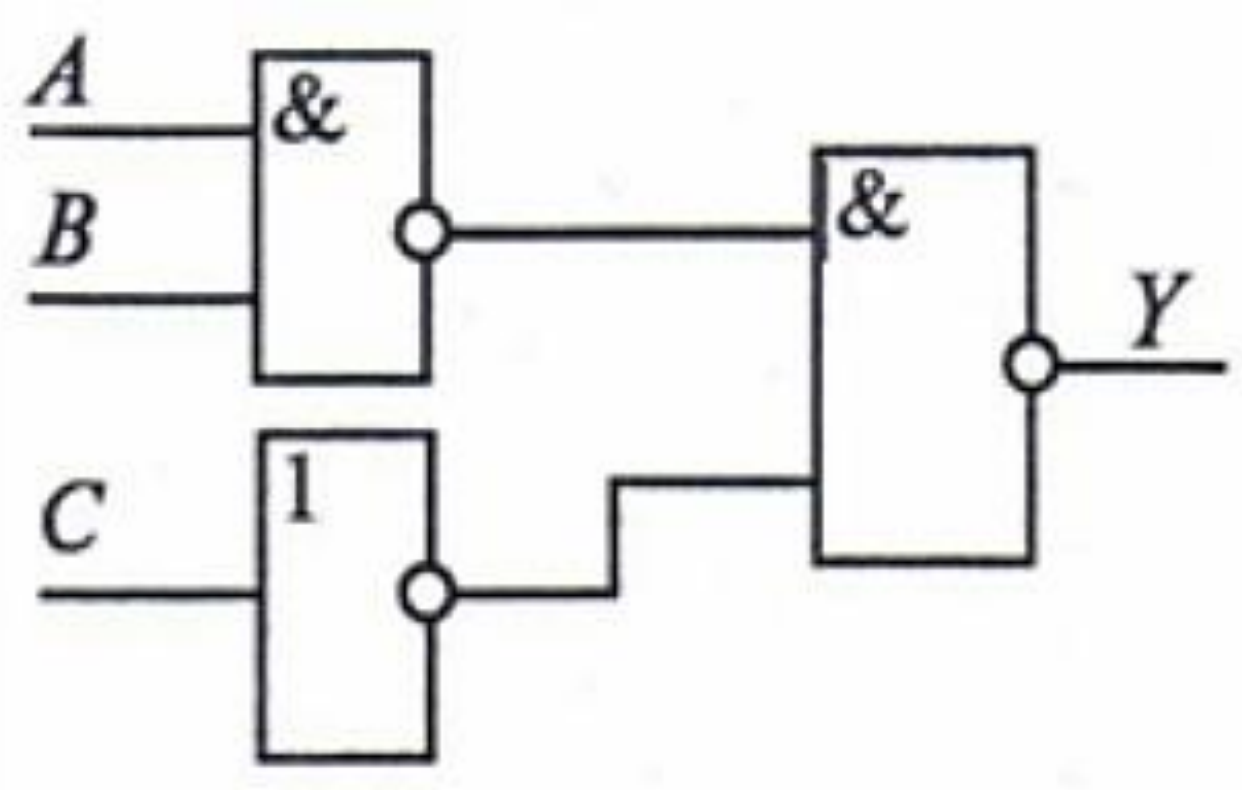

| x | y | $f_{21}$ | $f_{22}$ | f |
|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 |

Получаем функцию, которую реализует на выходе логическая схема:

$$f_1 = \bar{x} \wedge y$$

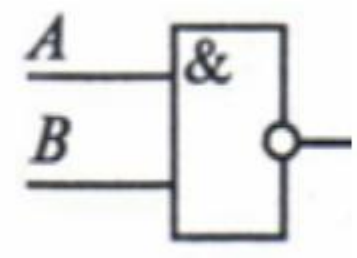
# Пример № 2

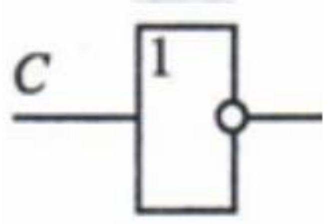
- По заданной схеме требуется определить функцию Y, реализующуюся данной схемой

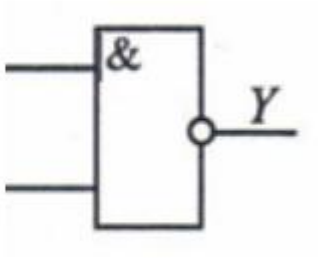


# Пример № 2. Решение

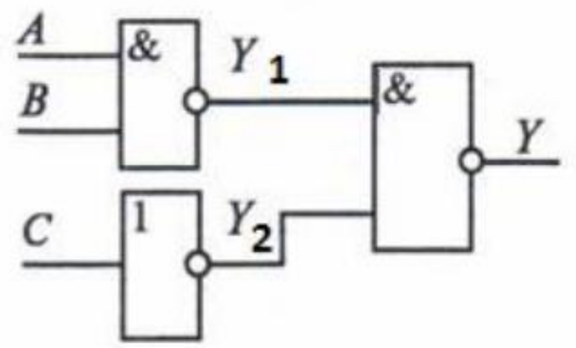
1. Подчитываются количество логических элементов, входящих в схему, в данном случаи их три штуки:



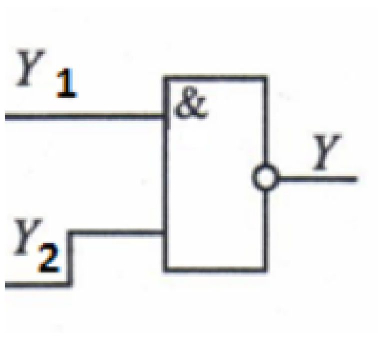
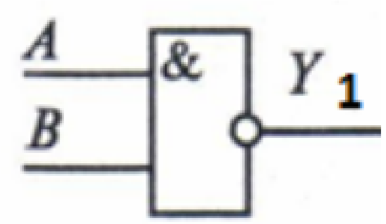
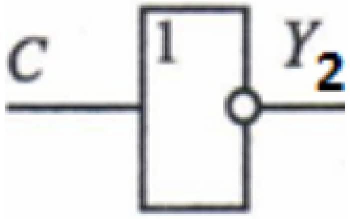




2. Выходы каждого логического элемента обозначаются проиндексированными функциями в порядке, начиная от последнего (обратный порядку приоритета), причем последний логический элемент имеет название конечной функции (Y1, Y2, Y).



# Пример № 2. Решение

3. Заполняем таблицу, по следующему принципу: записываются выходные функции каждого элемента в виде формул в соответствии с введенными обозначениями, в порядке начиная с последнего.

| Логические элементы | Логические функции |
|---|---|
|  | $Y=\neg(Y1\wedge Y2)$ |
|  | $Y1=\neg(A\wedge B)$ |
|  | $Y2=\neg C$ |

# Пример № 2. Решение

4. Производится подстановка одних выходных функций через другие, используя входные переменные и логические функции, полученные в таблице

$$Y = \neg(Y1 \wedge Y2) = \neg(\neg(A \wedge B) \wedge (\neg C))$$

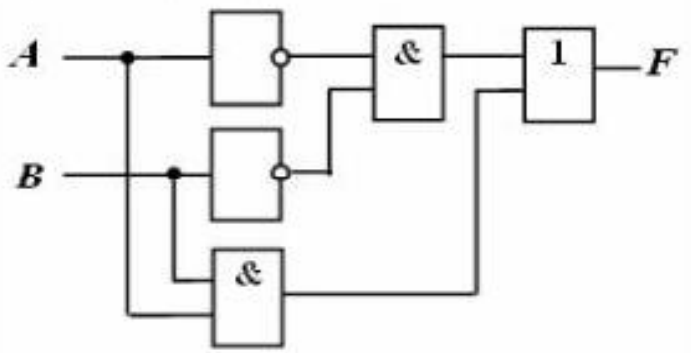
5. Записывается получившаяся булева функция через входные переменные.

$$Y = \neg(\neg(A \wedge B) \wedge (\neg C))$$

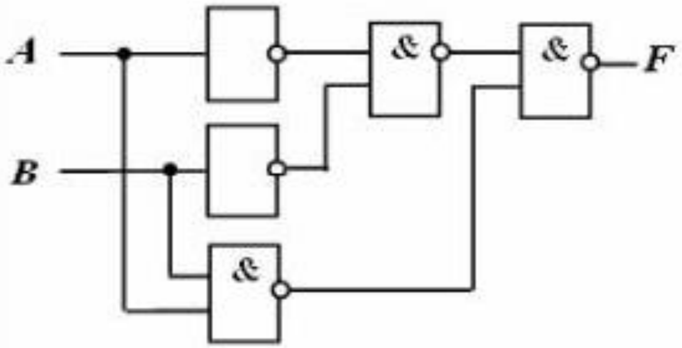
# Пример № 3

**Решение:**

Логической схеме




равносильна схема

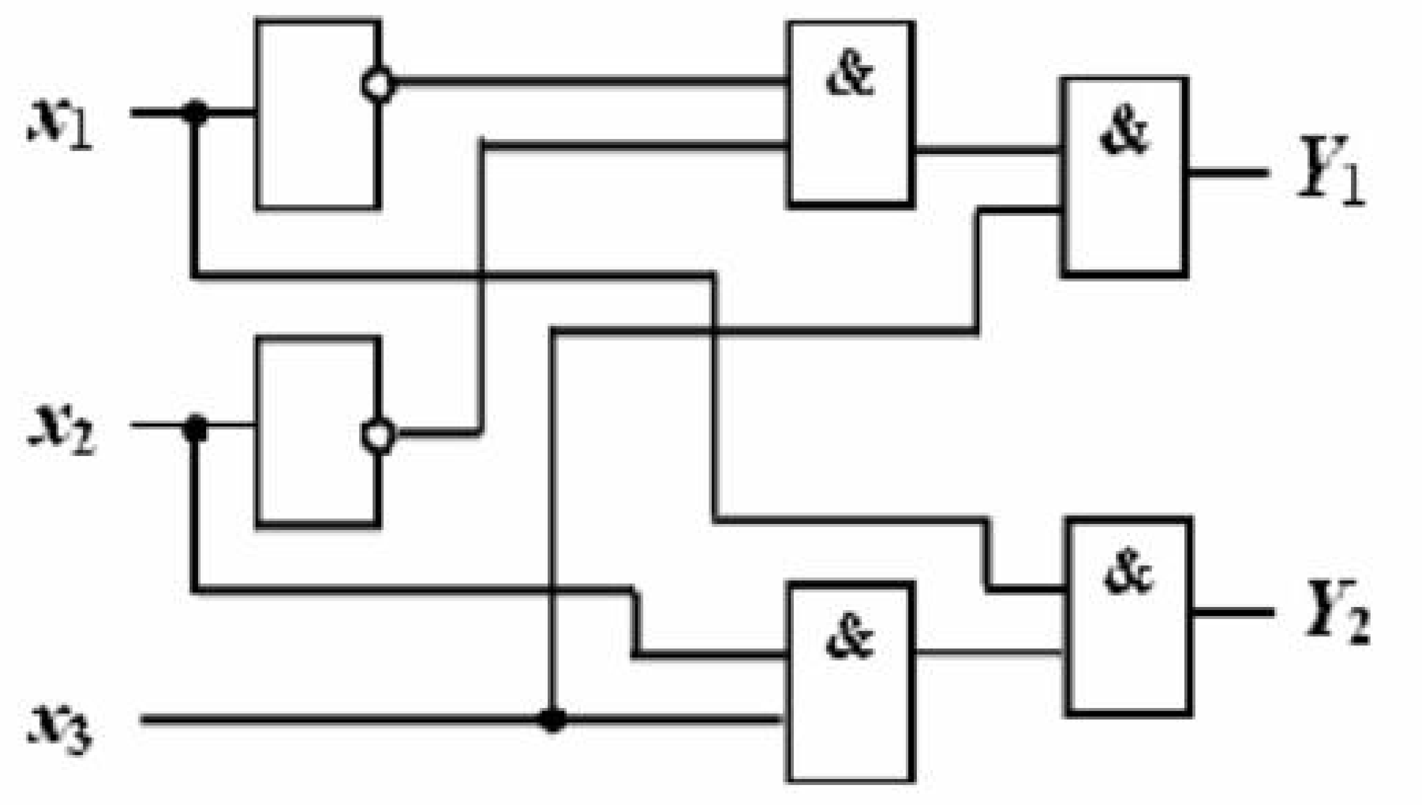



Если сравнить данную схему с исходной, то видно, что логический элемент **ИЛИ**, включенный на выходе исходной схемы, заменен на логический элемент **И-НЕ** с инверсией его входных сигналов в соответствии с законами де Моргана и двойного отрицания $A \vee B = (\overline{\overline{A} \& \overline{B}})$.

Итак, произведена равносильная замена (равносильное преобразование). Это можно проверить с помощью таблиц истинности или логических формул.

# Пример № 4

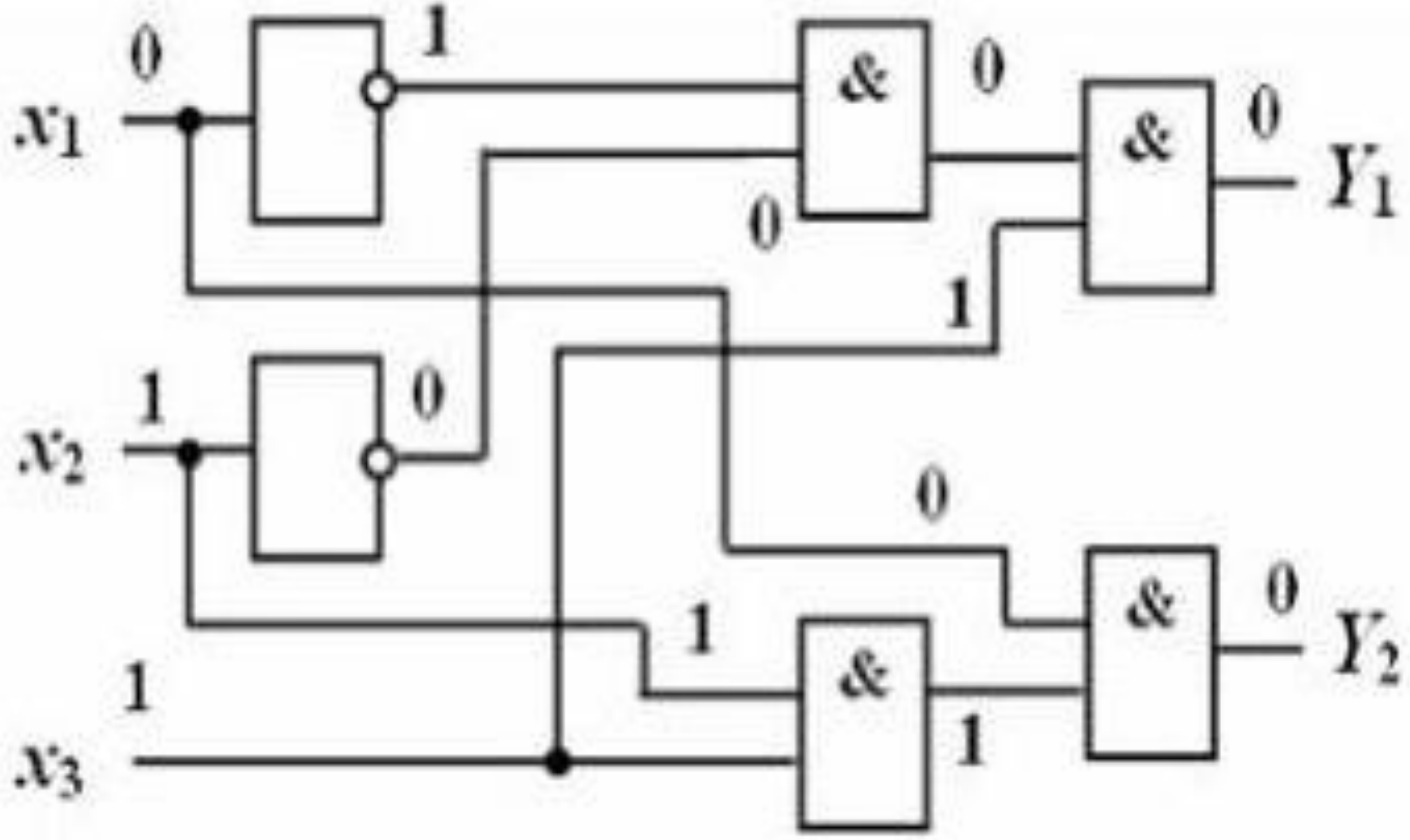
- Если на входы логической схемы




- подана следующая комбинация входных параметров
  $x_1 = 0$, $x_2 = 1$, $x_3 = 1$, то комбинацией значений на выходе будет…

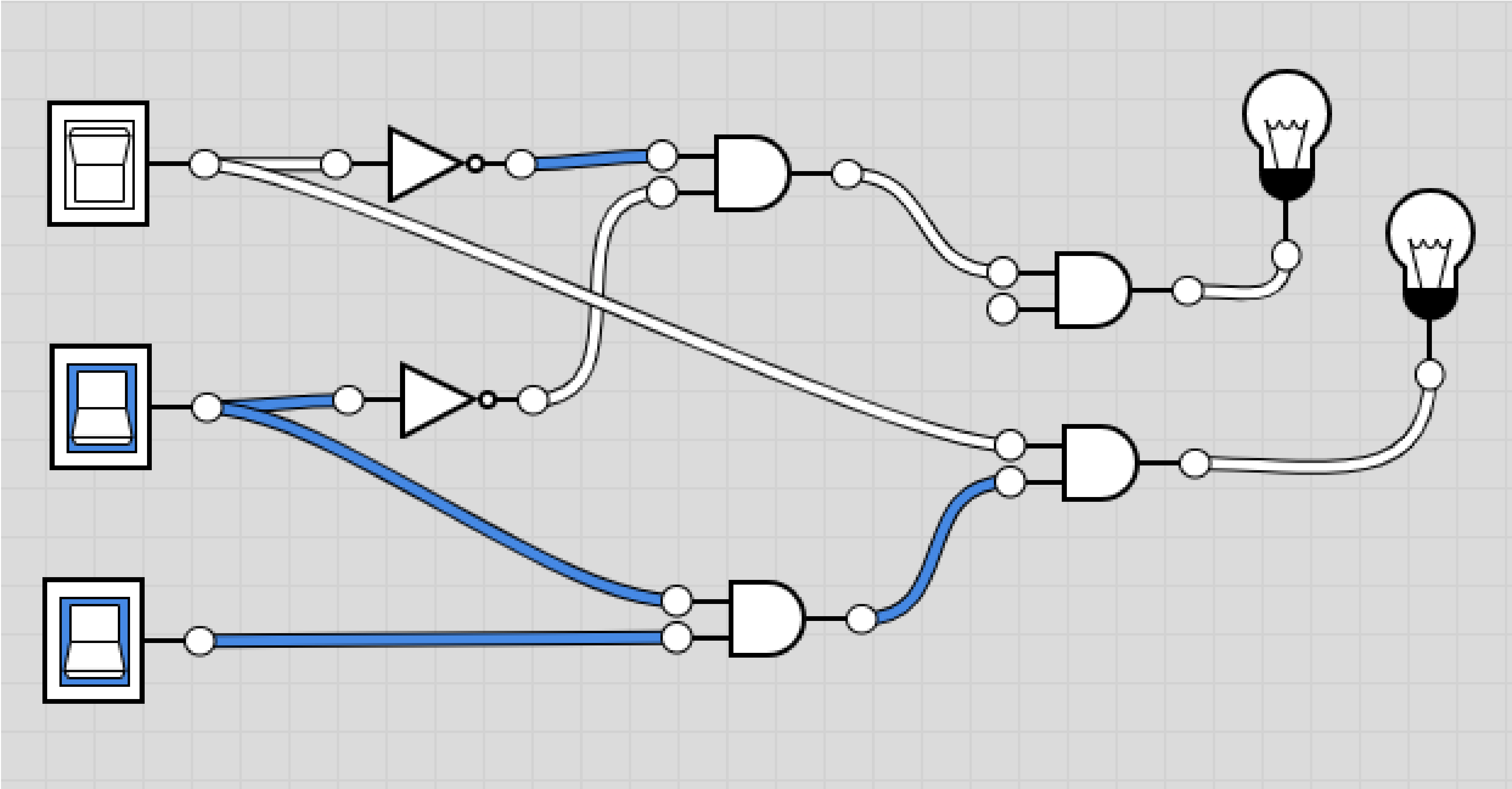
# Пример № 4 Решение

**Решение:**




Правильным решением будет $Y_1 = 0$, $Y_2 = 0$.

# Пример № 4 Решение в logic.ly



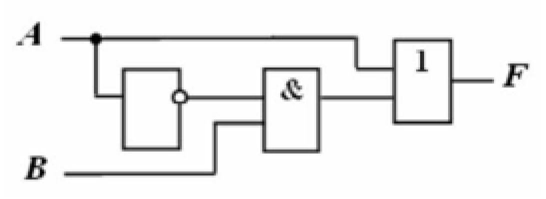
# Пример № 5

- Таблице истинности вида

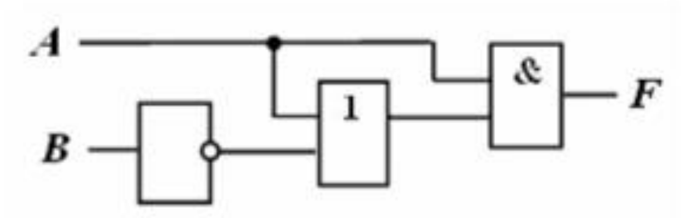
| A | B | F |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

- соответствует логическая схема …

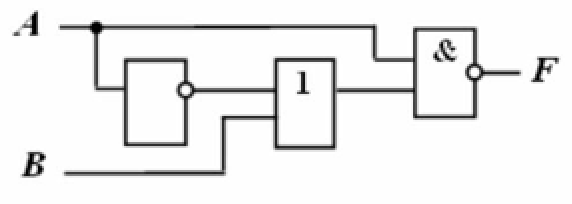
А)



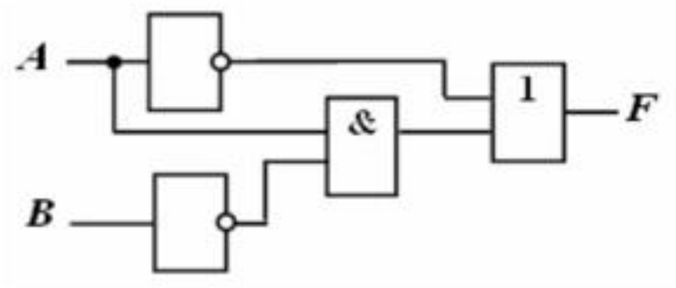

В)




Б)




Г)



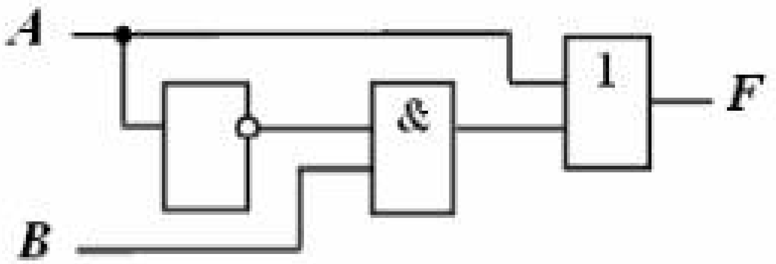
# Пример № 5 Решение

**Решение:**

Таблице истинности вида

| $A$ | $B$ | $F$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

из перечисленных схем соответствует логическая схема:




Логическим выражением, соответствующим данной логической схеме, будет

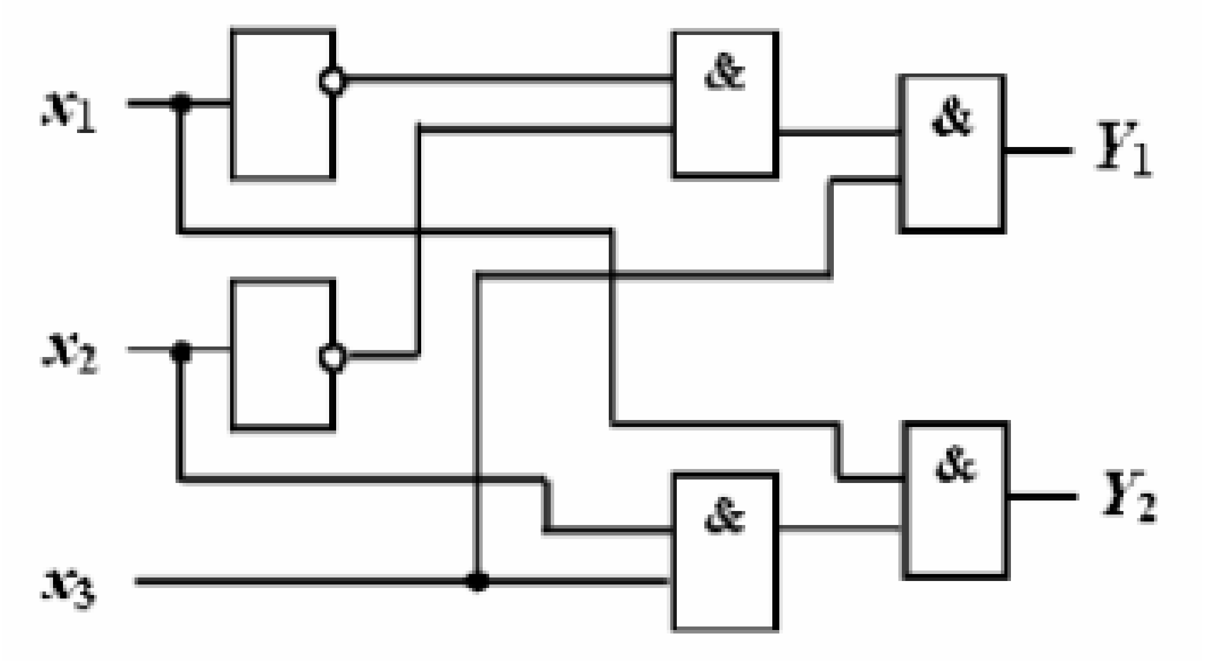
$F(A,B) = (\overline{A} \,\&\, B) \vee A.$

Проверим выполнимость таблицы истинности для полученного выражения и убедимся в соответствии друг другу таблицы истинности и логической схемы.

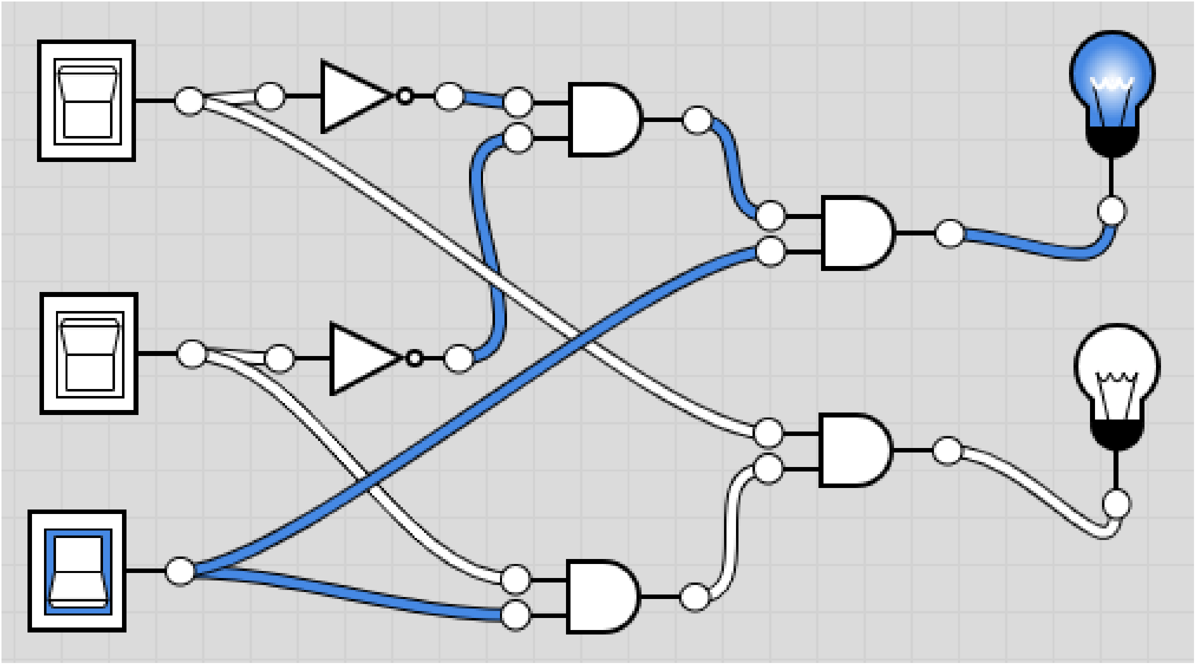
# Задача и решение

- Если на входе логической схемы подана следующая $x_1 = 0$, $x_2 = 0$, $x_3 = 1$, то комбинацией на выходе $Y_1$ и $Y_2$ чему будет равна?

Задача



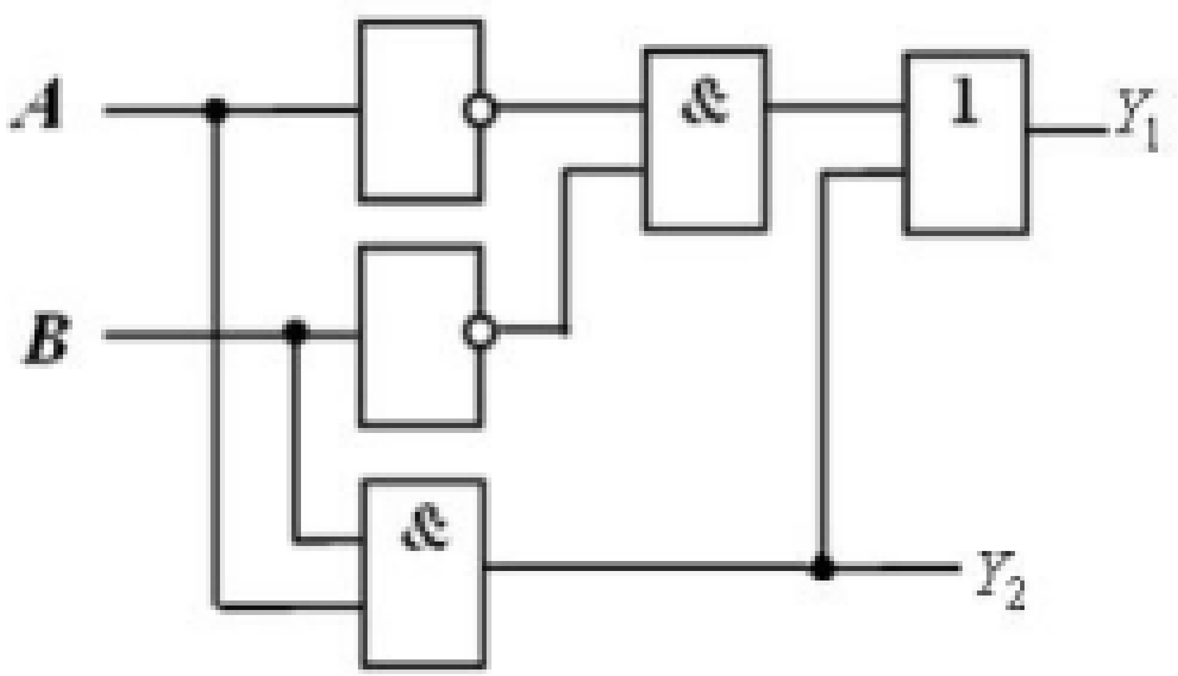

Решение в logic.ly



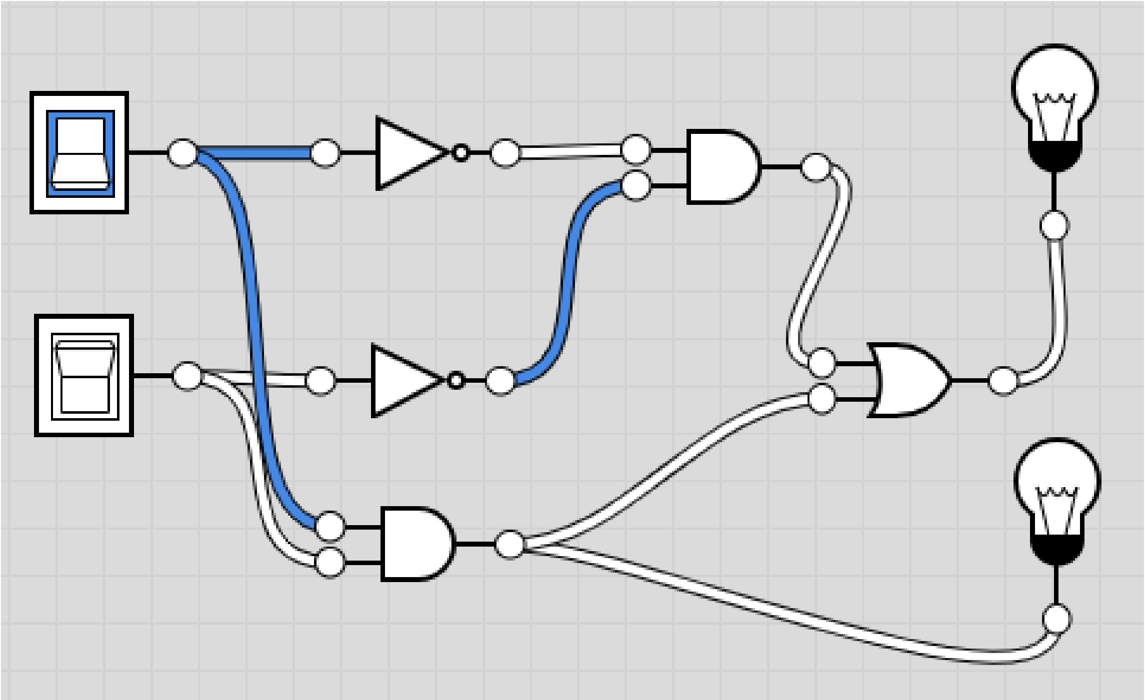
# Задача и решение

- Если на входе логической схемы подана следующая A =1, B = 0, то комбинацией на выходе $Y_1$ и $Y_2$ чему будет равна?

Задача



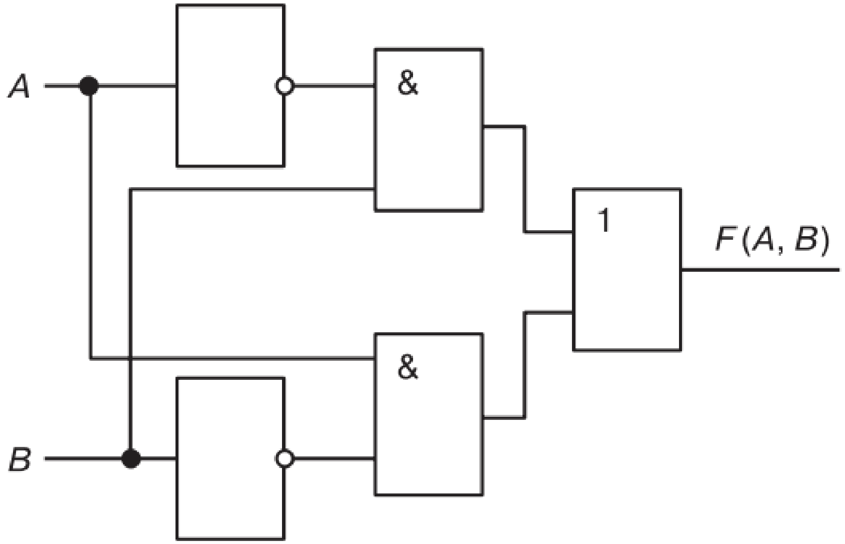

Решение в logic.ly



# Пример

**Пример.** По заданной логической функции $F(A, B) = \overline{A} \,\&\, B \vee A \,\&\, \overline{B}$ построим комбинационную схему

Построение начнём с логической операции, которая должна выполняться последней. В данном случае такой операцией является логическое сложение, следовательно, на выходе логической схемы должен быть дизъюнктор. На него сигналы подаются с двух конъюнкторов, на которые в свою очередь подаются один входной сигнал нормальный и один инвертированный (с инверторов).

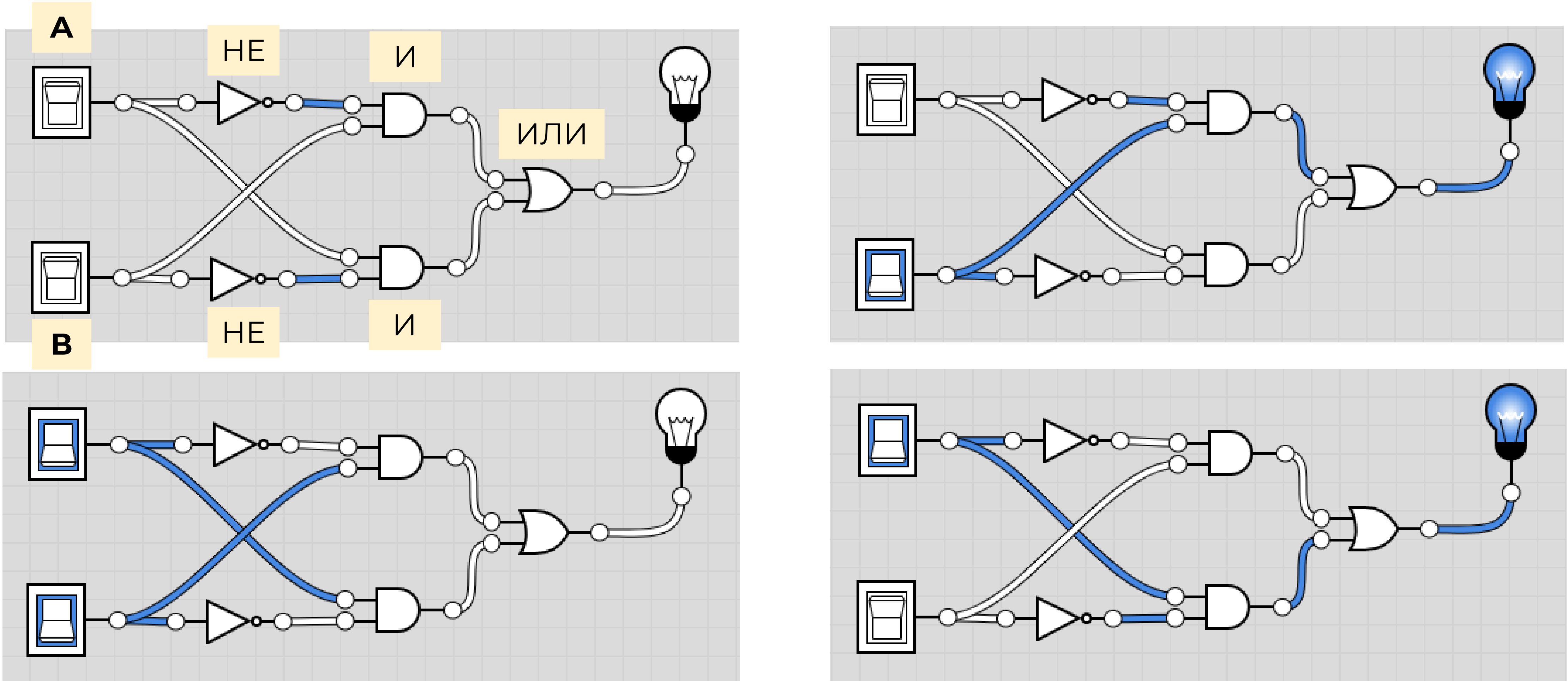



Комбинационная схема функции $F(A, B) = \overline{A} \,\&\, B \vee A \,\&\, \overline{B}$

# Пример решения в logic.ly

НЕ И ИЛИ И НЕ

$$F(A,\ B) = \overline{A}\ \&\ B \vee A\ \&\ \overline{B}$$



# Благодарю за внимание

**КУТУЗОВ** Виктор Владимирович

# Список использованных источников

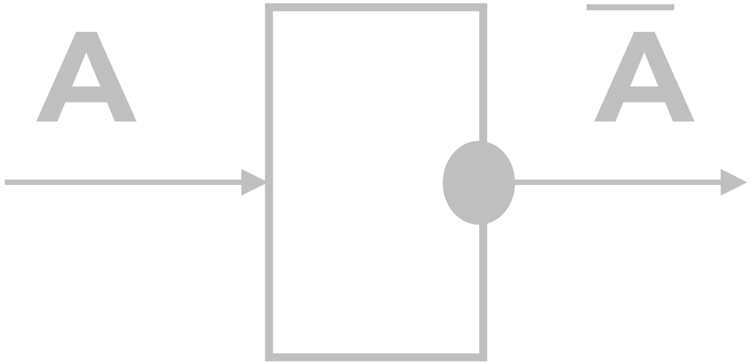

1. Рабочая программа дисциплины «Информатика» / Кутузов В.В.  – Могилев : Белорусско-Российский университет, 2023
2. Насыров И. А.  Лабораторный практикум. Основы построения цифровых логических устройств. Часть 1: Функции алгебры–логики и синтез логических схем. Учебно–методическое пособие. — Казань: Казанский университет, 2012. — 88 с. https://kpfu.ru/staff_files/F16500157/DigitElectrLabsPart1.pdf
3. Яшин В. Н. Информатика : учебник / В. Н. Яшин, А.Е. Колоденкова. — Москва: ИНФРА-М, 2021. — 522 с. https://znanium.com/catalog/product/1069776
4. Закляков В. Ф. Информатика: учеб. для вузов – 5-е изд., перераб. и доп. – М.: ДМК Пресс, 2021. – 750 с.
5. Поляков К. Ю. Информатика. 9 класс / К. Ю. Поляков, Е. А. Еремин. — М. : БИНОМ. Лаборатория знаний, 2017. — 288 с.
6. Поляков К. Ю. Информатика (базовый и углублённый уровни) (в 2 частях). 10 класс. Ч. 1 : учебник / К. Ю. Поляков, Е. А. Еремин. — М. : БИНОМ. Лаборатория знаний, 2019. — 352 с.: ил. https://djvu.online/file/ko8xgrJjGkkRP
7. Босова Л. Л. Информатика. 10 класс : учебник / Л. Л. Босова, А. Ю. Босова. — М. : БИНОМ. Лаборатория знаний, 2016. — 288 с. : Глава 4 Элементы теории множеств и  алгебры логики https://files.lbz.ru/authors/informatika/3/bosova-10-gl4.pdf
9. Элементы схемотехники в рамках курса информатики : методические указания к выполнению самостоятельной работы по информатике по теме «Основы схемотехники» для обучающихся по всем программам и форм обучения / сост. Ю. С. Бузыкова. – Хабаровск : Изд-во Тихоокеан. гос. ун-та, 2015. – 44 с. https://pnu.edu.ru/media/filer_public/c2/21/c2216330-1813-407f-b789-9a0b86f21e37/mu_osnovi_shemotehniki_busikova.pdf

# Список использованных источников


10. Панюкова, Е.В. Информатика : учеб.-метод. пособие / Е.В. Панюкова, Э.В. Егорова. – Тольятти : Изд-во ТГУ, 2012. – 148 с. https://dspace.tltsu.ru/bitstream/123456789/310/1/Егорова%201-85-11.pdf
11. Опарин Д.В. Практикум по основам алгебры логики. Часть I. Логические операции над высказываниями, формулы и функции алгебры логики – УФУ: Екатеринбург – 2015 – 26с. https://study.urfu.ru/Aid/Publication/13279/1/Oparin_1.pdf
12. Угринович Н. Д. Информатика и ИКТ. Профильный уровень : учебник для 10 класса / Н. Д. Угринович. — 3-е изд., испр. — М. : БИНОМ. Лаборатория знаний, 2008. — 387 с. https://sch4.clan.su/_ld/2/270___-10_-___._200.pdf
13. Практическая работа №1 Логические элементы и схемы https://portal.tpu.ru/SHARED/s/SHIDLOVSKIY/ur/Tab4/Практическая%20работа%20№1.pdf
14. Логические схемы терминалов релейной защиты http://digitalsubstation.com/blog/2016/11/02/logicheskie-shemy-terminalov-relejnoj-zashhity/
15. Информатика. Алгебра логики: Таблицы истинности. Центр онлайн-обучения «Фоксфорд» https://www.youtube.com/watch?v=qrj6Ekwqr-c
16. Логические элементы https://www.youtube.com/watch?v=3JCRI5YHbJU&t=1s
17. Логические выражения, таблицы истинности ,структурная логическая схема https://www.youtube.com/watch?v=7rcHdOStwcA
18. Презентация Свойства логических операций https://infedu.ru/2015/10/10/prezentatsiya-svoystva-logicheskih-operatsiy/

# Список использованных источников


19. Логические схемы и таблицы истинности
    https://function-x.ru/logicheskie_shemy_i_tablici_istinnosti.html
    https://function-x.ru/logicheskie_shemy_s2.html
20. Введение в логику, урок 1: Базовые понятия
    https://www.youtube.com/watch?v=eXI_TFW5Cdo
21. Введение в логику, урок 2: Представление функций
    https://www.youtube.com/watch?v=eFcz9agrOro
22. КАК работает ПРОЦЕССОР? ОБЪЯСНЯЕМ
    https://www.youtube.com/watch?v=qIhZrMg3_Tk
23. Логические элементы И, ИЛИ, Исключающее ИЛИ. История, Теория, Применение.
    https://www.youtube.com/watch?v=bXdiYU3IUJA
24. Логические элементы
    https://ru.wikipedia.org/wiki/Логические_элементы
25. logic.ly
    https://logic.ly/demo/
26. Основы логики и логические основы компьютера
    https://ppt-online.org/73297
27. Музей электронных раритетов
    http://www.155la3.ru
28. Tutorials inLogic Gates
    https://www.electronics-tutorials.ws/logic/logic_1.html




# Дополнительные материалы для самостоятельного изучения

# Дополнительные материалы для самостоятельного изучения

- Манушакян К. Г. **Цифровые устройства**: логические элементы: учебно-методическое пособие / К.Г. Манушакян, Н.Ю. Лахтина. – М.: МАДИ, 2021. – 68 с. https://lib.madi.ru/fel/fel1/fel22E569.pdf

- **Алгебра логики в симуляторе логических схем Logic.ly** https://dzen.ru/a/Y-zWi5RILSGfz24a

- **Элементы схемотехники в рамках курса информатики** : методические указания к выполнению самостоятельной работы по информатике по теме «Основы схемотехники» для обучающихся по всем программам и форм обучения / сост. Ю. С. Бузыкова. – Хабаровск : Изд-во Тихоокеан. гос. ун-та, 2015. – 44 с. https://pnu.edu.ru/media/filer_public/c2/21/c2216330-1813-407f-b789-9a0b86f21e37/mu_osnovi_shemotehniki_busikova.pdf