

Белорусско-Российский университет

Кафедра «Программное обеспечение  
информационных технологий»

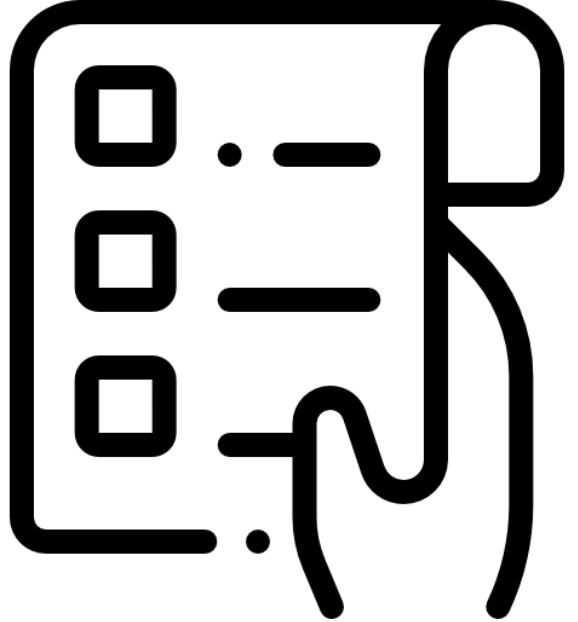
# ЭВМ, периферийные устройства и контроллеры

Тема: Уровни абстракции  
электронно-вычислительных  
систем

Кутузов Виктор Владимирович

Республика Беларусь, Могилев, 2025





# Содержание лекции

# Содержание лекции

## Тема: Уровни абстракции электронно-вычислительных систем

1. Рекомендуемые материалы по теме
2. Уровни абстракции электронно-вычислительных систем
3. Цифровая абстракция
4. Системы счисления и представление информации
5. Системы счисления. Преобразование чисел
6. Системы счисления. Арифметические операции
7. Булева алгебра. Основные логические операции
8. Уровень цифровой логики
9. За пределами цифровой абстракции. Транзисторы. MOS. CMOS

# Дополнительные материалы по теме на YouTube

## Тема: Уровни абстракции электронно-вычислительных систем

1. Дополнительные материалы по теме на YouTube
2. Уровни абстракции электронно-вычислительных систем
3. Системы счисления и представление информации
4. Булева алгебра
5. Цифровая логика. Базовые логические элементы
6. Физический уровень работы простейших элементов компьютерных комплектующих
7. Релейная логика
8. Ламповая логика. Применяются вакуумные лампы (электронные лампы)

# Дополнительные материалы по теме на YouTube

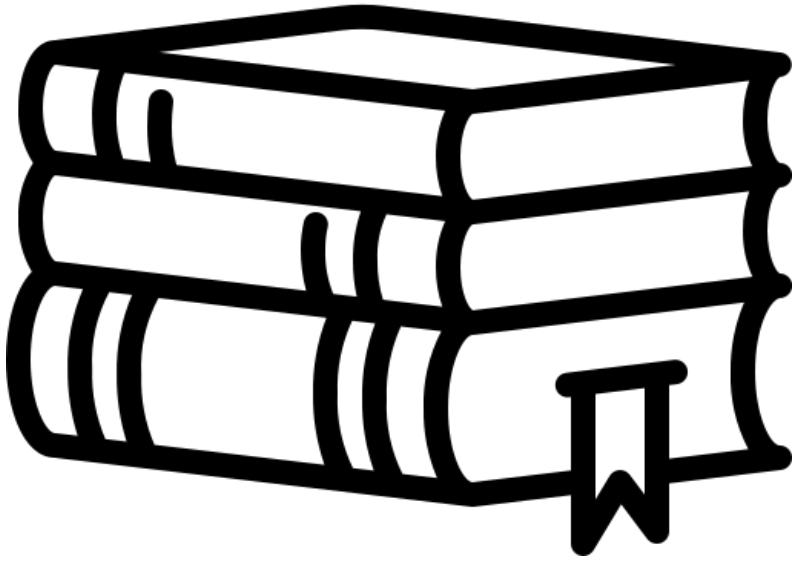
## Тема: Уровни абстракции электронно-вычислительных систем

9. Резисторно-транзисторная логика. Resistor-Transistor Logic (RTL). Применяются биполярные транзисторы
10. Диодно-транзисторная логика. Diode Transistor Logic (DTL). Используются биполярные транзисторы (BJT) и диоды
11. Транзисторно-транзисторная логика. TTL (Transistor-Transistor Logic). Используются биполярные транзисторы (BJT)
12. Логика с эмиттерной связью, Эмиттерно-связанная логика. Emitter-Coupled Logic (ECL)
13. Логика металл-оксид-полупроводник. Metal-Oxide-Semiconductor Logic (MOS). MOSFET

# Дополнительные материалы по теме на YouTube

## Тема: Уровни абстракции электронно-вычислительных систем

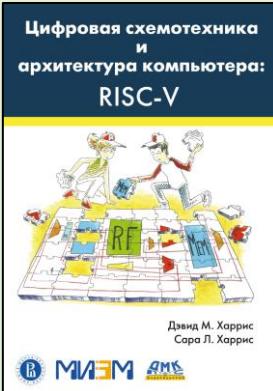
14. Комплементарная логика металл-оксид-полупроводник (КМОП). Complementary Metal-Oxide-Semiconductor Logic (CMOS)



## Рекомендуемые материалы по теме



# Литература



Сара Л. Харрис, Дэвид Харрис **Цифровая схемотехника и архитектура компьютера: RISC-V** / пер. с англ. В. С. Яценкова, А. Ю. Романова; под ред. А. Ю. Романова. – М.: ДМК Пресс, 2021. – 810 с. <https://rutracker.org/forum/viewtopic.php?t=6204850>

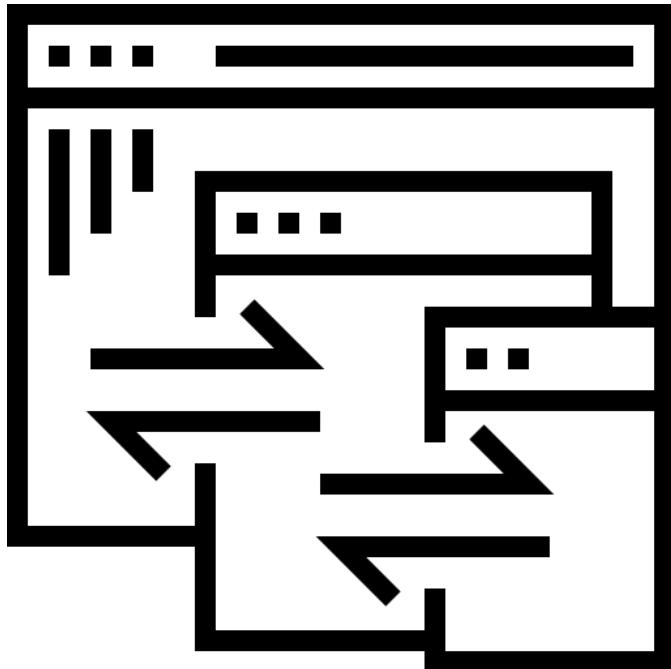


Белоус А. И., Красников Г. Я., Солодуха В. А. **Основы проектирования субмикронных микросхем.** Москва: ТЕХНОСФЕРА, 2020. – 782 с. <https://djvu.online/file/mXzkarl4cai0s>



**Электротехника и электроника. Наглядные пособия. Таблицы. Схемы** – Челябинск: ЮУрГУ, 2011. – 106 с.

<https://www.twirpx.com/file/1472423/>  
<https://djvu.online/file/uWScR1y9AmZXC>



# Уровни абстракции электронно- вычислительных систем



# Текущая ситуация

- За последних несколько десятилетий микропроцессоры буквально изменили наш мир до неузнаваемости.
- Ноутбук и смартфон сейчас обладают большей вычислительной мощностью, чем большой компьютер из недавнего прошлого, занимавший целую комнату.
- Внутри современного автомобиля представительского класса можно обнаружить около пятидесяти микропроцессоров.
- Именно прогресс в области микропроцессорной техники сделал возможным появление сотовых телефонов и сети Интернет, значительно продвинул вперед медицину и радикально изменил тактику и стратегию современных войн, кардинально изменил отдельные сферы промышленности и производства.
- Объем продаж мировой полупроводниковой промышленности вырос с 21 миллиарда долларов в 1985 году до 600 миллиардов долларов в 2022 году.
- При этом объем рынка товаров, содержащих чипы составляет десятки триллионов долларов. Общая схема полупроводниковой экосистемы достаточно простая и представляет собой производство чипов, которые в дальнейшем используются при создании конечного продукта.

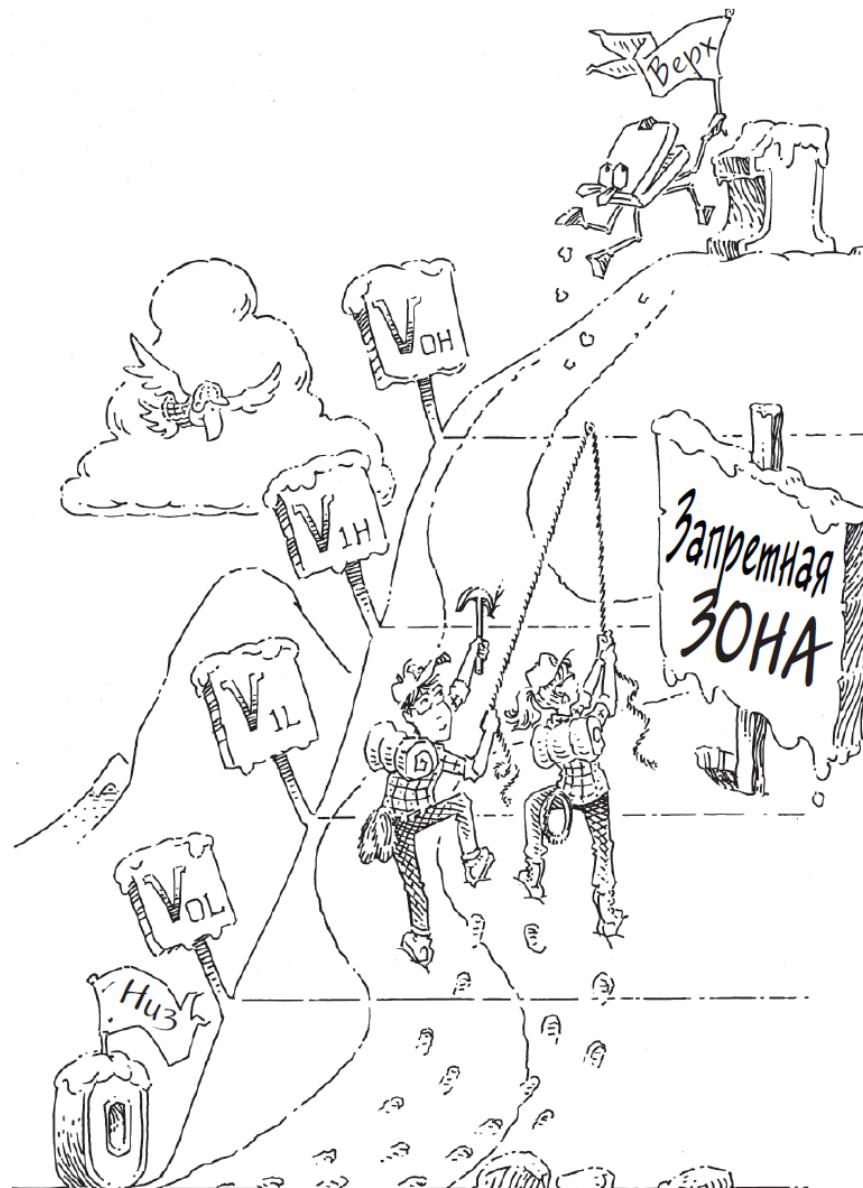
# Краткий план

- **При изучении данной дисциплины мы начнем с базовых элементов:** систем счисления, основ кодирования информации, физических принципов работы электронных устройств от простейших до сложных, перейдем на уровень цифровой логики.
- Перейдем к простейшим цифровым логическим элементам, которые принимают определенную комбинацию единиц и нулей на входах и трансформируют ее в другую комбинацию единиц и нулей на выходах.
- После этого мы с вами научимся объединять эти простейшие логические элементы в более сложные модули, такие как сумматоры, блоки памяти и другие элементы.
- Кратко коснемся вопросов программирования на языке ассемблера – родном языке микропроцессора.
- Затем из кирпичиков логических элементов мы рассмотрим как собирается полноценный микропроцессор, способный выполнять программы, разработанные на языке ассемблера.
- Рассмотрим все остальные элементы компьютера и периферийных устройств изучая принцип их работы, составные части и технические характеристики.
- **Но чтобы не утонуть в этом море информации нам нужно освоить навык управления сложностью – применение абстракций.**

# Искусство управления сложностью

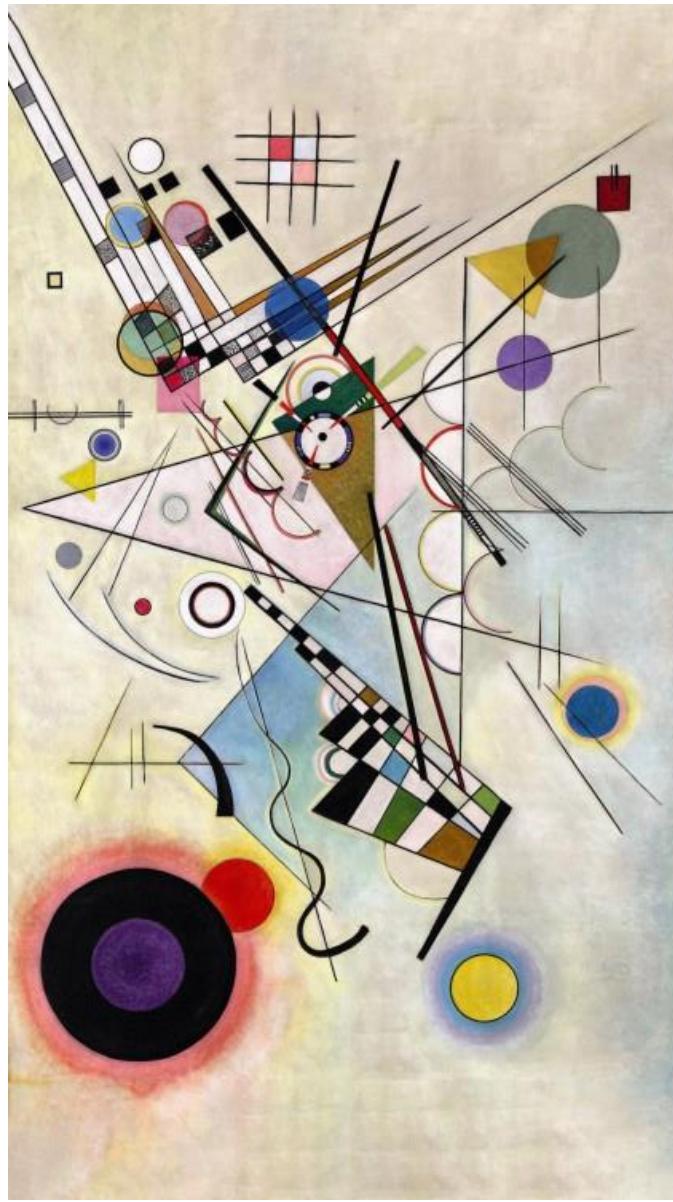
- Одной из характеристик, отличающих профессионального инженера-электронщика или программиста от дилетанта, является **систематический подход к управлению сложностью многоуровневой системы**.
- Современные цифровые системы построены из миллионов и миллиардов транзисторов.
- Человеческий мозг не в состоянии предсказать поведение подобных систем путем составления уравнений, описывающих движение каждого электрона в каждом транзисторе системы, и последующего решения этой системы уравнений.
- Для того чтобы разработать или просто понять как работает современный микропроцессор и (или) другие комплектующие компьютера и не утонуть при этом в море избыточной информации, необходимо научиться управлять сложностью изучаемых (разрабатываемых) систем.

# Уровни абстракции



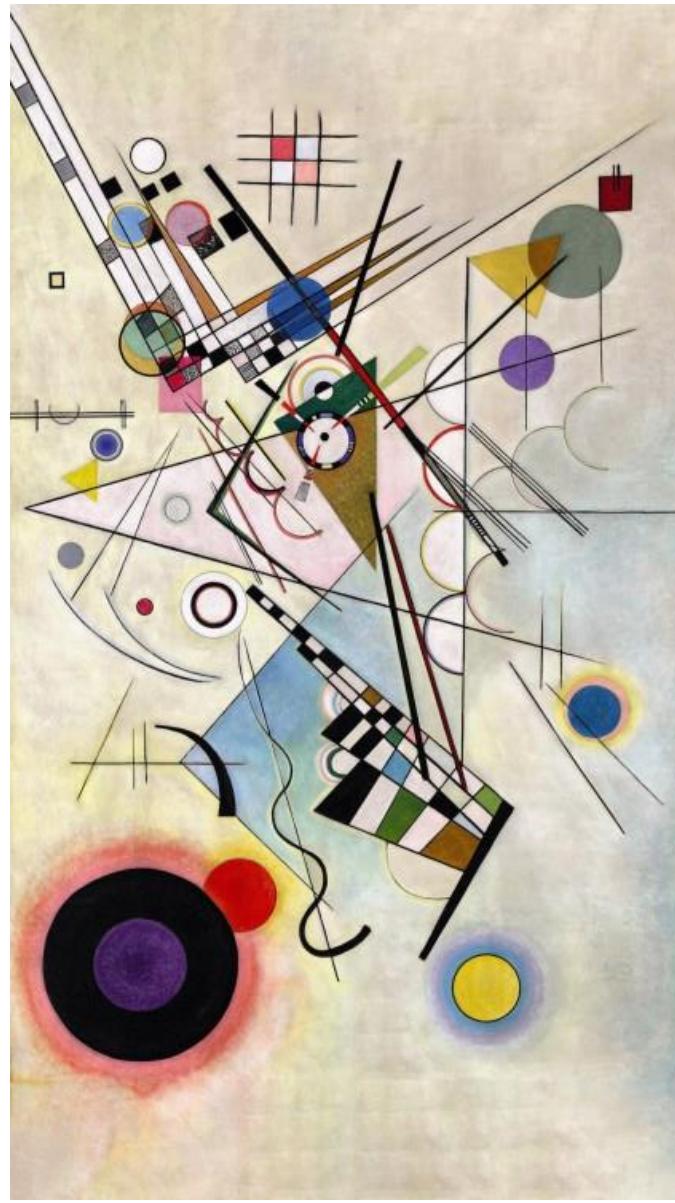
- Критически важный принцип управления сложностью системы – абстракция, подразумевающая исключение из рассмотрения тех элементов, которые в данном конкретном случае несущественны для понимания работы этой системы.
- Любую систему можно рассматривать с различных уровней абстракции.

# Абстракция



- **Абстракция** (отвлечение) процесс отвлечения (абстрагирования) от тех или иных характеристик объекта для их избирательного анализа; при этом наблюдаемый объект замещается его идеализированным теоретическим образом — абстрактным объектом.
- **Абстракции являются универсальным методом научного познания**, они необходимы для формирования понятий, узнавания и классификации объектов исследования на всех уровнях формирования знаний.
- Абстракция это когда берут какой-то сложный объект и убирают те детали, которые не имеют существенного значения для понимания какого бы то ни было свойства этого объекта.
- **Абстракции это не выдумка программистов.** Абстракции появились задолго до появления программирования и электроники так таинственной вообще. **Это более фундаментальная концепция!**

# Абстракция



- **Абстракция (абстрагирование)** — это процесс. Процесс уменьшения детализации в восприятии объекта.
- **Абстракция** — это объект. Объект без избыточных для решаемой задачи атрибутов.
- **Уровнем абстракции** определяется степень детализации предмета рассмотрения.

# Абстракция

- В контексте вычислительной техники **абстракция** — это представление, которое скрывает специфику реализации от потребителя сервисов (потребителем является компьютерная программа или человек), делая систему более простой и понятной.
- **Хороший пример — операционная система (ОС) компьютера.** Она абстрагирует все тонкости работы компьютера. Пользователю не нужно ничего знать о процессоре, памяти и работе с программами: он просто управляет операционной системой, а та разбирается со всеми нюансами. Все они скрыты за «занавесом» ОС или абстракцией.
- Системы, как правило, имеют несколько уровней абстракции. Это значительно упрощает разработку. При программировании разработчики создают компоненты, совместимые с определенным уровнем абстракции, и не заботятся о том, как та или иная функциональность реализована на нижележащих уровнях. Достаточно того, что эти компоненты работают с определенным уровнем абстракции. А то, что у него скрыто «под капотом», не имеет значения.

# Причины использования абстракций

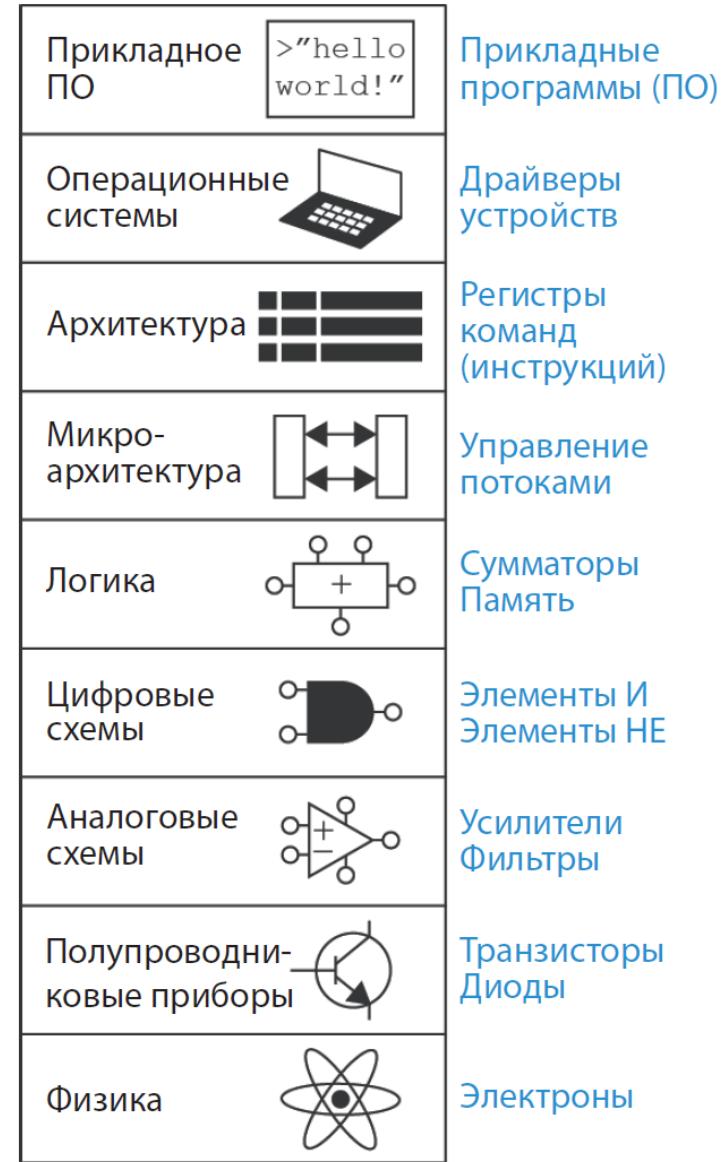
- Абстракции в вычислительных системах критически важны по нескольким ключевым причинам:
- **Борьба со сложностью.** Современные компьютеры — невероятно сложные системы (миллиарды транзисторов, слои ПО). Абстракция позволяет:
  - Разделить систему на понятные "слои" (например, железо → ОС → приложения).
  - Концентрироваться на одной задаче, игнорируя детали нижележащих уровней (программист не думает о физике транзисторов).
- **Ускорение разработки**
  - Повторное использование: Готовые абстракции (библиотеки, API, драйверы) избавляют от изобретения велосипедов.
  - Параллельная работа: Разные команды могут разрабатывать ОС, драйверы и приложения одновременно, соблюдая согласованные интерфейсы.
- **Независимость от "железа"** Абстракции обеспечивают переносимость. Например:
  - Программа, написанная под Windows, работает на любом совместимом железе.
  - Язык высокого уровня (Python) выполняется на разных процессорах благодаря слою абстракции (интерпретатор/виртуальная машина).
- **Безопасность и стабильность**
  - Скрытие деталей реализации (например, доступ к памяти через ОС) предотвращает ошибки и атаки.
  - Изменения в одном слое (апгрейд драйвера) не ломают другие уровни, если интерфейс сохранён.

# В чем достоинства абстракций?

- **1. Они помогают понять сложные вещи.** Абстракции делают возможным применение формальных методов, и алгоритмов, т.е. в конечном счете логики. (модель атома Бора, абстрактные структуры данных в программировании, закон всемирного тяготения, математический маятник, идеальный газ, материальная точка)
- **2. Они уменьшают время понимания сложных вещей** (схема метро, электрические принципиальные схемы)
- **3. Они позволяют масштабировать сложность, декомпозировать трудные задачи.** (автоматическая коробка передач, драйвер шагового двигателя, алгоритмы компрессии данных).
- **4. Они помогают переносить объекты на другую систему координат** (Языки программирования Java, гипервизоры, Docker контейнеры)
- **5. Они заменяют сложное на простое** (языки программирования, файловые системы, DNS адреса, иконки на рабочем столе, программы с GUI).
- **6. Они помогают уйти от ответственности** (out source компаний, модель OSI-7)

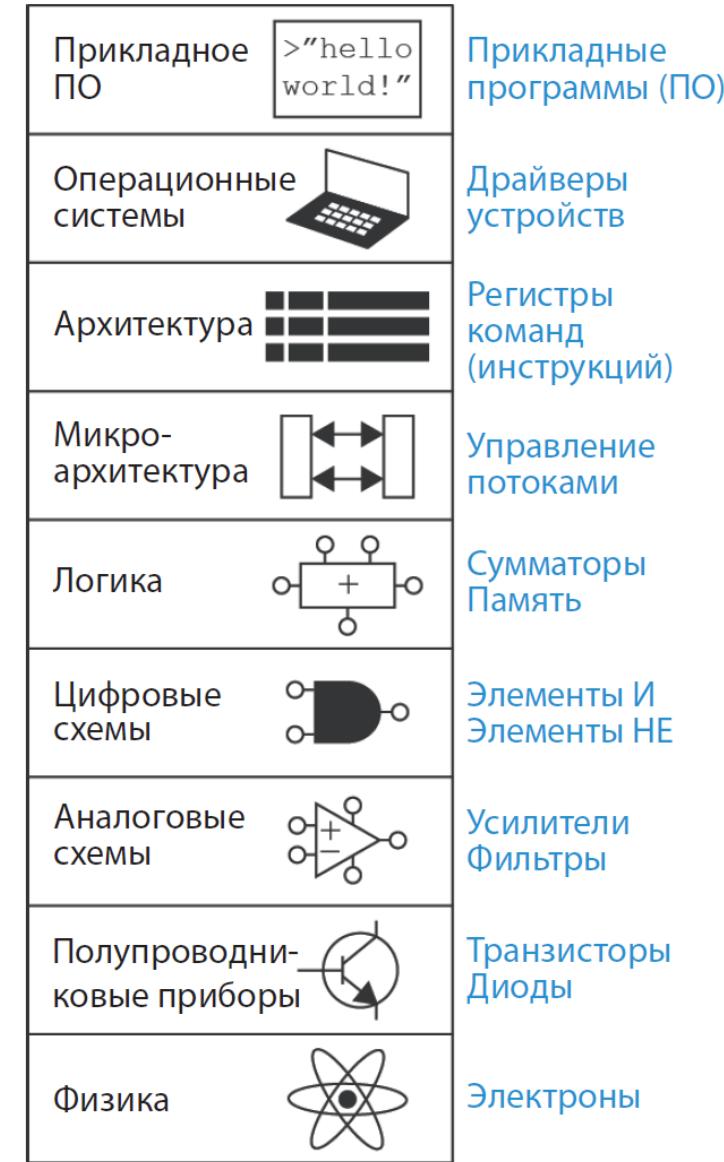
# Уровни абстракции

1. **Физика (Physics)** – поведение электронов в ЭВМ описывается квантовой механикой и уравнениями Максвелла (теоретическая физика)
2. **Устройство, полупроводниковые приборы (Devices)** – полупроводниковые устройства, такие как транзисторы, диоды (физика)
3. **Аналоговые схемы (Analog Circuits)** – полупроводниковые устройства соединены в функциональные компоненты, такие как усилители, фильтры и т.п. (физика)
4. **Цифровые схемы (Digital Circuits)** – уровень логических вентилей (logic gates), которые используют строго ограниченное число дискретных уровней напряжения. Базовые логические элементы И, НЕ и т.д. (проектирование схем, логика)
5. **Логический уровень (Logic)** – комбинированная логика, объединяющая набор вентилей в отдельные схемы, в том числе базовые блоки, такие как сумматоры и т.п. (проектирование схем, математическая логика и теория алгоритмов)
6. **Микроархитектура (Microarchitecture)** – соединение цифровых элементов в логические блоки, выполняющие определённые команды. Способ связи разработчиков «железа» и программистов (проектирование схем, методы оптимизации) – Связь логического и архитектурного уровней
7. **Архитектура (Architecture)** – описание компьютера с точки зрения программиста как некоторого набора ресурсов и команд (x86)
8. **Операционные системы (Operating systems)** – управление операциями нижнего уровня, такими как доступ к памяти и т.д. (программирование, в т.ч. низкоуровневое)
9. **Программное обеспечение, прикладное ПО (Application software)** – решение конкретных прикладных задач (программирование, как правило, высокоуровневое)



# Уровни абстракции

- На самом низком уровне абстракции находится физика, изучающая движение электронов. Поведение электронов описывается квантовой механикой и системой уравнений Максвелла.
- Рассматриваемая **современная электронная система** **состоит из полупроводниковых устройств** (devices), таких как транзисторы (а когда-то это были электронные лампы).
- Каждое такое устройство имеет четко определенные точки соединения с другими подобными устройствами. Эти точки мы будем называть контактами (в англоязычной литературе используется термин terminal).
- Любое электронное устройство может быть представлено абстрактной математической моделью**, описывающей изменяющуюся во времени взаимозависимость тока и напряжения. Такие же изменения тока и напряжения можно наблюдать на экране осциллографа, если подключить осциллограф к контактам реального устройства.
- Данный подход означает, что если рассматривать систему на **уровне устройств**, функции которых однозначно определены, то можно не учитывать поведение электронов внутри отдельных устройств этой системы.



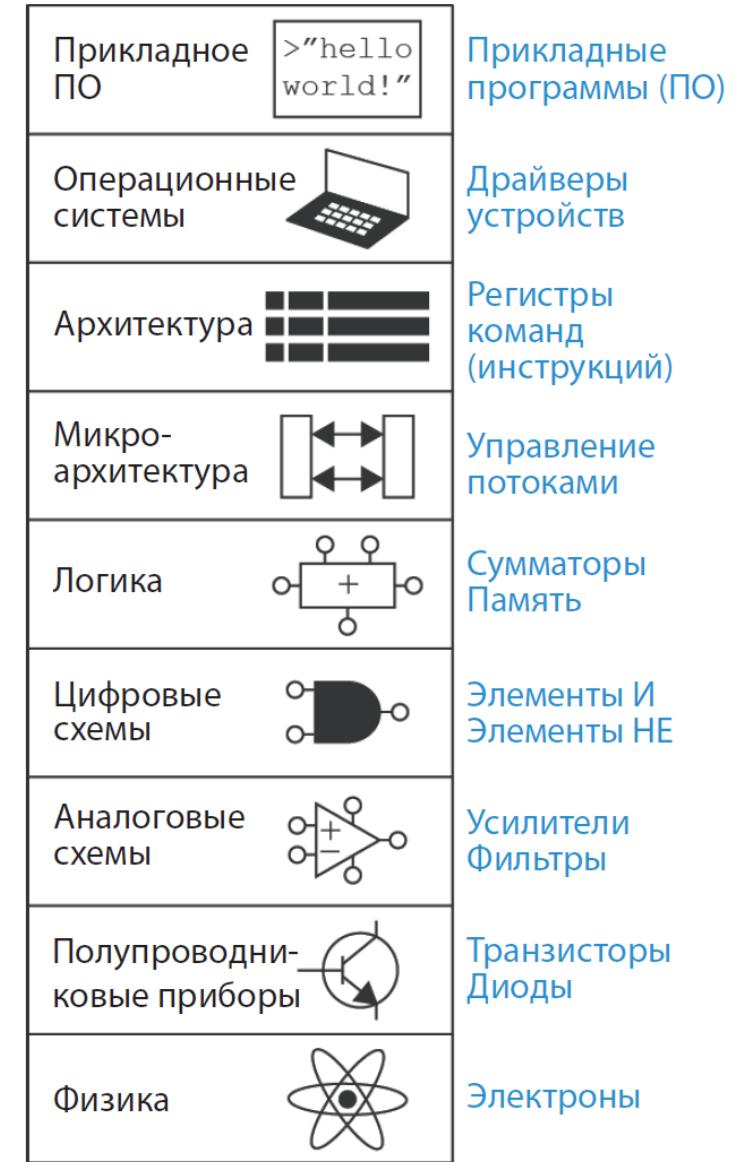
# Уровни абстракции

- **Следующий уровень абстракции – это аналоговые схемы** (analog circuits), в которых полупроводниковые устройства соединены таким образом, чтобы они образовывали функциональные компоненты, например усилители. Напряжение на входе и на выходе аналоговой цепи изменяется в непрерывном диапазоне.
- В отличие от аналоговых цепей, **цифровые схемы** (digital circuits), такие как логические элементы, используют два строго ограниченных дискретных уровня напряжения.
- **Один из этих дискретных уровней – это логический ноль, другой – логическая единица.**
- **Микроархитектурный уровень абстракции, или просто микроархитектура** (microarchitecture), связывает логический и архитектурный уровни абстракции.
- **Архитектурный уровень абстракции**, или архитектура (architecture), описывает компьютер с точки зрения программиста.



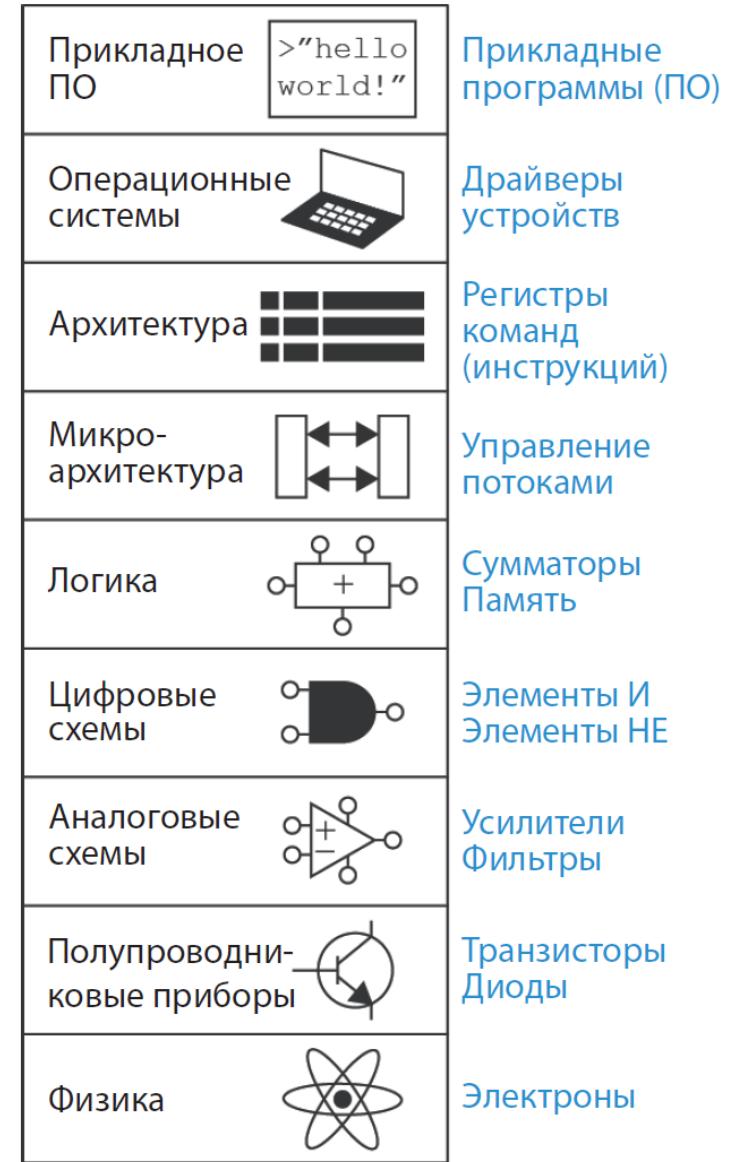
# Уровни абстракции

- **Микроархитектура** – это соединение простейших цифровых элементов в логические блоки, предназначенные для выполнения команд, определенных какой-то конкретной архитектурой.
- Отдельно взятая архитектура может быть реализована с использованием различных вариантов микроархитектур с разным соотношением **цены, производительности и потребляемой энергии**, и такое соотношение зачастую выбирается как баланс между этими тремя факторами.
- Процессоры Intel Core i7, Intel 80486 и AMD Athlon, например, используют одну и ту же архитектуру x86, но реализованную с применением трех разных микроархитектурных решений.



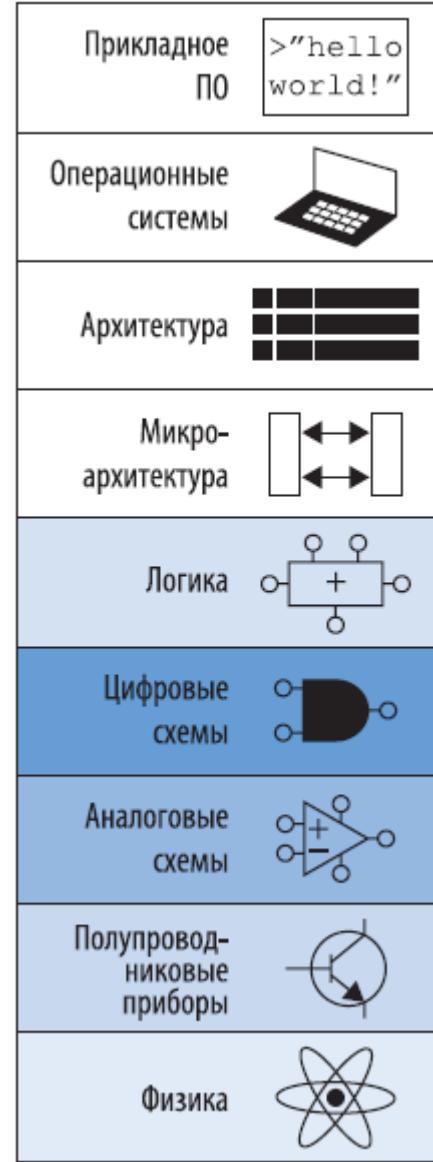
# Уровни абстракции

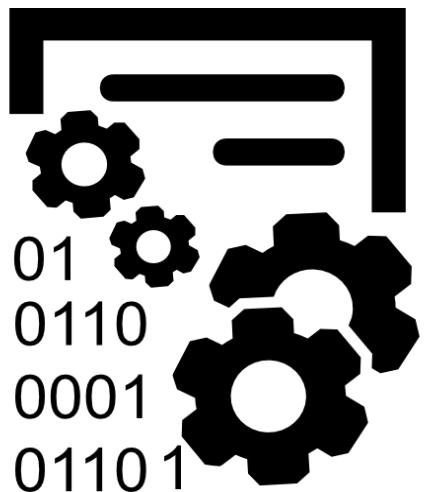
- **Область программного обеспечения.** Операционная система (operating system) управляет операциями нижнего уровня, такими как доступ к жесткому диску или управление памятью.
- И наконец, программное обеспечение использует ресурсы операционной системы для решения конкретных задач пользователя.
- Именно принцип абстрагирования от маловажных деталей позволяет вашей бабушке общаться с внуками в интернете, не задумываясь о квантовых колебаниях электронов или организации памяти компьютера.



# Три базовых принципа

- В дополнение к абстрагированию от несущественных деталей и конструкторской дисциплине разработчики электронных систем используют еще **три базовых принципа для управления сложностью системы: иерархичность, модульность конструкции и регулярность.**
- **Эти принципы применимы как к программному обеспечению, так и к аппаратной части компьютерных систем.**
  - **Иерархичность** – принцип иерархичности предполагает разделение системы на отдельные модули, а затем последующее разделение каждого такого модуля на фрагменты до уровня, позволяющего легко понять поведение каждого конкретного фрагмента.
  - **Модульность** – принцип модульности требует, чтобы каждый модуль в системе имел четко определенную функциональность и набор интерфейсов и мог быть легко и без непредвиденных побочных эффектов соединен с другими модулями системы.
  - **Регулярность** – принцип регулярности требует соблюдения единства при разработке отдельных модулей системы. Стандартные модули общего назначения, например такие как блоки питания, могут использоваться многократно, во много раз снижая количество модулей, необходимых для разработки новой системы.





# Цифровая абстракция



# Цифровая абстракция

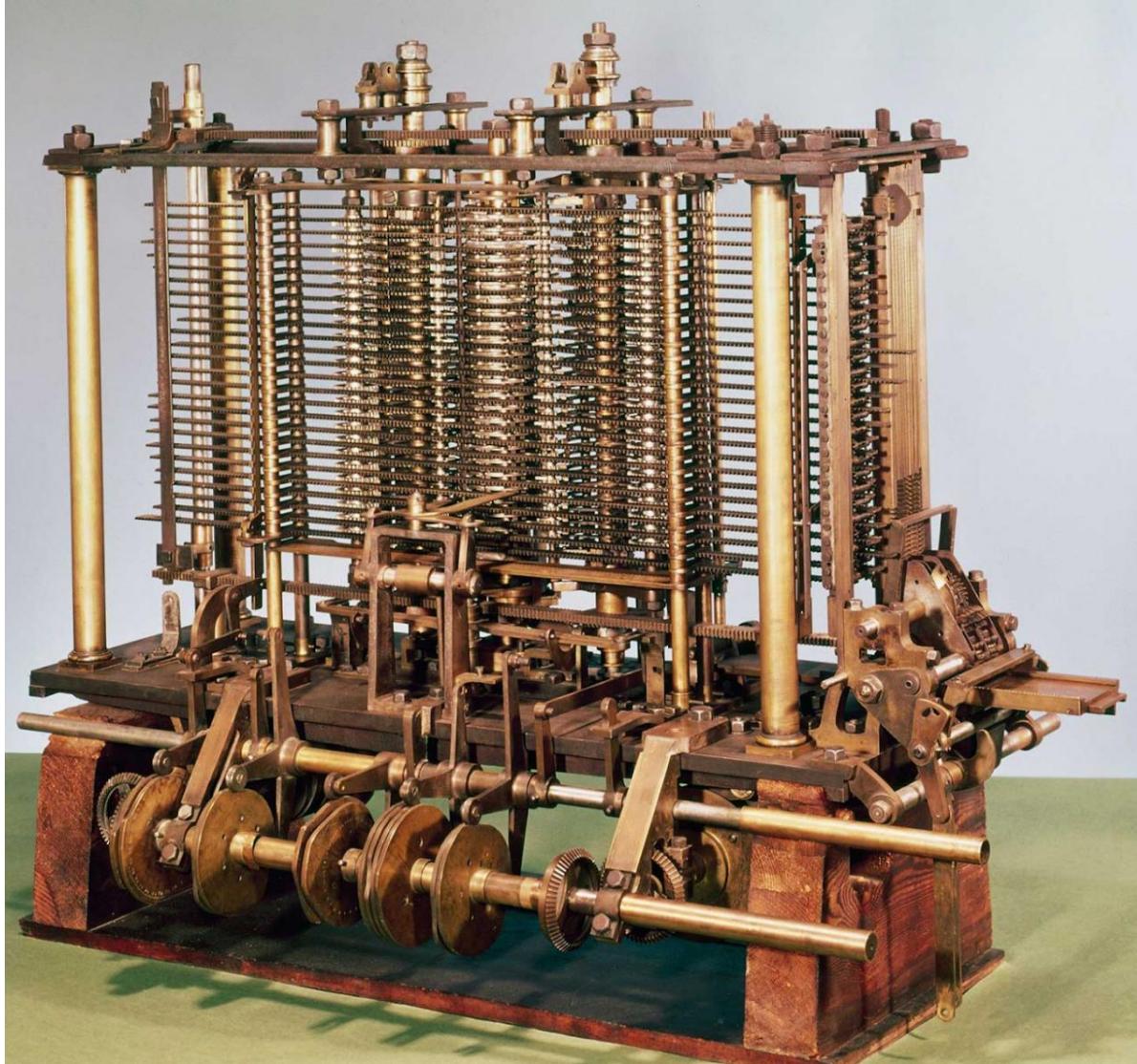
- **Большинство физических величин изменяются непрерывно.**
- Например, напряжение в электрическом проводе, частота колебаний или распределение массы – все это параметры, изменяющиеся непрерывно.
- **Цифровые системы**, с другой стороны, представляют информацию в виде дискретно меняющихся переменных с конечным числом строго определенных значений.
- **Одной из наиболее ранних цифровых систем стала аналитическая машина Чарльза Бэббиджа**, которая использовала переменные с десятью дискретными значениями.
- Начиная с 1834 года и до 1871 года **Бэббидж разрабатывал** и пытался построить этот **механический компьютер**.
- Шестеренки аналитической машины могли находиться в одном из десяти фиксированных положений, а каждое такое положение было промаркировано от 0 до 9, подобно механическому счетчику пробега автомобиля. показывает, как выглядел прототип аналитической машины.
- Каждый ряд шестеренок такой машины обрабатывал одну цифру.
- В своем механическом компьютере Бэббидж использовал 25 рядов шестеренок таким образом, чтобы машина обеспечивала вычисления с точностью до 25-го знака.



Чарльз Бэббидж  
1791 – 1871

Чарльз Бэббидж родился в 1791 году. Закончил Кембриджский университет и женился на Джорджиане Витмур. Он изобрел аналитическую машину – первый в мире механический компьютер. Чарльз Бэббидж также изобрел предохранительную решетку для локомотивов, спидометр и универсальный почтовый тариф. Ученый также очень интересовался отмычками для замков и почему-то ненавидел уличных музыкантов.

# Цифровая абстракция

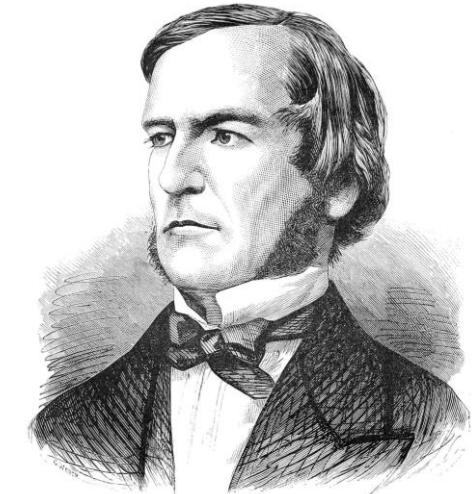


Аналитическая машина Бэббиджа

- В отличие от машины Бэббиджа **большинство компьютеров использует двоичный (бинарный) код.**
- В случае двоичного кода высокое напряжение – это **единица**, а низкое напряжение – **ноль**, поскольку гораздо легче оперировать двумя уровнями напряжения, чем десятью.
- **Объем информации**  $D$ , передаваемый одной дискретной переменной, которая может находиться в  $N$  различных состояниях, измеряется в единицах, называемых **битами**.  
$$D = \log_2 N \text{ бит}$$
- Двоичная переменная передает  $\log_2 2 = 1$  – один бит информации.
- Теперь вам, вероятно, понятно, почему единица информации называется битом.
- **Бит (bit)** – это сокращение от английского *binary digit*, что дословно переводится как двоичный разряд.

# Цифровая абстракция

- Джордж Буль разработал систему логики, использующую двоичные переменные, и эту систему сегодня называют его именем – булева логика.
- Логические переменные могут принимать значения **ИСТИНА (TRUE)** или **ЛОЖЬ (FALSE)**.
- В электронных компьютерах положительное напряжение обычно представляет единицу, а нулевое напряжение представляет ноль.
- Мы будем использовать понятия единица (1), **ИСТИНА (TRUE)** и **ВЫСОКИЙ УРОВЕНЬ СИГНАЛА (HIGH)** как синонимы.
- Аналогичным образом мы будем использовать ноль (0), **ЛОЖЬ (FALSE)** и **НИЗКИЙ УРОВЕНЬ СИГНАЛА (LOW)** как взаимозаменяемые термины.

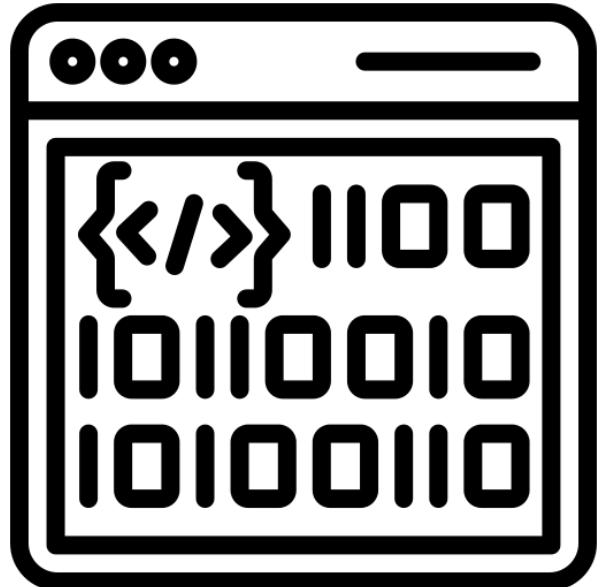


Джордж Буль  
1815–1864

Джордж Буль родился в семье небогатого ремесленника. Родители Джорджа не могли оплатить его формального образования, поэтому он осваивал математику самоучкой. Несмотря на это, Булю удалось стать преподавателем Королевского колледжа в Ирландии. В 1854 году Джордж Буль написал свою работу «Исследование законов мышления», которая впервые ввела в научный оборот двоичные переменные, а также три основных логических оператора И, ИЛИ, НЕ (AND, OR, NOT).

# Цифровая абстракция

- **Преимущества цифровой абстракции заключаются** в том, что разработчик цифровой системы может сосредоточиться исключительно на единицах и нулях, полностью игнорируя, каким образом логические переменные представлены на физическом уровне.
- Разработчика не волнует, представлены ли нули и единицы определенными значениями напряжения, врачающимися шестернями или уровнем гидравлической жидкости.
- Программист может продуктивно работать, не располагая детальной информацией об аппаратном обеспечении компьютера.
- Но **понимание того, как работает это аппаратное обеспечение, позволяет программисту гораздо лучше оптимизировать программу для конкретного компьютера.**
- Как вы могли видеть выше, **один-единственный бит не может передать большого количества информации.**
- Поэтому мы рассмотрим вопрос о том, каким образом набор битов можно использовать для представления десятичных чисел.
- В дальнейшем посмотрим, как группы битов могут представлять буквы и даже целую программу.
- А затем как это используется на аппаратном уровне.

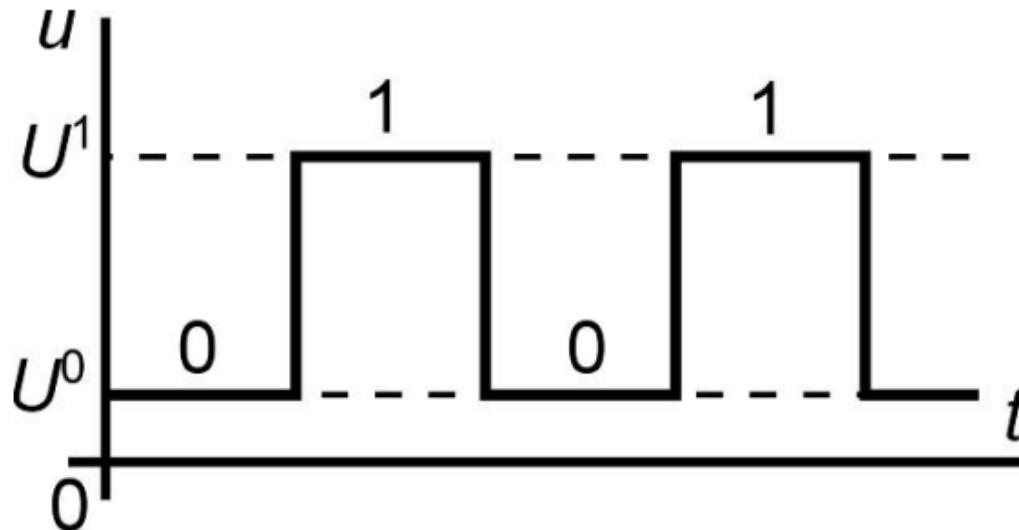


# Системы счисления и представление информации



# Цифровые устройства

- **Цифровые устройства** – электронные схемы, которые служат для обработки и преобразования цифровых сигналов.
- **Цифровой сигнал** – импульсы напряжения близкие по форме к прямоугольным.



Пример цифрового сигнала

**Символ «0» – логический ноль** – обозначает уровень низкого напряжения.

**Символ «1» – логическая единица** – обозначает уровень высокого напряжения.

**Напряжение логических единицы и нуля для каждого вида ИМС может быть различно!**

- Логические сигналы «0» или «1» несут определенную информацию.

# Системы счисления используемые в ЭВМ

В вычислительной технике в основном используются **позиционные системы счисления:**

**BIN** - двоичная,  
**DEC** - десятичная,  
**OCT** - восьмеричная,  
**HEX** - шестнадцатеричная.

Основные системы счисления десятичная и двоичная.

| <b>A<sub>10</sub></b> | <b>A<sub>2</sub></b> | <b>A<sub>8</sub></b> | <b>A<sub>16</sub></b> |
|-----------------------|----------------------|----------------------|-----------------------|
| 0                     | 0                    | 0                    | 0                     |
| 1                     | 1                    | 1                    | 1                     |
| 2                     | 10                   | 2                    | 2                     |
| 3                     | 11                   | 3                    | 3                     |
| 4                     | 100                  | 4                    | 4                     |
| 5                     | 101                  | 5                    | 5                     |
| 6                     | 110                  | 6                    | 6                     |
| 7                     | 111                  | 7                    | 7                     |
| 8                     | 1000                 | 10                   | 8                     |
| 9                     | 1001                 | 11                   | 9                     |
| 10                    | 1010                 | 12                   | A                     |
| 11                    | 1011                 | 13                   | B                     |
| 12                    | 1100                 | 14                   | C                     |
| 13                    | 1101                 | 15                   | D                     |
| 14                    | 1110                 | 16                   | E                     |
| 15                    | 1111                 | 17                   | F                     |

# **Системы счисления**

**BIN - Двоичная**

**OCT - Восьмеричная**

**DEC - Десятичная**

**HEX - Шестнадцатеричная**

# Десятичная система счисления

- Десятичная система счисления пришла в Европу из Индии, где она появилась не позднее VI века н. э.
- **В этой системе 10 цифр: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, однако информацию несет не только цифра, но и место, на котором цифра стоит (то есть ее позиция).**
- В десятичной системе счисления особую роль играют число 10 и его степени: 10, 100, 1000 и т. д.
- Самая правая цифра числа показывает число единиц, вторая справа – число десятков, следующая – число сотен и т. д.

# Двоичная система счисления

- В современной вычислительной технике, в устройствах автоматики и связи широко используется двоичная система счисления.
- В ней для изображения числа используются только две цифры: **0 и 1**.

# Двоичная система счисления



0

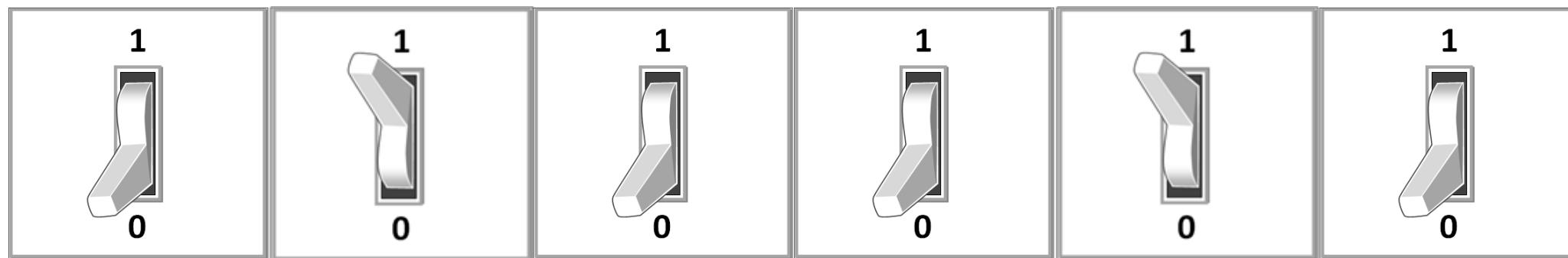
1

0

0

1

0



1

1

1

1

1

1

0

0

0

0

0

0

# Двоичная система счисления

- Одиночный бит может принимать одно из двух значений, 0 или 1.
- Несколько битов, соединенных в одной строке, образуют двоичное (binary) число.
- Каждая последующая позиция в двоичной строке имеет вдвое больший «вес», чем предыдущая позиция, так что двоичная система счисления – это система по основанию 2.
- В двоичном числе «вес» каждой позиции увеличивается (так же, как и в десятичном – справа налево) следующим образом:

**1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096,  
8192, 16384, 32768, 65536 и т.д.**

- Работая с двоичными числами, очень полезно для сохранения времени запомнить значения степеней двойки до  $2^{16}$ .

# Двоичная система счисления

- Цифровая информация в компьютере кодируется двоичным кодом.
- Двоичный код – это код, в котором используются два знака (0 и 1). Все данные в компьютере хранятся в двоичном коде.
- **Бит** — это наименьшая единица двоичного кода.
- **Бит (Binary Digit — двоичная цифра)** — это наименьшая единица представления информации.
- **Один бит может принимать только два значения: 1 (Да) или 0 (Нет).**
- **Двумя битами** можно закодировать **четыре значения:** 00, 01, 10, 11.
- **Тремя битами** можно закодировать **восемь значений:** 000, 001, 010, 011, 100, 101, 110, 111.

# Сколько разных чисел можно закодировать 1 БАЙТом?

$$00000000_2 = 0_{10}$$

$$00000001_2 = 1_{10}$$

$$00000010_2 = 2_{10}$$

$$00000011_2 = 3_{10}$$

⋮ ⋮ ⋮

$$11111111_2 = 255_{10}$$

# Бит, Байт

- **1 бит** = 0, 1
- **1 байт** = 8 бит – 0 .. 255
- **2 байта** = 16 бит – 0 .. 65 535
- **4 байта = 32 бита** – 0 .. 4 294 967 295
- **1 слово = 8 байт = 64 бит** – 0 .. 18 446 744 073 709 551 615  
Равно разрядности регистров процессора и/или шины данных.
- Все процессоры имеют несколько регистров.
- Процессор архитектуры **x86** имеет восемь 32-битных регистров.
- В архитектуре **x64** эти регистры расширены до 64 бит.



# БИТ – 1, 2, 3, 4 . . .

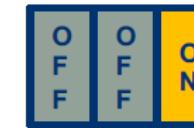
1 bit:



2 bits:



3 bits:



4 bits:



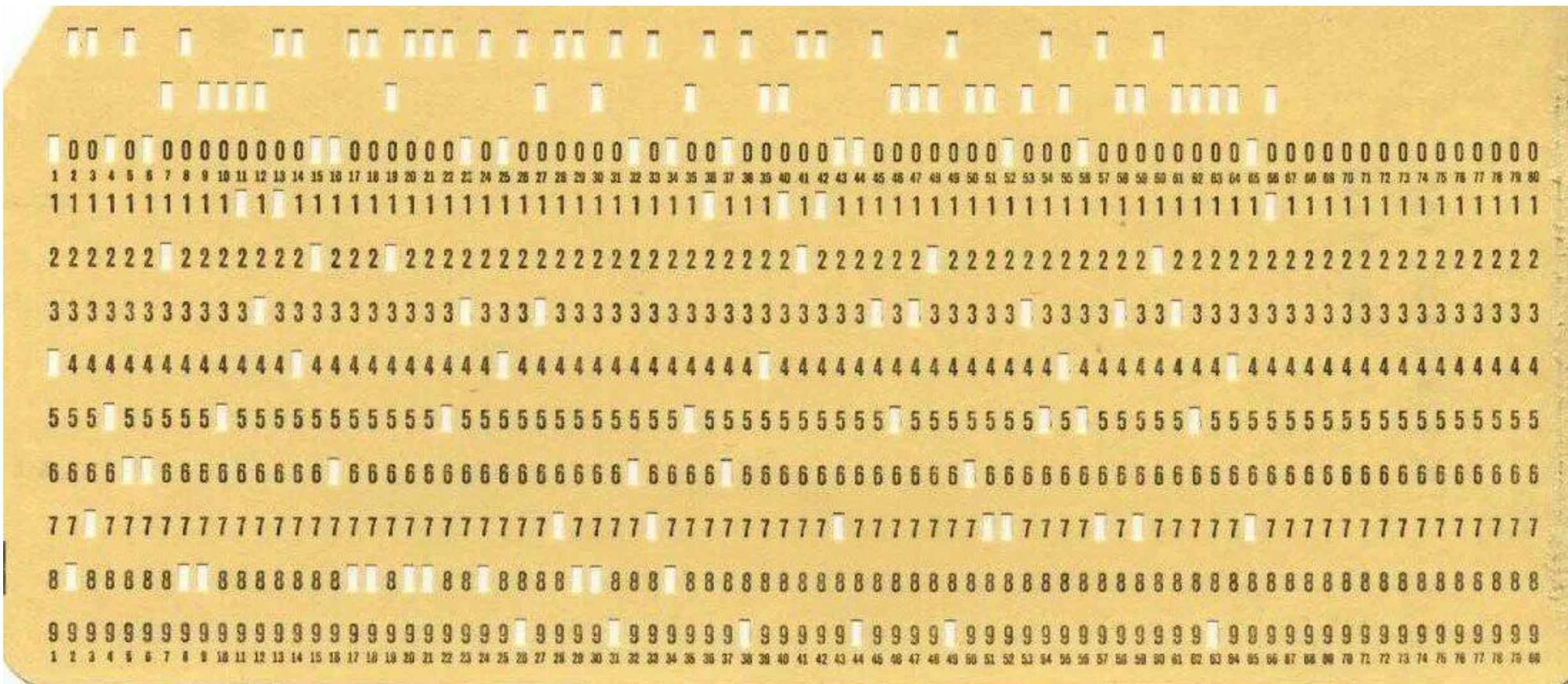
# Бит, Байт

- Группа из восьми битов называется **байт (byte)**.
- Байт представляет  $2^8 = 256$  цифровых комбинаций.
- Размер модулей, сохраненных в памяти компьютера, обычно измеряется именно в байтах, а не битах.
- Микропроцессор обрабатывает данные не целиком, а небольшими блоками, называемыми **словами**.
- **Размер слова (word) не является величиной, установленной раз и навсегда, а определяется архитектурой каждого конкретного микропроцессора.**
- Сейчас большинство компьютеров использовало **64-битные процессоры**. Такие процессоры обрабатывают информацию **блоками (словами) длиной 64 бита**.
- А еще не так давно верхом совершенства считались компьютеры, обрабатывающие информацию словами длиной 32 бита.
- Интересно, что и сегодня наиболее простые микропроцессоры и особенно те, что управляют работой таких бытовых устройств, как, например, тостеры или микроволновые печи, используют слова длиной 16 бит или даже 8 бит.

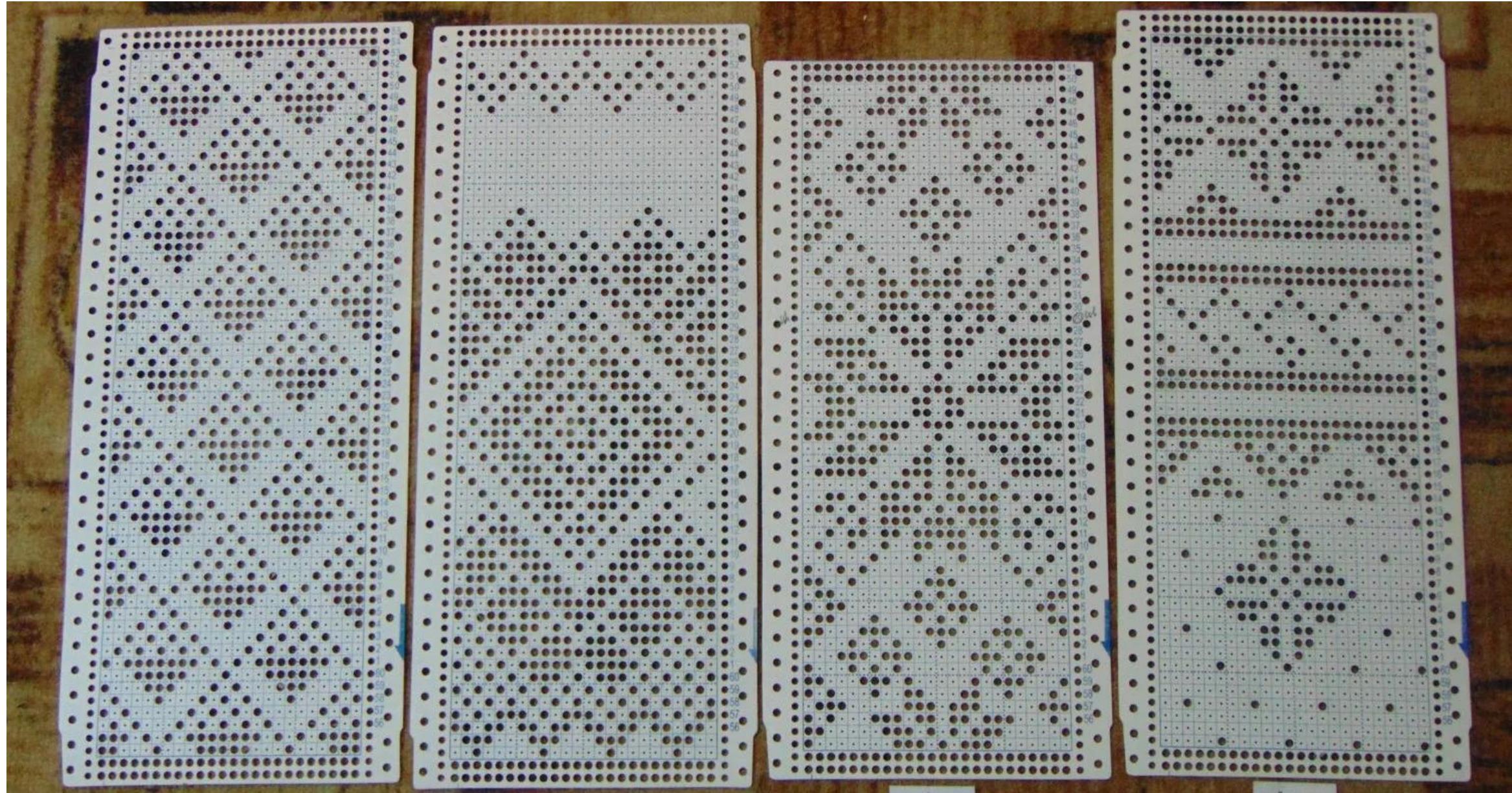
# Байт, Килобайт, Мегабайт, Гигабайт, . . .

- В силу удачного совпадения  $2^{10} = 1024 \approx 10^3$ .
- Этот факт позволяет нам использовать приставку кило (греческое название тысячи) для сокращенного обозначения  $2^{10}$ . Например,  $2^{10}$  байт – это один **килобайт (1 КБ)**.
- Подобным же образом,  **mega** (греческое название миллиона) обозначает  $2^{20} \approx 10^6$ , а  **гига** (греческое название миллиарда) указывает на  $2^{30} \approx 10^9$ .
- Зная, что  $2^{10} \approx 1$  тысяча,  $2^{20} \approx 1$  миллион,  $2^{30} \approx 1$  миллиард и помня значения степеней двойки до  $2^9$  включительно, будет легко приблизительно рассчитать в уме любую другую степень двух.
- Так же как **1024 байта называют килобайтом** (КБ), 1024 бита называют килобитом (Кб или кбит).
- **Аналогичным образом, МБ, Мб, ГБ и Гб используются для сокращенного обозначения миллиона и миллиарда байтов и битов.**

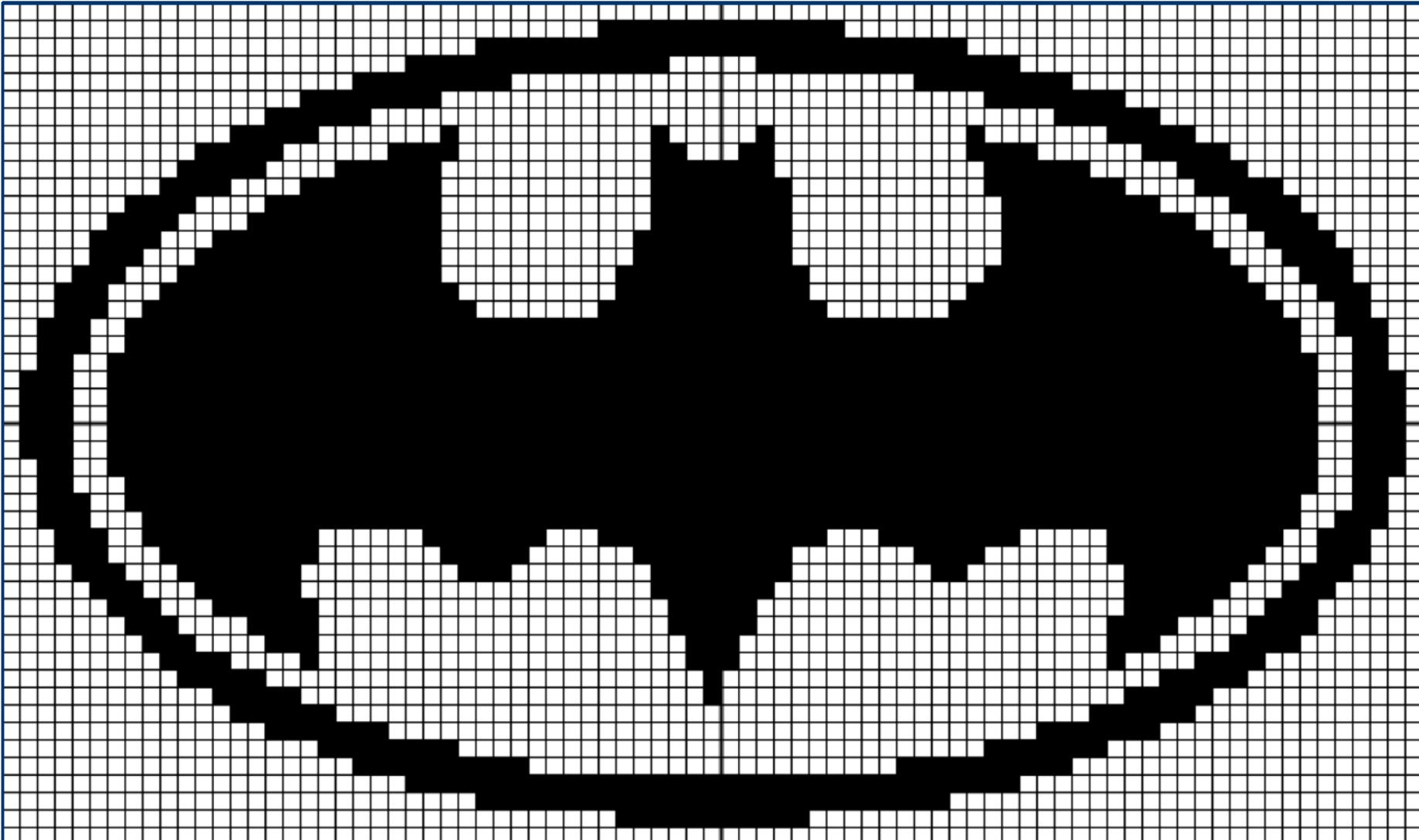
# Перфокарта



# Перфокарта

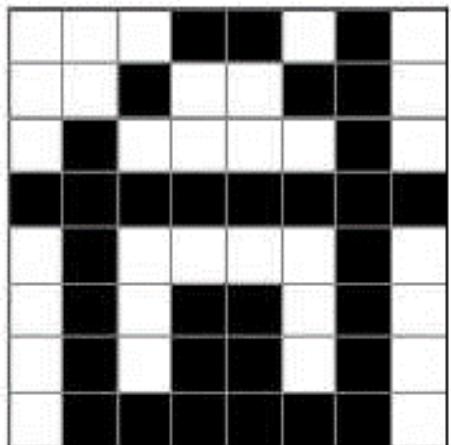
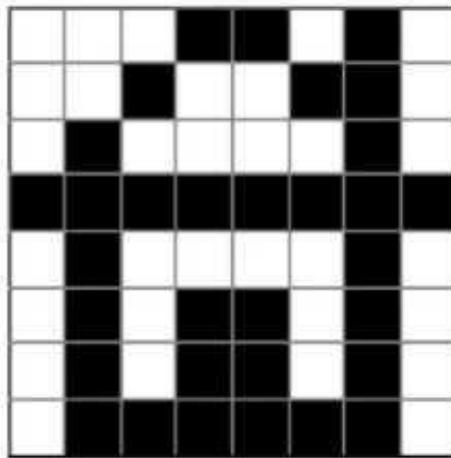
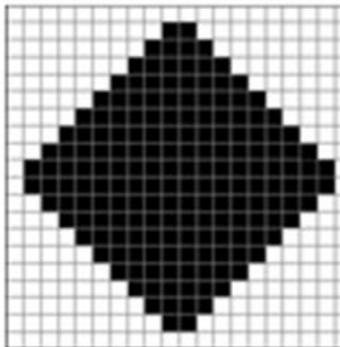
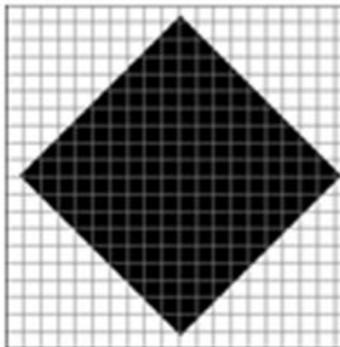


# Бинарное изображение



[ 0 , 1 ]

# Бинарное изображение



0 0 0 1 1 0 1 0  
1 A

|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |

|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |

1A  
26  
42  
FF  
42  
5A  
5A  
7E

| Образ символа |   |   |   |   |   |   |   | Коды строк матрицы |      |
|---------------|---|---|---|---|---|---|---|--------------------|------|
| 0             | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 0000 0000          | 00h  |
|               |   |   |   |   |   |   |   | 0000 1110          | 0E h |
|               |   |   |   |   |   |   |   | 0001 0010          | 12 h |
|               |   |   |   |   |   |   |   | 0010 0010          | 22 h |
|               |   |   |   |   |   |   |   | 0010 0010          | 22 h |
|               |   |   |   |   |   |   |   | 0011 1110          | 3E h |
|               |   |   |   |   |   |   |   | 0010 0010          | 22 h |
|               |   |   |   |   |   |   |   | 0010 0010          | 22 h |

Для первой строки получаем код  $1A_{16}$ .  
Для всего рисунка:  $1A2642FF425A5A7E_{16}$ .

# Кодирование текстовой информации

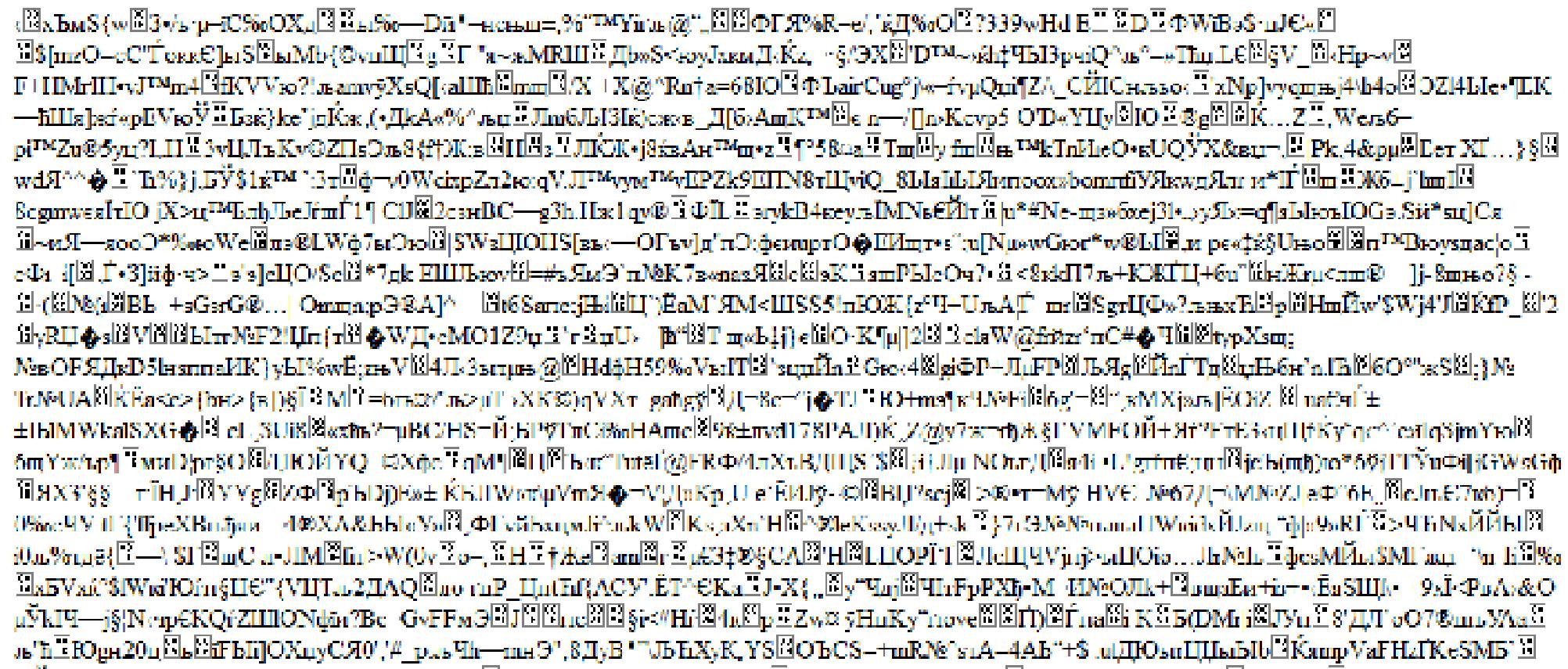
- Например буква «А» — это «11000000».
- Получается, что один символ — это 1 байт или 8 бит.
- При такой кодировке, путем нехитрых подсчетов можно посчитать, что мы можем зашифровать 256 символов.
- Для кодирования текстовой информации **данного количества символов более чем предостаточно.**
- Кодирование текстовой информации в компьютерных устройствах сводится к тому, что каждомуциальному символу присваивается уникальное десятичное значение от 0 и до 255 или его эквивалент в двоичной форме от 00000000 и до 11111111.
- Люди могут различать символы по их внешнему виду, а компьютерное устройство только по их уникальному коду.

# Кодирование текстовой информации и таблицы кодировок

- **Таблица кодировки** — это место, где прописано какому символу какой код относится.
- Все таблицы кодировки являются согласованными — это нужно, чтобы не возникало путаницы между документами, закодированными по одной таблице, но на разных устройствах.
- **На сегодняшний день существует множество таблиц кодировок.**
- **Наиболее популярные таблицы кодировки:** ASCII, MS-DOS, ISO, Windows, КОИ8, CP866, Mac, CP 1251, **Unicode (UTF-8, UTF-16, UTF-32)**, и другие. Из-за этого часто возникают проблемы с переносом текстовых документов между устройствами.
- Так получается, что если текстовая информация была закодирована по одной какой-то таблице, то и раскодирована она может быть только по этой таблице. Если попытаться раскодировать другой таблицей, то в результате получим только набор непонятных символов, но никак не читабельный текст.

# Проблема разных кодировок

- Проблема использования таких различных таблиц приводила к тому, что текст, написанный на одном компьютере, мог некорректно читаться на другом.
- Например:



亂碼 (jumbled text) from a Microsoft Word document. The text is a mix of various characters and symbols, including Chinese characters, English letters, and numbers, appearing as gibberish due to encoding problems.

Белорусско-Российский университет, Кафедра «Программное обеспечение информационных технологий»

# Таблица символов

Таблица символов

Шрифт: Arial Справка

|   |   |   |    |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|----|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ! | " | # | \$ | % | & | ' | ( | ) | * | + | , | - | . | / | 0 | 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8  | 9 | : | ; | < | = | > | ? | @ | A | B | C | D | E | F | G | H |
| I | J | K | L  | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | [ | \ |
| ] | ^ | _ | `  | a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p |
| q | r | s | t  | u | v | w | x | y | z | { |   | } | ~ | í | ¢ | £ | ¤ | ¥ |   |
| : | § | “ | ©  | ª | « | ¬ | - | ® | — | ° | ± | ² | ³ | · | µ | ¶ | · | , | ¹ |
| º | » | ¼ | ½  | ¾ | đ | À | Á | Â | Ã | Ä | Å | Æ | Ç | È | É | Ê | Ë | Ì | Í |
| Í | Ï | Ð | Ñ  | Ò | Ó | Ô | Õ | Ö | × | Ø | Ù | Ú | Û | Ü | Ý | þ | Þ | à | á |
| â | ã | ä | å  | æ | ç | è | é | ê | ë | í | í | î | ï | ð | ñ | ò | ó | ô | õ |
| ö | ÷ | ø | ù  | ú | û | ý | þ | ÿ | á | Á | ä | Ã | ă | Á | ă | ç | ć | ć |   |

Для копирования:  Выбрать Копировать

Дополнительные параметры

U+0021: Exclamation Mark

# Схема основной многоязычной плоскости Unicode

|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 0A | 0B | 0C | 0D | 0E | 0F |
| 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 1A | 1B | 1C | 1D | 1E | 1F |
| 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 2A | 2B | 2C | 2D | 2E | 2F |
| 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 3A | 3B | 3C | 3D | 3E | 3F |
| 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 4A | 4B | 4C | 4D | 4E | 4F |
| 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 5A | 5B | 5C | 5D | 5E | 5F |
| 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 6A | 6B | 6C | 6D | 6E | 6F |
| 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 7A | 7B | 7C | 7D | 7E | 7F |
| 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 8A | 8B | 8C | 8D | 8E | 8F |
| 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 9A | 9B | 9C | 9D | 9E | 9F |
| A0 | A1 | A2 | A3 | A4 | A5 | A6 | A7 | A8 | A9 | AA | AB | AC | AD | AE | AF |
| B0 | B1 | B2 | B3 | B4 | B5 | B6 | B7 | B8 | B9 | BA | BB | BC | BD | BE | BF |
| C0 | C1 | C2 | C3 | C4 | C5 | C6 | C7 | C8 | C9 | CA | CB | CC | CD | CE | CF |
| D0 | D1 | D2 | D3 | D4 | D5 | D6 | D7 | D8 | D9 | DA | DB | DC | DD | DE | DF |
| E0 | E1 | E2 | E3 | E4 | E5 | E6 | E7 | E8 | E9 | EA | EB | EC | ED | EE | EF |
| F0 | F1 | F2 | F3 | F4 | F5 | F6 | F7 | F8 | F9 | FA | FB | FC | FD | FE | FF |

- Латинская письменность
- Нелатинские европейские письменности
- Письменности Среднего Востока и Юго-Западной Азии
- Письменности Южной и Центральной Азии
- Письменности Африки
- Письменности Восточной Азии
- Письменности Юго-Восточной Азии
- Письменности Америки
- Письменности Индонезии и Океании
- Знаки
- Системы нотописи
- Идеограммы ККЯ
- Суррогатные пары UTF-16
- Область для частного использования

По состоянию на версию Юникода 14.0

# Кодовая таблица 0400 стандарта Unicode

|     | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | А | В | С | Д | Е | F |
|-----|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 040 | È | Ё | ҃ | Ѓ | Є | Ѕ | І | Ї | Ј | Љ | Њ | Ћ | Ќ | Ї | Ӯ | ҏ |
| 041 | А | Б | В | Г | Д | Е | Ж | З | И | Й | К | Л | М | Н | О | П |
| 042 | Р | С | Т | У | Ф | Х | Ц | Ч | Ш | Щ | Ђ | Ы | Ь | Э | Ю | Я |
| 043 | а | б | в | г | д | е | ж | з | и | й | к | л | м | н | о | п |
| 044 | р | с | т | у | ф | х | ц | ч | ш | щ | ъ | ы | ь | э | ю | я |
| 045 | è | ë | ђ | ѓ | € | ѕ | і | ї | ј | љ | њ | ћ | ќ | ѝ | ӻ | ҏ |
| 046 | Ѿ | ѿ | ҆ | Ҋ | ҂ | ҄ | ҈ | ҉ | Ҋ | Ҍ | ҈ | ҉ | Ҋ | ҈ | ҂ | ҃ |
| 047 | Ѱ | ѱ | Ѳ | ѳ | Ѷ | ѷ | ߵ | ߶ | Ѹ | ѹ | Ѻ | ѻ | ߴ | ߶ | ߵ | ߷ |
| 048 | Ҫ | ܲ | ܳ | ܲ | ܲ | ܲ | ܲ | ܲ | ܲ | ܲ | ܲ | ܲ | ܲ | ܲ | ܲ | ܲ |
| 049 | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ |
| 04A | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ |
| 04B | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ |
| 04C | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ |
| 04D | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ |
| 04E | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ |
| 04F | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ | Ӯ |

# Таблица символов Юникода

<https://unicode-table.com/ru/>

Таблица символов Юникода

Поиск символа

Юникод Эмоджи Наборы Инструменты Алфавиты HTML-мнемоники Alt-коды »

Выпускной вечер

Популярные наборы символов

Смотреть все >

Сердечки Красивые буквы Стрелки Символы для ников Звёздочки Кавычки Символы для VK Математические знаки

0 1 2 3 4 5 6 7 8 9 A B C D E F

|      |   |   |   |    |   |   |   |   |   |   |   |   |   |   |   |     |
|------|---|---|---|----|---|---|---|---|---|---|---|---|---|---|---|-----|
| 0000 | □ | □ | □ | □  | □ | □ | □ | □ | □ | □ | □ | □ | □ | □ | □ | □   |
| 0010 | □ | □ | □ | □  | □ | □ | □ | □ | □ | □ | □ | □ | □ | □ | □ | □   |
| 0020 | ! | " | # | \$ | % | & | ' | ( | ) | * | + | , | - | . | / |     |
| 0030 | 0 | 1 | 2 | 3  | 4 | 5 | 6 | 7 | 8 | 9 | : | ; | < | = | > | ?   |
| 0040 | @ | A | B | C  | D | E | F | G | H | I | J | K | L | M | N | O   |
| 0050 | P | Q | R | S  | T | U | V | W | X | Y | Z | [ | \ | ] | ^ | -   |
| 0060 | ‘ | a | b | c  | d | e | f | g | h | i | j | k | l | m | n | o   |
| 0070 | p | q | r | s  | t | u | v | w | x | y | z | { |   | } | ~ | DEL |
| 0080 | □ | □ | □ | □  | □ | □ | □ | □ | □ | □ | □ | □ | □ | □ | □ | □   |

Основная латиница

Открыть на отдельной странице

Диапазон: 0000-007F  
Количество символов: 128  
Тип: алфавит  
Языки: английский, немецкий, французский, итальянский, польский



# Например: буква «К» в Unicode U+041A

## Техническая информация

|                    |                            |
|--------------------|----------------------------|
| Название в Юникоде | Cyrillic Capital Letter Ka |
| Номер в Юникоде    | <b>U+041A</b>              |
| HTML-код           | &#1050;                    |
| CSS-код            | \041A                      |
| Раздел             | Кириллица                  |
| Строчная           | к                          |
| Версия Юникода:    | 1.1 (1993)                 |

## Свойства

|                                     |           |
|-------------------------------------|-----------|
| Версия                              | 1.1       |
| Блок                                | Кириллица |
| Тип парной зеркальной скобки (bidi) | Нет       |
| Композиционное исключение           | Нет       |
| Изменение регистра                  | 043A      |
| Простое изменение регистра          | 043A      |

## Кодировка

| Кодировка | hex         | dec (bytes) | dec       | binary                              |
|-----------|-------------|-------------|-----------|-------------------------------------|
| UTF-8     | D0 9A       | 208 154     | 53402     | 11010000 10011010                   |
| UTF-16BE  | 04 1A       | 4 26        | 1050      | 00000100 00011010                   |
| UTF-16LE  | 1A 04       | 26 4        | 6660      | 00011010 00000100                   |
| UTF-32BE  | 00 00 04 1A | 0 0 4 26    | 1050      | 00000000 00000000 00000100 00011010 |
| UTF-32LE  | 1A 04 00 00 | 26 4 0 0    | 436469760 | 00011010 00000100 00000000 00000000 |



Юникод

Эмоджи

Наборы

Инструменты

Алфавиты

HTML-мнемоники

Alt-коды

»



Выпускной вечер



Главная &gt; Наборы &gt; Смайлики-эмоджи «Лица»

## Смайлики-эмоджи «Лица»



U+1F600



U+1F603



U+1F604



U+1F642



U+1F643



U+1F609



U+1F618



U+1F617



U+263A



**Улыбающееся лицо с открытым ртом и смеющимися глазами >**

Номер в Юникоде: U+1F604

HTML-код: &amp;#128516;

Копировать



U+1F92A



U+1F61D



U+1F911



U+1F917



U+1F92D



U+1F92B



U+1F914



U+1F910

Смотрите также



Символы для ников



Символы для VK



Красивые буквы



Горячие символы



Сердечки



Забавные буквы



Символы для Facebook



Коронавирус



Дорожные знаки



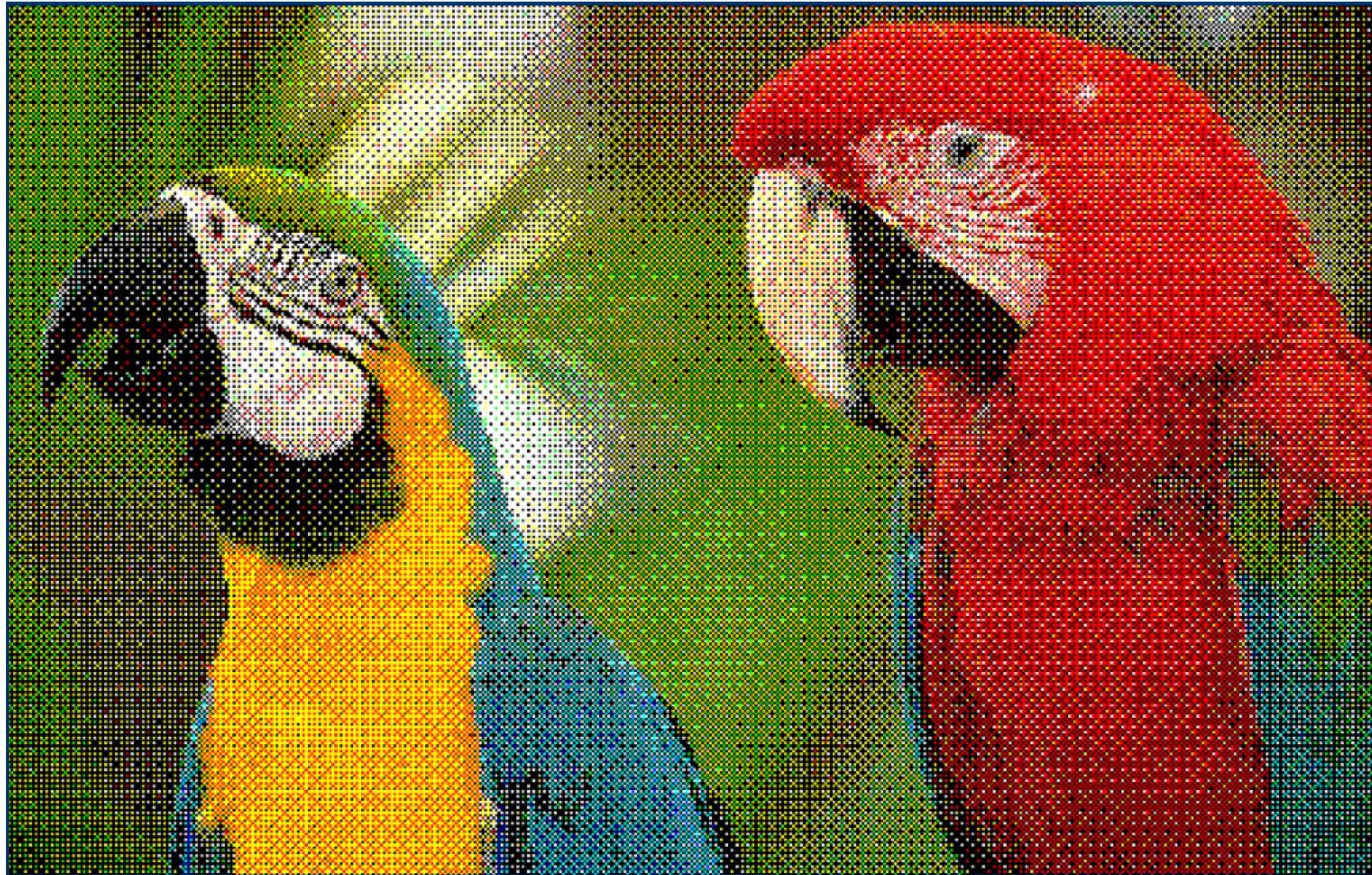
Топ-50 Эмоджи

# Изображение в оттенках серого



[ 0 : 255 ]

# Цветное изображение



( [ 0:255 ] , [ 0:255 ] , [ 0:255 ] )

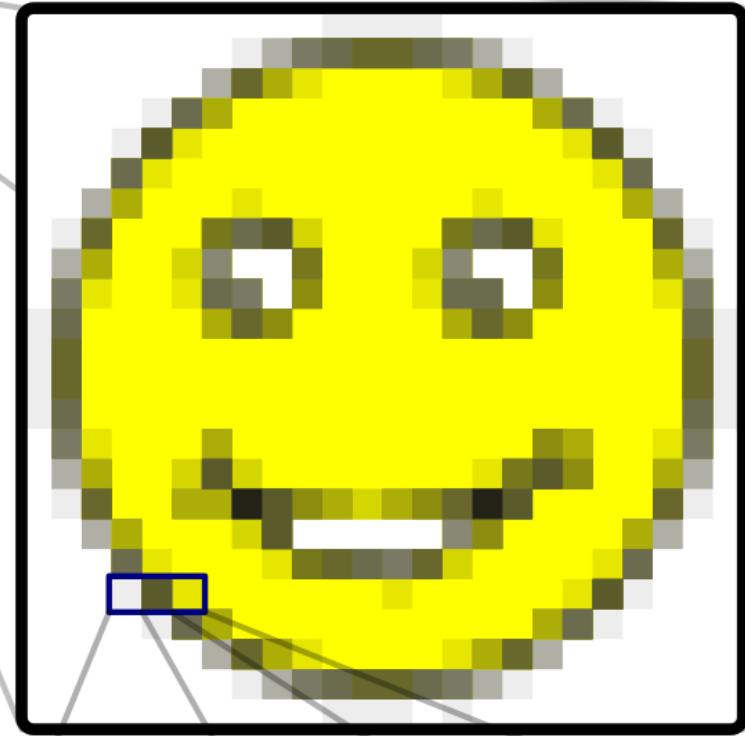
# Размер файла с изображением в зависимости от количества кодируемых цветов

| <b>Разрешение</b> | <b>16 цветов</b> | <b>256 цветов</b> | <b>65 536 цветов</b> | <b>16 777 216 цветов</b> |
|-------------------|------------------|-------------------|----------------------|--------------------------|
| 640x480           | 150 Кбайт        | 300 Кбайт         | 600 Кбайт            | 900 Кбайт                |
| 800x600           | 234,4 Кбайт      | 468,8 Кбайт       | 937,5 Кбайт          | 1,4 Мбайт                |
| 1024x768          | 384 Кбайт        | 768 Кбайт         | 1,5 Мбайт            | 2,25 Мбайт               |
| 1280x1024         | 640 Кбайт        | 1,25 Мбайт        | 2,5 Мбайт            | 3,75 Мбайт               |

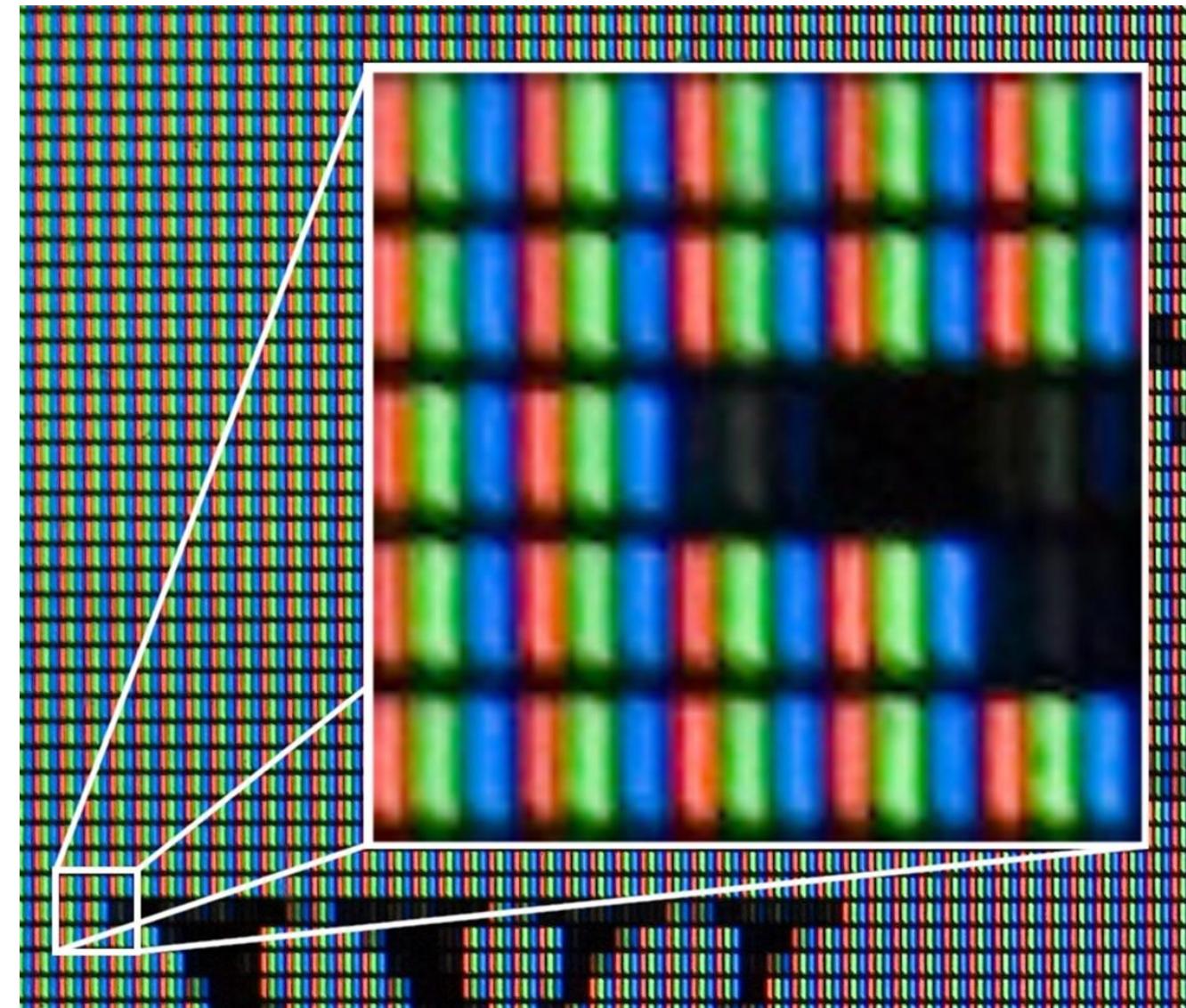
# RGB цвета и их кодирование

| Color             | HTML / CSS Name   | Hex Code<br>#RRGGBB | Decimal Code<br>(R,G,B) |
|-------------------|-------------------|---------------------|-------------------------|
| Black             | Black             | #000000             | (0,0,0)                 |
|                   | White             | #FFFFFF             | (255,255,255)           |
| Red               | Red               | #FF0000             | (255,0,0)               |
| Lime              | Lime              | #00FF00             | (0,255,0)               |
| Blue              | Blue              | #0000FF             | (0,0,255)               |
| Yellow            | Yellow            | #FFFF00             | (255,255,0)             |
| Cyan / Aqua       | Cyan / Aqua       | #00FFFF             | (0,255,255)             |
| Magenta / Fuchsia | Magenta / Fuchsia | #FF00FF             | (255,0,255)             |
| Silver            | Silver            | #C0C0C0             | (192,192,192)           |
| Gray              | Gray              | #808080             | (128,128,128)           |
| Maroon            | Maroon            | #800000             | (128,0,0)               |
| Olive             | Olive             | #808000             | (128,128,0)             |
| Green             | Green             | #008000             | (0,128,0)               |
| Purple            | Purple            | #800080             | (128,0,128)             |
| Teal              | Teal              | #008080             | (0,128,128)             |
| Navy              | Navy              | #000080             | (0,0,128)               |

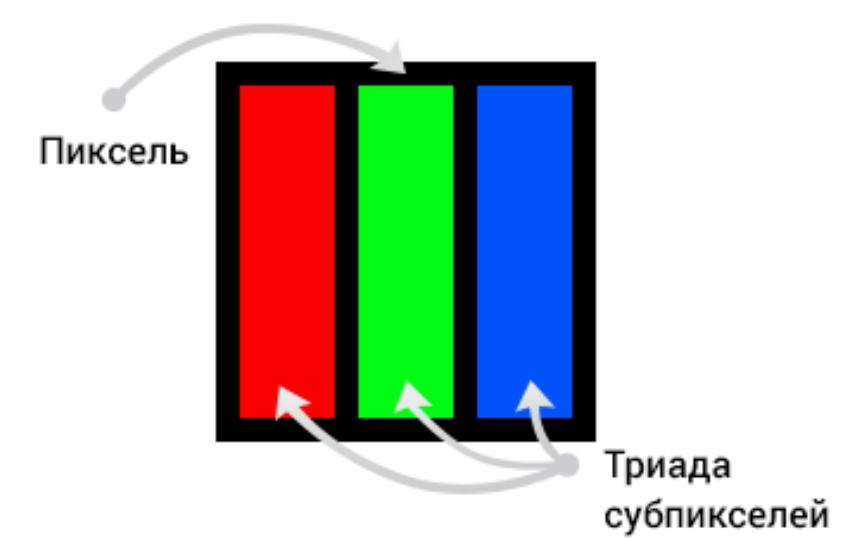
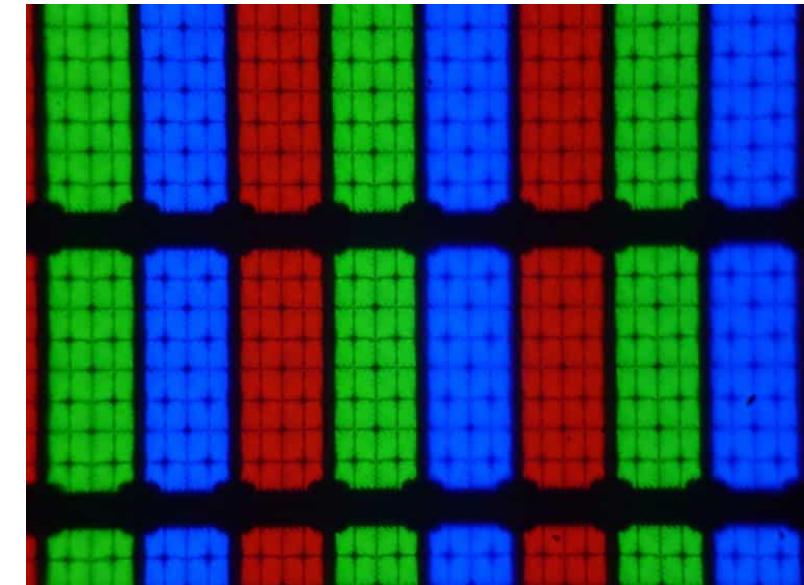
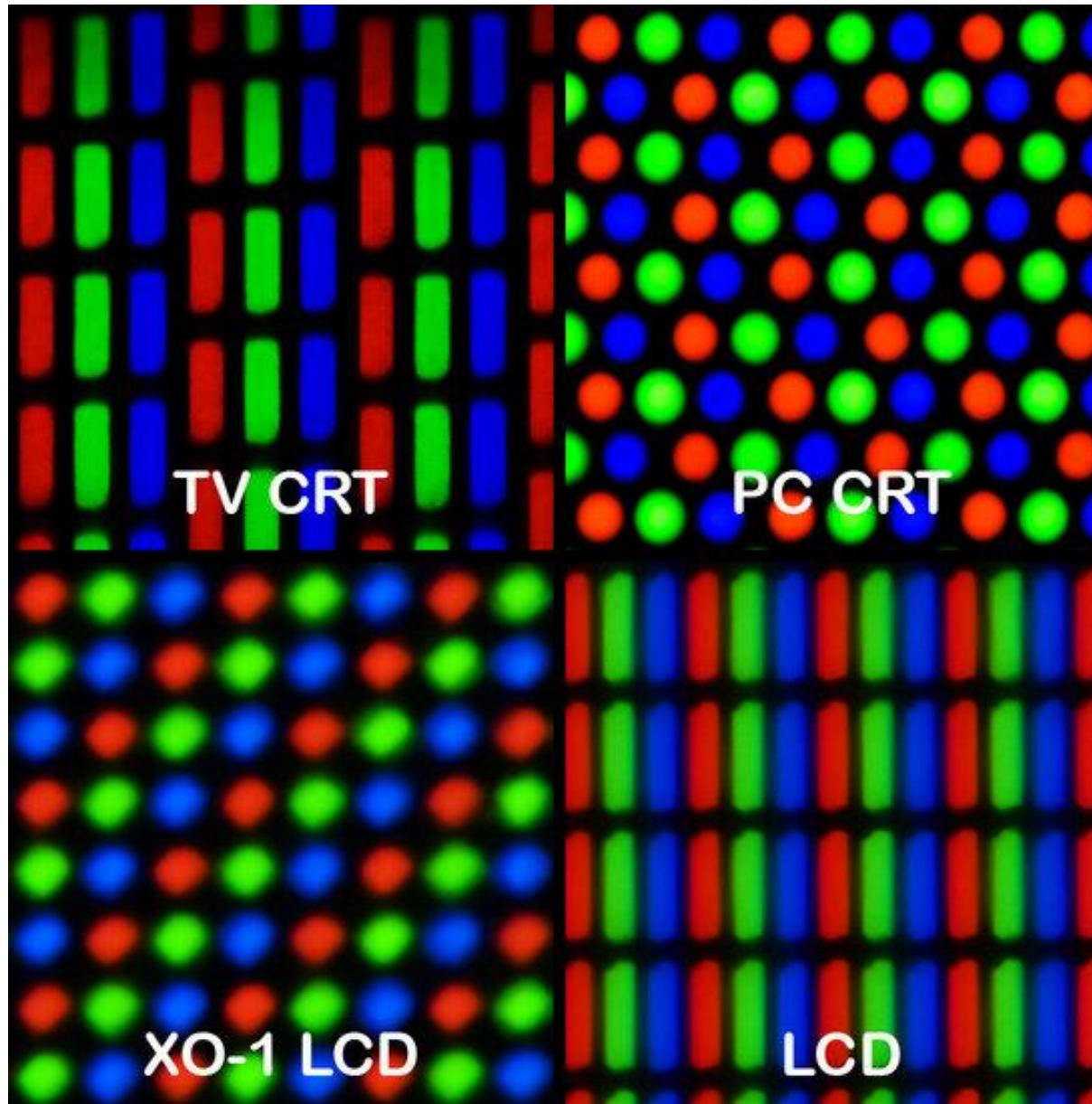
# Цветное изображение



|       |       |       |
|-------|-------|-------|
| R 93% | R 35% | R 90% |
| G 93% | G 35% | G 90% |
| B 93% | B 16% | B 0%  |



# Отображение цвета на мониторе

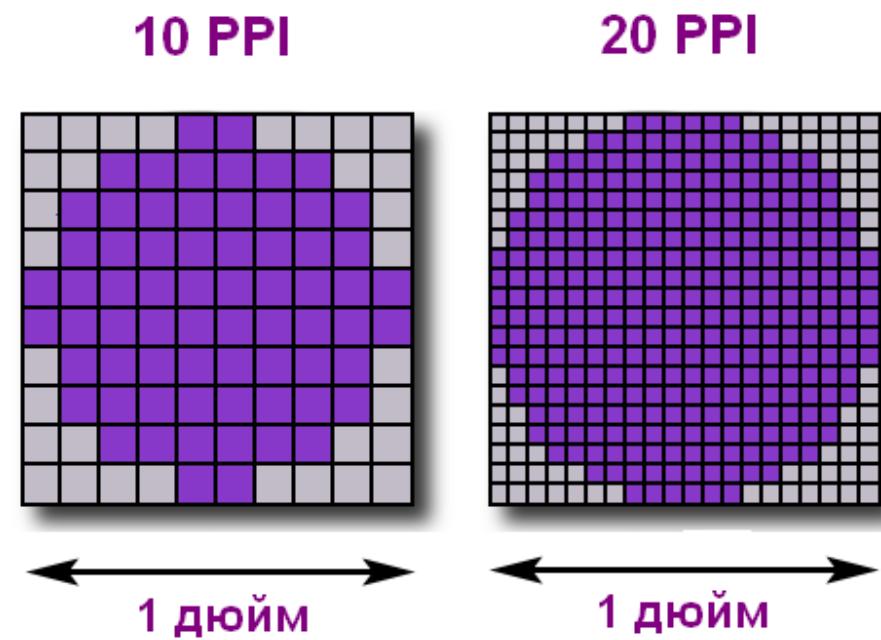


# Пикセル

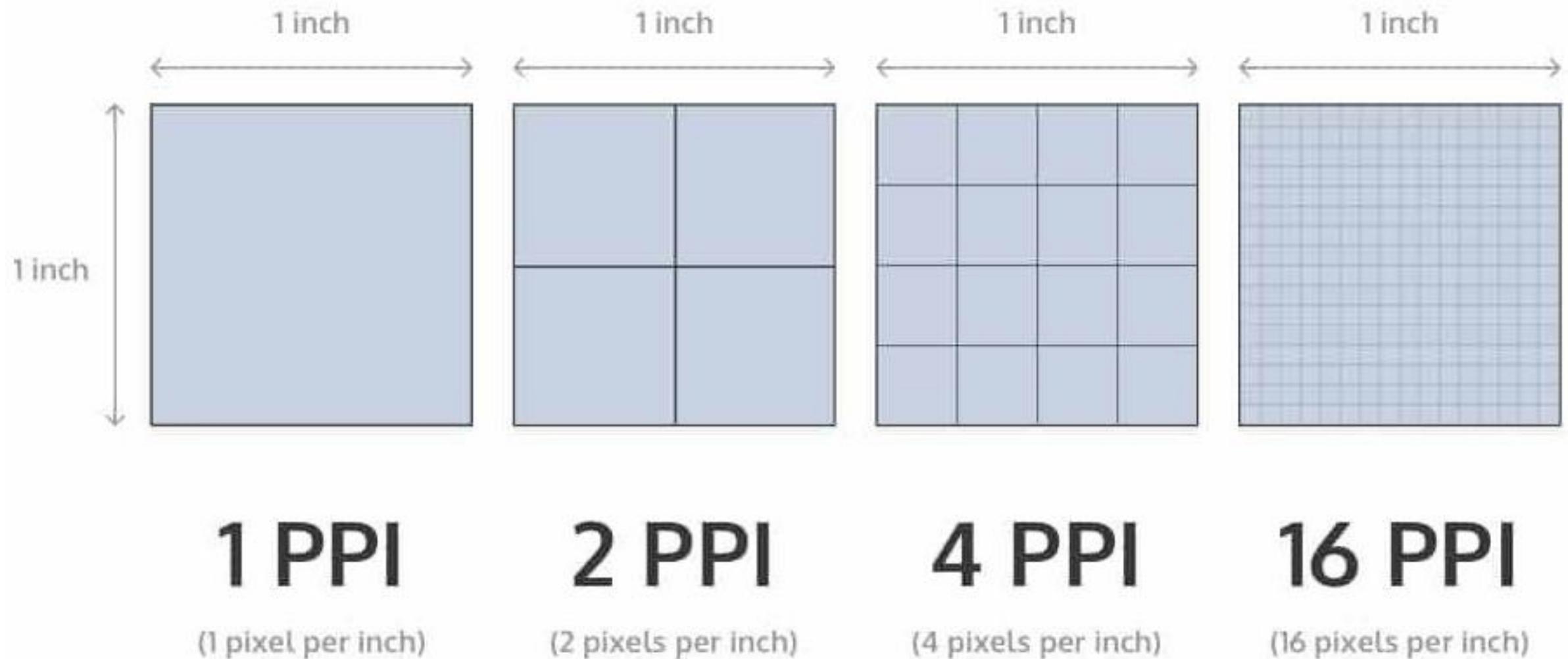
- **Пикセル** (англ. pixel = picture element, элемент рисунка) – это наименьший элемент рисунка, для которого можно задать свой цвет.
- **Чем больше пикселей на единицу площади содержит изображение, тем более оно детально.**
- **Качество растрового изображения** определяется двумя основными параметрами — **разрешением** (количеством точек по горизонтали и вертикали) **и используемой палитрой цветов** (набором цветов, при этом каждая точка может иметь количество цветов, равное 16, 256, 65 536 и т.д.).
- Разрешение задается указанием числа точек по горизонтали на число точек по вертикали. Например, 1024 на 768 точек.

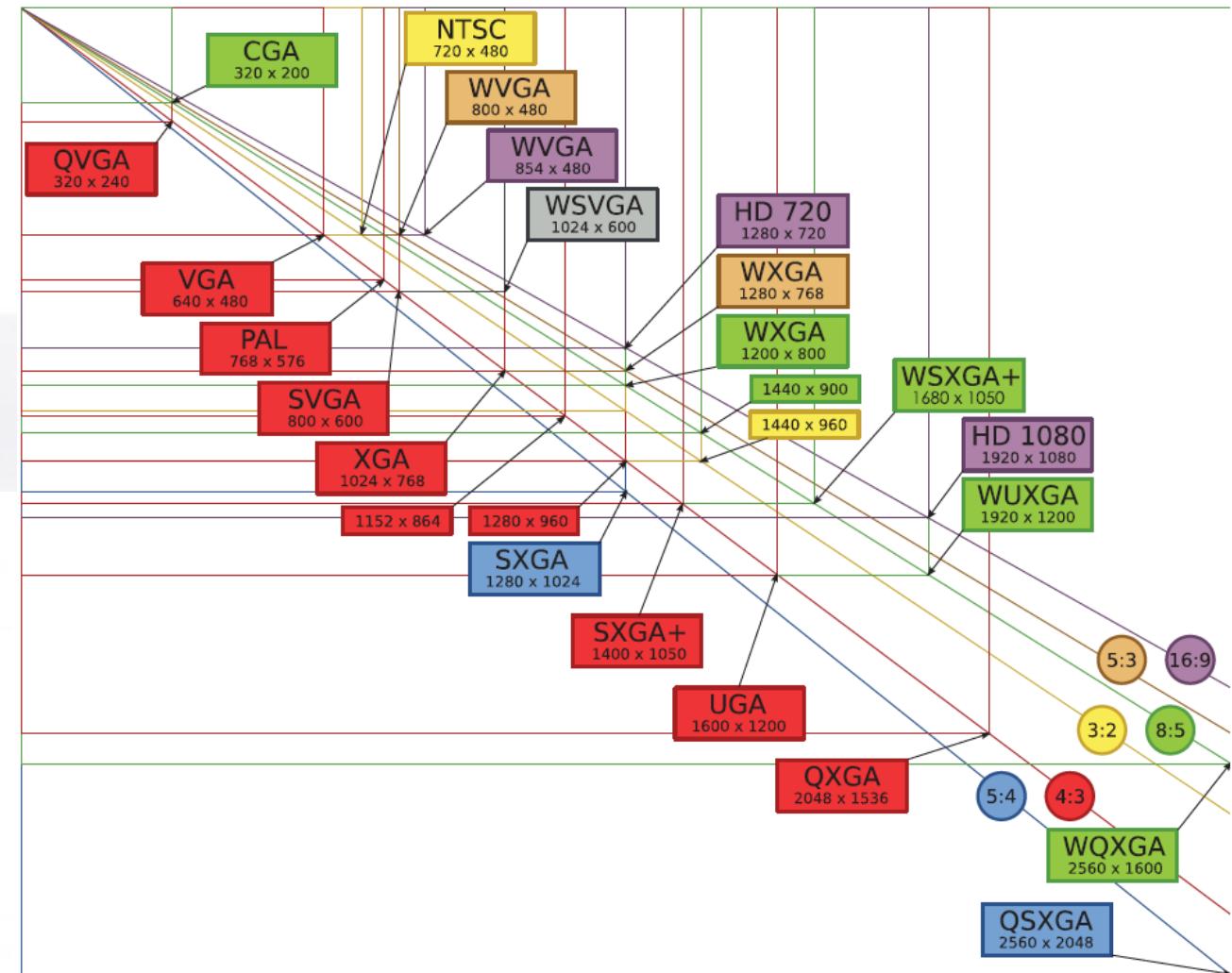
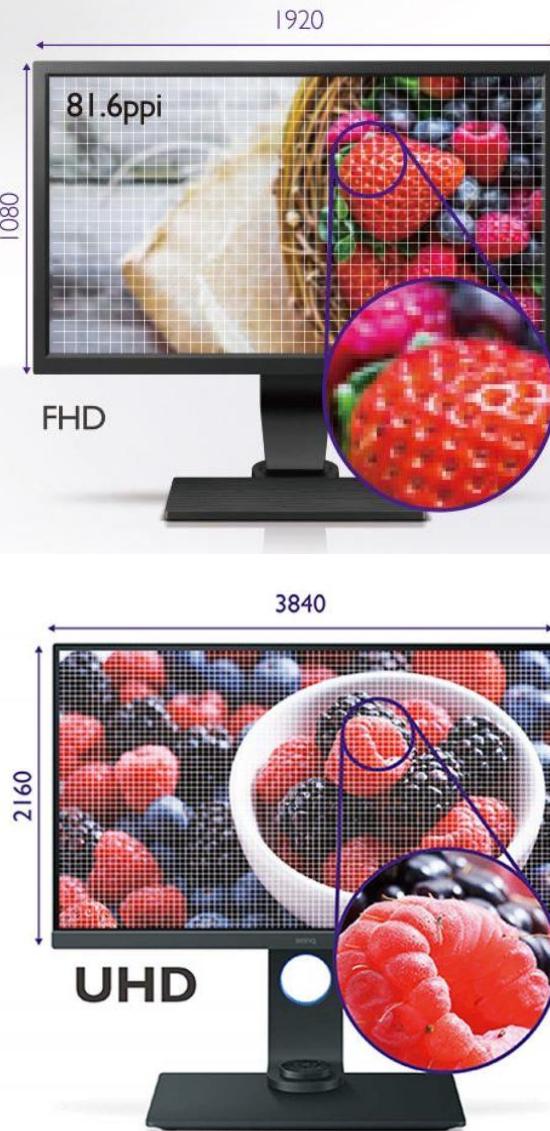
# Разрешение - dpi, ppi

- **Разрешение** – это количество пикселей, приходящихся на единицу линейного размера изображения.
- Для обозначения разрешающей способности различных процессов преобразования изображений (сканирование, печать, растеризация и т. п.) чаще всего используют следующие термины:
  - **dpi** (англ. dots per inch)  
**количество точек на дюйм.**
  - **ppi** (англ. pixels per inch)  
**количество (плотность)**  
**пикселей на дюйм.**



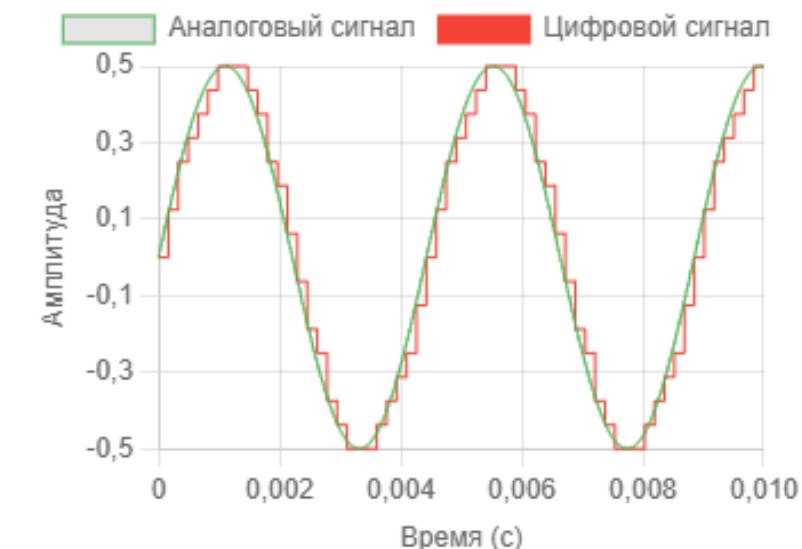
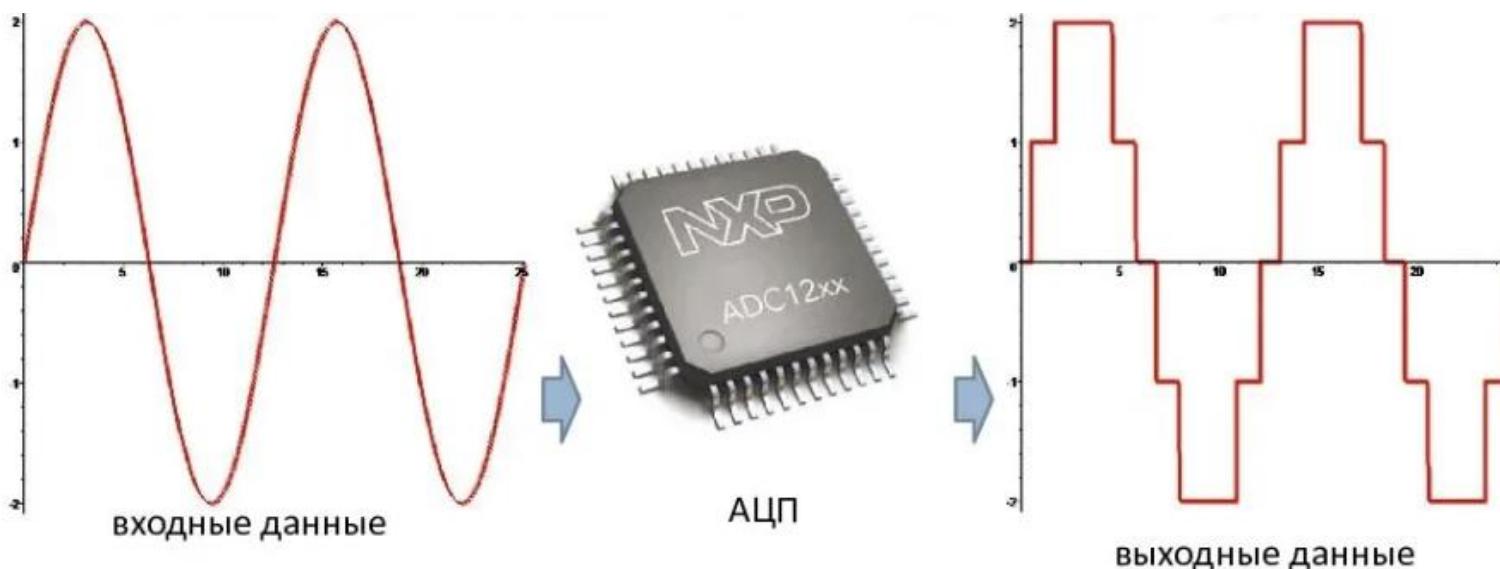
# PPI (pixels per inch) количество (плотность) пикселей на дюйм



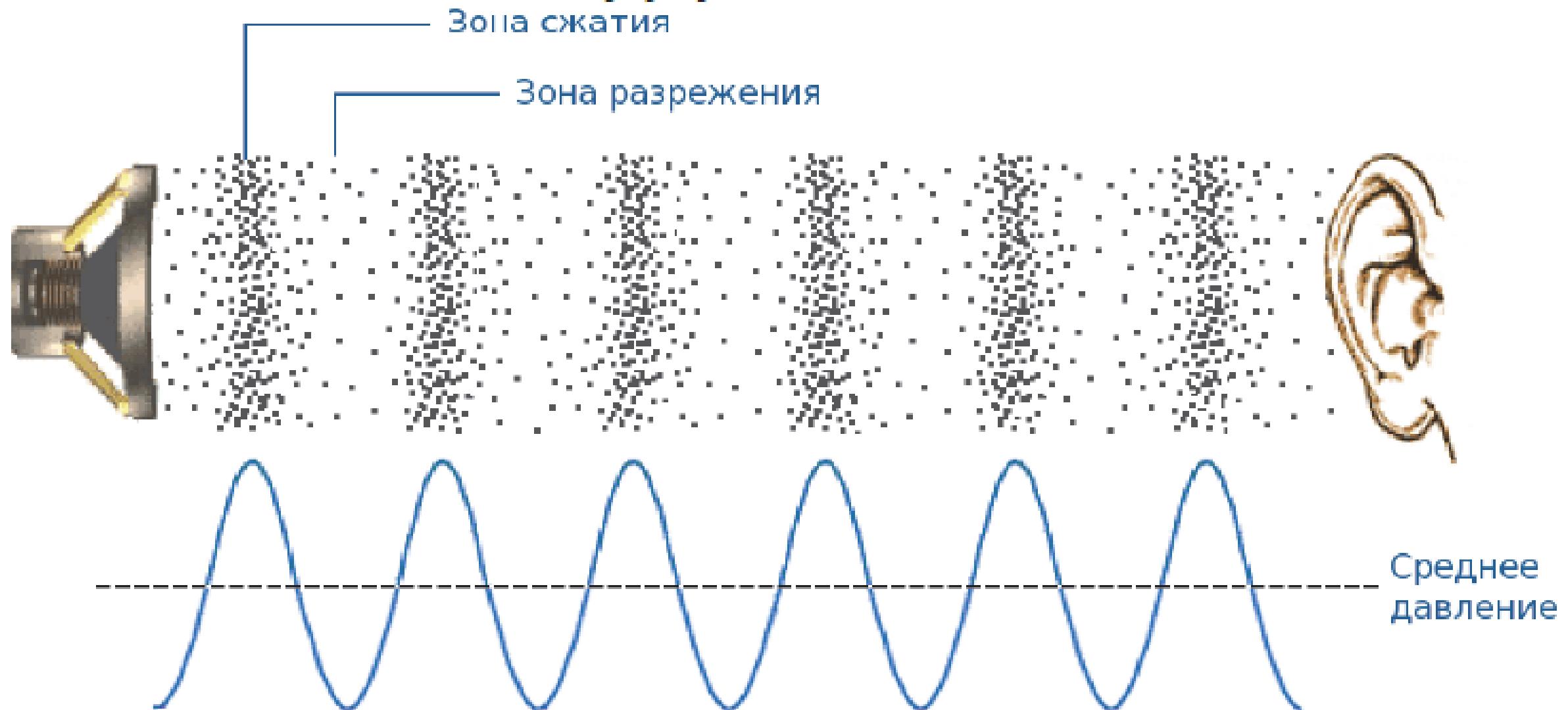


# Звук

- Звуковой (аудио) сигнал имеет аналоговую природу.
- Для того чтобы преобразовать его в дискретную форму используют специальный блок, входящий в состав звуковой карты компьютера, **АЦП (аналого-цифровой преобразователь)**. Основной принцип его работы заключается в том, что интенсивность звукового сигнала фиксируется не непрерывно, а периодически, в определенные моменты времени.



# Кодирование звуковой информации



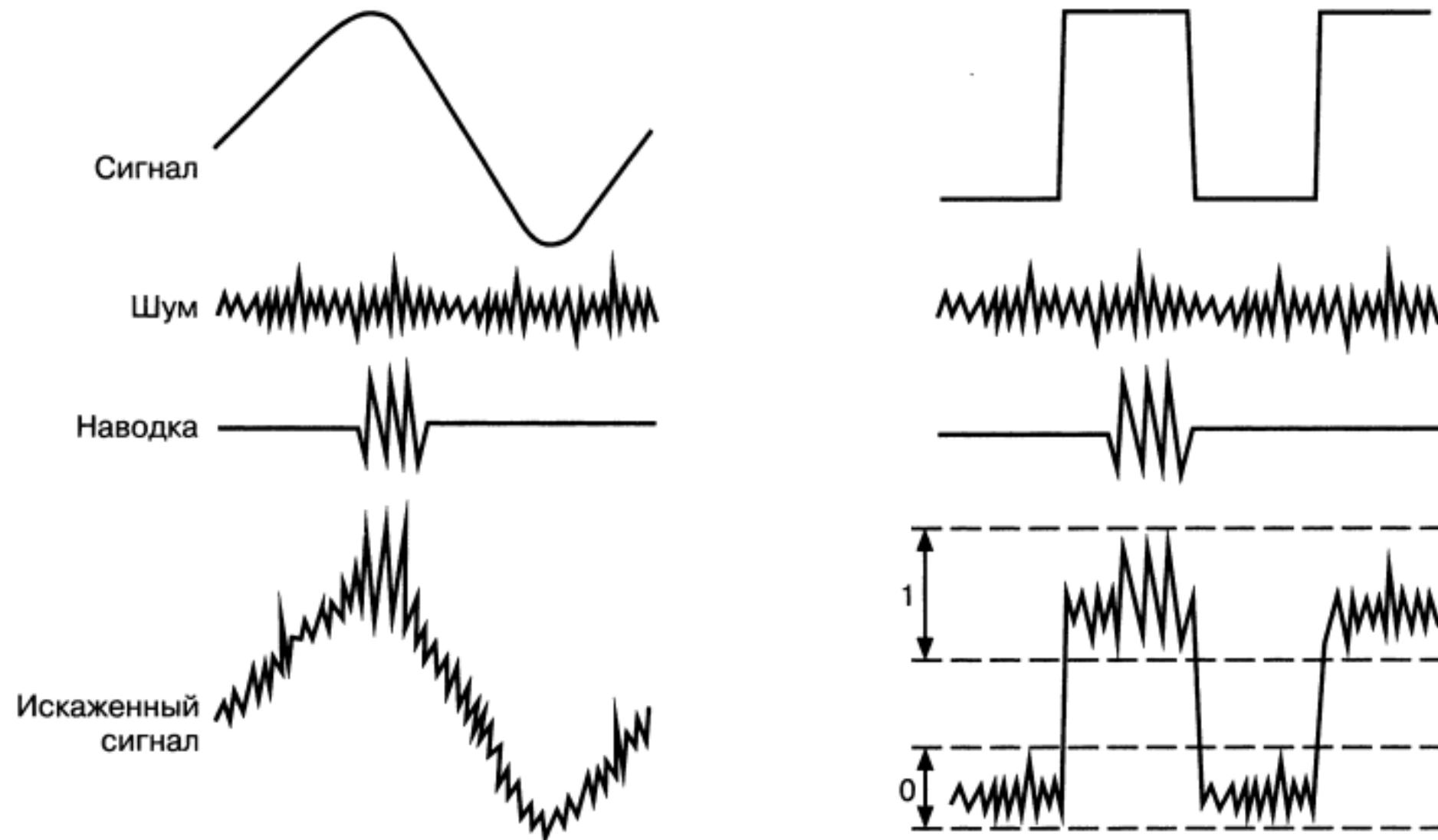
# Кодирование звуковой информации



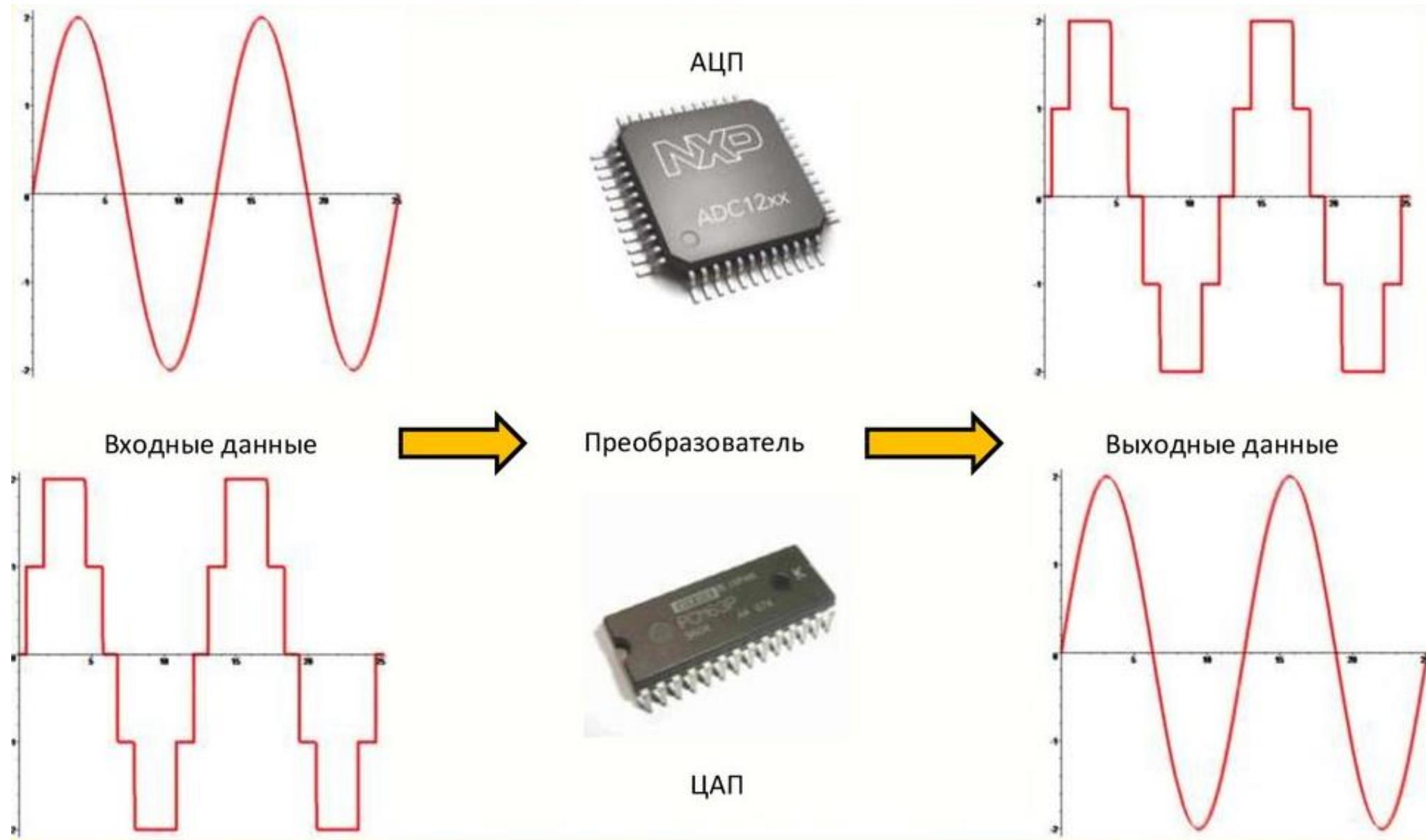
**АЦП – аналого-цифровое преобразование - оцифровка сигнала**

**ЦАП – цифро-аналоговое преобразование – преобразование в аналог**

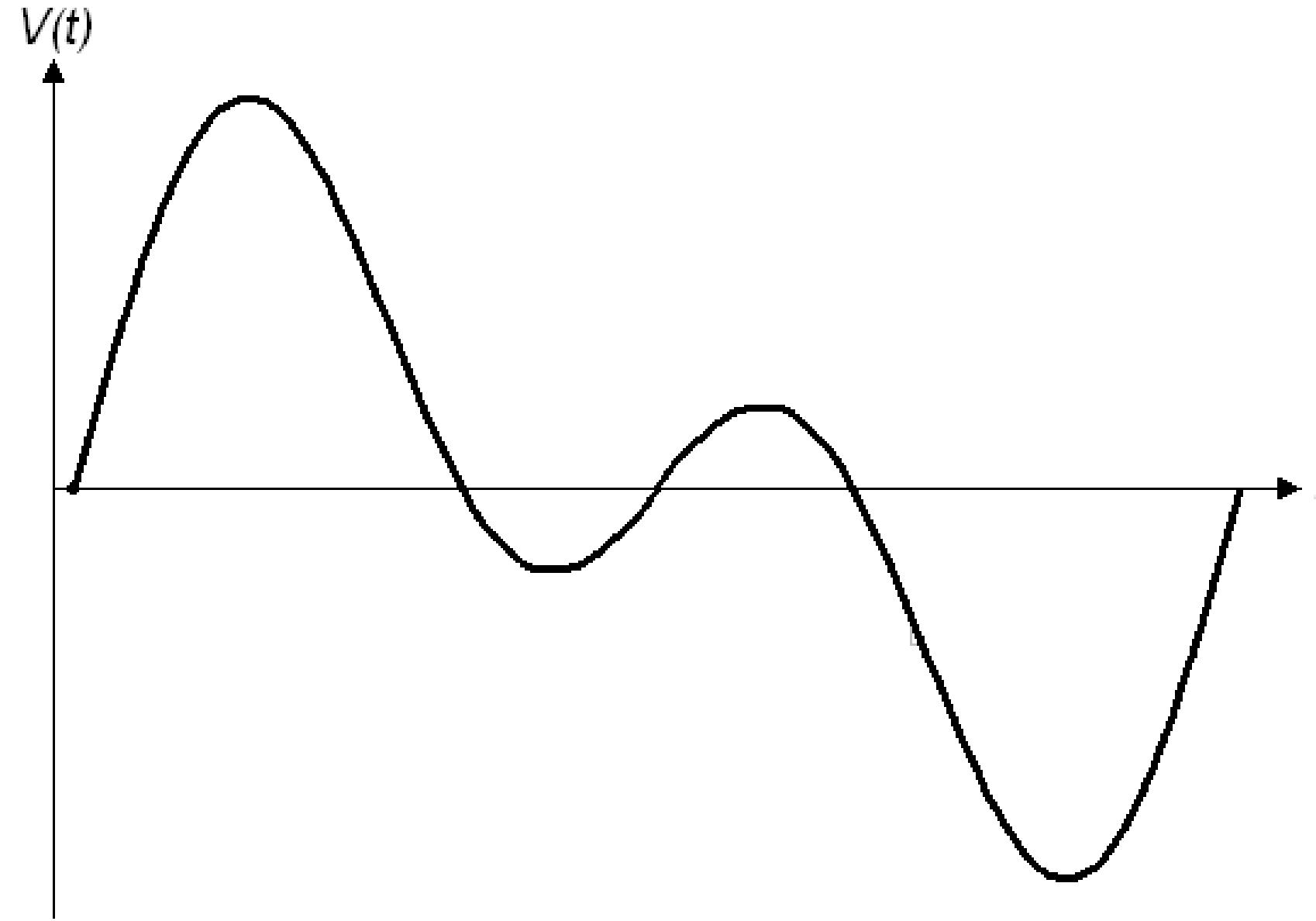
# Кодирование звуковой информации



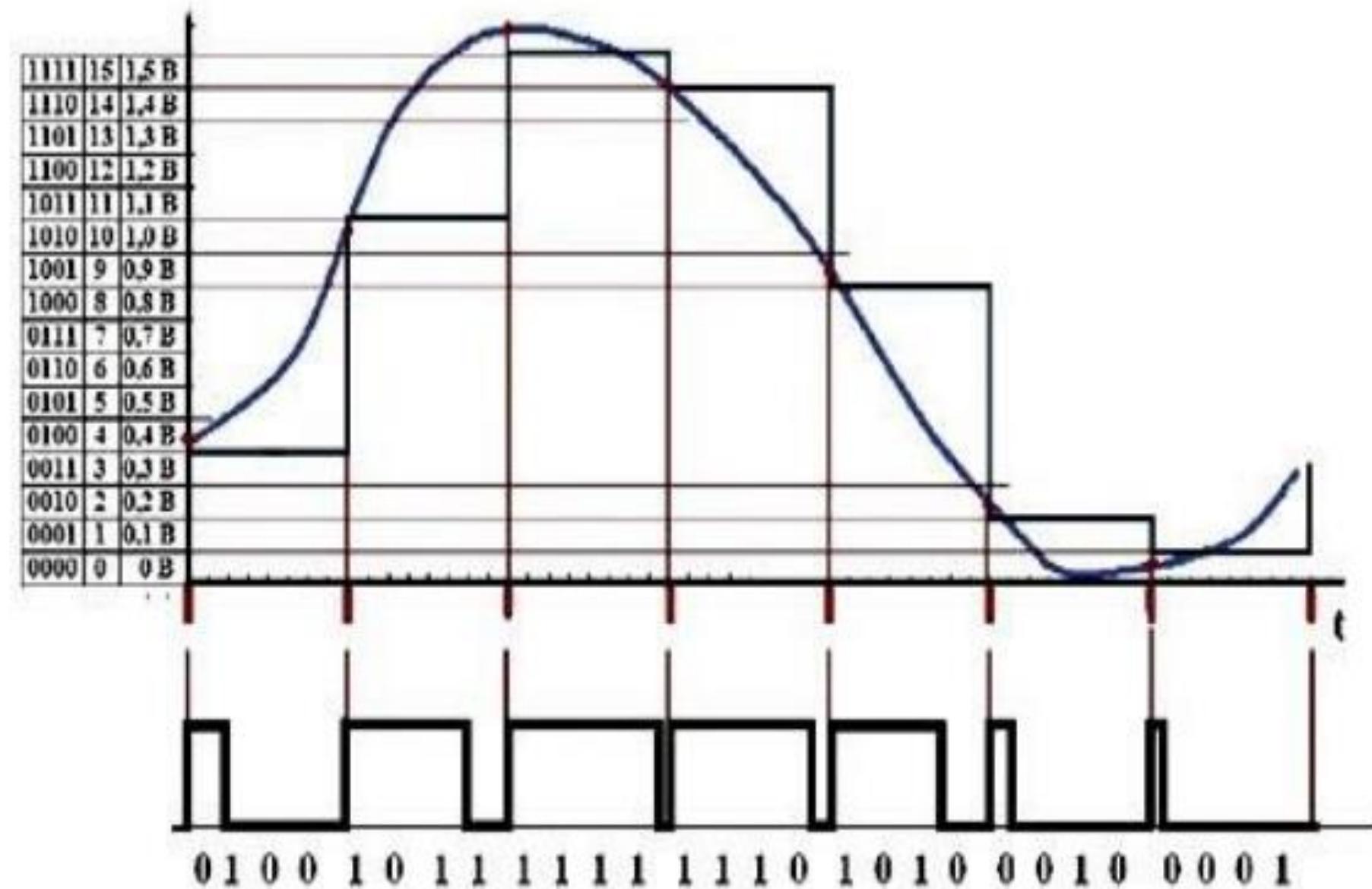
# Кодирование звуковой информации



# Кодирование звуковой информации

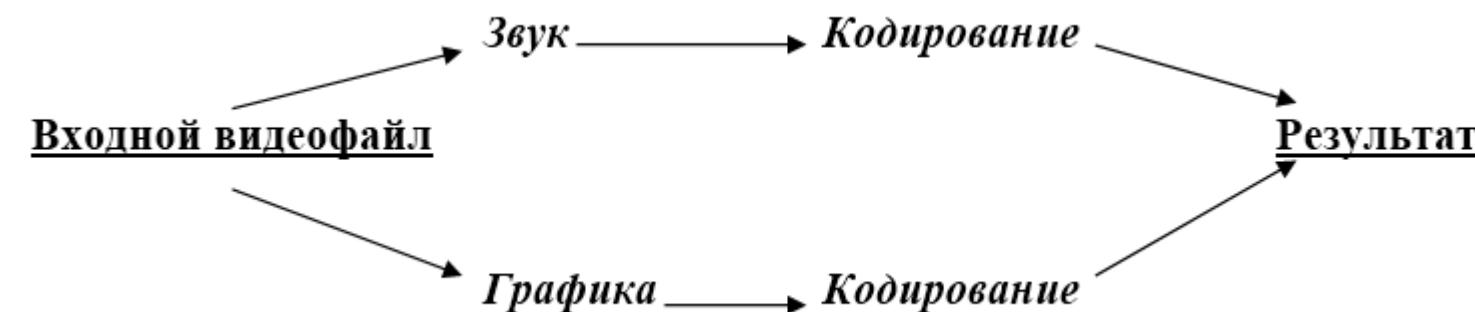


# Кодирование звуковой информации



# Кодирование видеоинформации

- Видеоинформация включает последовательность кадров и звуковое сопровождение, поэтому кодирование видеоинформации еще более сложная проблема, чем кодирование звуковой информации, так как нужно позаботиться не только о дискретизации непрерывных движений, но и о синхронизации изображения со звуковым сопровождением.
- В настоящее время для этого используется множество различных форматов, например один из форматов это AVI (Audio-Video Interleaved – чередующееся аудио и видео).



# Размер видеофайла в формате AVI

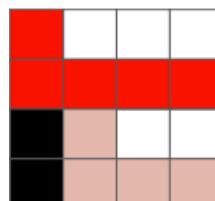
- Объем видеофайла примерно равен произведению количества информации в каждом кадре на число кадров. Число кадров вычисляется как произведение длительности видеоклипа  $\Delta t$  на скорость кадров  $v$ , то есть их количество в 1 с:

$$V = N * M * C * v * \Delta t.$$

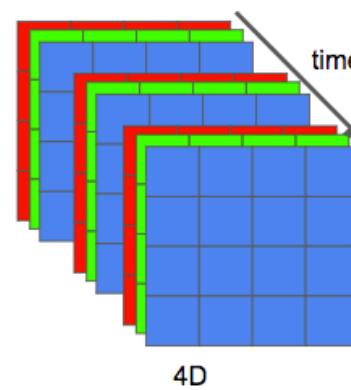
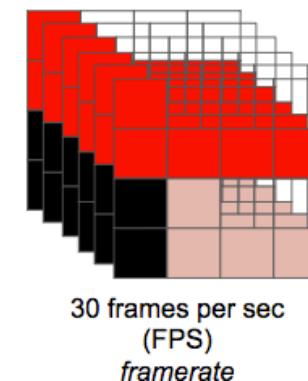
- При разрешении **800 × 600 точек**, разрядности цвета  $C = 16$ , скорости кадров  $v = 25$  кадров/с, видеоклип **длительностью 30 с будет иметь объем**:

$$\begin{aligned}800 * 600 * 16 * 25 * 30 &= 5\ 760\ 000\ 000 \text{ бит} = 576 * 10^7 \text{ бит} = \\&= 720\ 000\ 000 \text{ байт} = 72 * 10^7 \text{ байт} = \\&= 703\ 125 \text{ Кбайт} = 686,645 \text{ Мбайт.}\end{aligned}$$

- Это много для такого короткого видеофрагмента, поэтому на практике применяются различные способы компрессии, то есть сжатия звуковых и видео кодов.

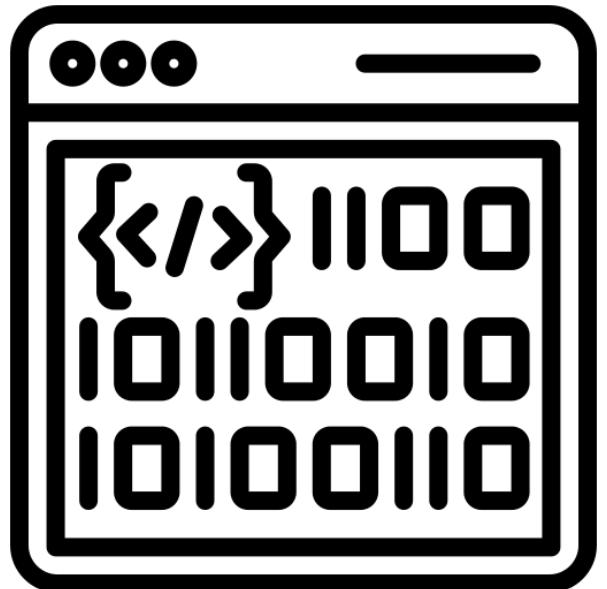


a single frame

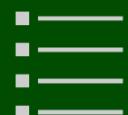


Количество бит в секунду, необходимое для показа видео, является его скоростью передачи — **битрейтом**.

**битрейт** = ширина \* высота \* бит глубина \* кадров в секунду



# Системы счисления. Преобразование чисел



# Системы счисления

- Для удобства работы ПК используют не только двоичную (2) и десятичную (10) системы счисления, но и 8, 16, 32, 64 системы счисления. Каждая применяется для своих задач.
- В ПК информация постоянно перекодируется из двоичной в 8, 10, 16, 32, 64 системы счисления и обратно.

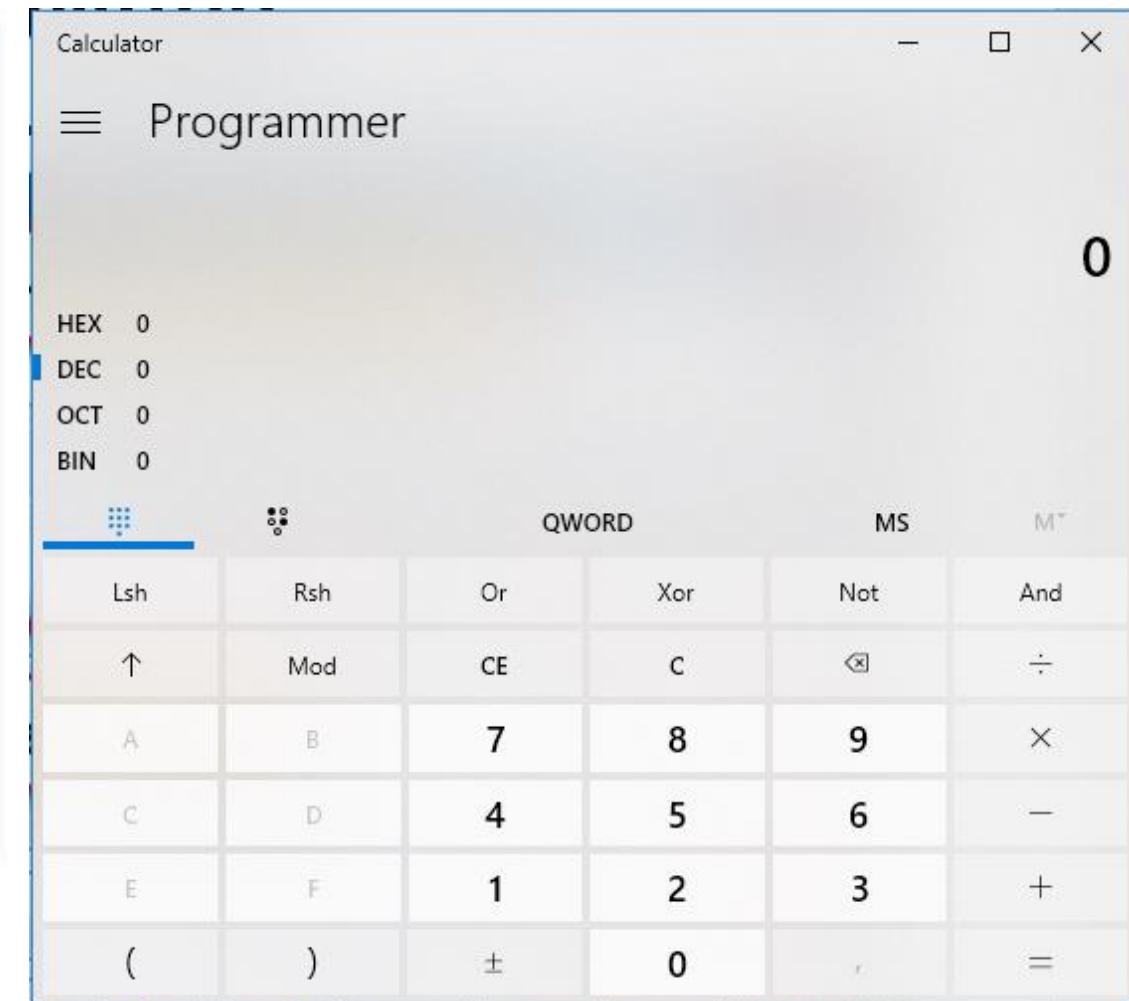
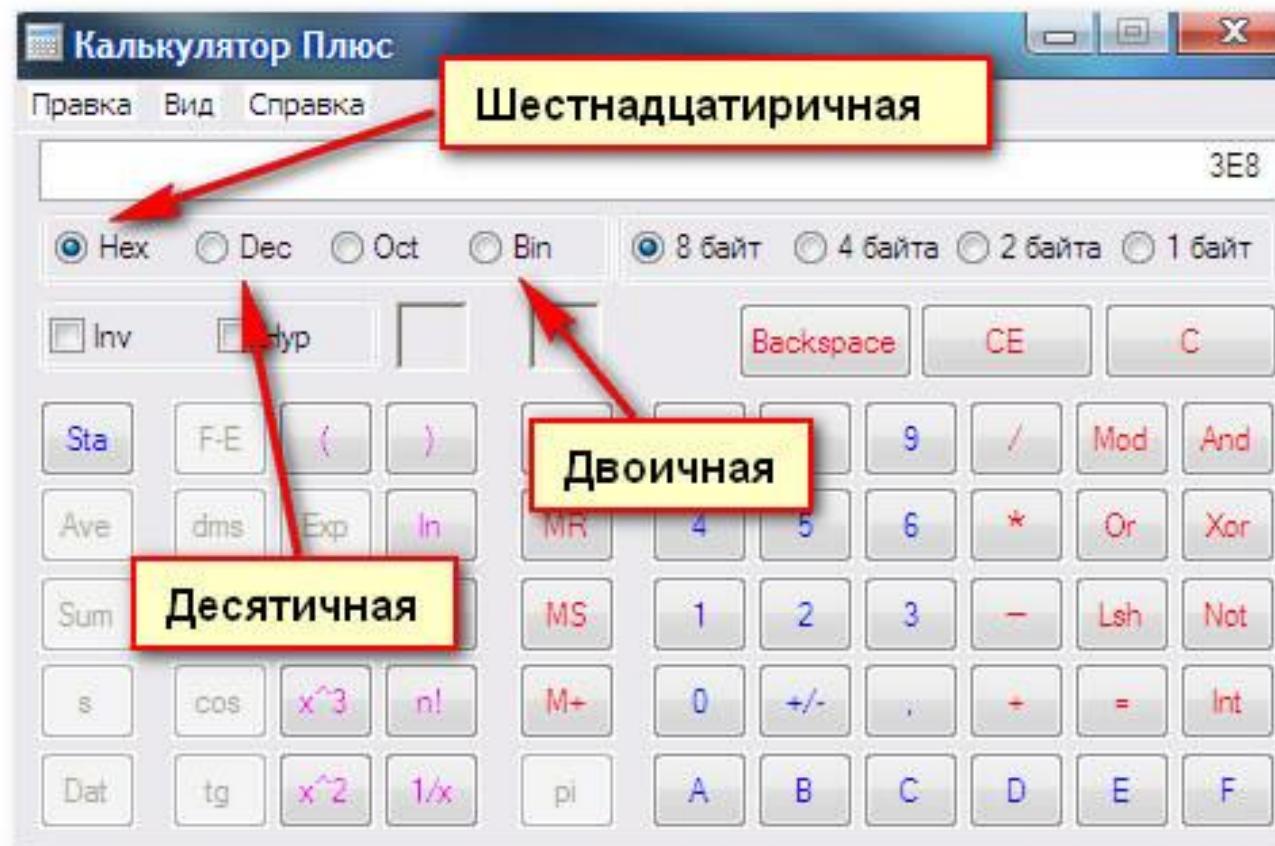
**Десятичная** – все вы её прекрасно знаете и изучали с первого класса. В качестве алфавита здесь используются цифры от 0 до 9.

**Двоичная** – счисление введенное в семнадцатом веке великим математиком Вильгельмом Лейбницем. В данный момент нашло широкое применение в персональных компьютерах и цифровой технике. Состоит всего из двух знаков 0 и 1.

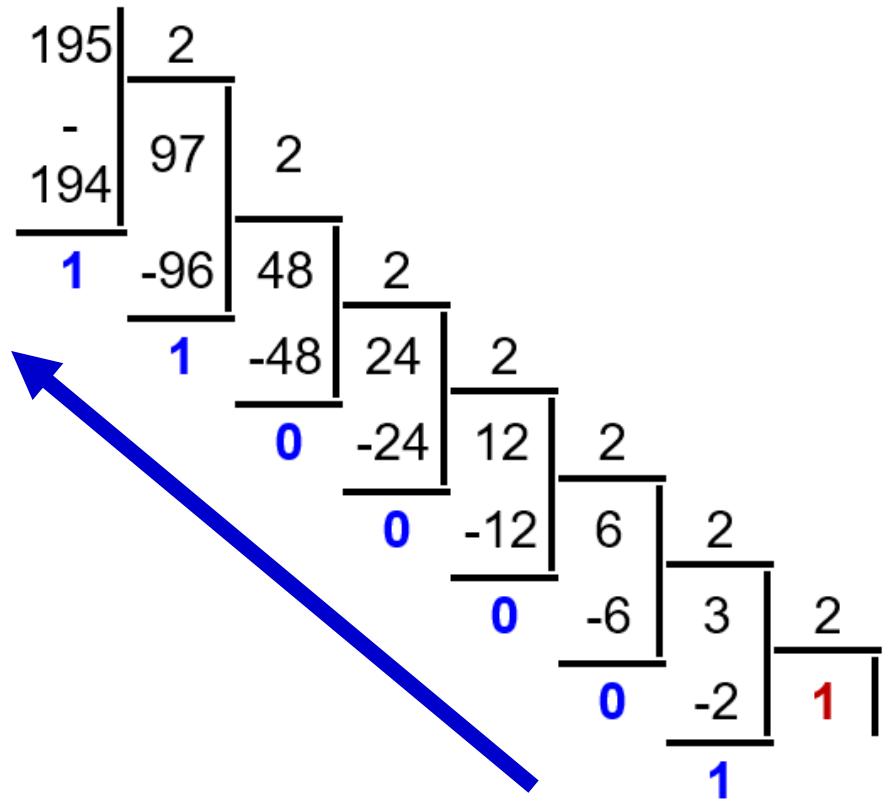
**Восьмеричная** – счисление, широко применяющееся в высокоуровневых языках программирования (например, Java и Python) и разработке цифровой аппаратуры. Свою популярность заслужила из-за легкого перевода в цифровой (двоичный) код. Состоит из цифр от 0 до 7.

**Шестнадцатеричная** – счисление используется в низкоуровневых языках программирования (язык Assembler'a) в информатике. Также в 16-ом виде представляются символы в стандарте Юникода. В её алфавит входят числа от 0 до 9 и латинские буквы A, B, C, D, E и F.

# Стандартный калькулятор Windows



# Перевод из 10 в 2 и из 2 в 10 систему счисления



$$195_{10} = 11000011_2$$

A diagram illustrating the conversion of the binary number  $1010_2$  to decimal. The digits are multiplied by powers of 2, with the result being the sum of the products.

$$\begin{array}{r} 3 \ 2 \ 1 \ 0 \\ 1010_2 \\ = 0 \times 2^0 + 1 \times 2^1 + 0 \times 2^2 + 1 \times 2^3 \\ = 0 + 2 + 0 + 8 = 10 \end{array}$$

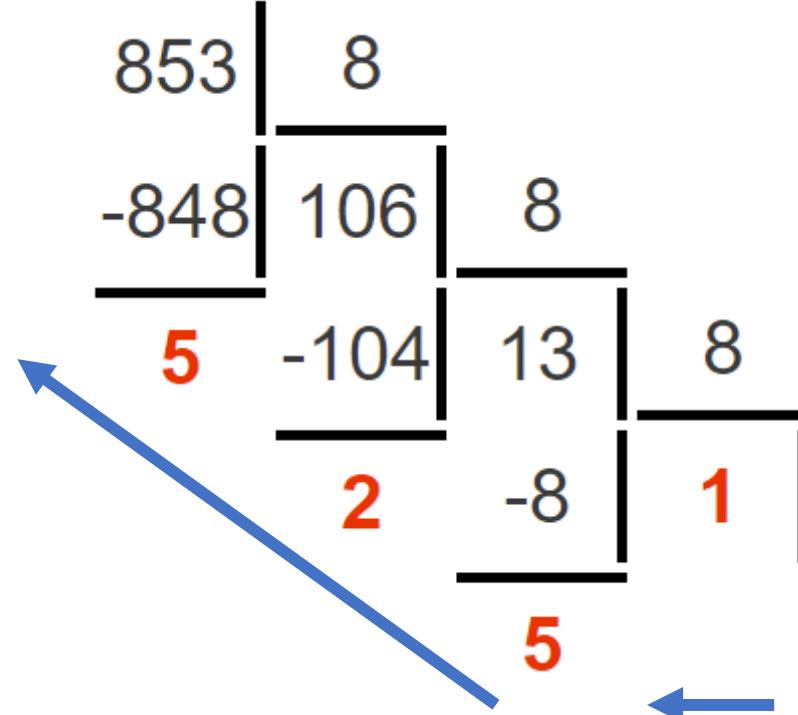
$$\begin{aligned} 1010_2 &= 0 \cdot 2^0 + 1 \cdot 2^1 + 0 \cdot 2^2 + 1 \cdot 2^3 = \\ &= 0 + 2 + 0 + 8 = 10_{10} \end{aligned}$$

# Восьмеричная система счисления

- **Восьмеричная** – счисление, широко применяющееся в высокоуровневых языках программирования (например, Java и Python) и разработке цифровой аппаратуры. Свою популярность заслужила из-за легкого перевода в цифровой (двоичный) код.
- **Состоит из цифр от 0 до 7**

| Десятичное число | Восьмеричное число | Десятичное число | Восьмеричное число |
|------------------|--------------------|------------------|--------------------|
| 0                | 0                  | 10               | 12                 |
| 1                | 1                  | 11               | 13                 |
| 2                | 2                  | 12               | 14                 |
| 3                | 3                  | 13               | 15                 |
| 4                | 4                  | 14               | 16                 |
| 5                | 5                  | 15               | 17                 |
| 6                | 6                  | 16               | 20                 |
| 7                | 7                  | 17               | 21                 |
| 8                | 10                 | 18               | 22                 |
| 9                | 11                 | 19               | 23                 |

# Перевод из 10 в 8 и из 8 в 10 систему счисления



$$853_{10} = 1525_8$$

$$\begin{aligned} 254_8 &= 2*8^2 + 5*8^1 + 4*8^0 = \\ &= 128+40+4 = \\ &= 172_{10} \end{aligned}$$

# Связь 8-ой с двоичной (2) системой

$$\begin{aligned}753_8 &= 7 \cdot 8^2 + 5 \cdot 8^1 + 3 \cdot 8^0 \\&= 7 \cdot 2^6 + 5 \cdot 2^3 + 3 \cdot 2^0\end{aligned}$$

$\overbrace{\quad\quad\quad}^{111_2} \quad \overbrace{\quad\quad\quad}^{101_2} \quad \overbrace{\quad\quad\quad}^{011_2}$

8 = 2<sup>3</sup>

$$\begin{aligned}753_8 &= (1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0) \cdot 2^6 + \\&\quad (1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0) \cdot 2^3 + \\&\quad (0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0) \cdot 2^0\end{aligned}$$

$$\begin{aligned}753_8 &= 1 \cdot 2^8 + 1 \cdot 2^7 + 1 \cdot 2^6 + \\&\quad 1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + \\&\quad 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 111101011_2\end{aligned}$$

# Связь 8-ой с двоичной (2) системой

$$8 = 2^3$$

Каждая восьмеричная цифра может быть записана как **три** двоичных (**триада**)!

$$1625_8 = \underbrace{001}_1 \underbrace{110}_6 \underbrace{010}_2 \underbrace{101}_5_2$$

|     |     |     |     |     |     |     |     |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 0   | 1   | 2   | 3   | 4   | 5   | 6   | 7   |
| 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |

# Перевод из двоичной (2) в восьмеричную (8) систему счисления

**Шаг 1.** Разбить на триады, начиная справа:

**001 001 011 101 111<sub>2</sub>**

**Шаг 2.** Каждую триаду записать одной восьмеричной цифрой:

**001 001 011 101 111<sub>2</sub>**

1      1      3      5      7

**Ответ:** **1001011101111<sub>2</sub> = 11357<sub>8</sub>**

# Права доступа к файлам и каталогам в Linux

| Код доступа        | --- | --x | -w- | -wx | r-- | r-x | rw- | rwx |
|--------------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Двоичная маска     | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
| Восьмеричная цифра | 0   | 1   | 2   | 3   | 4   | 5   | 6   | 7   |

| Числовой формат | Символьный формат | Права доступа              |
|-----------------|-------------------|----------------------------|
| 0               | ---               | Права отсутствуют          |
| 1               | --x               | Только выполнение          |
| 2               | -w-               | Только запись              |
| 3               | -wx               | Запись и выполнение        |
| 4               | r--               | Только чтение              |
| 5               | r-x               | Чтение и выполнение        |
| 6               | rw-               | Чтение и запись            |
| 7               | rwx               | Чтение, запись, выполнение |

# Права доступа к файлам и каталогам в Linux

```
mark@localhost:~/comcalc
Файл Правка Вид Поиск Терминал Справка
[mark@localhost comcalc]$ stat comcalc.cpp
  Файл: comcalc.cpp
    Размер: 2276        Блоков: 8        Блок В/В: 4096    обычный файл
Устройство: fd00h/64768d        Inode: 67694271        Ссылки: 1
Доступ: 0664 -rw-rw-r--) Uid: ( 1000/ mark)  Gid: ( 1000/ mark)
Контекст: unconfined_u:object_r:user_home_t:s0
Доступ: 2021-12-07 23:08:53.959698639 -0500
Модифицирован: 2021-12-07 23:08:53.959698639 -0500
Изменён: 2021-12-07 23:08:53.959698639 -0500
Создан: -
[mark@localhost comcalc]$ chmod 644 comcalc.cpp
[mark@localhost comcalc]$ stat -c "%a %n" comcalc.cpp
644 comcalc.cpp
[mark@localhost comcalc]$
```

# Шестнадцатеричная система счисления

- **Шестнадцатеричная система счисления** — это позиционная целочисленная система счисления с основанием 16.
- Шестнадцатеричная система счисления начала широко применяться с развитием компьютерной техники. **Как известно, компьютеры используют двоичный код. Но его использование неудобное, за счет длинных записей, а на перевод в десятичную систему уходило много времени и памяти. 16 кратно двум, поэтому вычисления производились быстрее.**
- Кроме этого, единица измерения информации — бит. В компьютерах, информация передается при помощи байтов. 1 байт = 8 бит.
- **Машинное слово** — это минимальная единица данных, состоящая из двух байт (16 бит). Таким образом, для записи команд удобно использовать именно шестнадцатеричную систему.

# Шестнадцатеричная система счисления

- Шестнадцатеричная система — это традиционная система счисления **с основанием 16**.
- Алфавит состоит из цифр **от 0 до 9** и латинских букв **от A до F**. Латинские буквы представляют собой десятичные числа от 10 до 15.

| Десятичное число        | 0  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  |
|-------------------------|----|----|----|----|----|----|----|----|----|----|
| Шестнадцатеричное число | 0  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  |
| Десятичное число        | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| Шестнадцатеричное число | A  | B  | C  | D  | E  | F  | 10 | 11 | 12 | 13 |

# Шестнадцатеричная система счисления

| $X_{10}$ | $X_{16}$ | $X_2$ | $X_{10}$ | $X_{16}$ | $X_2$ |
|----------|----------|-------|----------|----------|-------|
| 0        | 0        | 0000  | 8        | 8        | 1000  |
| 1        | 1        | 0001  | 9        | 9        | 1001  |
| 2        | 2        | 0010  | 10       | A        | 1010  |
| 3        | 3        | 0011  | 11       | B        | 1011  |
| 4        | 4        | 0100  | 12       | C        | 1100  |
| 5        | 5        | 0101  | 13       | D        | 1101  |
| 6        | 6        | 0110  | 14       | E        | 1110  |
| 7        | 7        | 0111  | 15       | F        | 1111  |

# Перевод из 10 в 16 и из 16 в 10 в систему счисления

$$\begin{array}{r} 987 \\ -976 \\ \hline 11 = B \end{array} \quad \begin{array}{r} 16 \\ | \\ 61 \\ -48 \\ \hline 3 \\ | \\ 13 = D \end{array}$$

$$987_{10} = 3DB_{16}$$

$$\begin{aligned} 3DB_{16} &= 3 \cdot 16^2 + 13 \cdot 16^1 + 11 \cdot 16^0 = \\ &= 768 + 208 + 11 = \\ &= 987_{10} \end{aligned}$$

|                         |    |    |    |    |    |    |    |    |    |    |
|-------------------------|----|----|----|----|----|----|----|----|----|----|
| Десятичное число        | 0  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  |
| Шестнадцатеричное число | 0  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  |
| Десятичное число        | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| Шестнадцатеричное число | A  | B  | C  | D  | E  | F  | 10 | 11 | 12 | 13 |

# Связь 16 с двоичной системой счисления

$$16 = 2^4$$

Каждая шестнадцатеричная цифра может быть записана как **четыре** двоичных (**тетрада**)!

$$7F1A_{16} = \underbrace{0111}_7 \quad \underbrace{1111}_{F_{(15)}} \quad \underbrace{0001}_1 \quad \underbrace{1010}_2 A_{(10)}$$

| 0    | 1    | 2    | 3    | 4    | 5    | 6    | 7    |
|------|------|------|------|------|------|------|------|
| 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 |

| 8    | 9    | A(10) | B(11) | C(12) | D(13) | E(14) | F(15) |
|------|------|-------|-------|-------|-------|-------|-------|
| 1000 | 1001 | 1010  | 1011  | 1100  | 1101  | 1110  | 1111  |

# Перевод из двоичной системы

$100101110111_2$

**Шаг 1. Разбить на тетрады, начиная справа:**

**0001 0010 1110 1111<sub>2</sub>**

**Шаг 2. Каждую тетраду записать одной шестнадцатеричной цифрой:**

**0001 0010 1110 1111<sub>2</sub>**

1

2

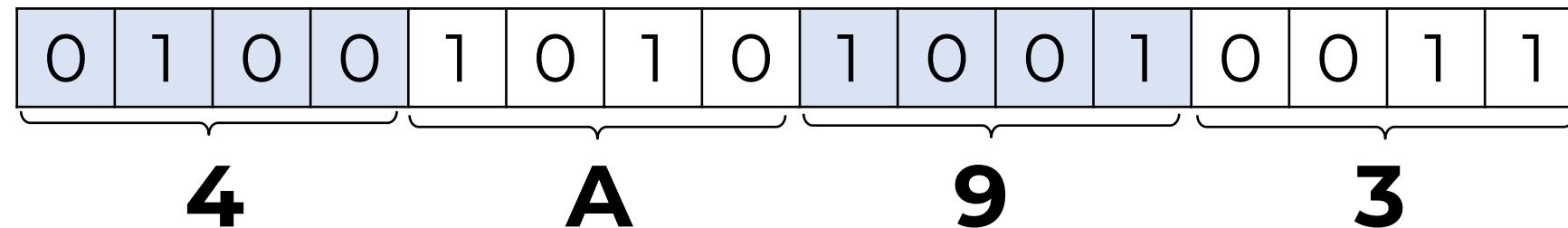
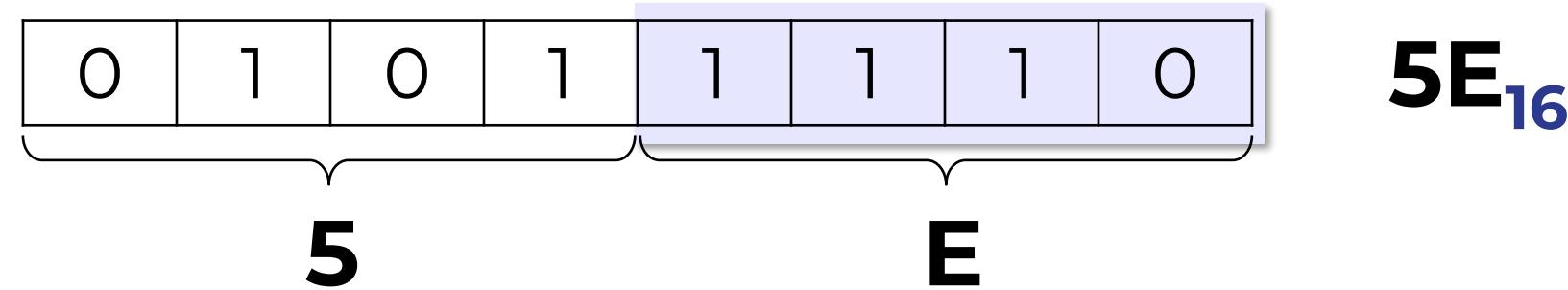
E

F

**Ответ:  $100101110111_2 = 12EF_{16}$**

# Сжатая запись двоичных кодов

Intel, AMD, ARM



$4A93_{16}$

# Применение шестнадцатеричной системы счисления

- **Шестнадцатеричная система**, как и восьмеричная активно применяется в компьютерных технологиях. При этом, запись чисел гораздо компактнее.
- **Шестнадцатеричная — применяется в следующих областях:**
  1. Низкоуровневое программирование (к примеру, ассемблер).
  2. Стандарт Юникод.
  3. Шестнадцатеричный цвет (RGB).
  4. Запись кодов ошибок.
  5. Представление данных в малоразрядных ЭВМ.и др.

# Применение шестнадцатеричной системы счисления

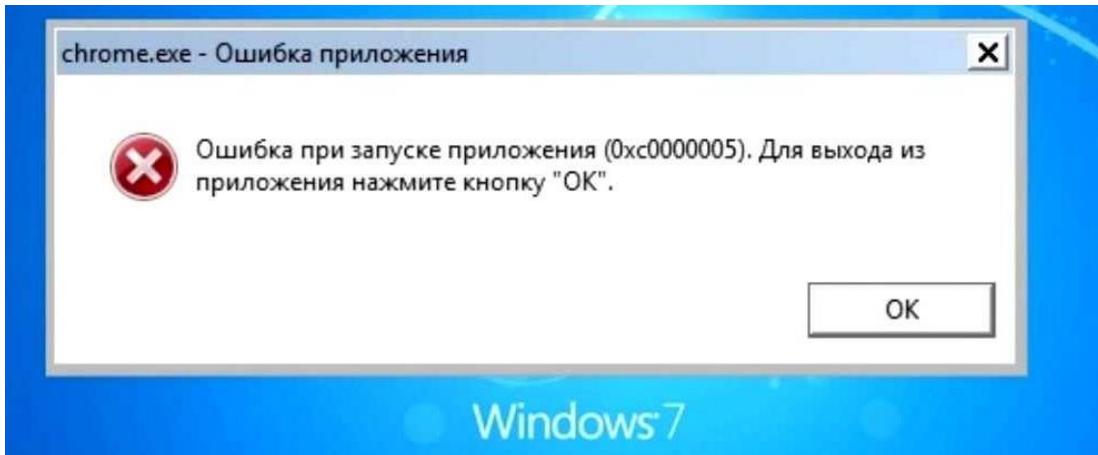


Таблица символов Юникода

Юникод Эмоджи Наборы Инструменты Алфавиты HTML-мнемоники Alt-коды »

Главная > Наборы > Смайлики-эмоджи «Лица»

Смотрите также

Смайлики-эмоджи «Лица»

Кнопка для копирования: Копировать

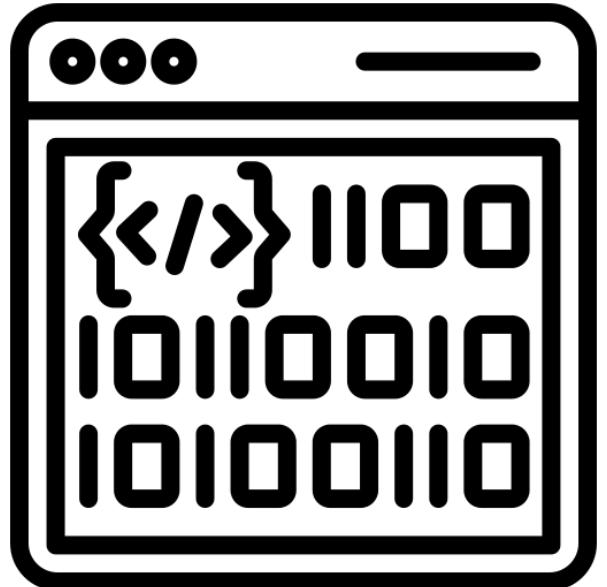
Характеристики символа:

- Номер в Юникоде: U+1F92A
- HTML-код: ⚡#12932;

Описание символа: Ухмыляющееся лицо с одним большим и одним маленьким глазом >

| Символ | Юникод  | HTML-код |
|--------|---------|----------|
| 😊      | U+1F600 | 😊        |
| 😁      | U+1F603 | 😁        |
| 😃      | U+1F604 | 😃        |
| 😆      | U+1F601 | 😆        |
| 😅      | U+1F606 | 😅        |
| 🤣      | U+1F605 | 🤣        |
| 😂      | U+1F923 | 😂        |
| 😅      | U+1F602 | 😅        |
| 😋      | U+1F642 | 😋        |
| 😛      | U+1F618 | 😛        |
| 😜      | U+1F92A | 😜        |
| 😝      | U+1F92B | 😝        |
| 😉      | U+1F917 | 😉        |
| 😘      | U+1F92D | 😘        |
| 😍      | U+1F92B | 😍        |
| 😘      | U+1F914 | 😘        |
| 😚      | U+1F910 | 😚        |
| 😉      | U+1F928 | 😉        |
| 😊      | U+1F610 | 😊        |
| 😃      | U+1F611 | 😃        |
| 😆      | U+1F636 | 😆        |
| 😅      | U+1F60F | 😅        |
| 🤣      | U+1F612 | 🤣        |
| 😂      | U+1F644 | 😂        |
| 😅      | U+1F62C | 😅        |



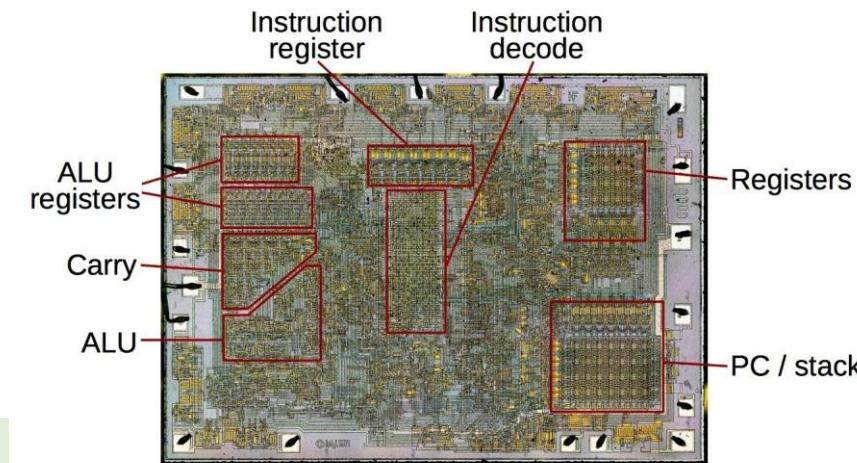


# Системы счисления. Арифметические операции

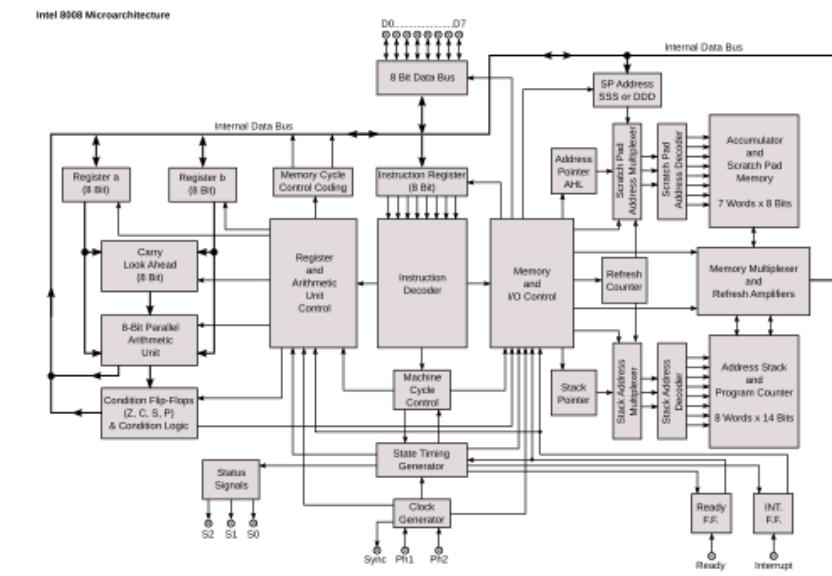


# Арифметические операции

- Арифметические операции в рассматриваемых позиционных системах счисления выполняются по законам, известным из десятичной арифметики.
- **Основные математические операции:** сложение, вычитание, умножение и деление.
- **В компьютере, в процессоре, за выполнение математических операций отвечает АЛУ (арифметико-логическое устройство).**
- **АЛУ** кроме арифметических операций (сложение, вычитание и т.д.), выполняет также логические операции (И, ИЛИ и т.д.) и побитовые операции (сдвиги, инверсии и пр.).
- **Все операции внутри АЛУ выполняются в двоичной системе.**



Внутреннее устройство чипа Intel 8008



Архитектура Intel 8008

# Перечень типовых математических (арифметических) операций выполняемых АЛУ

| Операция                                     | Описание  | Пример (в двоичной системе)  |
|--|---|--|
| <b>Сложение (+) (AND)</b>                    | Самая базовая операция ( $A+B$ )                            | $101+11=1000$  |
| <b>Вычитание (-) (OR)</b>                    | Реализуется через сложение с дополнительным кодом ( $A-B$ ) | $101-11=010$   |
| <b>Умножение (<math>\times</math>) (MUL)</b> | Последовательное сложение и сдвиги ( $A \times B$ )         | $101 \times 11 = 1111$   |
| <b>Деление (<math>\div</math>) (DIV)</b>     | Повторное вычитание и сдвиги ( $A \div B$ )                 | $110 \div 10 = 11$   |
| <b>Инкремент (INC)</b>                       | Увеличение на 1 ( $A+1$ )                                   | $101 \rightarrow 110$  |
| <b>Декремент (DEC)</b>                       | Уменьшение на 1 ( $A-1$ )                                   | $110 \rightarrow 101$  |
| <b>Сдвиги</b>                                | Логический/арифметический сдвиг влево/вправо                | Сдвиг влево (SHL) $0010 << 1 = 0100$<br>Сдвиг вправо (SHR/ASHR) $1010 >> 1 = 0101$ |
| <b>Смена знака (NEG)</b>                     | Через дополнительный код ( $-A$ )                           |  |

Примечание: Умножение и деление могут быть реализованы аппаратно или программно, в зависимости от архитектуры.

# Арифметико-логическое устройство

- АЛУ не хранит данные — работает с регистрами.
- АЛУ не управляет потоком выполнения — этим занимается УУ.
- Современные процессоры имеют несколько АЛУ (например, для целых чисел, для плавающей точки, векторные).
- АЛУ работает только с двоичными числами.
- АЛУ не имеет "списка команд" как у процессора, но выполняет набор микроопераций, соответствующих командам процессора.

# Пример: Процессор intel 8080

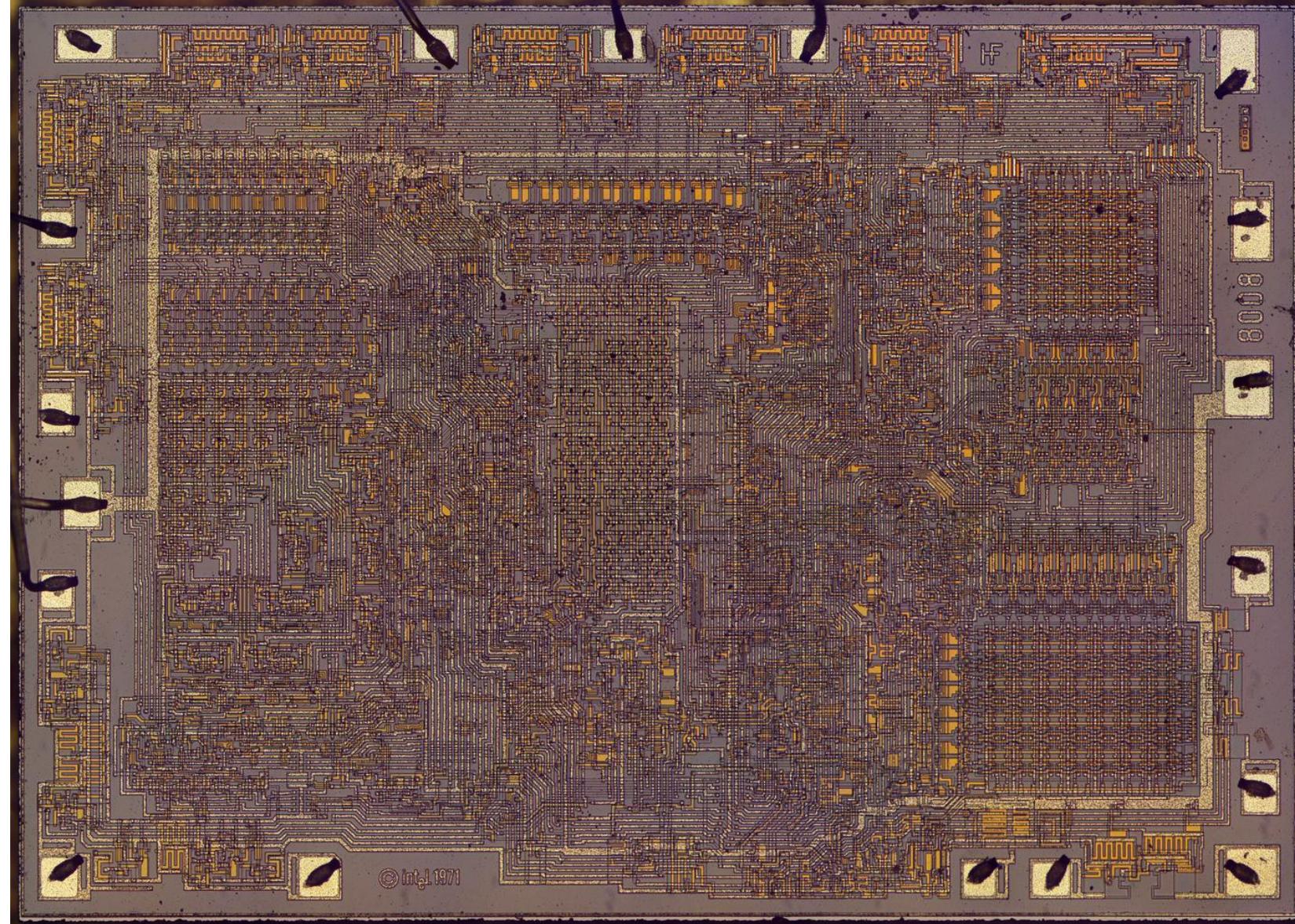


Фото кристалла процессора Intel 8008



An Intel C8008-1 processor variant with purple ceramic, gold-plated metal lid and pins

## General information

|                     |                                     |
|---------------------|-------------------------------------|
| Launched            | April 1972                          |
| Discontinued        | 1983 <sup>[1]</sup>                 |
| Marketed by         | Intel                               |
| Designed by         | Computer Terminal Corporation (CTC) |
| Common manufacturer | Intel                               |

## Performance

|                     |                    |
|---------------------|--------------------|
| Max. CPU clock rate | 500 kHz to 800 kHz |
|---------------------|--------------------|

|            |        |
|------------|--------|
| Data width | 8 bits |
|------------|--------|

|               |         |
|---------------|---------|
| Address width | 14 bits |
|---------------|---------|

## Architecture and classification

|             |   |
|-------------|---|
| Application | Computer terminals, calculators, bottling machines, 1970s ASEA industrial robots <sup>[2]</sup> (IRB 6), simple computers, etc. |
|-------------|---|

|                 |       |
|-----------------|-------|
| Technology node | 10 µm |
|-----------------|-------|

|                 |      |
|-----------------|------|
| Instruction set | 8008 |
|-----------------|------|

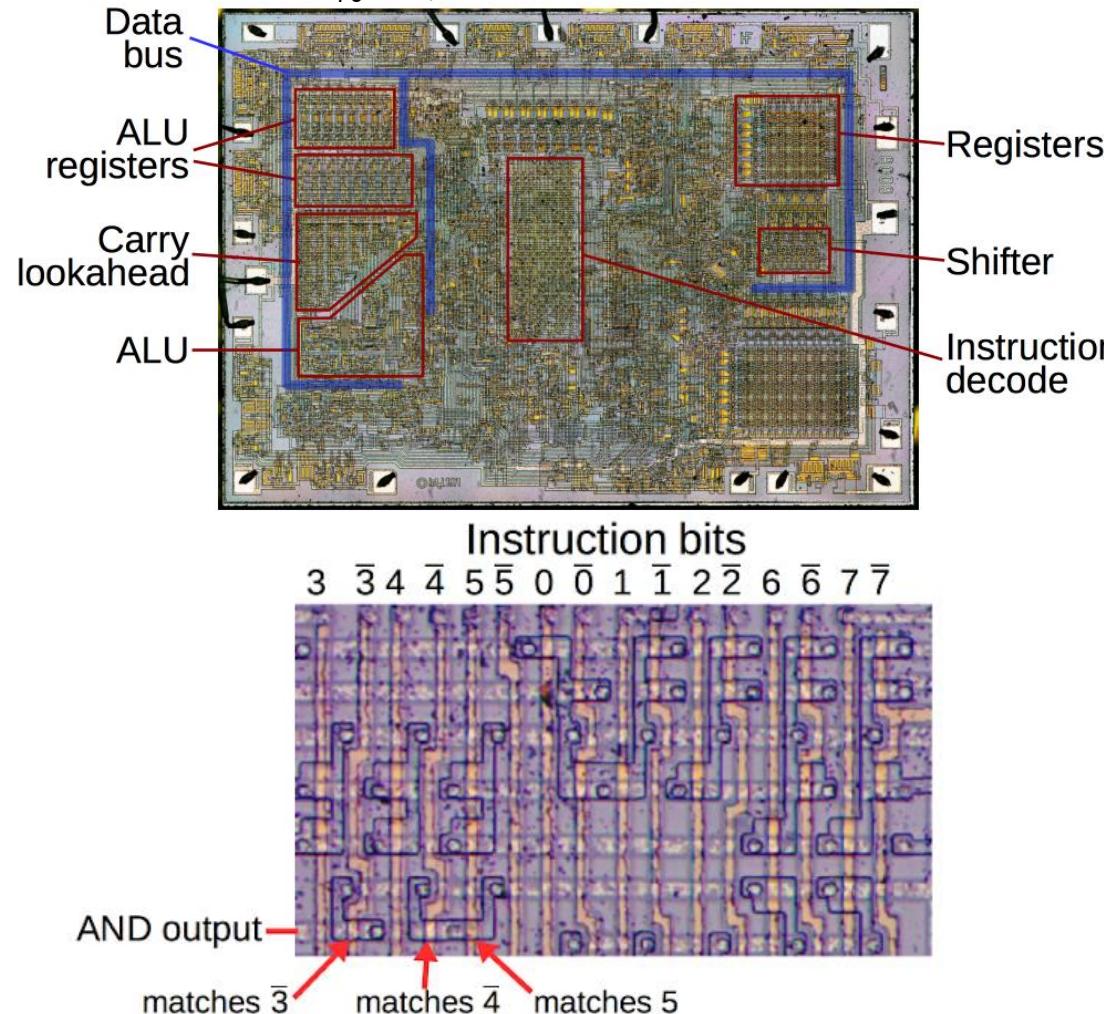
## Physical specifications

|             |                             |
|-------------|-----------------------------|
| Transistors | 3,500                       |
| Package     | 18-pin dual in-line package |
| Socket      | DIP18                       |

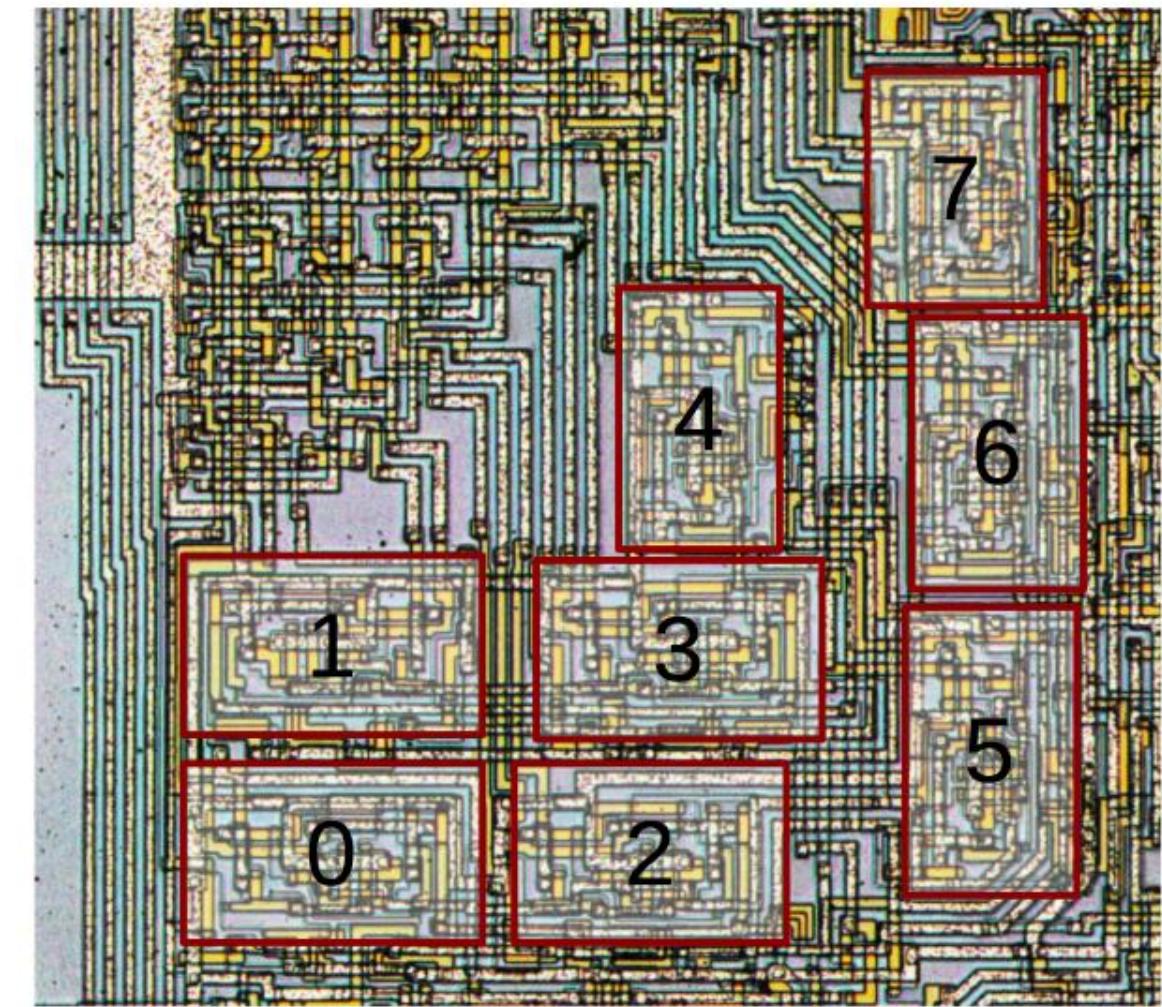
# Пример: Процессор intel 8080



Фотография микропроцессора intel 8008 с указанием важных функциональных блоков.

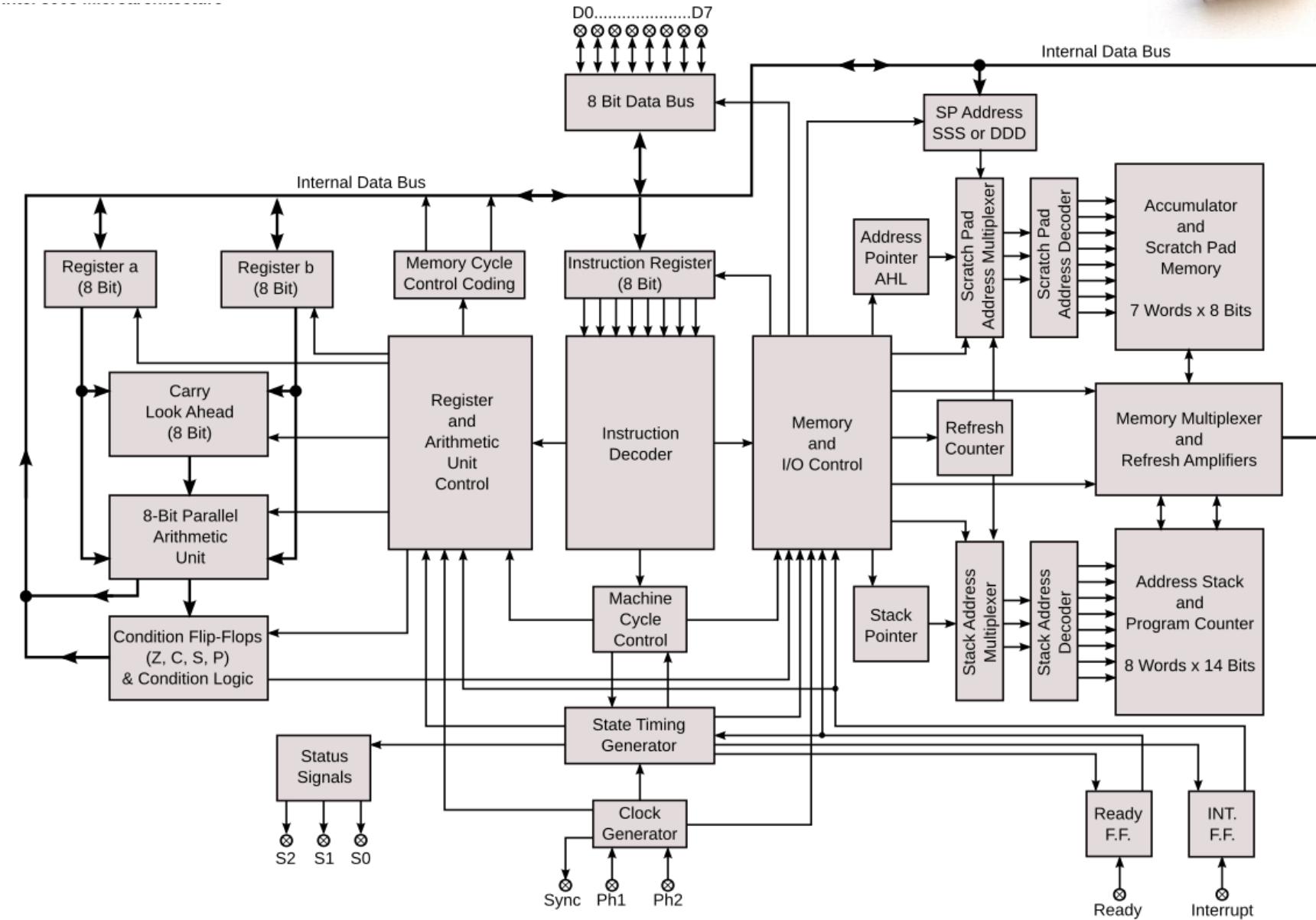


Часть команды 8008 декодируется PLA. три указанных транзистора соответствуют шаблону кода операции XX100XXX, что указывает на команду AND.



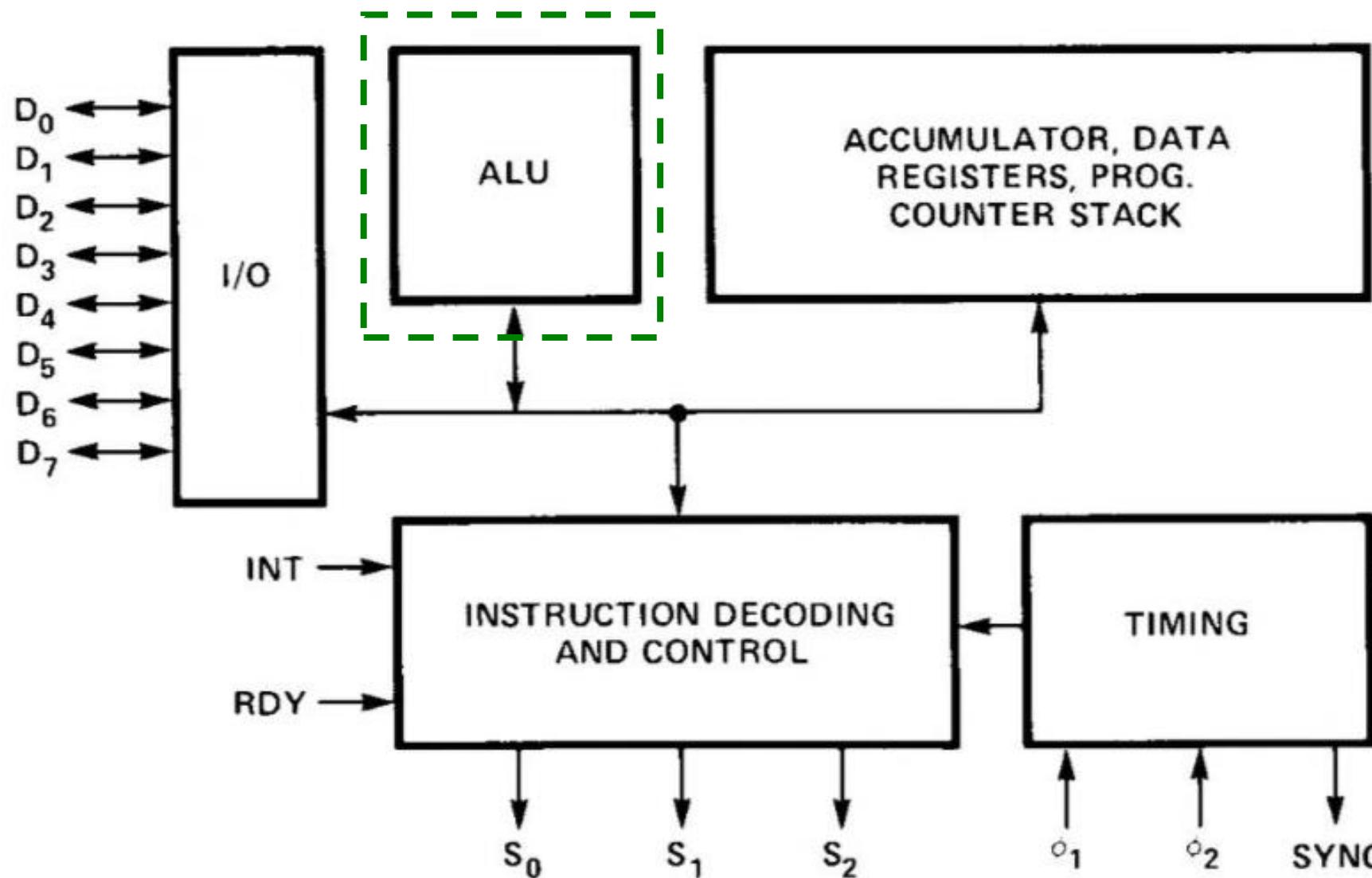
Расположение восьми блоков ALU на кристалле микропроцессора intel 8080. В отличие от большинства процессоров, блоки ALU в intel 8080 расположены в произвольном треугольном порядке.

# Процессор intel 8080



Структурная схема микропроцессора intel 8008

# Intel 8080 Арифметико-логическое устройство



Укрупненная структурная схема микропроцессора intel 8008

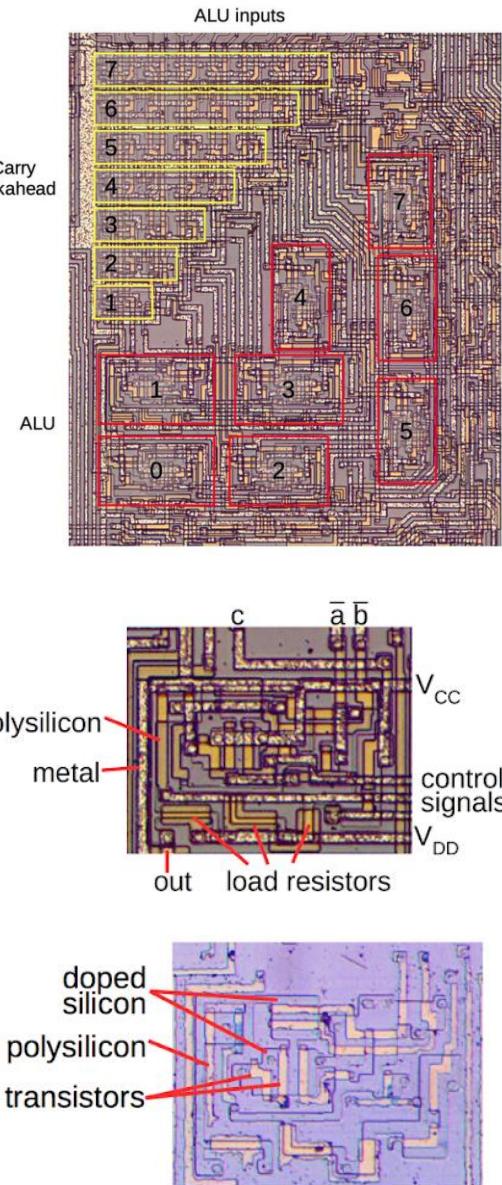


# Intel 8080 Арифметико-логическое устройство

- АЛУ процессора intel 8008 реализует четыре функции: сложение, И, исключающее ИЛИ и ИЛИ.
- Операция сложения добавляет два 8-битных числа. Остальные три операции являются стандартными логическими операциями. Операция И устанавливает выходной бит, если он установлен в первом И во втором числе. ИЛИ проверяет, установлен ли бит в первом ИЛИ во втором числе (или в обоих). XOR (исключающее ИЛИ) проверяет, установлен ли бит в первом ИЛИ во втором числе (но не в обоих).
- Концепция переноса при сложении является ключевой для арифметико-логического устройства.
- Двоичное сложение в процессоре похоже на сложение в столбик, которое изучают в начальной школе, только вместо десятичных чисел используются двоичные. Начиная с правого столбца, складываются два числа в каждом столбце, и может возникнуть перенос в следующий столбец. Таким образом, в каждом столбце арифметико-логическое устройство складывает два бита, а также бит переноса.
- В большинстве ранних микропроцессоров сложение в каждом столбце должно выполняться только после сложения в столбце справа и получения переноса. Перенос «прокатывается» по битам справа налево, замедляя процесс сложения. В 8008 однако используется схема быстрого прогнозирования переноса для параллельного формирования переносов для всех 8 столбцов перед выполнением сложения. Затем все столбцы можно сложить параллельно, не дожидаясь, пока перенос «пройдётся» по сумме. Эта схема упреждающего переноса является необычной для ранних микропроцессоров из-за своей сложности.
- Поскольку intel 8008 — это 8-битный процессор, арифметико-логическое устройство работает с двумя 8-битными аргументами. Большинство 8-битных процессоров (включая 8008) используют для арифметико-логического устройства конструкцию «битовых срезов», в которой один битовый срез арифметико-логического устройства повторяется восемь раз. Каждый однобитовый срез арифметико-логического устройства принимает два входных бита и бит переноса и выдаёт выходной бит.
- В большинстве 8-разрядных процессоров битовый ALU-срез организован путем укладки 8 прямоугольных ALU-срезов в виде компактного правильного блока. Однако в 8008 восемь алюминиевых пластин расположены нерегулярно — некоторые блоки даже расположены боком, — как показано на схеме ниже. Мотивацией для этого является то, что схема переноса данных занимает треугольное пространство на чипе.

# Intel 8080 Арифметико-логическое устройство

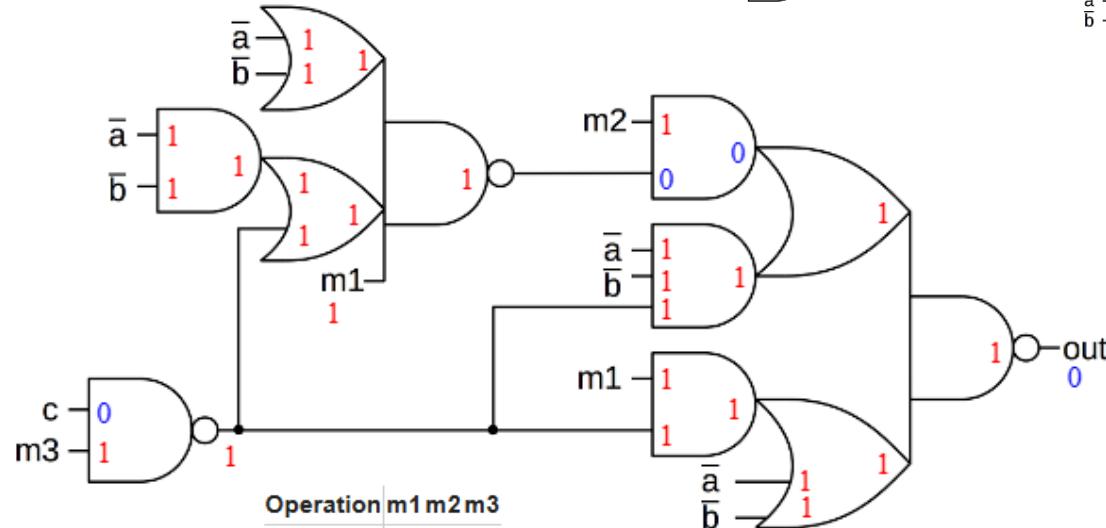
- Чтобы лучше использовать оставшееся пространство, арифметико-логическое устройство 8008 имеет необычную треугольную форму.
- Увеличив масштаб фотографии кристалла, мы можем рассмотреть один из срезов арифметико-логического устройства и увидеть, как устроена схема. Чип состоит из трёх слоёв (для упрощения). Самый верхний слой — это металлическая проводка. Это наиболее заметная часть, которая выглядит металлической (что неудивительно). На изображении ниже вы можете увидеть горизонтальные и вертикальные металлические дорожки. Слой поликремния находится под металлическим слоем и под микроскопом выглядит жёлтым/оранжевым. Поликремний может использоваться в качестве проводника, но, что более важно, он формирует затворы транзисторов, включая и выключая их. Нижний слой — это сам сероватый кремниевый кристалл, но его трудно разглядеть под другими слоями.
- На схеме выше перенос с и дополненные a и b входы подключаются через металлические провода вверху. Выход АЛУ находится внизу. Управляющие сигналы представлены горизонтальными металлическими линиями. Схема питается от металлических линий V<sub>CC</sub> (+5 вольт) и V<sub>DD</sub> (-9 вольт). Более яркие жёлтые области из поликремния — это транзисторы. Для каждого элемента схемы требуется «нагрузочный резистор», подключённый к V<sub>DD</sub>, чтобы на его выходе было низкое напряжение. Для повышения производительности они реализованы с использованием транзисторов, а не резисторов.
- При удалении металлического слоя с помощью кислоты кремниевый и поликремниевый слои становятся более заметными, как показано ниже. Чип формируется на кремниевой пластине, в некоторых областях которой содержатся примеси, создающие области полупроводникового кремния. Вы можете увидеть тёмные линии вдоль границы между легированным и нелегированным кремнием. Транзистор формируется там, где желтоватый поликремниевый провод пересекает легированный кремний. Транзистор образует переключатель между двумя кремниевыми сторонами, управляемый поликремниевым затвором. Каждый блок АЛУ содержит 20 транзисторов; на схеме ниже показаны два из них.



# Intel 8080 Моделирование одного блока АЛУ

Operation: Sum

| CAB   | Result |
|-------|--------|
| 0 0 0 | 0      |
| 0 0 1 | 1      |
| 0 1 0 | 1      |
| 0 1 1 | 0      |
| 1 0 0 | 0      |
| 1 0 1 | 0      |
| 1 1 0 | 0      |
| 1 1 1 | 1      |



Operation: AND

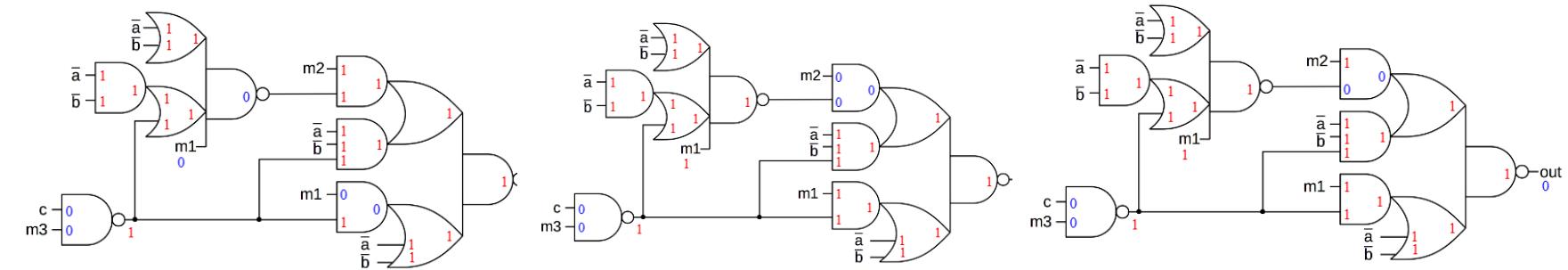
| CAB   | Result |
|-------|--------|
| 0 0 0 | 0      |
| 0 0 1 | 0      |
| 0 1 0 | 0      |
| 0 1 1 | 1      |
| 1 0 0 | 0      |
| 1 0 1 | 0      |
| 1 1 0 | 0      |
| 1 1 1 | 1      |

Operation: OR

| CAB   | Result |
|-------|--------|
| 0 0 0 | 0      |
| 0 0 1 | 1      |
| 0 1 0 | 1      |
| 0 1 1 | 1      |
| 1 0 0 | 0      |
| 1 0 1 | 1      |
| 1 1 0 | 1      |
| 1 1 1 | 1      |

Operation: XOR

| CAB   | Result |
|-------|--------|
| 0 0 0 | 0      |
| 0 0 1 | 1      |
| 0 1 0 | 1      |
| 0 1 1 | 0      |
| 1 0 0 | 0      |
| 1 0 1 | 1      |
| 1 1 0 | 1      |
| 1 1 1 | 0      |



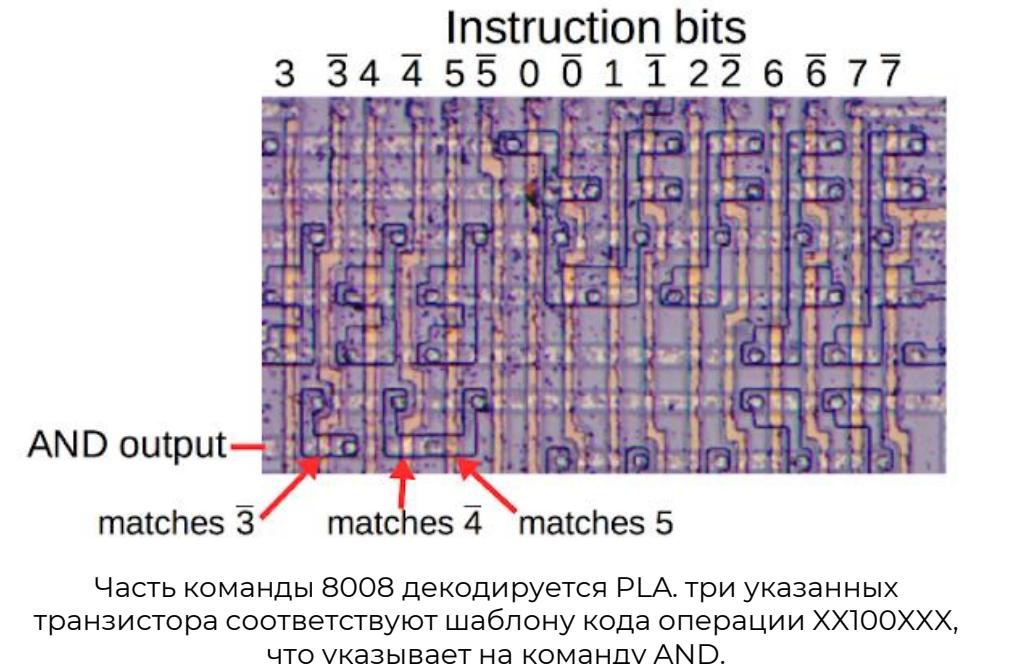
Внимательно изучив фотографии кристалла, можно составить **схему из 20 транзисторов** и их соединений в блоке АЛУ. На основе этого можно реконструировать логические элементы, из которых состоит схема. В результате получилась схема, показанная на рисунке, на которой изображен один бит блока АЛУ. Каждое АЛУ срезки принимает два входных сигнала ( $a$   $b$ ) и входной сигнал переноса  $c$ , а на выходе выдаёт один бит результата. Есть три линии выбора режима ( $m_1$ ,  $m_2$  и  $m_3$ ), которые выбирают одну из четырёх операций арифметико-логического устройства.

**Хотя этот фрагмент АЛУ выглядит так, будто он состоит из множества вентилей, физически он состоит всего из трёх вентилей: двух больших многоуровневых вентилей И-ИЛИ-НЕ-И и одного вентиля НЕ-И.** Логика И-ИЛИ-НЕ-И реализована на чипе в виде одного сложного вентиля, а не путём объединения более простых вентилей, поскольку один большой вентиль обеспечивает более высокую производительность при меньшем количестве схемных элементов, чем несколько маленьких вентилей. Одной из особенностей МОП-логики является то, что из неё так же легко создать логическую схему «И-ИЛИ-НЕ» (например), как и из обычной логической схемы «НЕ».

# Декодирование инструкций: как АЛУ определяет, какую операцию выполнять

- Процессор intel 8008 выполняет 8-битные инструкции, которые позволяют перемещать данные, выполнять операции ввода-вывода, переходить по веткам, вызывать подпрограммы и так далее. Логика декодирования команд анализирует команду и определяет, какую операцию следует выполнить, генерируя около 30 управляемых сигналов. **Более четверти команд выполняют операции с арифметико-логическим устройством**, и набор команд тщательно продуман таким образом, чтобы три бита команды указывали, какую из восьми операций следует выполнить. Анализируя эти биты, декодер команд генерирует управляемые линии режима ALU m1, m2 и m3.

- На примере инструкций AND показано, как это работает.**
- Все инструкции AND имеют битовую комбинацию xx100xxx** (где x — это 0 или 1).
- Например, инструкция AND с памятью имеет вид 10100111, а инструкция AND с константой — 00100100.
- Когда схема декодирования инструкций соответствует этому шаблону, она переводит управляющую линию m1 в низкое состояние, что заставляет АЛУ выполнить операцию И. Другие битовые шаблоны генерируют другие управляющие сигналы ALU.
- На приведенной схеме показана часть схемы декодирования команд. Биты команд (и их дополнения) расположены на желтых поликремниевых проводах, проходящих вертикально через схему. Каждая строка соответствует последовательности битов, к каждому биту команд подключен транзистор, который должен быть согласован. (Области легированного кремния, образующие транзисторы, обозначены черными контурами. Кружки обозначают соединения между транзистором и металлической линией ряда.) Например, три транзистора, отмеченные стрелками, соответствуют разряду 3 low, разряду 4 low и разряду 5 high, что определяет последовательность команд И. Таким образом, процессор использует сетку транзисторов в декодере команд для определения значения каждой команды.



# Сложение (+) (AND)

- **Сложение двоичных чисел** — это операция, аналогичная сложению в десятичной системе, но в двоичной используется всего две цифры: 0 и 1.
- Правила сложения двоичных цифр (битов):

| Бит 1 | Бит 2 | Результат суммы | Перенос (carry) |
|-------|-------|-----------------|-----------------|
| 0     | 0     | 0               | 0               |
| 0     | 1     | 1               | 0               |
| 1     | 0     | 1               | 0               |
| 1     | 1     | 0               | 1               |

- Если есть перенос с предыдущего разряда ( $\text{carry} = 1$ ), то его нужно учесть как дополнительный бит к сумме.

**Пример:** сложение двоичных чисел **1011** (11 в десятичной) и **1101** (13 в десятичной)

$$\begin{array}{r} & \text{1 (перенос)} \\ & 1 \ 0 \ 1 \ 1 \\ + & 1 \ 1 \ 0 \ 1 \\ \hline & 1 \ 1 \ 0 \ 0 \ 0 \end{array}$$

**Складываем поразрядно справа налево:**

**1 + 1 = 0**, перенос 1

**1 + 1 + перенос 1 = 1** (потому что  $1+1=0$  с переносом 1, итого  $0+1=1$ ), перенос 1

**0 + 1 + перенос 1 = 0**, перенос 1

**1 + 1 + перенос 1 = 1**, перенос 1

**Ответ: 11000**

(в десятичной системе это 24).

# Intel 8080 – Сложение (+) (AND)

- АЛУ процессора Intel 8080 использует 8-битный сумматор.
- Это означает, что при сложении двух 8-битных чисел, каждое АЛУ-бит складывает соответствующие биты операндов и входящий перенос из предыдущего, младшего бита.
- Результат этого сложения дает бит суммы для текущей позиции и бит переноса для следующей, старшей позиции.
- Этот перенос "перетекает" (ripples) от младшего бита к старшему.
- Для этого используется аккумулятор A и один из регистров в качестве operandов, а результат сохраняется в аккумуляторе с установкой флагов переноса, нуля и других.
- Если результат превышает 255, флаг переноса (Carry) позволяет продолжить сложение с более старшими байтами с помощью команды ADC для работы с 16-битными числами.
- В итоге даже если число результат вычисления переходит предел в 8-битных чисел, то результат все равно получается верным.

# Сложение двух 8-битных чисел

- Пример сложения двух 8-битных чисел
- Разберем пример:  $205 + 70 = 275$

## Шаг 1: Переведём в двоичную систему

$$205_{10} = 1100\ 1101_2$$

$$70_{10} = 0100\ 0110_2$$

## Шаг 2: Складываем в столбик

|   |   |   |   |   |   |   |   |            |
|---|---|---|---|---|---|---|---|------------|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | ← переносы |
| 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | (205)      |
| + | 0 | 1 | 0 | 0 | 0 | 1 | 1 | (70)       |
|   |   |   |   |   |   |   |   |            |
| 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | (275)      |

Результат: 9 бит  $\rightarrow 1\ 0001\ 0011_2 = 275_{10}$

Но у нас только 8 бит — старший бит (перенос) теряется из результата.

## Шаг 3: Результат в 8 битах

Младшие 8 бит:  $0001\ 0011_2 = 19_{10}$

Флаг переноса (Carry, C) = 1 — был выход за пределы 8 бит

- 275 больше, чем 255 — максимальное значение, которое можно представить в 8 битах (беззнаковое).

- Поэтому при обычном сложении произойдёт переполнение, и в аккумуляторе останется только младший байт:  $275 - 256 = 19$ , а флаг переноса (Carry) будет установлен в 1.

- Но чтобы получить правильный результат — 275, нужно использовать сложение с переносом (add with carry), то есть работать с 16-битным результатом, состоящим из двух байтов:

- Младший байт (LSB):  $275 \bmod 256 = 19$

- Старший байт (MSB):  $275 \div 256 = 1$  (целочисленное деление)

- Таким образом:

$$275_{10} = 0x0113 = 1 \cdot 256 + 19$$

# Вычитание (-) (OR)

- Поскольку АЛУ в основном "умеет" складывать, **вычитание** выполняется как:

$$A - B = A + (-B)$$

- где  $-B$  — это дополнительный код числа  $B$  (инверсия битов + 1).

- Пример:**

- $5 - 3 = 2$

- в 4-битной системе

- $5_{10} = 0101_2$
- $3_{10} = 0011_2$

- $-3$  в дополнительном коде:

**инвертируем биты → 1100 , прибавляем 1 → 1101**

- Складываем:  $0101 + 1101 = 10010$**

- Отбрасываем** перенос (5-й бит): остаётся  $0010_2 = 2_{10}$  — верно!

# Вычитание двух 8-битных чисел

- **Разберем пример:  $5_{10} - 2_{10} = 3_{10}$**

- **Шаг 1: Запись чисел в двоичной системе**

- Мы работаем с 8-битными числами (1 байт), поэтому каждое число представляется 8 битами.

- $5_{10} = 0000\ 0101_2$
- $2_{10} = 0000\ 0010_2$

- **Шаг 2: Как процессор выполняет вычитание?**

- Процессор не имеет отдельной схемы для вычитания.
- Вместо этого он использует алгебраическое правило:
- **$A - B = A + (-B)$** , где  $-B$  — это число в дополнительном коде (two's complement).

- **Шаг 3: Получаем дополнительный код числа -2**

- Чтобы представить  $-2$  в 8-битной двоичной системе:

- 1. Берём прямой код числа  $2: 2_{10} = 0000\ 0010_2$

- 2. **Инвертируем** все биты (обратный код):

- $0000\ 0010 = 1111\ 1101$

- 3. Прибавляем 1 (получаем дополнительный код):

- $1111\ 1101 + 1 = 1111\ 1110$

- **Таким образом:  $-2_{10} = 1111\ 1110_2$  (в дополнительном коде)**

- **Шаг 4: Сложение  $5 + (-2)$  в АЛУ**

- Теперь АЛУ выполняет сложение:

- $5_{10} + (-2_{10}) = 0000\ 0101_2 + 1111\ 1110_2$

- Выполним сложение по битам, начиная с младшего (справа):

Переносы:  $\begin{array}{r} 1\ 1\ 1\ 1\ 1\ 1\ 0\ 0 \\ 0\ 0\ 0\ 0 \quad 0\ 1\ 0\ 1 \leftarrow 5 \\ + 1\ 1\ 1\ 1 \quad 1\ 1\ 1\ 0 \leftarrow -2 \\ \hline \end{array}$

Результат:  $1\ 0\ 0\ 0\ 0 \quad 0\ 0\ 1\ 1 \leftarrow 9 \text{ бит!}$

- **Сложение дало 9 бит:** старший (девятый) бит — это **перенос (Carry)**.

- Младшие 8 бит:  $0000\ 0011_2 = 3_{10}$

- Девятый бит (перенос) = 1

- **Шаг 5: Что делает процессор с результатом?**

- Результат (8 бит) сохраняется в аккумуляторе A:

- **$A = 0000\ 0011_2 = 3_{10}$**

- Старший бит переноса (Carry) анализируется:

- При вычитании через сложение с дополнительным кодом, **наличие переноса = 1 означает, что заёма не было**.

- **После выполнения:  $A = 3$  Флаг C = 1 (нет заёма)**

# Умножение (\*) (MUL)

- Умножение в двоичной системе гораздо проще, чем в десятичной, потому что:
- Цифры только 0 и 1 → на каждом шаге либо прибавляем множимое, либо ничего не прибавляем.
- **Правило:**
- Если бит множителя = 1 → прибавляем сдвинутое множимое.
- Если бит = 0 → прибавляем 0.
- Каждый следующий разряд требует сдвига влево на 1 позицию (как при умножении на 10 в десятичной).

**Пример:  $5 \times 3 = 15$**

- Переведём в двоичную систему:
- $5_{10} = 101_2$
- $3_{10} = 11_2$
- Выполним умножение "в столбик":

$$\begin{array}{r} 1\ 0\ 1 & \leftarrow 5 \\ \times\ 1\ 1 & \leftarrow 3 \\ \hline 1\ 0\ 1 & \leftarrow 101 \times 1 \text{ (младший бит)} \\ +\ 1\ 0\ 1 & \leftarrow 101 \times 1 \text{ (со сдвигом влево на 1)} \\ \hline 1\ 1\ 1\ 1 & \leftarrow 15_{10} \end{array}$$

**Разбор:**

- Младший бит 3 = 1 → прибавляем 101
- Следующий бит = 1 → прибавляем 101 со сдвигом влево на 1 → 1010
- **Сумма:  $101 + 1010 = 1111_2 = 15_{10}$**
- **Ответ:  $1111_2 = 15_{10}$**

# Деление ( $\div$ ) (DIV)

- **Деление в двоичной системе похоже на деление "уголком" в десятичной,** но проще — на каждом шаге частное — либо 0, либо 1.
- **Правило:**
  - На каждом шаге сравниваем делитель с текущим остатком.
  - Если делитель  $\leq$  остатка  $\rightarrow$  ставим 1 в частное и вычитаем.
  - Если больше  $\rightarrow$  ставим 0.
  - Сдвигаем остаток влево на 1 бит (включаем следующий бит делимого).

**Пример:  $15 \div 3 = 5$**

- $15_{10} = 1111_2$
- $3_{10} = 11_2$
- $5_{10} = 101_2$
- Выполним деление:

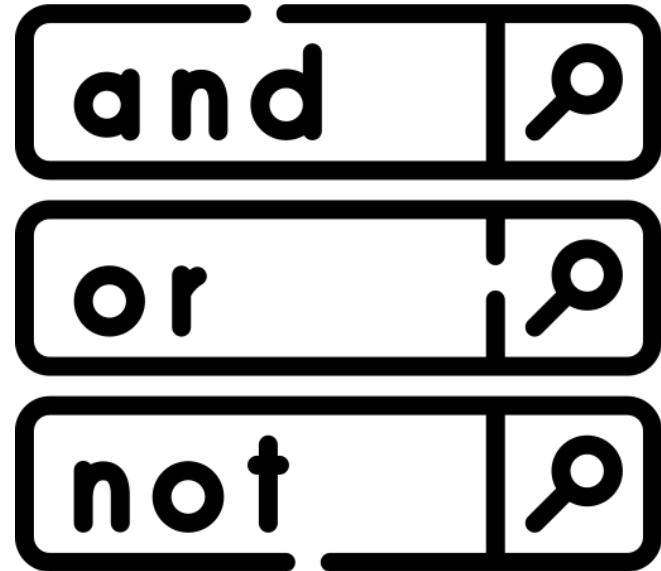
$$\begin{array}{r} 1\ 0\ 1 & \leftarrow \text{частное (5)} \\ \hline 11 | 1\ 1\ 1\ 1 & \leftarrow \text{делимое (15)} \\ - 1\ 1 & \leftarrow 11 \leq 11 \rightarrow 1, \text{ вычитаем} \\ \hline 0\ 1 \\ 0\ 0 & \leftarrow 11 > 01 \rightarrow 0 \\ \hline 1\ 1 & \leftarrow \text{сдвигаем дальше} \\ - 1\ 1 & \leftarrow 11 \leq 11 \rightarrow 1, \text{ вычитаем} \\ \hline 0 & \leftarrow \text{остаток 0} \end{array}$$

**Результат:**

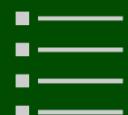
- Частное =  $101_2 = 5_{10}$
- Остаток = 0

# Арифметико-логическое устройство

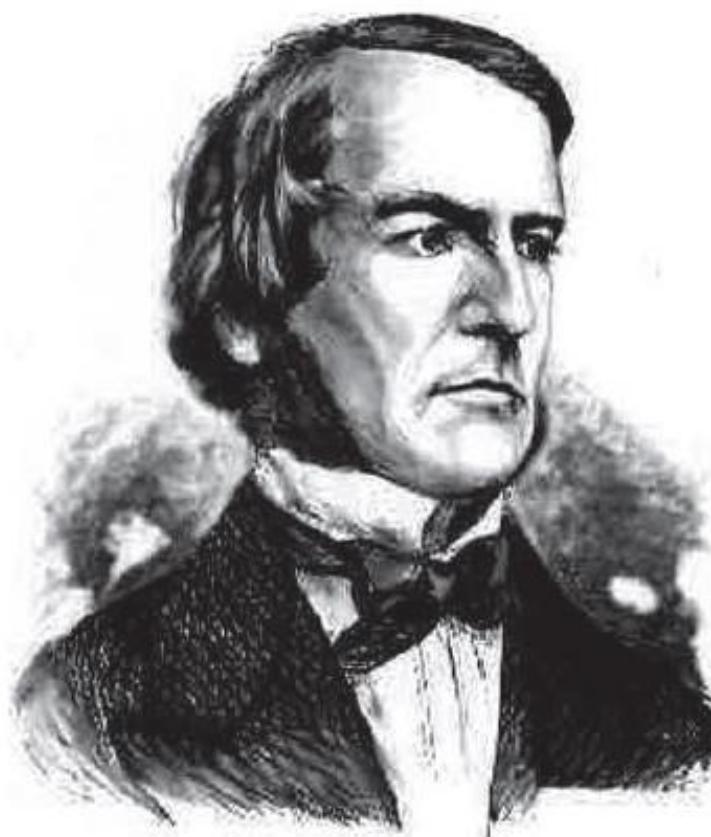
- В простых процессорах (например, Intel 8080) нет команды умножения и деления.
- **Умножение** реализуется программно через циклы:
  - Проверка битов множителя
  - Сдвиги
  - Сложение
- **Деление** реализуется программно:
  - Через цикл вычитания и сдвига.
  - или с использованием алгоритма восстановливающего / невосстанавливющего деления.
- **Современные процессоры имеют аппаратный умножитель в АЛУ или отдельный блок.**
- **Интересный факт:** Даже сложные операции ( $\sin$ ,  $\log$ ,  $\sqrt{}$ ) в компьютере в конечном счёте сводятся к сдвигам, сложениям и вычитаниям — всё это делает АЛУ.



Булева алгебра.  
Основные логические  
операции.



# 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 Джорж Буль.

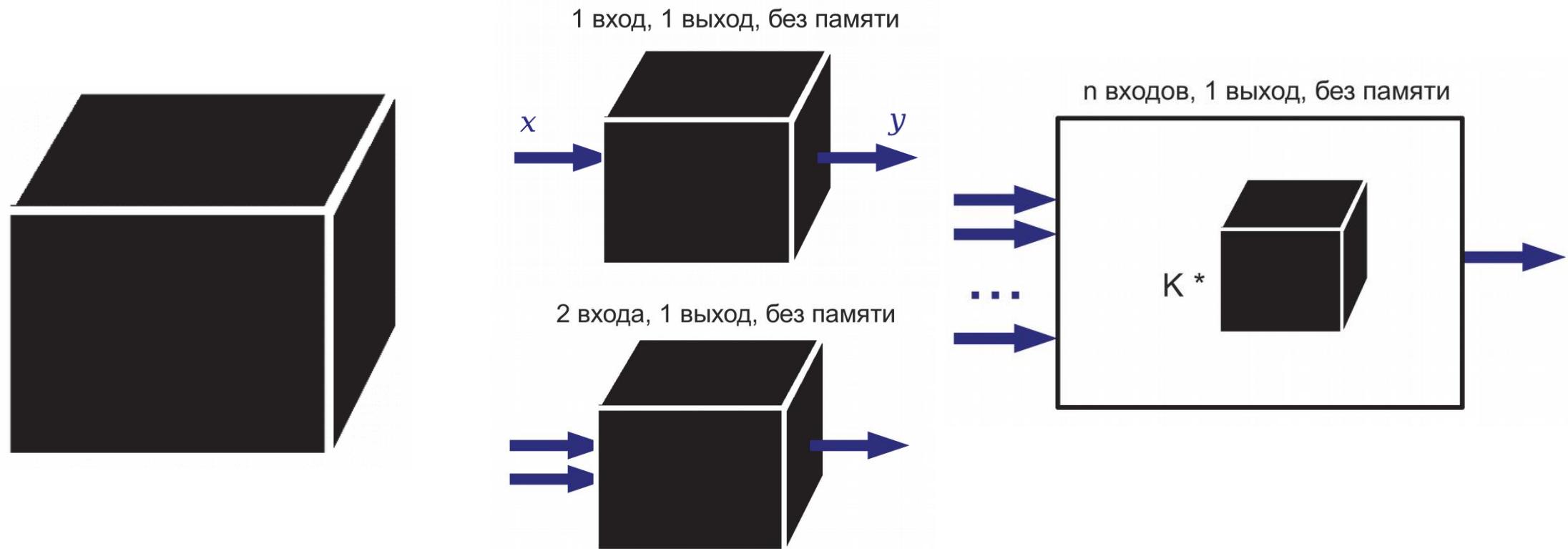
По его имени математический аппарат и получил название **булевой алгебры, или алгебры логики**.

**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 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)**.

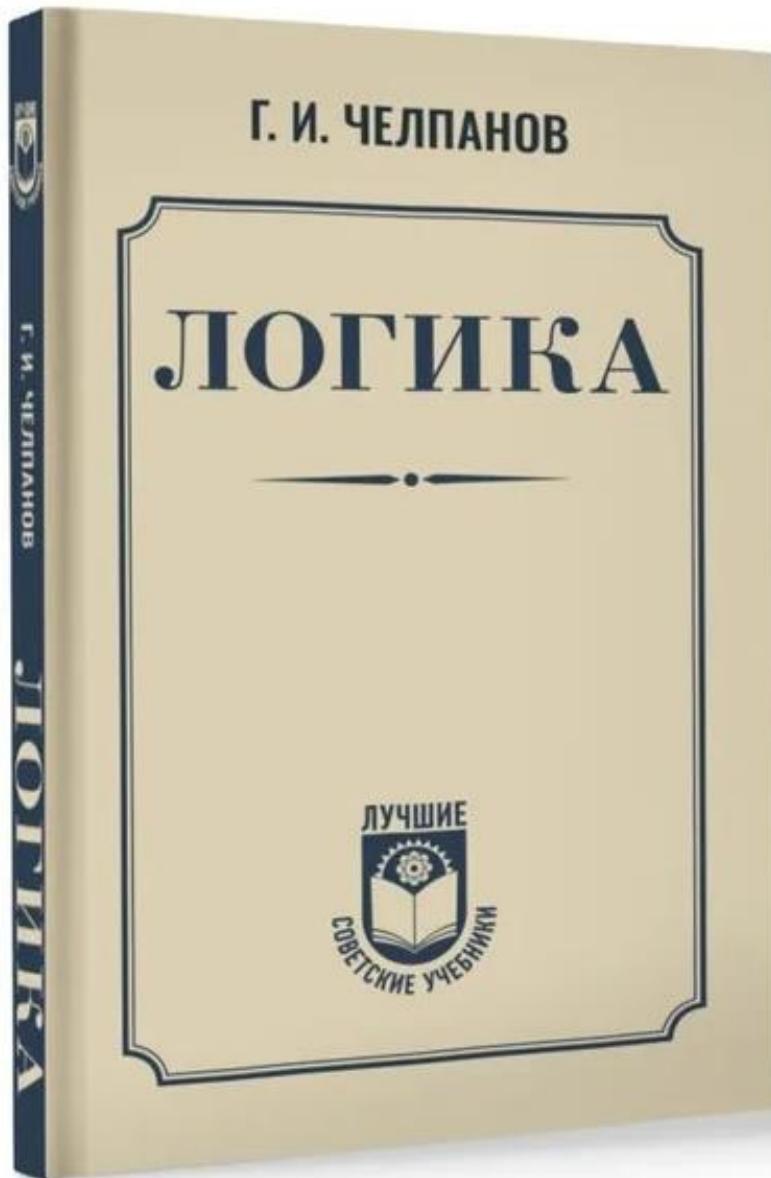
# Логические высказывания и переменные

- **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.**

# Логика



**ЛЕГЕНДАРНАЯ КНИГА  
по основам  
традиционной логики**



**Г. И. ЧЕЛПАНОВ**

Выдающийся психолог, философ, логик, педагог  
Создатель первой научной школы отечественной психологии  
Основатель Института экспериментальной психологии при Московском университете

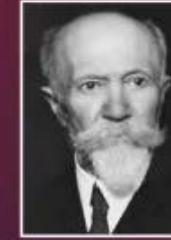
# УЧЕБНИК ЛОГИКИ

Обширная вступительная статья известного философа Б. В. Бирюкова

ИЗ НАСЛЕДИЯ МИРОВОЙ ФИЛОСОФСКОЙ МЫСЛИ  
ЛОГИКА

URSS

**Из наследия Н. О. Лосского**



**Н. О. ЛОССКИЙ**

Ярчайший представитель русской религиозной философии  
Один из основоположников интуитивизма

# ЛОГИКА

Гносеологическое введение в логику

**Суждение • Понятие**

Логические законы мышления • Теории понятия • Отношения между понятиями • Форма суждения • Недоразвитость знания и заблуждения • Однозначная и многозначная связь в суждении

**Доказательство • Умозаключение**

Непосредственное оправдание суждений • Непосредственные умозаключения • Теории силлогизма • Материально-сintéтические и индуктивные умозаключения • Гипотеза • Заключение учения о доказательствах

ИЗ НАСЛЕДИЯ МИРОВОЙ ФИЛОСОФСКОЙ МЫСЛИ  
ЛОГИКА

URSS



# Три основные логические операции

- Все возможные логические функции переменных можно образовать с помощью **трех основных операций**:
  - **Конъюнкция** (от лат. *conjunction* союз, связь) — логическая операция, по своему применению максимально приближённая к союзу «и». Синонимы: **логическое «И»**, логическое умножение, или просто «И».
  - **Дизъюнкция** (лат. *disjunction* — 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.

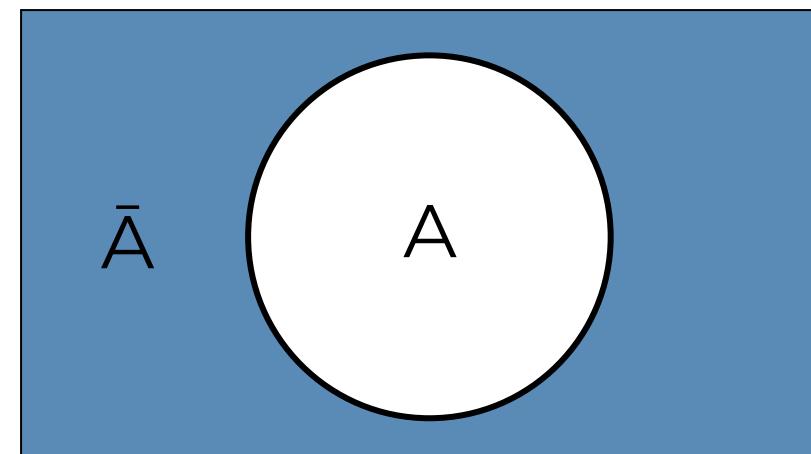
# «НЕ» - Инверсия - NOT

- 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 A истинно, то НЕ A ложно, и наоборот.
- Обозначения: НЕ, not,  $\neg$ ,  $\bar{}$ .

## Таблица истинности

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

## Диаграмма Венна (круг Эйлера)

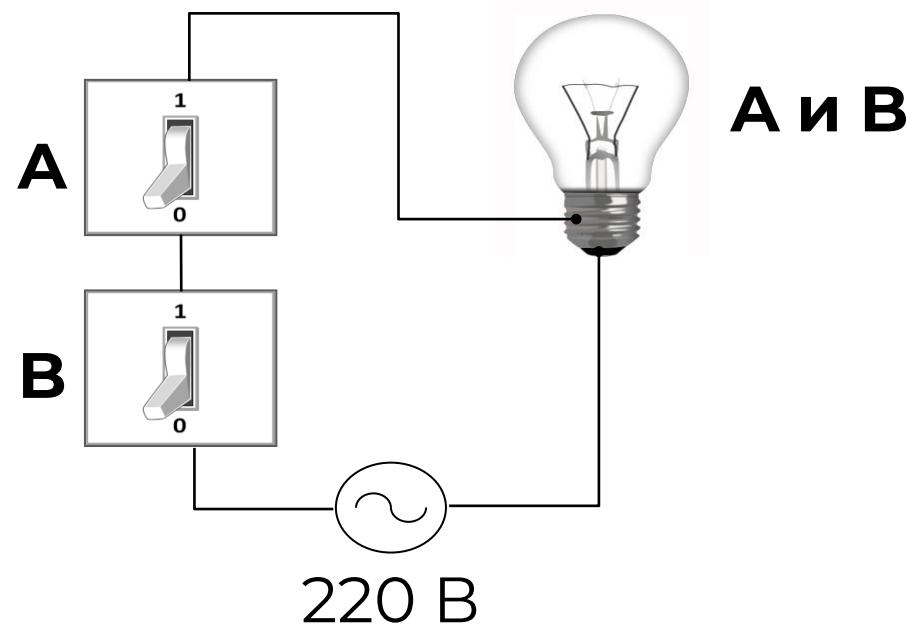


# «И» - Конъюнкция - AND

- **Логическая схема «И»**

называется также конъюнктором, выполняет операцию логического умножения (конъюнкции), может иметь от двух до восьми входов.

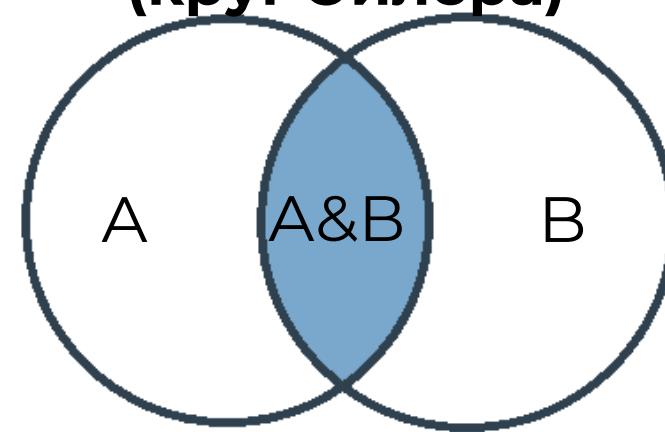
- **Обозначения:  $\wedge$ ,  $\times$ , &, И, and.**



## 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:  $\vee$ , |, ИЛИ, or, +.**

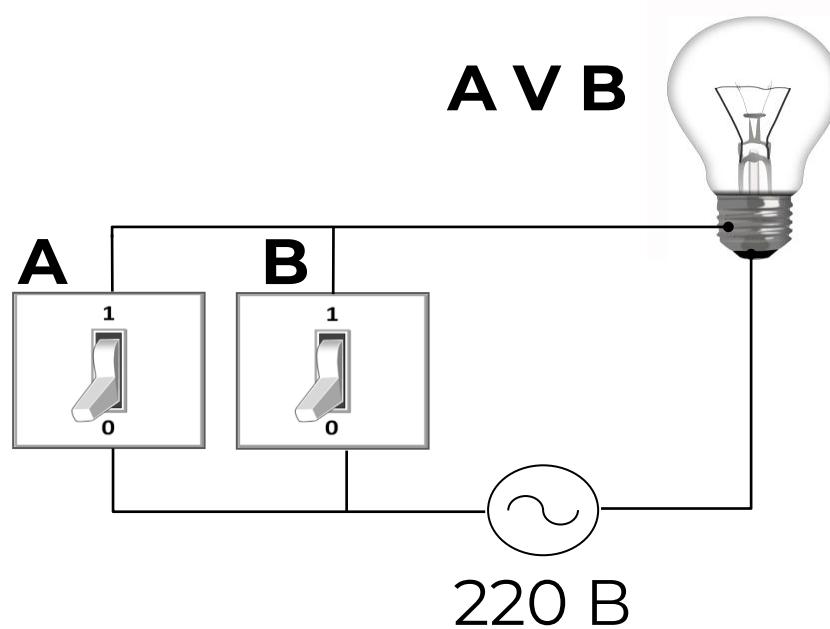
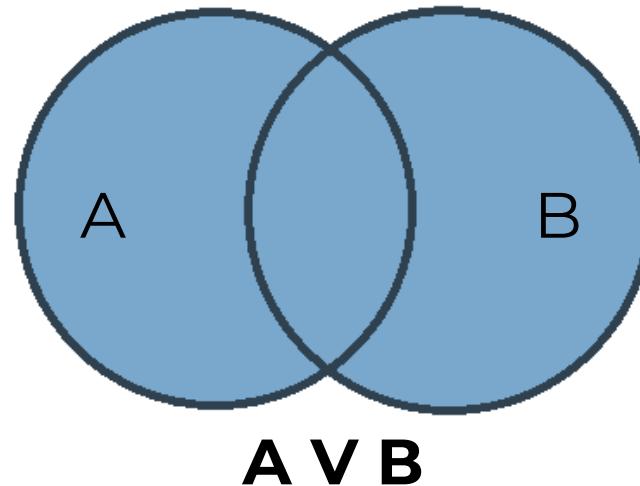


Таблица истинности

| A | B | $F = A \vee B$ |
|---|---|----------------|
| 0 | 0 | 0              |
| 0 | 1 | 1              |
| 1 | 0 | 1              |
| 1 | 1 | 1              |

Диаграмма Венна (круг Эйлера)



# 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.

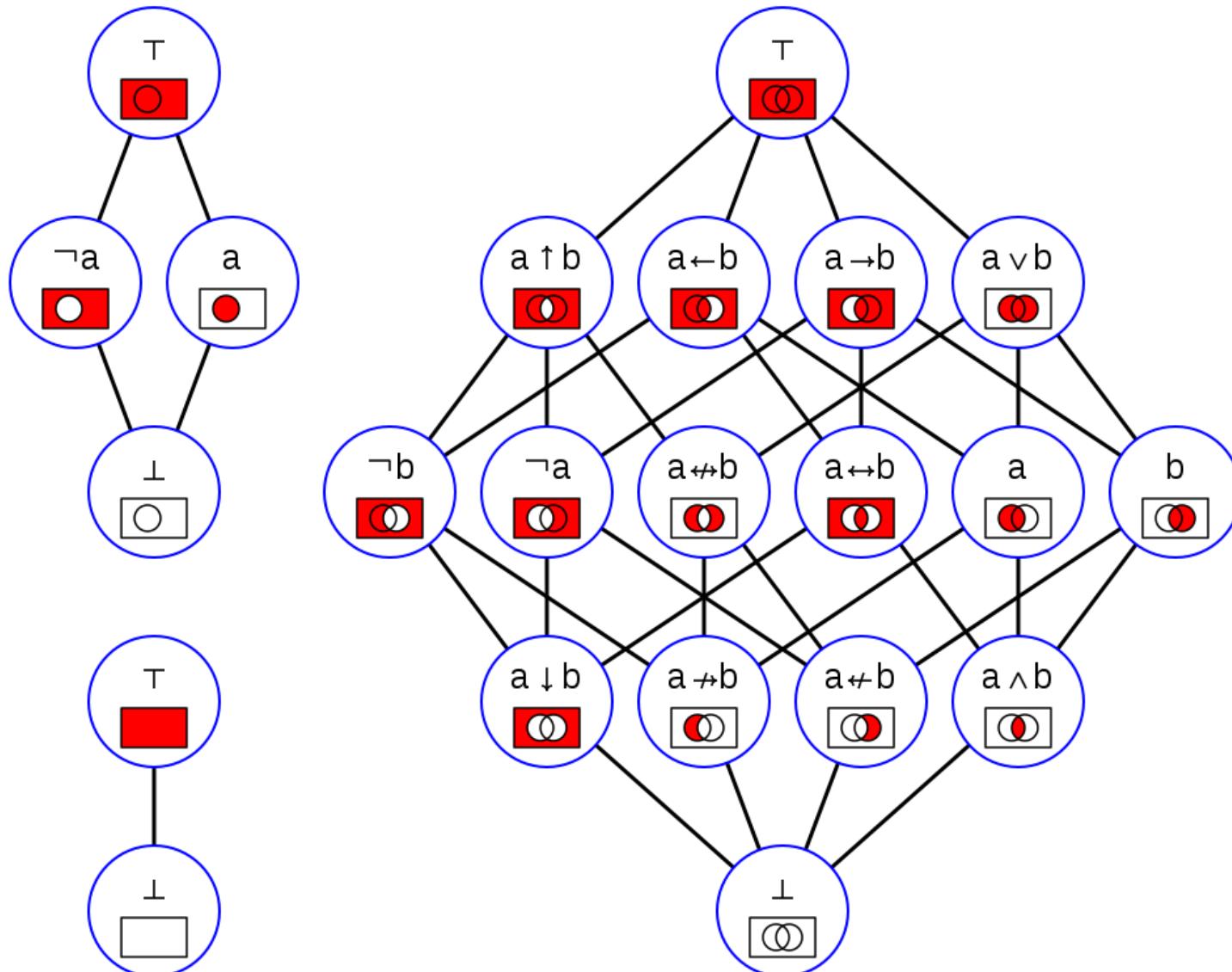
Логические элементы компьютера оперируют с сигналами, представляющими собой электрические импульсы. Есть импульс — логическое значение сигнала 1, нет импульса — значение 0. На вход логического элемента поступают сигналы-аргументы, на выходе появляется сигнал-функция.

Преобразование сигнала логическим элементом задается таблицей состояния, которая фактически является таблицей истинности, соответствующей логической функции.

# Логические операции

| Название операции                                      | Обозначение                           | Речевой оборот                |
|--|---------------------------------------|-------------------------------|
| <b>Инверсия</b> (отрицание, логическое НЕ)             | $\neg A$ ; $\bar{A}$                  | Не A; Неверно, что A;         |
| <b>Дизъюнкция</b> (логическое ИЛИ)                     | $A \vee B$ ; $A + B$                  | A или B или оба вместе;       |
| <b>Конъюнкция</b> (логическое И)                       | $A \wedge B$ ; $A \& B$ ; $A \cdot B$ | A и B вместе;                 |
| <b>Импликация</b> (следование)                         | $A \rightarrow B$                     | если A, то B;                 |
| <b>Эквивалентность</b> (тождество)                     | $A \leftrightarrow B$ ; $A \equiv B$  | A эквивалентно/равносильно B; |
| <b>Стрелка Пирса</b> (ИЛИ-НЕ)                          | $A \downarrow B$                      | Ни A, ни B;                   |
| <b>Штрих Шеффера</b> (И-НЕ)                            | $A   B$                               | Неверно, что A и B;           |
| Сложение по модулю 2 (строгая дизъюнкция, <b>XOR</b> ) | $A \oplus B$                          | Либо A, либо B;               |

# Логические связи



Тавтология  $T$

противоположное соединение  $\uparrow$

противоположное значение  $\leftarrow$

материальный условный  $\rightarrow$

логическая дизъюнкция  $\vee$

логическое отрицание  $\neg$

исключительная дизъюнкция  $\leftrightarrow$

биусловный  $\Leftrightarrow$

логическое утверждение

противоположная дизъюнкция  $\downarrow$

логическое дополнение  $\rightarrowtail$

противоположное примыкание  $\leftarrowtail$

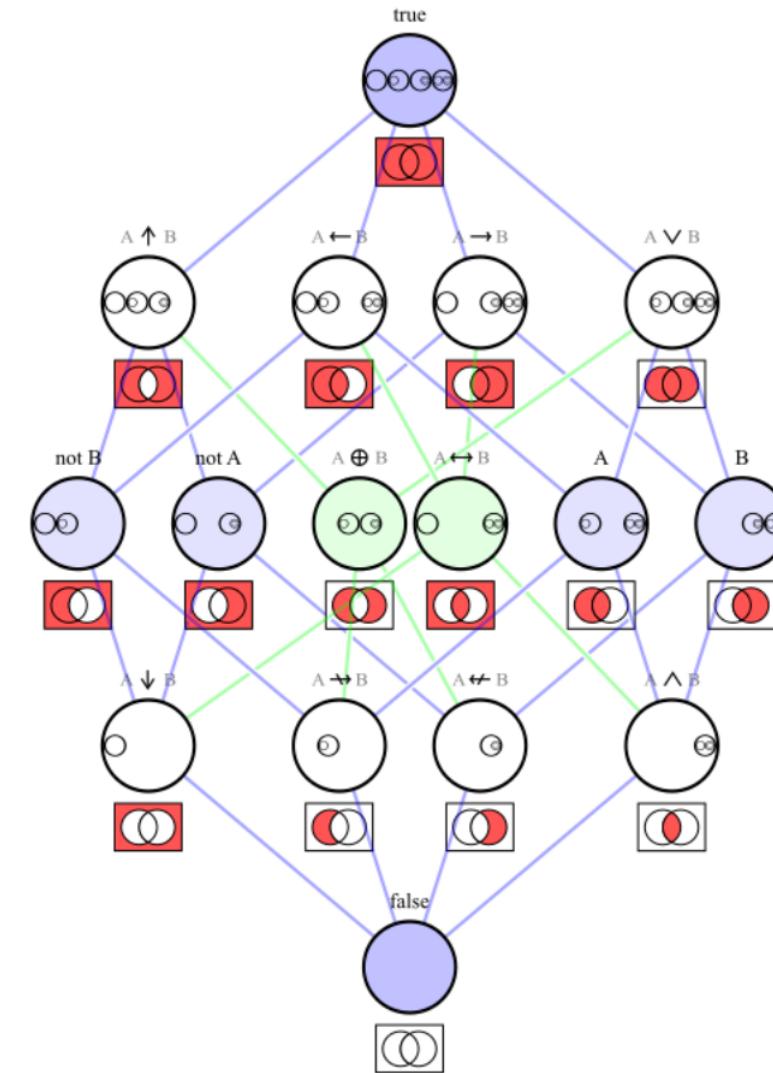
логическое соединение  $\wedge$

Противоречие  $\perp$

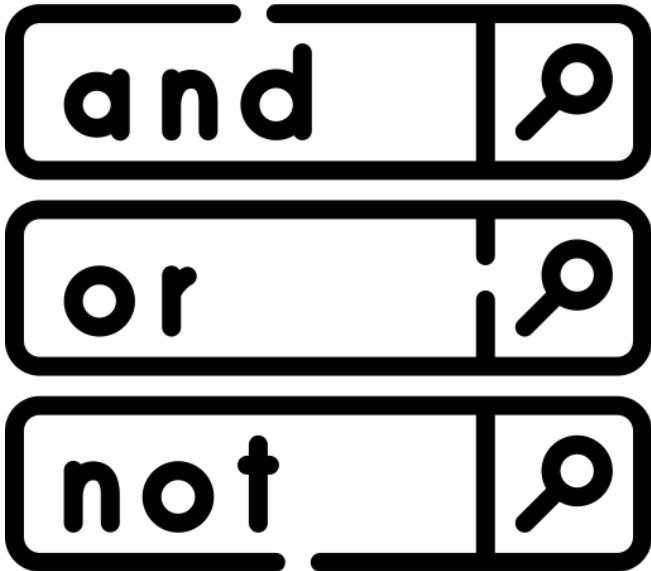
# Логические связки

Таблица логических связок и диаграмма Хассе

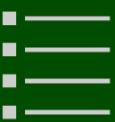
|   |   |   |   |   |
|---|---|---|---|---|
| A   | 0 | 1 | 0 | 1 |
| B   | 0 | 0 | 1 | 1 |
|   | ↓ | ↓ | ↓ | ↓ |
| false   | 0 | 0 | 0 | 0 |
| $A \wedge B \Leftrightarrow \bar{A} \leftrightarrow B \Leftrightarrow A \leftrightarrow \bar{B} \Leftrightarrow \bar{A} \downarrow \bar{B}$ | 0 | 0 | 0 | 1 |
| $A \leftrightarrow B \Leftrightarrow \bar{A} \wedge B \Leftrightarrow A \downarrow \bar{B} \Leftrightarrow \bar{A} \rightarrow \bar{B}$     | 0 | 0 | 1 | 0 |
| B   | 0 | 0 | 1 | 1 |
| $A \rightarrow B \Leftrightarrow \bar{A} \downarrow B \Leftrightarrow A \wedge \bar{B} \Leftrightarrow \bar{A} \leftrightarrow \bar{B}$     | 0 | 1 | 0 | 0 |
| A   | 0 | 1 | 0 | 1 |
| $A \oplus B \Leftrightarrow \bar{A} \leftrightarrow B \Leftrightarrow A \leftrightarrow \bar{B} \Leftrightarrow \bar{A} \oplus \bar{B}$     | 0 | 1 | 1 | 0 |
| $A \vee B \Leftrightarrow \bar{A} \rightarrow B \Leftrightarrow A \leftarrow \bar{B} \Leftrightarrow \bar{A} \uparrow \bar{B}$              | 0 | 1 | 1 | 1 |
| $A \downarrow B \Leftrightarrow \bar{A} \rightarrow B \Leftrightarrow A \leftrightarrow \bar{B} \Leftrightarrow \bar{A} \wedge \bar{B}$     | 1 | 0 | 0 | 0 |
| $A \leftrightarrow B \Leftrightarrow \bar{A} \oplus B \Leftrightarrow A \oplus \bar{B} \Leftrightarrow \bar{A} \leftrightarrow \bar{B}$     | 1 | 0 | 0 | 1 |
| $\bar{A}$   | 1 | 0 | 1 | 0 |
| $A \rightarrow B \Leftrightarrow \bar{A} \vee B \Leftrightarrow A \uparrow \bar{B} \Leftrightarrow \bar{A} \leftarrow \bar{B}$              | 1 | 0 | 1 | 1 |
| $\bar{B}$   | 1 | 1 | 0 | 0 |
| $A \leftarrow B \Leftrightarrow \bar{A} \uparrow B \Leftrightarrow A \vee \bar{B} \Leftrightarrow \bar{A} \rightarrow \bar{B}$              | 1 | 1 | 0 | 1 |
| $A \uparrow B \Leftrightarrow \bar{A} \leftarrow B \Leftrightarrow A \rightarrow \bar{B} \Leftrightarrow \bar{A} \vee \bar{B}$              | 1 | 1 | 1 | 0 |
| true  | 1 | 1 | 1 | 1 |



Logical connective (Логическая связка) [https://en.wikipedia.org/wiki/Logical\\_connective](https://en.wikipedia.org/wiki/Logical_connective)



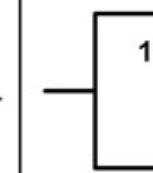
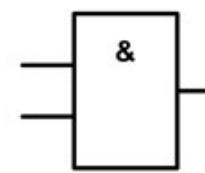
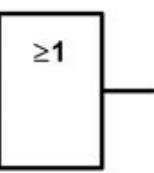
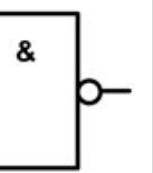
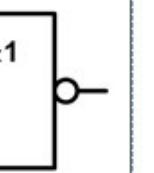
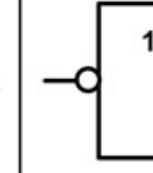
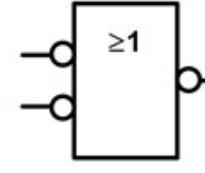
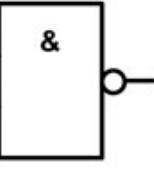
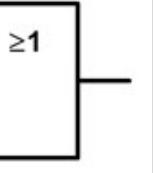
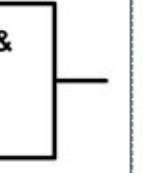
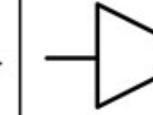
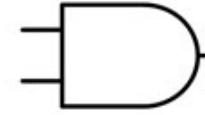
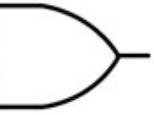
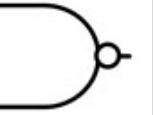
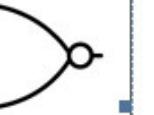
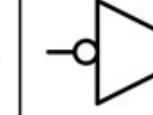
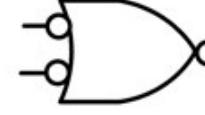
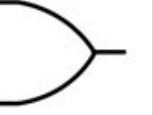
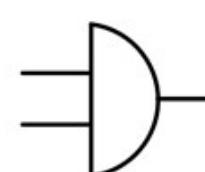
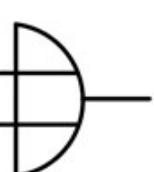
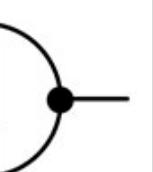
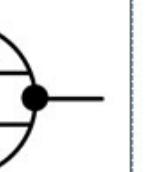
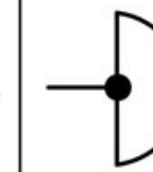
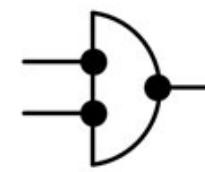
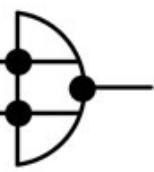
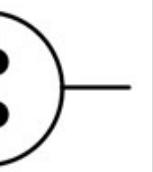
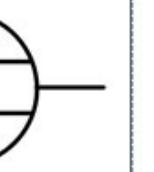
# Уровень цифровой логики



# Основные логические операции

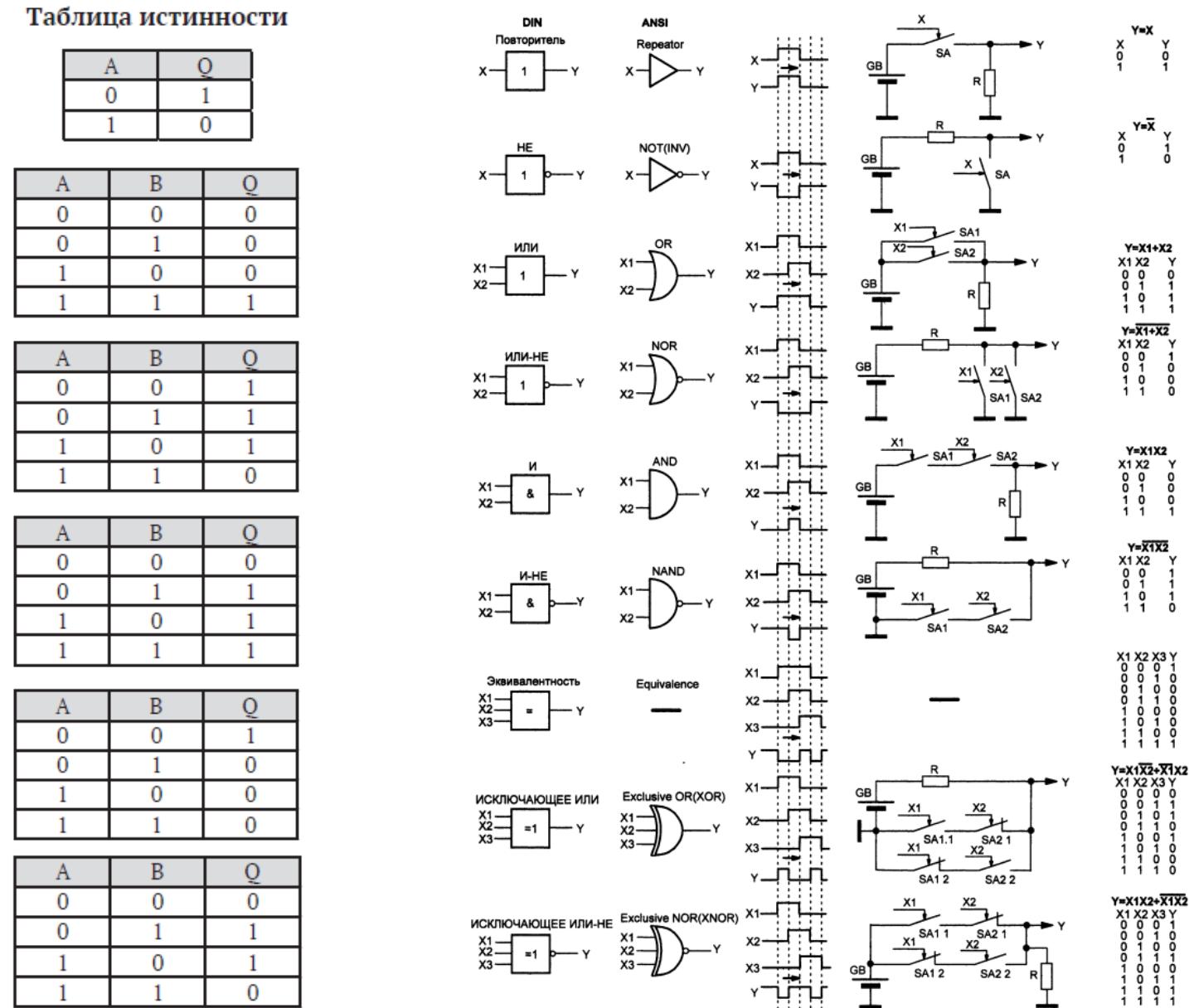
- В цифровых системах логические элементы (или логические вентили) являются строительными блоками, которые выполняют базовые логические операции.
- Логический элемент имеет один выходной терминал и один или несколько входных терминалов. Эти элементы обрабатывают двоичные сигналы (0 и 1) и являются основой для построения более сложных цифровых схем.
- **Логический элемент** – элемент устройства или функциональная группа, реализующая функцию или систему функций двоичной алгебры логики.
- Электронные логические схемы широко используются в калькуляторах, компьютерах, телефонных станциях и во всех приложениях, где задействованы системы с двумя состояниями.
- **Основные логические операции включают: И (AND), ИЛИ (OR), НЕ (NOT), И-НЕ (NAND), ИЛИ-НЕ (NOR), ИСКЛЮЧАЮЩЕЕ ИЛИ (XOR), ИСКЛЮЧАЮЩЕЕ ИЛИ-НЕ (XNOR).**

# Базовые логические элементы

| Логика     | «НЕ»  | «И»  | «ИЛИ»   | «И-НЕ»  | «ИЛИ-НЕ»  |
|------------|---|--|---|---|---|
| ГОСТ и IEC | Полож.<br>   |    |    |    |    |
|            | Отриц.<br>   |    |    |    |    |
| ANSI       | Полож.<br>   |    |    |    |    |
|            | Отриц.<br>   |    |    |    |    |
| DIN        | Полож.<br>  |   |   |   |   |
|            | Отриц.<br> |  |  |  |  |

# Базовые логические элементы

| Вентиль               | Символ | Уравнение                                      | Таблица истинности   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|-----------------------|--------|--|--|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| НЕ (NOT)              |        | $Q = \overline{A}$                             | <table border="1"> <thead> <tr> <th>A</th><th>Q</th></tr> </thead> <tbody> <tr> <td>0</td><td>1</td></tr> <tr> <td>1</td><td>0</td></tr> </tbody> </table>   | A | Q | 0 | 1 | 1 | 0 |   |   |   |   |   |   |   |   |   |
| A                     | Q      |  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0                     | 1      |  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1                     | 0      |  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| И (AND)               |        | $Q = A \cdot B = A \& B$                       | <table border="1"> <thead> <tr> <th>A</th><th>B</th><th>Q</th></tr> </thead> <tbody> <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> </tbody> </table> | A | B | Q | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 |
| A                     | B      | Q  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0                     | 0      | 0  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0                     | 1      | 0  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1                     | 0      | 0  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1                     | 1      | 1  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| И-НЕ (NAND)           |        | $Q = \overline{A \cdot B} = \overline{A \& B}$ | <table border="1"> <thead> <tr> <th>A</th><th>B</th><th>Q</th></tr> </thead> <tbody> <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> </tbody> </table> | A | B | Q | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| A                     | B      | Q  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0                     | 0      | 1  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0                     | 1      | 1  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1                     | 0      | 1  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1                     | 1      | 0  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| ИЛИ (OR)              |        | $Q = A + B$                                    | <table border="1"> <thead> <tr> <th>A</th><th>B</th><th>Q</th></tr> </thead> <tbody> <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> </tbody> </table> | A | B | Q | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| A                     | B      | Q  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0                     | 0      | 0  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0                     | 1      | 1  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1                     | 0      | 1  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1                     | 1      | 1  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| ИЛИ-НЕ (NOR)          |        | $Q = \overline{A + B}$                         | <table border="1"> <thead> <tr> <th>A</th><th>B</th><th>Q</th></tr> </thead> <tbody> <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> </tbody> </table> | A | B | Q | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| A                     | B      | Q  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0                     | 0      | 1  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0                     | 1      | 0  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1                     | 0      | 0  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1                     | 1      | 0  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| Исключающее ИЛИ (XOR) |        | $Q = A \oplus B$                               | <table border="1"> <thead> <tr> <th>A</th><th>B</th><th>Q</th></tr> </thead> <tbody> <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>0</td></tr> </tbody> </table> | A | B | Q | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| A                     | B      | Q  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0                     | 0      | 0  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0                     | 1      | 1  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1                     | 0      | 1  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1                     | 1      | 0  |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |



# Стандарты



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



**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 Commission, 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\\_Language\\_Diagrams\\_19620226.pdf](https://bitsavers.org/pdf/mil-std/MIL-STD-806B_Graphical_Symbols_For_Language_Diagrams_19620226.pdf)

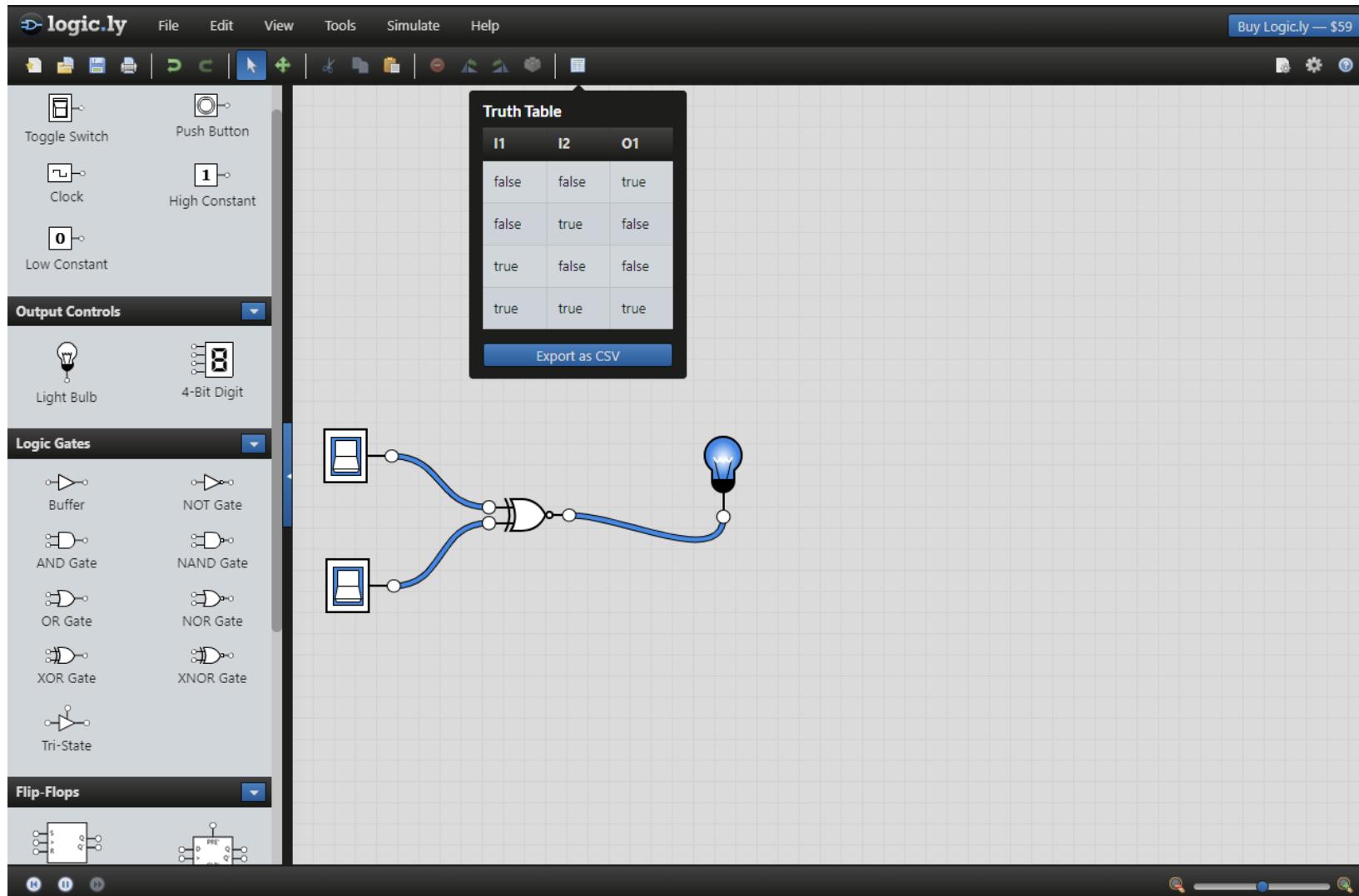
DIN EN 60617

[http://moodle.jrobi.hu/BOCI/moodle/Technikus/DINEN60617\(1999\).pdf](http://moodle.jrobi.hu/BOCI/moodle/Technikus/DINEN60617(1999).pdf)

ГОСТ 2.743-91

<https://meganorm.ru/Data2/1/4294849/4294849507.pdf>

# <https://logic.ly/demo/>

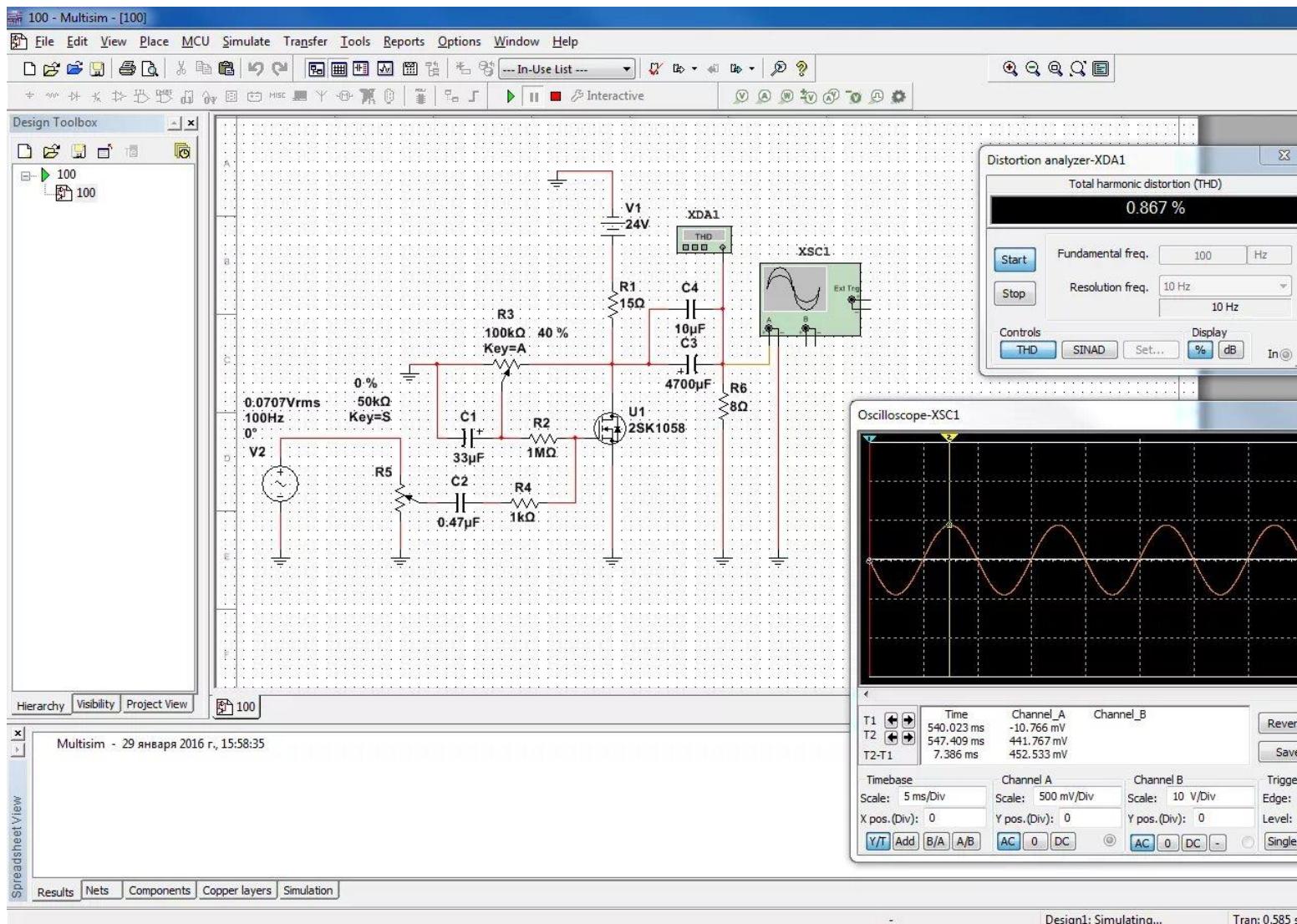


**Сервис logic.ly** позволяет строить логические схемы, используя входные (переключатели) и выходные сигналы (лампочку), а также логические элементы (NOT, OR, AND, XOR). Пользователь может построить схему любой сложности.

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.

# Multisim



**Multisim** – приложение для создания и тестирования электрических схемотехники.

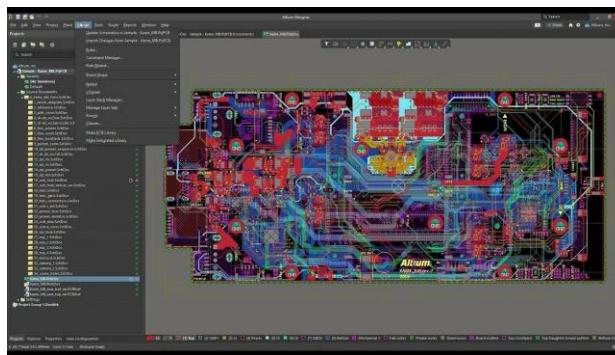
Оно включает сотни виртуальных электронных компонентов и измерительных устройств, умеет моделировать их работу и взаимодействие, что необходимо для проектирования, анализа и отладки электрических схем. Программа поддерживает имитацию различных режимов работы цепей и применяется на всех этапах их разработки.

NI Circuit Design Suite 14.3 x64 [Multisim & Ultiboard, 2022, ENG] <https://rutracker.org/forum/viewtopic.php?t=6225014>

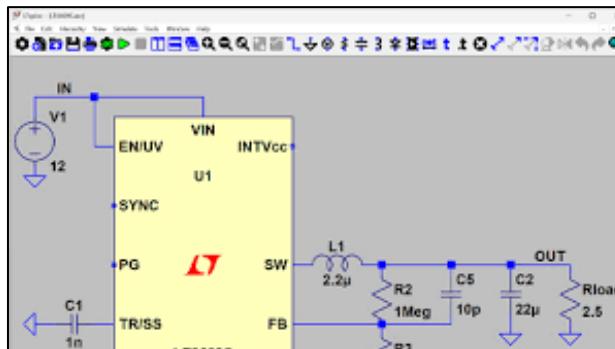
NI Circuit Design Suite 14.2 [Multisim & Ultiboard, EN/RU] <https://rutracker.org/forum/viewtopic.php?t=4994297>

Multisim & Ultiboard Power Pro 14.1.31 + Rus 14.1.31 PRO x86 x64 [2017, ENG + RUS] <https://rutracker.org/forum/viewtopic.php?t=5870078>

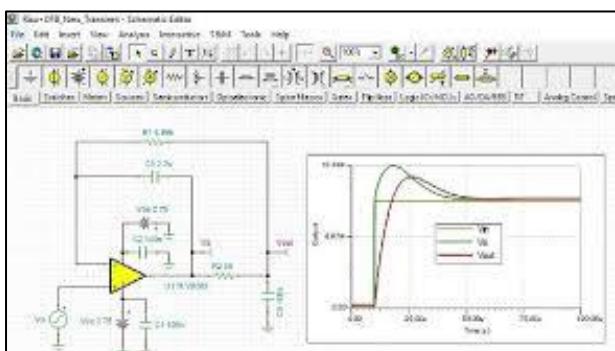
# Программное обеспечение



**Altium Designer** — это комплексный программный пакет для автоматизации проектирования электроники (EDA), который в основном используется для проектирования печатных плат (ПП). Он предоставляет единую среду для создания схем, компоновки печатных плат, моделирования и формирования производственных данных. Программное обеспечение разработано компанией Altium Limited и известно своей интегрированной средой проектирования и способностью работать с проектами от простых схем до сложных систем.



**LTspice (ранее SwitcherCAD)** - это бесплатный SPICE-симулятор, разработанный Analog Devices (ранее Linear Technology), который используется для моделирования аналоговых электронных схем. Он популярен как среди любителей, так и среди профессиональных инженеров благодаря своей бесплатной лицензии, широкому функционалу и высокой скорости работы. LTspice включает в себя редактор схем, симулятор и инструмент для анализа форм сигналов.



**TINA-TI** - SPICE-симулятор, предназначенный для проектирования, симуляции и отладки различных схем электронных устройств. TINA-TI представляет собой обычный SPICE-симулятор с простым, интуитивно понятным графическим интерфейсом, позволяющим освоить программу в кратчайшие сроки. Данный софт не имеет каких-либо ограничений на число используемых устройств и узлов, без проблем справляется с комплексными работами, идеально подходит для моделирования поведения различных аналоговых схем и импульсных источников питания. При помощи TINA-TI возможно «с чистого листа» создать проект любой сложности, объединить фрагменты уже готовых решений, проверить и определить некоторые качественные показатели схемы.

# Логический элемент: И (AND)

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

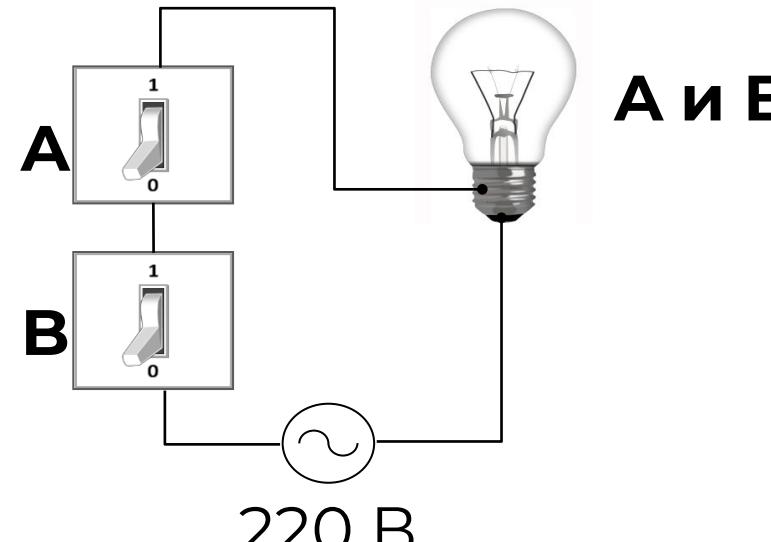
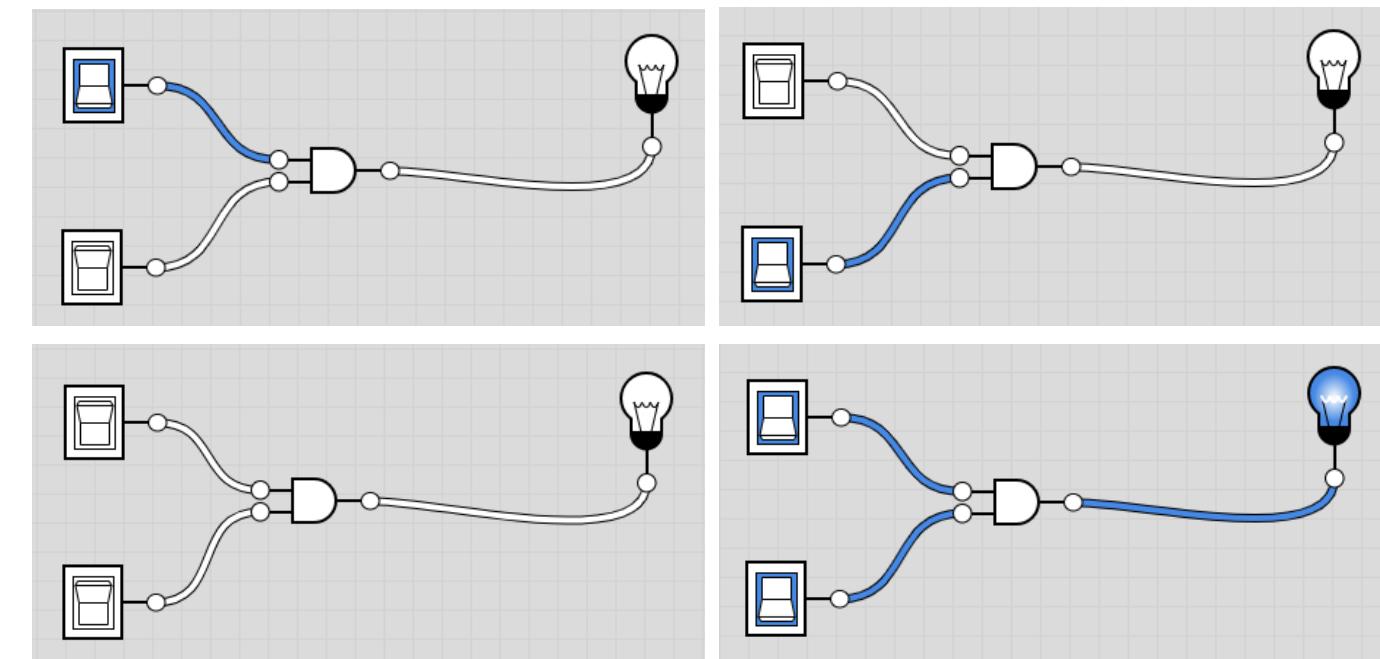
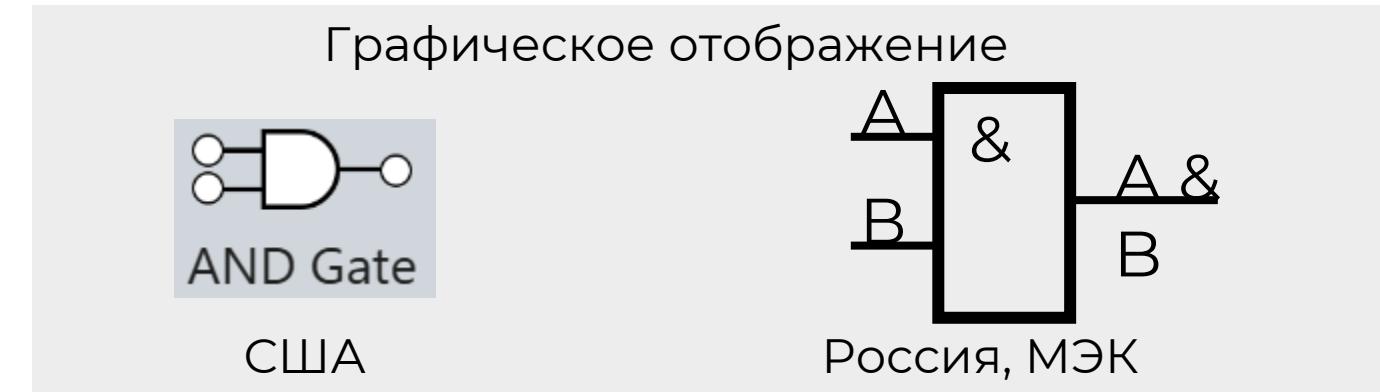


Таблица истинности

| A | B | А и В |
|---|---|-------|
| 0 | 0 | 0     |
| 0 | 1 | 0     |
| 1 | 0 | 0     |
| 1 | 1 | 1     |



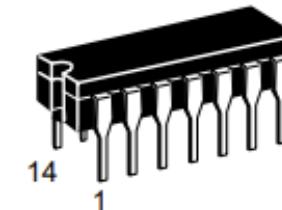
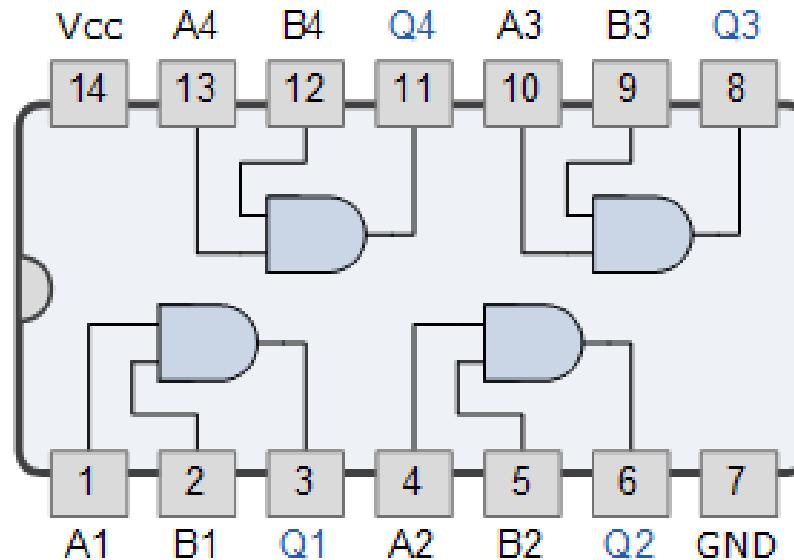
# Логический элемент: И (AND)

Пример микросхемы с логическим элементом «И»

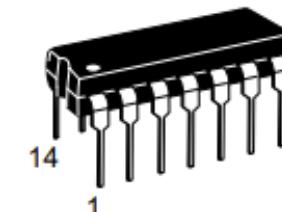


**QUAD 2-INPUT AND GATE**

**SN54/74LS08**



J SUFFIX  
CERAMIC  
CASE 632-08



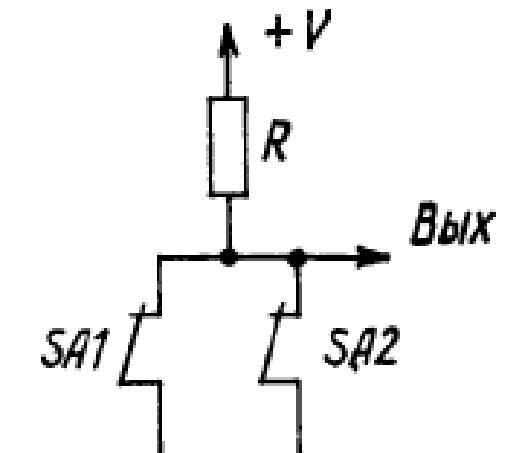
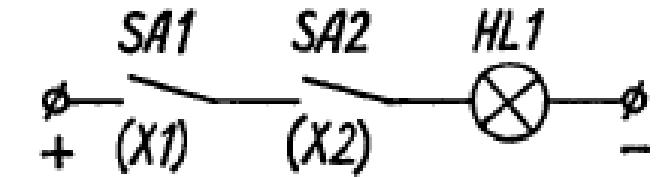
N SUFFIX  
PLASTIC  
CASE 646-06



D SUFFIX  
SOIC  
CASE 751A-02

## ORDERING INFORMATION

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



# Логический элемент: И (AND)

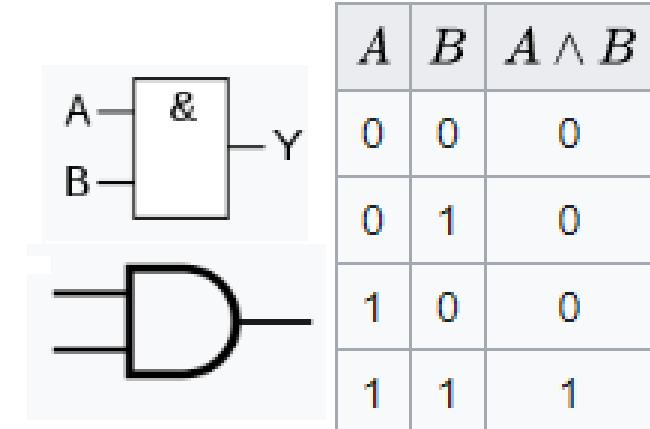
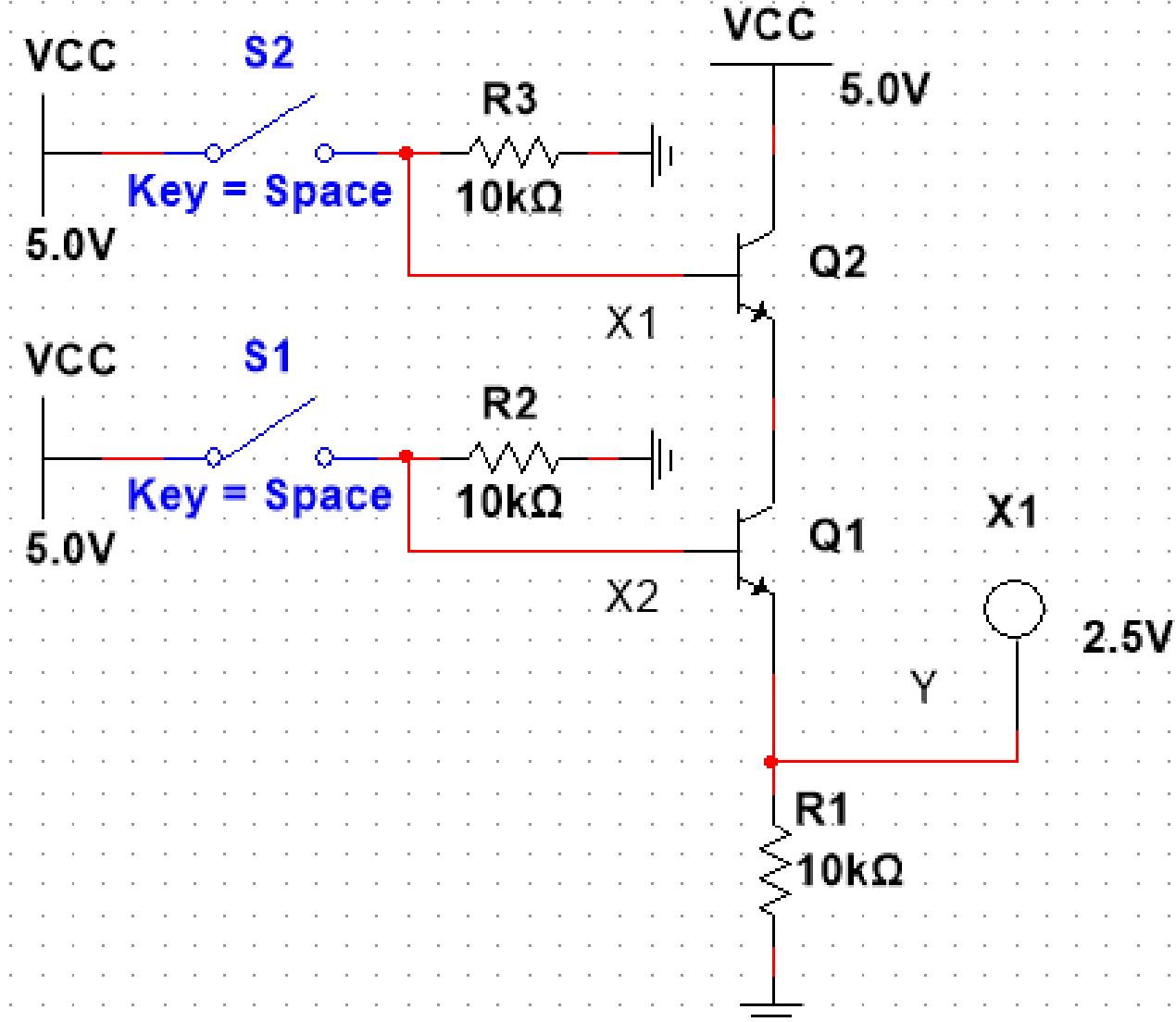


Схема элемента И на РТЛ (резисторно-транзисторная логика)

# Логический элемент: ИЛИ (OR)

Логическая схема «ИЛИ» называется также дизъюнктором, выполняет операцию логического сложения (дизъюнкции), может иметь от двух до восьми входов.

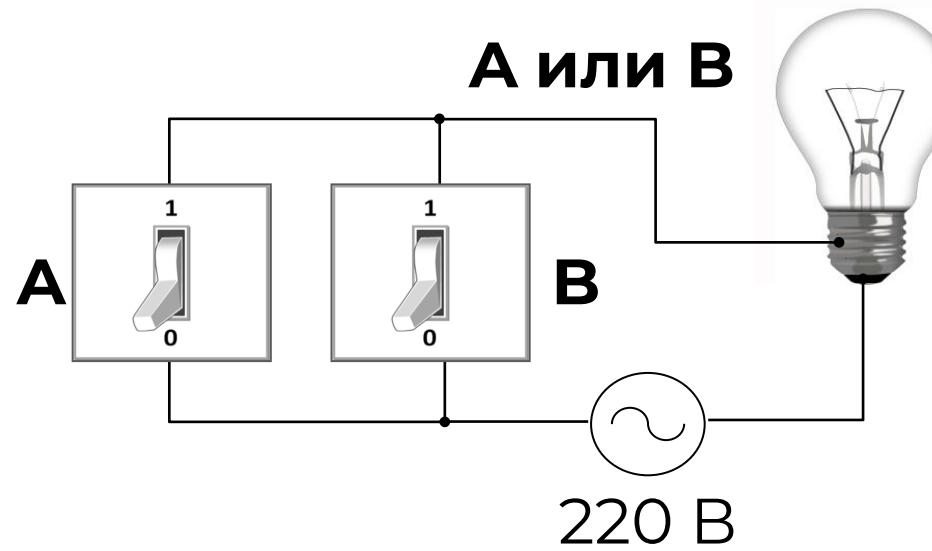
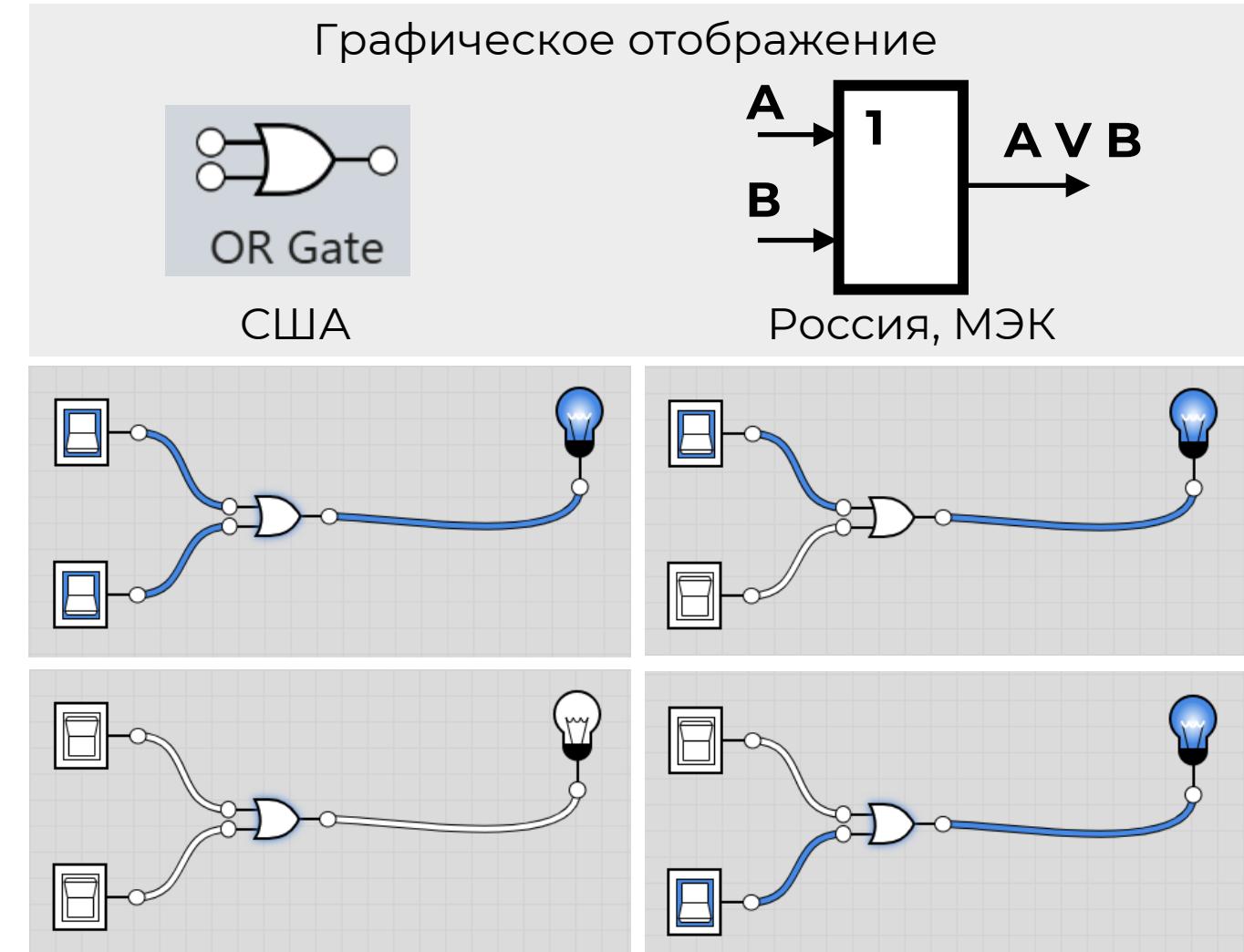


Таблица истинности

| A | B | А или B |
|---|---|---------|
| 0 | 0 | 0       |
| 0 | 1 | 1       |
| 1 | 0 | 1       |
| 1 | 1 | 1       |



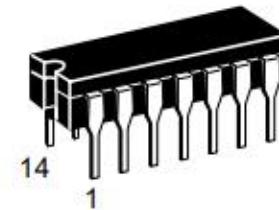
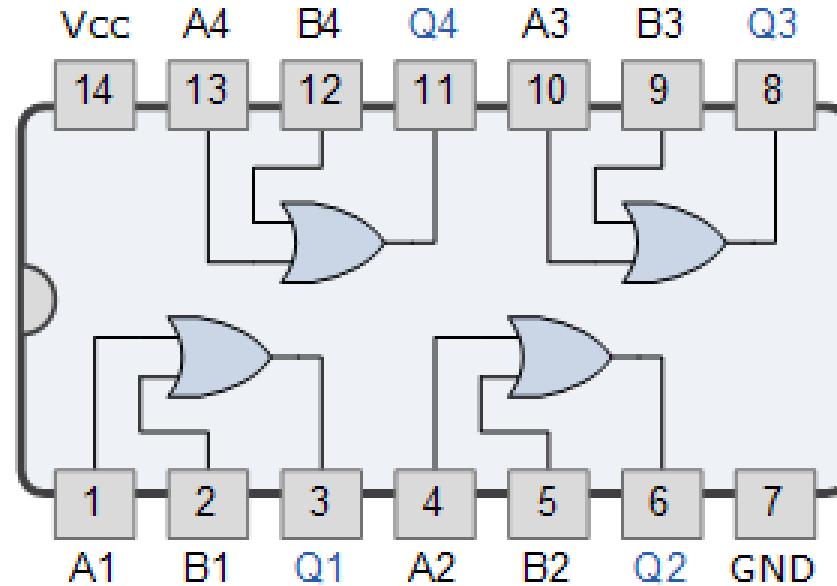
# Логический элемент: ИЛИ (OR)

Пример микросхемы с логическим элементом «ИЛИ»

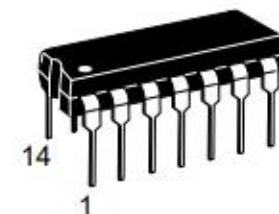


QUAD 2-INPUT OR GATE

SN54/74LS32



J SUFFIX  
CERAMIC  
CASE 632-08



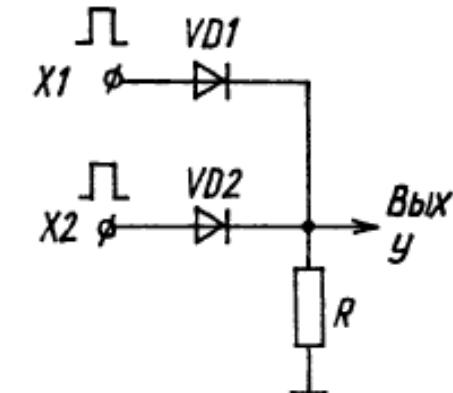
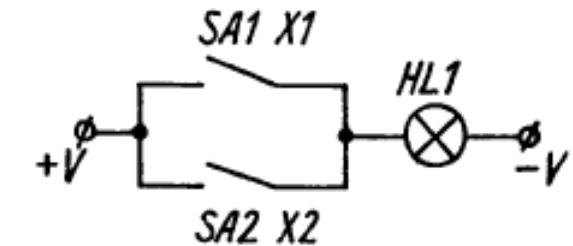
N SUFFIX  
PLASTIC  
CASE 646-06



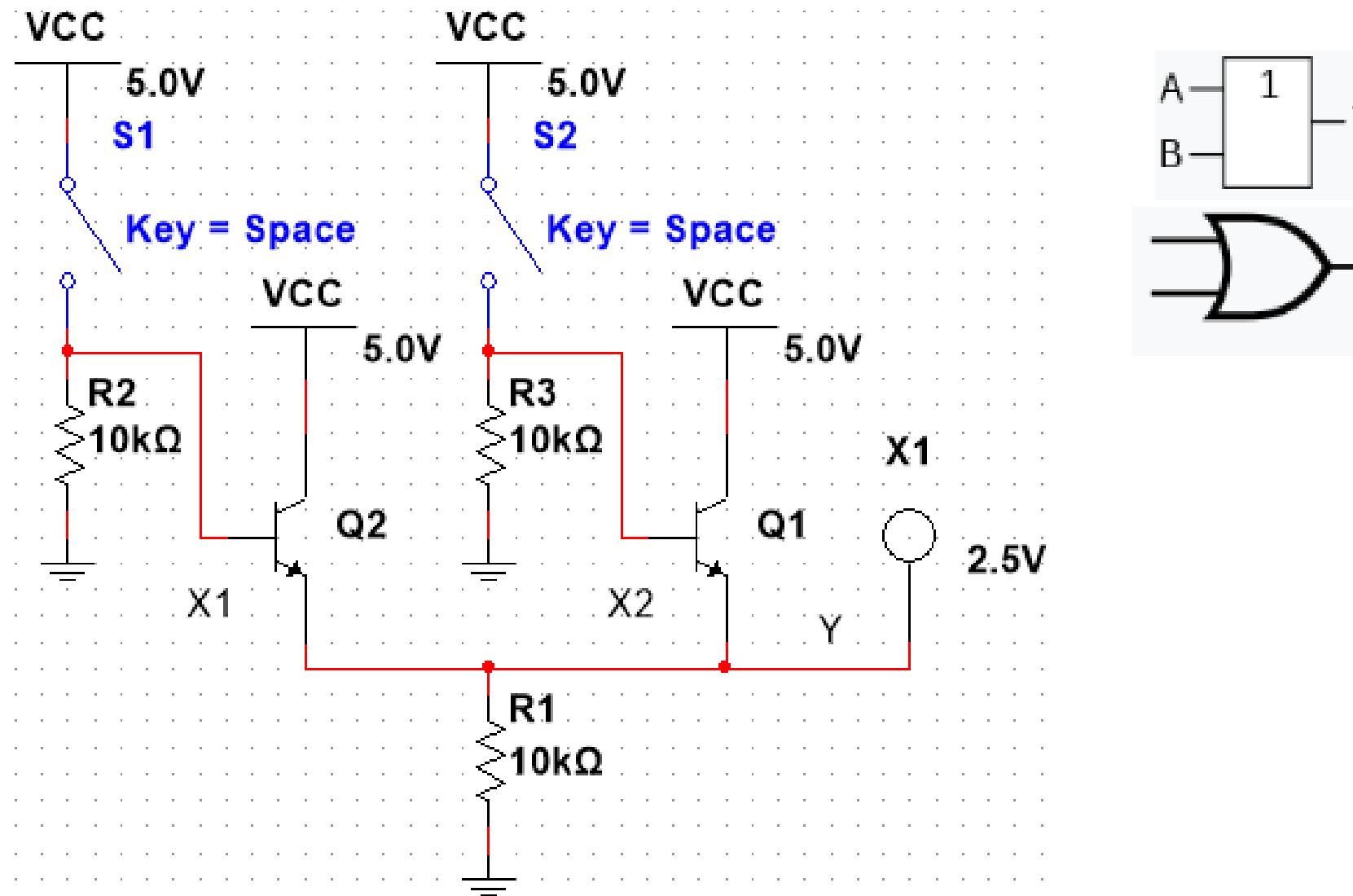
D SUFFIX  
SOIC  
CASE 751A-02

## ORDERING INFORMATION

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



# Логический элемент: ИЛИ (OR)



| A | B | $A \vee B$ |
|---|---|------------|
| 0 | 0 | 0          |
| 0 | 1 | 1          |
| 1 | 0 | 1          |
| 1 | 1 | 1          |

Схема элемента ИЛИ на РТЛ (резисторно-транзисторная логика)

# Логический элемент: НЕ (NOT)

**Логическая схема «НЕ»** – Инверсия – NOT – называется также инвертором, выполняет логическую операцию отрицания (инверсии).

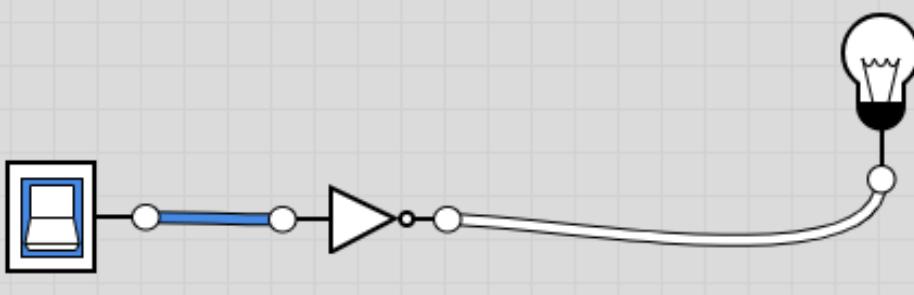
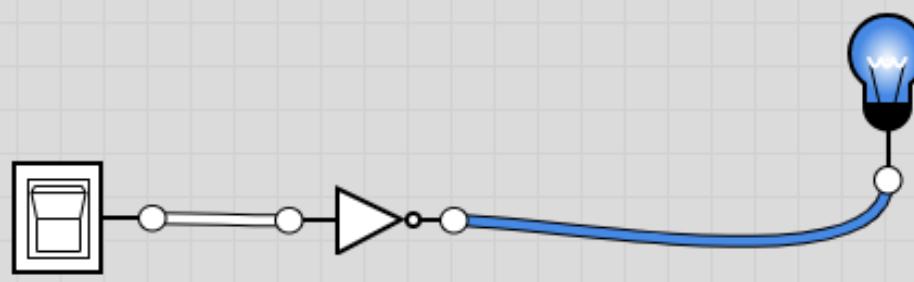
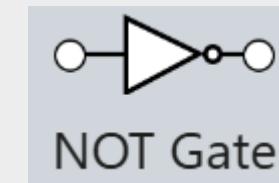


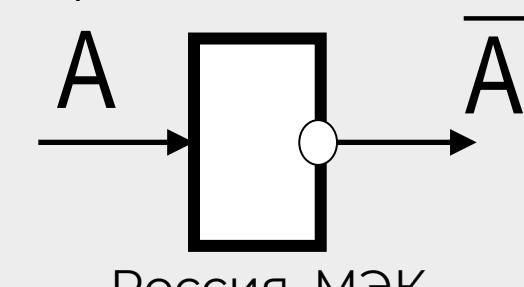
Таблица истинности

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

Графическое отображение

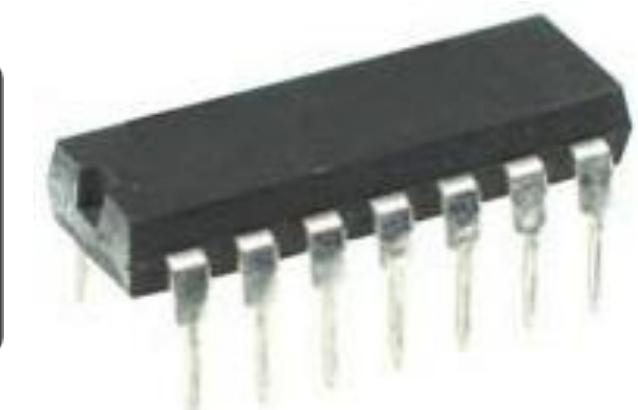
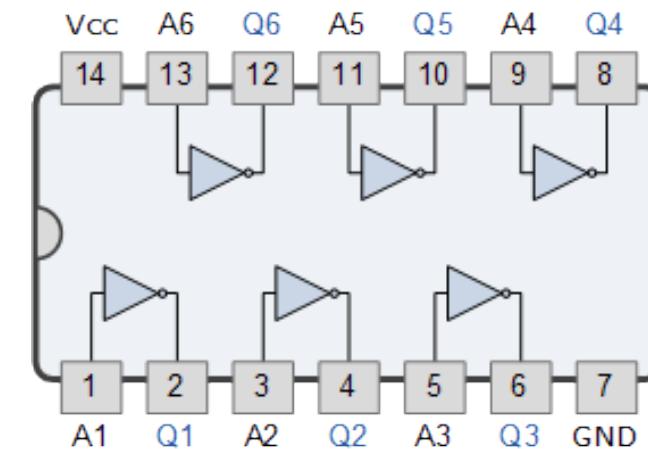


США



Россия, МЭК

Пример микросхемы с логическим элементом «НЕ»



IC 7404 (NOT gate Inverter)

# Логический элемент: НЕ (NOT)

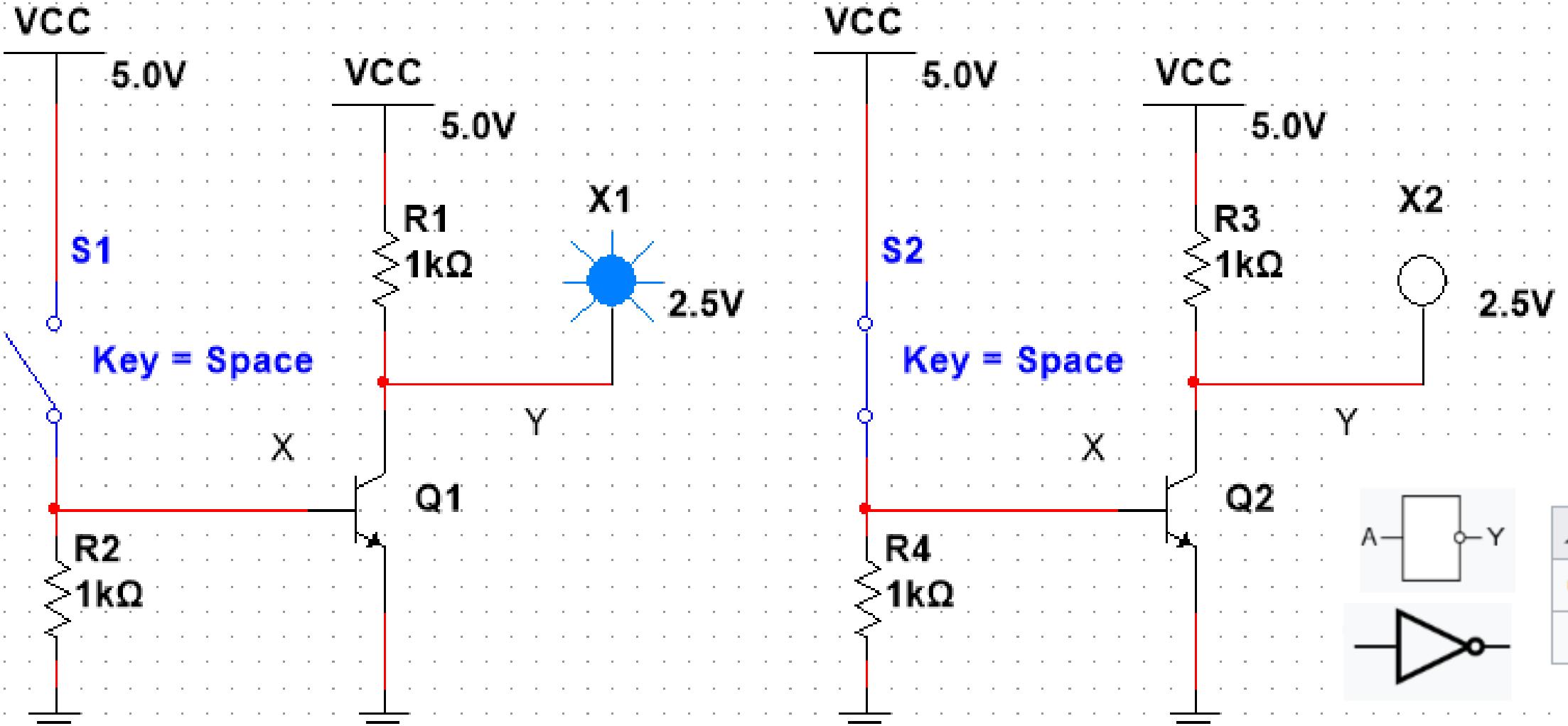


Схема элемента НЕ на РТЛ (резисторно-транзисторная логика)

# Логический элемент: И-НЕ (NAND)

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».

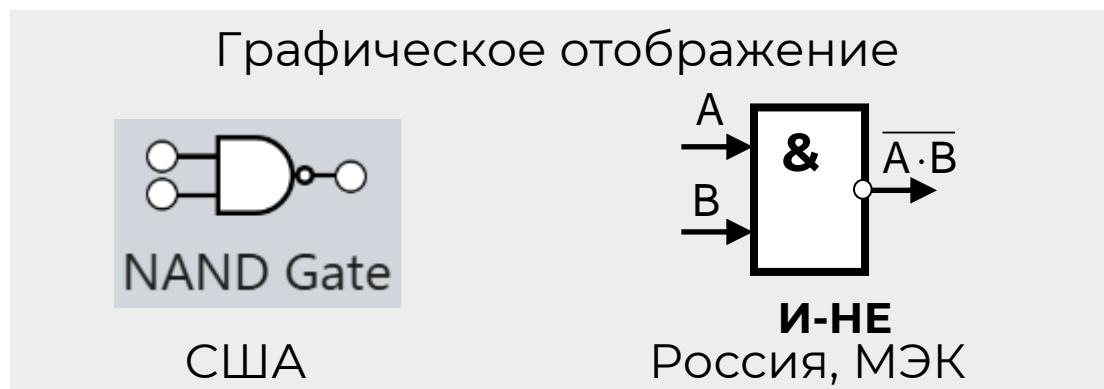
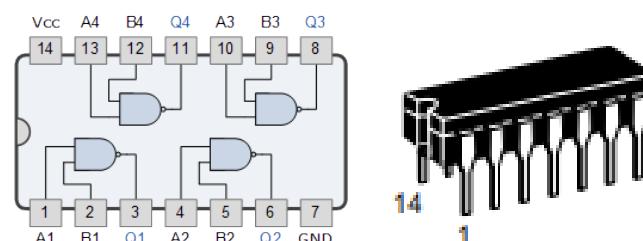


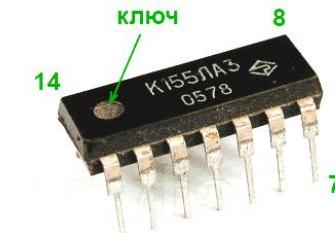
Таблица истинности

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

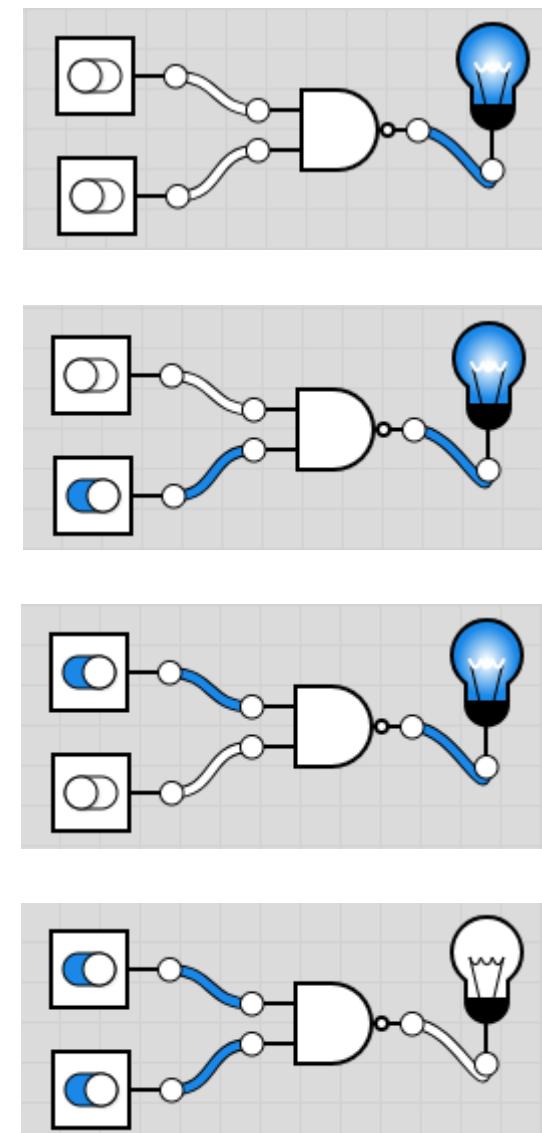
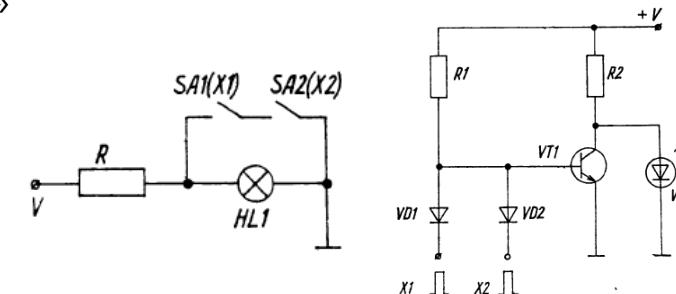
Пример микросхемы с логическим элементом «И-НЕ»



7400 Quad 2-input Logic NAND Gate



Микросхема К155ЛА3



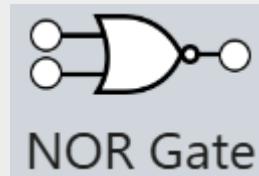
# Логический элемент: ИЛИ-НЕ (NOR)

Инверсия функции дизъюнкции.

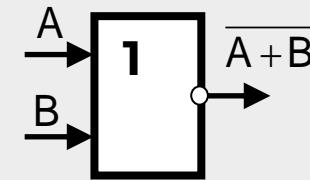
Операция «ИЛИ-НЕ» (стрелка Пирса)

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».

Графическое отображение



США

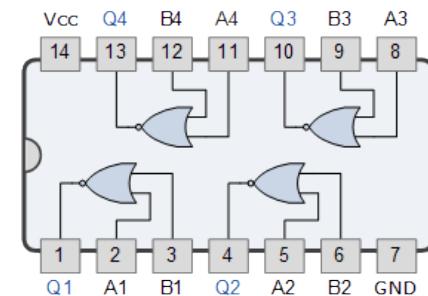


ИЛИ-НЕ  
Россия, МЭК

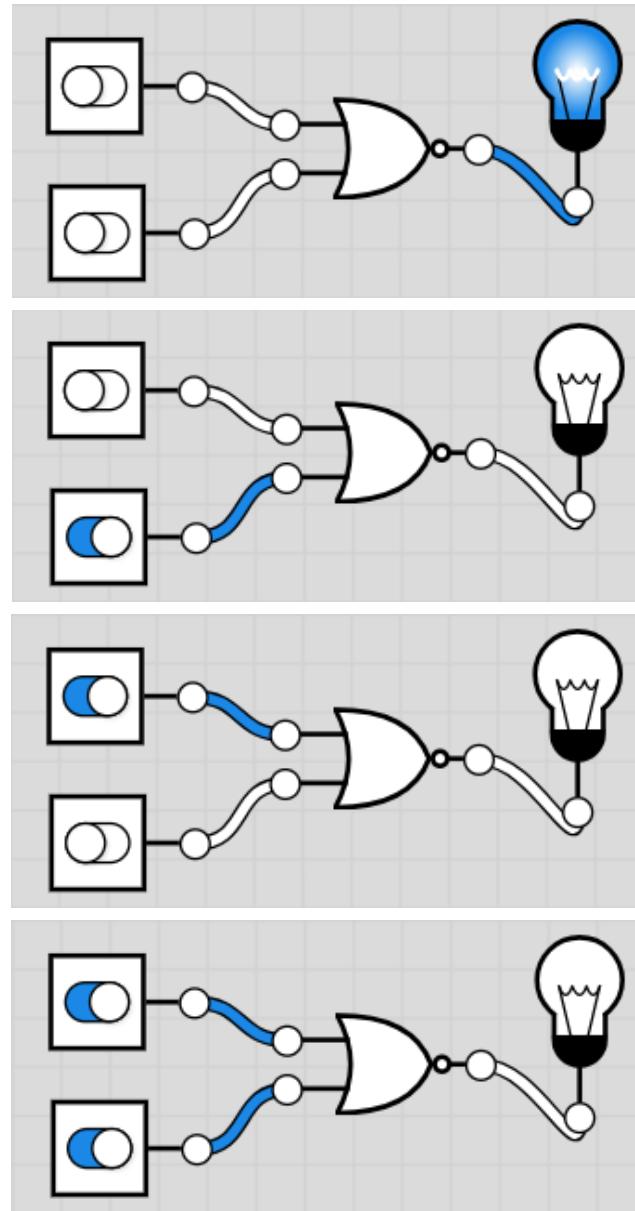
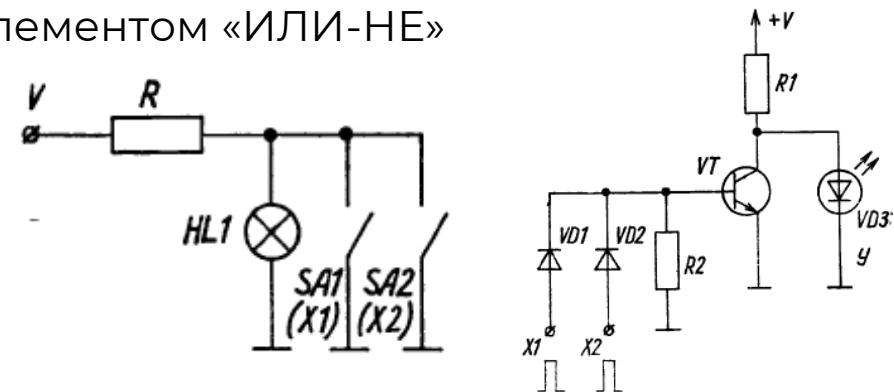
Таблица истинности

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

Пример микросхемы с логическим элементом «ИЛИ-НЕ»



7402 Quad 2-input Logic NOR Gate



# Логический элемент: Исключающее ИЛИ (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».

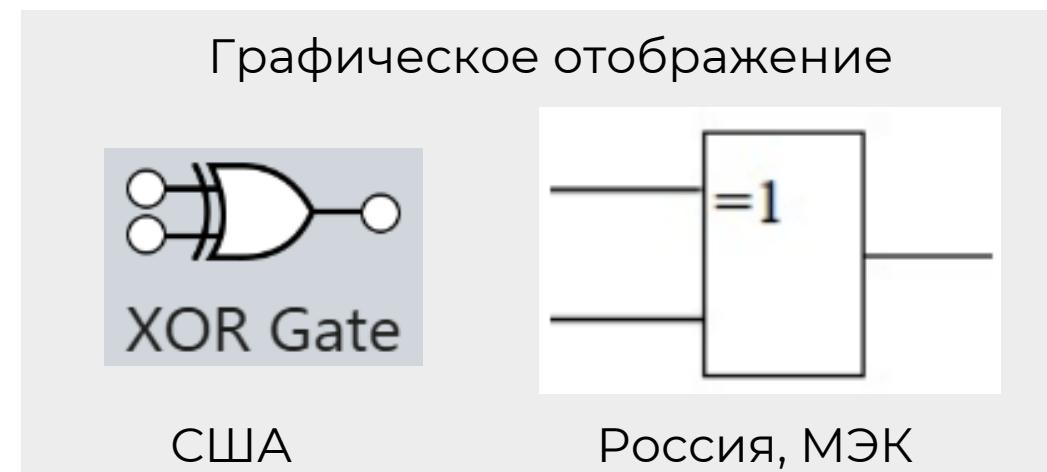
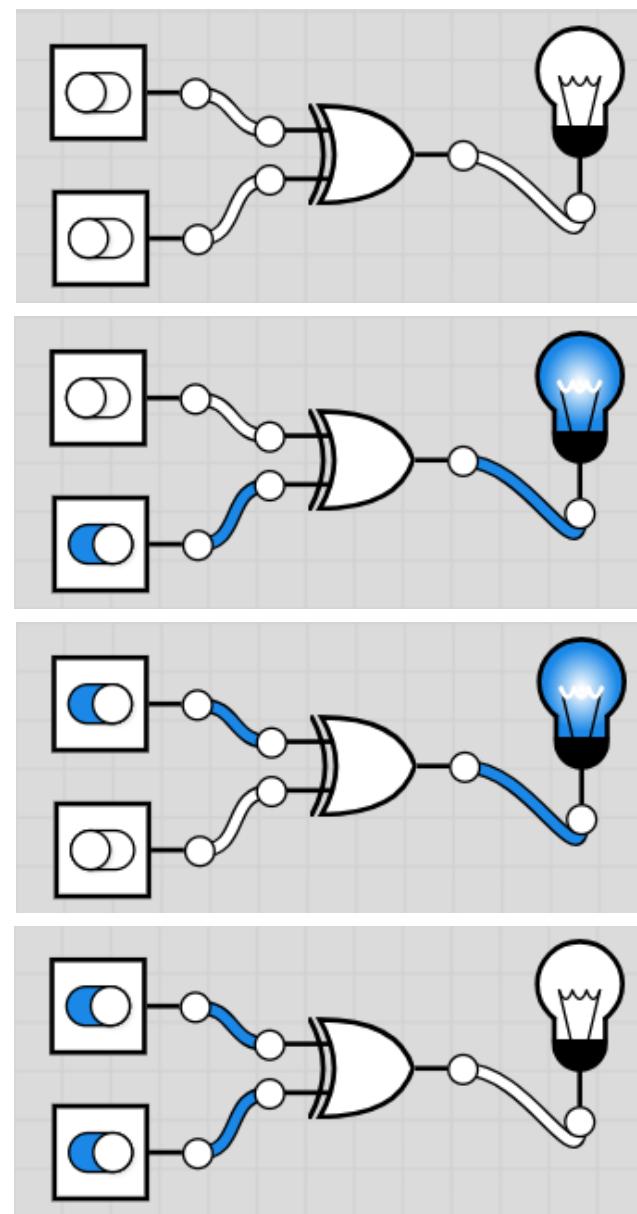


Таблица истинности

| A | B | $A \oplus B$ |
|---|---|--------------|
| 0 | 0 | 0            |
| 0 | 1 | 1            |
| 1 | 0 | 1            |
| 1 | 1 | 0            |

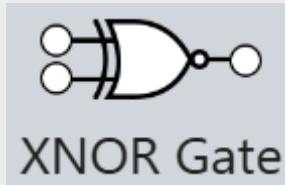


# Логический элемент: Исключающее ИЛИ-НЕ (XNOR)

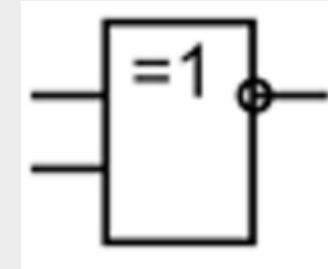
## ИСКЛЮЧАЮЩЕЕ ИЛИ-НЕ (EXCLUDING NOR, XNOR) –

логический элемент, для двухходового варианта которого выходной сигнал принимает значение логической единицы, если уровни входных сигналов совпадают (оба логических нуля или обе логические единицы). Стоит нарушить это условие, сигнал на выходе элемента примет значение логического нуля.

Графическое отображение



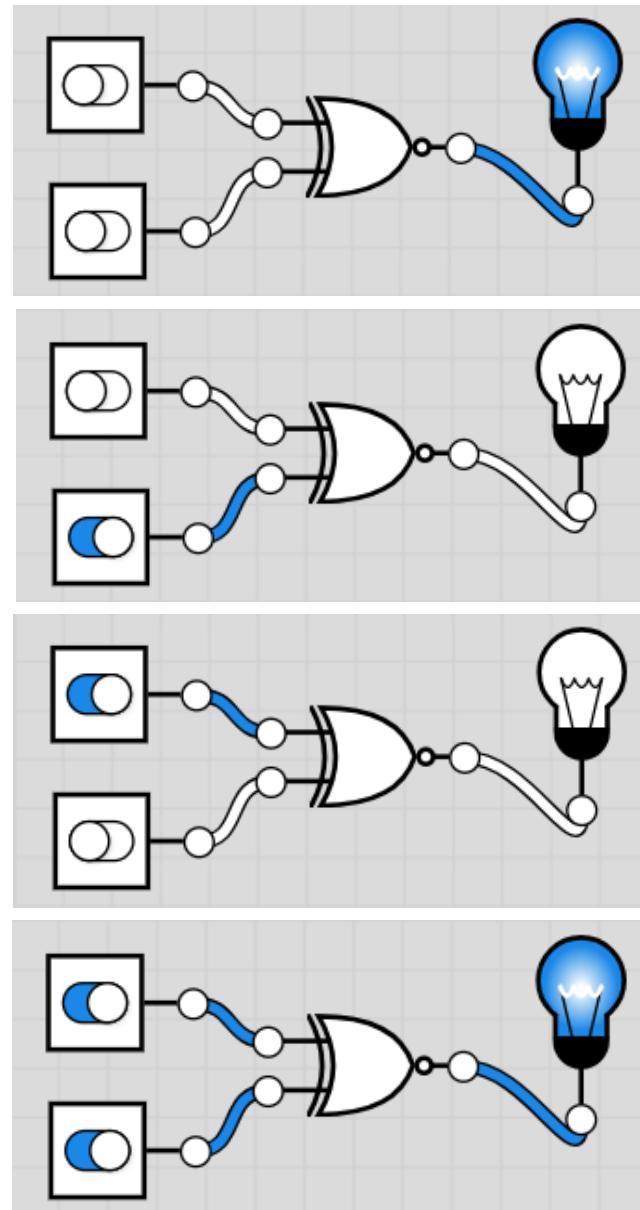
США



Россия, МЭК

Таблица истинности

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

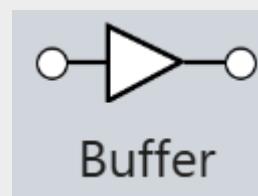


# Логический элемент: Буфер

**Цифровой буфер (или логический буфер)** — это элемент электронной схемы, используемый для копирования цифрового входного сигнала и его изоляции от любой выходной нагрузки. В типичном случае использования напряжения в качестве логических сигналов входное сопротивление логического буфера высокое, поэтому он потребляет мало тока от входной цепи, чтобы не нарушать сигнал.

Цифровой буфер важен для передачи данных между подключёнными системами. Буферы используются в регистрах (устройствах хранения данных) и шинах (устройствах передачи данных). Для подключения к общейшине следует использовать трёхпозиционный цифровой буфер, поскольку он имеет высокий импеданс («неактивное» или «отключённое») выходное состояние (в дополнение к логическому нулю и логической единице).

Графическое отображение



США

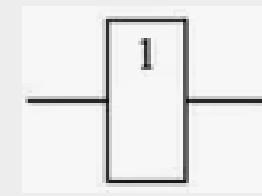
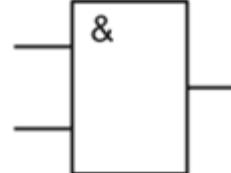
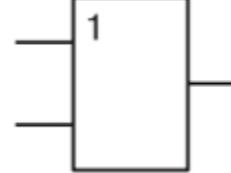
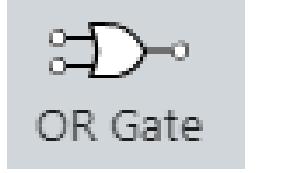
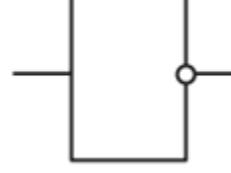
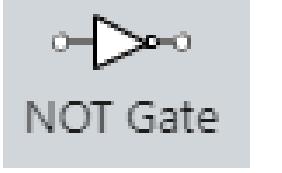
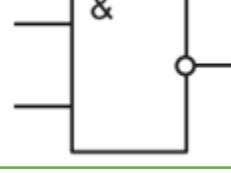
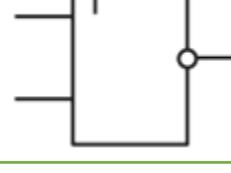


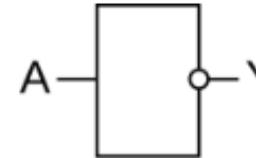
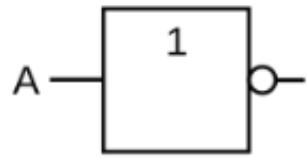
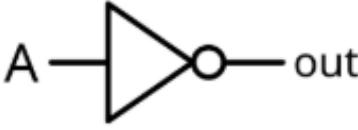
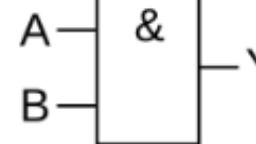
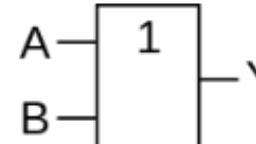
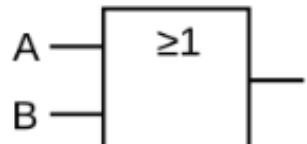
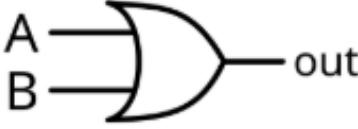
Таблица истинности

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

# Условные обозначения типовых логических элементов

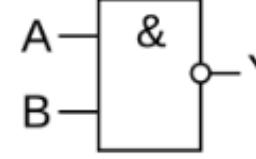
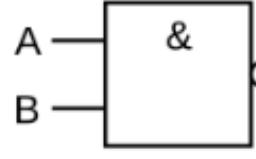
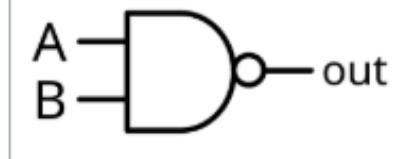
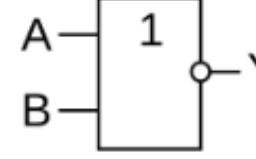
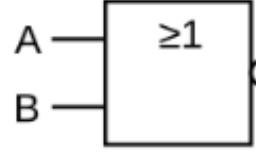
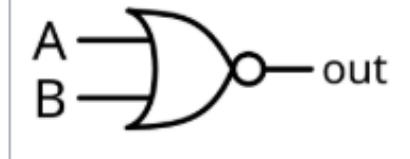
| Наименование элемента         | Условное обозначение   | Обозначение в logic.ly  | Название функции и её формула                         |
|-------------------------------|--|---|---|
| <b>И</b><br>AND               |    | <br>AND Gate   | Конъюнкция<br>$F = A \& B$                            |
| 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

# Условные обозначения типовых логических элементов

| Логический вентиль            | Условные графические обозначения  |  |   | Функция, запись   | Таблица истинности   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|-------------------------------|---|--|---|---|--|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|                               | ГОСТ 2.743-91   | IEC 60617-12 : 1997  | US ANSI 91-1984   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| <b>НЕ</b><br>(англ. NOT gate) |    |    |    | Отрицание<br>$Y = \bar{A}$<br>$Y = \neg A$<br>$Y = \tilde{A}$                 | <table border="1" data-bbox="2086 338 2291 568"> <tr> <td>A</td> <td>Y</td> </tr> <tr> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> </tr> </table>   | A | Y | 0 | 1 | 1 | 0 |   |   |   |   |   |   |   |   |   |
| A                             | Y   |  |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0                             | 1   |  |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1                             | 0   |  |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| <b>И</b><br>(англ. AND gate)  |    |    |    | Конъюнкция<br>$Y = A \wedge B$<br>$Y = A \cdot B$<br>$Y = A \& B$<br>$Y = AB$ | <table border="1" data-bbox="2060 597 2342 972"> <tr> <td>A</td> <td>B</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>  | A | B | Y | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 |
| A                             | B   | Y  |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0                             | 0   | 0  |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0                             | 1   | 0  |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1                             | 0   | 0  |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1                             | 1   | 1  |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| <b>ИЛИ</b><br>(англ. OR gate) |  |  |  | Дизъюнкция<br>$Y = A \vee B$<br>$Y = A + B$                                   | <table border="1" data-bbox="2060 986 2342 1360"> <tr> <td>A</td> <td>B</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> | A | B | Y | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| A                             | B   | Y  |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0                             | 0   | 0  |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0                             | 1   | 1  |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1                             | 0   | 1  |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1                             | 1   | 1  |   |   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |

Логический вентиль [https://ru.wikipedia.org/wiki/Логический\\_вентиль](https://ru.wikipedia.org/wiki/Логический_вентиль)

# Условные обозначения типовых логических элементов

| Логический<br>вентиль   | Условные графические обозначения   |   |  | Функция,<br>запись  | Таблица<br>истинности   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|---|--|---|--|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|   | ГОСТ 2.743-91  | IEC 60617-12 : 1997   | US ANSI 91-1984  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| <b>НЕ И (И-НЕ)</b><br>(англ. <i>NAND gate</i> )<br>Элемент Шеффера  |   |   |   | $Y = \overline{A \wedge B}$<br>$Y = A \bar{\wedge} B$<br>$Y = \overline{A \cdot B}$<br>$Y = \overline{AB}$<br>$Y = A B$ | <table border="1"> <thead> <tr> <th>A</th> <th>B</th> <th>Y</th> </tr> </thead> <tbody> <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> </tbody> </table> | A | B | Y | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| A   | B  | Y   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0   | 0  | 1   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0   | 1  | 1   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1   | 0  | 1   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1   | 1  | 0   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| <b>НЕ ИЛИ (ИЛИ-НЕ)</b><br>(англ. <i>NOR gate</i> )<br>Элемент Пирса |  |  |  | $Y = \overline{A \vee B}$<br>$Y = A \bar{\vee} B$<br>$Y = \overline{A + B}$<br>$Y = A - B$                              | <table border="1"> <thead> <tr> <th>A</th> <th>B</th> <th>Y</th> </tr> </thead> <tbody> <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> </tbody> </table> | A | B | Y | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| A   | B  | Y   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0   | 0  | 1   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0   | 1  | 0   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1   | 0  | 0   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1   | 1  | 0   |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |

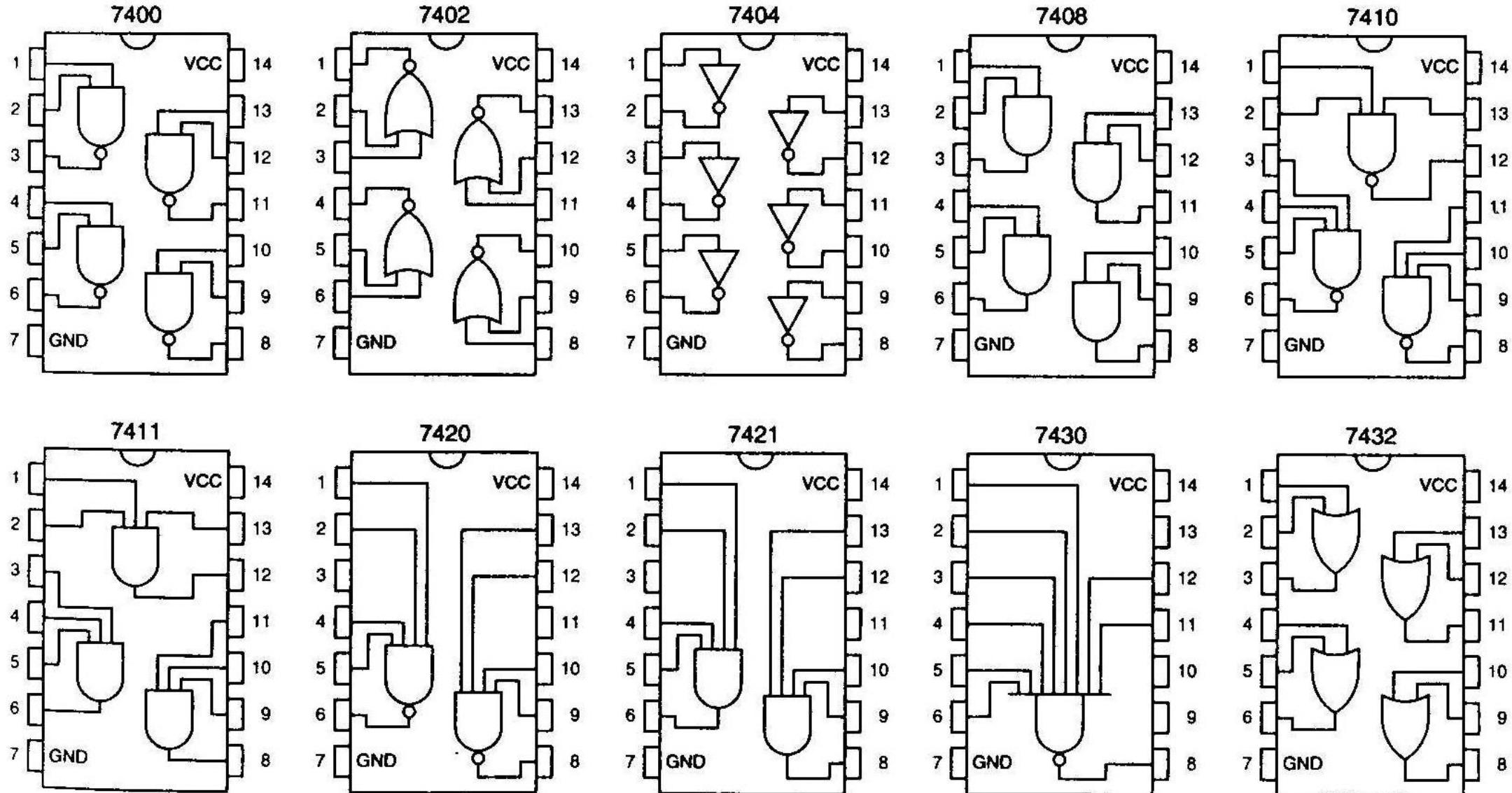
Логический вентиль [https://ru.wikipedia.org/wiki/Логический\\_вентиль](https://ru.wikipedia.org/wiki/Логический_вентиль)

# Условные обозначения типовых логических элементов

| Логический вентиль   | Условные графические обозначения |                     |                 | Функция, запись  | Таблица истинности  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|--|----------------------------------|---------------------|-----------------|--|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|  | ГОСТ 2.743-91                    | IEC 60617-12 : 1997 | US ANSI 91-1984 |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| Исключающее ИЛИ<br>(англ. <i>XOR gate</i> )<br>сложение по модулю 2        |                                  |                     |                 | Строгая дизъюнкция<br>$Y = A \vee B$<br>$Y = A \oplus B$   | <table border="1"> <tr> <th>A</th> <th>B</th> <th>Y</th> </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>0</td> </tr> </table> | A | B | Y | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| A  | B                                | Y                   |                 |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0  | 0                                | 0                   |                 |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0  | 1                                | 1                   |                 |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1  | 0                                | 1                   |                 |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1  | 1                                | 0                   |                 |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| Исключающее ИЛИ с инверсией<br>(англ. <i>XNOR gate</i> )<br>равнозначность |                                  |                     |                 | Эквиваленция<br>$Y = \overline{A} \vee \overline{B}$<br>$Y = \overline{A} \vee \overline{B}$<br>$Y = \overline{A \oplus B}$<br>$Y = A \odot B$ | <table border="1"> <tr> <th>A</th> <th>B</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>1</td> </tr> </table> | A | B | Y | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 |
| A  | B                                | Y                   |                 |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0  | 0                                | 1                   |                 |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 0  | 1                                | 0                   |                 |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1  | 0                                | 0                   |                 |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1  | 1                                | 1                   |                 |  |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |

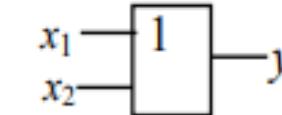
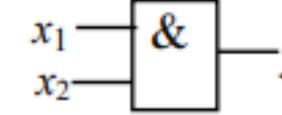
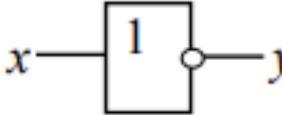
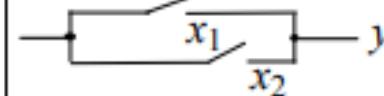
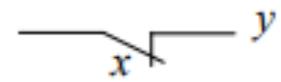
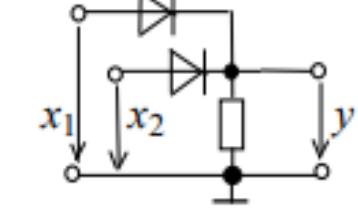
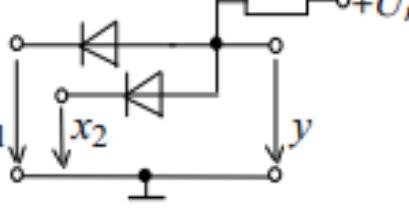
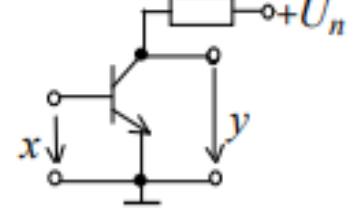
Логический вентиль [https://ru.wikipedia.org/wiki/Логический\\_вентиль](https://ru.wikipedia.org/wiki/Логический_вентиль)

# Микросхемы стандартной логики

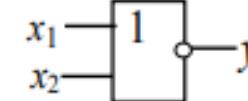
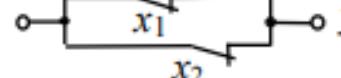
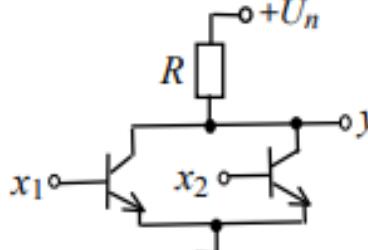
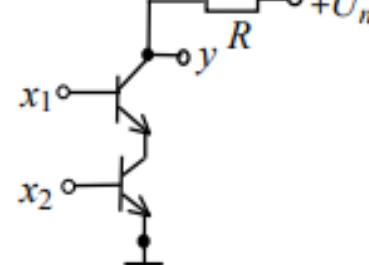


Интегральные схемы: чипы стандартной логики 74xx

# Формы отображения основных логических функций

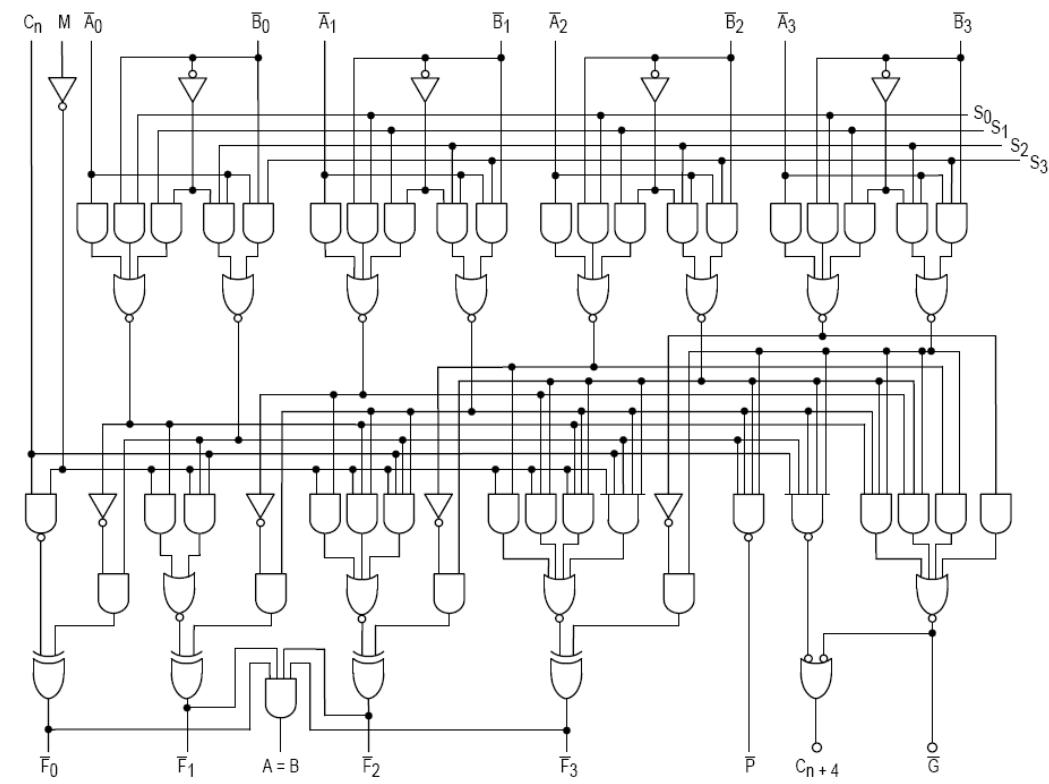
| Наименование              | Дизьюнкция  | Конъюнкция  | Инверсия  |     |   |   |   |   |   |   |   |   |   |   |   |   |   |       |       |     |   |   |   |   |   |   |   |   |   |   |   |   |   |     |     |   |   |   |   |
|---------------------------|---|---|---|-----|---|---|---|---|---|---|---|---|---|---|---|---|---|-------|-------|-----|---|---|---|---|---|---|---|---|---|---|---|---|---|-----|-----|---|---|---|---|
| Символическая             | $\vee$ или $+$  | $\wedge$ или $\cdot$  | $\bar{x}$   |     |   |   |   |   |   |   |   |   |   |   |   |   |   |       |       |     |   |   |   |   |   |   |   |   |   |   |   |   |   |     |     |   |   |   |   |
| Буквенная                 | ИЛИ   | И   | НЕ  |     |   |   |   |   |   |   |   |   |   |   |   |   |   |       |       |     |   |   |   |   |   |   |   |   |   |   |   |   |   |     |     |   |   |   |   |
| Условная<br>графическая   |   |    |    |     |   |   |   |   |   |   |   |   |   |   |   |   |   |       |       |     |   |   |   |   |   |   |   |   |   |   |   |   |   |     |     |   |   |   |   |
| Аналитическая             | $y = x_1 \vee x_2 = x_1 + x_2$  | $y = x_1 \wedge x_2 = x_1 x_2$  | $y = \bar{x}$   |     |   |   |   |   |   |   |   |   |   |   |   |   |   |       |       |     |   |   |   |   |   |   |   |   |   |   |   |   |   |     |     |   |   |   |   |
| Табличная<br>(истинности) | <table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <th><math>x_1</math></th> <th><math>x_2</math></th> <th><math>y</math></th> </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> | $x_1$   | $x_2$   | $y$ | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | <table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <th><math>x_1</math></th> <th><math>x_2</math></th> <th><math>y</math></th> </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> | $x_1$ | $x_2$ | $y$ | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | <table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <th><math>x</math></th> <th><math>y</math></th> </tr> <tr> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> </tr> </table> | $x$ | $y$ | 0 | 1 | 1 | 0 |
| $x_1$                     | $x_2$   | $y$   |   |     |   |   |   |   |   |   |   |   |   |   |   |   |   |       |       |     |   |   |   |   |   |   |   |   |   |   |   |   |   |     |     |   |   |   |   |
| 0                         | 0   | 0   |   |     |   |   |   |   |   |   |   |   |   |   |   |   |   |       |       |     |   |   |   |   |   |   |   |   |   |   |   |   |   |     |     |   |   |   |   |
| 0                         | 1   | 1   |   |     |   |   |   |   |   |   |   |   |   |   |   |   |   |       |       |     |   |   |   |   |   |   |   |   |   |   |   |   |   |     |     |   |   |   |   |
| 1                         | 0   | 1   |   |     |   |   |   |   |   |   |   |   |   |   |   |   |   |       |       |     |   |   |   |   |   |   |   |   |   |   |   |   |   |     |     |   |   |   |   |
| 1                         | 1   | 1   |   |     |   |   |   |   |   |   |   |   |   |   |   |   |   |       |       |     |   |   |   |   |   |   |   |   |   |   |   |   |   |     |     |   |   |   |   |
| $x_1$                     | $x_2$   | $y$   |   |     |   |   |   |   |   |   |   |   |   |   |   |   |   |       |       |     |   |   |   |   |   |   |   |   |   |   |   |   |   |     |     |   |   |   |   |
| 0                         | 0   | 0   |   |     |   |   |   |   |   |   |   |   |   |   |   |   |   |       |       |     |   |   |   |   |   |   |   |   |   |   |   |   |   |     |     |   |   |   |   |
| 0                         | 1   | 0   |   |     |   |   |   |   |   |   |   |   |   |   |   |   |   |       |       |     |   |   |   |   |   |   |   |   |   |   |   |   |   |     |     |   |   |   |   |
| 1                         | 0   | 0   |   |     |   |   |   |   |   |   |   |   |   |   |   |   |   |       |       |     |   |   |   |   |   |   |   |   |   |   |   |   |   |     |     |   |   |   |   |
| 1                         | 1   | 1   |   |     |   |   |   |   |   |   |   |   |   |   |   |   |   |       |       |     |   |   |   |   |   |   |   |   |   |   |   |   |   |     |     |   |   |   |   |
| $x$                       | $y$   |   |   |     |   |   |   |   |   |   |   |   |   |   |   |   |   |       |       |     |   |   |   |   |   |   |   |   |   |   |   |   |   |     |     |   |   |   |   |
| 0                         | 1   |   |   |     |   |   |   |   |   |   |   |   |   |   |   |   |   |       |       |     |   |   |   |   |   |   |   |   |   |   |   |   |   |     |     |   |   |   |   |
| 1                         | 0   |   |   |     |   |   |   |   |   |   |   |   |   |   |   |   |   |       |       |     |   |   |   |   |   |   |   |   |   |   |   |   |   |     |     |   |   |   |   |
| Контактная                |    |   |   |     |   |   |   |   |   |   |   |   |   |   |   |   |   |       |       |     |   |   |   |   |   |   |   |   |   |   |   |   |   |     |     |   |   |   |   |
| Схемо-<br>техническая     |   |  |  |     |   |   |   |   |   |   |   |   |   |   |   |   |   |       |       |     |   |   |   |   |   |   |   |   |   |   |   |   |   |     |     |   |   |   |   |

# Формы отображения основных логических функций

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

# Логическая реализация типовых устройств компьютера

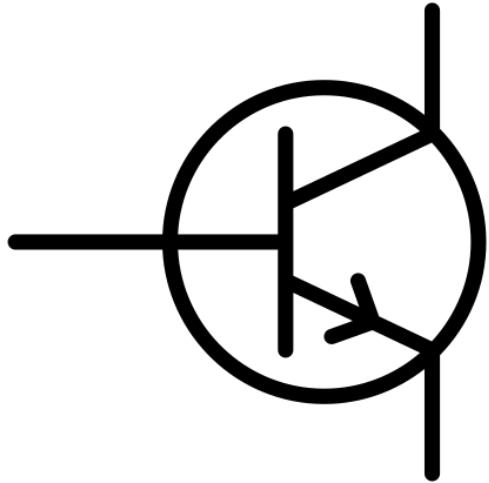
- Обработка любой информации на компьютере сводится к выполнению процессором различных арифметических и логических операций. Для этого в состав процессора входит так называемое арифметико-логическое устройство (АЛУ). Оно состоит из ряда устройств, построенных на рассмотренных выше логических элементах.
- Важнейшими из таких устройств являются:
  - триггеры,**
  - полусумматоры,**
  - сумматоры,**
  - шифраторы,**
  - десифраторы,**
  - счетчики,**
  - регистры.**
  - и многие другие элементы



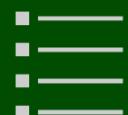
# Реализация может быть любой



Мега Компьютер Состоящий Из Миллионов Живых Людей. Сериал: Задача Трех Тел/ 3 Body Problem // 3 серия  
<https://www.youtube.com/watch?v=OluvYqRWIY>



За пределами  
цифровой абстракции.  
Транзисторы.  
MOS. CMOS.



# За пределами цифровой абстракции

- Цифровая система оперирует дискретными переменными.
- Но для представления этих переменных используются непрерывные физические величины, такие как напряжение в электрической цепи, положение шестеренок в механической передаче или уровень жидкости в гидравлическом цилиндре.
- Задача разработчика цифровой системы – определить, каким образом непрерывно меняющаяся величина соотносится с конкретным значением дискретной переменной.

**Рассмотрим, например, задачу** представления двоичного сигнала А напряжением в электрической цепи. Допустим, что напряжение 0 В соответствует значению  $A = 0$ , а напряжение 5 В соответствует  $A = 1$ .

Но реальная цифровая система должна быть устойчива к неизбежному в такой ситуации шуму, так что значение 4,97 В, вероятно, также следует толковать как  $A = 1$ .

А что делать, если напряжение равно 4,3 В? Или 2,8 В? Или 2,500000 В?

# Напряжение питания

- Предположим, что **минимальное напряжение в электронной цифровой системе**, называемое также напряжением земли (ground voltage, или просто ground, **GND**), **составляет 0 В**.
- **Самое высокое напряжение** в системе поступает от блока питания и, как правило, обозначается  **$V_{DD}$** .
- Транзисторные технологии семидесятых и восьмидесятых годов прошлого века в основном использовали  $V_{DD}$ , равное 5 В.
- С переходом на транзисторы меньшего размера  **$V_{DD}$**  последовательно снижали до **3,3 В, 2,5 В, 1,8 В, 1,5 В, 1,2 В** и даже ниже для экономии электроэнергии и для избегания перегрузки транзисторов.

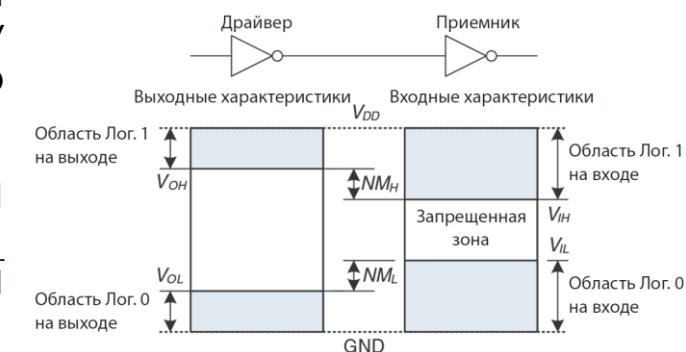
# Логические уровни

- Отображение непрерывно меняющейся переменной на различные значения дискретной двоичной переменной выполняется путем определения логических уровней.
- Первый логический элемент в рассматриваемой схеме называется источник (driver), а второй – приемник (receiver).**
- Выходной сигнал источника подключается ко входу приемника.
- Источник выдает выходной сигнал низкого напряжения (0) в диапазоне от 0 В до  $V_{OL}$  или выходной сигнал высокого напряжения (1) в диапазоне от  $V_{OH}$  до  $V_{DD}$ .
- Если приемник получает на вход сигнал в диапазоне от 0 до  $V_{IL}$ , он рассматривает такой сигнал как ноль.
- Если приемник получает на вход сигнал в диапазоне от  $V_{IH}$  до  $V_{DD}$ , он рассматривает такой сигнал как единицу.
- Если же по какой-либо причине, например наличия шумов или неисправности одного из элементов схемы, напряжение сигнала на входе приемника падает настолько, что попадает в запрещенную зону (forbidden zone) между  $V_{IL}$  и  $V_{IH}$ , то поведение этого логического элемента становится непредсказуемым.
- $V_{OH}$  и  $V_{OL}$  называются соответственно высоким и низким логическими уровнями выхода (output high and low logic levels), а  $V_{IH}$  и  $V_{IL}$  называются соответственно высоким и низким логическими уровнями входа (input high and low logic levels).

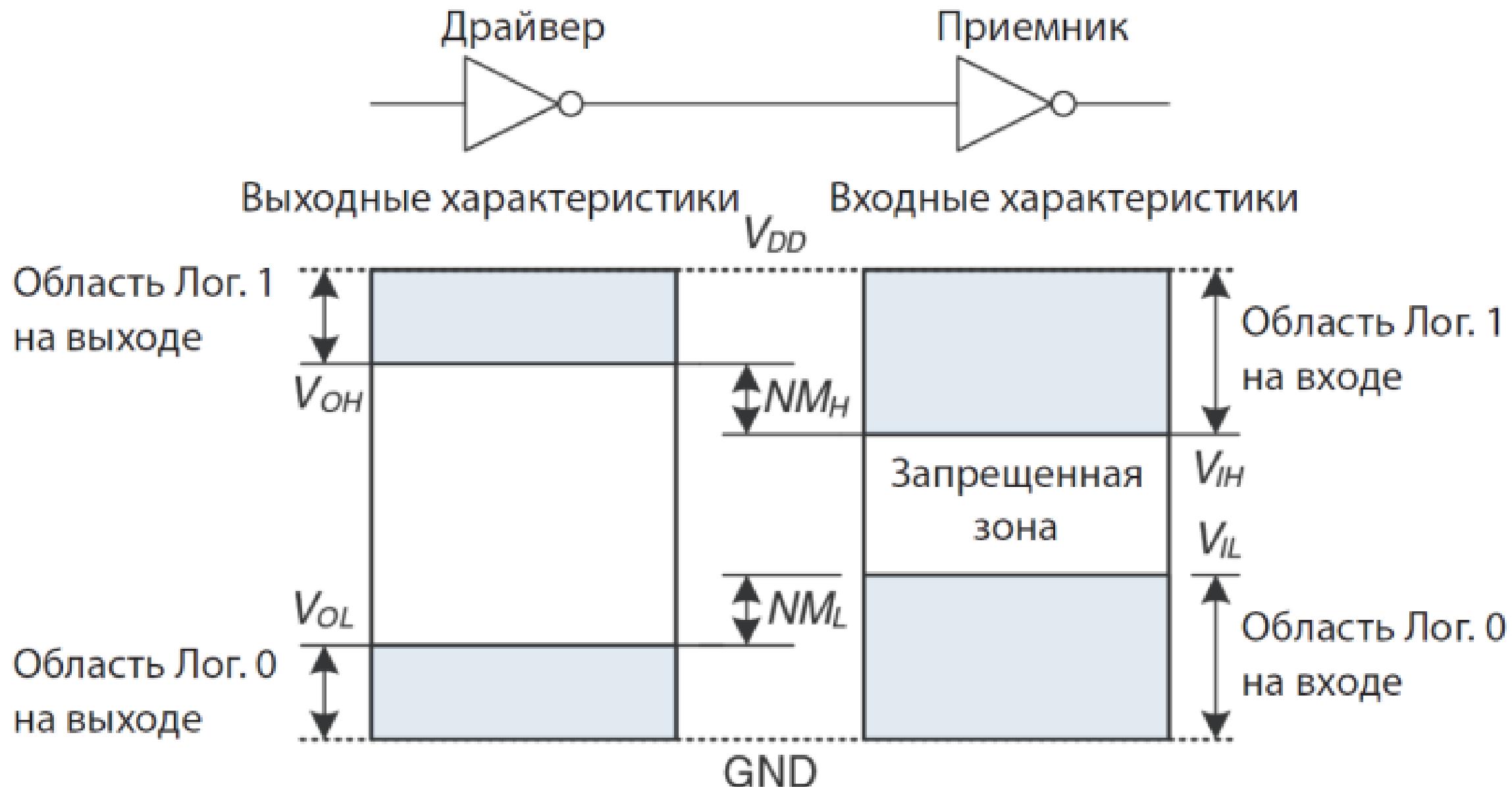
$V_{DD}$  обозначает напряжение стока (drain) в транзисторах, построенных на структуре металл-оксид-полупроводник (МОП). Такие транзисторы используются сегодня для создания самых современных микросхем.

Напряжение источника питания иногда также обозначают  $V_{CC}$ , как напряжение коллектора (collector) в биполярных транзисторах более ранних микросхем.

Напряжение земли (ground voltage, или просто ground) иногда обозначают как  $V_{SS}$  потому, что это напряжение на истоке (source) МОП-транзистора.



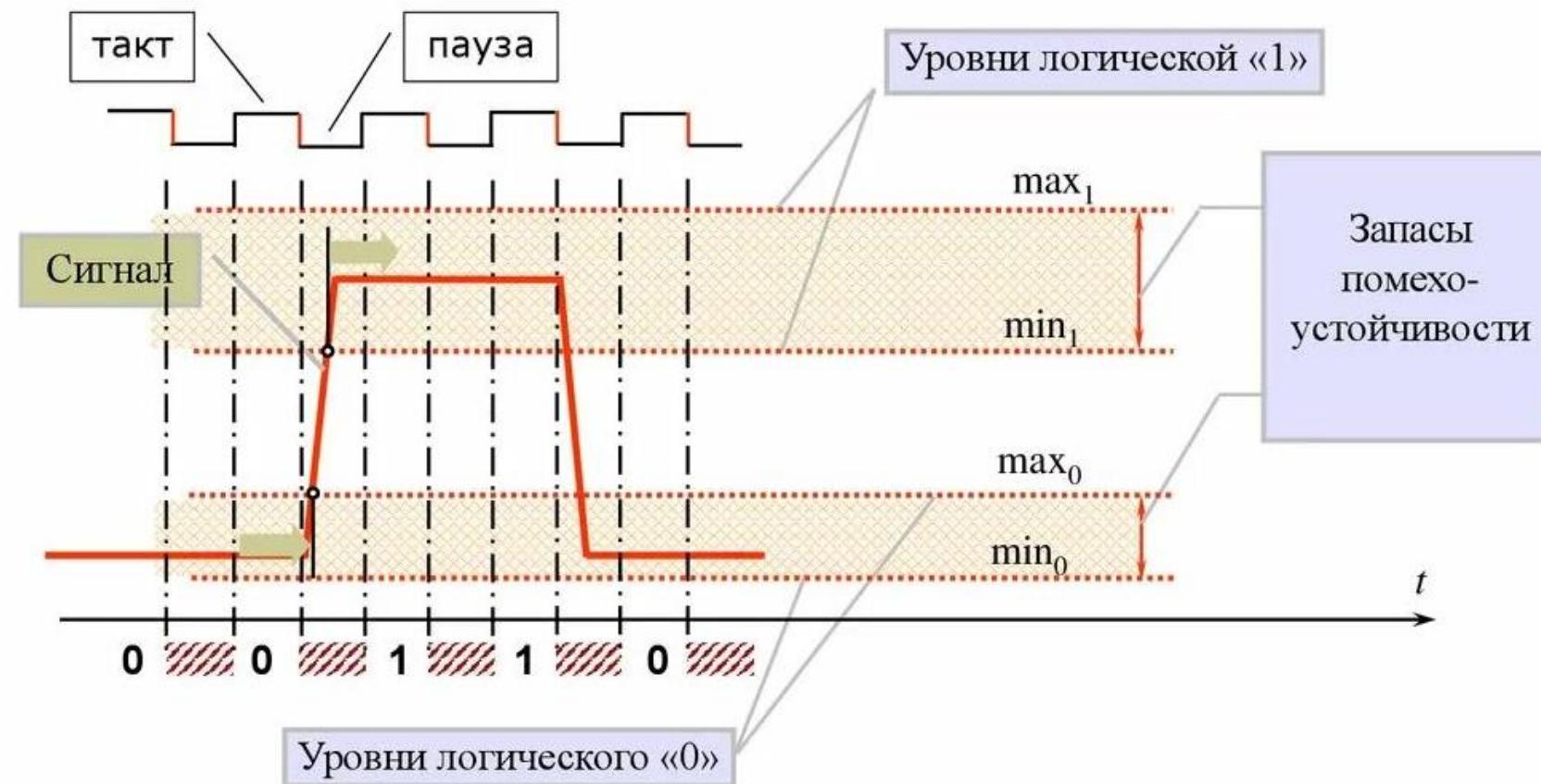
# Логические уровни и уровни шума



# Логические уровни и уровни шума

Идеальный цифровой сигнал имеет четкие дискретные уровни напряжений, напр., 0В, 5В).

Реальные цифровые схемы не могут устанавливать на выходе точные уровни напряжений и реальные цифровые сигналы могут принимать дискретные значения с отклонениями от номинального значения



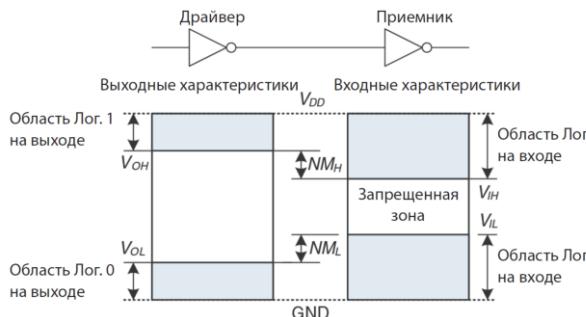
Принято считать логической единицей (1) наличие большого напряжения, а логическим нулем (0) — маленькое напряжение. Диапазоны значений ВЫСОКОГО (1) и НИЗКОГО (0) уровней для входов цифровых схем должны быть шире, чем для выходов, чтобы обеспечить однозначное восприятие уровня на границах диапазонов. Промежуток между диапазонами ВЫСОКОГО и НИЗКОГО сигналов для входа называется мертвым зоной или зоной неопределенного значения

# Допускаемые уровни шумов

- Для того чтобы выходной сигнал источника был правильно интерпретирован на входе приемника, необходимо, чтобы

$$V_{OL} < V_{IL} \text{ и } V_{OH} > V_{IH}$$

- В этом случае, даже если выходной сигнал источника будет загрязнен шумами, приемник по-прежнему сможет правильно определить логический уровень входного сигнала.
- **Допускаемый уровень шумов (noise margin)** – это то максимальное количество шума, присутствие которого в выходном сигнале источника не мешает приемнику корректно интерпретировать значение полученного сигнала.



$V_{OL}$  – Низкий логический уровень выхода (output low logic levels)

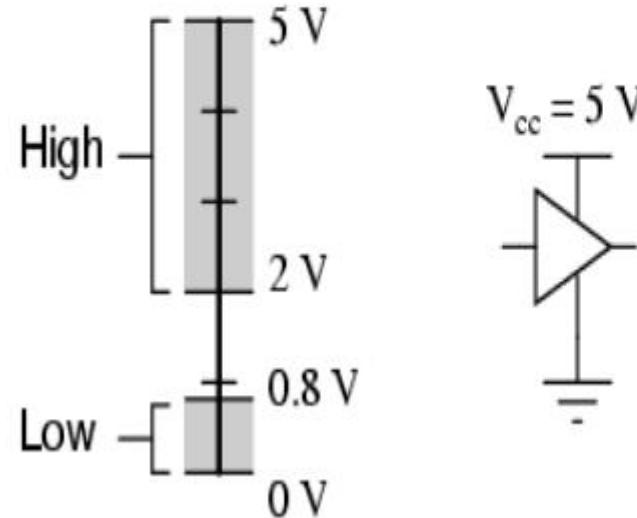
$V_{IL}$  – Низкий логический уровень входа (input low logic levels)

$V_{OH}$  – Высокий логический уровень выхода (output high logic levels)

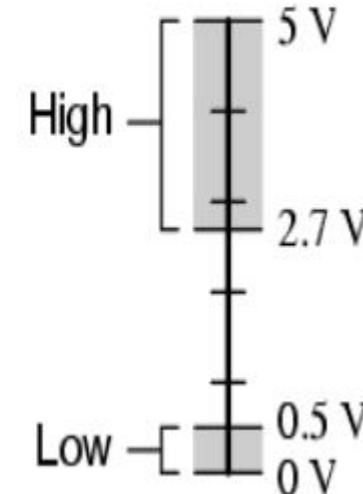
$V_{IH}$  – Высокий логический уровень входа (input high logic levels)

# Логические уровни

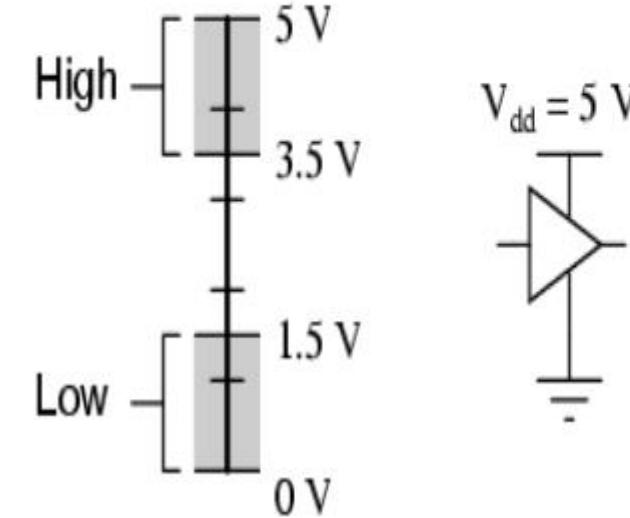
Acceptable TTL gate  
input signal levels



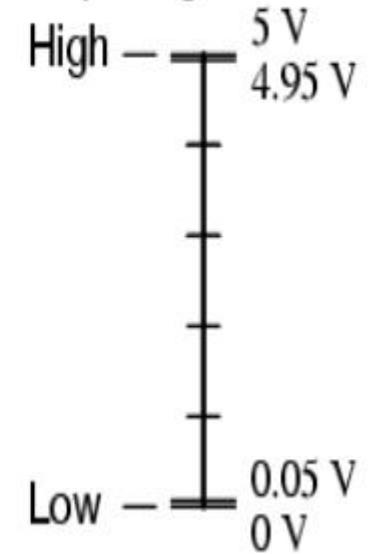
Acceptable TTL gate  
output signal levels



Acceptable CMOS gate  
input signal levels



Acceptable CMOS gate  
output signal levels

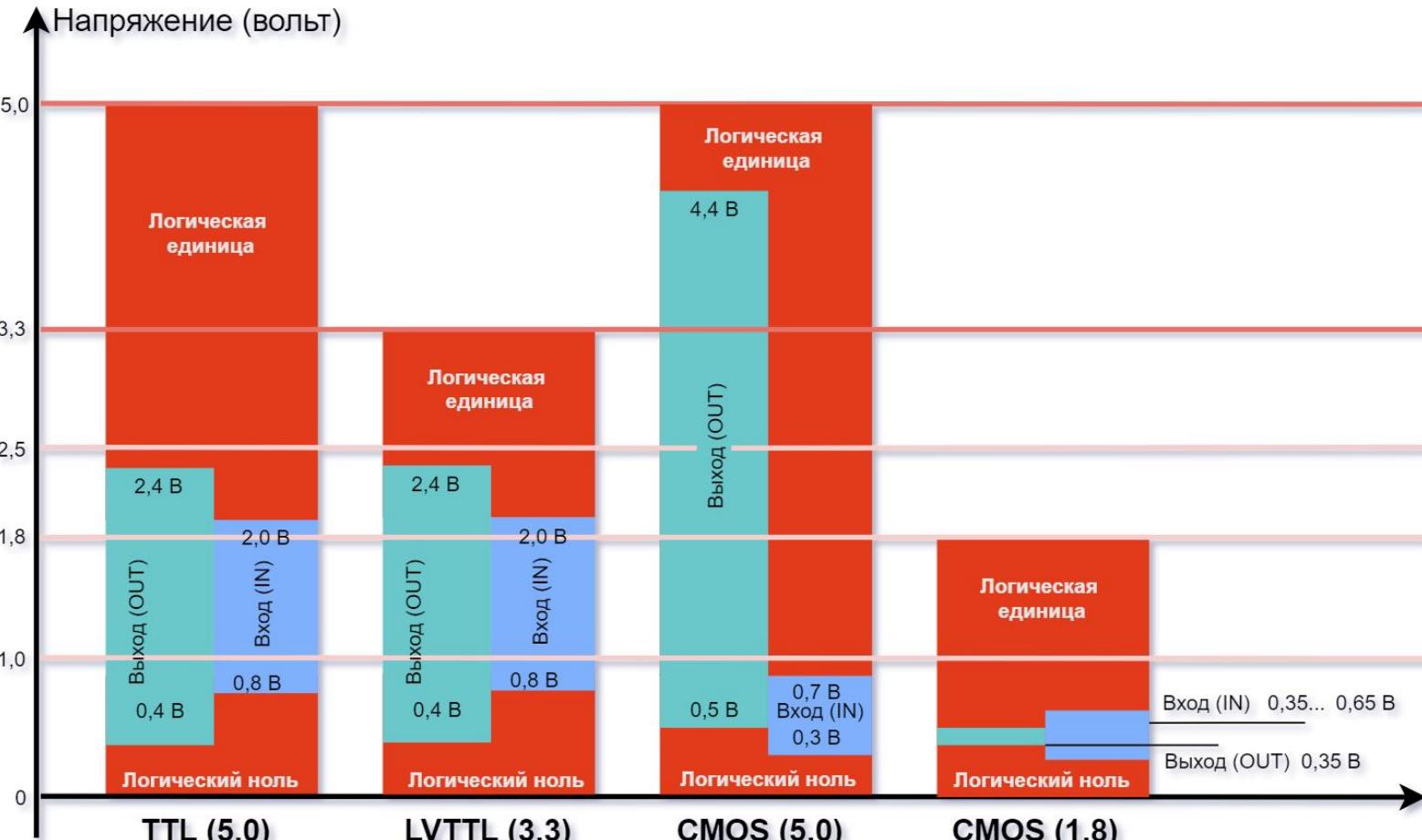


Что дает кодирование уровнями диапазонами напряжений

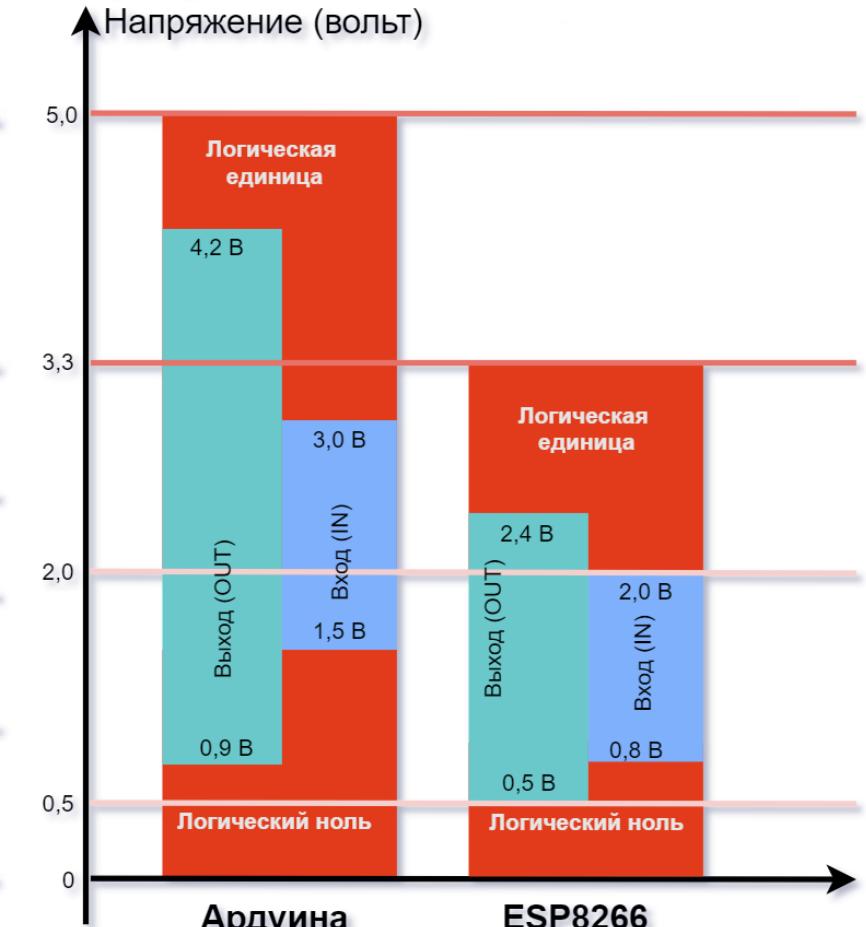
- Позволяет использовать цифровые элементы/схемы с достаточно значительными допусками параметров входных и выходных каскадов, что сильно удешевляет их производство.
- Допускает колебание параметров элементов/схем и соответствующих цифровых сигналов за счет изменения температур, электрической нагрузки и напряжения питания схем и т.п.
- Позволяет игнорировать влияние помех – паразитных напряжений, которые добавляются/вычитаются из рабочего напряжения при «прохождении» его через схему. Шумы возникают за счет емкостных и индуктивных связей между сигналами в схеме, помех приходящих по подключенным внешним цепям и цепям питания, за счет электромагнитных наводок.

# Логические уровни

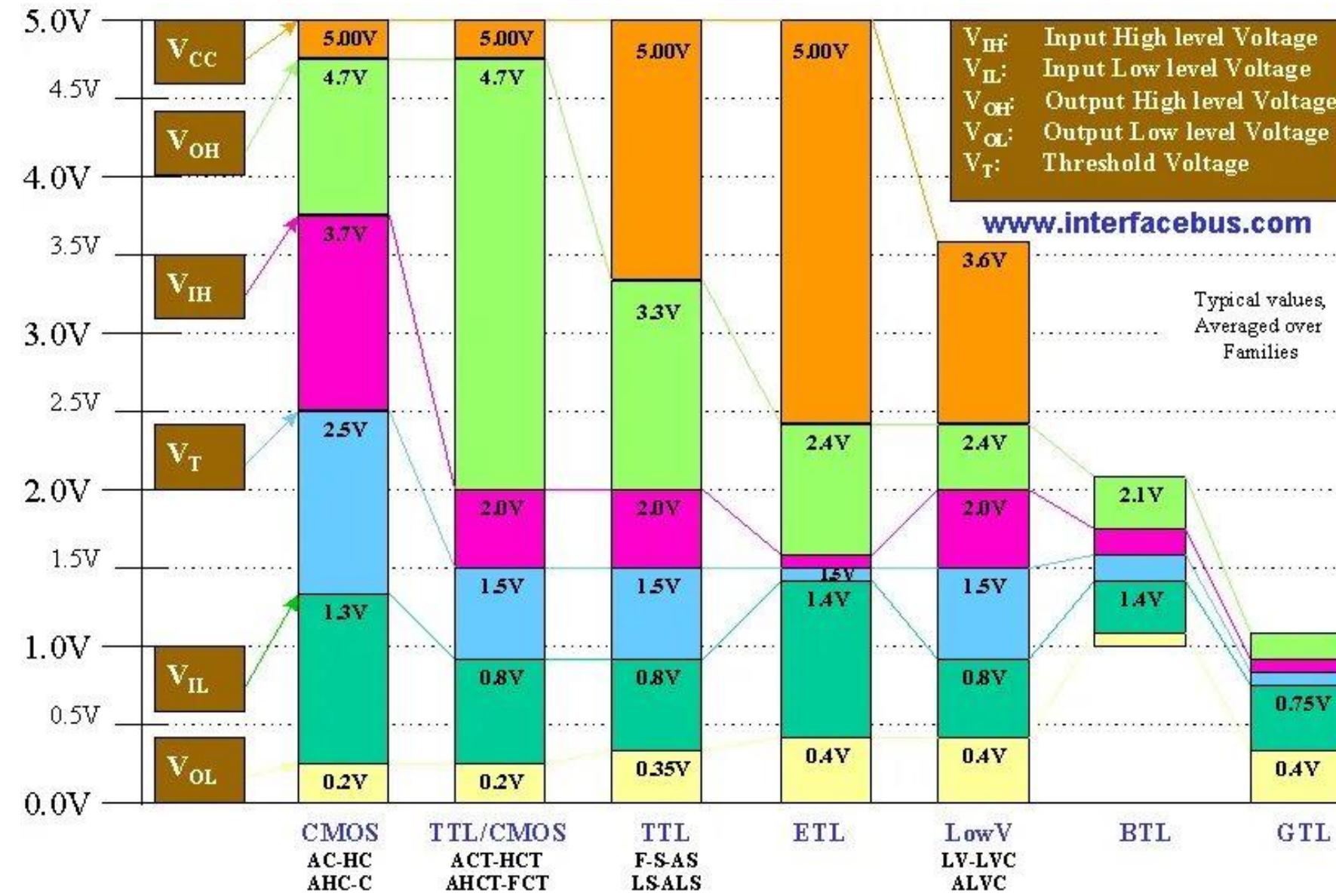
## Логические уровни



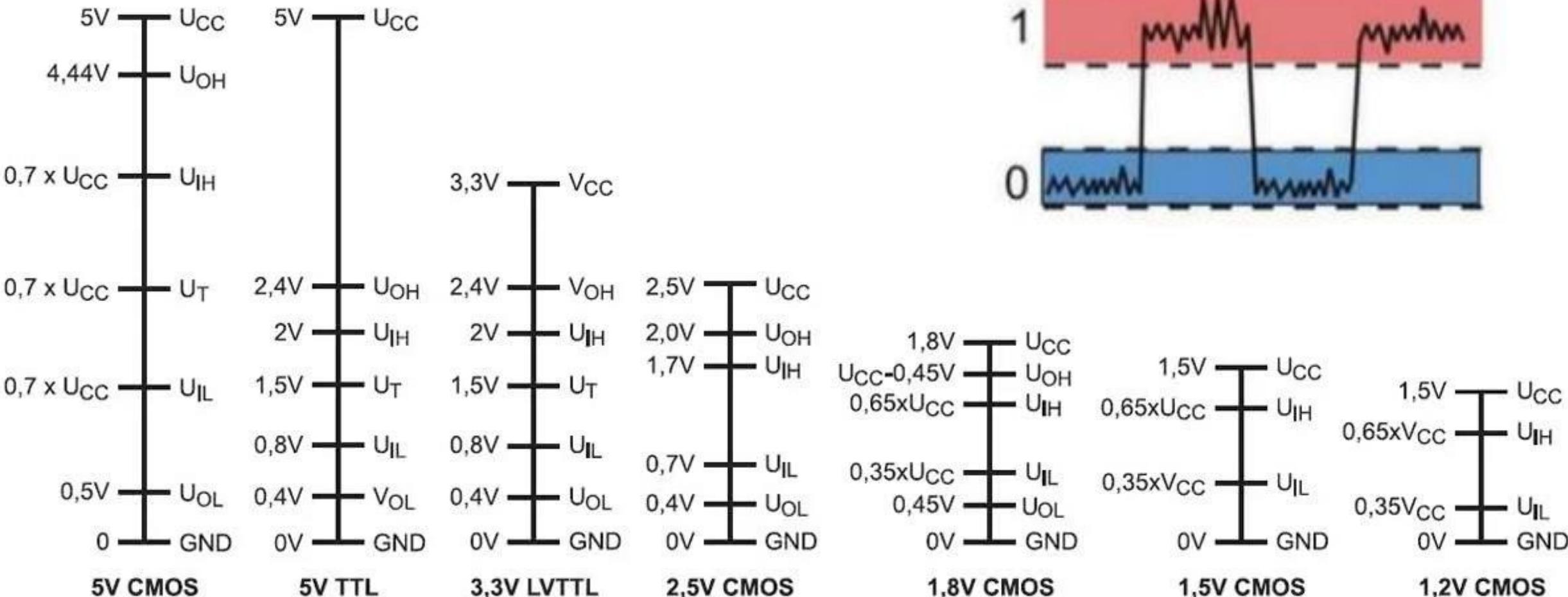
## Логические уровни Ардуины и ESP8266



# Логические уровни



# Логические уровни



# Допускаемые уровни шумов

## Пример РАСЧЕТ УРОВНЕЙ ШУМА

Рассмотрим схему с инверторами на [рис.](#)  $V_{O1}$  – это напряжение на выходе инвертора I1, а  $V_{I2}$  – напряжение на входе инвертора I2. Оба инвертора имеют следующие характеристики:  $V_{DD} = 5$  В,  $V_{IL} = 1,35$  В,  $V_{IH} = 3,15$  В,  $V_{OL} = 0,33$  В и  $V_{OH} = 3,84$  В. Определите нижний и верхний уровни шума. Может ли схема корректно обработать уровень шума в 1 В между  $V_{O1}$  и  $V_{I2}$ ?



**Рис. Схема с инверторами**

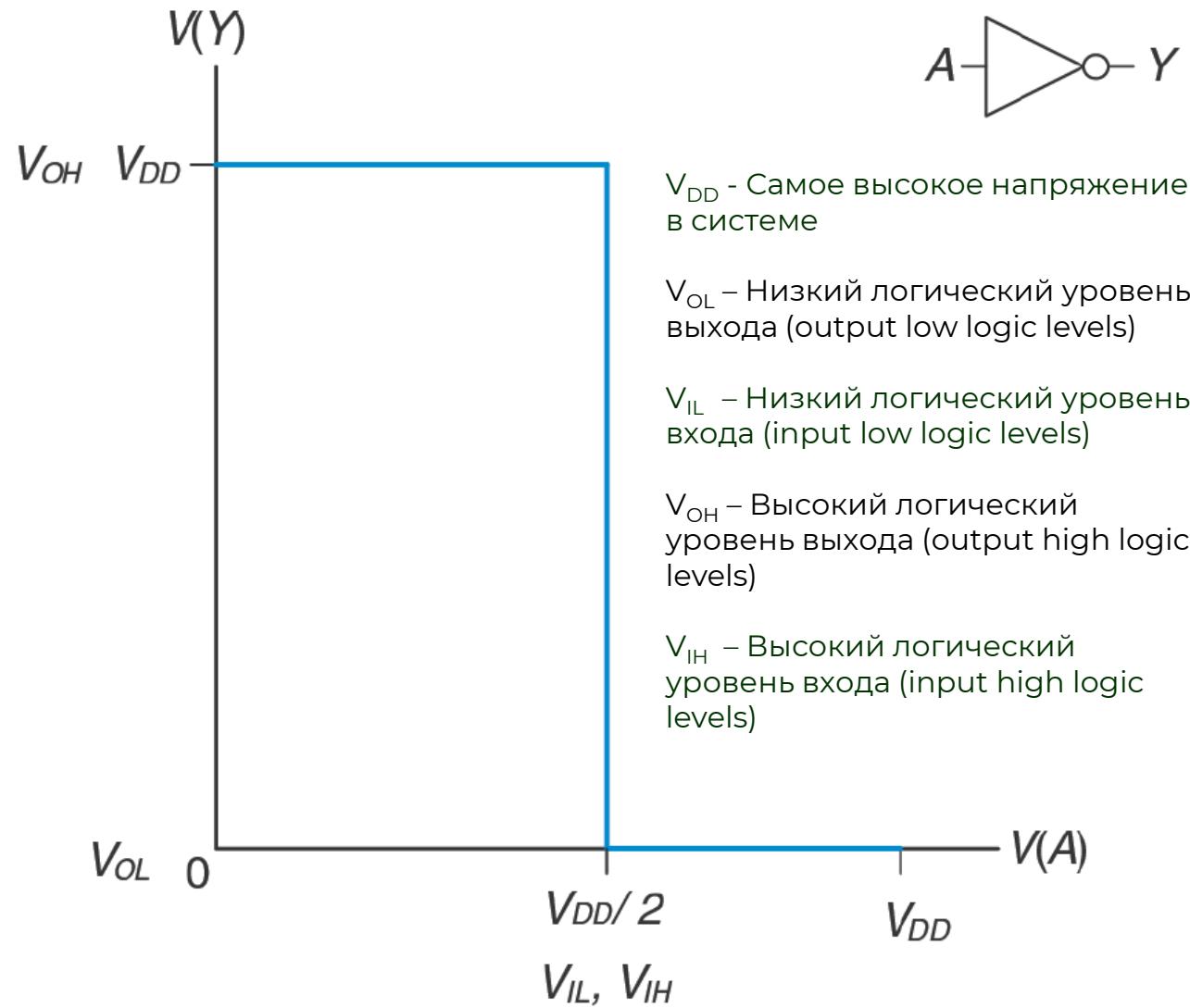
**Решение** Границы уровня шума инвертора следующие:  $NM_L = V_{IL} - V_{OL} = (1,35 \text{ В} - 0,33 \text{ В}) = 1,02 \text{ В}$ ,  $NM_H = V_{OH} - V_{IH} = (3,84 \text{ В} - 3,15 \text{ В}) = 0,69 \text{ В}$ . Схема может корректно обработать шум в 1 В, когда на выходе НИЗКИЙ уровень ( $NM_L = 1,02 \text{ В}$ ), но не когда на выходе ВЫСОКИЙ уровень ( $NM_H = 0,69 \text{ В}$ ). Например, предположим, что инвертор I1 имеет на выходе в наихудшем случае ВЫСОКОЕ значение,  $V_{O1} = V_{OH} = 3,84$  В. Если наличие шума вызовет падение напряжения на 1 В на входе инвертора I2, тогда  $V_{I2} = (3,84 \text{ В} - 1 \text{ В}) = 2,84 \text{ В}$ . Это меньше, чем допустимое входное значение ВЫСОКОГО уровня,  $V_{IH} = 3,15$  В, поэтому инвертор I2 может не принять правильное входное значение ВЫСОКОГО уровня.

# Передаточная характеристика

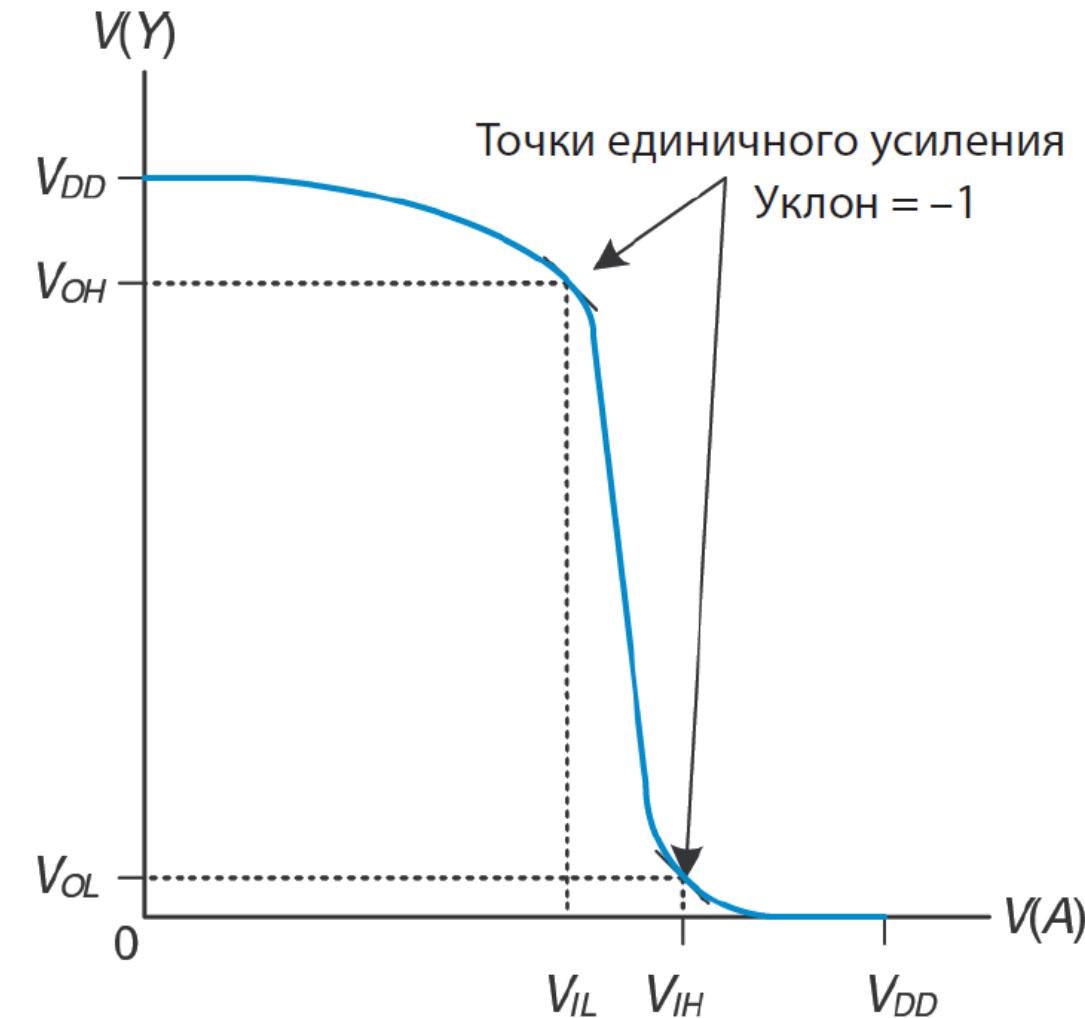
- Для понимания предела цифровой абстракции мы должны рассмотреть поведение логических элементов с аналоговой точки зрения.
- Передаточная характеристика (**DC transfer characteristics**) какого-либо логического элемента описывает напряжение на выходе этого элемента как функцию напряжения на его входе, когда входной сигнал изменяется настолько медленно, что выходной сигнал успевает изменяться вслед за ним.
- Такая характеристика называется передаточной, поскольку описывает взаимосвязь между входным и выходным напряжениями.

DC указывает на состояние, когда напряжение на входе электронной системы поддерживается постоянным или изменяется так медленно, что остальные параметры системы плавно изменяются вместе с ним. Исторически термин DC ведет свое происхождение от понятия постоянный ток (*direct current*) – метод передачи электрической энергии по схеме на расстояние, когда напряжение в линии поддерживается постоянным. В отличие от DC, переходная характеристика (*transient response*) схемы – это состояние, когда входное напряжение меняется быстро.

# Передаточные характеристики и уровни шума



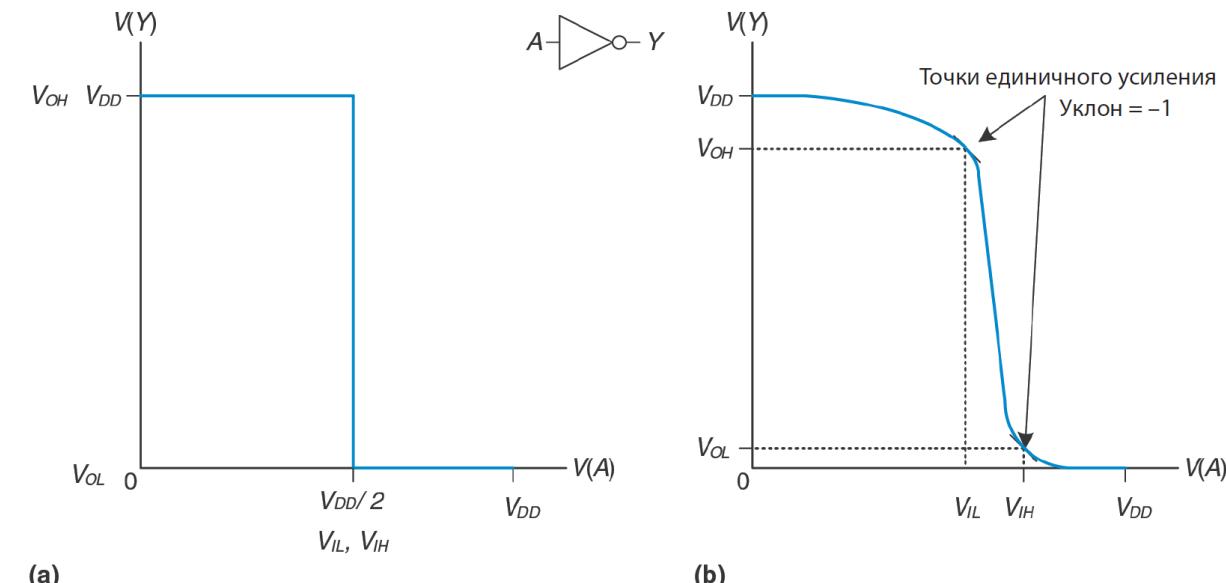
(a)



(b)

# Передаточные характеристики и уровни шума

- В случае идеального инвертора переключение будет резким в точке  $V_{DD}/2$ , как показано на рис. (а).
- Для  $V(A) < V_{DD}/2$   $V(Y) = V_{DD}$ .
- Для  $V(A) > V_{DD}/2$   $V(Y) = 0$ .
- В этом случае  $V_{IH} = V_{IL} = V_{DD}/2$ .
- $V_{OH} = V_{DD}$  и  $V_{OL} = 0$ .
- Напряжение при переключении реального инвертора изменяется постепенно между граничными значениями, как показано на рис. (б).
- Если входное напряжение  $V(A)$  равно 0, то напряжение на выходе  $V(Y) = V_{DD}$ .
- Если  $V(A) = V_{DD}$ , то  $V(Y) = 0$ .
- Но переход между этими конечными точками плавный и может находиться правее или левее значения  $V_{DD}/2$ .



- (a) В качестве логических уровней те две точки, где наклон передаточной характеристики  $dV(Y)/dV(A)$  равен  $-1$ .
- (b) Такие точки называются граничные коэффициенты передачи (unity gain points).
- Подобный выбор обычно максимизирует допускаемые уровни шумов.
- При уменьшении  $V_{IL}$   $V_{OH}$  увеличивается незначительно. Но если  $V_{IL}$  растет,  $V_{OH}$  падает практически отвесно.

# Статическая дисциплина

- Для того чтобы избежать попадания входных сигналов в запретные зоны, логические элементы должны разрабатываться в соответствии с **принципом статической дисциплины (static discipline)**.
- **Принцип статической дисциплины требует, чтобы при условии наличия логически корректных сигналов на входе каждый элемент системы выдавал логически корректные сигналы на выходе.**
- Применение принципа статической дисциплины ограничивает свободу разработчика в выборе аналоговых элементов для построения цифровых систем, но помогает обеспечить простоту и надежность разрабатываемых цифровых схем.
- **Используя этот принцип, разработчик поднимается с аналогового уровня абстракции на цифровой**, что увеличивает производительность разработчика, избавляя его от рассмотрения излишних деталей.

# Статическая дисциплина

- Выбор  $V_{DD}$  и логических уровней может быть произвольным, но этот выбор **должен обеспечить совместимость всех логических элементов**, обменивающихся данными в пределах одной цифровой системы.
- Поэтому **логические элементы обычно группируются в семейства логики (logic families)** таким образом, что любой элемент из одного семейства при соединении с любым другим элементом из этого же семейства автоматически обеспечивает соблюдение принципа статической дисциплины.
- **Логические элементы одного семейства соединяются друг с другом так же легко, как и блоки конструктора Lego, поскольку они полностью совместимы по напряжению источника питания и логическим уровням.**

# Статическая дисциплина

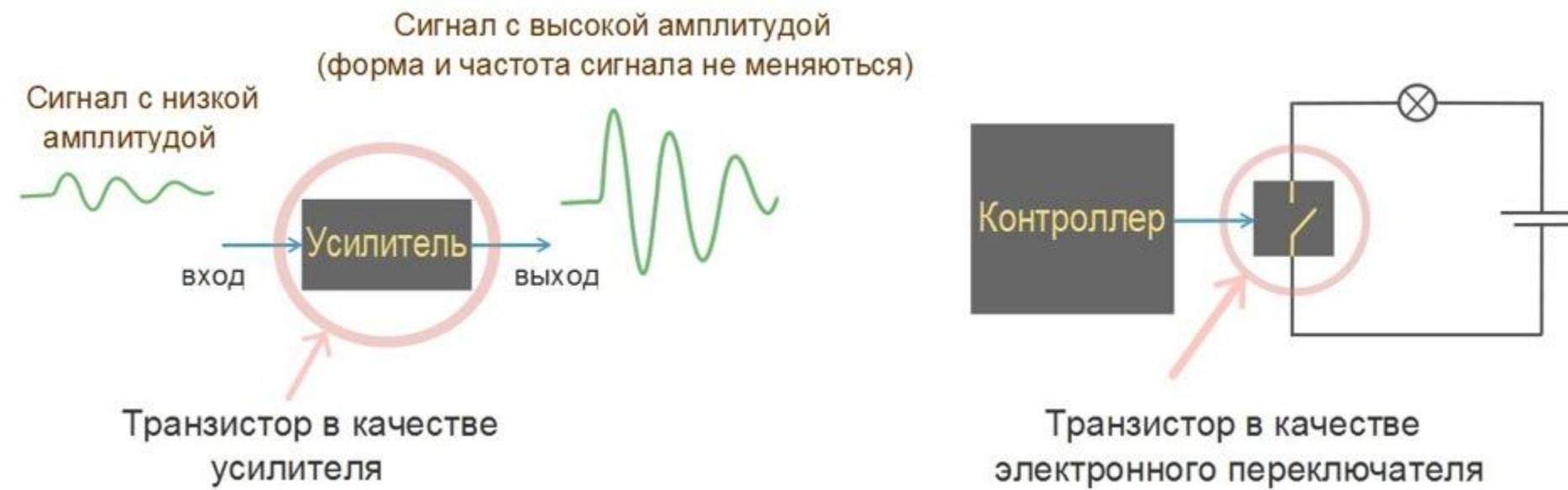
- Четыре основных семейства логических элементов доминировали **с 70-х по 90-е годы** прошлого века – это:
- ТТЛ** – **транзисторно-транзисторная логика** (Transistor-Transistor Logic, или **TTL**),
- КМОП** – **логика**, построенная на комплементарной структуре металл-оксид-полупроводник (Complementary Metal-Oxide-Semiconductor Logic, или **CMOS**),
- НТТЛ** – **низковольтная транзисторно-транзисторная логика** (Low-Voltage Transistor-Transistor Logic, или **LVTTL**)
- и **НКМОП** – **низковольтная логика** на комплементарной структуре металл-оксид-полупроводник (Low-Voltage Complementary Metal-Oxide-Semiconductor Logic, или **LVCMOS**).

# Основной элемент

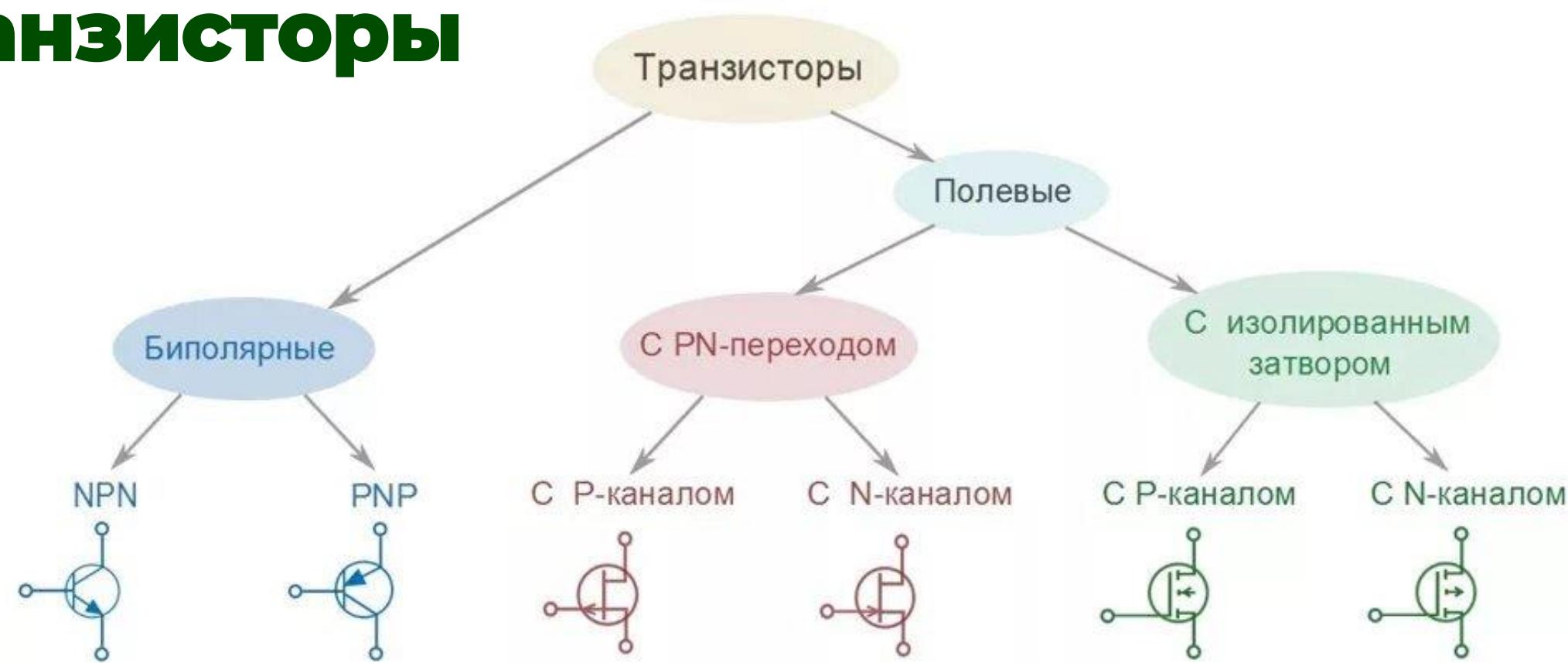
- **ТТЛ (TTL) и НТТЛ (LV-TTL)**
- **Основной элемент: Биполярные транзисторы (BJT — Bipolar Junction Transistor).**
- В ТТЛ и её низковольтной модификации НТТЛ используются биполярные транзисторы, часто с многоэмиттерной структурой для реализации логических функций (например, элементов И-НЕ). НТТЛ оптимизирована для работы при пониженном напряжении питания (например, 3.3 В вместо стандартных 5 В для классической ТТЛ), но сохраняет ту же базовую элементную базу — биполярные транзисторы.
- **КМОП (CMOS) и НКМОП (LVC-MOS)**
- **Основной элемент: МОП-транзисторы (MOSFET — Metal-Oxide-Semiconductor Field-Effect Transistor), а именно комплементарные пары рMOS и нMOS.**
- В КМОП и её низковольтной версии НКМОП логические схемы строятся на комбинации р-канальных (PMOS) и н-канальных (NMOS) транзисторов, образующих энергоэффективные комплементарные структуры. НКМОП адаптирована для работы при меньшем напряжении (например, 3.3 В, 2.5 В или ниже), но сохраняет принцип использования пары рMOS/нMOS.

# Транзистор

- **Транзистор** – электронный полупроводниковый прибор, предназначенный для усиления, генерирования и преобразования электрических сигналов.
- Если быть точнее, то транзистор позволяет регулировать силу электрического тока подобно тому, как водяной кран регулирует поток воды.
- Отсюда следуют **две основные функции прибора в электрической цепи — это усилитель и переключатель.**



# Транзисторы



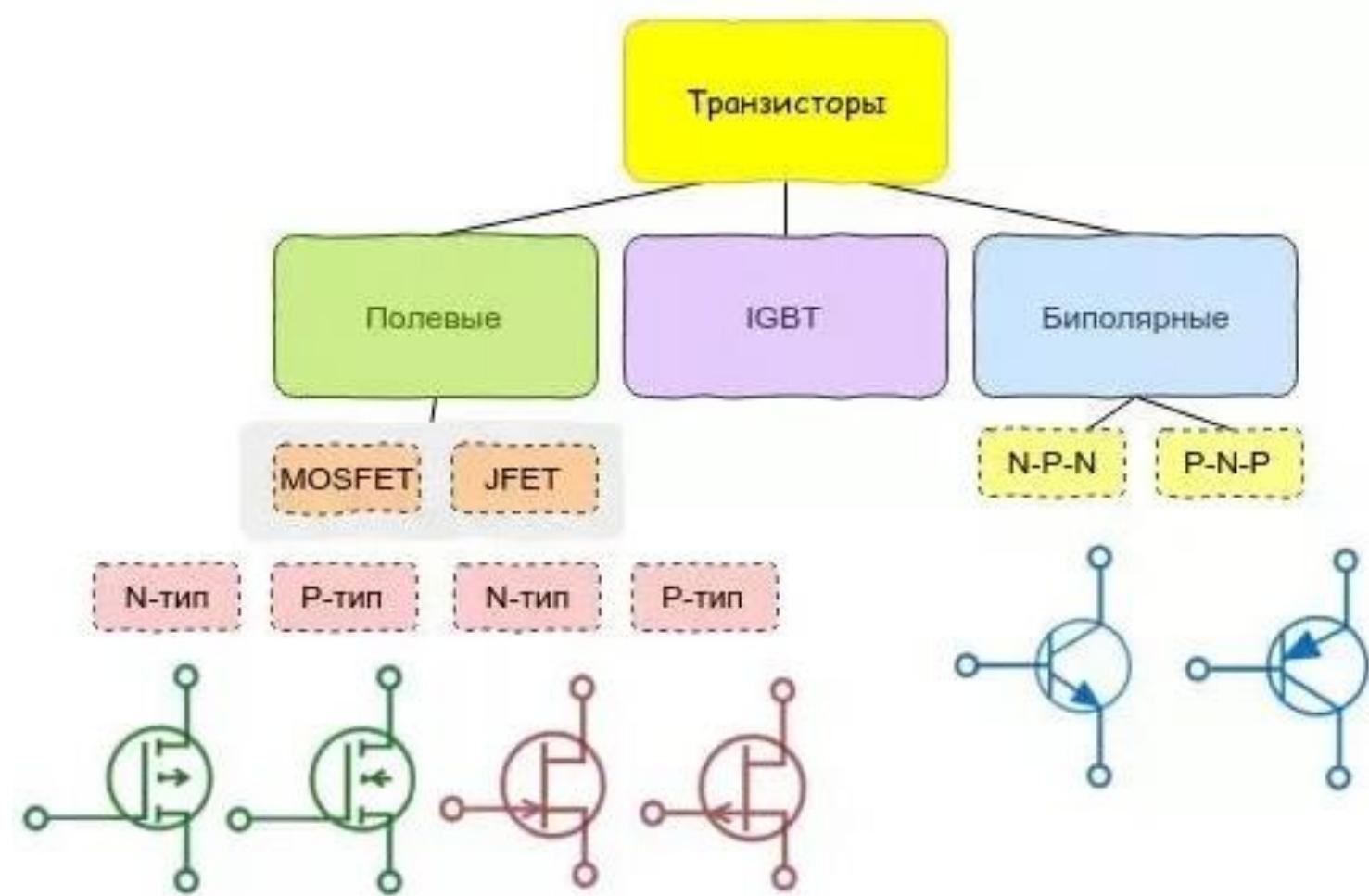
Классификация основных типов транзисторов и обозначение на схеме

**Устройство и обозначение транзисторов укрупненно разделяют на две большие группы.**

**Первая** – это **биполярные транзисторы** (БТ) (международный термин – BJT, Bipolar Junction Transistor). **Вторая** группа – это **унипольярные транзисторы**, еще их называют **полевыми** (ПТ) (международный термин – FET, Field Effect Transistor).

Полевые, в свою очередь, делятся на транзисторы с PN-переходом (**JFET** – Junction FET) и с изолированным затвором (**MOSFET**- Metal-Oxide-Semiconductor FET).

# Транзисторы



**JFET** (Junction Field-Effect Transistor) — это полевой транзистор с управляющим р-п переходом.

Он управляет электрическим полем, как и MOSFET, но затвор в JFET образован р-п переходом, а не изолированным диэлектриком (как в MOSFET).

**Он не является основой ни одного из стандартных цифровых логических семейств, включая RTL, DTL, TTL, ECL, CMOS и др.**

JFET применяются в основном в аналоговых схемах, а не в цифровой логике. Используются в усилителях, датчиках, аудиотехнике и т.д..

**IGBT** (Insulated-Gate Bipolar Transistor)

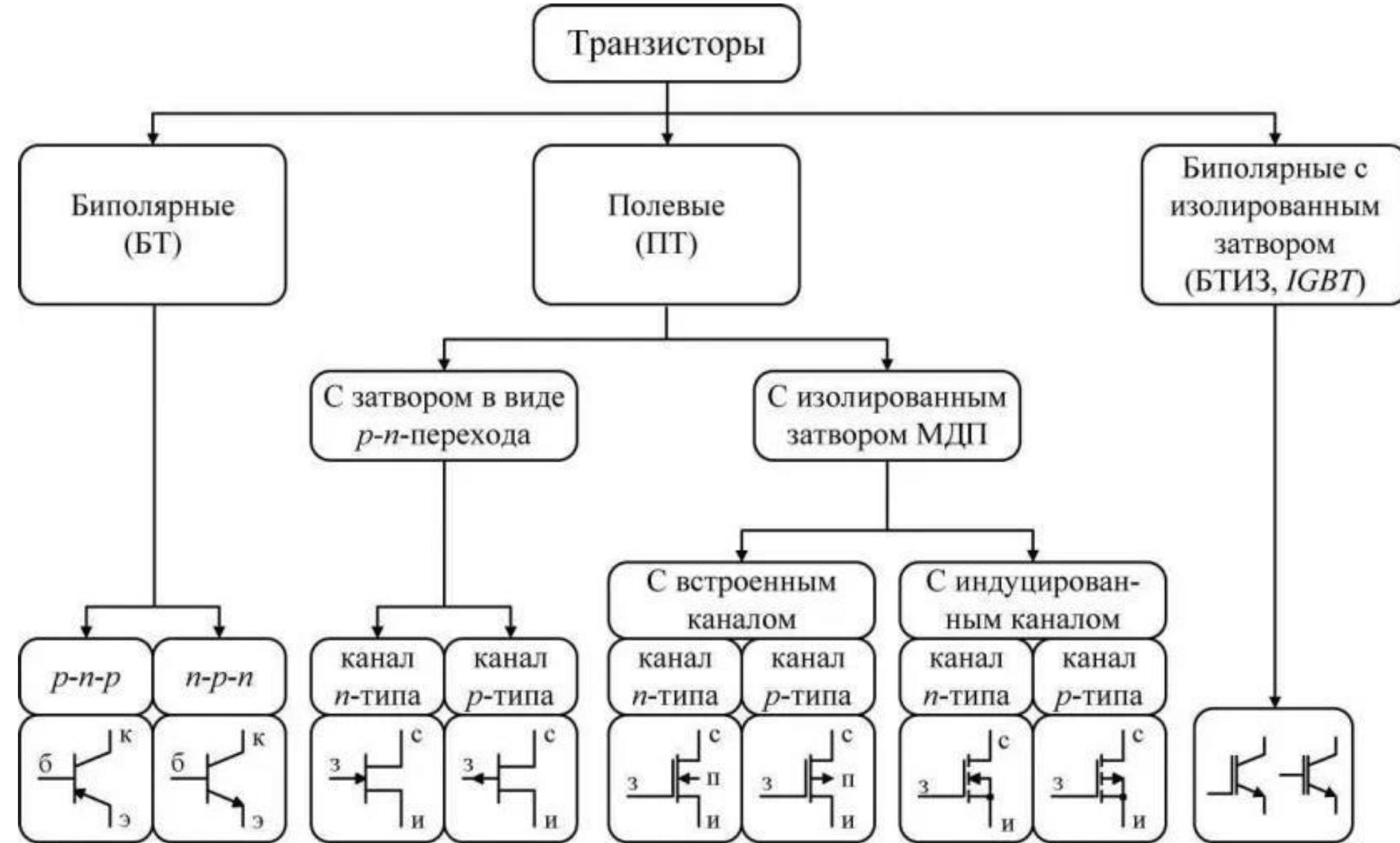
Транзистор с изолированным затвором и биполярным проводимым каналом.

Это гибридный полупроводниковый прибор, сочетающий в себе лучшие черты MOSFET и биполярного транзистора (BJT).

**IGBT — ключевой элемент в силовой электронике.** Используется там, где нужны: Высокое напряжение; Большой ток; Эффективное управление мощностью.

**Он не используется в цифровой логике (не является элементом TTL, CMOS и т.д.), но незаменим в силовой электронике.**

# Транзисторы



# Транзисторы

Приходите к нам в радиоэлектронику, у нас есть:



транзистор



транзистор



транзистор



транзистор



транзистор



транзистор



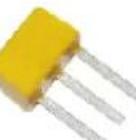
транзистор



транзистор



транзистор



транзистор



транзистор



транзистор



транзистор



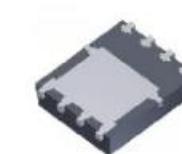
транзистор



транзистор



транзистор



транзистор



транзистор



транзистор



транзистор



транзистор



транзистор

# Семейства логики с уровнями напряжения 5 В и 3,3 В

| Семейство логики   | $V_{DD}$      | $V_{IL}$ | $V_{IH}$ | $V_{OL}$ | $V_{OH}$ |
|--|---------------|----------|----------|----------|----------|
| <b>TTL</b> – ТТЛ – транзисторно-транзисторная логика (Transistor-Transistor Logic, или TTL)  | 5 (4,75-5,25) | 0,8      | 2        | 0,4      | 2,4      |
| <b>CMOS</b> – КМОП – логика, построенная на комплементарной структуре металл-оксид-полупроводник (Complementary Metal-Oxide-Semiconductor Logic, или CMOS)                   | 5 (4,5-6)     | 1,35     | 3,15     | 0,33     | 3,84     |
| <b>LVTTL</b> – НТТЛ – низковольтная транзисторно-транзисторная логика (Low-Voltage Transistor-Transistor Logic, или LVTTL)   | 3,3 (3-3,6)   | 0,8      | 2        | 0,4      | 2,4      |
| <b>LVCMOS</b> – НКМОП – низковольтная логика на комплементарной структуре металл-оксид-полупроводник (Low-Voltage Complementary Metal-Oxide-Semiconductor Logic, или LVCMOS) | 3,3 (3-3,6)   | 0,9      | 1,8      | 0,36     | 2,7      |

$V_{DD}$  - Самое высокое напряжение в системе

$V_{OL}$  – Низкий логический уровень выхода (output low logic levels)

$V_{IL}$  – Низкий логический уровень входа (input low logic levels)

$V_{OH}$  – Высокий логический уровень выхода (output high logic levels)

$V_{IH}$  – Высокий логический уровень входа (input high logic levels)

Начиная с 90-х годов прошлого века четыре вышеперечисленных семейства распались на большое количество более мелких семейств в связи со все большим распространением устройств, требующих еще более низкого напряжения питания.

# Классификация логических семейств

| Семейство   | Тип транзисторов              | Напряжение питания | Скорость        | Энергопотребление     | Применение                   |
|-------------|-------------------------------|--------------------|-----------------|-----------------------|------------------------------|
| DTL         | Диоды + биполярные            | 5–15 В             | Низкая          | Высокое               | Устарело                     |
| TTL         | Биполярные                    | 5 В                | Средняя–высокая | Высокое               | Промышленность               |
| ECL         | Биполярные (дифференциальные) | -5.2 В             | Очень высокая   | Очень высокое         | Суперкомпьютеры              |
| CMOS        | MOSFET (p/n-канальные)        | 3–18 В             | Низкая–средняя  | Минимальное           | Общее назначение             |
| NMOS/PMOS   | MOSFET (только n или p)       | 5–12 В             | Средняя         | Высокое (статическое) | Устарело (ранние процессоры) |
| BiCMOS      | Биполярные + MOSFET           | 5 В                | Высокая         | Среднее               | Микропроцессоры              |
| LVTTL       | Биполярные (модифицированные) | 3.3 В              | Средняя         | Среднее               | Мобильные устройства         |
| LVCMOS      | MOSFET                        | 1.2–3.3 В          | Высокая         | Минимальное           | Современная электроника      |
| GTЛ/HSTL    | MOSFET (дифференциальные)     | 1.2–1.5 В          | Высокая         | Низкое                | Шины памяти                  |
| LVDS/CML    | MOSFET (дифференциальные)     | 1.2–2.5 В          | Очень высокая   | Среднее               | Высокоскоростные интерфейсы  |
| FinFET CMOS | 3D MOSFET                     | <1 В               | Очень высокая   | Минимальное           | Современные процессоры       |

# Семейства логических элементов

Список семейств логических элементов можно разделить на категории, которые перечислены здесь в приблизительном хронологическом порядке, а также с указанием их стандартных сокращений:

- Резисторно-транзисторная логика (RTL)
  - Транзисторная логика с прямой связью (DCTL)
  - Логика на однополюсных транзисторах с прямой связью (DCUTL)
  - Резистивно-ёмкостно-транзисторная логика (RCTL)
- Логика с эмиттерной связью (ECL)
  - Логика с положительным эмиттером (PECL)
  - Низковольтный PECL (LVPECL)
  - Микросхема с комплементарными транзисторами (CTuL)
- Диодно-транзисторная логика (DTL)
  - Логика на комплементарных транзисторных диодах (CTDL)
  - Логика с высоким порогом срабатывания (HTL)
- Транзисторно-транзисторная логика (TTL)
- Логика на основе металл-оксид-полупроводник (МОП)
  - Логика на МОП-транзисторах Р-типа (PMOS)
  - Логика на МОП-транзисторах N-типа (NMOS)
    - Логика NMOS с истощающей нагрузкой
    - NMOS с высокой плотностью (HMOS)
  - Комплементарная МОП (CMOS) логика
  - Биполярная МОП-логика (BiMOS)
    - Биполярная КМОП (BiCMOS)
- Интегрированная логика впрыска (I<sup>2</sup>L)
- Логика приемопередатчика Ганнинга (GTL)

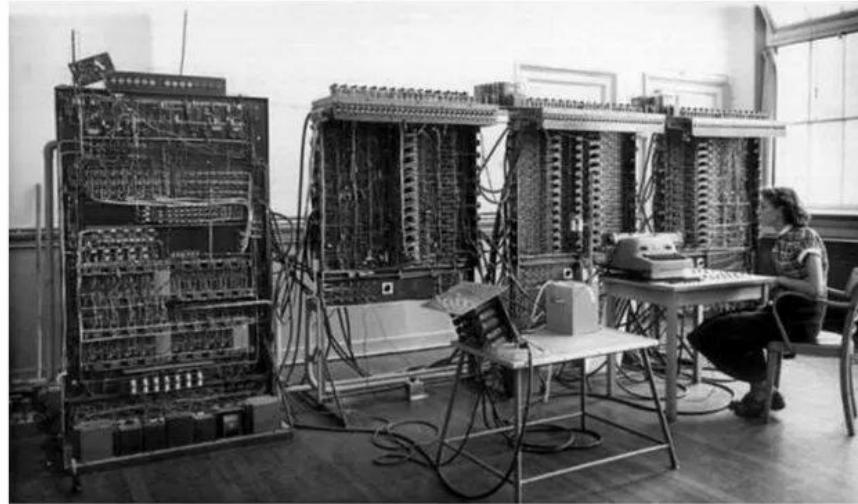
[https://en.wikipedia.org/wiki/Logic\\_family](https://en.wikipedia.org/wiki/Logic_family)

# Этапы развития логических элементов и их семейств

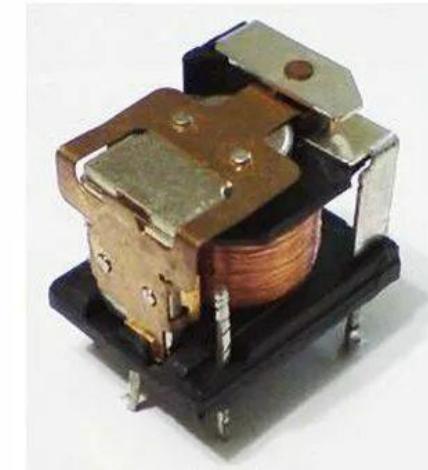
- **Первое поколение (1950-е – 1960-е): Дискретные компоненты**
  - Релейная логика
  - Ламповая логика
  - **RTL** (Resistor-Transistor Logic) - резисторно-транзисторная логика
  - **DTL** (Diode-Transistor Logic) - диодно-транзисторная логика
- **Второе поколение (1960-е – 1970-е): Интегральные схемы малой степени интеграции (SSI)**
  - **TTL** (Transistor-Transistor Logic) - транзисторно-транзисторная логика
  - **ECL** (Emitter-Coupled Logic) - логика с эмиттерной связью
- **Третье поколение (1970-е – настоящее время): Интегральные схемы большой и сверхбольшой степени интеграции (LSI, VLSI)**
  - **MOS** (Metal-Oxide-Semiconductor Logic) - логика металл-оксид-полупроводник
  - **CMOS** (Complementary Metal-Oxide-Semiconductor Logic) - комплементарная логика металл-оксид-полупроводник
  - **LVTTL** (Low-Voltage Transistor-Transistor Logic) - низковольтная транзисторно-транзисторная логика
  - **LVCMOS** (Low-Voltage Complementary Metal-Oxide-Semiconductor Logic) - низковольтная логика на комплементарной структуре металл-оксид-полупроводник
- **Современные и перспективные технологии**
  - **FinFET** (Fin Field-Effect Transistor): 3D-транзисторы
  - **Квантовые логические элементы**
  - **Нейроморфные вычисления**
  - **Молекулярные логические элементы**

# Релейная логика

## История ЭВМ

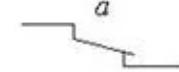
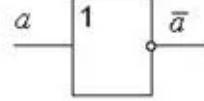
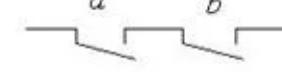
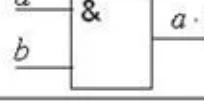
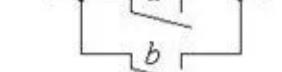
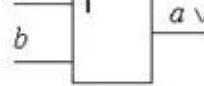
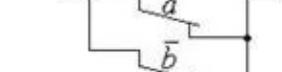
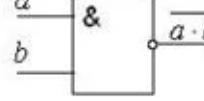
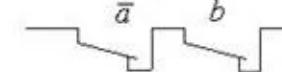
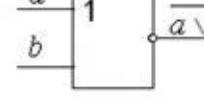


«Марк – 2» - релейная машина,  
изготовленная в 1947 году



Реле

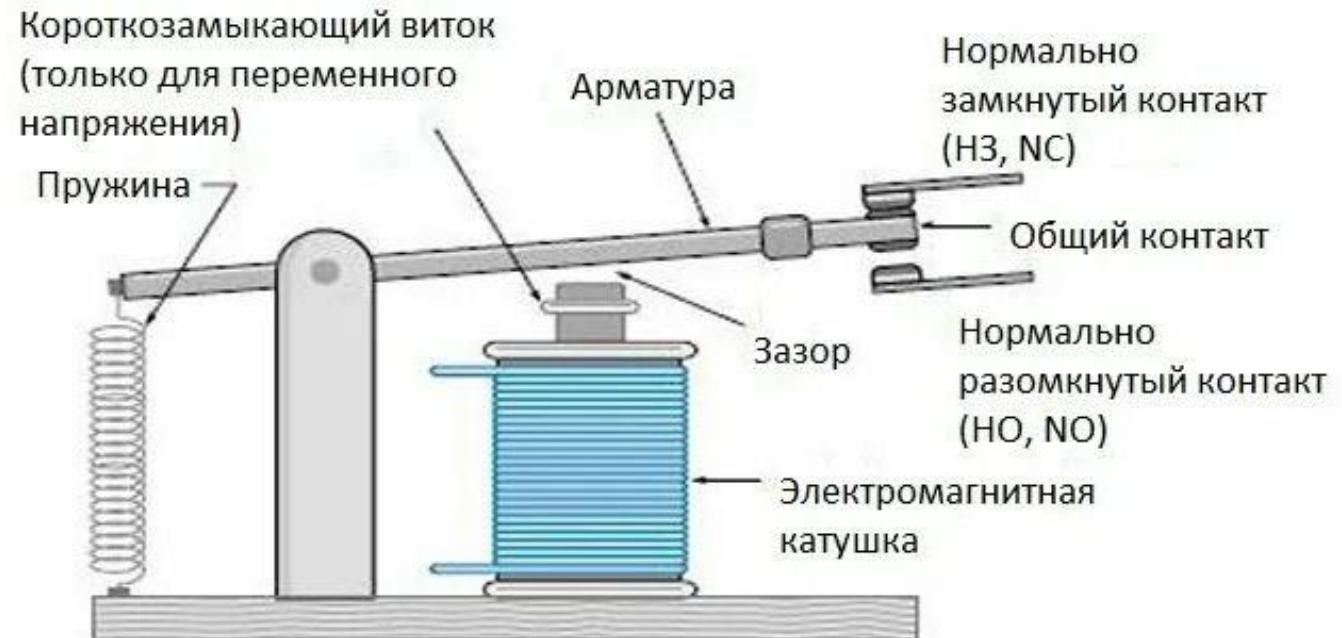
Основана на электромеханических реле

| Операция  | Обозначение при реализации  |   |
|---|---|---|
|   | На контактах реле   | На бесконтактных элементах  |
| Отрицание «НЕ»<br>$y = \bar{a}$   |  |  |
| Конъюнкция «И»<br>$y = a \cdot b$   |  |  |
| Дизъюнкция «ИЛИ»<br>$y = a \vee b$  |  |  |
| Отрицание конъюнкции<br>“И-НЕ”<br>$y = \overline{a \cdot b} = \bar{a} \vee \bar{b}$   |  |  |
| Отрицание дизъюнкции<br>“ИЛИ-НЕ”<br>$y = \overline{a \vee b} = \bar{a} \cdot \bar{b}$ |  |  |



# Релейная логика

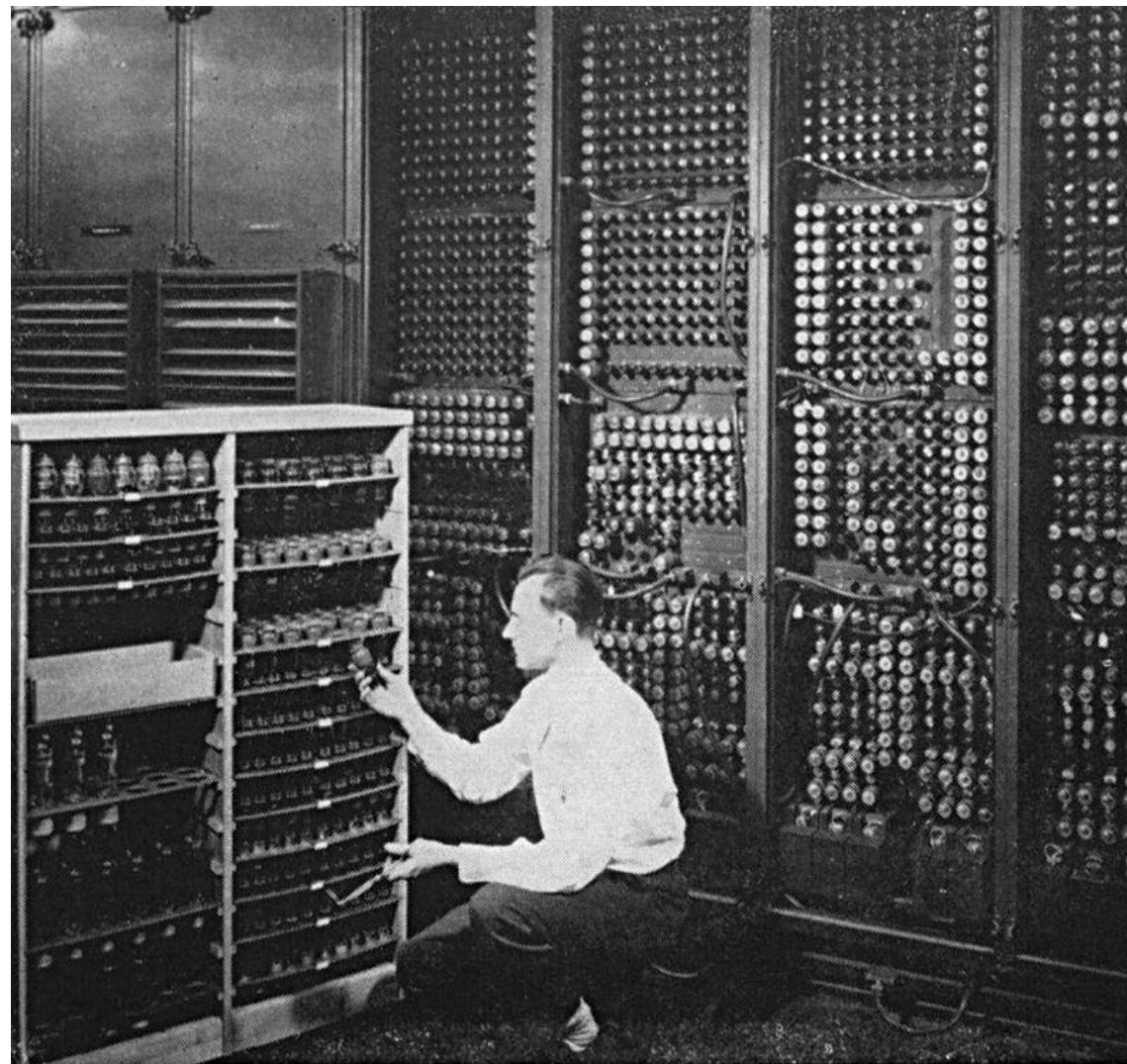
Ключевой элемент Релейной логики - электромеханическое реле



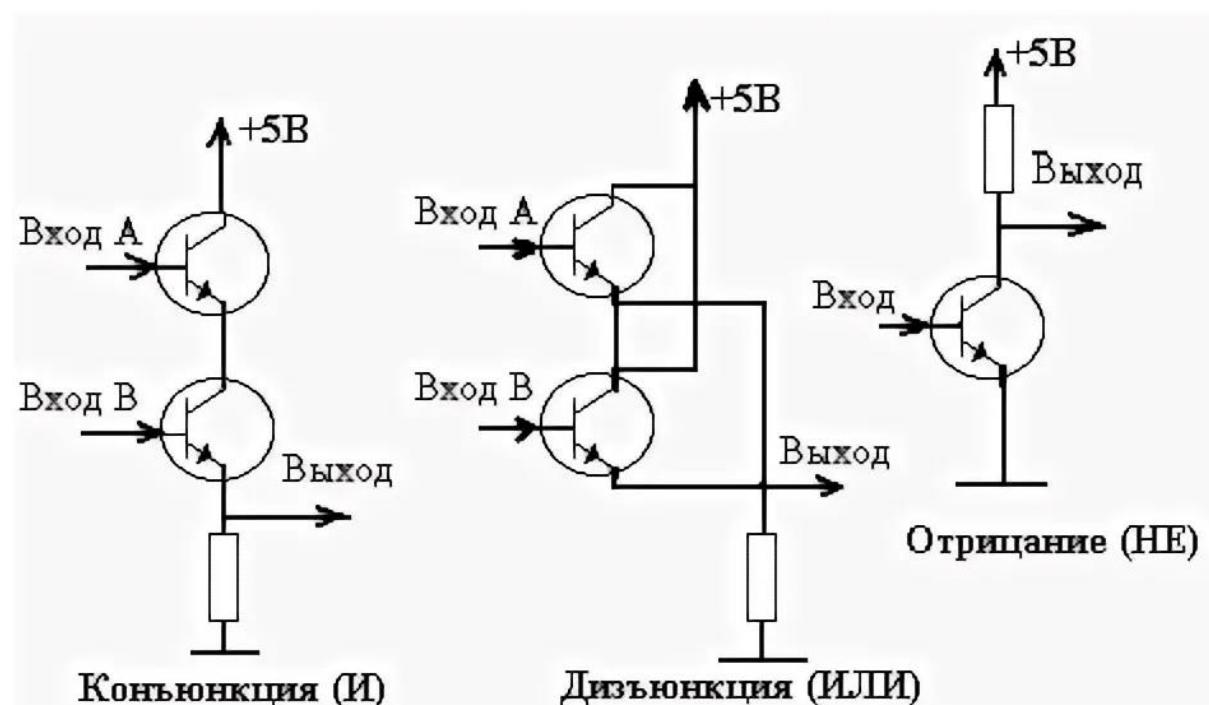
**Реле** - это устройство, которое перенаправляет потоки электроэнергии без необходимости ручного переключения контактов. Для того, чтобы отключить подачу тока или напротив - подать напряжение на нужную часть цепи, достаточно небольшого электросигнала.

Данный тип реле использует механически движущиеся контакты для соединения частей внешней цепи. Для того, чтобы контакты соединились, возле них помещают катушку, которая действует как электромагнит. При подаче слабого тока управления, этот электромагнит создаёт магнитную силу, притягивающие контакты друг к другу, в результате чего происходит соединение.

# Ламповая логика



Replacing a bad tube meant checking among ENIAC's 19,000 possibilities.



# Ламповая логика

Ключевой элемент Ламповой логики - вакуумные лампы (электронные лампы).



# RTL логика

## RTL (Resistor-Transistor Logic) - резисторно-транзисторная логика

**Резисторно-транзисторная логика** (РТЛ), иногда также называемая транзисторно-резисторной логикой (ТРЛ), представляет собой **класс цифровых схем, построенных с использованием резисторов** в качестве входной сети и биполярных транзисторов (BJT) в качестве переключающих устройств.

РТЛ — это самый ранний класс транзисторных цифровых логических схем; на смену ему пришли диодно-транзисторная логика (ДТЛ) и транзисторно-транзисторная логика (ТТЛ).

**Недостатком RTL является высокое энергопотребление** при включении транзистора из-за тока, протекающего через коллекторные и базовые резисторы. Это требует подачи большего тока и отвода тепла от схем RTL.

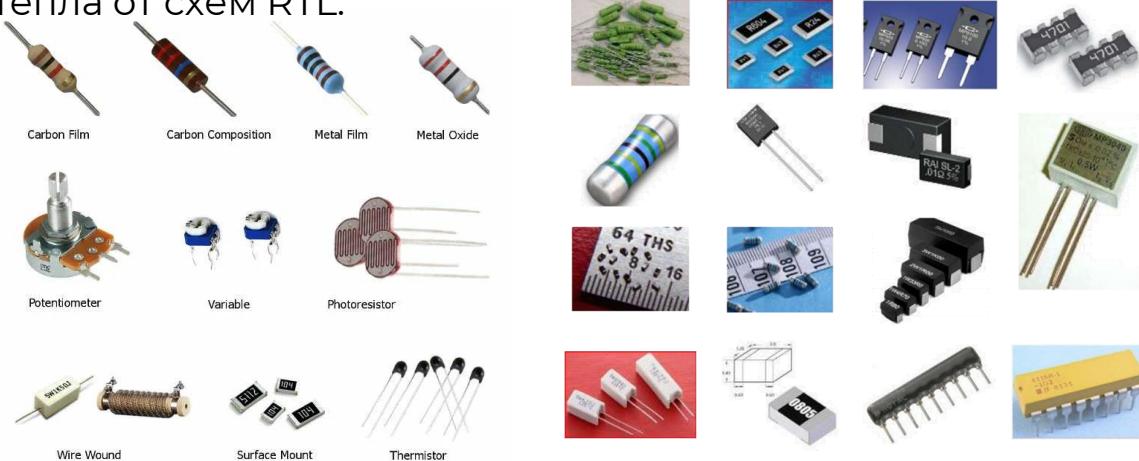
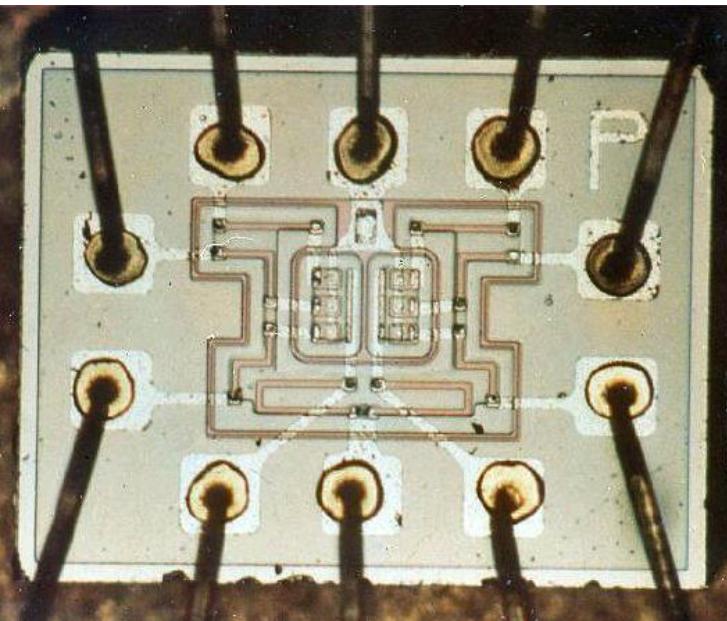
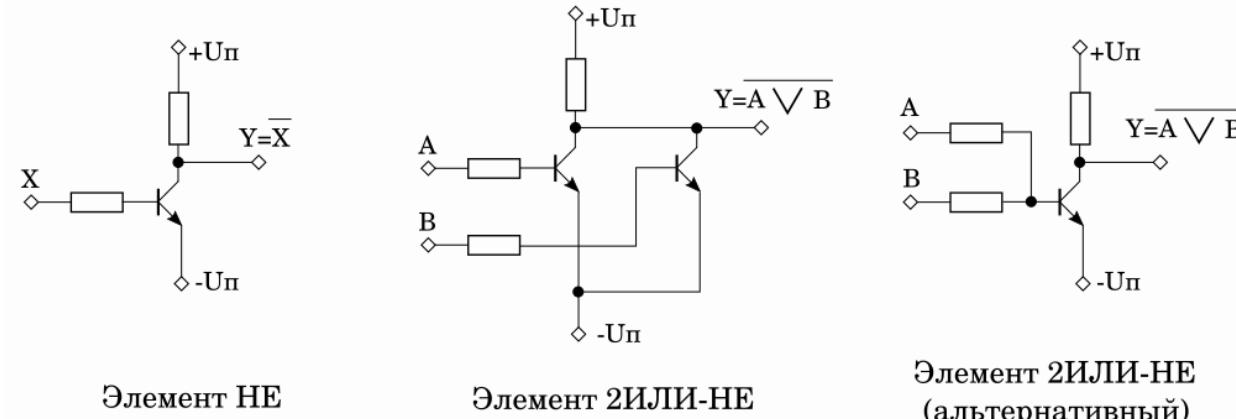


Рисунок 1.4 – Конструктивные разновидности современных резисторов



Фотография микросхемы с двумя 3-ходовыми вентилями NOR, которая использовалась для создания бортового компьютера Apollo. Соединения (по часовой стрелке, начиная с верхнего центра): заземление, входы (3), выход, питание ( $V_{\text{cc}}$ ), выход, входы (3). Шесть транзисторов (две группы по три) расположены в центре. Тонкие провода, идущие от клемм к транзисторам, — это резисторы.

# RTL логика

RTL (Resistor-Transistor Logic) - резисторно-транзисторная логика

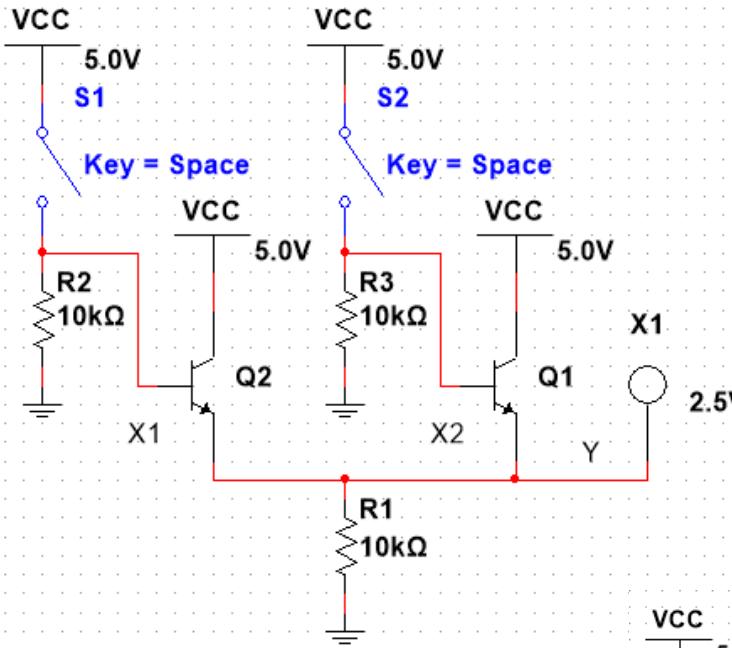


Схема элемента ИЛИ

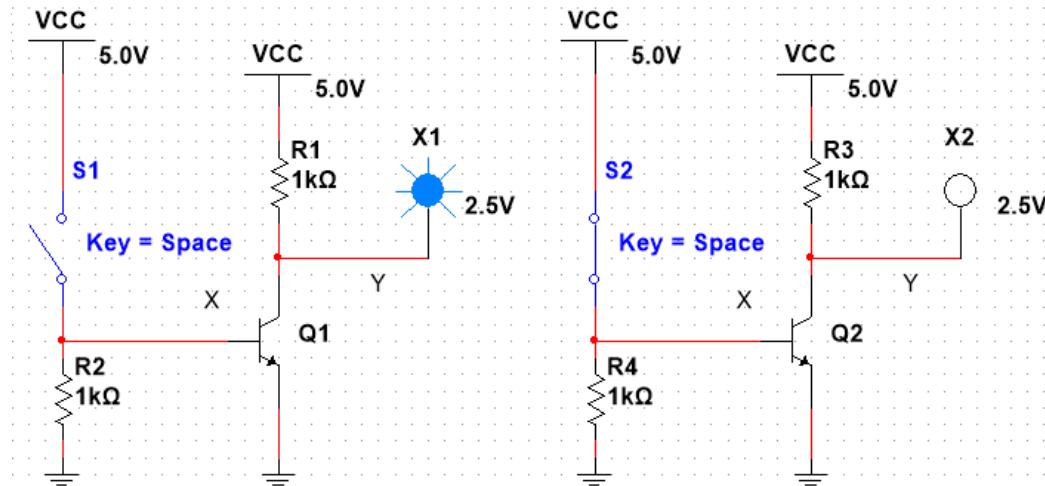
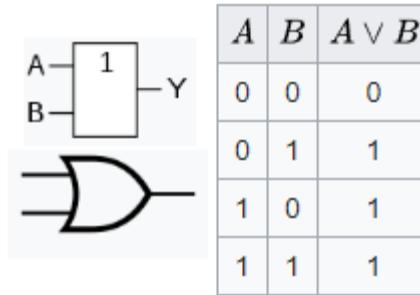


Схема элемента И

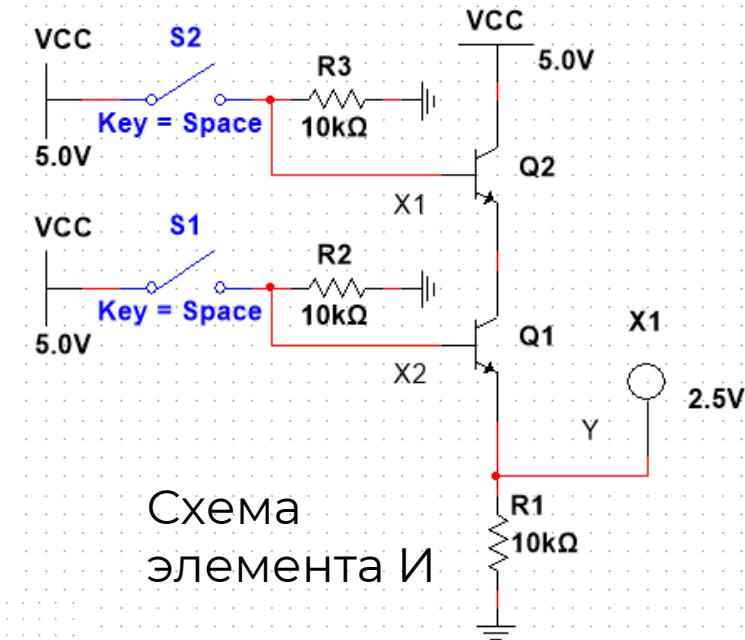
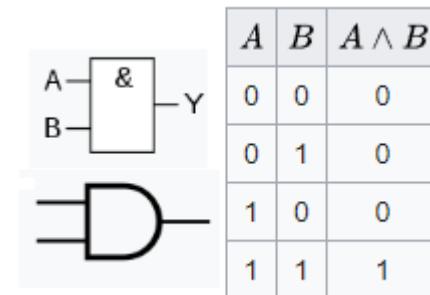


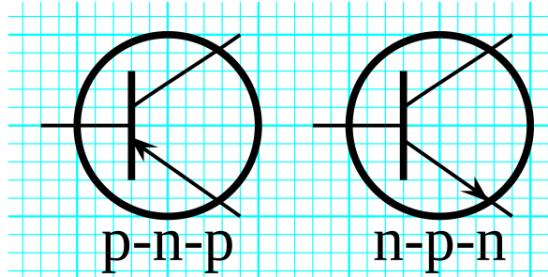
Схема элемента НЕ



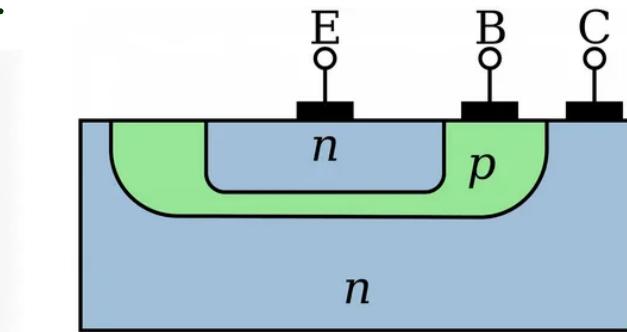
\* Значения напряжения на схемах показано примерно. При физической реализации нужно дорабатывать схему, иначе сгорит.

# RTL логика

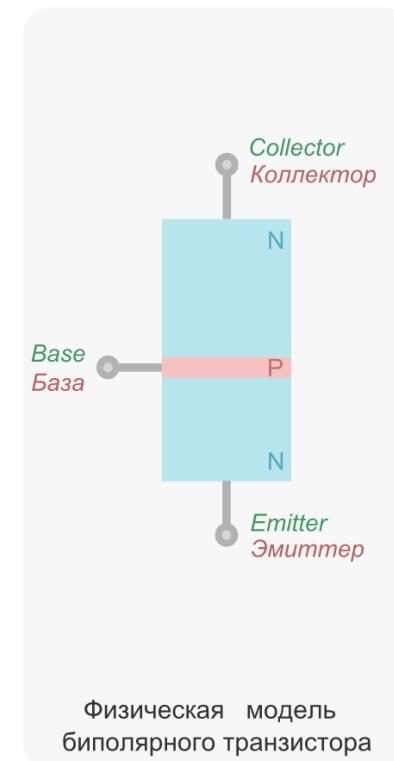
- Ключевой элемент RTL логики - Биполярные транзисторы.
- Биполярный транзистор (**Bipolar transistor**, **Bipolar junction transistor (BJT)**) представляет собой трехвыводной полупроводниковый пробор с тремя чередующимися слоями полупроводника разного вида проводимости, на границе раздела которых образуется два p-n перехода. Применяются в аналоговых устройствах.
- В современной электронике биполярные транзисторы уже практически не используются как силовые ключевые элементы.
- Биполярные транзисторы имеют два основных типа структуры: n-p-n и p-n-p.



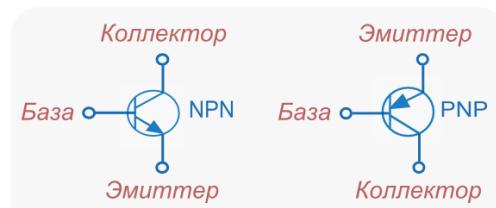
Обозначение биполярных транзисторов на схемах по ГОСТ 2.730



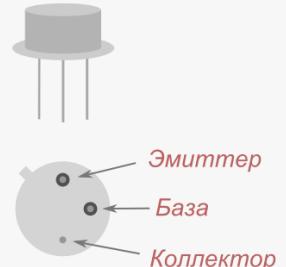
Упрощённая схема поперечного разреза планарного биполярного n-p-n транзистора.



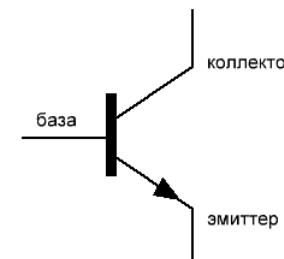
Физическая модель биполярного транзистора



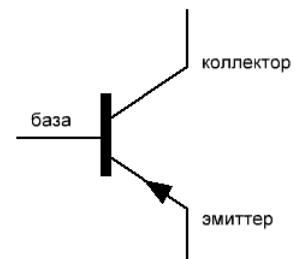
Обозначение на схеме биполярного транзистора



Биполярный транзистор в корпусе ТО типа



n-p-n транзистор



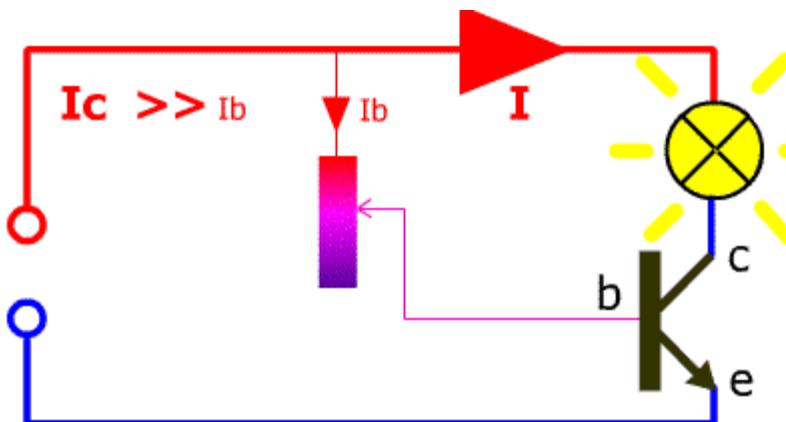
p-n-p транзистор

# Биполярный транзистор (BJT)

Работа его основана на движении носителей заряда через p-n переходы.

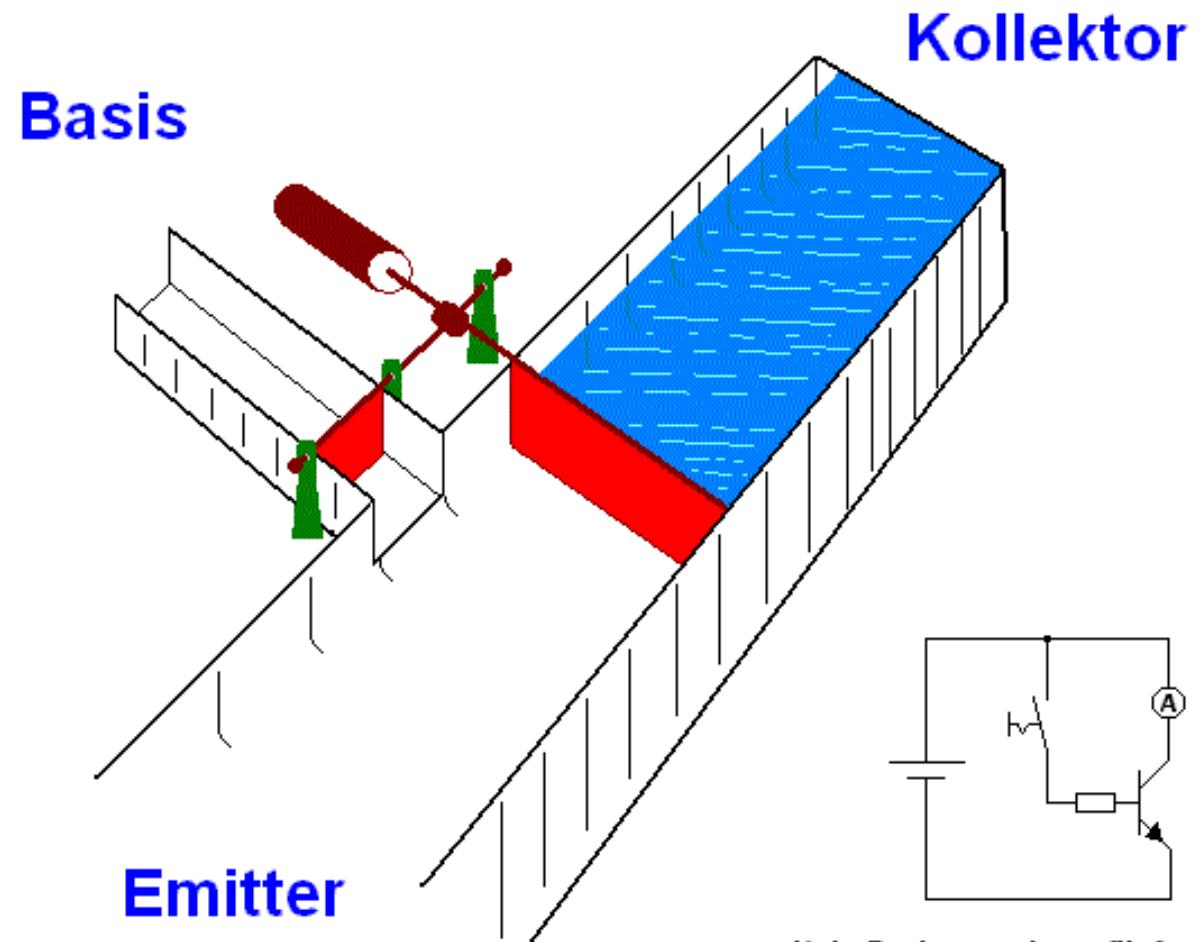
В активном режиме:

- 1.Малый ток базы  $I_b$  управляет большим током коллектора  $I_c$ .
- 2.Электроны (в NPN) или дырки (в PNP) проходят через тонкую базу.
- 3.При наличии достаточного напряжения между коллектором и эмиттером ток усиливается.



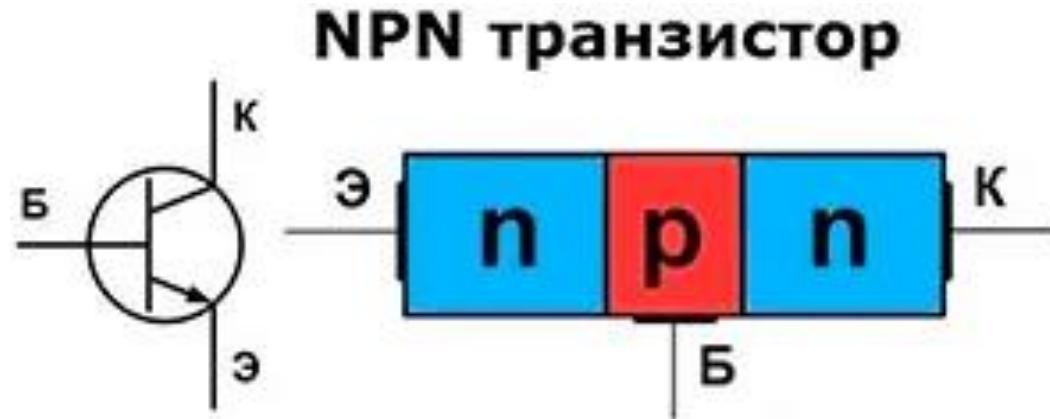
Передача тока через p-n переходы

При подаче напряжения на базу p-n переход эмиттер-база открывается, обеспечивая движение носителей заряда. В активном режиме большая часть носителей проходит через базу и попадает в коллектор, усиливая выходной ток.



Kein Basisstrom kann fließen;  
Der Transistor ist gesperrt.

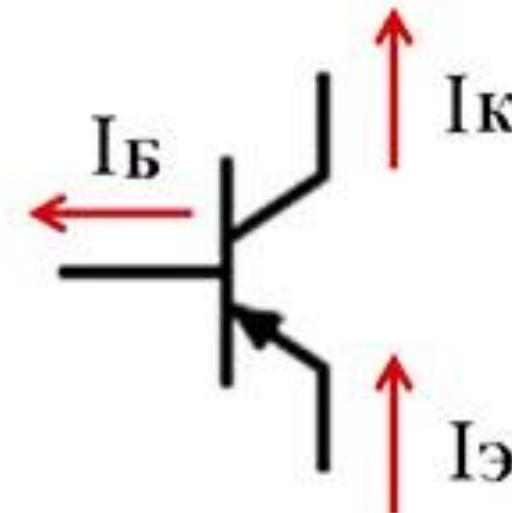
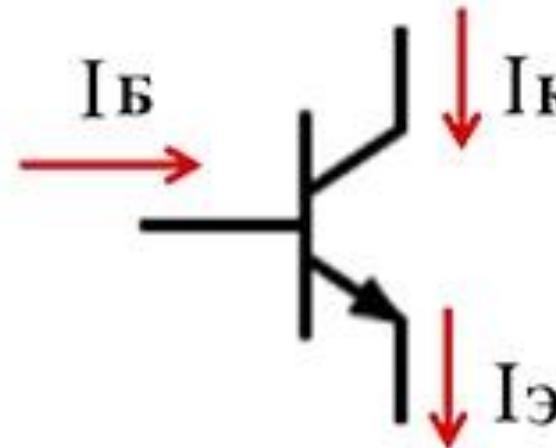
# Биполярный транзистор (BJT)



**NPN**



**PNP**



Биполярный транзистор (BJT, Bipolar Junction Transistor) – управление током через ток базы

# DTL логика

**DTL (Diode-Transistor Logic) - диодно-транзисторная логика**

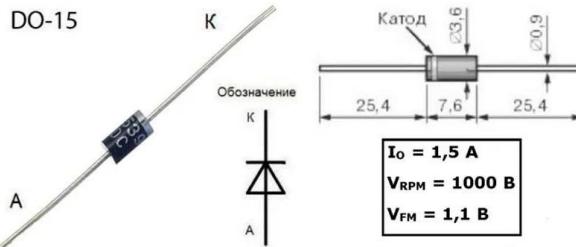
**Диодно-транзисторная логика – технология построения логических схем с применением диодов и биполярных транзисторов.**

Своё название технология получила благодаря реализации логических функций с помощью диодных цепей, а усиления и инверсии сигнала — с помощью транзистора.

**Недостатком ДТЛ** микросхем является задержка прохождения сигнала, что было устранено в следующем поколении – ТТЛ.

**Диод** – это электронный прибор, который позволяет пропускать электрический ток только в одном направлении. Он состоит из двух электродов – катода и анода, и полупроводникового материала, который разделяет электроды. Полупроводник в диоде обычно изготавливается из кремния или германия.

## Распиновка диода 1N5399



Диод имеет два контакта, которые называют **анодом** и **катодом**. При включении диода в электрическую цепь ток протекает от анода к катоду. Умение проводить ток только в одну сторону – **основное свойство диода**.

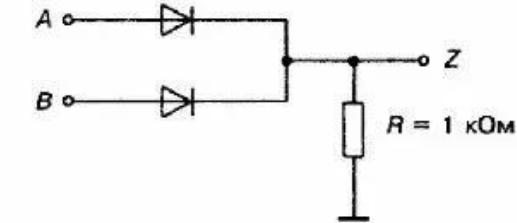


Диоды относятся к классу полупроводников и считаются активными электронными компонентами (**резисторы** и **конденсаторы** – пассивными).

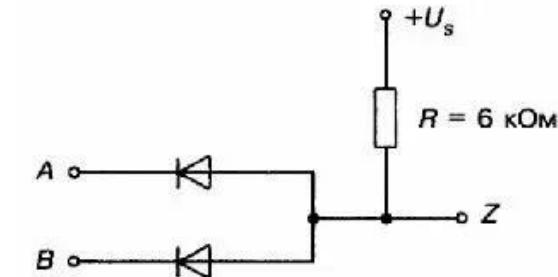


Цифровые часы, собранные только из дискретных транзисторов, диодов и резисторов, без интегральных схем. В этих часах используются 550 переключающих диодов и 196 транзисторов, которые делят частоту сети 60 Гц на один импульс в секунду и обеспечивают отображение часов, минут и секунд.

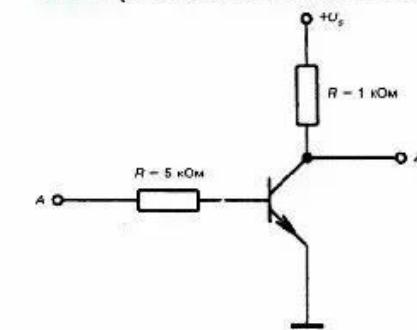
**ИЛИ** (положительная логика)



**И** (положительная логика)

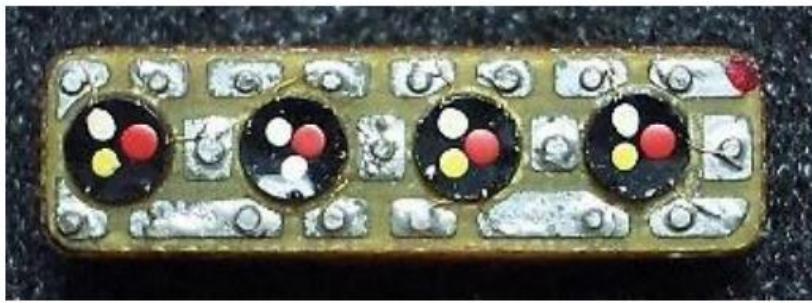
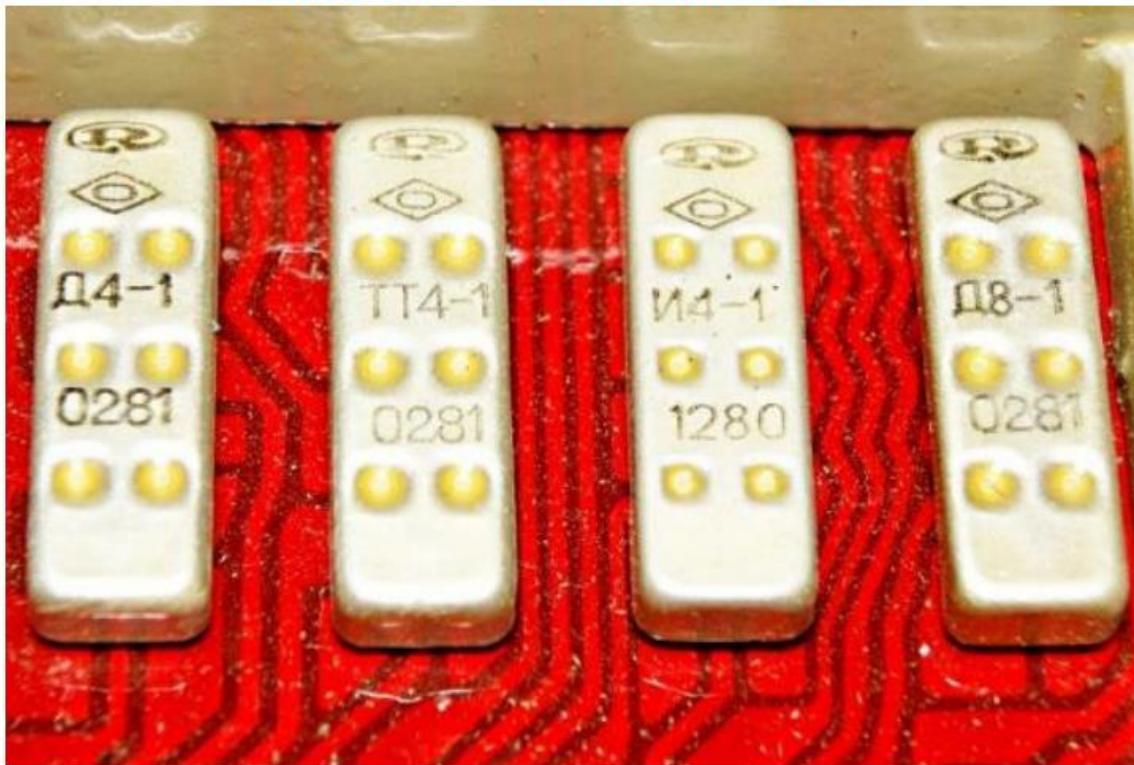


**НЕ** (положительная логика)



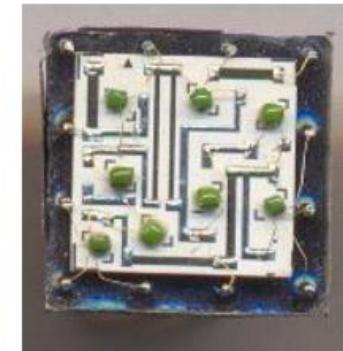
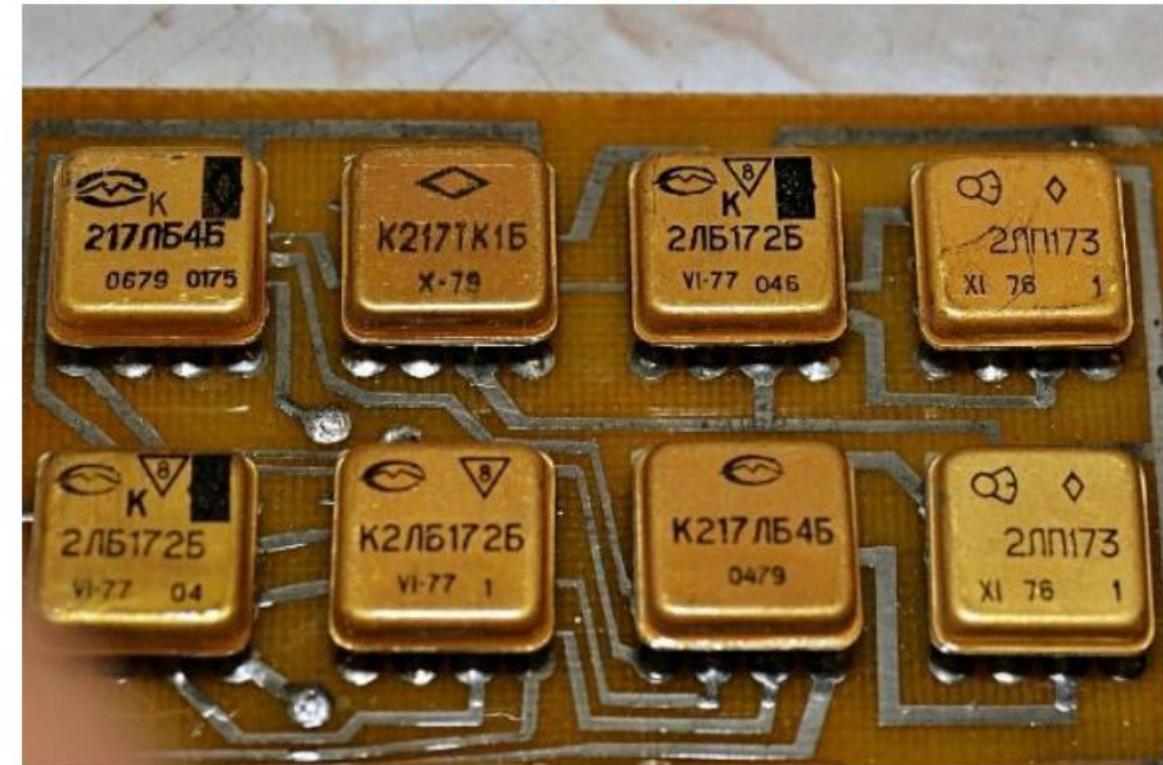
# RTL и DTL логика

Резисторно-транзисторная логика



серия Р12-2,  
1970-е годы

Диодно-транзисторная логика



Микросхемы серии 217,  
выполненные на тонких  
пленках, 1970-е годы

# TTL логика

**TTL** (Transistor-Transistor Logic) - транзисторно-транзисторная логика

**В настоящее время применяются два вида ТТЛ-микросхем — с напряжением питания 3 В и 5 В**, но, независимо от этого, логическому нулю соответствует низкий уровень напряжения около нуля, а логической единице — высокий уровень, близкий к напряжению питания. Поэтому дополнительного согласования между этими ТТЛ-микросхемами обычно не требуется.

**Базовые логические элементы ТТЛ являются элементной базой для микросхем среднего и высокого быстродействия.**

**Базовые логические элементы ТТЛ реализуют логическую функцию И-НЕ** (являются элементами Шеффера) и содержат каскад на много-эмиттерном транзисторе, выполняющий логическую функцию И, и транзисторный ключ-инвертор.

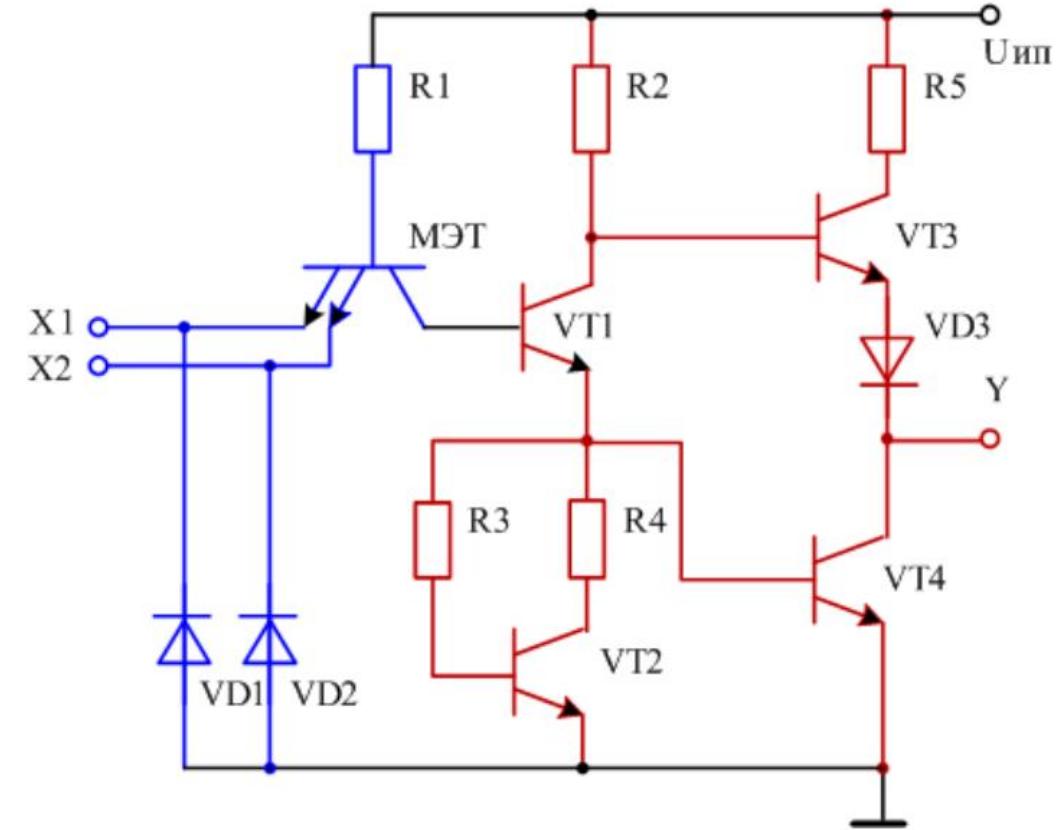
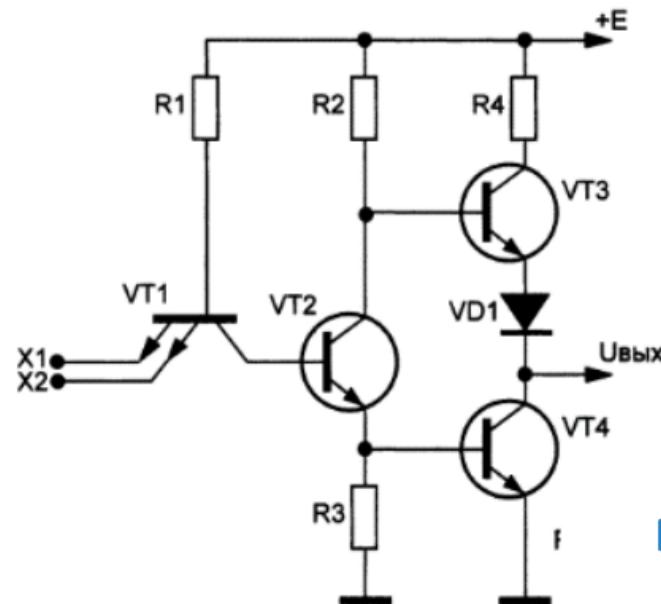


Схема базового логического элемента ТТЛ  
(реализует логическую функцию **И-НЕ**)

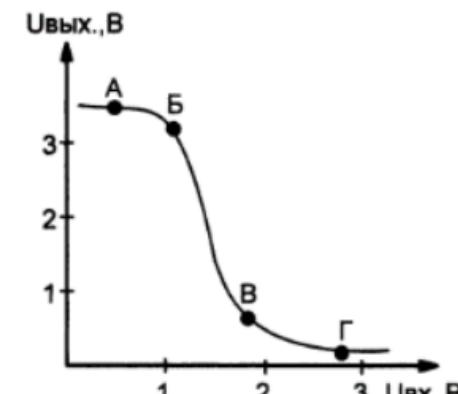
# TTL логика

**TTL** (Transistor-Transistor Logic) - транзисторно-транзисторная логика

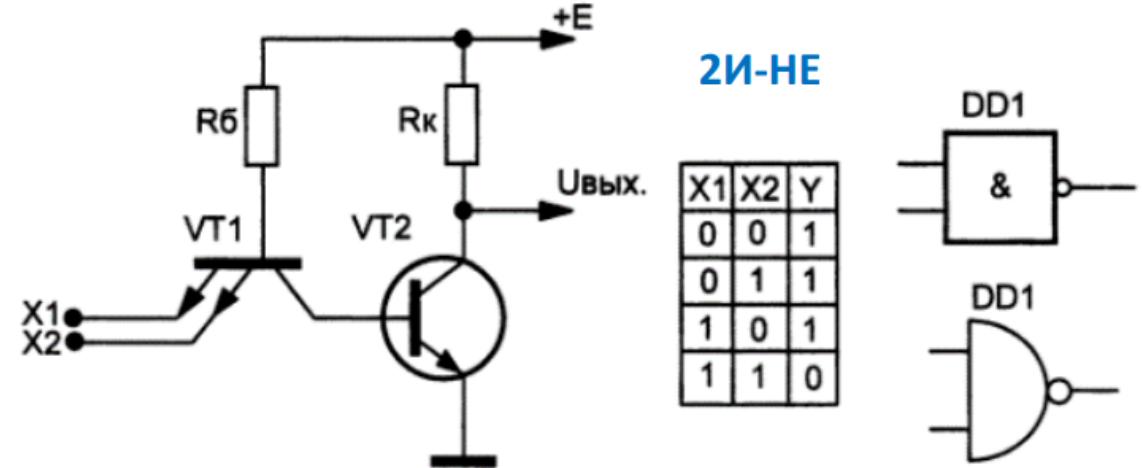
Логика построена на основе биполярных транзисторов и резисторов. Более высокое быстродействие по сравнению с РТЛ и ДТЛ, и более простая технология изготовления. Дополнительным бонусом является усиление выходного сигнала.



Реальная схема 2И-НЕ



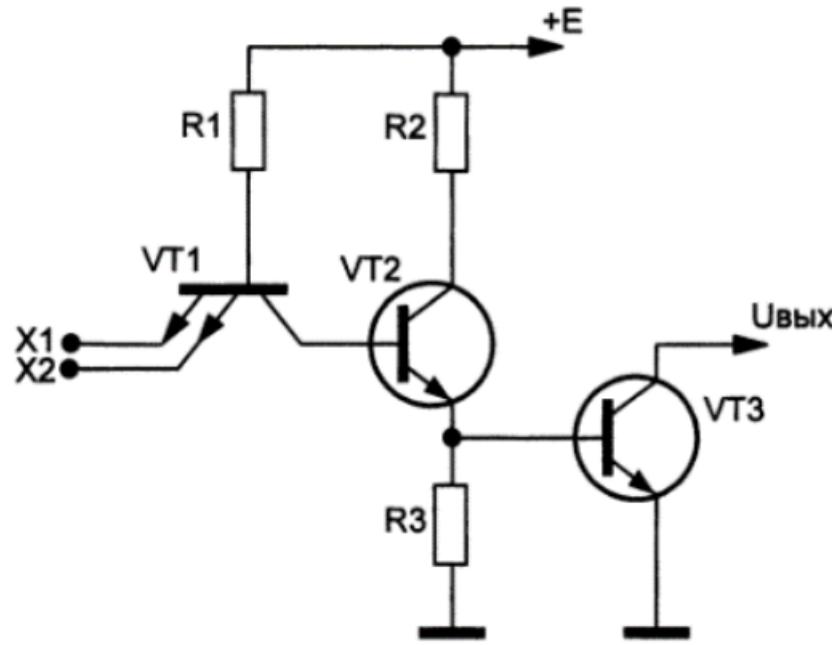
Если напряжение на входах =  $U_H$  ( $> 3.3$  В), VT2 и VT4 находятся в режиме насыщения (участок В-Г на графике), при этом VT3 закрыт, выходное напряжение L ( $\sim 0.2$  В). Если на одном из входов напряжение  $U_L$ , VT2 и VT4 будут закрыты, VT3 – открыт, диод также открыт, на выходе – уровень H (точка А).



Базовый логический элемент ТТЛ повторяет структуру ДТЛ микросхем, но вместо входной диодной сборки в нем использован многоэмиттерный транзистор VT1, к коллекторному выходу которого подключен транзистор VT2, выполняющий функцию инвертирующего усилителя. Заряд, возникающий в базе транзистора VT2, быстро рассасывается через транзистор VT1, что существенно влияет на скорость переключения ТТЛ-схемы.

# TTL логика

**TTL** (Transistor-Transistor Logic) - транзисторно-транзисторная логика



Дополнительный вход управления EZ позволяет переводить микросхему в состояние "отключено" (sleep), подав  $U_L$  на этот вход. Микросхема переходит в высокоимпедансное состояние и резко снижает потребление энергии.

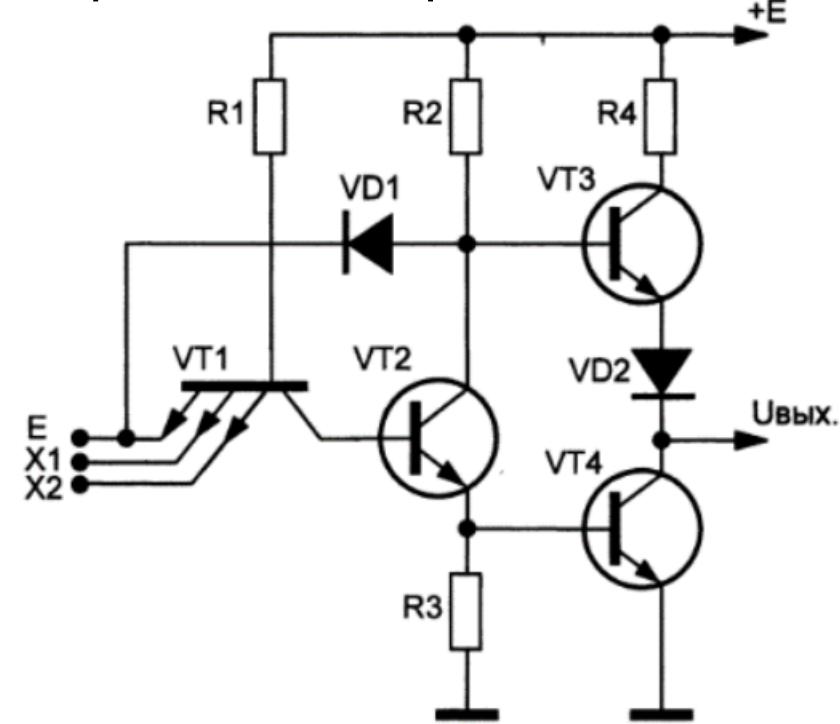
Для повышения нагрузочной способности логического элемента и обеспечения возможности работы его выходного каскада при повышенном напряжении в выходном каскаде используют схему с открытым коллектором. Смещающий резистор подключается "снаружи".

*Достоинства ТТЛ:*

- простота технической реализации, дешевизна
- относительно высокое быстродействие
- умеренное энергопотребление

*Недостатки ТТЛ:*

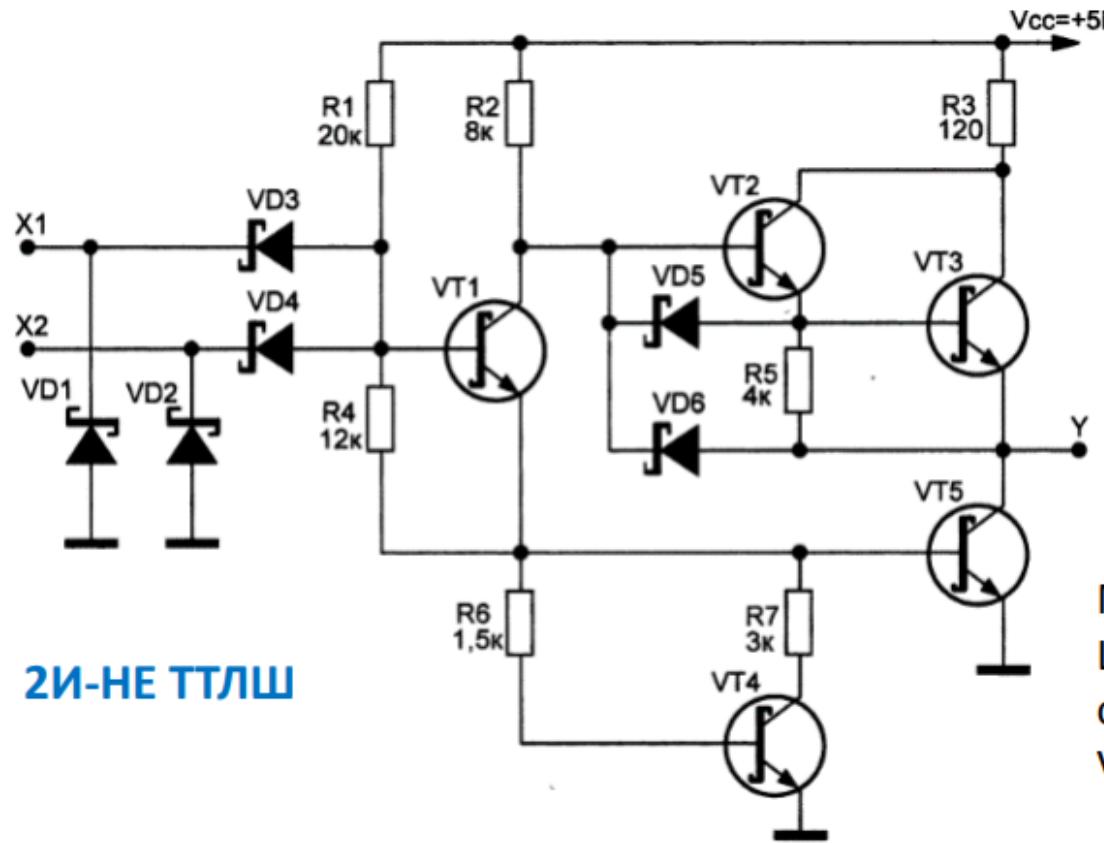
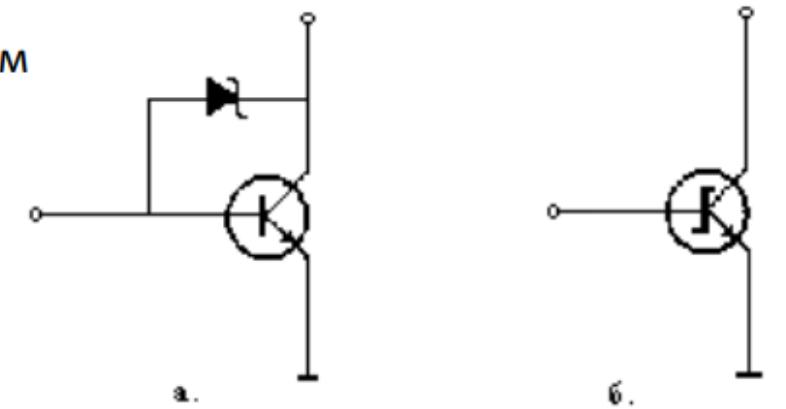
- малый коэффициент разветвления по выходам
- низкая помехоустойчивость (даже по сравнению с ДТЛ)



# TTL логика с диодами Шоттки

## ТРАНЗИСТОРНО-ТРАНЗИСТОРНАЯ ЛОГИКА С ДИОДАМИ ШОТТКИ

Является развитием логики TTL. Отличается повышенным быстродействием при одновременном снижении потребляемой мощности за счёт использования диодов и транзисторов Шоттки. Недостатком микросхем TTLШ в сравнении с TTL является меньшая помехоустойчивость из-за меньшего размаха выходного напряжения.



2И-НЕ TTLШ

Транзистор Шоттки отличается от обычного транзистора тем, что в нем коллектор соединен с базой с помощью диода Шоттки, который предотвращает работу транзистора в режиме насыщения. Благодаря этому соединению исключается эффект накопления электронов в области базы, т. е. в конечном счете уменьшаются задержки, зависящие от эффекта накопления.

Многоэмиттерный вход заменен схемой И из диодов Шоттки. Диоды с заземлённым выводом подавляют осцилляции и отражения во входных сигнальных линиях. VT2 и VT3 представляют собой транзистор Дарлингтона.

# ECL логика

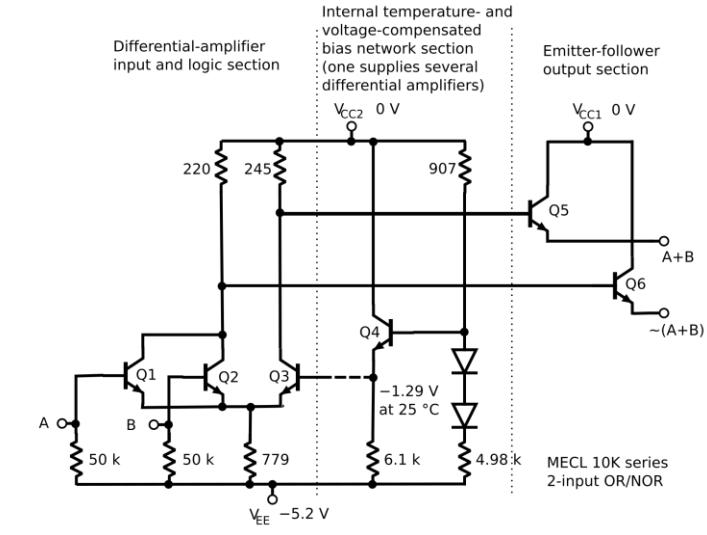
## ECL (Emitter-Coupled Logic) - логика с эмиттерной связью

**Эмиттерно-связанная логика (ЭСЛ, ECL) — способ построения логических элементов на основе дифференциальных транзисторных каскадов.**

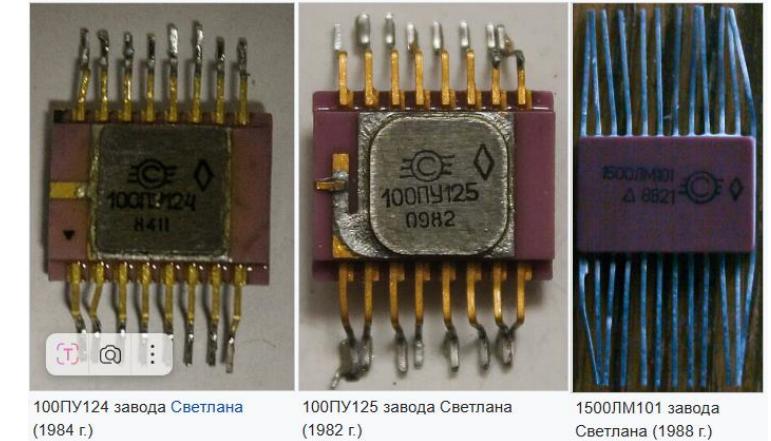
ЭСЛ является самой быстродействующей из всех типов логики, построенной на биполярных транзисторах. Это объясняется тем, что транзисторы в ЭСЛ работают в линейном режиме, не переходя в режим насыщения, выход из которого замедлен. Низкие значения логических перепадов в ЭСЛ-логике способствуют снижению влияния на быстродействие паразитных ёмкостей.

**Основная деталь ЭСЛ-логики — схема** потенциального сравнения, собранная не на диодах (как в ДТЛ), а **на транзисторах**. Схема представляет собой транзисторы, соединённые эмиттерами и подключенные к корпусу (или питанию) через резистор. При этом транзистор, у которого напряжение на базе выше, пропускает через себя основной ток. Как правило, один транзистор в схеме сравнения подключен к опорному уровню, равному напряжению логического порога, а остальные транзисторы являются входами. Выходные цепи схемы сравнения поступают на усилительные транзисторы, а с них — на выходные эмиттерные повторители.

Базовая схема логического элемента ИЛИ/НЕ  
Motorola ECL 10 000, 1972 г.



ЭСЛ советской разработки

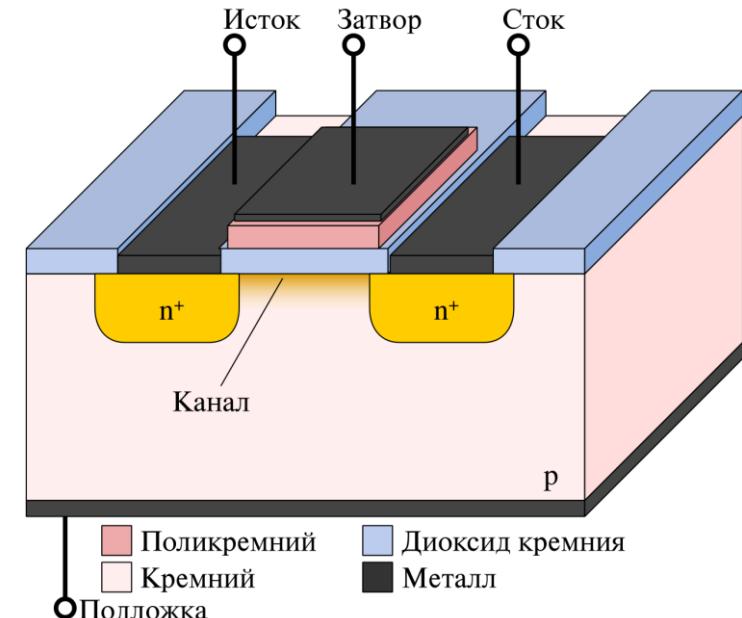


## Интегральные схемы большой и сверхбольшой степени интеграции (LSI, VLSI)

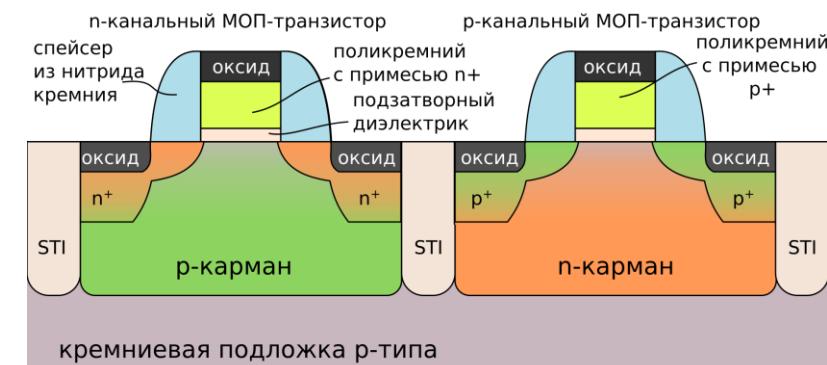
- Третье поколение (1970-е – настоящее время):
- **Интегральные схемы большой и сверхбольшой степени интеграции (LSI, VLSI):**
  - **MOS** (Metal-Oxide-Semiconductor Logic) - логика металл-оксид-полупроводник
  - **CMOS** (Complementary Metal-Oxide-Semiconductor Logic) - комплементарная логика металл-оксид-полупроводник
  - **LVTTL** (Low-Voltage Transistor-Transistor Logic) - низковольтная транзисторно-транзисторная логика
  - **LVCMOS** (Low-Voltage Complementary Metal-Oxide-Semiconductor Logic) - низковольтная логика на комплементарной структуре металл-оксид-полупроводник
  - и др.

# МОП (MOS)

- Схемы металл-оксид-полупроводник (МОП) - MOS (Metal-Oxide-Semiconductor Logic) являются важнейшими компонентами в электронике, образуя основу полевых МОП-транзисторов (МОП-транзисторов).
- МОП — это семейство логических схем, основанных на использовании полевых транзисторов (metal–oxide–semiconductor field-effect transistor (MOSFET, MOS-FET, MOS FET, or MOS transistor)), где током управляет электрическое поле, создаваемое напряжением на затворе.
- Это основополагающая технология для создания интегральных схем, используемая для усиления, переключения и выполнения цифровых логических операций.
- Основным и ключевым элементом МОП логики является МОП-транзистор. Он состоит из трех основных слоев: затвор (gate), исток (source) и сток (drain). Затвор отделен от полупроводникового слоя тонкой оксидной пленкой, что позволяет управлять током, протекающим между истоком и стоком.
- Работа МОП-транзистора основана на управлении током между истоком и стоком с помощью напряжения, приложенного к затвору. Когда на затвор подается положительное напряжение (для n-канальных транзисторов), создается электрическое поле, которое привлекает электроны к области под затвором, образуя проводящий канал. В случае p-канальных транзисторов процесс происходит наоборот, с использованием дырок как носителей заряда. Электрическое поле, создаваемое в слое оксида, модулирует проводимость канала.



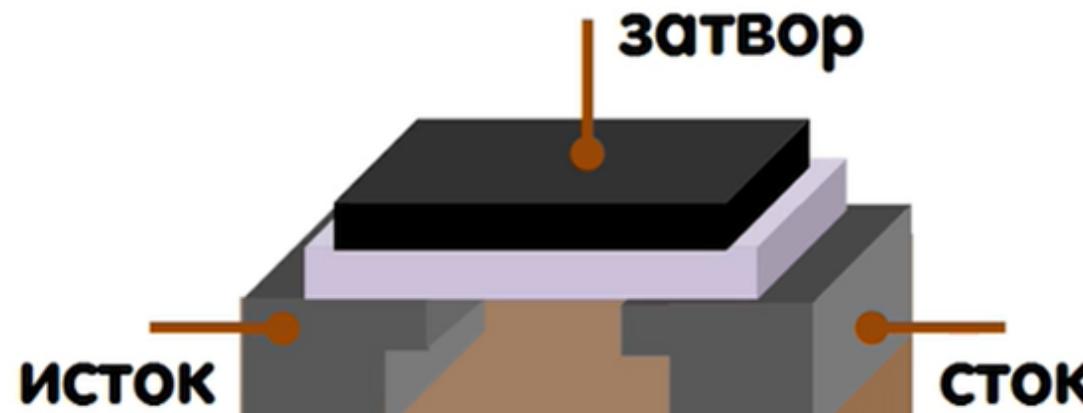
Структура n-канального МОП-транзистора с индуцированным каналом



МОП-транзисторы в современных интегральных микросхемах

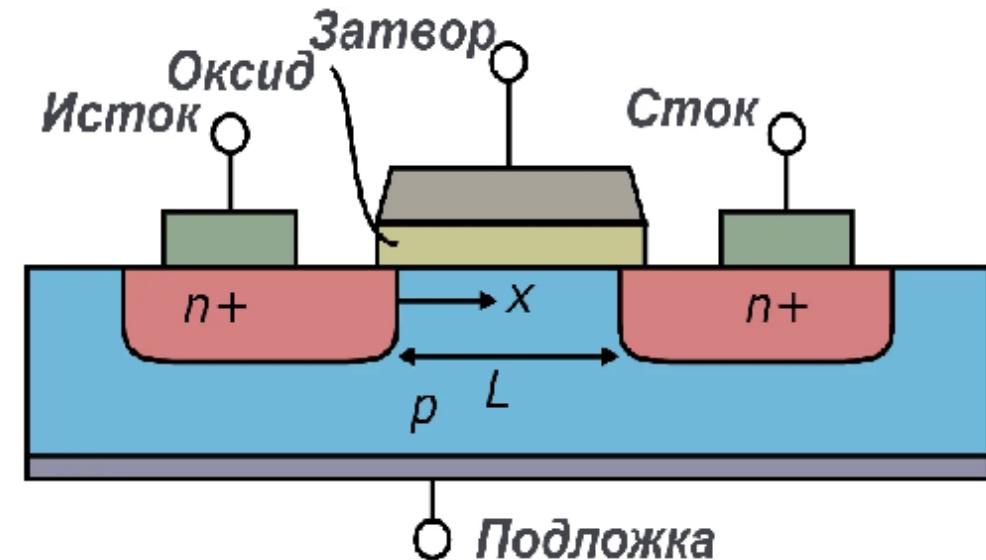
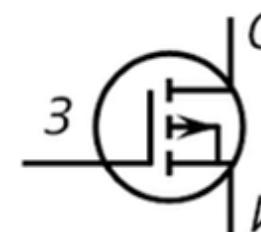
# МОП (MOS)

**МОП (металл-оксид-полупроводник)**



**ИСТОК**

**СТОК**

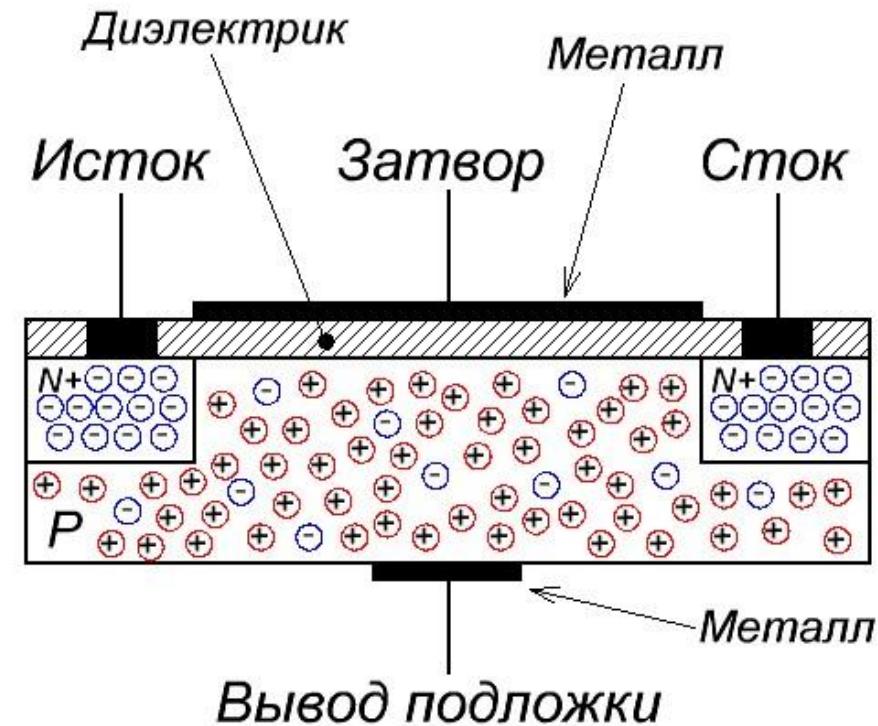
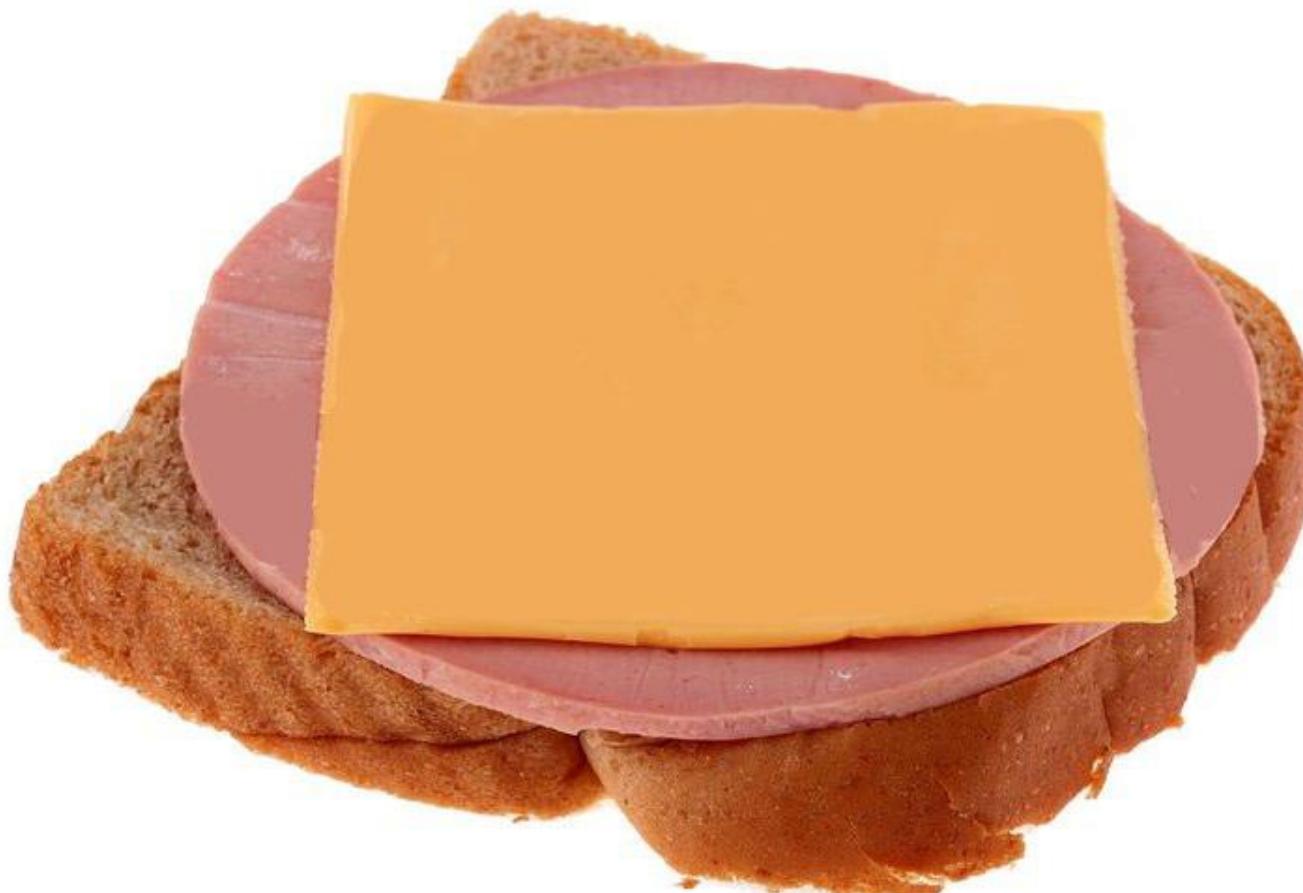


**МДП (металл-диэлектрик-полупроводник)  
MOS (metal-oxide-semiconductor)  
MOSFET**

Полевой транзистор с изолированным затвором по принципу Металл-Оксид-Полупроводник (МОП, МДП, MOS, MOSFET) – сопротивление управляемое напряжением

# МОП (MOS)

## Полевой транзистор МОП (MOSFET)

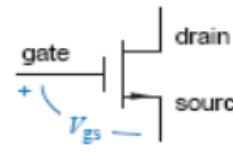


С точки зрения еды на вашем столе, МОП-транзистор будет больше похож на бутерброд.

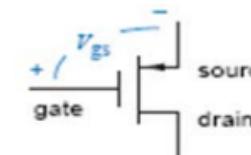
**Полупроводник Р-типа** — толстый кусок хлеба, **диэлектрик** — тонкий слой колбасы, **слой металла** — тонкая пластинку сыра. В результате у нас получается вот такой бутерброд.

# Функционирование МОП-транзистора

В цифровых схемах МОП-транзисторы работают в ключевом режиме (открыт-закрыт).



Управление nМОП-транзистором

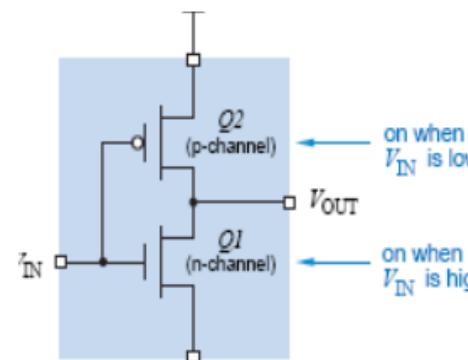


Управление рМОП-транзистором

Если на ЗАТВОР относительно ИСТОКА подано напряжение  $U_{gs} > |U_{th}|$ , то сопротивление между ИСТОКОМ и СТОКОМ становится низким и между ними может протекать ток .

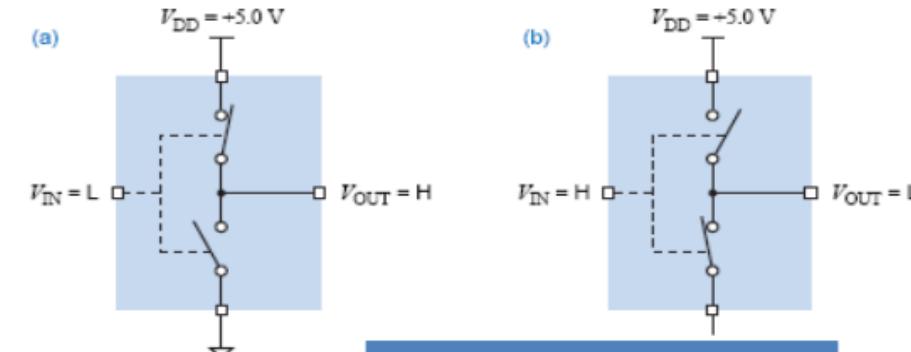
Если на ЗАТВОР относительно ИСТОКА подано напряжение  $U_{gs} < |U_{th}|$ , то сопротивление между ИСТОКОМ и СТОКОМ становится высоким и ток между ними не протекает.

*U<sub>th</sub>* (threshold voltage) – пороговое напряжение «открывания» транзистора, для цифровых микросхем равно примерно половине напряжения питания..



Управление двухтактным каскадом на комплементарных рМОП, нМОП транзисторах

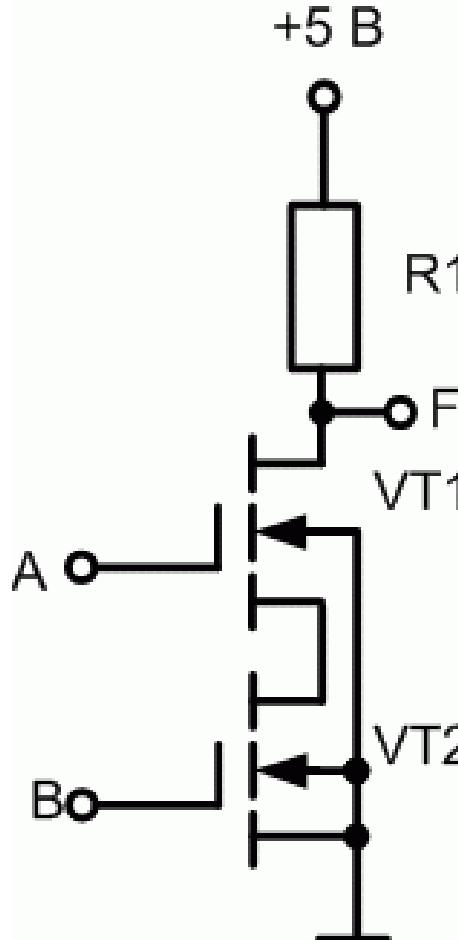
Полярность управляющего (З-И) и питающего (С-И) напряжений:  
нМОП – положительное напряжение.  
рМОП – отрицательное напряжение



Эквивалентная схема двухтактного каскада

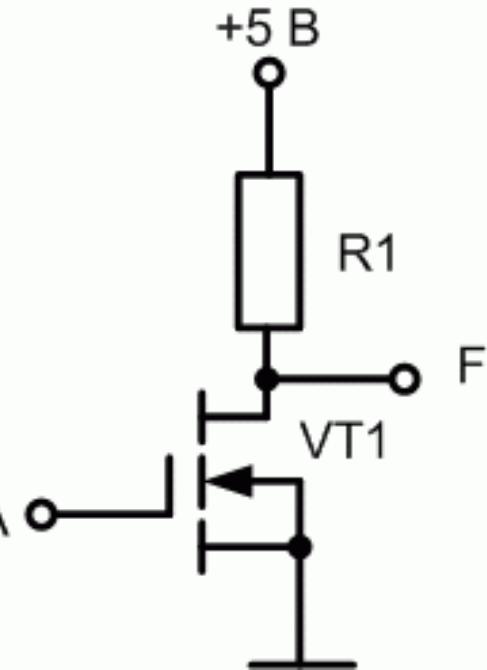
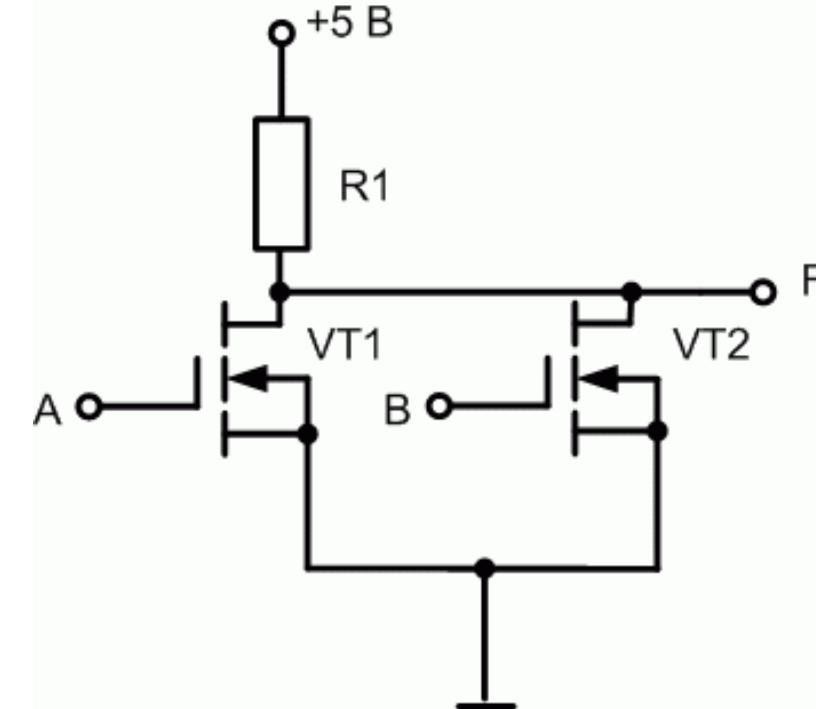
# Схемы МОП (MOS)

Схемы **nМОП**-технологии базируются на полевых (МОП) транзисторах с индуцированным каналом n-типа. **Базовым элементом данной технологии является схема И-НЕ.**



Логический элемент **И-НЕ** nМОП-технологии

Логическое умножение осуществляется за счет последовательного соединения каналов транзисторов VT1 и VT2. Канал между истоком и стоком в nМОП-транзисторе индуцируется в том случае, когда на затвор (вход схемы) подается положительный относительно подложки потенциал. Цель от +5 В до земли замкнется только в одном случае, когда  $A=B=1$ , поскольку в этом случае оба транзистора открываются и образуется единый канал, замыкающий цепь.

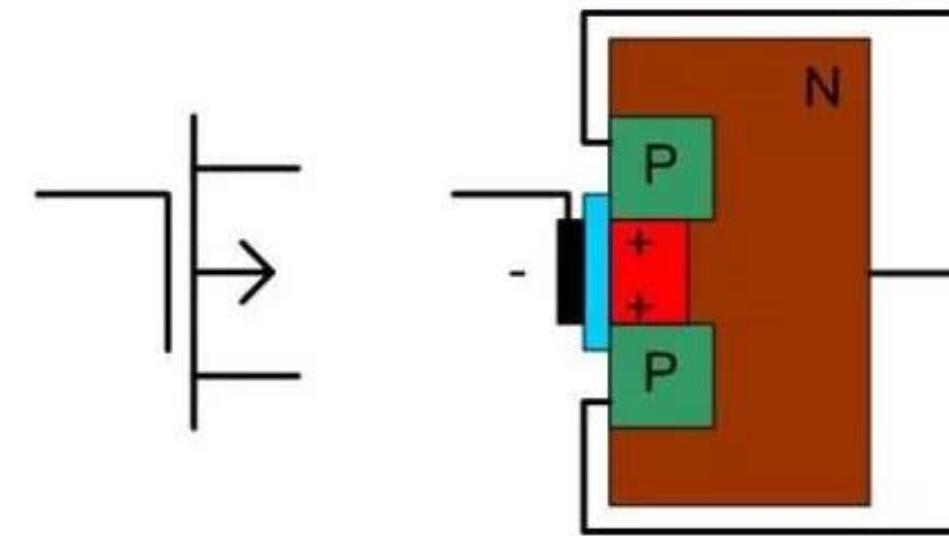


Логический элемент **НЕ**  
nМОП-технологии

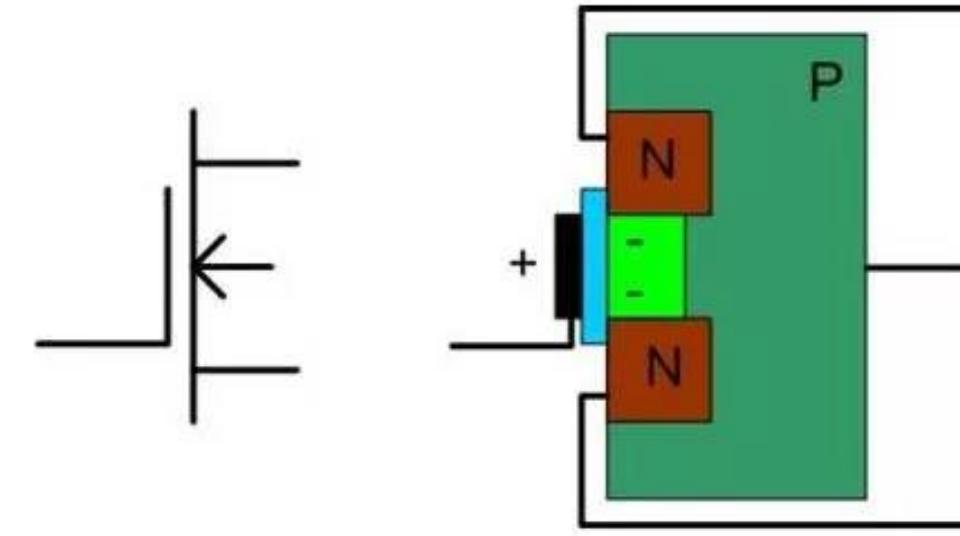
Функция ИЛИ-НЕ осуществляется за счет параллельного соединения таких транзисторов: при подаче хотя бы на один вход единицы индуцируется канал в соответствующем транзисторе и замыкается цепь от +5 В до земли. Следовательно, на выходе будет потенциал, соответствующий падению напряжения в канале транзистора, т.е. 0,2 В, при этом  $F=0$ .

Логический элемент **ИЛИ-НЕ** nМОП-технологии

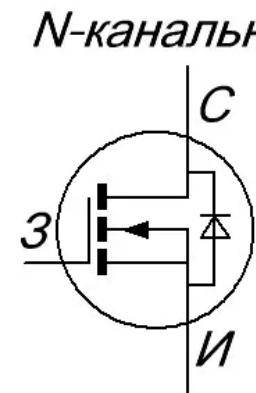
# МОП (MOS) - рMOS / nMOS



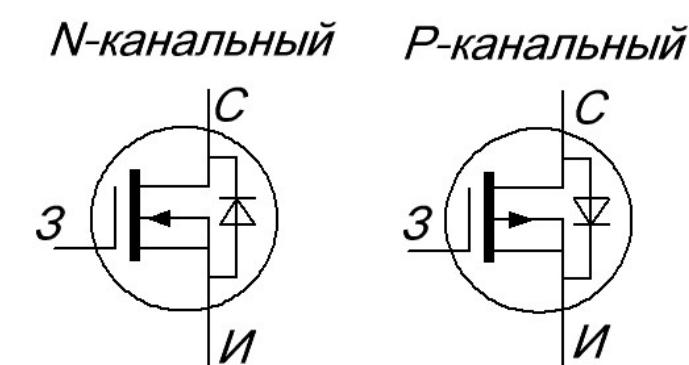
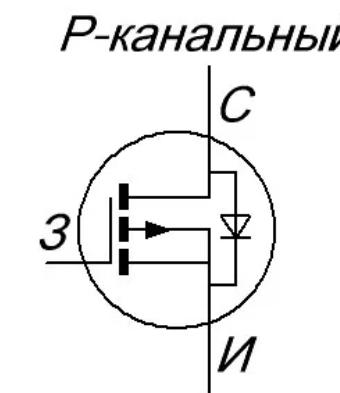
Р-канальный МОП-транзистор



N-канальный МОП-транзистор



с индуцированными каналами



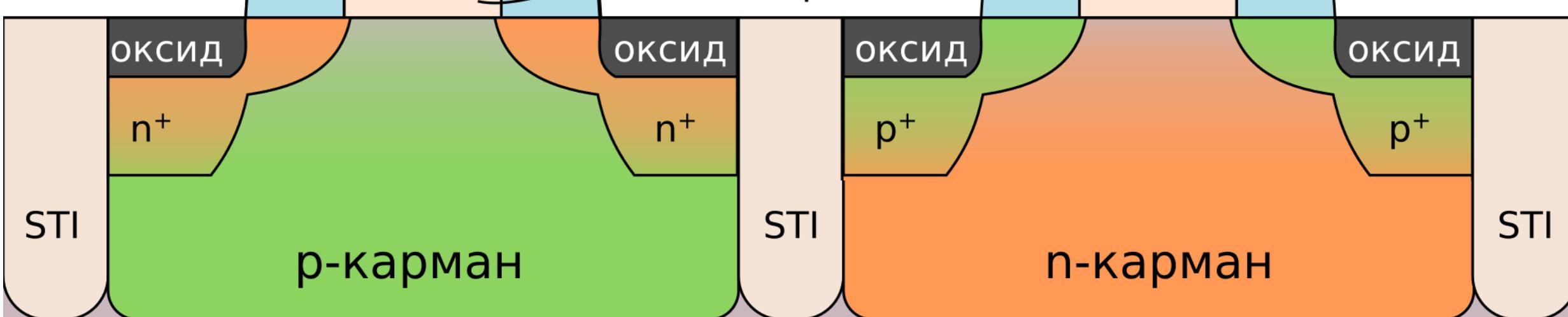
со встроенными каналами

# МОП (MOS) - pMOS / nMOS

n-канальный МОП-транзистор

спейсер  
из нитрида  
кремния

оксид  
поликремний  
с примесью n+  
подзатворный  
диэлектрик



p-канальный МОП-транзистор

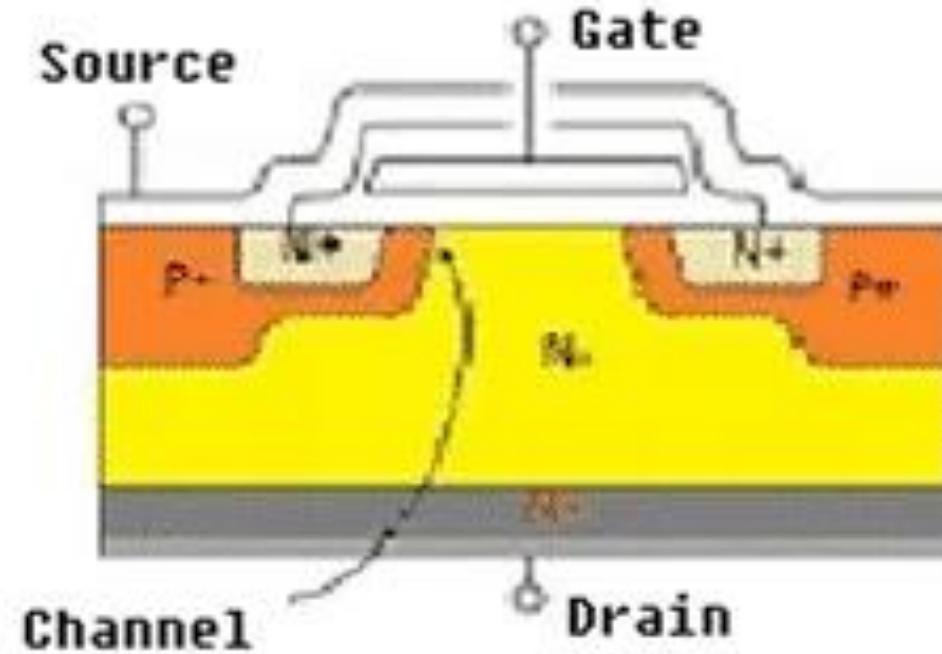
поликремний  
с примесью  
p+



кремниевая подложка р-типа

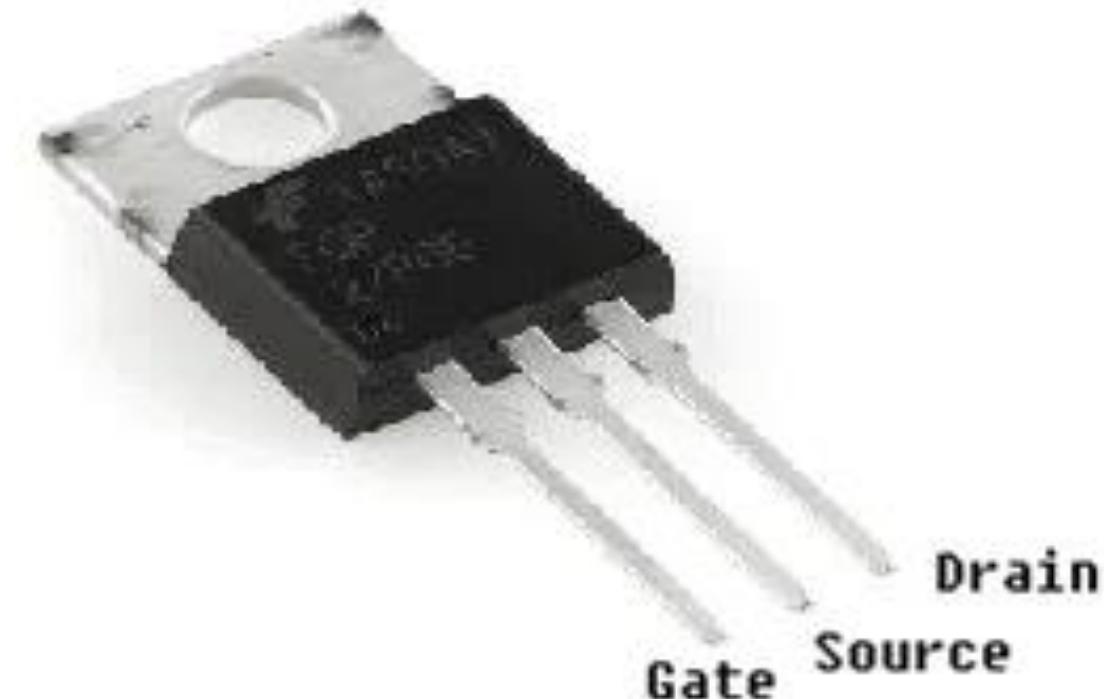
**МОП-транзисторы** в современных интегральных микросхемах

# МОП (MOS)



Structural View

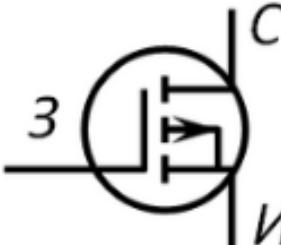
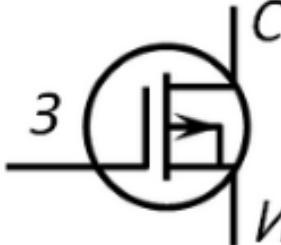
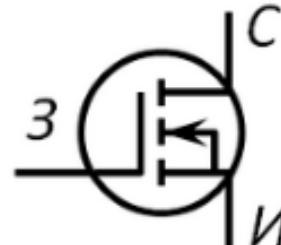
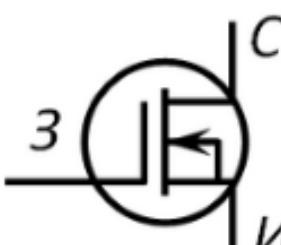
## MOSFET



Physical View

# МОП (MOS) - рMOS / nMOS

Условные графические обозначения полупроводниковых приборов регламентируются ГОСТ 2.730-73

|  | Индукционный<br>канал  | Встроенный<br>канал  |
|--|--|--|
| Р-канал  |   |   |
| N-канал  |  |  |
| Условные обозначения: З — затвор (G — Gate), И — исток (S — Source),<br>С — сток (D — Drain) |  |  |

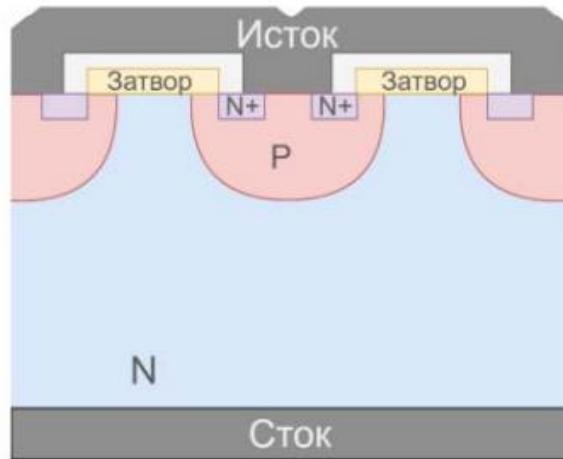
# МОП (MOS)

## VD MOSFET транзистор

VD (Vertical Double-Diffused) MOSFET.

Планарные вертикальные полевые транзисторы с двойной диффузией

### VD (Vertical Double-Diffused) MOSFET



Структура планарных VD MOSFET

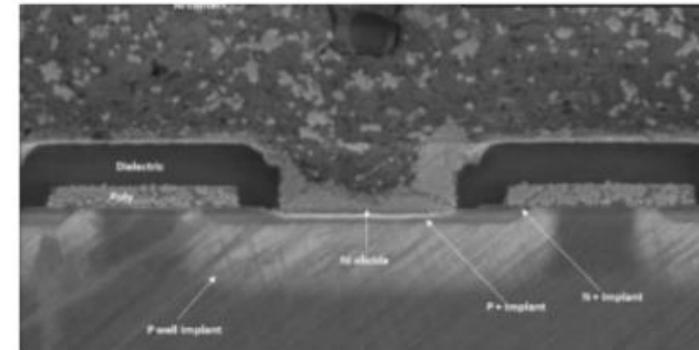
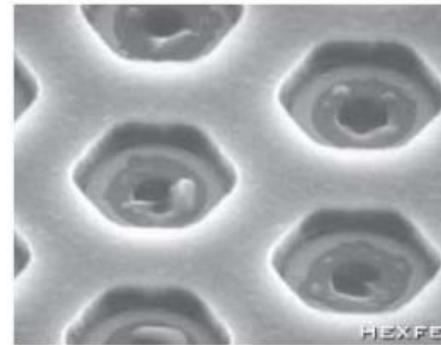


Фото разреза планарного VD MOSFET



Форма единичных ячеек на кристалле планарного транзистора HEXFET

#### Преимущества

- + Высокое напряжение сток-исток ( $V_{ds}$ )
- + Высокая устойчивость к лавинному пробою

#### Недостатки

- Высокие  $Q_g$  и  $C_{iss}$
- Высокая стоимость кристалла из-за его размера

#### Особенности

- $V_{ds}$  до 1500В
- $R_{ds(ON)}$  единицы Ом
- 5-8 слоев литографии
- Шаг ячейки ~20 мкм

#### Импульсные источники питания



#### Драйверы электродвигателей



#### DC/DC преобразователи

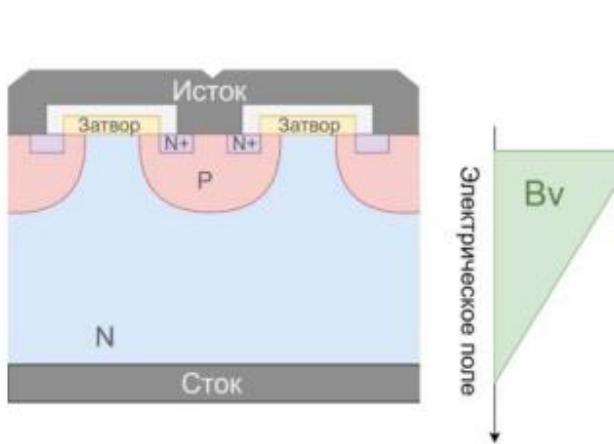


# МОП (MOS)

## SJ MOSFET

SJ (Super Junction) MOSFET.

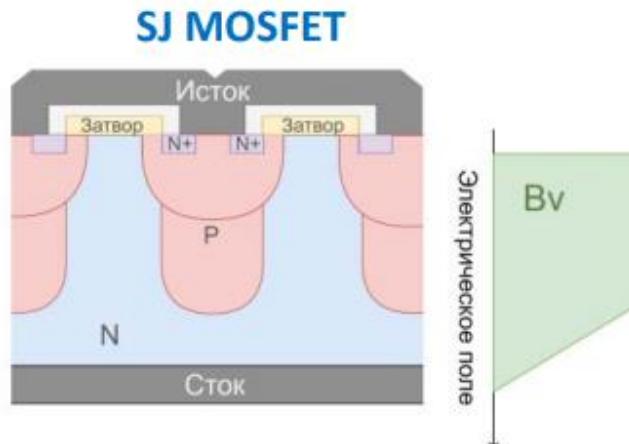
Полевые транзисторы с супер переходом



Структура планарных VD MOSFET  
и распределение электрического  
поля в кристалле транзистора

### Преимущества

- +  $R_{ds(ON)}$  ниже чем у VD MOSFET
- +  $Q_g$  и  $C_{iss}$  ниже чем у VD MOSFET.
- + Меньшие габариты кристалла по сравнению с VD MOSFET
- + Нет значительного увеличения  $R_{ds(ON)}$  при увеличении рабочего  $V_{ds}$



Структура SJMOSFET и  
распределение электрического поля  
в кристалле транзистора

### Недостатки

- Сложный производственный процесс.
- Ниже устойчивость к лавинному пробою в сравнении с планарными MOSFET

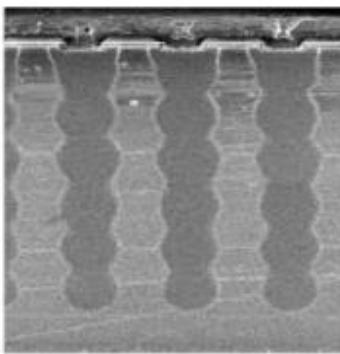


Фото разреза SJ MOSFET

### Особенности

- $V_{ds}$  до 250В
- $R_{ds(ON)}$  единицы...десятки мОм
- До 15 слоев литографии
- Шаг ячейки ~13 мкм

Резонансные преобразователи с PFC



Импульсные источники питания



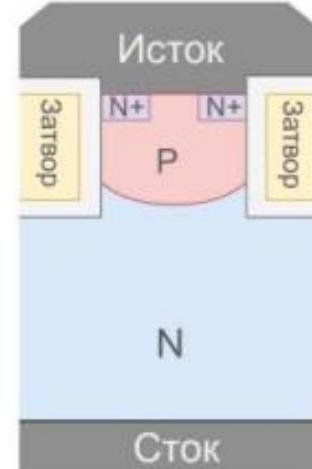
Бесперебойные источники питания



# МОП (MOS)

## Trench MOSFET

Полевой транзистор с затвором в «траншее»



Структура Trench MOSFET

### Преимущества

- + Низкое  $R_{ds(On)}$ .
- +  $I_d$  выше чему VD и SJ MOSFET
- + Малые габариты единичной структуры
- + Малые габариты кристалла по сравнению с планарными MOSFET

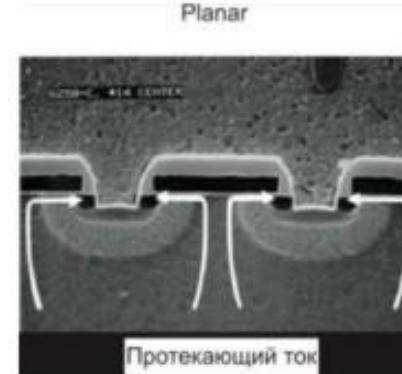


Фото разреза планарного VD MOSFET с указанием пути протекания тока

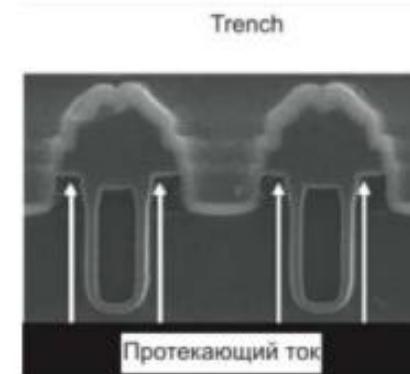
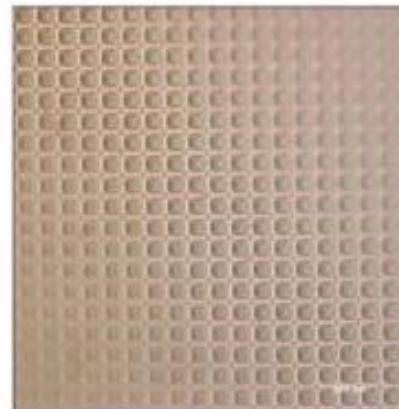


Фото разреза Trench MOSFET с указанием пути протекания тока



Форма единичных ячеек на кристалле Trench MOSFET

### Недостатки

- Низкое напряжение сток-исток ( $V_{ds}$ )
- Низкие динамические характеристики

### Особенности

- $V_{ds}$  до 250В
- $R_{ds(ON)}$  единицы – десятки мОм
- 5-7 слоев литографии
- Шаг ячейки 5 мкм



Бесперебойные источники питания



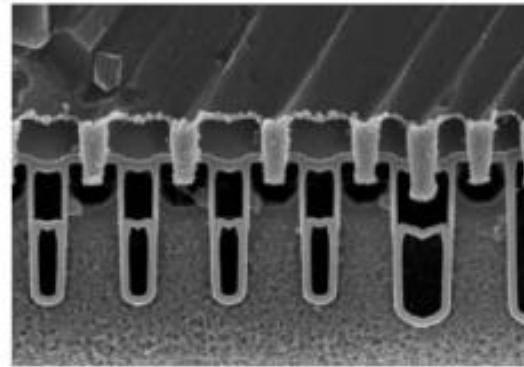
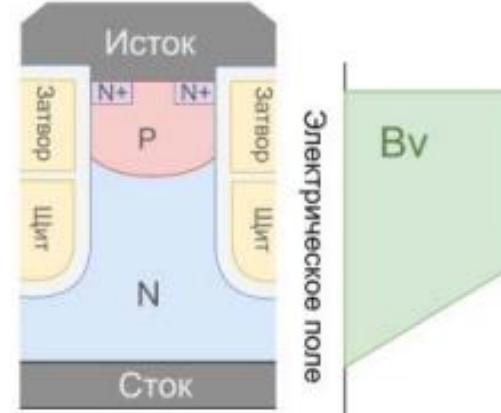
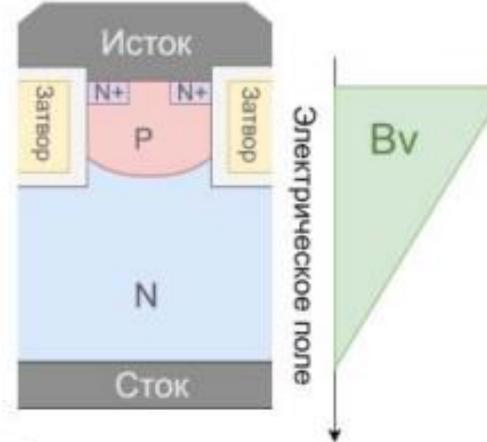
DC/DC преобразователи



# МОП (MOS)

**SGT MOSFET.** Полевой транзистор с экранированным затвором

## SGT MOSFET. Полевой транзистор с экранированным затвором



Структура Trench MOSFET и распределение электрического поля в кристалле транзистора

Структура SGT MOSFET и распределение электрического поля в кристалле транзистора

Фото разреза SGT MOSFET

### Преимущества

- +  $R_{ds(ON)}$  ниже чем у Trench MOSFET.
- +  $Q_g$  и  $C_{iss}$  ниже чем у Trench MOSFET.
- + Выше устойчивость к лавинному пробою
- + Экран выполняет роль снаббера и уменьшает потери на переключение.

### Недостатки

- Сложный производственный процесс.

### Особенности

- $V_{ds}$  до 200В
- $R_{ds(ON)}$  десятые доли...единицы мОм
- 6-9 слоев литографии
- Шаг ячейки ~3 мкм



Системы управления батареями

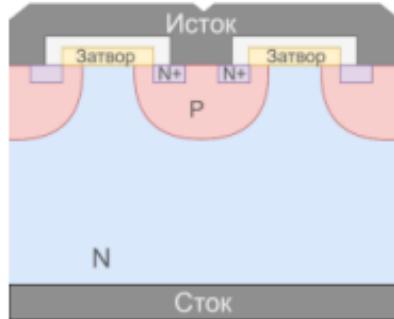


# МОП (MOS)

Солнечные преобразователи

**SiC MOSFET.** Полевые транзисторы на основе карбида кремния

**SiC MOSFET. Полевые транзисторы на основе карбида кремния**



Структура планарного Si MOSFET



Дрейфовый N-слой в 10 раз  
тоньше благодаря высокой  
электрической прочности SiC.

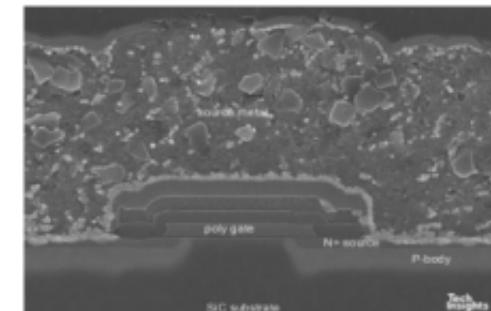


Фото разреза планарного SiC MOSFET

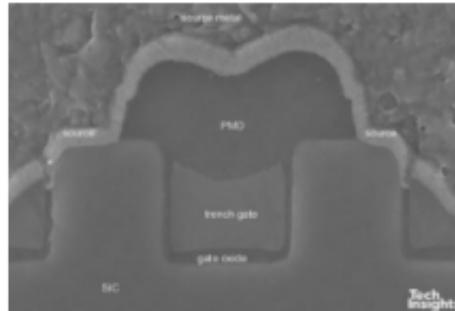


Фото разреза Trench SiC MOSFET

## Преимущества

- + Высокое  $V_{ds}$
- + Высокая частота переключений
- + Низкое  $R_{ds(ON)}$
- + Электрические свойства слабо зависят от температуры
- + Высокая теплопроводность

## Недостатки

- Высокая сложность производства
- Высокая стоимость

## Особенности

- $V_{ds}$  до 1700В
- $R_{ds(ON)}$  десятые доли, единицы мОм
- Шаг ячейки ~5 мкм
- ~300 операций при производстве



Модульные бесперебойные  
источники питания



Электрозарядные станции



Драйверы маломощных  
промышленных электродвигателей



# Характеристики МОП-транзисторов

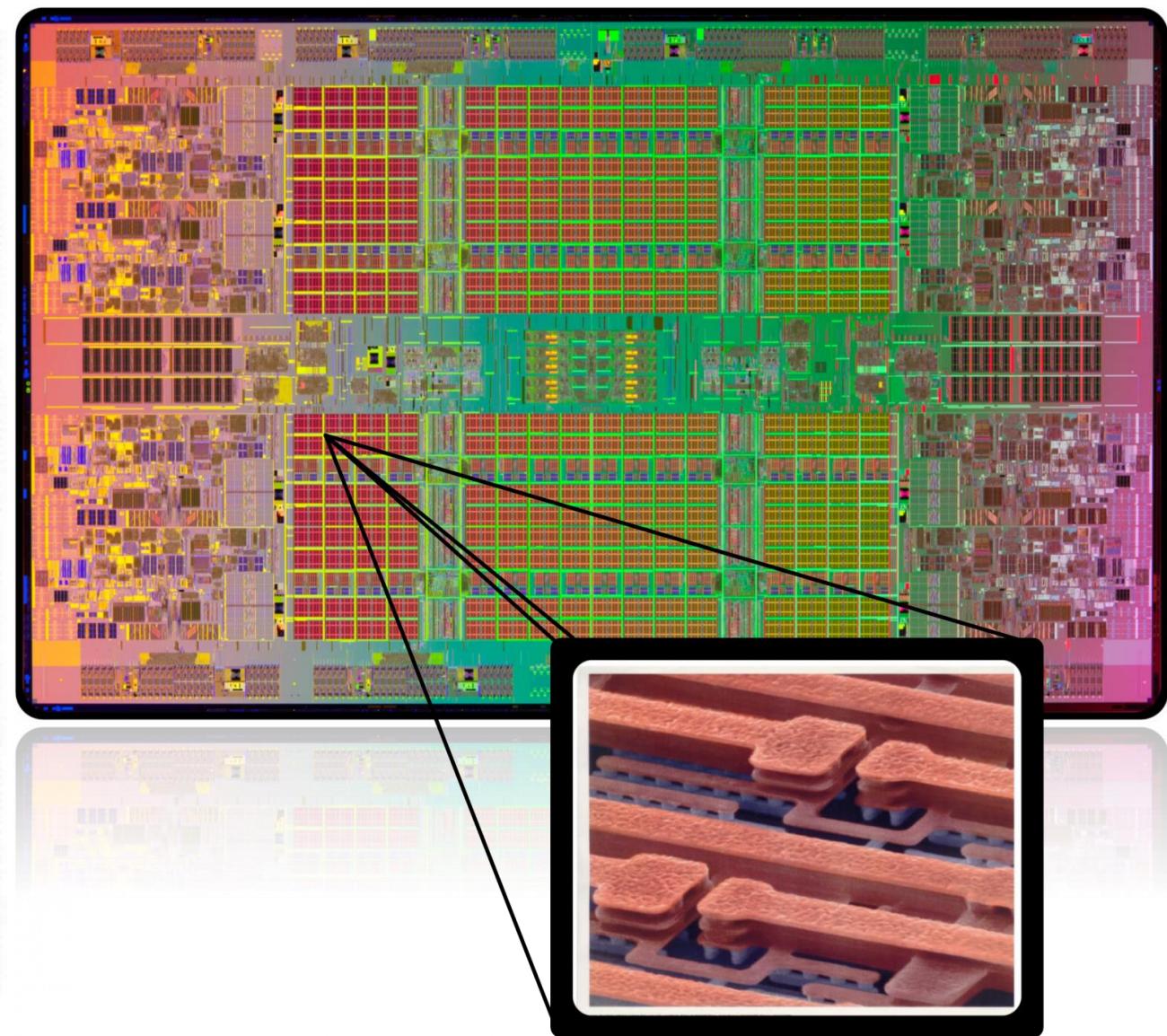
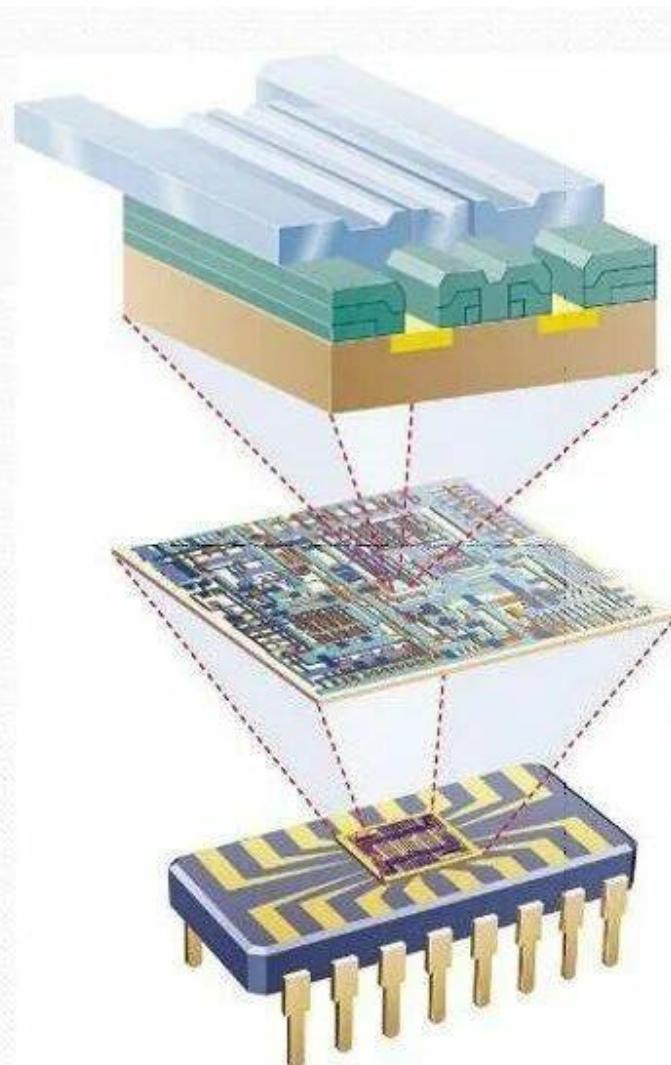
- **Большое входное сопротивление** (неск. МОм) – низкое потребление, большой коэффициент разветвления по выходу.
- **Малое выходное сопротивление** И-С открытого транзистора – большая выходная мощность, большой коэффициент разветвления.
- **Высокое напряжение переключения**, дающее большой запас помехоустойчивости.
- **Малая площадь каскадов/схем на кристалле микросхемы**
- **Большая входная емкость З-И** - большой импульсный ток при переключении: эл-маг. помехи, большое потребление
- **Большая входная емкость С-И** - большая длительность переключения выхода и задержка распространения сигнала.

# МОП (MOS)

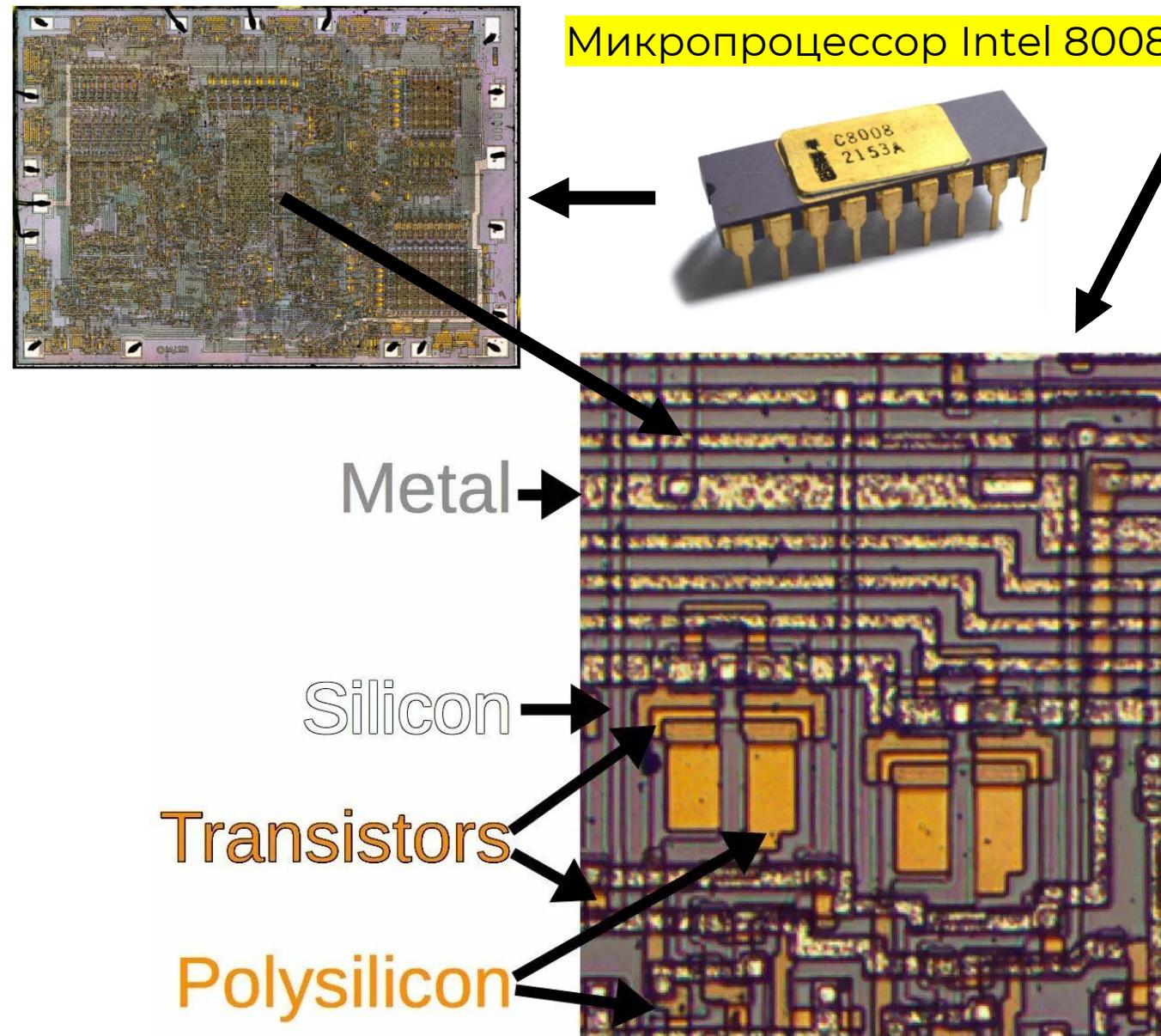
- **Примеры использования МОП-транзисторов в компьютере:**

- **Центральный процессор (CPU):** МОП-транзисторы являются основными строительными блоками микропроцессоров. Они используются для выполнения логических операций и управления потоком данных внутри процессора. Например, современные процессоры Intel и AMD содержат миллиарды МОП-транзисторов.
- **Оперативная память (RAM):** В модулях оперативной памяти (DRAM) МОП-транзисторы используются для хранения битов информации. Они позволяют быстро записывать и считывать данные, необходимые для работы компьютера.
- **Видеокарта (GPU):** Графические процессоры также содержат огромное количество МОП-транзисторов, которые используются для обработки графики и выполнения сложных вычислений, необходимых для игр и других графических приложений.
- **Твердотельные накопители (SSD):** В SSD МОП-транзисторы используются в качестве ячеек памяти для хранения данных. Они обеспечивают быстрый доступ к данным и более высокую надежность по сравнению с традиционными жесткими дисками.
- **Материнская плата:** На материнской плате МОП-транзисторы используются в различных цепях питания и управления, обеспечивая стабильную работу всех компонентов компьютера.
- **МОП-логика — основа цифровой революции.** Её развитие (CMOS, FinFET, GAAFET) определяет прогресс микроэлектроники: от первых микропроцессоров (Intel 4004, 1971) до нейросетевых ускорителей.
- **Несмотря на физические ограничения, технологии MOS остаются доминирующими благодаря сочетанию плотности, энергоэффективности и стоимости.**

# Современные транзисторы



# Транзисторы в микросхемах (Intel 8008)

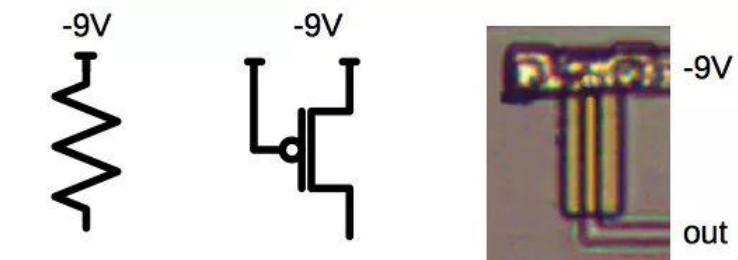
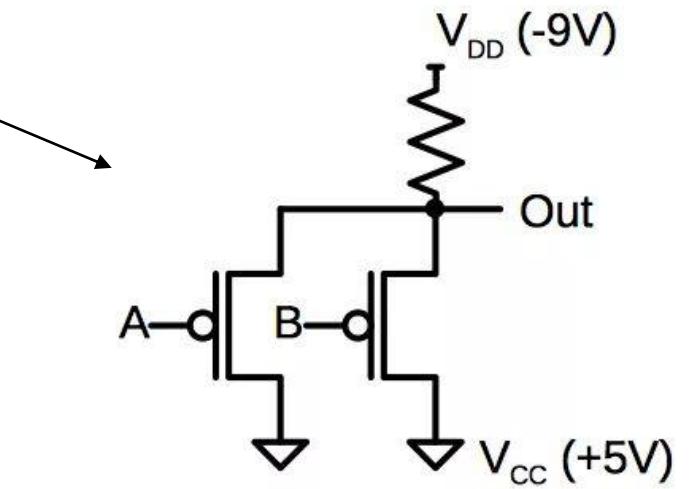
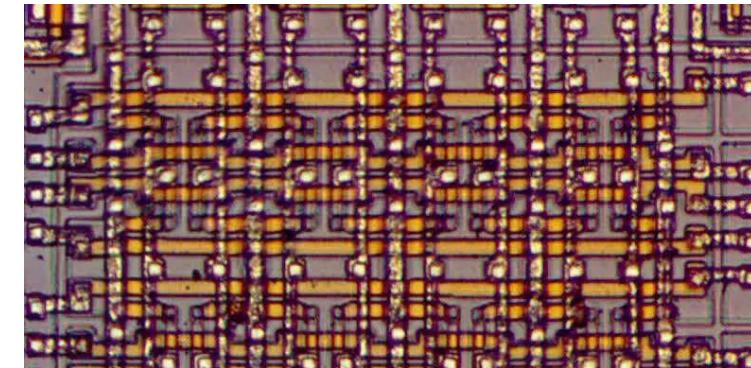


Давайте представим чип как комбинацию трех слоев. На рисунке ниже изображен фрагмент чипа крупным планом с указанием этих слоев. Самый верхний слой – это металлическая проводка. Это наиболее заметная часть микросхемы, что не удивительно (металл видно сразу). На данном фрагменте чипа провода располагаются преимущественно горизонтально. Под металлическими проводами лежит слой из поликристаллического кремния (поликремния), который на фото, сделанном под микроскопом, имеет выраженный оранжевый цвет. Основой чипа служит кремниевая «вафля», которая на фото выглядит пурпурно-серой. Чистый кремний – это практически непроводящий материал. Полупроводниковый кристалл из него получают путем легирования – добавления примесей на определенные участки. Находящийся в самом низу кремниевый слой визуально трудно различить, но можно увидеть черные линии вдоль границ между участками чистого и легированного кремния. На фото видны несколько таких кремниевых "проводов".

**Ключевыми элементами чипа являются транзисторы, и транзисторы формируются там, где поликремний пересекают провода из легированного кремния.** На фото участки поликремния в местах формирования транзисторов выглядят ярко оранжевыми.

# Транзисторы в микросхемах (Intel 8008)

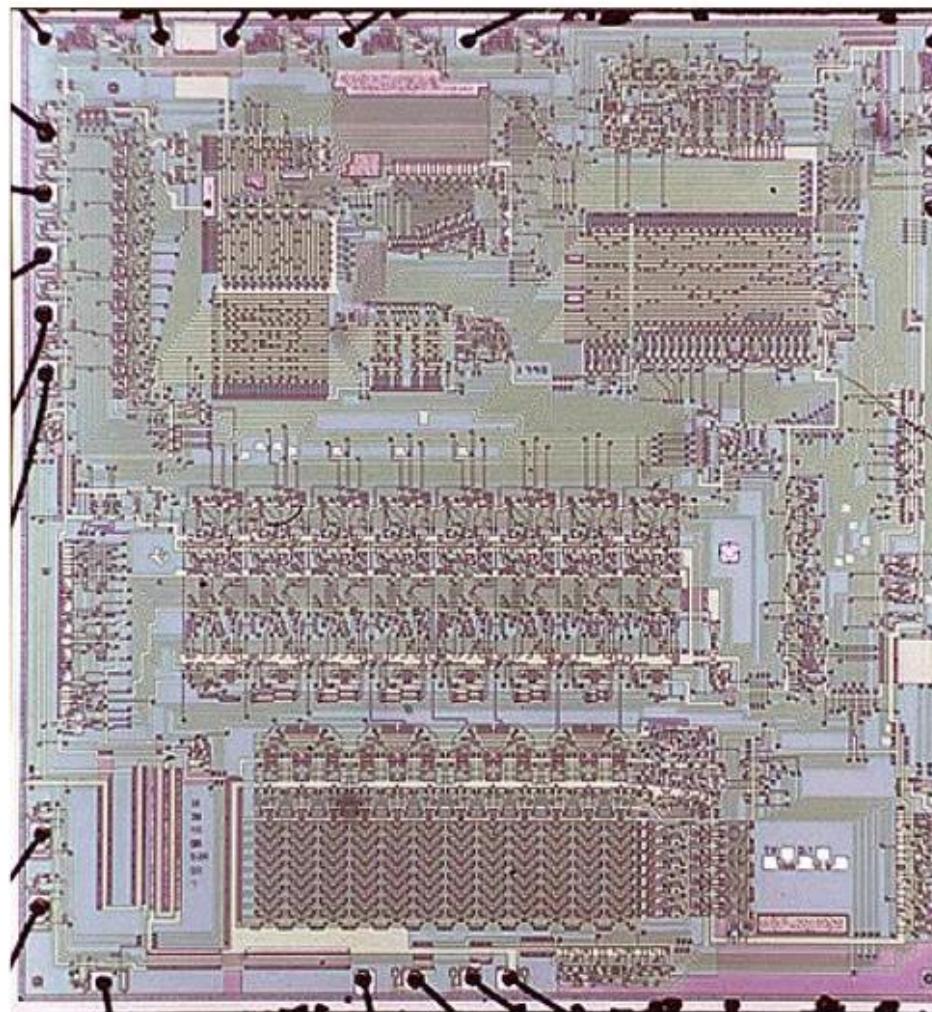
- Процессор 8008 использует **полевые транзисторы с изолированным затвором типа PMOS (р-канальные МОП-транзисторы)**. Упрощенно можно представить PMOS-транзистор как переключатель между двумя кремниевыми проводами, управляемый входным сигналом, который подается на поликремниевый затвор. Когда на затвор подается низкий уровень сигнала (0 на входе), транзистор закрыт и на выходе сигнал высокий (1). Если вы знакомы с принципом работы транзисторов типа NMOS (n-канальных МОП-транзисторов), которые использовались, например, в микропроцессоре 6502, то PMOS могут поначалу вводить в недоумение, потому что работают с точностью до наоборот.
- Схема логического вентиля NAND на транзисторах PMOS**  
Показана на рисунке справа. Когда на обоих входах сигнал высокий, транзисторы выключены и за счет резистора выходной сигнал низкий. Когда на любом из входов сигнал низкий, транзистор проводит ток, соединяя выход с полюсом +5 В, и на выходе сигнал высокий. Таким образом реализуется логическая функция "не-И" (NAND). Для обеспечения совместимости с 5-вольтовыми цепями TTL (на биполярных транзисторах) в схемах на PMOS (соответственно, и в процессоре 8008) используются необычные потенциалы входного напряжения: -9 В и +5 В.
- Из технических соображений резистор фактически выполняется заодно с транзистором. На рисунке ниже показано, как подключается транзистор, чтобы он мог работать как понижающий резистор. **Фрагмент фото справа показывает, как эта схема выглядит на чипе.** Металлический контакт -9 В – сверху, сам транзистор – посередине, а кремниевый выход – внизу.



# Транзисторы в микросхемах (Intel 8008)

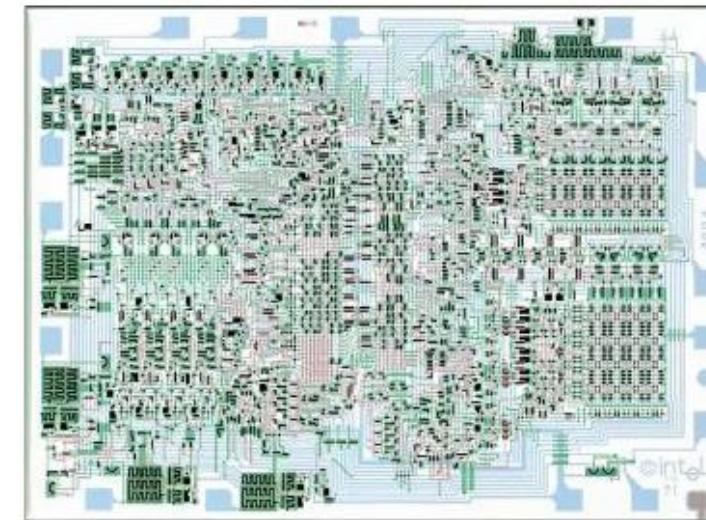
- **Процессоры intel 4004, и 8008 – использовали транзисторы РМОС обогащенного типа (enhancement-mode) с поликремниевым затвором, которые применялись сравнительно недолго. Этим обусловлено уникальное положение этих чипов в аспекте развития технологий полупроводникового производства.**
- **Процессор 8008 (как и современные процессоры) использует транзисторы MOS.** Эти транзисторы прошли долгий путь, прежде чем были окончательно приняты в отрасли, так как были медленнее и менее надежны, чем биполярные транзисторы, использовавшиеся в большинстве компьютеров в 1960-е годы.
- **К концу 1960-х транзисторы MOS в интегральных микросхемах стали использоваться более широко; стандартным вариантом этой технологии были транзисторы рMOS с металлическим затвором.** Металл затворов транзисторов также использовался для электрического соединения компонентов чипа. Чипы, по существу, включали в себя два функциональных слоя: сам кремний и металлическую проводку сверху.

# Транзисторы в процессорах

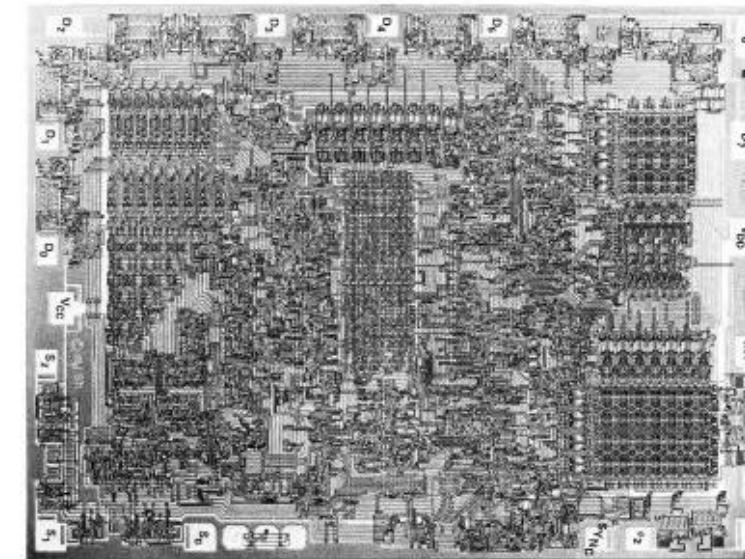


**TMX 1795**  
3078 transistors

1mm



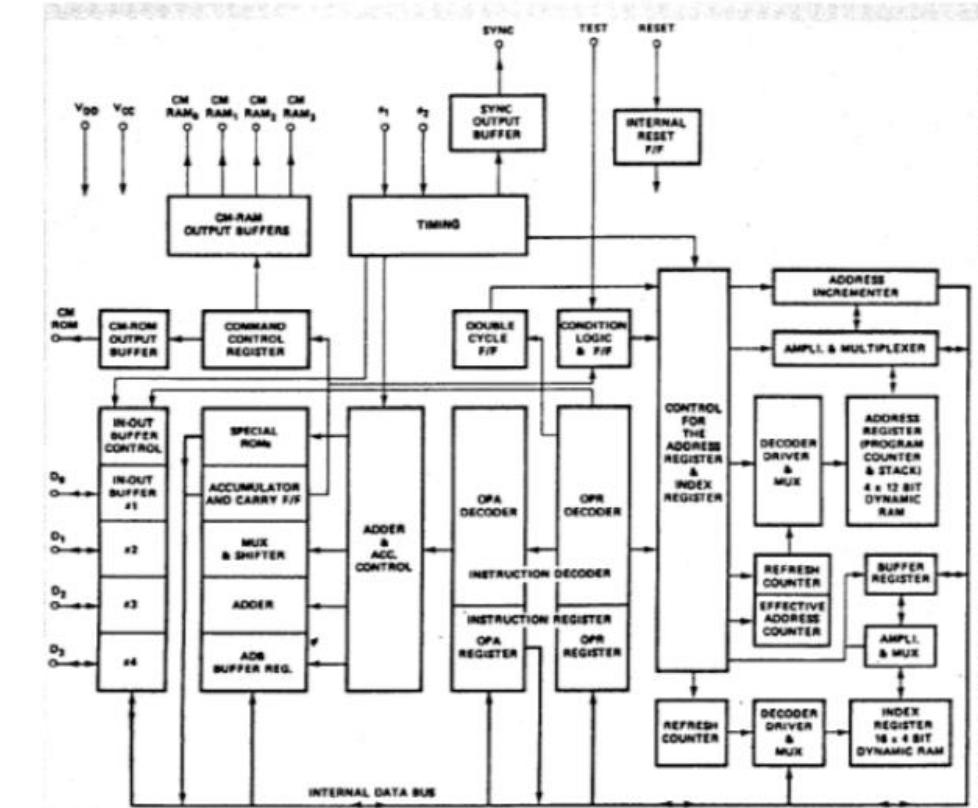
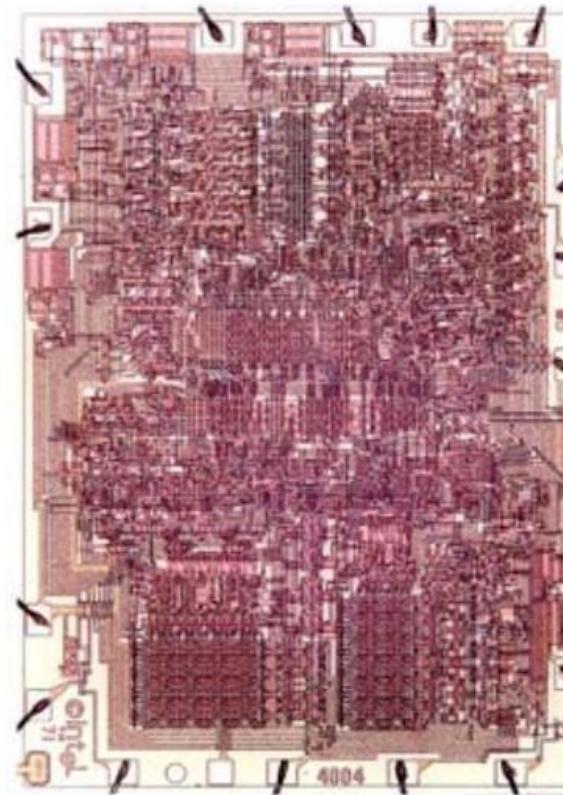
**4004**  
2300 transistors



**8008**  
3098 transistors

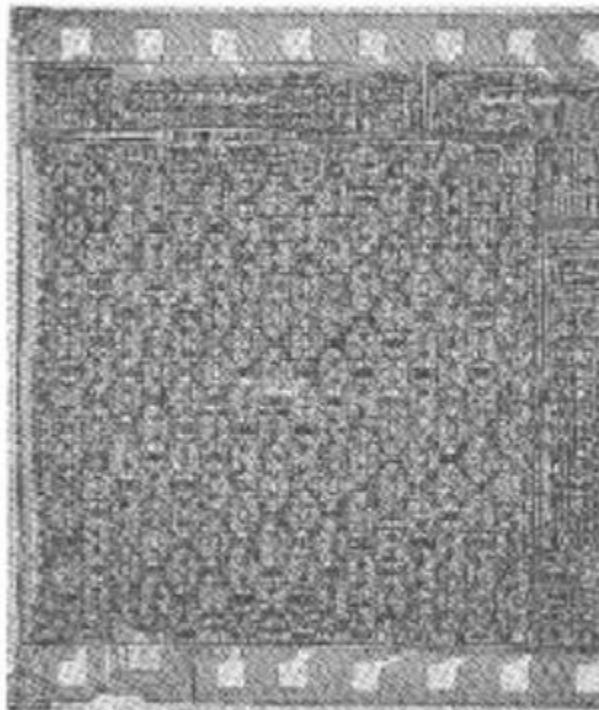
# Транзисторы в процессорах

## Intel 4004



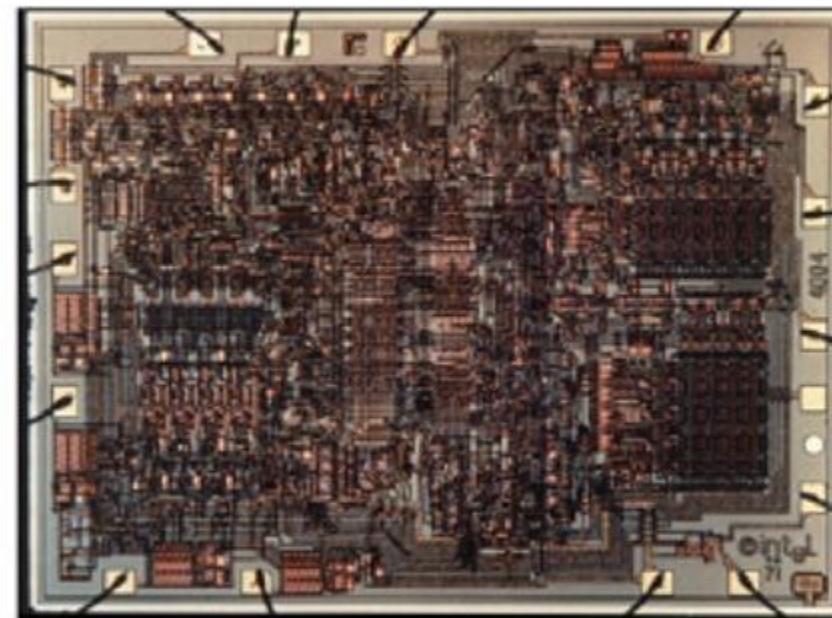
- **Transistors 2300**
- **Process 10 μ**
- **Area 83 mm<sup>2</sup>**
- **Clock 100 kHz**

# Примеры МОП-интегральных схем



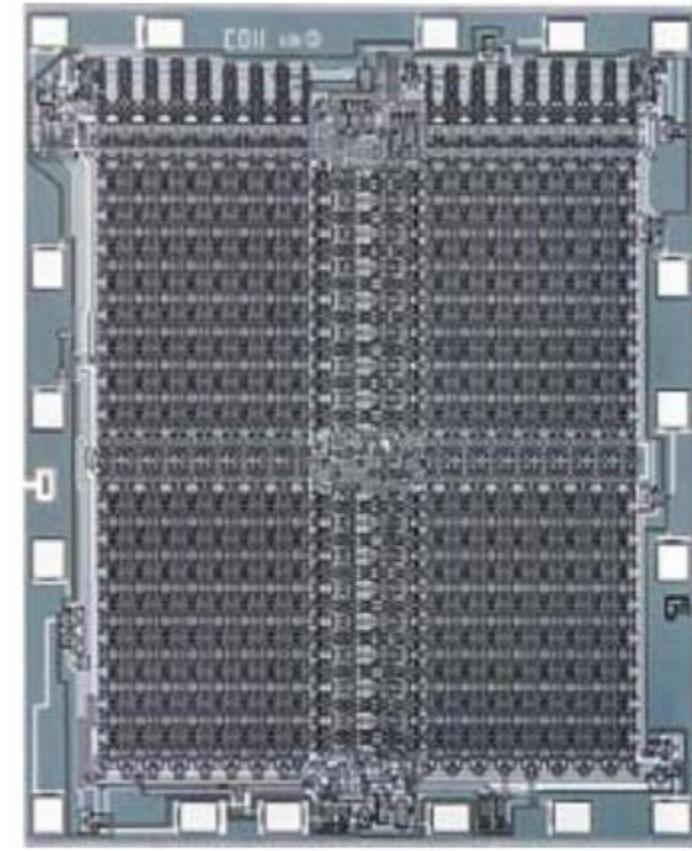
Intel 1101

256-bit SRAM



Intel 4004

4-bit  $\mu$ Proc



Intel 1103

1K bit DRAM

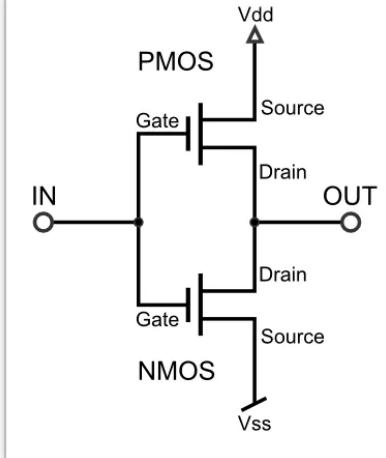
**В процессорах 1970-х годов обычно использовались только nMOS-транзисторы.** Они недорогие, но потребляют энергию в режиме ожидания

# CMOS (КМОП)

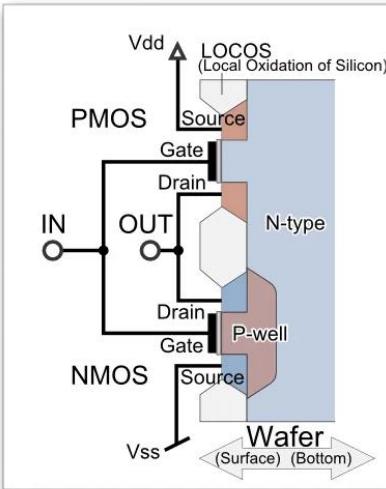
- **СМОС-технология** – одна из нескольких технологий разработки и построения схем электроники.
- Само название технологии является аббревиатурой английского выражения – complementary metal-oxide-semiconductor.
- В русской транскрипции – **КМОП**, или **комплементарная логика с транзисторами на металл-оксид-полупроводнике**.
- Работа транзисторов МОП-структуры основана на полевом эффекте, открытом ещё в 20-х годах девятнадцатого века.
- В микросхемах, построенных по СМОС-технологии, применяются пары полевых транзисторов с одинаковыми параметрами, но с разными проводимостями изолированных затворов.
- Если у одного транзистора затвор р-типа, то у другого он п-типа; следствием такой структуры является более высокое быстродействие.

CMOS Invertor

Model chart

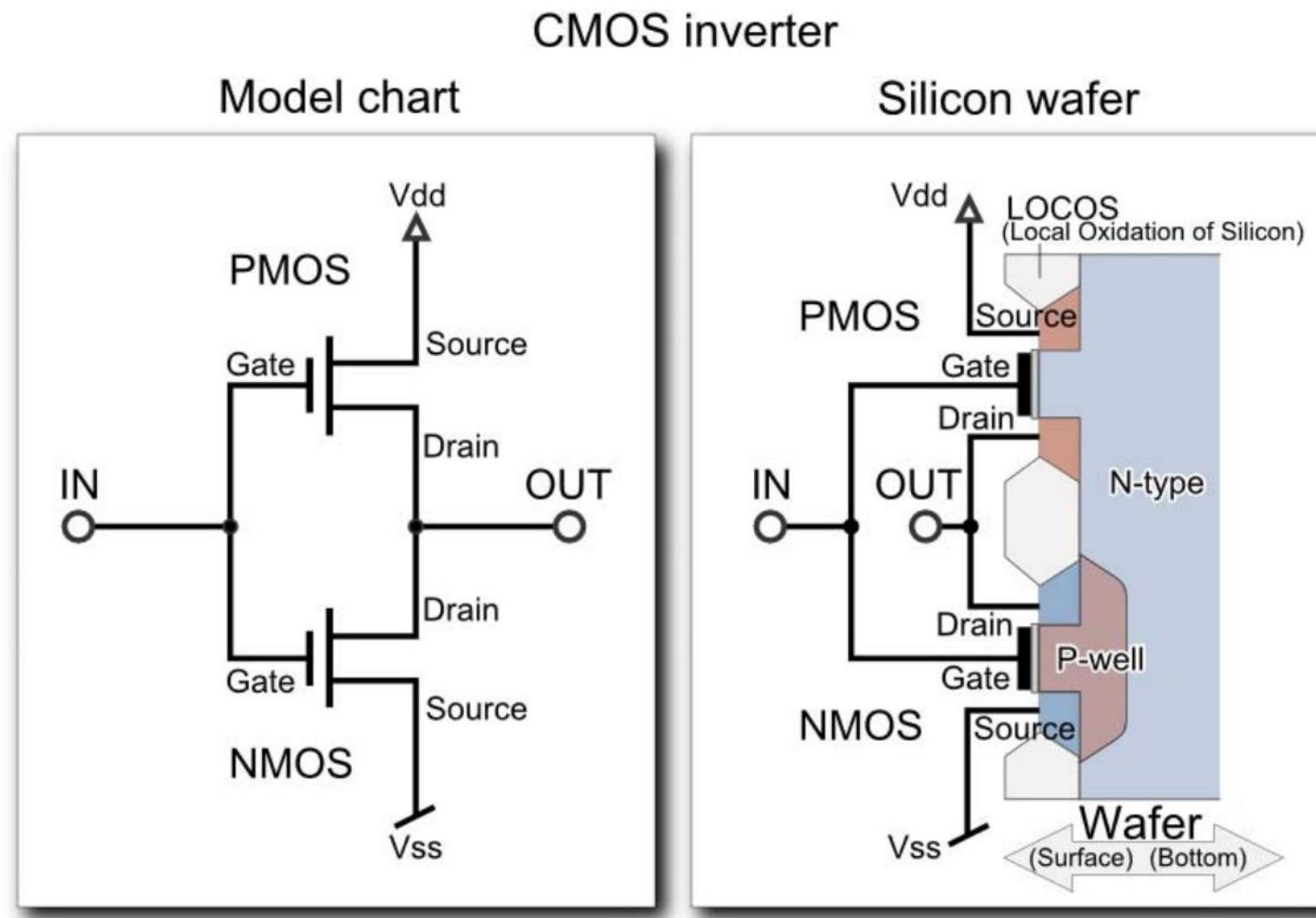


Silicon wafer



# CMOS (КМОП)

CMOS представляет собой комбинацию двух MOSFET транзисторов.



CMOS строится на паре комплементарных (дополняющих) МОП-транзисторов:

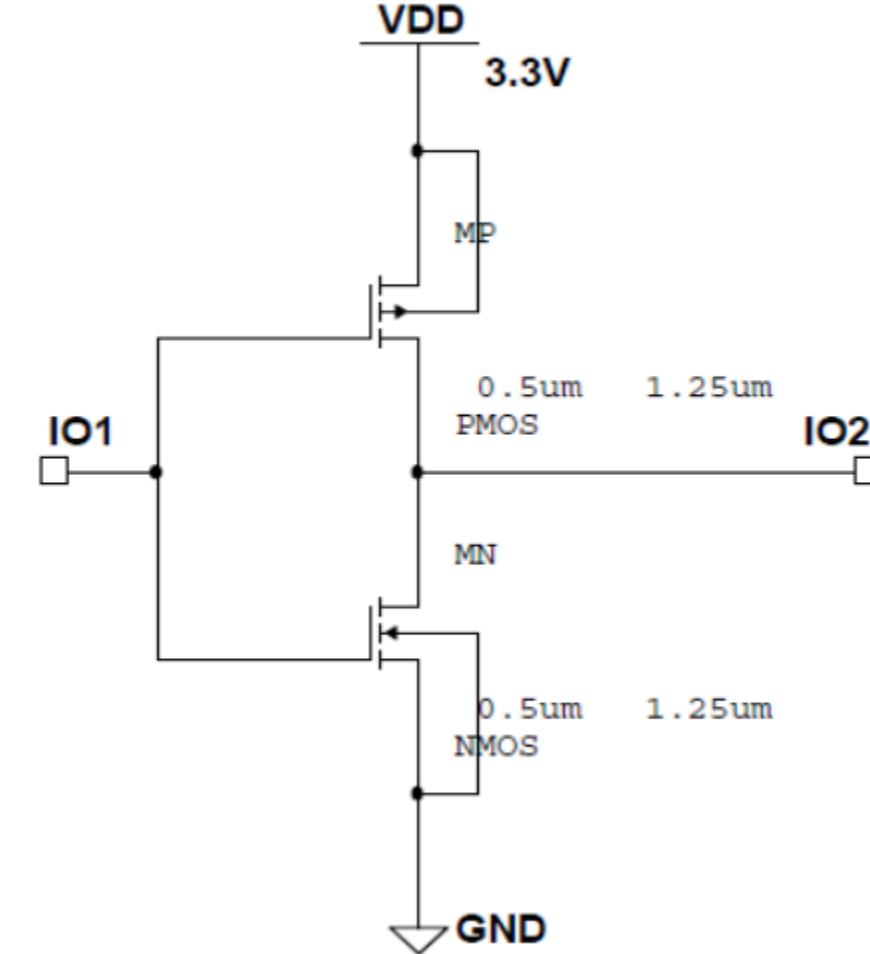
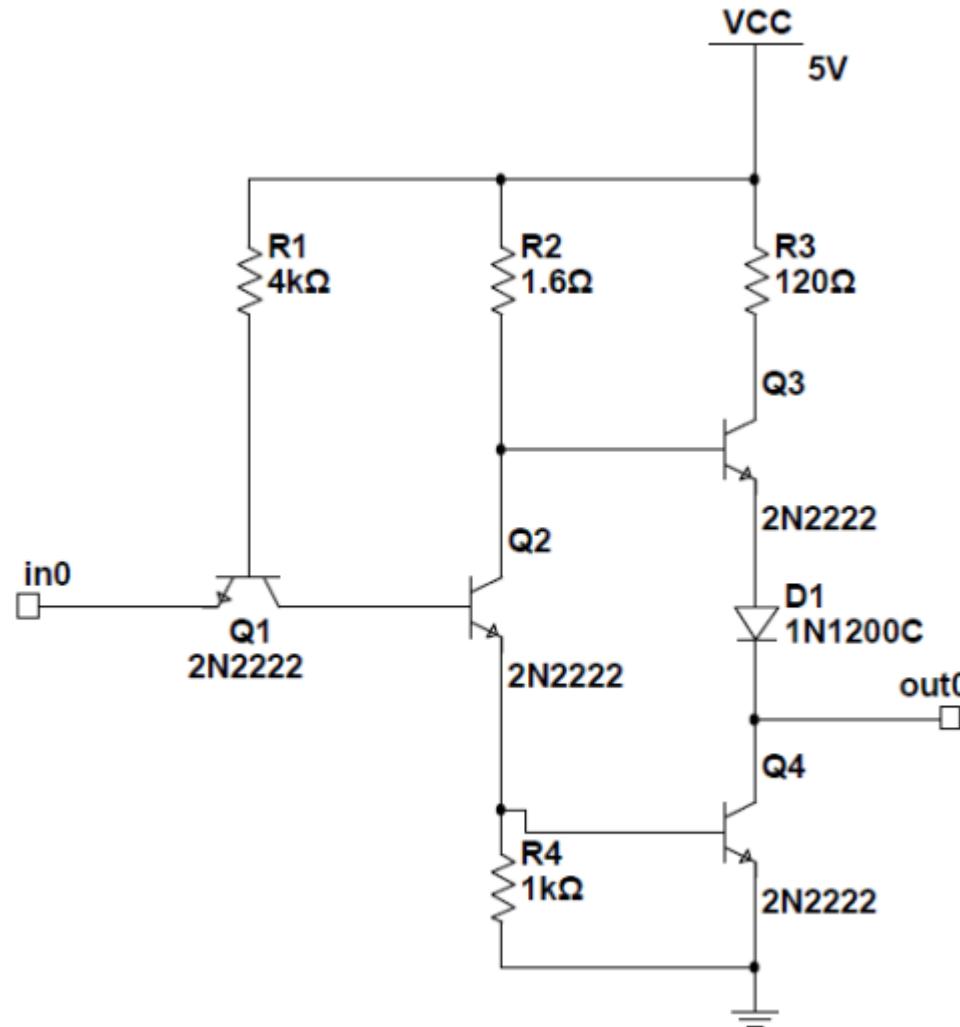
**PMOS (р-канальный):**

- Канал из **р-типа** (дырочная проводимость),
- Включается **низким напряжением** на затворе (0 В),
- Подключён к питанию ( $V_{DD}$ ).

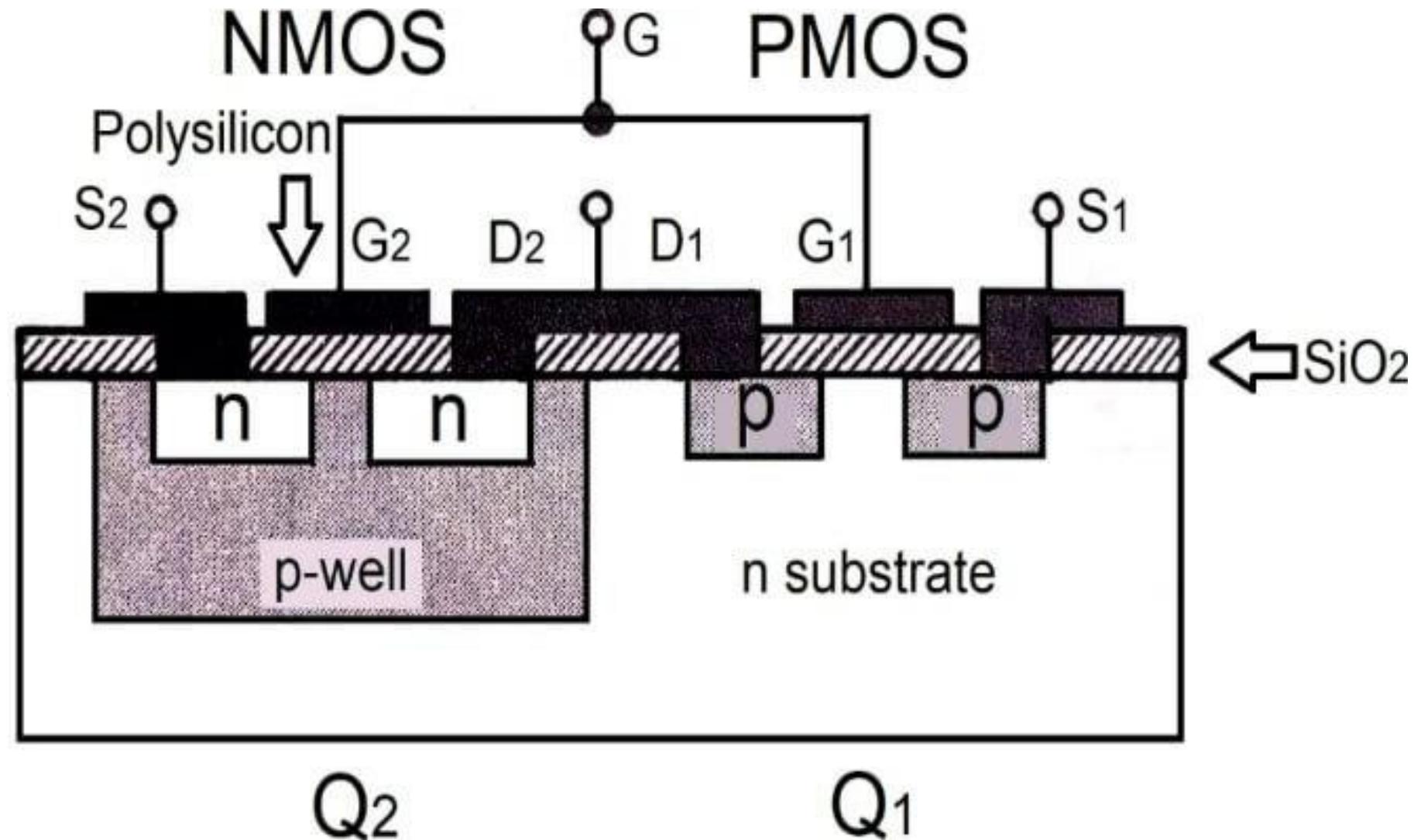
**NMOS (n-канальный):**

- Канал из **n-типа** (электронная проводимость),
- Включается **высоким напряжением** на затворе ( $V_{DD}$ ),
- Подключён к земле (**GND**).

# Пример аналогичных схем ТТЛ и КМОП: инвертор



# CMOS (КМОП)



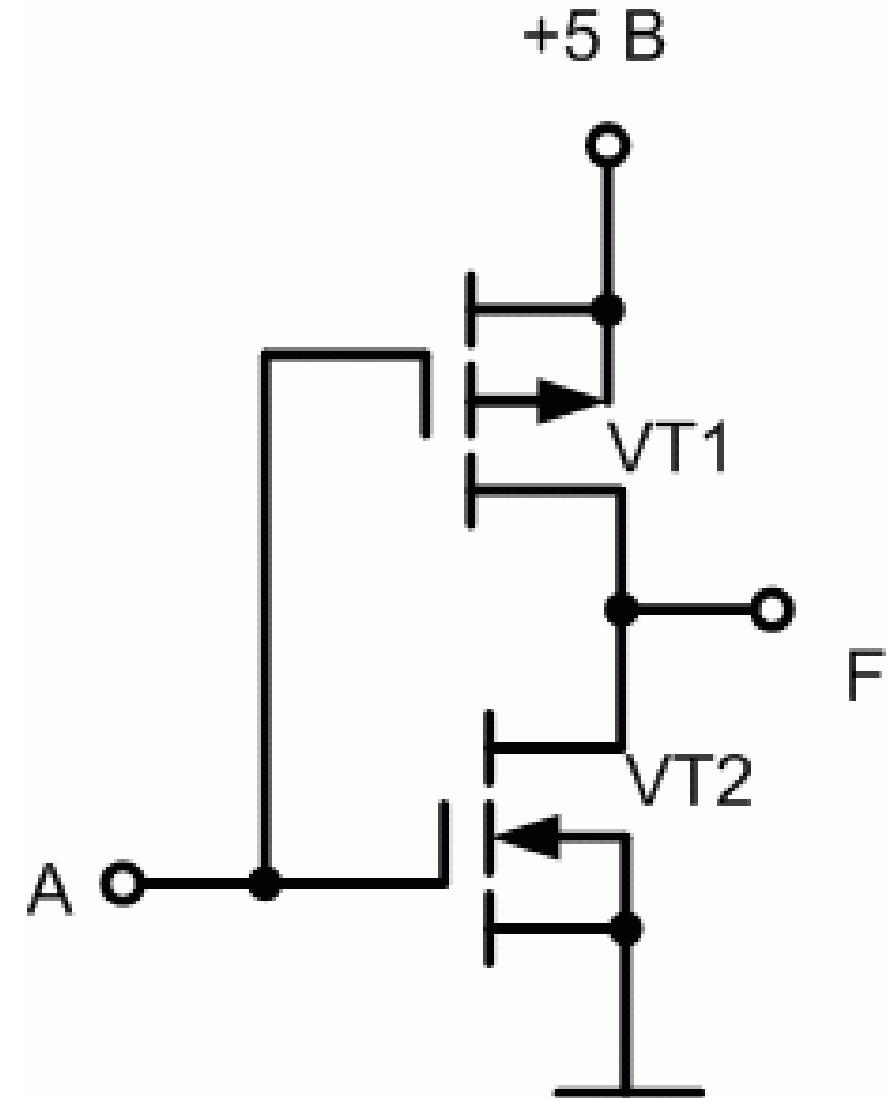
Поперечное сечение CMOS

# CMOS (КМОП)

- **СМОС (КМОП – комплементарная структура металл – оксид – полупроводник, от английского complementary metal–oxide–semiconductor)** – это набор полупроводниковых технологий для создания интегральных микросхем, основанный на использовании комплементарных пар полевых транзисторов п- и р-типа с изолированным затвором (обычно диоксид кремния).
- **СМОС-технология обеспечивает очень низкое энергопотребление в статическом режиме и высокое быстродействие**, что делает её доминирующей в производстве современных цифровых микросхем, включая микропроцессоры, память и датчики изображений в цифровых камерах.
- **СМОС – основа современной цифровой электроники.**
- Эта технология **используется в 95% всех микросхем**, от смартфонов до суперкомпьютеров. Разберём её детально: от физики до современных трендов.

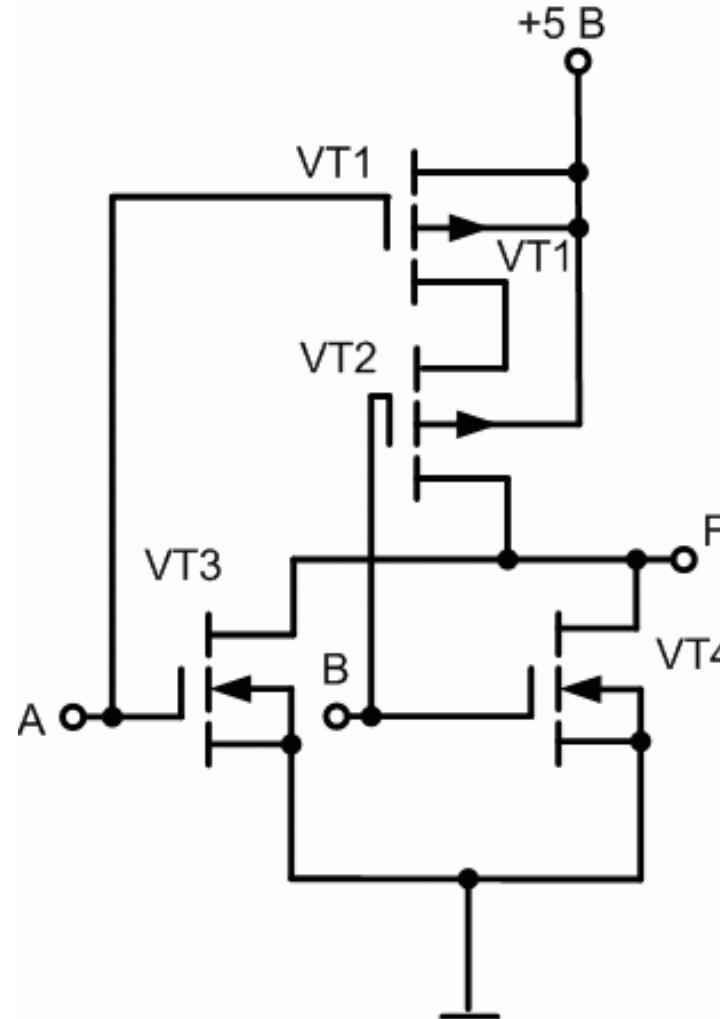
# CMOS (КМОП)

- Схемы КМОП-технологии (К – от термина комплементарный, что означает комплексный) базируются на полевых (МОП) транзисторах с индуцированным каналом и р-, и п-типа.**
- Базовым элементом данной технологии является схема инвертора (Логический элемент НЕ).
- Принципиальное отличие КМОП-схем от nMOP-технологии заключается в отсутствии в схеме активных сопротивлений.**
- К каждому входу схемы подключена пара транзисторов с различным типом канала. Транзисторы с каналом р-типа подключены подложкой к источнику питания, поэтому образование канала в них будет происходить при достаточной большой разности потенциалов между подложкой и затвором, причем потенциал на затворе должен быть отрицательным относительно подложки.
- Такое состояние обеспечивается подачей на затвор потенциала земли (т.е. логического 0). Транзисторы с каналом п-типа подключены подложкой к земле, поэтому образование канала в них будет происходить при подаче на затвор потенциала источника питания (т.е. логической 1). Одновременная подача на такие пары транзисторов с разным типом каналов логического нуля или логической единицы приводит к тому, что один транзистор пары обязательно будет открыт, а другой закрыт. Таким образом, создаются условия к подключению выхода либо к источнику питания, либо к земле.
- Так, в простейшем случае, для схемы инвертора (Логический элемент НЕ) при  $A=0$  транзистора VT1 будет открыт, а VT2 закрыт. Следовательно, выход схемы F будет подключен через канал VT1 к источнику питания, что соответствует состоянию логической единицы:  $F=1$ . При  $A=1$  транзистор VT1 будет закрыт (на затворе и подложке одинаковые потенциалы), а VT2 открыт. Следовательно, выход схемы F будет подключен через канал транзистора VT2 к земле. Это соответствует состоянию логического нуля:  $F=0$ .



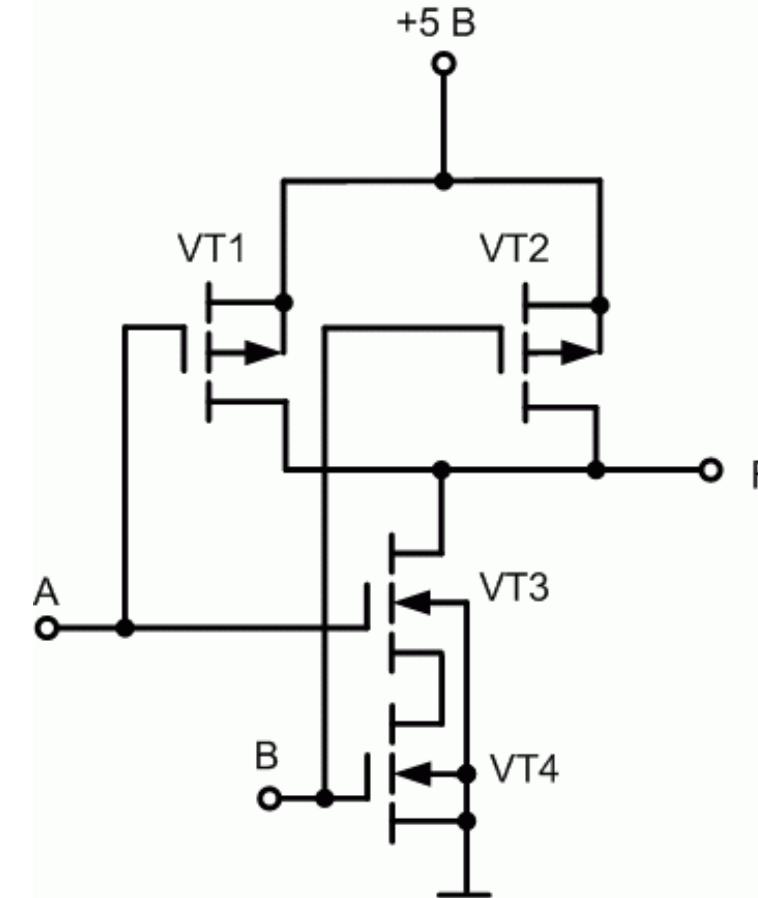
Логический элемент НЕ КМОП-технологии

# CMOS (КМОП) – Элементы ИЛИ-НЕ и И-НЕ



Логическое сложение осуществляется за счет последовательного соединения р-каналов транзисторов VT1 и VT2. При подаче хотя бы одной единицы единого канала у данных транзисторов не образуется. В то же время благодаря параллельному соединению VT3 и VT4 осуществляется открытие соответствующего транзистора в нижней части схемы, обеспечивающее подключение выхода F к земле. Получается  $F=0$  при подаче хотя бы одной логической 1 – это правило ИЛИ-НЕ.

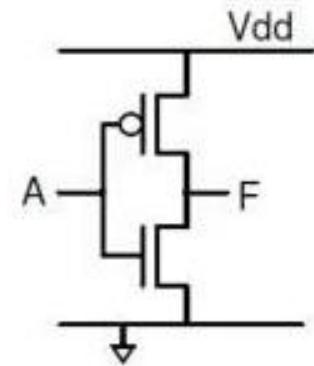
Логический элемент **ИЛИ-НЕ** КМОП-технологии



Логический элемент **И-НЕ** КМОП-технологии

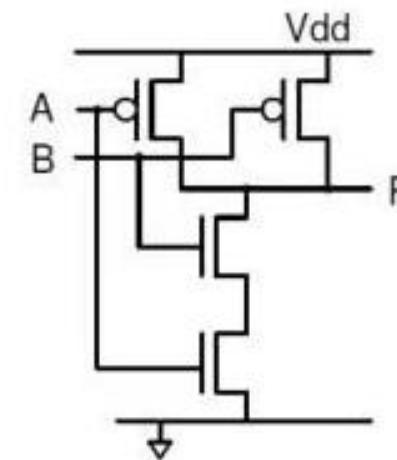
Функция И-НЕ осуществляется за счет параллельного соединения VT1 и VT2 в верхней части схемы и последовательного соединения VT3 и VT4 в нижней части. При подаче хотя бы на один вход нуля единый канал на VT3 и VT4 не образуется, выход будет отключен от земли. В то же время хотя бы один транзистор в верхней части схемы (на затвор которого подан логический ноль) будет обеспечивать подключение выхода F к источнику питания:  $F=1$  при подаче хотя одного нуля – правило И-НЕ.

# CMOS (КМОП) – Элементы



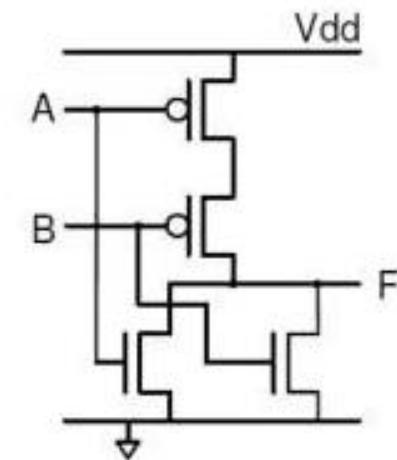
CMOS INVERTER

| A | F |
|---|---|
| L | H |
| H | L |



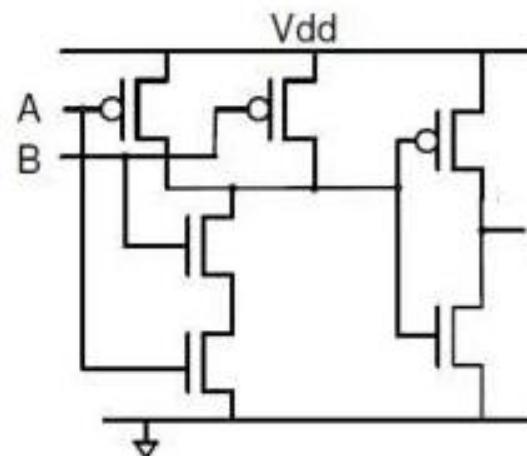
CMOS NAND

| A | B | F |
|---|---|---|
| L | L | H |
| L | H | H |
| H | L | H |
| H | H | L |



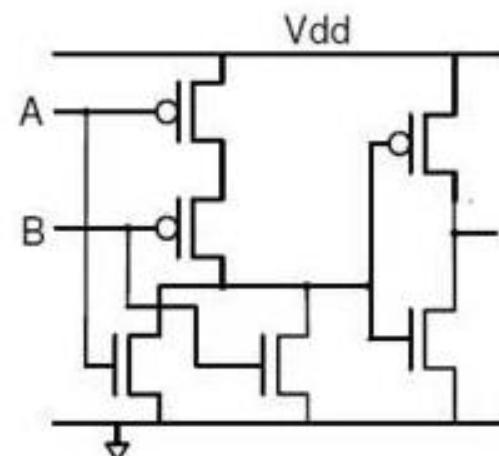
CMOS NOR

| A | B | F |
|---|---|---|
| L | L | H |
| L | H | L |
| H | L | L |
| H | H | L |



CMOS AND

| A | B | F |
|---|---|---|
| L | L | L |
| L | H | L |
| H | L | L |
| H | H | H |



CMOS OR

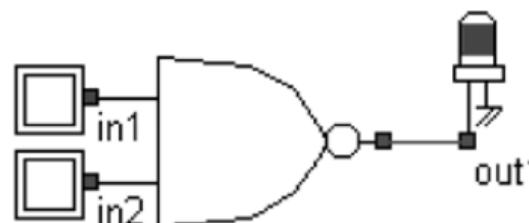
| A | B | F |
|---|---|---|
| L | L | L |
| L | H | H |
| H | L | H |
| H | H | H |

# CMOS (КМОП) – Элемент И-НЕ (Nand)

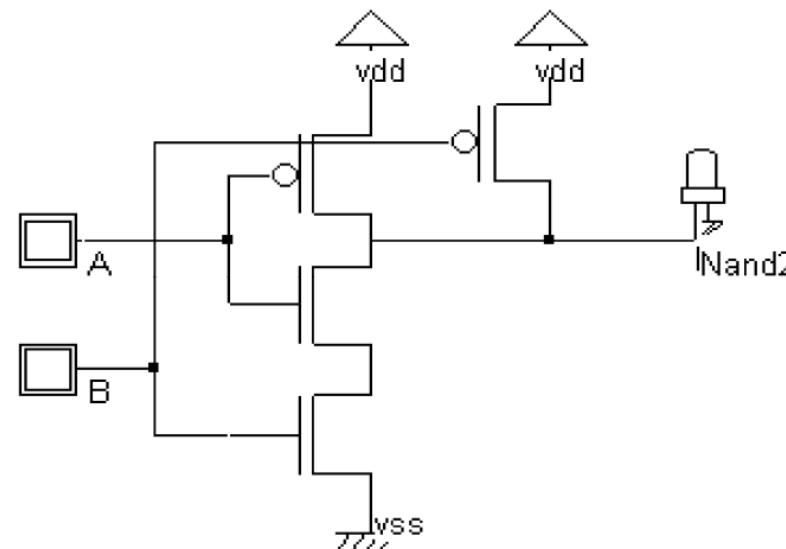


| in1 | in2 | Out |
|-----|-----|-----|
| 0   | 0   | 1   |
| 0   | 1   | 1   |
| 1   | 0   | 1   |
| 1   | 1   | 0   |

Таблица истинности



Обозначения элемента  
И-НЕ (NAND)

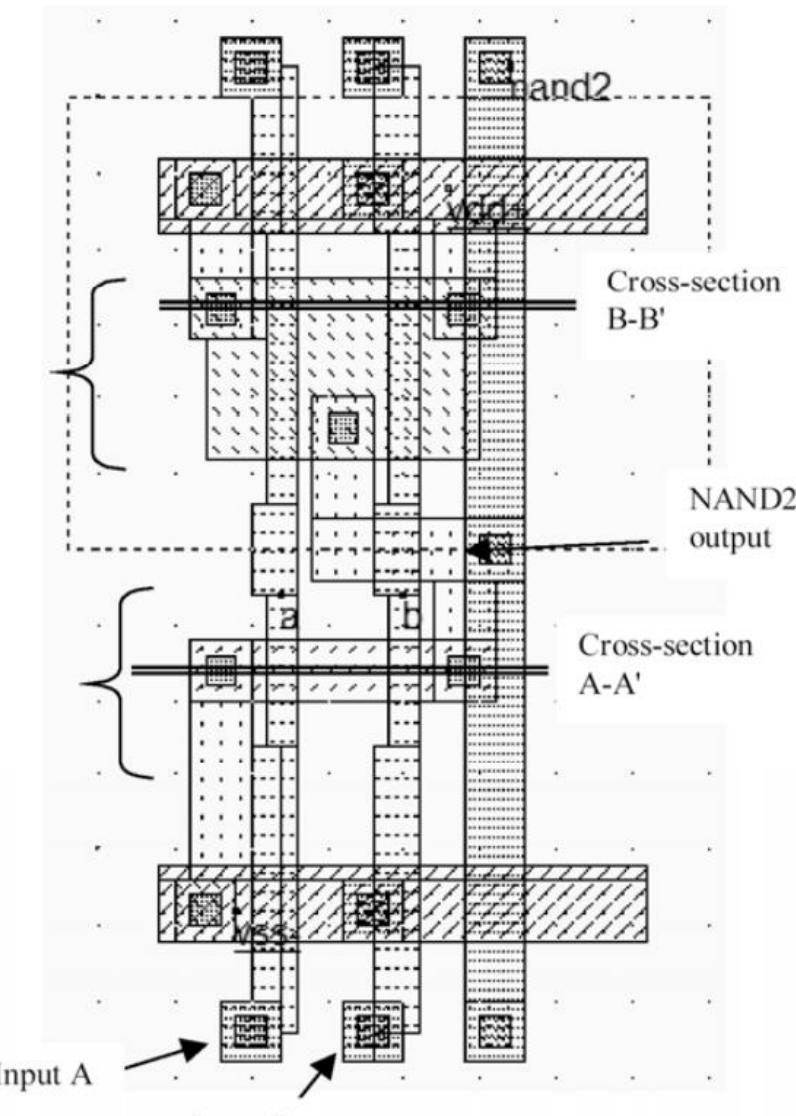


Электрическая схема ячейки NAND

**В CMOS проектировании элемент NAND состоит из двух nMOS транзисторов**, соединенных последовательно, которые связаны с двумя pMOS в параллель. Последовательные транзисторы nMOS связывают выход с землей для единственной комбинации на входе схемы A=1, B=1. Для трех других комбинаций подсхема из nMOS транзисторов не работает, но хотя бы один из pMOS транзисторов связывает выход с питанием VDD.

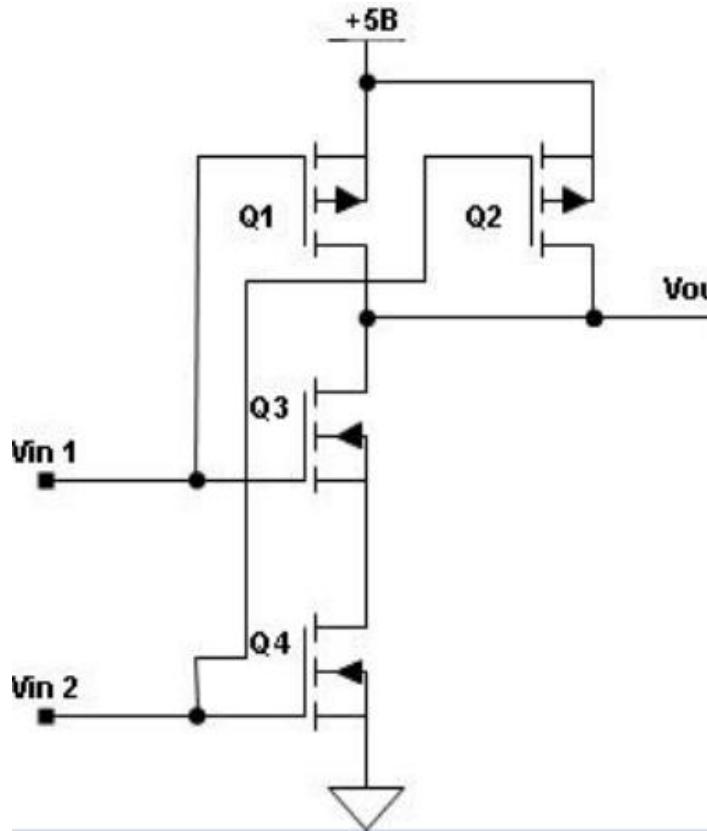
Pmos devices

Nmos devices

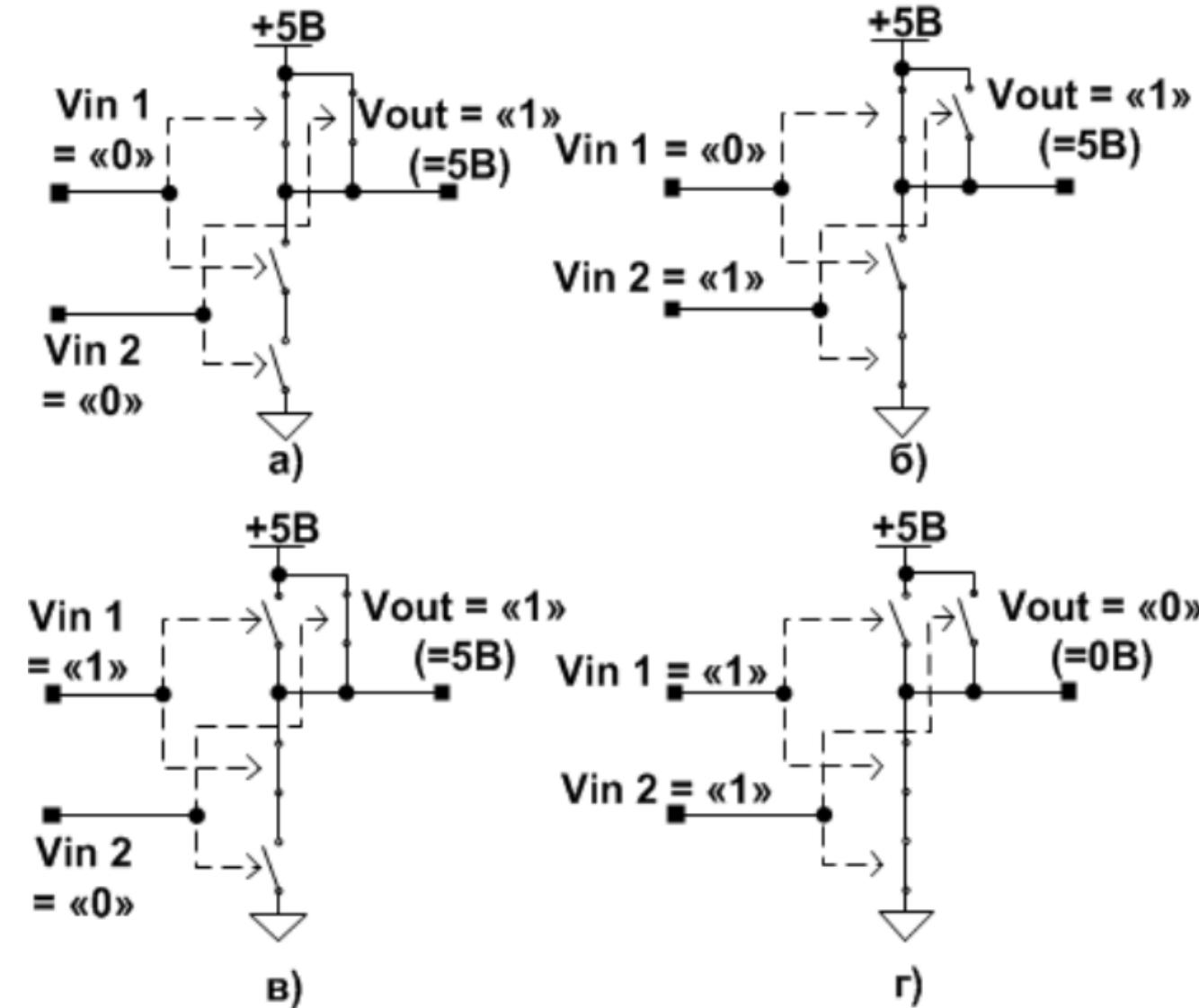


Топология ячейки NAND

# CMOS (КМОП) – Элемент И-НЕ (Nand)

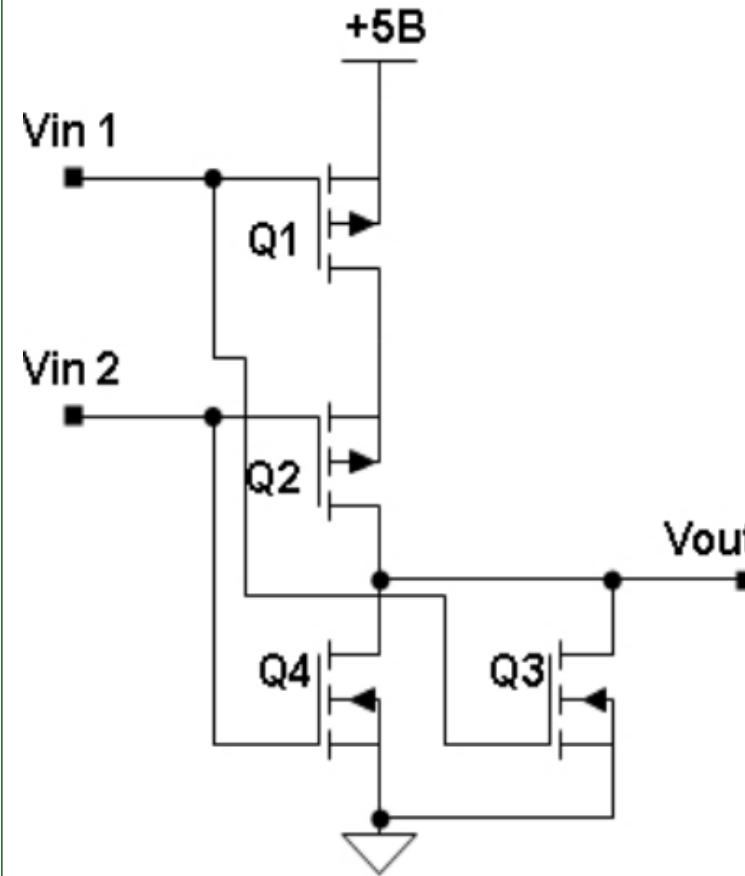


Эквивалентная функциональная схема

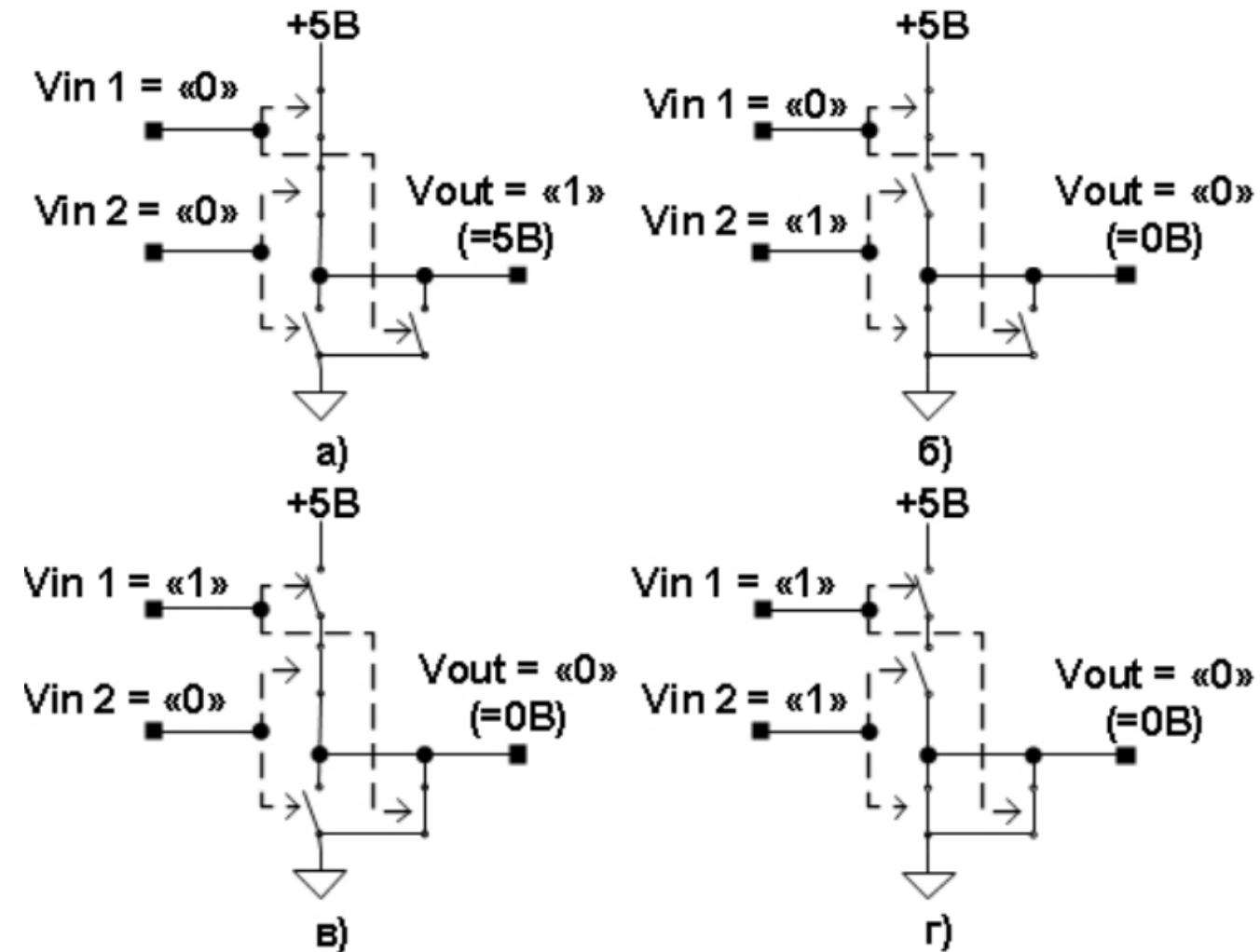


Принципиальная электрическая схема  
и условное обозначение

# CMOS (КМОП) – Элемент ИЛИ-НЕ (NOR)



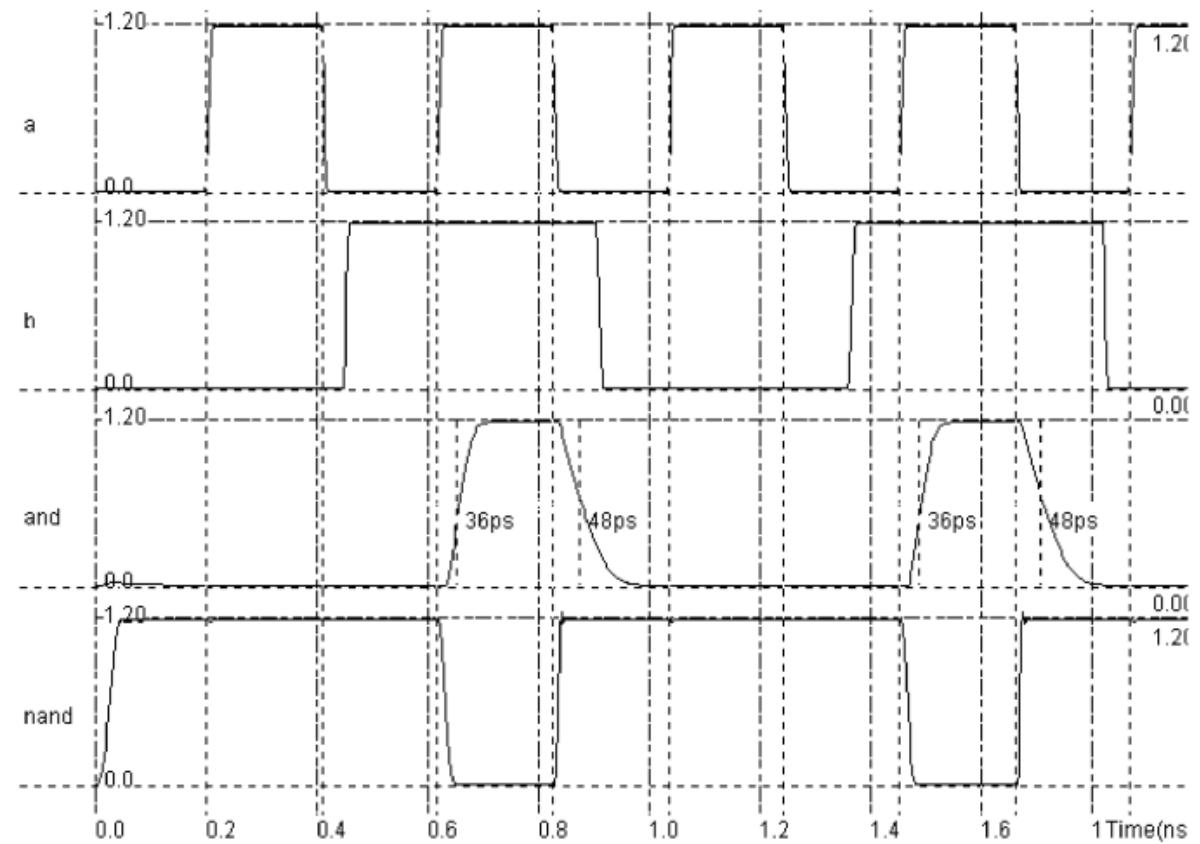
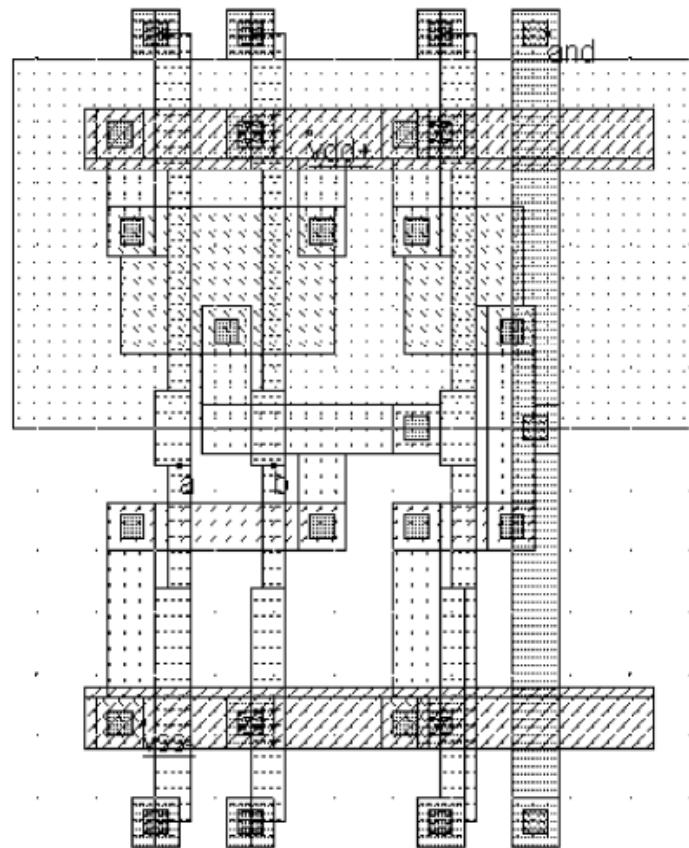
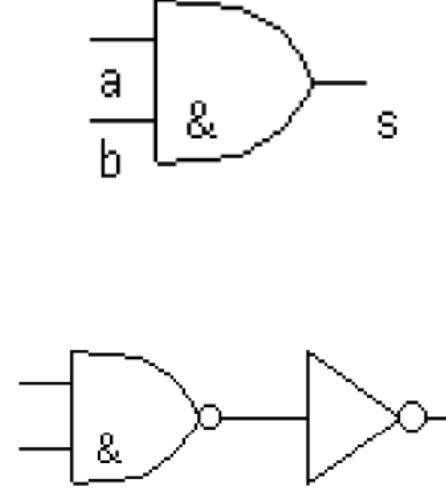
Эквивалентная функциональная схема



Принципиальная электрическая схема  
и условное обозначение

# CMOS (КМОП) – Элемент И (AND)

**Элемент И (AND) формируется из NAND2 и инвертора.** Для CMOS технологии элементы с отрицанием (NAND, NOR, INV) выполняют операции быстрее и проще чем элементы без отрицания (AND, OR, Buffer). Задержка ячейки значительно выше, чем для отдельного NAND2 элемента из-за задержки каскада инвертора.

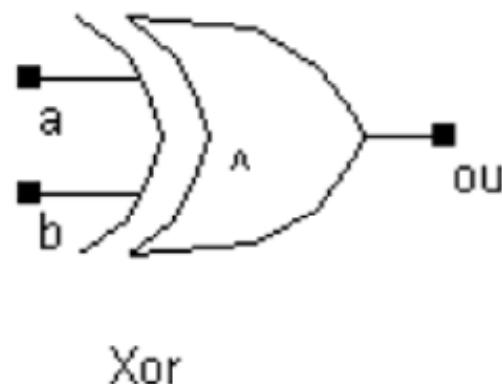


Топология и результаты моделирования ячейки AND

# CMOS (КМОП) – Элемент XOR

**Элемент XOR.** Существует множество вариантов реализации функции XOR в CMOS технологии. Один из вариантов, со средней эффективностью по проектированию, но перспективный для CMOS состоит в построении логической схемы XOR по Булевым уравнениям.

Предлагаемый вариант состоит из элемента «transmission-gate» для реализации оператора XOR. Таблица истинности для элемента XOR может быть прочтена в следующем виде: если  $B=0$ , то  $OUT=A$ , если  $B=1$ , то  $OUT = \text{Inv}(A)$ . Заметим, что nMOS и pMOS транзисторы, расположенные в середине используются как «ключи передачи сигнала».



Обозначения элемента  
XOR

**XOR 2 inputs**

| A | B | OUT |
|---|---|-----|
| 0 | 0 | 0   |
| 0 | 1 | 1   |
| 1 | 0 | 1   |
| 1 | 1 | 0   |

Таблица истинности

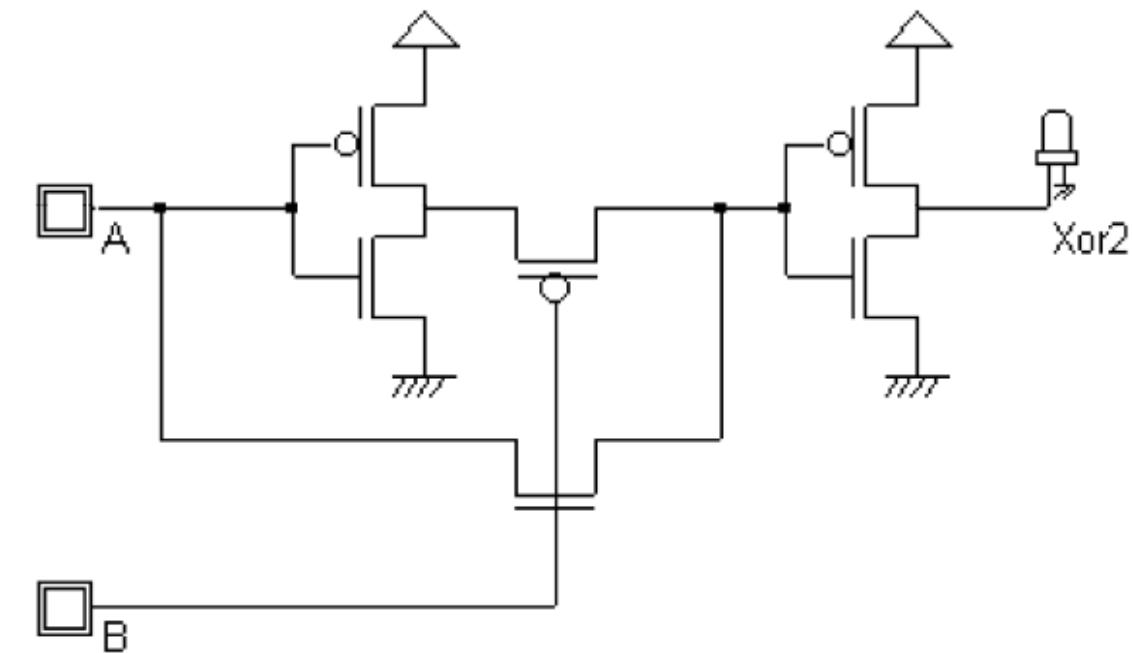
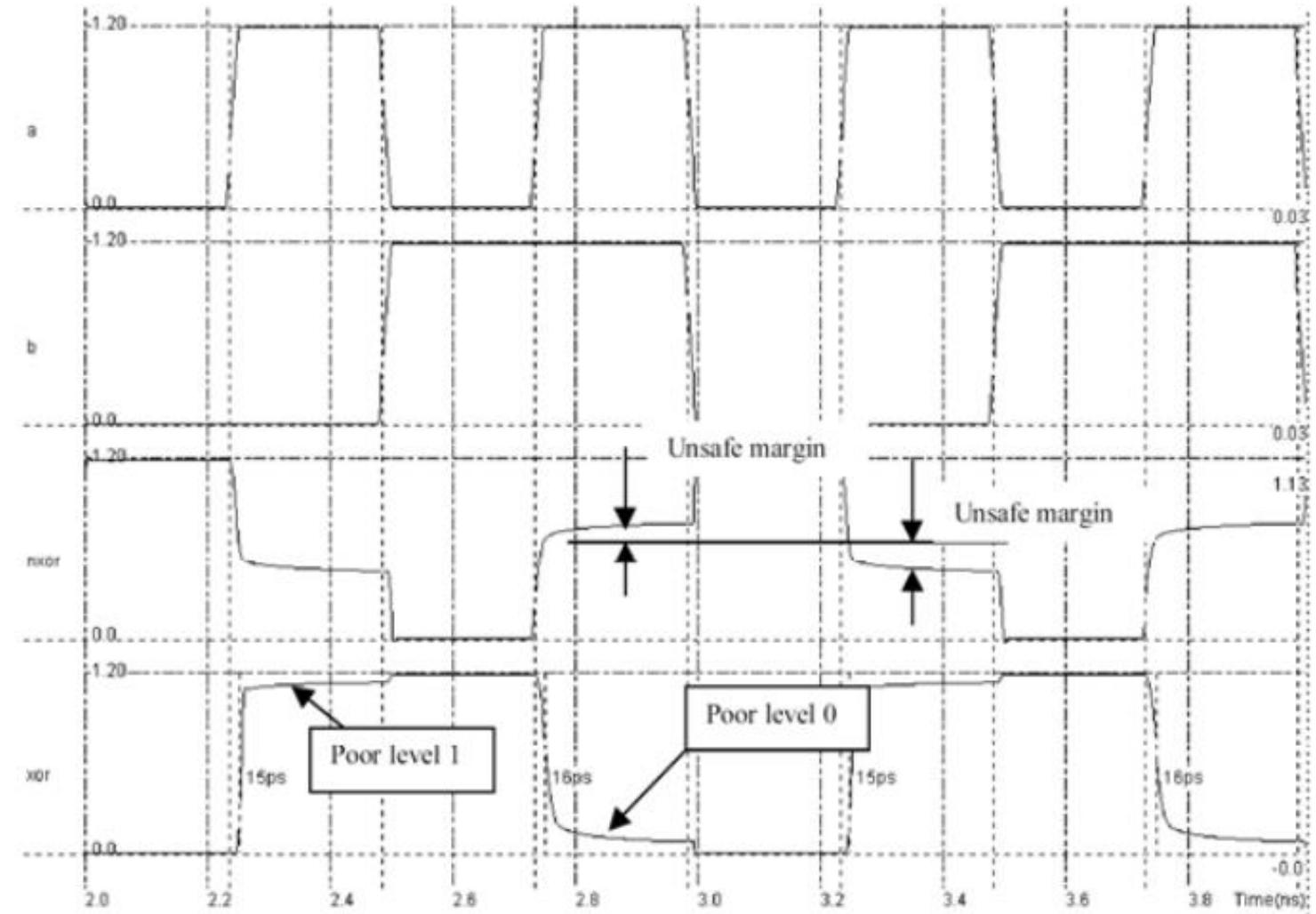
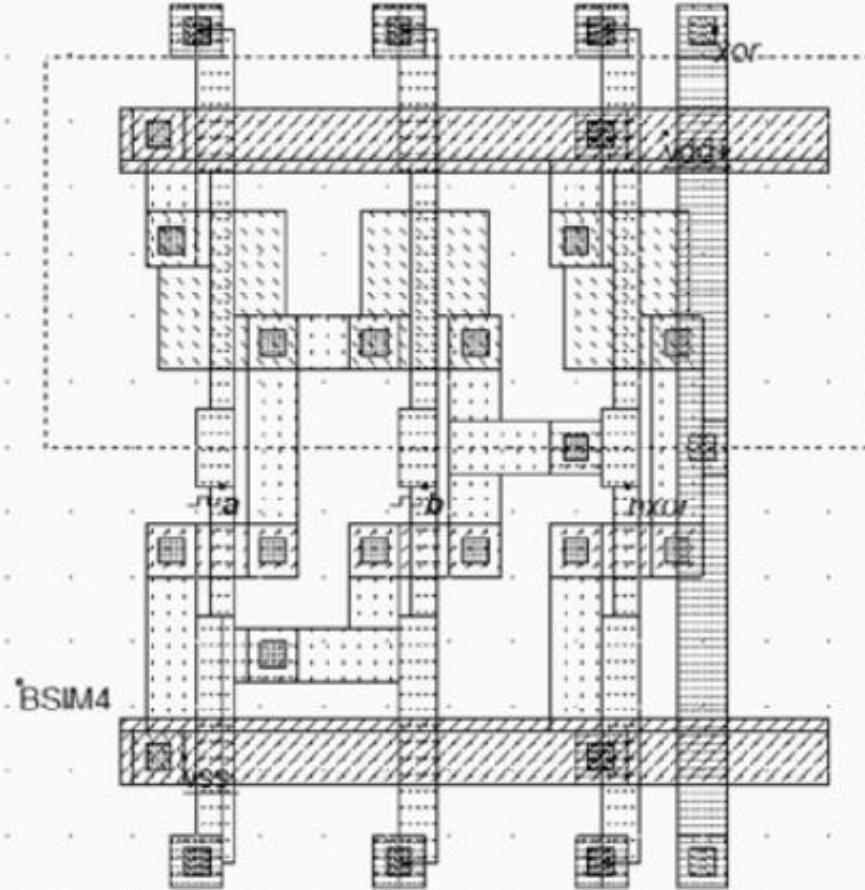


Схема ячейки XOR

# CMOS (КМОП) – Элемент XOR



Топология и результаты моделирования ячейки XOR

# CMOS (КМОП) – функция 2И-НЕ

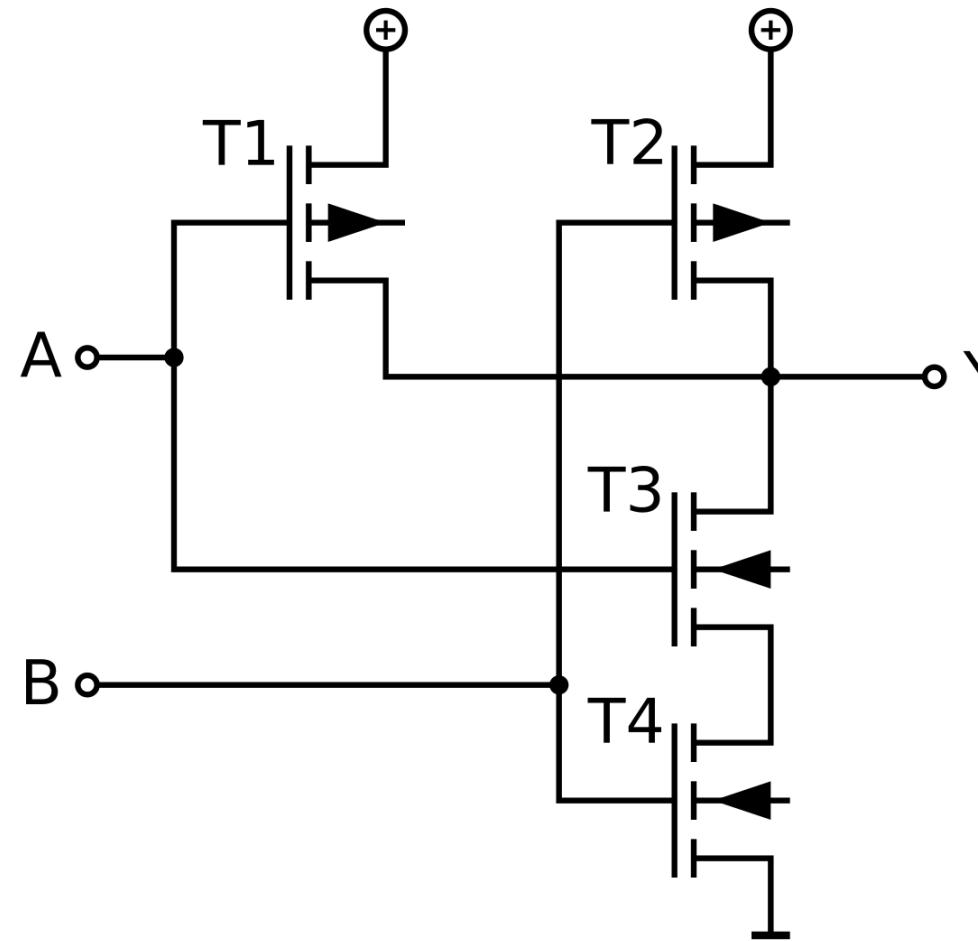
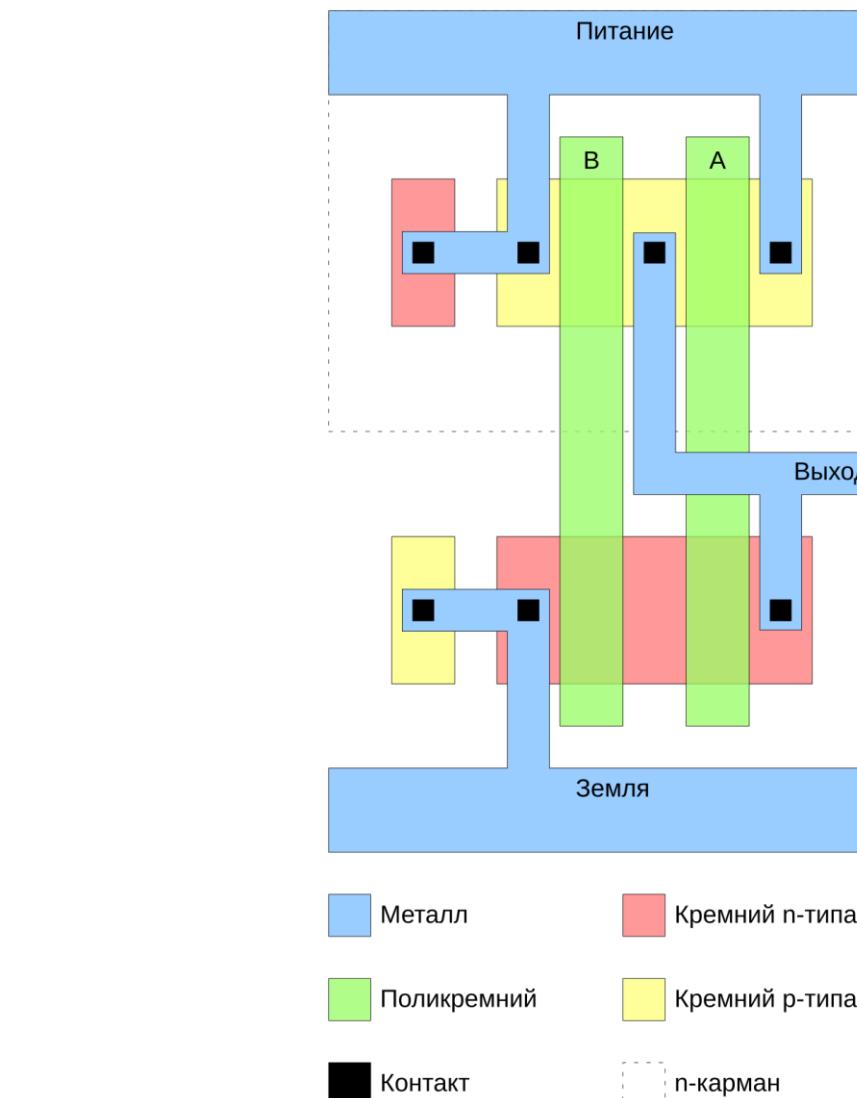
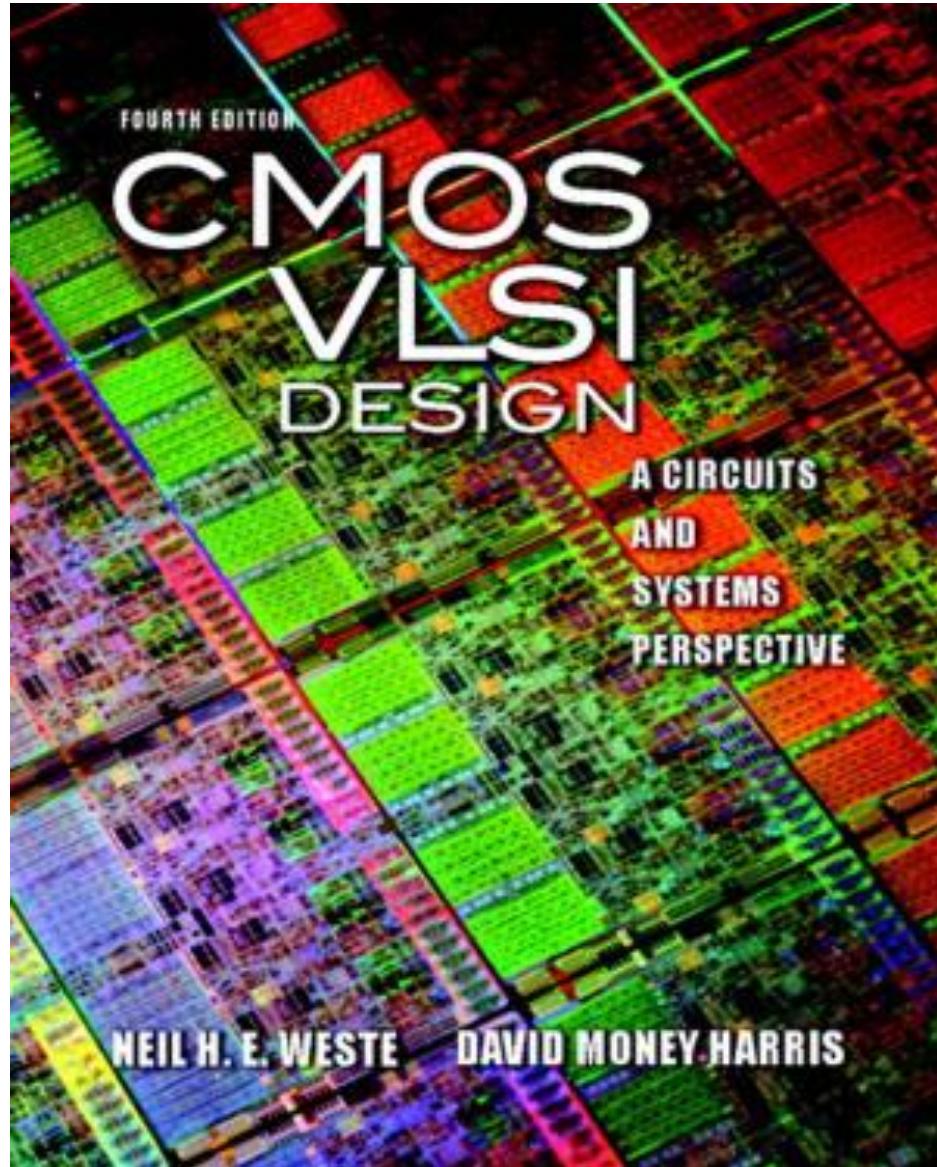


Схема логического элемента, выполняющего логическую функцию 2И-НЕ



Топология логического элемента 2И-НЕ в КМОП интегральной схеме

# CMOS (КМОП)



Weste Neil H.E., Harris David. **CMOS VLSI design** 4th edition. – Pearson Education, 2011.  
– 839 p. – ISBN: 0321547748.

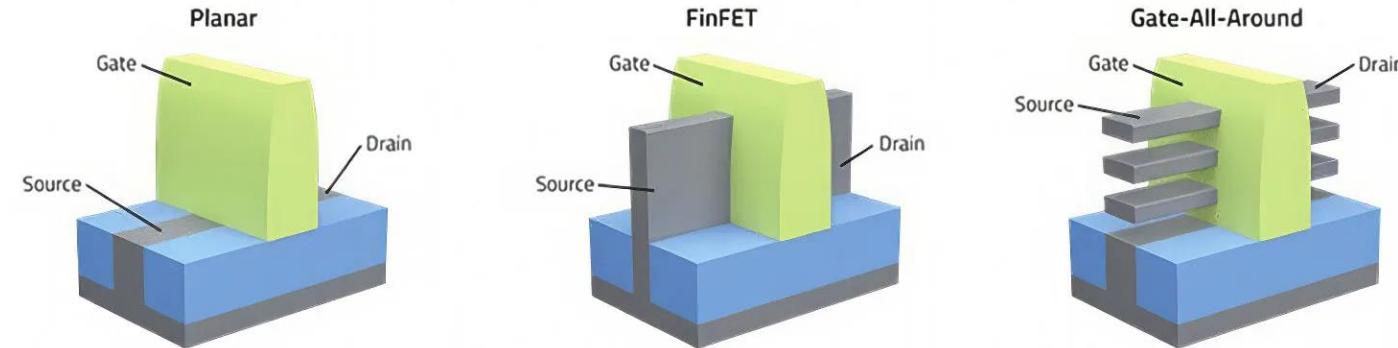
<https://www.twirpx.com/file/2540943/>

For both introductory and advanced courses in VLSI design, this authoritative, comprehensive textbook is highly accessible to beginners, yet offers unparalleled breadth and depth for more experienced readers. The Fourth Edition of "CMOS VLSI Design: A Circuits and Systems perspective" presents broad and in-depth coverage of the entire field of modern CMOS VLSI Design. The authors draw upon extensive industry and classroom experience to introduce today's most advanced and effective chip design practices. They present extensively updated coverage of every key element of VLSI design, and illuminate the latest design challenges with 65 nm process examples. This book contains unsurpassed circuit-level coverage, as well as a rich set of problems and worked examples that provide deep practical insight to readers at all levels.

Этот авторитетный, всеобъемлющий учебник, предназначенный как для вводных, так и для продвинутых курсов по проектированию СБИС, в высшей степени доступен для начинающих, но предлагает непревзойденную широту и глубину для более опытных читателей. В четвертом издании "Проектирование КМОП-СБИС: взгляд с точки зрения схем и систем" представлен широкий и углубленный обзор всей области современного проектирования КМОП-СБИС. Авторы опираются на обширный промышленный и аудиторский опыт, чтобы представить самые передовые и эффективные методы проектирования микросхем на сегодняшний день. В них подробно освещаются все ключевые элементы проектирования СБИС и освещаются новейшие проблемы проектирования с помощью примеров 65-нм технологического процесса. Эта книга содержит непревзойденное описание схем, а также богатый набор проблем и отработанных примеров, которые дают глубокое практическое представление читателям всех уровней.

# Эволюция CMOS: от микронов до нанометров

| Поколение                    | Годы   | Техпроцесс   | Напряжение | Особенности   |
|------------------------------|--------|--------------|------------|---|
| <b>Ранний CMOS</b>           | 1970-е | 10–5 мкм     | 5–15 В     | Серии CD4000, K564                                    |
| <b>Стандарт CMOS</b>         | 1980-е | 3–1 мкм      | 5 В        | 74HC, 74HCT   |
| <b>Низковольтный CMOS</b>    | 1990-е | 0.8–0.35 мкм | 3.3–2.5 В  | LVC MOS, переход к КМОП-логике в компьютерах          |
| <b>Глубокое субмикронное</b> | 2000-е | 180–90 нм    | 1.8–1.2 В  | Рост утечек, появление ультранизковольтных стандартов |
| <b>Нанометровая эра</b>      | 2010-е | 45–14 нм     | 1.0–0.7 В  | FinFET(3D-транзисторы), борьба с утечками             |
| <b>Современные нормы</b>     | 2020-е | 7–2 нм       | <0.7 В     | GAA (Gate-All-Around) FET                             |



# Почему CMOS победил другие технологии?

| Технология | Проблемы                                    | Почему CMOS лучше                           |
|------------|---|---|
| TTL        | Высокое энергопотребление, низкая плотность | В $10^5$ раз меньше статической мощности    |
| ECL        | Очень высокое энергопотребление             | Энергоэффективность при сохранении скорости |
| NMOS       | Сквозной ток, высокие потери                | Отсутствие статического тока в CMOS         |
| JFET       | Невозможно масштабировать                   | Подходит для VLSI/ULSI                      |

## Главные преимущества CMOS:

- Энергоэффективность ( $P \propto CV^2f$ ),
- Масштабируемость (от 10 мкм до 2 нм),
- Высокая помехоустойчивость (размах сигнала  $\approx V_{DD}$ ).

# Современные разновидности CMOS

| Тип         | Описание                                 | Где используется                              |
|-------------|--|---|
| LVCMOS      | Низковольтный CMOS (1.8 В, 1.2 В, 0.8 В) | Интерфейсы FPGA, SoC (например, Raspberry Pi) |
| FinFET CMOS | 3D-транзисторы с «плавником» (Fin)       | Процессоры Apple M-series, Intel Core 14–5 нм |
| FD-SOI      | Тонкоплёночный кремний на изоляторе      | IoT-устройства (низкое энергопотребление)     |
| GAA CMOS    | Gate-All-Around (затвор окружает канал)  | Samsung 3GAA (3 нм), Intel 20A (5 нм)         |
| RFSOI       | CMOS для радиочастот                     | 5G-модули, фазированные антенные решётки      |

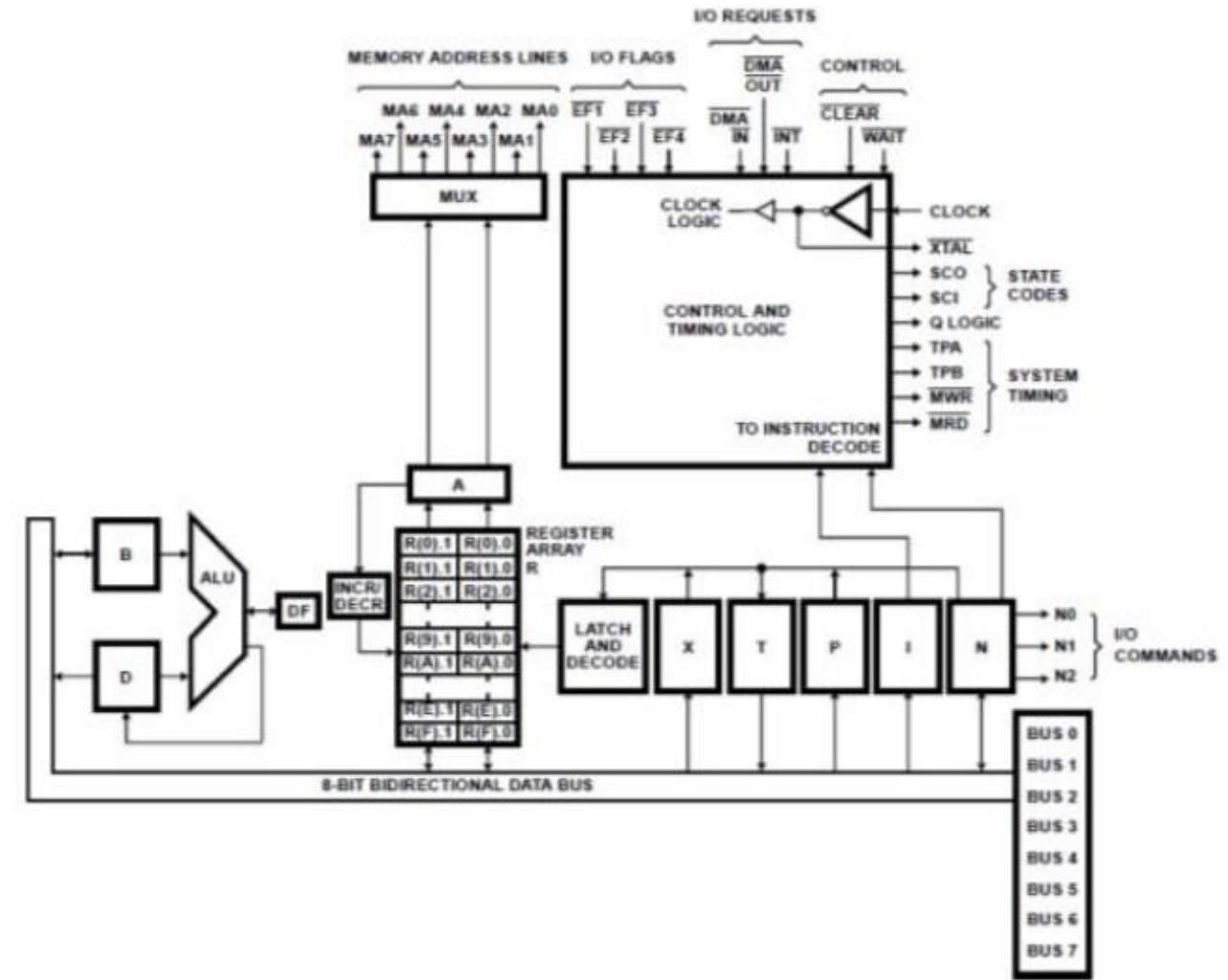
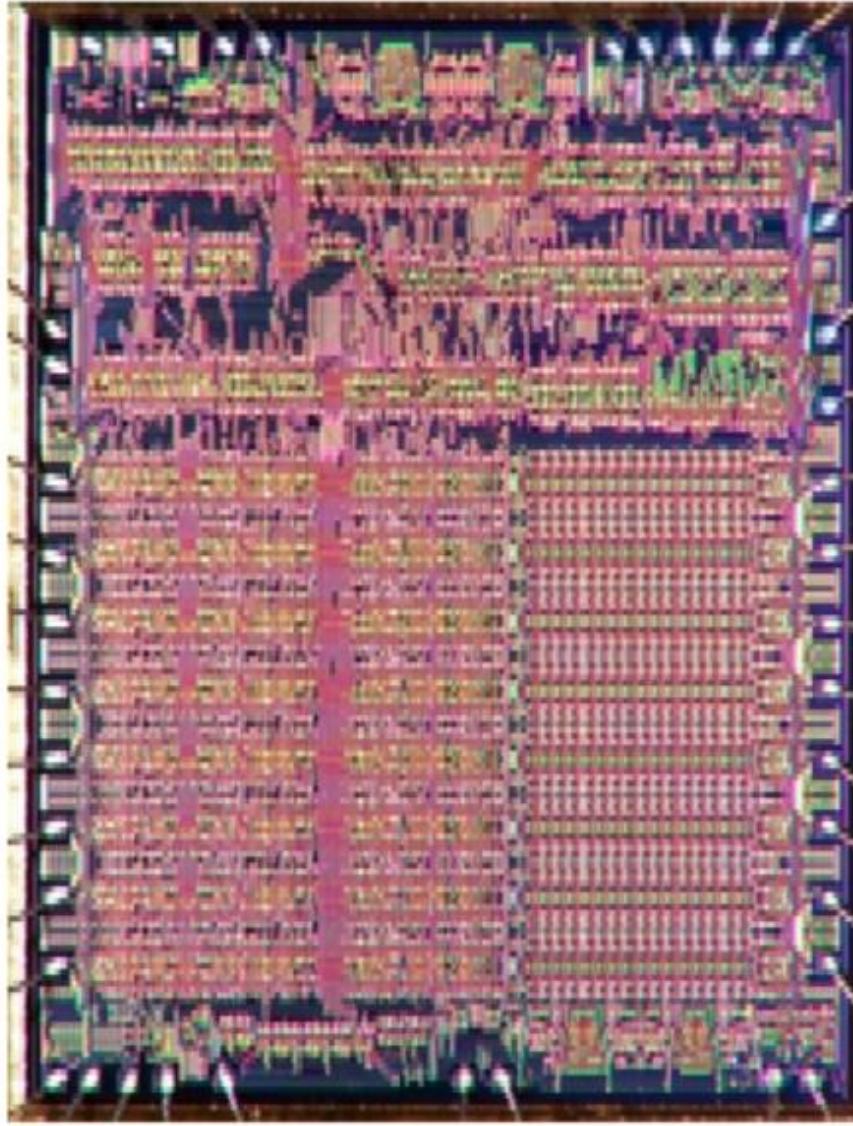
# Где применяется CMOS?

**CMOS — основа всей современной цифровой электроники**

| Область                           | Примеры                                      |
|-----------------------------------|--|
| <b>Микропроцессоры</b>            | Intel, AMD, Apple M1/M2, ARM-процессоры      |
| <b>Память</b>                     | SRAM, Flash (в части управления), регистры   |
| <b>Микроконтроллеры</b>           | STM32, AVR, PIC                              |
| <b>FPGA и ASIC</b>                | Программируемые логические схемы             |
| <b>Системы на кристалле (SoC)</b> | Snapdragon, Exynos, Raspberry Pi             |
| <b>Цифровые датчики</b>           | CMOS-матрицы в камерах (в отличие от CCD)    |
| <b>Таймеры, счётчики, шины</b>    | UART, SPI, I <sup>2</sup> C, USB контроллеры |

Примечание: Термин "CMOS-камера" — это не логика, а светочувствительная матрица, построенная по CMOS-технологии (в отличие от CCD).

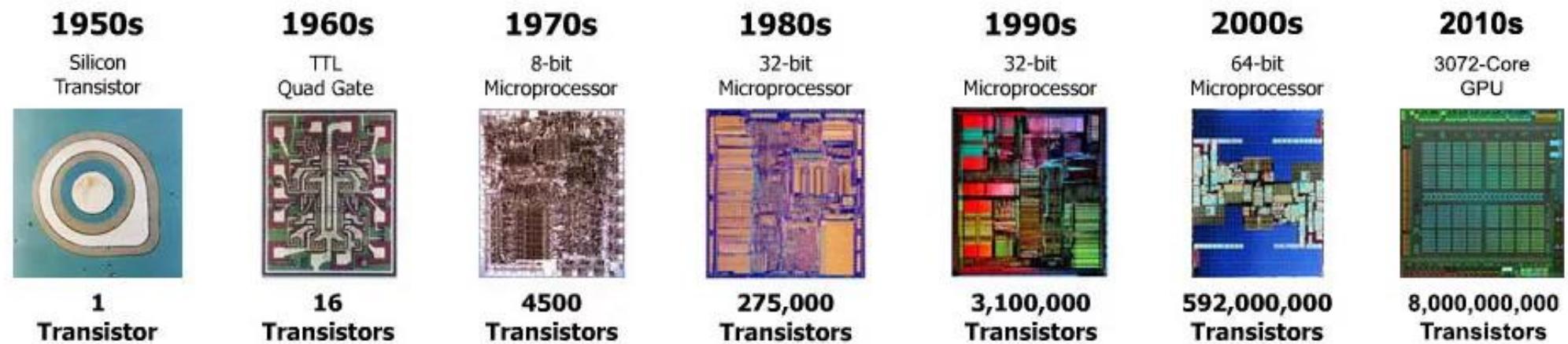
# Примеры (CMOS) КМОП-интегральных схем



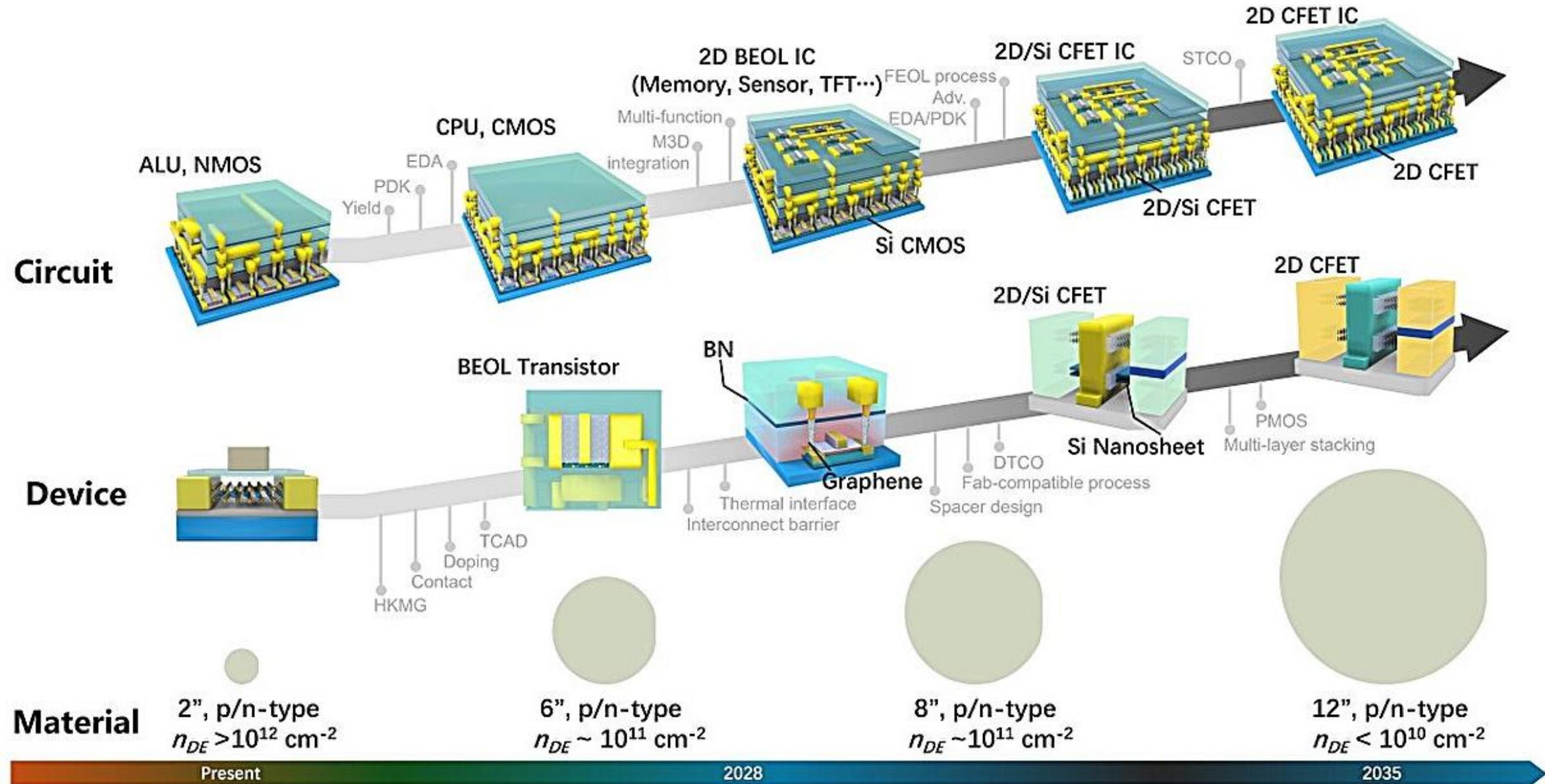
8-битный RCA 1802, примерно 1975 год: первый микропроцессор на КМОП-технологии (который попал на Юпитер!)

# CMOS в цифрах: Статистика и рекорды

- Плотность транзисторов:
  - Intel 4 (7 нм): **290 млн транзисторов/мм<sup>2</sup>**,
  - TSMC N3 (3 нм): **300 млн транзисторов/мм<sup>2</sup>**.
- Энергопотребление:
  - Apple M2 Ultra: **60 Вт** при 2,4 ТГц вычислений,
  - IoT-чипы (например, STM32U5): **26 нА** в режиме ожидания.
- Скорость:
  - Современные CMOS: до **5 ГГц** (Intel Core i9),
  - Экспериментальные CMOS на SiGe: **700 ГГц**.

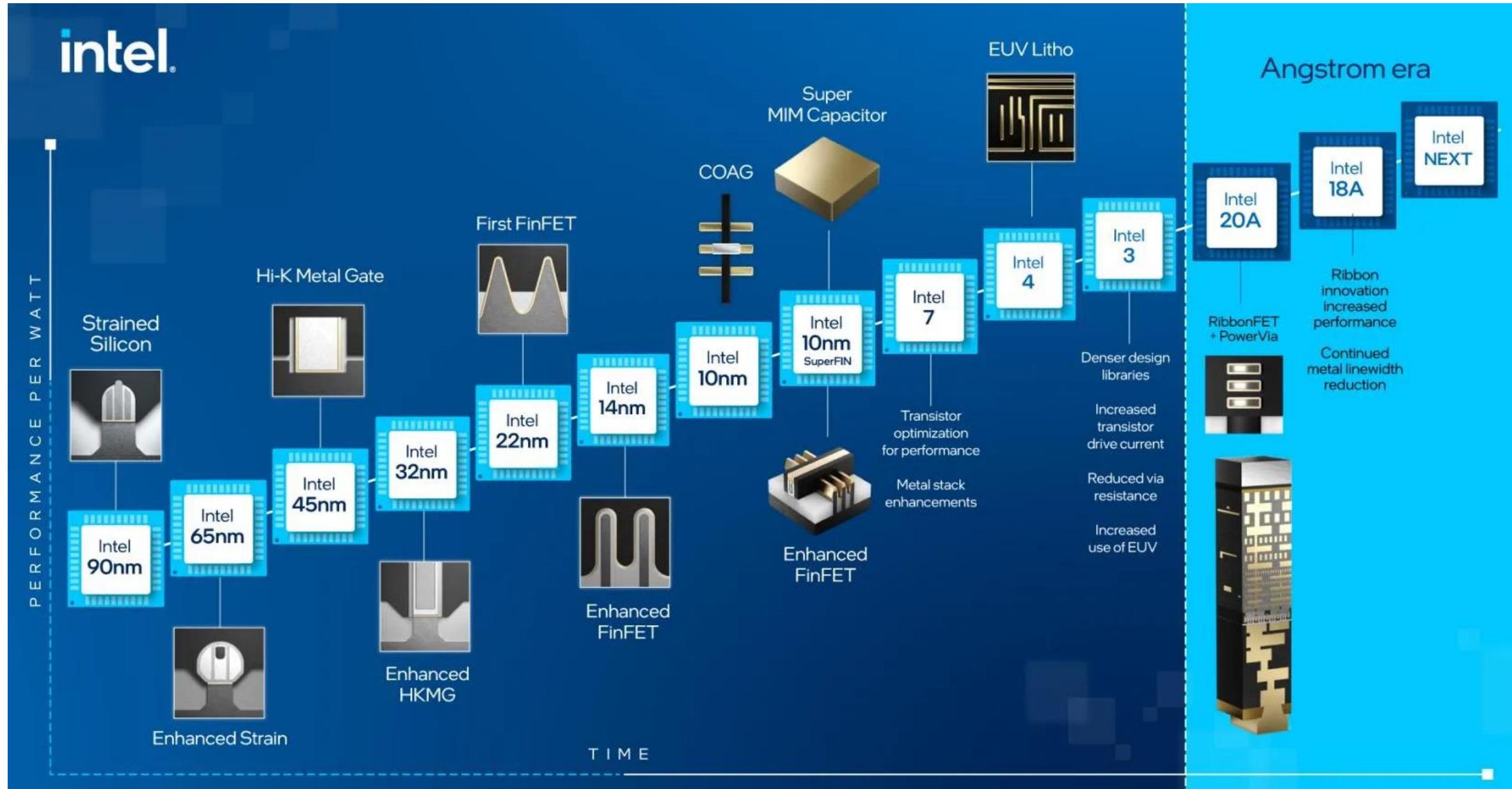


# Развитие технологий



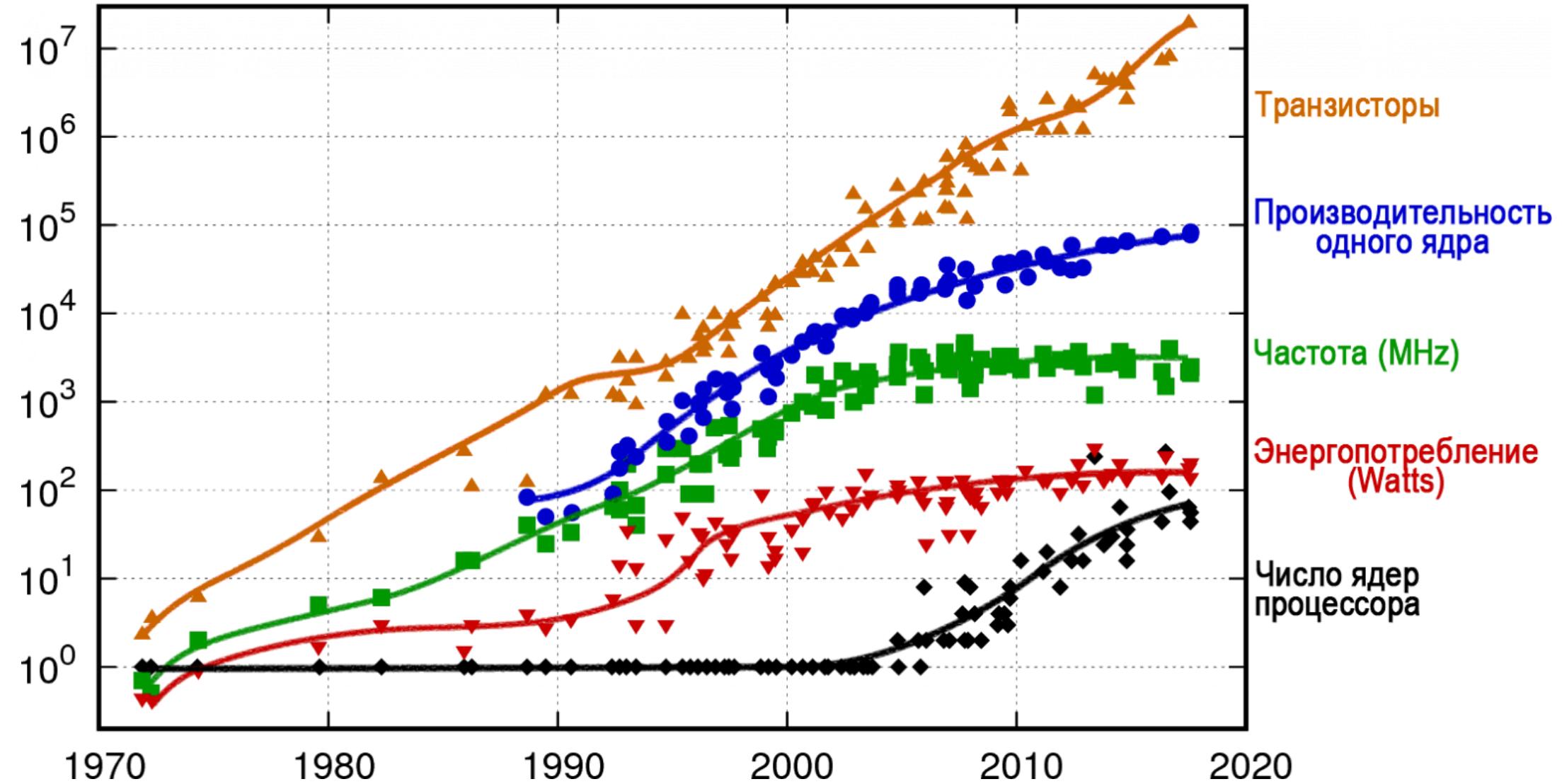
Roadmap for the future development of 2D information materials layout planning. Credit: Science China Press

# Развитие технологий

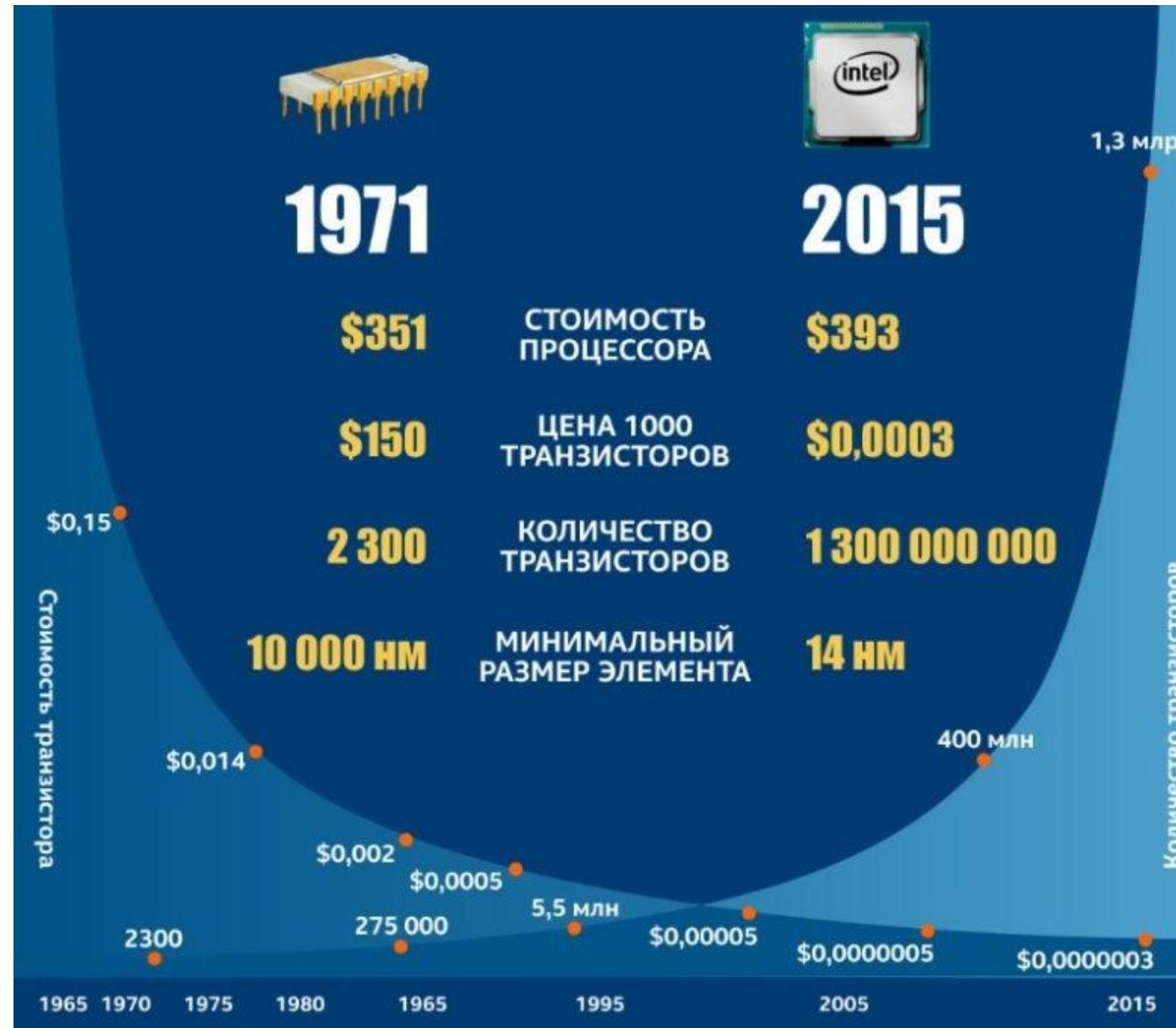


# Тенденции в микроэлектронике

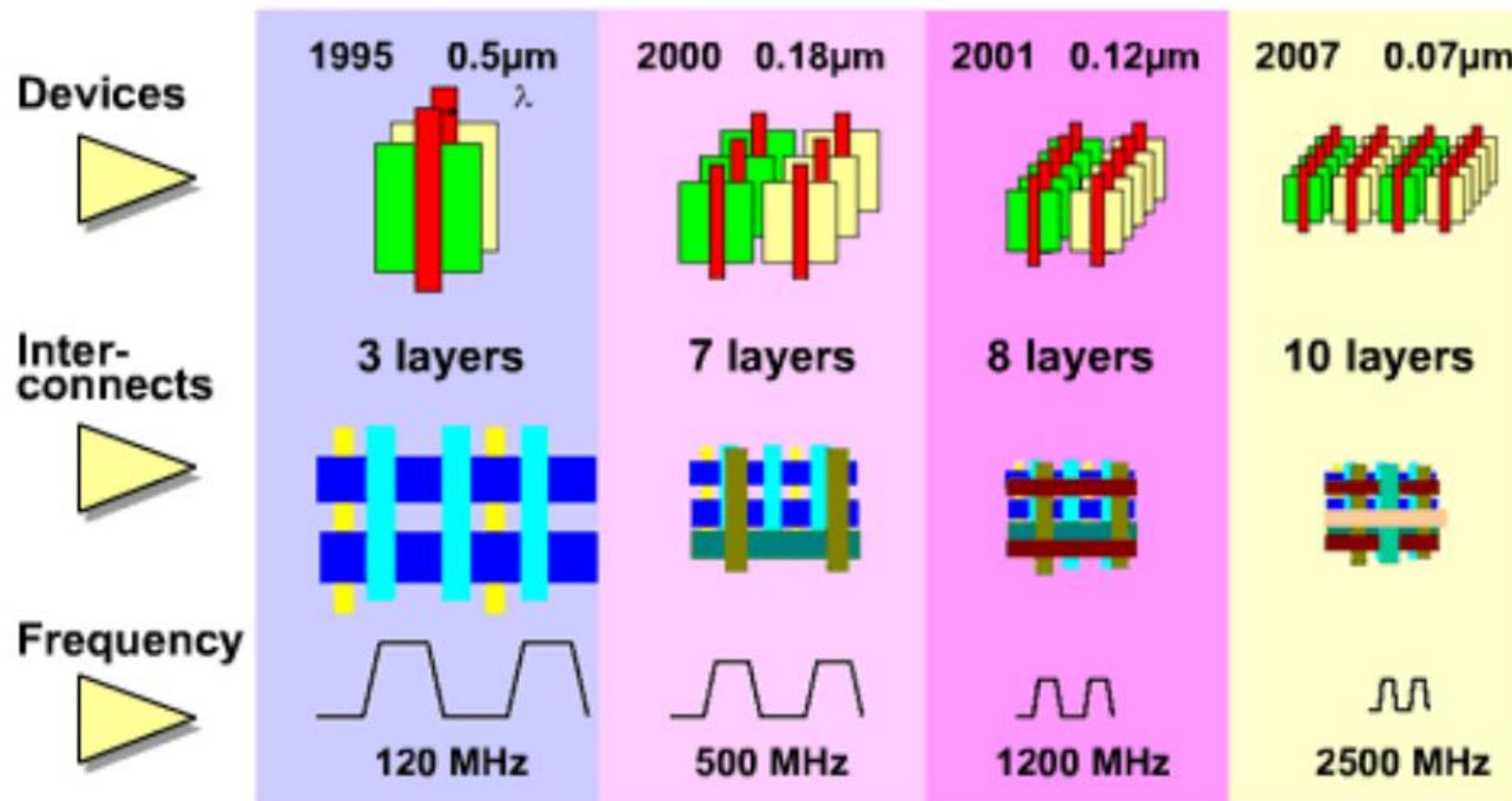
РОСТ ПРОИЗВОДИТЕЛЬНОСТИ ПРОЦЕССОРОВ ПО ГОДАМ



# Тенденции в микроэлектронике

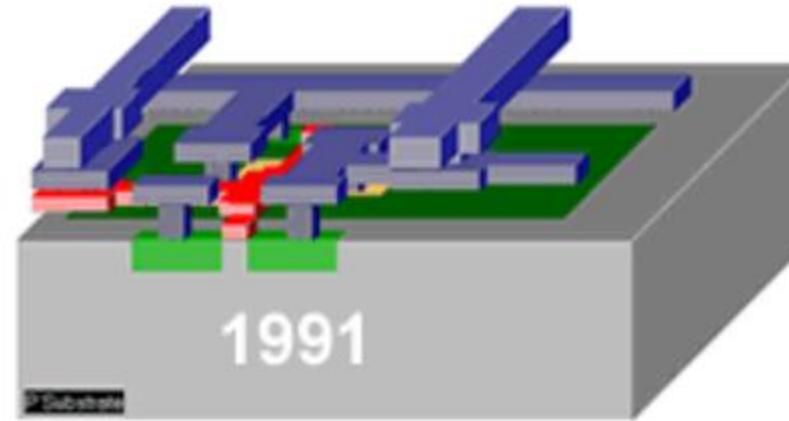


# Тенденции в микроэлектронике

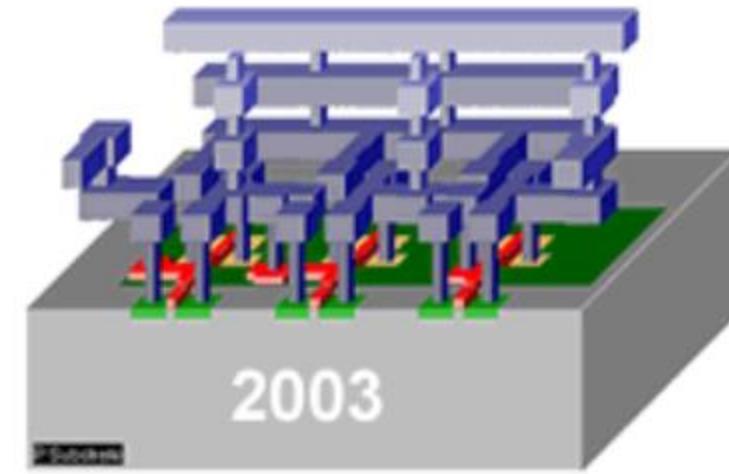


**Основной тенденцией микроэлектроники является обеспечение тех же функций на меньшей площади**, в этом случае можно на тех же площадях обеспечивать больше функций. Число слоев металлизации увеличивается, большее число слоев дает больше возможностей по сокращению площади кристалла, активные области транзисторов могут быть ближе расположены. Сокращение площадей приводит к уменьшению паразитных емкостей, **повышается тактовая частота, уменьшается необходимое напряжение питания**.

# Рост сложности проектирования при уменьшении технологических норм



- 0.7µm, 2 metal layers
- Up to 100,000 devices on a chip
- CPU frequency 50MHz

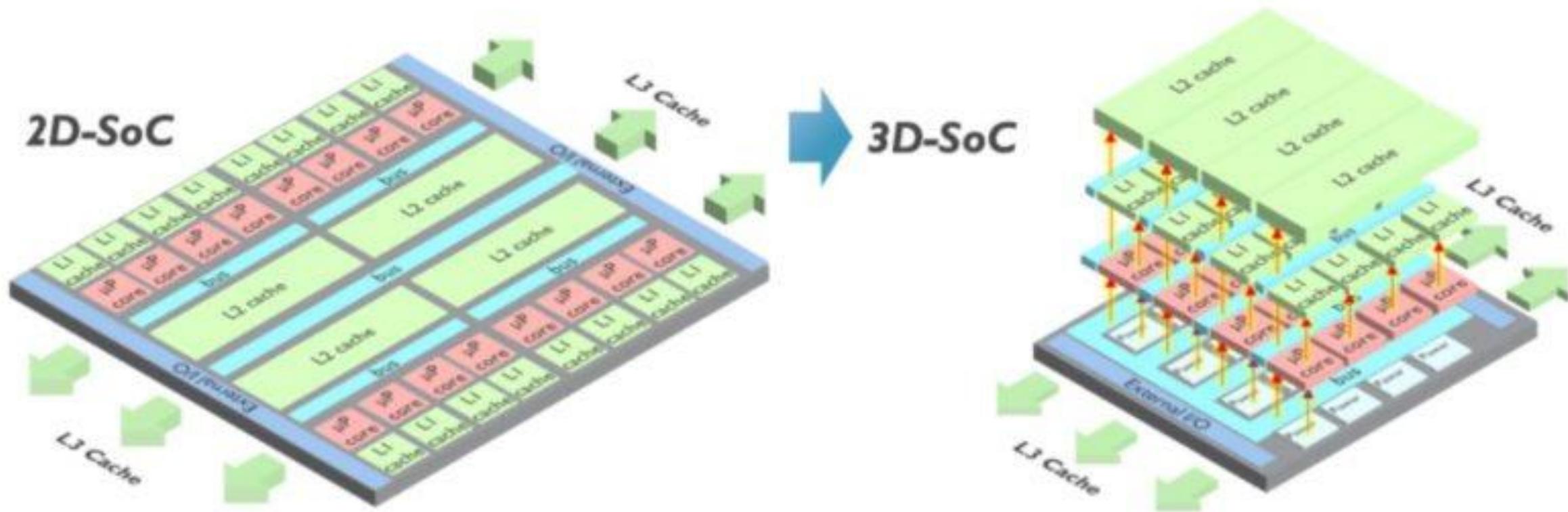


- 0.12µm, 7 metal
- Up to 100,000,000 devices
- CPU frequency 1.5GHz



Прогресс в сложности проектирования (технология 0,7 и 0,12 мкм, 2 и 7 слоев металлизации, число транзисторов на кристалле, тактовая частота)

# Тенденции в микроэлектронике



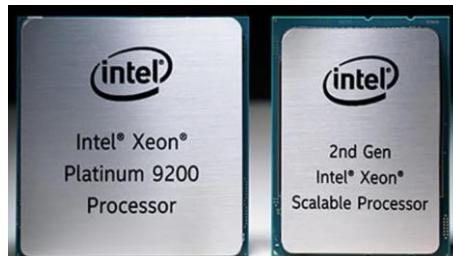
# Современные цифровые вычислительные системы



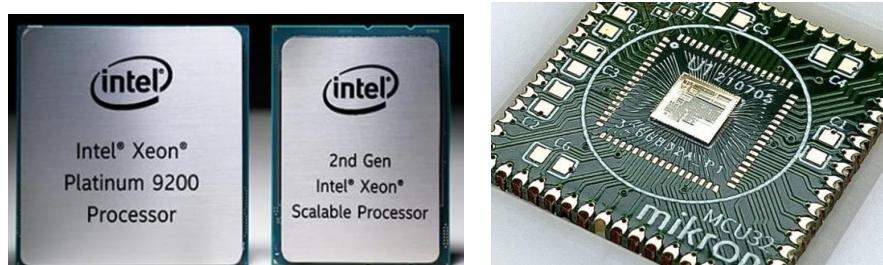
GPU



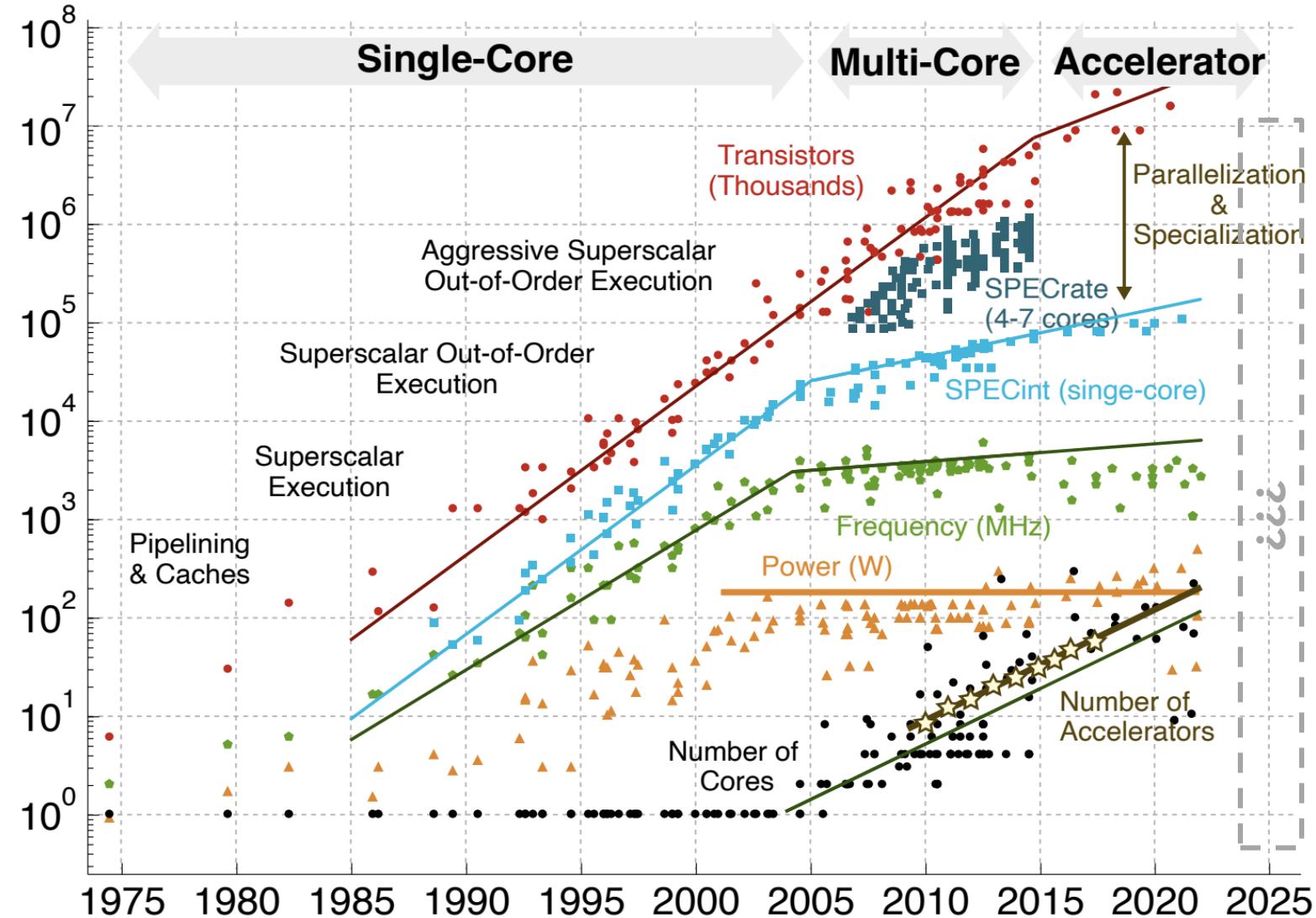
System-on-Chip



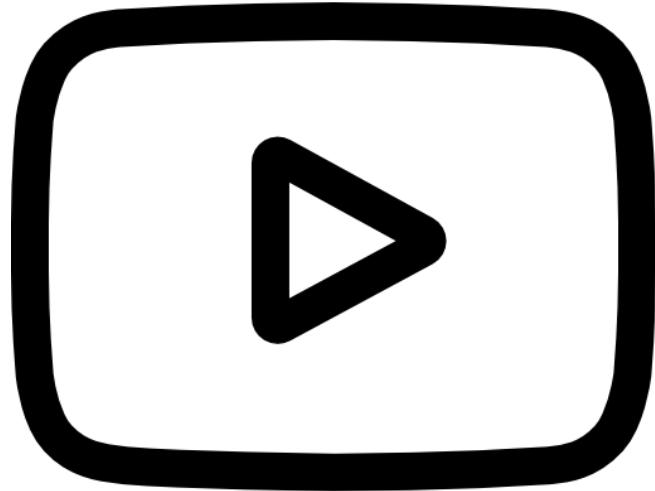
CPU



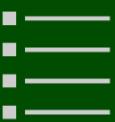
Микроконтроллер

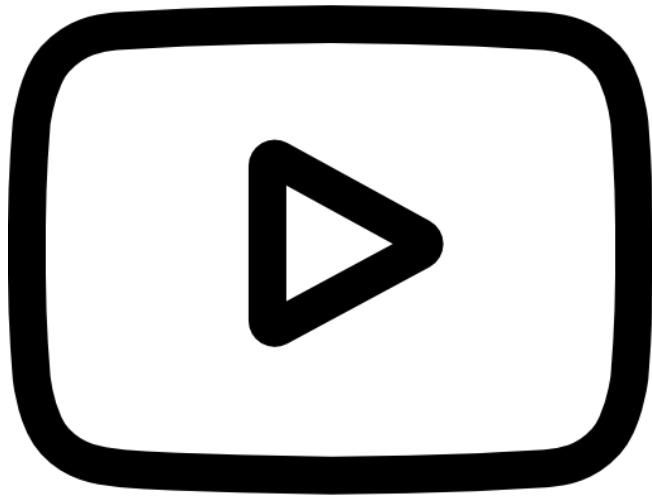


C. Batten, M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, K. Rupp & [Y. Shao, IEEE Micro'15] & [C. Leiserson, Science'20]



Дополнительные  
материалы по  
теме на YouTube

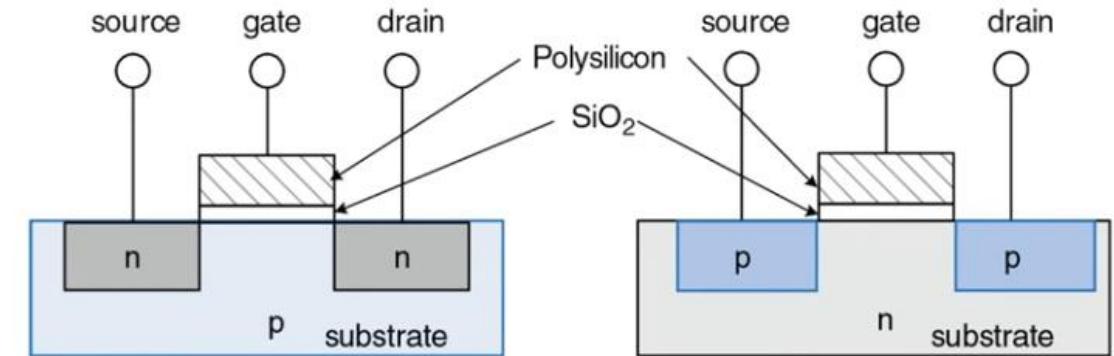
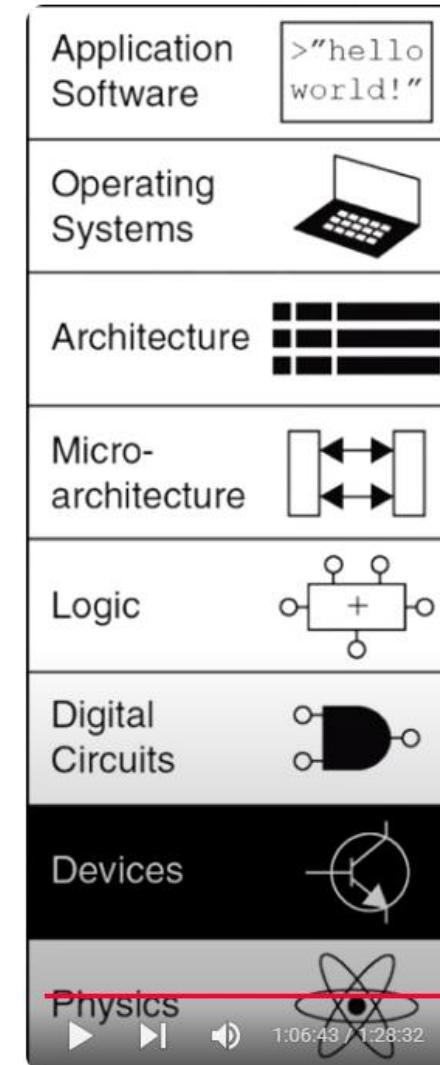




# Уровни абстракции электронно- вычислительных систем



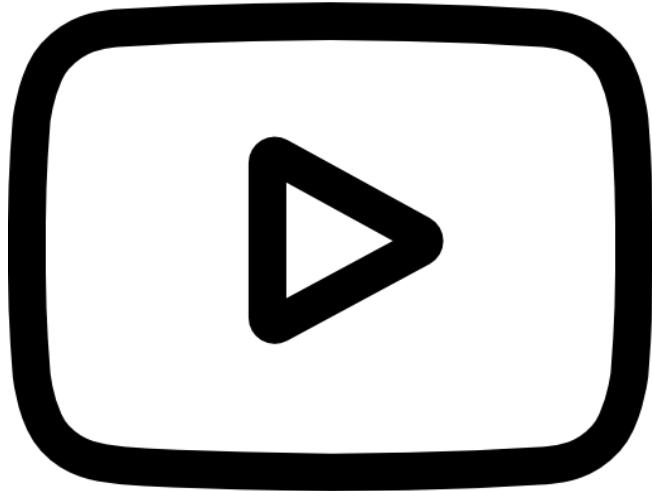
YouTube 1:28:32



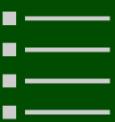
nMOS

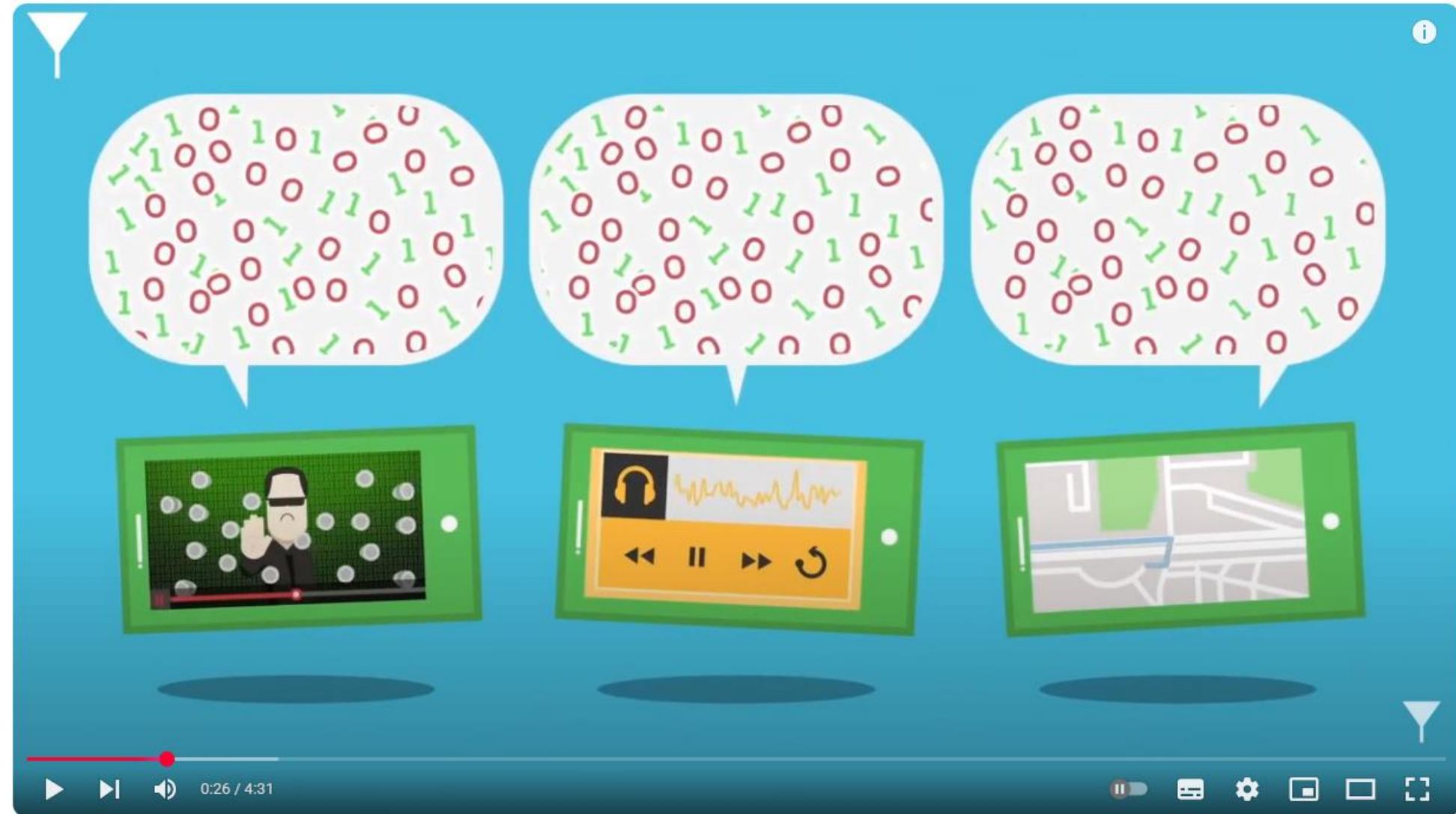


АПС Л1. Вводная (ПМ, ИВТ, ПИН) (2021)  
<https://www.youtube.com/watch?v=JmiMaKVolol>



# Системы счисления и представление информации

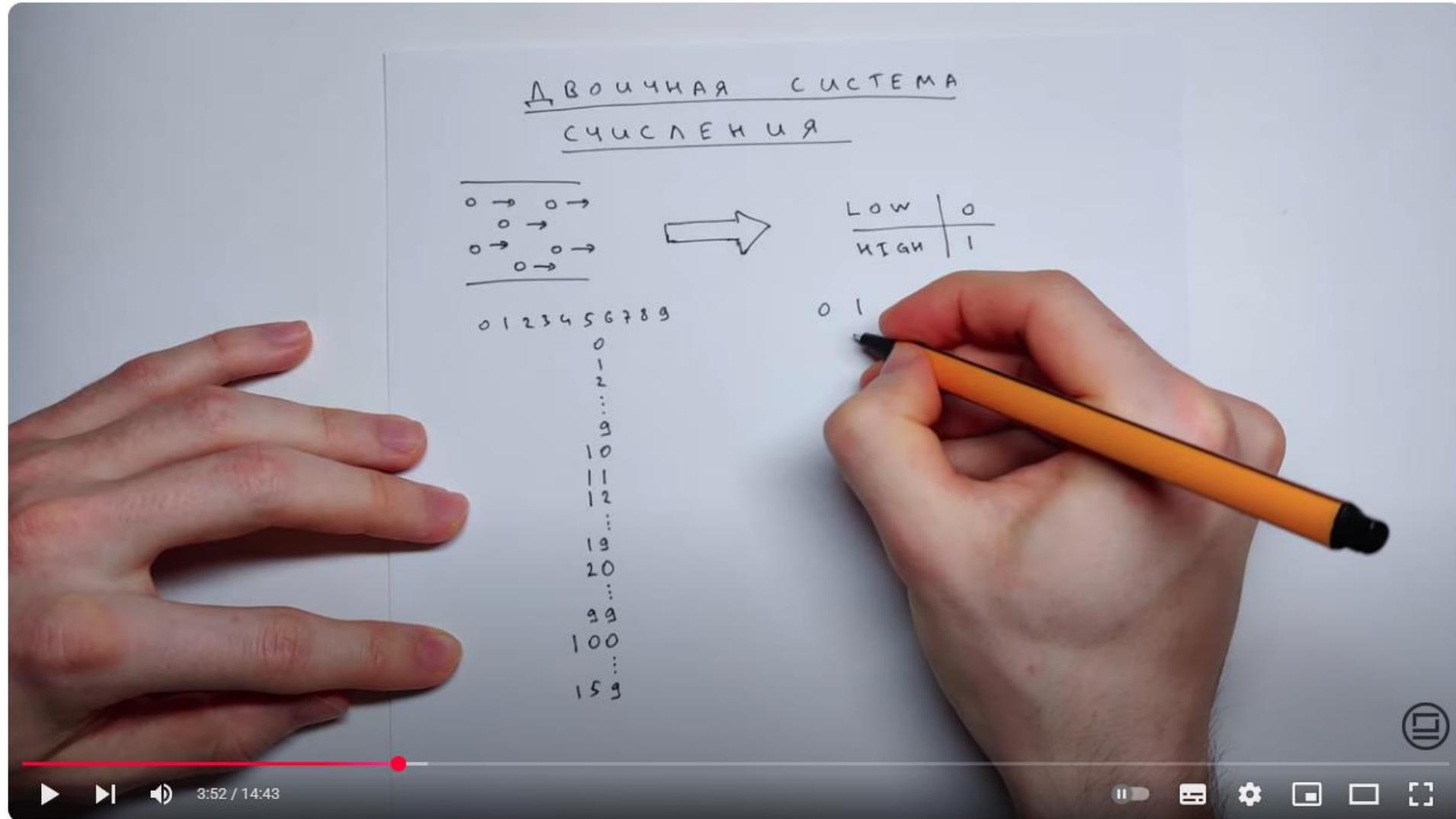




Как на самом деле работает двоичный код? (2018)  
<https://www.youtube.com/watch?v=GL3kPXjSaWY>



Применение двоичной системы счисления в реальной жизни (2020)  
<https://www.youtube.com/watch?v=VawNmOMMxCE>



Двоичная система счисления - язык понятный компьютеру (2020)  
<https://www.youtube.com/watch?v=yOGgS-JWdV4>

## Разложение числа на слагаемые по основанию два

- Отличие двоичной и десятичной систем только в количестве символов для записи цифр
- Запишем разряды и основание таблицы более компактно, используя цифры десятичной системы

| Десятичная | Двоичная |
|------------|----------|
| 0          | 0        |
| 1          | 1        |
| 2          | 10       |
| 3          | 11       |

| Число     | 1         | 1         | 0      | 1      |
|-----------|-----------|-----------|--------|--------|
| Разряд    | 11        | 10        | 1      | 0      |
| Основание | $10^{11}$ | $10^{10}$ | $10^1$ | $10^0$ |



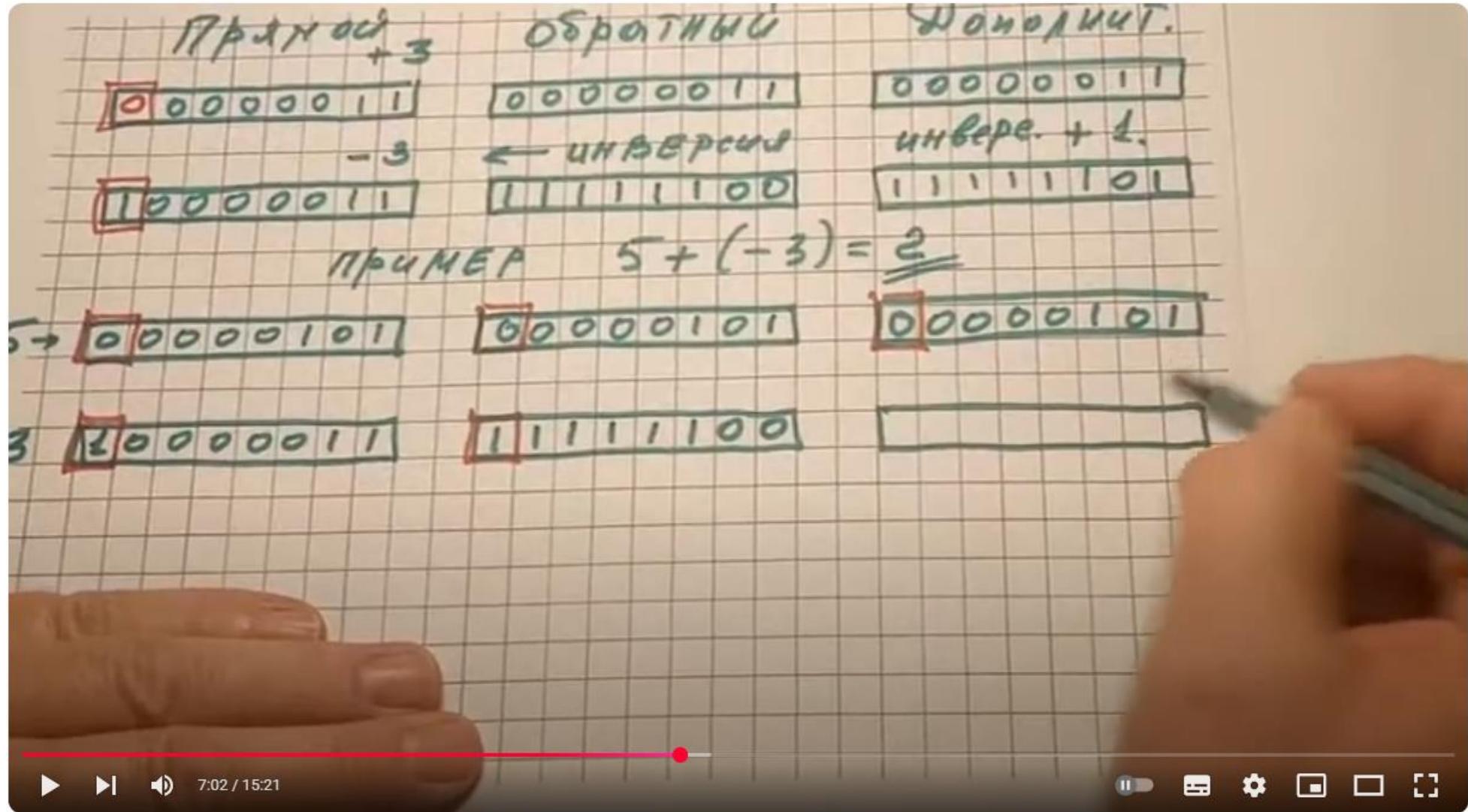
| Число     | 1     | 1     | 0     | 1     |
|-----------|-------|-------|-------|-------|
| Разряд    | 3     | 2     | 1     | 0     |
| Основание | $2^3$ | $2^2$ | $2^1$ | $2^0$ |

$$\begin{aligned}
 & 1 * 10^{11} + 1 * 10^{10} + 0 * 10^1 + 1 * 10^0 = 1101 \\
 & 1 * 2^3 + 1 * 2^2 + 0 * 2^1 + 1 * 2^0 \\
 & 8 + 4 + 0 + 1 = 13
 \end{aligned}$$

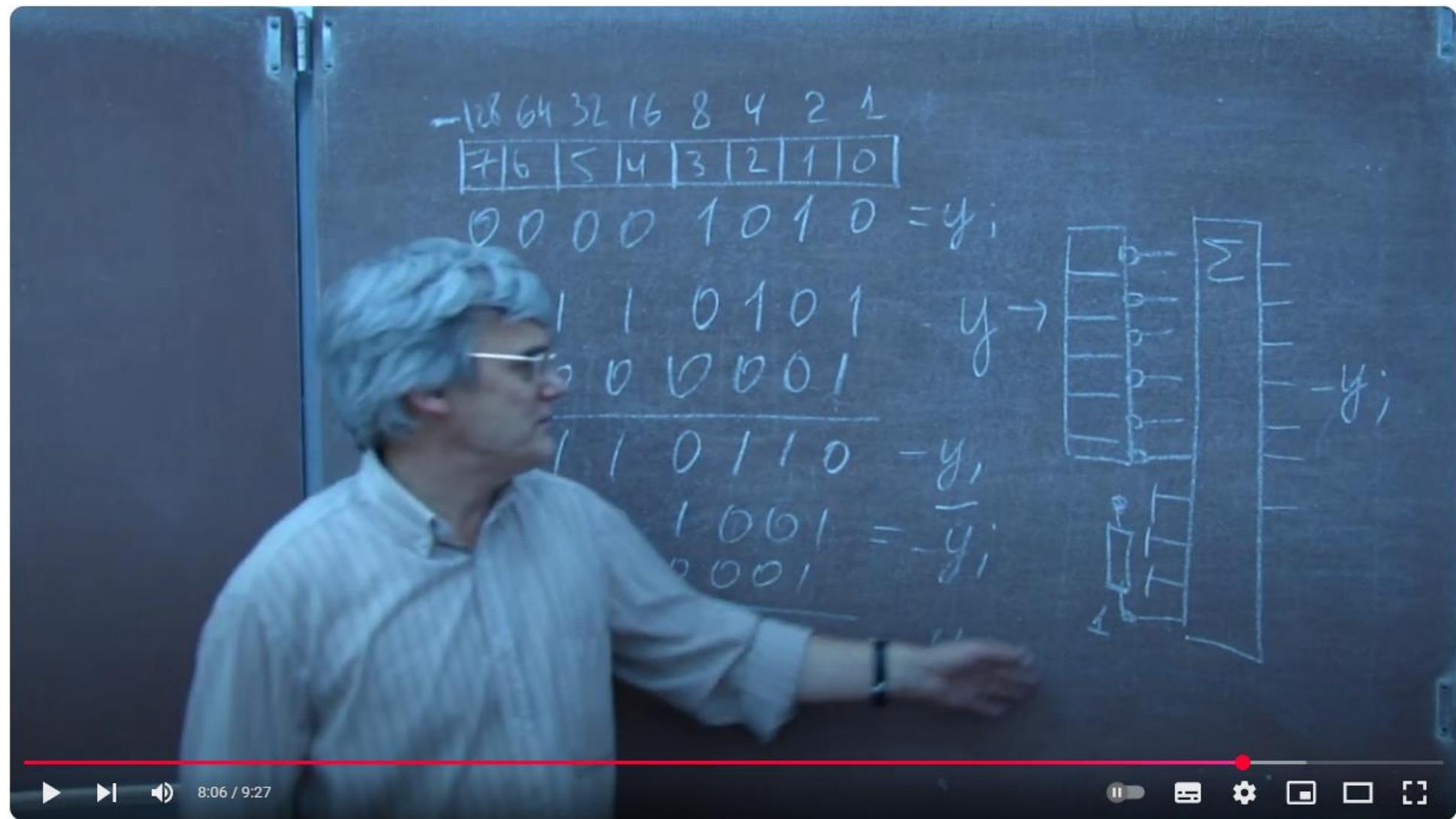
Мы преобразовали число из двоичной системы в десятичную

1101 -> 13

Двоичная система счисления. Максимально просто и подробно (2019)  
[https://www.youtube.com/watch?v=Ro8jdy\\_krko](https://www.youtube.com/watch?v=Ro8jdy_krko)



Прямой Обратный Дополнительный (2020)  
[https://www.youtube.com/watch?v=Gn8P\\_TYa\\_AU](https://www.youtube.com/watch?v=Gn8P_TYa_AU)



## Лекция 110. Арифметика отрицательных чисел в микропроцессорах (2013)

<https://www.youtube.com/watch?v=Zd0mM5pgORY>

*Двоичная система.*

0, 1

0 + 0 = 0      0 - 0 = 0  
0 + 1 = 1      1 - 1 = 0  
1 + 0 = 1      1 - 0 = 1  
1 + 1 = 10      10 - 1 = 1

0 - 0  
1 - 1  
2 - 10  
3 - 11  
4 - 100  
5 - 101  
6 - 110  
7 - 111  
8 - 1000

$$\begin{array}{r} 10110 \\ + 10110 \\ \hline 1000011 \end{array}$$

$$\begin{array}{r} 10110 \\ - 10110 \\ \hline 0110 \end{array}$$

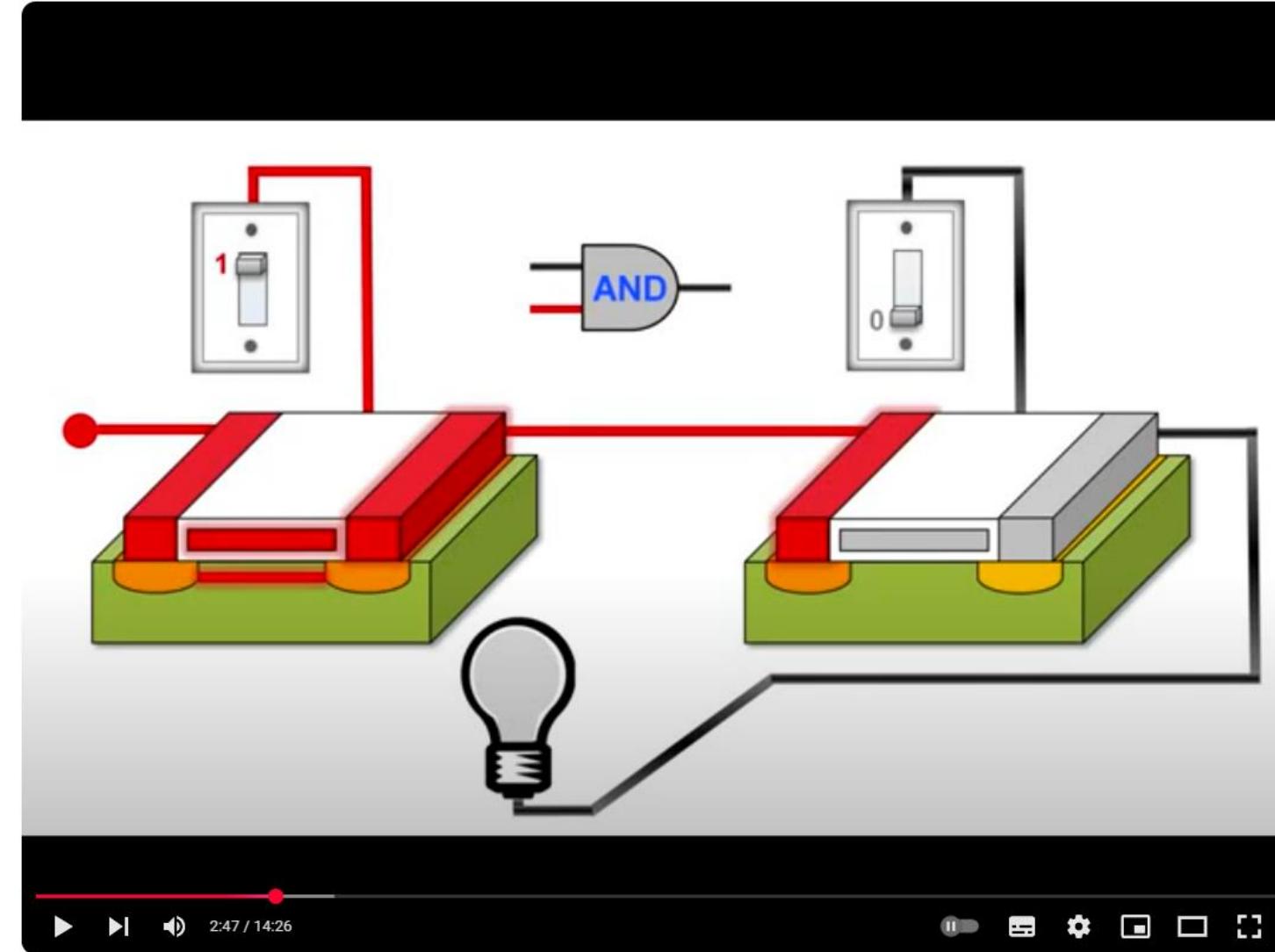
7:08 / 9:58

Os

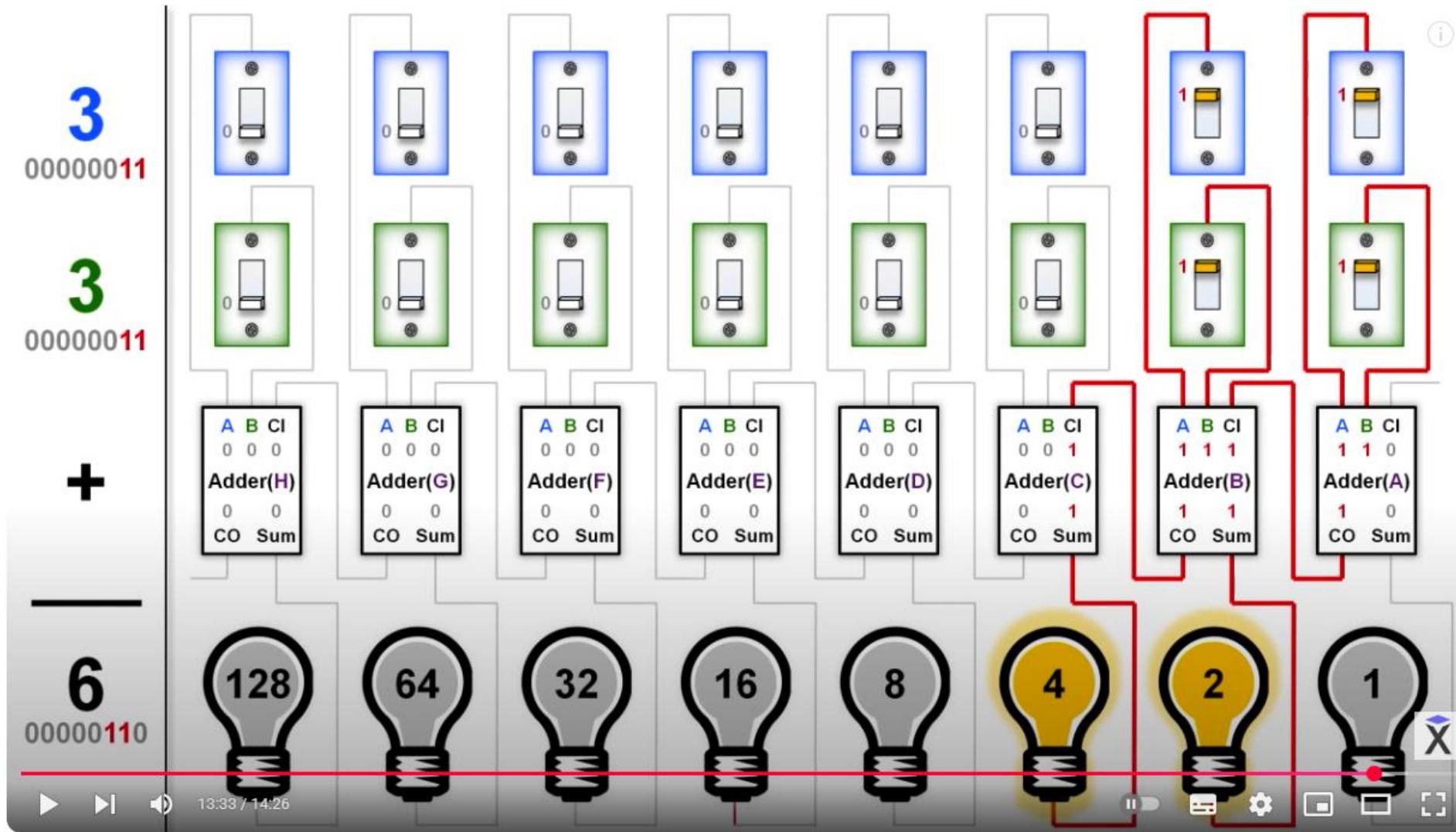
ИНФОРМАТИКА 8 класс: Двоичная система счисления. Двоичная арифметика (2018)  
<https://www.youtube.com/watch?v=NIPeCMjri3Q>



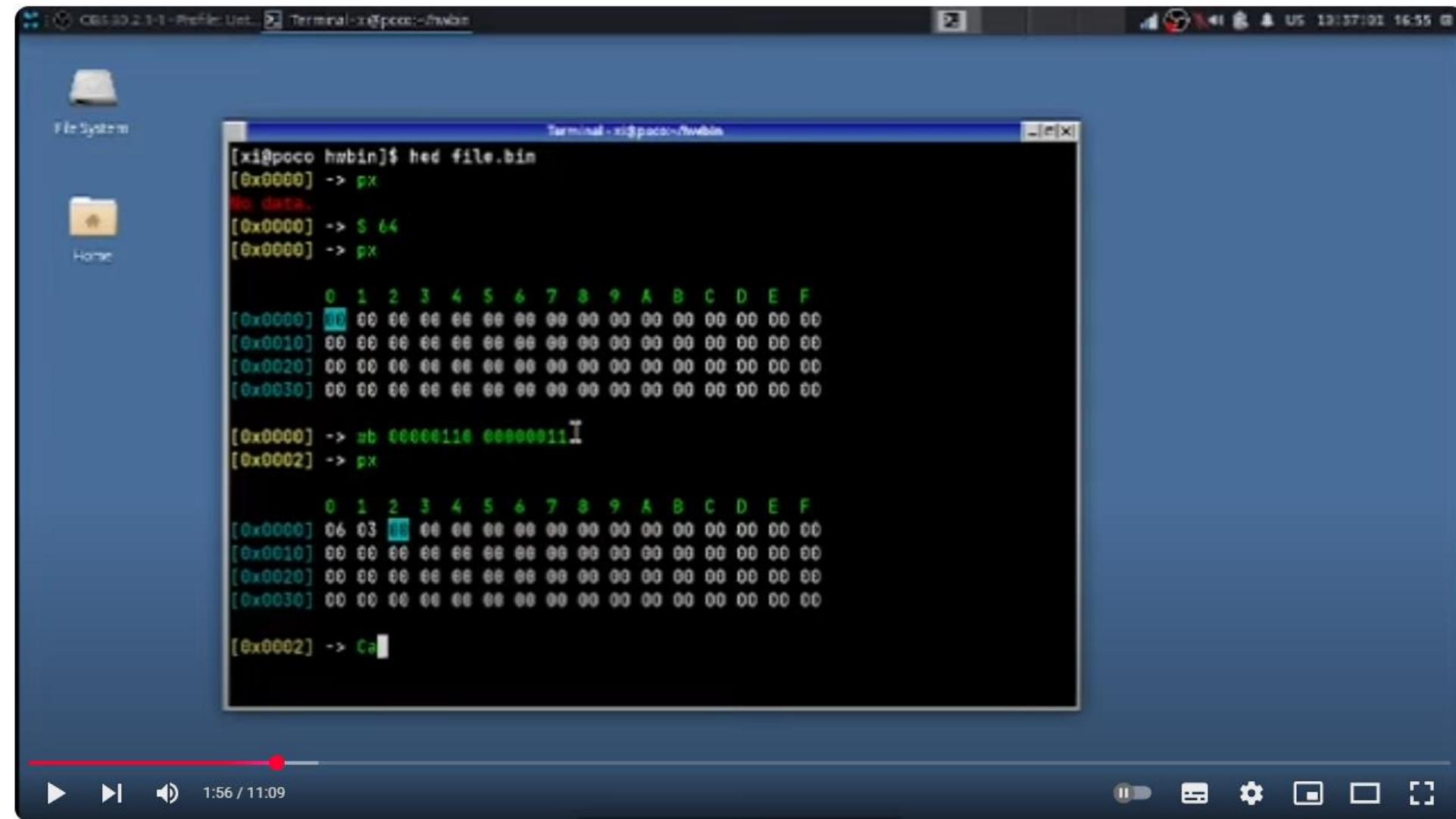
Информатика 10 класс. Представление чисел в компьютере (УМК БОСОВА Л.Л., БОСОВА А.Ю.) (2020) <https://www.youtube.com/watch?v=kMvWakrKZJE>



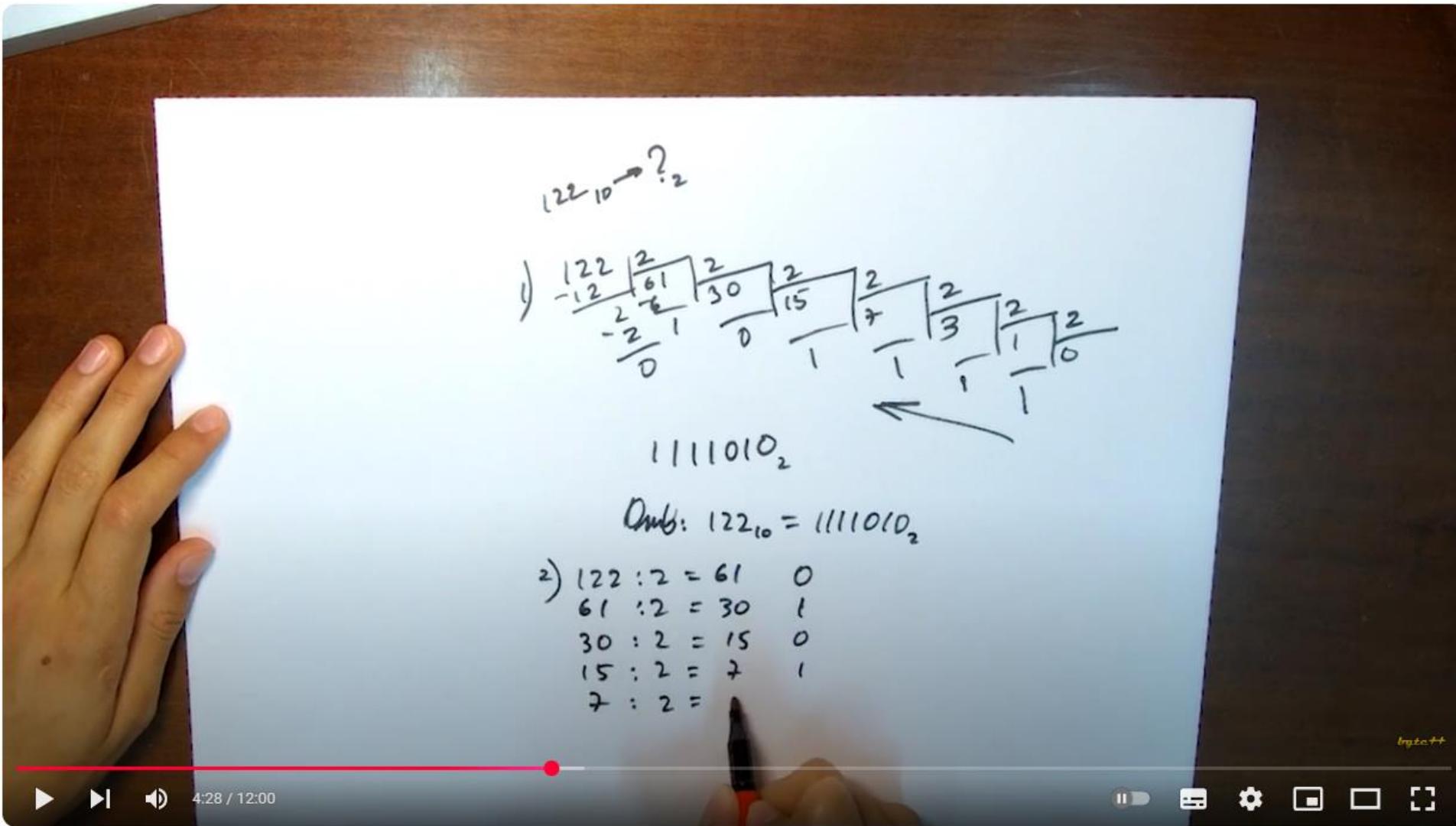
Сложение в микропроцессоре. Как работает двоичная система в процессоре (2012)  
[https://www.youtube.com/watch?v=wi\\_QNI8EZhA](https://www.youtube.com/watch?v=wi_QNI8EZhA)



Как компьютеры складывают числа | Хекслет (2015)  
<https://www.youtube.com/watch?v=YuSgZ173Utg>

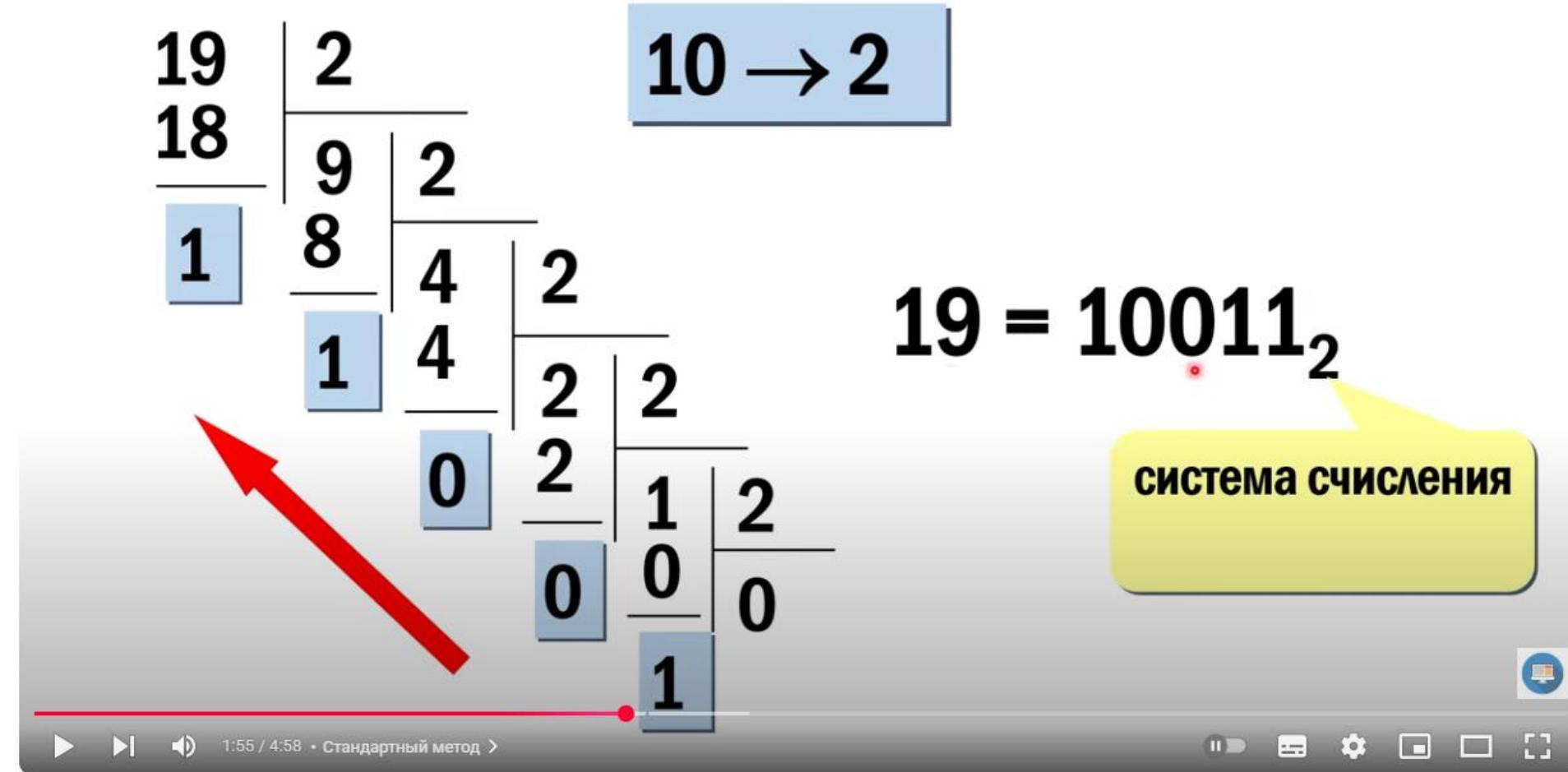


Как написать Hello, World! на двоичном коде (не шестнадцатеричный) | GovnBin (2024)  
[https://www.youtube.com/watch?v=2-WuohF\\_vGo](https://www.youtube.com/watch?v=2-WuohF_vGo)



Двоичная система счисления. Урок 1 (2017)  
<https://www.youtube.com/watch?v=FGRIYjHfzSY>

# Универсальный способ



Перевод из десятичной в двоичную систему счисления (2021)  
<https://www.youtube.com/watch?v=7KYxTj4UrhY>

# Метод подстановки

**2 → 10**

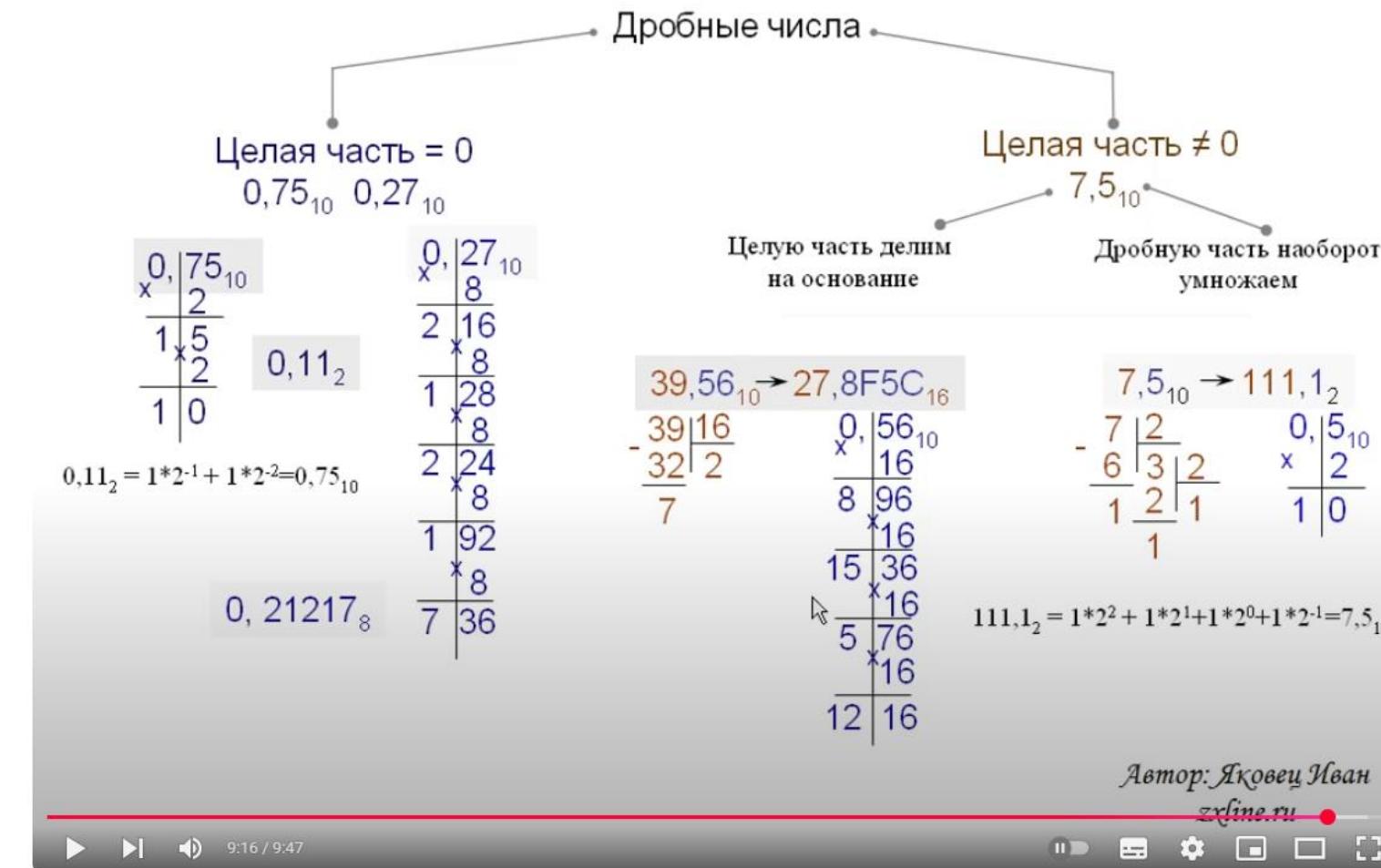


$$64 + 8 + 4 + 2 + 1 = 79_{10}$$

1:14 / 2:36 • Метод подбора >

Перевод из двоичной в десятичную систему счисления (2021)  
<https://www.youtube.com/watch?v=VVh8btCx1nk>

## Перевод дробных чисел из 10 в 2,8,16 системы счисления



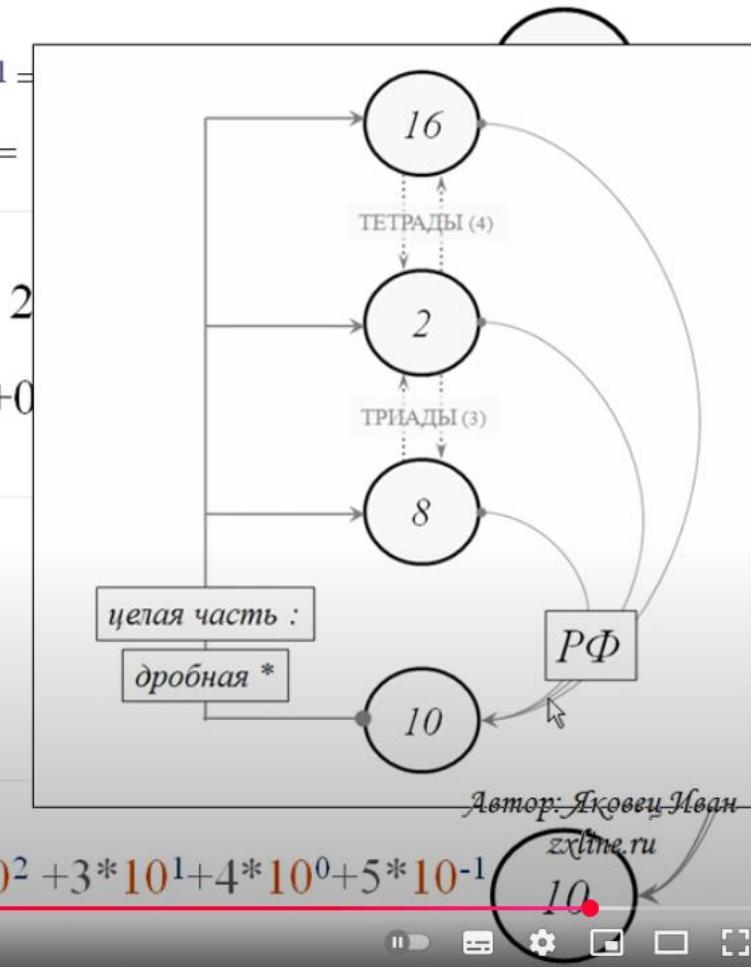
Перевод дробных чисел из 10 системы в 2, 8, 16 (2013)  
<https://www.youtube.com/watch?v=yLCZXD510Vo>

Перевод из 16,2,8  $\Leftrightarrow$  10

$$\begin{aligned}9A,B_{16} &= 9 \cdot 16^1 + 10 \cdot 16^0 + 11 \cdot 16^{-1} = \\&= 154 + 11 \cdot 0,0625 =\end{aligned}$$

$$11,11_2 = 1 \cdot 2^1 + 1 \cdot 2^0 + 1 \cdot 2^{-1} + 1 \cdot 2^{-2} = 2$$

$$\begin{array}{cccccc}16 & 8 & 4 & 2 & 1 & 0,5 & 0,25 \\1 & 0 & 1 & 0 & 1,1 & 1_2 & = 16+0+4+0+1+0\end{array}$$

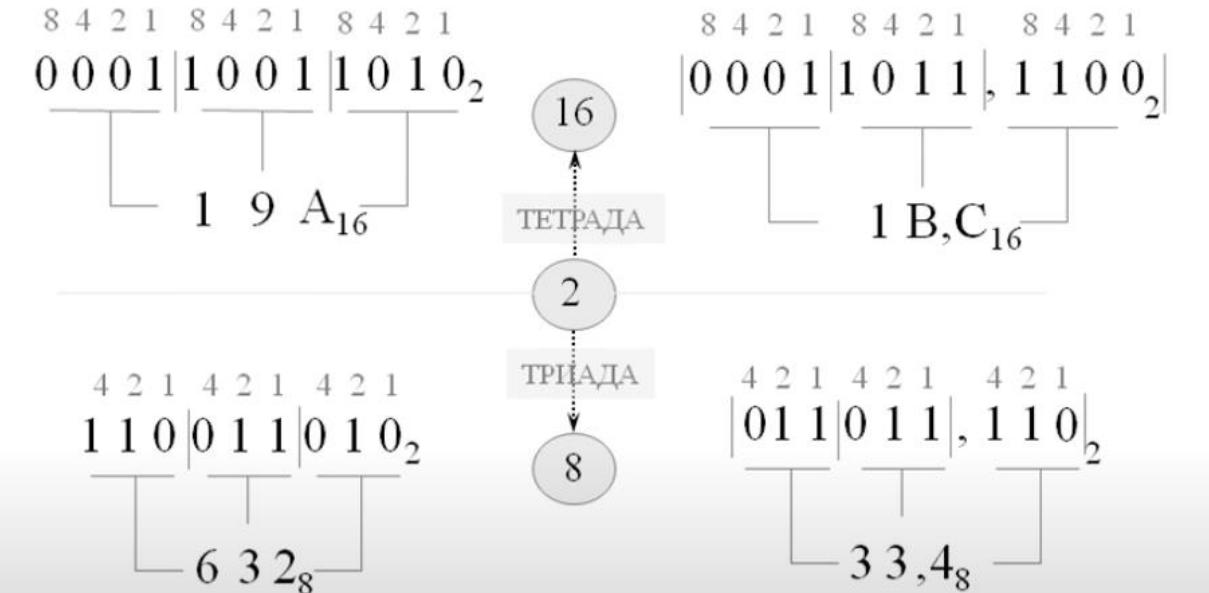


$$\underline{\underline{5 \ 3 \ 4,5}}_{10} = 500+30+4+0,5 = 5 \cdot 10^2 + 3 \cdot 10^1 + 4 \cdot 10^0 + 5 \cdot 10^{-1}$$

▶ 2 ▶ 1 0 -1 12:12 / 13:53

Перевод из 2, 8, 16 систем счисления в десятичную систему счисления (2013)  
<https://www.youtube.com/watch?v=G-ptY2HOcBk>

Перевод из  $2 \Leftrightarrow 8, 2 \Leftrightarrow 16 :$



Автор: Яковец Иван  
zxfline.ru



Перевод из двоичной в восьмеричную и шестнадцатеричную системы счисления целых и дробных чисел (2013) [https://www.youtube.com/watch?v=qSB\\_fkx3LYs](https://www.youtube.com/watch?v=qSB_fkx3LYs)

Сложение, вычитание и умножение двоичных чисел

$$1+0=1$$

$$1+1=10_2$$

$$1+1+1=11_2$$

$$1+1+1+1=100_2$$

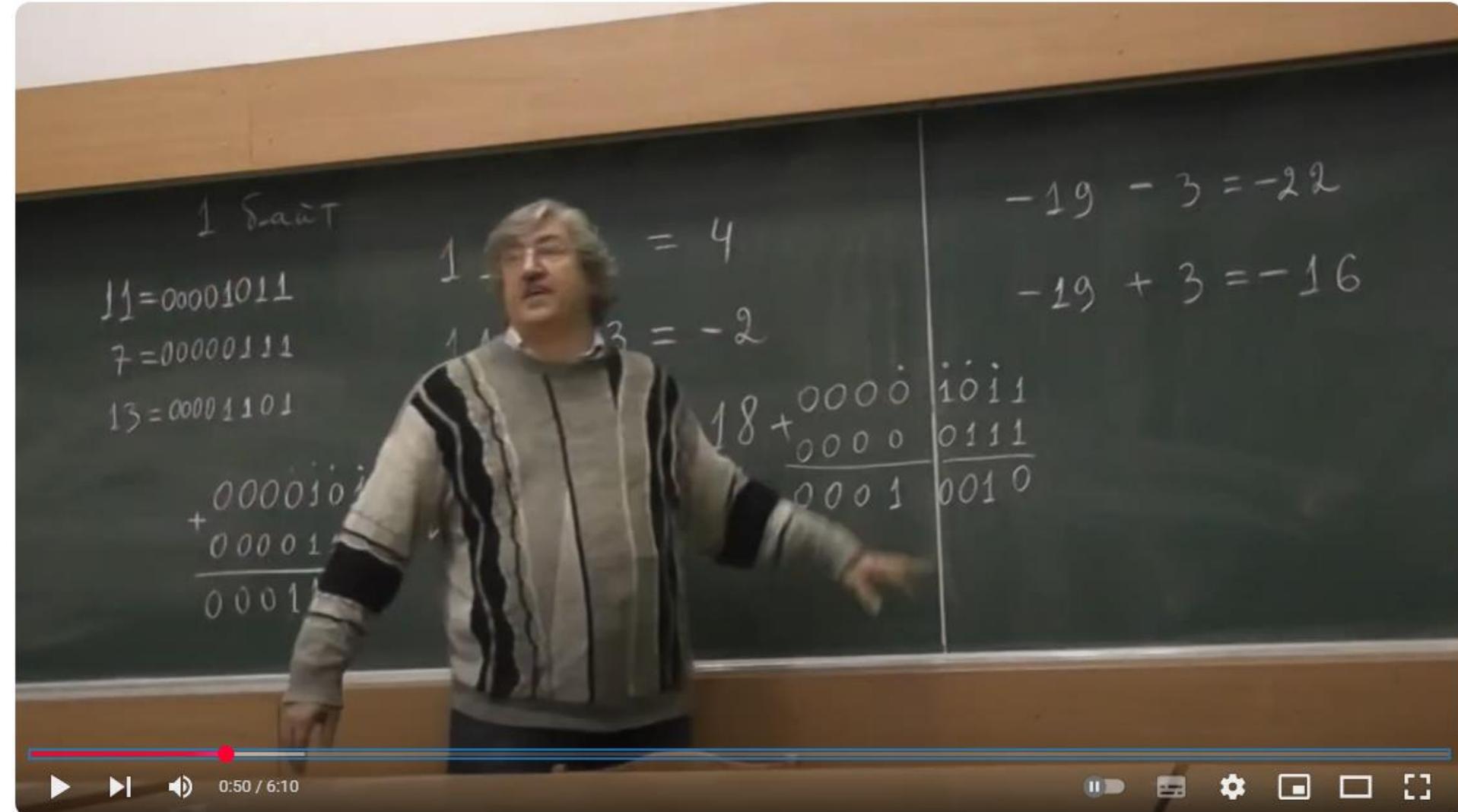
$$\begin{array}{r} 1011010 \\ + 111011 \\ \hline 10010101 \end{array}$$

$$\begin{array}{r} 10010101 \\ - 111011 \\ \hline 1011010 \end{array}$$

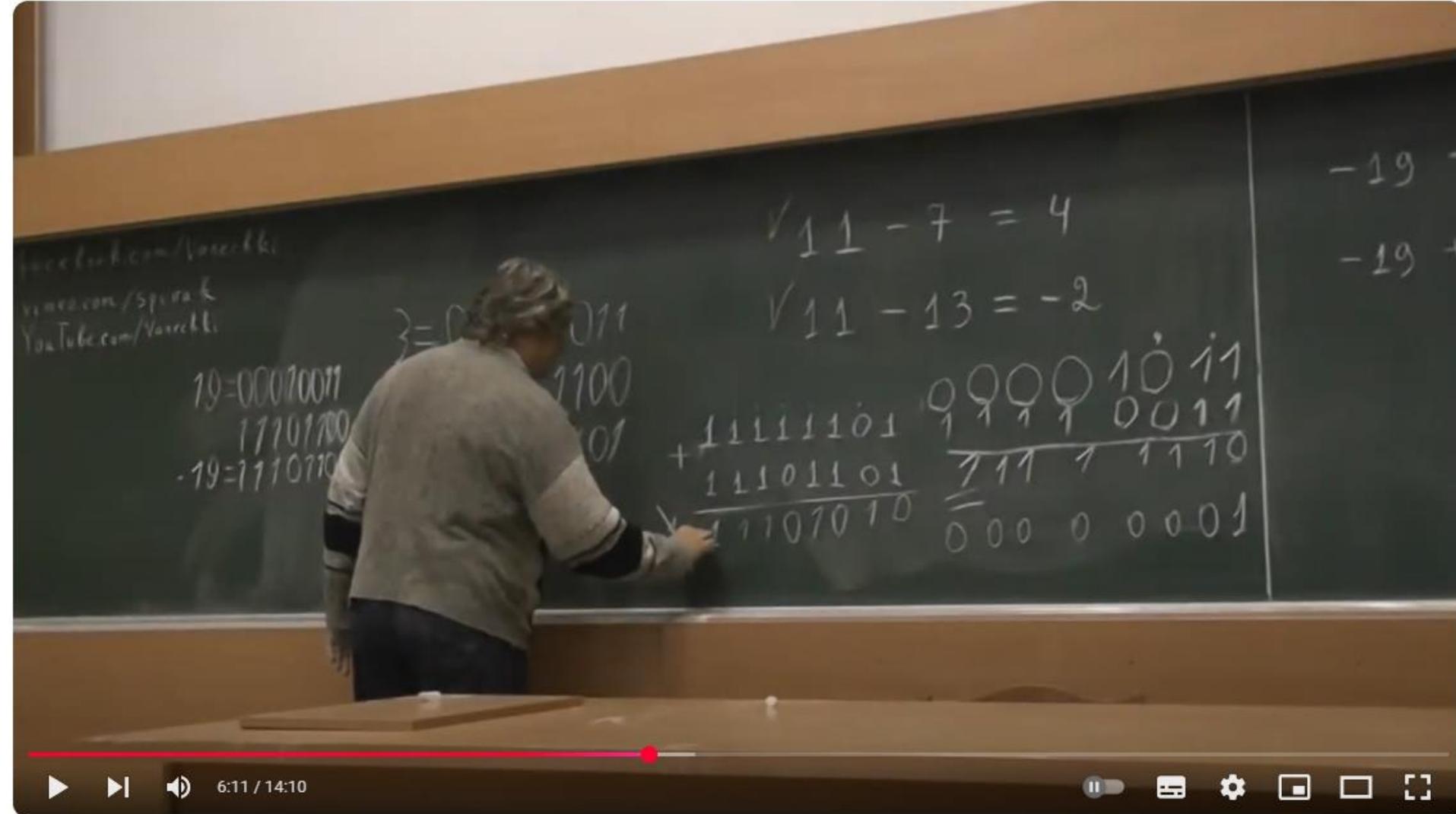
$$\begin{array}{r} 1011010 \\ \times 111011 \\ \hline 101101 \\ 101101 \\ 101101 \\ \hline 101001011110 \end{array}$$



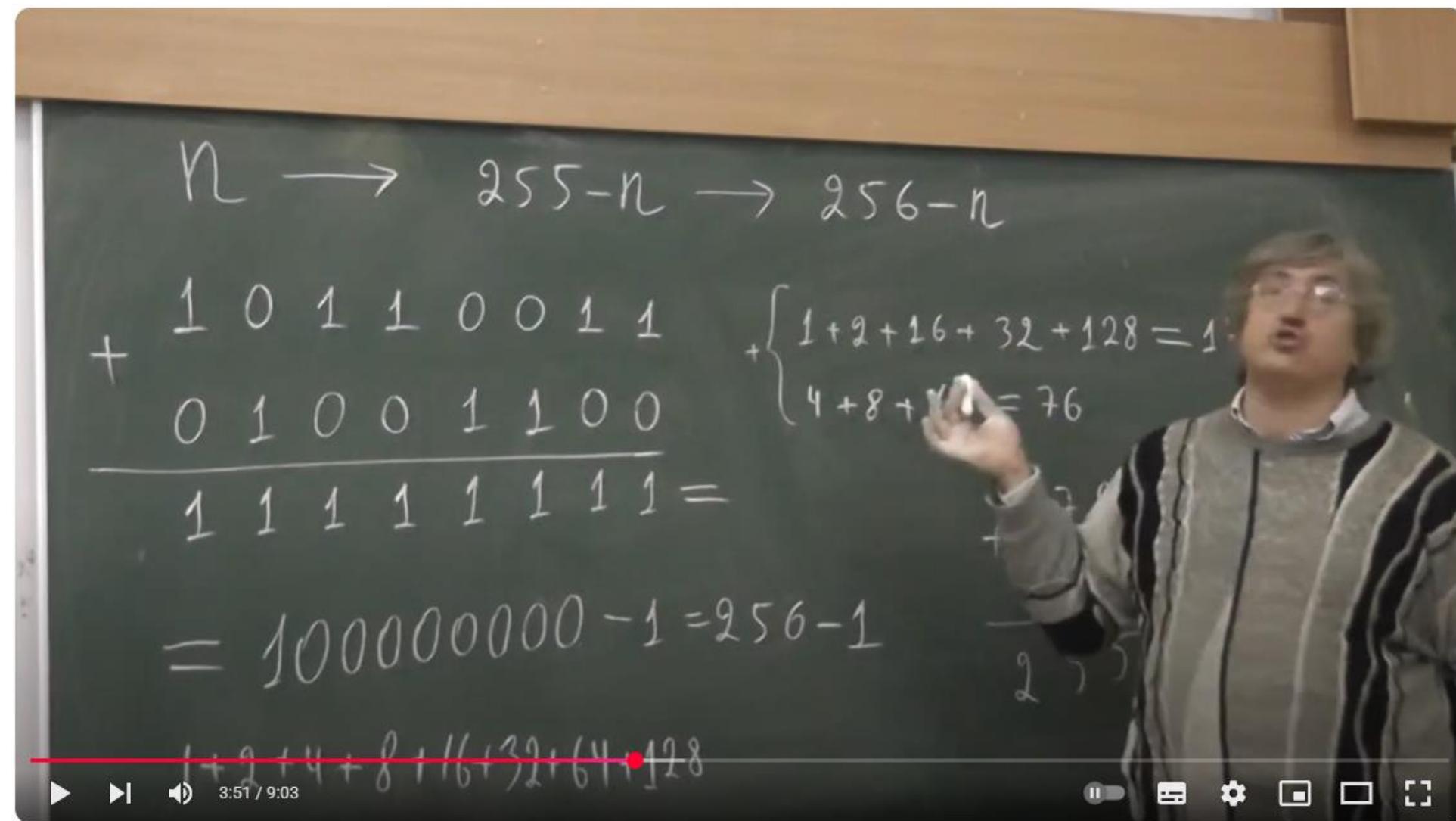
Системы счисления: Сложение, вычитание и умножение двоичных чисел (2014)  
<https://www.youtube.com/watch?v=DUFM11-n9Nc>



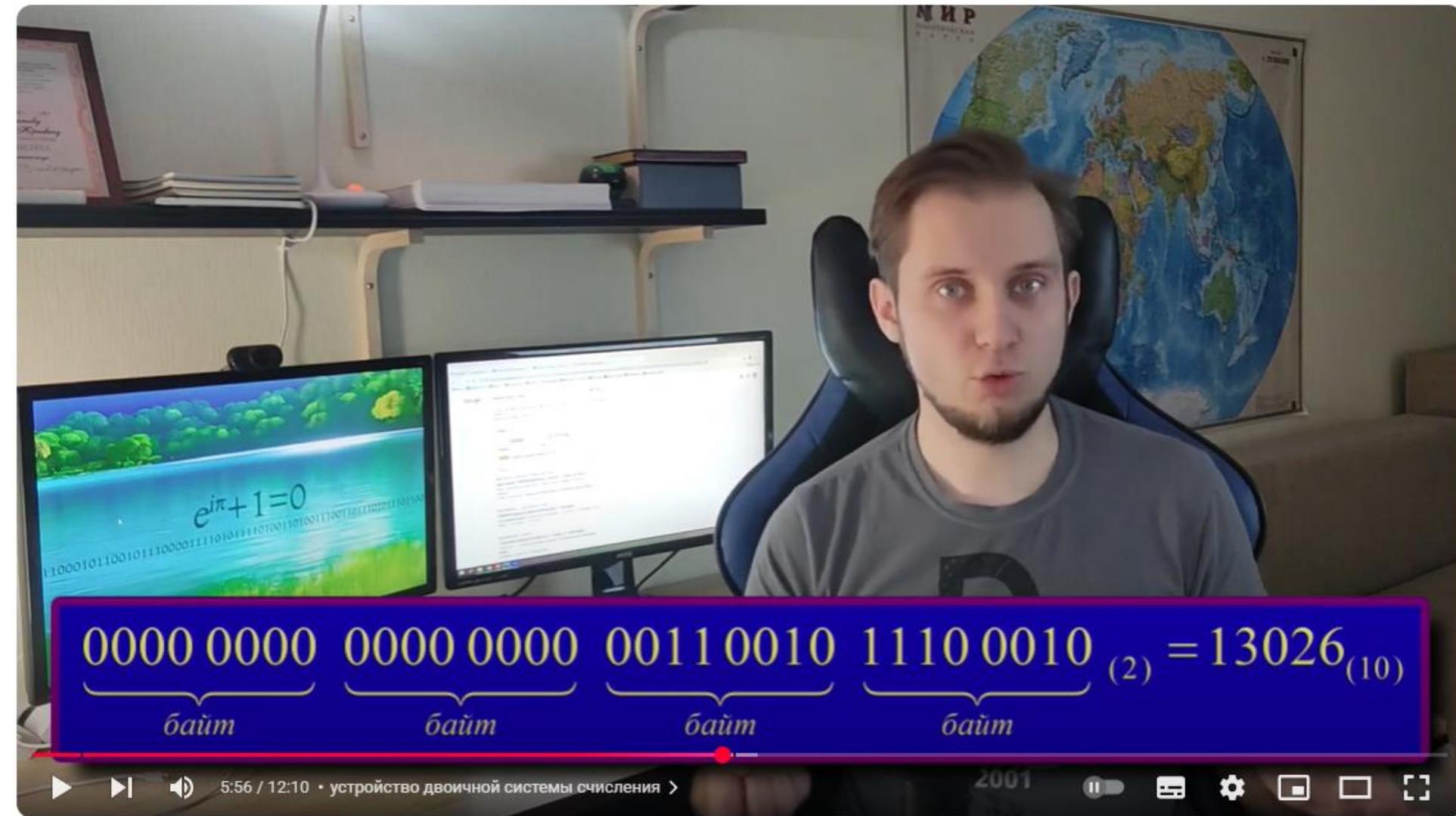
59 Запись отрицательных чисел, или Дополнительный код (2018)  
<https://www.youtube.com/watch?v=Jk7rg7SJxyl>



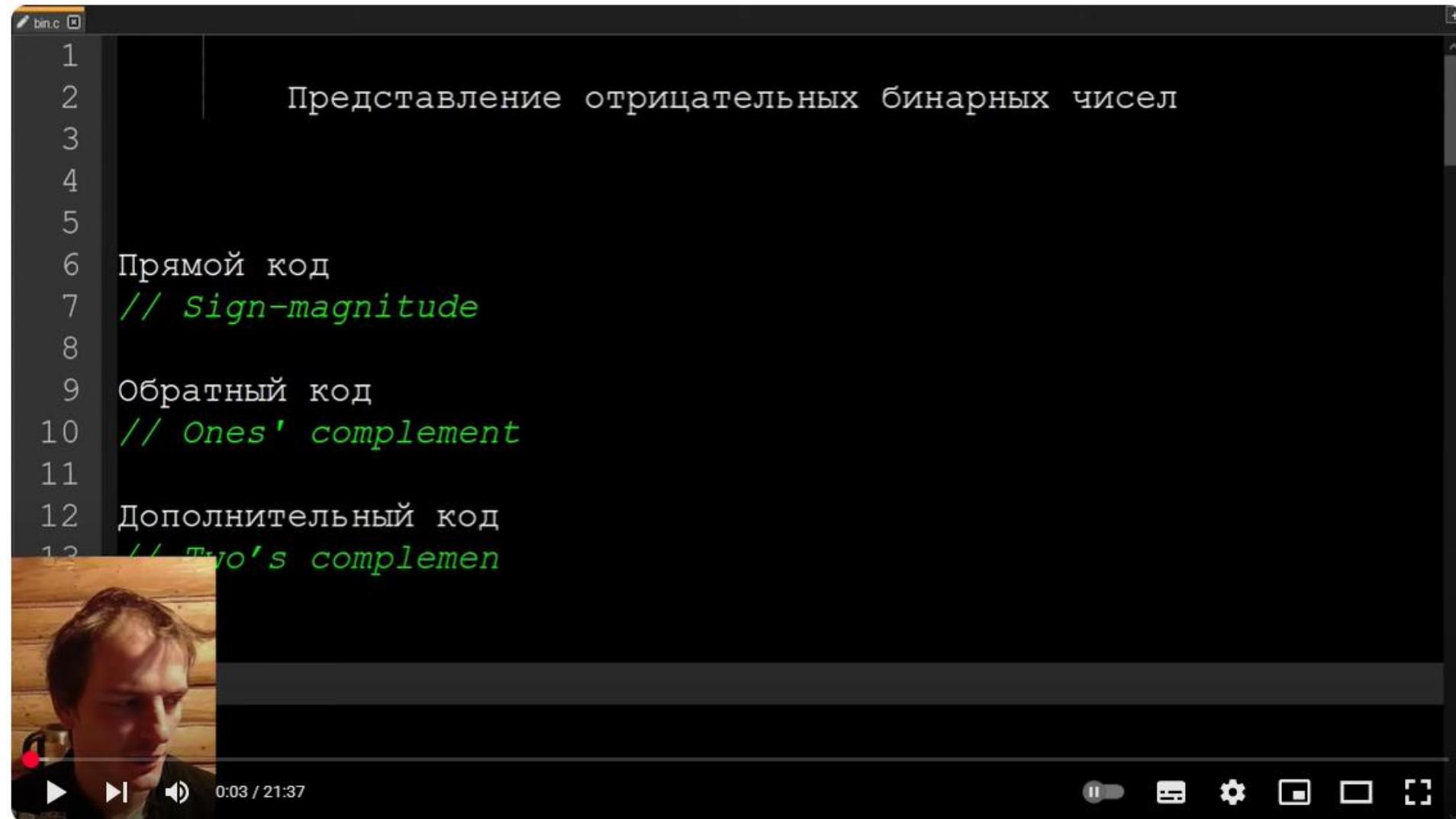
60 Примеры использования дополнительного кода (2018)  
[https://www.youtube.com/watch?v=uue5djB\\_CAI](https://www.youtube.com/watch?v=uue5djB_CAI)



61 Дополнительный код: инвертируем все биты и прибавляем 1 (2018)  
<https://www.youtube.com/watch?v=laNxfYfF6eo>



Как устроен двоичный код (2023)  
<https://www.youtube.com/watch?v=Ft7LvI9re0E>



Отрицательные двоичные числа: дополнительный и обратный код (2022)  
<https://www.youtube.com/watch?v=1CsmDrmmZcA>

The image shows a YouTube video player interface. In the top left corner, there is a red play button icon. To its right, the word "YouTube" is written in white, followed by the time "6:25". Below the video player, there is a set of four small icons: a plus sign (+), a minus sign (-), a multiplication symbol (×), and a division symbol (÷). To the right of these icons are two orange-bordered boxes containing binary arithmetic rules:

|                 |              |
|-----------------|--------------|
| $0 \cdot 0 = 0$ | $0 + 0 = 0$  |
| $0 \cdot 1 = 0$ | $0 + 1 = 1$  |
| $1 \cdot 0 = 0$ | $1 + 0 = 1$  |
| $1 \cdot 1 = 1$ | $1 + 1 = 10$ |

Below these boxes is a binary multiplication problem:

$$\begin{array}{r} & 1 & 1 & 1 & 1 \\ \times & 1 & 1 & 1 & 1 \\ \hline \end{array}$$

Underneath the multiplication problem is a binary addition problem:

$$\begin{array}{r} & 1 & 1 & 1 & 1 \\ + & 1 & 1 & 1 & 1 \\ \hline \end{array}$$

Below the addition problem is another binary addition problem:

$$\begin{array}{r} & 1 & 1 & 1 & 1 \\ + & 1 & 1 & 1 & 1 \\ \hline \end{array}$$

Below the second addition problem is a final binary addition problem:

$$\begin{array}{r} & 1 & 1 & 1 & 1 \\ + & 1 & 1 & 1 & 1 \\ \hline \end{array}$$

At the bottom of the video player, there is a progress bar with a red slider indicating the video is at 9:54 of 11:48. Below the progress bar are standard video control buttons: play/pause, volume, and settings.

Арифметические действия в двоичной системе счисления (2018)  
<https://www.youtube.com/watch?v=x92pfduxhqY>

# Сложение в двоичной системе счисления

| Таблица сложения |   |     |
|------------------|---|-----|
| a                | b | a+b |
| 0                | 0 | 0   |
| 0                | 1 | 1   |
| 1                | 0 | 1   |
| 1                | 1 | 10  |

$\begin{array}{r} 1 \\ + 1 \\ \hline 10 \end{array}$

$\begin{array}{r} 09_{10} \\ + 1_{10} \\ \hline 10_{10} \end{array}$

$\begin{array}{r} 10_2 \\ + 1_2 \\ \hline 10_2 \end{array} \rightarrow 10_2 = 1 \cdot 2^1 + 0 \cdot 2^0 = 2 + 0 = 2_{10}$

$0, 1, 10, 11, 100, 10$

$11111011 + 11100100 = 11001111$

Двоичная арифметика. Сложение и умножение чисел в двоичной системе счисления.  
8 класс Информатика (2020) <https://www.youtube.com/watch?v=4vtWVliX-3w>



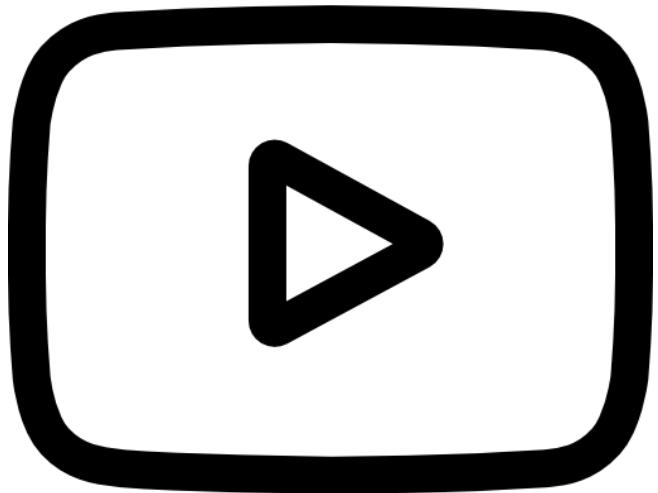
## Пример сложения в двоичной системе счисления

$$\begin{array}{r}
 1001 + 111 \\
 \hline
 0111
 \end{array}
 \quad \rightarrow \quad
 \begin{array}{r}
 + 1001 \\
 \hline
 0111
 \end{array}
 \quad \rightarrow \quad
 \begin{array}{r}
 + 1001 \\
 \hline
 0111
 \end{array}
 \quad \xrightarrow{\left\{ \begin{array}{l} 1+1=2 \geq 2 \\ 2 \% 2 = 0 \end{array} \right.}
 \quad \rightarrow \quad
 \begin{array}{r}
 + 1001 \\
 \hline
 0111
 \end{array}
 \quad \xrightarrow{\left\{ \begin{array}{l} 1+0+1=2 \geq 2 \\ 2 \% 2 = 0 \end{array} \right.}$$

Дополняем нулями

$$\begin{array}{r}
 + 1001 \\
 \hline
 0111
 \end{array}
 \quad \xrightarrow{\left\{ \begin{array}{l} 1+0+1=2 \geq 2 \\ 2 \% 2 = 0 \end{array} \right.}
 \quad
 \begin{array}{r}
 + 1001 \\
 \hline
 0111
 \end{array}
 \quad \xrightarrow{\left\{ \begin{array}{l} 1+0+1=2 \geq 2 \\ 2 \% 2 = 0 \end{array} \right.}
 \quad
 \begin{array}{r}
 + 1001 \\
 \hline
 0111
 \end{array}
 \quad \xrightarrow{\left\{ \begin{array}{l} 1+0+1=2 \geq 2 \\ 2 \% 2 = 0 \end{array} \right.}$$

Значения в десятичной системе счисления  
 $1001 = 9$   
 $111 = 7$   
 $10000 = 16$



# Булева алгебра



Законы алгебры логики

Простые:

$$\begin{array}{lll} A \cdot A = A & A + A = A & | A \cdot (A + B) = A \\ A \cdot \bar{A} = 0 & A + \bar{A} = 1 & | A + A \cdot B = A \end{array}$$

Свойства операций:

$$\begin{array}{lll} A \cdot B = B \cdot A & B = B + A \\ A \cdot (B \cdot C) = (A \cdot B) \cdot C & A + (B + C) = (A + B) + C \\ A \cdot (B + C) = A \cdot B + A \cdot C & B \cdot C = (A + B) \cdot (A + C) \end{array}$$

Законы отрицания:

$$\bar{\bar{A}} = A$$

Представление

$$A \rightarrow B = \bar{A} + B$$

XOR,  $\Rightarrow$ ,  $\Leftrightarrow$ :

$$\begin{array}{ll} A \cdot B + \bar{A} \cdot \bar{B} \\ A \cdot \bar{B} + \bar{A} \cdot B \end{array}$$

Progress bar: 1:00 / 13:02

Control icons: play, pause, volume, settings, full screen, exit full screen.

Алгебра логики: Законы алгебры логики. (2014)  
<https://www.youtube.com/watch?v=7XA77xNVBv4>

## Операции алгебры логики

1) НЕ

 $\neg A$ 

2) И

 $A \wedge B$  $A \vee B \wedge \neg C \wedge D \rightarrow E$ 

3) ИЛИ

 $A \vee B$ 

4) XOR

 $A \oplus B$ 

5) импликация

 $A \rightarrow B$ 

6) эквиваленция

 $A \leftrightarrow B$ 

1:51 / 4:34

Информатика. Алгебра логики: Операции алгебры логики.(2014)  
<https://www.youtube.com/watch?v=3yAEUKy68lw>

HE

| A | $\neg A$ |
|---|----------|
| 0 | 1        |
| 1 | 0        |

ИМПЛИКАЦИЯ

| A | B | $A \rightarrow B$ |
|---|---|-------------------|
| 0 | 0 | 1                 |
| 0 | 1 | 1                 |
| 1 | 0 | 0                 |
| 1 | 1 | 1                 |

ИЛИ

| A | B | $A \vee B$ |
|---|---|------------|
| 0 | 0 | 0          |
| 0 | 1 | 0          |
| 1 | 0 | 0          |
| 1 | 1 | 1          |

ИСКЛЮЧАЮЩИЙ ИЛИ

| A | B | $A \oplus B$ |
|---|---|--------------|
| 0 | 0 | 0            |
| 0 | 1 | 1            |
| 1 | 0 | 1            |
| 1 | 1 | 0            |

2:02 / 10:15

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100

Информатика. Алгебра логики: Таблицы истинности (2014)  
<https://www.youtube.com/watch?v=qrj6Ekwqr-c>

Представление целых чисел в памяти ПК

Беззнаковое:  $01100011_2$   $\begin{array}{r} -99 \\ 98 \end{array} \quad \begin{array}{r} 2 \\ 49 \\ 1 \end{array}$

$0\dots255$

Со знаком:

прямой код:  $11100011_2$  *знаковый бит*

$-127\dots127$

обратный код:  $10011100_2$   $-127\dots127$

дополнительный код:  $10011101_2$

$99 - 98 = 1$

$011_2 - 98 = 1$

Информатика. Архитектура ПК: Представление целых чисел в памяти ПК (2014)  
<https://www.youtube.com/watch?v=g6Y86fAqXEY>

The image shows a screenshot of a YouTube video player. The main content is a presentation slide with a light gray background. In the center, there is large blue text that reads "Логические основы компьютеров". Below this, in black text, is "§ 16. Логика и компьютер". At the bottom of the slide, there is a dark footer bar containing a play button, a progress bar showing "0:02 / 7:35", the text "© Ю. Поляков, Е. А. Ерёмин, 2018", and a URL "http://kpoliakov.spb.ru". To the right of the slide, there is a vertical black sidebar. At the top of this sidebar, there is a small video frame showing a woman with short dark hair, labeled with the number "2". The YouTube interface includes a red play button icon in the top left corner of the video area.

# Логические основы компьютеров

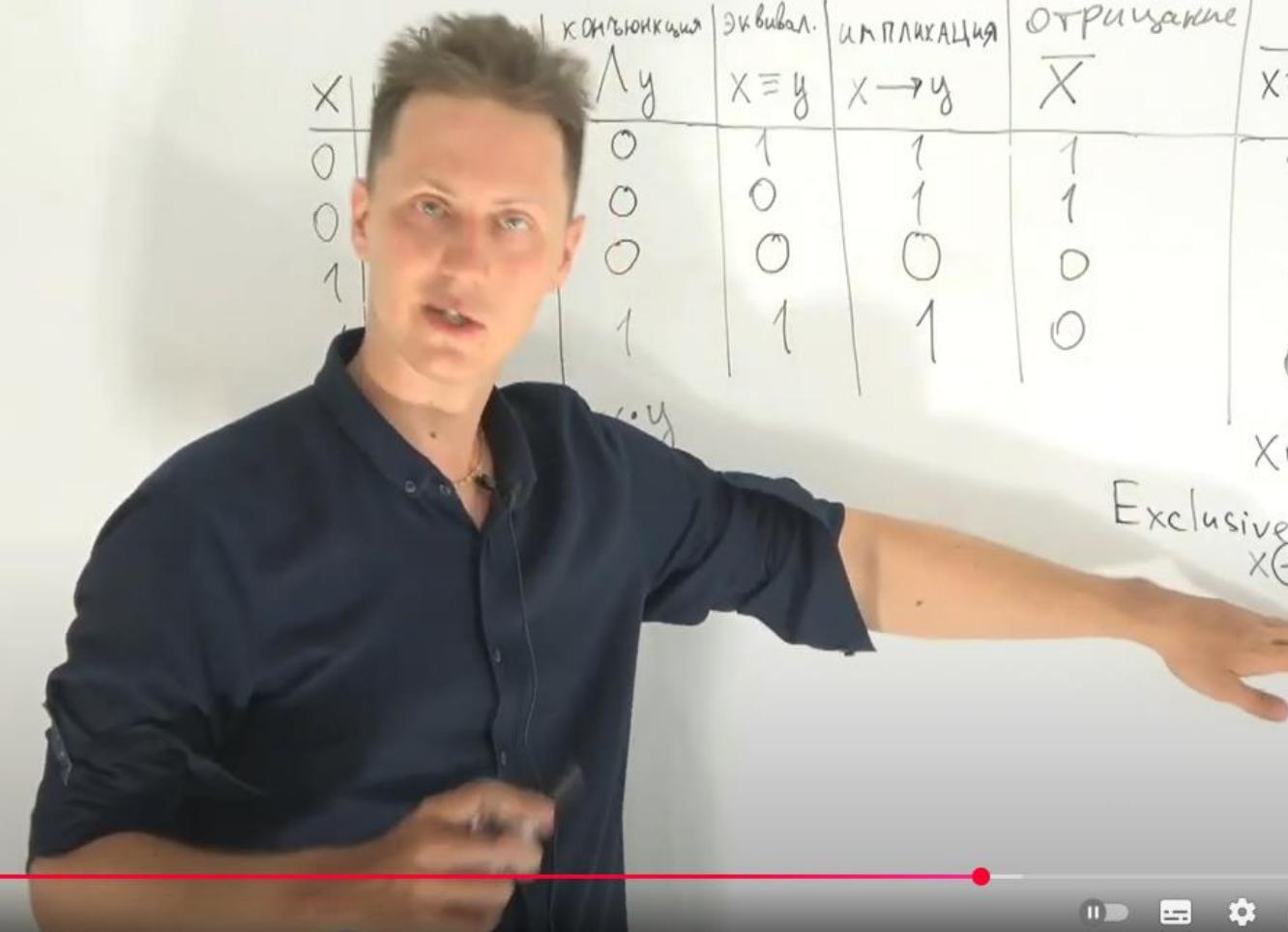
## § 16. Логика и компьютер

Урок 21. Логические операции "И", "ИЛИ", "НЕ", "исключающее ИЛИ". ИКТ 10 класс по Полякову (2021) <https://www.youtube.com/watch?v=YLz1KOtZs8w>

АЛГЕБРА ЛОГИКИ

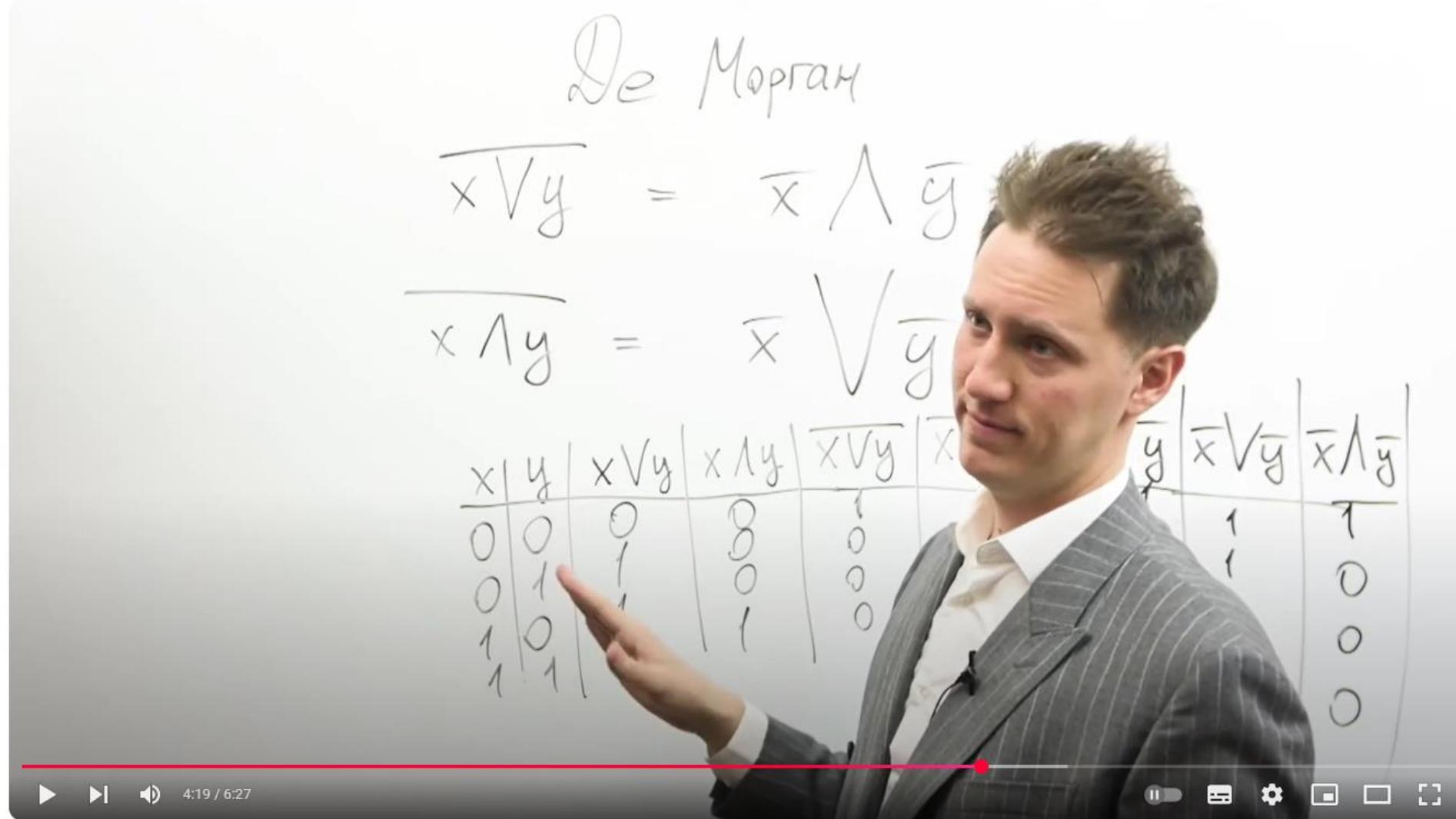
| X | Y | ЛУ | $X \equiv Y$ | $X \rightarrow Y$ | $\bar{X}$ | $\bar{X} \equiv Y$ |
|---|---|----|--------------|-------------------|-----------|--------------------|
| 0 | 0 | 1  | 1            | 1                 | 1         | 0                  |
| 0 | 1 | 0  | 0            | 1                 | 1         | 1                  |
| 1 | 0 | 0  | 0            | 0                 | 0         | 1                  |
| 1 | 1 | 1  | 1            | 1                 | 0         | 0                  |

$X \oplus Y$   
Exclusive OR  
 $X \oplus Y$

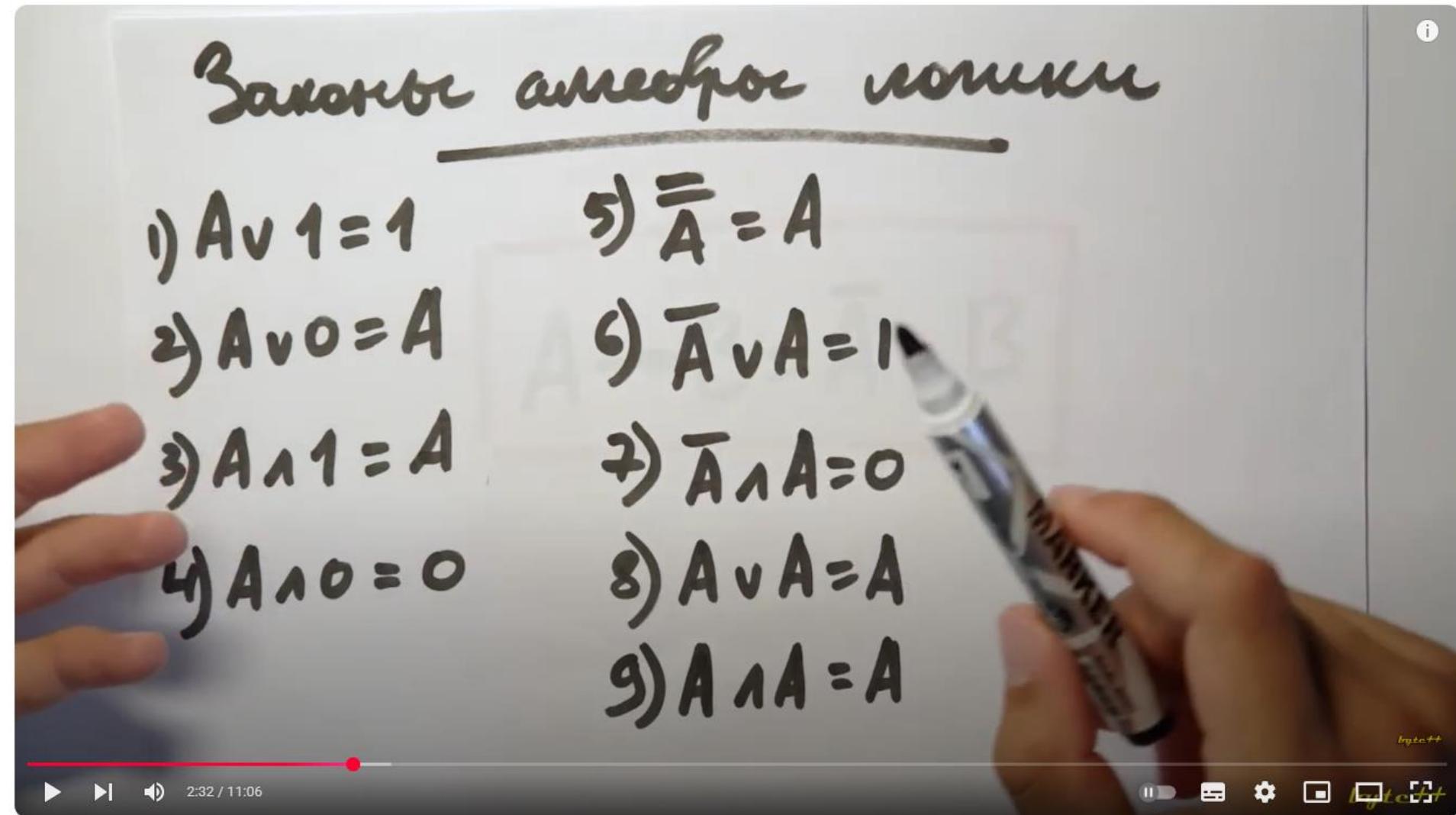


11:32 / 15:52

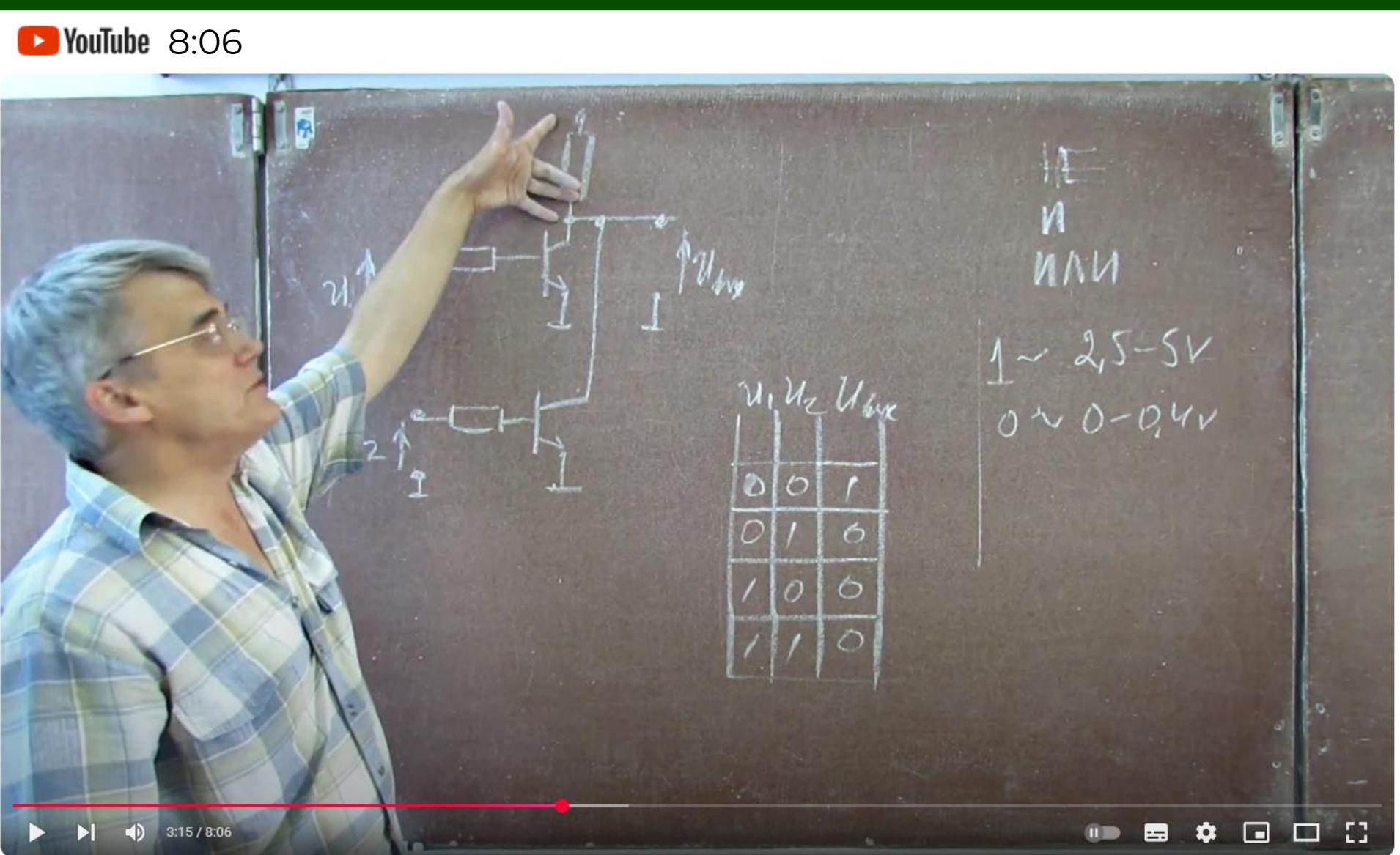
50 уроков Информатики: Алгебра логики (2021)  
[https://www.youtube.com/watch?v=\\_x\\_Vhwt384M](https://www.youtube.com/watch?v=_x_Vhwt384M)



Законы де Моргана | 13/50 урок Информатики | Школково (2022)  
[https://www.youtube.com/watch?v=tO6HBxAt\\_fw](https://www.youtube.com/watch?v=tO6HBxAt_fw)

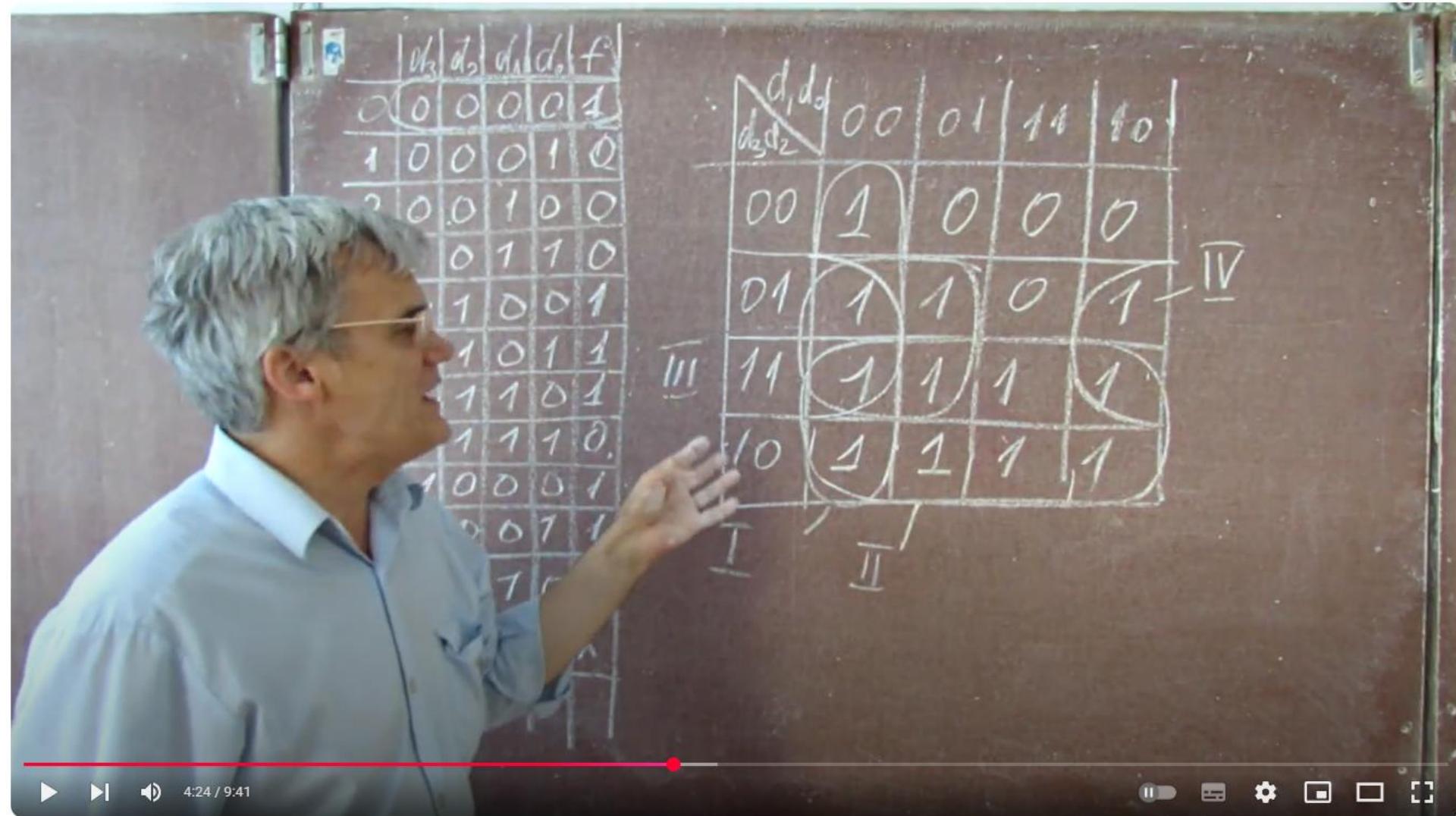


Законы алгебры логики / Закон де Моргана + доказательство [Алгебра логики] #5 (2020)  
<https://www.youtube.com/watch?v=yUP59rLpFZw>

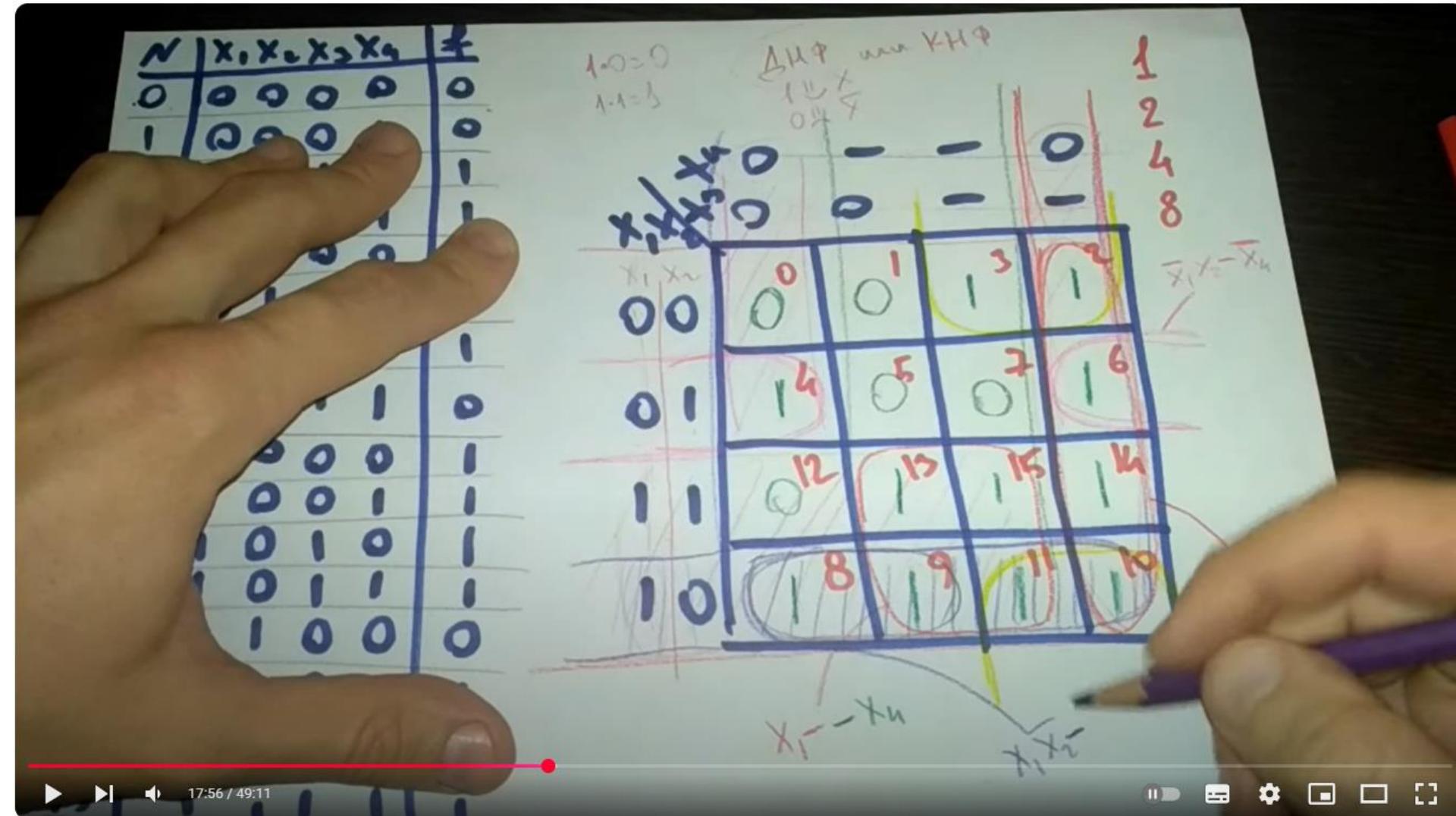


## Лекция 67. Теорема де Моргана (2013)

<https://www.youtube.com/watch?v=Sv64kxLGBFc>



Лекция 80. Карта Карно (2013)  
<https://www.youtube.com/watch?v=a37anDvo0bs>



Карты Карно. Как они работают. Большой выпуск (2019)  
<https://www.youtube.com/watch?v=wIEiX9ROSoE>

## ПОСТРОИТЬ ТАБЛИЦУ ИСТИННОСТИ ДЛЯ ЛОГИЧЕСКОГО ВЫРАЖЕНИЯ

$$F = A^1 \cdot B^4 + A^2 \cdot C^5 + B^3 \cdot C$$

1. КОЛИЧЕСТВО СТРОК?  $2^N + 1$   $N = 3$   $2^3 + 1 = 8 + 1 = 9$

2. КОЛИЧЕСТВО СТОЛБЦОВ?  $N + O$   $N = 3$   $O = 5$   $N + O = 3 + 5 = 8$

3. СТРОИМ ТАБЛИЦУ  $8 \times 9$ :

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



Построение таблиц истинности (2018)  
<https://www.youtube.com/watch?v=R5iuMQFPmI8>

## Построение таблиц истинности

$$\begin{matrix} 2 & 1 \\ \mathbf{A} \vee \mathbf{B} \wedge \mathbf{A} \end{matrix}$$

- 1) К - количество переменных 2
- 2) О - количество операций и их порядок 2
- 3) Количество строк таблицы =  $2^k$  4
- 4) Количество столбцов таблицы =  $K+O$  4
- 5) Заполняем таблицу



## Построение таблиц истинности

i

A, B, C  
0 1  
ложь истина

$\bar{A} \wedge B \vee C$

| A | B | C | $\bar{A}$ | $\bar{A} \wedge B$ | $\bar{A} \wedge B \vee C$ |
|---|---|---|-----------|--------------------|---------------------------|
| 0 | 0 | 0 | 1         | 0                  | 0                         |
| 0 | 0 | 1 | 1         | 0                  | 1                         |
| 0 | 1 | 0 | 1         | 1                  | 1                         |
| 0 | 1 | 1 | 1         | 1                  | 0                         |
| 1 | 0 | 0 | 0         | 0                  | 0                         |
| 1 | 0 | 1 | 0         | 0                  | 0                         |
| 1 | 1 | 0 | 0         | 0                  | 0                         |
| 1 | 1 | 1 | 0         | 0                  | 0                         |



▶ ▶ 🔍 5:28 / 6:06 • Построение таблицы истинности >

Os

|| ⏸ ⏹ ⚙ ☐ ☐ ☐ ☐ ☐ ☐

ИНФОРМАТИКА 8 класс: Построение таблиц истинности для логических выражений (2018) <https://www.youtube.com/watch?v=vWLGKw0U5TU>

YouTube 14:19

| $c_i \backslash a \backslash b$ | $v \downarrow c_o$ |
|---------------------------------|--------------------|
| 000                             | 00                 |
| 001                             | 10                 |
| 010                             | 10                 |
| 011                             | 01                 |
| 100                             | 10                 |
| 101                             | 01                 |
| 110                             | 01                 |
| 111                             | 11                 |

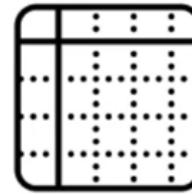
$V:$

| $c_i \backslash a \backslash b$ | 00 | 01 | 11 |
|---------------------------------|----|----|----|
| 00                              | 00 | 01 | 10 |
| 01                              | 01 | 10 | 11 |
| 11                              | 10 | 11 | 00 |

$c_o:$

| $c_i \backslash a \backslash b$ | 00 | 01 | 11 |
|---------------------------------|----|----|----|
| 00                              | 00 | 01 | 10 |
| 01                              | 01 | 10 | 11 |
| 11                              | 10 | 11 | 11 |

Минимизация функций. Карты Карно. Цифровая техника (2016)  
[https://www.youtube.com/watch?v=Fyruie9\\_uDKk](https://www.youtube.com/watch?v=Fyruie9_uDKk)



## Построение таблиц истинности для логических выражений

подсчитать  $n$  - число переменных в выражении

подсчитать общее число логических операций в выражении

установить последовательность выполнения логических операций

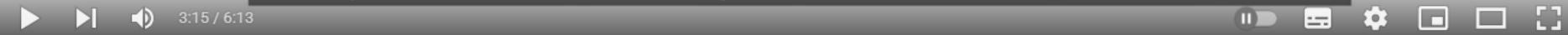
определить число столбцов в таблице

заполнить шапку таблицы, включив в неё переменные и операции

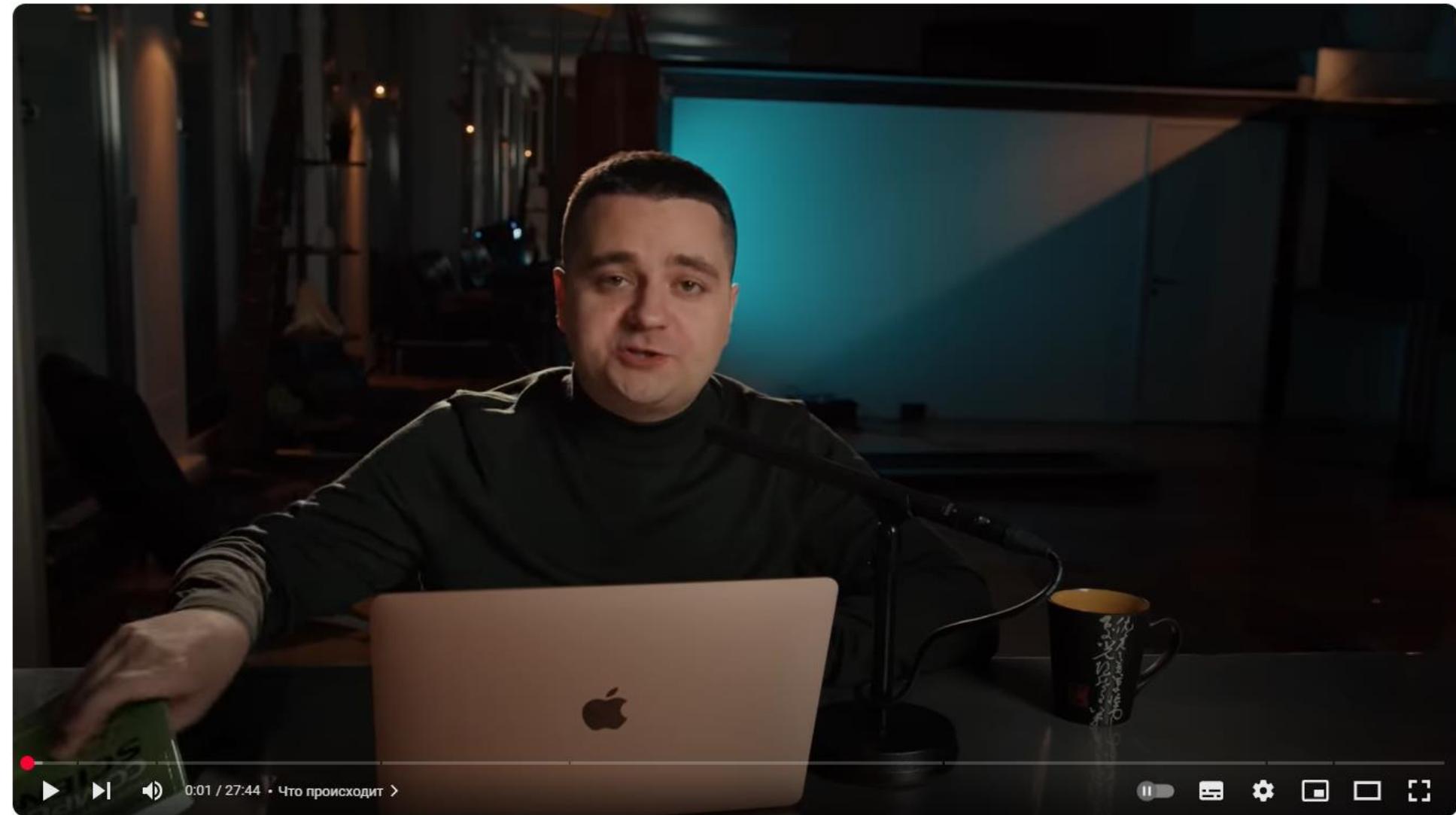
определить число строк в таблице без шапки:  $m = 2^n$

выписать наборы входных переменных

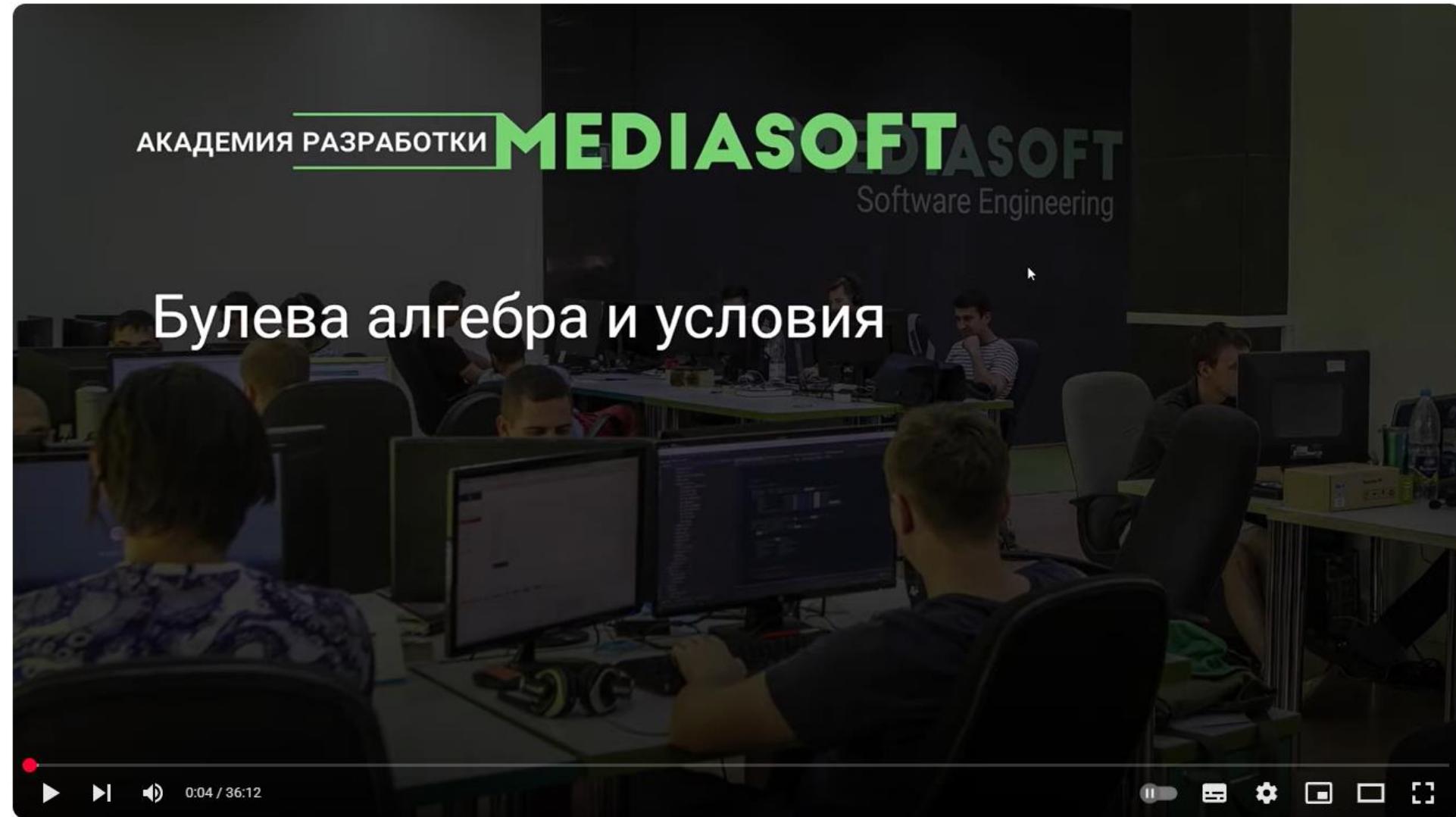
проводить заполнение таблицы по столбцам, выполняя логические операции в соответствии с установленной последовательностью



Информатика 8 класс. Таблицы истинности (УМК БОСОВА Л.Л.,  
БОСОВА А.Ю.) (2020) <https://www.youtube.com/watch?v=jynqE6QMiHw>



Теоретический минимум по CS: логические вентили, булева алгебра, сумматоры.  
Читаем главу 1 «Основы» (2022) <https://www.youtube.com/watch?v=s7g7ZUCZ2q8>

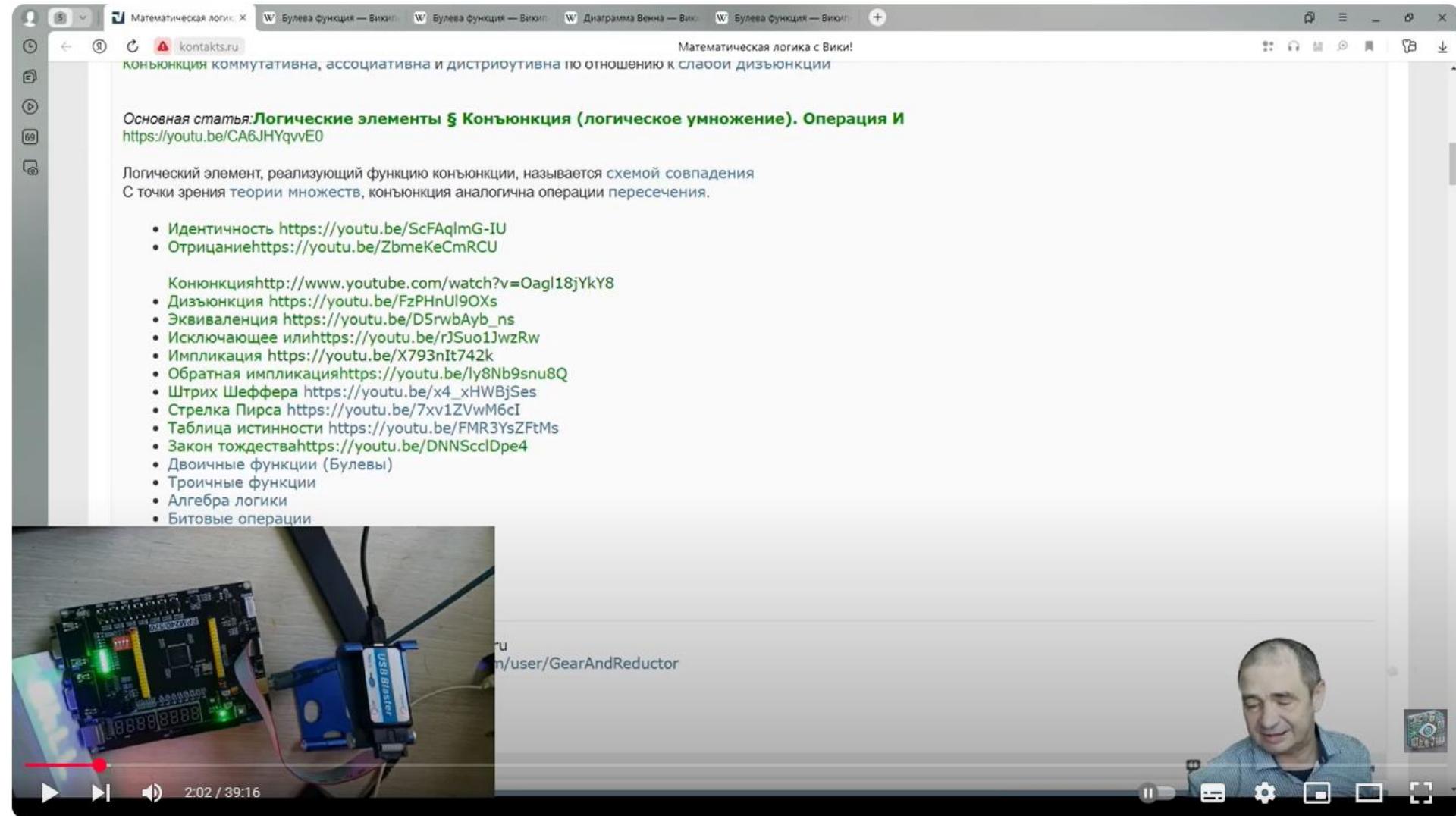


Лекция №6 / Основы программирования / Булева алгебра и условия (2021)  
<https://www.youtube.com/watch?v=EDEIPk2Y5Zc>

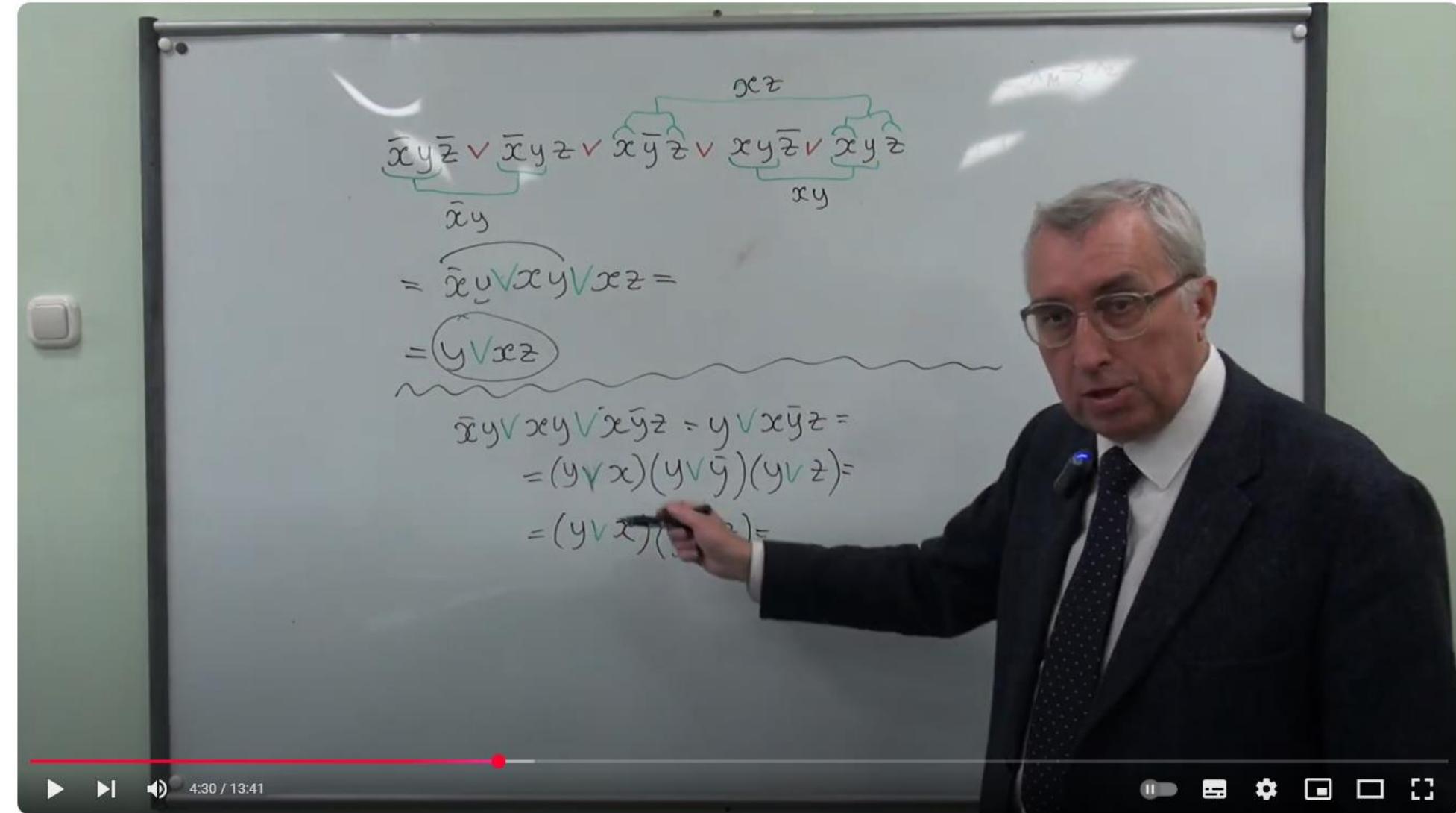
| $f(x,y,z)$  | $g(x,y,z)$                      | $f(x,y,z)$  | $g(x,y,z)$                             |
|---|---------------------------------|---|--|
| $\bar{xyz} \downarrow y$                                  | $\bar{xz} \leftarrow y$         | $\bar{xyz} \oplus y$                                      | $\bar{xz} \leftarrow y$                |
| $\bar{\bar{y} \vee x} \rightarrow \bar{xyz}$              | $x \rightarrow \bar{yz}$        | $\bar{\bar{y} \rightarrow x} \downarrow \bar{yz}$         | $x \downarrow \bar{yz}$                |
| $(\bar{yz} \leftarrow x) \vee \bar{y \vee z}$             | $\bar{yz} \oplus x$             | $(\bar{yz} \oplus x) \leftarrow \bar{x \vee y \vee z}$    | $\bar{yz} \rightarrow x$               |
| $\bar{\bar{x \vee y \vee xyz} \oplus y}$                  | $xz \oplus y$                   | $\bar{\bar{x \vee y}} \downarrow xyz \rightarrow y$       | $xz \leftarrow y$                      |
| $\bar{\bar{x \vee y}} \downarrow \bar{xy} \rightarrow z$  | $z \downarrow \bar{xy}$         | $\bar{\bar{x \oplus y}} \downarrow z \rightarrow x$       | $x \downarrow z$                       |
| $\bar{yz} \vee \bar{x \vee y \vee z} \oplus x$            | $\bar{yz} \oplus x$             | $y \rightarrow \bar{z} \oplus x$                          | $\bar{yz} \downarrow x$                |
| $\bar{xz} \rightarrow \bar{yz} \leftarrow \bar{x \vee y}$ | $z \rightarrow y \leftarrow x$  | $\bar{xz} \leftarrow \bar{yz} \rightarrow \bar{x \vee y}$ | $z \oplus y \rightarrow x$             |
| $\bar{yz} \vee \bar{xyz} \rightarrow x$                   | $\bar{yz} \vee z \rightarrow x$ | $\bar{yz} \vee \bar{xyz} \rightarrow x$                   | $\bar{yz} \rightarrow z \rightarrow x$ |
| $\bar{y \vee z} \oplus \bar{x \vee y \vee z}$             | $y \oplus \bar{x \vee z}$       | $\bar{y \vee z} \oplus \bar{x \vee y \vee z}$             | $y \downarrow \bar{x \vee z}$          |
| $\bar{xyz} \downarrow x \downarrow z$                     | $y \downarrow x \rightarrow z$  | $\bar{xyz} \downarrow x \downarrow z$                     | $y \downarrow x \oplus z$              |
| 30  |                                 |   |  |



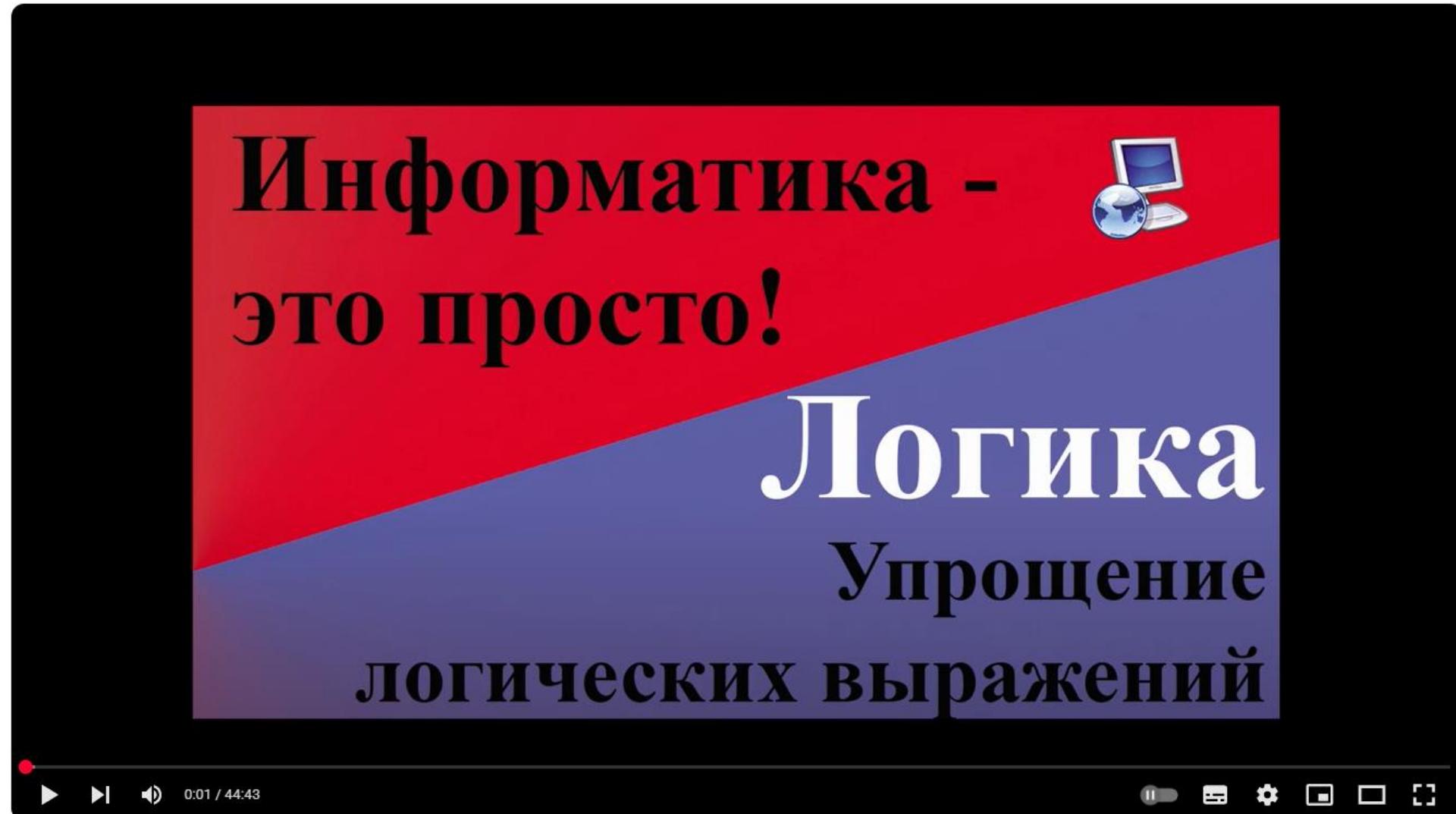
Решение логических выражений | 30 примеров | Булева алгебра | Гайд (2024)  
[https://www.youtube.com/watch?v=\\_HqeGGpJyCw](https://www.youtube.com/watch?v=_HqeGGpJyCw)



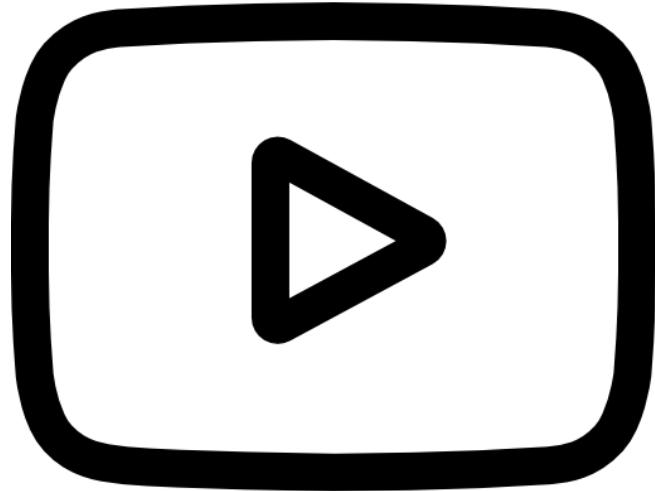
Двоичные функции (Булевы функции). Математическая логика с Вики! (2024)  
<https://www.youtube.com/watch?v=Xz36ONw1QCE>



Три способа упрощения логической функции (2014)  
<https://www.youtube.com/watch?v=5U4P56A2ePY>



Логика - Упрощение логических выражений. Законы алгебры логики(2018)  
<https://www.youtube.com/watch?v=sNI5dB8I1qc>



# Цифровая логика. Базовые логические элементы



## Логические элементы

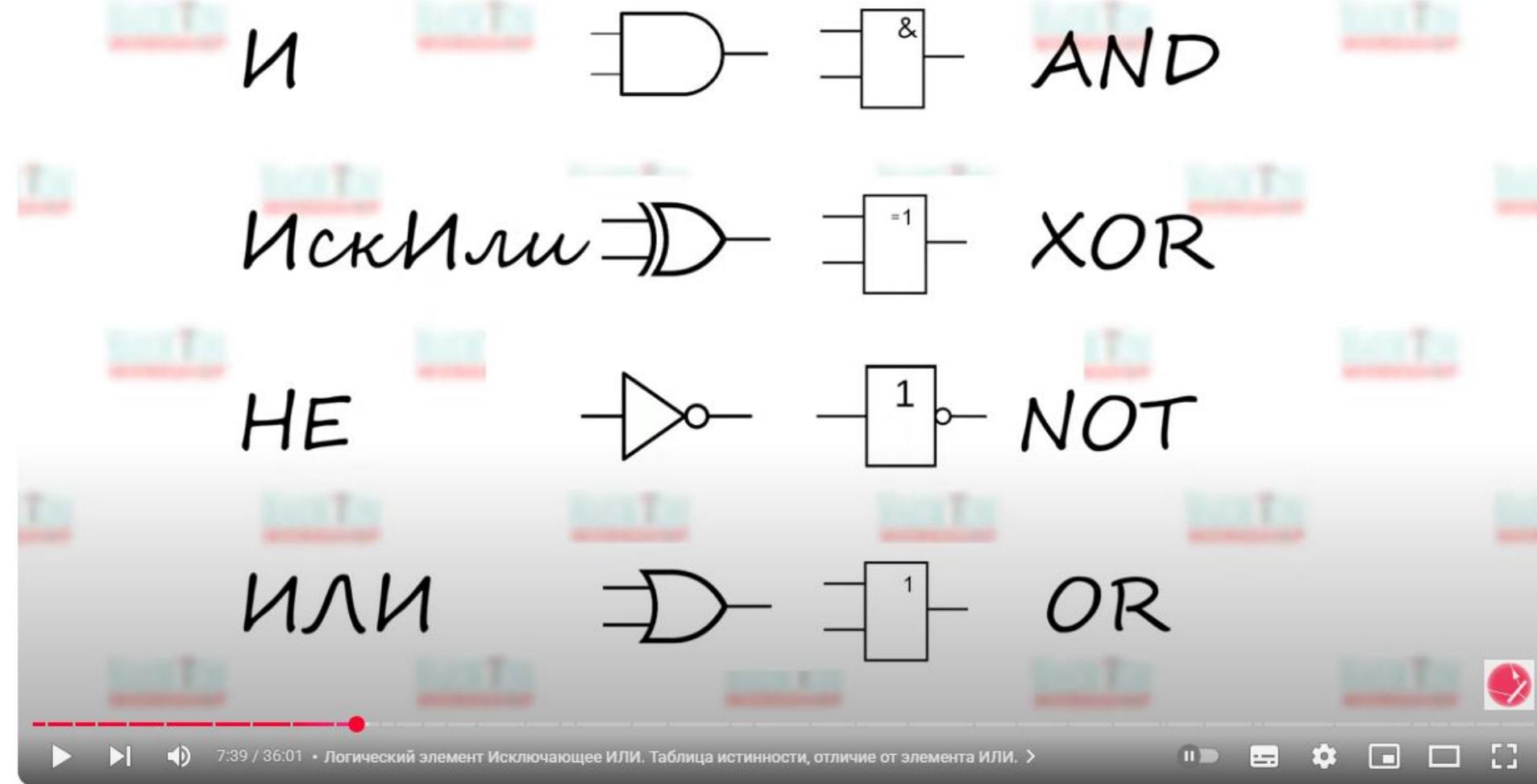
Математические основы информатики



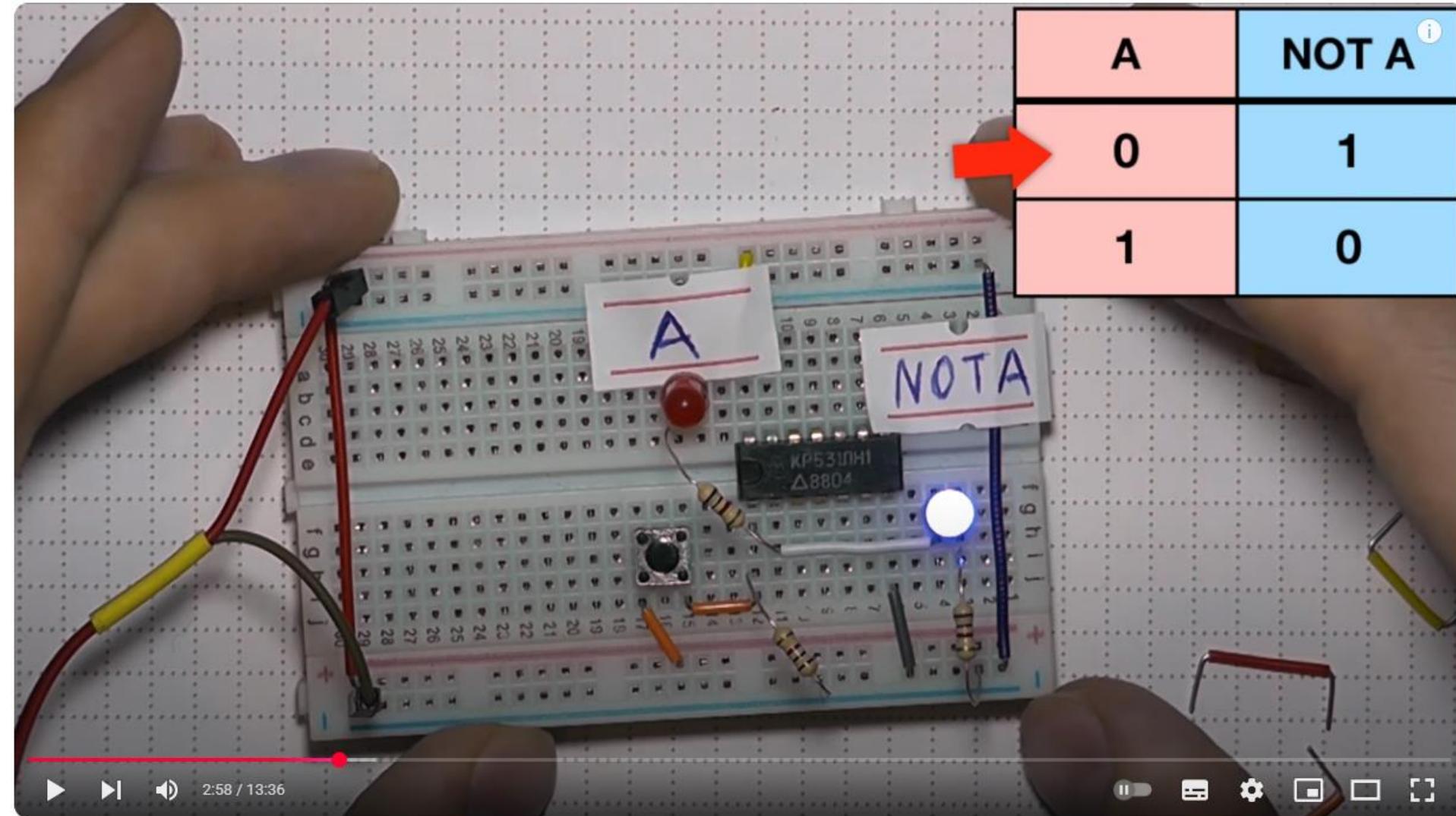
0:02 / 15:44



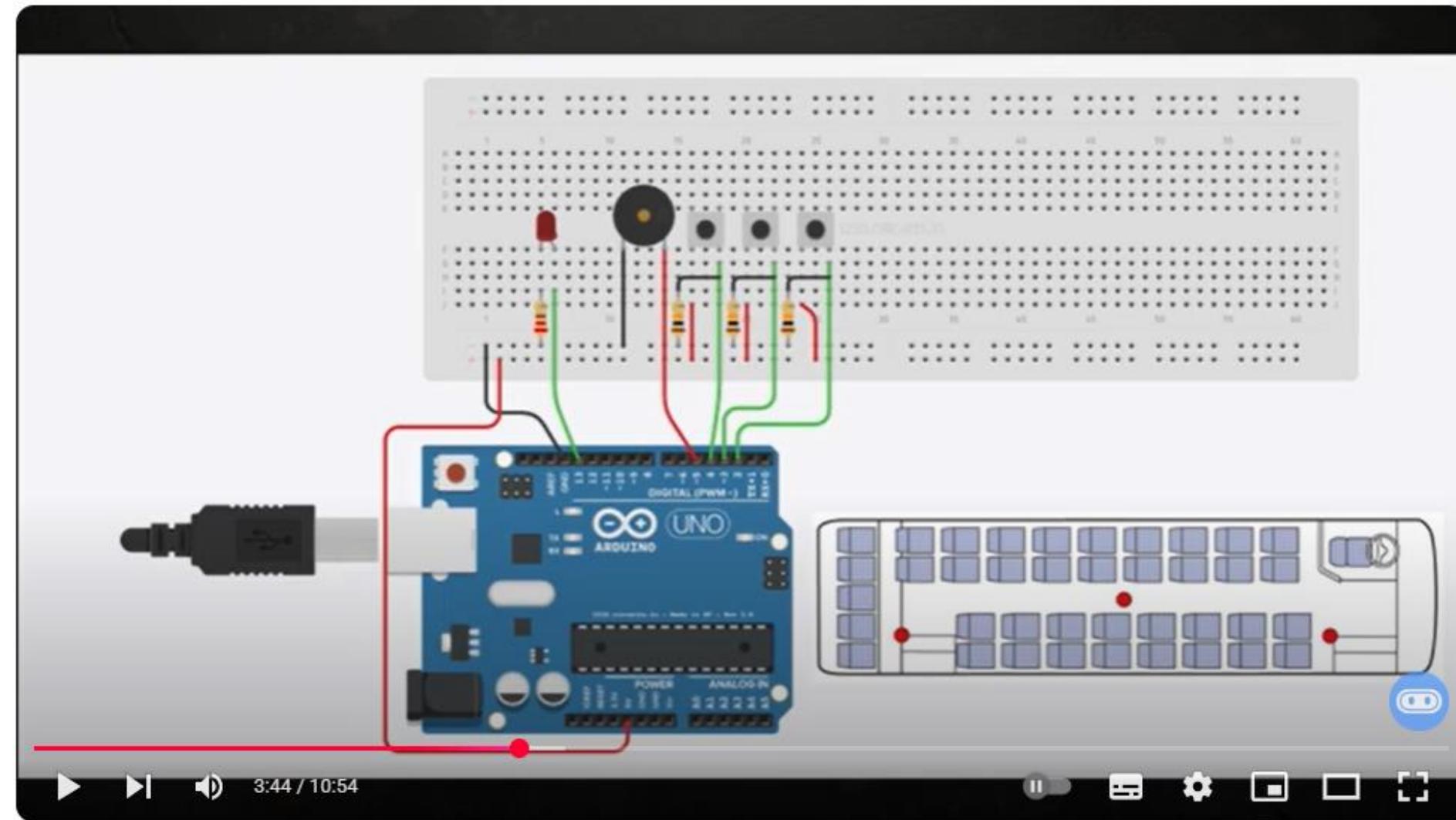
Логические элементы (2017)  
<https://www.youtube.com/watch?v=3JCR15YHbJU>



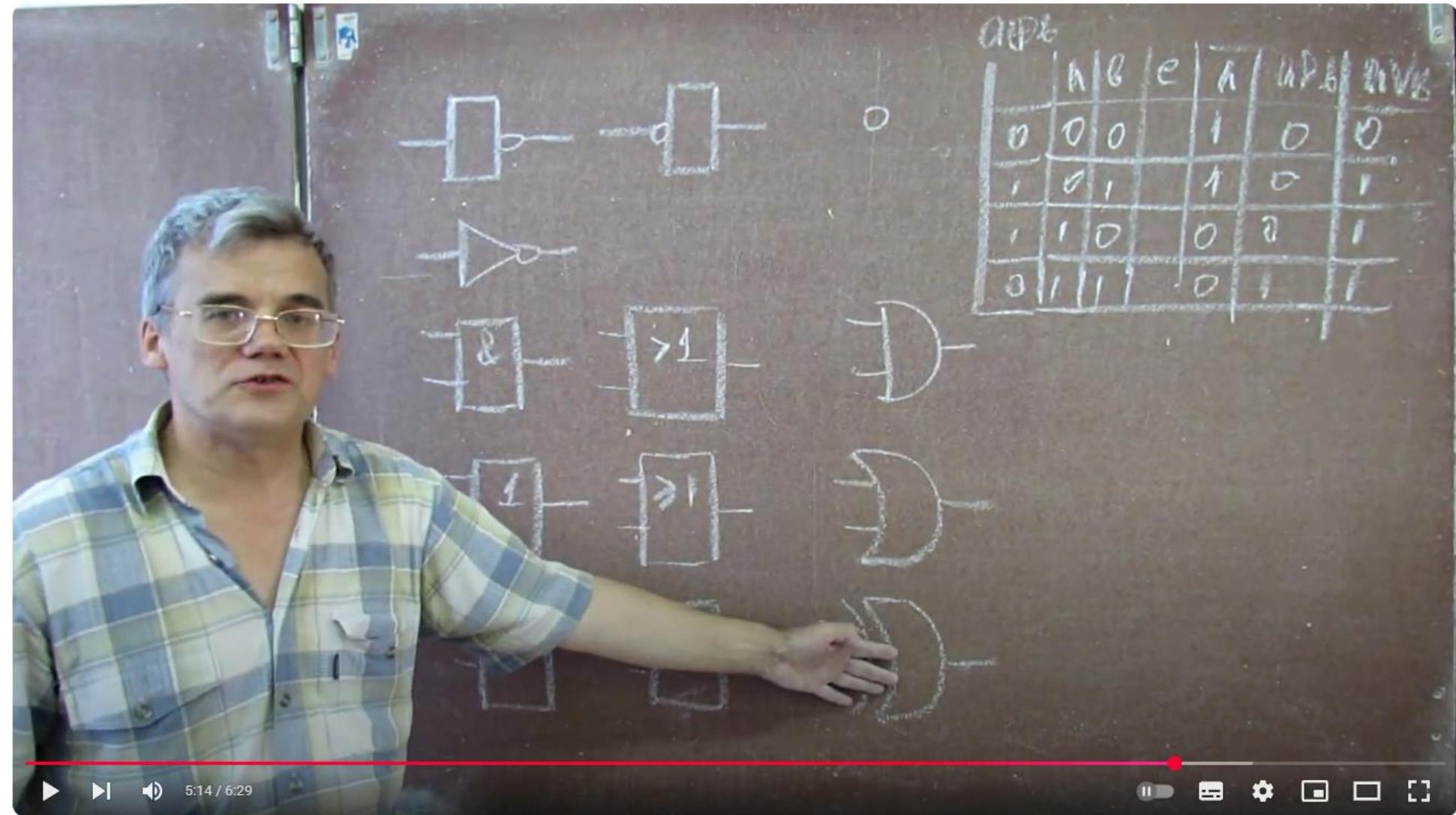
Логические элементы И, ИЛИ, Исключающее ИЛИ. История, Теория, Применение (2021) <https://www.youtube.com/watch?v=bXdiYU3IUJA>



[Электроника] Логические Элементы, И, ИЛИ, НЕ, подробный обзор, и тестирование на микросхемах! (2018) <https://www.youtube.com/watch?v=rva16jfbdWE>



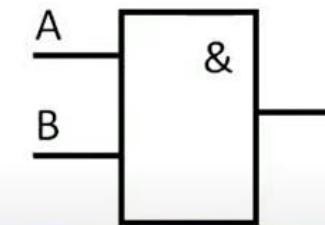
Должен знать каждый программист - логические операции И, ИЛИ, НЕ. Уроки Arduino для начинающих (2016) <https://www.youtube.com/watch?v=vSw5P4Ey9zc>



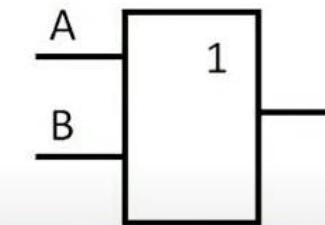
Лекция 71. Обозначение логических элементов (2013)  
<https://www.youtube.com/watch?v=s5MQoR-LN34>

## Логические элементы

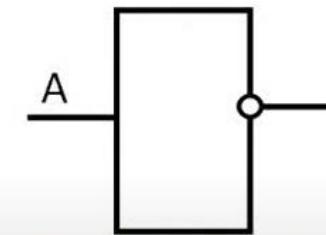
**Логический элемент** – устройство, которое после обработки двоичных сигналов выдаёт значение одной из логических операций.



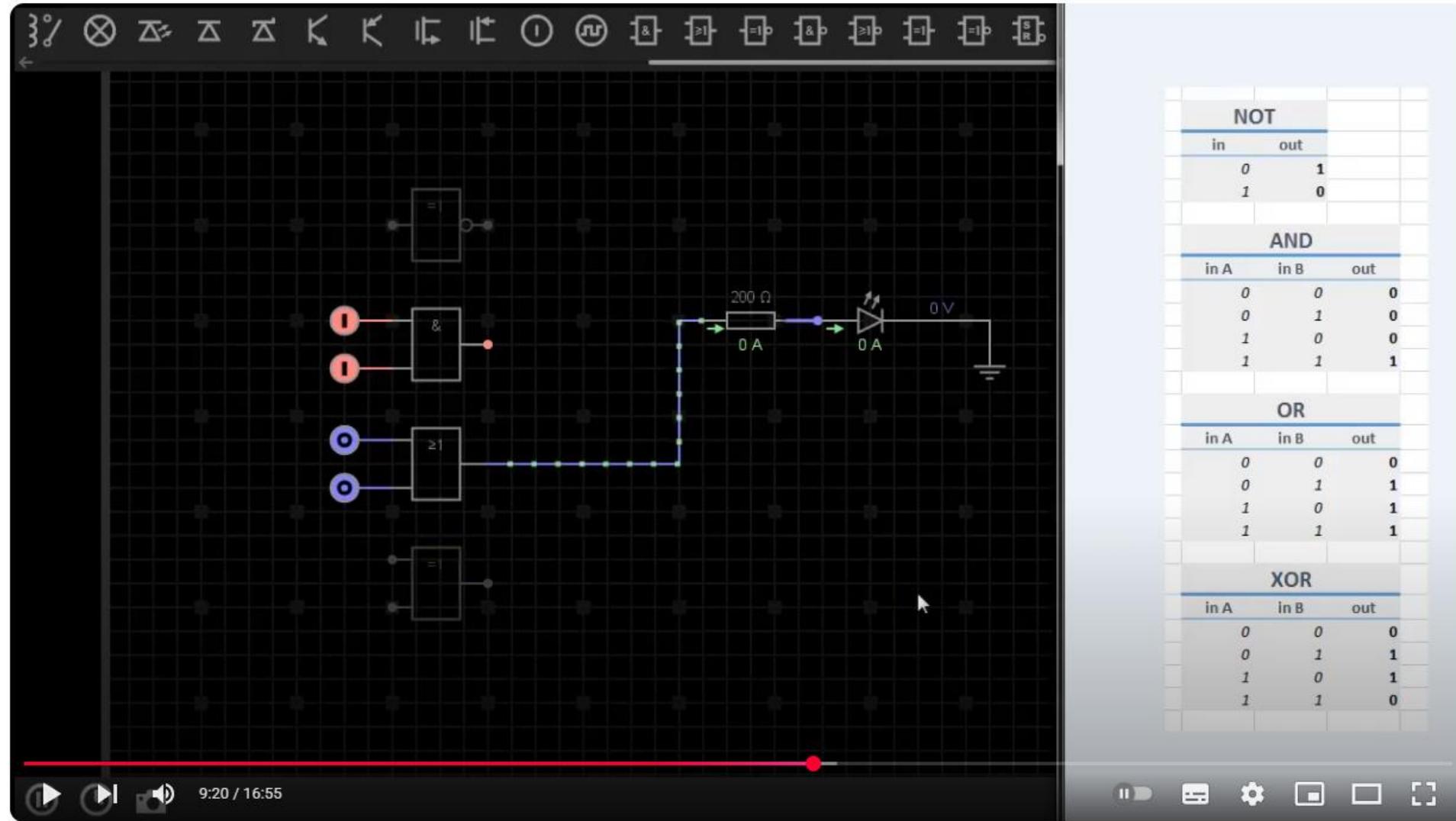
И (конъюнктор)



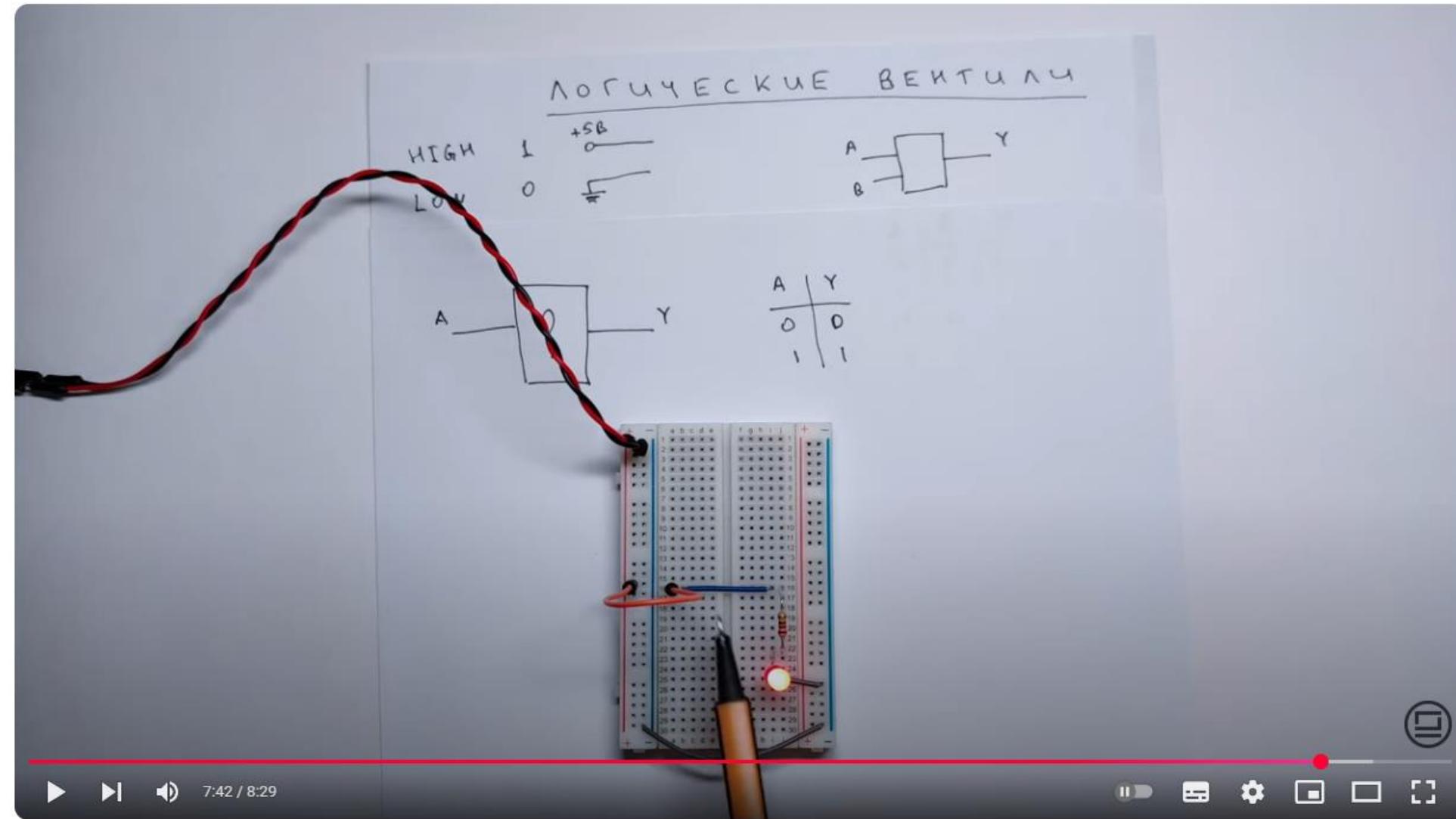
ИЛИ (дизъюнктор)



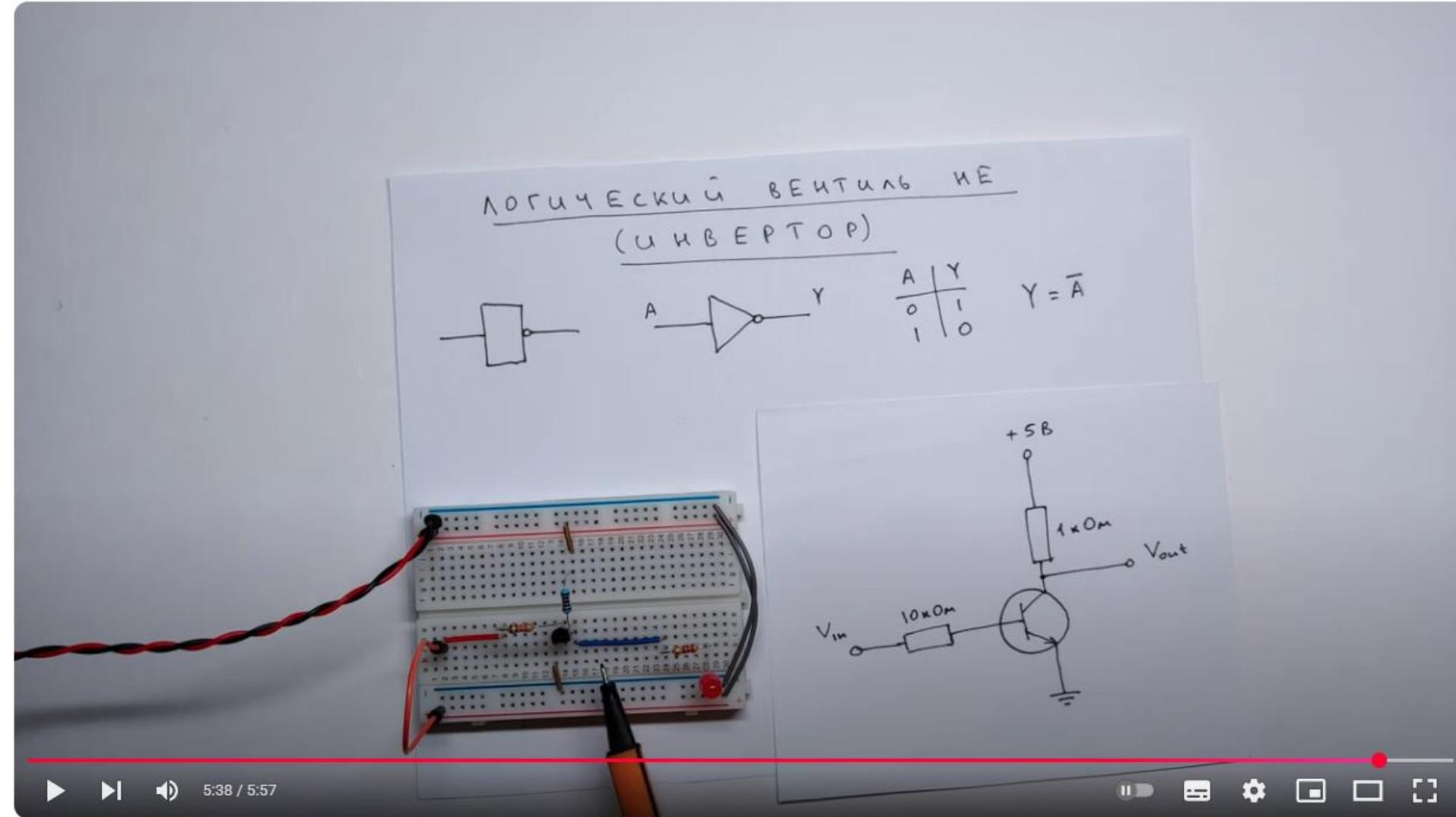
НЕ (инвертор)



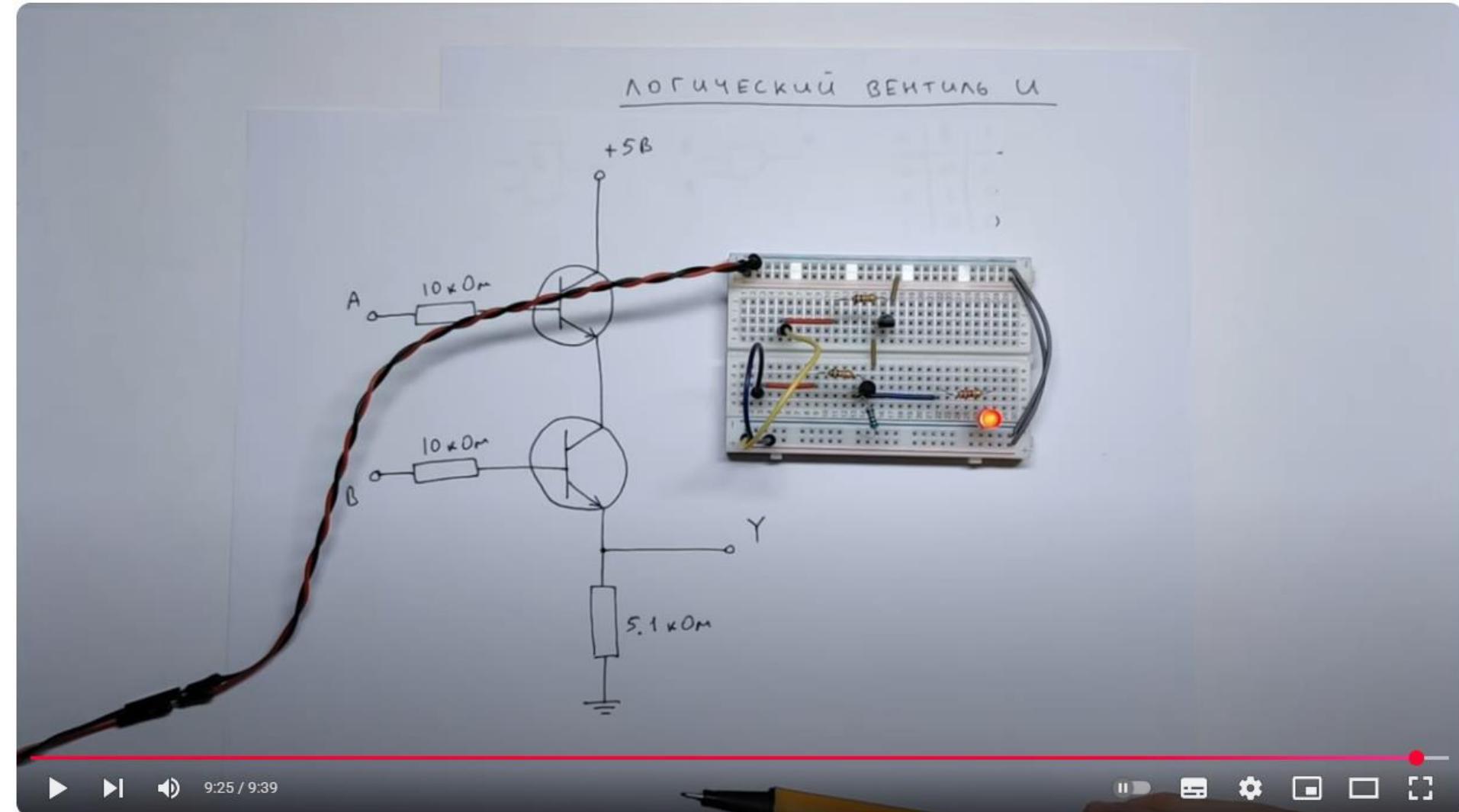
Урок №27. Базовые логические элементы (2016)  
<https://www.youtube.com/watch?v=7TEGOGejvI>



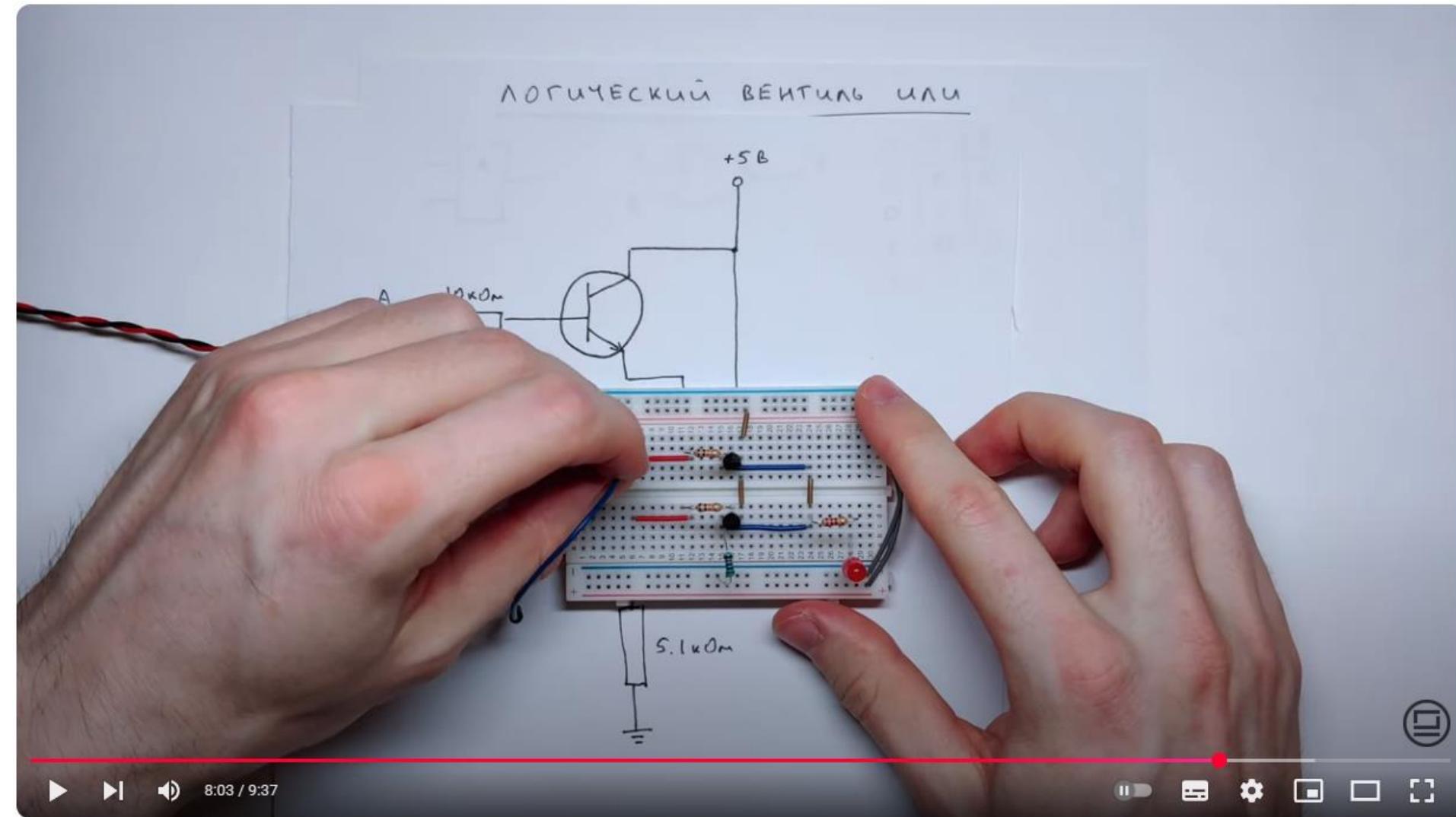
Введение в логические вентили (2020)  
<https://www.youtube.com/watch?v=n5Zo2wJS5XI>



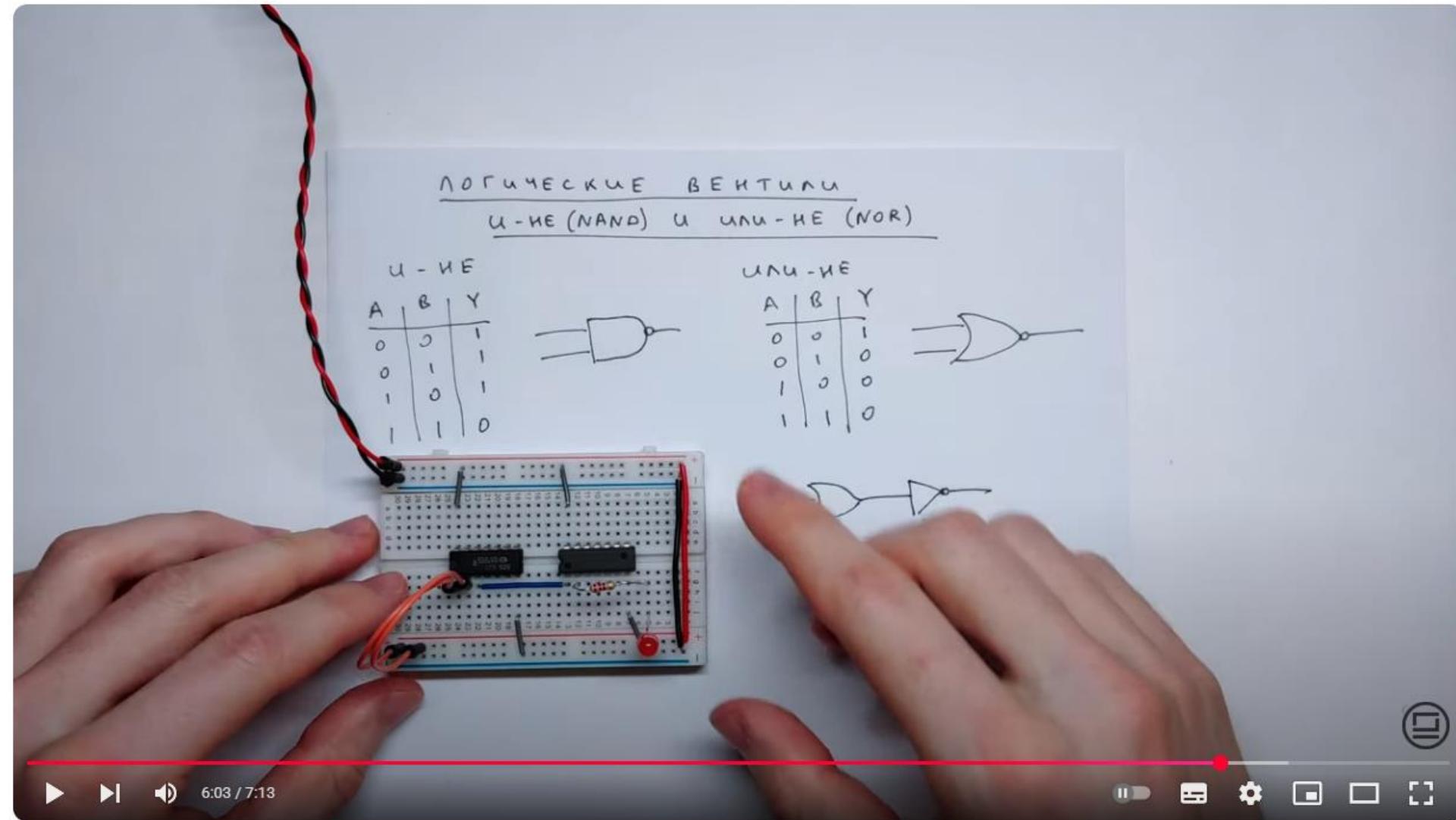
Логический вентиль НЕ (инвертор) - как устроен и как собрать самостоятельно (2020) <https://www.youtube.com/watch?v=vg67x1javjo>



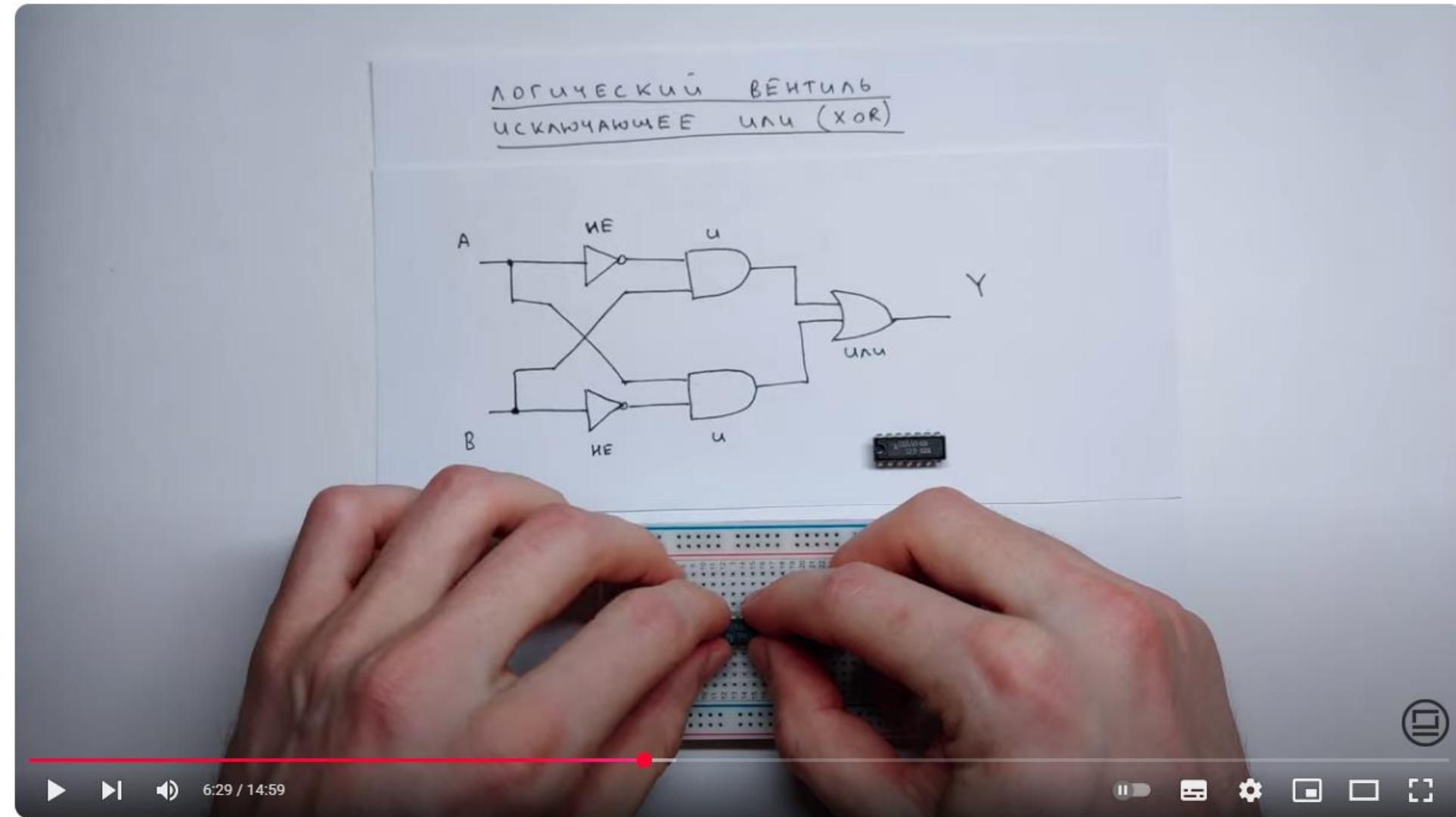
Логический вентиль И - как работает и как собрать на макетной плате на основе транзисторов (2020) <https://www.youtube.com/watch?v=o8kAhNwiPjw>



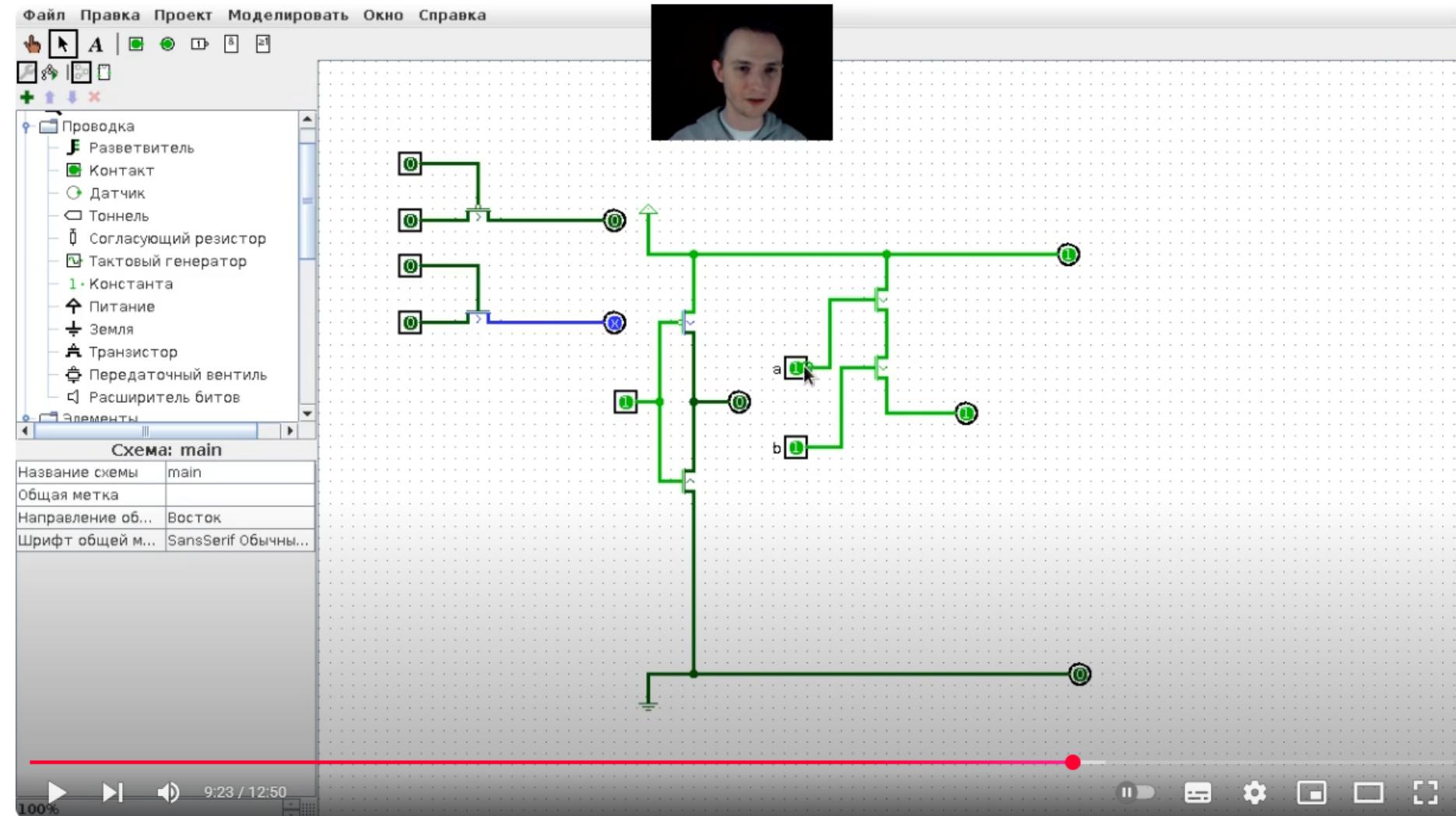
Логический вентиль ИЛИ - объяснение работы и пример сборки на макетной плате на основе транзисторов (2020) <https://www.youtube.com/watch?v=qaQ0cLC66GE>



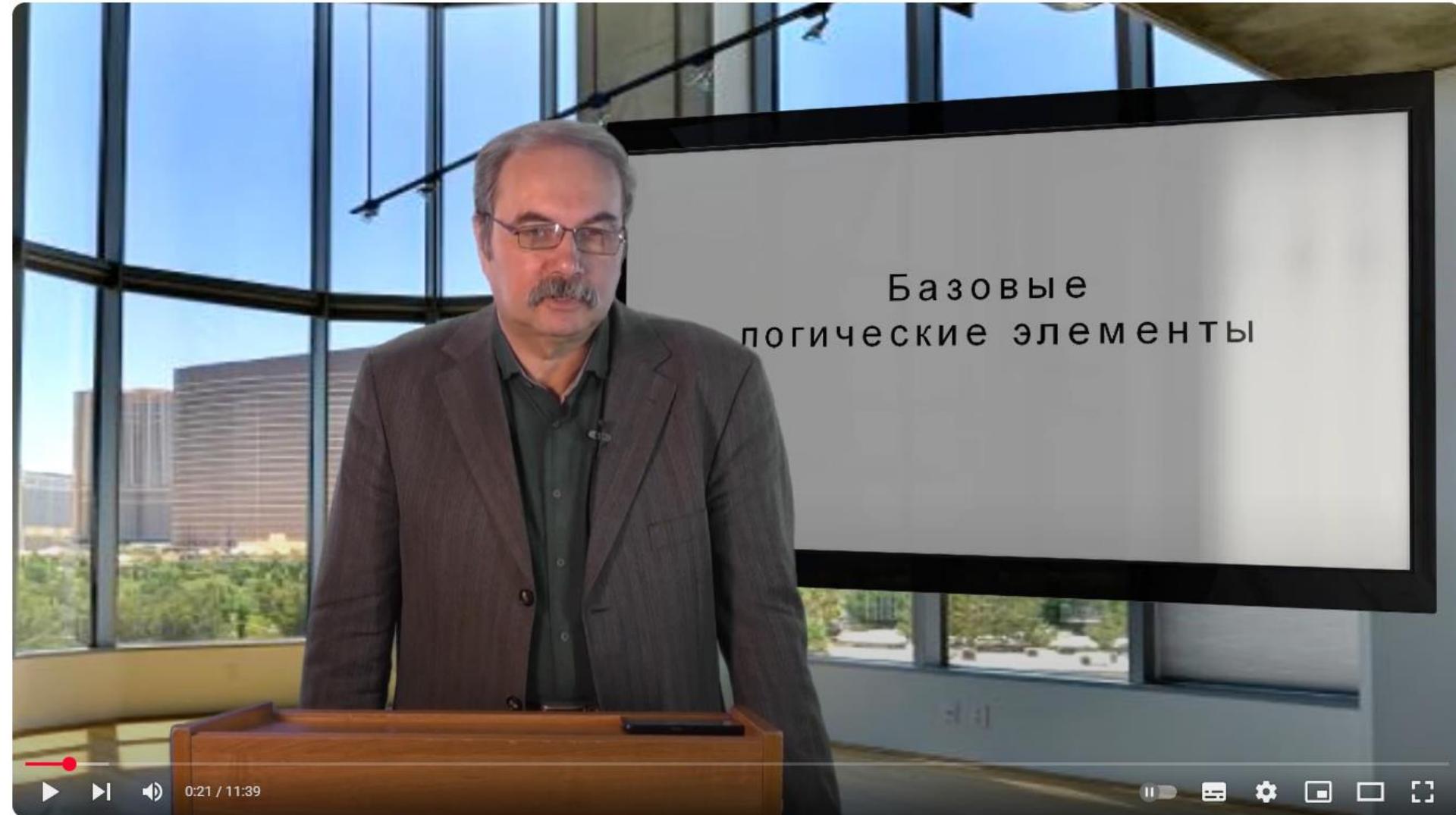
Логические вентили И-НЕ и ИЛИ-НЕ (2020)  
<https://www.youtube.com/watch?v=NKcGeBzOYbM>



Вентиль Исключающее ИЛИ (XOR) - объяснение принципа работы и пример на макетной плате (2020) <https://www.youtube.com/watch?v=Ik6be8jS9RA>



Цифровая техника - И, ИЛИ, НЕ на транзисторах (2015)  
<https://www.youtube.com/watch?v=CPp3Fp3Y2dw>



Зубаков А.П. Базовые логические элементы (2022)  
[https://www.youtube.com/watch?v=QcmDN\\_dFYXQ](https://www.youtube.com/watch?v=QcmDN_dFYXQ)

Белорусско-Российский университет, Кафедра «Программное обеспечение информационных технологий»

**Базовые элементы цифровых устройств**

U. Tietze, Ch. Schenk  
HALBLEITER-SCHALTUNGSTECHNIK  
Kollektivverantwortlicher Autoren  
SPRINGER-VERLAG BERLIN HEIDELBERG NEW YORK 1982

У.ТИЦЕ  
К.ШЕНК  
ПОЛУПРОВОДНИКОВАЯ  
СХЕМОТЕХНИКА

Перевод с немецкого  
под редакцией  
д-ра техн. наук А.Г. Алексенко  
МОСКВА «ИРН» 1982

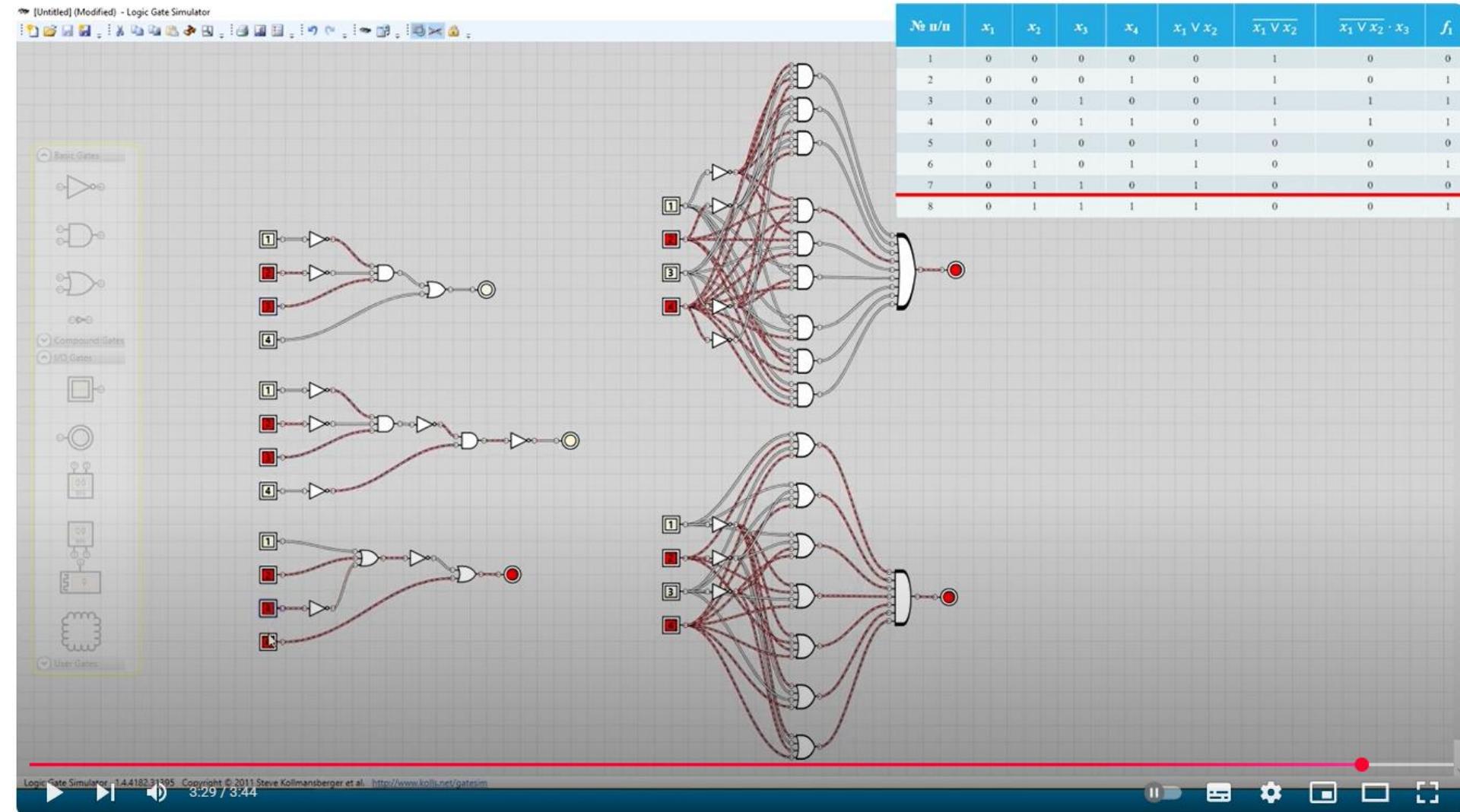
|       |  |     |
|-------|--|-----|
| 9.    | Базовые логические схемы . . . . .                                     | 100 |
| 9.1.  | Основные логические функции . . . . .                                  | 100 |
| 9.2.  | Составление логических функций . . . . .                               | 102 |
| 9.3.  | Применение основных логических функций . . . . .                       | 105 |
| 9.4.  | Схемотехническая реализация основных логических функций . . . . .      | 106 |
| 9.5.  | Интегральные триггеры . . . . .  | 117 |
| 9.6.  | Полупроводниковые запоминающие устройства . . . . .                    | 122 |
| 19.   | Комбинационные логические схемы . . . . .                              | 318 |
| 19.1. | Преобразователи кодов . . . . .  | 319 |
| 19.2. | Мультиплексор и демультиплексор . . . . .                              | 326 |
| 19.3. | Комбинационное устройство сдвига . . . . .                             | 328 |
| 19.4. | Компьютеры . . . . .   | 329 |
| 19.5. | Сумматоры . . . . .  | 331 |
| 19.6. | Умножители . . . . .   | 339 |
| 19.7. | Цифровые функциональные преобразователи . . . . .                      | 341 |
| 20.   | Интегральные схемы со структурами последовательностного типа . . . . . | 344 |
| 20.1. | Двоичные счетчики . . . . .  | 344 |
| 20.2. | Двоично-десятичный счетчик в коде 8421 . . . . .                       | 351 |
| 20.3. | Счетчик с предварительной установкой . . . . .                         | 353 |
| 20.4. | Регистры сдвига . . . . .  | 354 |
| 20.5. | Получение псевдослучайных последовательностей . . . . .                | 356 |
| 20.6. | Первичальная обработка асинхронного сигнала . . . . .                  | 359 |
| 20.7. | Систематический синтез последовательностных схем . . . . .             | 362 |

Юрий Н. Новиков, 2020

Заметки к слайду

Слайд 1 из 18    Тема Office    Русский (Россия)    0:01 / 1:00:05

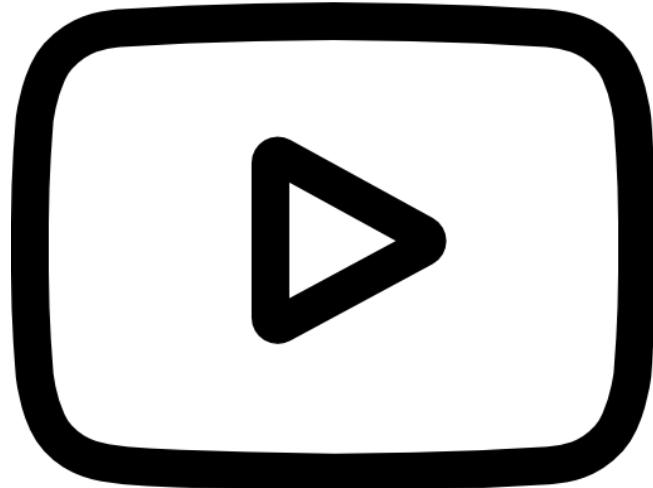
Базовые элементы цифровой электроники. Часть 1.(2020)  
<https://www.youtube.com/watch?v=Dc5XN3gu5Jo>



LOGIC GATE SIMULATOR: построение логических схем в основных базисах (2023) <https://www.youtube.com/watch?v=SbInkTLkmK4>

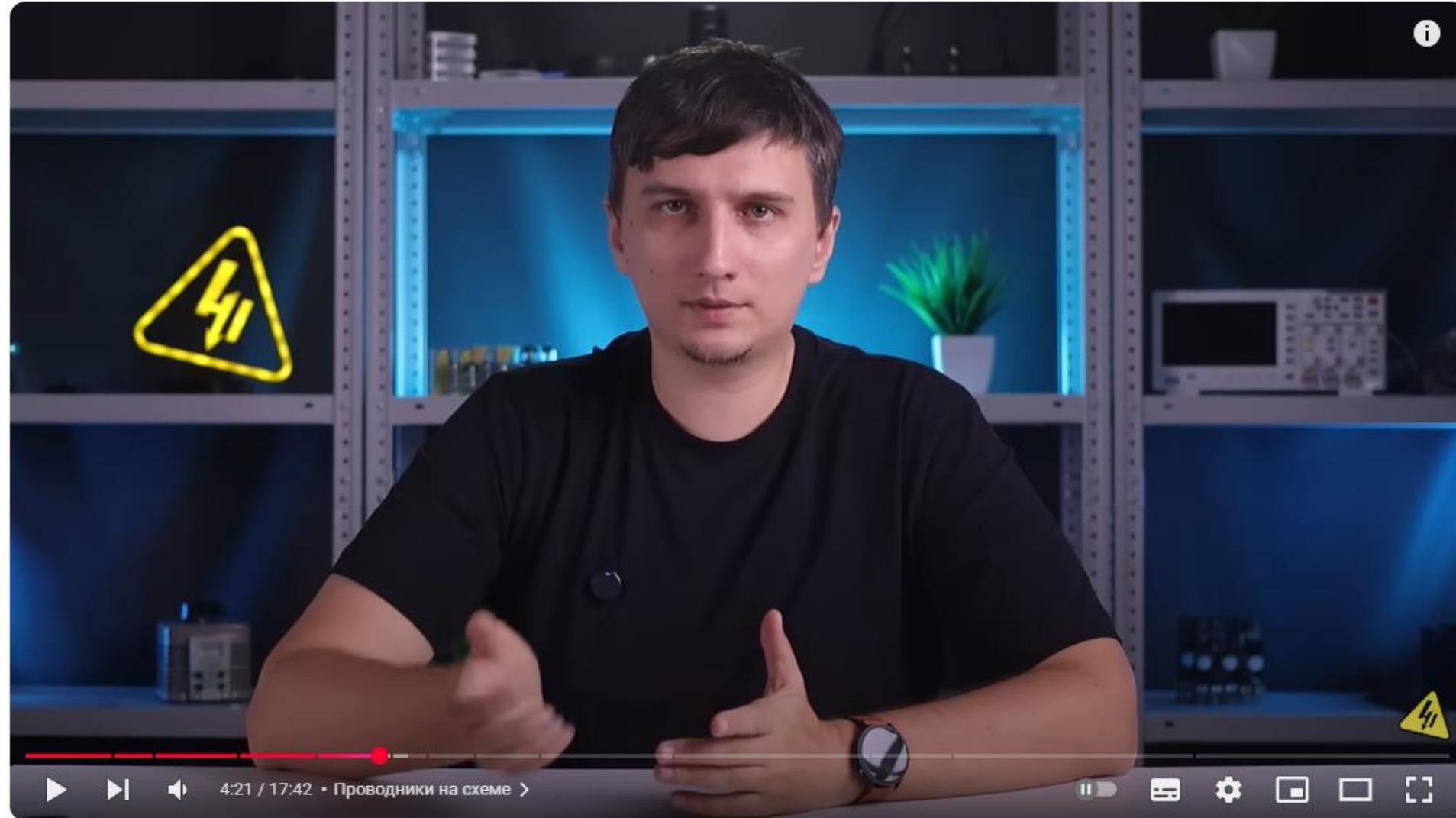


Digital Logic Gates from Transistors, AND, NAND, OR, NOR, XOR, XNOR, Buffer, and Inverter (2022)  
Цифровые логические элементы на основе транзисторов, AND, NAND, OR, NOR, XOR, XNOR, буфера и инвертора (2022)  
<https://www.youtube.com/watch?v=nB6724G3b3E>



**Физический уровень  
работы простейших  
элементов  
компьютерных  
комплектующих**

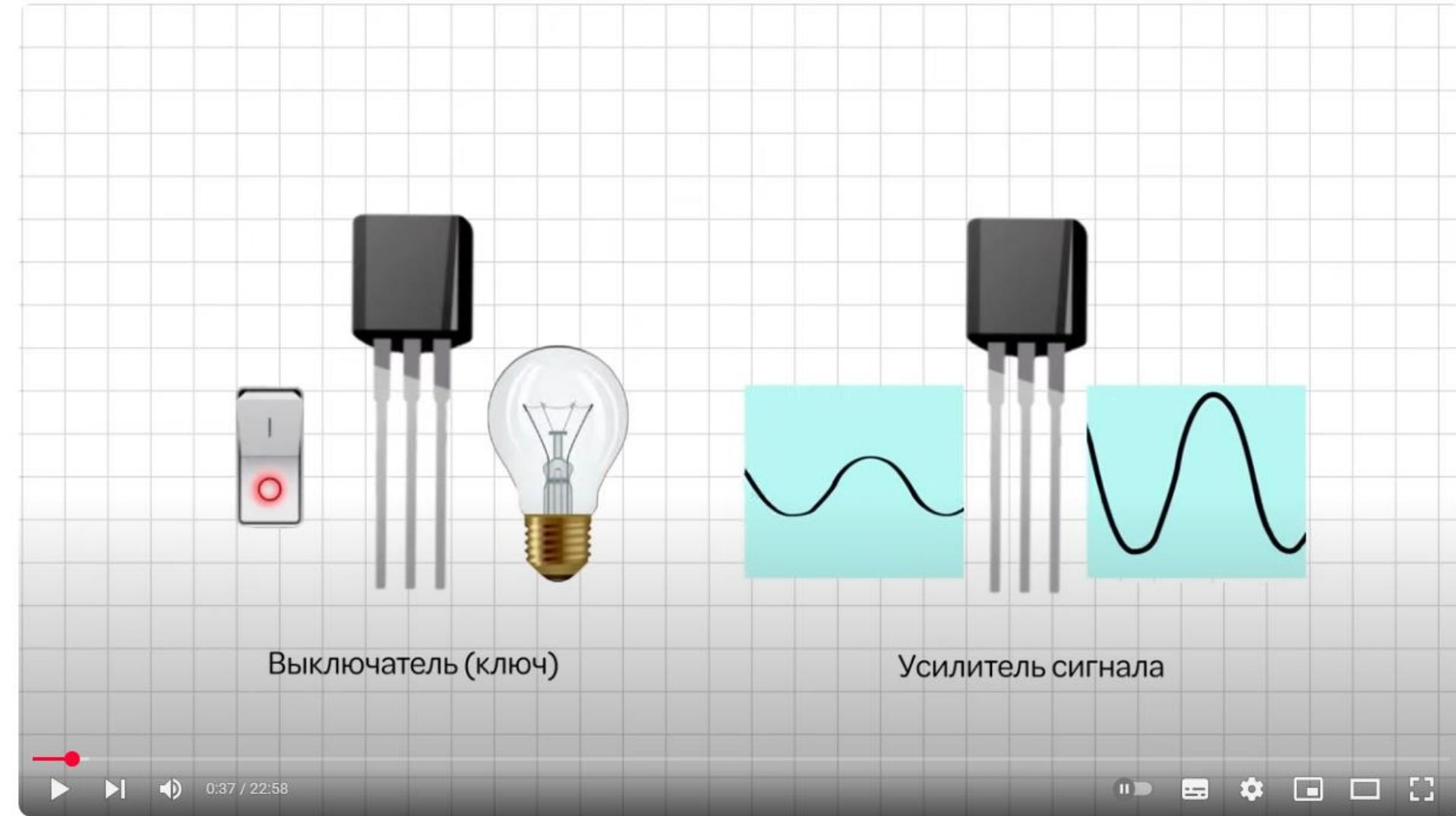




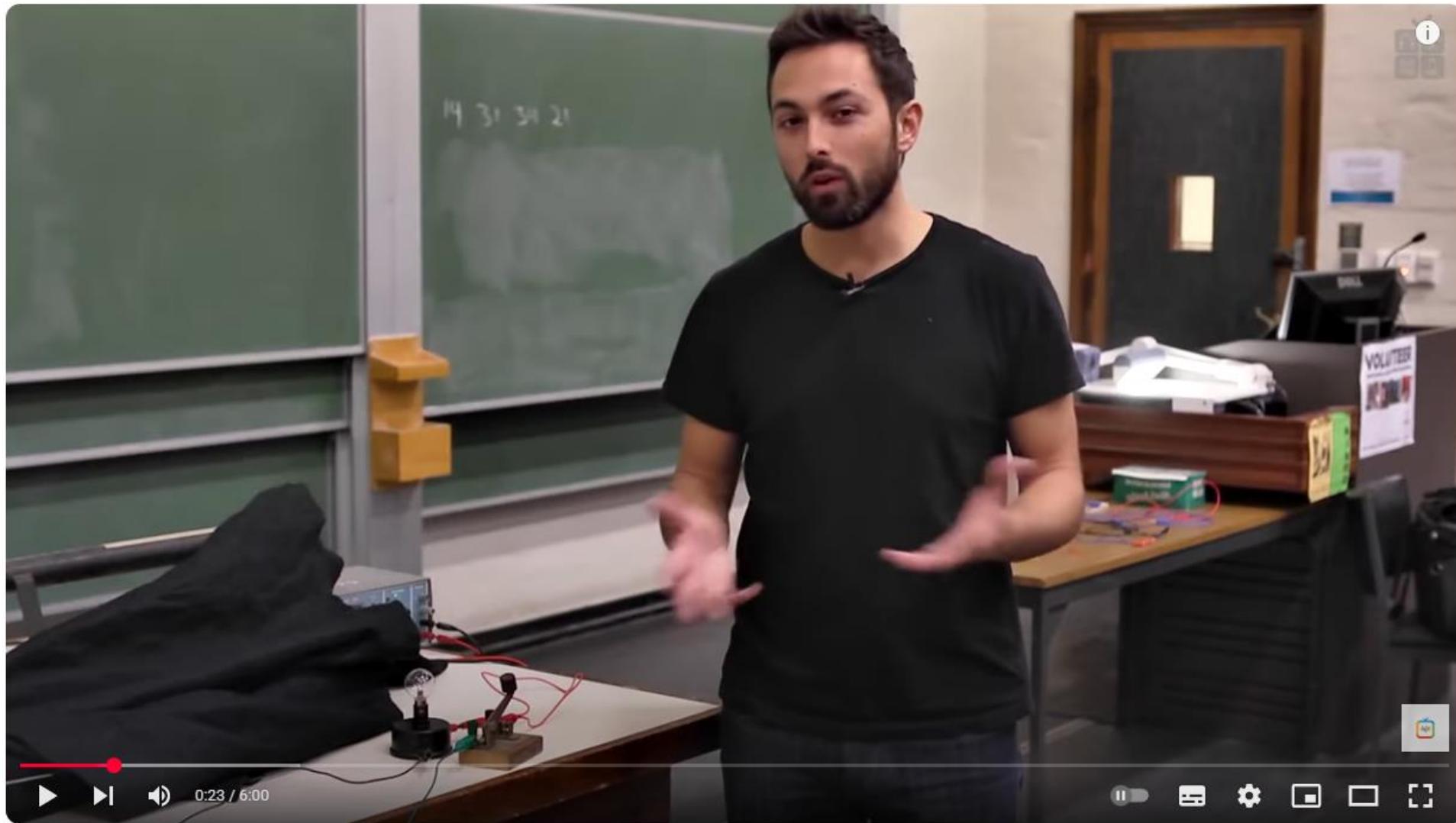
Где начало СХЕМЫ? Понимаем, читаем, изучаем схемы. Понятное объяснение! (2025)  
<https://www.youtube.com/watch?v=u6f6v4YTjN4>



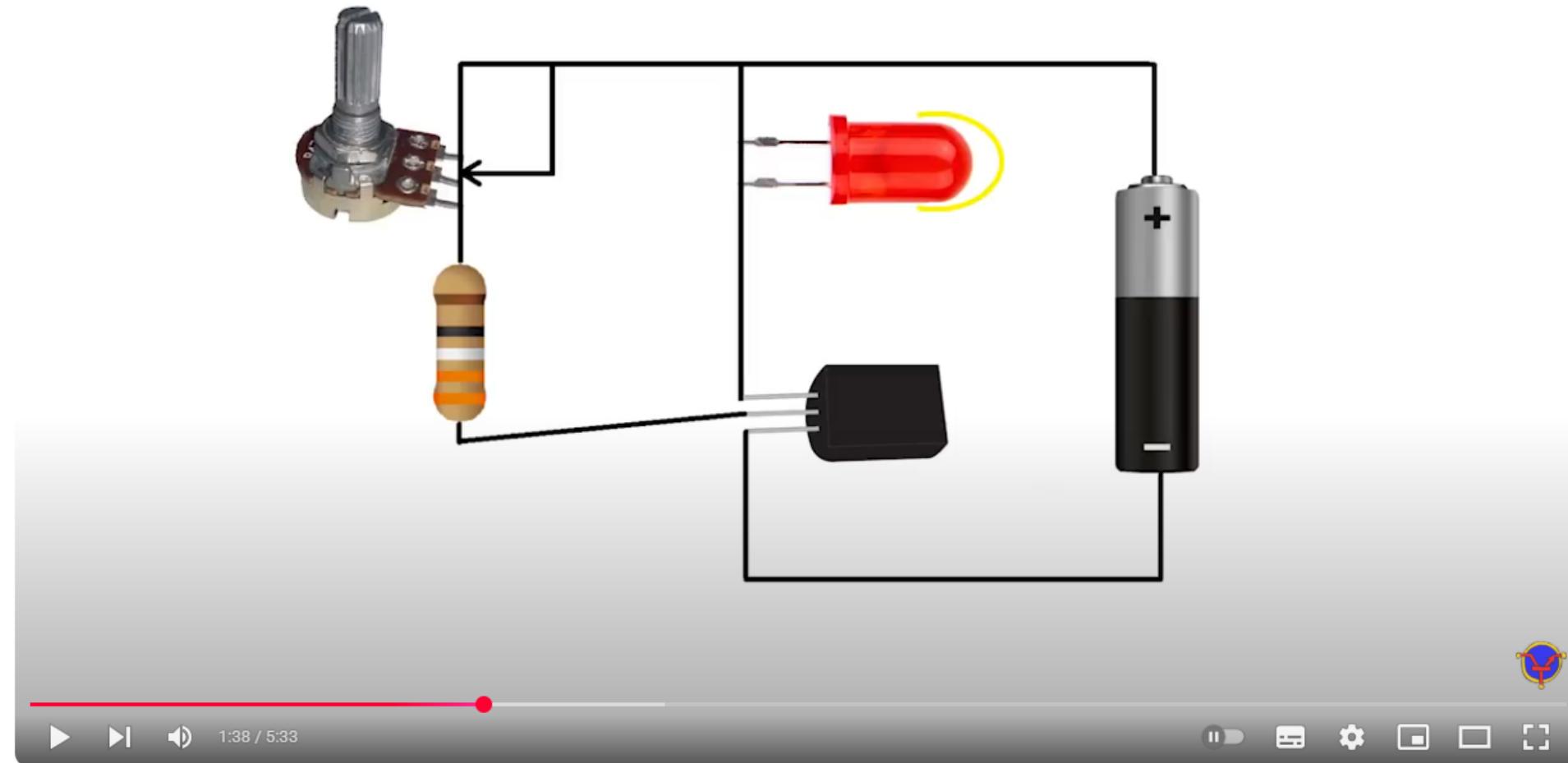
Транзисторы и их применение (1987)  
[https://www.youtube.com/watch?v=y\\_vbpE-R-z8](https://www.youtube.com/watch?v=y_vbpE-R-z8)



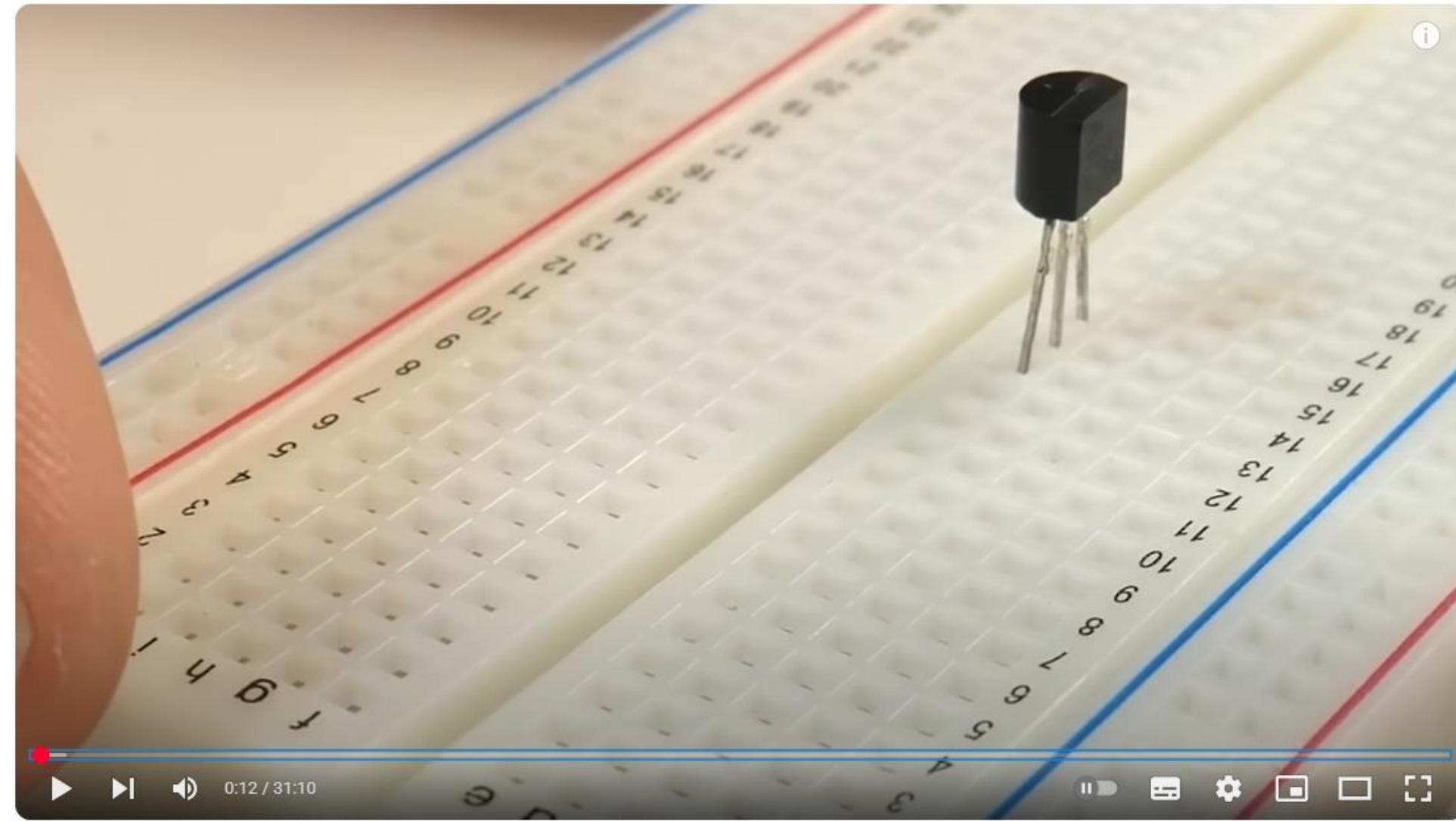
ТАКОЕ НЕ ПОКАЖУТ В ВУЗах - Как работают и для чего нужны транзисторы ?  
Что такое PN переход? (2024) <https://www.youtube.com/watch?v=fHsqtFTnOzY>



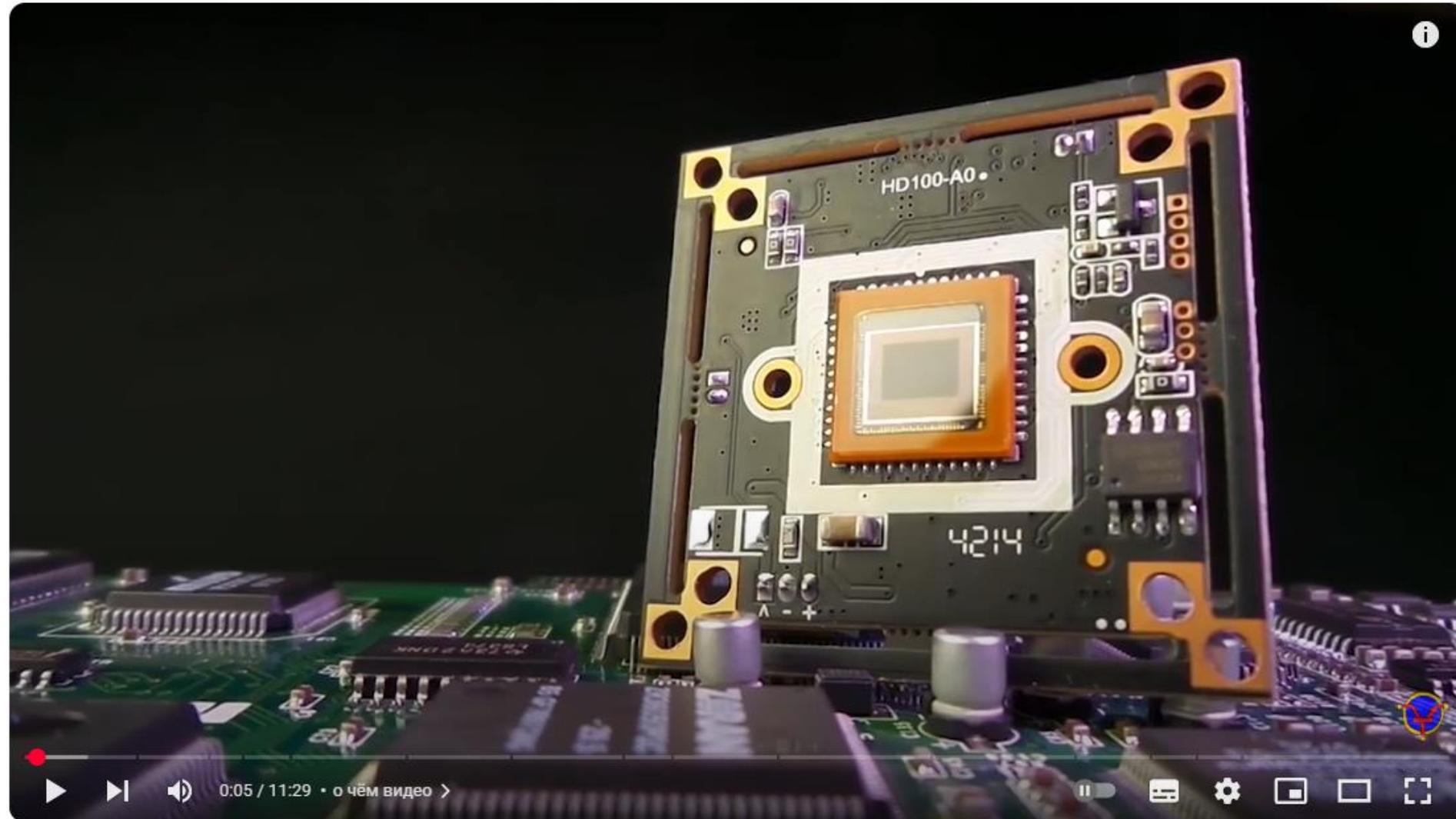
Как работает транзистор [Veritasium] (2016)  
<https://www.youtube.com/watch?v=-z65gKETbf8>



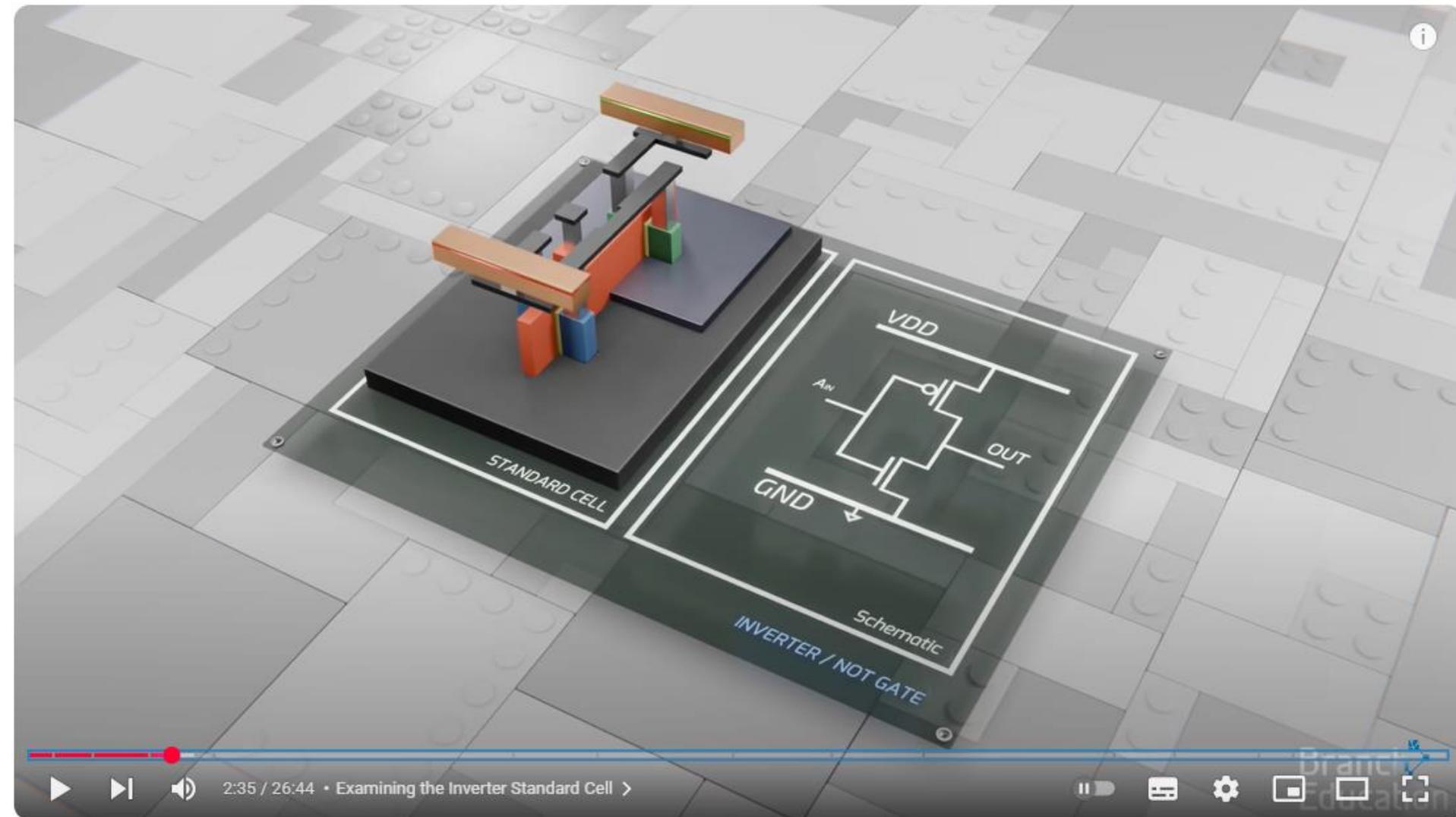
Как работают схемы с транзисторами (2019)  
[https://www.youtube.com/watch?v=whn\\_fU95QKI](https://www.youtube.com/watch?v=whn_fU95QKI)



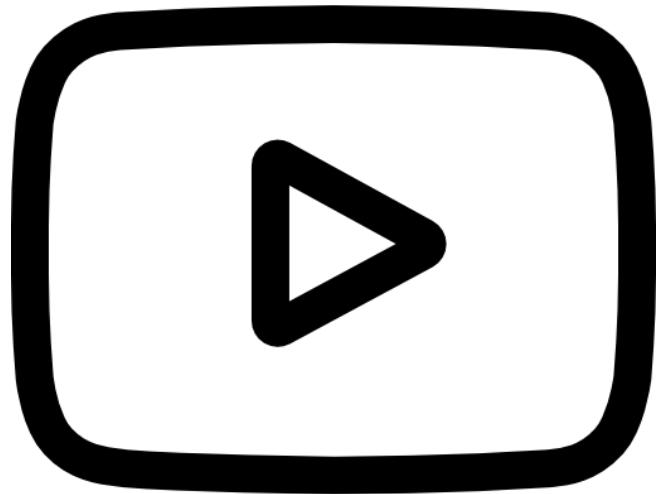
Магия транзисторов: как мы научили компьютеры думать с помощью кусочков кремния?(2023) [https://www.youtube.com/watch?v=\\_5W\\_GZOPa8E](https://www.youtube.com/watch?v=_5W_GZOPa8E)



КАК РАБОТАЕТ ПРОЦЕССОР | КАК ТРАНЗИСТОРЫ НАУЧИЛИСЬ СЧИТАТЬ? (2020)  
[https://www.youtube.com/watch?v=W6thMGQOX\\_k](https://www.youtube.com/watch?v=W6thMGQOX_k)

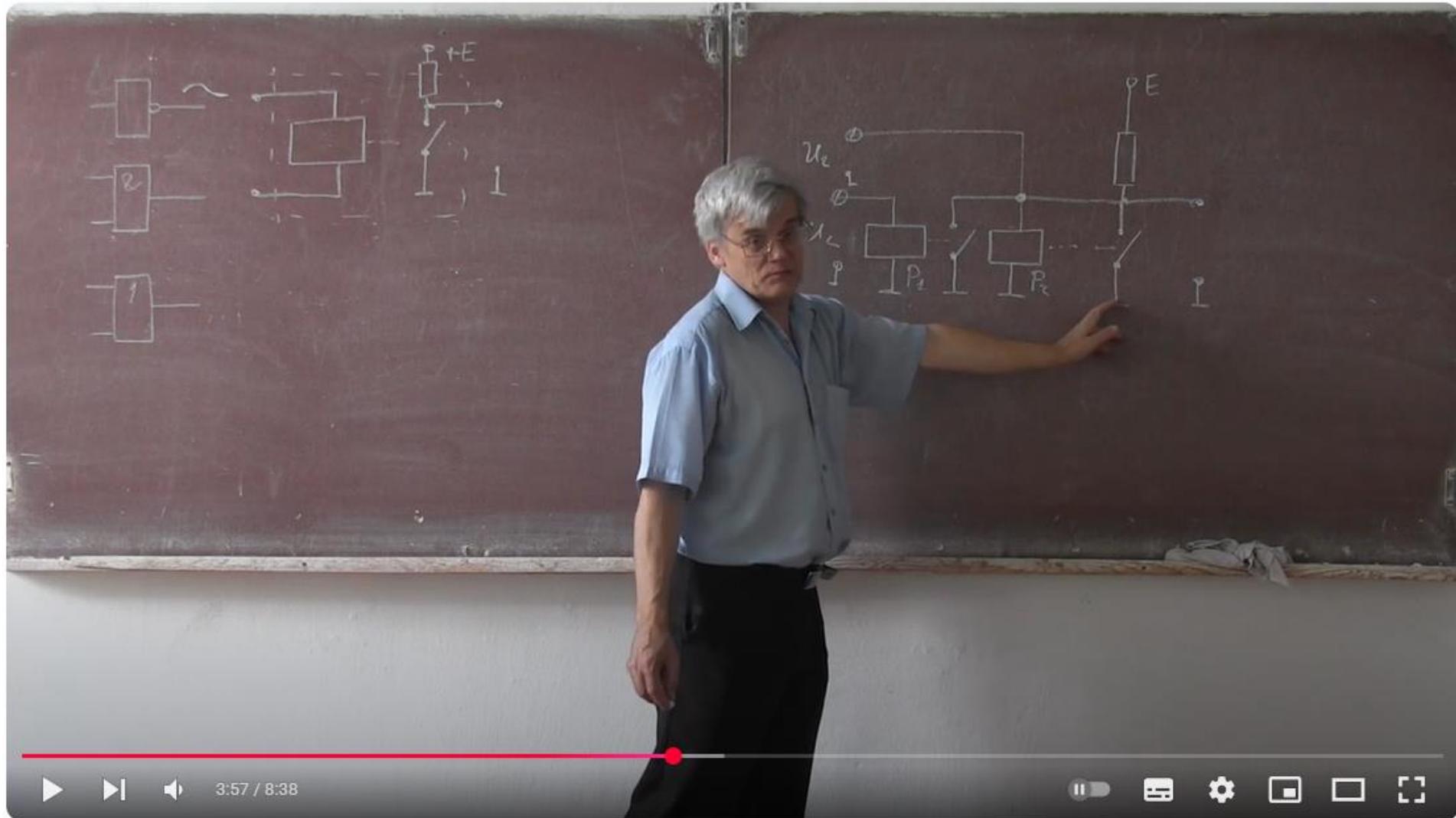


How do Transistors Build into a CPU? How do Transistors Work?(2025)  
Как транзисторы встраиваются в центральный процессор? Как работают транзисторы? (2025)  
[https://www.youtube.com/watch?v=\\_Pqfjer8-O4](https://www.youtube.com/watch?v=_Pqfjer8-O4)



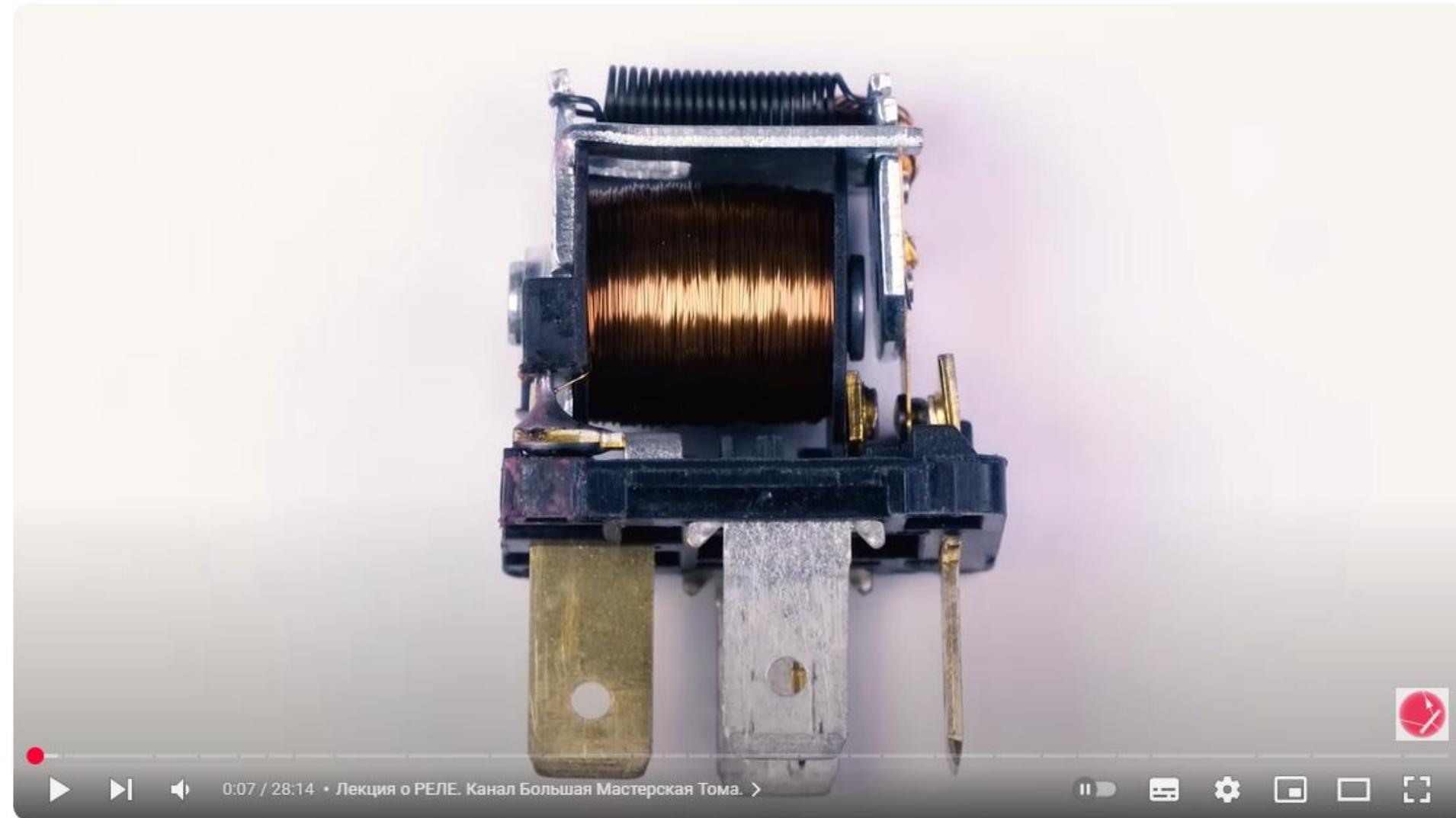
# Релейная логика



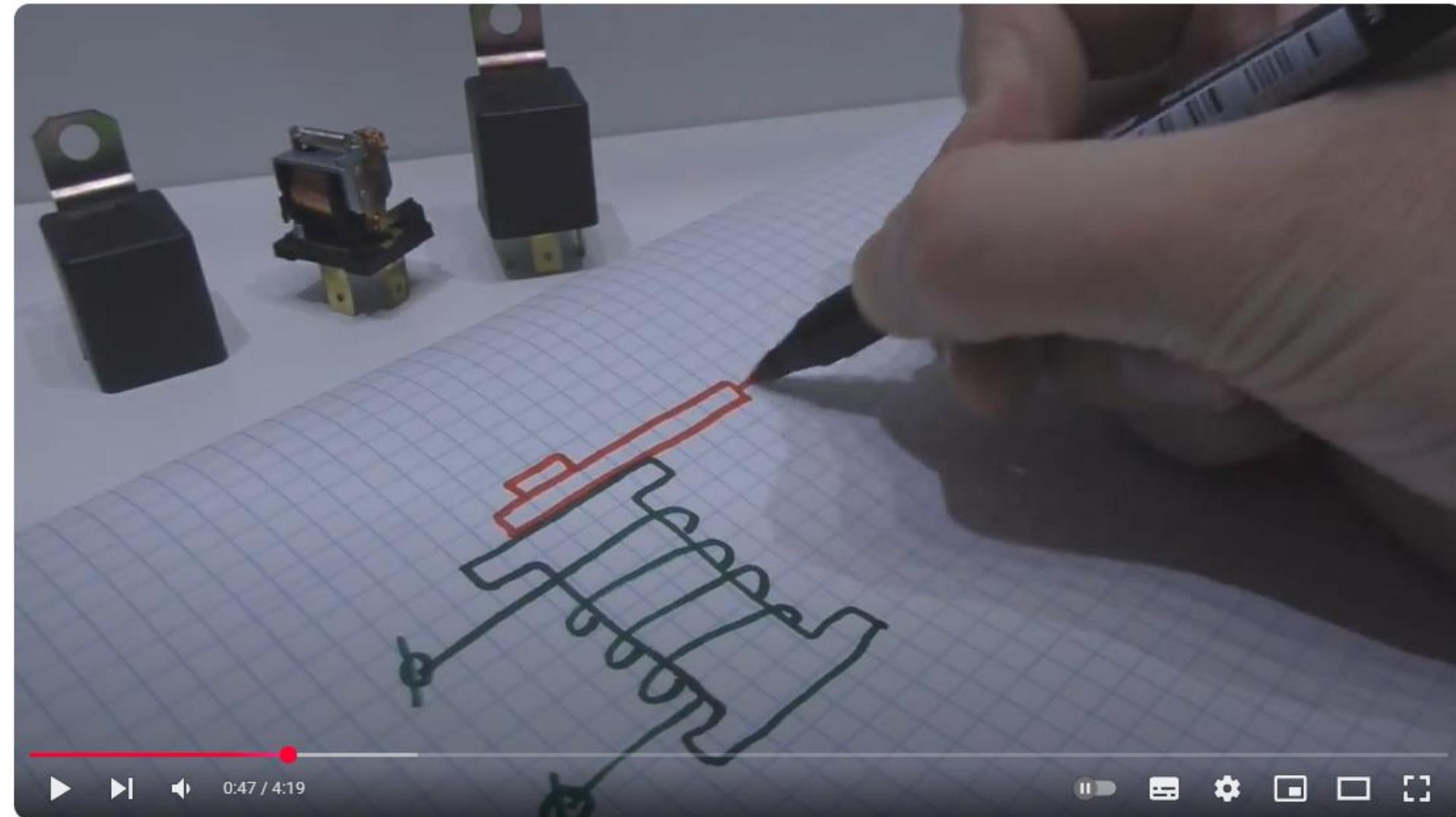


▶ ▶ 🔍 3:57 / 8:38 ⏴ ⏵ ⏷ ⏸ ⏹ ⏺

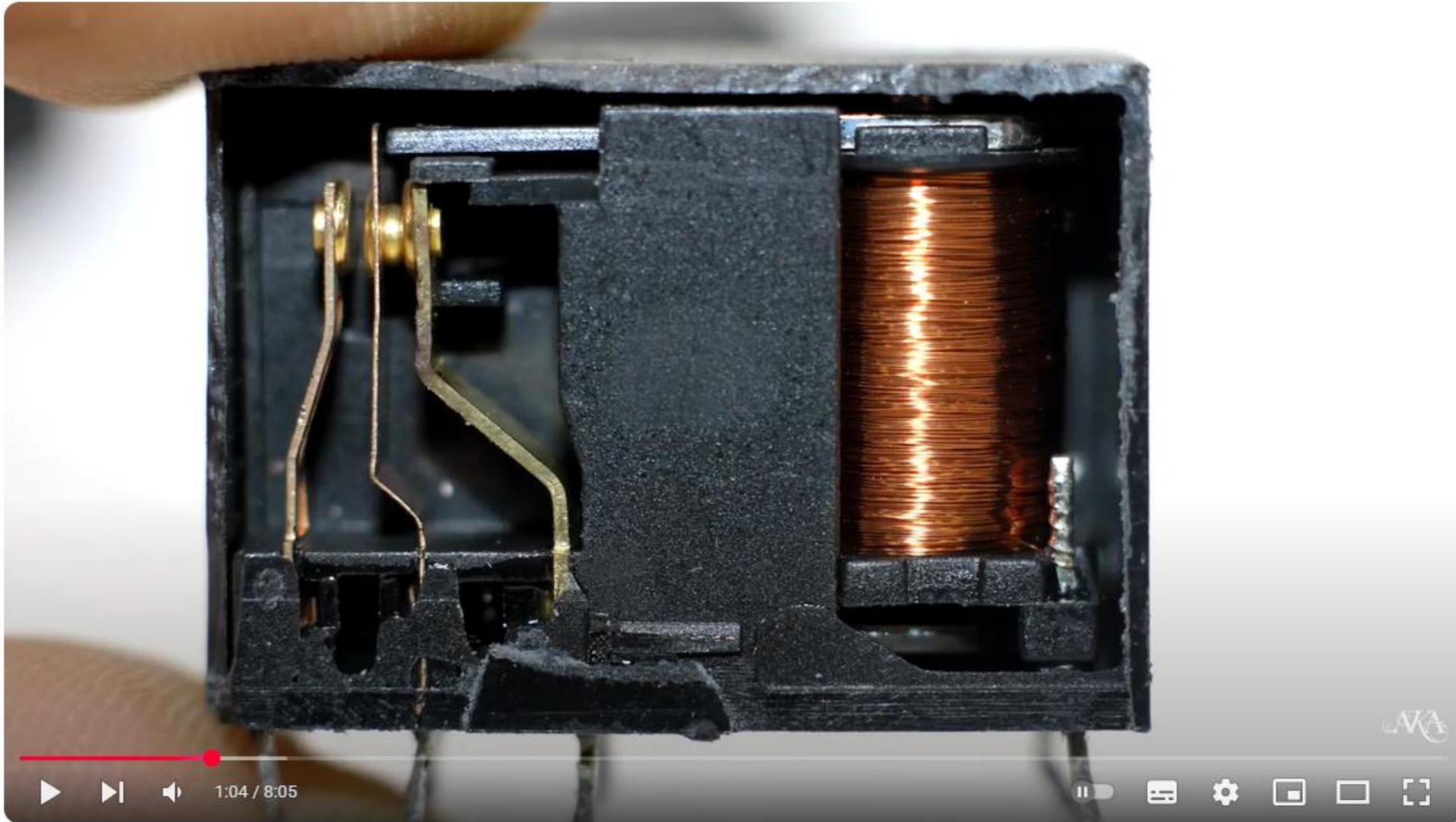
Лекция 261. Простейшая логика на реле (2015)  
<https://www.youtube.com/watch?v=gz7luCwHVNM>



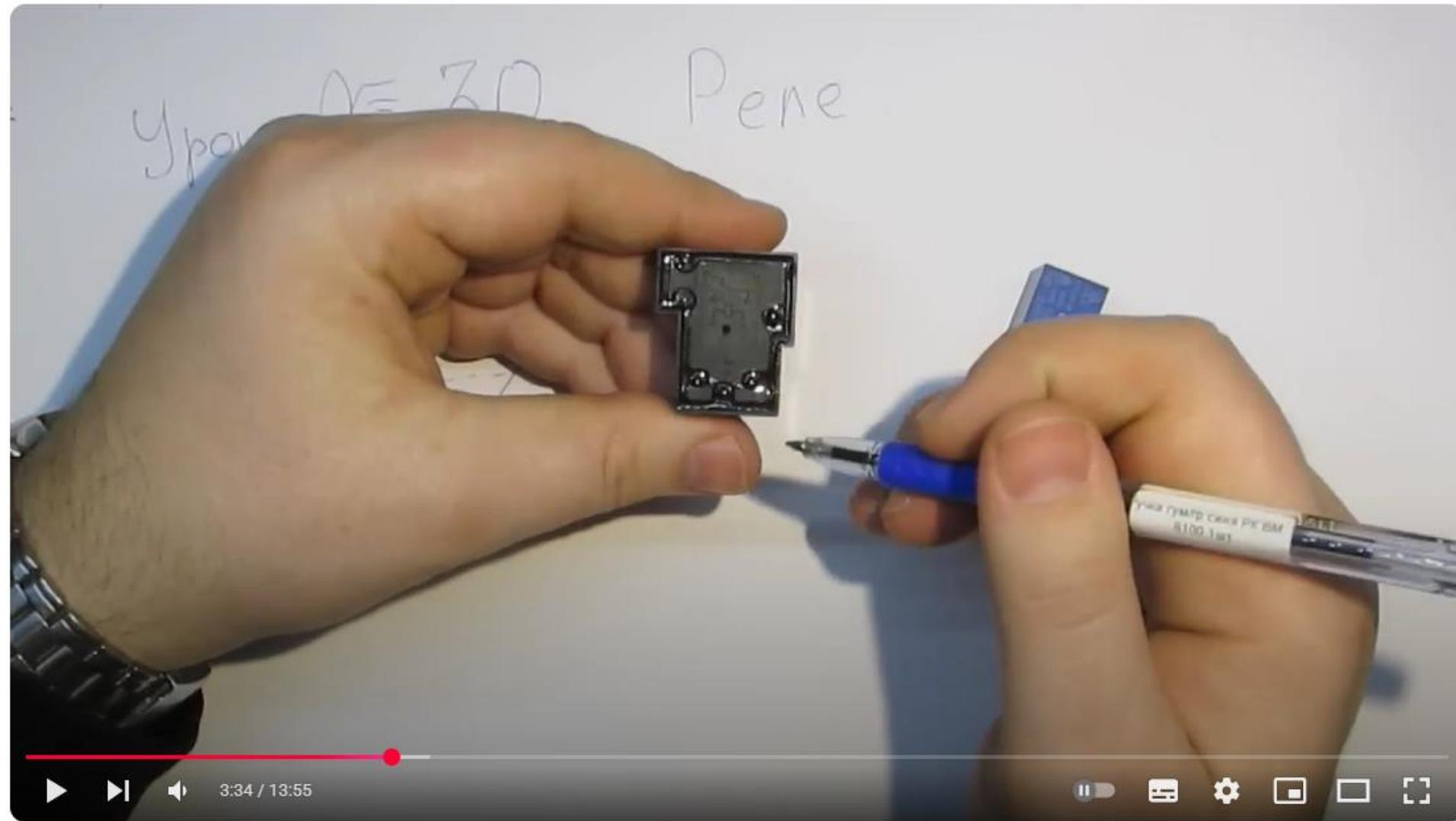
ВСЁ что Вы хотели знать о РЕЛЕ. Виды и способы подключения -- в Теории и на Практике! (2021)  
<https://www.youtube.com/watch?v=SagbSOhlFlc>



Как работает реле? Что такое реле? (2019)  
<https://www.youtube.com/watch?v=uIVlout6NCo>



Электромагнитное реле - как это работает ? (2017)  
<https://www.youtube.com/watch?v=I8KFFvOA3mg>



Урок №30. Реле. (2017)  
<https://www.youtube.com/watch?v=c4wDE9PstGw>

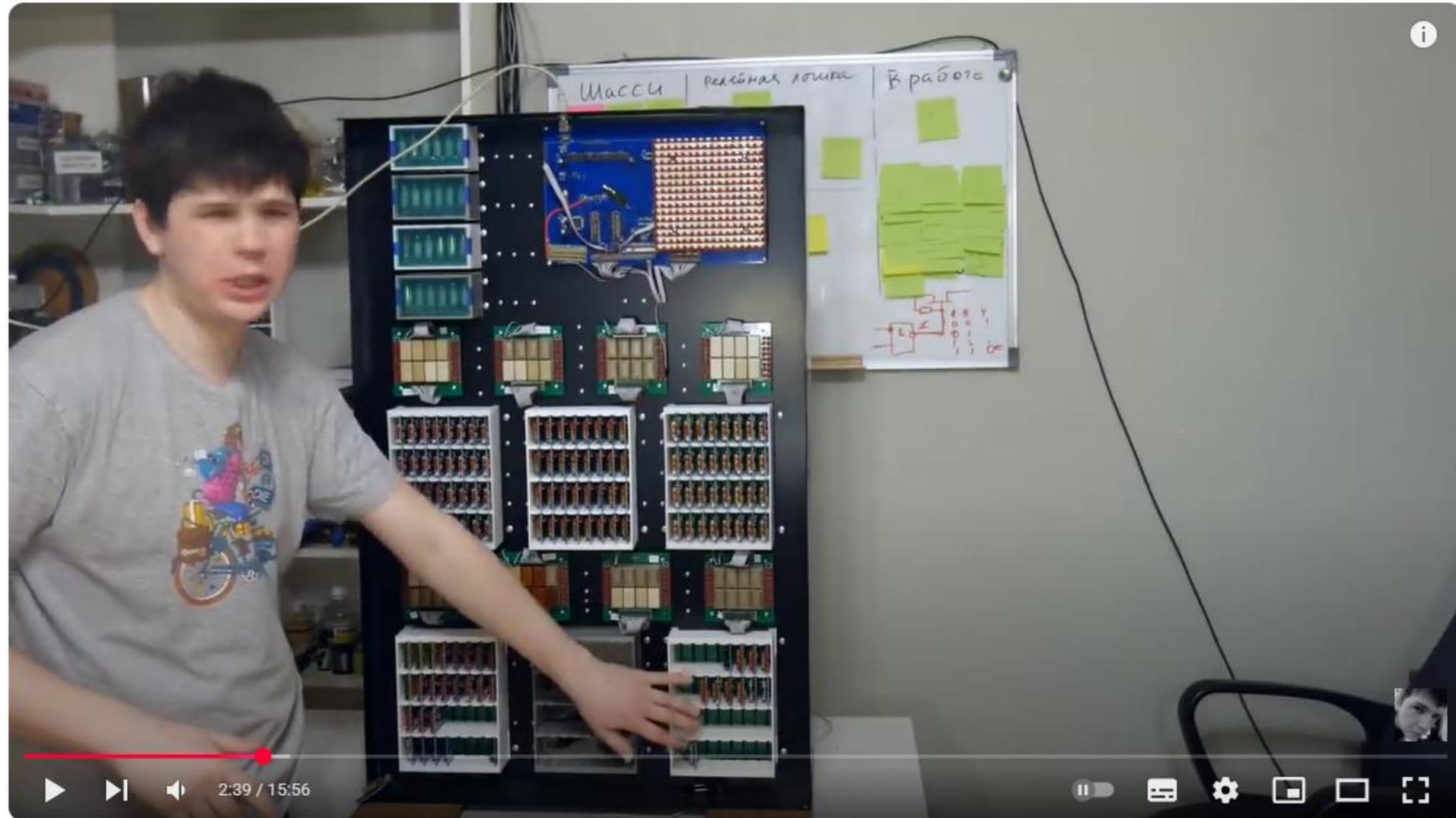
# КАК РАБОТАЮТ РЕЛЕ

## Рассмотрим:

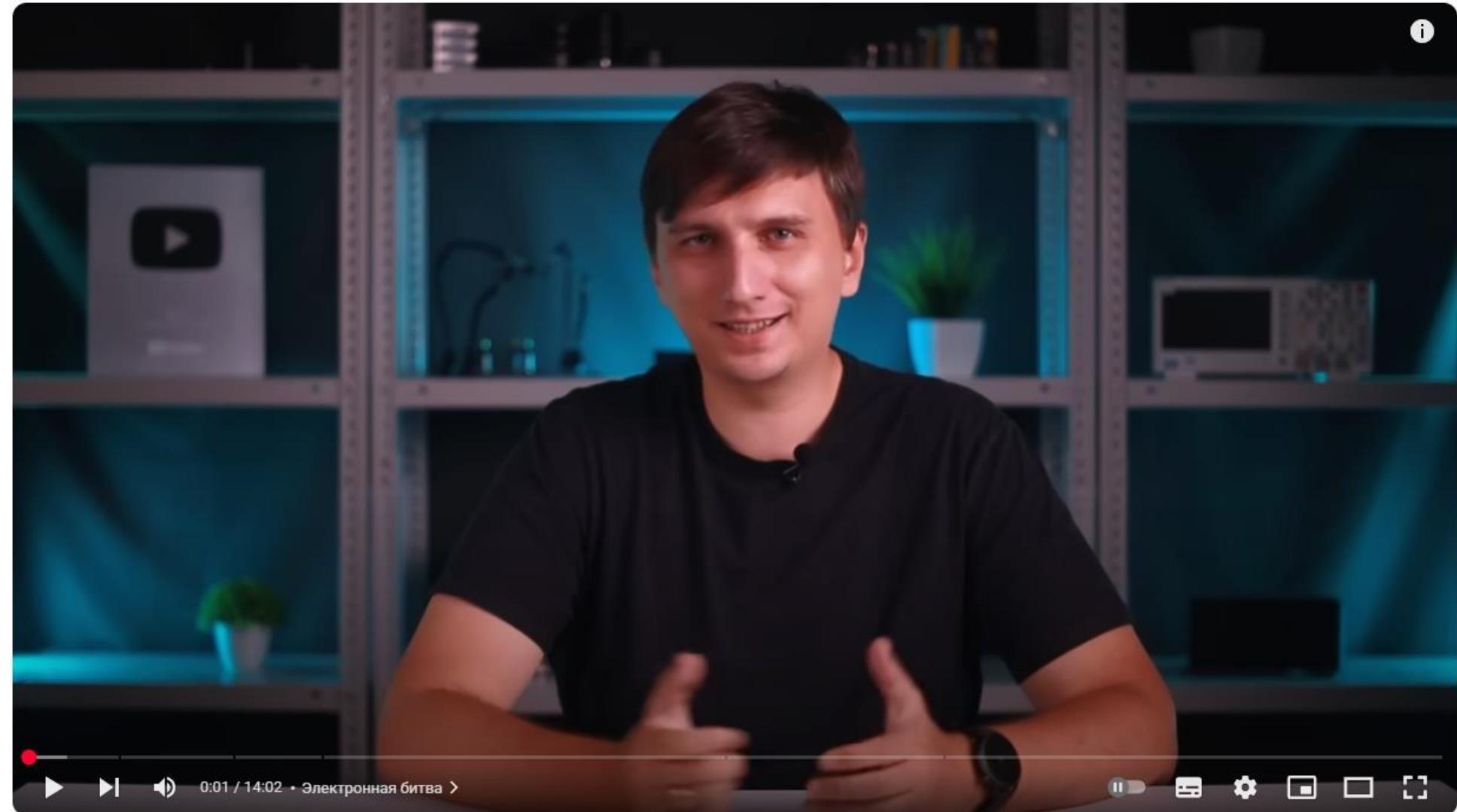
- Что такое реле, зачем они нужны
- Основные части
- Как работают электромагнитные реле
- Типы реле
- Демпферные диоды



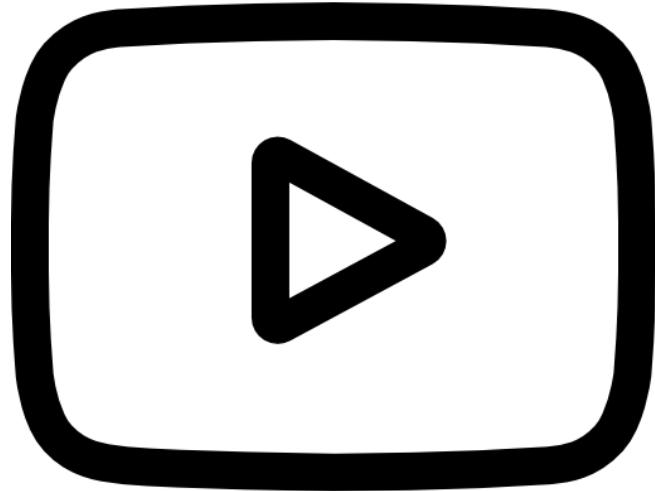
Как работает электромагнитное реле? Принцип работы (2023)  
<https://www.youtube.com/watch?v=bVmAzMQpj7Q>



Пожалуй, самый быстрый релейный компьютер в мире (2019)  
[https://www.youtube.com/watch?v=5hhbGBIP3\\_4](https://www.youtube.com/watch?v=5hhbGBIP3_4)

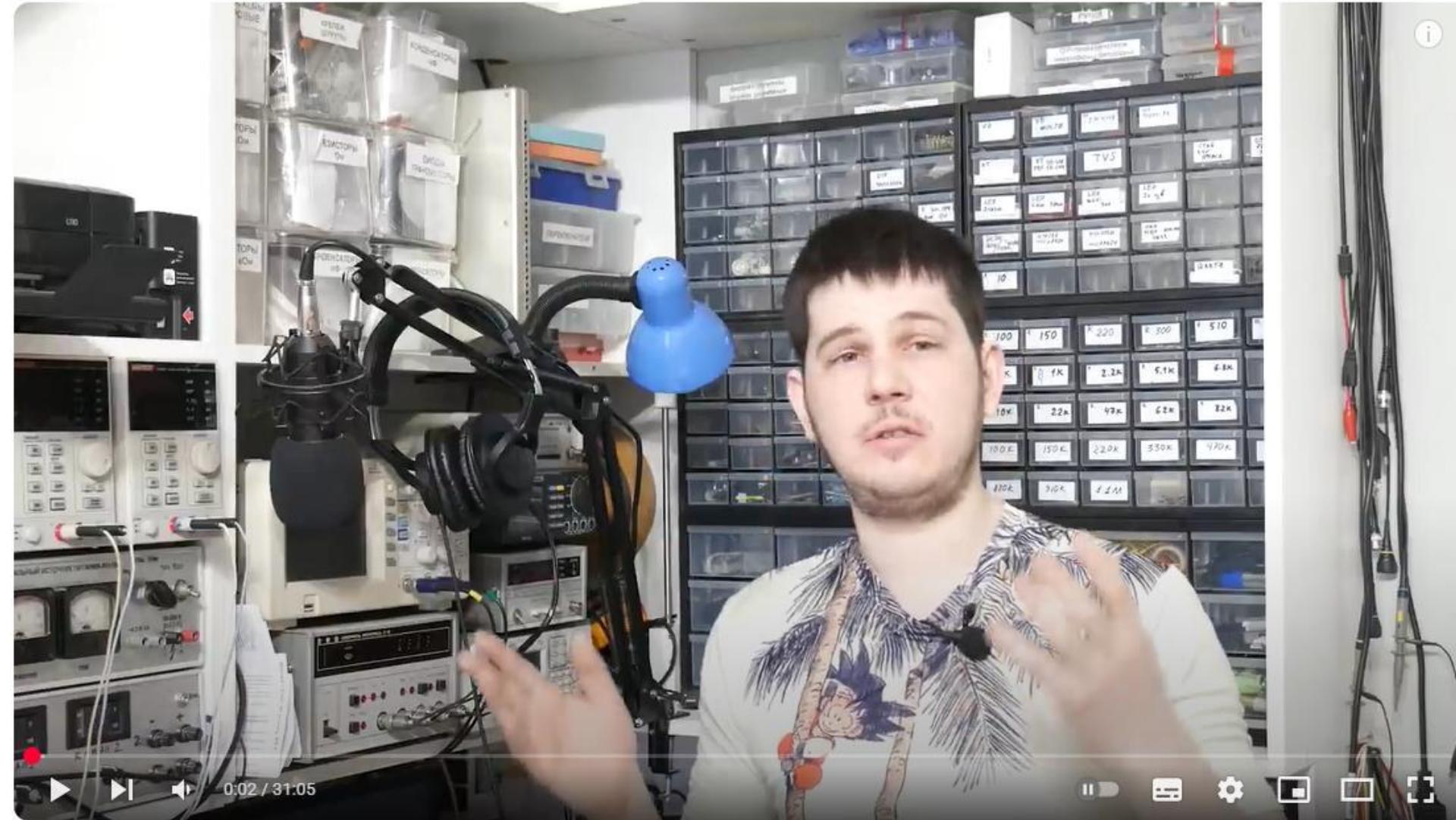


Реле против Транзисторов: Что лучше и в каких ситуациях? (2024)  
<https://www.youtube.com/watch?v=xXiZuy9hTO4>

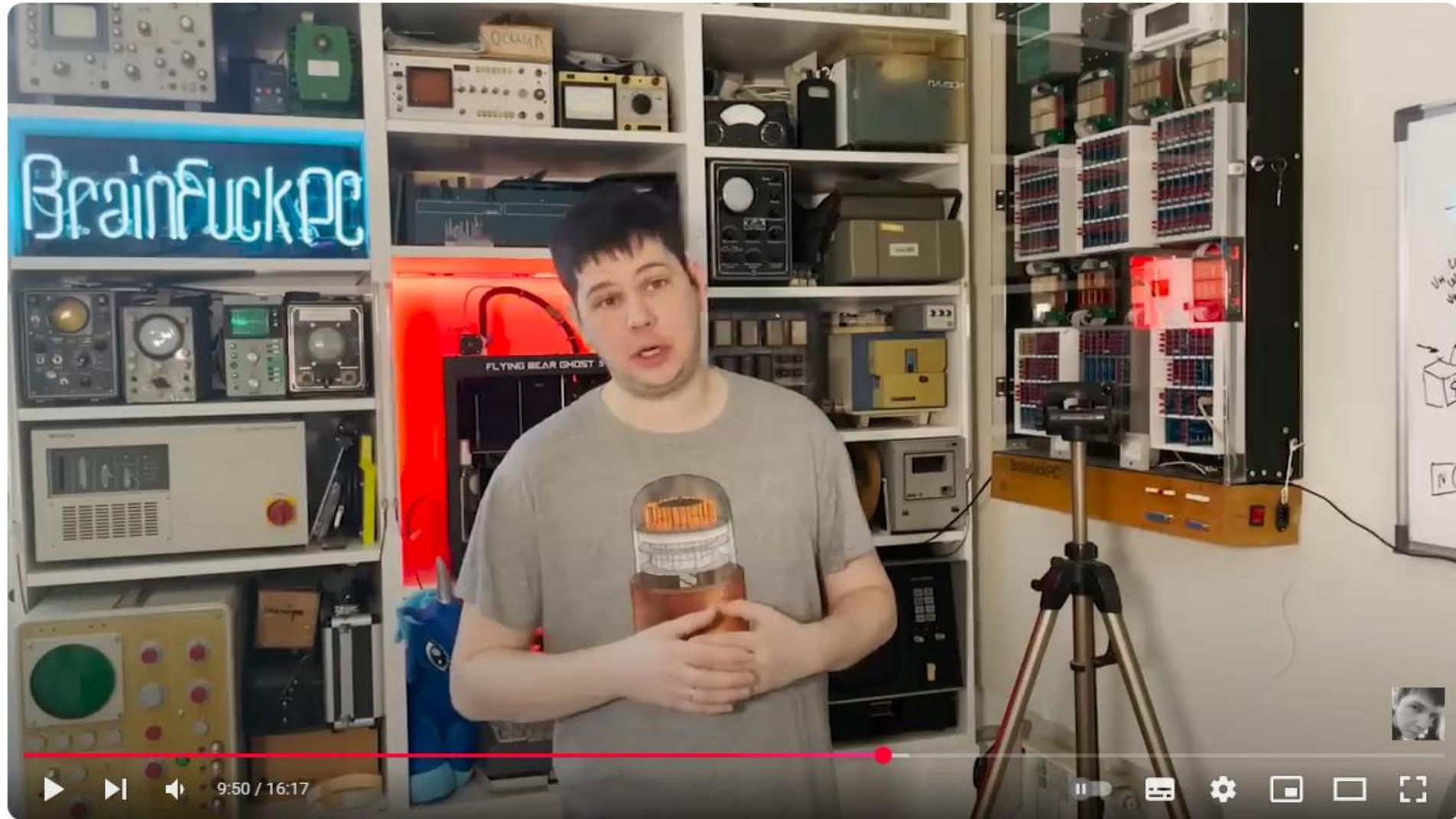


**Ламповая логика.  
Применяются  
вакуумные лампы  
(электронные лампы).**





Ламповый компьютер? Это просто! (2020)  
<https://www.youtube.com/watch?v=fQ3Wv26qflg>



Хроники лампового компьютера: Все красиво, все сломали (2024)  
<https://www.youtube.com/watch?v=fVTqflfuy8>

YouTube 29:02

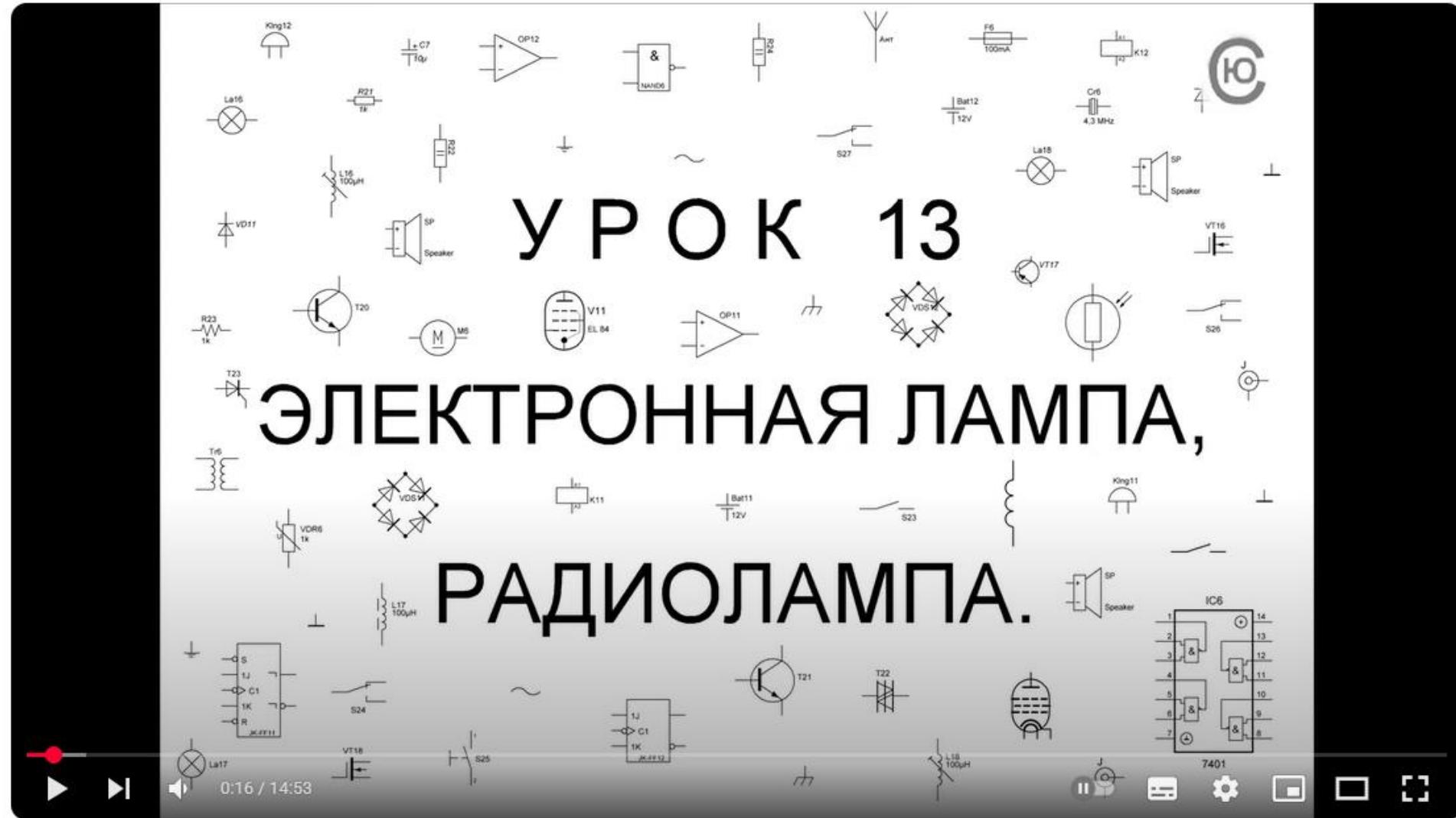


Как это сделано? Ламповые компьютеры (2020)  
<https://www.youtube.com/watch?v=LRdius6-Uz4>



Лампы-транзисторы-микросхемы (2016)

<https://www.youtube.com/watch?v=L-LrApms9ko>



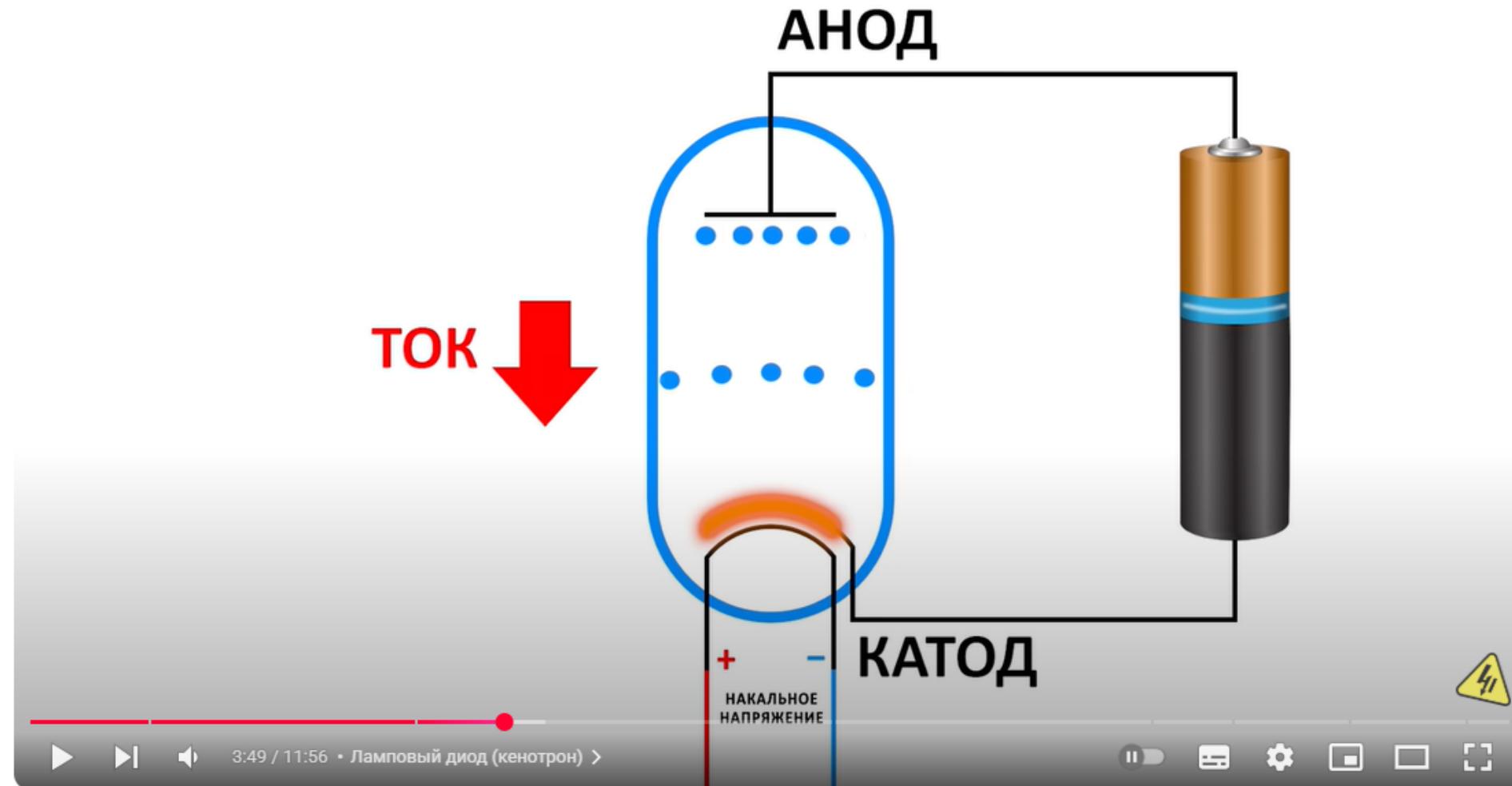
Урок 13. Электронная лампа. Радиолампа (2020)  
<https://www.youtube.com/watch?v=92tspHm4XFc>

Урок №48

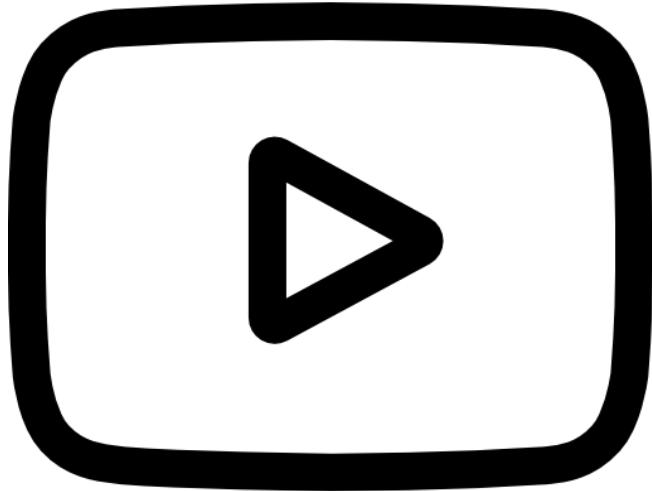
# Радиолампа



Урок №48. Радиолампа (2019)  
<https://www.youtube.com/watch?v=cSXAJw5Y9B0>



ЛАМПОВОЕ УСИЛЕНИЕ. Как устроена РАДИОЛАМПА? Понятное объяснение! (2023)  
<https://www.youtube.com/watch?v=AJiel4dwtLA>



## Резисторно-транзисторная логика

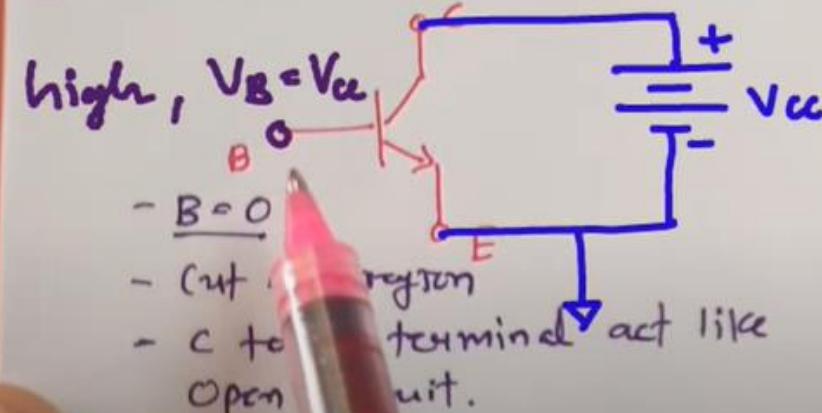
### Resistor-Transistor Logic (RTL).

Применяются биполярные  
транзисторы.



## Resistor Transistor Logic [RTL]

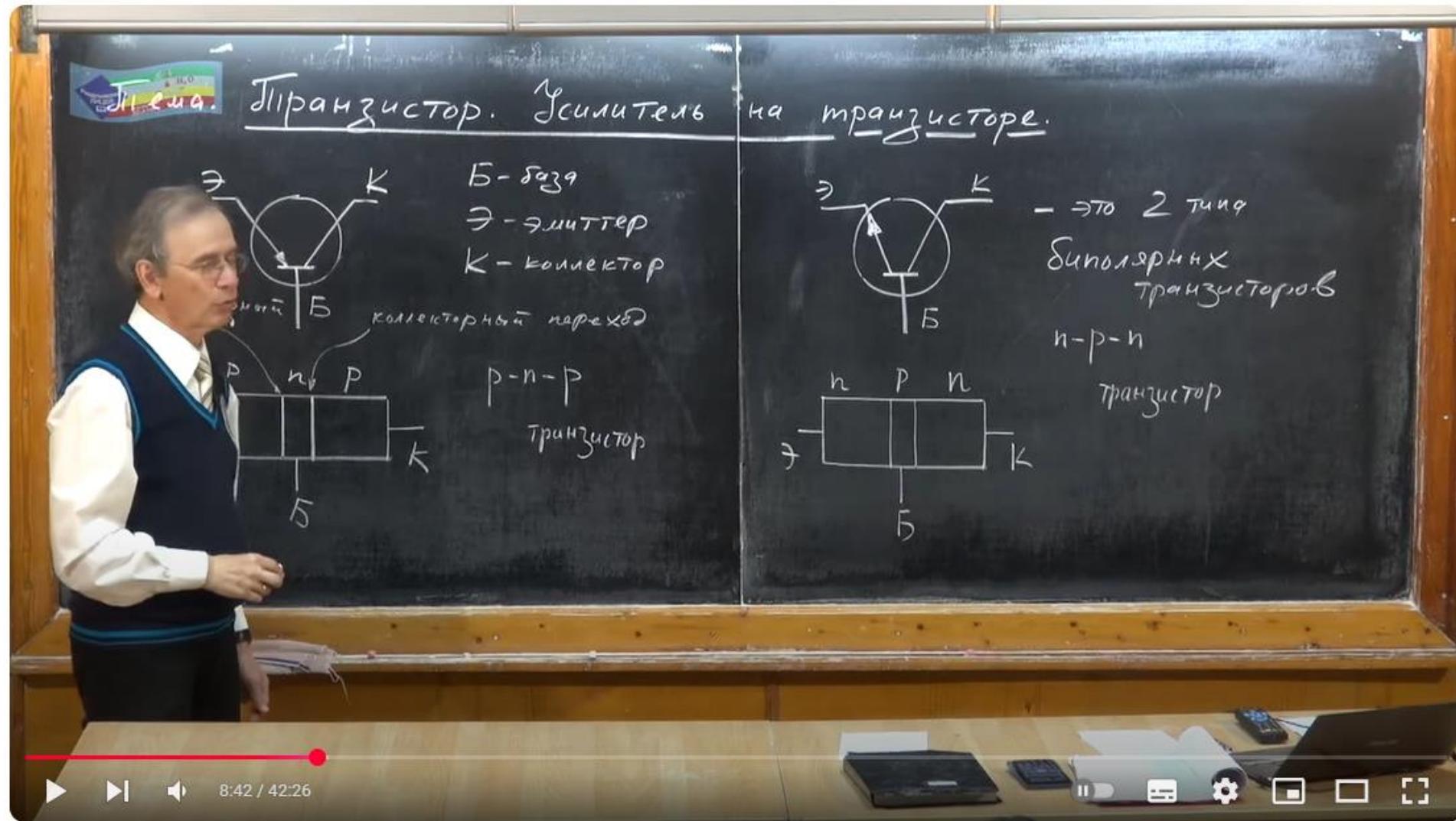
- This logic family includes resistors and transistors in its Integrated Circuit. So, it is referred as RTL family.
- Here we use transistors in saturation and cut off regions. so, speed of this logic family is low.



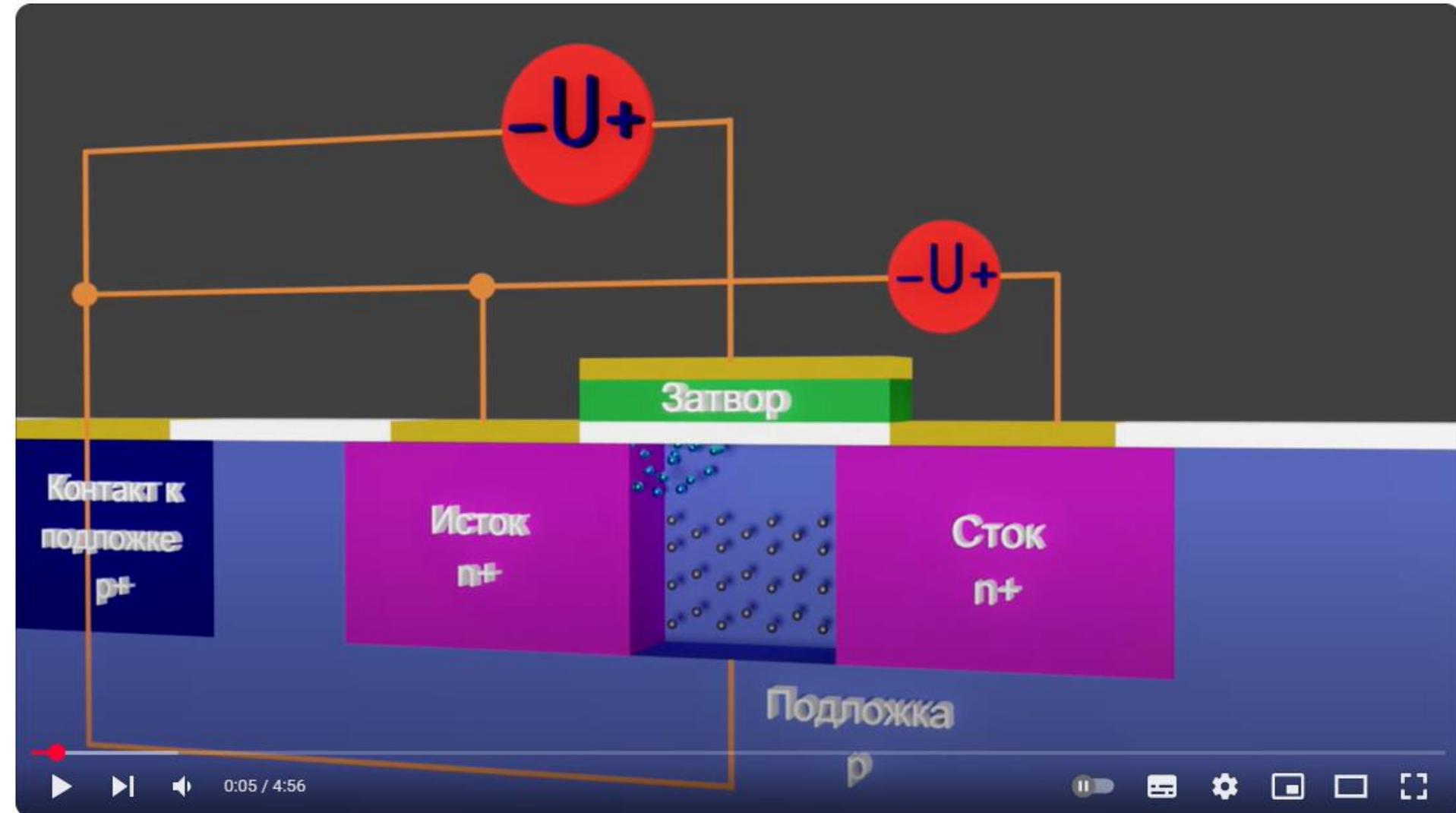
2:05 / 12:47 • Operating Region of Transistor in RTL &gt;



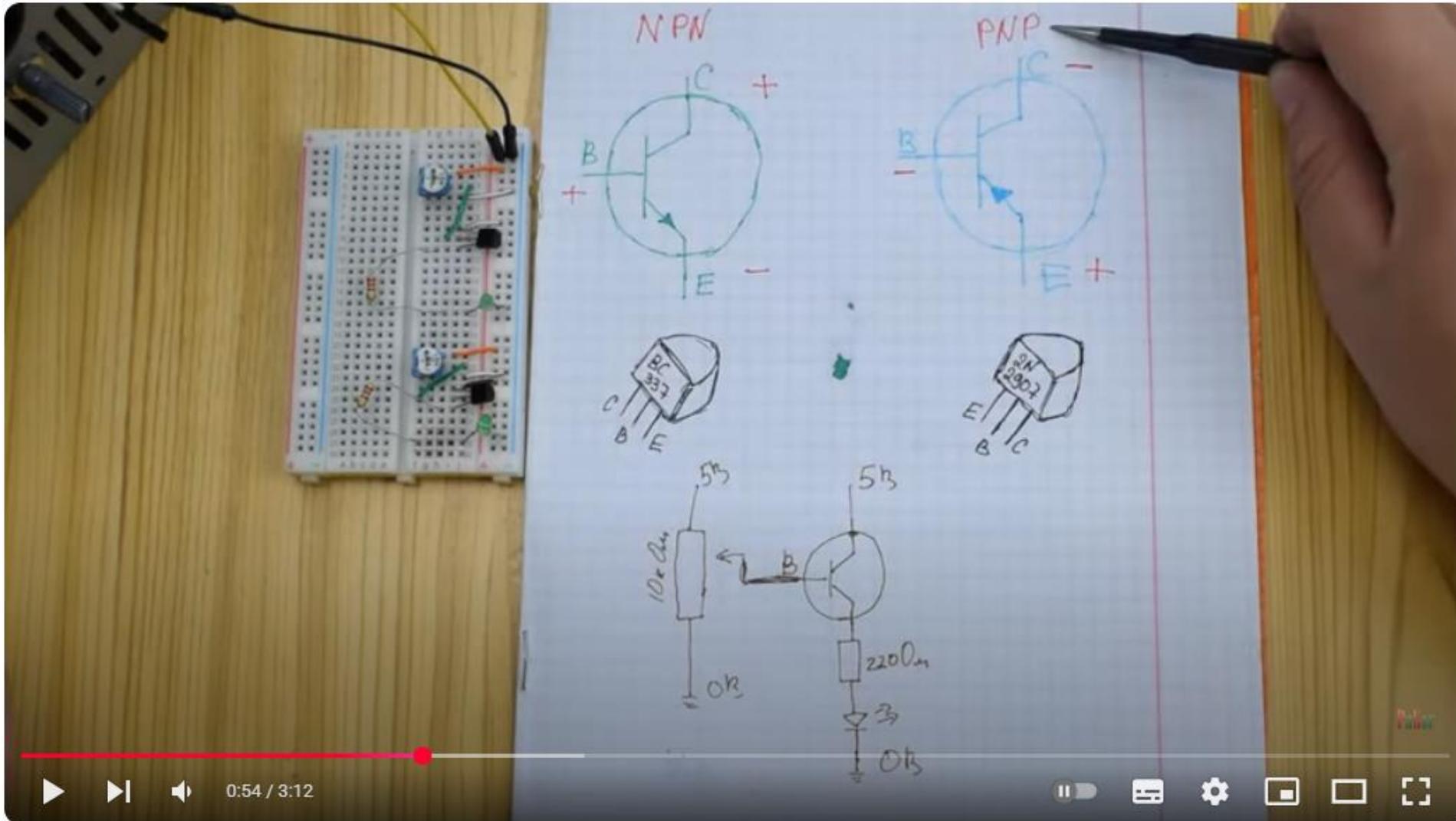
Resistor Transistor Logic Basics: RTL NOT Gate and RTL NOR Gate (2021)  
 Основы резистивно-транзисторной логики: вентиль RTL NOT и вентиль RTL NOR (2021)  
<https://www.youtube.com/watch?v=hvy4YASs-CQ>



Урок 308. Транзистор. Усилитель на транзисторе (2016)  
<https://www.youtube.com/watch?v=kBolzQc4j3c>

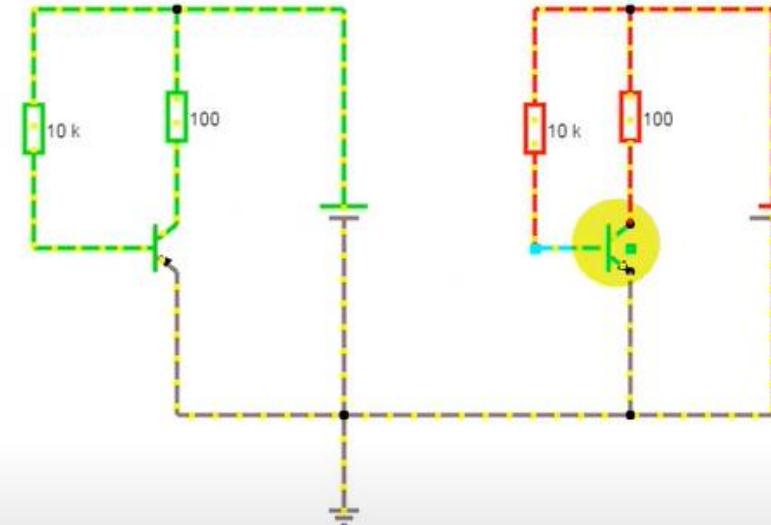


Принцип работы биполярного транзистора(2025)  
<https://www.youtube.com/watch?v=sas3zVZudDc>



В чем разница между PNP и NPN транзисторами? (2018)  
<https://www.youtube.com/watch?v=rHnJnnMtARA>

Reset  
Stopped  
Simulation Speed  
Current Speed  
Power Brightness

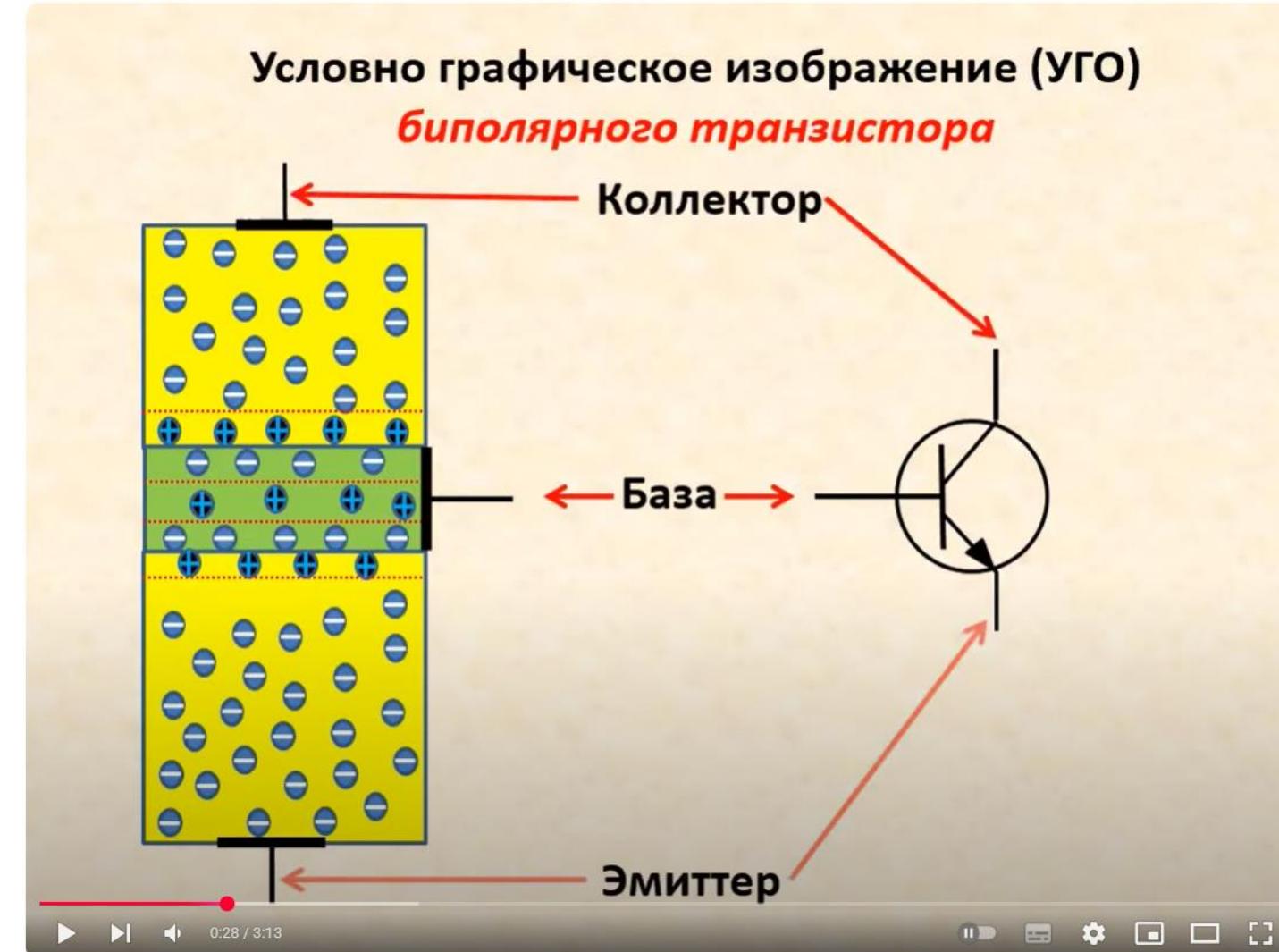


transistor (NPN) beta=100  
cutoff  
 $I_c = 0 \text{ A}$   
 $I_b = 0 \text{ A}$   
 $V_{be} = -5 \text{ V}$   
 $V_{bc} = 0 \text{ V}$   
 $V_{ce} = -5 \text{ V}$



## Урок №19. NPN и PNP биполярные транзисторы (2015)

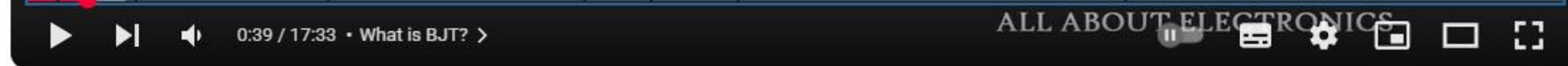
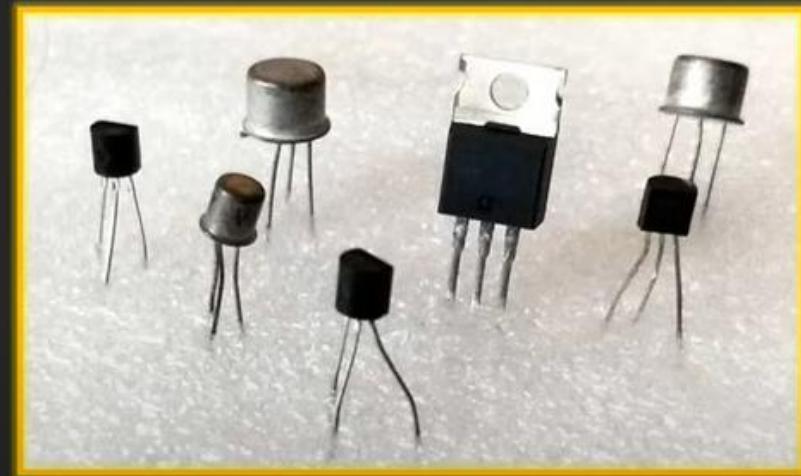
<https://www.youtube.com/watch?v=yQB2EnEwIIO>



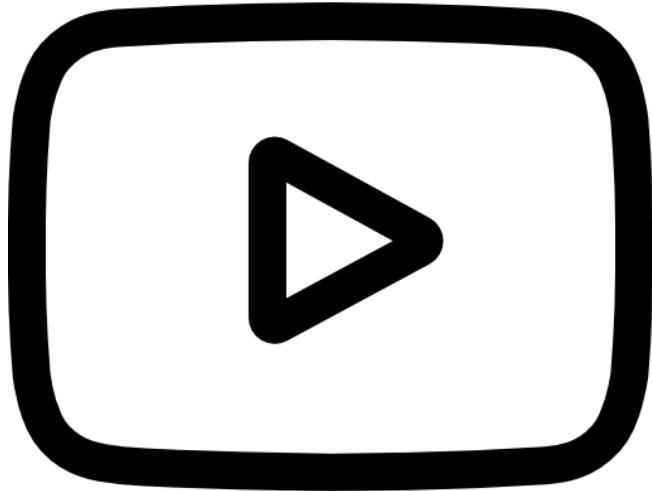
Биполярный транзистор (2020)  
[https://www.youtube.com/watch?v=LgsdYwxSA\\_k](https://www.youtube.com/watch?v=LgsdYwxSA_k)



## Bipolar Junction Transistor (BJT)



Introduction to Bipolar Junction Transistor (BJT) (2019)  
Знакомство с биполярным транзистором (BJT) (2019)  
<https://www.youtube.com/watch?v=-VwPSDQmdjM>

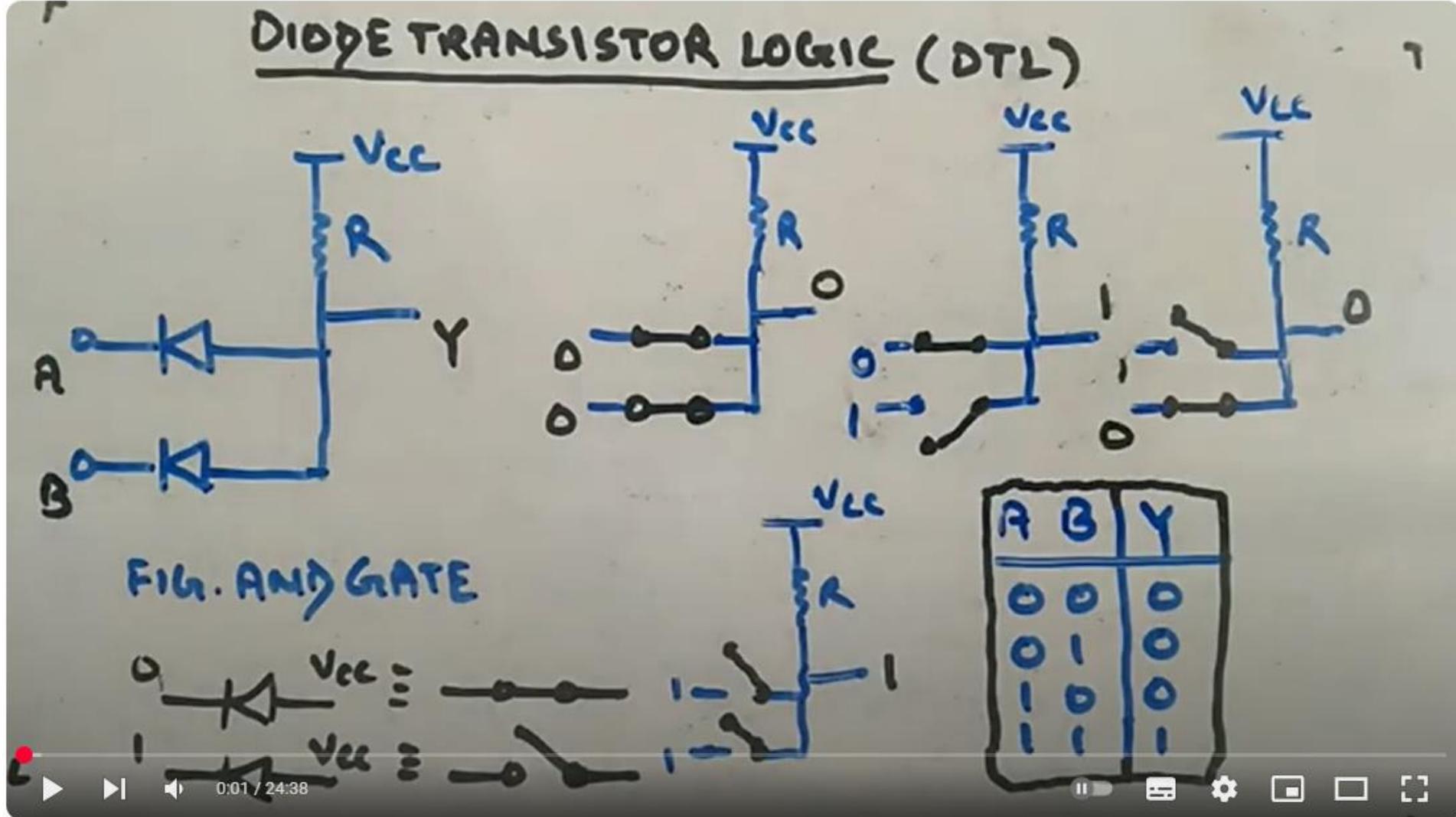


# Диодно-транзисторная логика

## Diode Transistor Logic (DTL)

Используются биполярные  
транзисторы (BJT) и  
диоды.





Diode Transistor Logic (DTL) - Digital Circuits and Logic Design (2022)  
Диодная транзисторная логика (DTL) — цифровые схемы и проектирование логических схем (2022)  
<https://www.youtube.com/watch?v=WD-Ipu-lvAM>

DIODE TRANSISTOR LOGIC (DTL)

SOLUTION  $I_{CCH} - Y=1$

$$I_y = 0$$

$$-5V + 5k \Omega I_R + 0.7 + 0.2 = 0$$

$$I_R = \frac{5 - 0.7 - 0.2}{5k \Omega}$$

$$= 0.82mA$$

$$I_{CCH} = I_y + I_R = .82mA$$

$V_{OL(H)} = 0.7V$   $V_{CE(SAT)} = 0.2V$

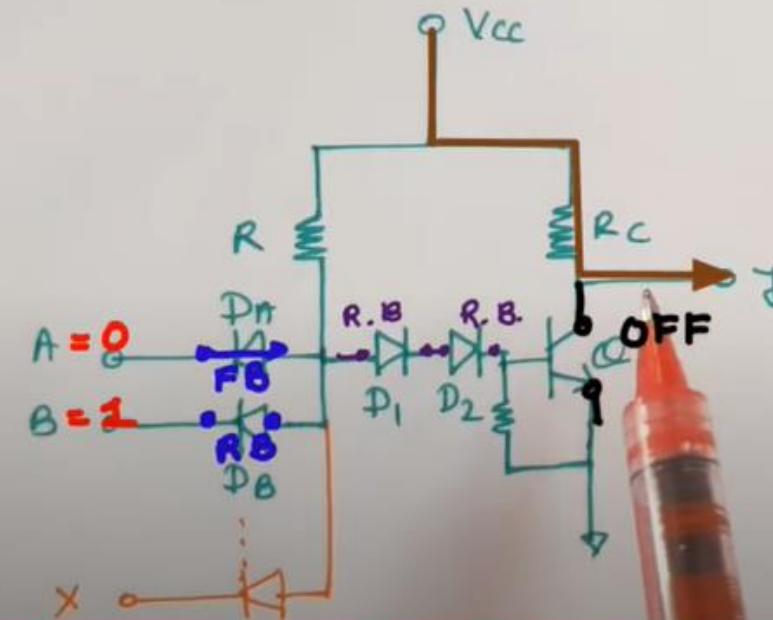
~~$V_{BE(SAT)} = 0.7V$   $P_D = ?$~~

5:35 / 9:41

Question on Diode Transistor Logic (DTL) - Digital Circuits and Logic Design (2022)  
 Вопрос о диодной транзисторной логике (ДТЛ) — цифровые схемы и проектирование логических схем (2022)  
<https://www.youtube.com/watch?v=1e3L9ZTMn-E>

## Diode Transistor Logic [DTL]

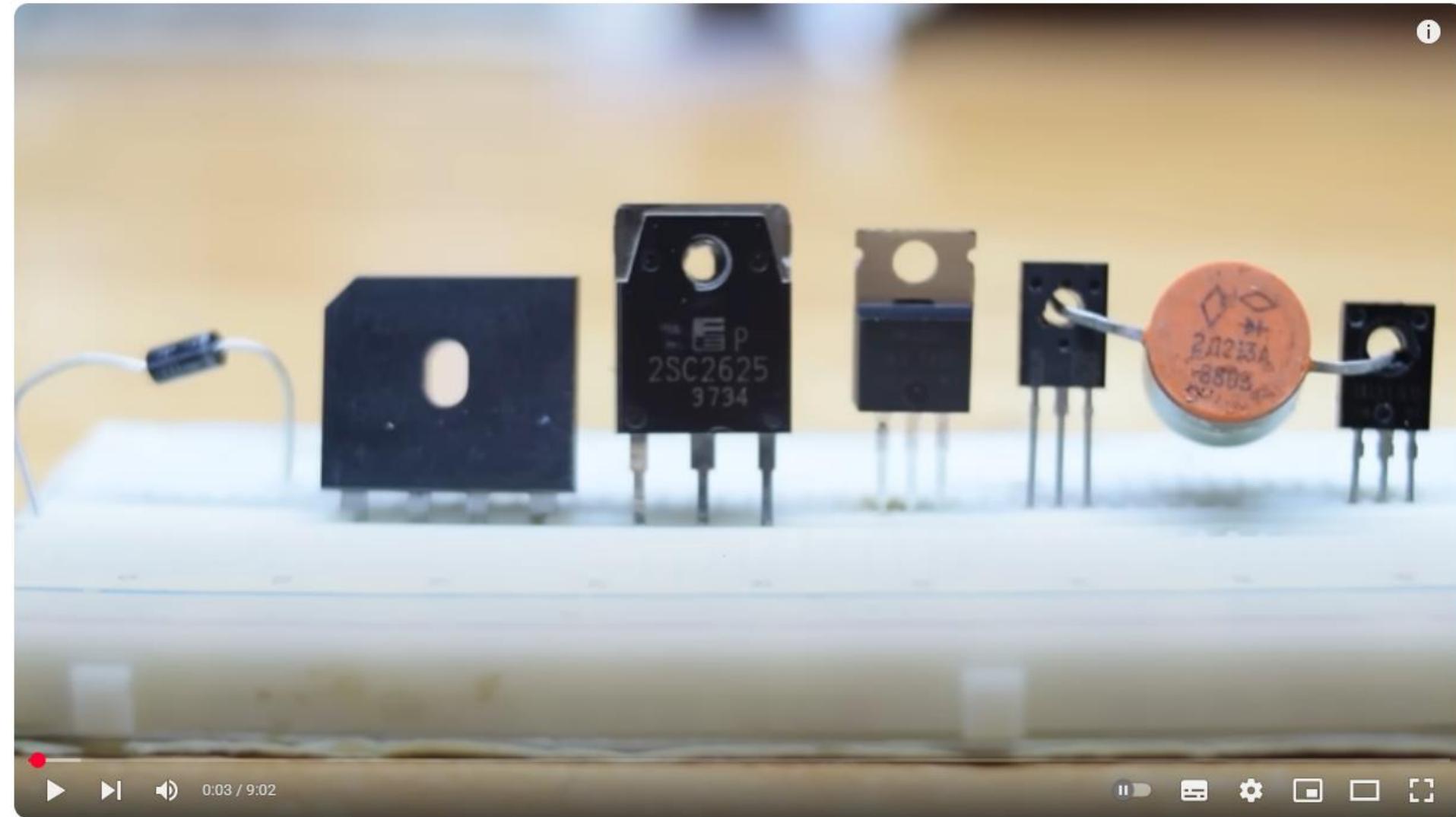
- RTL families has low NM, low fan out, slow speed & higher power dissipation. so, we don't use RTL in recent IC's.
- DTL has improved NM and Fan Out compared to RTL.



| A | B | Y |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

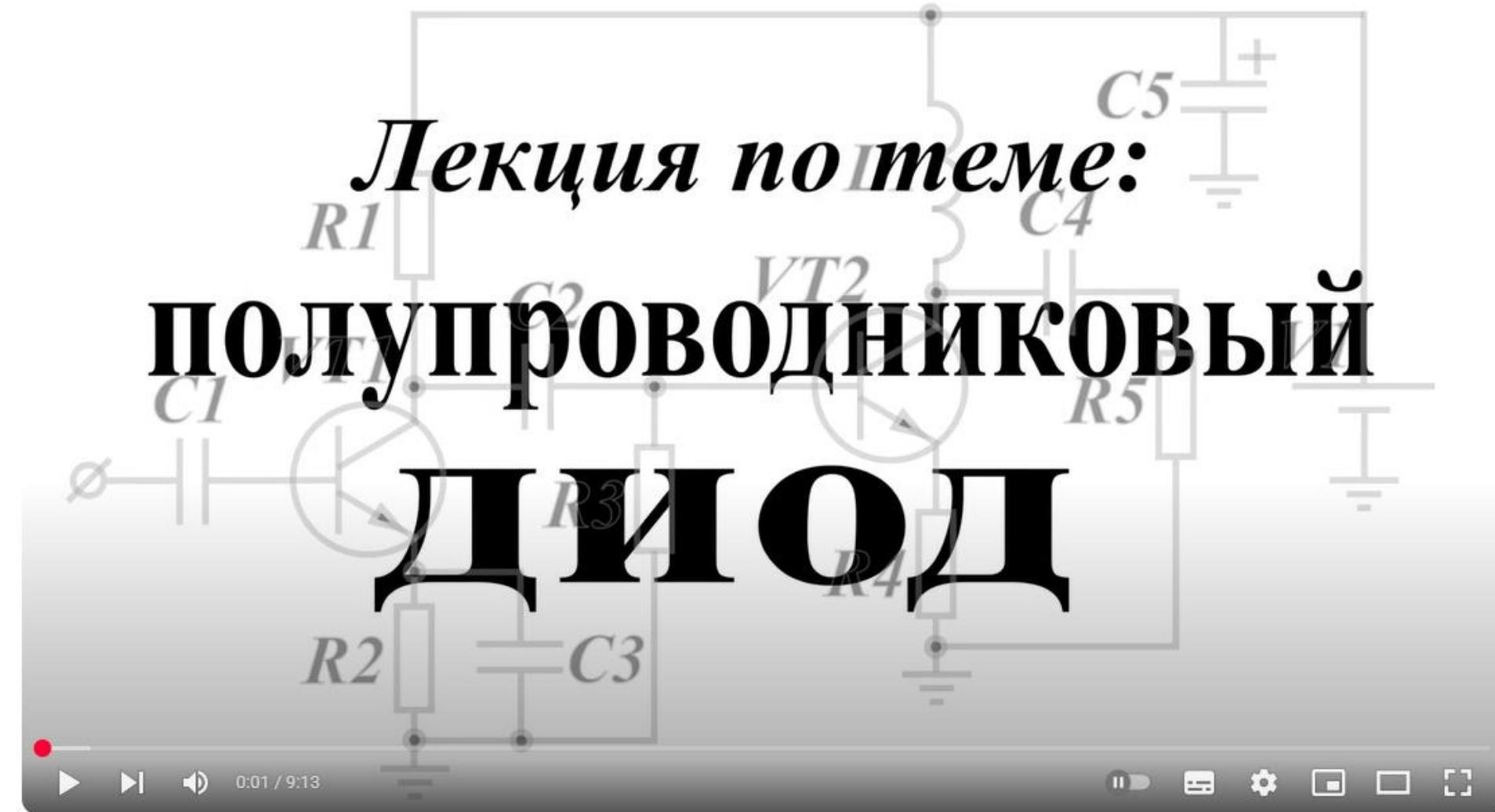
5:27 / 9:24 • DTL NAND Gate Working

Diode Transistor Logic (DTL): DTL NAND Gate Circuit and Working (2021)  
Диодная транзисторная логика (ДТЛ): схема и принцип работы вентиля ДТЛ И-НЕ (2021)  
<https://www.youtube.com/watch?v=bGsque2SIEQ>

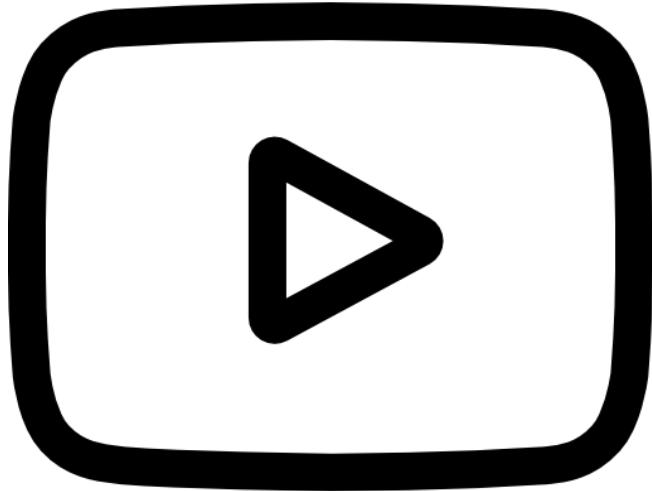


Полупроводники. Как работают транзисторы и диоды. Самое понятное объяснение (2019)  
<https://www.youtube.com/watch?v=OMGdSCaMVDO>

# Лекция по теме: полупроводниковый диод



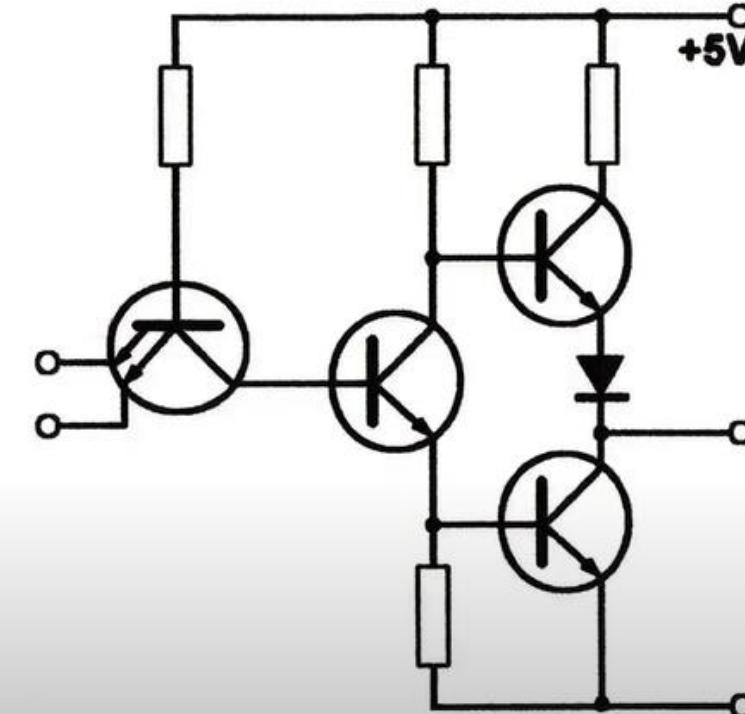
Принцип работы полупроводникового диода. ВАХ диода (2022)  
<https://www.youtube.com/watch?v=v5dzA8AmcWM>



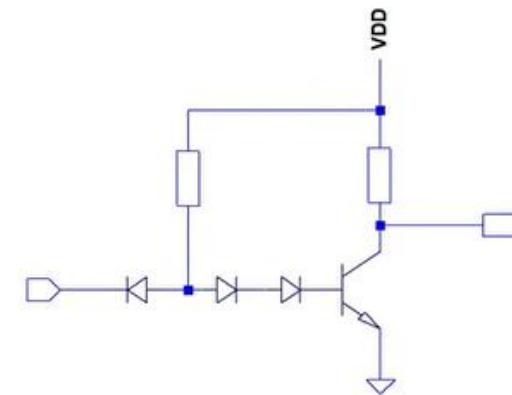
**Транзисторно-транзисторная логика  
TTL (Transistor-Transistor Logic)**  
**Используются биполярные транзисторы (BJT)**



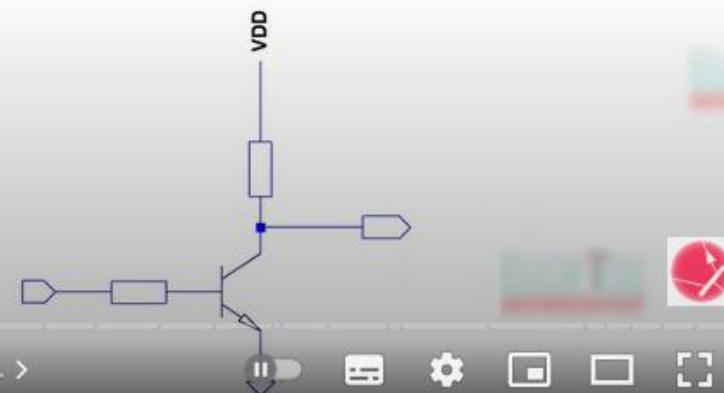
# ТТЛ-логика



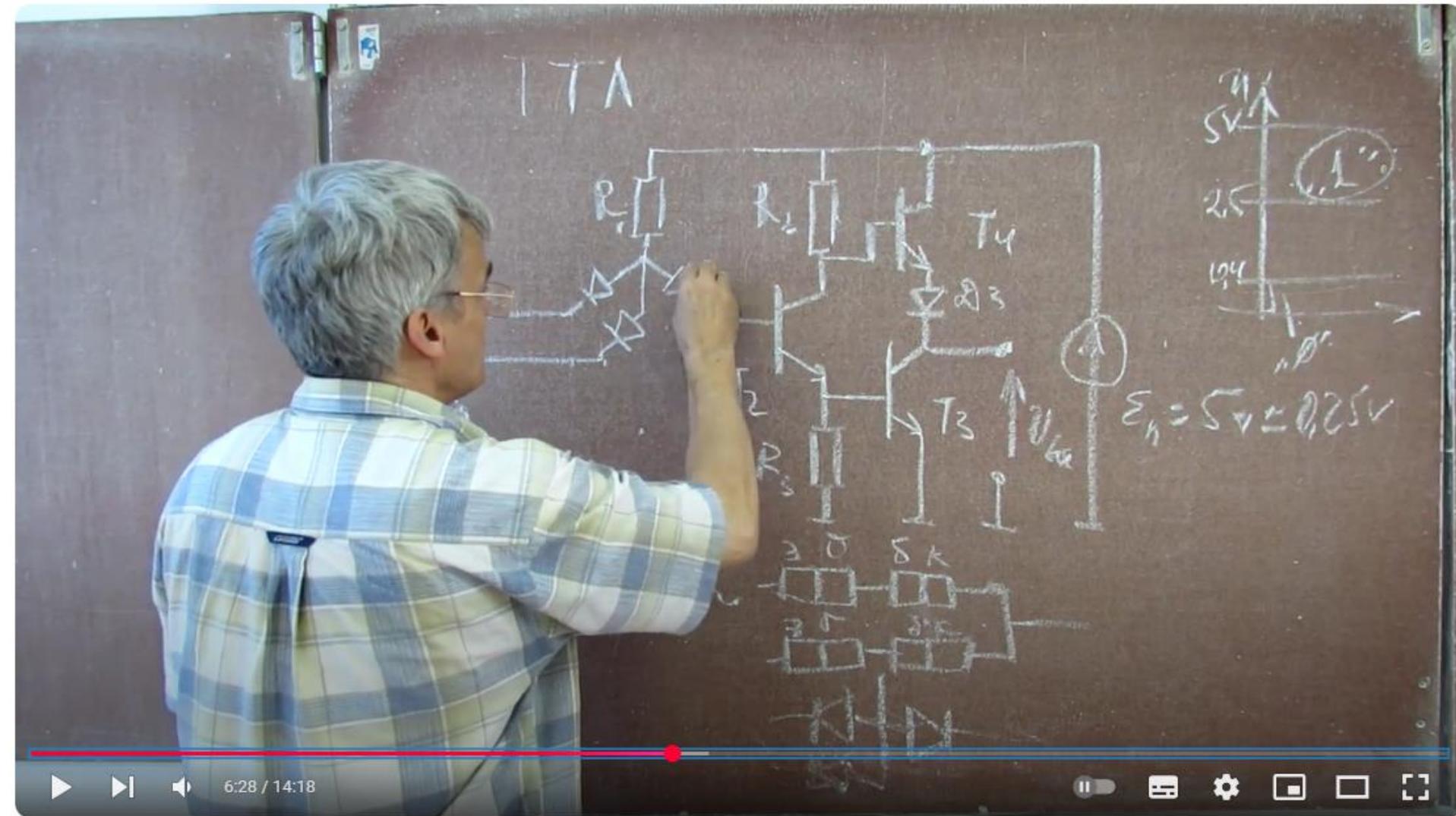
**DTL** ДИОДНО-ТРАНЗИСТОРНАЯ ЛОГИКА



**RTL** РЕЗИСТОРНО-ТРАНЗИСТОРНАЯ ЛОГИКА



Логические элементы И, ИЛИ, Исключающее ИЛИ. История, Теория, Применение (2021)  
<https://www.youtube.com/watch?v=bXdiYU3IUJA>



## Лекция 74. Транзисторно-транзисторная логика (ТТЛ) (2013)

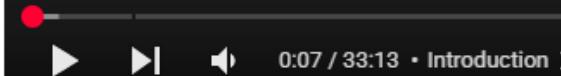
[https://www.youtube.com/watch?v=qOtHc\\_x70hY](https://www.youtube.com/watch?v=qOtHc_x70hY)



## Digital Electronics

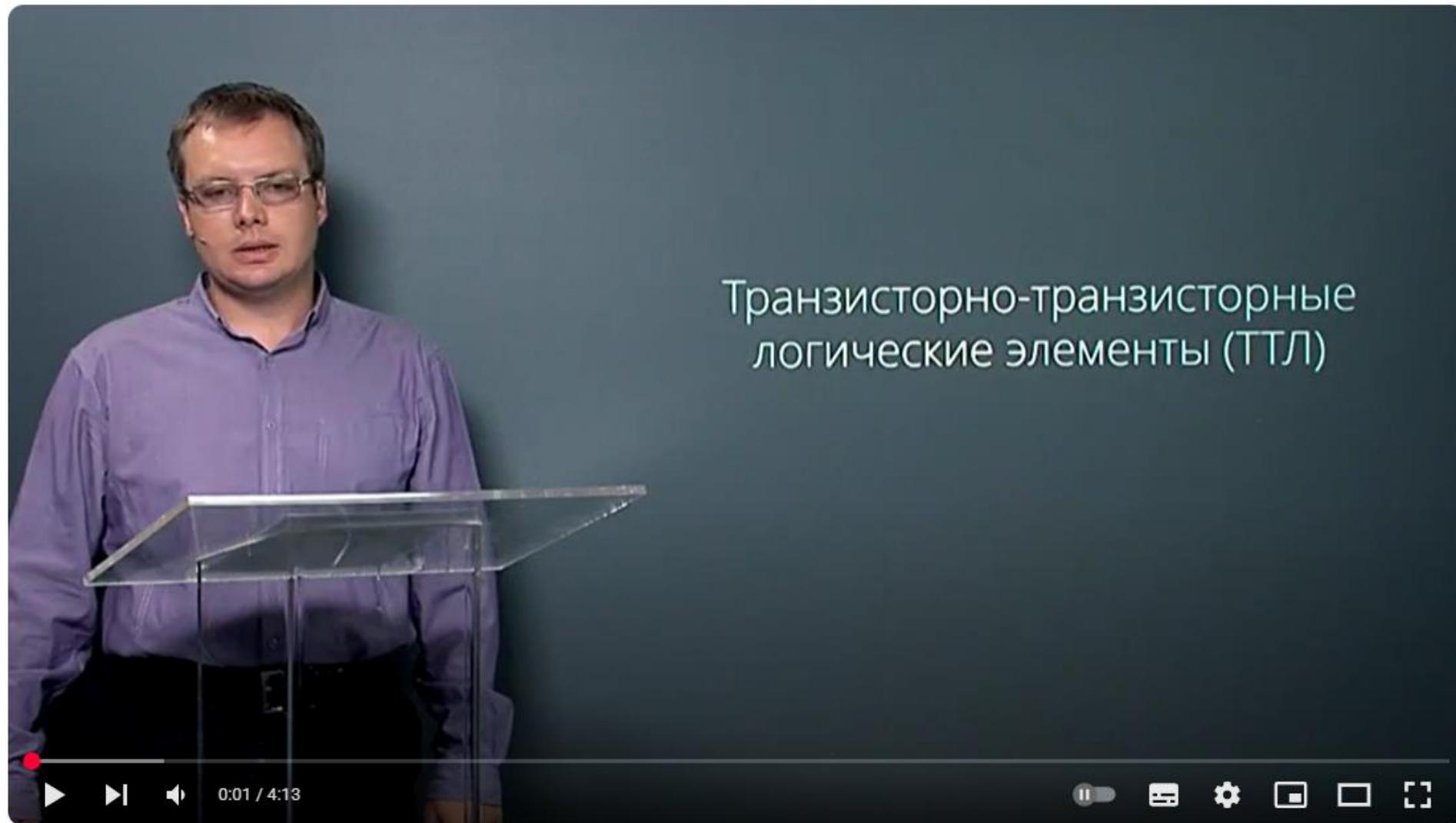
### Transistor Transistor Logic (TTL)

Hey friends, welcome to the YouTube channel ALL ABOUT ELECTRONICS.



ALL ABOUT ELECTRONICS

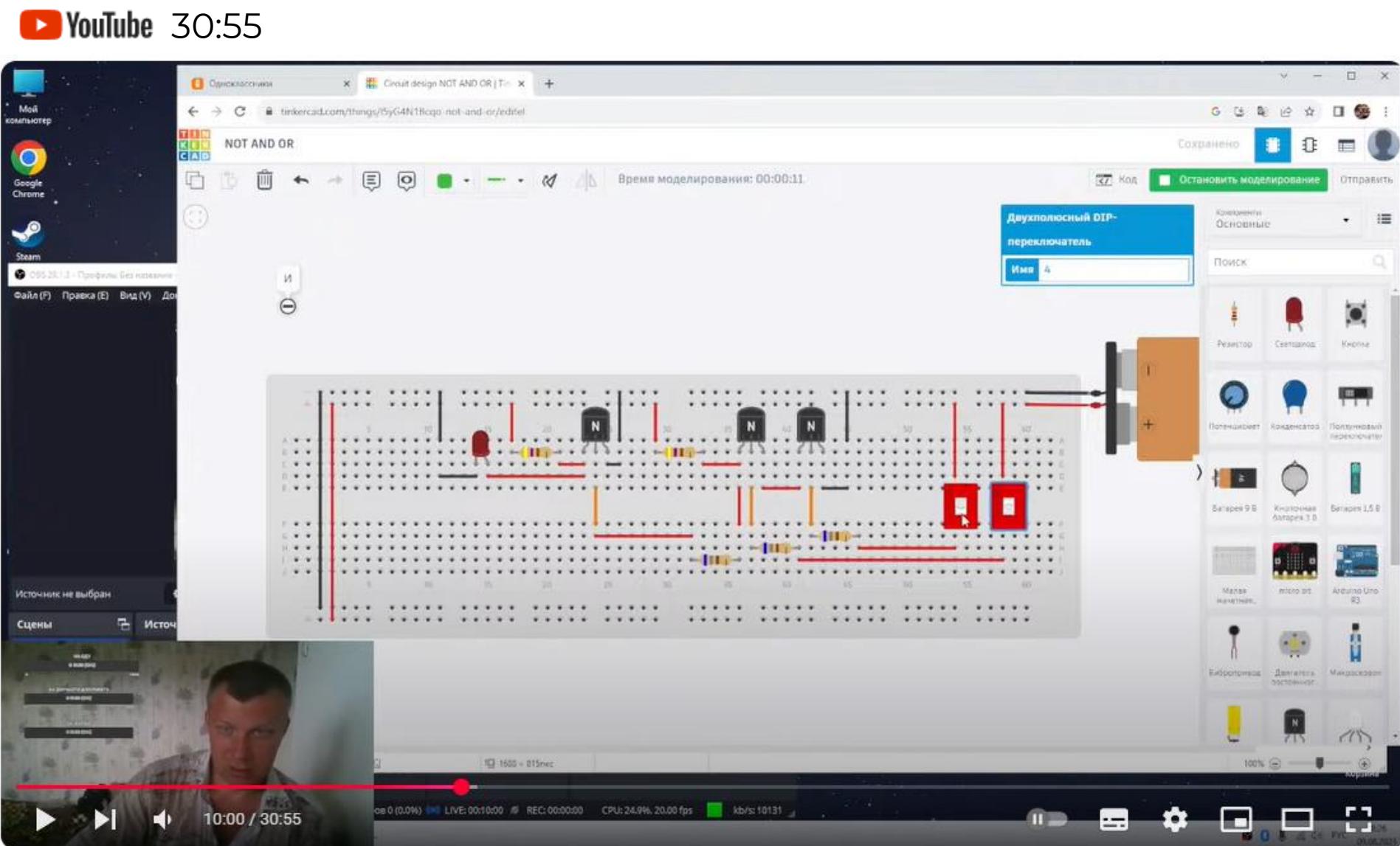
TTL Logic Explained | TTL Inverter Circuit | Noise Margin and Fanout of TTL Circuits (2025)  
Объясненная логика TTL | Схема TTL-инвертора | Уровень шума и разветвление TTL-цепей (2025)  
[https://www.youtube.com/watch?v=I74vWLGif\\_o](https://www.youtube.com/watch?v=I74vWLGif_o)



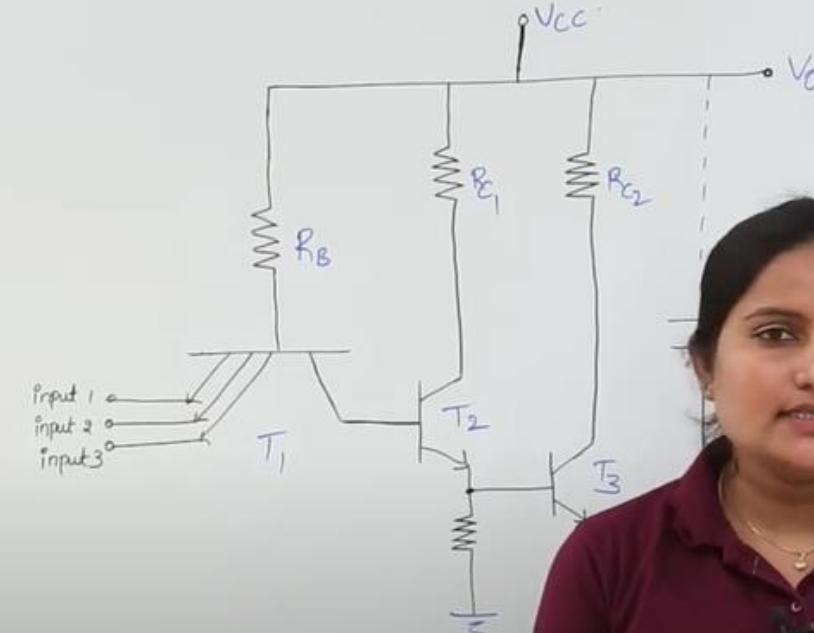
4.2.2 Транзисторно транзисторные логические элементы (2017)  
<https://www.youtube.com/watch?v=x1gEFFctTLk>



Элементы транзисторно транзисторной логики (2014)  
<https://www.youtube.com/watch?v=8tolhnUvN2A>

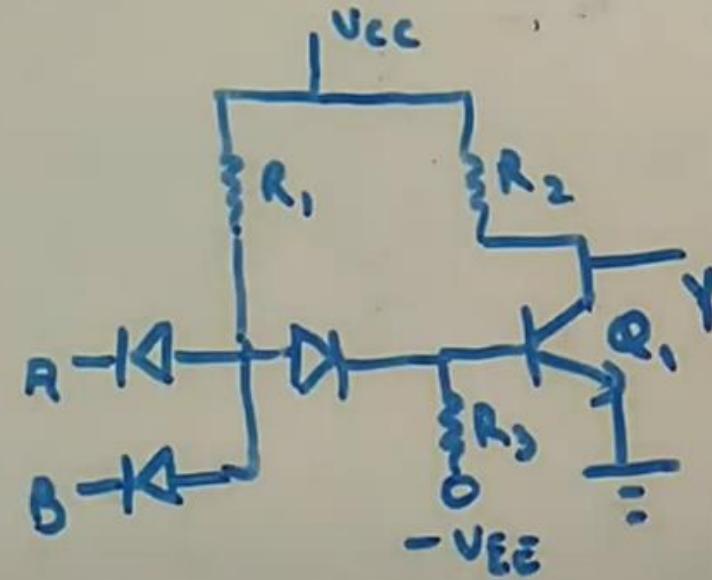


Занятие 7: TTL Транзисторно-транзисторная логика (2023)  
[https://www.youtube.com/watch?v=684AaZq\\_-I](https://www.youtube.com/watch?v=684AaZq_-I)

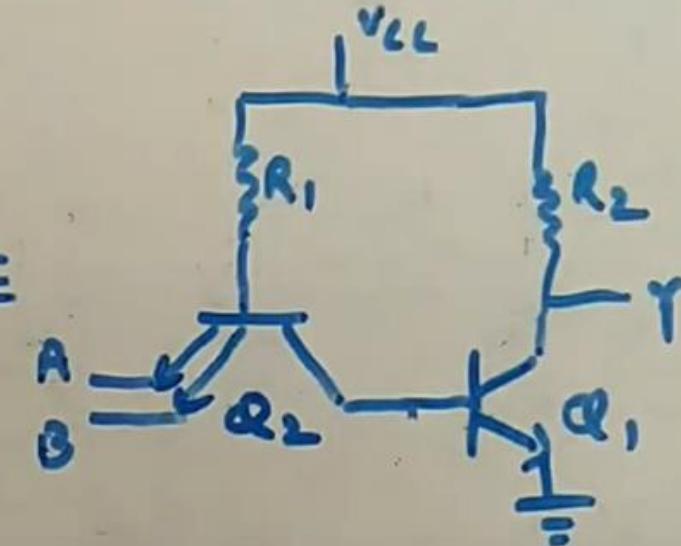
TRANSISTOR- TRANSISTOR LOGIC (TTL)

Transistor Transistor Logic (2018)  
Транзисторно-транзисторная логика (2018)  
<https://www.youtube.com/watch?v=sW1FEm2yNnA>

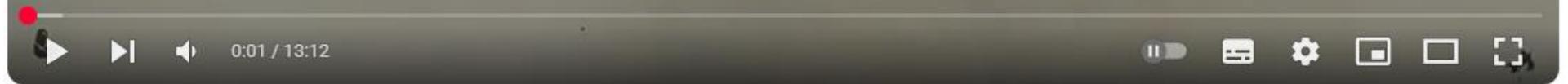
## TRANSISTOR TRANSISTOR LOGIC



**FIG. DTL NAND**



**FIG. TTL NAND**



Transistor Transistor Logic (TTL) - Digital Circuits and Logic Design (2022)  
 Транзисторная транзисторная логика (ТТЛ) — цифровые схемы и логическое проектирование (2022)  
<https://www.youtube.com/watch?v=hcyC5EzOVuU>



## Digital Electronics

### TTL NAND and NOR gates

The image shows a YouTube video player interface. At the top left is the YouTube logo and the time '16:32'. The title 'Digital Electronics' is in white at the top right. A large yellow text box in the center contains the title 'TTL NAND and NOR gates'. The video progress bar at the bottom shows a red playhead at 0:08 of a 16:32 minute video. Below the progress bar are standard YouTube controls: play, pause, volume, and a search bar with the text 'TTL NAND gate internal circuit and its working >'. To the right of the search bar is the channel name 'ALL ABOUT ELECTRONICS' followed by icons for settings, notifications, and sharing. The bottom of the video frame features a decorative border with a red sunburst graphic.

TTL Logic: TTL NAND and NOR gates Explained (2025)  
<https://www.youtube.com/watch?v=sDn2hi5ndCA>

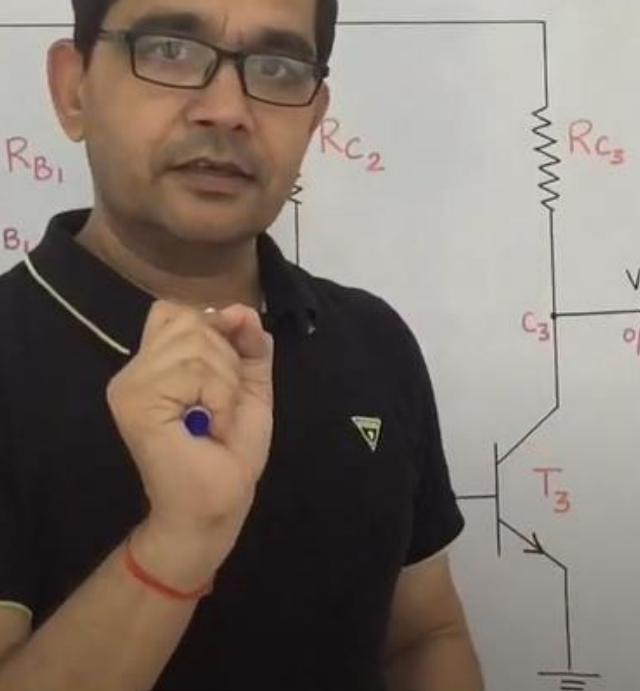
YouTube 13:06



Hackaday Logic Series: TTL Electrical characteristics (2015)  
<https://www.youtube.com/watch?v=b7koDIN4m-Q>

Logic 0 = 0.2V ( $V_{CEsat}$ )

Logic 1 = 5V

Transistor-Transistor Logic (TTL)RTL

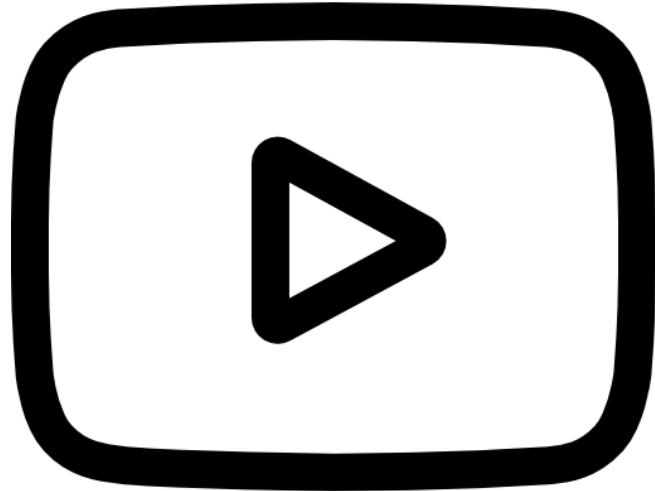
- Poor Noise Margin
- Poor Fan Out
- Low Speed
- High Power Dissip.

DTL

- Improved Noise Margin & Fan Out
- Slow speed



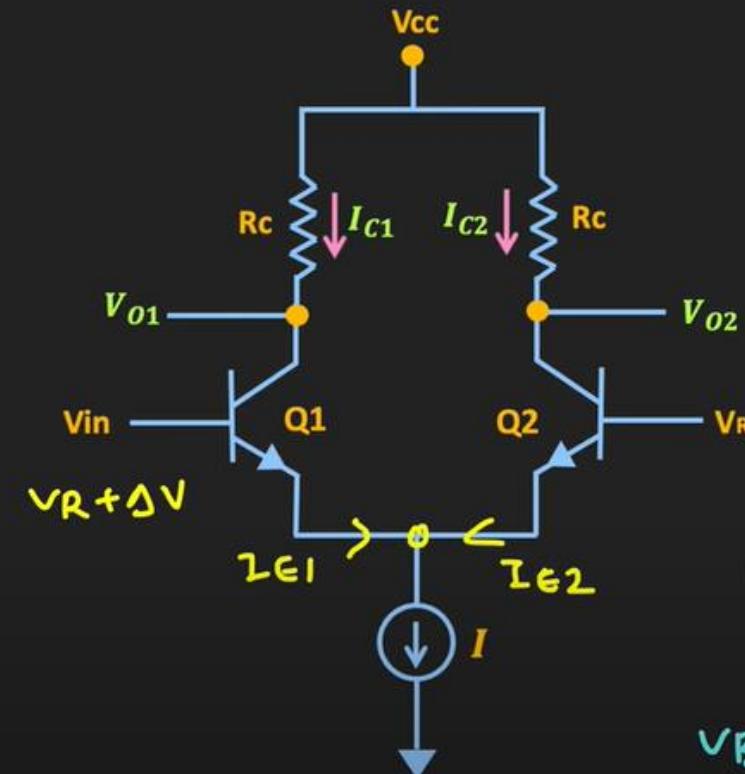
7. TTL (Transistor -Transistor Logic) Circuit | Digital Logic Families | TECH GURUKUL By Dinesh Arya (2018)  
 7. Схема ТТЛ (транзисторно-транзисторной логики) | Семейства цифровых логических схем | TECH GURUKUL от Динеша Арья (2018)  
[https://www.youtube.com/watch?v=exM\\_4Vbsxl4](https://www.youtube.com/watch?v=exM_4Vbsxl4)



**Логика с эмиттерной  
связью, Эмиттерно-  
связанная логика  
Emitter-Coupled Logic (ECL)**



## Basic Element of ECL Logic



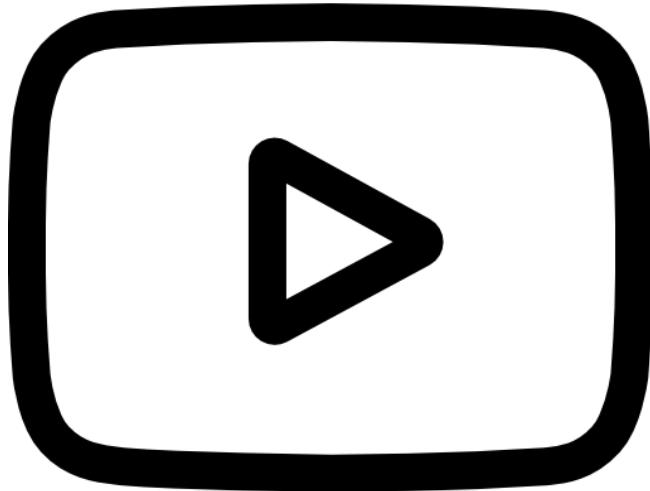
$$V_{BE1} - V_{BE2} = V_T \ln \left[ \frac{I_E1}{I_E2} \right]$$

$$V_{B1} - V_{B2} = V_T \ln \left[ \frac{I_E1}{I_E2} \right]$$

$$I_E1 + I_E2 = I$$

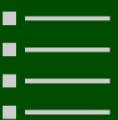
When  $V_{in} = V_R + \Delta V \Rightarrow I_E1 = 99\% \text{ of } I$

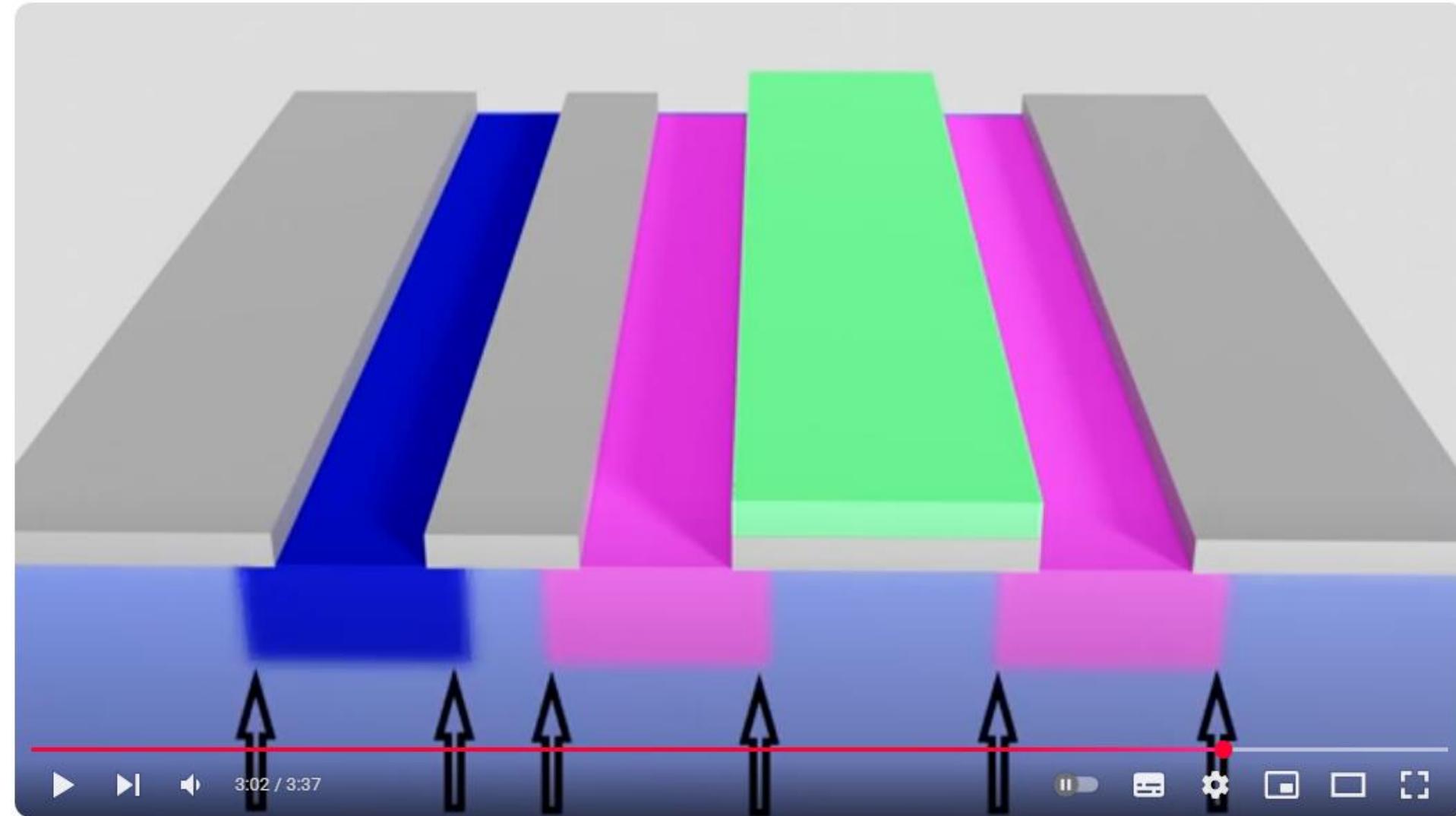
$$V_{B1} - V_{B2} = V_T \ln \left[ \frac{99}{1} \right] = 115mV$$



**Логика металл-оксид-  
полупроводник**

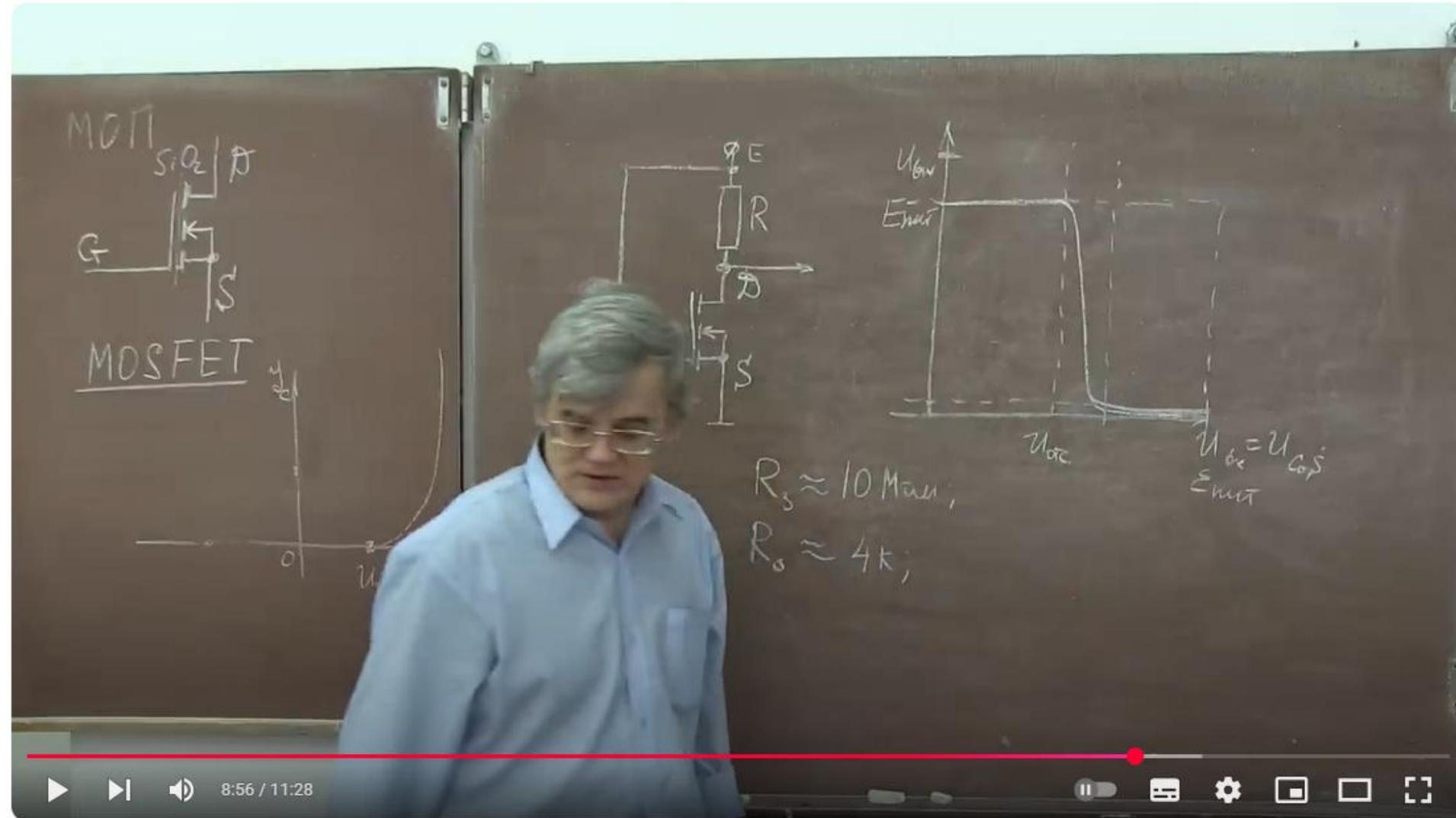
**Metal-Oxide-Semiconductor  
Logic (MOS)  
MOSFET**



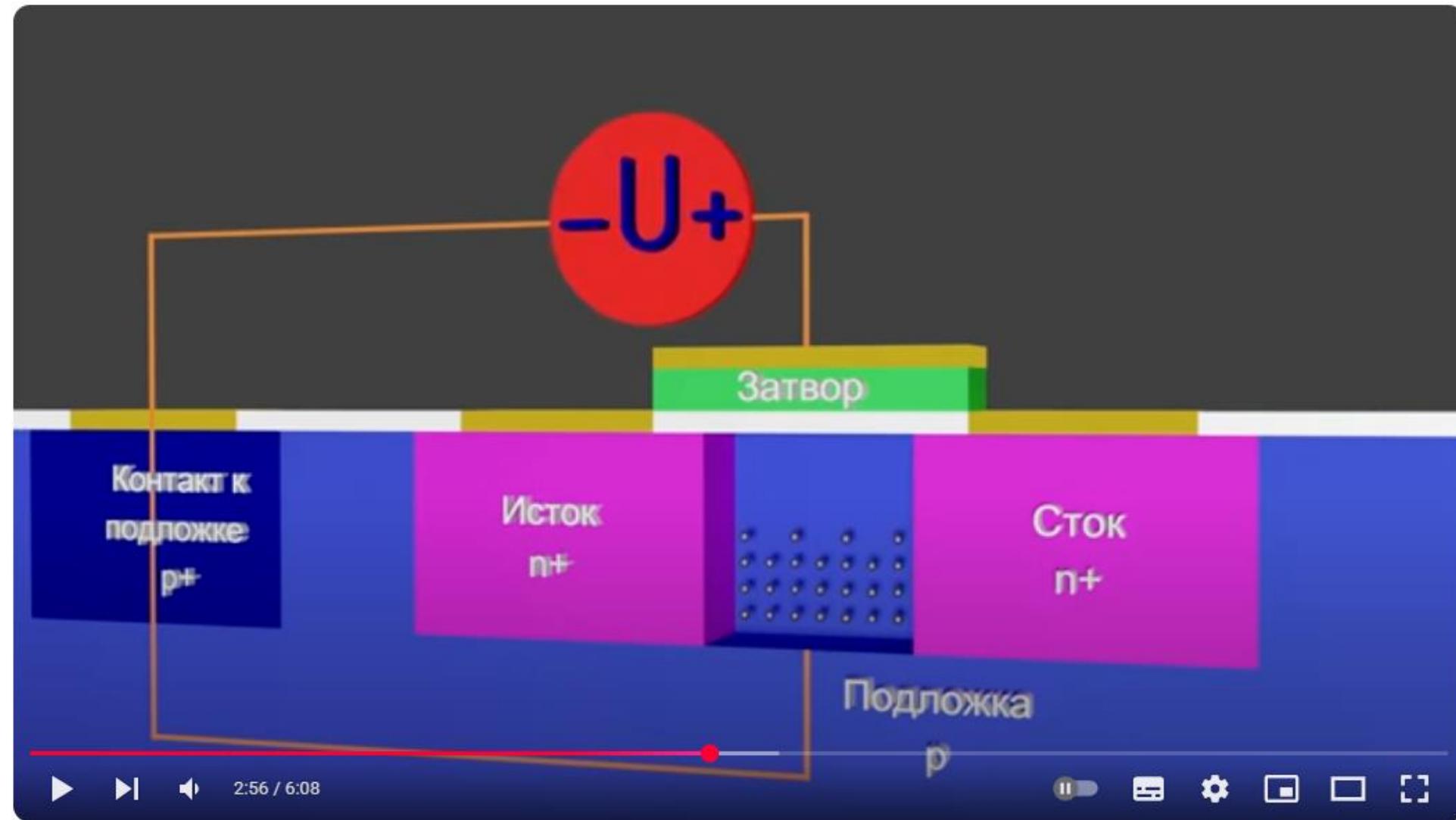


#### 4. Изготовление МОП-транзистора (2020)

<https://www.youtube.com/watch?v=CJxt3IOJX5k>

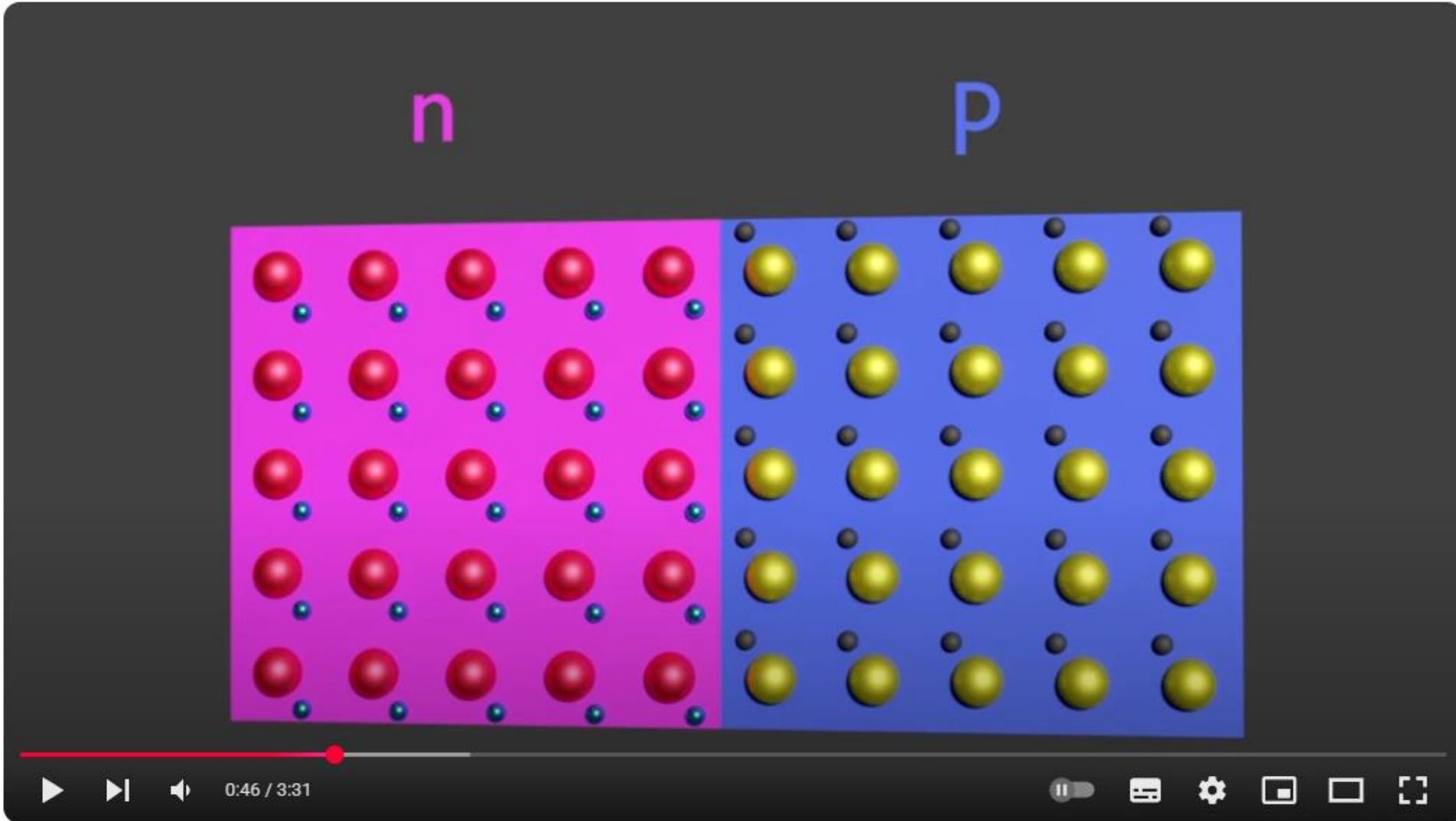


Лекция 137. МОП-транзистор (2014)  
<https://www.youtube.com/watch?v=uKPkLr1AQdc>

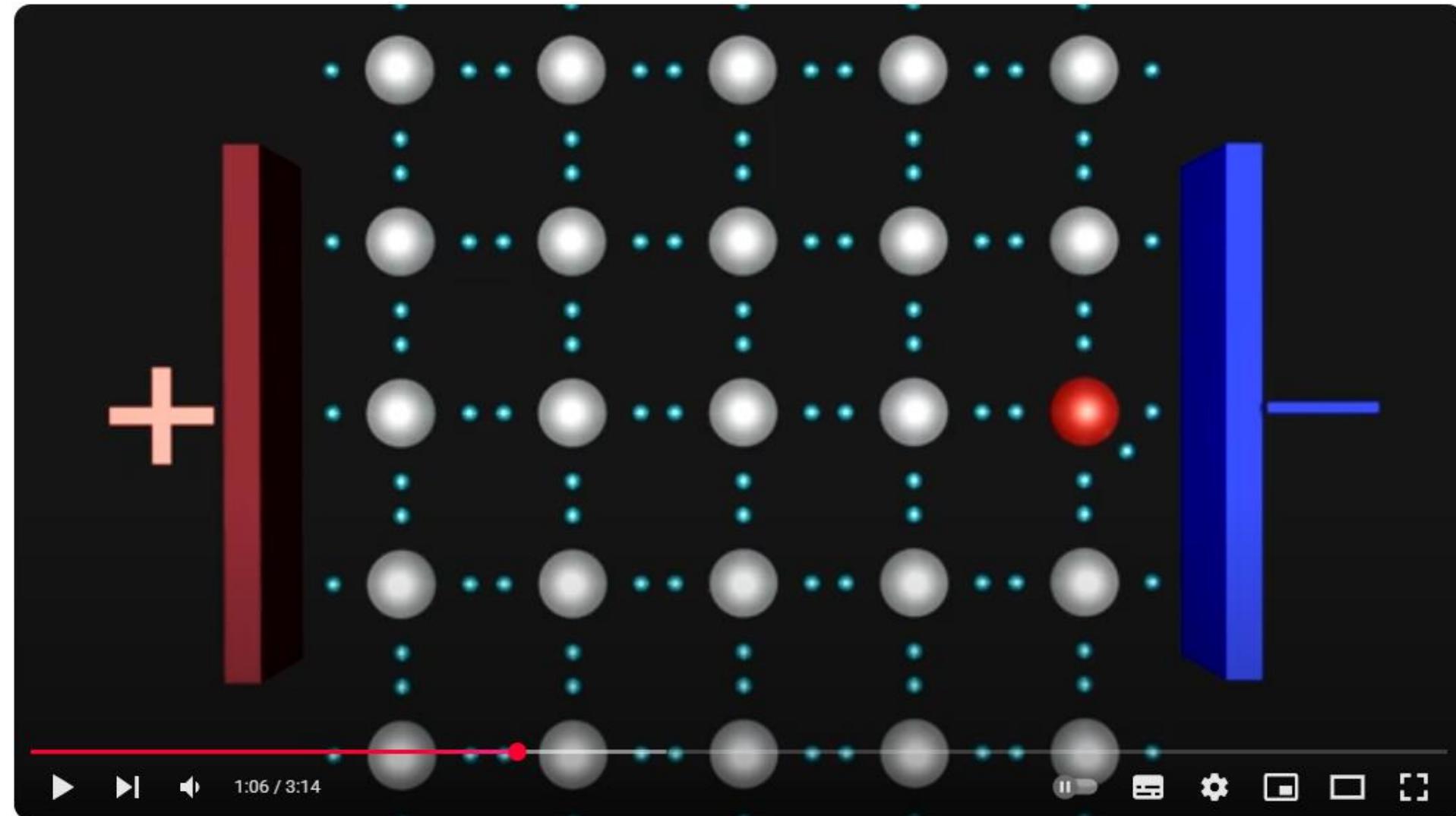


## 5. Принцип работы МОП-транзистора (2021)

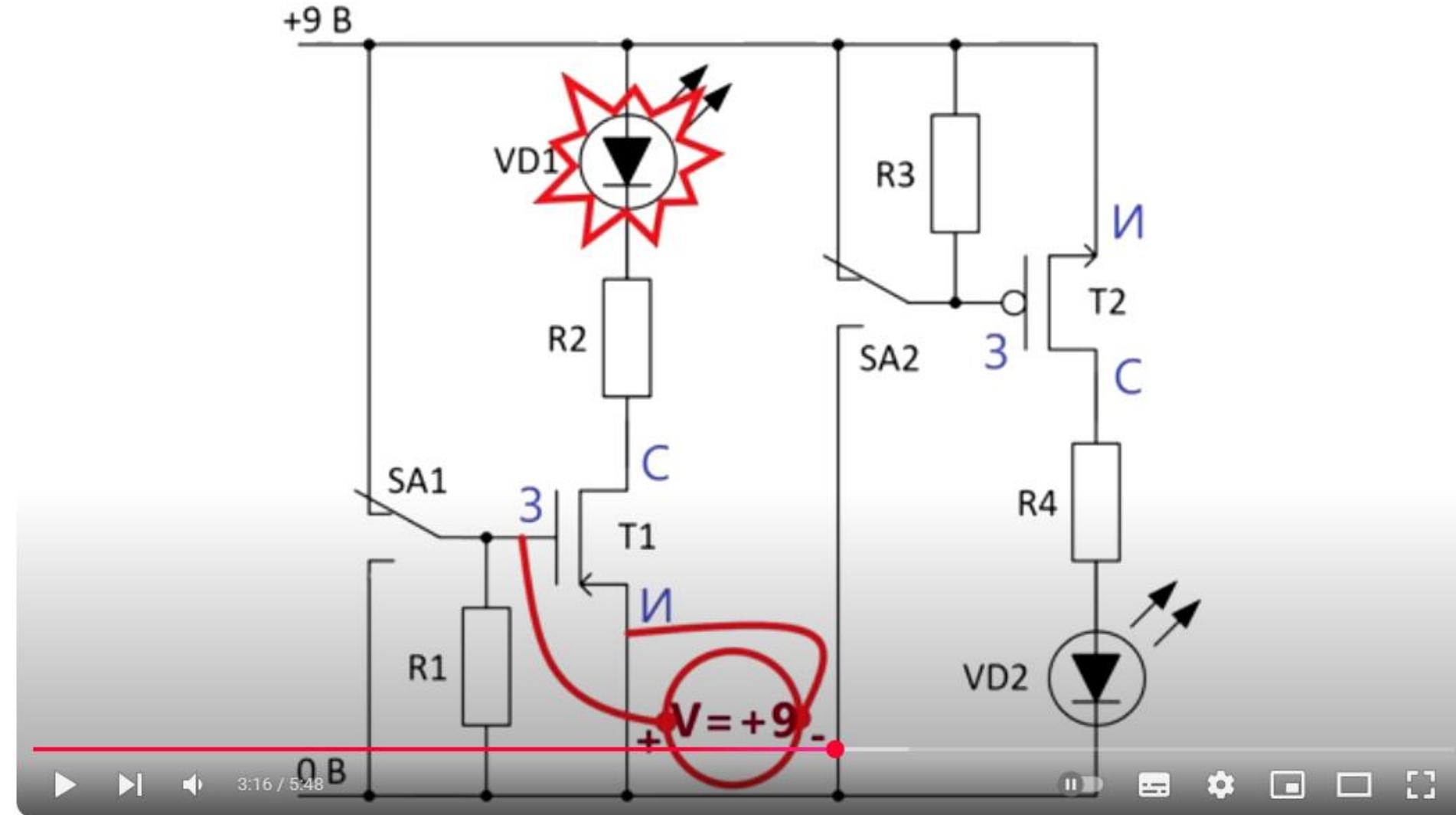
<https://www.youtube.com/watch?v=-lwpk--mJHM>



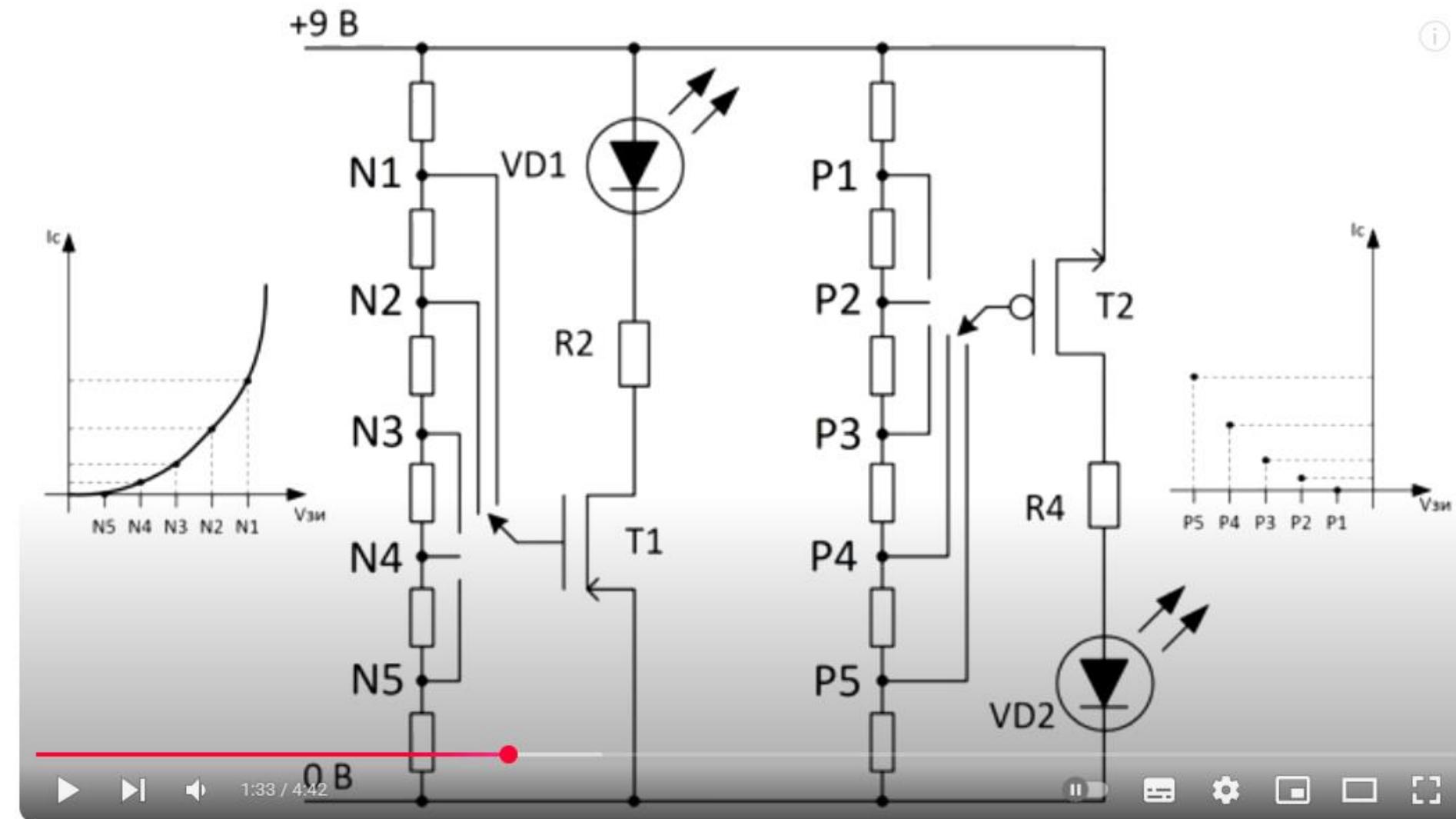
3. Что такое рп-переход и зачем он нужен (2021)  
[https://www.youtube.com/watch?v=r\\_1A-oPRNPw](https://www.youtube.com/watch?v=r_1A-oPRNPw)



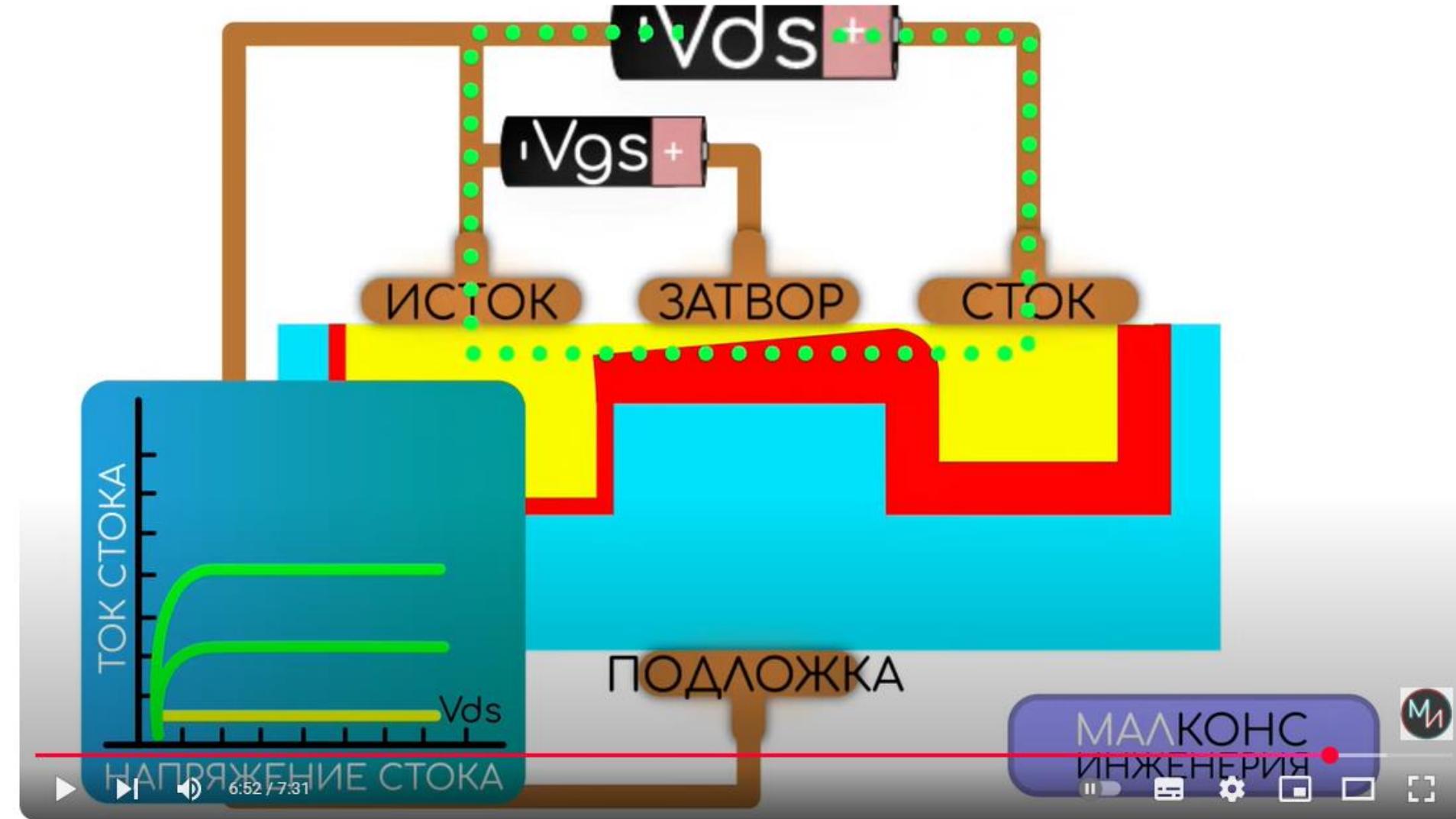
2. Зачем нужны примеси в кремнии (2020)  
<https://www.youtube.com/watch?v=jDurmeWpEd4>



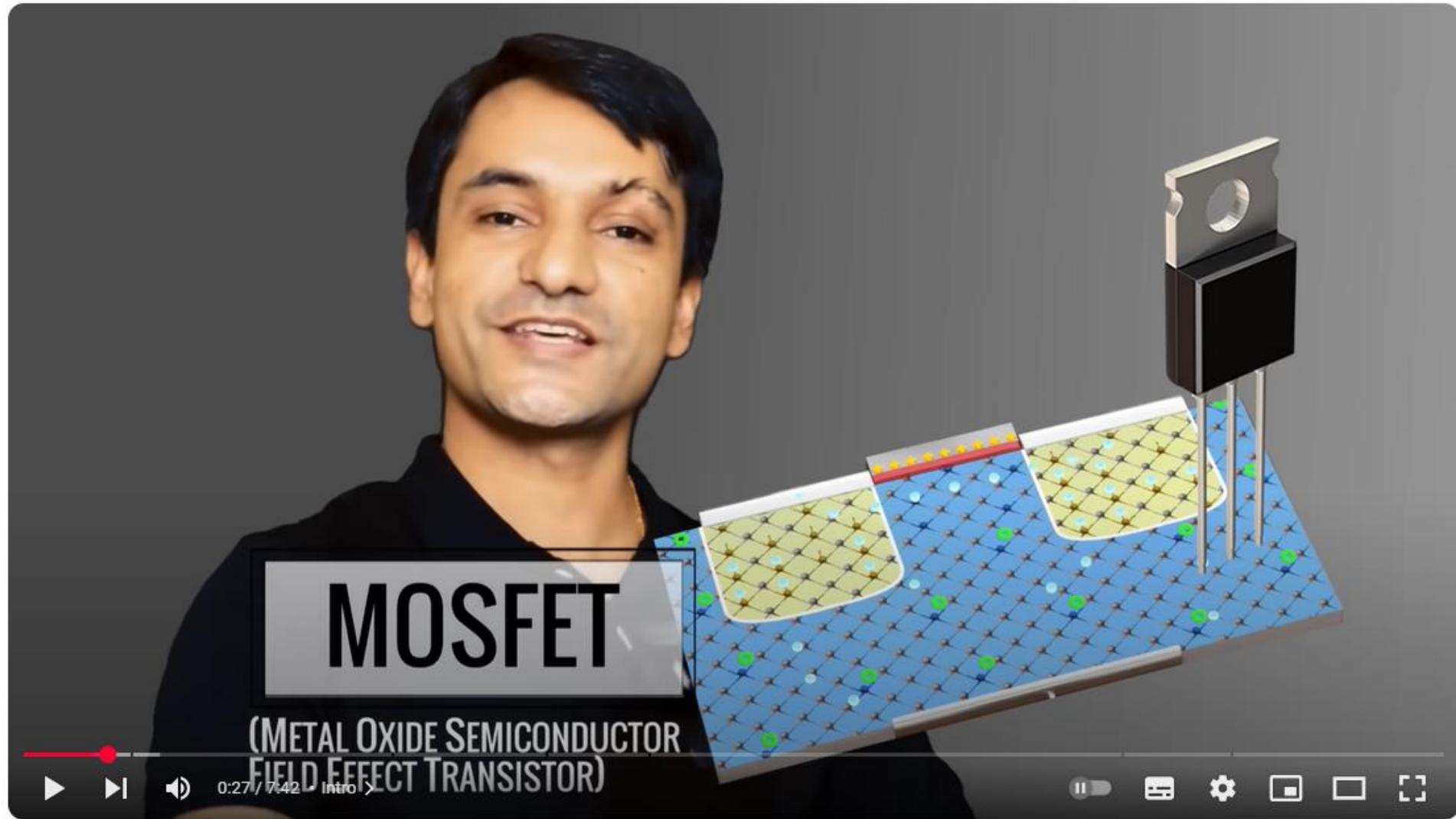
Принцип работы МОП-транзистора на примере простейшей схемы (часть 1) (2022)  
[https://www.youtube.com/watch?v=24eURhl\\_WbM](https://www.youtube.com/watch?v=24eURhl_WbM)



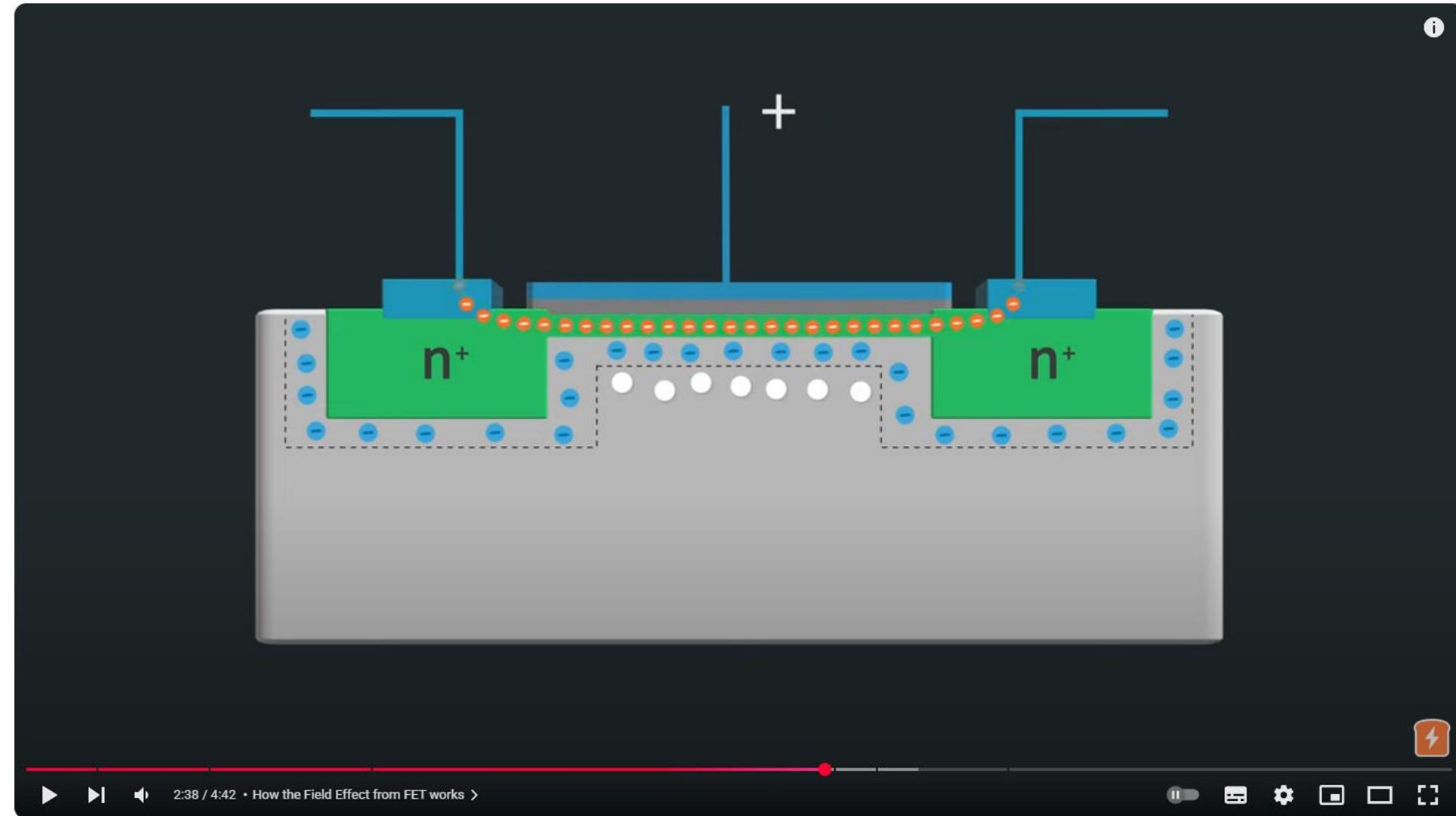
Принцип работы МОП-транзистора на примере простейшей схемы (часть 2) (2022)  
<https://www.youtube.com/watch?v=hvooC4EIAT8>



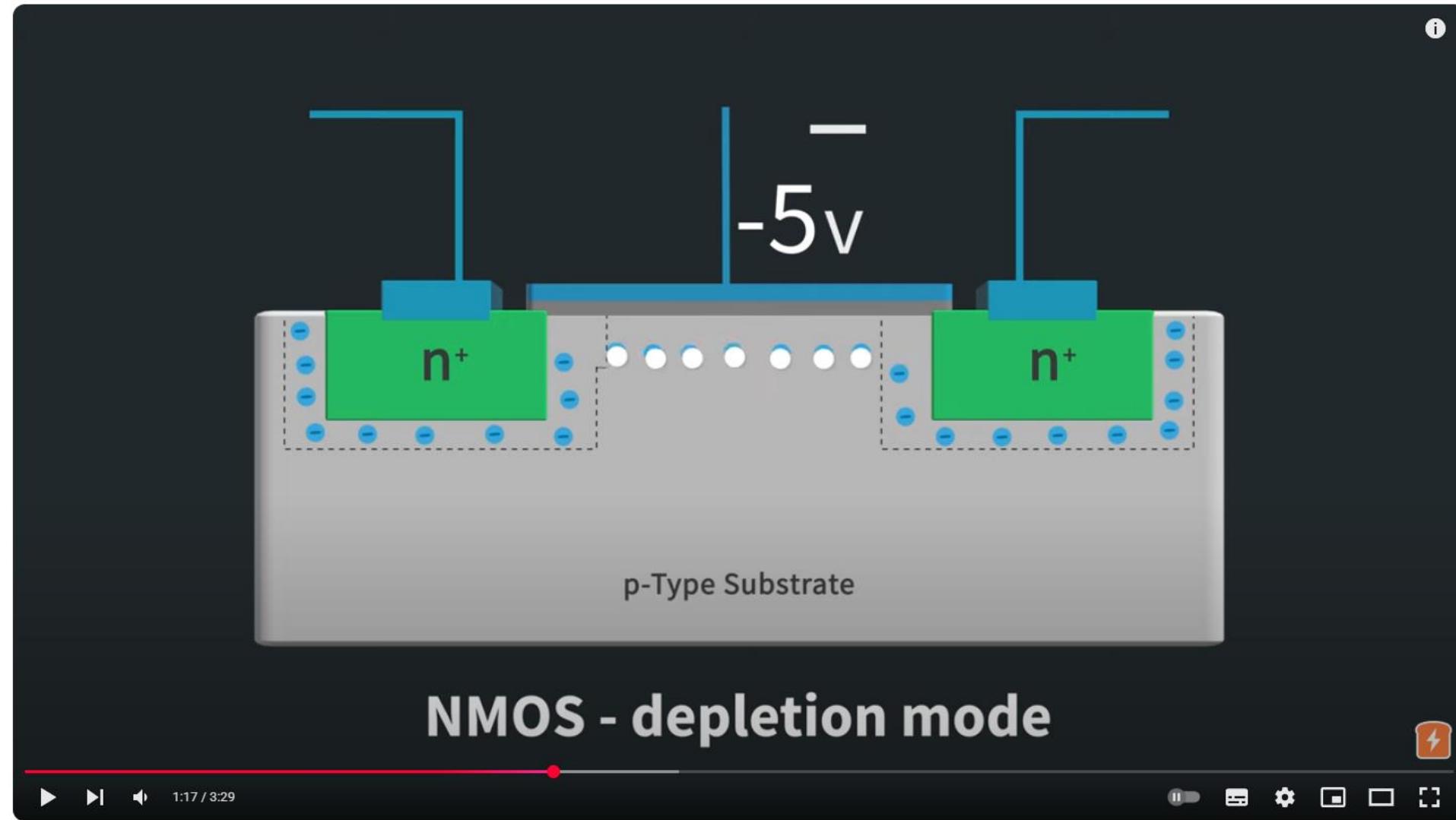
МОП MOSFET ТРАНЗИСТОР. ПРИНЦИП РАБОТЫ В АНИМАЦИИ. БЕЗ ЛИШНЕЙ ВОДЫ И ФОРМУЛ (2021) <https://www.youtube.com/watch?v=ZUly6WzORHc>



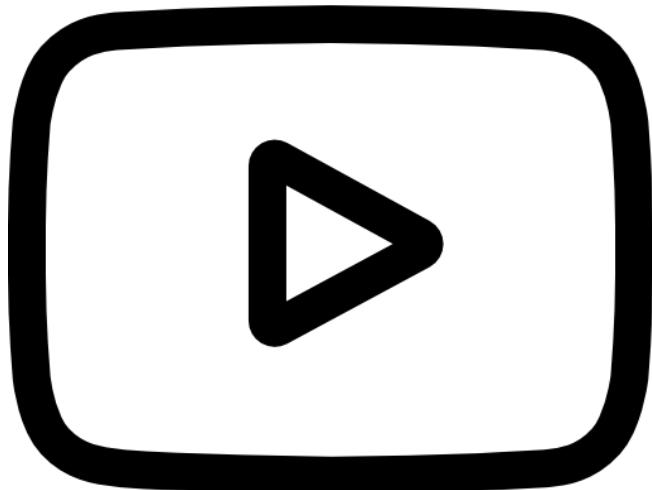
Working of Transistors | MOSFET (2018)  
Работа транзисторов | MOSFET (2018)  
<https://www.youtube.com/watch?v=stM8dgcY1CA>



How a MOSFET Works - with animation! | Intermediate Electronics (2020)  
[https://www.youtube.com/watch?v=Bfvyj88Hs\\_o](https://www.youtube.com/watch?v=Bfvyj88Hs_o)

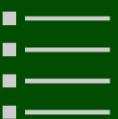


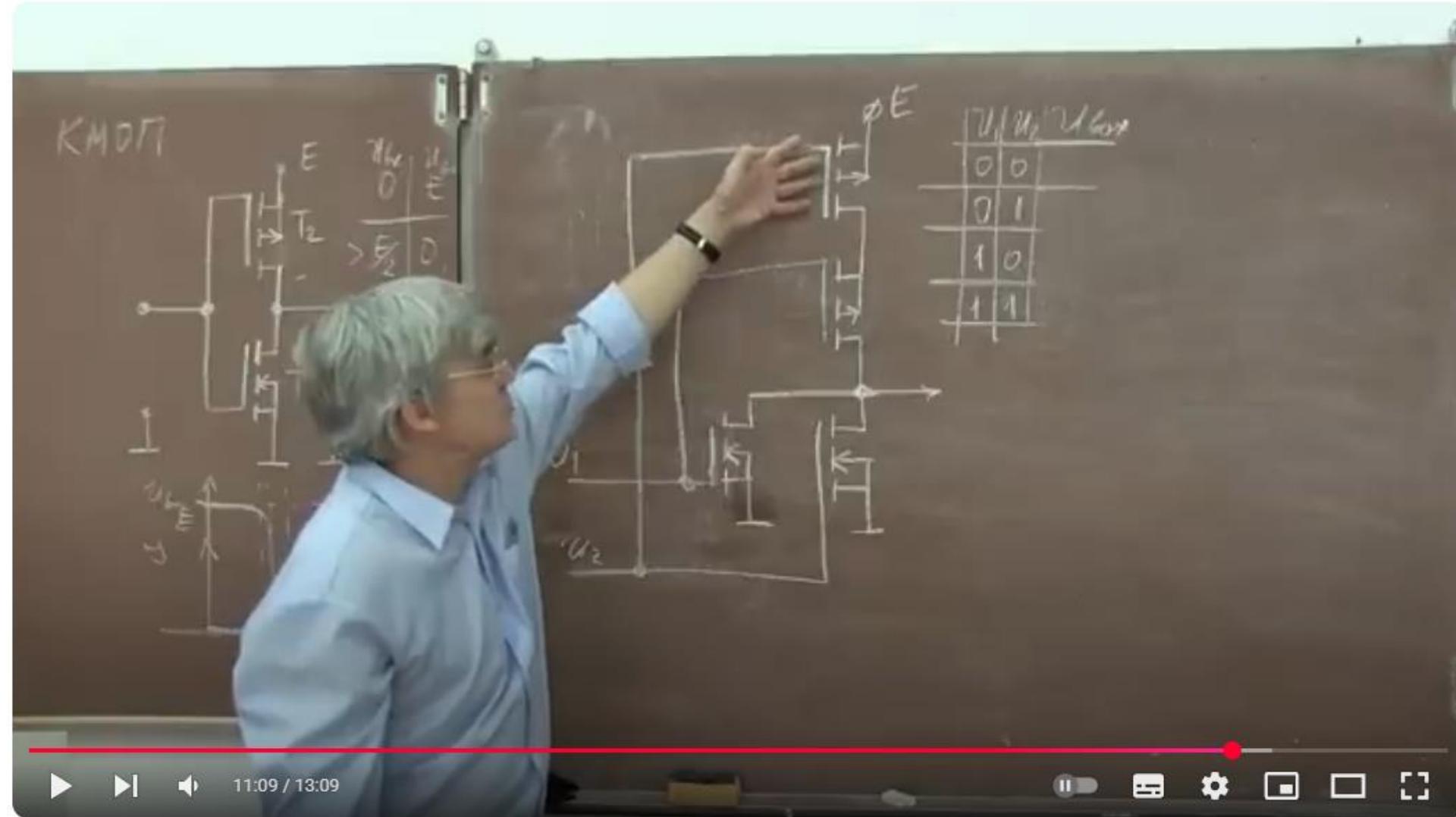
NMOS vs PMOS and Enhancement vs Depletion Mode MOSFETs | Intermediate Electronics (2020)  
<https://www.youtube.com/watch?v=kY-kaOPriaE>



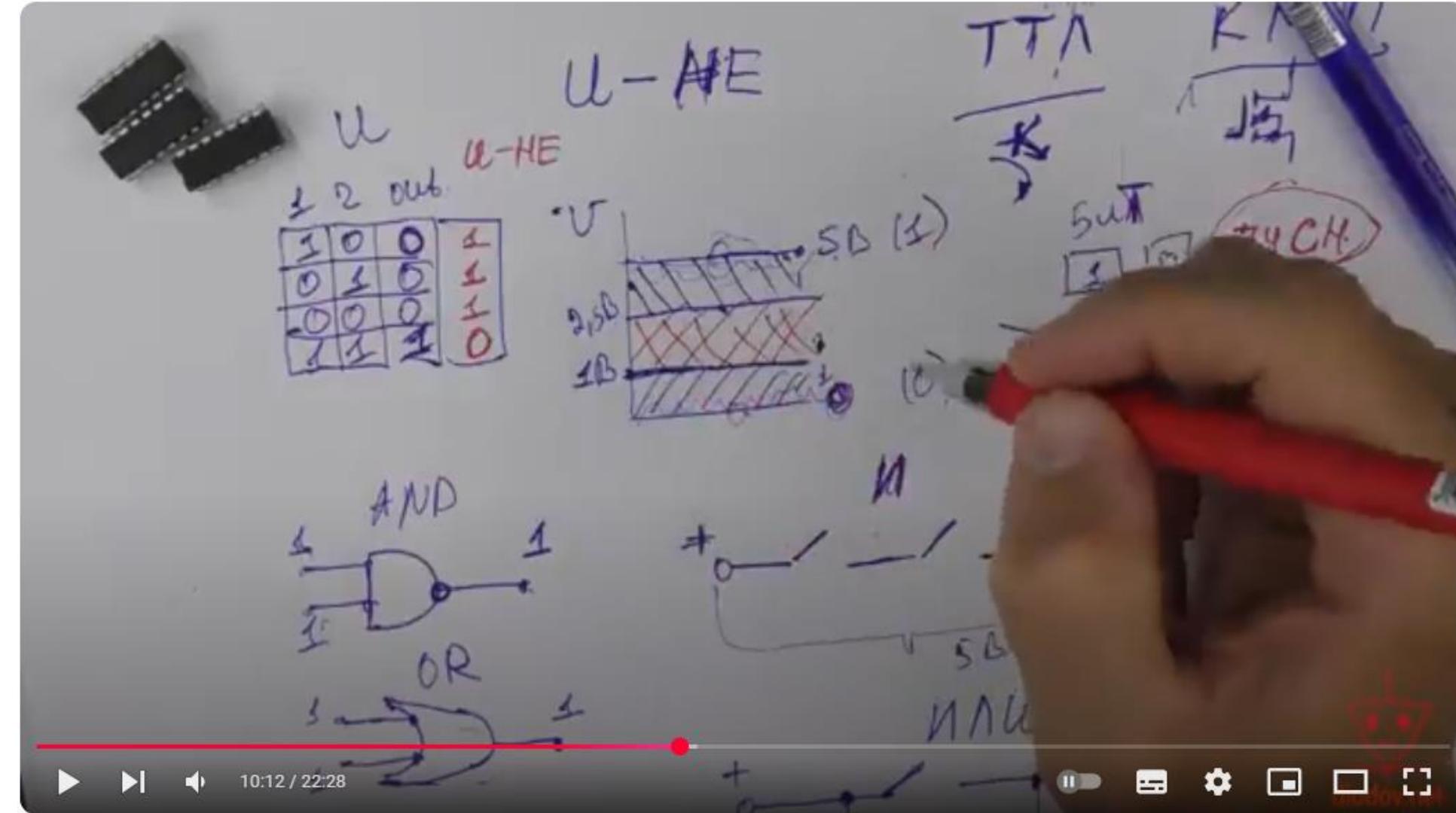
**Комплементарная логика  
металл-оксид-  
полупроводник (КМОП)**

**Complementary Metal-  
Oxide-Semiconductor Logic  
(CMOS)**

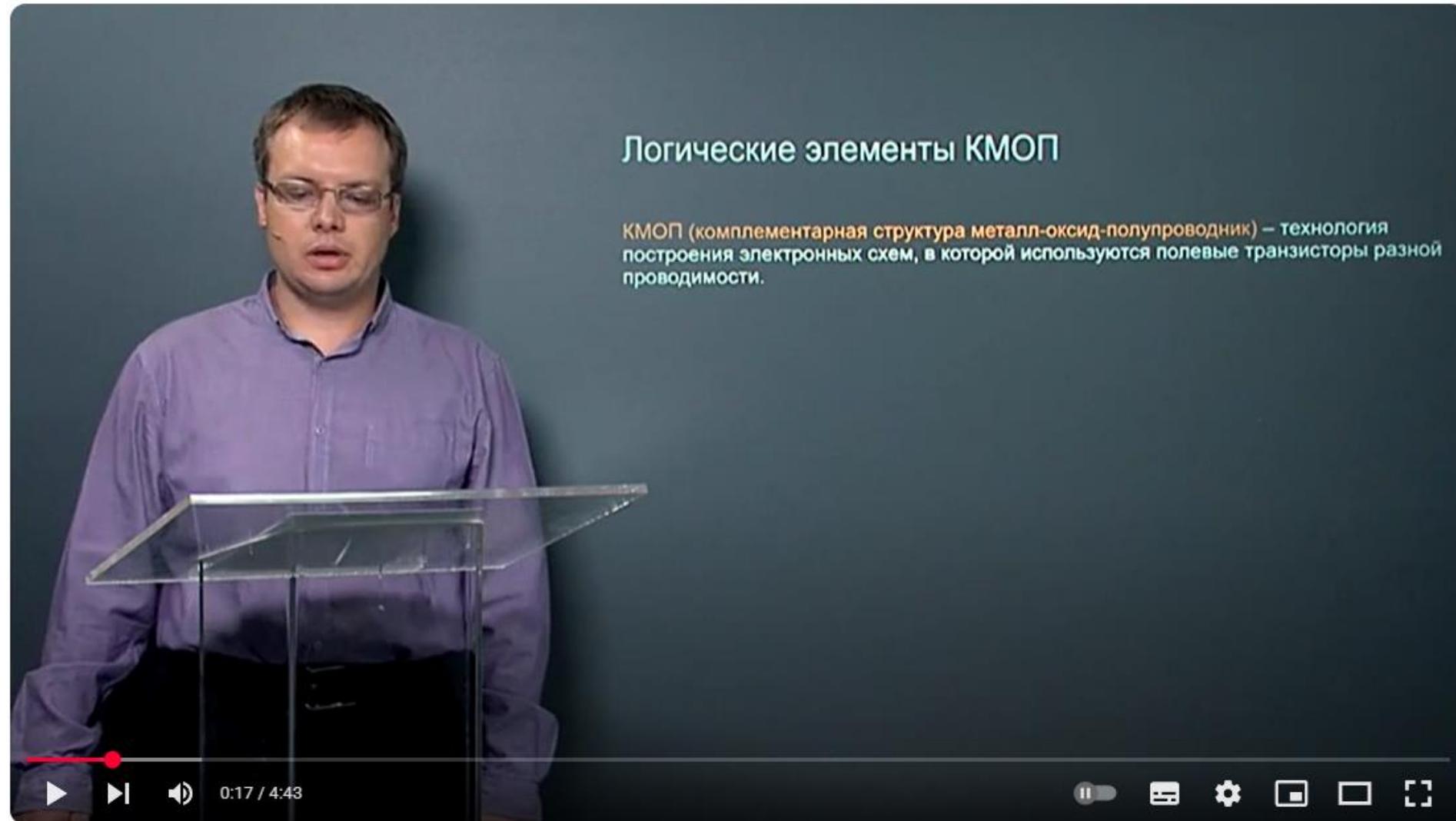




Лекция 138. КМОП-логика (2014)  
<https://www.youtube.com/watch?v=9eIkCWrpIqY>

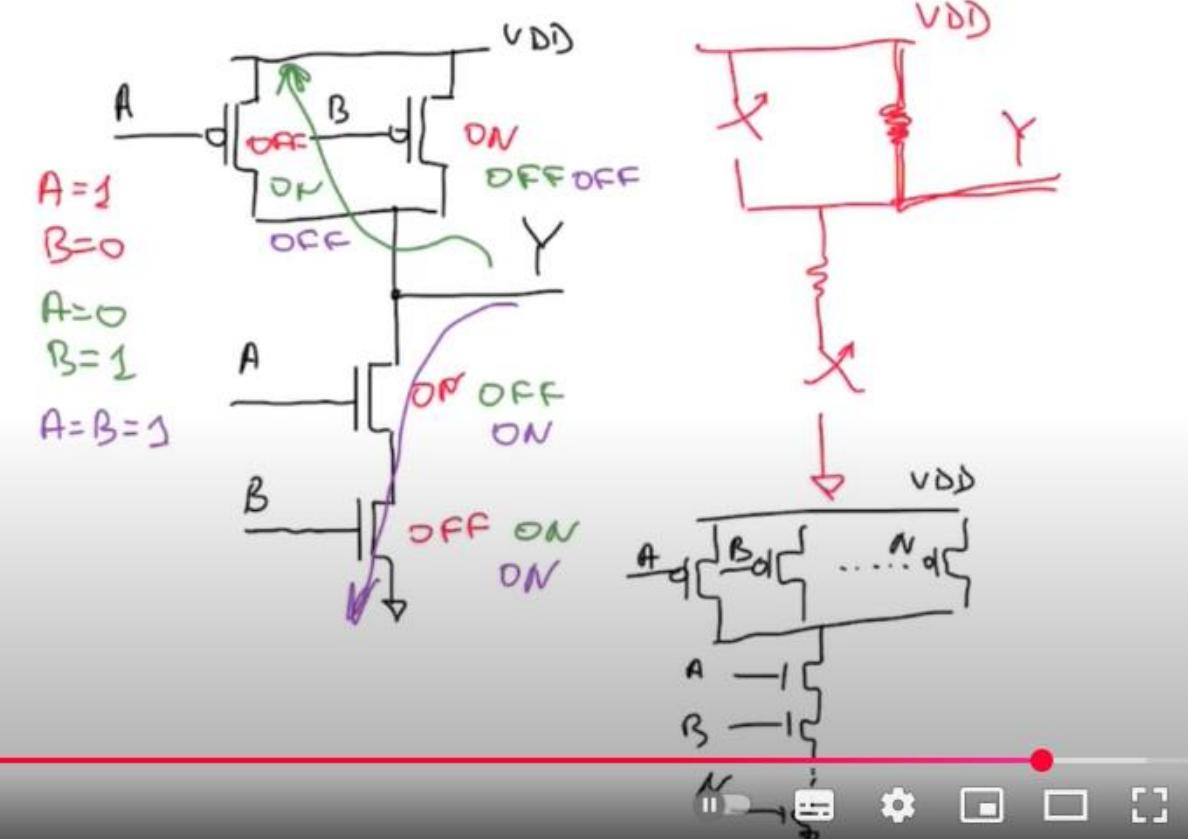
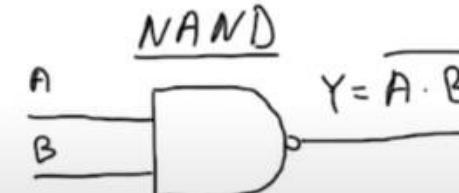
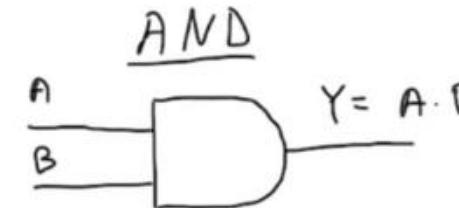


Что такое КМОП микросхемы. CMOS. На реальных примерах (2021)  
<https://www.youtube.com/watch?v=iYyp13bRuFk>



4 2 3 Логические элементы КМОП (2017)  
<https://www.youtube.com/watch?v=banmRPMATEw>

## Логические И и И-НЕ (AND NAND )

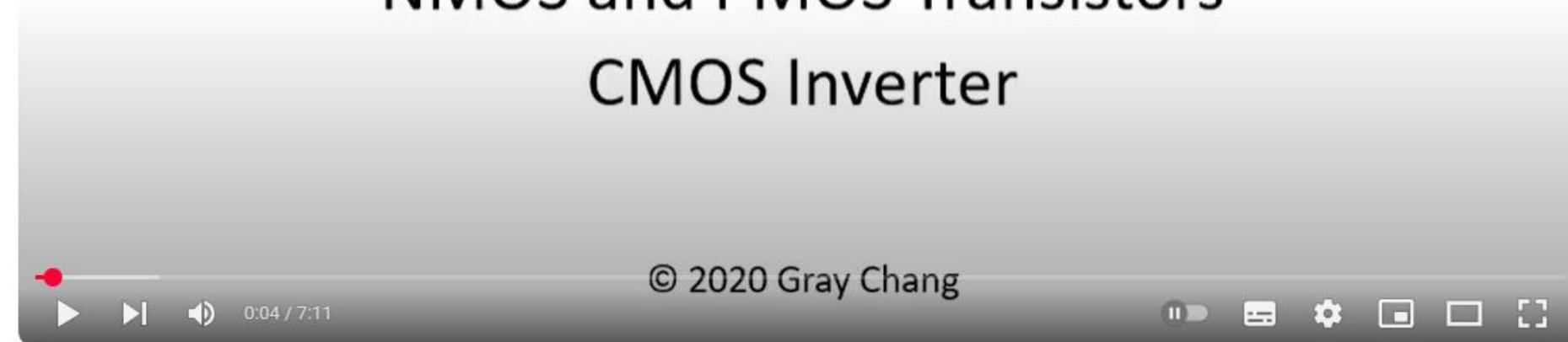


AND и NAND из КМОП транзисторов(2019)  
[https://www.youtube.com/watch?v=Yw5To\\_b7Yzo](https://www.youtube.com/watch?v=Yw5To_b7Yzo)

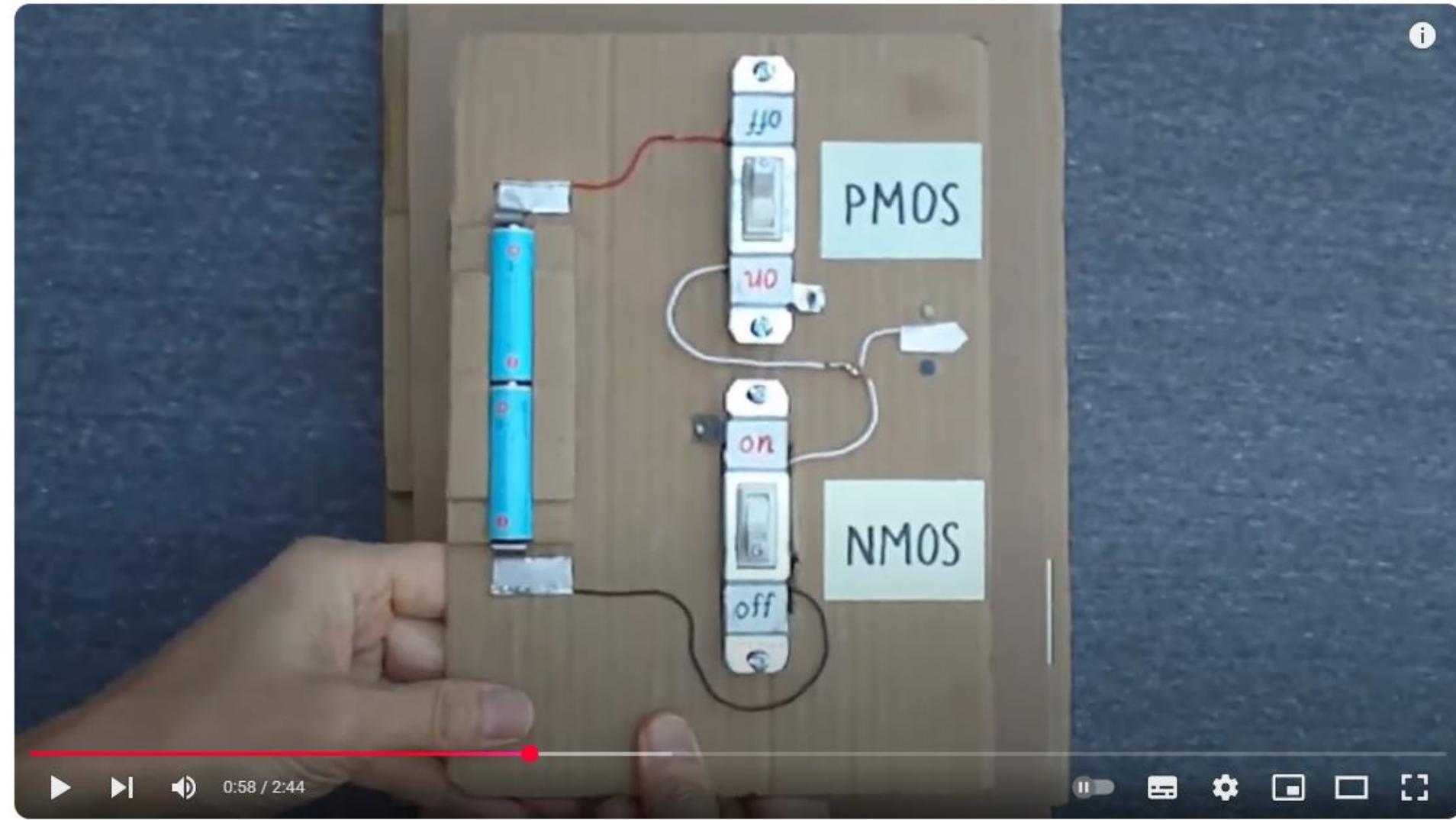


# CMOS Technology

NMOS and PMOS Transistors  
CMOS Inverter



CMOS Tech: NMOS and PMOS Transistors in CMOS Inverter (3-D View) (2020)  
Технология CMOS: NMOS и PMOS-транзисторы в CMOS-инверторе (трехмерный вид) (2020)  
<https://www.youtube.com/watch?v=oSrUsM0hoPs>



CMOS Inverter Model Using Light Switches (2020)  
Модель инвертора CMOS с использованием выключателей (2020)  
<https://www.youtube.com/watch?v=b4SmRKrapq4>



# ЭВМ, периферийные устройства и контроллеры

Тема: Уровни абстракции электронно-  
вычислительных систем

**Благодарю  
за внимание**

**КУТУЗОВ** Виктор Владимирович

# Список использованных источников

1. Рабочая программа дисциплины «ЭВМ, периферийные устройства и контроллеры» для студентов направлений подготовки 09.03.01 «Информатика и вычислительная техника» и 09.03.04 «Программная инженерия» / Кутузов В. В. – Могилев : Белорусско-Российский университет, 2025
2. Фотографии и картинки взяты с сайтов Яндекс.Картинки, Гугл.Картинки, Pinterest, иконки с flaticon.com
3. Отдельная информация генерировалась при помощи больших языковых моделей (LLM, Large Language Model). Прорабатывались идеи и структура отдельных разделов, уточнялась и перепроверялась отдельная информация, выполнялся поиск. Использовались: DeepSeek-R1, Qwen3-235B-A22B-2507, Qwen3-235B-A22B, perplexity.ai, Gemini-2.5-Flash-Lite-Preview, Gemini-2.0-Flash, Gemini 2.0. Вся сгенерированная информация многократно перепроверялась и дополнялась с интернет ресурсов.
4. Сервис перевода текстов DeepL Translate  
<https://www.deepl.com/>
5. Сервис автоматического перевода текста от Яндекса встроенного в Яндекс.Браузер  
<https://browser.yandex.ru>
6. Сара Л. Харрис, Дэвид Харрис Цифровая схемотехника и архитектура компьютера: RISC-V / пер. с англ. В. С. Яценкова, А. Ю. Романова; под ред. А. Ю. Романова. – М.: ДМК Пресс, 2021. – 810 с.: ил. ISBN 978-5-97060-961-3 <https://rutracker.org/forum/viewtopic.php?t=6204850>
7. Белоус А. И., Красников Г. Я., Солодуха В. А. Основы проектирования субмикронных микросхем. Москва: ТЕХНОСФЕРА, 2020. – 782 с. <https://djvu.online/file/mXzkarl4cai0s>
8. Электротехника и электроника. Наглядные пособия. Таблицы. Схемы – Челябинск: ЮУрГУ, 2011. – 106 с.  
<https://www.twirpx.com/file/1472423/>  
<https://djvu.online/file/uWScR1y9AmZXC>

# Список использованных источников

9. Кто производит чипы? Обзор и перспективы.  
<https://borodulin.expert/blog/prochee/kto-proizvodit-chipy-obzor-i-perspektivy/>
10. Примеры абстракций в технике и повседневности  
<https://habr.com/ru/articles/715146/>
11. Ты не видишь всей картины!  
<https://habr.com/ru/articles/887692/>
12. Абстракция  
<https://glossary.cncf.io/ru/abstraction/>
13. Примеры абстракций в технике и повседневности  
<https://habr.com/ru/articles/715146/>
14. Введение в анализ производительности и оптимизацию программ. Раздел I. Архитектурные механизмы, влияющие на производительность. Уровни параллелизма.  
[https://hpc-education.unn.ru/files/courses/optimization/1\\_2\\_Architecture.pdf](https://hpc-education.unn.ru/files/courses/optimization/1_2_Architecture.pdf)
15. Электроника. Цифровые устройства  
[https://portal.tpu.ru/SHARED/g/GREBENNIKOVVV/students/Tab4/Tab/05\\_Электроника\\_22\\_ЛЕКЦИИ\\_ЛЭ\\_2022.pdf](https://portal.tpu.ru/SHARED/g/GREBENNIKOVVV/students/Tab4/Tab/05_Электроника_22_ЛЕКЦИИ_ЛЭ_2022.pdf)
16. Острейковский, В. А. Информатика. Теория и практика: Учеб, пособие / В.А. Острейковский, И.В. Полякова. — М.: Издательство Оникс, 2008. — 608 с.
17. Дэвид М. Харрис, Сара Л. Харрис Цифровая схемотехника и архитектура компьютера. / пер. с англ. Imagination Technologies. – М.: ДМК Пресс, 2018. – 792 с.

# Список использованных источников

18. Федотова, Е. Л. Информатика : учебное пособие / Е.Л. Федотова. — 2-е изд., перераб. и доп. — Москва : ИНФРА-М, 2022. — 453 с. — (Высшее образование: Бакалавриат). — DOI 10.12737/1200564. - ISBN 978-5-16-016625-4. - Текст : электронный. - URL:  
<https://znanium.com/catalog/product/1200564>
19. CSC111: Introduction to CS through Programming. Lecture 2: HOW COMPUTERS WORK. (R. Jordan Crouser, Assistant Professor of Computer Science, Smith College )  
<https://jcrouser.github.io/CSC111/lectures/02-how-computers-work.pdf>
20. Краткое объяснение кодирования текстовой информации. Информатика  
<https://bingoschool.ru/manual/kratkoe-obyasnenie-kodirovaniya-tekstovoj-informaczii.-informatika/>
21. Информатика, 10 класс. Урок № 14. Кодирование текстовой информации  
<https://resh.edu.ru/subject/lesson/5225/conspect/203083/>
22. Схема основной многоязычной плоскости Юникода  
[https://ru.wikipedia.org/wiki/%D8%A5%D8%BD%D9%85%D9%83%D9%84%D9%84%D9%82%D9%84%D9%8A%D9%84#media/Файл:Roadmap\\_to\\_Unicode\\_BMP\\_multilingual.svg](https://ru.wikipedia.org/wiki/%D8%A5%D8%BD%D9%85%D9%83%D9%84%D9%84%D9%82%D9%84%D9%8A%D9%84#media/Файл:Roadmap_to_Unicode_BMP_multilingual.svg)
23. Плоскость (Юникод)  
[https://ru.wikipedia.org/wiki/%D0%9F%D0%BB%D0%BE%D0%BF%D0%B5%D0%BA%D1%81%D0%BD%D0%BE%D0%B9\\_%D8%A5%D8%BD%D9%85%D9%84%D9%84](https://ru.wikipedia.org/wiki/%D0%9F%D0%BB%D0%BE%D0%BF%D0%B5%D0%BA%D1%81%D0%BD%D0%BE%D0%B9_%D8%A5%D8%BD%D9%85%D9%84%D9%84)
24. Кириллическая заглавная буква ка. Кириллица.  
<https://unicode-table.com/ru/041A/>
25. Смайлики-эмоджи «Лица»  
<https://unicode-table.com/ru/sets/faces/>

# Список использованных источников

26. Петрунина Е.Б. Лекции по информатике: Учеб.-метод. пособие. – СПб.: НИУ ИТМО; ИХиБТ, 2014. – 105 с  
<https://books.ifmo.ru/file/pdf/1599.pdf>
27. Аналого-цифровые и цифро-аналоговые преобразователи  
<https://slide-share.ru/nalogo-cifrovie-ifro-analogovie-preobrazovateli-316355>
28. Закляков В. Ф. Информатика: учеб. для вузов – 5-е изд., перераб. и доп. – М.: ДМК Пресс, 2021. – 750 с.
29. Звуковые карты  
<https://ppt-online.org/287222>
30. 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>
31. Особенности Linux \ Работа с файлами \ Права доступа  
[https://linuxcookbook.ru/books/informatika1/2\\_os/1\\_linux/03\\_file/9\\_chmod/index.html](https://linuxcookbook.ru/books/informatika1/2_os/1_linux/03_file/9_chmod/index.html)
32. Определение прав доступа к файлам и папкам  
[https://okmysite.com/php/php\\_uchebnik/prava\\_dostupa\\_k\\_fajlam\\_i\\_papkam\\_v\\_php.html](https://okmysite.com/php/php_uchebnik/prava_dostupa_k_fajlam_i_papkam_v_php.html)
33. Системы счисления  
<https://guides.hexlet.io/ru/numeral-systems/>
34. Шестнадцатеричная система счисления  
<https://sistemy-schisleniya.ru/pozitsionnye/shestnadtsaterichnaya>
35. Революционный микропроцессор 8008: взгляд 50 лет спустя  
[https://www.nix.ru/computer\\_hardware\\_news/hardware\\_news\\_viewer.html?id=213371](https://www.nix.ru/computer_hardware_news/hardware_news_viewer.html?id=213371)

# Список использованных источников

36. Intel 8008  
[https://ru.wikipedia.org/wiki/Intel\\_8008](https://ru.wikipedia.org/wiki/Intel_8008)
37. [Перевод] Инженерный анализ схемы ускоренного переноса процессора Intel 8008  
<https://news.myseldon.com/ru/news/index/240748940>
38. Reverse-engineering the surprisingly advanced ALU of the 8008 microprocessor (Реверс-инжиниринг удивительно продвинутого арифметико-логического устройства микропроцессора 8008)  
<https://www.righto.com/2017/02/reverse-engineering-surprisingly.html>
39. Die photos and analysis of the revolutionary 8008 microprocessor, 45 years old (Фотографии и анализ революционного микропроцессора 8008, которому исполнилось 45 лет)  
[https://www.righto.com/2016/12/die-photos-and-analysis-of\\_24.html](https://www.righto.com/2016/12/die-photos-and-analysis-of_24.html)
40. Intel 8008 / MCS-8 Users Manual (Intel 8008 8-bit Parallel Central Processor Unit and MCS-8 Micro Computer Set Users Manual. Includes the 8008-1, SIM8-01, MP7-08 and the Bootstrap Loader Rev. 2, Nov. 1972.)  
<https://archive.org/details/intel-8008-mcs-8-users-manual/mode/2up>
41. И. А. Насыров 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](https://kpfu.ru/staff_files/F16500157/DigitElectrLabsPart1.pdf)
42. Элементы теории множеств и алгебры логики  
<https://files.lbz.ru/authors/informatika/3/bosova-10-gl4.pdf>

# Список использованных источников

43. 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>
44. И. А. Насыров 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](https://kpfu.ru/staff_files/F16500157/DigitElectrLabsPart1.pdf)
45. Яшин, В. Н. Информатика : учебник / В.Н. Яшин, А.Е. Колоденкова. — Москва: ИНФРА-М, 2021. — 522 с. — (Высшее образование: Бакалавриат). — DOI 10.12737/1069776. - ISBN 978-5-16-015924-9. - Текст : электронный. - URL: <https://znanium.com/catalog/product/1069776>
46. 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. : ил.
47. Физика двоичной логики  
<https://habr.com/ru/companies/timeweb/articles/653159/>
48. Логическая связка  
[https://ru.wiki34.com/wiki/Conectiva\\_lógica](https://ru.wiki34.com/wiki/Conectiva_lógica)
49. Logical connective (Логическая связка)  
[https://en.wikipedia.org/wiki/Logical\\_connective](https://en.wikipedia.org/wiki/Logical_connective)
50. Лекция 4. Основы логических элементов  
[https://farabi.university/storage/files/321504225667596ff77e4dd311126007\\_Лекция%204%20-%20Основы%20логических%20элементов%20+.pdf](https://farabi.university/storage/files/321504225667596ff77e4dd311126007_Лекция%204%20-%20Основы%20логических%20элементов%20+.pdf)

# Список использованных источников

51. Цифровой синтез: практический курс / под общ. ред. А. Ю. Романова, Ю. В. Панчула. – М.: ДМК Пресс, 2020. – 556 с. ISBN 978-5-97060-850-0  
<https://rutracker.org/forum/viewtopic.php?t=6258743>
52. ГОСТ Р МЭК 60617-DB-12M-2015  
<http://www.omegametall.ru/Data2/1/4293763/4293763109.pdf>
53. Построение логических элементов на транзисторах  
<https://habr.com/ru/articles/811401/>
54. Лекция 4. Основы логических элементов  
<https://www.rlocman.ru/review/article.html?di=619699>
55. Логический вентиль  
[https://ru.wikipedia.org/wiki/Логический\\_вентиль](https://ru.wikipedia.org/wiki/Логический_вентиль)
56. Практическая работа №1 Логические элементы и схемы  
<https://portal.tpu.ru/SHARED/s/SHIDLOVSKIY/ur/Tab4/Практическая%20работа%20Nº1.pdf>
57. Основы логики и логические основы компьютера  
<https://ppt-online.org/73297>
58. Мега Компьютер Состоящий Из Миллионов Живых Людей. Сериял: Задача Трех Тел/ 3 Body Problem // 3 серия <https://www.youtube.com/watch?v=OluvvYqRWIY>
59. Функциональная схемотехника/ Кустарев П.В. // Кафедра ВТ 2010-18, Университет ИТМО  
[https://picloud.pw/media/resources/posts/2018/03/15/lec\\_schemo\\_slide.pdf](https://picloud.pw/media/resources/posts/2018/03/15/lec_schemo_slide.pdf)

# Список использованных источников

60. Логические уровни или где заканчивается ноль и начинается единица.  
<https://dzen.ru/a/YA9xU43-ezstKTys>
61. OpenWrt » Hardware » Serial Console  
<https://web.archive.org/web/20171215115024/http://wiki.openwrt.org/doc/hardware/port.serial>
62. Logic Threshold Voltage Levels  
[https://web.archive.org/web/20171225154900/http://www.interfacebus.com/voltage\\_threshold.html](https://web.archive.org/web/20171225154900/http://www.interfacebus.com/voltage_threshold.html)
63. Translatory napięć poziomów logicznych  
<https://elektronikab2b.pl/technika/13733-translatory-napiec-poziomow-logicznych?start=0&showall=1>
64. Транзистор — что это и с чем его едят?!  
<https://www.drive2.ru/b/1135837/>
65. Logic family  
[https://en.wikipedia.org/wiki/Logic\\_family](https://en.wikipedia.org/wiki/Logic_family)
66. Твердотельное реле против электромеханики: что выбрать?  
<https://dzen.ru/a/W5qAHwLiSwCqad95>
67. Резисторно-транзисторная логика  
[https://en.wikipedia.org/wiki/Resistor-transistor\\_logic](https://en.wikipedia.org/wiki/Resistor-transistor_logic)
68. Биполярные транзисторы  
<https://fresh-web-studio.github.io/artemsdobnikov/math/bipolar-transistors.html>
69. Биполярный транзистор  
[https://ru.wikipedia.org/wiki/Биполярный\\_транзистор](https://ru.wikipedia.org/wiki/Биполярный_транзистор)

# Список использованных источников

70. Принцип работы транзисторов  
<https://radioskot.ru/publ/teoria/princzip-raboty-tranzistorov>
71. Микросхемы цифровые  
[https://l7805cv.ru/digital\\_ic-dtl.html](https://l7805cv.ru/digital_ic-dtl.html)
72. Diode-transistor logic  
[https://en.wikipedia.org/wiki/Diode-transistor\\_logic](https://en.wikipedia.org/wiki/Diode-transistor_logic)
73. Физика и схемотехника интегральных схем. Тема 4. Схемотехника цифровых интегральных схем (часть 1)  
<https://edu2u.ru/sandbox/data/FSIC/L4.pdf>
74. Дисциплина «Микроэлектроника» ТЕМА: «Основные схемотехнические структуры цифровой интегральной микроэлектроники»  
<https://edu.study.tusur.ru/publications/5585/download>
75. Эмиттерно-связанная логика  
[https://ru.wikipedia.org/wiki/%D0%95%D0%BC%D0%B8%D1%82%D1%80%D0%B5%D0%BD%D0%BD%D0%BA%D0%B0%D0%BA](https://ru.wikipedia.org/wiki/%D0%95%D0%BC%D0%B8%D1%82%D1%80%D0%B5%D0%BD%D0%BD%D0%BA%D0%B0-%D1%81%D1%8F%D1%8A%D0%BD%D0%BA%D0%B0%D0%BA)
76. Emitter-coupled logic  
[https://en.wikipedia.org/wiki/Emitter-coupled\\_logic](https://en.wikipedia.org/wiki/Emitter-coupled_logic)
77. Эмиттерно-связанная логика  
<https://ru.ruwiki.ru/wiki/%D0%95%D0%BC%D0%B8%D1%82%D1%80%D0%B5%D0%BD%D0%BD%D0%BA%D0%B0%D0%BA>
78. Что такое схема металл-оксид-полупроводник (МОП)?  
<https://www.vrsystems.ru/chto-takoe-shema-metall-oksid-polyprovodnik-mop.htm>
79. МОП-транзистор <https://ru.wikipedia.org/wiki/%D0%9C%D0%9E%D0%A1-%D1%84%D1%80%D0%BE%D0%BD%D0%BD%D0%BA%D0%BE%D0%BC>

# Список использованных источников

80. Полевой транзистор МОП (MOSFET)  
<https://www.ruselectronic.com/polevoj-mop-tranzistor/?nowprocket=1>
81. Лекция 16: Реализация логических элементов  
<https://intuit.ru/studies/courses/685/541/lecture/12194%20Лекция%202016>
82. Схема включения моп транзисторов  
<https://logoslab.ru/library/shema-vkljuchenija-mop-tranzistorov.html>
83. Активные электронные компоненты MOSFET транзисторы  
<https://razumdom.ru/lectures/MOSFET.pdf>
84. Прикладная электроника для студентов  
[https://razumdom.ru/articles/prikladnaya\\_elektronika\\_dlya\\_studentov/](https://razumdom.ru/articles/prikladnaya_elektronika_dlya_studentov/)
85. Функциональная схемотехника. Кафедра ВТ, Университет ИТМО 2010-18 к.т.н., доц. Кустарев П. В.  
[https://picloud.pw/media/resources/posts/2018/03/15/lec\\_schemo\\_slide.pdf](https://picloud.pw/media/resources/posts/2018/03/15/lec_schemo_slide.pdf)
86. Introduction to CMOS VLSI Design. Technology Introduction. Peter Kogge, University of Notre Dame, Fall 2015, 2018 <https://www3.nd.edu/~kogge/courses/cse40462-VLSI-fa18/www/Public/Lectures/Intro.pdf>
87. CMOS-технология  
[https://ncontrol.ru/blog/azbuka\\_kontrolya/cmos](https://ncontrol.ru/blog/azbuka_kontrolya/cmos)
88. Введение в технологию CMOS  
<https://www.ariat-tech.ru/blog/an-introduction-to-cmos-technology.html>
89. Логические элементы КМОП-технологии  
<https://intuit.ru/studies/courses/685/541/lecture/12194?page=2>

# Список использованных источников

90. Introduction to Electronics. Phys104 By : Behailu Abebe. Overview. Review the definition of voltage, electric current, resistance and power. Introduction to various electronic components Introduction to digital electronics circuits design and operation. Mar 26, 2019  
[https://www.slideserve.com/tamal/introduction-to-electronics-powerpoint-ppt-presentation#google\\_vignette](https://www.slideserve.com/tamal/introduction-to-electronics-powerpoint-ppt-presentation#google_vignette)
91. Ланцов, В. Н. Проектирование заказных интегральных схем на КМОП: учеб. пособие / В. Н. Ланцов; Владимир. гос. ун-т. – Владимир: Изд-во Владимир. гос. ун-та, 2009. – 224 с. – ISBN 978-5-89368-941-9  
<https://dspace.www1.vlsu.ru/bitstream/123456789/1314/3/00806.pdf>
92. КМОП  
<https://ru.wikipedia.org/wiki/КМОП>
93. Weste Neil H.E., Harris David. CMOS VLSI design 4th edition. — Pearson Education, 2011. — 839 p.  
<https://www.twirpx.com/file/2540943/>
94. Researchers develop a roadmap for the development of information technology based on 2D materials  
[https://phys.org/news/2024-06-roadmap-technology-based-2d-materials.html?deviceType=mobile&utm\\_medium=organic&utm\\_source=yandexsmartcamera](https://phys.org/news/2024-06-roadmap-technology-based-2d-materials.html?deviceType=mobile&utm_medium=organic&utm_source=yandexsmartcamera)
95. Производство чипов размером 1 нм  
<https://radioskot.ru/publ/teoria/tehnologiya-proizvodstva-chipov-razmerom-1-nm>
96. Введение в проектирование на языке Verilog  
<https://docs.google.com/presentation/d/1TqlFGqrBzD166VZrZCWKF1KIriF4yPpPZ9JolRr6bsQ>

# Список использованных источников (YouTube)

## Уровни абстракции электронно-вычислительных систем

1. АПС Л1. Водная (ПМ, ИВТ, ПИН) (2021)  
<https://www.youtube.com/watch?v=JmiMaKVol0>

## Системы счисления и представление информации

1. Как на самом деле работает двоичный код? (2018)  
<https://www.youtube.com/watch?v=GL3kPXjSaWY>
2. Применение двоичной системы счисления в реальной жизни (2020)  
<https://www.youtube.com/watch?v=VawNm0MMxCE>
3. Двоичная система счисления - язык понятный компьютеру (2020)  
<https://www.youtube.com/watch?v=yOGgS-JWdV4>
4. Двоичная система счисления. Максимально просто и подробно (2019)  
[https://www.youtube.com/watch?v=Ro8jdy\\_kpko](https://www.youtube.com/watch?v=Ro8jdy_kpko)
5. Прямой Обратный Дополнительный (2020)  
[https://www.youtube.com/watch?v=Gn8P\\_TYa\\_AU](https://www.youtube.com/watch?v=Gn8P_TYa_AU)
6. Лекция 110. Арифметика отрицательных чисел в микропроцессорах (2013)  
<https://www.youtube.com/watch?v=Zd0mM5pgORY>
7. ИНФОРМАТИКА 8 класс: Двоичная система счисления. Двоичная арифметика (2018)  
<https://www.youtube.com/watch?v=NIPeCMjpri3Q>
8. Информатика 10 класс. Представление чисел в компьютере (УМК БОСОВА Л.Л., БОСОВА А.Ю.) (2020)  
<https://www.youtube.com/watch?v=kMvWakrKZJE>

# Список использованных источников (YouTube)

9. Сложение в микропроцессоре. Как работает двоичная система в процессоре (2012)  
[https://www.youtube.com/watch?v=wi\\_QNI8EZhA](https://www.youtube.com/watch?v=wi_QNI8EZhA)
10. Как компьютеры складывают числа | Хекслет (2015)  
<https://www.youtube.com/watch?v=YuSgZ173Utq>
11. Как написать Hello, World! на двоичном коде (не шестнадцатеричный) | GovnBin (2024)  
[https://www.youtube.com/watch?v=2-WuohF\\_vGo](https://www.youtube.com/watch?v=2-WuohF_vGo)
12. Двоичная система счисления. Урок 1 (2017)  
<https://www.youtube.com/watch?v=FGRIYjHfzSY>
13. Перевод из десятичной в двоичную систему счисления (2021)  
<https://www.youtube.com/watch?v=7KYxTj4UrhY>
14. Перевод из двоичной в десятичную систему счисления (2021)  
<https://www.youtube.com/watch?v=VVh8btCxlnk>
15. Перевод дробных чисел из 10 системы в 2, 8, 16 (2013)  
<https://www.youtube.com/watch?v=yLCZXD510Vo>
16. Перевод из 2, 8, 16 систем счисления в десятичную систему счисления (2013)  
<https://www.youtube.com/watch?v=G-ptY2HOcBk>
17. Перевод из двоичной в восьмеричную и шестнадцатеричную системы счисления целых и дробных чисел (2013) [https://www.youtube.com/watch?v=qSB\\_fkx3LYs](https://www.youtube.com/watch?v=qSB_fkx3LYs)
18. Системы счисления: Сложение, вычитание и умножение двоичных чисел (2014)  
<https://www.youtube.com/watch?v=DUFMI1-n9Nc>

# Список использованных источников (YouTube)

19. 59 Запись отрицательных чисел, или Дополнительный код (2018)  
<https://www.youtube.com/watch?v=Jk7rg7SJxyl>
20. 60 Примеры использования дополнительного кода (2018)  
[https://www.youtube.com/watch?v=uue5djB\\_CAI](https://www.youtube.com/watch?v=uue5djB_CAI)
21. 61 Дополнительный код: инвертируем все биты и прибавляем 1 (2018)  
<https://www.youtube.com/watch?v=laNxfYfF6eo>
22. Как устроен двоичный код (2023)  
<https://www.youtube.com/watch?v=Ft7Lvl9reoE>
23. Отрицательные двоичные числа: дополнительный и обратный код (2022)  
<https://www.youtube.com/watch?v=1CsmDrmmZcA>
24. Арифметические действия в двоичной системе счисления (2018)  
<https://www.youtube.com/watch?v=x92pfbuxhqY>
25. Двоичная арифметика. Сложение и умножение чисел в двоичной системе счисления. 8 класс Информатика (2020)  
<https://www.youtube.com/watch?v=4vtWVliX-3w>
26. Алгоритмы. Позиционная система счисления (2020)  
<https://www.youtube.com/watch?v=xhfzQjo3pHc>

# Список использованных источников (YouTube)

## Булева алгебра

1. Алгебра логики: Законы алгебры логики. (2014)  
<https://www.youtube.com/watch?v=7XA77xNVBv4>
2. Информатика. Алгебра логики: Операции алгебры логики.(2014)  
<https://www.youtube.com/watch?v=3yAEUKy68lw>
3. Информатика. Алгебра логики: Таблицы истинности (2014)  
<https://www.youtube.com/watch?v=qrij6Ekwqr-c>
4. Информатика. Архитектура ПК: Представление целых чисел в памяти ПК (2014)  
<https://www.youtube.com/watch?v=g6Y86fAqKEY>
5. Урок 21. Логические операции "И", "ИЛИ", "НЕ", "исключающее ИЛИ". ИКТ 10 класс по Полякову (2021)  
<https://www.youtube.com/watch?v=YLz1KOtZs8w>
6. 50 уроков Информатики: Алгебра логики (2021)  
[https://www.youtube.com/watch?v=\\_x\\_Vhwt384M](https://www.youtube.com/watch?v=_x_Vhwt384M)
7. Законы де Моргана | 13/50 урок Информатики | Школково (2022)  
[https://www.youtube.com/watch?v=tO6HBxAt\\_fw](https://www.youtube.com/watch?v=tO6HBxAt_fw)
8. Законы алгебры логики / Закон де Моргана + доказательство [Алгебра логики] #5 (2020)  
<https://www.youtube.com/watch?v=yUP59rLpFZw>
9. Лекция 67. Теорема де Моргана (2013)  
<https://www.youtube.com/watch?v=Sv64kxLGBFc>

# Список использованных источников (YouTube)

10. Лекция 80. Карта Карно (2013)  
<https://www.youtube.com/watch?v=a37anDvo0bs>
11. Карты Карно. Как они работают. Большой выпуск (2019)  
<https://www.youtube.com/watch?v=wIEiX9ROSoE>
12. Построение таблиц истинности (2018)  
<https://www.youtube.com/watch?v=R5iuMQFPml8>
13. Построение таблиц истинности. Информатика 8 класс(2020)  
<https://www.youtube.com/watch?v=IHF0b5Tw9Zw>
14. ИНФОРМАТИКА 8 класс: Построение таблиц истинности для логических выражений (2018)  
<https://www.youtube.com/watch?v=vWLGKwOU5TU>
15. Минимизация функций. Карты Карно. Цифровая техника (2016)  
[https://www.youtube.com/watch?v=Fyruie9\\_uDKk](https://www.youtube.com/watch?v=Fyruie9_uDKk)
16. Информатика 8 класс. Таблицы истинности (УМК БОСОВА Л.Л., БОСОВА А.Ю.) (2020)  
<https://www.youtube.com/watch?v=iynqE6QMhW>
17. Теоретический минимум по CS: логические вентили, булева алгебра, сумматоры. Читаем главу 1 «Основы» (2022) <https://www.youtube.com/watch?v=s7g7ZUCZ2q8>
18. Лекция №6 / Основы программирования / Булева алгебра и условия (2021)  
<https://www.youtube.com/watch?v=EDEIPk2Y5Zc>
19. Решение логических выражений | 30 примеров | Булева алгебра | Гайд (2024)  
[https://www.youtube.com/watch?v=\\_HqeGGpJyCw](https://www.youtube.com/watch?v=_HqeGGpJyCw)

# Список использованных источников (YouTube)

20. Двоичные функции (Булевы функции). Математическая логика с Вики! (2024)  
<https://www.youtube.com/watch?v=Xz36ONw1QCE>
21. Три способа упрощения логической функции (2014)  
<https://www.youtube.com/watch?v=5U4P56A2ePY>
22. Логика - Упрощение логических выражений. Законы алгебры логики(2018)  
<https://www.youtube.com/watch?v=sNI5dB8I1qc>

# Список использованных источников (YouTube)

## Цифровая логика. Базовые логические элементы

1. Логические элементы (2017) <https://www.youtube.com/watch?v=3JCR15YHbJU>
2. Логические элементы И, ИЛИ, Исключающее ИЛИ. История, Теория, Применение (2021) <https://www.youtube.com/watch?v=bXdiYU3IUJA>
3. [Электроника] Логические Элементы, И, ИЛИ, НЕ, подробный обзор, и тестирование на микросхемах! (2018) <https://www.youtube.com/watch?v=rva16jfbdWE>
4. Должен знать каждый программист - логические операции И, ИЛИ, НЕ. Уроки Arduino для начинающих (2016) <https://www.youtube.com/watch?v=vSw5P4Ey9zc>
5. Лекция 71. Обозначение логических элементов (2013) <https://www.youtube.com/watch?v=s5MQoR-LN34>
6. Логические элементы (анализ электронных схем) (2021) <https://www.youtube.com/watch?v=4jER2JsjRhk>
7. Урок №27. Базовые логические элементы (2016) <https://www.youtube.com/watch?v=7TEGOGeplvl>
8. Введение в логические вентили (2020) <https://www.youtube.com/watch?v=n5Zo2wJS5XI>
9. Логический вентиль НЕ (инвертор) - как устроен и как собрать самостоятельно (2020) <https://www.youtube.com/watch?v=vg67x1javjo>

# Список использованных источников (YouTube)

10. Логический вентиль И - как работает и как собрать на макетной плате на основе транзисторов (2020) <https://www.youtube.com/watch?v=o8kAhNwiPjw>
11. Логический вентиль ИЛИ - объяснение работы и пример сборки на макетной плате на основе транзисторов (2020) <https://www.youtube.com/watch?v=qaQ0cLC66GE>
12. Логические вентили И-НЕ и ИЛИ-НЕ (2020) <https://www.youtube.com/watch?v=NKcGeBzOYbM>
13. Вентиль Исключающее ИЛИ (XOR) - объяснение принципа работы и пример на макетной плате (2020) <https://www.youtube.com/watch?v=Ik66e8jS9RA>
14. Цифровая техника - И, ИЛИ, НЕ на транзисторах (2015) <https://www.youtube.com/watch?v=CPp3Fp3Y2dw>
15. Зубаков А.П. Базовые логические элементы (2022) [https://www.youtube.com/watch?v=QcmDN\\_dFYXQ](https://www.youtube.com/watch?v=QcmDN_dFYXQ)
16. Базовые элементы цифровой электроники. Часть 1.(2020) <https://www.youtube.com/watch?v=Dc5XN3gu5Jo>
17. LOGIC GATE SIMULATOR: построение логических схем в основных базисах (2023) <https://www.youtube.com/watch?v=SbInkTLkmK4>
18. Digital Logic Gates from Transistors, AND, NAND, OR, NOR, XOR, XNOR, Buffer, and Inverter (2022) Цифровые логические элементы на основе транзисторов, AND, NAND, OR, NOR, XOR, XNOR, буфера и инвертора (2022) <https://www.youtube.com/watch?v=nB6724G3b3E>

# Список использованных источников (YouTube)

## Физический уровень работы простейших элементов компьютерных комплектующих

1. Где начало СХЕМЫ? Понимаем, читаем, изучаем схемы. Понятное объяснение! (2025)  
<https://www.youtube.com/watch?v=u6f6v4YTjN4>
2. Транзисторы и их применение (1987)  
[https://www.youtube.com/watch?v=y\\_vbpE-R-z8](https://www.youtube.com/watch?v=y_vbpE-R-z8)
3. ТАКОЕ НЕ ПОКАЖУТ В ВУЗах - Как работают и для чего нужны транзисторы ? Что такое PN переход? (2024) <https://www.youtube.com/watch?v=fHsqtFTnOzY>
4. Как работает транзистор [Veritasium] (2016)  
<https://www.youtube.com/watch?v=-z65gKETbf8>
5. Как работают схемы с транзисторами (2019)  
[https://www.youtube.com/watch?v=whn\\_fU95QKI](https://www.youtube.com/watch?v=whn_fU95QKI)
6. Магия транзисторов: как мы научили компьютеры думать с помощью кусочков кремния?(2023)  
[https://www.youtube.com/watch?v=\\_5W\\_GZOPa8E](https://www.youtube.com/watch?v=_5W_GZOPa8E)
7. КАК РАБОТАЕТ ПРОЦЕССОР | КАК ТРАНЗИСТОРЫ НАУЧИЛИСЬ СЧИТАТЬ? (2020)  
[https://www.youtube.com/watch?v=W6thMGQOX\\_k](https://www.youtube.com/watch?v=W6thMGQOX_k)
8. Лекция 261. Простейшая логика на реле (2015)  
<https://www.youtube.com/watch?v=gz7luCwHVNM>
9. ВСЁ что Вы хотели знать о РЕЛЕ. Виды и способы подключения -- в Теории и на Практике! (2021)  
<https://www.youtube.com/watch?v=SagbSOhlFlc>

# Список использованных источников (YouTube)

10. Как работает реле? Что такое реле? (2019)  
<https://www.youtube.com/watch?v=uIVlout6NCo>
11. Электромагнитное реле - как это работает ? (2017)  
<https://www.youtube.com/watch?v=l8KFFvOA3mg>
12. Урок №30. Реле. (2017)  
<https://www.youtube.com/watch?v=c4wDE9PstGw>
13. Как работает электромагнитное реле? Принцип работы (2023)  
<https://www.youtube.com/watch?v=bVmAzMQpj7Q>
14. Пожалуй, самый быстрый релейный компьютер в мире (2019)  
[https://www.youtube.com/watch?v=5hhbGBIP3\\_4](https://www.youtube.com/watch?v=5hhbGBIP3_4)
15. Реле против Транзисторов: Что лучше и в каких ситуациях? (2024)  
<https://www.youtube.com/watch?v=xXiZuy9hTO4>

# Список использованных источников (YouTube)

## Ламповая логика. Применяются вакуумные лампы (электронные лампы)

1. Ламповый компьютер? Это просто! (2020)  
<https://www.youtube.com/watch?v=fQ3Wv26qflg>
2. Хроники лампового компьютера: Все красиво, все сломали (2024)  
<https://www.youtube.com/watch?v=fVTqflfuly8>
3. Как это сделано? Ламповые компьютеры (2020)  
<https://www.youtube.com/watch?v=LRdius6-Uz4>
4. Лампы-транзисторы-микросхемы (2016)  
<https://www.youtube.com/watch?v=L-LrApmS9ko>
5. Урок 13. Электронная лампа. Радиолампа (2020)  
<https://www.youtube.com/watch?v=92tspHm4XFc>
6. Урок №48. Радиолампа (2019)  
<https://www.youtube.com/watch?v=cSXAJw5Y9B0>
7. ЛАМПОВОЕ УСИЛЕНИЕ. Как устроена РАДИОЛАМПА? Понятное объяснение! (2023)  
<https://www.youtube.com/watch?v=AJieI4dwtLA>

# Список использованных источников (YouTube)

**Резисторно-транзисторная логика. Resistor-Transistor Logic (RTL). Применяются биполярные транзисторы.**

1. Resistor Transistor Logic Basics: RTL NOT Gate and RTL NOR Gate (2021)  
Основы резистивно-транзисторной логики: вентиль RTL NOT и вентиль RTL NOR (2021)  
<https://www.youtube.com/watch?v=hvy4YASs-CQ>
2. Урок 308. Транзистор. Усилитель на транзисторе (2016)  
<https://www.youtube.com/watch?v=kBolzQc4j3c>
3. Принцип работы биполярного транзистора(2025)  
<https://www.youtube.com/watch?v=sas3zVZudDc>
4. В чем разница между PNP и NPN транзисторами? (2018)  
<https://www.youtube.com/watch?v=rHnJnnMtARA>
5. Урок №19. NPN и PNP биполярные транзисторы (2015)  
<https://www.youtube.com/watch?v=yQB2EnEwII0>
6. Биполярный транзистор (2020)  
[https://www.youtube.com/watch?v=LgsdYwxSA\\_k](https://www.youtube.com/watch?v=LgsdYwxSA_k)
7. Introduction to Bipolar Junction Transistor (BJT) (2019)  
Знакомство с биполярным транзистором (BJT) (2019)  
<https://www.youtube.com/watch?v=-VwPSDQmdjM>

# Список использованных источников (YouTube)

**Диодно-транзисторная логика. Diode Transistor Logic (DTL) Используются биполярные транзисторы (BJT) и диоды.**

1. Diode Transistor Logic (DTL) - Digital Circuits and Logic Design (2022)  
Диодная транзисторная логика (DTL) — цифровые схемы и проектирование логических схем (2022)  
<https://www.youtube.com/watch?v=WD-Ipu-lvaM>
2. Question on Diode Transistor Logic (DTL) - Digital Circuits and Logic Design (2022)  
Вопрос о диодной транзисторной логике (ДТЛ) — цифровые схемы и проектирование логических схем (2022) <https://www.youtube.com/watch?v=1e3L9ZTMn-E>
3. Diode Transistor Logic (DTL): DTL NAND Gate Circuit and Working (2021)  
Диодная транзисторная логика (ДТЛ): схема и принцип работы вентиля ДТЛ И-НЕ (2021)  
<https://www.youtube.com/watch?v=bGsque2SIEQ>
4. Полупроводники. Как работают транзисторы и диоды. Самое понятное объяснение (2019)  
<https://www.youtube.com/watch?v=OMGdSCaMVDO>
5. Принцип работы полупроводникового диода. ВАХ диода (2022)  
<https://www.youtube.com/watch?v=v5dzA8AmcWM>

# Список использованных источников (YouTube)

**Транзисторно-транзисторная логика. TTL (Transistor-Transistor Logic) Используются биполярные транзисторы (BJT).**

1. Логические элементы И, ИЛИ, Исключающее ИЛИ. История, Теория, Применение (2021)  
<https://www.youtube.com/watch?v=bXdjYU3IUJA>
2. Лекция 74. Транзисторно-транзисторная логика (ТТЛ) (2013)  
[https://www.youtube.com/watch?v=qOtHc\\_x70hY](https://www.youtube.com/watch?v=qOtHc_x70hY)
3. TTL Logic Explained | TTL Inverter Circuit | Noise Margin and Fanout of TTL Circuits (2025)  
Объясненная логика TTL | Схема TTL-инвертора | Уровень шума и разветвление TTL-цепей (2025)  
[https://www.youtube.com/watch?v=l74vWLGi\\_f\\_o](https://www.youtube.com/watch?v=l74vWLGi_f_o)
4. 4 2 2 Транзисторно транзисторные логические элементы (2017)  
<https://www.youtube.com/watch?v=x1gEFFctTLk>
5. Элементы транзисторно транзисторной логики (2014)  
<https://www.youtube.com/watch?v=8tolhnUvN2A>
6. Занятие 7: TTL Транзисторно-транзисторная логика (2023)  
[https://www.youtube.com/watch?v=684AaZq\\_-lI](https://www.youtube.com/watch?v=684AaZq_-lI)
7. Transistor Transistor Logic (2018)  
Транзисторно-транзисторная логика (2018)  
<https://www.youtube.com/watch?v=sW1FEm2yNnA>
8. Transistor Transistor Logic (TTL) - Digital Circuits and Logic Design (2022)  
Транзисторная транзисторная логика (ТТЛ) — цифровые схемы и логическое проектирование (2022)  
<https://www.youtube.com/watch?v=hcyC5EzOVuU>

# Список использованных источников (YouTube)

9. TTL Logic: TTL NAND and NOR gates Explained (2025)  
Логика TTL: объяснены элементы TTL NAND и NOR (2025)  
<https://www.youtube.com/watch?v=sDn2hi5ndCA>
10. Hackaday Logic Series: TTL Electrical characteristics (2015)  
Серия Hackaday Logic: Электрические характеристики TTL (2015)  
<https://www.youtube.com/watch?v=b7koDIN4m-Q>
11. 7. TTL (Transistor -Transistor Logic) Circuit | Digital Logic Families | TECH GURUKUL By Dinesh Arya (2018)  
7. Схема ТТЛ (транзисторно-транзисторной логики) | Семейства цифровых логических схем | TECH GURUKUL от Динеша Арья (2018)  
[https://www.youtube.com/watch?v=exM\\_4Vbsxl4](https://www.youtube.com/watch?v=exM_4Vbsxl4)

## Логика с эмиттерной связью, Эмиттерно-связанная логика. Emitter-Coupled Logic (ECL)

1. ECL Logic Explained | ECL OR and NOR gate explained (2025)  
Объяснение логики ECL | Объяснение работы логических элементов ECL OR и NOR (2025)  
<https://www.youtube.com/watch?v=gZdgOcE9FNO>

# Список использованных источников (YouTube)

## Логика металл-оксид-полупроводник. Metal-Oxide-Semiconductor Logic (MOS). MOSFET.

1. 4. Изготовление МОП-транзистора (2020)  
<https://www.youtube.com/watch?v=CJxt3lOJX5k>
2. Лекция 137. МОП-транзистор (2014)  
<https://www.youtube.com/watch?v=uKPkLr1AQdc>
3. 5. Принцип работы МОП-транзистора (2021)  
<https://www.youtube.com/watch?v=-lwPk--mJHM>
4. 3. Что такое pn-переход и зачем он нужен (2021)  
[https://www.youtube.com/watch?v=r\\_1A-oPRNPw](https://www.youtube.com/watch?v=r_1A-oPRNPw)
5. 2. Зачем нужны примеси в кремнии (2020)  
<https://www.youtube.com/watch?v=jDurmeWpEd4>
6. Принцип работы МОП-транзистора на примере простейшей схемы (часть 1) (2022)  
[https://www.youtube.com/watch?v=24eURhl\\_WbM](https://www.youtube.com/watch?v=24eURhl_WbM)
7. Принцип работы МОП-транзистора на примере простейшей схемы (часть 2) (2022)  
<https://www.youtube.com/watch?v=hvooC4EIAT8>
8. МОП MOSFET ТРАНЗИСТОР. ПРИНЦИП РАБОТЫ В АНИМАЦИИ. БЕЗ ЛИШНЕЙ ВОДЫ И ФОРМУЛ (2021)  
<https://www.youtube.com/watch?v=ZUly6Wz0RHc>
9. Working of Transistors | MOSFET (2018)  
Работа транзисторов | MOSFET (2018)  
<https://www.youtube.com/watch?v=stM8dgcY1CA>

# Список использованных источников (YouTube)

## Логика металл-оксид-полупроводник. Metal-Oxide-Semiconductor Logic (MOS). MOSFET.

10. How a MOSFET Works - with animation! | Intermediate Electronics (2020)  
Как работает МОП-транзистор - с анимацией! | Промежуточная электроника (2020)  
[https://www.youtube.com/watch?v=Bfyuj88Hs\\_o](https://www.youtube.com/watch?v=Bfyuj88Hs_o)
11. NMOS vs PMOS and Enhancement vs Depletion Mode MOSFETs | Intermediate Electronics (2020)  
NMOS против PMOS и МОП-транзисторы в режиме усиления и в режиме истощения | Промежуточная электроника (2020) <https://www.youtube.com/watch?v=kY-kaOPriaE>

# Список использованных источников (YouTube)

## Комплементарная логика металл-оксид-полупроводник (КМОП). Complementary Metal-Oxide-Semiconductor Logic (CMOS)

1. Лекция 138. КМОП-логика (2014)  
<https://www.youtube.com/watch?v=9eIkCWpPlqY>
2. Что такое КМОП микросхемы. CMOS. На реальных примерах (2021)  
<https://www.youtube.com/watch?v=iYvp13bRuFk>
3. 4 2 3 Логические элементы КМОП (2017)  
<https://www.youtube.com/watch?v=banmRPMATEw>
4. AND и NAND из КМОП транзисторов(2019)  
[https://www.youtube.com/watch?v=Yw5To\\_b7Yzo](https://www.youtube.com/watch?v=Yw5To_b7Yzo)
5. CMOS Tech: NMOS and PMOS Transistors in CMOS Inverter (3-D View) (2020)  
Технология CMOS: NMOS и PMOS-транзисторы в CMOS-инверторе (трехмерный вид) (2020)  
<https://www.youtube.com/watch?v=oSrUsM0hoPs>
6. CMOS Inverter Model Using Light Switches (2020)  
Модель инвертора CMOS с использованием выключателей (2020)  
<https://www.youtube.com/watch?v=b4SmRKrqpq4>