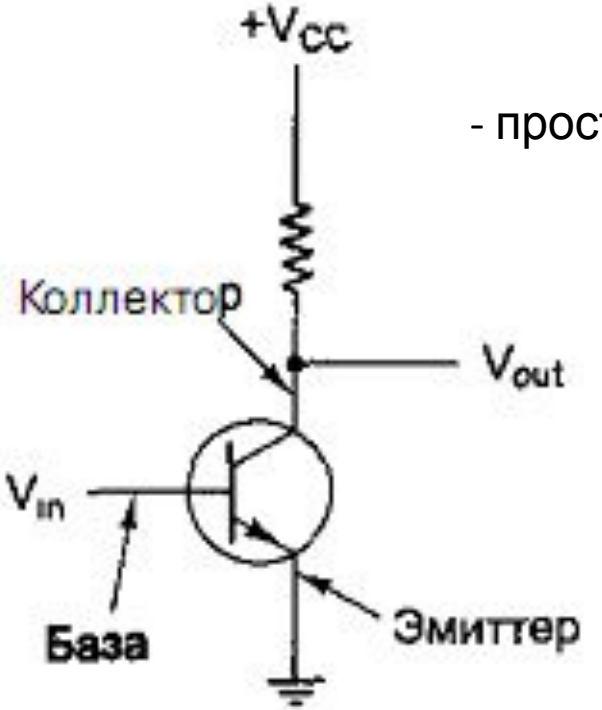
Цифровой логический уровень

Транзистор – основной элемент цифровой электроники.

- если входное напряжение V_{in} ниже определенного значения, транзистор выключен и не проводит ток – в результате $V_{out} = V_{cc}$ (внешнее напряжение, обычно 5 вольт)

- если входное напряжение V_{in} превышает критическое значение, транзистор открывается и проводит ток – в результате $V_{out} = 0$

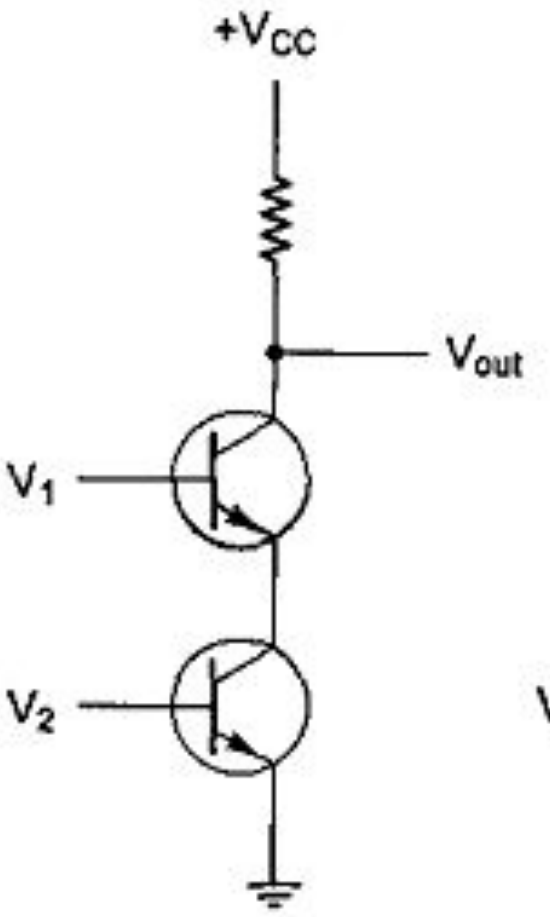
Таким образом если V_{in} высокое, то V_{out} низкое, и наоборот. То есть транзистор является **инвертором** – превращает 0 в 1, а 1 в 0.



- простой транзисторный инвертер

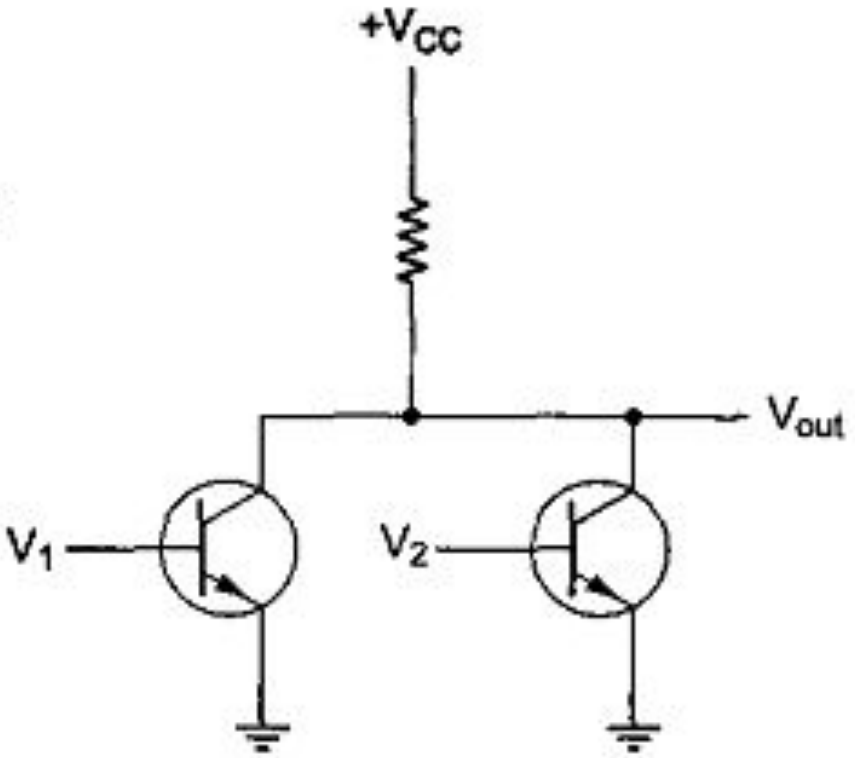
Резистор ограничивает протекаемый ток, чтобы транзистор не сгорел

Время переключения – несколько наносекунд



Вентиль И-НЕ

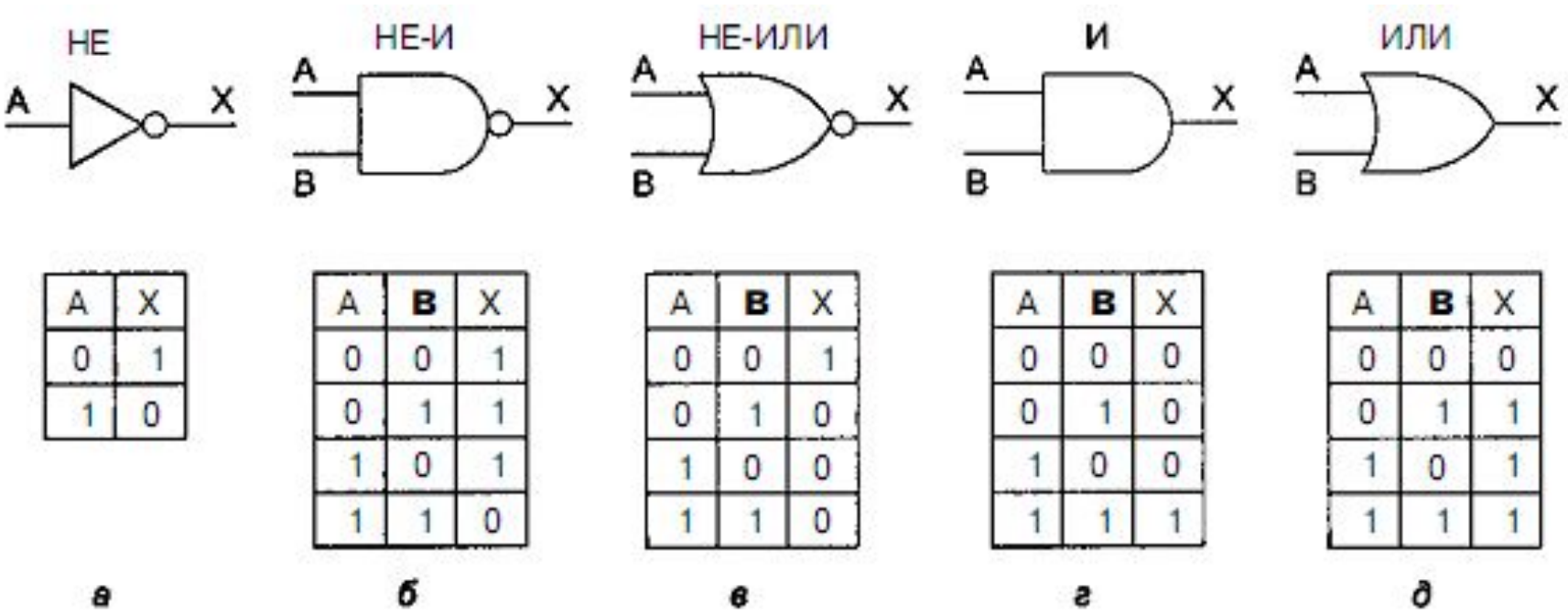
Если и V1, и V2 высокие, то оба транзистора проводят ток, и V_{out} низкое



Вентиль ИЛИ-НЕ

Если хотя бы одно из V1 V2 высокое, то ток уходит на землю, и V_{out} низкое

Обозначения основных вентилей и таблицы истинности



A	X
0	1
1	0

в

A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

б

A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

в

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

г

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

д

Вентили НЕ-И и НЕ-ИЛИ требуют по 2 транзистора, а вентили И и ИЛИ – по 3.

На практике вентили делают несколько по-другому, но всё равно НЕ-И и НЕ-ИЛИ проще и чаще используются в цифровых электронных схемах

Классификация технологий производства вентилей

Биполярная

- ТТЛ (транзисторно-транзисторная логика)
- ЭСЛ (эмиттерно-связанная логика) – более высокая скорость работы

МОП (металл-оксид-полупроводник)

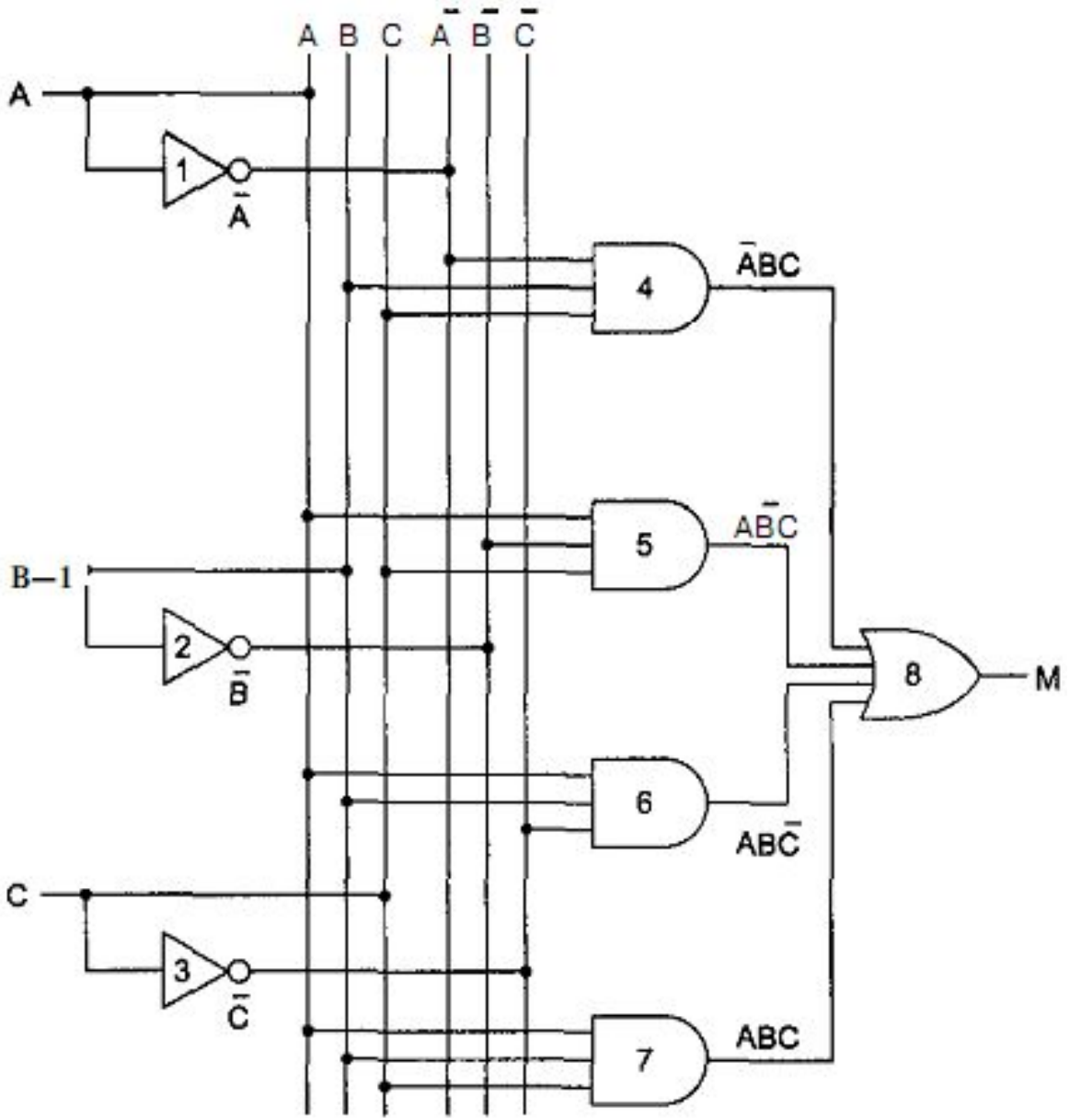
Работают медленнее ТТЛ и ЭСЛ, но компактны и потребляют мало энергии – можно разместить много на ограниченной площади.

Недорогие процессоры и память часто производят (по крайне мере, до недавних пор производили) по технологии комплиментарных МОП.

Стандартное напряжение работы +3.3 В.

Функция большинства – на выходе 1, если большинство переменных = 1

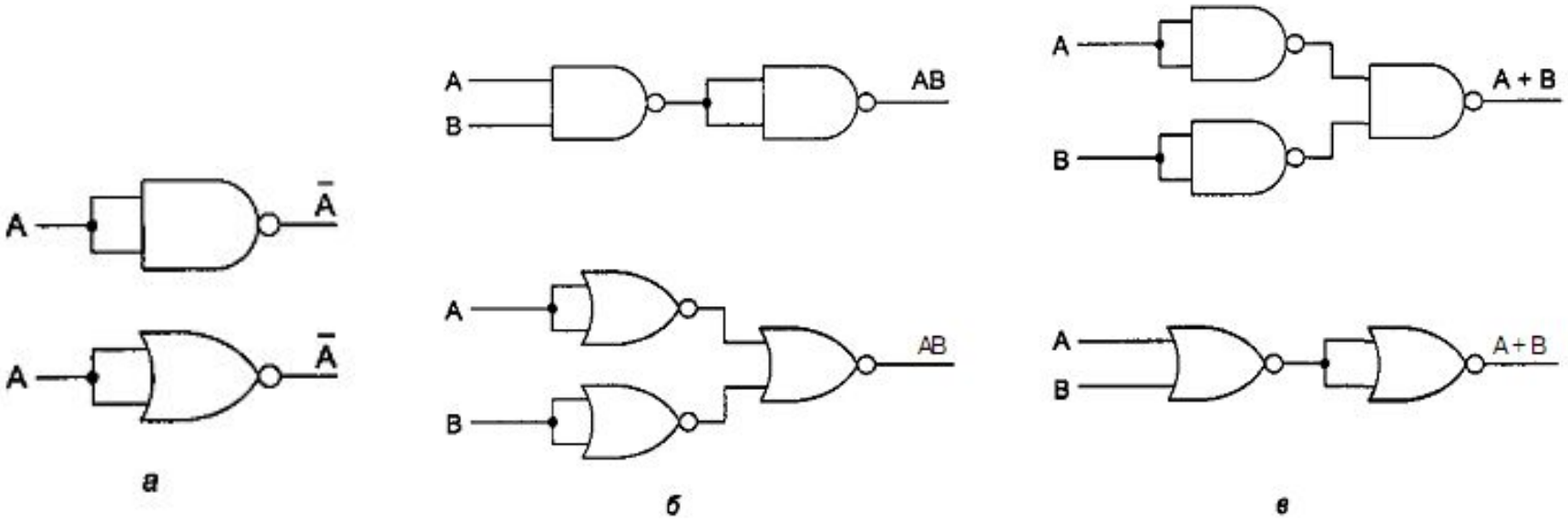
A	B	C	M
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1



Алгоритм построения схемы для любой булевой функции

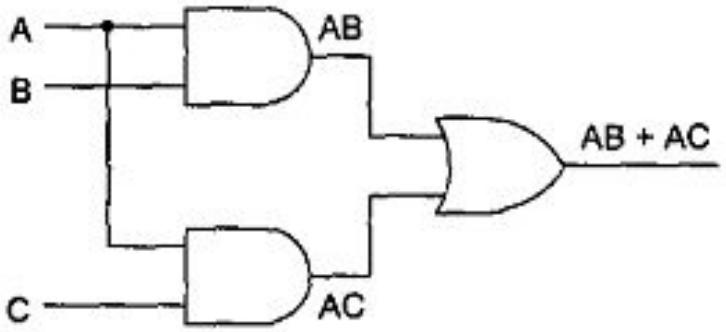
1. Составить таблицу истинности для данной функции.
2. Обеспечить инверторы, чтобы порождать инверсии для каждого входного сигнала.
3. Нарисовать вентиль И для каждой строки таблицы истинности с результатом 1.
4. Соединить вентили И с соответствующими входными сигналами.
5. Вывести выходы всех вентилей И в вентиль ИЛИ.

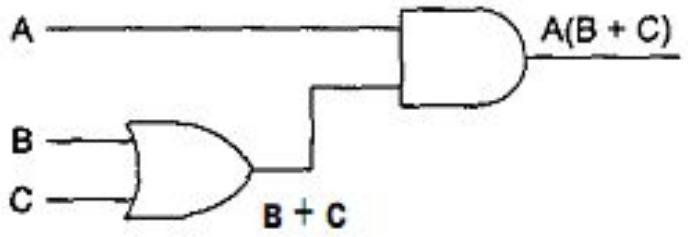
Построенную схему можно преобразовать, чтобы использовать только один тип вентилей - И-НЕ *или* ИЛИ-НЕ



НЕ И ИЛИ

Одну и ту же функцию можно реализовать разными схемами с разным числом элементов





A	B	C	AB	AC	AB+AC
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	0	0
0	1	1	0	0	0
1	0	0	0	0	0
1	0	1	0	1	1
1	1	0	1	0	1
1	1	1	1	1	1

A	B	C	A	B+C	A(B + C)
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	1	0
0	1	1	0	1	0
1	0	0	1	0	0
1	0	1	1	1	1
1	1	0	1	1	1
1	1	1	1	1	1

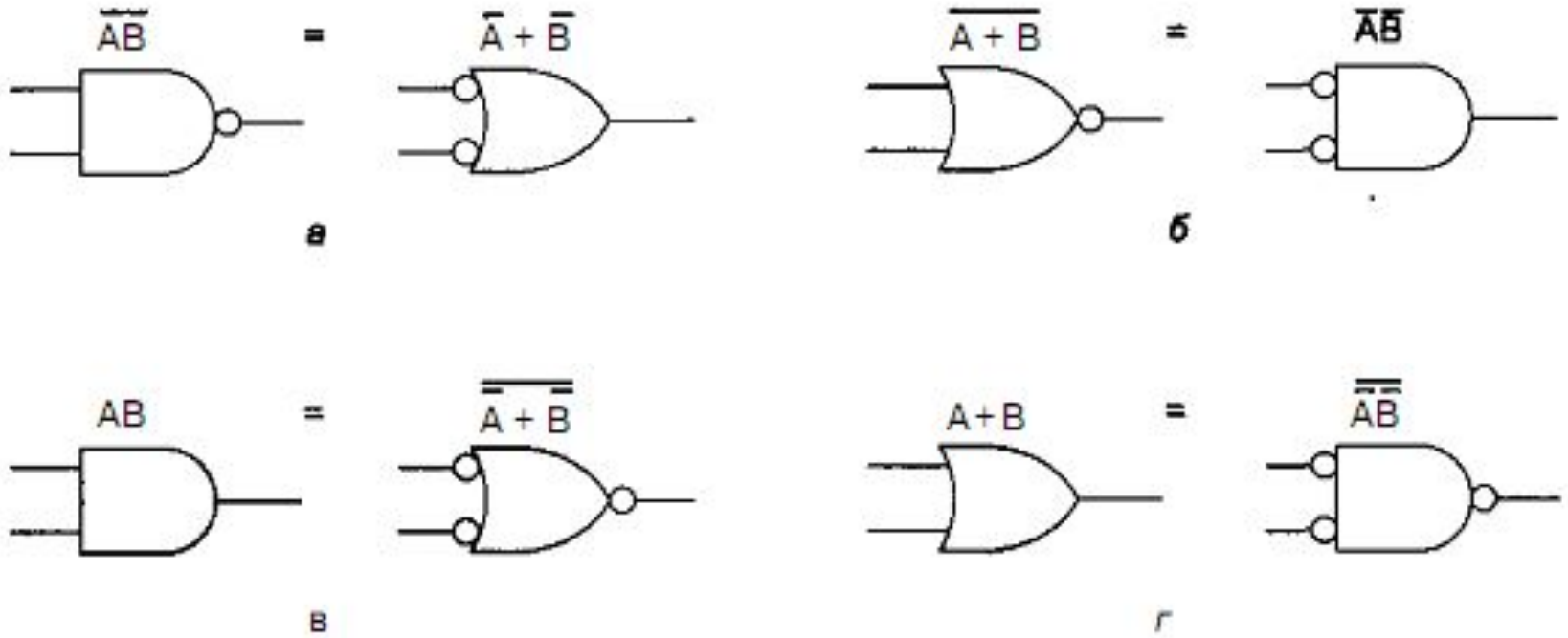
Для минимизации булевых функций используются законы булевой алгебры

$a \vee b = b \vee a$;	$a \wedge b = b \wedge a$.	1 коммутативность переместительность
$a \vee (b \vee c) = (a \vee b) \vee c$;	$a \wedge (b \wedge c) = (a \wedge b) \wedge c$.	2 ассоциативность сочетательность
3.1 конъюнкция относительно дизъюнкции $a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$	3.2 дизъюнкция относительно конъюнкции $a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$	3 дистрибутивность распределительность
$a \vee \neg a = 1$;	$a \wedge \neg a = 0$.	4 комплементность дополнительность (свойства отрицаний)
$\neg(a \vee b) = \neg a \wedge \neg b$;	$\neg(a \wedge b) = \neg a \vee \neg b$.	5 законы де Моргана
$a \vee (a \wedge b) = a$;	$a \wedge (a \vee b) = a$.	6 законы поглощения
$a \vee (\neg a \wedge b) = a \vee b$;	$a \wedge (\neg a \vee b) = a \wedge b$.	7 Блейка-Порецкого
$a \vee a = a$;	$a \wedge a = a$.	8 Идемпотентность
$\neg\neg a = a$.		9 инволютивность отрицания
$a \vee 0 = a$;	$a \wedge 1 = a$.	
$a \vee 1 = 1$;	$a \wedge 0 = 0$.	10 свойства констант
дополнение 0 есть 1 $\neg 0 = 1$;	дополнение 1 есть 0 $\neg 1 = 0$.	

Идемпотентност

Иллюстрация к законам Де-Моргана

$$\neg(a \vee b) = \neg a \wedge \neg b; \qquad \neg(a \wedge b) = \neg a \vee \neg b.$$

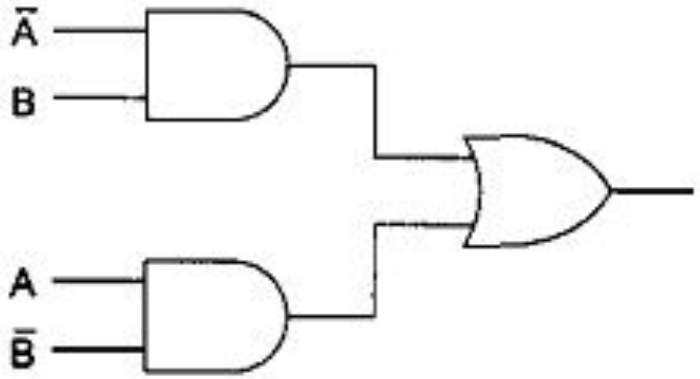


Альтернативные обозначения некоторых вентилей: НЕ-И (а); НЕ-ИЛИ (б); И (е); ИЛИ (г)

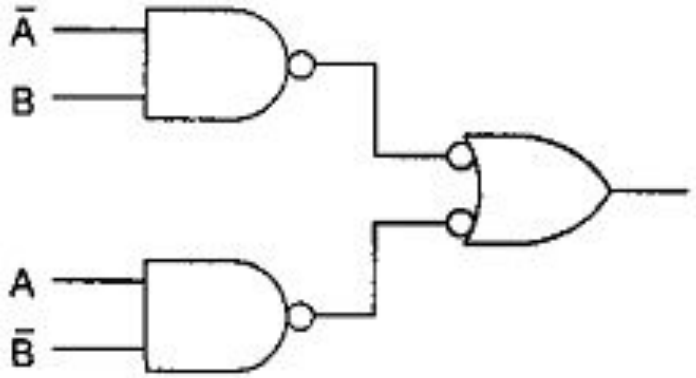
Примеры схем для функции XOR (Исключающее ИЛИ)

A	B	XOR
0	0	0
0	1	1
1	0	1
1	1	0

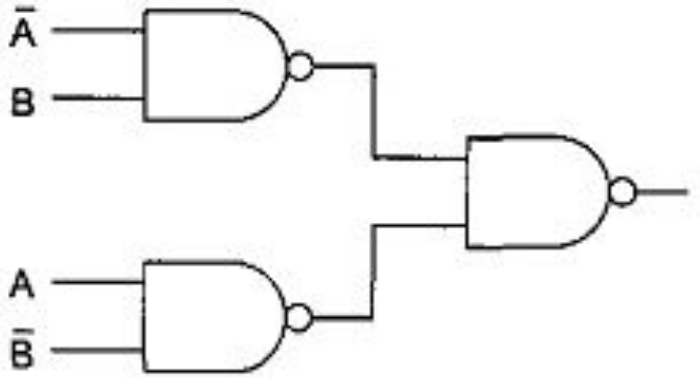
а



б



в



г

Позитивная и негативная логика

Позитивная логика:
0 – 0 вольт, 1 – 3.3 или 5 вольт

Негативная логика:
1 – 0 вольт, 0 – 3.3 или 5 вольт

Одна и та же схема реализует разную функцию в негативной и позитивной логике:

A	B	F
0V	0V	0V
0V	5V	0V
5V	0V	0V
5V	5V	5V

а

A	B	F
0	0	0
0	1	0
1	0	0
1	1	1

б

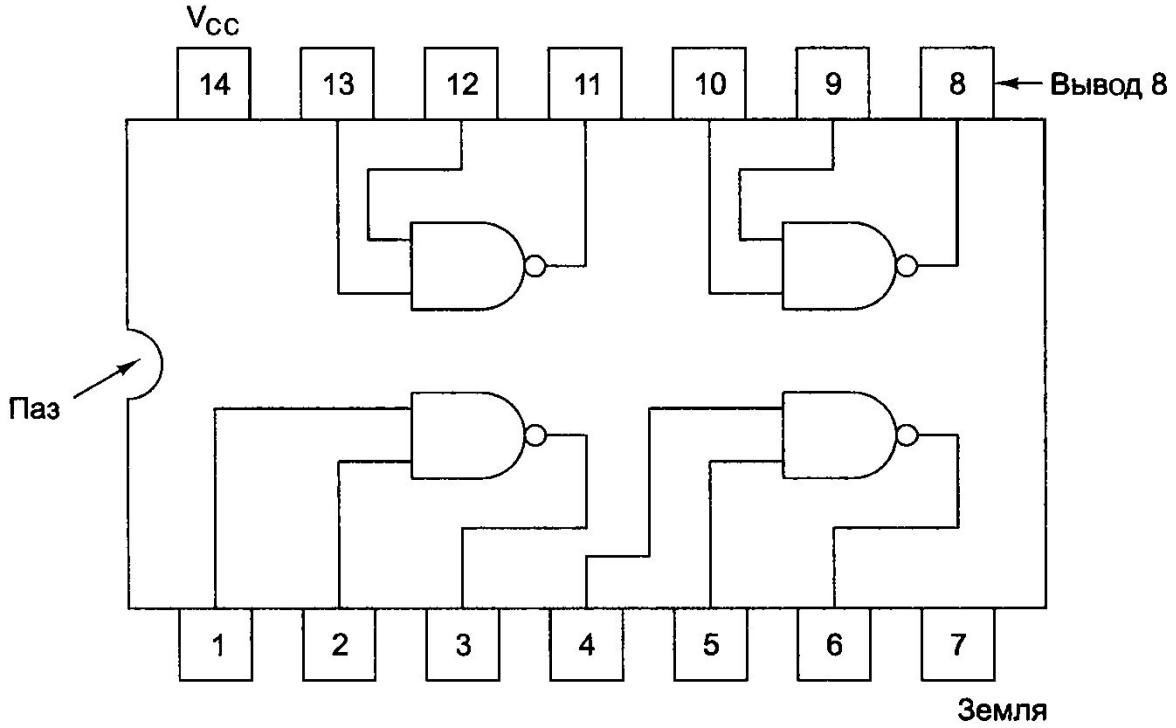
A	B	F
1	1	1
1	0	1
0	1	1
0	0	0

в

Электрические характеристики устройства (а); позитивная логика (б); негативная логика (s)

Интегральные схемы

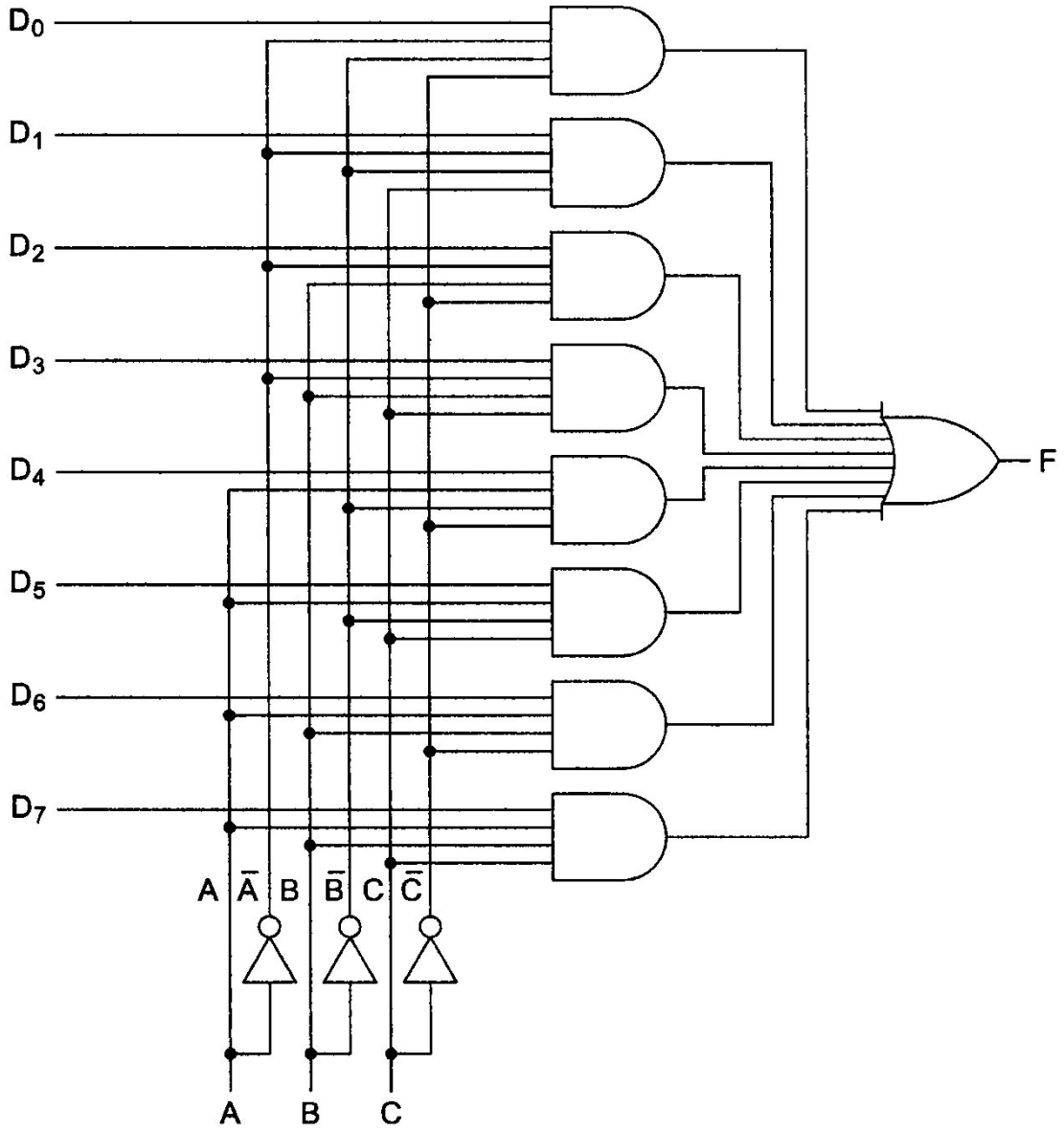
- МИС (малая интегральная схема): от 1 до 10 вентилей.
- СИС (средняя интегральная схема): от 1 до 100 вентилей.
- БИС (большая интегральная схема): от 100 до 100 000 вентилей.
- СБИС (сверхбольшая интегральная схема): более 100 000 вентилей.



МИС из четырех вентилей

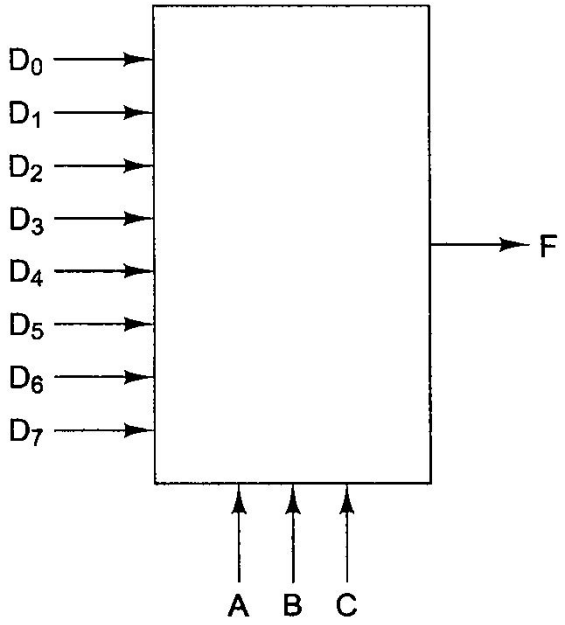
Сейчас на одну микросхему помещают уже десятки миллионов транзисторов

Мультиплексор

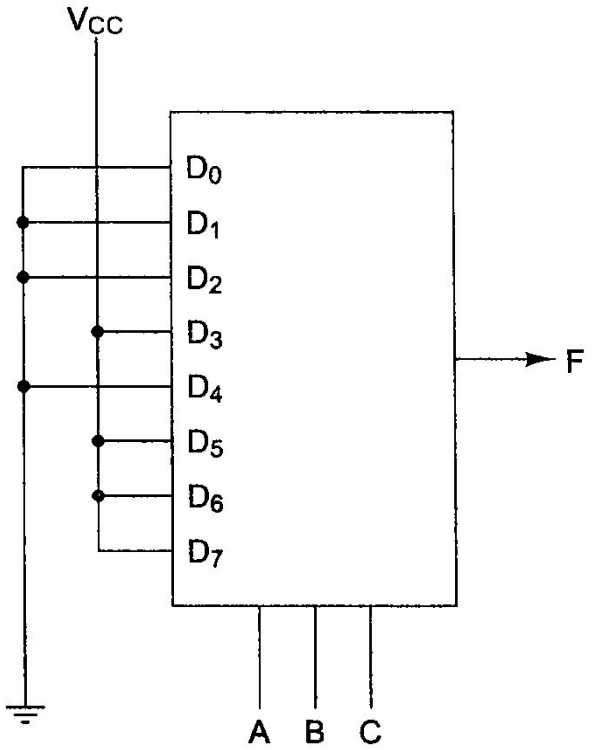


Линии управления A,B,C кодируют 3-разрядное двоичное число, которое определяет, какую из 8-ми ***входных линий*** соединить с выходом

Схемы на мультиплексорах



Обозначение мультиплексора



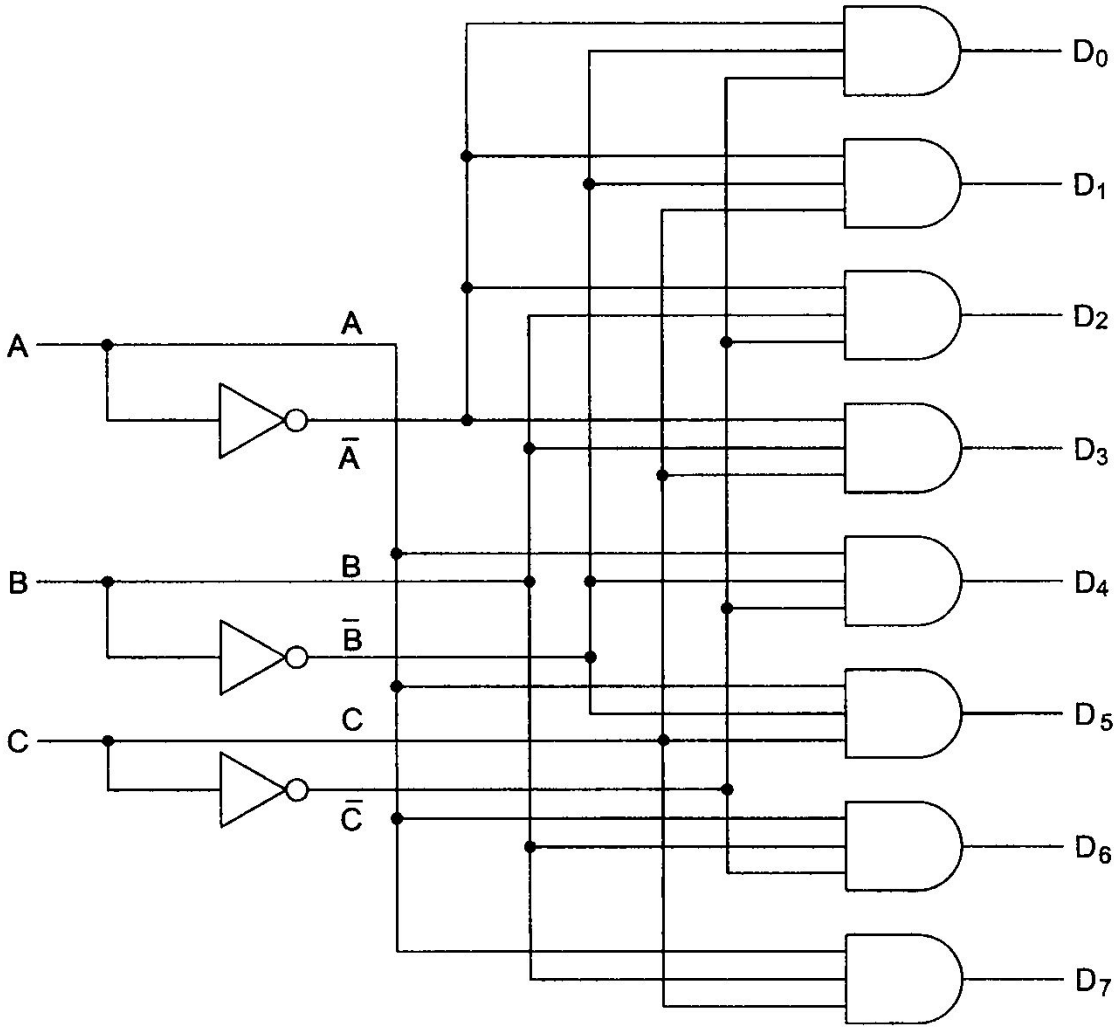
Пример реализации функции большинства на мультиплексоре (просто подаём 1 на те линии, для которых ответ по табличке истинности = 1)

Вывод: на мультиплексоре легко реализовать любую логическую функцию по её таблице истинности

А ещё с помощью мультиплексора можно преобразовывать параллельный код в последовательный

Декодер

получает на вход n-разрядное число i и выставляет 1 на i-й линии

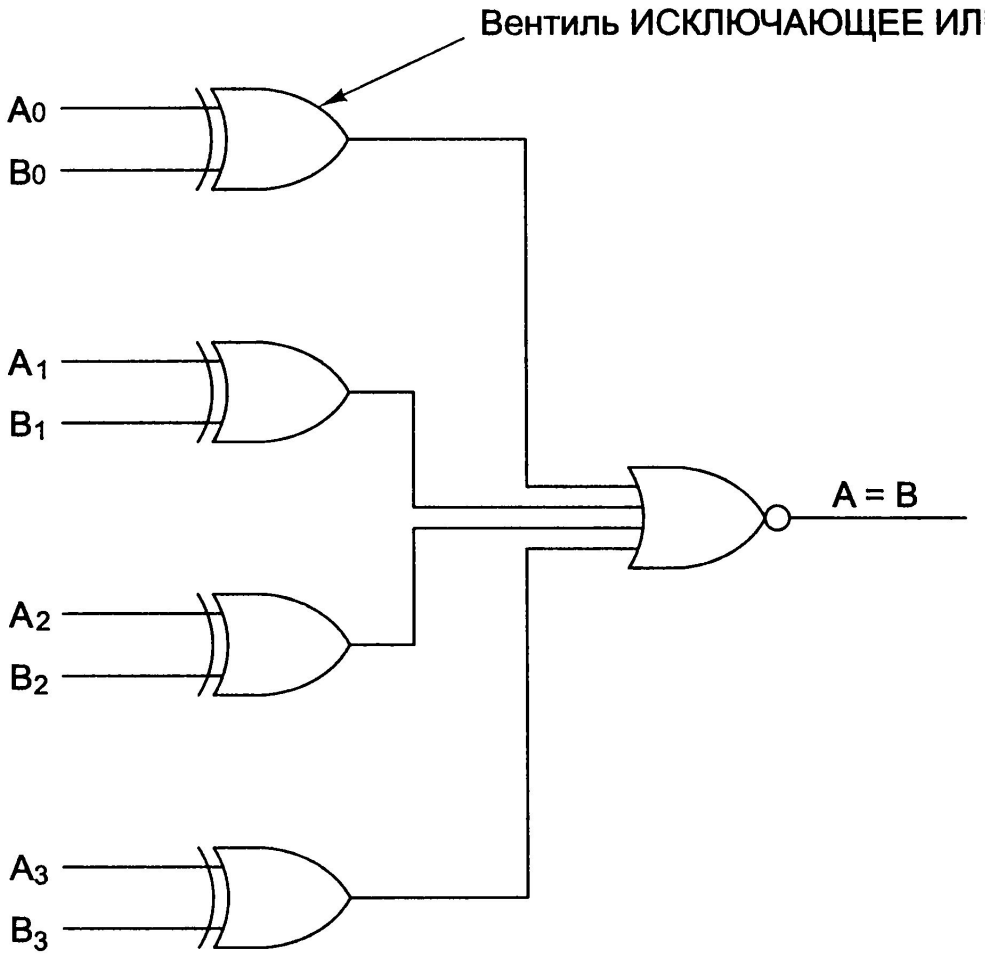


Принцип действия: каждый вентиль запускается уникальной комбинацией входов

Пример применения: на плате имеется 8 микросхем памяти по 1 мегабайту, нужно выбрать одну из них

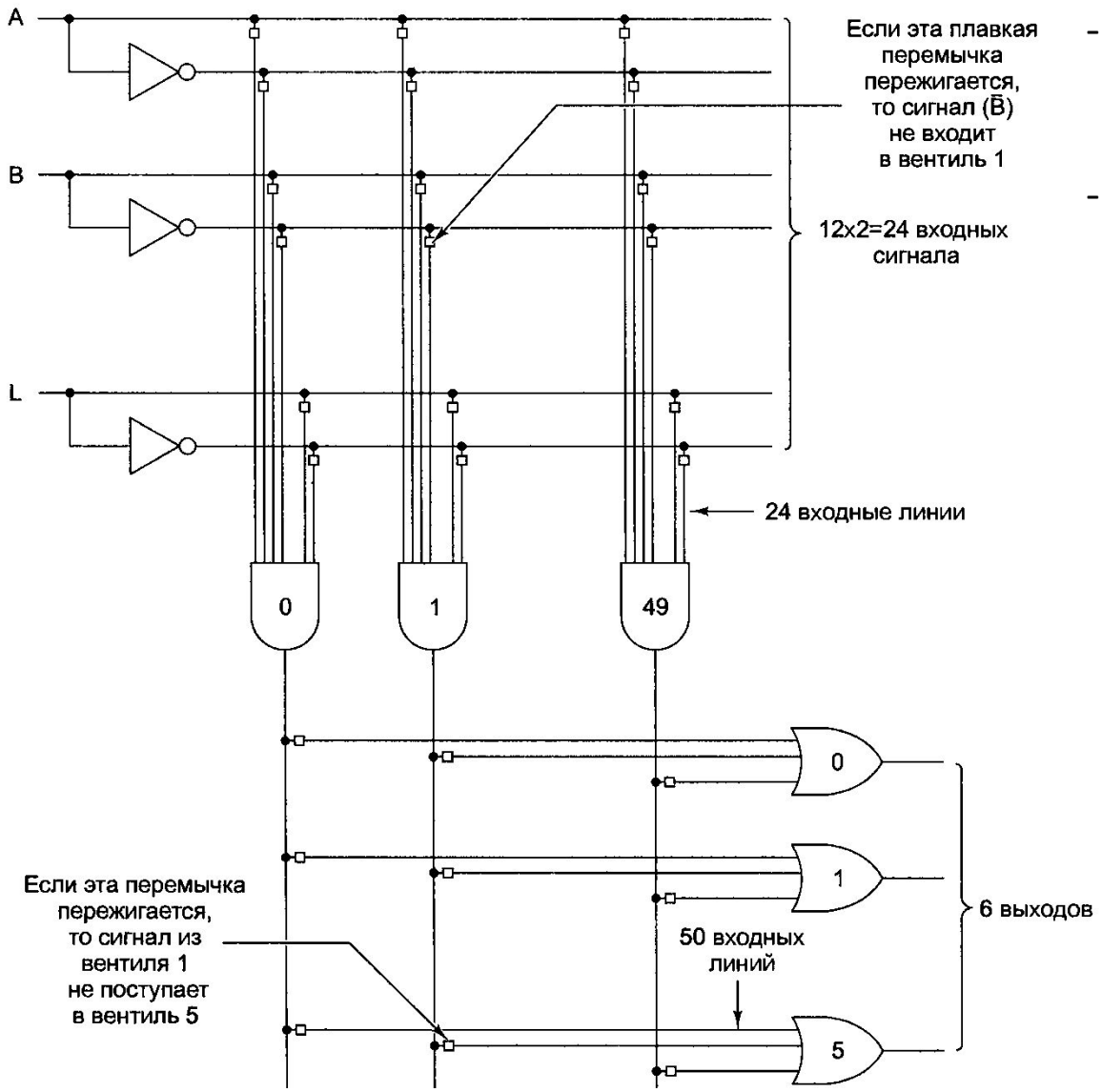
Компаратор

сравнивает n-разрядные слова на равенство/неравенство



Программируемые логические матрицы

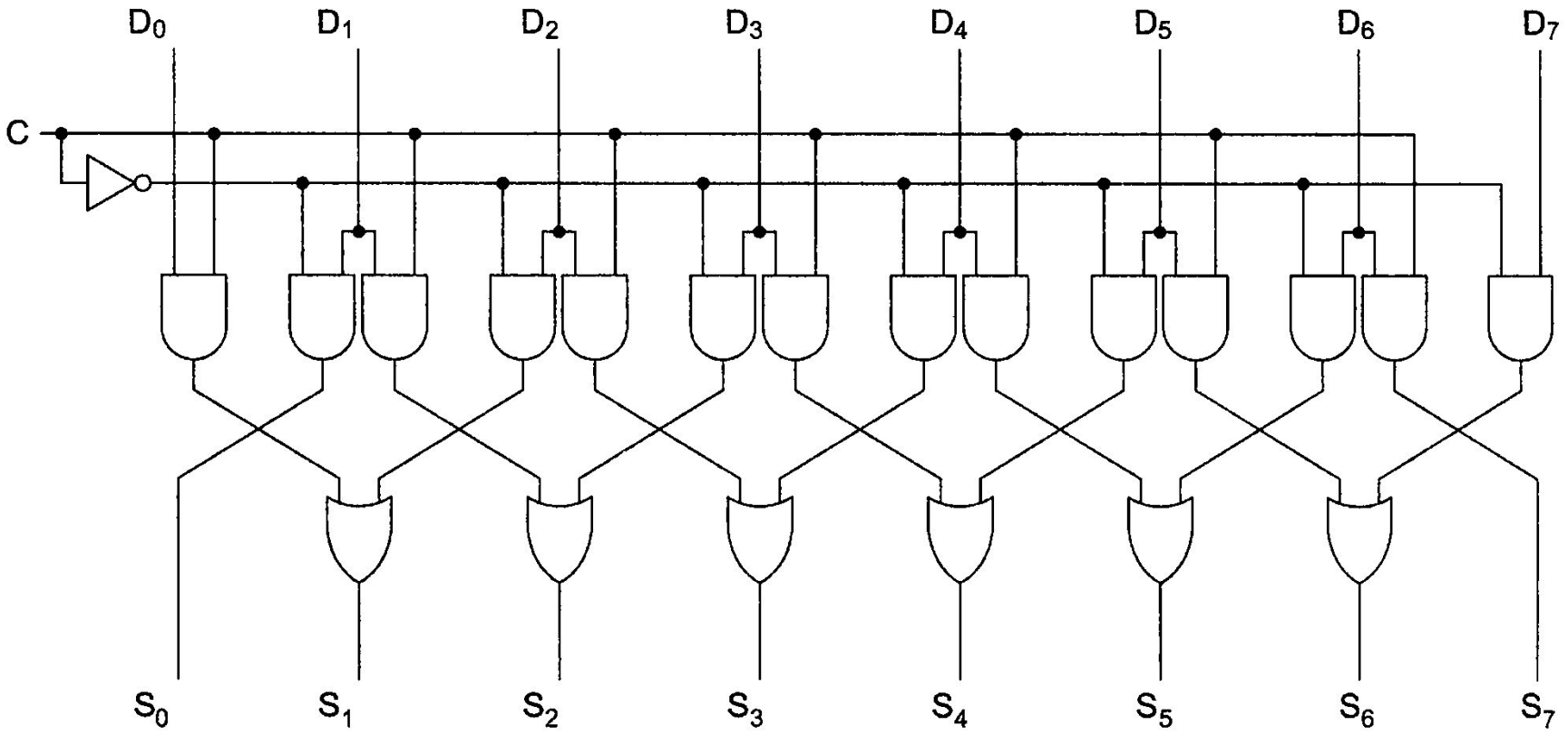
Содержат плавкие перемычки. Пережигая их, можно получать разные логические схемы



- ПЛМ с 12 входами и 6 выходами
- Обычно дешевле сразу заказать нужную конфигурацию на заводе

Арифметические схемы

Схема сдвига:

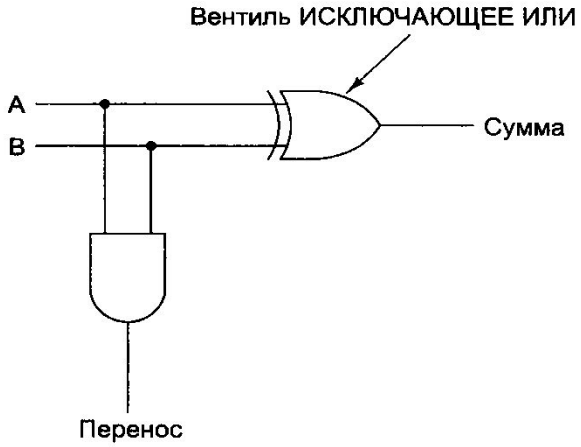


D – входные линии, S – выходные линии

C – направление сдвига (0 – влево, 1 – вправо)

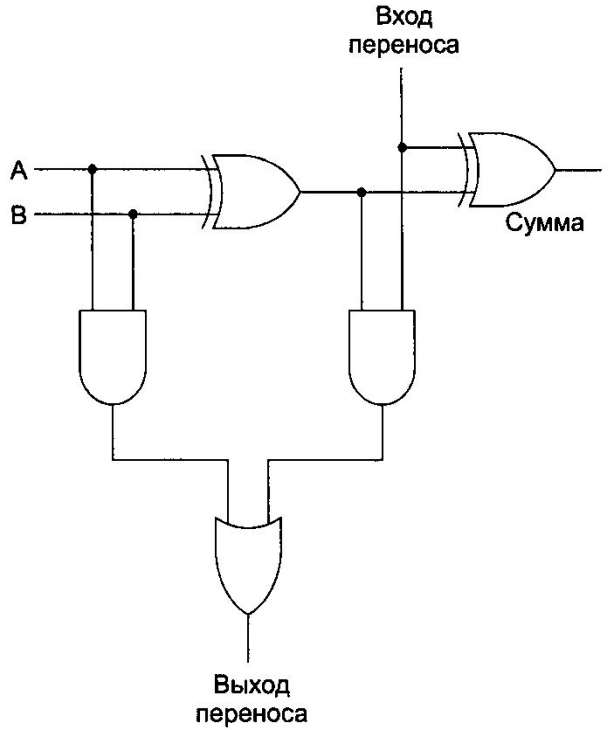
Полусумматор

A	B	Сумма	Перенос
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



Полный одноразрядный сумматор

A	B	Вход переноса	Сумма	Выход переноса
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



Сумматоры многоразрядных чисел

Простой подход:

Соединить последовательно N одноразрядных полных сумматоров (получится ***сумматор со сквозным переносом***).

Минус – скорость работы в N раз ниже, чем у одноразрядного сумматора.

Более быстрый подход:

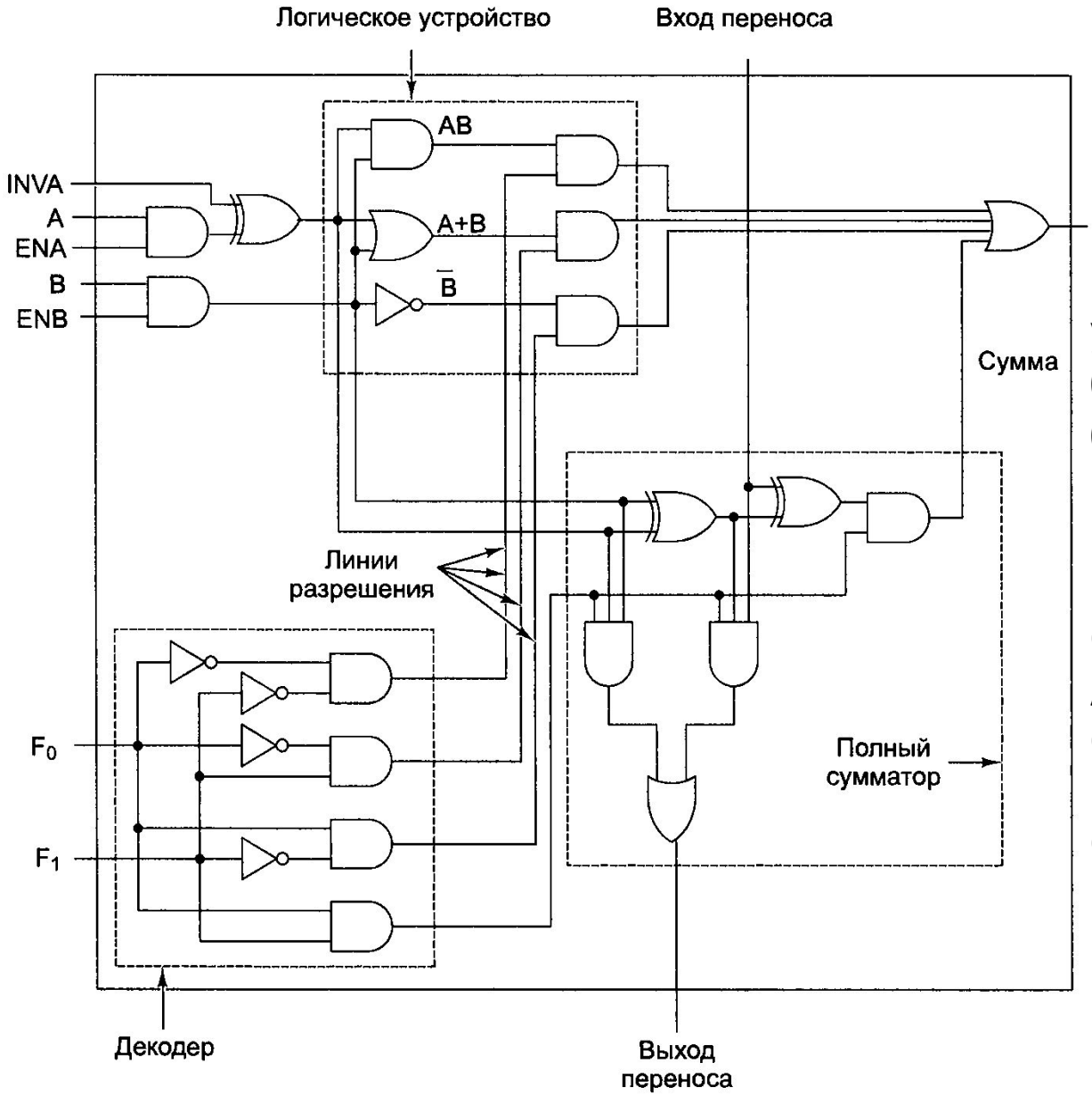
Пусть N=32. Разобьём 32-битный сумматор на две половины: нижний (младший) - **L** и два верхних (старших) - **U0** и **U1**, при этом:

- **U0** предполагает, что перенос в 16-й разряд = 0
- **U1** предполагает, что перенос в 16-й разряд = 1

В конце расчёта берётся верная старшая часть, а неверная отбрасывается.

Можно каждый 16-битный сумматор ещё разбить на 8-битные, и т.д.

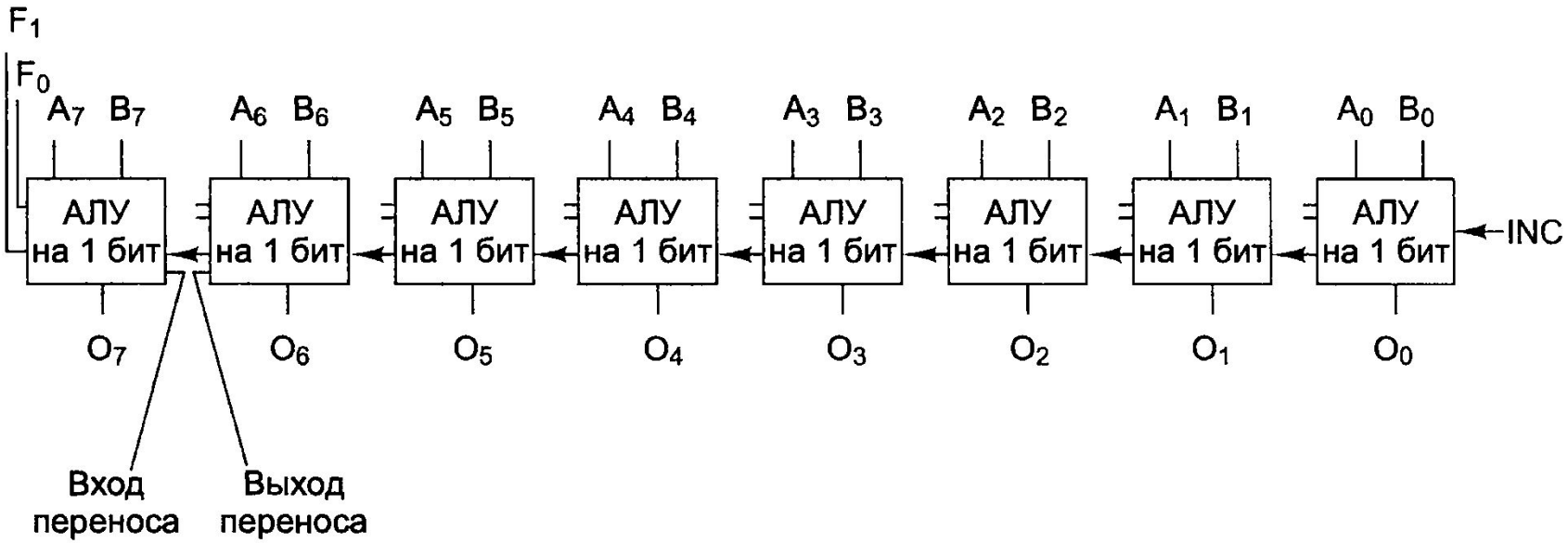
Простейшее одноразрядное арифметико-логическое устройство
(одноразрядная микропроцессорная секция)



F0 И F1 – команда управления:
00 – считать A И B
01 – считать A ИЛИ B
10 – считать НЕ B
11 – считать A+B

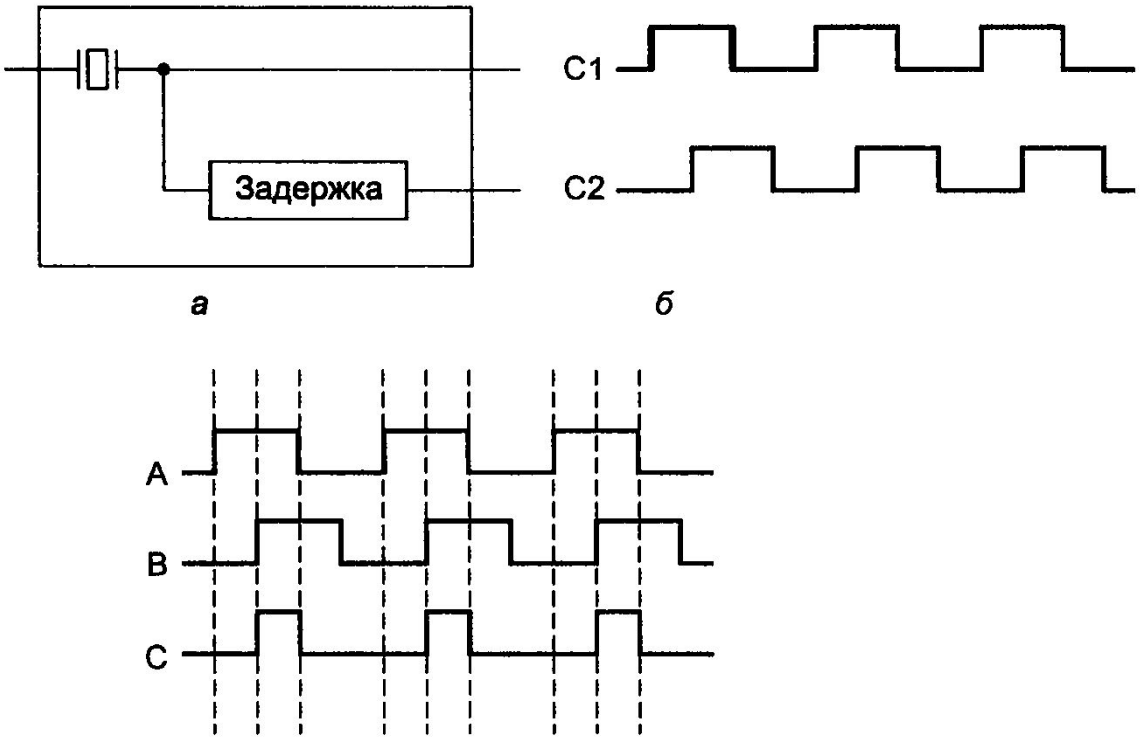
Сигнал ENA=0 – считать, что A=0
Сигнал ENB=0 – считать, что B=0
Сигнал INVA=1 – работать с инвертированным A

Простейшее 8-разрядное арифметико-логическое устройство



Сигнал INC позволяет считать A+1 или A+B+1

Тактовые генераторы (генераторы импульсов)

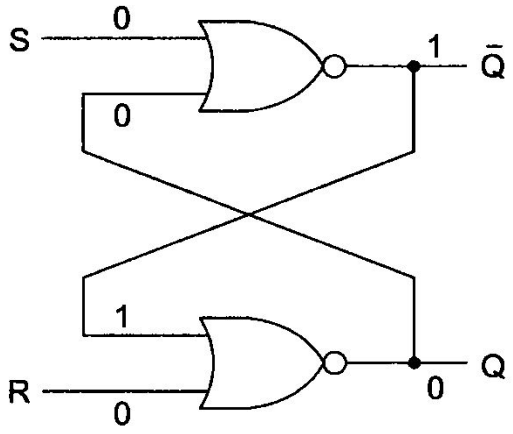


Если нужно на каждом такте выполнить несколько событий в определённом порядке, то можно сделать ответвление от сигнала тактового генератора и вставить схему задержки.

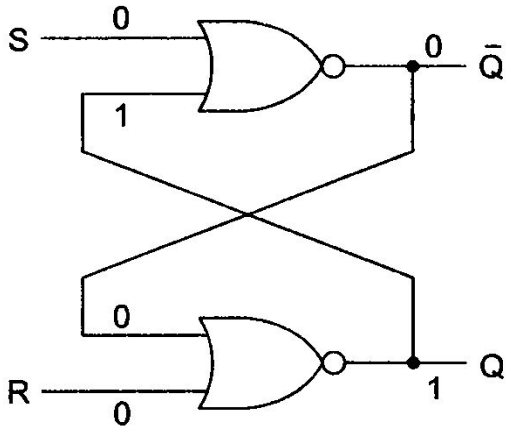
Синхронный генератор – время пика = времени спада (A и B на рисунке)
Асинхронный генератор – время пика <> времени спада (сигнал C на рисунке)

Устройство памяти

SR-защелка:



а



б

A	B	НЕ-ИЛИ
0	0	1
0	1	0
1	0	0
1	1	0

в

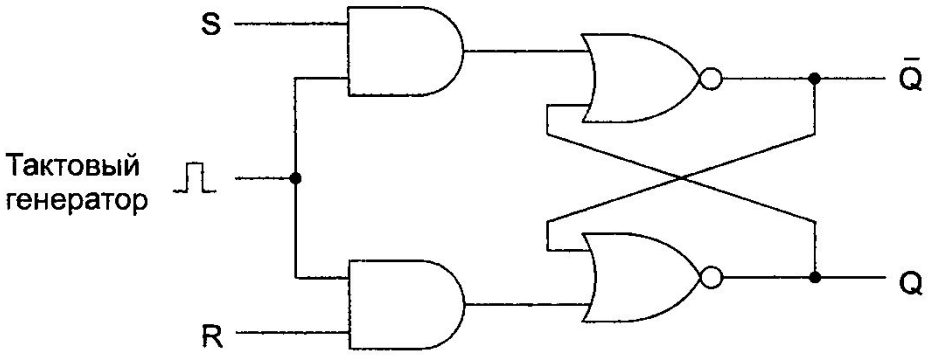
В режиме хранения S=R=0, и защелка может находиться в одном из двух устойчивых состояний (хранить бит 0 или 1)

S = 1 – заносит в защёлку 1

R = 1 – заносит в защёлку 0

Одновременно S=1 и R=1 – некорректное действие

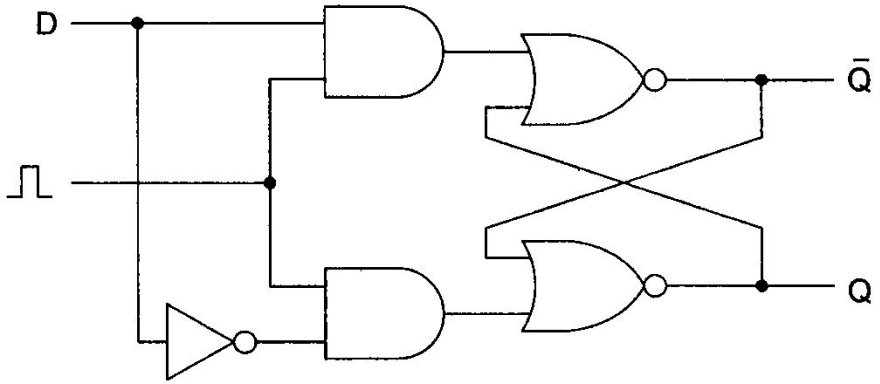
Синхронная SR-защелка:



Появление единицы на синхронизирующем входе – ***включение*** или ***стробирование***

Синхронная D-защелка (элемент памяти в 1 бит):

Нет проблемы неоднозначности при S=R=1



Такая схема требует 11 транзисторов.
Существуют элементы памяти всего на 6 транзисторах.

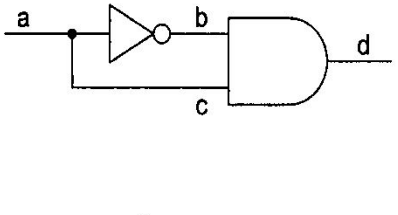
Отличия защелок и триггеров

Защелка (latch) запускается ***уровнем сигнала***

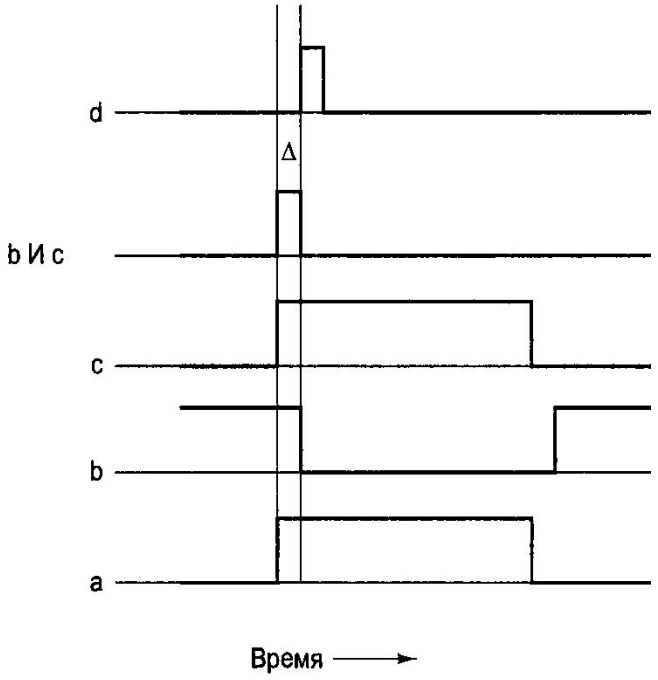
Триггер (flip-flop) запускается ***перепадом сигнала*** (с 0 на 1 или наоборот)

В отечественной литературе защелка называется триггером, а триггер – Т-триггером.

Для создания триггера можно применить схему, дающую очень короткий импульс на D-защелку (ей этого хватит, чтобы сработать).
Простейшая схема:



а

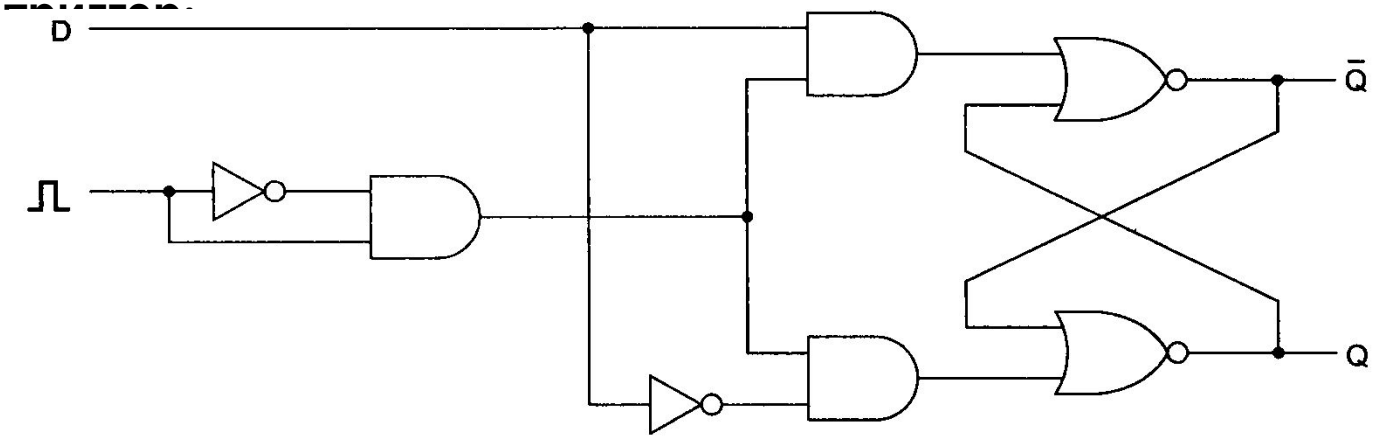


б

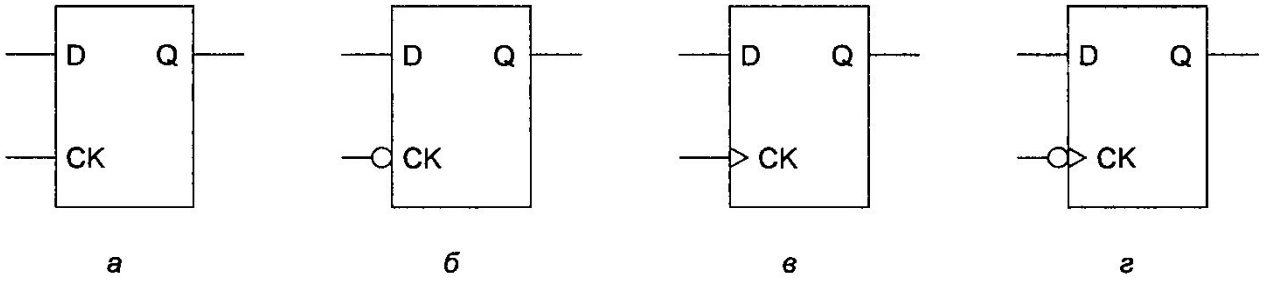
Смысл в том, что элемент «НЕ» срабатывает с небольшой задержкой, и на короткое время на элемент «И» подадутся две единицы.

На схеме показано напряжение в разных точках схемы как функция от времени

В результате получаем D-



Обозначения защелок и триггеров:



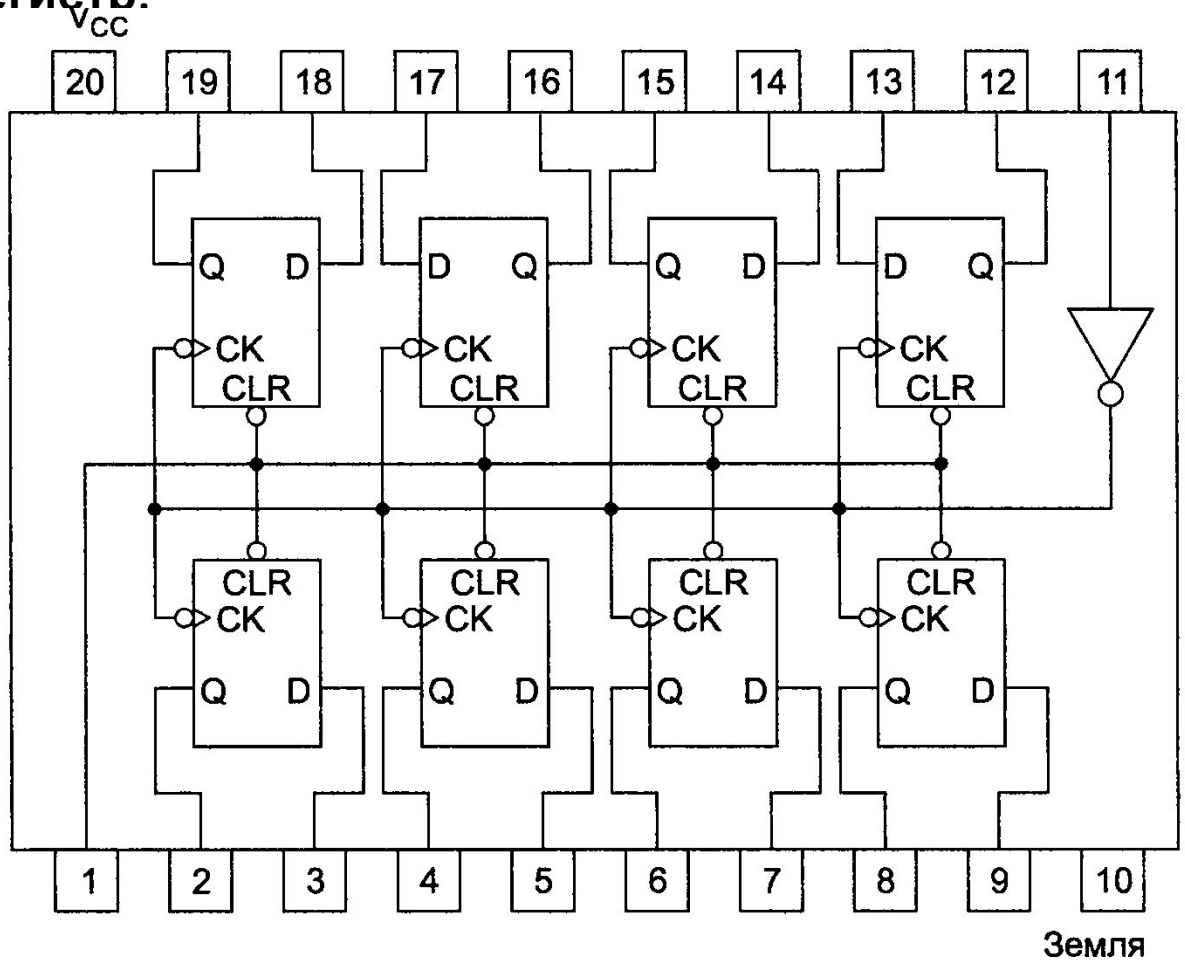
а – защелка, загружающая значение при СК=1,

б – защелка, загружающая значение при СК=0,

в – триггер, устанавливающий значение на фронте синхросигнала,

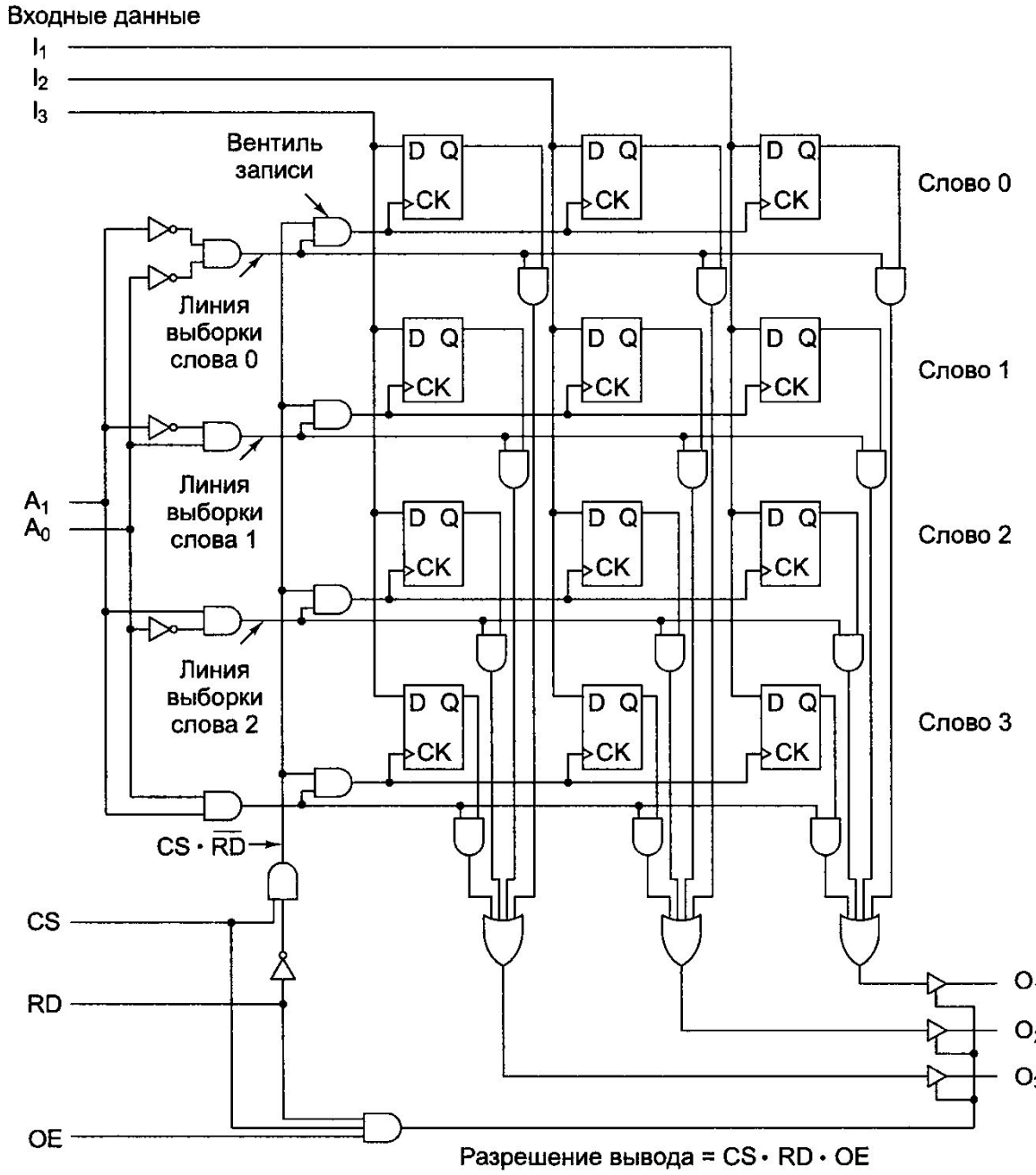
г - триггер, устанавливающий значение на спаде синхросигнала,

8-битный регистр:



Синхросигнал подаётся на 11-й вход. Он инвертируется на входе в микросхему, а потом ещё раз на входе в каждый триггер просто для усиления сигнала (иначе мощности сигнала не хватит на запуск 8-ми триггеров)

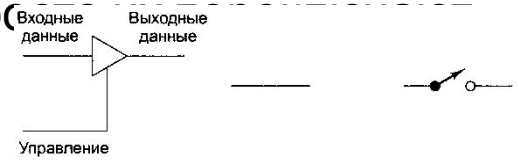
Организация памяти большого объёма



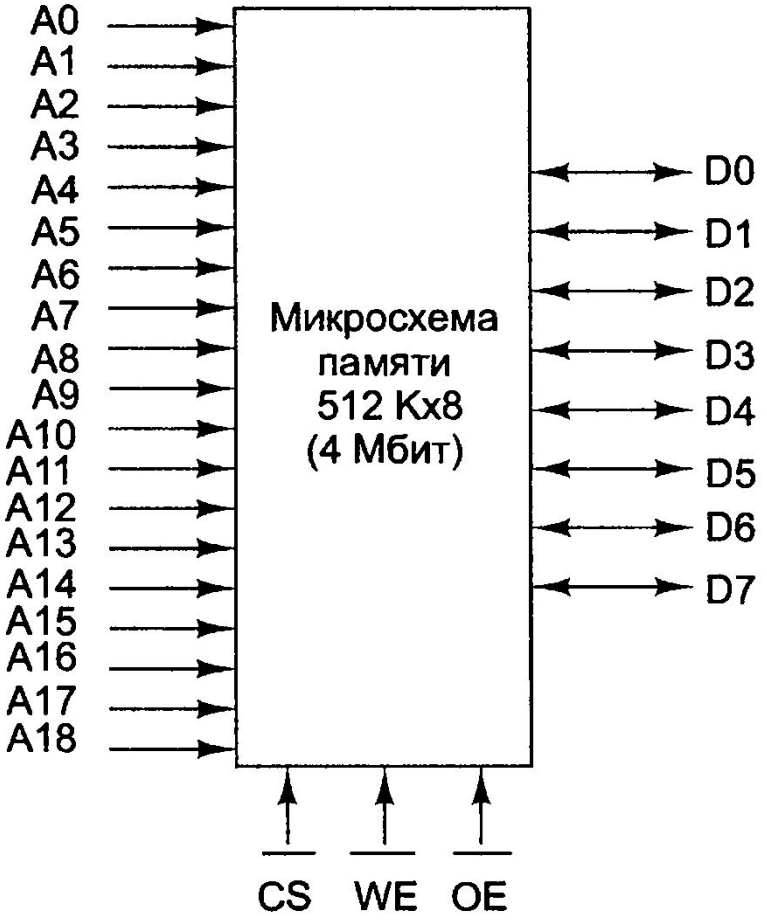
На рисунке память содержит четыре 3-разрядных слова
Каждая операция считывает или записывает одно слово

A0 и **A1** – адресные входы
I0, **I1**, **I2** – входы для данных
O0,**O1**,**O2** – выходы для данных
CS (chip select) - выбор элемента памяти
RD (read) – чтобы отличить чтение от записи (1-чтение, 0 – запись)
OE (output enable) – разрешение выдачи выходных сигналов

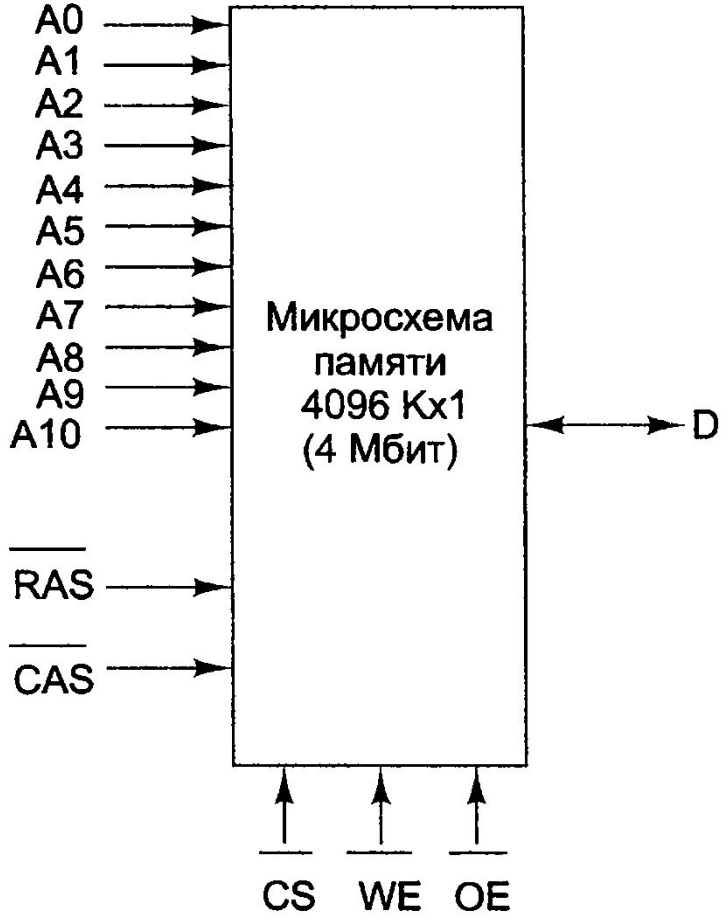
На практике для входных и выходных линий используют одни и те же проводники, про



Разные подходы к построению схем памяти



а



б

а – возвращает 8-битное слово

б – возвращает 1 бит, но работает в 2 раза медленнее: сначала подаётся номер строки (и сигнал RAS), потом номер столбца (и сигнал CAS)