БАЗОВЫЕ ЛОГИЧЕСКИЕ ОПЕРАЦИИ

В алгебре высказываний, как и в обычной алгебре, вводится ряд операций. Логические связки И, ИЛИ и НЕ заменяются логическими операциями: ***конъюнкцией, дизъюнкцией и инверсией.*** Это основные логические операции, при помощи которых можно записать любую логическую функцию.

1. Логическая операция ИНВЕРСИЯ (ОТРИЦАНИЕ)

- соответствует частице НЕ
- обозначается черточкой над именем переменной или знаком ¬ перед переменной
- **Инверсия логической переменной истинна, если сама переменная ложна, и, наоборот, инверсия ложна, если переменная истинна.**

Таблица истинности инверсии имеет вид:

A	$\overline{A}$
0	1
1	0

2. Логическая операция ДИЗЪЮНКЦИЯ (ЛОГИЧЕСКОЕ СЛОЖЕНИЕ)

- соответствует союзу ИЛИ
- обозначается знаком **V** или **+** или ‖
- **Дизъюнкция двух логических переменных ложна тогда и только тогда, когда оба высказывания ложны.**
  Это определение можно обобщить для любого количества логических переменных, объединенных дизъюнкцией.

  A **V** B **V** C =0, только если A=0, B=0, C=0.
  Таблица истинности дизъюнкции имеет следующий вид:

A	B	A V B
0	0	0
0	1	1
1	0	1
1	1	1

3. Логическая операция КОНЪЮНКЦИЯ (ЛОГИЧЕСКОЕ УМНОЖЕНИЕ)

- соответствует союзу И
- обозначается знаком & или Λ, или ·
- **Конъюнкция двух логических переменных истинна тогда и только тогда, когда оба высказывания истинны.**
  Это определение можно обобщить для любого количества логических переменных, объединенных конъюнкцией.
  A & B & C=1, только если A=1, B=1, C=1.
  Таблица истинности конъюнкции имеет следующий вид:

A	B	A & B
0	0	0
0	1	0
1	0	0
1	1	1

ЛОГИЧЕСКИЕ ВЫРАЖЕНИЯ И ТАБЛИЦЫ ИСТИННОСТИ

- Сложные высказывания можно записывать в виде формул. Для этого простые логические высказывания нужно обозначить как логические переменные буквами и связать их с помощью знаков логических операций. Такие формулы называются ***логическими выражениями.*** Например:

$$(A \vee B) \& (\overline{A} \vee \overline{B})$$

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

- Чтобы определить значение логического выражения необходимо подставить значения логических переменных в выражение и выполнить логические операции. Операции в логическом выражении выполняются слева направо с учетом скобок в следующем порядке:
  1. инверсия;
  2. конъюнкция;
  3. дизъюнкция;
  4. импликация и эквивалентность.

  Для изменения указанного порядка выполнения логических операций используются круглые скобки.

Таблицы истинности

- Для каждого составного высказывания (логического выражения) можно построить *таблицу истинности*, которая определяет истинность или ложность логического выражения при всех возможных комбинациях исходных значений простых высказываний (логических переменных).
- При построении таблиц истинности целесообразно руководствоваться определенной последовательностью действий:

1) записать выражение и определить порядок выполнения операций

2) определить количество строк в таблице истинности. Оно равно количеству возможных комбинаций значений логических переменных, входящих в логическое выражение (определяется по формуле $Q=2^n$, где n - количество входных переменных)

3) определить количество столбцов в таблице истинности (= количество логических переменных + количество логических операций)

4) построить таблицу истинности, обозначить столбцы (имена переменных и обозначения логических операций в порядке их выполнения) и внести в таблицу возможные наборы значений исходных логических переменных.

5) заполнить таблицу истинности, выполняя базовые логические операции в необходимой последовательности и в соответствии с их таблицами истинности

Теперь мы можем определить значение логической функции для любого набора значений логических переменных.

Например, построим таблицу истинности для логической функции:

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

Количество входных переменных в заданном выражении равно трем *(A,B,C)*. Значит, количество входных наборов, а значит и строк $Q=2^3=8$. Количество столбцов равно 6 (3 переменные + 3 операции). Столбцы таблицы истинности соответствуют значениям исходных выражений *A,B,C*, промежуточных результатов $\overline{A}$ и (*B* V *C*), а также искомого окончательного значения сложного арифметического выражения $\overline{A} \& (B \vee C)$

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

A	B	C	$\bar{A}$	B V C	$\bar{A} \& (B \vee C)$
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

A	B	C	$\overline{A}$	B V C	$\overline{A} \& (B \vee C)$
0	0	0	1	0	0
0	0	1	1	1	1
0	1	0	1	1	1
0	1	1	1	1	1
1	0	0	0	0	0
1	0	1	0	1	0
1	1	0	0	1	0
1	1	1	0	1	0

Задание. Постройте таблицу истинности для данного логического выражения:

$$(A \vee B) \& (\overline{A} \vee B)$$

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

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

Равносильные логические выражения. Логические выражения, у которых последние столбцы таблиц истинности совпадают, называются *равносильными.* Для обозначения равносильных логических выражений используется знак **=**.

Например: $\overline{A}\,\&\,\overline{B} = \overline{A \vee B}$

Логические законы и правила преобразования логических выражений

Равносильности формул логики высказываний часто называют *законами логики*. Законы логики отражают наиболее важные закономерности логического мышления.

В алгебре высказываний законы логики записываются в виде формул, которые позволяют проводить эквивалентные преобразования логических выражений в соответствие с законами логики.

Знание законов логики позволяет проверять правильность рассуждений и доказательств. Нарушения этих законов приводят к логическим ошибкам и вытекающим из них противоречиям.

Перечислим наиболее важные из них:

1. **Закон тождества.** Всякое высказывание тождественно самому себе:

$$A = A$$

Этот закон сформулирован древнегреческим философом Аристотелем. **Закон тождества** утверждает, что мысль, заключенная в некотором высказывании, остается неизменной на протяжении всего рассуждения, в котором это высказывание фигурирует.

2. **Закон непротиворечия.** Высказывание не может быть одновременно истинным и ложным. Если высказывание ***A*** — истинно, то его отрицание ***не A*** должно быть ложным. Следовательно, логическое произведение высказывания и его отрицания должно быть ложно:

$$\overline{A} \,\&\, A = 0$$

Закон непротиворечия говорит о том, что никакое предложение не может быть истинно одновременно со своим отрицанием. *“Это яблоко спелое” и “Это яблоко не спелое”*

3. **Закон исключенного третьего.** Высказывание может быть либо истинным, либо ложным, третьего не дано. Это означает, что результат логического сложения высказывания и его отрицания всегда принимает значение истина:

$$A \vee \overline{A} = 1$$

Закон исключенного третьего говорит о том, что для каждого высказывания имеются лишь две возможности: это высказывание либо истинно, либо ложно. Третьего не дано.

“Сегодня я получу 5 либо не получу”. Истинно либо суждение, либо его отрицание.

4. **Закон двойного отрицания.** Если дважды отрицать некоторое высказывание, то в результате мы получим исходное высказывание:

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

Закон двойного отрицания. Отрицать отрицание какого-нибудь высказывания - то же, что утверждать это высказывание.

“ Неверно, что 2× 2^{1} 4”

5. Законы идемпотентности. В алгебре логики нет показателей степеней и коэффициентов.

Конъюнкция одинаковых «сомножителей» равносильна одному из них:

$$A \,\&\, A = A$$

Дизъюнкция одинаковых «слагаемых» равносильна одному:

$$A \vee A = A$$

6. Законы де Моргана:

$$\overline{A \vee B} = \overline{A} \,\&\, \overline{B} \qquad \overline{A \,\&\, B} = \overline{A} \vee \overline{B}$$

Смысл законов де Моргана (Август де Морган (1806-1871) - шотландский математик и логик) можно выразить в кратких словесных формулировках:

отрицание логической суммы эквивалентно логическому произведению отрицаний слагаемых;

отрицание логического произведения эквивалентно логической сумме отрицаний множителей.

7. **Правило коммутативности.** В обычной алгебре слагаемые и множители можно менять местами. В алгебре высказываний можно менять местами логические переменные при операциях логического умножения и логического сложения:

Логическое умножение:

$$A \& B = B \& A$$

Логическое сложение:

$$A \vee B = B \vee A$$

8. **Правило ассоциативности.** Если в логическом выражении используются только операция логического умножения или только операция логического сложения, то можно пренебрегать скобками или произвольно их расставлять:

Логическое умножение:

$$(A \& B) \& C = A \& (B \& C)$$

Логическое сложение:

$$(A \vee B) \vee C = A \vee (B \vee C)$$

9. Правило дистрибутивности. В отличие от обычной алгебры, где за скобки можно выносить только общие множители, в алгебре высказываний можно выносить за скобки, как общие множители, так и общие слагаемые:

Дистрибутивность умножения относительно сложения:

$$(A \& B) \vee (A \& C) = A \& (B \vee C)$$

Дистрибутивность сложения относительно умножения:

$$(A \vee B) \& (A \vee C) = A \vee (B \& C)$$

10. $A \& 1 = A$ $\qquad$ $A \& 0 = 0$

11. $A \vee 1 = 1$ $\qquad$ $A \vee 0 = A$

12. Законы поглощения:

$$A \& (A \vee B) = A \qquad A \vee (A \& B) = A \& B$$

ЛОГИЧЕСКИЕ ОСНОВЫ КОМПЬЮТЕРА

Логические элементы

В основе обработки компьютером информации лежит алгебра логики, разработанная Дж. Булем. Знания из области математической логики можно использовать для конструирования различных электронных устройств.

Нам известно, что 0 и 1 в логике не просто цифры, а обозначение состояний какого-то предмета нашего мира, условно называемых "ложь" и "истина". Таким предметом, имеющим два фиксированных состояния, может быть электрический ток. Были созданы устройства управления электричеством - электронные схемы, состоящие из набора полупроводниковых элементов. Такие электронные схемы, которые преобразовывают сигналы только двух фиксированных напряжений электрического тока стали называть *логическими элементами*.

Логические элементы *— это электронные устройства, которые преобразуют проходящие через них двоичные электрические сигналы по определенному закону*.

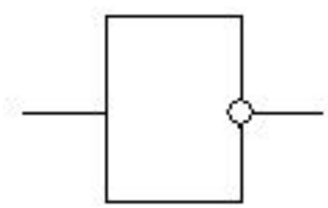
Логические элементы имеют один или несколько входов, на которые подаются электрические сигналы, обозначаемые условно **0**, если отсутствует электрический сигнал, и **1**, если имеется электрический сигнал.

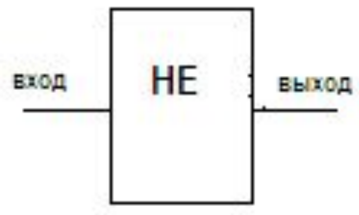
Также логические элементы имеют один выход, с которого снимается преобразованный электрический сигнал.

Было доказано, что все электронные схемы компьютера могут быть реализованы с помощью трёх базовых логических элементов **И, ИЛИ, НЕ.**

Логический элемент НЕ (инвертор)

Простейшим логическим элементом является ***инвертор***, выполняющий функцию отрицания (инверсию). У этого элемента один вход и один выход. На функциональных схемах он обозначается:



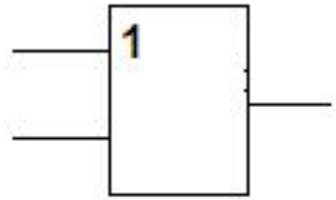


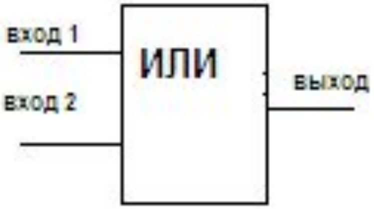
Если на вход поступает сигнал, соответствующий 1, то на выходе будет 0. И наоборот.

вход	*выход*
1	0
0	1

Логический элемент ИЛИ (дизъюнктор)

Логический элемент, выполняющий логическое сложение, называется ***дизъюнктор.*** Он имеет, как минимум, два входа. На функциональных схемах он обозначается:



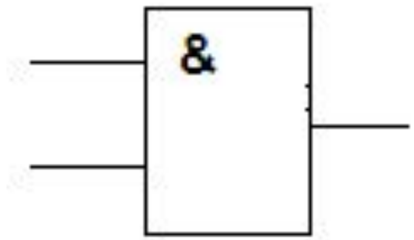


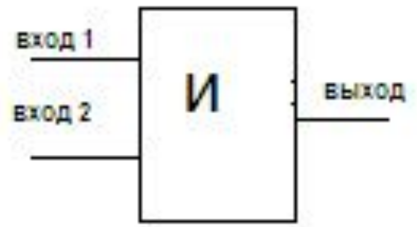
Если хотя бы на один вход поступает сигнал 1, то на выходе будет сигнал 1.

вход 1	*вход 2*	*выход*
0	**0**	**0**
0	**1**	**1**
1	**0**	**1**
1	**1**	**1**

Логический элемент И (конъюнктор)

Логический элемент, выполняющий логическое умножение, называется ***конъюнктор.*** Он имеет, как минимум, два входа. На функциональных схемах он обозначается:





На выходе этого элемента будет сигнал 1 только в том случае, когда на все входы поступает сигнал 1. Когда хотя бы на одном входе будет ноль, на выходе также будет ноль.

вход 1	*вход 2*	*выход*
0	0	0
0	1	0
1	0	0
1	1	1

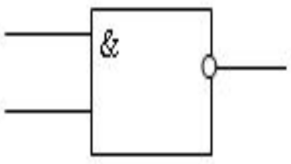
Другие логические элементы построены из трех простейших базовых элементов и выполняют более сложные логические преобразования информации.

Рассмотрим еще два логических элемента, которые играют роль базовых при создании более сложных элементов и схем.

Логический элемент И-НЕ

Логический элемент И-НЕ выполняет логическую функцию штрих Шеффера (И-НЕ), он имеет, как минимум, два входа. На функциональных схемах он обозначается:

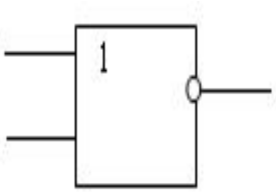
вход 1	*вход 2*	*выход*
0	**0**	**1**
0	**1**	**1**
1	**0**	**1**
1	**1**	**0**



Логический элемент ИЛИ-НЕ

Логический элемент ИЛИ-НЕ выполняет логическую функцию стрелка Пирса (И-НЕ), он имеет, как минимум, два входа. На функциональных схемах он обозначается:

вход 1	*вход 2*	*выход*
0	**0**	**1**
0	**1**	**0**
1	**0**	**0**
1	**1**	**0**

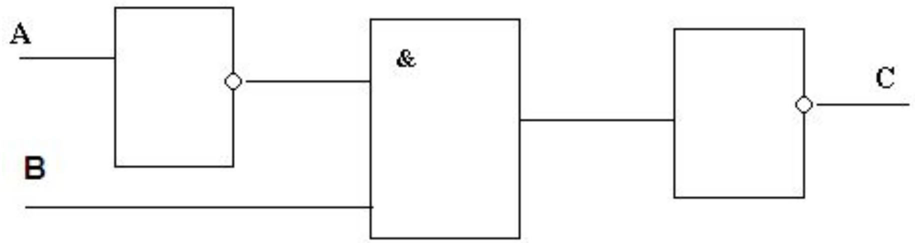


Функциональные схемы

Сигнал, выработанный одним логическим элементом, можно подавать на вход другого элемента, это дает возможность образовывать цепочки из отдельных логических элементов — *функциональные схемы.*

Функциональная (логическая) схема – это схема, состоящая из логических элементов, которая выполняет определённую функцию. Анализируя функциональную схему, можно понять, как работает логическое устройство, т.е. дать ответ на вопрос: какую функцию она выполняет.

Важной формой описания функциональных схем является структурная формула. Покажем на примере, как выписывают формулу по заданной функциональной схеме.



Ясно, что элемент “И” осуществляет логическое умножение значений ¬A и B. Над результатом в элементе “НЕ” осуществляется операция отрицания, т.е. вычисляется значение выражения:

$$\overline{\overline{A} \,\&\, B}$$

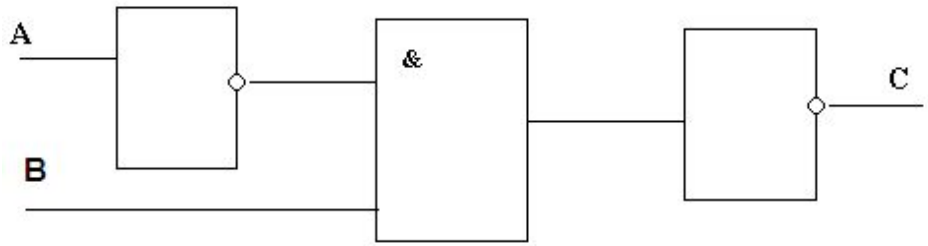
Таким образом структурной формулой данной функциональной схемы является формула:

$$C = \overline{\overline{A} \,\&\, B}$$

Таблица истинности функциональной схемы

Для функциональной схемы можно составить таблицу истинности, то есть таблицу значений сигналов на входах и выходах схемы, по которой можно понять какую функцию выполняет данная схема. ***Таблица истинности -*** это табличное представление логической (функциональной) схемы в котором перечислены все возможные сочетания значений входных сигналов вместе со значением выходного сигнала для каждого из этих сочетаний.

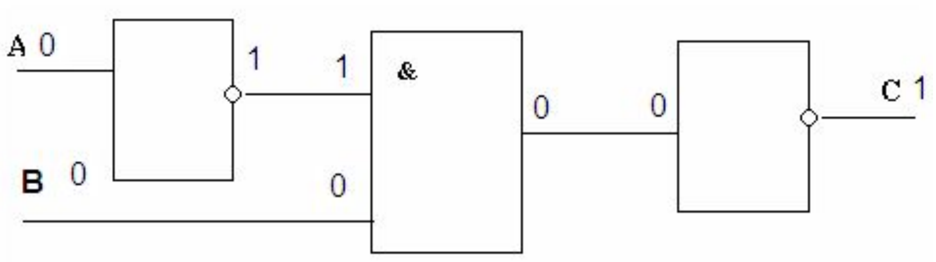
Составим таблицу истинности для данной логической схемы:



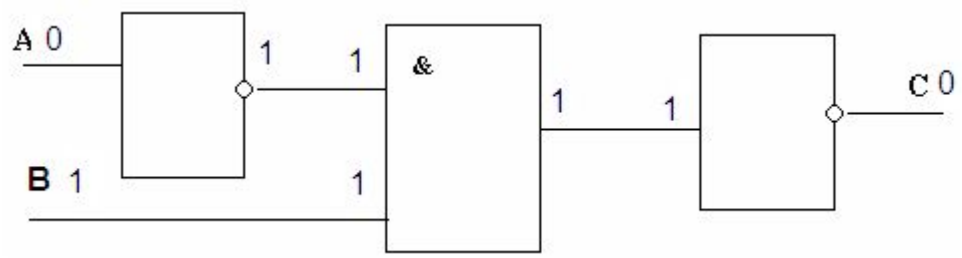
Начертим таблицу: количество столбцов = количество входов + количество выходов, количество строк = 2 $^{\text{количество входов}}$. В данной таблице 3 столбца и 4 строки. Заполним первые столбцы всеми возможными вариантами входных сигналов

A (вход 1)	B (вход 2)	C (выход)
0	0	
0	1	
1	0	
1	1	

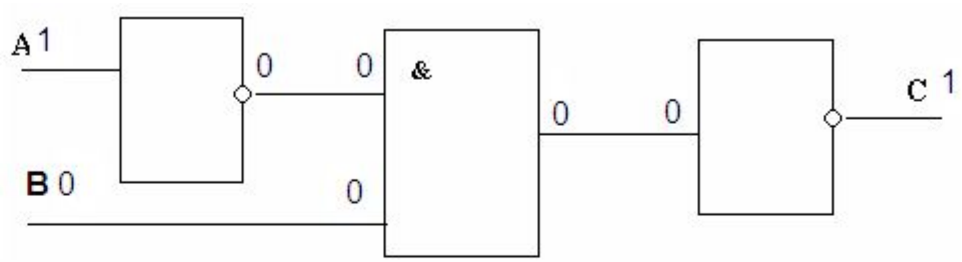
Рассмотрим первый вариант входных сигналов: A=0, B=0. Проследим по схеме, как проходят и преобразуются входные сигналы. Результат, полученный на выходе (C=1), запишем в таблицу.



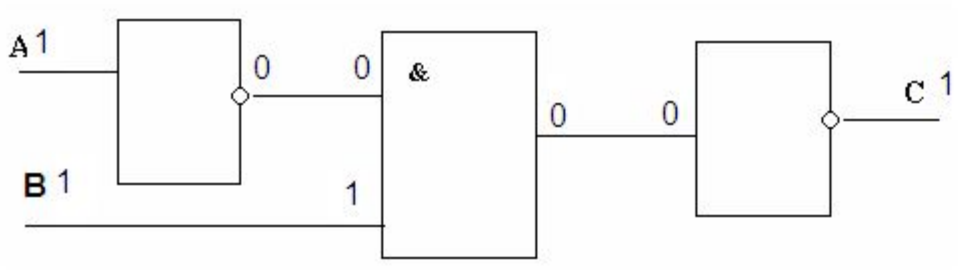
Рассмотрим второй вариант входных сигналов: A=0, B=1. Проследим по схеме, как проходят и преобразуются входные сигналы. Результат, полученный на выходе (C=0), запишем в таблицу.



Рассмотрим третий вариант входных сигналов: A=1, B=0. Проследим по схеме, как проходят и преобразуются входные сигналы. Результат, полученный на выходе (C=1), запишем в таблицу.



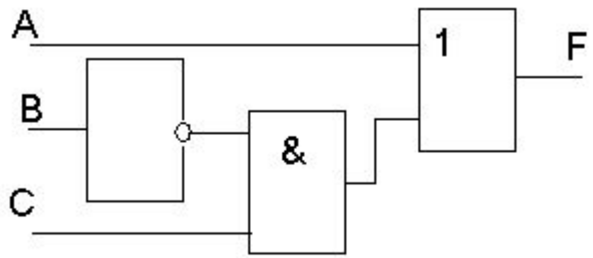
Рассмотрим четвёртый вариант входных сигналов: A=1, B=1. Проследим по схеме, как проходят и преобразуются входные сигналы. Результат, полученный на выходе (C=1), запишем в таблицу.



В результате получаем таблицу истинности данной логической схемы:

A (вход 1)	**B (вход 2)**	**C (выход)**
0	**0**	**1**
0	**1**	**0**
1	**0**	**1**
1	**1**	**1**

Задание. Построить таблицу истинности для данной логической схемы и записать формулу для данной схемы:



Логическая реализация типовых устройств компьютера

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

Выясним , как из логических элементов разрабатываются логические устройства.

Этапы конструирования логического устройства.

Конструирование логического устройства состоит из следующих этапов:

1. Построение таблицы истинности по заданным условиям работы проектируемого узла (т.е. по соответствию его входных и выходных сигналов).

2. Конструирование логической функции данного узла по таблице истинности, ее преобразование (упрощение), если это возможно и необходимо.

3. Составление функциональной схемы проектируемого узла по формуле логической функции.

После этого остается только реализовать полученную схему.

Задание. Построить логическую схему для заданной таблицы истинности:

Запишем логическую функцию по данной таблице истинности:

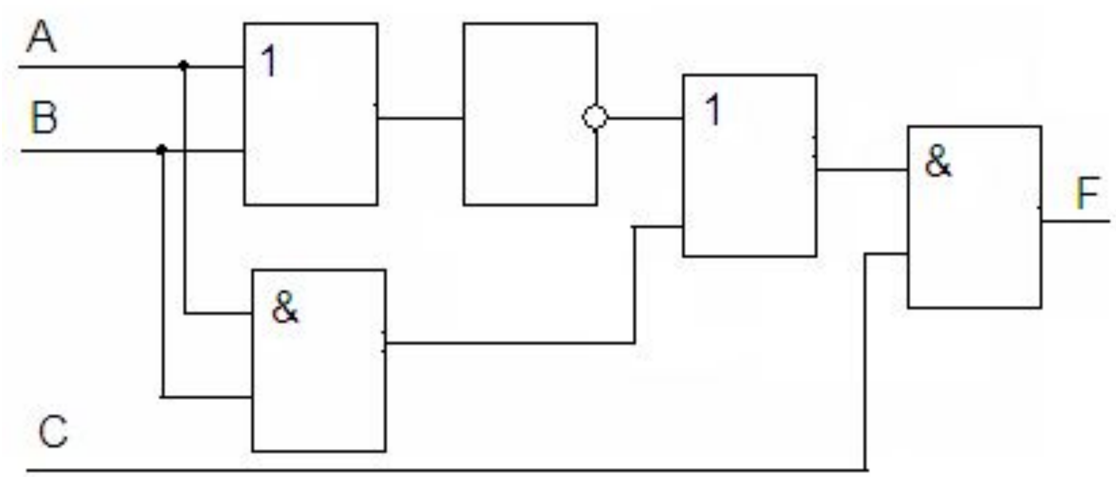
$$F = \overline{A} \& \overline{B} \& C \vee A \& B \& C$$

Упростим полученное логическое выражение:

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

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

Построим логическую схему для данного выражения:



Попробуем, действуя по этому плану, сконструировать устройство для сложения двух двоичных чисел ***(одноразрядный полусумматор).***

Пусть нам необходимо сложить двоичные числа **A** и **B**. Через **P** и **S** обозначим первую и вторую цифру суммы: **A + B = PS**. Вспомните таблицу сложения двоичных чисел.

1. Таблица истинности, определяющая результат сложения, имеет вид:

Слагаемые		Перенос	Сумма
A	**B**	**P**	**S**
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

2. Сконструируем функции *P(A,B)* и *S(A,B)* по этой таблице:

$$P(A,B) = A \,\&\, B \qquad S(A,B) = \overline{A} \,\&\, B \vee A \,\&\, \overline{B}$$

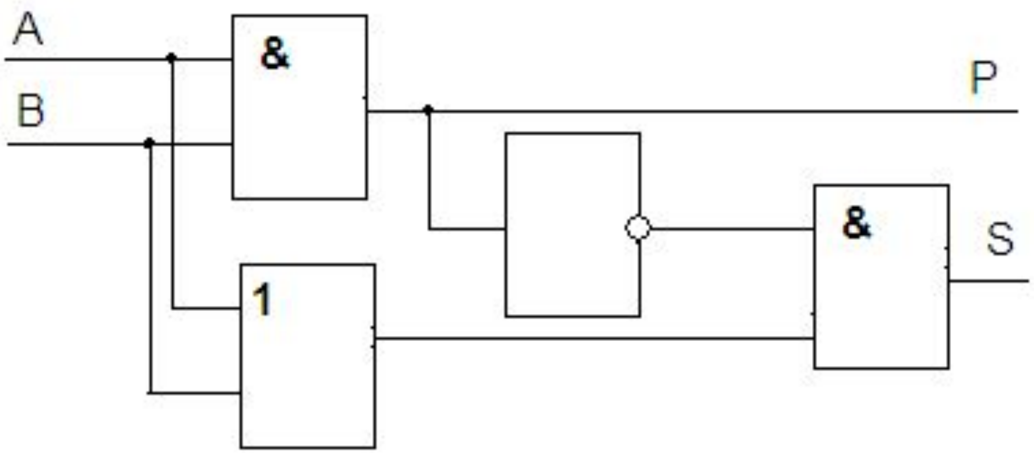
Преобразуем вторую формулу, пользуясь законами логики:

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

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

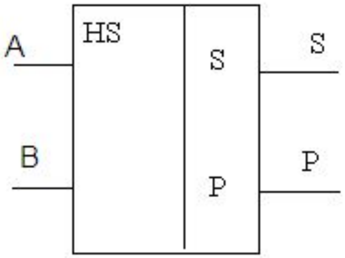
3. Теперь можно построить функциональную схему одноразрядного полусумматора:

$$P(A,B) = A \,\&\, B \qquad S(A,B) = (A \vee B) \,\&\, \overline{(A \,\&\, B)}$$



Чтобы убедиться в том, как работает схема, проследите за прохождением сигналов в каждом из четырёх случаев и составьте таблицу истинности данной логической схемы.

Условное обозначение одноразрядного сумматора:



Полный одноразрядный сумматор.

Одноразрядный двоичный сумматор на три входа и два выхода называется *полным одноразрядным сумматором*.

Логика работы одноразрядного сумматора на три входа или полного сумматора приведена в таблице, где **A, B** - суммируемые двоичные цифры , **Po** - перенос из младшего разряда, ***S*** - образующаяся сумма данного разряда и осуществляет перенос **P** в следующий старший разряд.

Слагаемые		**Перенос из младшего разряда**	**Сумма**	**Перенос**
A	**B**	**P_0**	**S**	**P**
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

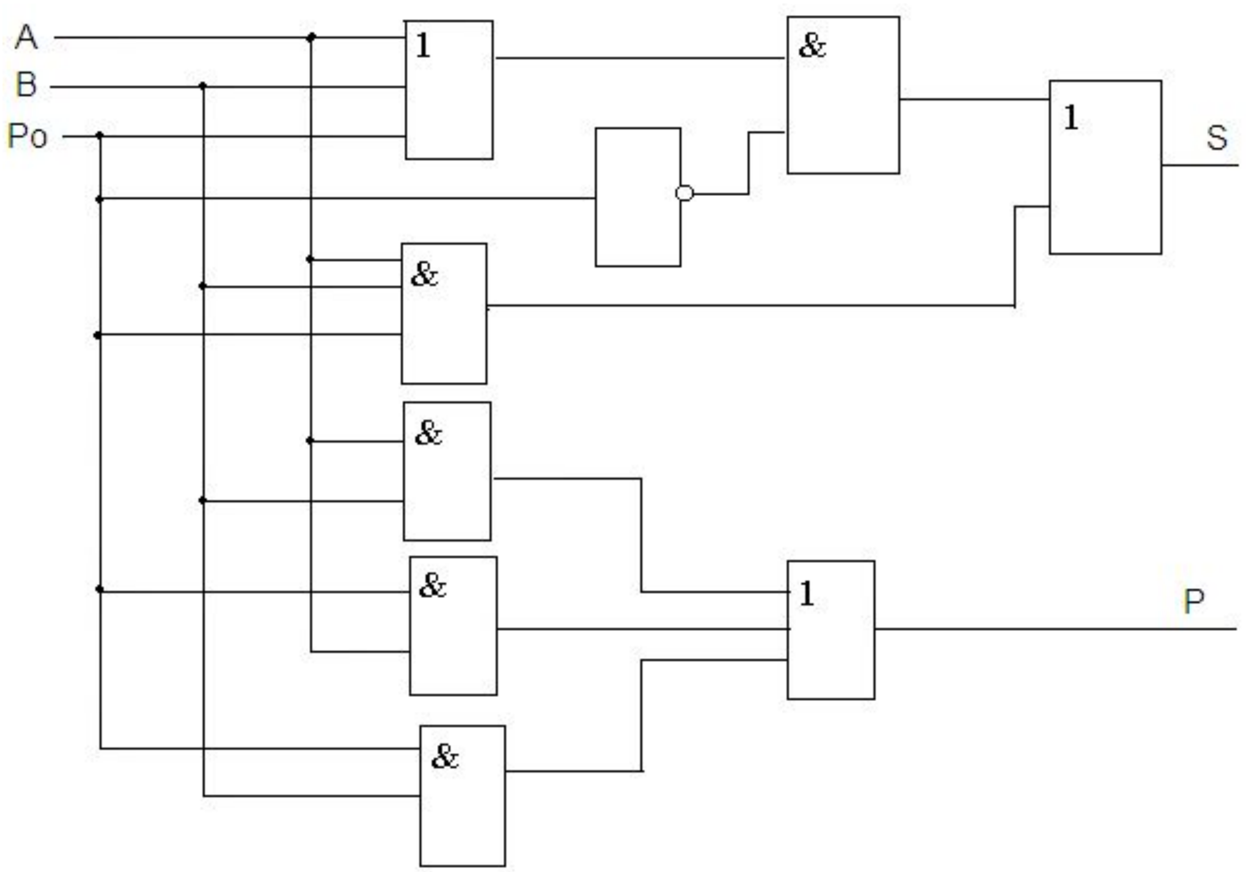
Формула переноса: $P = A \& B \& \overline{P}_0 \vee \overline{A} \& B \& P_0 \vee A \& \overline{B} \& P_0 \vee A \& B \& P_0$

Формула для вычисления суммы: $S = \overline{A} \& B \& \overline{P}_0 \vee A \& \overline{B} \& \overline{P}_0 \vee \overline{A} \& \overline{B} \& P_0 \vee A \& B \& P_0$

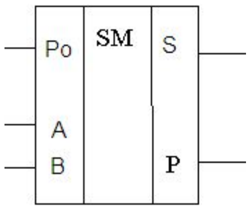
После преобразования формулы переноса и суммы принимают вид:

$$P = A \& B \vee A \& P_0 \vee B \& P_0 \qquad S = (A \vee B \vee P_0) \& \overline{P_0} \vee (A \& B \& P_0)$$

Теперь можно построить схему полного одноразрядного сумматора с учётом переноса из младшего разряда.

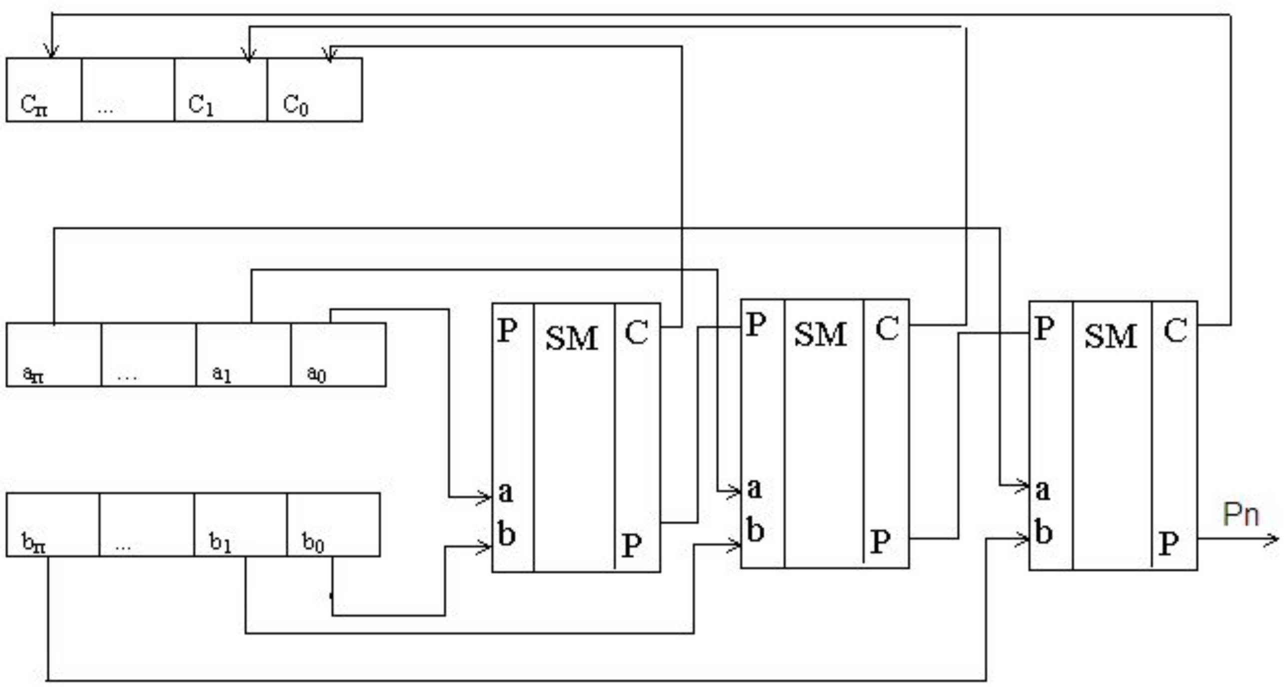


Сумматор - это электронная логическая схема, выполняющая суммирование двоичных чисел поразрядным сложением. Сумматор является центральным узлом арифметико-логического устройства процессора. Находит он применение и в других устройствах компьютера. В реальных электронных схемах сумматор изображается так:



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

На рисунке показано, как из N сумматоров можно составить устройство для сложения двух N-разрядных двоичных кодов, это схема многоразрядного сумматора.



ТРИГГЕР

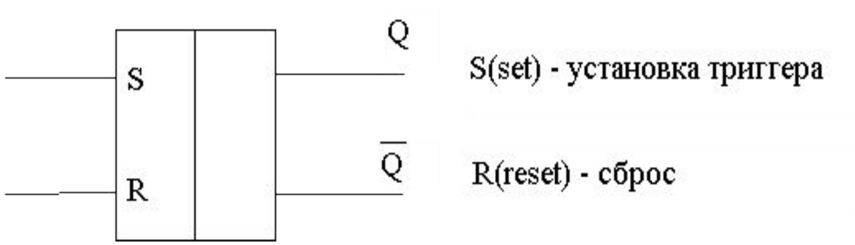
Триггер - электронная схема, применяемая для хранения значения одноразрядного двоичного кода.

Воздействуя на входы триггера, его переводят в одно из двух возможных состояний (0 или 1). С поступлением сигналов на входы триггера в зависимости от его состояния либо происходит переключение, либо исходное состояние сохраняется. При отсутствии входных сигналов триггер сохраняет свое состояние сколь угодно долго.

Термин *триггер* происходит от английского слова *trigger* - защёлка, спусковой крючок. Для обозначения этой схемы в английском языке чаще употребляется термин *flip-flop*, что в переводе означает "хлопанье". Это звукоподражательное название электронной схемы указывает на её способность почти мгновенно переходить ("перебрасываться") из одного электрического состояния в другое.

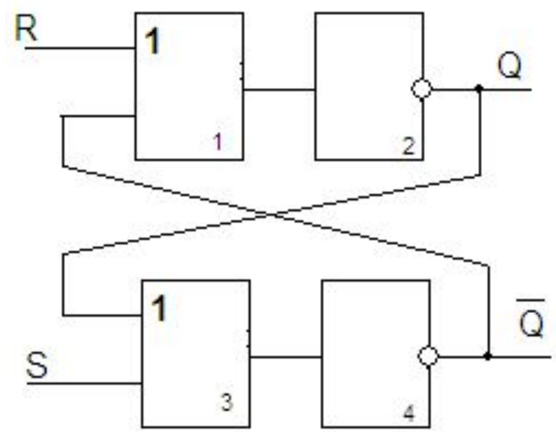
Существуют разные варианты исполнения триггеров в зависимости от элементной базы (И-НЕ, ИЛИ-НЕ) и функциональных связей между сигналами на входах и выходах (*RS, JK, T, D* и другие).

Самый распространённый тип триггера - это *RS*-триггер (*S* и *R* соответственно от английских *set* - установка, и *reset* - сброс). Условное обозначение *RS*-триггера:



RS-триггер

RS-триггер построен на 2-х логических элементах: ИЛИ - НЕ либо И – НЕ. Как, правило, триггер имеет 2 выхода: прямой и инверсный ($\overline{Q}$)

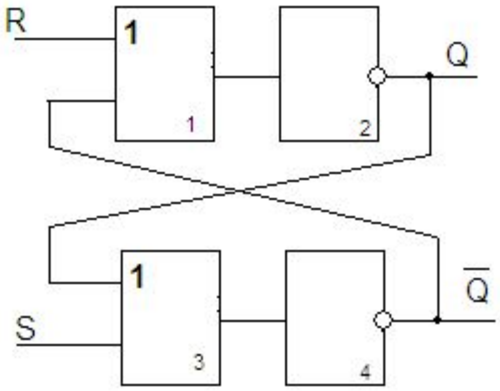


Как он работает?

Пусть на вход элемента №1 подан сигнал 1, а на вход элемента № 3 - 0. На выходе элемента №1 независимо от того, какой второй сигнал поступит на вход, будет 1, т.к. это элемент ИЛИ (по свойствам дизъюнкции). Пройдя через элемент № 2 сигнал примет значение 0 (Q=0). Следовательно, и на втором входе элемента № 3 установится сигнал 0. На выходе элемента №3 - 0. Пройдя через элемент № 4 сигнал изменится на 1. Следовательно, $\overline{Q}$= 1.

Убедимся, что данное устройство сохраняет информацию. Запомните, что S=0, R=1, Q=0, $\overline{Q}$=1. В момент прекращения входных сигналов (S=0, R=0) на выходе =1. Это напряжение подается на вход элемента № 1. На выходе элемента №1 сохраняется 1, и на Q - сигнал 0. На входах элемента №3 - 0, следовательно $\overline{Q}$ =1. Таким образом, при отсутствии на внешних входах сигналов 1 триггер поддерживает постоянное напряжение на своих выходах. Чтобы изменить напряжение на выходах триггера, надо подать сигнал 1 на вход элемента № 3. Тогда Q=1, $\overline{Q}$ =0.

RS-триггер



Вход		Выход		Режим работы
S	R	Q	$\overline{Q}$	
0	0	0	0	Хранение
1	0	1	0	Запись 1
0	1	0	1	Запись 0
1	1	X	X	Запрещение ($Q \neq \overline{Q}$)

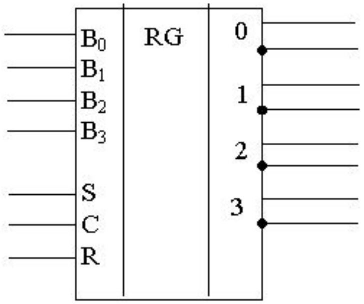
РЕГИСТРЫ

Функциональная схема компьютера, состоящая из триггеров, предназначенная для запоминания многоразрядных кодов и выполнения над ними некоторых логических преобразований называется ***регистром.***

Упрощенно регистр можно представить как совокупность ячеек, в каждой из которых может быть записано одно из двух значений: 0 или 1, то есть один разряд двоичного числа.

С помощью регистров можно выполнять следующие операции: установку, сдвиг, преобразование. Основными типами регистров являются параллельные и последовательные (сдвигающие).

Совокупность регистров, используемых ЭВМ для запоминания программы работы, исходных и промежуточных результатов называется оперативной памятью (ОП).



Регистры содержатся в различных вычислительных узлах компьютера - процессоре, периферийных устройствах и т.д.

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

РЕГИСТРЫ

Существует несколько типов регистров, отличающихся видом выполняемых операций.

Некоторые важные регистры имеют свои названия, например:

сдвиговый регистр - предназначен для выполнения операции сдвига;

счетчики - схемы, способные считать поступающие на вход импульсы. К ним относятся *T*-триггеры (название от англ. *tumble* - опрокидываться). Этот триггер имеет один счетный вход и два выхода. Под действием сигналов триггер меняет свое состояние с нулевого на единичное и наоборот. Число перебрасываний соответствует числу поступивших сигналов;

счетчик команд - регистр устройства управления процессора (УУ), содержимое которого соответствует адресу очередной выполняемой команды; служит для автоматической выборки программы из последовательных ячеек памяти;

регистр команд - регистр УУ для хранения кода команды на период времени, необходимый для ее выполнения. Часть его разрядов используется для хранения кода операции, остальные - для хранения кодов адресов операндов.

В ЭВМ применяются регистры 8, 16, 32, 48 и 64 разрядов.