Информатика

Курс лекций часть 4

Масловский Владимир Михайлович, к.т.н., доцент кафедра ИУ-10
РУНЦ «Безопасность» МГТУ им. Р.Э. Баумана, тел. 499 263 6794,
E-mail: zi@bmstu.ru, mvm481@rambler.ru

Логические основы построения цифровых автоматов

1. Основные законы и постулаты цифровых автоматов
2. Представление функций алгебры логики
3. Логический синтез переключательных и вычислительных схем
3.1. Синтез переключательных схем
3.2. Синтез вычислительных схем
4. Основы элементной базы цифровых автоматов
4.1.Логические элементы
4.2 Схемотехника логических элементов
4.3 Элементы интегральных схем

Контрольные вопросы

Логические основы построения цифровых автоматов

Слово *логика* означает совокупность правил, которым подчиняется процесс мышления.

Сам термин "логика" происходит от древнегреческого logos, означающего "слово, мысль, понятие, рассуждение, закон". *Формальная логика* - наука о формах и законах мышления. Законы логики отражают в сознании человека свойства, связи и отношения объектов окружающего мира. Логика как наука позволяет строить формальные модели окружающего мира, отвлекаясь от содержательной стороны. Основными формами мышления являются *понятия, суждения* и *умозаключения*.

Понятие - это форма мышления, которая выделяет существенные признаки предмета или класса предметов, отличающие его от других. Например, компьютер, человек, ученики.

Суждения - это форма мышления, в которой утверждается или отрицается связь между предметом и его признаком, отношения между предметами или факт существования предмета и которая может быть либо истинной, либо ложной. Языковой формой выражения суждения является повествовательное предложение. Вопросительные и побудительные предложения суждениями не являются

Логические основы построения цифровых автоматов

Суждения рассматриваются не с точки зрения их смысла и содержания, а только с точки зрения их истинности или ложности. Истинным будет суждение, в котором связь понятий правильно отражает свойства и отношения реальных объектов. "Дважды два равно четырем" - истинное суждение, а вот "Процессор предназначен для печати" - ложное. Суждения могут быть простыми и сложными. "Весна наступила, и грачи прилетели" - сложное суждение, состоящее из двух простых. Простые суждения (высказывания) выражают связь двух понятий. Сложные - состоят из нескольких простых суждений.

Умозаключение - прием мышления, позволяющий на основе одного или нескольких суждений-посылок получить новое суждение (знание или вывод).

Примерами умозаключений являются доказательства теорем в геометрии. Посылками умозаключения по правилам формальной логики могут быть только истинные суждения. Тогда и умозаключение будет истинным. Иначе можно прийти к ложному умозаключению

Алгебра логики. История логики

Алгебра — раздел математики, исследующий операции, аналогичные *сложению, умножению, вычитанию* и *делению* и выполняемые не только над числами, но и над другими математическими объектами, например, многочленами, векторами, матрицами, операторами и т.д., над объектами самой различной природы.

Возникла алгебра в связи с поисками общих приемов решения однотипных арифметических задач. В основе найденных алгеброй общих приемов лежат действия над величинами (составление и решение уравнений), выраженных буквами, независимо от их конкретного числового значения. Введение символики имело исключительное значение и явилось огромным шагом вперед в развитии математики, так как введение буквенных обозначений сделало запись сжатой и удобной для построения исчислений. Применение буквенных обозначений облегчило и исследование общих свойств числовых систем и общих методов решения задач при помощи уравнений.

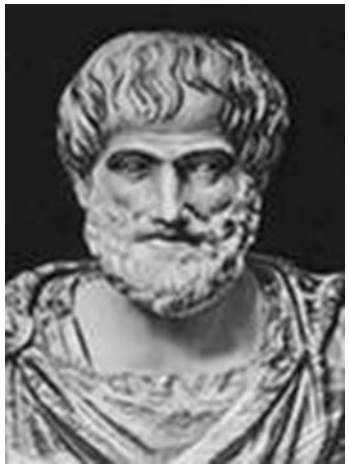
Логические основы построения цифровых автоматов

Как грамматика изучает формы отдельного слова и формы сочетания слов в предложении, отвлекаясь от конкретного содержания языковых выражений; как математика рассматривает количественные и пространственные отношения и формы, отвлекаясь от конкретных материальных предметов, так и *формальная логика* исследует формы отдельных мыслей и формы сочетаний их в отвлечении от конкретного содержания суждений, умозаключений, доказательств и понятий. Составной частью формальной логики является *математическая логика.*

Математическая логика изучает вопросы применения математических методов для решения логических задач и построения логических схем, которые лежат в основе работы любого компьютера. Суждения в математической логике называют *высказываниями* или *логическими выражениями*. Подобно тому, как для описания действий над переменными был разработан раздел математики алгебра, так и для обработки логических выражений в математической логике была создана *алгебра высказываний*, или *алгебра логики*.

Логические основы построения цифровых автоматов

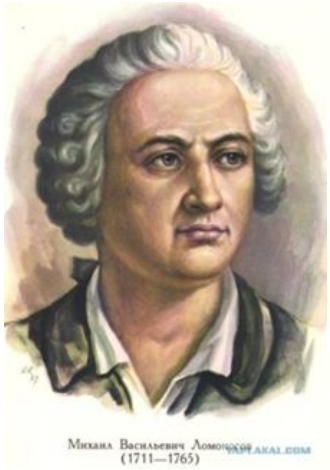
Зародилась логика в лоне единой нерасчлененной науки — античной философии, которая тогда объединяла всю совокупность знаний о мире и о самом человеке и его мышлении. В IV в. до н. э. логика начинает развиваться под влиянием возросшего интереса к ораторскому искусству. Это характерно не только для Древней Греции, но и для Древней Индии, Древнего Китая, Древнего Рима и феодальной России.



Как известно, в первом сочинении *Аристотеля* (384 — 322 до н. э.) по логике проблемы логики рассматривались в связи с теорией ораторского искусства.

Декарт Рене (1596-1650, фр. философ, математик) предложил в логике использовать математические методы.

Логические основы построения цифровых автоматов



Первый русский фундаментальный труд по логике, написанный *М.В. Ломоносовым* (1711 — 1765), называется «Краткое руководство к красноречию».

Основы математической логики заложил немецкий ученый и философ *Готфрид Вильгельм Лейбниц* (1646 — 1716). Он сделал попытку построить первые логические исчисления, считал, что можно заменить простые рассуждения действиями со знаками и привел соответствующие правила.

Но Лейбниц высказал только идею, а развил ее окончательно англичанин *Джордж Буль* (1815 — 1864). Буль считается основоположником математической логики как самостоятельной дисциплины. В его работах логика обрела свой алфавит, свою орфографию и грамматику. Недаром начальный раздел математической логики называют *алгеброй логики,* или *булевой алгеброй.*

Алгебра логики – раздел математики, изучающий высказывания, рассматриваемые со стороны их логических значений (истинности или ложности) и логических операций над ними.

Приведем еще одно определение алгебры логики, устанавливающее связь с высказыванием.

Алгебра логики (логика высказываний) — один из основных разделов математической логики, в котором методы алгебры используются в логических преобразованиях *высказываний*.

Попробуем разобраться что же такое логическое высказывание?

Логическое высказывание — это любое повествовательное предложение, в отношении которого можно однозначно сказать, истинно оно или ложно.

Или же

Высказывания

Высказывание — это термин математической логики, которым обозначается предложение какого-либо языка (естественного или искусственного), рассматриваемого лишь в связи с его истинностью. Например: «Земля — планета солнечной системы.»	Истина
«2+8<5»	Ложь
«Всякий квадрат есть параллелограмм.»	Истина
«Каждый параллелограмм есть квадрат.»	Ложь

Логические основы построения цифровых автоматов

ОСНОВНЫЕ ЗАКОНЫ АЛГЕБРЫ ЛОГИКИ

В алгебре логики выполняются следующие основные законы, позволяющие производить *тождественные преобразования логических выражений*:

Закон	Для ИЛИ	Для И
Переместительный	$x \vee y = y \vee x$	$x \cdot y = y \cdot x$
Сочетательный	$x \vee (y \vee z) = (x \vee y) \vee z$	$x \cdot (y \cdot z) = (x \cdot y) \cdot z$
Распределительный	$x \cdot (y \vee z) = x \cdot y \vee x \cdot z$	$x \vee (y \cdot z) = (x \vee y) \cdot (x \vee z)$
Правила де Моргана	$\overline{x \vee y} = \overline{y} \cdot \overline{x}$	$\overline{x \cdot y} = \overline{y} \vee \overline{x}$
Идемпотенции	$x \vee x = x$	$x \cdot x = x$
Поглощения	$x \vee (x \cdot y) = x$	$x \cdot (x \vee y) = x$
Склеивания	$(x \cdot y) \vee (\overline{x} \cdot y) = y$	$(x \vee y) \cdot (\overline{x} \vee y) = y$
Операция переменной с ее инверсией	$x \vee \overline{x} = 1$	$x \cdot \overline{x} = 0$
Операция с константами	$x \vee 0 = x;\ x \vee 1 = 1$	$x \cdot 1 = x;\ x \cdot 0 = 0$
Двойного отрицания	$\overline{\overline{x}} = x$	

Рассмотрим основные законы алгебры логики.

1. Переместительный закон (закон коммутативности) для логического умножения и логического сложения: $a \cdot b = b \cdot a;\ a+b = b+a.$

2. Сочетательный закон (закон ассоциативности) для логического умножения и логического сложения: $(a \cdot b) \cdot c = a \cdot (b \cdot c);\ (a+b)+c = a+(b+c).$

3. Распределительный закон (дистрибутивный закон): $a+b \cdot c=(a+b)(a+c);$

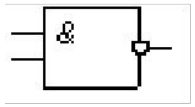
$(b+c) \cdot a=a \cdot b+a \cdot c;$

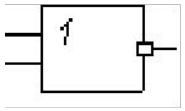
$a+b \cdot c=(a \cdot 1+b \cdot c)=a \cdot (1+b+c)+b \cdot c = a \cdot a+a \cdot b+a \cdot c+b \cdot c=a \cdot (a+b)+c \cdot (a+b)=$

$= (a+b) \cdot (a+c).$

4. Закон поглощения: $a+a \cdot b=a \cdot (1+b)=a;\ a \cdot (a+b)=a \cdot a+a \cdot b=a.$

5. Закон склеивания: $a \cdot b = a \cdot \bar{b}$; $(a+b) \cdot (a+\bar{b}) = a$; $a + \bar{a} \cdot b = a + b$; $a \cdot (\bar{a} + b) = a \cdot b$.





6. Закон отрицания (закон двойственности, закон де Моргана): $\overline{a+b} = \bar{a} \cdot \bar{b}$;

$\overline{a \cdot b} = \bar{a} + \bar{b}$; $\overline{a+b} = a \downarrow b$ - «стрелка Пирса» (функция «Вебба») ,

$a \cdot b = a / b$ - «штрих Шеффера», .

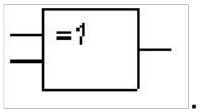
7. Закон двойного отрицания: $a = \overline{\overline{a}}$.

8. Закон умножения на 1 и 0 : $a \cdot 1 = a$; $a \cdot 0 = 0$; $a \cdot a = a$.

9. Закон сложения с 1 и 0: $a + 1 = 1$; $a + 0 = a$; $a + a = a$.

10. Закон исключения и противоречия: $a + \overline{a} = 1$; $a \cdot \overline{a} = 0$.

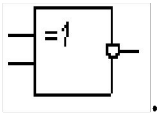
11. Константы: $\overline{1} = 0$; $\overline{0} = 1$.

12. Операция «Исключающее ИЛИ» (неравнозначность, сумма по модулю 2): $a \oplus b = a \cdot \overline{b} + \overline{a} \cdot b$ 

.

13. Операция сравнения (равнозначность, эквивалентность):

$$\overline{a \oplus b} = \bar{a} \cdot \bar{b} + a \cdot b$$



.

Законы 1 – 10 свидетельствуют о том, что алгебра логики обладает свойством двойственности (дуальности) относительно операций логического сложения и умножения. Двойственность определяется как изменение всех знаков операций И на знаки операций ИЛИ или всех знаков операций ИЛИ на знаки операций И.

Логическую функцию для удобства записи и последующего синтеза выражают в виде суммы произведений переменных или в виде произведений их сумм. Первая запись называется дизъюнктивной нормальной формой (ДНФ), вторая - конъюнктивной нормальной формой (КНФ).

Дизъюнктивная нормальная форма это запись логической функции в виде суммы произведений переменных.

Конъюнктивная нормальная форма это запись логической функции в виде произведения сумм переменных.

Для каждой логической функции может существовать несколько равносильных дизъюнктивных и конъюнктивных форм, однако существует только один вид ДНФ или КНФ, в котором функция может быть записана единственным образом (совершенные нормальные формы СДНФ и СКНФ). В СДНФ функция записывается в виде логической суммы конституент единицы (минтермов), а в СКНФ - в виде логического произведения конституент нуля (макстермов).

Конституенты единицы и нуля - это комбинации переменных, при которых функция соответственно обращается в единицу или нуль.

Минтермом (или элементарной конъюнкцией Q_i, или конституентой единицы) называется логическое произведение прямых или инверсных переменных, причем каждая переменная в произведении встречается только один раз: $Q_i = a \cdot b \cdot \bar{c} \ldots n = 1$

Макстермом (или элементарной дизъюнкцией D_i, или конституентой нуля) называется логическая сумма прямых или инверсных переменных, причем каждая переменная встречается в сумме только один раз: $D_i = (a + b + \bar{c} \ldots \bar{n}) = 0$

Количество минтермов и макстермов заданного числа аргументов совпадает с числом различных наборов аргументов $N = 2^n$.

1. Между индексами i одноименных минтермов и макстермов булевых *n* переменных существуют следующие соотношения: $\overline{Q}_i = D_{2^n-1-i}$ $\overline{D}_i = Q_{2^n-1-i}$

где индекс *i* – десятичное число и соответствует двоичному коду, отвечающему комбинации значений аргументов функции.

2. Логическая сумма всех минтермов любого числа переменных равна единице, т.е.

$$\sum_{i=0}^{2^n-1} Q_i = 1$$

3. Логическое произведение всех макстермов любого числа переменных равно нулю, т.е.

$$\prod_{i=0}^{2^n-1} D_i = 0$$

4. Логическое произведение минтермов, имеющих разные индексы, равно нулю, т.е: $Q_i \cdot Q_j = 0$, при $i \neq j$.

5. Логическая сумма неодинаковых макстермов равна единице, т. е. $D_i + D_j = 1$, при $i \neq j$.

Для построения СДНФ логической функции $F_{СДНФ}$ от n переменных, заданной таблицей истинности, необходимо по каждому набору переменных, на котором функция принимает значение 1, записать конъюнкцию – минтерм вида $Q_i = a \cdot \bar{b} \cdot c, \ldots n$

и все такие конъюнкции соединить знаками дизъюнкции. При этом переменные, имеющие значение нуля, инвертируются

$$a = 0 \rightarrow \bar{a} \quad a = 1 \rightarrow a \qquad F_{СДНФ} = \sum_{i=0}^{2^n - 1} F_i \cdot Q_i$$

где i – десятичные числа, соответствующие тем наборам аргументов, на которых $F = F_i = 1$.

Для построения СКНФ логической функции $F_{СКНФ}$ от n переменных, заданной таблицей истинности, необходимо по каждому набору переменных, на котором функция принимает значение 0, записать дизъюнкцию – макстерм вида $D_i = a + b + c + \ldots + n$

и все такие дизъюнкции соединить знаками конъюнкции. При этом переменные, имеющие значение единицы, инвертируются:

$$a = 1 \rightarrow \bar{a},\ a = 0 \rightarrow a \qquad F_{СКНФ} = \prod_{i=0}^{2^n-1} (F_i + D_i)$$

где i – десятичные числа, соответствующие тем наборам аргументов, на которых $F=F_i=0$.

Для уменьшения числа логических элементов, реализующих функцию, применяются различные методы минимизации. Под минимизацией логической функции понимают нахождение наиболее простого ее представления в виде суперпозиции операций, составляющих какую-либо фиксированную, функционально полную систему.

Логическая функция может быть упрощена непосредственно алгебраическим преобразованием с помощью законов алгебры логики (склеивания и поглощения). Но, как правило, такие преобразования требуют громоздких выкладок, а также знаний и навыков. Для функций, имеющих большое число переменных (больше трех) и большое число слагаемых, существуют специальные методы. Наиболее часто применяют методы с использованием карт Вейча и карт Карно, представляющим собой прямоугольную таблицу с числом клеток 2^n. Каждой строке (столбцу) этой таблицы соответствует определенная комбинация аргументов (переменных), каждая клетка соответствует определенному набору значений аргументов так, что при всяком переходе из одной клетки в соседнюю вдоль строки или столбца изменяется значение лишь одного аргумента функции. Карты Карно отличаются от диаграмм Вейча порядком расположения аргументов, перечисляемых в циклическом коде (коде Грея) двоичных чисел.

Алгоритм минимизации логической функции состоит из следующих этапов:

1. Логическую функцию следует привести к СДНФ (СКНФ). Для этого необходимо воспользоваться законами алгебры логики 2, 3, 7-10, а в каждый из членов функции, в которых отсутствуют аргументы, ввести выражение вида $x_i + \overline{x_i} = 1$

и воспользоваться равенством $y\left(x_i + \overline{x_i}\right) = yx_i + y\,\overline{x_i}$ для СДНФ

$x_i\,\overline{x_i} = 0$, $x_i\,\overline{x_i} = \left(y + x_i\right)\left(y + \overline{x_i}\right)$ для СКНФ),

где i - отсутствующий в члене аргумент.

2. Заполнить минтермами (макстермами) прямоугольную таблицу, количество клеток которой равно числу возможных минтермов 2^n. Минтермы логической функции отмечаются 1 в соответствующих клетках таблицы, а макстермы - 0. Два минтерма, находящиеся в соседних клетках, могут быть заменены одним логическим произведением, содержащим на одну переменную меньше, на основании законов дистрибутивности, склеивания и сложения (умножения) с 1 и 0. В общем случае наличие единиц (нулей) в 2^K соседних клетках позволяет исключить K переменных.

3. В заполненной таблице обводят прямоугольными контурами все единицы (нули). При проведении контуров придерживаются следующих правил: контур должен быть прямоугольным; внутри контура должны быть только клетки, заполненные 1 (0); количество клеток находящихся внутри контура, должно быть целой степенью числа 2, т.е. 1, 2, 4, 8, 16; одни и те же клетки, заполненные 1 (0), могут входить в несколько контуров; самая нижняя и самая верхняя строки таблицы, а также самый левый и самый правый столбцы считаются соседними; контуров должно быть как можно меньше, а сами контуры - как можно большими.

4. Выражение минимальной ДНФ (МДНФ) функции записывается следующим образом. Каждый контур в МДНФ представляется членом, число переменных в котором на K меньше общего n – числа аргументов функции, т.е. равно $n – K$. Каждый член МДНФ составляет лишь из тех аргументов, которые для соответствующего контура являются общими, т.е. имеют значения без инверсии либо с инверсией. В общем случае функция может иметь несколько минимальных форм, которым соответствуют различные, но равноценные по числу членов МДНФ.

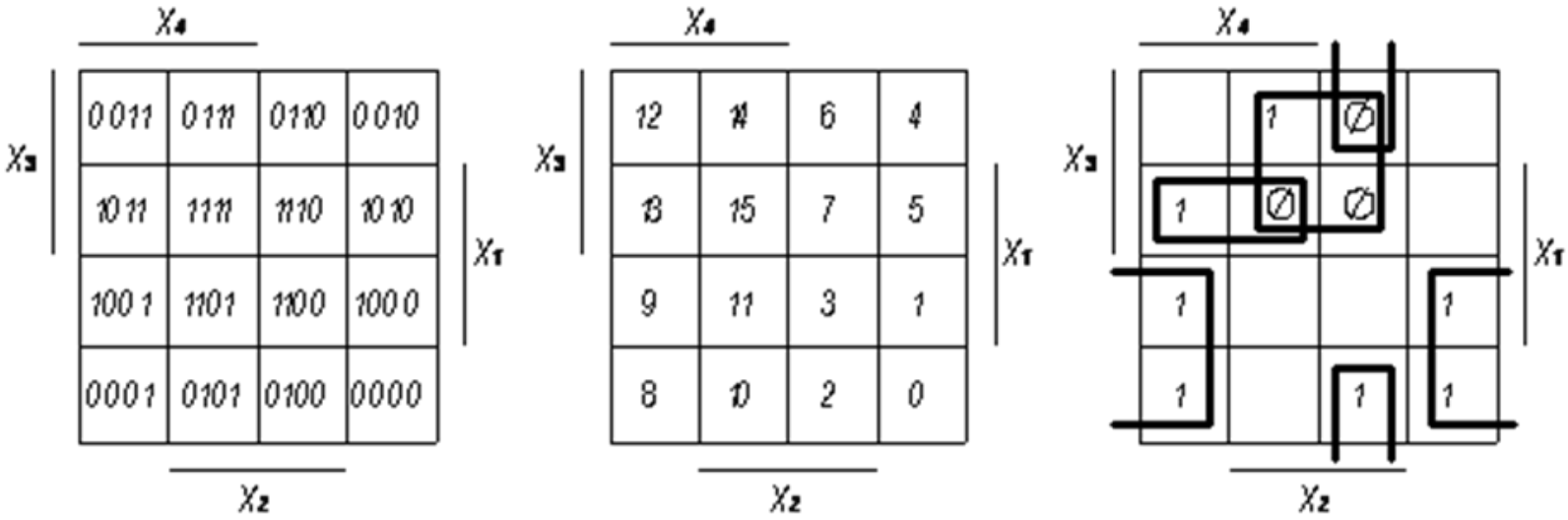
Например: Задана функция четырех переменных (аргументов)

$$F = 1110000011000110$$

В СДНФ

$$F = \overset{1}{0000} + \overset{3}{1000} + \overset{3}{0100} + \overset{4}{1100} + \overset{5}{0010} + \overset{8}{1010} + \overset{7}{0110} + \overset{8}{1110} + \overset{9}{0001} + \overset{10}{1001} +$$
$$+ \overset{11}{0101} + \overset{13}{1101} + \overset{13}{0011} + \overset{14}{11011} + \overset{15}{0111} + \overset{16}{111}$$

Наносим на карту Карно, произведем накрытие



Запишем результат накрытый в виде дезъюнкции конъюнкций (суммы произведений):

$$F = X_3X_2 + \overline{X_3}\,\overline{X_2} + X_4X_3X_1 + \overline{X_4}X_2\overline{X_1}$$

Функция до минимизации имела 16 наборов переменных, после четыре.

Логический синтез переключательных и вычислительных схем

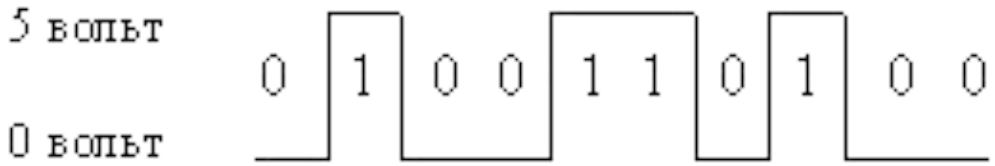
Рассмотрим какая же связь между алгеброй логики и двоичным кодированием?

Математический аппарат алгебры логики очень удобен для описания того, как функционируют аппаратные средства компьютера, поскольку основной системой счисления в компьютере является двоичная, в которой используются цифры 1 и 0, а значений логических переменных тоже два: “1” и “0”.

Из этого следует два вывода:

1. одни и те же устройства компьютера могут применяться для обработки и хранения как числовой информации, представленной в двоичной системе счисления, так и логических переменных;
2. на этапе конструирования аппаратных средств алгебра логики позволяет значительно упростить логические функции, описывающие функционирование схем компьютера, и, следовательно, уменьшить число элементарных логических элементов, из десятков тысяч которых состоят основные узлы компьютера.

В каком виде записываются в памяти компьютера и в регистрах процессора данные и команды?
Данные и команды представляются в виде двоичных последовательностей различной структуры и длины. Существуют различные физические способы кодирования двоичной информации. Мы уже рассмотрели способы записи двоичной информации на магнитных дисках и на CD-ROM. В электронных устройствах компьютера **двоичные единицы чаще всего кодируются более высоким уровнем напряжения, чем двоичные нули** (или наоборот), например:



Что такое логический элемент компьютера?

Логический элемент компьютера **— это часть электронной логической схемы, которая реализует элементарную логическую функцию.**

Логическими элементами компьютеров являются электронные схемы И, ИЛИ, НЕ, И—НЕ, ИЛИ—НЕ и другие (называемые также **вентилями**), а также **триггер.**

С помощью этих схем можно реализовать любую логическую функцию, описывающую работу устройств компьютера. Обычно у вентилей бывает от двух до восьми входов и один или два выхода.

Чтобы представить два логических состояния — “1” и “0” в вентилях, соответствующие им входные и выходные сигналы имеют один из двух установленных уровней напряжения. Например, +5 вольт и 0 вольт. Высокий уровень обычно соответствует значению “истина” (“1”), а низкий — значению “ложь” (“0”).

Каждый логический элемент имеет свое условное обозначение, которое выражает его логическую функцию, но не указывает на то, какая именно электронная схема в нем реализована. Это упрощает запись и понимание сложных логических схем.

Работу логических элементов описывают с помощью таблиц истинности.

Таблица истинности **это табличное представление логической схемы (операции), в котором перечислены все возможные сочетания значений истинности входных сигналов (операндов) вместе со значением истинности выходного сигнала (результата операции) для каждого из этих сочетаний.**

С х е м а И

Схема И реализует конъюнкцию двух или более логических значений. Условное обозначение на структурных схемах схемы **И** с двумя входами представлено на рисунке

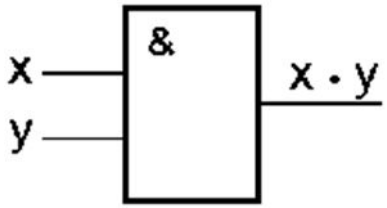


Таблица истинности схемы **И**

x	y	x · y
0	0	0
0	1	0
1	0	0
1	1	1

Единица на выходе схемы И будет тогда и только тогда, когда на всех входах будут единицы. Когда хотя бы на одном входе будет ноль, на выходе также будет ноль.

Связь между выходом z этой схемы и входами x и y описывается соотношением: $\mathbf{z = x \cdot y}$
(читается как **"x и y"**). Операция конъюнкции на структурных схемах обозначается знаком **"&"** (читается как **"амперсэнд"**), являющимся сокращенной записью английского слова **and.**

С х е м а ИЛИ

Схема ИЛИ реализует дизъюнкцию двух или более логических значений. Когда хотя бы на одном входе схемы **ИЛИ** будет единица, на её выходе также будет единица.

Условное обозначение на структурных схемах схемы **ИЛИ** с двумя входами представлено ниже Знак **"1"** на схеме — от устаревшего обозначения дизъюнкции как **">=1"** (т.е. значение дизъюнкции равно единице, если сумма значений операндов больше или равна 1). Связь между выходом **z** этой схемы и входами **x** и **y** описывается соотношением: **z = x v y** (читается как **"x или y"**).

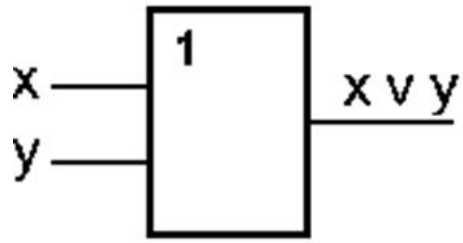


Таблица истинности схемы **ИЛИ**

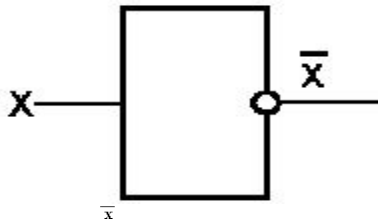
x	y	x v y
0	0	0
0	1	1
1	0	1
1	1	1

С х е м а НЕ

Схема **НЕ** (инвертор) реализует операцию отрицания. Связь между входом **x** этой схемы и выходом **z** можно записать соотношением $Z = \overline{x}$, где

$\overline{x}$ читается как **"не x"** или **"инверсия x".**

Если на входе схемы **0,** то на выходе **1.** Когда на входе **1,** на выходе **0.**
Условное обозначение на структурных схемах инвертора — на рисунке



$\overline{x}$

Таблица истинности схемы **НЕ**

x	
0	1
1	0

С х е м а И—НЕ

Схема **И—НЕ** состоит из элемента **И** и инвертора и осуществляет отрицание результата схемы **И.** Связь между выходом **z** и входами **x** и **y** схемы записывают следующим образом: $z = \overline{x \cdot y}$

где $\overline{x \cdot y}$ читается как **"инверсия x и y".** Условное обозначение на структурных схемах схемы **И—НЕ** с двумя входами представлено на рисунке

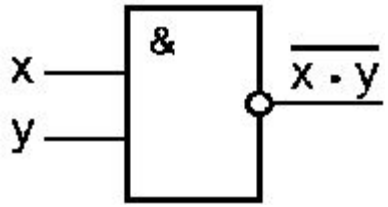


Таблица истинности схемы И—НЕ

x	y	$\overline{x \cdot y}$
0	0	1
0	1	1
1	0	1
1	1	0

С х е м а ИЛИ—НЕ

Схема **ИЛИ—НЕ** состоит из элемента **ИЛИ** и инвертора и осуществляет отрицание результата схемы **ИЛИ.** Связь между выходом **z** и входами **x** и **y** схемы записывают следующим образом:

$$z = \overline{x \vee y}$$

где $\overline{x \vee y}$ читается как **"инверсия x или y ".** Условное обозначение на структурных схемах схемы **ИЛИ—НЕ** с двумя входами представлено на рисунке

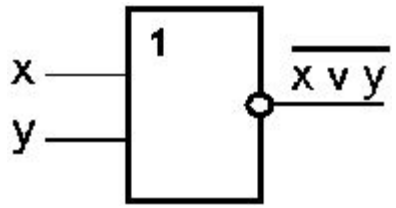


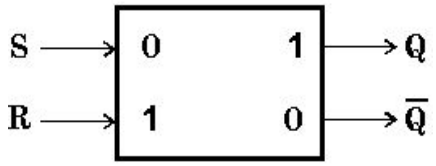
Таблица истинности схемы ИЛИ—НЕ

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

Триггер **— это электронная схема, широко применяемая в регистрах компьютера для надёжного запоминания одного разряда двоичного кода. Триггер имеет два устойчивых состояния, одно из которых соответствует двоичной единице, а другое — двоичному нулю.**

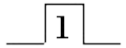
Термин **триггер** происходит от английского слова **trigger** — защёлка, спусковой крючок. Для обозначения этой схемы в английском языке чаще употребляется термин **flip-flop**, что в переводе означает “хлопанье”. Это звукоподражательное название электронной схемы указывает на её способность почти мгновенно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

Самый распространённый тип триггера — так называемый RS-триггер (S и R, соответственно, от английских *set* — установка, и *reset* — сброс). Условное обозначение триггера — на рисунке.

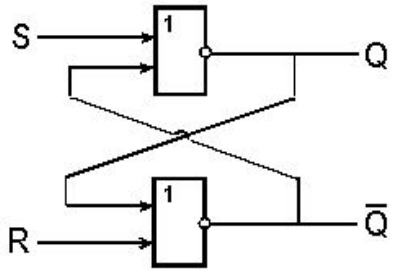


Он имеет два симметричных входа S и R и два симметричных выхода Q и $\overline{Q}$

причем выходной сигнал Q является логическим отрицанием сигнала $\overline{Q}$

На каждый из двух входов S и R могут подаваться входные сигналы в виде кратковременных импульсов 

Наличие импульса на входе будем считать единицей, а его отсутствие — нулем. На рисунке ниже показана реализация триггера с помощью вентилей ИЛИ—НЕ и соответствующая таблица истинности.



S	R	Q	$\overline{Q}$
0	0	запрещено	
0	1	1	0
1	0	0	1
1	1	хранение бита	

Проанализируем возможные комбинации значений входов R и S триггера, используя его схему и таблицу истинности схемы ИЛИ—НЕ

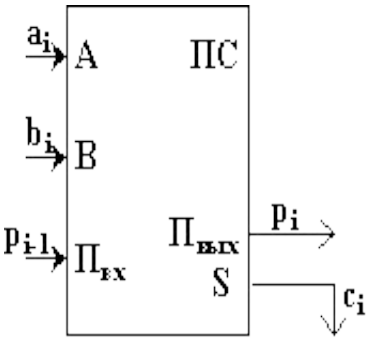
1. Если на входы триггера подать S=“1”, R=“0”, то (независимо от состояния) на выходе Q верхнего вентиля появится “0”. После этого на входах нижнего вентиля окажется R=“0”, Q=“0” и выход $\overline{Q}$ станет равным “1”.
2. Точно так же при подаче “0” на вход S и “1” на вход R на выходе $\overline{Q}$ появится “0”, а на Q — “1”.
3 Если на входы R и S подана логическая “1”, то состояние Q и $\overline{Q}$ не меняется.
4. Подача на оба входа R и S логического “0” может привести к неоднозначному результату, поэтому эта комбинация входных сигналов запрещена.

Поскольку один триггер может запомнить только один разряд двоичного кода, то для запоминания байта нужно 8 триггеров, для запоминания килобайта, соответственно, 8 x 2^{10}= 8192 триггеров. Современные микросхемы памяти содержат миллионы триггеров.

<u>Сумматор</u> — это электронная логическая схема, выполняющая суммирование двоичных чисел.

Сумматор служит, прежде всего, центральным узлом арифметико-логического устройства компьютера, однако он находит применение также и в других устройствах машины.

Многоразрядный двоичный сумматор, предназначенный для сложения многоразрядных двоичных чисел, **представляет собой комбинацию одноразрядных сумматоров,** с рассмотрения которых мы и начнём. Условное обозначение одноразрядного сумматора на рисунке



При сложении чисел A и B в одном *i*-ом разряде приходится иметь дело с тремя цифрами:

1. цифра a_i первого слагаемого;

2. цифра b_i второго слагаемого;

3. перенос p_{i-1} из младшего разряда.

В результате сложения получаются две цифры:

1. цифра c_i для суммы;

2. перенос p_i из данного разряда в старший.

Таким образом, **одноразрядный двоичный сумматор есть устройство с тремя входами и двумя выходами**, работа которого может быть описана следующей таблицей истинности:

Входы			Выходы	
Первое слагаемое	Второе слагаемое	Перенос	Сумма	Перенос
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

Если требуется складывать двоичные слова длиной два и более бит, то можно использовать последовательное соединение таких сумматоров, причём **для двух соседних сумматоров выход переноса одного сумматора является входом для другого.**

Например, схема вычисления суммы $C = (c_3\ c_2\ c_1\ c_0)$ двух двоичных трехразрядных чисел $A = (a_2\ a_1\ a_0)$ и $B = (b_2\ b_1\ b_0)$ может иметь вид: