**Лекция 3. Логические основы работы ЭВМ**

## Алгебра логики (булева алгебра) – это раздел математики, возникший в XIX веке благодаря усилиям английского математика Дж. Буля. Поначалу булева алгебра не имела никакого практического значения. Однако уже XX веке ее положения нашли применение в описании функционирования и разработке различных электронных схем. Законы и аппарат алгебры логики стал использоваться при проектировании различных частей компьютеров (память, процессор). Хотя это не единственная сфера применения данной науки.

## Что же собой представляет алгебра логики? Во-первых, она изучает методы установления истинности или ложности сложных логических высказываний с помощью алгебраических методов. Во-вторых, булева алгебра делает это таким образом, что сложное логическое высказывание описывается функцией, результатом вычисления которой может быть либо истина, либо ложь (1, либо 0). При этом аргументы функции (простые высказывания) также могут иметь только два значения: 0, либо 1.

## 3.2. Какая 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. В электронных устройствах компьютера **двоичные единицы чаще всего кодируются более высоким уровнем напряжения, чем двоичные нули** (или наоборот), например:

http://techn.sstu.ru/kafedri/%D0%BF%D0%BE%D0%B4%D1%80%D0%B0%D0%B7%D0%B4%D0%B5%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F/1/MetMat/shaturn/inform/%D0%9B%D0%B5%D0%BA%D1%86%D0%B8%D1%8F%203/0005.gif

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”).

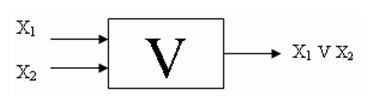
**Каждый логический элемент имеет свое условное обозначение, которое выражает его логическую функцию, но не указывает на то, какая именно электронная схема в нем реализована.** Это упрощает запись и понимание сложных логических схем.

Алгебра логики предусматривает множество логических операций. Однако три из них заслуживают особого внимания, т.к. с их помощью можно описать все остальные, и, следовательно, использовать меньше разнообразных устройств при конструировании схем. Такими операциями являются **конъюнкция** (И), **дизъюнкция** (ИЛИ) и **отрицание** (НЕ). Часто конъюнкцию обозначают **&,**ᴧ, дизъюнкцию - **||**, **V** а отрицание - чертой над переменной, обозначающей высказывание.

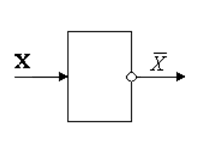
    Функция ***конъюнкции***  (логического умножения)  истинна тогда, когда истинны одновременно оба высказывания.

$IMAGE1$

***Дизъюнкция***. Читается X1 ИЛИ X2: часто это высказывание называют логическим сложением. Функция дизъюнкции истинна тогда, когда хотя бы одно из высказываний истинно.



    Логическое отрицание – ***инверсия***.     Инверсия – это высказывание, которое истинно, если исходное высказывание ложно; и, наоборот, ложно, если исходное высказывание истинно.



**Таблицы истинности**

       Логические операции удобно описывать так называемыми **таблицами истинности**, в которых отражают результаты вычислений сложных высказываний при различных значениях исходных простых высказываний. Простые высказывания обозначаются переменными (например, A и B).

В ЭВМ используются различные устройства, работу которых прекрасно описывает алгебра логики. К таким устройствам относятся группы переключателей, триггеры, сумматоры.

        Кроме того, связь между булевой алгеброй и компьютерами лежит и в используемой в ЭВМ системе счисления. Как известно она двоичная. Поэтому в устройствах компьютера можно хранить и преобразовывать как числа, так и значения логических переменных.

В ЭВМ применяются электрические схемы, состоящие из множества переключателей. Переключатель может находиться только в двух состояниях: замкнутом и разомкнутом. В первом случае – ток проходит, во втором – нет. Описывать работу таких схем очень удобно с помощью алгебры логики. В зависимости от положения переключателей можно получить или не получить сигналы на выходах.

**Вентили**

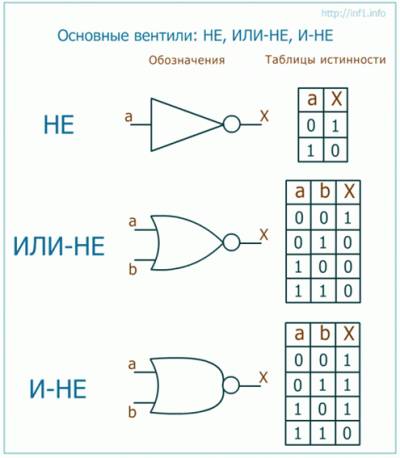
          В основе построения компьютеров, а точнее аппаратного обеспечения, лежат так называемые **вентили**. Они представляют собой достаточно простые элементы, которые можно комбинировать между собой, создавая тем самым различные схемы. Одни схемы подходят для осуществления **арифметических операций**, а на основе других строят различную **память** ЭВМ.

              Простейший вентиль представляет собой транзисторный инвертор, который преобразует низкое напряжение в высокое или наоборот (высокое в низкое). Это можно представить как преобразование логического нуля в логическую единицу или наоборот. т.е. получаем вентиль **НЕ**.

          Соединив пару транзисторов различным способом, получают вентили **ИЛИ-НЕ** и **И-НЕ**. Эти вентили принимают уже не один, а два и более входных сигнала.        Выходной сигнал всегда один и зависит (выдает высокое или низкое напряжение) от входных сигналов. В случае вентиля ИЛИ-НЕ получить высокое напряжение (логическую единицу) можно только при условии низкого напряжении на всех входах. В случае вентиля И-НЕ все наоборот: логическая единица получается, если все входные сигналы будут нулевыми. Как видно, это обратно таким привычным логическим операциям как И и ИЛИ. Однако обычно используются вентили И-НЕ и ИЛИ-НЕ, т.к. их реализация проще: И-НЕ и ИЛИ-НЕ реализуются двумя транзисторами, тогда как логические И и ИЛИ тремя.

***Выходной сигнал вентиля можно выражать как функцию от входных.***

           Транзистору требуется очень мало времени для переключения из одного состояния в другое (время переключения оценивается в наносекундах). И в этом одно из существенных преимуществ схем, построенных на их основе.

[](http://apx2013.ucoz.ru/_si/0/44136441.png)

**Сумматор и полусумматор**

Арифметико-логическое устройство процессора (АЛУ) обязательно содержит в своем составе такие элементы как **сумматоры**. Эти схемы позволяют складывать двоичные числа.

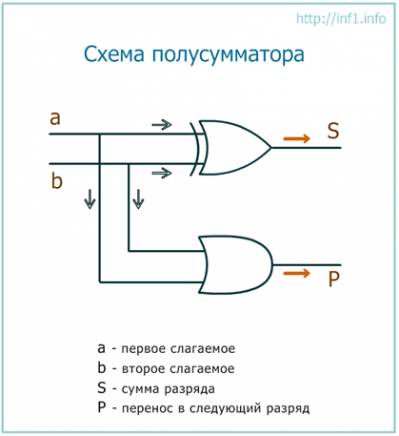
Как происходит сложение? Допустим, требуется сложить двоичные числа 1001 и 0011. Сначала складываем младшие разряды (последние цифры): 1+1=10. Т.е. в младшем разряде будет 0, а единица – это перенос в старший разряд. Далее: 0 + 1 + 1(от переноса) = 10, т.е. в данном разряде снова запишется 0, а единица уйдет в старший разряд. На третьем шаге: 0 + 0 + 1(от переноса) = 1. В итоге сумма равна 1100.

**Полусумматор**

         Теперь не будем обращать внимание на перенос из предыдущего разряда и рассмотрим только, как формируется сумма текущего разряда. Если были даны две единицы или два нуля, то сумма текущего разряда равна 0. Если одно из двух слагаемых равно единице, то сумма равна единицы. Получить такие результаты можно при использовании вентиля **ИСКЛЮЧАЮЩЕГО ИЛИ.**

           Перенос единицы в следующий разряд происходит, если два слагаемых равны единице. И это реализуемо вентилем И.

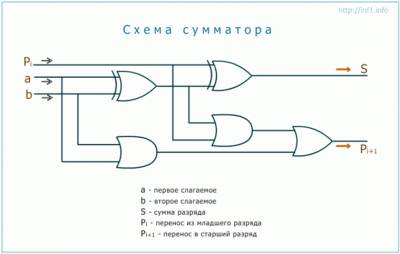
        Тогда сложение в пределах одного разряда (без учета возможной пришедшей единицы из младшего разряда) можно реализовать изображенной ниже схемой, которая называется **полусумматором**. У полусумматора два входа (для слагаемых) и два выхода (для суммы и переноса). На схеме изображен полусумматор, состоящий из вентилей **ИСКЛЮЧАЮЩЕЕ ИЛИ и И.**

[](http://apx2013.ucoz.ru/_si/0/44206781.png)

**Сумматор**

         В отличие от полусумматора **сумматор** учитывает перенос из предыдущего разряда, поэтому имеет не два, а три входа.

         Чтобы учесть перенос приходится схему усложнять. По сути она получается, состоящей из двух полусумматоров.

[](http://apx2013.ucoz.ru/_si/0/61914101.png)

        Рассмотрим один из случаев. Требуется сложить 0 и 1, а также 1 из переноса. Сначала определяем сумму текущего разряда. Судя по левой схеме ИСКЛЮЧАЮЩЕЕ ИЛИ, куда входят a и b, на выходе получаем единицу. В следующее ИСКЛЮЧАЮЩЕЕ ИЛИ уже входят две единицы. Следовательно, сумма будет равна 0.

       Теперь смотрим, что происходит с переносом. В один вентиль И входят 0 и 1 (a и b). Получаем 0. Во второй вентиль (правее) заходят две единицы, что дает 1. Проход через вентиль ИЛИ нуля от первого И и единицы от второго И дает нам 1.

       Проверим работу схемы простым сложением 0 + 1 + 1 = 10. Т.е. 0 остается в текущем разряде, и единица переходит в старший. Следовательно, логическая схема работает верно.

       Работу данной схемы при всех возможных входных значениях можно описать следующей таблицей истинности.

[](http://apx2013.ucoz.ru/_si/0/52431688.png)

**Триггер как элемент памяти. Схема RS-триггера**

      Память (устройство, предназначенное для хранения данных и команд) является важной частью компьютера. Можно сказать, что она его и определяет: если вычислительное устройство не имеет памяти, то оно уже не компьютер.

     Элементарной единицей компьютерной памяти является бит. Поэтому требуется устройство, способное находиться в двух состояниях, т.е. хранить единицу или ноль. Также это устройство должно уметь быстро переключаться из одного состояния в другое под внешним воздействием, что дает возможность изменять информацию. Ну и наконец, устройство должно позволять определять его состояние, т.е. предоставлять во вне информацию о своем состоянии.

     Устройством, способным запоминать, хранить и позволяющим считывать информацию, является триггер. Он был изобретен в начале XX века Бонч-Бруевичем.

    К триггерным принято относить все устройства, имеющих два устойчивых состояния. В основе любого триггера находится кольцо из двух инверторов, показанное на рис.12.1. Общепринято это кольцо изображать в виде так называемой защелки, которая показана на рис.

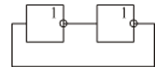


Рис. Кольцо из двух инверторов

      Разнообразие триггеров весьма велико. Наиболее простой из них так называемый **RS-триггер**, который собирается из двух вентилей. Обычно используют вентили ИЛИ-НЕ или И-НЕ.

**RS-триггер на вентилях ИЛИ-НЕ**

    RS-триггер «запоминает», на какой его вход подавался сигнал, соответствующий единице, в последний раз. Если сигнал был подан на S-вход, то триггер на выходе постоянно «сообщает», что хранит единицу. Если сигнал, соответствующий единице, подан на R-вход, то триггер на выходе имеет 0. Не смотря на то, что триггер имеет два выхода, имеется в виду выход Q. (Q с чертой всегда имеет противоположное Q значение.)

     Другими словами, вход S (set) отвечает за установку триггера в 1, а вход R (reset) – за установку триггера в 0. Установка производится сигналом, с высоким напряжением (соответствует единице). Просто все зависит от того, на какой вход он подается.

       Большую часть времени на входы подается сигнал равный 0 (низкое напряжение). При этом триггер сохраняет свое прежнее состояние.

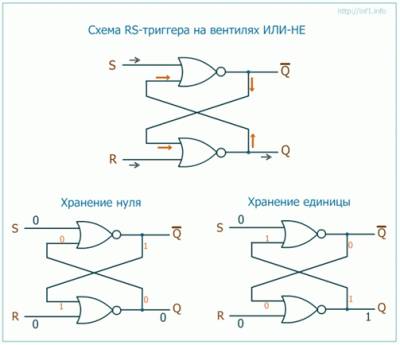
       Возможны следующие ситуации:

* Q = 1, сигнал подан на S, следовательно, Q не меняется.
* Q = 0, сигнал подан на S, следовательно, Q = 1.
* Q = 1, сигнал подан на R, следовательно, Q = 0.
* Q = 0, сигнал подан на R, следовательно, Q не меняется.

      Ситуация, при которой на оба входа подаются единичные сигналы, недопустима.

   Как триггер сохраняет состояние? Допустим, триггер выдает на выходе Q логический 0. Тогда судя по схеме, этот 0 возвращается также и в верхний вентиль, где инвертируется (получается 1) и уже в этом виде передается нижнему вентилю. Тот в свою очередь снова инвертирует сигнал (получается 0), который и имеется на выходе Q. Состояние триггера сохраняется, он хранит 0.

     Теперь, допустим, был подан единичный сигнал на вход S. Теперь в верхний вентиль входят два сигнала: 1 от S и 0 от Q. Поскольку вентиль вида ИЛИ-НЕ, то на выходе из него получается 0. Ноль идет на нижний вентиль, там инвертируется (получается 1). Сигнал на выходе Q становится соответствующим 1.

[](http://apx2013.ucoz.ru/_si/0/69141467.png)

***Т*‑триггер**

        Триггер типа *Т* называется триггером со счётным входом. Он изменяет своё состояние на противоположное каждый раз, когда на его вход приходит очередной сигнал.  Обозначение триггера произошло от первой буквы английского слова *toggle* – защёлка.   
          Условное графическое обозначение *Т*-триггера показано на рис. *Т*‑триггер имеет один вход *Т* и два выхода *Q* и Õ. *T* - счётный вход триггера.

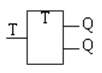


Рис. Условное графическое обозначение *T*‑триггера

         Принцип работы триггера иллюстрирует его таблица истинности



      Состояние его выхода меняется на противоположное при поступлении на вход счётного сигнала *Т=1* и сохраняется неизменным при *Т=0*. В соответствии с табл.12.5 характеристическое уравнение *Т*‑триггера имеет вид:

http://apx2013.ucoz.ru/_si/0/05647365.png

        Согласно этому уравнению *Т*‑триггер сохраняет неизменное состояние при *Т=0*, когда *Q n+1=Q n* и при *Т=1*, когда *Q n+1=*http://apx2013.ucoz.ru/_si/0/78792763.png

        Т‑триггер может быть реализован введением в *RS*‑ и *D*‑триггеры обратных связей

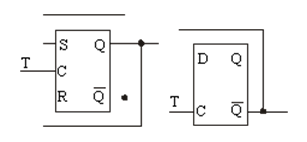


Рис. Схемы *T*‑триггера

***JK*‑триггер**

*JK*‑триггер наиболее широко используемый универсальный триггер, обладающий характеристиками всех других типов триггеров. *JK*‑триггер в отличии от *RS*‑триггера не имеет запрещенных комбинаций входных сигналов, которые следует исключать при работе цифровых систем.

        На рис. показано условное графическое обозначение *JK*‑триггера.

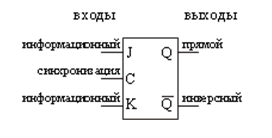


Рис. Условное графическое обозначение *JK*‑триггера

         Рассмотрим табл., иллюстрирующую принципы работы *JK*‑триггера.

Таблица истинности для *JK*‑триггера

[](http://apx2013.ucoz.ru/_si/0/75360073.png)

       Из табл. видно, что когда на оба входа *J* и *K* подается уровень логического *0*, триггер блокируется, и состояния его выходов не изменяются. В этом случае триггер находится в режиме хранения.

         Строки 2 и 3 табл. описывают режимы, соответствующие установке триггера в состояние *0* и *1*. Строка 4 соответствует переключательному режиму работы *JK*‑триггера. Если на обоих входах *J* и *K* установлен уровень логической *1*, то следующие друг за другом тактовые импульсы будут вызывать перебросы уровней сигналов на выходах триггера от *1* к *0*, от *0* к *1* и т.д. Такая работа триггера напоминает последовательно производимые переключения тумблера, откуда и происходит название режима.

          Характеристическое уравнение *JK*‑триггера имеет вид:

http://apx2013.ucoz.ru/_si/0/30841816.png

*JK*‑триггер может быть реализован с использованием двух элементов И и *RS*‑триггера

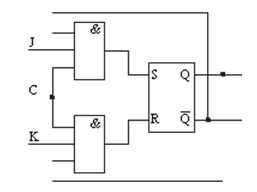


Рис. Реализация *JK*‑т