Раздел 1. Основные функциональные элементы ЭВМ. Архитектуры.

Лекция 1 Основные характеристики и основные функциональные элементы ЭВМ.

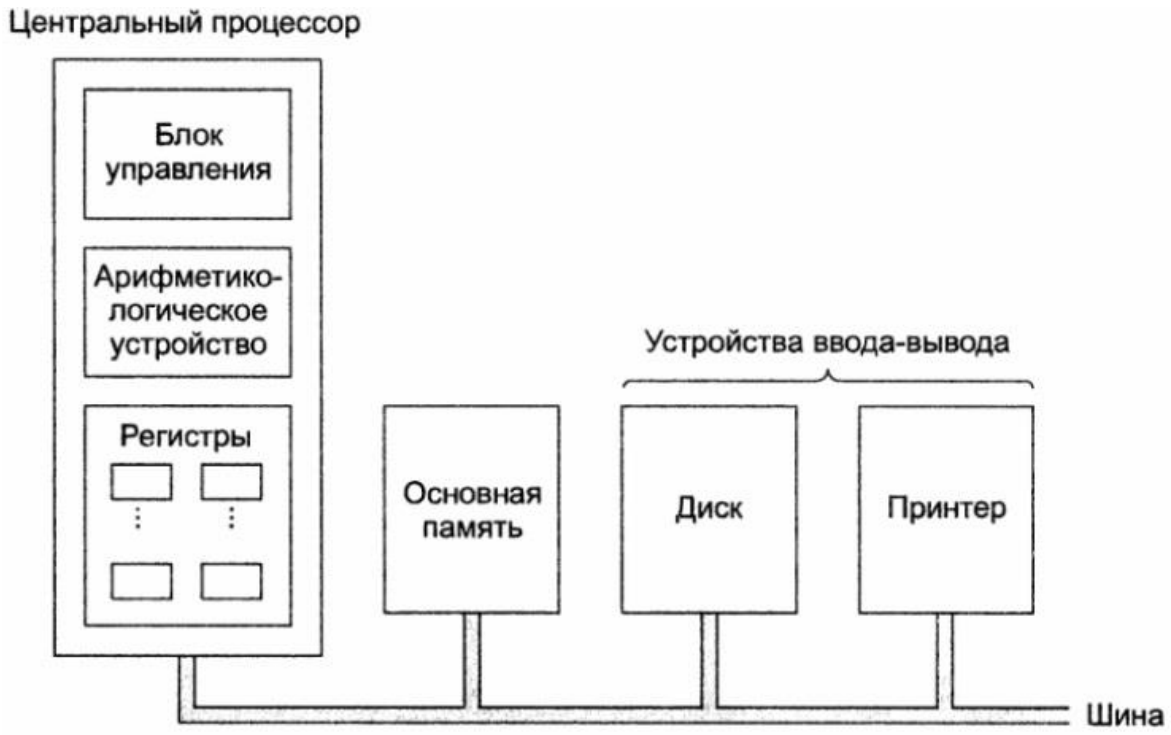


Рис. 1. Схема компьютера с одним центральным процессором и двумя устройствами ввода-вывода

Архитектура ЭВМ включает в себя как структуру, отражающую состав ПК, так и программно–математическое обеспечение. Структура ЭВМ - совокупность элементов и связей между ними. Основным принципом построения всех современных аппаратных средств является программное управление. Основы учения об архитектуре вычислительных машин были заложены Джон фон Нейманом. Совокупность этих принципов породила классическую (фон-неймановскую) архитектуру ЭВМ. Фон Нейман не только выдвинул основополагающие принципы логического устройства ЭВМ, но и предложил ее структуру, представленную на рисунке.

Положения фон Неймана:

- Компьютер состоит из нескольких основных устройств (арифметико-логическое устройство, управляющее устройство, память, внешняя память, устройства ввода и вывода)
- Арифметико-логическое устройство – выполняет логические и арифметические действия, необходимые для переработки информации, хранящейся в памяти
- Управляющее устройство – обеспечивает управление и контроль всех устройств компьютера (управляющие сигналы указаны пунктирными стрелками)

• Данные, которые хранятся в запоминающем устройстве, представлены в двоичной форме
• Программа, которая задает работу компьютера, и данные хранятся в одном и том же запоминающем устройстве
• Для ввода и вывода информации используются устройства ввода и вывода

Один из важнейших принципов – принцип хранимой программы – требует, чтобы программа закладывалась в память машины так же, как в нее закладывается исходная информация.

Арифметико-логическое устройство и устройство управления в современных компьютерах образуют процессор ЭВМ. Процессор, который состоит из одной или нескольких больших интегральных схем называется микропроцессором или микропроцессорным комплектом. Процессор – функциональная часть ЭВМ, выполняющая основные операции по обработке данных и управлению работой других блоков.

Процессор является преобразователем информации, поступающей из памяти и внешних устройств.

Запоминающие устройства обеспечивают хранение исходных и промежуточных данных, результатов вычислений, а также программ. Они включают: оперативные (ОЗУ), сверхоперативные СОЗУ), постоянные (ПЗУ) и внешние (ВЗУ) запоминающие устройства. Оперативные ЗУ хранят информацию, с которой компьютер работает непосредственно в данное время (резидентная часть операционной системы, прикладная программа, обрабатываемые данные). В СОЗУ хранится наиболее часто используемые процессором данные. Только та информация, которая хранится в СОЗУ и ОЗУ, непосредственно доступна процессору. Внешние запоминающие устройства (накопители на магнитных дисках, например, жесткий диск или винчестер) с емкостью намного больше, чем ОЗУ, но с существенно более медленным доступом, используются для длительного хранения больших объемов информации. Например, операционная система (ОС) хранится на жестком диске, но при запуске компьютера резидентная часть ОС загружается в ОЗУ и находится там до завершения сеанса работы ПК. ПЗУ (постоянные запоминающие устройства) и ППЗУ (перепрограммируемые постоянные запоминающие устройства) предназначены для постоянного хранения информации, которая записывается туда при ее изготовлении, например, ППЗУ для BIOS.

В качестве устройства ввода информации служит, например, клавиатура. В качестве устройства вывода – дисплей, принтер и т.д.

В построенной по схеме фон Неймана ЭВМ происходит последовательное считывание команд из памяти и их выполнение. Номер (адрес) очередной ячейки памяти, из которой будет извлечена следующая команда программы, указывается специальным устройством – счетчиком команд в устройстве управления.

Лекция 2. Основные логические элементы. Их роль при построении различных узлов и устройств ЭВМ.

Основные понятия алгебры логики

Логической основой компьютера является алгебра логики, которая рассматривает логические операции над высказываниями.

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

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

Таблица 1. Основные логические операции

Обозначение операции	Читается	Название операции	Альтернативные обозначения
¬	НЕ	Отрицание (инверсия)	Черта сверху
∧	И	Конъюнкция (логическое умножение)	· &
∨	ИЛИ	Дизъюнкция (логическое сложение)	+
→	Если … то	Импликация	⊃
↔	Тогда и только тогда	Эквиваленция	~
XOR	Либо …либо	Исключающее ИЛИ (сложение по модулю 2)	⊕

НЕ Операция, выражаемая словом «не», называется **отрицанием** и обозначается чертой над высказыванием (или знаком ¬). Высказывание ¬A истинно, когда A ложно, и ложно, когда A истинно.

Пример. Пусть A=«Сегодня пасмурно», тогда ¬A=«Сегодня не пасмурно».

И Операция, выражаемая связкой «и», называется **конъюнкцией** (лат. conjunctio – соединение) или логическим умножением и обозначается точкой « • » (может также обозначаться знаками ∧ или &). Высказывание A • B истинно тогда и только тогда, когда оба высказывания A и B истинны.

Пример. Высказывание «Число 6 делится на 2, и число 6 делится на 3» - истинно, а высказывание «Число 6 делится на 2, и число 6 больше 10» - ложно.

ИЛИ Операция, выражаемая связкой «или» (в неисключающем смысле этого слова), называется **дизъюнкцией** (лат. disjunctio – разделение) или логическим сложением и обозначается знаком $\vee$ (или плюсом). Высказывание $A \vee B$ ложно тогда и только тогда, когда оба высказывания A и B ложны.

Пример: Высказывание «Число 6 делится на 2 или число 6 больше 10» - истинно, а высказывание «Число 6 делится на 5 или число 6 больше 10» - ложно.

ЕСЛИ … ТО Операция, выражаемая связками «если …, то», «из … следует», «... влечет …», называется **импликацией**(лат. implico – тесно связаны) и обозначается знаком $\rightarrow$. Высказывание $A \rightarrow B$ ложно тогда и только тогда, когда A истинно, а B ложно.

Пример. Высказывание «если студент сдал все экзамены на «отлично», то он получит стипендию». Очевидно, эту импликацию следует признать ложной лишь в том случае, когда студент сдал на «отлично» все экзамены, но стипендии не получил. В остальных случаях, когда не все экзамены сданы на «отлично» и стипендия получена (например, в силу того, что студент проживает в малообеспеченной семье) либо когда экзамены вообще не сданы и о стипендии не может быть и речи, импликацию можно признать истинной.

РАВНОСИЛЬНО Операция, выражаемая связками «тогда и только тогда», «необходимо и достаточно», «... равносильно …», называется **эквиваленцией** или **двойной импликацией** и обозначается знаком $\leftrightarrow$ или $\sim$. Высказывание $A \leftrightarrow B$ истинно тогда и только тогда, когда значения A и B совпадают.

Пример: Высказывание «Число является четным тогда и только тогда, когда оно делится без остатка на 2» является истинным, а высказывание «Число является нечетным тогда и только тогда, когда оно делится без остатка на 2» - ложно.

ЛИБО … ЛИБО Операция, выражаемая связками «Либо … либо», называется **исключающее ИЛИ** или **сложением по модулю 2** и обозначается XOR или $\oplus$. Высказывание $A \oplus B$ истинно тогда и только тогда, когда значения A и B не совпадают.

Пример. Высказывание «Число 6 либо нечетно либо делится без остатка на 2» является истинным, а высказывание «Либо число 6 четно либо число 6 делится на 3» – ложно, так как истинны оба высказывания входящие в него.

Замечание. Импликацию можно выразить через дизъюнкцию и отрицание:

$$A \rightarrow B = \neg A \vee B .$$

Эквиваленцию можно выразить через отрицание, дизъюнкцию и конъюнкцию:

$A \leftrightarrow B = (\neg A \vee B) \wedge (\neg B \vee A)$.

Исключающее ИЛИ можно выразить через отрицание, дизъюнкцию и конъюнкцию:

$A \text{ XOR } B = (\neg A \wedge B) \vee (\neg \text{B} \,\&\, \text{A})$.

Вывод. Операций отрицания, дизъюнкции и конъюнкции достаточно, чтобы описывать и обрабатывать логические высказывания.

Порядок выполнения логических операций задается круглыми скобками. Но для уменьшения числа скобок договорились считать, что сначала выполняется операция отрицания («не»), затем конъюнкция («и»), после конъюнкции – дизъюнкция («или») и исключающего или и в последнюю очередь – импликация и эквиваленция.

С помощью логических переменных и символов логических операций любое высказывание можно формализовать, то есть заменить логической формулой (логическим выражением).

Логическая формула - это символическая запись высказывания, состоящая из логических величин (констант или переменных), объединенных логическими операциями (**логическими связками**).

Логическая функция - это функция логических переменных, которая может принимать только два значения: 0 или 1. В свою очередь, сама логическая переменная (аргумент логической функции) тоже может принимать только два значения: 0 или 1.

Пример. $F(\text{A,B}) = \text{A} \,\&\, \text{B} \vee A$ – логическая функция двух переменных A и B.

Значения логической функции для разных сочетаний значений входных переменных – или, как это иначе называют, наборов входных переменных – обычно задаются специальной таблицей. Такая таблица называется **таблицей истинности**.

Приведем таблицу истинности основных логических операций (табл. 2)

Таблица 2

A	B	$\neg A$	A & B	$A \vee B$	$A \rightarrow B$	$A \leftrightarrow B$	$A \text{ XOR } B$
1	1	0	1	1	1	1	0
1	0	0	0	1	0	0	1
0	1	1	0	1	1	0	1
0	0	1	0	0	1	1	0

Опираясь на данные таблицы истинности основных логических операций, можно составлять таблицы истинности для более сложных формул.

Алгоритм построения таблиц истинности для сложных выражений:

1. Определить количество строк:
   - количество строк = 2^n+ строка для заголовка,

- n - количество простых высказываний.

2. Определить количество столбцов:
   - количество столбцов = количество переменных + количество логических операций;
   - определить количество переменных (простых выражений);
   - определить количество логических операций и последовательность их выполнения.

Пример 1. Составить таблицу истинности для формулы И–НЕ, которую можно записать так: $\neg(A \& B)$.

1. Определить количество строк:

На входе два простых высказывания: A и B, поэтому n=2 и количество строк $=2^2+1=5$.

2. Определить количество столбцов:

Выражение состоит из двух простых выражений (A и B) и двух логических операций (1 инверсия, 1 конъюнкция), т.е. количество столбцов таблицы истинности = 4.

3. Заполнить столбцы с учетом таблиц истинности логических операций (табл. 3).

Таблица 3. Таблица истинности для логической операции

A	B	$A \& B$	$\neg(A \& B)$
1	1	1	0
1	0	0	1
0	1	0	1
0	0	0	1

Подобным образом можно составить таблицу истинности для формулы ИЛИ–НЕ, которую можно записать так:
$\neg(A \vee B)$.

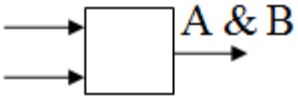
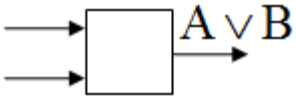
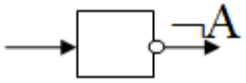
Логические элементы

Логические формулы можно также представлять с помощью языка логических схем.

Электронные схемы, реализующие логические операции, называют **логическими элементами**.

Существует три базовых логических элемента, которые реализуют три основные логические операции:

- логический элемент «И» – логическое умножение – конъюнктор;
- логический элемент «ИЛИ» – логическое сложение – дизъюнктор;
- логический элемент «НЕ» – инверсию – инвертор.

конъюнктор	дизъюнктор	инвертор
		

Поскольку любая логическая операция может быть представлена в виде комбинации трех основных, любые устройства компьютера, производящие обработку или хранение информации, могут быть собраны из базовых логических элементов, как из “кирпичиков”.

Логические элементы компьютера оперируют с сигналами, представляющими собой электрические импульсы. Есть импульс – логический смысл сигнала – 1, нет импульса – 0. На входы логического элемента поступают сигналы-значения аргументов, на выходе появляется сигнал-значение функции.

Преобразование сигнала логическим элементом задается таблицей состояний, которая фактически является таблицей истинности, соответствующей логической функции, только представлена в форме логических схем. В такой форме удобно изображать цепочки логических операций и производить их вычисления.

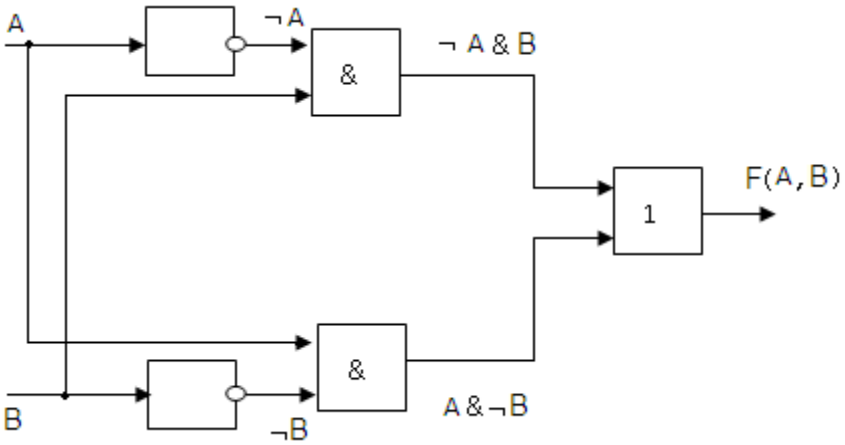
Алгоритм построения логических схем.

1. Определить число логических переменных.
2. Определить количество логических операций и их порядок.
3. Изобразить для каждой логической операции соответствующий ей логический элемент.
4. Соединить логические элементы в порядке выполнения логических операций.

Пример. По заданной логической функции $F(A,B) = \neg A \,\&\, B \vee A \,\&\, \neg B$ построить логическую схему.

Решение.

1. Число логических переменных = 2 (A и B).
2. Количество операций = 5 (2 инверсии, 2 конъюнкции, 1 дизъюнкция). Сначала выполняются операции инверсии, затем конъюнкции, в последнюю очередь операция дизъюнкции.
3. Схема будет содержать 2 инвертора, 2 конъюнктора и 1 дизъюнктор.
4. Построение надо начинать с логической операции, которая должна выполняться последней. В данном случае такой операцией является логическое сложение, следовательно, на выходе должен быть дизъюнктор. На него сигналы подаются с двух конъюнкторов, на которые, в свою очередь, подаются один входной сигнал нормальный и один инвертированный (с инверторов).



Логические законы и правила преобразования логических выражений

Если две формулы A и B одновременно, то есть при одинаковых наборах значений входящих в них переменных, принимают одинаковые значения, то они называются **равносильными.**

В алгебре логики имеется ряд законов, позволяющих производить равносильные преобразования логических выражений.

1. Закон двойного отрицания: $A = \neg(\neg A)$;
2. Переместительный (коммутативный) закон:
   - для логического сложения: $A \vee B = B \vee A$;
   - для логического умножения: $A \wedge B = B \wedge A$;
3. Сочетательный (ассоциативный) закон:
   - для логического сложения: $(A \vee B) \vee C = A \vee (B \vee C)$;
   - для логического умножения: $(A \wedge B) \wedge C = A \wedge (B \wedge C)$;
4. Распределительный (дистрибутивный) закон:
   - для логического сложения: $(A \vee B) \wedge C = (\mathrm{A} \,\&\, \mathrm{C}) \vee (\mathrm{B} \,\&\, \mathrm{C})$;
   - для логического умножения: $(A \wedge B) \vee C = (A \vee C) \wedge (B \vee C)$;
5. Законы де Моргана:
   - для логического сложения: $\neg(A \vee B) = \neg \mathrm{A} \,\&\, \neg B$;
   - для логического умножения: $\neg(A \wedge B) = \neg A \vee \neg B$;
6. Закон идемпотентности:
   - для логического сложения: $A \vee A = A$;
   - для логического умножения: $A \wedge A = A$;
7. Законы исключения констант:
   - для логического сложения: $A \vee 1 = 1$, $A \vee 0 = A$;
   - для логического умножения: $A \wedge 1 = A$, $A \wedge 0 = 0$;
8. Закон противоречия: $\mathrm{A} \,\&\, \neg A = 0$;
9. Закон исключения третьего: $A \vee \neg A = 1$;
10. Закон поглощения:
    - для логического сложения: $A \vee (A \wedge B) = A$;
    - для логического умножения: $A \wedge (A \vee B) = A$;
11. Правило исключения импликации: $A \to B = \neg A \vee B$;
12. Правило исключения эквиваленции: $A \leftrightarrow B = (A \to B) \wedge (B \to A)$.

Справедливость этих законов можно доказать составив таблицу истинности выражений в правой и левой части и сравнив соответствующие значения.
Основываясь на законах, можно выполнять упрощение сложных логических выражений. Такой процесс замены сложной логической функции более простой, но равносильной ей, называется минимизацией функции.
Пример. Упростить логическое выражение $\neg(A \vee B) \wedge (A \,\&\, \neg B)$.
Решение:
Согласно закону де Моргана:
$\neg(A \vee B) \wedge (A \,\&\, \neg B) \vee A = \neg A \,\&\, \neg B \,\&\, (A \,\&\, \neg B) \vee A$.
Согласно сочетательному закону:
$\neg A \,\&\, \neg B \,\&\, (A \,\&\, \neg B) \vee A = \neg A \,\&\, A \,\&\, \neg B \,\&\, \neg B \vee A$.
Согласно закону противоречия и закону идемпотентности:
$\neg A \,\&\, A \,\&\, \neg B \,\&\, \neg B \vee A = 0 \wedge \neg B \,\&\, \neg B = 0 \,\&\, \neg B \vee A$.
Согласно закону исключения 0:
$0 \,\&\, \neg B = 0$
Окончательно получаем
$\neg(A \vee B) \wedge (A \,\&\, \neg B) \vee A = 0 \vee A = A$ /

Тема 1.2 Функциональные схемы и узлы ЭВМ

Элементы и узлы ЭВМ.

Элемент ЭВМ – наименьшая конструктивная и функциональная часть ЭВМ, которая используется при ее логическом проектировании и технологической реализации. По назначению они различаются на логические, запоминающие и вспомогательные.

Логические элементы реализуют логические операции и применяются как для построения сложных логических схем (узлов), так и для управления работой отдельных блоков и устройств ЭВМ.

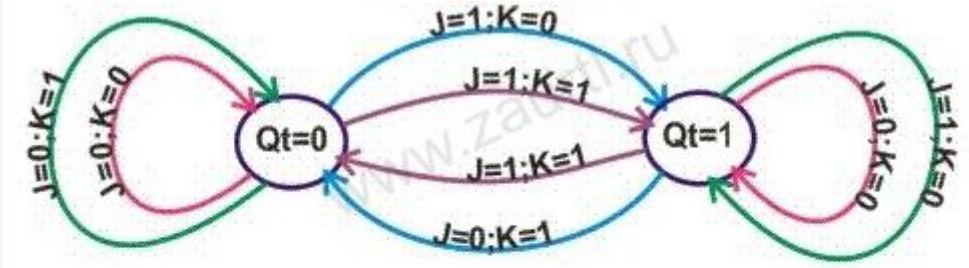
Запоминающие элементы предназначены для хранения и выдачи двоичной информации.

Вспомогательные элементы используются чаще всего для энергетического обеспечения и согласования работы различных блоков ЭВМ.

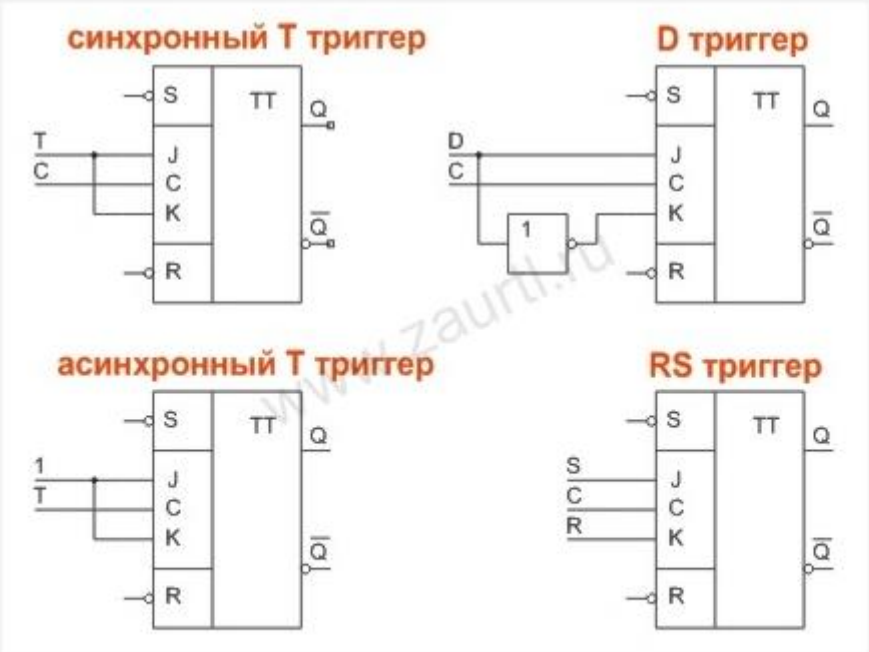
Рассмотрим принцип построения и функционирования элементов и узлов широко применяемых в ЭВМ.

Триггер – элементарный цифровой автомат с двумя устойчивыми состояниями. Состояние 0 на выход Q соответствует выключенному состоянию, а Q=1 – включенному. Триггеры осуществляют запоминание информации и остаются в заданном состоянии после прекращения действия переключающих сигналов. Они широко применяются широко применяются при цифровой обработке информации.
По способу организации логических связей, определяющие особенности функционирования, различают триггеры RS, T, D, JK. Из них JK триггер называется универсальным, так как из него можно получить все остальные виды триггеров.

Принцип работы JK триггера хорошо поясняется на графе переходов.



Схемы включения JK триггера:



Асинхронный T триггер – счетный триггер, каждые два сигнала на входе T формируют один сигнал на выходе.

Синхронный T триггер – счетный триггер, каждые два сигнала на входе C формируют один сигнал на выходе, если на входе T присутствует логическая 1.

Синхронный D триггер – реализует функцию временной задержки. Функционирует в соответствии со следующей таблицей переходов.

Qt	D	C	Qt+1
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

Асинхронный RS триггер – элементарный цифровой автомат с двумя устойчивыми состояниями и двумя входами R и S, функционирующий в соответствии со следующей таблицей переходов.

Асинхронный RS триггер – элементарный цифровой автомат с двумя устойчивыми состояниями и двумя входами R и S, функционирующий в соответствии со следующей таблицей переходов.

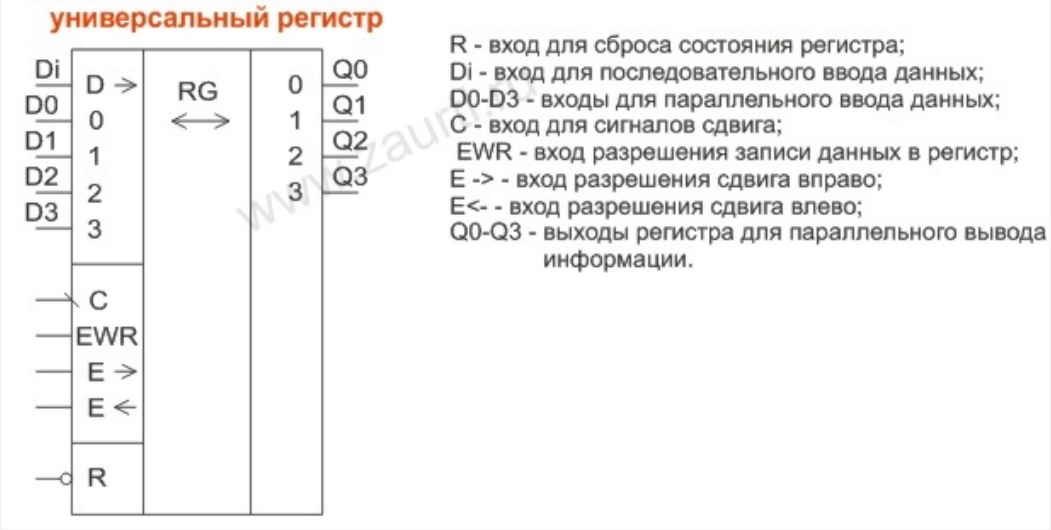
Qt	R	S	Qt+1	описание
0	0	0	0	хранение 0
0	0	1	1	установка 1
0	1	0	0	подтверждение 0
0	1	1	x	неопределенное
1	0	0	1	хранение 1
1	0	1	1	подтверждение 1
1	1	0	0	установка 0
1	1	1	x	неопределенное

Синхронный RS триггер – отличается от асинхронных RS триггеров тем, что кроме информационных входов имеет вход синхронизации C. При C=0 триггер находится в режиме хранения информации. При C=1 синхронный триггер работает как асинхронный RS триггер.

Регистры – это узлы ЭВМ, служащие для хранения информации в виде машинных слов или его частей, а так же для выполнения над словами некоторых логических преобразований. Они представляют собой цифровые автоматы Мили, выполненные на триггерах.
Регистры способны выполнять следующие операции:
- установка регистра в состояние 0 или 1 (на всех выходах);
- прием и хранение в регистре n разрядного слова;
- сдвиг хранимого в регистре двоичного кода слова в право или в лево на заданное значение разрядов;

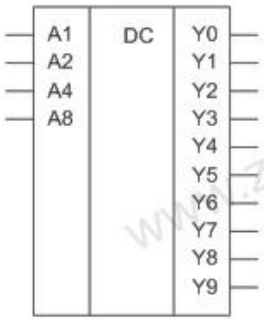
- преобразование кода хранимого слова в последовательный, и наоборот, при приеме или при выдачи двоичных данных;
- поразрядные логические операции.

Ниже показано условно графическое обозначение универсального регистра и назначение его выводов:



Счетчики – узлы ЭВМ, которые осуществляют счет и хранение кода числа подсчитанных сигналов. Они представляют собой цифровые автоматы Мура, в которых новое состояние счетчика определяется его предыдущим состоянием и состоянием логической переменной на входе.
Внутреннее состояние счетчиков характеризуется коэффициентом пересчета К, определяющим число его устойчивых состояний. Основными параметрами являются разрешающая способность (минимальное время между двумя сигналами, которые надежно фиксируются) или максимальное быстродействие и информационная емкость. Обозначение и назначение выводов реверсивного счетчика показано на рисунке ниже.

Дешифратор, или избирательная схема, – это узел ЭВМ, в котором каждой комбинации входных сигналов соответствует наличие сигнала на одной вполне определенной шине на выходе (комбинационное устройство). Дешифраторы широко используются для преобразования двоичных кодов в управляющие сигналы для различных устройств ЭВМ.

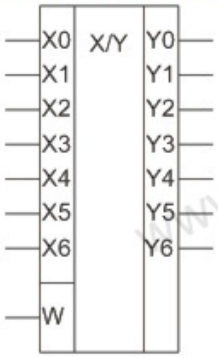


A8	A4	A2	A1	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7	Y8	Y9
0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0	0
0	0	1	1	0	0	0	1	0	0	0	0	0	0
0	1	0	0	0	0	0	0	1	0	0	0	0	0
0	1	0	1	0	0	0	0	0	1	0	0	0	0
0	1	1	0	0	0	0	0	0	0	1	0	0	0
0	1	1	1	0	0	0	0	0	0	0	1	0	0
1	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	1	0	0	0	0	0	0	0	0	0	1

Шифратор, или кодер, – это узел ЭВМ, преобразующий унитарный код в некоторый позиционный код. Если выходной код является двоичным позиционным, то шифратор называется двоичным. С помощью шифраторов возможно преобразование цифр десятичных чисел в двоичное представление с использованием любого другого двоично-десятичного кода.

Преобразователи кодов – это узлы ЭВМ, предназначенные для кодирования чисел. В число преобразователей кодов входят: двоично-десятичные преобразователи, преобразователи цифровой индикации, преобразователи прямого кода двоичных чисел в обратный или дополнительный код и т. д.

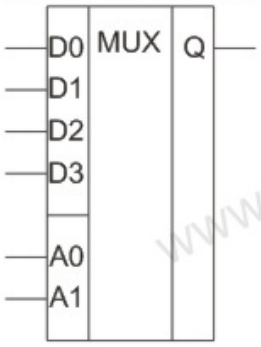
преобразователь кода и таблица его функционирования



Xi	W	Yi
0	0	0
1	0	1
0	1	1
1	1	0

Мултиплексоры – это узлы, преобразующие параллельные цифровые коды в последовательные. В этом устройстве выход соединяется с одним из входов в зависимости от значения адресных входов. Мультиплексоры широко используются для синтеза комбинационных устройств, так как это способствует значительному уменьшению числа используемых микросхем.

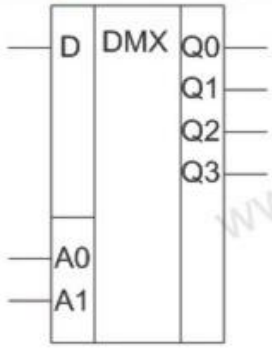
мультиплексор и его таблица функционирования



A1	A0	Выход
0	0	Q=D0
0	1	Q=D1
1	0	Q=D2
1	1	Q=D3

Демультиплексоры – это узлы, преобразующие информацию из последовательной формы в параллельную. Информационный вход D подключается к одному из выходов Qi определяемый адресными сигналами A0 и A1.

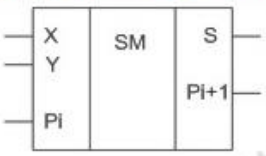
демультиплексор и его таблица функционирования



A1	A0	Q1	Q2	Q3	Q4
0	0	D	0	0	0
0	1	0	D	0	0
1	0	0	0	D	0
1	1	0	0	0	D

Сумматор – это узел, в котором выполняется арифметическая операция суммирования цифровых кодов двух двоичных чисел.

одноразрядный сумматор и таблица его функционирования



X	Y	Pi	S	Pi+1
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

Используя одноразрядные сумматоры можно построить многоразрядные сумматоры.

схема соединения одноразрядных полных сумматоров для получения трехразрядного сумматора