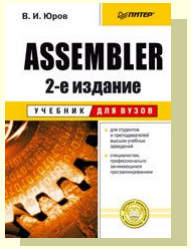
Литература:

Олифер В.Г., Олифер Н.А. Сетевые операционные системы: Учебник для вузов. – СПб.: Питер, 2006. (www.rus-lib.ru)



Юров В.И. Assembler: Учебник для вузов. 2-е изд. – СПб.: Питер, 2007.

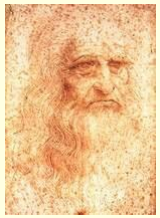
Эндрю Таненбаум Современные операционные системы. 2-е изд. – СПб.: Питер. 2007

Вильям Столлингс Операционные системы. – М.: Издательский дом «Вильямс», 2002.

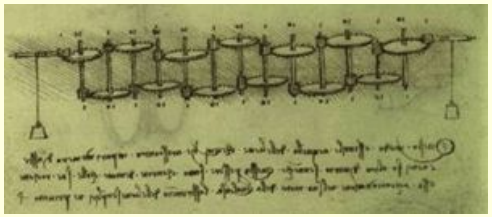
А.В. Гордеев, А.Ю. Молчанов Системное программное обеспечение. – СПб.: Питер, 2003

1. Основные принципы устройства и функционирования ЭВМ

1.1 Эволюция развития ЭВМ

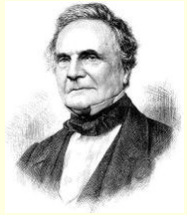


Леонардо да Винчи (**1452-1519** гг) – проект механической счетной машины, описанный в собрании рукописей Codex Madrid и Codex Atlanticus.



1623 – ученный Вильгельм Шиккард, «Часы для счета».

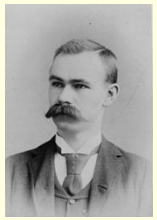
1642 – французский ученный и философ Блез Паскаль, первый арифмометр, выполнявший четыре основных действия.



1823 – английский ученный Чарльз Беббидж разрабатывает проект «Разностной машины» ставшей прообразом программно-управляемой машины.

1. Основные принципы устройства и функционирования ЭВМ

1.1 Эволюция развития ЭВМ



1880-е годы – американец Герман Холлерит разрабатывает машину, работающую с таблицами данных, машина управлялась программой записанной на перфокартах.

1938 – немецкий инженер-кибернетик Конрад Цузе создает механическую вычислительную машину Z-1 использующую, в отличии от предыдущих, двоичную систему счисления.
В **1941** создает электромеханическую вычислительную машину Z-3, основанную на телефонном реле, которая умела выполнять операции с плавающей точкой.

1944 – Говард Айкен, электромеханическая вычислительная машина MARK-1, которая оперировала десятичной системой счисления.

1945 – Джон фон Нейман разработал концепцию ЭВМ (EDVAC) с вводимыми в память программой и данными, главными элементами концепции были принцип хранимой программы и принцип па-раллельной организации вычислений. Сама машина была завершена в 1950 г.

1. Основные принципы устройства и функционирования ЭВМ

1.1 Эволюция развития ЭВМ

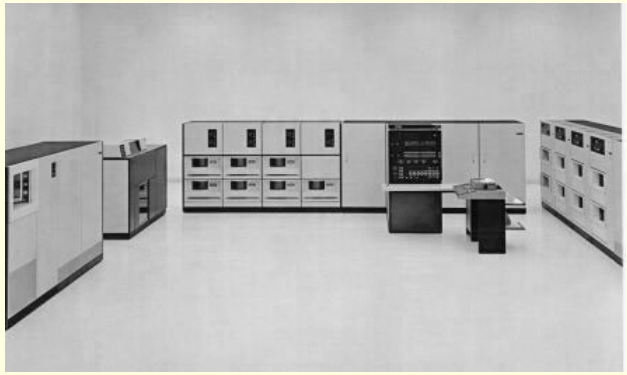
1946 – в США, в университете Пенсильвания, была создана первая универсальная ЭВМ – ENIAC, содержащая 18 тыс. ламп, весила 30 тонн, занимала площадь 200 м^2. В ней использовались десятичные операции, программирование производилось с помощью переключателей.

1945-1954 (1-е поколение ЭВМ) – время становление машин с фон-неймановской архитектурой. Машины 1-го поколения строились на ламповой элементной базе. Программы составлялись на языке Assembler.

1. Основные принципы устройства и функционирования ЭВМ

1.1 Эволюция развития ЭВМ

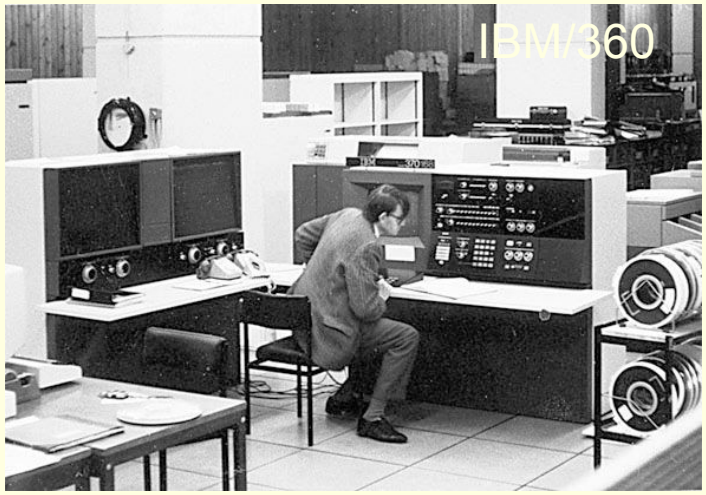
1955-1964 (2-е поколение ЭВМ). Вместо ламп и реле стали применяться транзисторы, в качестве ОП стали использоваться магнитные сердечники. В архитектуре ЭВМ стали использоваться процессоры ввода-вывода, позволявшими производить ввод-вывод информации одновременно с процессом вычисления. Появились трансляторы языков высокого уровня Algol, FORTRAN, COBOL. Для эффективного управления ресурсами ЭВМ стали впервые использоваться ОС.



1965-1970 (3-е поколение ЭВМ). В качестве элементной базы стали использоваться интегральные микросхемы. Появились недорогие и малогабаритные машины – мини-ЭВМ. Увеличение мощности ЭВМ сделало возможным одновременно выполнять несколько программ, что приводит к созданию более сложных многозадачных ОС. Наблюдается тенденция к созданию семейств ЭВМ, т.е. машины становятся совестимыми снизу вверх на программно-аппаратном уровне (пример серия IBM System 360, отечественный аналог серия ЕС).

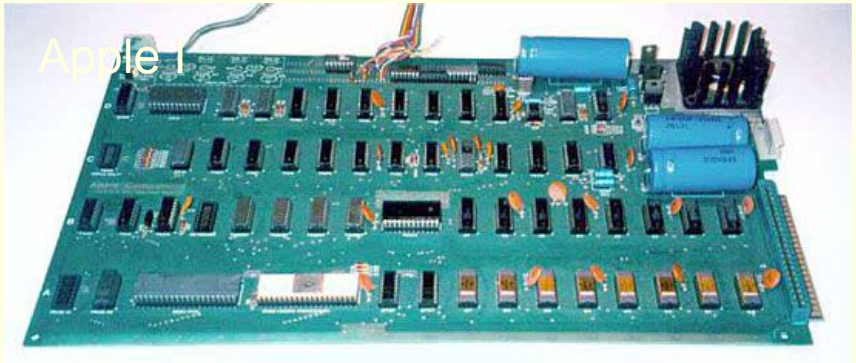
1. Основные принципы устройства и функционирования ЭВМ

1.1 Эволюция развития ЭВМ



1970-1984 (4-е поколение ЭВМ). Смена элементной базы на большие и сверхбольшие интегральные схемы (БИС и СБИС).

- **1971** – фирмой Intel был выпущен первый микропроцессор 4-х разрядный i4004. До этого было три направления развития ЭВМ: суперЭВМ, большие ЭВМ (мэйнфреймы) и мини-ЭВМ), теперь к ним прибавилось четвертое – микропроцессорное.
- **1972** – создание 8-ми разрядного МП i8008.
- **1975** – фирма MITS на основе i8008 начинает выпускать первый ПК Altair 8800.



1. Основные принципы устройства и функционирования ЭВМ

1.1 Эволюция развития ЭВМ

- **1981** – корпорация IBM на базе 16-ти разрядного процессора i8086 выпускает ПК IBM PC, который послужил начальной точкой развития одной из современных платформы ПК получившей название PC.
- **1983** – компания Apple выпускает первый ПК с графическим пользовательским интерфейсом Apple Lisa, который не завоевал большой популярности, но послужил прообразом более популярной линейки компьютеров Apple Macintosh.

С средины 1980-х (5-е поколение ЭВМ). ЭВМ с использованием принципиально новых технологий, т. к. управления потоками данных, элементы искусственного интеллекта

1. Основные принципы устройства и функционирования ЭВМ

1.2 Основные типы современных ЭВМ

- **СуперЭВМ и большие ЭВМ (мейнфрейм)** – уникальная сверхпроизводительная система используемая для решения сложнейших задач, требующих гигантский объемов вычислений.
- **Кластер** - это группа вычислительных машин (часто ПК), которые связаны между собою и функционируют как один узел обработки информации.
- **Сервер** – компьютер, предоставляющий свои ресурсы другим пользователям.
- **Персональный компьютер (ПК)** – компьютер, предназначенный для работы в условиях предприятия или дома, настройка, обслуживание и установка программного обеспечения выполняется самим пользователем.
- **Встраиваемая ЭВМ** – микроЭВМ, предназначенная в системах управления и контроля.

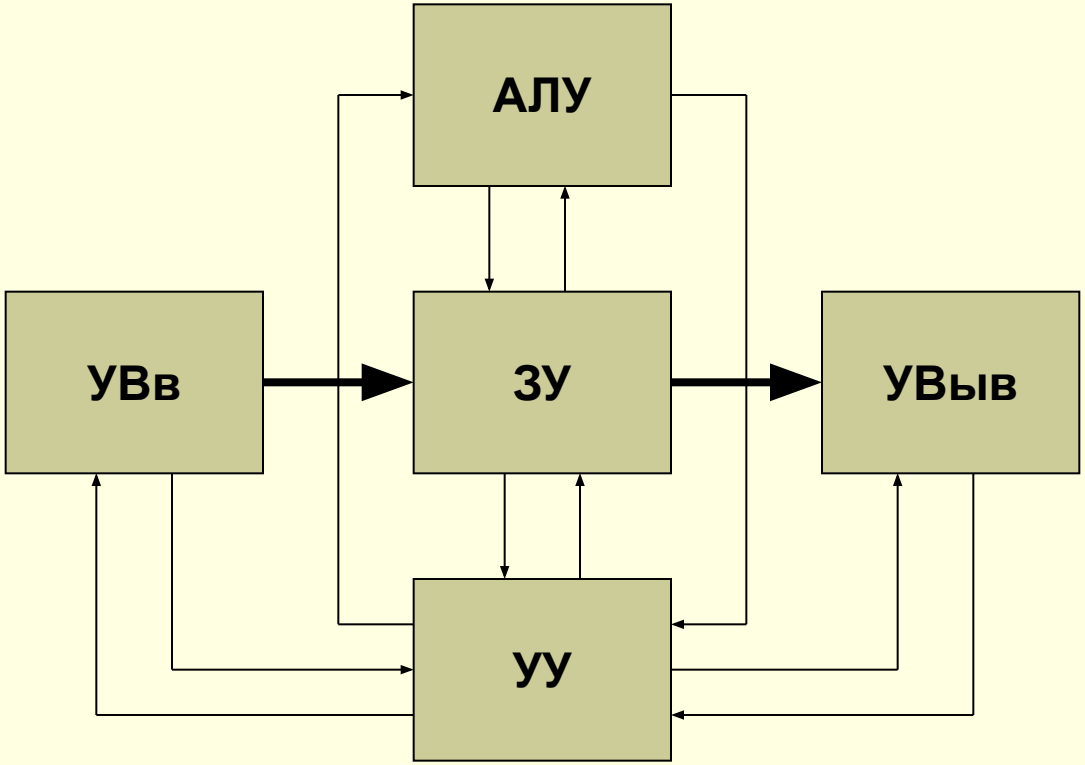
1. Основные принципы устройства и функционирования ЭВМ

1.2 Основные типы современных ЭВМ

- **Профессиональная рабочая станция** – специализированный компьютер, ориентированный на профессиональную деятельность в определенной области, оснащен дополнительным оборудование и специализированным программным обеспечением.
- **Сетевая рабочая станция (сетевой ПК)** - ПК который предназначен для функционирования в вычислительной сети, обычно настройка, техническая поддержка и установка программного на такой компьютер осуществляется централизованно.
- **Портативный компьютер (ноутбук)** – ПК, обладающий мощностью персонального компьютера, способный в течение определенного времени работать без подключения к электрической сети.
- **Карманный компьютер (КПК)** – компьютер, обладающий малыми размерами (не больше записной книжки) и способный функционировать автономно в течение длительного промежутка времени.
- **Смартфон** – телефон с функциями компьютера.
- **Коммуникатор** – КПК со встроенным GSM/GPRS модулем, позволяющим совершать телефонные звонки и выходить в сеть Internet.
- **Терминал** – устройство, которое не предназначено для работы в автономном режиме, выполняющие операции по вводу и передаче команд пользователя более мощному компьютеру и выдаче пользователю результата.

1. Основные принципы устройства и функционирования ЭВМ

1.3 Структура ЭВМ



- блок для выполнения арифметических и логических операций (**АЛУ** - арифметико-логическое устройство);
- блок для хранения информации или память (**ЗУ** – запоминающее устройство);
- устройство для ввода исходных данных (**УВв**) и для вывода результатов (**УВыв**);
- устройство для управления блоками ЭВМ (**УУ** – устройство управления)

1. Основные принципы устройства и функционирования ЭВМ

1.3 Структура ЭВМ

Элементы современных ЭВМ:

- **Центральный процессор (ЦП) -** устройство, непосредственно осуществляющее процесс обработки данных и программное управление этим процессом.
- **Оперативное запоминающие устройство (ОЗУ)** - предназначено для приема, хранения и выдачи информации необходимой для выполнения операций в ЦП.
- **ВЗУ** – запоминающие устройство, предназначенное для хранения больших массивов информации, обычно ВЗУ строятся на разновидностях магнитных носителей.

В современных ЭВМ в качестве ЗУ используется комбинация ОЗУ с ВЗУ, которая называется виртуальная память.

1. Основные принципы устройства и функционирования ЭВМ

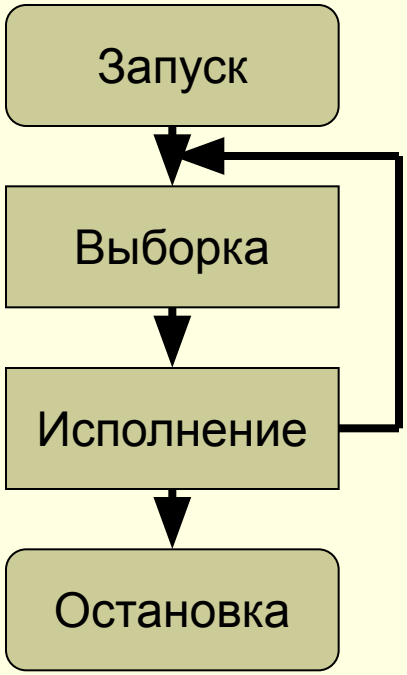
1.4 Основные свойства архитектуры ЭВМ

Общие архитектурные свойства современных ЭВМ:

- **Принцип хранимой программы.** Код программы и ее данные находятся в одном адресном пространстве в ОЗУ.
- **Принцип микропрограммирования.** Процессор имеет блок микропрограммного управления, этот блок для каждой машинной команды имеет набор действий-сигналов, который нужно сгенерировать для физического выполнения требуемой команды.
- **Линейное адресное пространство.** Совокупность ячеек памяти, которым последовательно присваиваются адреса.
- **Последовательное выполнение команд.** Процессор выбирает из памяти команды строго последовательно. Для изменения прямолинейного хода выполнения программы необходимо использовать специальные команды.
- **Безразличие к целевому назначению данных.** С точки зрения процессора нет принципиальной разницы между данными и командами. Данные и команды находятся в одном адресном пространстве в виде последовательности нулей и единиц. Машине все равно, какую логическую нагрузку несут эти данные.

1. Основные принципы устройства и функционирования ЭВМ

1.5 Архитектура и основные свойства современных ЦП



Выполнение команд процессор включает:

- выборка очередной команды из памяти;
- декодирование команды;
- генерация адреса, при которой определяются адреса аргументов команды;
- выполнение команды с помощью АЛУ;
- запись результатов.

1. Основные принципы устройства и функционирования ЭВМ

1.5 Архитектура и основные свойства современных ЦП

Архитектура - абстрактное понятие, которое отражает структурную, схемотехническую и логическую организацию ЦП .

Основные архитектуры:

- **CISC (Complete Instruction Set Computing)** – процессоры с полным набором команд;
- **RISC (Reduced Instruction Set Computer)** – процессоры с сокращенным набором команд, ориентированный на быстрое и эффективное выполнение относительно небольшого набора команд, в отличие от команд процессоров CISC-архитектуры, все команды имеют фиксированную длину, за счет чего пропускается этап генерации адреса для определения адресов аргументов и значительно повышается скорость выполнения.

1. Основные принципы устройства и функционирования ЭВМ

1.5 Архитектура и основные свойства современных ЦП

Особенности архитектуры современных ЦП:

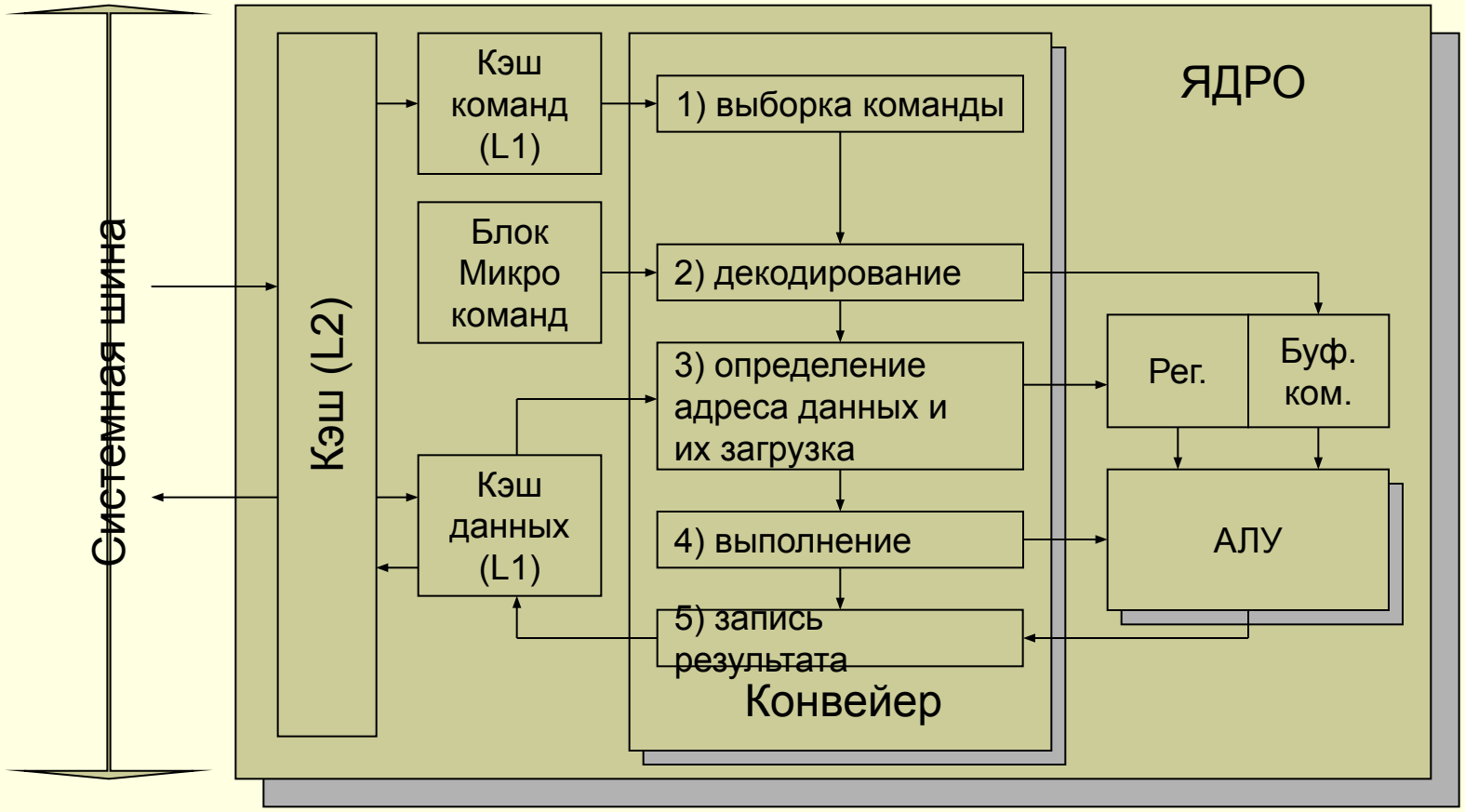
- **Суперскалярная архитектура** – в основе архитектуры лежит принцип **конвейеризации вычислений**.

*Важным элементом суперскалярной архитектуры является **конвейер** – специальное устройство, реализующее такой метод обработки команд внутри процессора, при котором исполнение команды разбивается на несколько этапов. При этом очередная команда после выборки попадает на декодирование. Таким образом, блок выборки свободен и может выбирать следующую команду. В результате на конвейере могут находиться несколько команд в разной стадии выполнения. Процессоры, имеющие один конвейер, называются **скалярными**, два и более – **суперскалярными**.*

- **Несколько АЛУ** и специализированные вычислительные блоки.
- **Несколько уровней кэш-памяти**, значительно ускоряющей обмен информацией между ЦП и ОЗУ .
- **Много ядерная архитектура**, когда на одном кристалле располагаются несколько процессоров.

1. Основные принципы устройства и функционирования ЭВМ

1.5 Архитектура и основные характеристики современных ЦП



Структура ЦП серии P6

1. Основные принципы устройства и функционирования ЭВМ

1.5 Архитектура и основные свойства современных ЦП

Основные характеристики ЦП:

- **Тактовая частота**

Такт *- сигнал фиксированной продолжительность, используемый для синхронизации работ устройств ЭВМ.*

- **Разрядность шины данных** определяется, какой объем информации может обрабатывать МП за один такт, измеряется в битах.
- **Разрядность адресной шины** емкость адресуемой МП памяти.
- **Количество уровней и размерность кэш-памяти.**

Кэш память МП *- промежуточная сверхскоростная оперативная память являющейся буфером между контроллером сравнительно медленной системной памяти и процессором .*

- **Количество ядер в МП.**

1. Основные принципы устройства и функционирования ЭВМ

1.6 Оперативное запоминающие устройство

ОЗУ - предназначена для приема, хранения и выдачи информации (чисел, символов, команд, констант), т.е. всей информации необходимой для выполнения операций в ЦП. ОЗУ представляет собой набор однотипный ячеек (в роли последних выступают конденсаторы или триггеры), которые хранят информацию в виде электрических импульсов. Каждая из ячеек имеет свой уникальный адрес.

Основные типы памяти:

- **Статическая память** – в качестве носителя информации используются триггеры, которые могут иметь одно из двух состояний.
- **Динамическая память** – в качестве носителя информации используются конденсаторы.

Основные характеристики ОЗУ:

- **Емкость** - характеризуется максимальный объем информации, которую может хранить память.
- **Время доступа -** время за которое ЦП получает требуемую информацию**.**

1. Основные принципы устройства и функционирования ЭВМ

1.7 Внешнее запоминающие устройство

ВЗУ - применяются для хранения больших объемов информации, которые не используются в данный момент времени процессором, информация на таких устройствах хранится блоками, каждый из которых имеет свой уникальный адрес.

Основные типы ВЗУ:

- **Произвольного доступа** – позволяют обратиться к любому произвольному блоку хранимой информации (винчестеры, накопители на оптических дисках и т.д.).
- **Последовательного доступа** – позволяют получить доступ к нужному блоку информации только после прочтения всех предыдущих блоков (накопители на магнитной ленте).

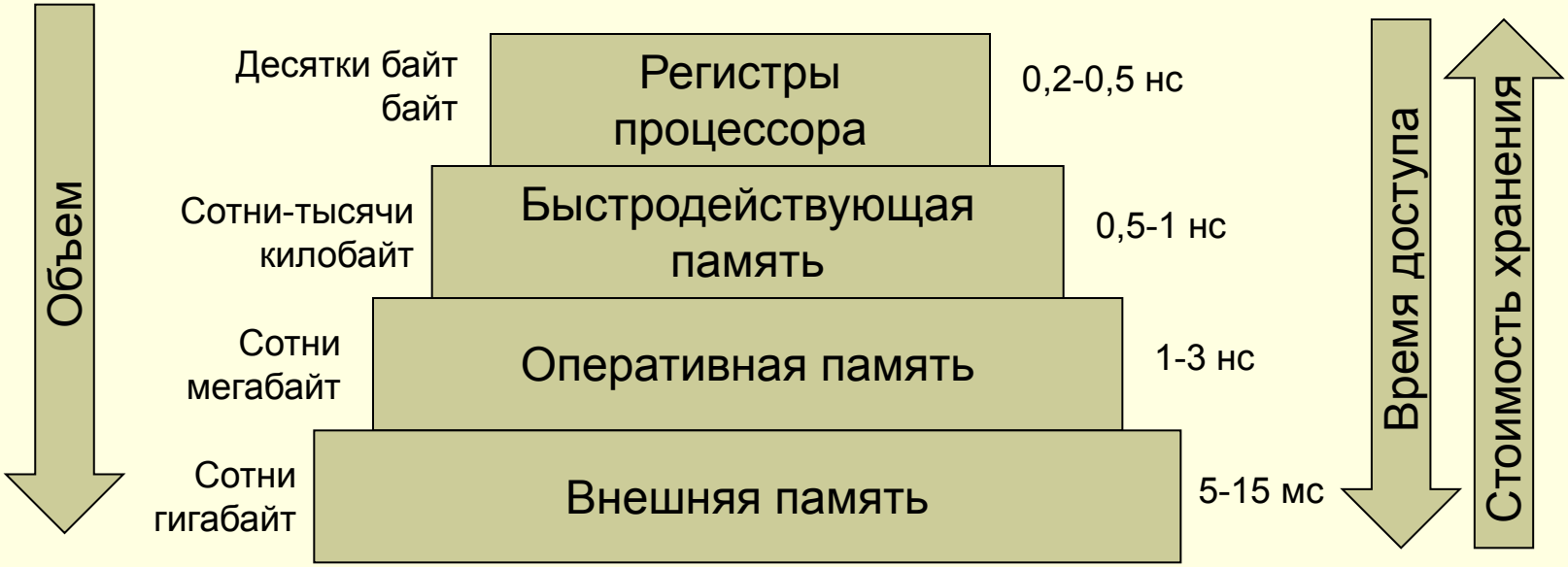
Основные характеристики ВЗУ:

- **Емкость** - характеризуется максимальный объем информации, которую может хранить ВЗУ.
- **Размер кэш-памяти.**
- **Время доступа -** время за которое ЦП получает требуемую информацию**.**

1. Основные принципы устройства и функционирования ЭВМ

1.8 Кэш

Память вычислительной машины представляет собой иерархию запоминающих устройств (ЗУ), отличающихся средним временем доступа к данным, объемом и стоимостью хранения одного бита.



Иерархия запоминающих устройств

1. Основные принципы устройства и функционирования ЭВМ

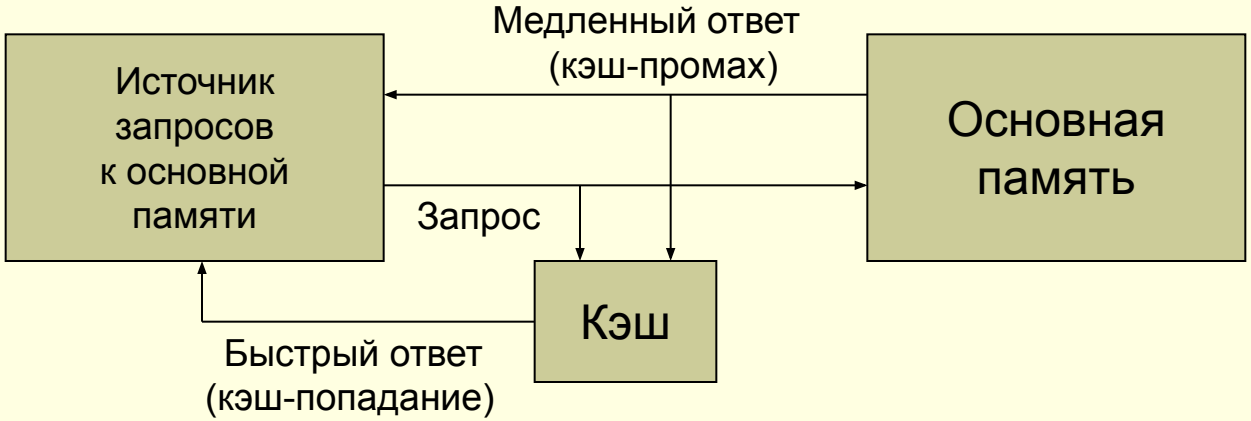
1.8 Кэш

Кэш память (кэш) - это способ совместного функционирования двух типов запоминающих устройств, отличающихся временем доступа и стоимостью хранения данных, который за счет динамического копирования в «быстрое» ЗУ наиболее часто используемой информации из «медленного» ЗУ позволяет, с одной стороны, уменьшить среднее время доступа к данным, а с другой стороны, экономить более дорогую быстродействующую память.

Кэширование — это универсальный метод, пригодный для ускорения доступа к оперативной памяти, к диску и к другим видам запоминающих устройств.

1. Основные принципы устройства и функционирования ЭВМ

1.8 Кэш



Принцип работы кэш-памяти

При каждом обращении к основной памяти по физическому адресу просматривается содержимое кэш-памяти с целью определения, не находятся ли там нужные данные. Кэш-память не является адресуемой, поэтому поиск нужных данных осуществляется по содержимому — по взятому из запроса значению поля адреса в оперативной памяти. Далее возможен один из двух вариантов развития событий:

если данные обнаруживаются в кэш-памяти, то есть произошло **кэш-попадание**, они считываются из нее и результат передается источнику запроса;

если нужные данные отсутствуют в кэш-памяти, то есть произошел **кэш-промах**, они считываются из основной памяти, передаются источнику запроса и одновременно с этим копируются и кэш-память.

1. Основные принципы устройства и функционирования ЭВМ

1.8 Кэш

Высокое значение вероятности нахождения нужных данных в кэш-памяти объясняется наличием у данных двух объективных свойств:

- **Временная локальность**. Если произошло обращение по некоторому адресу, то следующее обращение по тому же адресу с большой вероятностью произойдет в ближайшее время.

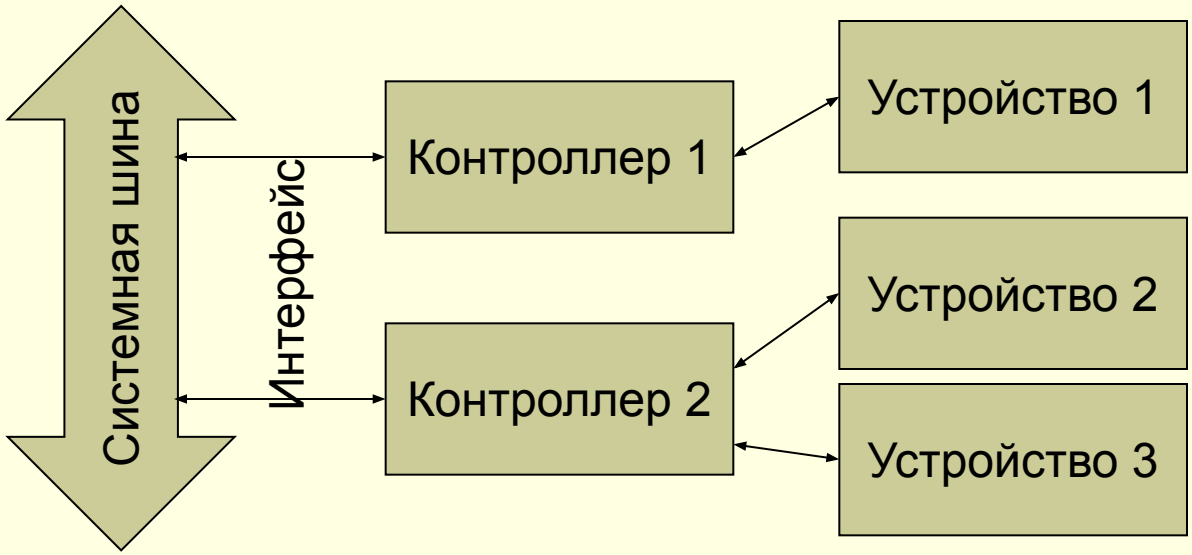
- **Пространственная локальность.** Если произошло обращение по некоторому адресу, то с высокой степенью вероятности в ближайшее время произойдет обращение к соседним адресам.

1. Основные принципы устройства и функционирования ЭВМ

1.9 Устройства ввода-вывода

К устройствам ввода-вывода относятся: клавиатура, мышь, монитор, принтер, сканер, модем, сетевая карта и т.д. Любое устройство управляется контроллером.

Контроллер – блок ЭВМ предназначенный для управления внешними устройствами или группой внешних устройств.



1. Основные принципы устройства и функционирования ЭВМ

1.9 Устройства ввода-вывода

Интерфейс представляет собой совокупность стандартизованных аппаратных и программных средств, обеспечивающих обмен информацией между устройствами. В основе построения интерфейсов лежат унификация и стандартизация (использование единых способов кодирования данных, форматов данных, стандартизация соединительных элементов - разъемов и т. д.). Наличие стандартных интерфейсов позволяет унифицировать передачу информации между устройствами независимо от их особенностей, и производителей устройств.

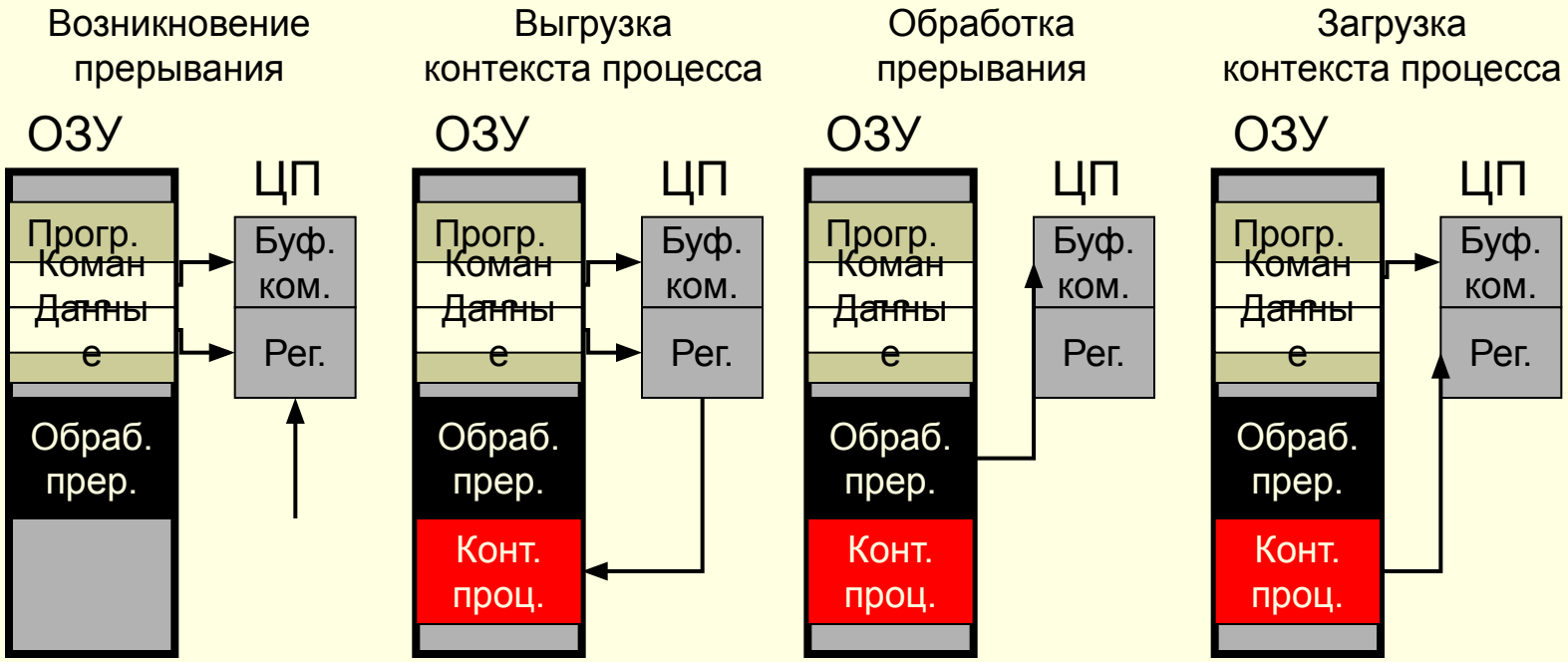
Интерфейсы устройств ЭВМ делятся на две категории:

- **Последовательные интерфейсы -** информация передается последовательно по битам (интерфейсы стандартов COM и USB).
- **Параллельные интерфейсы** - данные предаются порцией за раз по параллельным каналам, каждый бит по отдельному каналу (LPT и ATA).

1. Основные принципы устройства и функционирования ЭВМ

1.10 Прерывания

Прерывания – механизм ЭВМ, используемый для выполнения незапланированных действий, который прерывает выполнение основного потока команд и переводит процессор на выполнение потока команд для обработки прерывания с последующим возвратом к исходным командам.



Механизм обработки прерывания

1. Основные принципы устройства и функционирования ЭВМ

1.10 Прерывания

В зависимости от источника прерывания делятся на три класса:

- **Внешние (аппаратные) прерывания** – возникают вследствие подачи некоторой аппаратурой (например, контроллером принтера) электрического сигнала, который передается (возможно, проходя через другие блоки компьютера, например контроллер прерываний) на специальный вход прерывания процессора. Внешние прерывания обслуживаются драйверами устройств.
- **Внутренние прерывания (исключения)** - возникают при появлении аварийной ситуации в ходе исполнения некоторой инструкции программы и обрабатываются специальными модулями ядра.
- **Программные прерывания** - возникают при выполнении особой команды процессора, выполнение которой имитирует прерывание.

1. Основные принципы устройства и функционирования ЭВМ

1.10 Прерывания

Существую два способа реализации прерываний:

- **векторный** – в ЦП передается информация об уровне приоритета прерывания, а так же информация о начальном адресе программы обработчика возникшего прерывания;
- **опрашиваемый –** в ЦП передается только приоритет прерывания, ЦП самостоятельно определяет каким устройством вызвано прерывание путем вызова всех обработчиков прерывания для данного уровня приоритета, пока один из обработчиков не подтвердит что прерывание пришло из обслуживаемого им устройства.

1. Основные принципы устройства и функционирования ЭВМ

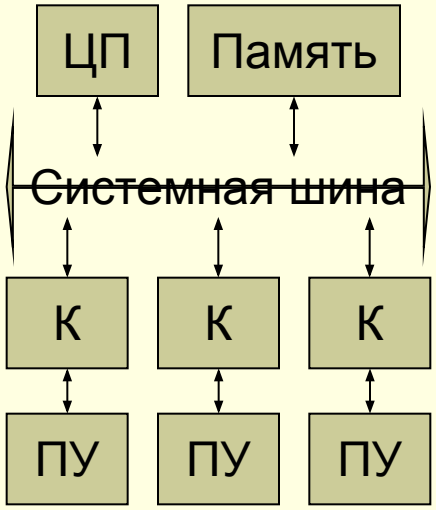
1.10 Прерывания

Маскирование прерываний – способ обработки нескольких одновременно возникших прерываний, при котором обслуживаются прерывания с наибольшим приоритетом, остальные прерывания игнорируются (маскируются).

1. Основные принципы устройства и функционирования ЭВМ

1.10 Структура ПК

Современные ПК построенные на принципе **открытой архитектуры**. При использование открытой архитектуры производитель ПК должен придерживаться общеизвестных стандартов на интерфейсы устройств ПК.



***ЦП** – центральный процессор;*

***К** – контроллер;*

***ПУ** – периферийное устройство*

В ПК используется структура с одним общим интерфейсом, называемым **системной шиной**. При такой структуре все устройства компьютера обмениваются информацией и управляющими сигналами через общий интерфейс - **системную шину**.

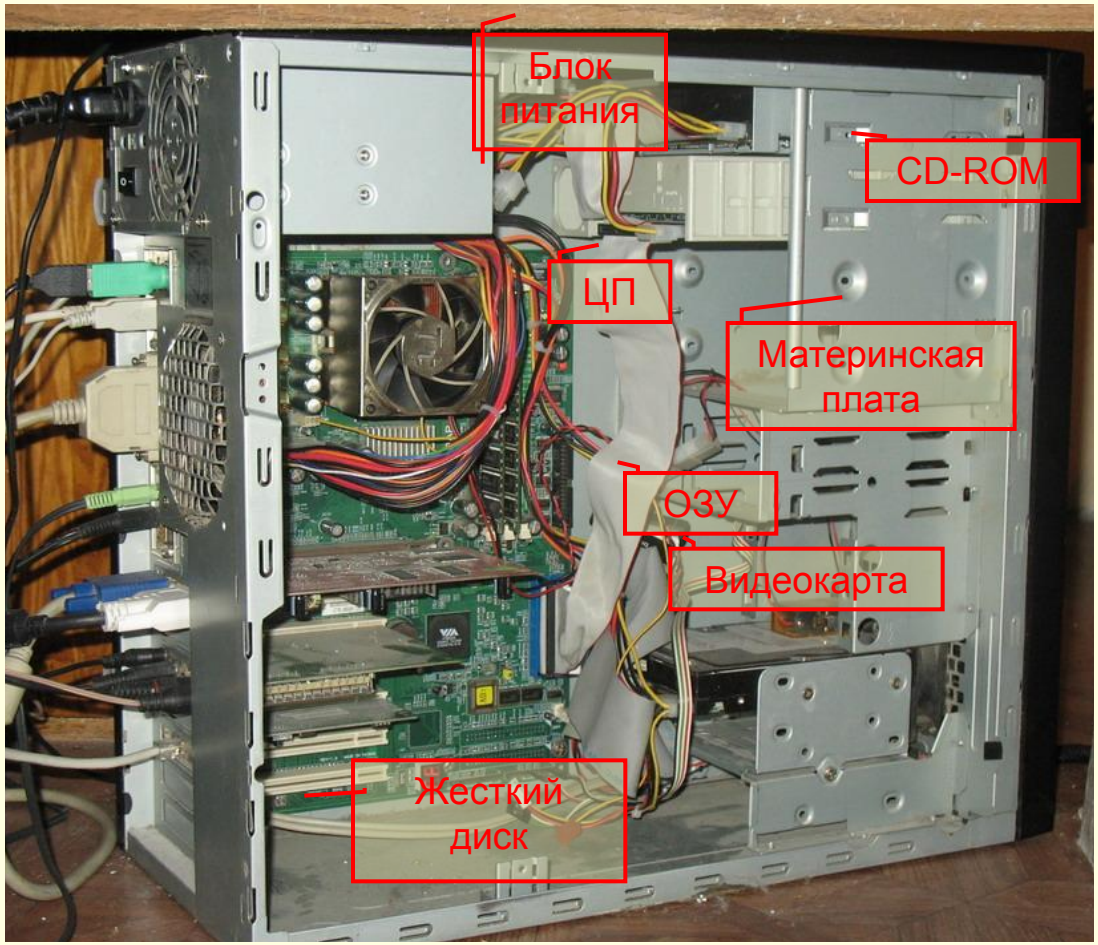
Все устройства ПК подключаются через разъемы (шины), которые соединены с системной или локальной шиной.

Основная электронная часть ПК конструктивно располагается в системном блоке выполняющая функции механического и электрического соединения компонентов ЭВМ по средствам системной шины, называется **системной (материнской) платой**.

Большая часть устройств системной платы помещена в одну или несколько больших микросхем, называемых **набором микросхем (chipset)**.

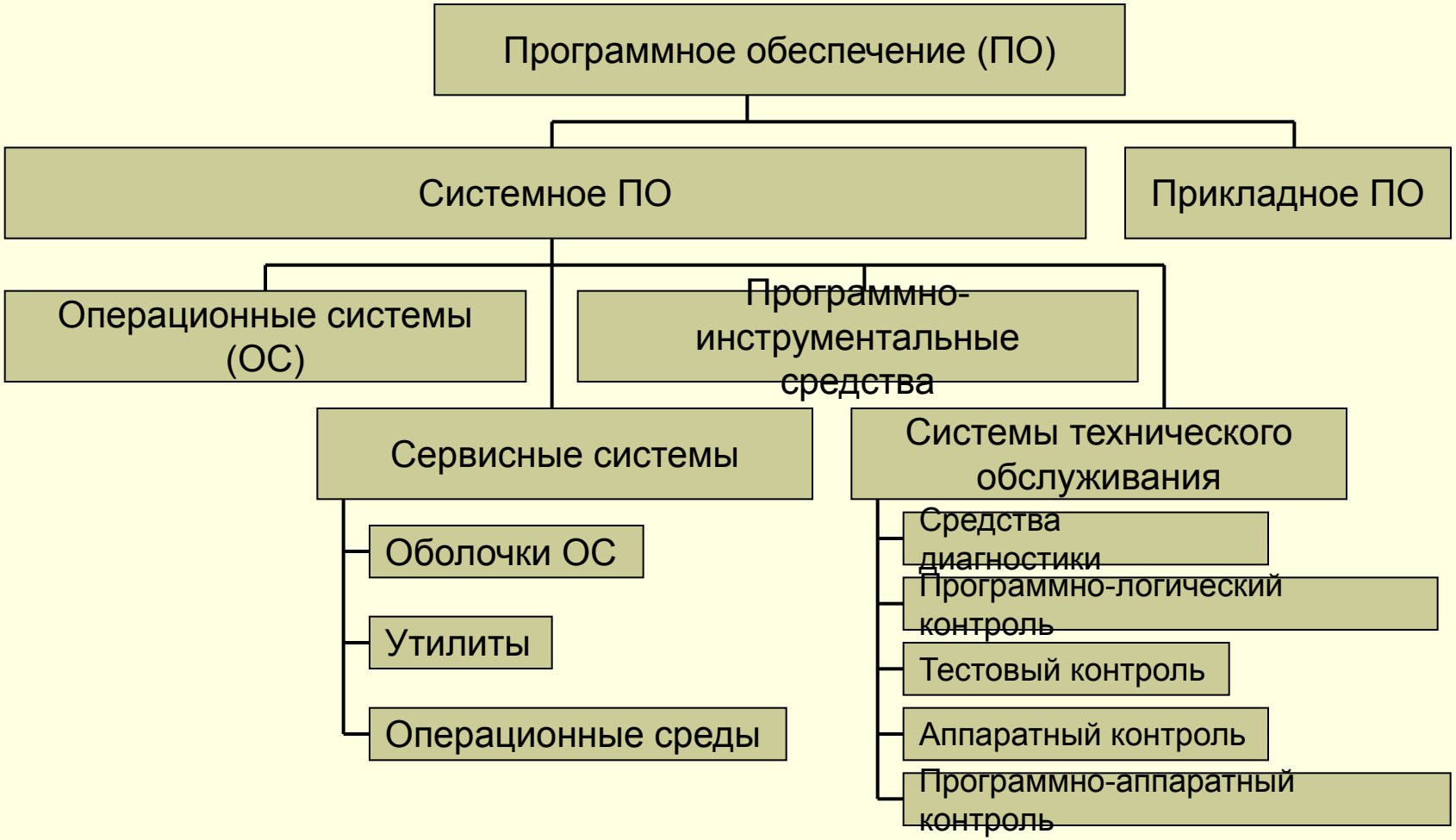
1. Основные принципы устройства и функционирования ЭВМ

1.10 Структура ПК



2. Программное обеспечение

2.1 Классификация программного обеспечения



2. Программное обеспечение

2.1 Классификация программного обеспечения

Программа – формализованное описание последовательности действий устройств компьютера по реализации той или иной задачи. Совокупность программ и сопровождающей их документации, предназначенная для решения задач на ПК, называется **программным обеспечением (ПО) (software).**

Системное программное обеспечение предназначено для управления компьютером, создания и поддержки выполнения программ пользователя, а также для предоставления пользователю набора всевозможных услуг.

Прикладное программное обеспечение предназначено для решения определенных классов задач пользователя.

2. Программное обеспечение

2.1 Классификация программного обеспечения

Системное ПО делится на:

Операционные системы. Операционная система (ОС) - совокупность программ, управляющих работой всех устройств ПК, процессом выполнения прикладных программ и взаимодействием пользователя с ПК.

Сервисные системы расширяют возможности ОС, предоставляя пользователю, а также выполняемым программам набор разнообразных дополнительных услуг. К сервисным системам относят:

- **Оболочки ОС** - это программные продукты, которые делают общение пользователя с компьютером более комфортным.
- **Утилиты** - это служебные программы, которые предоставляют пользователю ряд дополнительных услуг.
- **Операционные среды**. **Операционная среда** - полнофункциональная надстройка над ОС, которая формирует новую среду выполнения программ и выполняет все функции оболочки.

2. Программное обеспечение

2.1 Классификация программного обеспечения

Системы технического обслуживания – совокупность программно-аппаратных средств ПК для обслуживания сбоев в процессе работы ПК. Эти средства можно разделить на пять категорий:

- **средства диагностики** – обеспечивают автоматический поиск ошибок и выявление неисправностей с их локализацией;
- **программно-логический контроль** – основан на использовании избыточного кода исходных и промежуточных данных ПК, что позволяет находить ошибки при изменении отдельных битов информации;
- **тестовый контроль** – осуществляется с помощью специальных тестов для проверки правильности работы ПУ или его устройств;
- **аппаратный контроль** – ведется автоматически с помощью встроенного в ПК оборудования;
- **программно-аппаратный контроль –** включает программный и аппаратный контроль.

2. Программное обеспечение

2.1 Классификация программного обеспечения

Программно-инструментальные средства - это программные продукты, предназначенные для разработки программного обеспечения.

Языки программирования делятся на два класса:

- **Машинно-зависимые языки (низкого уровня)** – являются внутренними языками ЭВМ и представляет собой систему инструкций и данных, которые не требуют преобразования и могут непосредственно интерпретироваться и исполняться аппаратными средствами ЭВМ.
- **Машинно-независимые языки (высокого уровня)** – не требуют от пользователя полного знания архитектуры ЭВМ на котором реализуется программа и позволяют пользователю записывать программу в виде команд близких к человеческому пониманию, которые пред выполнением программы преобразуются в команды ЭВМ.

Процесс преобразования команд языков высокого уровня в команда языка низкого уровня называется **трансляцией**, соответственно программа выполняющие это действие – **транслятор.** Работа трансляторов строится по одному из двух принципов:

- **Интерпретация** подразумевает пооператорную трансляцию и последующее выполнение оттранслированного оператора исходной программы.
- **Компиляция** разделяется на два этапа: сначала программа полностью переводится на машинный язык, а затем оттранслированная программа может многократно выполняться.

2. Программное обеспечение

2.1 Классификация программного обеспечения

Существуют современные языки программирования использующие обе разновидности трансляторов, к примеру Java. В таких языках исходный текст программы компилируется в специальный низкоуровневый двоичный код – **байт-код**, который интерпретируется при непосредственном исполнении. Для работы бай-кода на ЭВМ необходимо наличие **виртуальной машины**, которая будет выполнять функции транслятора байт-кода в код ЦП, на котором будет исполняться программа.

Достоинство такого метода – это **кроссплатформенность** программы, т.е. способность программы работать на любом семействе ЭВМ, где есть необходимая виртуальная машина. Недостаток – снижение быстродействия за счет необходимости преобразования байт-кода в код ЦП.

2. Программное обеспечение

2.1 Развитие системного ПО

В развитие которого можно выделить следующие этапы.

До 50-х годов ЭВМ использовались только создателями и программирование производилось исключительно на машинном языке, все задачи организации вычислительного процесса решались вручную программистом.

В 50-е годы, с ростом вычислительно мощности ЭВМ, заметный рост стал наблюдаться и в области автоматизации программирования и организации вычислительных работ. В эти годы появились первые алгоритмические языки, и таким образом к библиотекам математических и служебных подпрограмм добавился новый тип системного программного обеспечения – **трансляторы**.

Выполнение каждой программы стало включать большое количество вспомогательных работ: загрузка нужного транслятора, запуск транслятора и получение результирующей программы в машинных кодах, связывание программы с библиотечными подпрограммами, загрузка программы в оперативную память, запуск программы, вывод результатов. Для организации эффективной работы в штат многих вычислительных центров были введены должности **операторов**, профессионально выполнявших работу по организации вычислительного процесса для всех пользователей этого центра.

2. Программное обеспечение

2.1 Развитие системного ПО

Для решения проблемы автоматизации вычислительного процесса были разработаны **системы пакетной обработки** (наиболее известные IBSYS, SAGE, SABRE, MERCURY), которые автоматизировали всю последовательность действий оператора по организации вычислительного процесса. Ранние системы пакетной обработки явились прообразом современных операционных систем, они стали первыми системными программами, предназначенными не для обработки данных, а для управления вычислительным процессом.

В ходе реализации систем пакетной обработки был разработан формализованный **язык управления заданиями**, с помощью которого программист сообщал системе и оператору, какие действия и в какой последовательности он хочет выполнить на вычислительной машине.

Оператор составлял **пакет заданий**, которые в дальнейшем без его участия последовательно запускались на выполнение управляющей программой — **монитором**. Кроме того, монитор был способен самостоятельно обрабатывать наиболее часто встречающиеся ошибки.

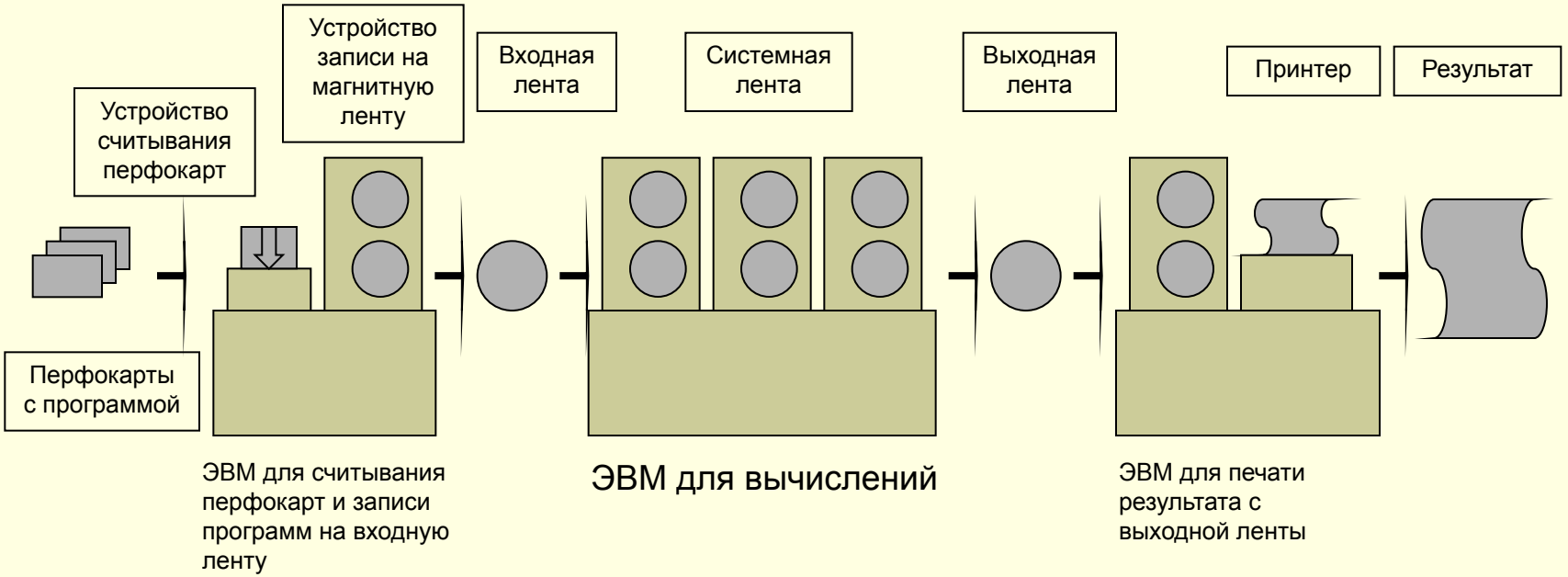
Пакет обычно представлял собой набор перфокарт, но для ускорения работы он мог переноситься на более удобный и емкий носитель, например на магнитную ленту или магнитный диск.

Системы пакетной обработки заданий сокращали затраты времени на вспомогательные действия по организации вычислительного процесса, но их недостатком было то, что программист-пользователь лишался непосредственного доступа к компьютеру, что снижало эффективность работы.

2. Программное обеспечение

2.1 Развитие системного ПО

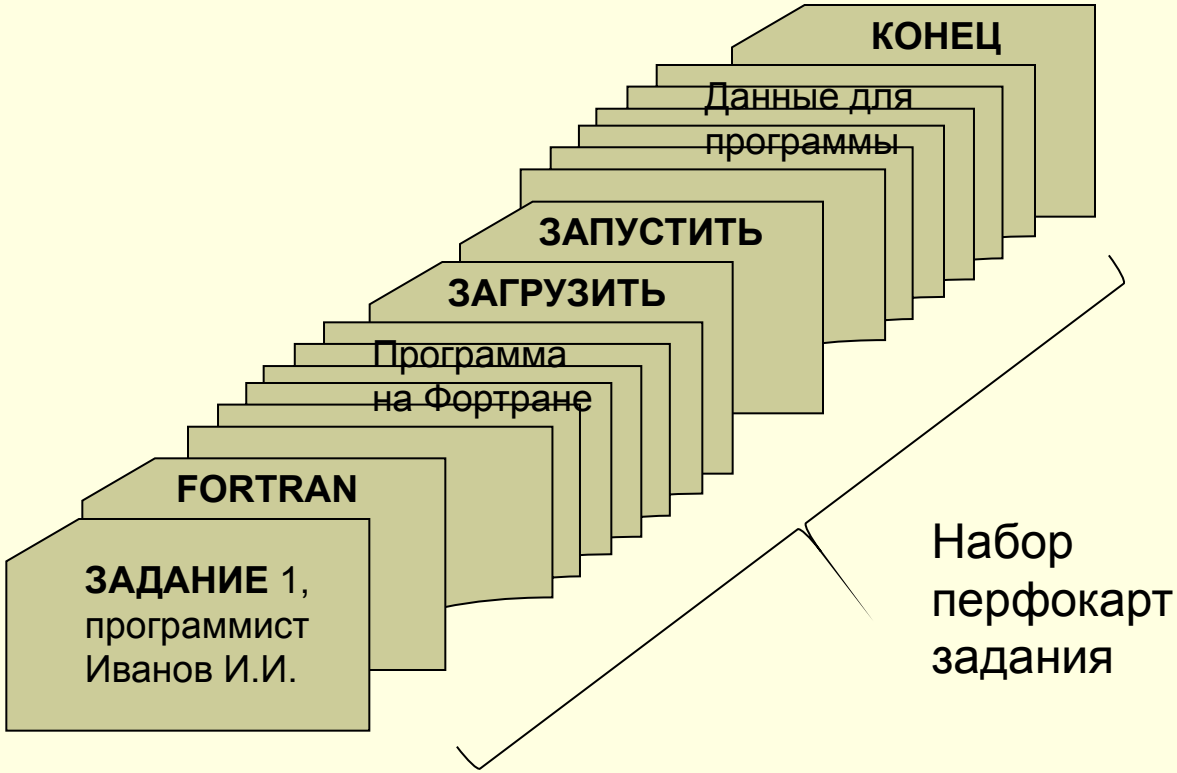
Процесс выполнение программы в системе пакетной обработки



2. Программное обеспечение

2.1 Развитие системного ПО

Структура типичного задания



2. Программное обеспечение

2.1 Развитие системного ПО

В период 1965-70 годов были реализованы практически все основные механизмы, присущие современным ОС.

В условиях резко возросших возможностей ЭВМ выполнение только одной программы в каждый момент времени оказалось крайне неэффективным. Решением стало **мультипрограммирование** — способ организации вычислительного процесса, при котором в памяти компьютера находилось одновременно несколько программ, попеременно выполняющихся на одном процессоре.

Мультипрограммирование было реализовано в следующих вариантах:

- **Мультипрограммные системы пакетной обработки** имели своей целью обеспечение максимальной загрузки аппаратуры компьютера. В этом режиме процессор не простаивал, пока одна программа выполняла операцию ввода-вывода, ЭВМ переключался на другую готовую к выполнению программу. В результате достигалась сбалансированная загрузка всех устройств компьютера.
- Вариант **систем разделения времени** рассчитан на многотерминальные системы, когда каждый пользователь интерактивно работает за своим терминалом. Такой вариант мультипрограммирования был нацелен на создание для каждого отдельного пользователя иллюзии единоличного владения вычислительной машиной за счет периодического выделения каждой программе своей доли процессорного времени. В системах разделения времени эффективность использования оборудования ниже, чем в системах пакетной обработки, что явилось платой за удобства работы пользователя. Примерами систем разделения времени являлись ОС TSS/360, CTSS и MULTICS.

2. Программное обеспечение

2.1 Развитие системного ПО

Реализация мультипрограммирования потребовала внесения очень важных изменений в аппаратуру компьютера. В процессорах появился **привилегированный** и **пользовательский** режимы работы, специальные регистры для быстрого переключения одной программы на другую, средства защиты областей памяти, а также развитая система прерываний.

В **привилегированном режиме**, предназначенном для работы программных модулей ОС, процессор мог выполнять все команды, в том числе те из них, которые позволяли осуществлять распределение и защиту ресурсов компьютера. Программам, работающим в **пользовательском режиме**, некоторые команды процессора были недоступны. Так же для разделения периферийных устройств между одновременно работающими программами была введена **система прерываний**.

Еще одной важной тенденцией этого периода является создание **семейств программно-совместимых машин и ОС** для них. Примерами семейств программно-совместимых машин, являются серии машин IBM/360 и IBM/370 (аналоги этих семейств советского производства — машины серии ЕС), PDP-11 (СМ). Вскоре идея программно-совместимых машин стала общепризнанной.

2. Программное обеспечение

2.2 Развитие системного ПО

В 1969 году Министерство обороны США инициировало работы по объединению суперкомпьютеров оборонных и научно-исследовательских центров в единую сеть. Эта сеть получила название **ARPANET** и явилась отправной точкой для создания самой известной ныне глобальной сети — **Интернета**.

В начале 70-х годов появились первые **сетевые операционные системы**, которые позволяли организовать распределенное хранение и обработку данных между несколькими компьютерами, связанными электрическими связями.

К середине 70-х годов наряду с мэйнфреймами широкое распространение получили мини-компьютеры, такие как PDP-11, Nova, HP, которые использовали преимущества больших интегральных схем, позволившие реализовать достаточно мощные функции при сравнительно невысокой стоимости компьютера. Архитектура мини-компьютеров была значительно упрощена по сравнению с мэйнфреймами, что нашло отражение и в их ОС. Многие функции мультипрограммных многопользовательских ОС мэйнфреймов были усечены, учитывая ограниченность ресурсов мини-компьютеров. Эти ОС не всегда были многопользовательскими, что во многих случаях оправдывалось невысокой стоимостью компьютеров.

2. Программное обеспечение

2.2 Развитие системного ПО

С появлением мини-компьютеров для них была создана ОС **UNIX**. Первоначально эта ОС предназначалась для поддержания режима разделения времени в мини-компьютере PDP-7. С середины 70-х годов началось массовое использование ОС **UNIX** для мини-компьютеров. К этому времени программный код для ОС UNIX был на 90 % написан на языке высокого уровня С. Широкое распространение эффективных С-компиляторов сделало UNIX уникальной для своего времени ОС, обладающей возможностью сравнительно легкого переноса на различные типы компьютеров, т.е. имевшей одну из важных характеристик ОС – **мобильность.**

С начала 80-х компьютеры стали широко использоваться неспециалистами, что потребовало разработки «дружественного» программного обеспечения, и предоставление этих «дружественных» функций стало прямой обязанностью ОС.

Первая версия наиболее популярной ОС раннего этапа развития ПК — **MS-DOS** компании Microsoft — была лишена всех выше перечисленных особенностей. Это однопрограммная однопользовательская ОС с интерфейсом командной строки способная стартовать с дискеты. Основными задачами для нее были управление файлами, расположенными на гибких и жестких дисках, а также поочередный запуск программ. MS-DOS не была защищена от программ пользователя, так как процессор Intel 8088 не поддерживал привилегированного режима. Разработчики первых ПК считали, что при индивидуальном использовании компьютера и ограниченных возможностях аппаратуры нет смысла в поддержке мультипрограммирования.

2. Программное обеспечение

2.2 Развитие системного ПО

Недостающие функции для MS-DOS и подобных ей ОС компенсировались внешним программами, предоставлявшими пользователю удобный интерфейс (например, Norton Commander) или средства тонкого управления дисками ,например, PC Tool).

Первой широко распространенная ОС с «дружественным» графическим интерфейсом считается ОС **MacOS** для компьютеров фирмы Apple. С появлением этой ОС фирма Microsoft также приступила к разработке ОС с графическим интерфейсом. Как результатом в 1991 году была выпущена операционная среда **Windows 3.0**, которая являлась надстройкой над ОС MS-DOS, предоставляя пользователю графический интерфейс и многозадачную среду для выполнения приложений.

В 1991 году Финским студентом Линусом Торвальдсом разрабатывается ядро ОС **Linux**, которое является свободно распространяемым клоном ОС UNIX. На основе этого множество программистов и компаний по всему миру разрабатывают свои варианты (**дистрибутивы**) ОС Linux.

В конце 90-х особое внимание стало уделяться **корпоративным ОС**, которые отличаются способностью хорошо и устойчиво работать в крупных сетях, характерных для больших предприятий, имеющих отделения в десятках городов и, возможно, в разных странах. К настоящему времени достаточно явно определились лидеры в классе корпоративных ОС — это ОС семейства Microsoft Windows NT (Windows 2000, XP, 2003), ОС семейства Linux, а также UNIX-системы различных производителей аппаратных платформ.

2. Программное обеспечение

2.3 Назначение и функции локальной ОС

ОС выполняет две группы функций:

- **предоставление пользователю или программисту вместо реальной аппаратуры компьютера расширенной виртуальной машины**, с которой удобней работать и которую легче программировать;
- **повышение эффективности использования компьютера** путем рационального управления его ресурсами в соответствии с некоторым критерием.

К числу основных ресурсов современных вычислительных систем могут быть отнесены такие ресурсы, как процессоры, ЗУ, ПУ. Все ресурсы распределяются между процессами.

Процесс (задача) – программа в стадии выполнения.

Управление ресурсами включает решение следующих общих, не зависящих от типа ресурса задач:

- **планирование ресурса**, т.е. определение, какому процессу, когда и в каком количестве (если ресурс может выделяться частями) следует выделить данный ресурс;
- **удовлетворение запросов на ресурсы**;
- **отслеживание состояния и учет использования ресурса** — то есть поддержание оперативной информации о том, занят или свободен ресурс и какая доля ресурса уже распределена;
- **разрешение конфликтов между процессами**.

2. Программное обеспечение

2.3 Назначение и функции локальной ОС

Для решения этих общих задач управления ресурсами разные ОС используют различные алгоритмы, особенности которых и конечном счете и определяют облик ОС в целом, включая характеристики производительности, область применения и даже пользовательский интерфейс.

Функции операционной системы автономного компьютера обычно группируются либо в соответствии с типами локальных ресурсов, которыми управляет ОС, либо в соответствии со специфическими задачами.

Наиболее важными подсистемы ОС:

- **Управления процессами.** Для каждого вновь создаваемого процесса ОС генерирует системные информационные структуры, которые содержат данные о потребностях процесса в ресурсах вычислительной системы, а также о фактически выделенных ему ресурсах. В мультипрограммной ОС одновременно может существовать несколько процессов. Поскольку процессы часто одновременно претендуют на одни и те же ресурсы, то в обязанности ОС входит поддержание очередей заявок процессов на ресурсы, например очереди к процессору. Важной задачей операционной системы является защита ресурсов, выделенных данному процессу, от остальных процессов.

2. Программное обеспечение

2.3 Назначение и функции локальной ОС

- **Управления памятью.** Память является для процесса таким же важным ресурсом, как и процессор, так как процесс может выполняться процессором только в том случае, если его коды и данные (не обязательно все) находятся в оперативной памяти. Управление памятью включает распределение имеющейся физической памяти между всеми существующими в системе в данный момент процессами, загрузку кодов и данных процессов в отведенные им области памяти, настройку адресно-зависимых частей кодов процесса на физические адреса выделенной области, а также защиту областей памяти каждого процесса.
- **Управления файлами и внешними устройствами.** Способность ОС к «экранированию» сложностей реальной аппаратуры очень ярко проявляется в одной из основных подсистем ОС — **файловой системе**. ОС виртуализирует отдельный набор данных, хранящихся на внешнем накопителе, в виде **файла** — простой неструктурированной последовательности байтов, имеющей символьное имя. При выполнении своих функций файловая система тесно взаимодействует с подсистемой управления внешними устройствами.
- **Защиты данных и администрирования.** Безопасность данных вычислительной системы обеспечивается средствами отказоустойчивости ОС, направленными на защиту от сбоев и отказов аппаратуры и ошибок программного обеспечения, а также средствами защиты от несанкционированного доступа. В последнем случае ОС защищает данные от ошибочного или злонамеренного поведения пользователей системы.

2. Программное обеспечение

2.3 Назначения и функции локальной ОС

- **Интерфейса прикладного программирования.** Возможности операционной системы доступны прикладному программисту в виде набора функций, называющегося **интерфейсом прикладного программирования (Application Programming Interface, API)**. От конечного пользователя эти функций скрыты за оболочкой алфавитно-цифрового или графического пользовательского интерфейса. Приложения выполняют обращения к функциям API с помощью **системных вызовов**. Способ, которым приложение получает услуги операционной системы, очень похож на вызов подпрограмм.
- **Пользовательского интерфейс.** ОС обеспечивает удобный интерфейс для человека, работающего за терминалом. В ранних операционных системах пакетного режима функции пользовательского интерфейса были сведены к минимуму и не требовали наличия терминала. Команды языка управления заданиями набивались на перфокарты, а результаты выводились на печатающее устройство. Современные ОС поддерживают развитые функции пользовательского интерфейса для интерактивной работы за терминалами двух типов: **алфавитно-цифровыми** и **графическими**.

2. Программное обеспечение

2.4 Архитектура ОС

2.4.1 Ядро и вспомогательные модули

Современные ОС представляют собой хорошо структурированные модульные системы, способные к развитию, расширению и переносу на новые платформы.

Какой-либо единой архитектуры ОС не существует, но существуют универсальные подходы к структурированию ОС.

Наиболее общим подходом к структуризации операционной системы является разделение всех ее модулей на две группы:

- **ядро** — модули, выполняющие основные функции ОС;
- **модули**, выполняющие вспомогательные функции ОС, такие как управление процессами, памятью, устройствами ввода-вывода, поддержание приложений и т. п.

Ядро составляет сердцевину операционной системы, без него ОС является полностью неработоспособной и не сможет выполнить ни одну из своих функций.

Приложения могут обращаться к ядру с запросами для выполнения тех или иных действий, которые называются **системными вызовами.**

Функции ядра, которые могут вызываться приложениями, образуют **интерфейс прикладного программирования — API**.

2. Программное обеспечение

2.4 Архитектура ОС

Некоторые функции, выполняемые модулями ядра, часто используются, поэтому скорость их выполнения определяет производительность всей системы в целом. Для обеспечения высокой скорости работы ОС часто используемые модули ядра или большая их часть постоянно находятся в оперативной памяти, то есть являются **резидентными**.

Вспомогательные модули ОС, которые используются редко загружаются в оперативную память, только на время выполнения своих функций, являются **транзитными**. Такая организация ОС экономит оперативную память компьютера.

Разделение операционной системы на ядро и модули-приложения обеспечивает легкую расширяемость ОС.

В современных ОС большинство основных функций располагается в ядре, такие системы называются ОС с **монолитным ядром.**

2. Программное обеспечение

2.4 Архитектура ОС

2.4.2 Работа ОС в привилегированном и пользовательском режимах

Важным свойством архитектуры ОС, основанной на ядре, является возможность защиты кодов и данных операционной системы за счет выполнения функций ядра в привилегированном режиме. Для обеспечения привилегии операционной системе аппаратура компьютера должна поддерживать как минимум два режима работы — **пользовательский режим** и **привилегированный режим.** Подразумевается, что операционная система или некоторые ее части работают в привилегированном режиме, а приложения — в пользовательском режиме.

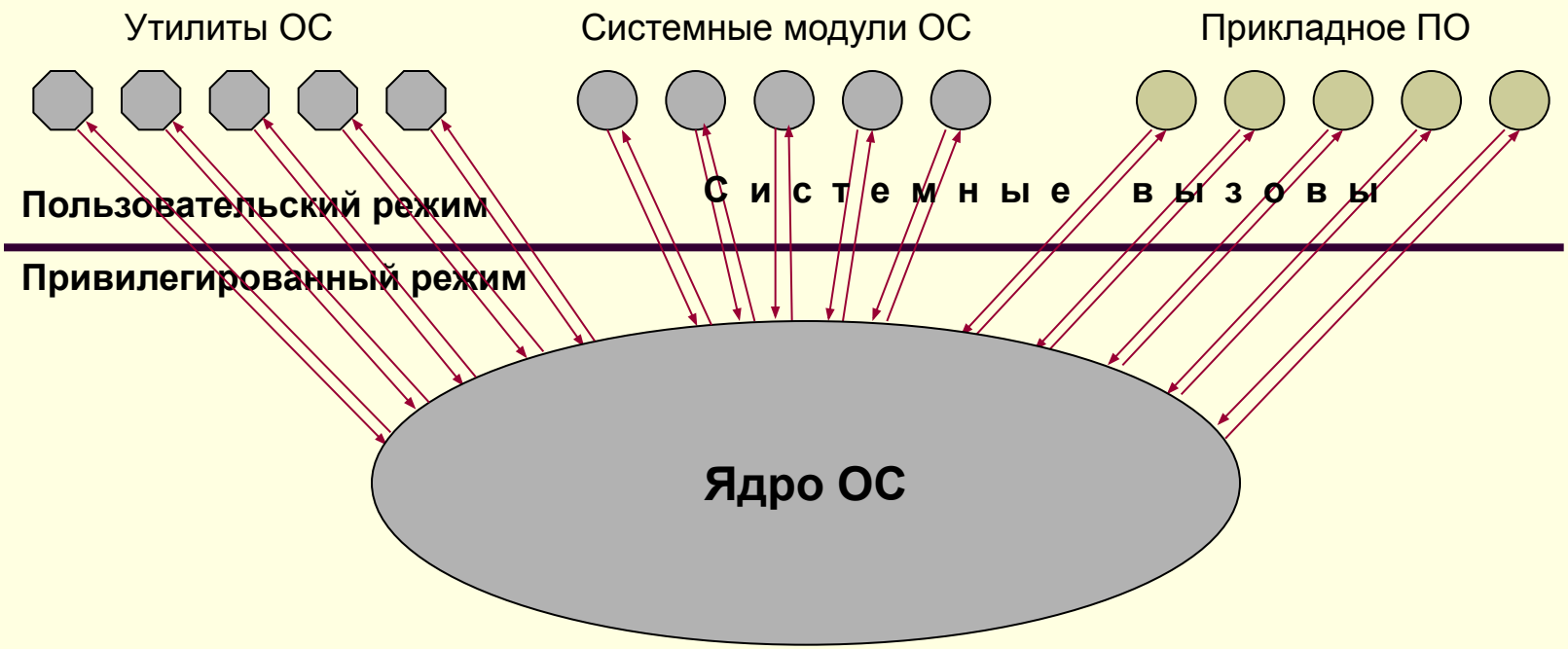
Обычно ядро является той частью ОС, которая работает в привилегированном режиме. Иногда это свойство — работа в привилегированном режиме — служит основным определением понятия «ядро».

Приложения ставятся в подчиненное положение за счет запрета выполнения в пользовательском режиме некоторых критичных команд.

2. Программное обеспечение

2.4 Архитектура ОС

Архитектура ОС с ядром в привилегированном режиме

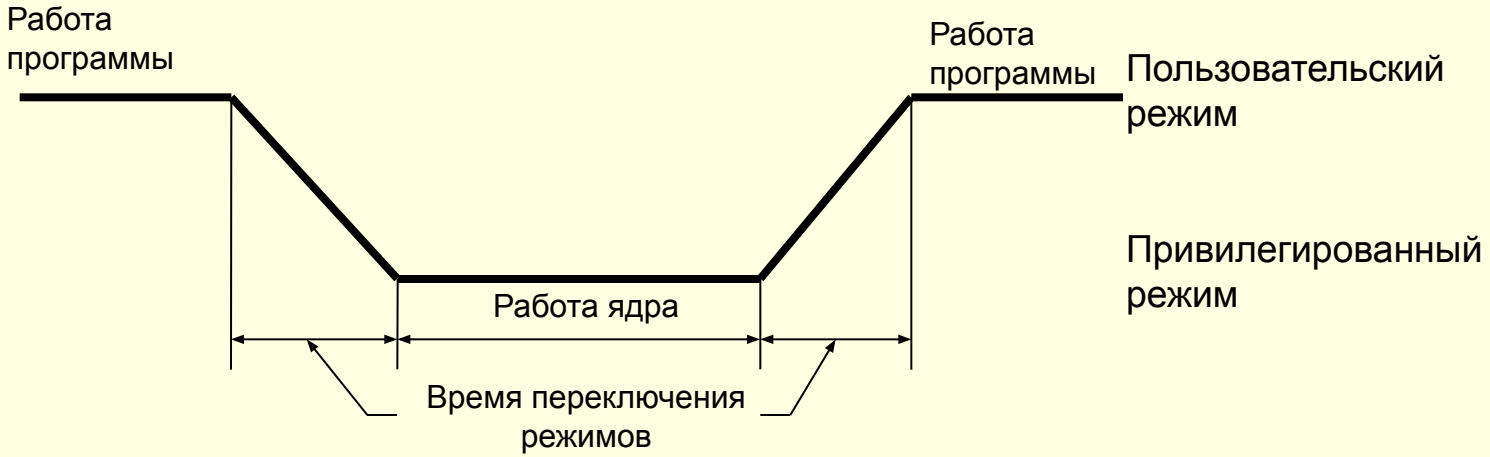


2. Программное обеспечение

2.4 Архитектура ОС

Повышение устойчивости ОС, обеспечиваемое переходом ядра в привилегированный режим, достигается за счет некоторого замедления выполнения системных вызовов. Системный вызов привилегированного ядра инициирует переключение процессора из пользовательского режима в привилегированный, а при возврате к приложению — переключение из привилегированного режима в пользовательский.

Смена режимов при выполнении системного вызова к привилегированному ядру



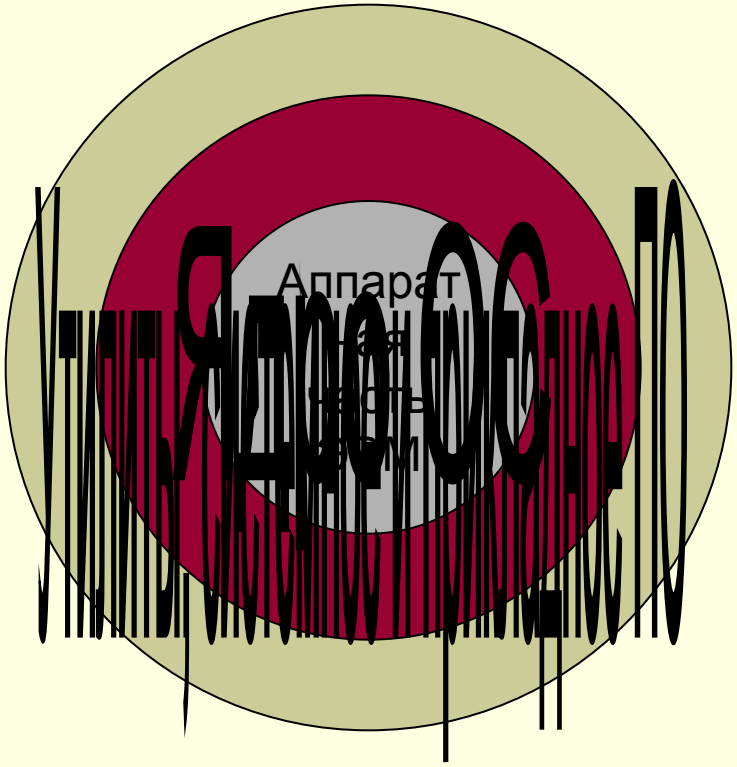
В современных ОС большинство основных функций располагается в ядре, такие системы называются ОС с **монолитным ядром.**

2. Программное обеспечение

2.4 Архитектура ОС

2.4.3 Многослойная структура и аппаратная зависимость ОС

Трехслойная схема вычислительной системы



Вычислительную систему, работающую под управлением ОС на основе ядра, можно рассматривать как систему, состоящую из трех иерархически расположенных слоев: нижний слой образует аппаратура, промежуточный — ядро, а утилиты, обрабатывающие программы и приложения, составляют верхний слой системы.

Многослойный подход является универсальным и эффективным способом организации сложных систем любого типа, в том числе и программных. В соответствии с этим подходом система состоит из иерархии слоев. Каждый слой обслуживает вышележащий слой, выполняя для него некоторый набор функций.

2. Программное обеспечение

2.4 Архитектура ОС

Большая часть модели многослойной структуры программы реализуется с помощью библиотек.

Библиотека – программный модуль, содержащий в своем составе набор функций и различных ресурсов (текст, графика, звук и т.д.), которыми может воспользоваться любая программа.

Библиотеки используются как на этапе разработки программы программистом, так и на этапе выполнения программы.

Библиотеки бывают двух типов:

- **статические** – при использовании статических библиотек в процессе компиляции необходимые функции библиотеки копируются в исполняемый модуль программы;
- **динамические** – при компиляции в программу вставляются вызовы нужных функций библиотек. Для работы такой программы обязательно наличие этих библиотек в системе.

Для программиста библиотеки доступны в виде **исходного кода** написанного на любом алгоритмическом языке или в виде **бинарного файла**.

Благодаря многослойному подходу в любой ОС можно выделить достаточно компактный слой машинно-зависимых компонентов ядра и сделать остальные слои ОС общими для разных аппаратных платформ, что облегчает перенос ОС с платформы на платформу.

Если код ОС может быть сравнительно легко перенесен с аппаратной платформы одного типа на аппаратную платформу другого типа, то такую ОС называют **переносимой, или мобильной**