Оглавление

[Вопрос 1: История ОС. Современное состояние. Основные концепции. 2](#_Toc11542597)

[Вопрос 2: Архитектура ЭВМ. Назначение основных компонентов ЭВМ. 5](#_Toc11542598)

[Вопрос 3: Схема процессора. 6](#_Toc11542599)

[Вопрос 4: Жизненный цикл программы. 7](#_Toc11542600)

[Вопрос 5: Стек. 8](#_Toc11542601)

[Вопрос 6: Понятие ОС и её основные функции. 9](#_Toc11542602)

[Вопрос 7: Архитектура Windows. 10](#_Toc11542603)

[Вопрос 8: Ядро Windows. 12](#_Toc11542604)

[Вопрос 9: Командный интерфейс ОС. Batch – файлы. 13](#_Toc11542605)

[Вопрос 10: Буферизация ввода-вывода. 14](#_Toc11542606)

[Вопрос 11: Понятие критической секции, пример программы. 15](#_Toc11542607)

[Вопрос 12: Сетевые ОС. 16](#_Toc11542608)

[Вопрос 13: Пример программы на Aссемблер. 17](#_Toc11542609)

[Вопрос 14: Файлы и файловая система. 18](#_Toc11542610)

[Вопрос 15: Файловая система FAT. 19](#_Toc11542611)

[Вопрос 16: Файловая система NTFS. 20](#_Toc11542612)

[Вопрос 17: Каналы. 21](#_Toc11542613)

[Вопрос 18: Система ввода-вывода. 22](#_Toc11542614)

[Вопрос 19: Система прерываний. 23](#_Toc11542615)

[Вопрос 20: Управление процессами и потоками(диспетчеризация и планирование). 24](#_Toc11542616)

[Вопрос 21: Алгоритмы планирования, основанные на квантовании. 25](#_Toc11542617)

[Вопрос 22: Алгоритмы планирования, основанные на приоритетах. 26](#_Toc11542618)

[Вопрос 23: Смешанные алгоритмы планирования. 27](#_Toc11542619)

[Вопрос 24: Алгоритм Джонсона. 28](#_Toc11542620)

[Вопрос 25: Синхронизация процессов. Способы синхронизации. 29](#_Toc11542621)

[Вопрос 26: Проблема тупиков. Алгоритм банкира. 30](#_Toc11542622)

[Вопрос 27: Виды памяти. 31](#_Toc11542623)

[Вопрос 28: Управление памятью. Свопинг. 32](#_Toc11542624)

[Вопрос 29: Виртуальная память. Страничная организация памяти. 33](#_Toc11542625)

[Вопрос 30: Виртуальная память. Сегментная организация памяти. 34](#_Toc11542626)

[Вопрос 31: Виртуальная память. Странично-сегментная организация памяти. 35](#_Toc11542627)

[Вопрос 32: Кеш-память. 36](#_Toc11542628)

[Вопрос 33: Управление кучей. 37](#_Toc11542629)

[Вопрос 34: Метод близнецов и выделение памяти на основе последовательности Фибоначчи. 38](#_Toc11542630)

[Вопрос 35: Клиент-серверная технология. Протоколы сетевого взаимодействия. 39](#_Toc11542631)

[Вопрос 36: Регистровая память. 40](#_Toc11542632)

# Вопрос 1: История ОС. Современное состояние. Основные концепции.

**Операционная система** – совокупность программных средств, обеспечивающая управление аппаратным обеспечением компьютера и прикладными программами, а также их взаимодействие между собой и пользователем.

17 век- арифмометры Pascal   
Использовали механизм зубчатых колес для сложения и вычитания. В СССР в 20-е годы такие арифмометры были и использовались.   
1834 год Чарльз Бэббидж создаёт два проекта универсальной аналитической механической счётной машины. Несмотря на то, что проекты не были реализованы, он указал базовую архитектуру машины и сформировал принцип программного управления. Его ученица Ада считается автором первого в мире языка программирования.   
До второй мировой войны немец Цузе создаёт электромеханическую машину Z1.  
1945 год инсельванский университет США - Первая ЭВМ ЭНИАК-1 (Джон Мочли, T. Эккерт).  
  
Архитектура ЭВМ базировалось на идеях Бэббеджа. К ним подключился Джон фон Нейман. Была создана первая версия ЭВМ ЭДВАК-1. Фон Нейман сумел получить некоторые основополагающие идеи относительно принципов работы ЭВМ: 1)должна быть использовано двоичная система счисления; 2)команда и данные хранятся вместе в оперативной памяти; 3) арифметическое устройство должно выполнять операцию сложения: специализация других операциях является нецелесообразной; 4) следует использовать параллелизм в процессе обработки команд и данных.   
Команды в первой ЭВМ набирались вручную и вводились с пультом. Создали первый язык Ассемблер (он приближённый к машинному языку).   
Переход к языкам более высокого уровня. В деле разработке языков сыграл роль Ноэм Хомский. Он создал теорию формальных языков и грамматики. Эта теория дала возможность полной автоматизации процесса разбора программы. Осталось необходимой создать компилятор (программу, предназначенную для разбора других программ).   
1957-1959 года - Джон Бэкус разработал язык Фортран (получил премию Филдинга).   
Компилятор с Фортран переводит исходный текст программы на языке Фортран в язык Ассемблер для данной машины, либо в машинный код. Компилятор выполняет задачи лексического анализа, синтаксического анализа, генерации кода.

Первые операционные системы появились в конце 50-х годов, среди них,SAGE, ATLAS, SOS. Задача ОС состоит в том, чтобы обеспечить ввод программы, её размещения в оперативной памяти, передачу ей управления, а после завершения обработки удаление программы из памяти. Особенно роль ОС взросла в связи с реализацией режим мультипрограммирования - одновременного выполнения нескольких программ в ЭВМ. Проблема состоит в том, чтобы защитить адресное пространство одной программы от другой, а также обеспечить синхронное использование ресурсов ЭВМ. 1954 год - машина "Стрела", разработчик Лебедев. У нас использовались "Минск 32".   
В середине 60-х годов прошлого века в мире доминируют фирма IBM, она выпускает ОС (System/360). Одновременно развивалась технология микрочипов (микропроцессоров). Первый микропроцессор - i404 (1971 год, фирма Intel), работал на частоте 108 кГц (использовались для светофора и калькулятора).   
  
Микропроцессор - интегральная схема смонтированная на небольшой кремниевой пластине (5\*5 мм), содержащая 1000000 транзисторов, соединенных между собой сверх тонкими проводниками из алюминия.   
1975 год основатель фирмы Microsoft Билл Гейтс продвигает реализацию интерпретатор языка Бейсик(BASIC). Одним из первых ЭВМ был Apple.   
1981 год появляются ОС MS-DOS и 16-битовый персональный компьютер работающий под управлением MS-DOS.   
Конкуренцию MS-DOS составляет Apple MACINTOSH.

1982 год начинает работать сеть персональных ЭВМ (она обеспечивала передачу электронной почты). 1983 год компьютеры оснащаются жесткими дисками (винчестерами) объемом 10 Мб. Появляется Windows, как графический расширение OС MS-DOS, мышь. ООП - языки Simula Smolltolk.  
Идеи ООП были воплощены в язык Си 1980 год ->C++ (Cтрауструп) -> развивается распределенное программирование объектов, которые стали существовать как доступные единицы программирования в сети.   
1996 год - появился язык Java, который является языком сетевого программирования, ориентированным на интернет.   
Java - фирма Sun Microsystems. Особенность Java - возможность его использования под управлением любой iOS, которая способна выполнять так называемую программу виртуальный Java-машины, в которой транслируется программа на языке Java (bi-code).Бай-код, в отличие от Ассемблер, не зависит от типа ОС. 2002 год - [платформа.NET](https://vk.com/away.php?to=http%3A%2F%2F%EF%EB%E0%F2%F4%EE%F0%EC%E0.NET&cc_key=).   
Параллельно, хотя в меньшем объеме, развиваются специализированные языки: язык логического программирования (Prolog), функционированного программирования (LISP).

В процессе эволюции возникло несколько важных понятий, **концепций:**  
Системные вызовы – это интерфейс между ОС и пользовательской программой. Они создают, удаляют и используют различные объекты, главные из которых процессы и файлы.   
Прерывание – сигнал, генерируемый внешним устройством для немедленной реакции ЦП на событие.  
Исключительная ситуация – exception – это событие, возникающее при выполнении программной команды, которая по каким-то причинам не может быть выполнена до конца.  
Файлы предназначены для хранения информации на внешних носителях.  
Под процессом понимается экземпляр выполняющийся в ОС задачи со всеми ее данными.

Главным требованием, предъявляемым к **современной операционной системе**, является выполнение ею основных функций эффективного управления ресурсами и обеспечение удобного интерфейса для пользователя и прикладных программ. Современная ОС, как правило, должна поддерживать мультипрограммную обработку, виртуальную память, многооконный графический интерфейс пользователя, а также выполнять многие другие необходимые функции и услуги  
Существуют три наиболее популярных операционных систем для компьютеров: Microsoft Windows, Apple Mac Os X, Linux.

 История ОС насчитывает примерно полвека. Она во многом определялась и определяется развитием элементной базы и вычислительной аппаратуры   
  
- Первые цифровые вычислительные машины, появившиеся в начале 40-х годов, работали без операционных систем, все задачи организации вычислительного процесса решались вручную каждым программистом с пульта управления   
  
- Прообразом современных операционных систем явились мониторные системы середины 50-х, которые автоматизировали действия оператора по выполнению пакета заданий   
  
- В период 1965-1975 гг. были реализованы практически все основные концепции, присущие современным ОС: мультипрограммирование, мультипроцессирование, многотерминальный режим, виртуальная память, файловые системы, разграничение доступа и сетевая работа   
  
- С середины 70-х началось массовое использование UNIX, уникальной для того времени ОС, которая сравнительно легко переносилась на различные типы компьютеров.   
  
- Начало 80-х годов связано с появлением персональных компьютеров. Это потребовало разработки «дружественного интерфейса». Персональные компьютеры способствовали бурному росту локальных сетей, в результате поддержка сетевых функций стала для ОС персональных компьютеров необходимым условием.   
  
- В 90-е годы практически все ОС, занимающие заметное место на рынке стали сетевыми. ОС получили средства для работы со всеми основными технологиями локальных (Ethernet, Fast Ethernet, Token Ring, FDDI) и глобальных (ISDN, ATM) сетей.   
  
- Особое внимание в течение последнего десятилетия уделялось корпоративным сетевым операционным системам. Они отличаются способностью хорошо и устойчиво работать в крупных сетях, которые характерны для крупных предприятий, способностью беспроблемно работать на различных аппаратных платформах.

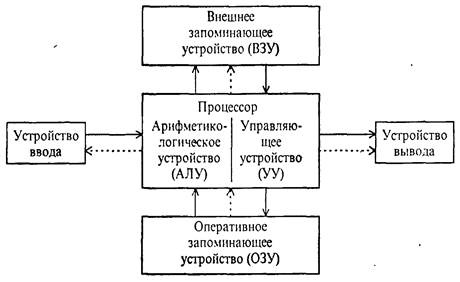
# Вопрос 2: Архитектура ЭВМ. Назначение основных компонентов ЭВМ.

Под***архитектурой ЭВМ*** понимается совокупность общих принципов организации аппаратно-программных средств и их характеристик, определяющая функциональные возможности ЭВМ при решении соответствующих классов задач.

ЭВМ (электронно-вычислительная машина) - это комплекс технических и программных средств, предназначенные для автоматизации подготовки и решения задач пользователей (предназначенных для автоматической обработки информации, вычислений, автоматического управления).

ЭВМ состоит из блока оперативной памяти, блока центрального процессора, специализированных блоков, обеспечивающих подключение и обмен информации с внешними устройствами. Блок центрального процессора – для обработки данных в соответствии с реализуемой программой, блок оперативной памяти – для хранения команд данных. Для ввода и вывода информации используются устройства ввода(например, клавиатура и т. д) и вывода(например, дисплей, принтер и т. д). Процессор образуют арифметико-логическое устройство и управляющее устройство (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)). Оперативные ЗУ хранят информацию, с которой компьютер работает непосредственно в данное время. Внешние запоминающие устройства (например, жесткий диск или винчестер) используются для длительного хранения больших объемов информации.

В ЭВМ происходит последовательное считывание команд из памяти и их выполнение. Такая архитектура ЭВМ была предложена Д. фон Нейманом.



# Вопрос 3: Схема процессора.

Процессор состоит из блока управления, блока регистров, арифметико-логического устройства. АЛУ выполняет арифметические и логические операции, например: суммирование операндов. Все вычисления производятся в двоичной системе счисления. Операнды на АЛУ поступают из регистра аккумулятора АС и одного из регистров регистрового блока. Данные вместе с кодом команды поступают в регистровый блок из оперативной памяти, адрес обращения которой выставляется на адресном счётчике РС и далее через регистр адреса MAR и идёт на шину адреса А.

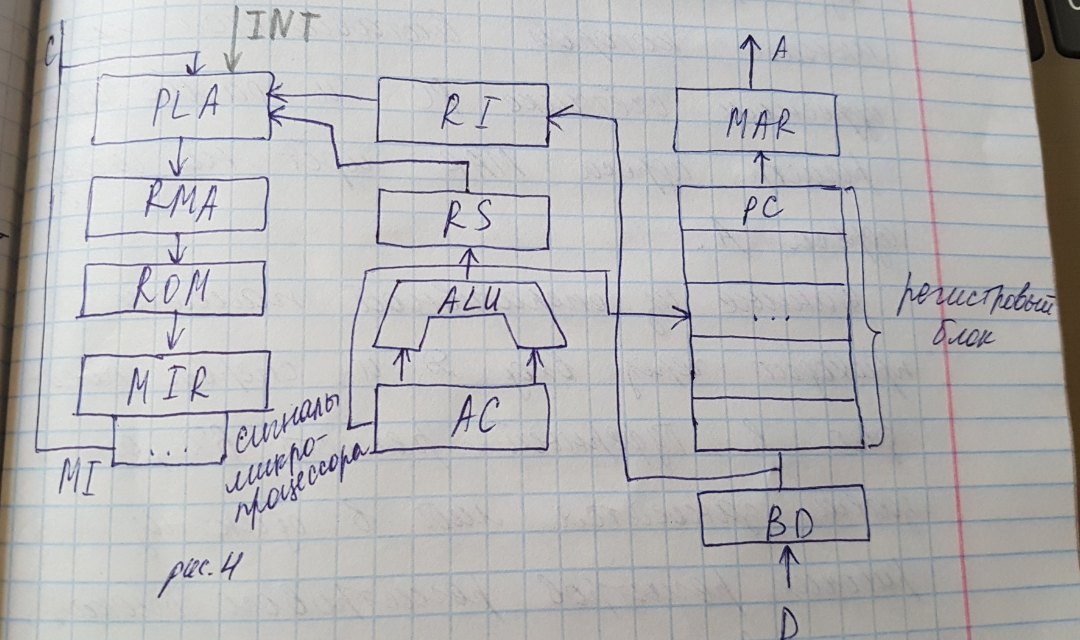
Данные из оперативной памяти приходят через вход Д и сперва попадают в буферный регистр ВД, а затем заносятся либо в один из рабочих регистров регистрового блока, либо в аккумулятор АС.

Код выполняемой команды поступает в регистр RI и определяет адрес микропрограммы реализующей команду. По шине С передаются команды для управления устройством компьютера.

Код принятой команды (в регистре RI) преобразуется в адрес микропрограммы на схеме PLA – программируемая логическая матрица. Первый адрес микропрограммы заносится в регистр RMA.

Микропрограммы представляют собой набор микрокоманд, зашитых в ROM. Микрокоманды считываются одна за одной из ROM и попадают в регистр MIR, где образуют множество сигналов микропроцессора, а так же адрес следующей микрокоманды для управления другими узлами компьютера.

Регистр RS содержит набор признаков выполняемых операций в АЛУ, такими признаками являются: обнуление результата, положительность результата и т. д.



В PLA код принятой программы преобразуется в адрес микропрограмм, первый ее адрес заносится в регистр PMА, дальше передается в регистр ROM, где представляет собой набор микрокоманд, дальше передается в регистр MIR, где образуют множ-во сигналов микропроцессора.

# Вопрос 4: Жизненный цикл программы.

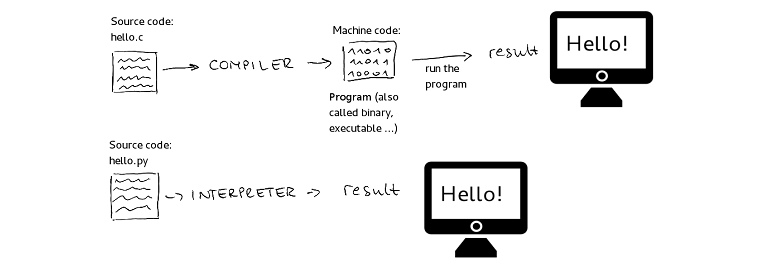
Жизненным циклом программы называется отрезок времени от принятия решения о необходимости создания программы до снятия программы с эксплуатации.

С момента создания программы в интегрированных средах( Visual Studio и т.д.) далее происходит ее трансляция (преобразование программы, представленной на одном из языков программирования, в программу на другом языке.), т.е. компиляция (создается exe файл) либо интерпретация(покомандного выполнения программы без предварительной компиляции).

Задача транслятора – перевести исходный код программы в машинный. Компилятор строит объектный код. При компиляции отлавливаются ошибки программы, а при интерпретации - ошибка отлавливается и код не выполняется.

Различают два этапа трансляции: лексический и синтаксический разбор. Лексический разбор - определение лексем. Лексемы - имена переменных, константы, названия операторов, классов, меток. Цель синтаксического анализа - определить правильность записи операторов. Для этого нужны правила грамматики.

Байт-код — это компактное представление программы, уже прошедшей синтаксический и семантический анализ. По форме байт-код похож на машинный код.



# Вопрос 5: Стек.

Стеком называется одномерная структура данных, включение и исключение элементов которого осуществляется с помощью указателя стека в соответствии с правилом "последним введен, первым выведен" (last-in, first-out − LIFO). Стек может быть реализован статически (на основе массива) и динамически (на основе списка).

В ЭВМ для организации стека выделяется область оперативной памяти, а для ее адресации и доступа к стеку используется регистр— указатель стека.

При записи числа в стек указатель стека модифицируется так, чтобы он указывал на следующую свободную ячейку, и в нее записываются данные.

При извлечении из стека данные считываются из той ячейки ОП, на которую показывает указатель, затем указатель стека модифицируется так, чтобы указывать на предпоследнее сохранённое в стеке значение.

Push – вставка в стек.  
Pop – удаление из стека.

Стек имеет **2 назначения**: 1) в нем сохраняются состояния регистров при прерывании; 2) используются при компиляции(чтобы построить объектный код из исходного), т.е. делаем синтаксический или лексический анализ.

При прерываниях извлекаем процесс или поток из стека и загружаем в стек обратно!!!

# Вопрос 6: Понятие ОС и её основные функции.

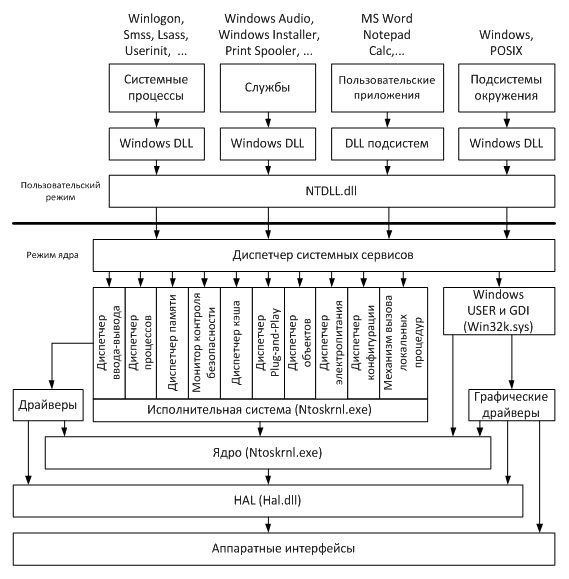
Операцио́нная систе́ма— комплекс взаимосвязанных программ, предназначенных для управления ресурсами компьютера и организации взаимодействия с пользователем.

Выполняет 2 функции:

1. Повышение эффективности использования компьютера путём управления его ресурсами в соответствии с некоторым критерием;
2. Предоставление пользователю вместо реальной аппаратуры, расширенной виртуальной машины, с которой удобнее работать и легче программировать.

Ресурсы распределяются между процессами. Процесс (задача) представляет программу в стадии выполнения. Программа - статический объект, представляющий собой файл с кодами и данными. Процесс - динамический объект, который возникает после запуска программу на выполнение, т.е. создать единицу вычислительных работ. \_\_Управление ресурсами вычислительной системы с целью наиболее эффективного их использования является одним из основных назначений ОС. \_\_Критерий эффективности в соответствии с которым ОС организует управление ресурсами ЭВМ может быть различным: пропускная способность вычислительной системы, время её реакции.  
Управление ресурсами включает решение следующих общих задач: 1)планирование ресурсов - определение какому процессу, когда и в каком количестве (если ресурс может выделяться частями) следует выделить данный ресурс; 2) удовлетворение запросов на ресурсы; 3)отслеживание состояния и учет использования ресурсов, т.е. поддержание оперативной информации о том, занят или свободен ресурсы, и какая доля ресурса уже распределена; 4)разрешение конфликтов между процессами .

# Вопрос 7: Архитектура Windows.



Архитектура Windows - многоуровневая конструкция, состоящая из двух основных компонентов: пользовательского режима и режима ядра.

Пользовательский режим:

Процессор : системный процессор – компоненты Windows, отвечающие за решение важных системных задач (например, аварийное завершение системы). Основные системные процессы: Winlogon – процесс входы в систему и выхода из неё, Lsass – проверяет правильность введённых имени пользователя и пароля, Userinit – процесс инициализации пользовательской среды.

Службы(главная служба, системный журнал, служба управления интернетом, служба памяти). Служба памяти – приложения, работающие в фоновом режиме и не требующие взаимодействия с пользователем. Службы могут быть как частью ОС(Windows Audio – служба для работы со звуком, Print Spooler – диспетчер печати), так и частью пользовательского приложения (СУБД SQL Server)

Пользовательские приложения: User Applikation – подсистемы окружения (environment subsystems) эмулятор позволяет выполнять команды к приложениям и некоторым системным функциям.

Windows DLL – динамически подключаемая библиотека. Каждая DLLэкспортирует набор Windows API функций, которые могут вызывать процесс. Windows API – способ взаимодействия процессов пользовательского режима с модулем режима ядра. Основные DLL – Kornel32.dll – базовые функции, в том числе работы с процессами и потоками, управление памятью, вводом(выводом); Advapi32.dll – функции связанные с управлением безопасностью и доступом к регистру(regedit – команда захода в регистр).

NTDLL.dll – библиотека содержит некоторые вспомогательные функции для библиотек компилятора, в так же специальные точки входа используемые ядром для инициализации потоков. Данная библиотека преобразует вызовы Vin API в вызовы функции Native API и переключает процессы вниз ядра.

Режим ядра:

* Диспетчер системных сервисов работает в режиме ядра, перехватывает вызовы функции от NTDLL, проверяет их параметры и вызывают в соответствие функции из Ntoskrnl.exe
* Исполнительская система и ядро представляют собой совокупность компонентов называемых диспетчерами которые реализуют основные задачи ОС: 1) Диспетчер это процессов (process meneger) - управление процессами и потоками, создает и завершает, приостанавливает и восстанавливает процессы и потоки, хранит о них информацию; 2)Диспетчер памяти  - управление виртуальной памятью и отображение её на физическую(в файл на диске); 3) Монитор контроля безопасности -  управление безопасностью, устанавливает правила защиты на локальном компьютере, охраняет ресурсы ОС, выполняет защиту и регистрацию исполняемых объектов; 4)Диспетчер ввода\вывода - подсистема ввода\вывода; 5) Диспетчер объектов  - создает, удаляет, управляет объектами - абстрактными типами данных, используемых для предоставления ресурсов системы. Под объектом могут быть семафоры, мьютексы, таймеры порты; 6) Диспетчер электропитания (Power meneger) - управление электропитанием и энергопотреблением; 7)Диспетчер кэша : диспетчер PNP-meneger (plug&play) - управление внешними устройствами и файловыми системами; 8) Диспетчер конфигурации - он реализует реестр. Реестр содержит конфигурационные данные для системы в файлах; 9) Механизм вызова локальных процедур (Local Procedure Call) представляет собой вариант более общего средства удаленного вызова процедур( rpc), которая используется для связи клиентов и сервером, расположенных на разных машинах в сети; 10) Windows User и GDT - отвечает за пользовательский графический интерфейс (окна, меню, кнопки, элементы рисования). Он реализован в драйвере Win32k.sys; 11)Диспетчер Open - отвечает за создание экранного интерфейса и за процессы, использующие оконные функции  
  Взаимодействие диспетчера ввода\вывода с устройствами обеспечивают драйверы - программные модули, работающие в режиме ядра, обладающие максимальной полной информацией о конкретном устройстве.

Ядро - (программный модуль) - центральная часть ОС, обеспечивающая приложениям координированный доступ к ресурсам компьютера (процессорное время, память, внешнее обеспечение, принтер, модем).  
Ядро осуществляет управление процессами и потоками, управление памятью, диспетчеризацией потоков, управлением ресурсами и т.д.   
Драйвера и ядро не взаимодействуют с физическими устройствами напрямую - посредником между программными компонентами режима ядра и аппаратурой является **HAL** - уровень абстрагирования от оборудования. Он позволяет скрыть от всех программных компонентов особенности аппаратной платформы.

# Вопрос 8: Ядро Windows.

Ядро - (программный модуль) - центральная часть ОС, обеспечивающая приложениям координированный доступ к ресурсам компьютера (процессорное время, память, внешнее обеспечение, принтер, модем).  
Ядро осуществляет управление процессами и потоками, управление памятью, диспетчеризацией потоков, управлением ресурсами и т.д.   
Драйвера и ядро не взаимодействуют с физическими устройствами напрямую - посредником между программными компонентами режима ядра и аппаратурой является HAL - уровень абстрагирования от оборудования. Он позволяет скрыть от всех программных компонентов особенности аппаратной платформы. Дра́йвер — компьютерное программное обеспечение, с помощью которого другое программное обеспечение (операционная система) получает доступ к аппаратному обеспечению некоторого устройства.

ДИСПЕТЧЕРА ИЗ ПРЕДЫДУЩЕГО ВОПРОСА!!!

# Вопрос 9: Командный интерфейс ОС. Batch – файлы.

Командная строка — принцип организации пользовательского интерфейса на основе ввода текстовых команд с клавиатуры и текстового вывода результатов на экран.

MD – создание каталога.

RD – уничтожение пустого каталога. Для этого предварительно нужно удалить из него все файлы командой DEL.

CD – переход в другой каталог.

TYPE – просмотр текстового файла.

DEL – удаление файлов. Атрибут /p – подтверждение запроса на удаление.

COPY – копирование файлов. Кроме того используется для объявления содержимого нескольких файлов.

BATCH файл - это набор команд DOS, хранящийся в обычном текстовом файле, которые можно запустить, просто набрав имя файла в командной строке. Такие файлы помогают избежать повторного набора одних и тех же команд по много раз. BATCH файл должен иметь расширение ".BAT".

Команды BATCH файла :

* CALL - вызывает из одного BATCH-файла другой BATCH-файл.
* ECHO - выводит сообщения на экран.
* FOR - исполнение одной и той же команды несколько раз.
* IF - команда проверки некоторого условия.
* PAUSE - выводит на экран сообщение и ждет ответа пользователя.
* GOTO - переход к заданному месту исполняемого BACTH-файла.
* REM - позволяет вставить комментарий.

Например, для создания трехстрочного файла с именем "dirab.bat", который показывает содержание дисков A и B, используется следующий набор команд:   
COPY CON dirab.bat <Enter>   
CLS <Enter>   
DIR A: <Enter>   
DIR B: <Enter>   
^Z <Enter>

# Вопрос 10: Буферизация ввода-вывода.

Буфером ввода-вывода называются область оперативной памяти(ОП), предназначенную для временного хранения записи в файлах. Обычно длина буфера выбирается кратной длине кластера. Буферы ввода-вывода предназначаются для решения 2 задач: 1)Устранение несоответствия между размером логической записи файла и размером кластера(логическая единица хранения данных в таблице размещения файлов, объединяющая группу секторов), который записывается на диск.  Несколько логических записей могут быть объединены в одну физическую.  2) Это снижение влияния внешних устройств на скорость работы процессора, которая значительно превышает скорость работы внешних устройств.  Для ускорения  ввода-вывода данных обычно используют несколько буферов ввода-вывода, которые организованы в кольцевую очередь.

# Вопрос 11: Понятие критической секции, пример программы.

Критическая секция – участок кода программы, в котором производится доступ к общему ресурсу, который не должен быть одновременно использован более чем одним потоком исполнения.

Чтобы исключить эффект гонок по отношению к критическим данным необходимо обеспечить, чтобы в каждый момент времени в критической секции находился только один поток.

<windows.h> системные API-вызовы  
CRITICAL\_SECTION cs - адрес переменной, представляющий критическую секцию

**LeaveCriticalSection** – системный вызов.

**CreateFile**– создание файла (stdPath – имя файла, Generic\_write - создание файла для записи, 0 – режим доступа(если функция завершается успешно объект не может совместно использоваться, пока дискриптор не закроется, NULL – дискриптор защиты; CREATE\_ALWAYS - будет создавать всегда файл; )   
**WriteFile** – запись в файл (дескриптор файла;данные,которые будут записаны;число байтов;указатель на переменную,в которую записывается;)

int\_main  
0-начальный размер стека

**CreateThread** - создание потока(  
0-кол-во реальных записанных байтов  
0-начальный размер стека  
myThread - ф-ция, которую выполняет поток  
(void\*)Buff - параметр, который передается в ф-цию  
CREATE\_SUSPENDED - создание потока приостановлено и не запускается до вызова ф-ции Resume Thread (если 0, то поток запускается сразу)  
&muThread ID - id потока);

**BOOL b = SetThreadPriorityBoost**(myHandle, false) - ф-ция отключает (try) или включает (false) способность системы временно превышать приоритет потока;  
if(b) - если переменная истинно, то SetThreadPriority - устанавливает приоритет потоку;  
WaitForSIngObject(myHAndle, 10000) - программа будет ждать пока поток не закончится либо 10000 милисекунд.

# Вопрос 12: Сетевые ОС.

Сетевые ОС составляют основу любой вычислительной сети, отличаются от обычных OС тем ,что предоставляют сервисы, необходимые для работы компьютеров в сети. К числу таких сервисов относится : считывание файлов с удаленного компа, запуск программы на удаленном компе, отправка сообщений по сети, использование сетевого принтера, работа с удаленным рабочим столом и т. д. Такие сервисы предоставляются специализированными модулями (подсистемами называемыми службами).

В качестве примера сетевых служб можно назвать: TELNET - подсистема обеспечивает работу с удаленным компьютером  FTP  - обеспечивает пересылку файлов с удалённых машин и т,д. Каждый компьютер в сети в значительной степени автономен, поэтому под **сетевой ОС** понимается совокупность операционных систем отдельных компьютеров, взаимодействующих с целью обмена сообщениями и разделение ресурсов по единым правила - протоколом. В узком смысле сетевая ОС - операционная система отдельного компа, предоставляющая ему возможность работать в сети.



В сетевой ОС можно выделить следующие части: средства управления локальными ресурсами: функции распределения оперативной памяти между процессами, планирование и диспетчеризация процессами, управление периферийными устройствами.

Серверная часть представлена - средства предоставления собственных ресурсов и услуг в общее пользование, например, эти средства обеспечивают блокировку файлов и записей,, обработку запросов удалённого доступа к собственной файловой системе и базе данных.

Клиентская часть представляет средства доступа к удаленным услугам и ресурсам и их использование.  Основная функция клиентской части  - формирование запросов к серверу, например, запуск программы, передача сообщений, чтения файлов.

Транспортные средства ОС – средства, совместно с коммуникационной системой обеспечивающие обмен сообщениями в сети.

Основными двумя технологиями реализации сетевых ОС является одноранговая и клиент-серверная. Одноранговая технология предполагает, что компьютеры в сети равноправны, т.е. нету выделенного сервера. Клиент-серверная, технология, т.е. выделяется 1 сервер в сети, которые выполняет обработку запроса клиентов.

# Вопрос 13: Пример программы на Aссемблер.

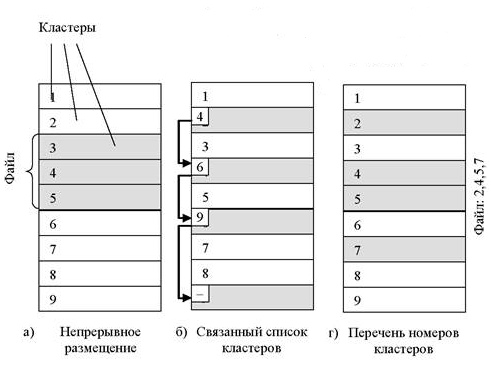
ma sm  
.data  
per1 ab"Hello",$ ;сообщение для вывода  
.code  
begin: ;начало пpогpаммы  
mov ex.@datа   
mov ds, ax ; пересылка в ds через пересылку ax,  
mov ah, 09h ;функция DOS 09h вывода на экран  
mov dx, offset per1 ;загрузка в регистр dx адреса сообщения для вывода

int 21h ;вызов прерывания равное 21h  
mov ah, 4c00h  
int 21h ;вызов прерывания равное 21h  
end begin ;конец программы  
Обработка ассемблерной программы основана на вызове ф-ции ОС и механизме программных прерываний. Ассемблерная программа процессора Pendium использует область данных и область кода. Вывод строки выполняется ф-цией ОС: 09h. В регистр [dx] помещается адрес переменной per1. 21h - обработчик программного прерывания с этим номером выполнит эту ф-цию. Далее ассемблерный код переводится в машинный и называется объектным модулем. В объектном модуле адреса переменных определены только как относительные( имеется смещение). Далее формируется загрузочный модуль - машинный код настроенный на конкретные адреса в памяти.

# Вопрос 14: Файлы и файловая система.

**Файл** - именованная область внешней памяти, в которую можно считывать данные.  
**Файловая система** - часть ОС, вмещающая:  
- совокупность всех файлов на диске  
- наборы структур данных используемых для управления файлами (каталоги)  
-комплекс системных программных средств, реализующих операции над файлами создание, удаление, поиск, запись ...  
т.е. файловая система берёт на себя функции по именованию файлов, обеспечению доступа данных,...  
**Типы файлов** : обычные файлы, каталоги - особый тип файлов, содержащие системную справочную информацию о наборе файлов по какому-то признаку. Специальные файлы - файлы, ассоциируемые с устройствами ввода-вывода, используемые для унификации механизма доступа к файлам и внешним устройством.

**Имена файлов**: простое; полное - имя идёт от корня до файла; относительное - относительно какого-то открытого каталога.   
Физическая организация файловой системы - диск разбивается на дорожки, нумерация дорожки происходит с края, каждая дорожка разбивается на фрагменты - сектора (512 байт). Запись на диск осуществляется благодаря способности головки чтения записи изменять магнитное свойство.  
ОС при работе с диском использует соответствующую единицу дискового пространства - кластер. При создании файла место на диске определяется файлами.  
Раздел - непрерывная часть физического диска, которую он предоставляет пользователю как логическое устройство. У файла есть атрибуты (свойства файла).  
**Типы размещения файлов**: а)непрерывное размещение - файлу предоставляется последовательность кластеров диска, образующих непрерывной участок дисковой памяти; б)размещение файлов в виде связанного списка кластера(логическая единица хранения данных в таблице размещения файлов объединяющая группу секторов) при таком способе кластер содержит указатель на следующий кластер; г)перечень номеров кластеров: в этом случае адресная информация минимальна расположение файла может быть задана номером кластера. При этом способе мы избегаем фрагментацию.



# Вопрос 15: Файловая система FAT.

Файловая система FAT была разработана Биллом Гейтсом и Марком Макдональдом.

Сейчас существуют три типа файловой системы FAT:   
• FAT12 – поддерживает очень небольшие объемы дисков, поэтому сейчас она применяется только на дискетах.   
• FAT16 – используется на винчестерах и поддерживает диски объемом до 2 Гб, поэтому сейчас данная файловая система практически не используется.   
• FAT32 – теоретически поддерживаются диски объемом до 2 Гб. Поддерживается, начиная с операционной системы Windows 95 OSR2. Данная файловая система сейчас достаточно популярна, хотя в последние годы многие пользователи Windows XP предпочитают использовать NTFS.

https://pp.userapi.com/c851020/v851020904/141773/vNpAwJmv3TA.jpg

Рис. 1. Структура файловой системы FAT

– Загрузочный сектор   
В начале раздела диска с файловой системой FAT располагается загрузочный сектор. Он необходим для начальной загрузки компьютера. Так же в нем располагается информация о параметрах данного раздела.

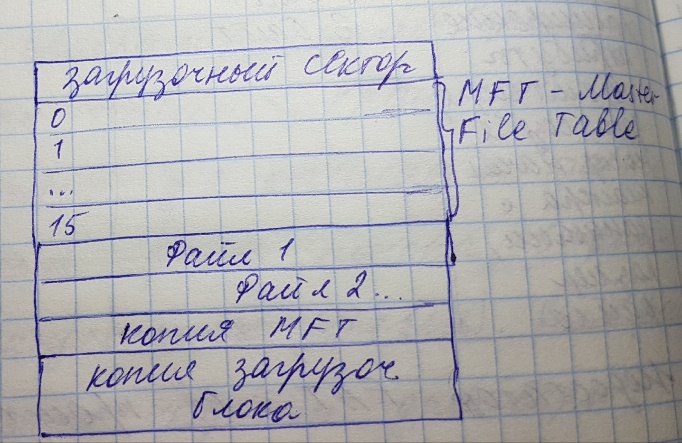
– Таблица размещения файлов .  
Вся область данных диска разделена на кластеры – блоки, размер которых задается при форматировании диска.

– Корневой каталог   
Область диска, в котором располагается информация о корневом каталоге. Размер ее ограничен, поэтому в корневом каталоге диска может находиться не более 512-ти файлов и подкаталогов.

– Область данных   
Часть раздела, на которой размещается содержимое файлов и каталогов.

Файловая система FAT обладает линейным поиском. (чтоб найти последний FAT, проход с верху и сравнивание каждого FAT с каждым)

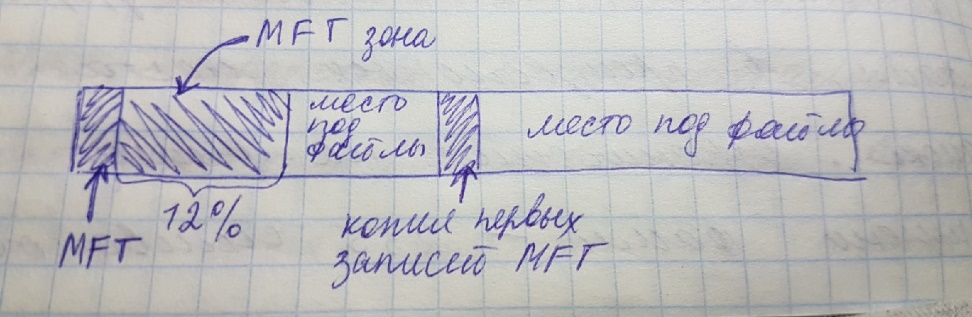
# Вопрос 16: Файловая система NTFS.



Все файлы на диске NTFS идентифицируются номером файла, который определяется позицией файлов MFT.   
Весь том NTFS делится на кластеры, каждый элемент системы представляет собой файл. Самый главный файл - MFT - представляет собой каталог всех остальных файлов диска включая самого себя.   
Каждая запись MFT имеет фиксированный размер 1 кб. Первые 16 файлов носит служебный характер.   
В NTFS файл целиком размещается в MFT, если позволяет его размер. Копия первых 3 записей хранится ровно посередине диска, остальное MFT файл может располагаться, как и любой другой файл, в произвольных местах диска.

В этой файловой системе отсутствуют ограничения на размер файлов и каталогов.

Для поиска файлов используются индексные деревья (B-дерево,balanced tree). FAT занимает весь файл, MFT - кусочек. В NTFS понятие файла заменено понятием потока. Поток - объект хранения чего-то. В поток можно добавлять информацию(кем создан файл), атрибуты файла,. В NTFS если не хватает места, то она выделяет область для хранения extend. Первые 12% диска отводится под зона MFT -пространство в которое растет метафайл MFT. 88% - пространство для хранения файла. Механизм использования зоны MFT: когда файлы нельзя уже записывать в обычное пространство, MFT зона сокращается, освобождая таким образом места для записи файла. При освобождении места в обычной области MFT зона может снова расширяться. При этом не исключена ситуация, когда в этой зоне остались обычные файлы.



В-дерево - все файлы разделяются, упорядочиваются по имени или индексу, выбирается средина списка и заносится в корень дерева, дальше делятся пополам и с каждого берется половина) (NTFS) Поиск- если ключ содержится в текущем узле, возвращаем его. Иначе определяем интервал и переходим к соответствующему сыну. Повторяем пока ключ не найден или не дошли до листа.

# Вопрос 17: Каналы.

**Канал** —механизм передачи сообщений между процессами т.е. канал реализует механизм межпроцессного взаимодействия. Через канал можно передавать данные только между двумя процессами. Один из процессов создает канал, другой – открывает его. После этого оба процесса могут передавать данные через канал в одну или обе стороны. Различные реализации каналов могут быть именованные или анонимные, синхронными или асинхронными, использовать буферизацию сообщений или нет.   
 [\\Имя\_сервера\pipe\Имя\_канала](file:///\\Имя_сервера\pipe\Имя_канала)

Если процесс открывает канал, созданный на другой рабочей станции, он должен указать имя сервера. Если процесс создает или открывает канал на своей рабочей станции, то указывается **точка**: [\\.\pipe\Имя\_канала](file:///\\.\pipe\Имя_канала) (pipe – тип объекта, который создается)

CЕРВЕРНАЯ ЧАСТЬ  
**CreateNamedPipe**  (аргументы: PIPE\_ACCSESS\_DUPLEX - создает канал для чтения и записи  
PIPE\_WAIT - Режим работы канала (может быть ориентировано на передачу потока байт, либо на передачу сообщения, в нашем случае передаются байты). Канал будет работать в блокирующим режиме, когда процесс переводится в состояние ожидания до завершения операции в канале  
1 - прототип функции, максимальное количество реализации, которая может быть создана для канала  
1024\*16 - Размер используемый для записи  
1024\*16 - Размер используемый для чтения  
NMPWAIT\_USE\_DEFAULT\_WAIT - определяется время ожидания для реализации команд  
NULL - передаётся адрес переменной, содержащий атрибуты защиты для канала)

Этот процесс может создать канал только на одной рабочей станции, поэтому для создания канала имя сервера не указывается.  
ConnectNamedPipe - установка соединения с каналом   
КЛИЕНТСКАЯ ЧАСТЬ  
  
WriteFile - запись в канал

Каналы являются основополагающими для подхода исчисления процессов, и появились в взаимодействующих последовательных процессах (CSP), формальной модели параллелизма. Каналы используются во многих производных языках программирования, таких как Occam, Limbo (через языки Newsqueak и Алеф). Они также используются в потоковой библиотеке libthread языка программирования Си в ОС Plan 9, а также в Stackless Python и языке Go.

# Вопрос 18: Система ввода-вывода.

Система ввода-вывода (Input-Output System) – часть операционной системы, обеспечивающая управление внешними устройствами, подключенными к ЭВМ.

Основной задачей системы ввода-вывода является обеспечение непрерывной организации (планирования, управления) и двусторонней передачи данных между основной памятью и внешними устройствами с целью достижения максимального перекрытия во времени работы этой аппаратуры и процессора.

ОС осуществляет функции управления вводом-выводом. Драйвер - программный модуль обычно с расширением sys, обеспечивает интерфейс (управление,взаимодействие) между ОС и внешним устройством. Непосредственно драйвер не работает с устройством напрямую, т.е. от него скрыты особенности реализации внешнего устройства, а сигнальное управление внешним устройством реализуется через контроллер (адаптер). Центральным элементом подсистемы ввода-вывода являются диспетчер, он осуществляет выбор нужного драйвера и передачи ему запроса на ввод-вывод (IRP пакет). Драйвер вызывает процедуру HALL, чтобы преобразовать команду записи в последовательности сигналов, причём последовательность сигналов вырабатывается контроллером внешнего устройства.

Состав систем ввода-вывода и, следовательно, перечень драйверов устройств в различных операционных системах не совпадают, что объясняется имеющимися отличиями в аппаратуре ввода-вывода, а также множеством методов, используемых для управления этой аппаратурой. Вместе с тем в большинстве операционных систем существует некоторое ядро системы ввода-вывода, получившее название базовой системы ввода-вывода.

При построении систем ввода-вывода аппаратура ввода-вывода рассматривается как совокупность аппаратурных процессоров, которые способны работать параллельно и независимо друг от друга, а также относительно центрального процессора.

# Вопрос 19: Система прерываний.

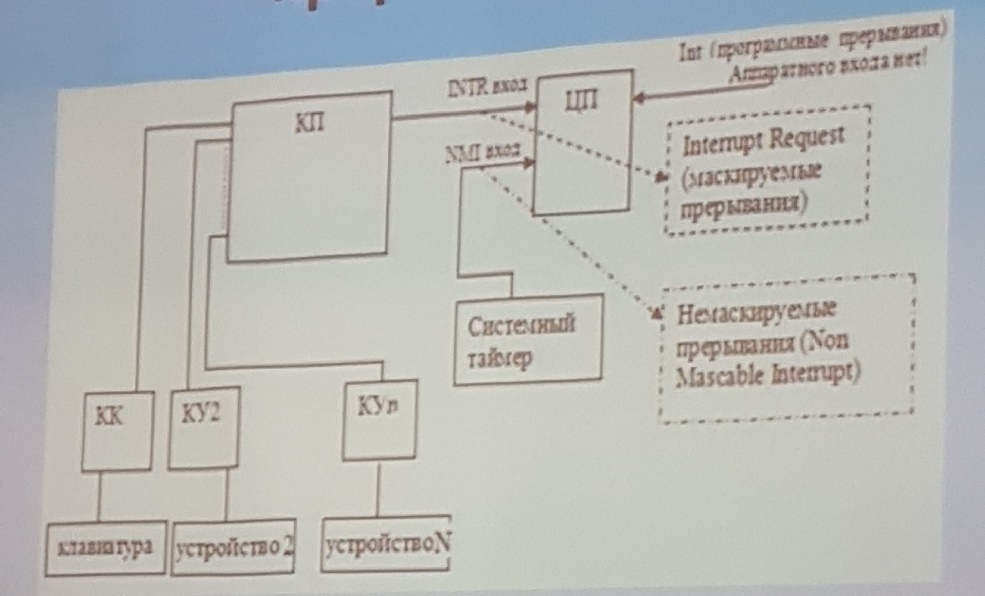
Прерывание (англ. interrupt) — сигнал, заставляющий ЭВМ менять обычный порядок выполнения команд процессором. Прерывание извещает процессору о наступлении высокоприоритетного события, требующего прерывания текущего кода, выполняемого процессором.

Прерывание делится на аппаратное (вызывается физическими устройствами и возникает асинхронно) и программное (вызывается событиями, которые связаны с работой процессора и являются синхронными с его операциями). С каждым прерыванием связывают число, которое называется номером прерывания. Система умеет распознавать какое прерывание и с каким номером произошло, и запускает соответственно этому номеру программу обработки прерывания.   
Обработчик прерываний является частью ОС, предназначенный для выполнения ответных действий на условия вызвавшие прерывание. Например, прерывание №9 – прерывание от клавиатуры.

IRQ1 – IRQ-прерывание, 1-приоритет прерывания.

При возникновении прерывание с высоким уровнем процессор сохраняет состояние прерванного потока и запускает связанный с прерыванием диспетчер системных прерываний. Тот в свою очередь поднимает уровень IRQL (L-уровень) вызывает программу обработки прерывания, а она вызывает драйвер устройства, инициализирующий прерывание. После выполнения этой процедуры диспетчер понижает уровень IRQL , а затем загружается сохраненное состояние машины.   
Прерванный поток продолжает выполнение с того места, в котором был прерван.   
Последовательность действий для прерываний:  
-приём сигнала, идентификация   
-Запоминание состояния прерванного потока  
-Передача управления обработчику прерываний  
-Обработка прерываний  
-Восстановление прерванного потока

Программы могут вызывать сами прерывание с заданным номером, они используют команду INT.



# Вопрос 20: Управление процессами и потоками(диспетчеризация и планирование).

Одной из основных подсистем ОС является подсистема управления процессами и потоками, которая занимается их созданием и уничтожением, поддерживает взаимодействие между ними, распределяет процессорное время между несколькими процессами и потоками, осуществляет синхронизацию потоков.   
В настоящее время в большинстве ОС определены 2 типа единицы работы. Более крупная единица - **процесс** или задача, требует для своего выполнения более мелкие работы - **поток**, нить. В системе unix отсутствует понятие потока, там только процессы.  
ОС обеспечивает каждый процесс отдельным виртуальным адресным пространством и распределяет процессорное время между потоками одной программы.  
Все потоки одного процесса используют одно и тоже адресное пространство, используют одни и те же ресурсы, при этом потоки разных процессов защищены друг от друга.  
Создание процесса включает загрузку его кодов и данных исполняемой программы с диска в оперативную память. Для этого ОС должна обнаружить местоположение такой программы на диске, выделить оперативную память исполняемой программе нового процесса и считать программу в выделенном участке памяти.

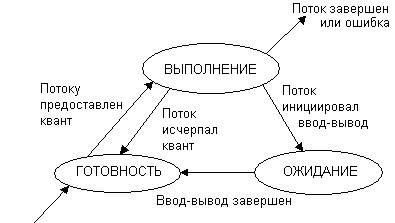
Переход от выполнения одного потока к другому осуществляется в результате планирования и диспетчеризации.   
Работа по определению того, в какой момент необходимо прервать выполнение текущего активного потока и какому потоку предоставить возможность выполняться называется **планированием**.  
Планирование потоков осуществляется на основе информации, хранящейся в описателях процессов и потоков при планировании могут приниматься во внимание приоритет потоков, время их ожидания в очереди, накопленное время выполнения и так далее.  
Планирование потоков включает в себя решение следующих задач:   
1)Определение момента времени для смены текущего активного потока;  
2)Выбор для выполнения потока из очереди готовых;  
Создать процесс - создать описатель процесса, в качестве которого выступает одна или несколько информационных структур, содержащих сведения о процессе необходимые ОС для управления им. В число таких сведений входит идентификатор процесса, данные о расположение в памяти исполняемого модуля, приоритет процесса.

Существуют две разновидности приоритетного планирования: относительное и абсолютное. В системах с относительными приоритетами активный поток выполняется до тех пор, пока он сам не покинет процессор, перейдя в состояние ожидания. В системах с абсолютными приоритетами выполнение активного потока прерывается, если в очереди готовых потоков появился поток с более высоким приоритетом. Динамическое планирование - решение принимается во время работы системы на основе анализа текущей ситуации. Статическое планирование - планировщик принимает решение о работе системы заранее.

**Диспетчеризация** заключается в реализации, найденного в результате планирования, решения, т.е. переключение процессора с одного потока на другой. Прежде чем прервать поток, ОС запоминает его текст, контекст отражает состояние аппаратуры компьютера в момент прерывание, значения счётчика команд, содержимые регистром (флаги, маски прерываний).   
Диспетчеризация сводится к следующему:  
1)сохранение текущего потока, которое следует сменить;   
2)загрузка контекста нового потока;  
3)запуск нового потока на выполнение

# Вопрос 21: Алгоритмы планирования, основанные на квантовании.

В основе многих вытесняющих алгоритмов планирования лежит концепция квантования, согласно которому каждому потоку поочередно для выполнения предоставляется ограниченный непрерывный период процессорного времени — квант. Смена активного потока происходит, если:   
- поток завершился и покинул систему;   
- произошла ошибка;   
- поток перешел в состояние ожидания;   
- исчерпан квант процессорного времени, отведенный данному потоку.

Граф состояний потока в системе с квантованием: 

Поток, который исчерпал свой квант, переводится в состояние готовности и ожидает, когда ему будет предоставлен новый квант процессорного времени, а на выполнение выбирается новый поток из очереди готовых.

Кванты могут быть одинаковыми для всех потоков или различными. Они могут быть фиксированной величины, а могут изменяться в разные периоды жизни потока.

# Вопрос 22: Алгоритмы планирования, основанные на приоритетах.

Приоритет - это число, характеризующее степень привилегированности потока при использовании ресурсов вычислительной машины. Приоритет может быть положительным, отрицательным значением. Приоритет процессу назначается ОС при его создании. При назначении приоритета учитывается является ли этот процесс системным или пользовательским; статус пользователя, запустившегося процесса; было ли явное указание пользователя на присваивание процессу определенного уровня приоритета.  
В Windows 32 уровня приоритетов и 2 класса потоков: потоки реального времени, потоки с переменными приоритетами.   
Диапазон от 1 до 15 включительно отведен для потоков с переменными, от 1 до 16 - для потоков реального времени.  
Существует два типа приоритетного планирования: обслуживания с относительными и абсолютными приоритетами. На выполнение из очереди которых выбирается поток, имеющий наивысший приоритет. Активный поток выполняется до тех пор, пока он сам не покинет процесс перейдя в ожидание. (относительный приоритет планирования).  
С абсолютными приоритетами активный поток может покинуть процесс, если в очереди готовых появился поток, приоритет которого выше активного потока. В этом случае прерванный поток переходит в состояние готовности.



# Вопрос 23: Смешанные алгоритмы планирования.

Смешанный алгоритм планирования используется во многих ОС. Например, в основе планирования лежит квантование, но величина кванта и/или порядок выбора потока из очереди готовых определяется приоритетами потоков. Таким образом так реализовано планирование в системе Windows, в которой квантование сочетается с динамическими абсолютными приоритетами. На выполнение выбирается готовый поток с наивысшим приоритетом. Ему выделяется квант времени. Если во время выполнения в очереди готовых появляется поток с более высоким приоритетом, то он вытесняет выполняемый поток, который возвращается в очередь готовых, причем он становится впереди всех остальных потоков имеющих такой же приоритет.

В системах UNIX системному процессору выделяется большой квант процессорного времени.

# Вопрос 24: Алгоритм Джонсона.

Алгоритм Джонсона[1] предназначен для решения задачи поиска всех кратчайших путей на графе.

Граф — абстрактный математический объект, представляющий собой множество *вершин* графа и набор *рёбер*, то есть соединений между парами вершин.

Для заданного ориентированного взвешенного графа алгоритм находит кратчайшие расстояния между всеми парами вершин за время https://pp.userapi.com/c852036/v852036153/14a41d/-2ltchz0O5s.jpg .   
В случае разреженных графов алгоритм Джонсона эффективнее алгоритма Флойда-Уоршела, имеющего сложность https://pp.userapi.com/c852036/v852036153/14a424/0ldSXpP8gis.jpg. Он применим к в случае произвольных (в том числе отрицательных) вещественных весов, в отличие от алгоритм Дейкстры, имеющего ту же сложность.

# Вопрос 25: Синхронизация процессов. Способы синхронизации.

Потребность в синхронизации потоков возникает только в мультипрограммной операционной системе и связана с совместным использованием аппаратных(сканер, принтер) и информационных ресурсов вычислительной системы. Отсутствие синхронизации может приводить к гонкам и тупикам. Любое взаимодействие процессов или потоков связано с их синхронизацией, которая заключается в согласовании их скоростей путем приостановки потока до наступления некоторого события и последующей его активизации при наступлении этого события.

Синхронизация необходима для исключения кода при обмене данными между потоками, разделений данных, при доступе к процессору и устройствам ввода-вывода. Во многих OС эти средства называются средства межпроцессорного взаимодействия.

Способы синхронизации:

Мьютекс - булевская переменная, например: если мьютекс красный цвет - вход запрещён, зелёный -разрешен. Мьютекс стоит на входе критической секции. Перед тем когда процесс заходит в критическую секцию , он проверяет: зелёный - заходим, красный - поток (процесс) становится в очередь.

Критическая секция - область кода, где используются разделяемые ресурсы. Например: имеется два процесса, работают с одним файлом, один процесс может работать, другой будет ожидать.   
Мьютекс - упрощенной семафор.   
Критическая секция работает с потоками одного процесса, а мьютекс - с разными процессами.   
Семафор - объект, ограничивающий количество потоков, которые могут войти в заданный участок кода.   
Семафор в отличие от мьютекса может иметь значение > 1, например 3. Он выступает как неотрицательная переменная. Если семафор >нуля, то процесс заходит и вычитается единица из самой переменной, а при выходе один прибавляется. Если семафор = нулю, то процесс не зайдёт.

# Вопрос 26: Проблема тупиков. Алгоритм банкира.

Проблема тупиков возникает, когда блокируются два процесса. Тупики могут называться взаимной блокировкой - дедлоками.

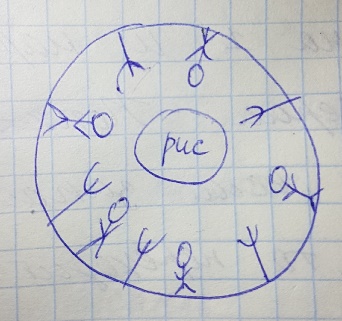
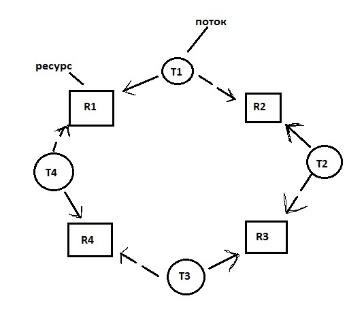
 Рассмотрим пример тупика. У каждого китайца должно быть 2 палки. Один китаец хватает палку слева, а правой палки нет. В этой ситуации нужно дать нескольким китайцам поесть, а потом передать палочки другим.

Рис - это ресурсы. Китайцы - процессы или потоки, которые ведут борьбу за ресурсы . Если каждый процесс возьмёт немного по основному ресурсу, то не выполнится ни какой.

Невозможность потоков завершить начатую работу из-за возникновения взаимных блокировок снижает производительность вычислительной системы.

Взаимная блокировка нескольких потоков: сплошная стрелка означает, что соответствующий ресурс был выделен потоку, пунктирная стрелка - потоку необходим ресурс, но он не может быть пока выделен, т.к. он занят другими потоками.

Алгоритм банкира:

Есть память 100 мб, A, B, C, D - процессы. Для каждого процесса известен объем памяти, который может быть востребован.

|  |  |  |
| --- | --- | --- |
|  | Запрос | Max память |
| А | 20 | 40 |
| B | 40 | 45 |
| C | 30 | 30 |
| D | 50 | 50 |

Выделяем память процессу, если он гарантированно завершится. Банкир выделяет память только если памяти хватит для завершения последнего процесса.

# Вопрос 27: Виды памяти.

Память компьютера делится на внешнюю (основную) и внутреннюю.   
К внутренней памяти относятся:   
1. Оперативная память — это устройства, где размещены данные, которые процессор обрабатывает в определенный промежуток времени. При этом выполняется следующее условие: в любой момент существует условие работы с любой ячейкой оперативной памяти. В оперативной памяти сохраняется временная информация, которая изменяется по мере выполнения процессором различных операций, таких как запись, считывание, сохранение. При отключении компьютера вся информация, которая находилась в оперативной памяти исчезает, если она не была сохранена на других носителях информации.   
2. Регистры — это сверхскоростная память процессора. Они сохраняют адрес команды, саму команду, данные для её выполнения и результат.

3. Кэш-память — это промежуточное запоминающее устройство, используемое для ускорения обмена между процессором и RAM (оперативная память). В современных процессорах используется несколько уровней кэш-памяти.   
4. Постоянная память — это электронная память предназначена для длительного сохранения программы и данных. Используется оно для чтения данных. Как правило, эта информация записывается при изготовлении компьютера и служит для начальной загрузки оперативной системы, проверки работоспособности компьютера.   
 Внешняя память рассчитана на длительное хранение программ и данных. Она реализуется с помощью специальных устройств, которые в зависимости от способов записи и считывания делятся на магнитные, оптические и магнитооптические.   
Основными характеристиками внешней памяти являются её объем, скорость обмена информацией, способ и время доступа к данным.   
 К внешней памяти принадлежат также накопители на гибких дисках (дискетах). Наиболее распространёнными являются дискеты диаметром 3,5 дюйма.

Полупроводниковая память, форма электронного устройства хранения данных, обычно используется для компьютерной памяти и реализуется на полупроводниковой интегральной схеме

# Вопрос 28: Управление памятью. Свопинг.

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.

Разбиение памяти на несколько непрерывных разделов может быть статическим, либо динамическим (то есть процесс выделения нового раздела памяти происходит непосредственно при появлении новой задачи). Распределение памяти перемещаемыми разделами :   
перемещаемые разделы используются как метод для работы с фрагментацией памяти, заключающийся в перемещение всех занятых участков в сторону старших или младших адресов, чтобы свободная память образовала единую область.

Одним из наиболее популярных способов управления памятью в современных операционных системах является так называемая виртуальная память. Наличие в ОС механизма виртуальной памяти позволяет программисту писать программу так, как будто в его распоряжении имеется однородная оперативная память большого объема, часто существенно превышающего объем имеющейся физической памяти. В действительности все данные, используемые программой, хранятся на диске и при необходимости частями (сегментами или страницами) отображаются в физическую память. При перемещении кодов и данных между оперативной памятью и диском подсистема виртуальной памяти выполняет трансляцию виртуальных адресов, полученных в результате компиляции и компоновки программы, в физические адреса ячеек оперативной памяти.  
Функциями ОС по управлению памятью являются отслеживание свободной и занятой памяти; выделение памяти процессам и освобождение памяти при завершении процессов; защита памяти; вытеснение процессов из оперативной памяти на диск, когда размеры основной памяти недостаточны для размещения в ней всех процессов, и возвращение их в оперативную память, когда в ней освобождается место, а также настройка адресов программы на конкретную область физической памяти.

Подкачка страниц, она же свопинг - один из механизмов виртуальной памяти, при котором отдельные фрагменты памяти (обычно неактивные) перемещаются из ОЗУ на жёсткий диск (или другой внешний накопитель, такой как Флеш-память), освобождая ОЗУ для загрузки других фрагментов памяти. Такими фрагментами в современных ЭВМ являются страницы памяти.

# Вопрос 29: Виртуальная память. Страничная организация памяти.

Виртуальная память - между оперативной памятью и диском перемещаются части ( сегменты, страницы и т.д.) образов процесса.  При нехватке памяти в ОП на диск выгружается только часть образов процесса.

Механизм виртуальной памяти предполагает разделение памяти :

1)страничное

2)сегментное

3)странично-сегментное

Согласно страничному распределению памяти вся оперативная память, а так же виртуальная память процесса делится на страницы.

При создании процесса ОС загружает в ОП несколько виртуальных страниц. Копия виртуального адресного пространства хранится на диске. Смежные виртуальные страницы необязательно располагать в смежных физических страницах. Для каждого процесса ОС создает таблицу страниц – структуру, в которой хранятся записи о всех виртуальных записях страницы процесса. Запись таблицы называется дескриптором страницы, включает следующую информацию: 1)номер физической страницы в ОП; 2)признак присутствия = 1, если такая виртуальная страница находится в ОП; 3)признак модификации = 1, если производится запись по адресу, относящемуся к данной странице; 4)признак обращения к странице = 1, при каждом обращении к странице.

При каждом обращении к памяти выполняется поиск номера виртуальной страницы, содержащие требуемый адрес, затем по этому номеру определяется нужный элемент таблицы страниц и из его извлекается физический номер страницы. Далее анализируется признак присутствия, если данная виртуальная страница находится в ОП, то выполняется преобразование виртуального адреса в физический. Если нужная виртуальная страница в данный момент выгружена на диск, то происходит страничное прерывание. Если в памяти имеется свободная физическая страница, то загрузка выполняется немедленно, если нет - то решается вопрос о том, какую страницу выгрузить из ОП.

# Вопрос 30: Виртуальная память. Сегментная организация памяти.

Виртуальная память - между оперативной памятью и диском перемещаются части ( сегменты, страницы и т.д.) образов процесса.  При нехватке памяти в ОП на диск выгружается только часть образов процесса.

Механизм виртуальной памяти предполагает разделение памяти :

1)страничное

2)сегментное

3)странично-сегментное

Сегментное распределение позволяет обращаться к любым частям программы. В программе определяют следующие сегменты:   
-CS- сегмент код;   
-DS - сегмент данных;  
-SS - сегмент стека;  
-es, fs, gs - дополнительные сегменты;  
CS создает система, DS управляется программно, компилятор определяет его размер. Размеры сегментов разные. Деление виртуального адресного пространства процесса на сегменты осуществляется компилятором.

При загрузке процессов в ОП помещается только часть его сегментов. Для каждого загружаемого сегмента ОС ищет непрерывный участок свободной памяти. Если во время выполнения процесса происходит обращение по виртуальным адресу, относящемуся к сегменту, который отсутствует в памяти, то происходит прерывание.   
ОС организует загрузку нужного сегмента с диска. При отсутствии места в ОП ОС выбирает сегмент на выброску, используя критерии аналогичные при страничной организации.   
На этапе создания процесса время его загрузки в ОП система создает таблицу сегментов. Происходит преобразование виртуального адреса в физический. Физический адрес получается следующим образом: в процессоре имеется регистр CR0, в нем находится адрес начала таблицы сегментов. По таблице сегментов находим по виртуальному адресу соответствующий физический адрес. В таблице сегментов находится адрес начала сегментов. Отыскиваем по адресу начало сегментов + добавляется смещение. Недостаток сегментного распределения – фрагментация(наличие большого числа несмежных участков свободной памяти маленького размера, в связи с чем вновь поступающие программы не могут поместиться ни в одном из участков).

# Вопрос 31: Виртуальная память. Странично-сегментная организация памяти.

Виртуальная память - между оперативной памятью и диском перемещаются части ( сегменты, страницы и т.д.) образов процесса.  При нехватке памяти в ОП на диск выгружается только часть образов процесса.

Механизм виртуальной памяти предполагает разделение памяти :

1)страничное

2)сегментное

3)странично-сегментное

Виртуальное адресное пространство процесса делится на сегменты, перемещение данных между памятью и диском осуществляется страницами. Сначала определяется адрес начала сегмента с помощью CR0 (таблицы сегментов), к этому адресу прибавляется адрес начала виртуальной страницы из таблицы страниц. Затем добавляется смещение относительно начала страницы, т.е. реальный адрес равен: адрес начала сегмента + адрес начало страницы в сегменте + смещение внутри страницы.

# Вопрос 32: Кеш-память.

Кеш-память - небольшая оперативная память, которая находится на отдельном блоке или же микросхеме.   
Различают кеш-данных, кеш-команд. Кеш-память позволяет ускорить обработку в 1,2 раза.   
Содержимое кеш-памяти представляет собой совокупность записей обо всех загруженных в неё элементов данных из основной памяти. Каждая запись об элементе данных включает в себя: значение элемента данных, адрес его в ОП и дополнительную информацию.   
При каждом обращении к ОП просматривается содержимое кеша, чтобы определить находится ли там нужные данные.   
Поиск данных осуществляется по содержимому и возможен 1 из 2 вариантов развития событий: 1)если данные обнаружив в кеше, то произошло попадание, они считаются из кеша; 2)если данные отсутствуют в кеше, то произошел кеш-промах, они считываются из ОП, передаются источнику запроса и одновременно копируется в кэш.

# Вопрос 33: Управление кучей.

Куча - динамическая область памяти, выделяемая процессу при его создании.   
Максимальный размер памяти кучи зависит от типа ОС. Для 32-разрядных Windows приблизительно 2 ГБ, для 64-разрядных - 4 ГБ. Программа используя функции подобные malloc может получать указатели на область памяти принадлежащие кучи. Программа использует кучу для размещения динамически создаваемых структур данных, т.е. при конструировании объекта указываются размеры запрашиваемые под объект памяти.  
Освобождение памяти занятой под какой-либо объект осуществляется командой free.

HeapCreate - создание кучи (аргументы: 1й - создание кучи NULL, HEAP\_NO\_SERIALIZE- куча может использовать только одним потоком; 2й -первоначальный размер в байтах; 3й - максимальный размер кучи в байтах)

HeapAlloc - распределение размера (аргументы: 1й - идентификатор кучи; 2й - флаги; 3й - объём памяти получаемые в байтах)

LPVOID HeapReAlloc - изменение размера кучи ( HANDLE hHeap, DWORD dwFlage - флаг изменения размера блока памяти, LPVOID IpMem - адрес перераспределяемой памяти, DWORD dwBytes - новый размер кучи в байтах )

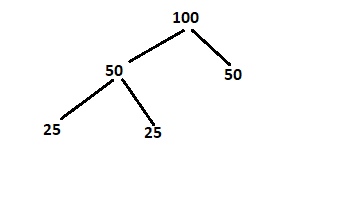
BOOL HeapDestroy -освобождение памяти, занятой кучей (HANDLE hHeap - дескриптор кучи)

DWORD HeapSize(HANDLE hHeap, DWORD Flags - управляющие флаги, LPCVOID IpMem - адрес на блок памяти) - ф-ция определения размера блока памяти

BOOL HeapFree (HANDLE hHeap- дескриптор, DWORD dwFlags - флаги, LPVOID IpMem - адрес освобождаемого блока памяти) - ф-ция освобождения памяти

# Вопрос 34: Метод близнецов и выделение памяти на основе последовательности Фибоначчи.

**Метод близнецов** является схемой выделения памяти. Всю память всё время делим пополам. Пример: есть 10 мб памяти, нужно 40 мб.

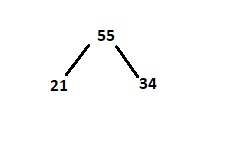


Будем требовать 50 мб , 10 мб не будет использоваться.  
Недостатки : не вся память используется.

**Метод Фибоначчи:**

1. 1 1 2 3 5 8 13 21 34 55 89 134…

Нам нужно 40мб памяти, будем выделять 55мб памяти



# Вопрос 35: Клиент-серверная технология. Протоколы сетевого взаимодействия.

В основе клиент-серверной технологии лежат два основных понятия: клиент и сервер.  
Клиент - компьютер, осуществляющий запрос к серверу на выполнение каких-либо действий или предоставление какой-либо информации. Сервер - компьютер, обычно более мощный, чем компьютер-клиент. Модель функционирования такой системы заключается в следующем: клиент делает запрос серверу, сервер (серверная часть) получает запрос, выполняет его и отсылает результат клиенту (клиентская часть).

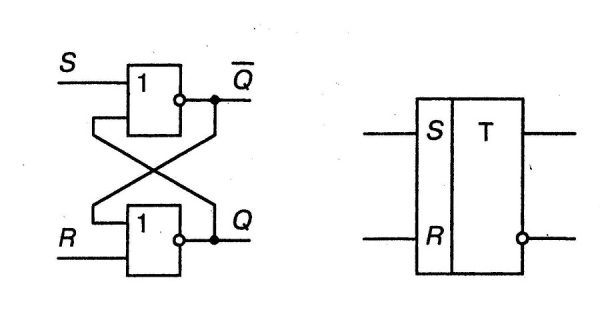
Сокеты включают следующие свойства: 1)поддержка взаимодействия клиента и сервера в синхронном режиме (сервер будет ждать, пока клиент не закончит какую-то задачу).; 2)сокет может работать с несколькими клиентами одновременно; 3)поддержка пространство имен (Active Directory): означает, что я пишу не адрес компьютера.   
  
Работа с клиентом: Для работы с сервером клиент должен инициализировать сокет, получить адрес сервера (getaddrinfo). После получения адреса сервера клиент выдает команду connect; когда подключение установлено, клиент может отправлять и получать данные через сокет используя API функции send, recv.  
  
Работа с сервером: сервер создает сокет( сервер-сокет) для прослушивания сети, это выполняется с помощью команды listen. Если сервер ориентирован на установку соединения, он выполняет в отношении сокета операцию прослушивания (listen) - указывая запас или количество соединений, которая сервер просит сокет удерживать, пока он не сможет их принять. Затем он использует команду accept, чтобы позволить клиенту подключиться к сокету. Сервер узнаёт клиента по выставленному клиентом имени сервера (тогда начинает работать accept). Когда соединение установлено, функция accept возвращает новый сокет, который представляет собой серверный конечный пункт соединения (исходный сокет, используемый для прослушивания, для установки связи не используется, т.к. он используется только для получения запросов на подключение). На стороне сервера используются функции recv, send.

 Для взаимодействия сервера и клиента используют протоколы (TCP и UDP).   
Протокол TCP более медленный, но более надежный. Отличие от UDP - TCP обеспечивает обмен служебной информацией, т.е. клиент не сможет начать передавать данные пока сервер не даст подтверждение готовности и наоборот. На это тратится время, но повышается надежность передачи данных.   
UDP более быстрый, но менее надёжный, обмен подтверждения о готовности передачи не выполняется. В результате клиент посылает данные, не зная, готов ли их принять сервер или нет.

# Вопрос 36: Регистровая память.

Регистровая память - вид компьютерной оперативной памяти, модули которой содержат регистр между микросхемами памяти и системным контроллером памяти. Регистр состоит из отдельных ячеек (ячейка хранит 1 бит информации). Наличие регистров уменьшает электрическую нагрузку на контроллер и позволяет устанавливать больше модулей памяти в одном канале.

Чтобы построить ячейку памяти для хранения сигнала, принцип работы и её следующий: при подаче на её вход 0, на выходе ячейки будет 0 и наоборот, при подаче 1 - будет 1.   
Данная схема называется RS-триггером и представлена на рисунке.(S=1, R=0;на выходе , Q=1)

 Разрешенными комбинациями на входах триггеров являются:   
S=1 R=0  
R=1 S=0  
S=0 R=0  
S=1 R=1

S=1, R=0 в этом случае когда на выходе Q\_ будет 0, а на выходе Q - 1, после того как сигнал на входе S пропадет, единичный сигнал с выхода Q поступает на вход схемы 1(1), тем самым обеспечивается 0 на выходе Q\_, а на выходе Q остается 1. Таким образом триггер сохраняет сигнал на входе S. Данная схема выполняет ф-цию запоминающего устройства и иллюстрирует принцип работы полупроводниковой памяти. Она не обеспечивает сохранение сигнала после отключения питания. На данном принципе реализуется так же регистровая память. Регистр состоит из отдельных ячеек, каждую из которых можно рассматривать как триггер

.   
Из-за использования регистров возникает дополнительная задержка при работе с памятью. Каждое чтение и запись буферизуются в регистре на один такт, прежде чем попадут с шины памяти в чип DRAM, поэтому регистровая память считается на один такт более медленной, чем нерегистровая (UDIMM, unregistered DRAM). Для памяти SDRAM эта задержка существенна только для первого цикла в серии запросов (burst).