Управление памятью

- Память - это важнейший ресурс, требующий тщательного управления со стороны мультипрограммной операционной системы.

Роль память

- Процессор может выполнять инструкции программы только в том случае, если они находятся в памяти.
- Память распределяется как между модулями прикладных программ, так и между модулями самой операционной системы.

Управление памятью

- В ранних ОС - сводилось к загрузке программы и ее данных из некоторого внешнего накопителя (перфоленты, магнитной ленты или магнитного диска) в память.
- С появлением мультипрограммирования ОС - распределяли имеющуюся память между несколькими одновременно выполняющимися программами.

Функции ОС по управлению памятью в мультипрограммной системе

- Отслеживание свободной и занятой памяти;
- Выделение памяти процессам и освобождение памяти по завершении процессов;
- Вытеснение кодов и данных процессов из оперативной памяти на диск (полное или частичное), когда размеры основной памяти не достаточны для размещения в ней всех процессов, и возвращение их в оперативную память, когда в ней освобождается место;
- Настройка адресов программы на конкретную область физической памяти.

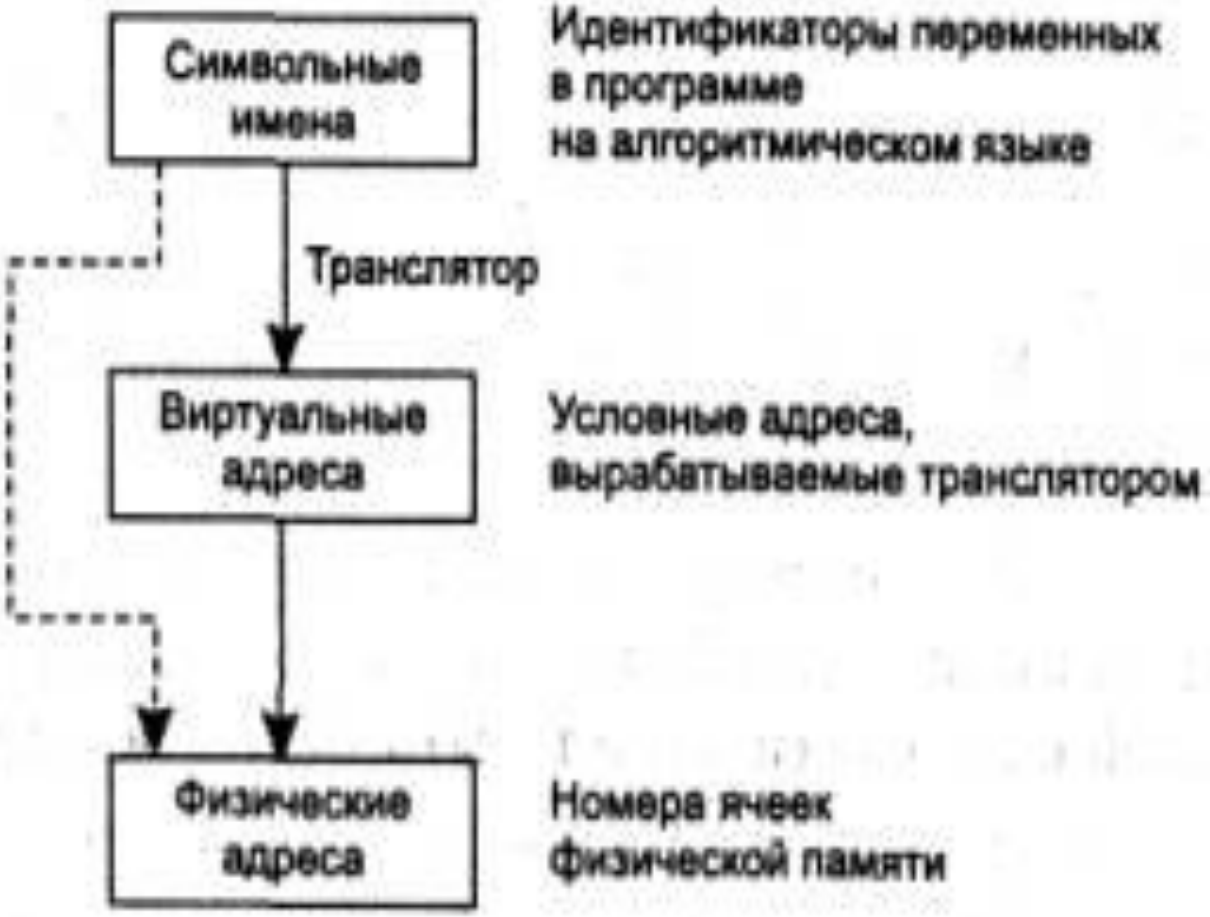
- Динамическое распределение памяти, то есть выполнять запросы приложений на выделение им дополнительной памяти во время выполнения. После того как приложение перестает нуждаться в дополнительной памяти, оно может возвратить ее системе.
- Дефрагментация памяти. Выделение памяти случайной длины в случайные моменты времени из общего пула памяти приводит к фрагментации и, вследствие этого, к неэффективному ее использованию.

- Создание новых служебных информационных структур (описатели процессов и потоков, таблицы распределения ресурсов, буферы, используемые процессами для обмена данными, синхронизирующие объекты и т. п.

- В некоторых ОС во время установки резервируется некоторый фиксированный объем памяти для системных нужд.
- В других же ОС используется более гибкий подход, при котором память для системных целей выделяется динамически.
- Разные подсистемы ОС при создании своих таблиц, объектов, структур и т. п. обращаются к подсистеме управления памятью с запросами.

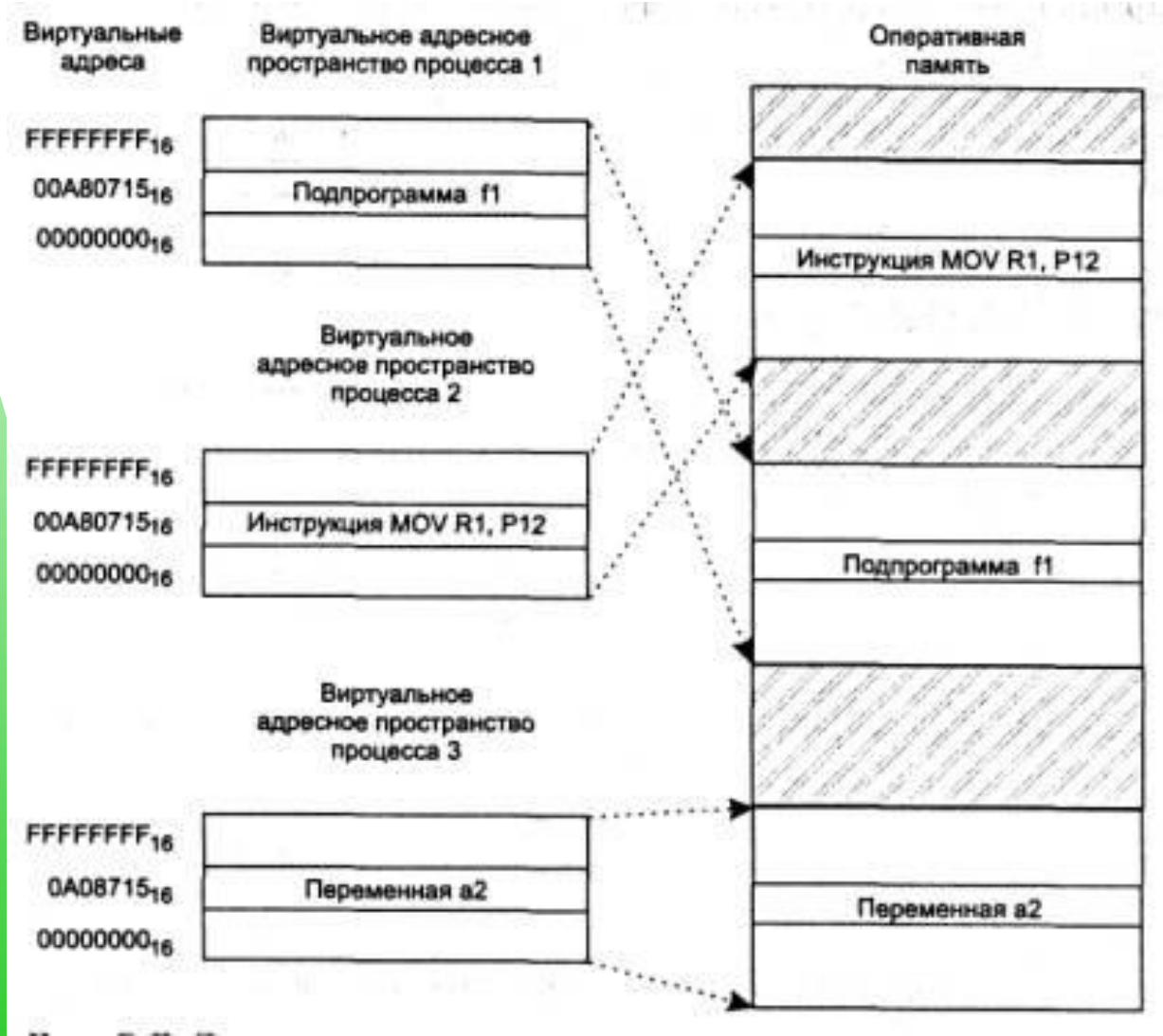
- Защита памяти состоит в том, чтобы не позволить выполняемому процессу записывать или читать данные из памяти, назначенной другому процессу. Эта функция реализуется программными модулями ОС в тесном взаимодействии с аппаратными средствами.

Типы адресов



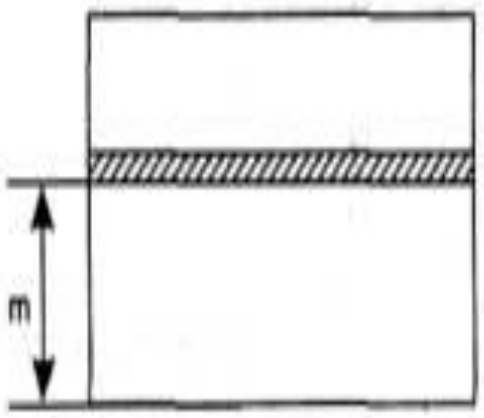
- Совокупность виртуальных адресов процесса называется *виртуальным адресным пространством.*
- Каждый процесс имеет собственное виртуальное адресное пространство — транслятор присваивает виртуальные адреса переменным и кодам каждой программе независимо

Виртуальные адресные пространства нескольких программ



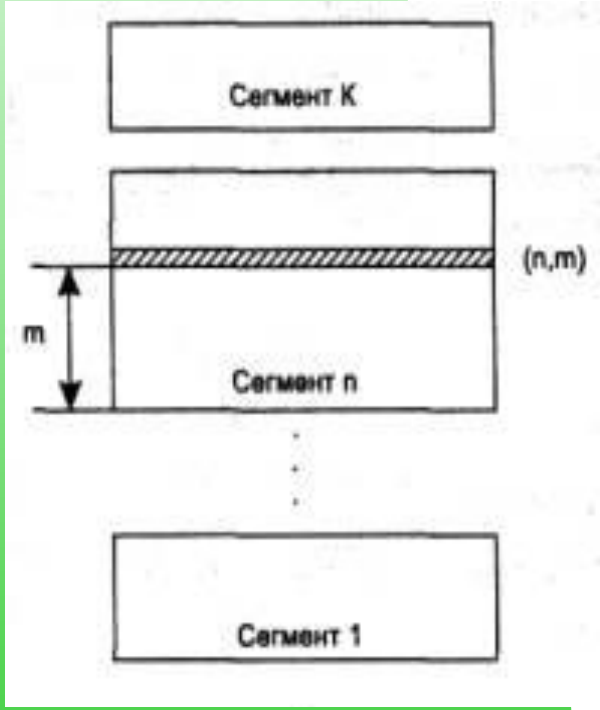
- Совпадение виртуальных адресов переменных и команд различных процессов не приводит к конфликтам, когда эти переменные одновременно присутствуют в памяти, операционная система отображает их на разные физические адреса.

Способы структуризации виртуального адресного пространства



- Непрерывная *линейная* последовательность виртуальных адресов (*плоской (flat)).* Виртуальным адресом является единственное число, представляющее собой смещение относительно начала (обычно это значение 000...000) виртуального адресного пространства. Адрес такого типа называют линейным виртуальным адресом.

Способы структуризации виртуального адресного пространства



- Виртуальное адресное пространство делится на части - сегменты (или секции, или области, или другими терминами). В этом случае помимо линейного адреса может быть использован виртуальный адрес, представляющий собой *пару* чисел *(n, m),* где n определяет сегмент, а m — смещение внутри сегмента.

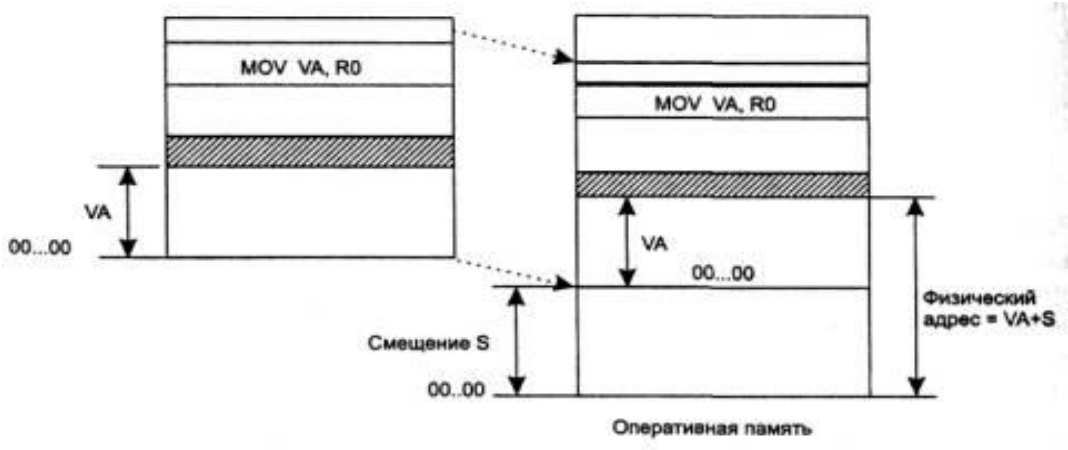
- ОС отображает индивидуальные виртуальные адресные пространства всех одновременно выполняющихся процессов на общую физическую память. При этом ОС отображает либо все виртуальное адресное пространство, либо только определенную его часть.

Подходы к преобразованию виртуальных адресов в физические

- Замена виртуальных адресов на физические выполняется один раз для каждого процесса во время начальной загрузки программы в память. Специальная системная программа — *перемещающий загрузчик* — на основании имеющихся у нее исходных данных о начальном адресе физической памяти, а также информации, предоставленной транслятором об адресно-зависимых элементах программы, выполняет загрузку программы, совмещая ее с заменой виртуальных адресов физическими.

Подходы к преобразованию виртуальных адресов в физические

- Программа загружается в память в неизмененном виде в виртуальные адреса, то есть операнды инструкций и адреса переходов имеют те значения, которые выработал транслятор (виртуальная и физическая память процесса представляют собой единые непрерывные области адресов), операционная система выполняет преобразование виртуальных адресов в физические по следующей схеме. При загрузке операционная система фиксирует смещение действительного расположения программного кода относительно виртуального адресного пространства. Во время выполнения программы при каждом обращении к оперативной памяти выполняется преобразование виртуального адреса в физический



Различают

- *максимально возможное* виртуальное адресное пространство процесса;
- *назначенное (выделенное)* процессу виртуальное адресное пространство.

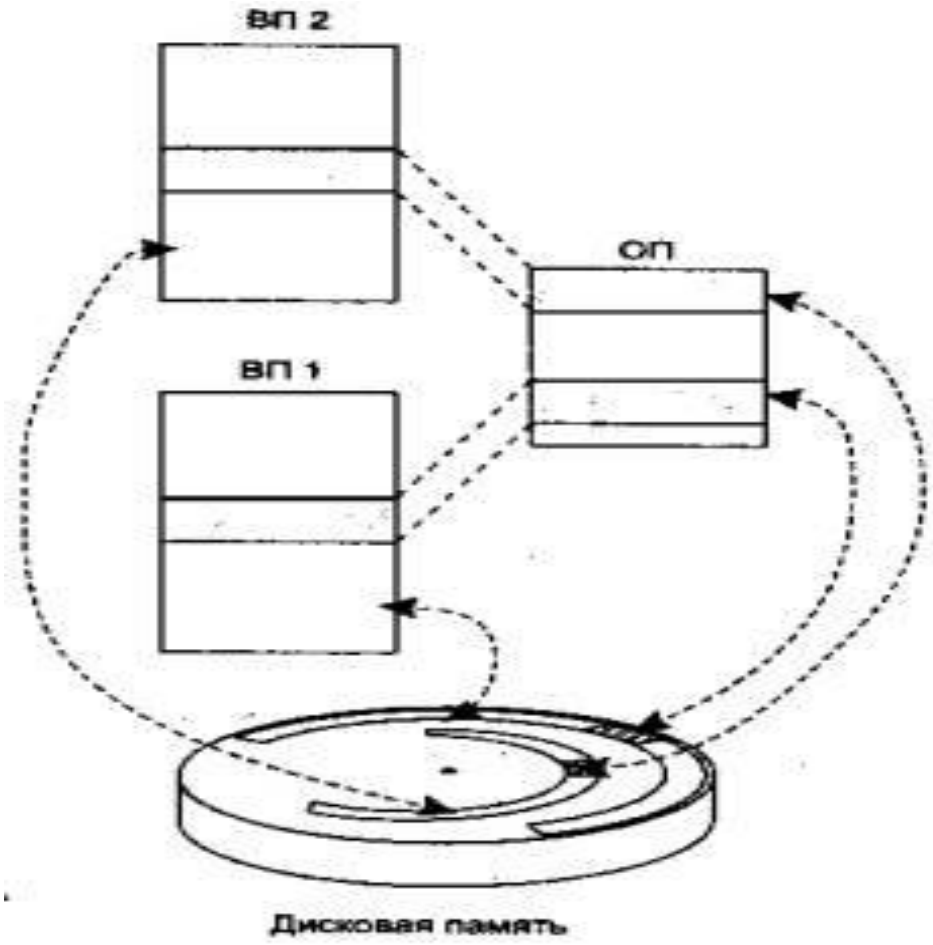
Назначенное виртуальное адресное пространство

- Представляет собой набор виртуальных адресов, действительно нужных процессу для работы.
- ОС обычно следит за корректностью использования процессом виртуальных адресов — процессу не разрешается оперировать с виртуальным адресом, выходящим за пределы назначенных ему сегментов.

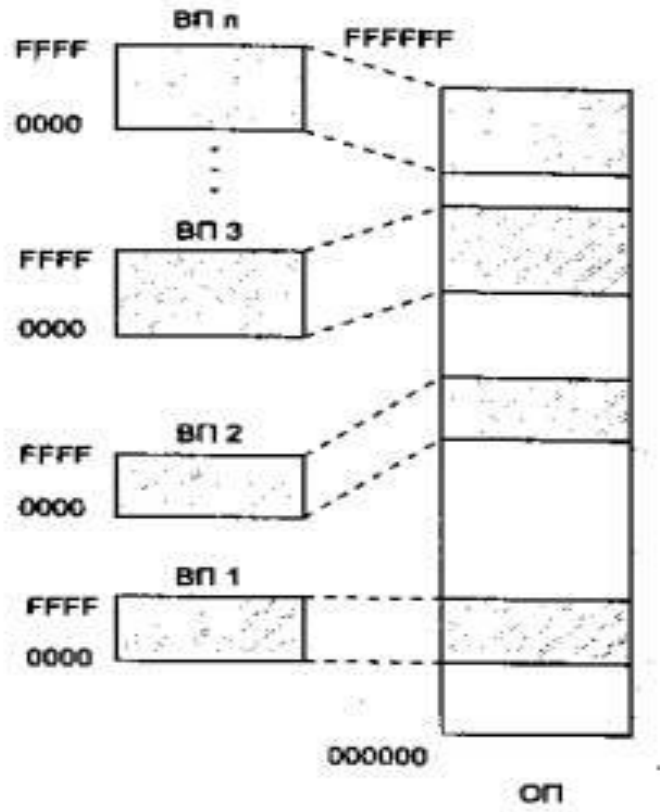
Максимальный размер виртуального адресного пространства

- Ограничивается только разрядностью адреса, присущей данной архитектуре компьютера, и не совпадает с объемом физической памяти, имеющимся в компьютере.

- ОС для хранения данных виртуального адресного пространства процесса, не помещающихся в оперативную память, использует внешнюю память



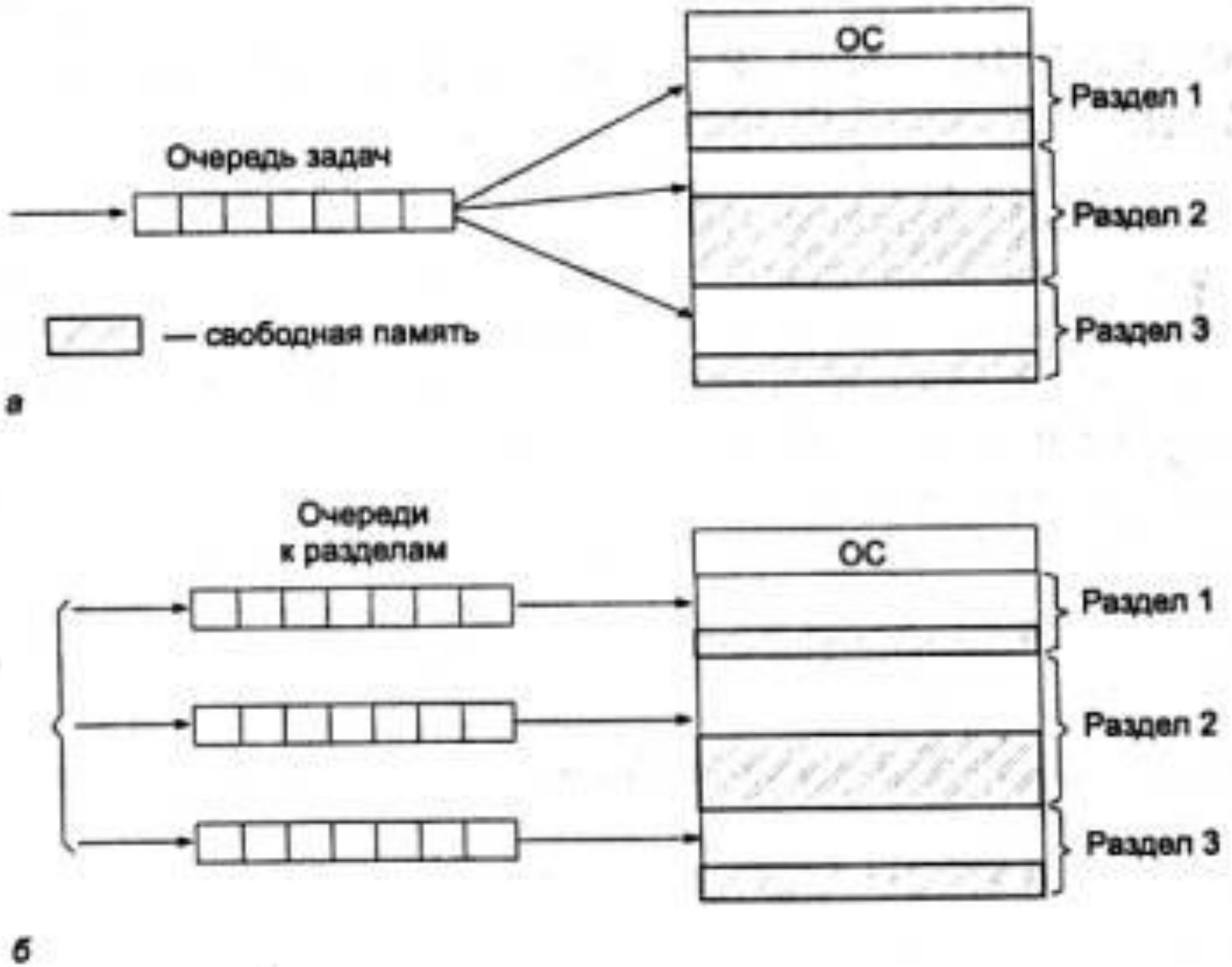
- В мини-компьютерах разрядности поля адреса нередко не хватало для того, чтобы охватить всю имеющуюся оперативную память. Несколько процессов могло быть загружено в память одновременно и целиком.



Алгоритмы распределения памяти

Распределение памяти фиксированными разделами

- Память разбивается на несколько областей фиксированной величины, называемых *разделами.* Такое разбиение может быть выполнено вручную оператором во время старта системы или во время ее установки. После этого границы разделов не изменяются.
- Очередной новый процесс, поступивший на выполнение, помещается либо в общую очередь (рис. а), либо в очередь к некоторому разделу (рис. *б*).


ОС
Раздел 1
Раздел 2
Раздел 3
Очередь задач
— свободная память
а
Очереди
к разделам
ОС
Раздел 1
Раздел 2
Раздел 3
б

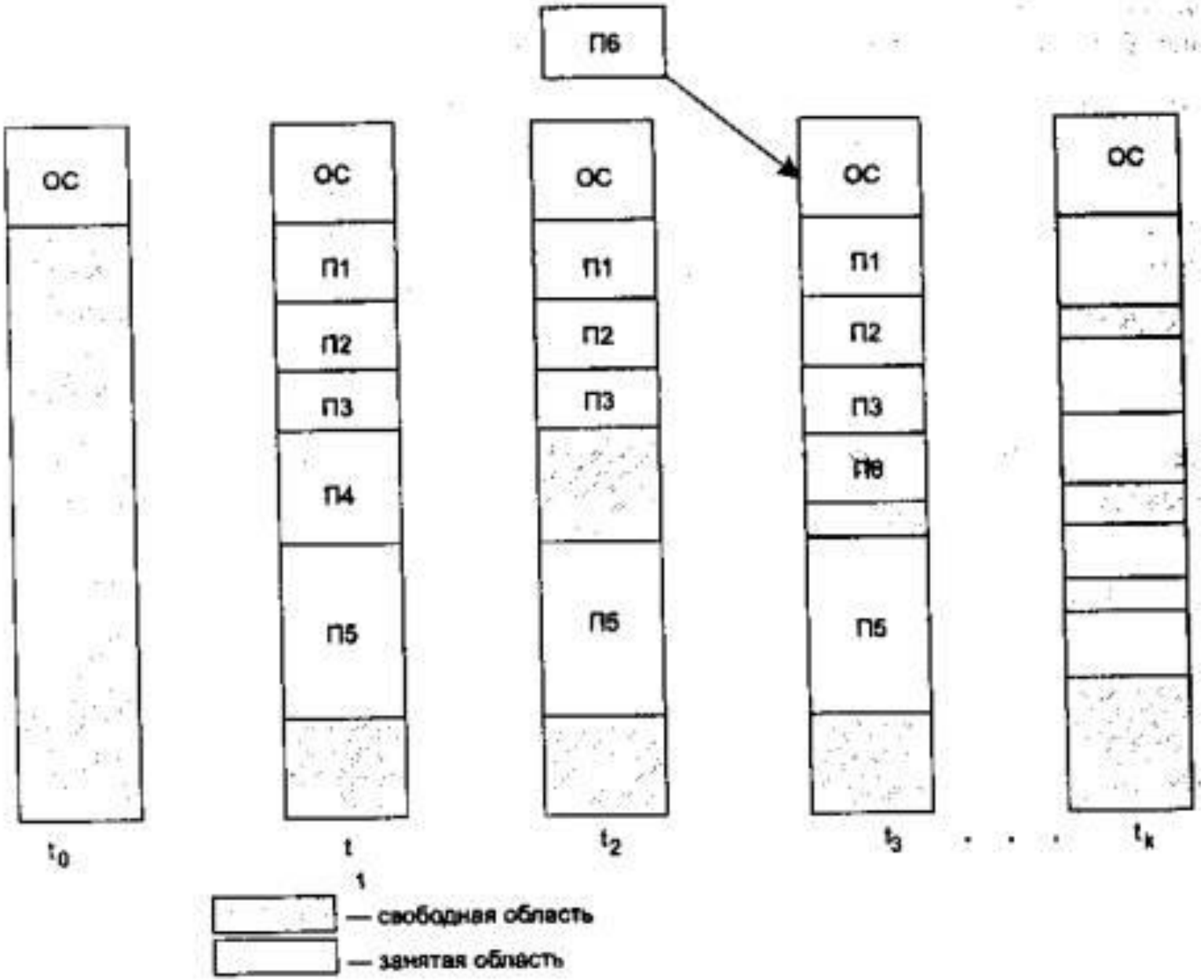
Подсистема управления памятью выполняет задачи:

- Сравнивает объем памяти, требуемый для вновь поступившего процесса, с размерами свободных разделов и выбирает подходящий раздел.
- Осуществляет загрузку программы в один из разделов и настройку адресов. Уже на этапе трансляции разработчик программы может задать раздел, в котором ее следует выполнять. Это позволяет сразу, без использования перемещающего загрузчика, получить машинный код, настроенный на конкретную область памяти.

- Такой способ управления памятью применялся в ранних мультипрограммных ОС. Однако и сейчас метод распределения памяти фиксированными разделами находит применение в системах реального времени.

Распределение памяти динамическими разделами

- Память машины не делится заранее на разделы.
- Сначала вся память, отводимая для приложений, свободна. Каждому вновь поступающему на выполнение приложению на этапе создания процесса выделяется вся необходимая ему память (если достаточный объем памяти отсутствует, то приложение не принимается на выполнение и процесс для него не создается). После завершения процесса память освобождается, и на это место может быть загружен другой процесс. Таким образом, в произвольный момент времени оперативная память представляет собой случайную последовательность занятых и свободных участков (разделов) произвольного размера.


П6
ОС
ОС
П1
П2
П3
П4
П5
ОС
П1
П2
П3
П5
ОС
П1
П2
П3
П5
ОС
t_0
t
1
t_2
t_3
t_k
— свободная область
— занятая область

Функции операционной системы

- Ведение таблиц свободных и занятых областей, в которых указываются начальные адреса и размеры участков памяти.
- При создании нового процесса — анализ требований к памяти, просмотр таблицы свободных областей и выбор раздела, размер которого достаточен для размещения кодов и данных нового процесса.
- Загрузка программы в выделенный ей раздел и корректировка таблиц свободных и занятых областей Данный способ предполагает, что программный код не перемещается во время выполнения, а значит, настройка адресов может быть проведена единовременно во время загрузки.
- После завершения процесса корректировка таблиц свободных и занятых областей.

Недостаток

- фрагментация памяти

Фрагментация

- наличие большого числа несмежных участков свободной памяти очень маленького размера (фрагментов)

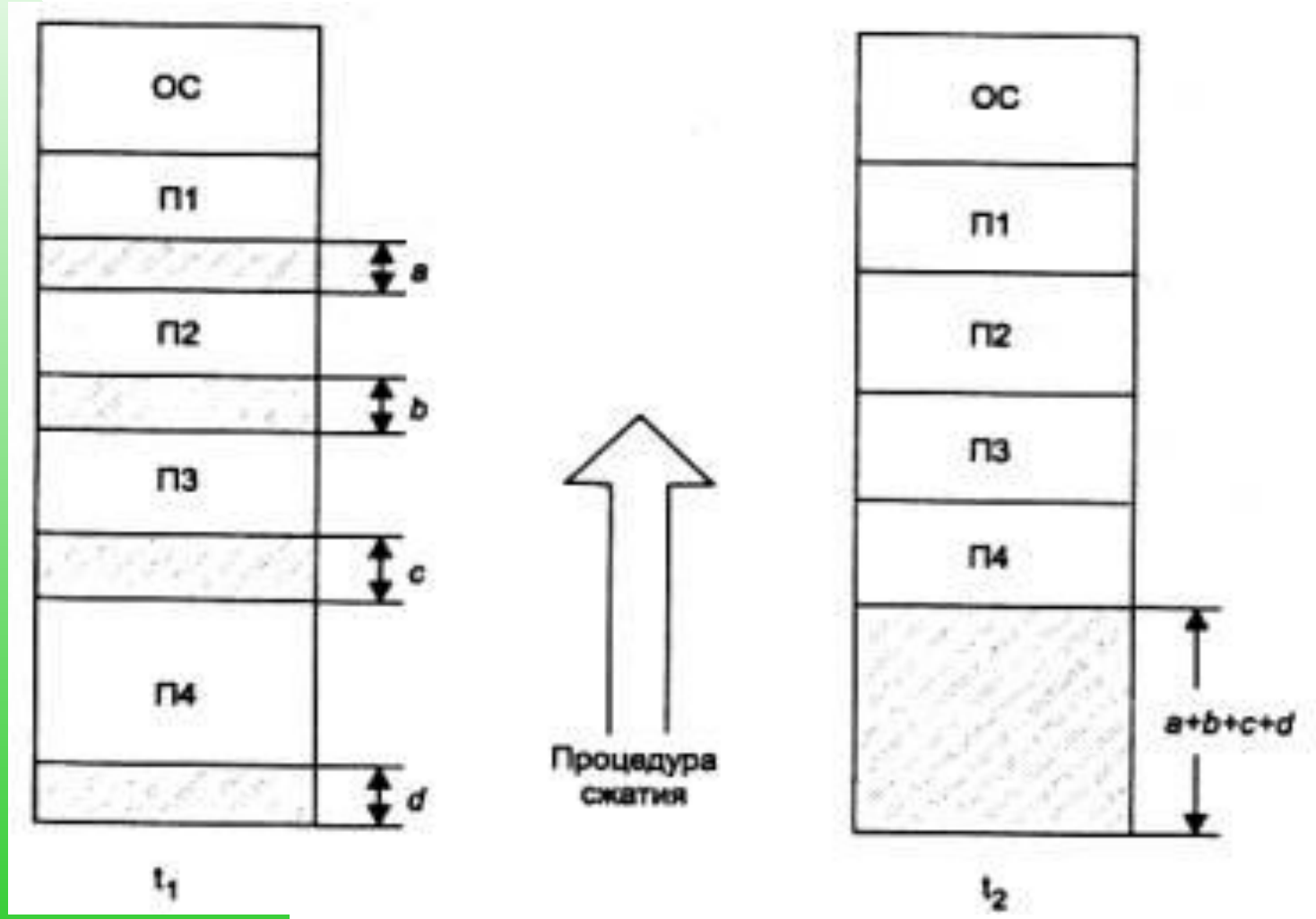
- Настолько маленького участка, что ни одна из вновь поступающих программ не может поместиться ни в одном из участков, хотя суммарный объем фрагментов может составить значительную величину, намного превышающую требуемый объем памяти.

- Распределение памяти динамическими разделами лежит в основе подсистем управления памятью операционных системах 60-70-х годов, например OS/360.

Перемещаемые разделы

- Это перемещение всех занятых участков в сторону старших или младших адресов, так, чтобы вся свободная память образовала единую свободную область

Перемещаемые разделы



Сжатие

- Это копирование содержимого разделов из одного места памяти в другое, корректируя таблицы свободных и занятых областей

- Сжатие может выполняться либо при каждом завершении процесса, либо только тогда, когда для вновь создаваемого процесса нет свободного раздела достаточного размера.

- Программы перемещаются по оперативной памяти в ходе своего выполнения, то невозможно выполнить настройку адресов с помощью перемещающего загрузчика.

- Процедура сжатия и приводит к более эффективному использованию памяти, но требует значительного времени для перемещения фрагментов.
- Такой подход был использован в ранних версиях OS/2, в которых память распределялась сегментами, а возникавшая при этом фрагментация устранялась путем периодического перемещения сегментов.

Свопинг и виртуальная память

- Объем оперативной памяти ограничивает число одновременно выполняющихся программ и размеры их виртуальных адресных пространств.
- Когда задачи мультипрограммной ОС являются вычислительными, то для хорошей загрузки процессора может оказаться достаточным всего 3-5 задач.
- Если вычислительная система загружена выполнением интерактивных задач, то для эффективного использования процессора может потребоваться уже несколько десятков, а то и сотен задач.

- Когда имеющейся оперативной памяти недостаточно, был предложен метод организации вычислительного процесса, при котором образы некоторых процессов целиком или частично временно выгружаются на диск.

- Виртуализация оперативной памяти дисковой памятью позволяет повысить уровень мультипрограммирования — объем оперативной памяти компьютера теперь не столь жестко ограничивает количество одновременно выполняемых процессов, поскольку суммарный объем памяти, занимаемой образами этих процессов, может существенно превосходить имеющийся объем оперативной памяти.

Виртуализация оперативной памяти

- размещение данных в запоминающих устройствах разного типа, например часть кодов программы — в оперативной памяти, а часть — на диске;
- выбор образов процессов или их частей для перемещения из оперативной памяти на диск и обратно;
- перемещение по мере необходимости данных между памятью и диском;
- преобразование виртуальных адресов в физические.

- Все действия по организации совместного использования диска и оперативной памяти — выделение места для перемещаемых фрагментов, настройка адресов, выбор кандидатов на загрузку и выгрузку — осуществляются операционной системой и аппаратурой процессора автоматически, без участия программиста, и никак не сказываются на логике работы приложений.

- Пользователи столкнулись с проблемой размещения в памяти программы, размер которой превышает имеющуюся в наличии свободную память.
- Разбиение программы на части, называется оверлеем.
- Когда первый оверлей заканчивал свое выполнение, он вызывал другой оверлей. Все оверлеи хранились на диске и перемещались между памятью и диском средствами операционной системы на основании явных директив программиста, содержащихся в программе. Этот способ, несмотря на внешнее сходство, имеет принципиальное отличие от виртуальной памяти, заключающееся в том, что разбиение программы на части и планирование их загрузки в оперативную память должны были выполняться заранее программистом во время написания программы.

Подходы виртуализации памяти

- *свопинг (swapping)* — образы процессов выгружаются на диск и возвращаются в оперативную память *целиком;*
- *виртуальная память (virtual memory)* — между оперативной памятью и диском перемещаются *части* (сегменты, страницы и т. п.) образов процессов.

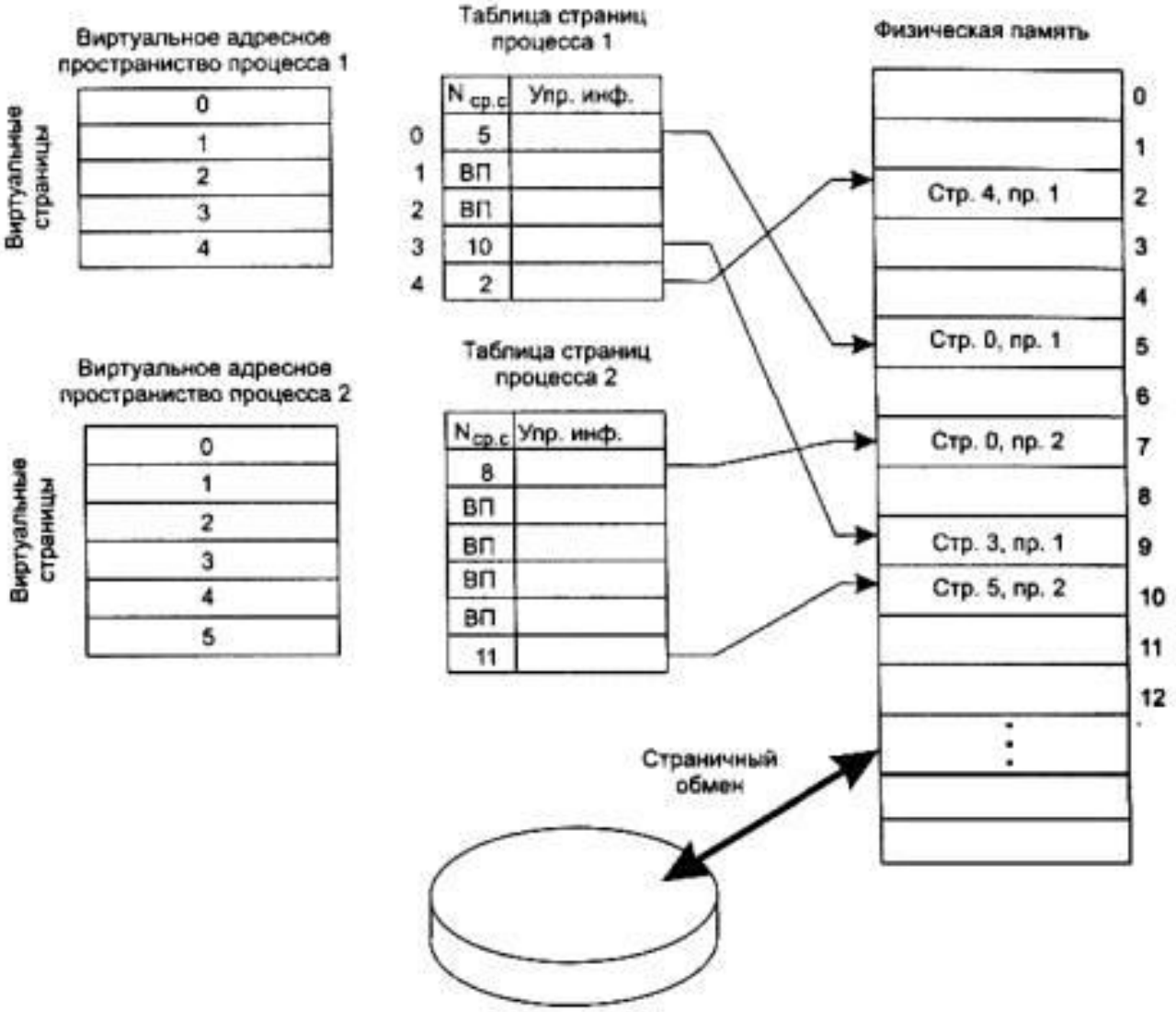
Реализация виртуальной памяти

- *Страничная виртуальная память* организует перемещение данных между памятью и диском страницами — частями виртуального адресного пространства, фиксированного и сравнительно небольшого размера.
- *Сегментная виртуальная память* предусматривает перемещение данных сегментами — частями виртуального адресного пространства произвольного размера, полученными с учетом смыслового значения данных.
- *Сегментно-страничная виртуальная память* использует двухуровневое деление: виртуальное адресное пространство делится на сегменты, а затем сегменты делятся на страницы. Единицей перемещения данных здесь является страница. Этот способ управления памятью объединяет в себе элементы обоих предыдущих подходов.

- Для временного хранения сегментов и страниц на диске отводится либо специальная область, либо специальный файл (страничный файл).
- Чем больше страничный файл, тем больше приложений может одновременно выполнять ОС (при фиксированном размере оперативной памяти)

- Увеличение числа одновременно работающих приложений за счет увеличения размера страничного файла замедляет их работу, так как значительная часть времени при этом тратится на перекачку кодов и данных из оперативной памяти на диск и обратно.
- Размер страничного файла в современных ОС является настраиваемым параметром.

Страничное распределение



- Виртуальное адресное пространство каждого процесса делится на части одинакового, фиксированного для данной системы размера, называемые *виртуальными страницами (virtual pages).* Размер виртуального адресного пространства процесса не кратен размеру страницы, поэтому последняя страница каждого процесса дополняется фиктивной областью.
- Вся оперативная память машины также делится на части такого же размера, называемые *физическими страницами* (или блоками, или кадрами).
- Размер страницы выбирается равным степени двойки: 512, 1024, 4096 байт и т. д.

Для каждого процесса операционная система создает *таблицу страниц* — информационную структуру, содержащую записи обо всех виртуальных страницах процесса.

Запись таблицы, называемая *дескриптором страницы,* включает следующую информацию:

- *номер физической страницы,* в которую загружена данная виртуальная страница;
- *признак присутствия,* устанавливаемый в единицу, если виртуальная страница находится в оперативной памяти;
- *признак модификации* страницы, который устанавливается в единицу всякий раз, когда производится запись по адресу, относящемуся к данной странице;
- *признак обращения* к странице, называемый также *битом доступа,* который устанавливается в единицу при каждом обращении по адресу, относящемуся к данной странице.

Аппаратная схема процессора

- Из специального регистра процессора извлекается адрес таблицы страниц активного процесса. На основании начального адреса таблицы страниц, номера виртуальной страницы p (старшие разряды виртуального адреса) и длины отдельной записи в таблице страниц L (системная константа) определяется адрес нужного дескриптора в таблице страниц: a=AT+(p*L).
- Из этого дескриптора извлекается номер соответствующей физической страницы — n.
- К номеру физической страницы присоединяется смещение s (младшие разряды виртуального адреса).

- Типичная машинная инструкция требует 3-4 обращений к памяти (выборка команды, извлечение операндов, запись результата). И при каждом обращении происходит либо преобразование виртуального адреса в физический, либо обработка страничного прерывания.

Схема преобразования виртуального адреса в физический

- Страничное распределение памяти может быть реализовано в упрощенном варианте, без выгрузки страниц на диск. В этом случае все виртуальные страницы всех процессов постоянно находятся в оперативной памяти.
- Такой режим работы системы управления памятью используется в некоторых специализированных ОС, когда требуется высокая реактивность системы и способность выполнять переменный набор приложений (пример — ОС семейства Novell NetWare 3.x и 4.x).
- Достоинство страничной организации позволяет успешно бороться с фрагментацией физической памяти.

Сегментное распределение

- Виртуальное адресное пространство процесса делится на части — *сегменты,* размер которых определяется с учетом смыслового значения содержащейся в них информации.
- Отдельный сегмент может представлять собой подпрограмму, массив данных и т. п.
- Деление виртуального адресного пространства на сегменты осуществляется компилятором на основе указаний программиста или по умолчанию.

- При загрузке процесса в оперативную память помещается только часть его сегментов, полная копия виртуального адресного пространства находится в дисковой памяти.
- Для каждого загружаемого сегмента операционная система подыскивает непрерывный участок свободной памяти достаточного размера.

- Система с сегментной организацией функционирует аналогично системе со страничной организацией: при каждом обращении к оперативной памяти выполняется преобразование виртуального адреса в физический, время от времени происходят прерывания, связанные с отсутствием нужных сегментов в памяти, при необходимости освобождения памяти некоторые сегменты выгружаются.

- Отличие сегментной организации памяти от страничной является возможность задания дифференцированных прав доступа процесса к его сегментам.
- Например, один сегмент данных, содержащий исходную информацию для приложения, может иметь права доступа «только чтение», а сегмент данных, представляющий результаты, — «чтение и запись». Это свойство дает принципиальное преимущество сегментной модели памяти над страничной.

Недостатки сегментного распределения

- **Фрагментация**, которая возникает из-за непредсказуемости размеров сегментов
- **Избыточность,** во многих случаях для работы программы вовсе не требуется загружать весь сегмент целиком, достаточно было бы одной или двух страниц

Сегментно-страничное распределение

- Так же как и при сегментной организации памяти, виртуальное адресное пространство процесса разделено на сегменты. Это позволяет определять разные права доступа к разным частям кодов и данных программы.
- Перемещение данных между памятью и диском осуществляется не сегментами, а страницами. Для этого каждый виртуальный сегмент и физическая память делятся на страницы равного размера, что позволяет более эффективно использовать память, сократив до минимума фрагментацию.

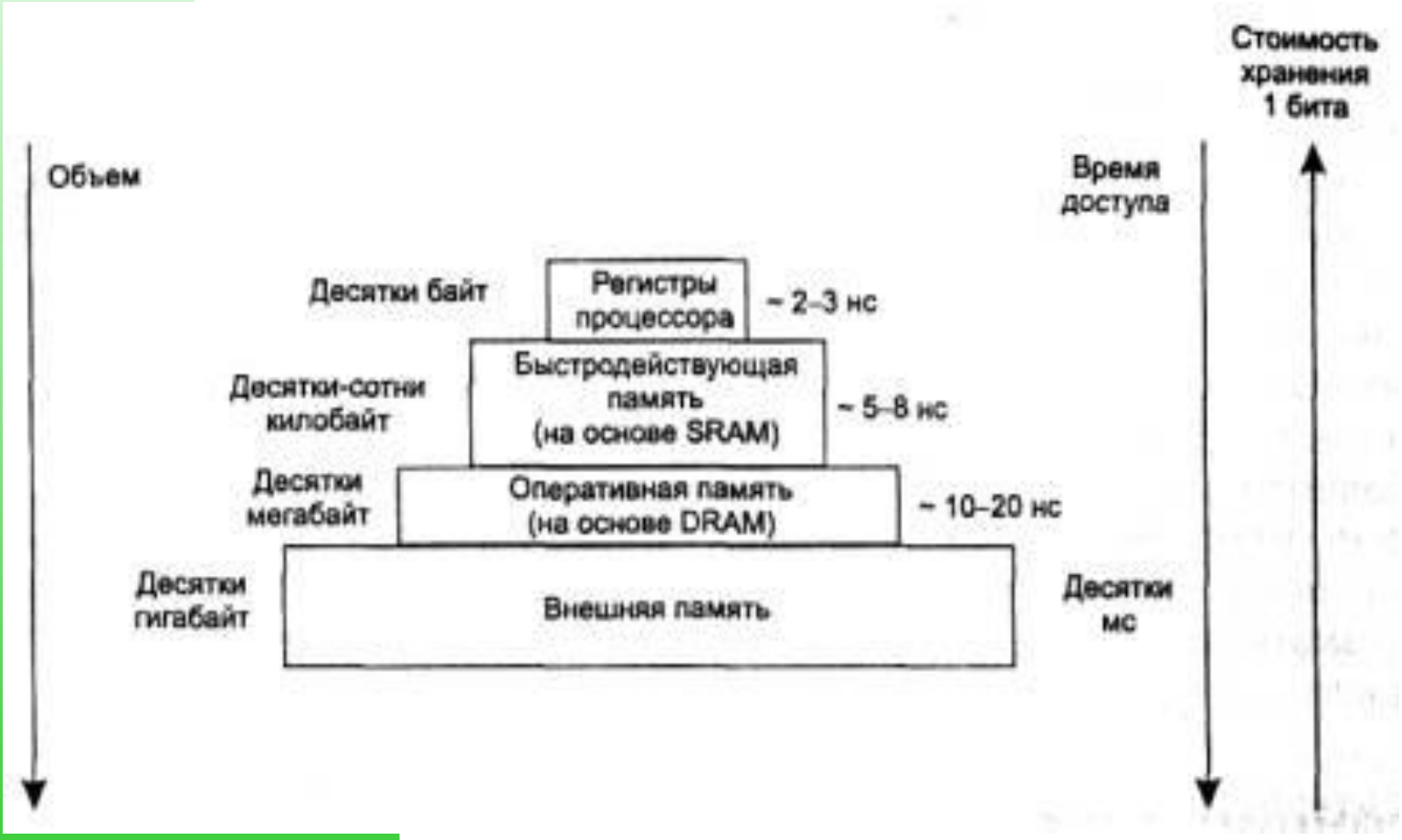
Способы сегментно-страничной организации

- Линейным виртуальным адресом, который равен сдвигу данного байта относительно границы общего линейного виртуального пространства
- Парой чисел, одно из которых является номером сегмента, а другое — смещением относительно начала сегмента

Кэширование данных. Иерархия запоминающих устройств

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

Иерархия запоминающих устройств



- Таким образом, чем больше объем устройства, тем менее быстродействующим оно является.
- Стоимость хранения данных в расчете на один бит также увеличивается с ростом быстродействия устройств.
- Однако пользователю хотелось бы иметь и недорогую, и быструю память. Кэш-память представляет некоторое компромиссное решение этой проблемы

Кэш-память

- *Кэш-память,* или просто *кэш (cache),* — 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 уменьшить среднее время доступа к данным и экономить более дорогую быстродействующую память.

Кэширование

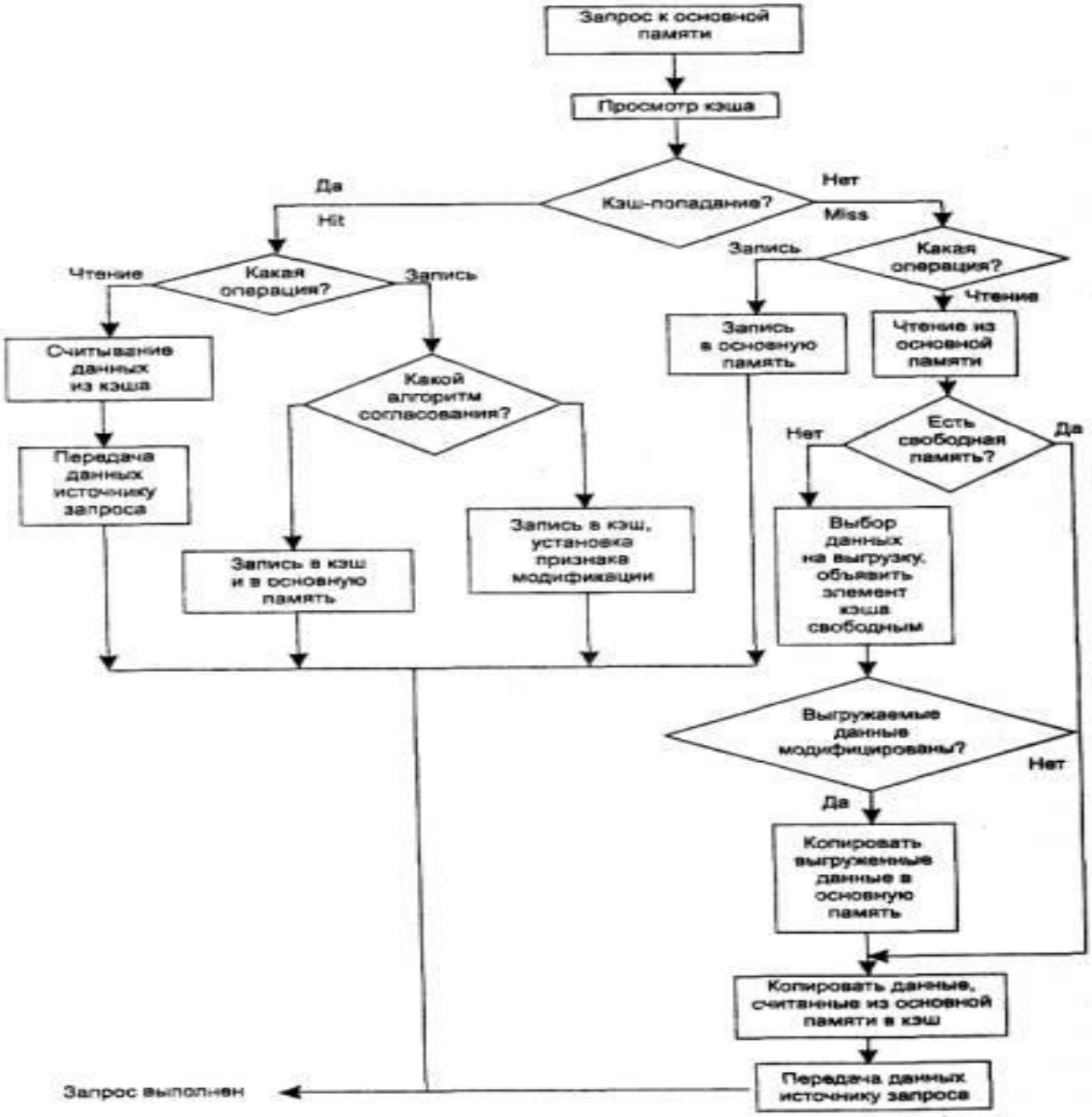
- Это универсальный метод, пригодный для ускорения доступа к оперативной памяти, к диску и к другим видам запоминающих устройств.

- Кэширование применяется для уменьшения среднего времени доступа к оперативной памяти, то в качестве кэша используют быстродействующую статическую память.
- Если кэширование используется системой ввода-вывода для ускорения доступа к данным, хранящимся на диске, то в этом случае роль кэш-памяти выполняют буферы в оперативной памяти, в которых оседают наиболее активно используемые данные.

Принцип действия кэш-памяти

Структура кэш-памяти

Адрес данных в основной памяти	Данные	Управляющая информация


Запрос к основной памяти
Просмотр кэша
Кэш-попадание?
Да
Hit
Нет
Miss
Какая операция?
Чтение
Запись
Какая операция?
Чтение
Запись
Считывание данных из кэша
Передача данных источнику запроса
Какой алгоритм согласования?
Запись в кэш и в основную память
Запись в кэш, установка признака модификации
Запись в основную память
Чтение из основной памяти
Есть свободная память?
Нет
Да
Выбор данных на выгрузку, объявить элемент кэша свободным
Выгружаемые данные модифицированы?
Нет
Да
Копировать выгруженные данные в основную память
Копировать данные, считанные из основной памяти в кэш
Передача данных источнику запроса
Запрос выполнен

Выводы

- Оперативная память является важнейшим ресурсом вычислительной системы, требующим тщательного управления со стороны мультипрограммной операционной системы. Особая роль памяти объясняется тем, что процессор может выполнять инструкции программы только в том случае, если они находятся в памяти.

- Память распределяется как между модулями прикладных программ, так и между модулями самой операционной системы.
- Функциями ОС по управлению памятью в мультипрограммной системе являются:

  - отслеживание наличия свободной и занятой памяти;

  - выделение памяти процессам и освобождение памяти при завершении процессов;

  - вытеснение кодов и данных процессов из оперативной памяти на диск (полное или частичное), когда размеры основной памяти не достаточны для размещения в ней всех процессов, и возвращение их в оперативную намять, когда в ней освобождается место;

  - настройка адресов программы на конкретную область физической памяти;

  - защита памяти процессов от взаимного вмешательства.

- На разных этапах жизненного цикла программы для представления переменных и кодов требуются три типа адресов: символьные (имена, используемые программистом), виртуальные (условные числа, вырабатываемые компилятором) и физические (адреса фактического размещения в оперативной памяти).
- Совокупность виртуальных адресов процесса называется виртуальным адресным пространством. Диапазон возможных адресов виртуального пространства у всех процессов является одним и тем же.
- Виртуальное адресное пространство может быть плоским (линейным) или структурированным.

- Необходимо различать максимально возможное виртуальное адресное пространство процесса, которое определяется только разрядностью виртуального адреса и архитектурой компьютера, и назначенное (выделенное) процессу виртуальное адресное пространство, состоящее из набора виртуальных адресов, действительно нужных процессу для работы.
- Виртуальное адресное пространство процесса делится на две непрерывные части: системную и пользовательскую. Системная часть является общей для всех процессов, в ней размещаются коды и данные операционной системы.

- Наиболее эффективным способом управления памятью является виртуальная память, вытеснившая в современных ОС методы распределения памяти фиксированными, динамическими или перемещаемыми разделами.
- Виртуальная память использует дисковую память для временного хранения не помещающихся в оперативную память данных и кодов выполняемых процессов ОС.

- В настоящее время все множество реализаций виртуальной памяти может быть представлено тремя классами:

- страничная виртуальная память организует перемещение данных между памятью и диском страницами — частями виртуального адресного пространства фиксированного и сравнительно небольшого размера (достоинства — высокая скорость обмена, низкий уровень фрагментации; недостатки — сложно организовать защиту данных, разделенных на части механически);

- сегментная виртуальная память предусматривает перемещение данных сегментами — частями виртуального адресного пространства произвольного размера, полученными с учетом смыслового значения данных (достоинства — «осмысленность» сегментов упрощает их защиту; недостатки — медленное преобразование адреса, высокий уровень фрагментации);

- сегментно-страничная виртуальная память сочетает достоинства обоих предыдущих подходов.

- Сегменты виртуальной памяти могут быть разделяемыми между несколькими процессами. Разделяемые сегменты используются либо для экономии физической памяти, когда несколько пользователей работают с одним кодовым сегментом приложения, либо в качестве средства обмена данными между процессами.
- Для ускорения доступа к данным в вычислительных системах широко используется принцип кэширования. В компьютерах существует иерархия запоминающих устройств, в которой нижний уровень занимают емкая, но относительно медленная дисковая память, затем располагается оперативная память, а верхний уровень составляет сверхоперативная память процессорного кэша. Каждый уровень памяти (кроме нижнего) выполняет роль кэша по отношению к нижележащему.

- Каждая запись в кэш-памяти об элементе данных включает в себя:

  - значение элемента данных;

  - адрес, который этот элемент данных имеет в основной памяти;

  - дополнительную информацию, которая используется для реализации алгоритма замещения данных в кэше и обычно включает признак модификации и признак действительности данных.

- При кэшировании данных из оперативной памяти широко используются две основные схемы отображения: случайное отображение и детерминированное отображение

- При случайном отображении элемент оперативной памяти может быть размещен в произвольном месте кэш-памяти. Для того чтобы в дальнейшем можно было найти нужные данные в кэше, они помещаются туда вместе со своим адресом оперативной памяти.
- Детерминированный (прямой) способ отображения предполагает, что любой элемент основной памяти всегда отображается в одно и то же место кэш-памяти. В этом случае кэш-память разделена на строки, каждая из которых предназначена для хранения одной записи об одном элементе данных и имеет свой номер.
- Во многих современных процессорах кэш-память строится на основе сочетания этих двух подходов, что позволяет найти компромисс между сравнительно низкой стоимостью кэша с прямым отображением и интеллектуальностью алгоритмов замещения в кэше со случайным отображением.