Операционные системы и среды

Лекция №5

Управление памятью

Иерархия памяти

- Память вычислительной машины представляет собой иерархию запоминающих устройств (ЗУ), отличающихся средним временем доступа к данным, объемом и стоимостью хранения одного бита (рис. 3.1)

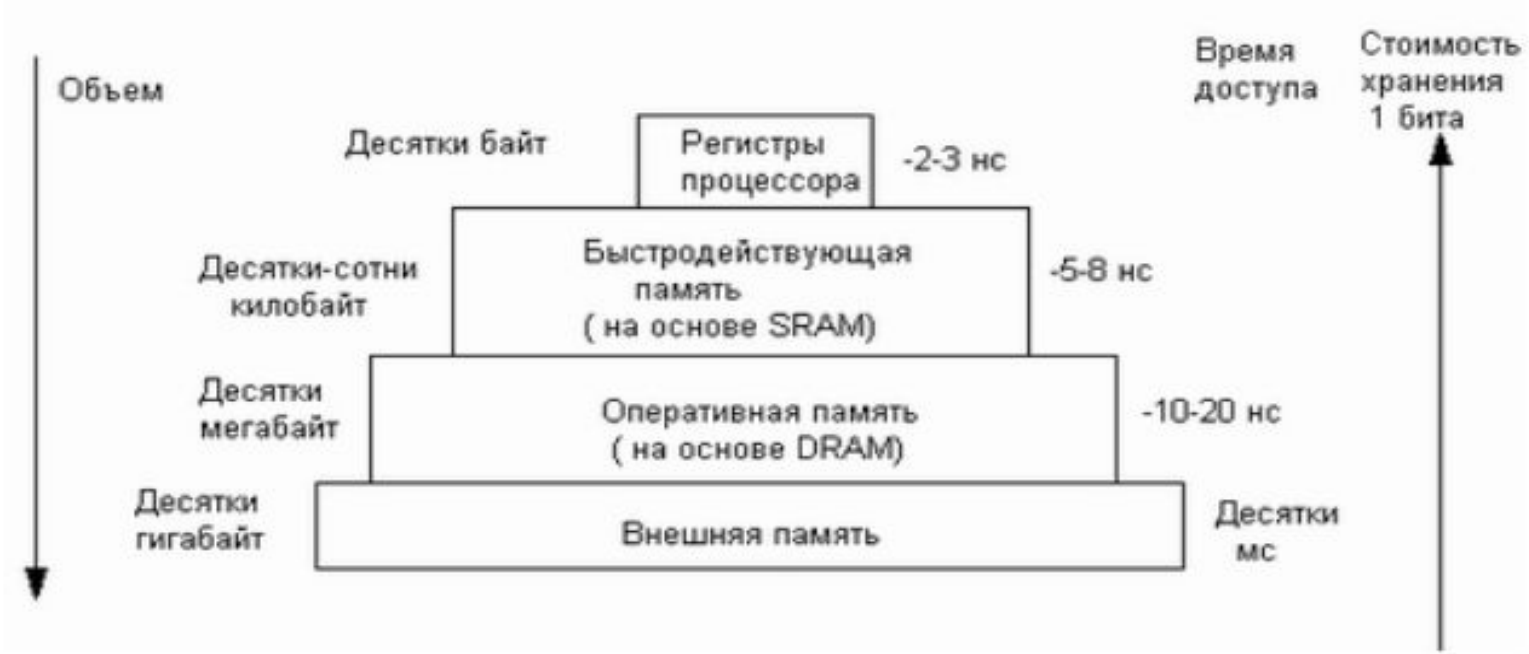


Рис. 3.1. Иерархия запоминающих устройств

- Фундаментом этой пирамиды запоминающих устройств служит внешняя память, как правило, представляемая жестким диском. Она имеет большой объем (десятки и сотни гигабайт), но скорость доступа к данным является невысокой. Время доступа к диску измеряется миллисекундами.
- На следующем уровне располагается более быстродействующая (время доступа1 равно примерно 10-20 наносекундам) и менее объемная (от десятков мегабайт до нескольких гигабайт) оперативная память, реализуемая на относительно медленной динамической памяти DRAM. Для хранения данных, к которым необходимо обеспечить быстрый доступ, используются компактные быстродействующие запоминающие устройства на основе статической памяти SRAM (Кэш), объем которых составляет от нескольких десятков до нескольких сотен килобайт, а время доступа к данным обычно не превышает 8 нс.

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

- Верхушку в этой пирамиде составляют внутренние регистры процессора, которые также могут быть использованы для промежуточного хранения данных. Общий объем регистров составляет несколько десятков байт, а время доступа определяется быстродействием процессора и равно в настоящее время примерно 2-3 нс.
- Все перечисленные характеристики ЗУ быстро изменяются по мере совершенствования вычислительной аппаратуры. В данном случае важны не абсолютные значения времени доступа или объема памяти, а их соотношение для разных типов запоминающих устройств.
- Таким образом, можно констатировать печальную закономерность — чем больше объем устройства, тем менее быстродействующим оно является. Более того, 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.

Управление памятью

- Под памятью (memory) как правило подразумевается оперативная память компьютера. В отличие от памяти жесткого диска, которую называют внешней памятью (storage), оперативной памяти для сохранения информации требуется постоянное электропитание.
- Память является важнейшим ресурсом, требующим тщательного управления со стороны мультипрограммной операционной системы. Особая роль памяти объясняется тем, что процессор может выполнять инструкции протравы 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 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.
- Помимо первоначального выделения памяти процессам при их создании ОС должна также заниматься динамическим распределением 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. Выделение памяти случайной длины в случайные моменты времени из общего пула памяти приводит к фрагментации и, вследствие этого, к неэффективному ее использованию. Дефрагментация памяти тоже является функцией операционной системы.

Типы адресации

- Для идентификации переменных и команд на разных этапах жизненного цикла программы используются символьные имена (метки), виртуальные адреса и физические адреса (рис. 5.1):
  - Символьные имена присваивает пользователь при написании программы на алгоритмическом языке или ассемблере.
  - Виртуальные адреса, называемые иногда математическими, или логическими адресами, вырабатывает транслятор, переводящий программу на машинный язык. Поскольку во время трансляции в общем случае не известно, в какое место оперативной памяти будет загружена программа, то транслятор присваивает переменным и командам виртуальные (условные) адреса, обычно считая по умолчанию, что начальным адресом программы будет нулевой адрес.
  - Физические адреса соответствуют номерам ячеек оперативной памяти, где в действительности расположены или будут расположены переменные и команды.

- Совокупность виртуальных адресов процесса называется виртуальным адресным пространством.

Рис. 3.1. Типы адресов

- Диапазон возможных адресов виртуального пространства у всех процессов является одним и тем же. Например, при использовании 32-разрядных виртуальных адресов этот диапазон задается границами 00000000 16и FFFFFFFF16. Тем не менее каждый процесс имеет собственное виртуальное адресное пространство — транслятор присваивает виртуальные адреса переменным и кодам каждой программе независимо (рис. 3.2).

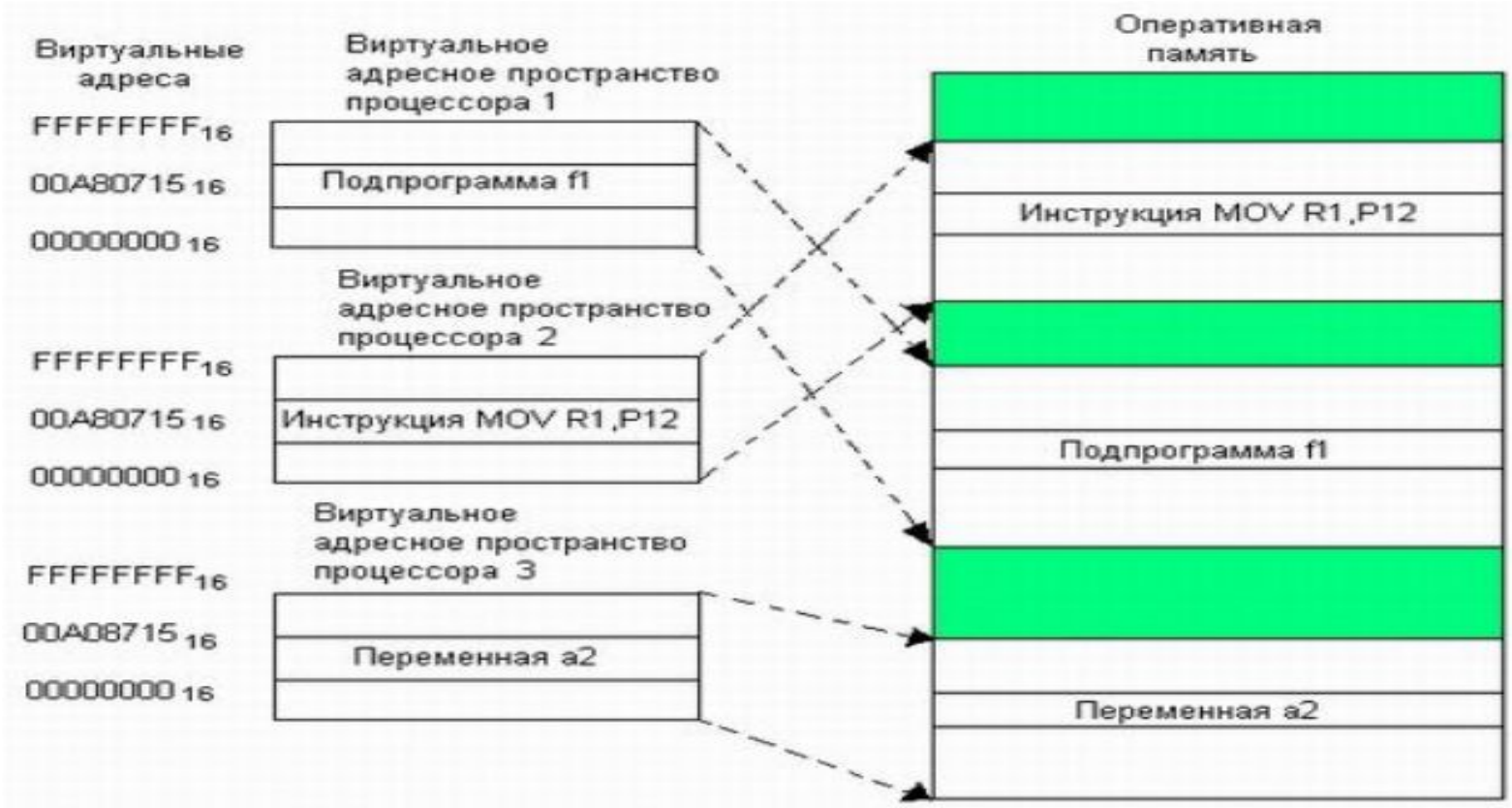


Рис. 3.2. Виртуальные адресные пространства нескольких программ

- В разных операционных системах используются разные способы структуризации виртуального адресного пространства:
  - В одних ОС виртуальное адресное пространство процесса подобно физической памяти представлено в виде непрерывной линейной последовательности виртуальных адресов. Такую структуру адресного пространства называют также плоской (flat). При этом виртуальным адресом является единственное число, представляющее собой смещение относительно начала (обычно это значение 000...000) виртуального адресного пространства (рис. 3.3, а). Адрес такого типа называют линейным виртуальным адресом.
  - В других ОС виртуальное адресное пространство делится на части, называемые сегментами (или секциями, или областями, или другими терминами). В этом случае помимо линейного адреса может быть использован виртуальный адрес, представляющий собой пару чисел (и, т), где n определяет сегмент, а т — смещение внутри сегмента (рис. 3.3, б).

- Средства аппаратной поддержки ОС. Часть функций ОС может выполняться и аппаратными средствами. К операционной системе относят, естественно, не все аппаратные устройства компьютера, а только средства аппаратной поддержки ОС, то есть те, которые прямо участвуют в организации вычислительных процессов:
  - средства поддержки привилегированного режима,
  - систему прерываний,
  - средства переключения контекстов процессов,
  - средства защиты областей памяти и т. п.

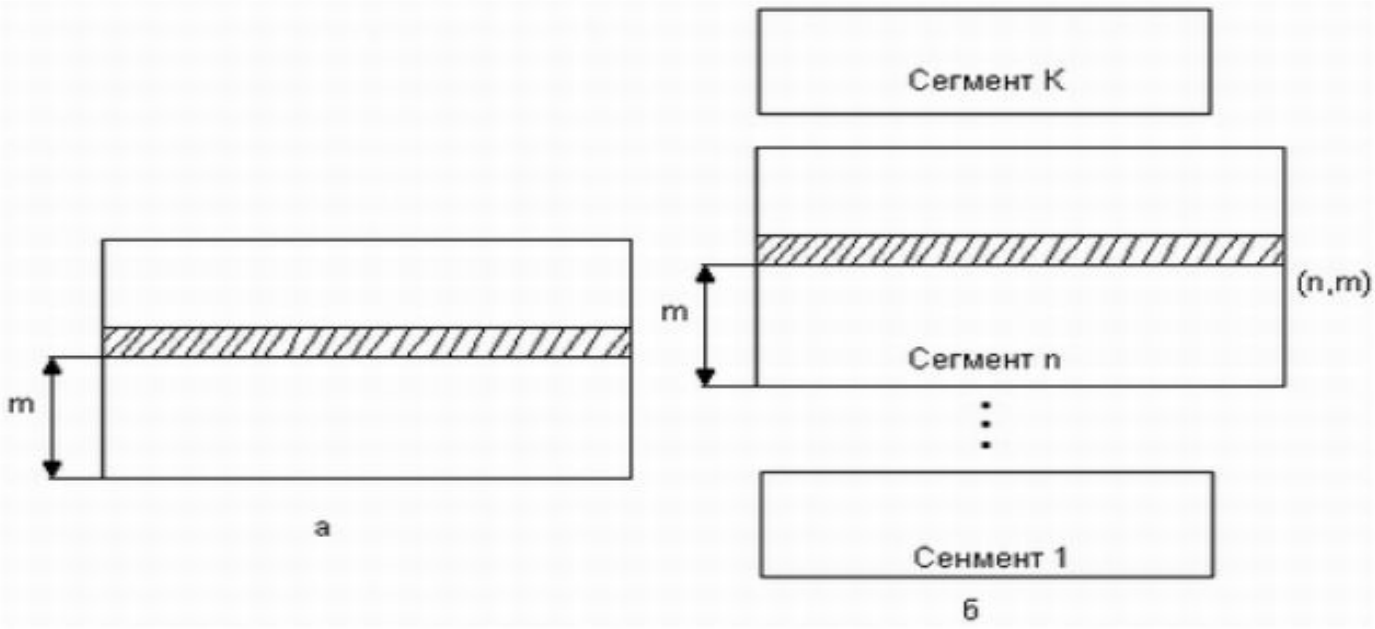


Рис. 3.3. Типы виртуальных адресных пространств: плоское (а), сегментированное (б)

- Существуют и более сложные способы структуризации виртуального адресного пространства, когда виртуальный адрес образуется тремя или даже более числами. Задачей операционной системы является отображение индивидуальных виртуальных адресных пространств всех одновременно выполняющихся процессов на общую физическую память.

- Существуют два принципиально отличающихся подхода 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, в которую предстоит загружать программу, а также информации, предоставленной транслятором об адресно-зависимых элементах программы, выполняет загрузку программы, совмещая ее с заменой виртуальных адресов физическими.
  - программа загружается в память в неизмененном виде в виртуальных адресах, то есть операнды инструкций и адреса переходов имеют те значения, которые выработал транслятор. В наиболее простом случае, когда виртуальная и физическая память процесса представляют собой единые непрерывные области адресов, операционная система выполняет преобразование виртуальных адресов в физические по следующей схеме.

- Во втором случае 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. Схема такого преобразования показана на рис. 3.4.

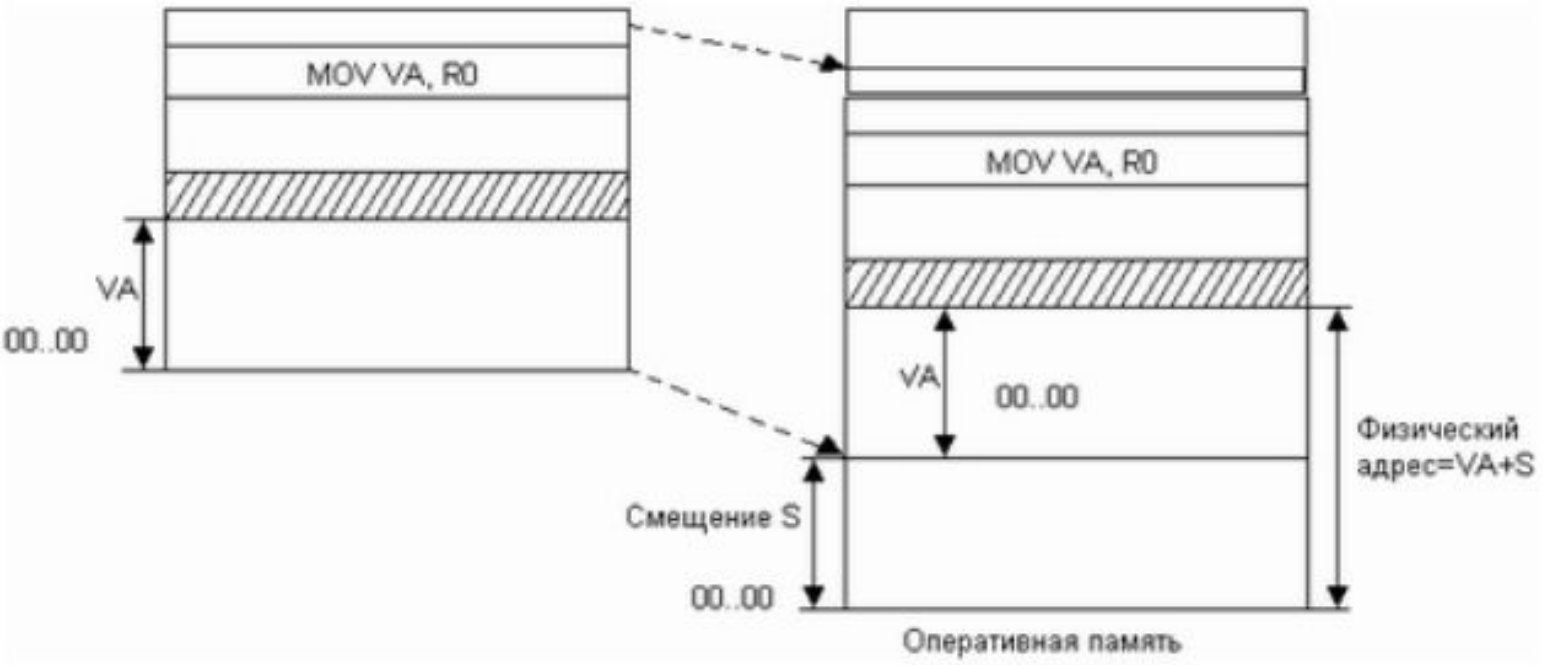


Рис. 3.4. Схема динамического преобразования адресов

- Пусть, например, некоторая программа, работающая под управлением этой ОС, загружена в физическую память начиная с физического адреса S. ОС запоминает значение начального смещения S и во время выполнения программы помещает его в специальный регистр процессора. При обращении к памяти виртуальные адреса данной программы преобразуются в физические путем прибавления к ним смещения S. Например, при выполнении инструкции пересылки данных, находящихся по адресу VA, виртуальный адрес VA заменяется физическим адресом VA+S.
- Такой способ является более гибким: в то время как перемещающий загрузчик жестко привязывает программу к первоначально выделенному ей участку памяти, динамическое преобразование виртуальных адресов позволяет перемещать программный код процесса в течение всего периода его выполнения.

Виртуальная память и свопинг

- Сегодня для машин универсального назначения типична ситуация, когда объем виртуального адресного пространства превышает доступный объем оперативной памяти. В таком случае операционная система для хранения данных виртуального адресного пространства процесса, не помещающихся в оперативную память, использует внешнюю память, которая в современных компьютерах представлена жесткими дисками (рис. 3.4, а). Именно на этом принципе основана виртуальная память — наиболее совершенный механизм, используемый в операционных системах для управления памятью.
- Однако соотношение объемов виртуальной и физической памяти может быть и обратным. Так, в мини-компьютерах 80-х годов разрядности поля адреса нередко не хватало для того, чтобы охватить всю имеющуюся оперативную память. Несколько процессов могло быть загружено в память одновременно и целиком (рис. 3.4, б).

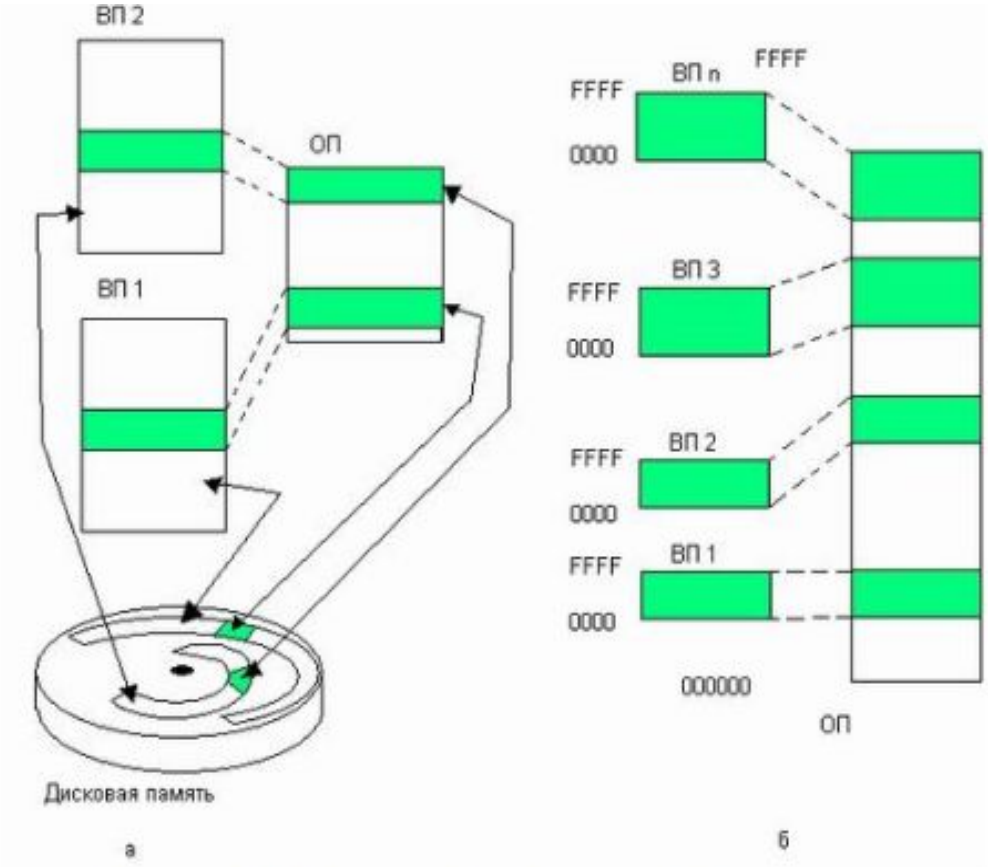


Рис. 3.4. Соотношение объемов виртуального адресного пространства и физической памяти: виртуальное адресное пространство превосходит объем физической памяти (а), виртуальное адресное пространство меньше объема физической памяти (б)

- Необходимо подчеркнуть, что виртуальное адресное пространство и виртуальная память — это различные механизмы и они не обязательно реализуются в операционной системе одновременно. Можно представить себе ОС, в которой поддерживаются виртуальные адресные пространства для процессов, но отсутствует механизм виртуальной памяти. Это возможно только в том случае, если размер виртуального адресного пространства каждого процесса меньше объема физической памяти.
- Подмена (виртуализация) оперативной памяти дисковой памятью позволяет повысить объем оперативной памяти компьютера поскольку суммарный объем памяти, занимаемой процессами, может существенно превосходить имеющийся объем оперативной памяти.

- Виртуальным называется ресурс, который пользователю или пользовательской программе представляется обладающим свойствами, которыми он в действительности не обладает. В данном случае в распоряжение прикладного программиста предоставляется виртуальная оперативная память, размер которой намного превосходит всю имеющуюся в системе реальную оперативную память.
- Виртуализация оперативной памяти осуществляется совокупностью программных модулей ОС и аппаратных схем процессора и включает решение следующих задач:
  - 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; Q преобразование виртуальных адресов в физические.

- Виртуализация памяти может быть осуществлена на основе двух различных подходов:

  - 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.

- Свопинг представляет собой частный случай виртуальной памяти и, следовательно, более простой в реализации способ совместного использования оперативной памяти и диска. Однако подкачке свойственна избыточность: когда ОС решает активизировать процесс, для его выполнения, как правило, не требуется загружать в оперативную память все его сегменты полностью — достаточно загрузить небольшую часть кодового сегмента с подлежащей выполнению инструкцией и частью сегментов Данных, с которыми работает эта инструкция, а также отвести место под сегмент стека.

  Для временного хранения сегментов и страниц на диске отводится либо специальная область, либо специальный файл, которые во многих ОС по традиции продолжают называть областью, или файлом свопинга, хотя перемещение информации между оперативной памятью и диском осуществляется уже не в форме полного замещения одного процесса другим, а частями. Другое популярное название этой области — страничный файл (page file, или paging file). Текущий размер страничного файла является важным параметром, оказывающим влияние на возможности операционной системы: чем больше страничный файл, тем больше приложений может одновременно выполнять ОС (при фиксированном размере оперативной памяти).

Алгоритмы управления памятью

- Рассмотрим наиболее общие подходы к распределению памяти, которые были характерны для разных периодов развития операционных систем. Некоторые из них сохранили актуальность и широко используются в современных ОС, другие же представляют в основном только познавательный интерес, хотя их и сегодня можно встретить в специализированных системах. На рис. 3.7 все алгоритмы распределения памяти разделены на два класса:
  - алгоритмы, в которых используется перемещение сегментов процессов между оперативной памятью и диском,
  - алгоритмы, в которых внешняя память не привлекается.

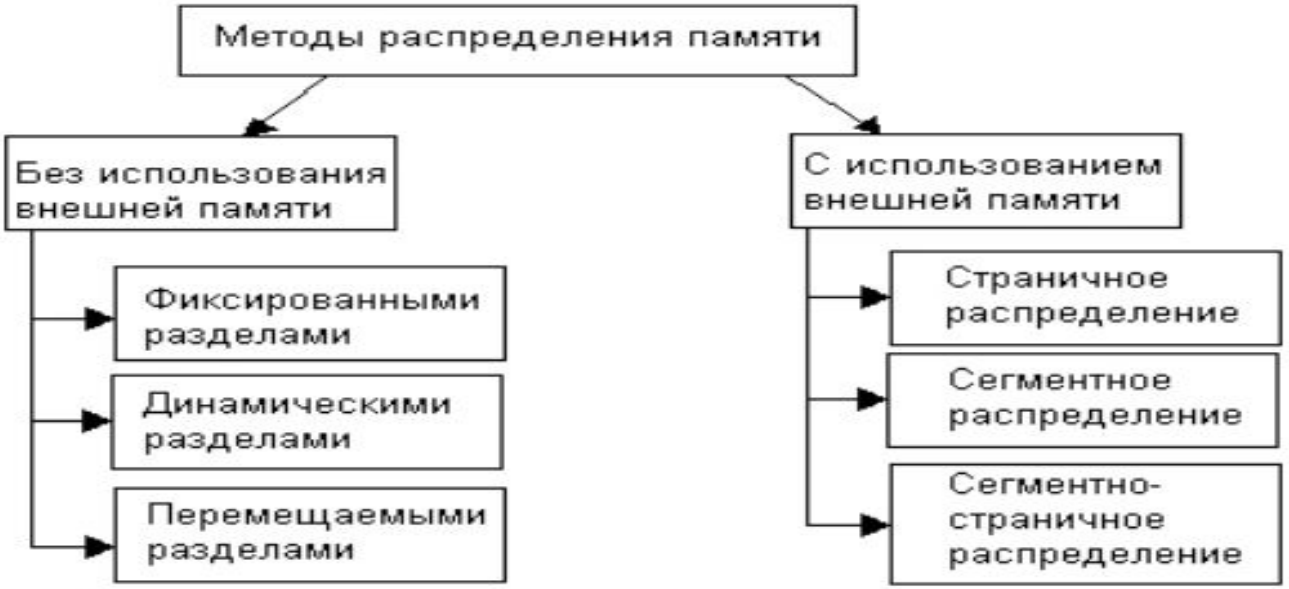


Рис. 3.7. Классификация методов распределения памяти

Распределение памяти фиксированными разделами

- Простейший способ управления оперативной памятью состоит в том, что 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. 3.8, а), либо в очередь к некоторому разделу (рис. 3.8, б).

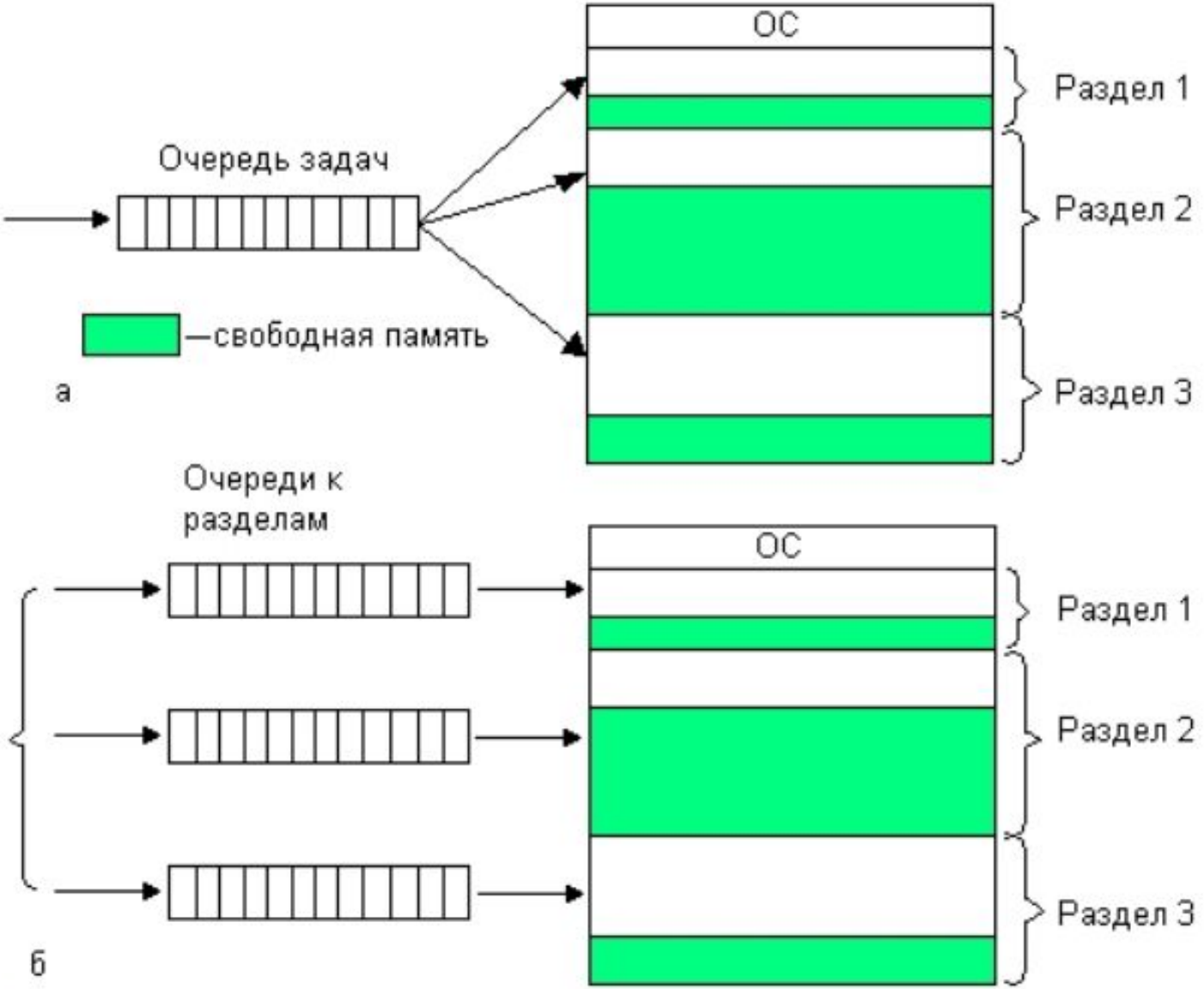


Рис. 3.8. Распределение памяти фиксированными разделами: с общей очередью (а), с отдельными очередями (б)

- При очевидном преимуществе — простоте реализации, данный метод имеет существенный недостаток — жесткость. Так как в каждом разделе может выполняться только один процесс, то уровень мультипрограммирования заранее ограничен числом разделов. Независимо от размера программы она будет занимать весь раздел. С другой стороны, разбиение памяти на разделы не позволяет выполнять процессы, программы которых не помещаются ни в один из разделов, но для которых было бы достаточно памяти нескольких разделов. Такой способ управления памятью применялся в ранних мультипрограммных ОС.

Распределение памяти динамическими разделами

- В этом случае 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. На рис. 3.9 показано состояние памяти в различные моменты времени при использовании динамического распределения. Так, в момент t0 в памяти находится только ОС, а к моменту t1 память разделена между 5 процессами, причем процесс П4, завершаясь, покидает память. На освободившееся от процесса П4 место загружается процесс П6, поступивший в момент t3.

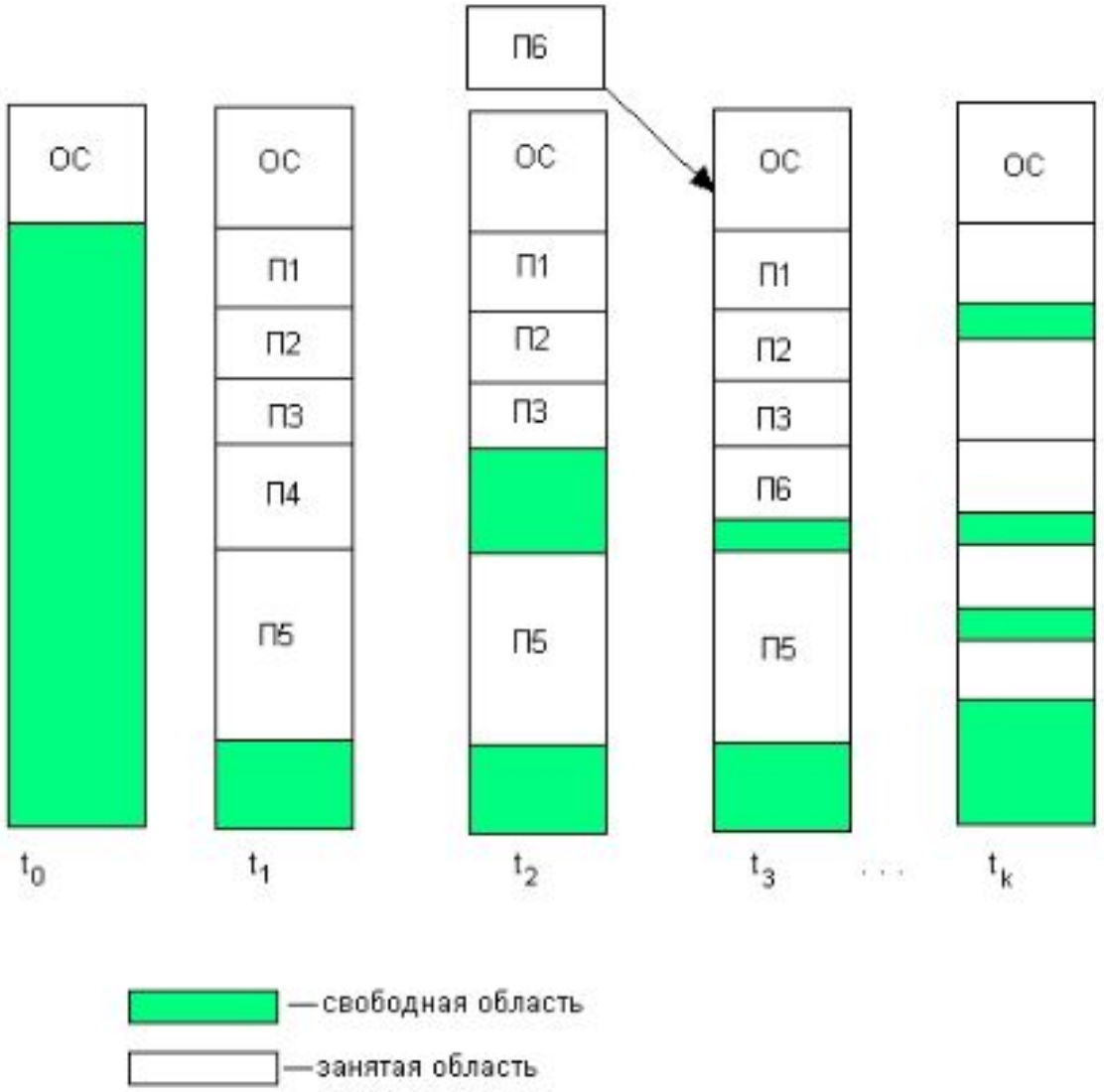


Рис. 3.9. Распределение памяти динамическими разделами

- По сравнению с методом распределения памяти фиксированными разделами данный метод обладает гораздо большей гибкостью, но ему присущ очень серьезный недостаток — фрагментация памяти.
- Фрагментация — это наличие большого числа несмежных участков свободной памяти очень маленького размера (фрагментов). Настолько маленького, 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 многих мультипрограммных операционных системах 60-70-х годов, в частности такой операционной системы, как OS/360.

Перемещаемые разделы

- Одним из методов борьбы с фрагментацией является перемещение всех занятых участков в сторону старших или младших адресов, так, чтобы вея свободная память образовала единую свободную область (рис 3.10). ОС должна еще время от времени копировать содержимое разделов из одного места памяти в другое, корректируя таблицы свободных и занятых областей. Эта процедура называется сжатием.

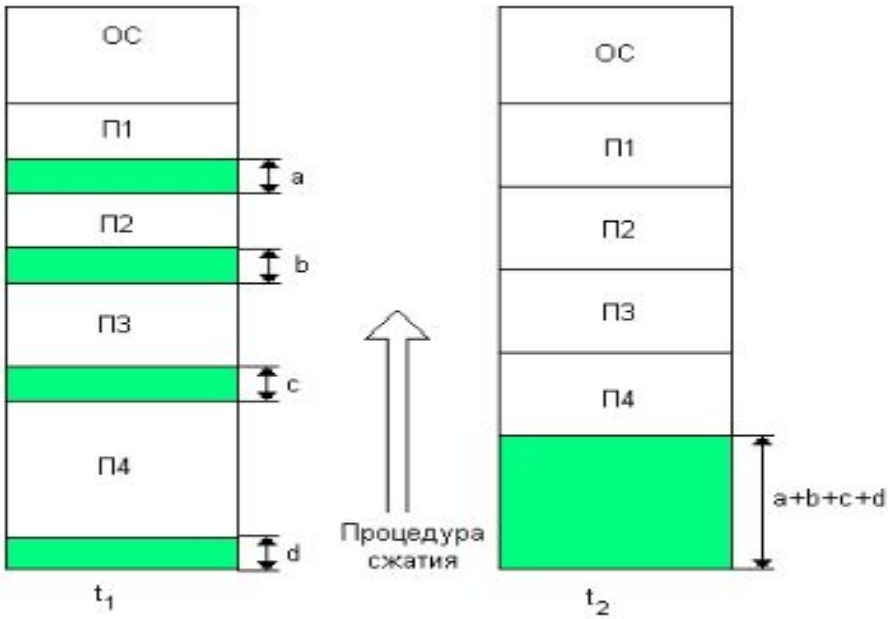


Рис. 3.10. Распределение памяти перемещаемыми разделами

- Хотя процедура сжатия и приводит к более эффективному использованию памяти, она может потребовать значительного времени, что часто перевешивает преимущества данного метода.
- Так как программы перемещаются по оперативной памяти в ходе своего выполнения, то в данном случае невозможно выполнить настройку адресов с помощью перемещающего загрузчика. Здесь более подходящим оказывается динамическое преобразование адресов.

Алгоритмы управления памятью с использованием виртуальной памяти

- Ключевой проблемой виртуальной памяти, возникающей в результате многократного изменения местоположения в оперативной памяти образов процессов или их частей, является преобразование виртуальных адресов в физические. Решение этой проблемы, в свою очередь, зависит от того, какой способ структуризации виртуального адресного пространства принят в данной системе управления памятью.
- 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.
  - 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.

Страничное распределение

- На рис. 3.11 показана схема страничного распределения памяти. Виртуальное адресное пространство каждого процесса делится на части
одинакового, фиксированного для данной системы размера, называемые
виртуальными страницами (virtual pages). В общем случае размер виртуального адресного пространства процесса не кратен размеру стр[illegible]ется фи

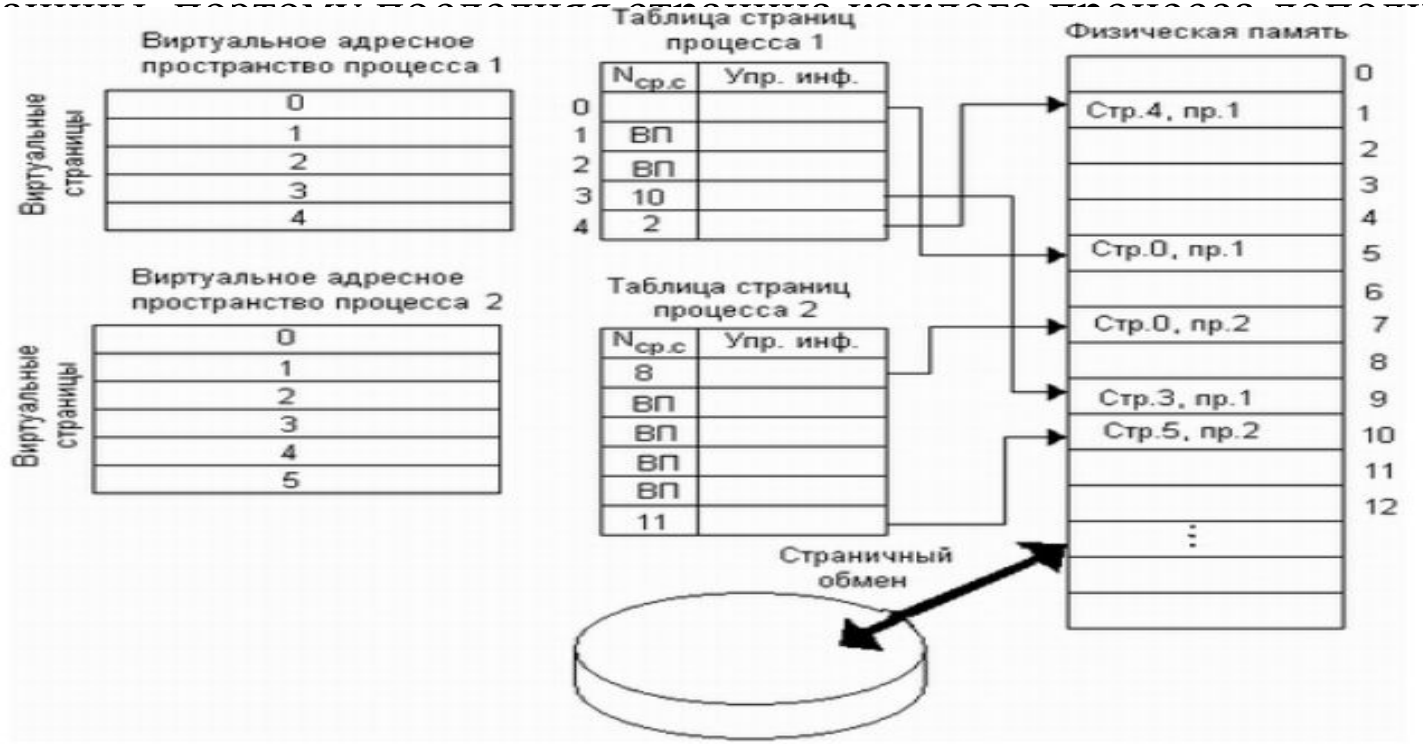


Рис. 3.11. Страничное распределение памяти

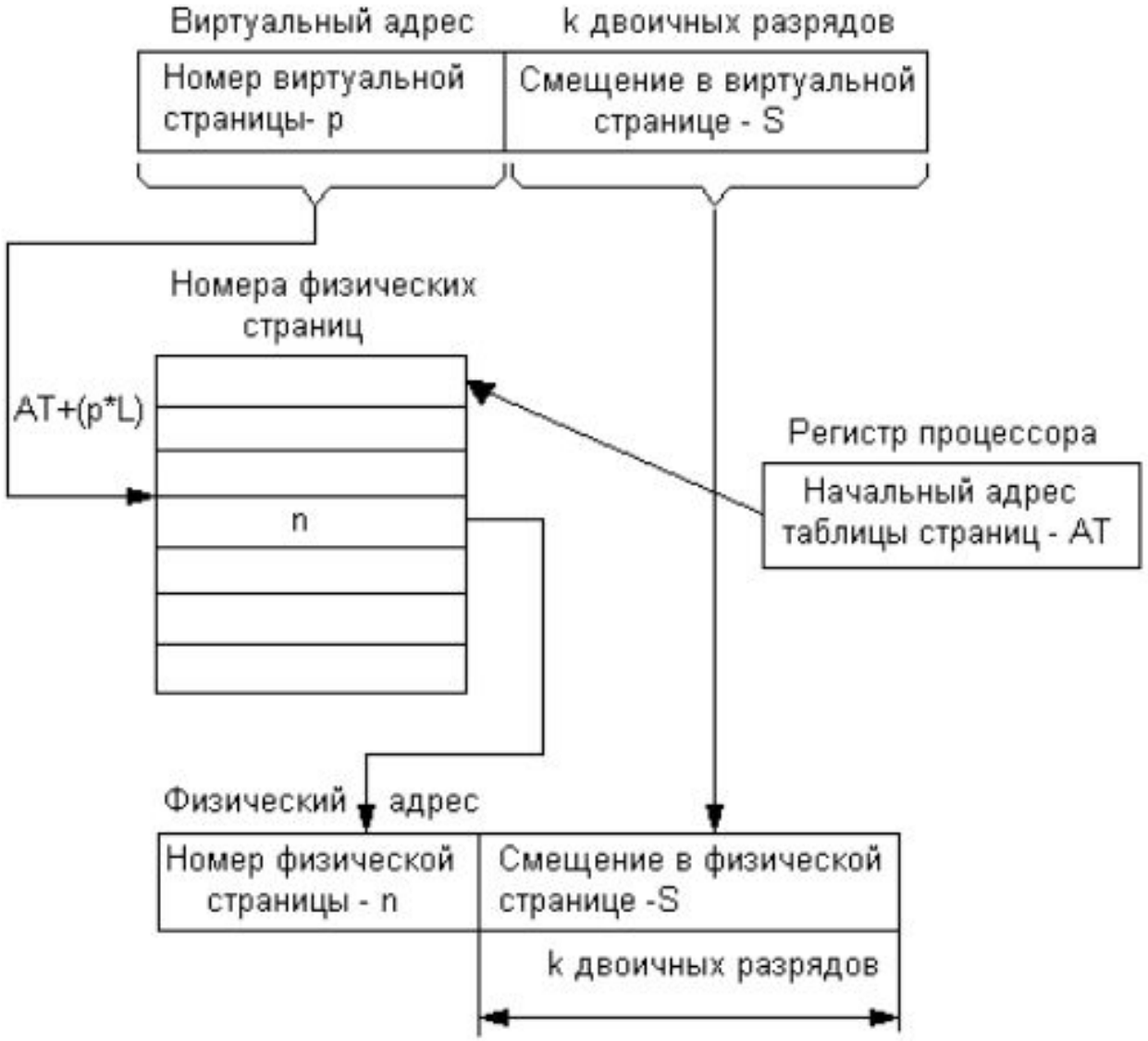


Рис. 3.12. Схема преобразования виртуального адреса в физический

- 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. Это позволяет упростить механизм преобразования адресов. Организация перемещение данных между памятью и диском ведется страницами — частями виртуального адресного пространства, фиксированного и сравнительно небольшого размера.
  При создании процесса ОС загружает в оперативную память несколько его виртуальных страниц (начальные страницы кодового сегмента и сегмента данных). Копия всего виртуального адресного пространства процесса находится на диске. Смежные виртуальные страницы не обязательно располагаются в смежных физических страницах. Для каждого процесса операционная система создает таблицу страниц — информационную структуру, содержащую записи обо всех виртуальных страницах процесса.

- При каждом обращении к памяти выполняется поиск номера виртуальной страницы, содержащей требуемый адрес, затем по этому номеру определяется нужный элемент таблицы страниц, и из него извлекается описывающая страницу информация1. Далее анализируется признак присутствия, и, если данная виртуальная страница находится в оперативной памяти, то выполняется преобразование виртуального адреса в физический, то есть виртуальный адрес заменяется указанным в записи таблицы физическим адресом. Если же нужная виртуальная страница в данный момент выгружена на диск, то происходит так называемое страничное прерывание. Выполняющийся процесс переводится в состояние ожидания, и активизируется другой процесс из очереди процессов, находящихся в состоянии готовности. Параллельно программа обработки страничного прерывания находит на диске требуемую виртуальную страницу (для этого операционная система должна помнить положение вытесненной страницы в страничном файле диска) и пытается загрузить ее в оперативную память. Если в памяти имеется свободная физическая страница, то загрузка выполняется немедленно, если же свободных страниц нет, то на основании принятой в данной системе стратегии замещения страниц решается вопрос о том, какую страницу следует выгрузить из оперативной памяти.

- После того как выбрана страница, которая должна покинуть оперативную память, обнуляется ее бит присутствия и анализируется ее признак модификации. Если выталкиваемая страница за время последнего пребывания в оперативной памяти была модифицирована, то ее новая версия должна быть переписана на диск. Если нет, то принимается во внимание, что на диске уже имеется предыдущая копия этой виртуальной страницы, и никакой записи на диск не производится. Физическая страница объявляется
свободной. Из соображений безопасности в некоторых системах освобождаемая страница обнуляется, с тем чтобы невозможно было использовать содержимое выгруженной страницы.

Сегментное распределение

- При страничной организации виртуальное адресное пространство процесса делится на равные части механически, без учета смыслового значения данных. Такой подход не позволяет обеспечить дифференцированный доступ к разным частям программы, а это свойство могло бы быть очень полезным во многих случаях. Кроме того, разбиение виртуального адресного пространства на «осмысленные» части делает возможным совместное использование фрагментов программ разными процессами. При отображении в физическую память сегменты, содержащие коды подпрограммы из обоих виртуальных пространств, проецируются на одну и ту же область физической памяти. Таким образом оба процесса получат доступ к одной и той же копии подпрограммы (рис. 3.13).

- Итак, 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, в соответствии с принятыми в системе соглашениями.
- Максимальный размер сегмента определяется разрядностью виртуального адреса. Сегменты не упорядочиваются друг относительно друга, так что общего для сегментов линейного виртуального адреса не существует, виртуальный адрес задается парой чисел: номером сегмента и линейным виртуальным адресом внутри сегмента (рис. 3.14).

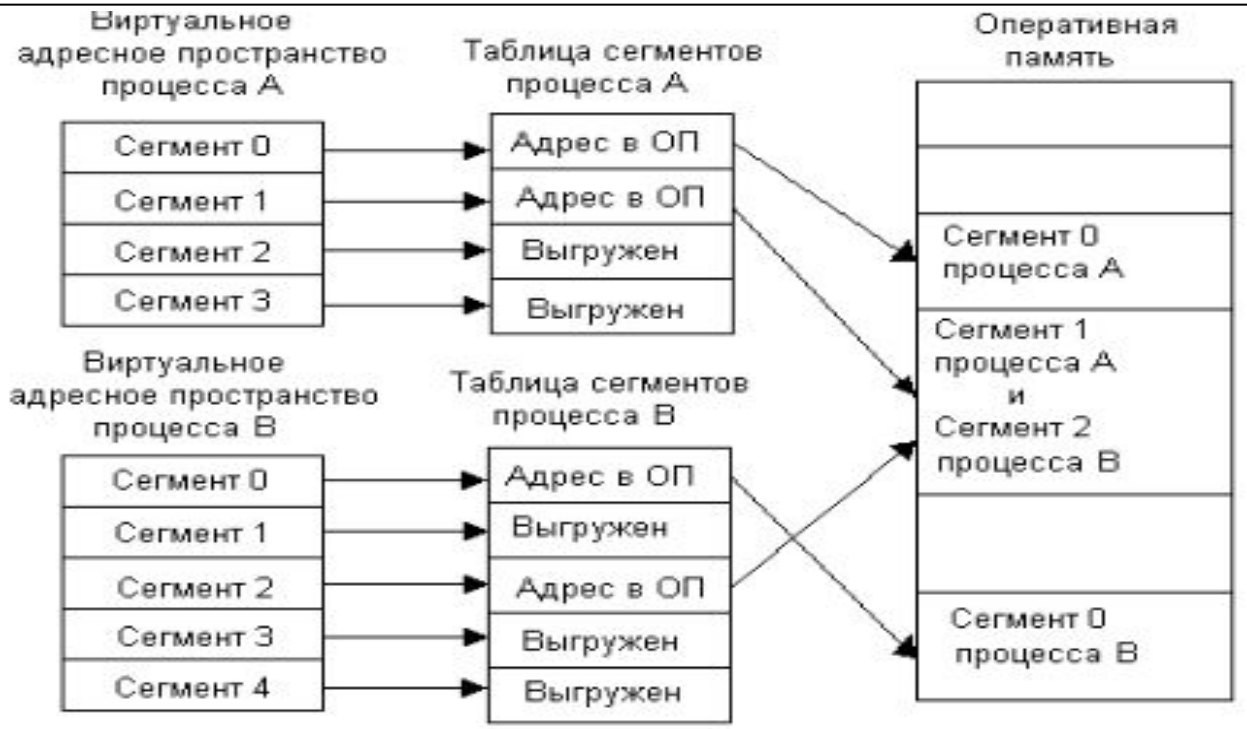


Рис. 3.13. Распределение памяти сегментами

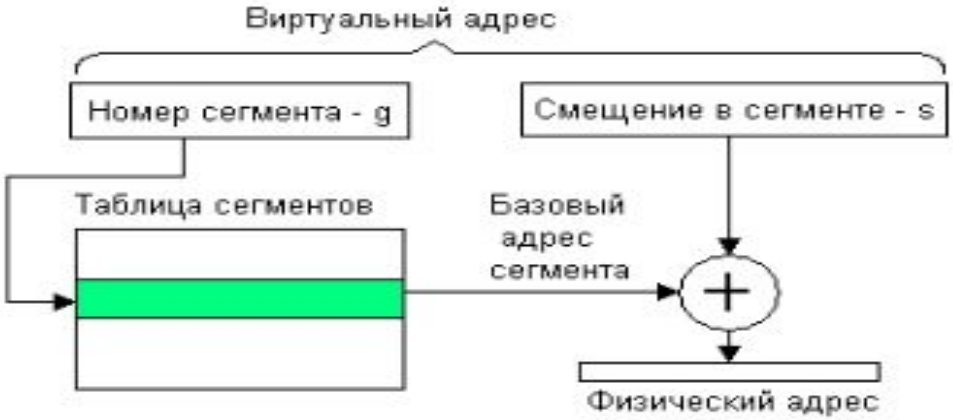


Рис. 3.14. Преобразование виртуального адреса при сегментной организации памяти

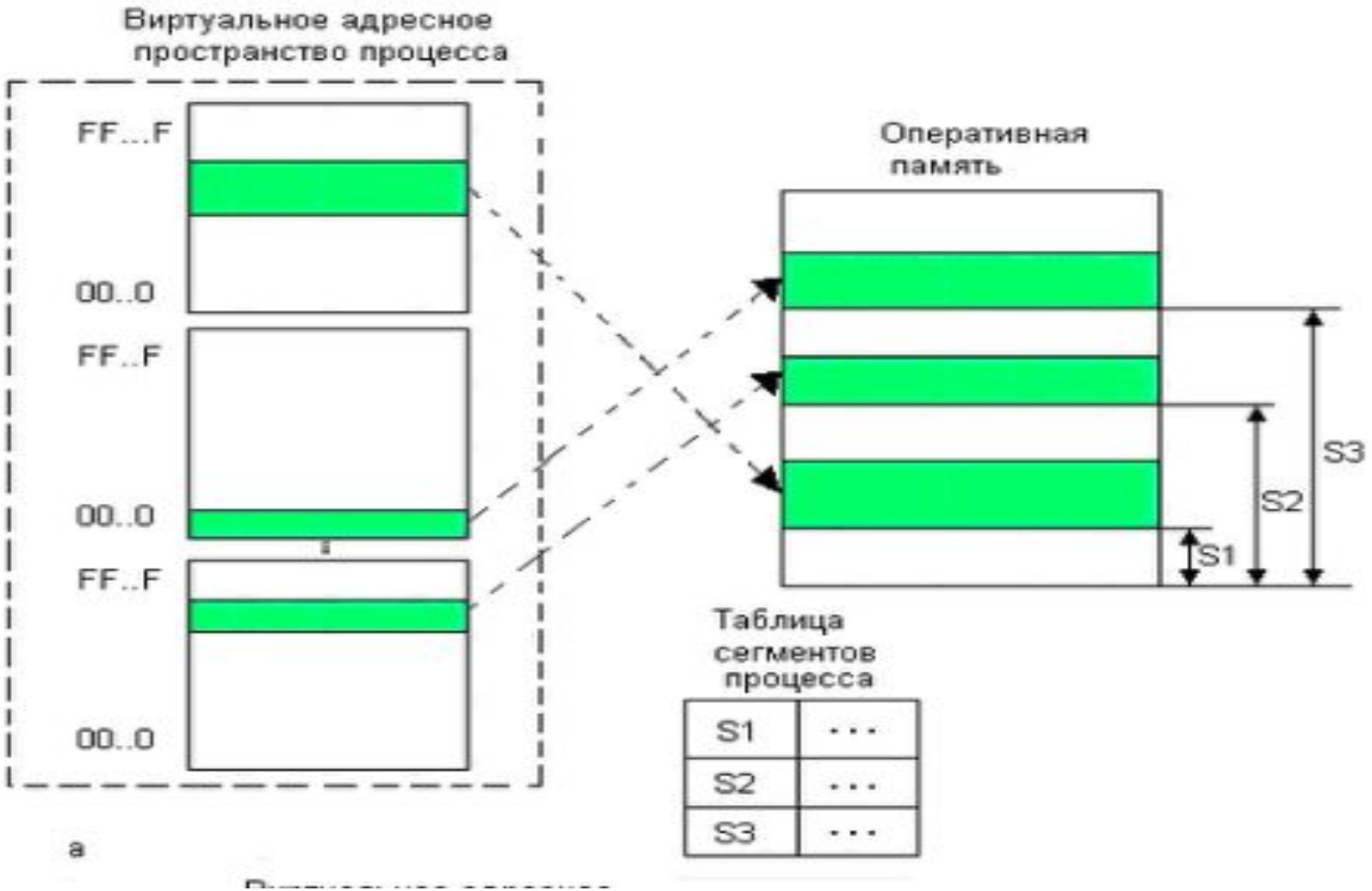
- При загрузке процесса в оперативную память помещается только часть его сегментов, полная копия виртуального адресного пространства находится в дисковой памяти. Для каждого загружаемого сегмента операционная система подыскивает непрерывный участок свободной памяти достаточного размера. Смежные в виртуальной памяти сегменты одного процесса могут занимать в оперативной памяти несмежные участки. Если во время выполнения процесса происходит обращение по виртуальному адресу, относящемуся к сегменту, который в данный момент отсутствует в памяти, то происходит прерывание. ОС приостанавливает активный процесс, запускает на выполнение следующий процесс из очереди, а параллельно организует загрузку нужного сегмента с диска. При отсутствии в памяти места, необходимого для загрузки сегмента, операционная система выбирает сегмент на выгрузку, при этом она использует критерии, аналогичные рассмотренным выше критериям выбора страниц при страничном способе управления памятью.

- Если виртуальные адресные пространства нескольких процессов включают один и тот же сегмент, то в таблицах сегментов этих процессов делаются ссылки на один и тот же участок оперативной памяти, в который данный сегмент загружается в единственном экземпляре. Как видно, сегментное распределение памяти имеет очень много общего со страничным распределением. Механизмы преобразования адресов этих двух способов управления памятью тоже весьма схожи, однако в них имеются и существенные отличия, которые являются следствием того, что сегменты в отличие от страниц имеют произвольный размер. Виртуальный адрес при сегментной организации памяти может быть представлен парой (g, s), где g — номер сегмента, а s — смещение в сегменте. Физический адрес получается путем сложения базового адреса сегмента, который определяется по номеру сегмента g из таблицы сегментов и смещения s (рис. 3.14). Главный недостаток сегментного распределения — это фрагментация, которая возникает из-за непредсказуемости размеров сегментов. В процессе работы системы в памяти образуются небольшие участки свободной памяти, в которые не может быть загружен ни один сегмент. Суммарный объем, занимаемый фрагментами, может составить существенную часть общей памяти системы, приводя к ее неэффективному использованию.

Сегментно-страничное распределение

- Метод сегментно-страничного распределения памяти представляет собой комбинацию страничного и сегментного механизмов управления памятью и направлен на реализацию достоинств обоих подходов.
  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.

- В большинстве современных реализаций сегментно-страничной организации памяти в отличие от набора виртуальных диапазонов адресов при сегментной организации памяти (рис. 3.15, а) все виртуальные сегменты образуют одно непрерывное линейное виртуальное адресное пространство (рис. 3.16, б).



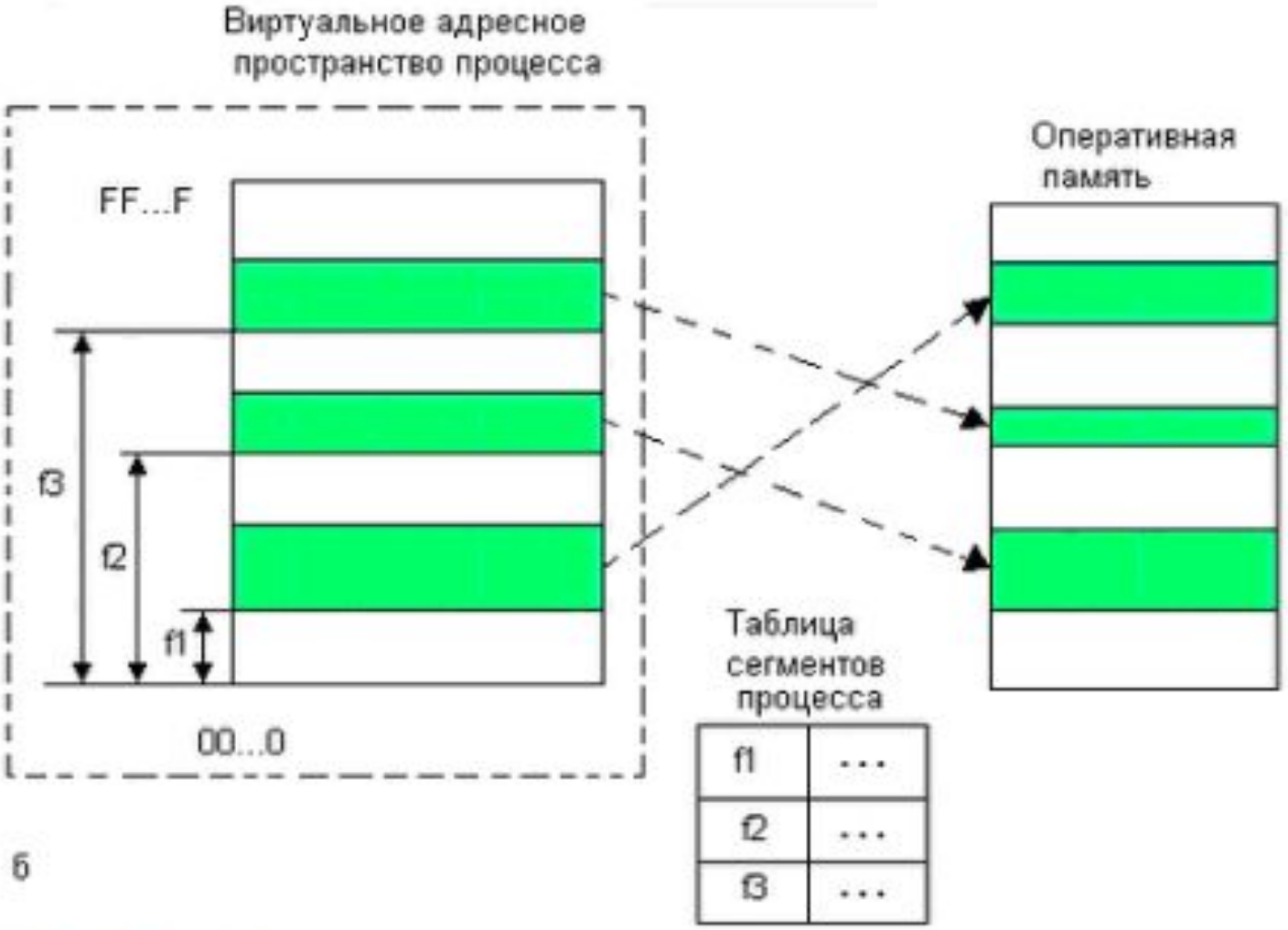


Рис. 3.16. Два способа сегментации при сегментно-страничной организации памяти

- Координаты байта в виртуальном адресном пространстве при сегментно-страничной организации можно задать двумя способами:
  - Во-первых, линейным виртуальным адресом, который равен сдвигу данного байта относительно границы общего линейного виртуального пространства,
  - Во-вторых, парой чисел, одно из которых является номером сегмента, а другое — смещением относительно начала сегмента. При этом в отличие от сегментной модели, для однозначного задания виртуального адреса вторым способом необходимо каким-то образом указать также начальный виртуальный адрес сегмента с данным номером.

- Системы виртуальной памяти ОС с сегментно-страничной организацией используют второй способ, так как он позволяет непосредственно определить принадлежность адреса некоторому сегменту и проверить права доступа процесса к нему.
- Для каждого процесса операционная система создает отдельную таблицу сегментов, в которой содержатся описатели (дескрипторы) всех сегментов процесса. Описание сегмента включает назначенные ему права доступа и другие характеристики, подобные тем, которые содержатся в дескрипторах сегментов при сегментной организации памяти. Однако имеется и принципиальное отличие. В поле базового адреса указывается не начальный физический адрес сегмента, отведенный ему в результате загрузки в оперативную память, а начальный линейный виртуальный адрес сегмента в пространстве виртуальных адресов (на рис. 3.16 базовые физические адреса обозначены SI, S2, S3, а базовые виртуальные адреса — fl, f2, f3).
- Наличие базового виртуального адреса сегмента в дескрипторе позволяет однозначно преобразовать адрес, заданный в виде пары (номер сегмента, смещение в сегменте), в линейный виртуальный адрес байта, который затем преобразуется в физический адрес страничным механизмом.

- Преобразование виртуального адреса в физический происходит в два этапа (рис. 3.17):
- 1. На первом этапе работает механизм сегментации. Исходный виртуальный адрес, заданный в виде пары (номер сегмента, смещение), преобразуется в линейный виртуальный адрес. Для этого на основании базового адреса таблицы сегментов и номера сегмента вычисляется адрес дескриптора сегмента. Анализируются поля дескриптора и выполняется проверка возможности выполнения заданной операции. Если доступ к сегменту разрешен, то вычисляется линейный виртуальный адрес путем сложения базового адреса сегмента, извлеченного из дескриптора, и смещения, заданного в исходном виртуальном адресе.
  2. На втором этапе работает страничный механизм. Полученный линейный виртуальный адрес преобразуется в искомый физический адрес. В результате преобразования линейный виртуальный адрес представляется в том виде, в котором он используется при страничной организации памяти, а именно в виде пары (номер страницы, смещение в странице). Благодаря тому что размер страницы выбран равным степени двойки, эта задача решается простым отделением некоторого количества младших двоичных разрядов. При этом в старших разрядах содержится номер виртуальной страницы, а в младших — смещение искомого элемента относительно начала страницы.

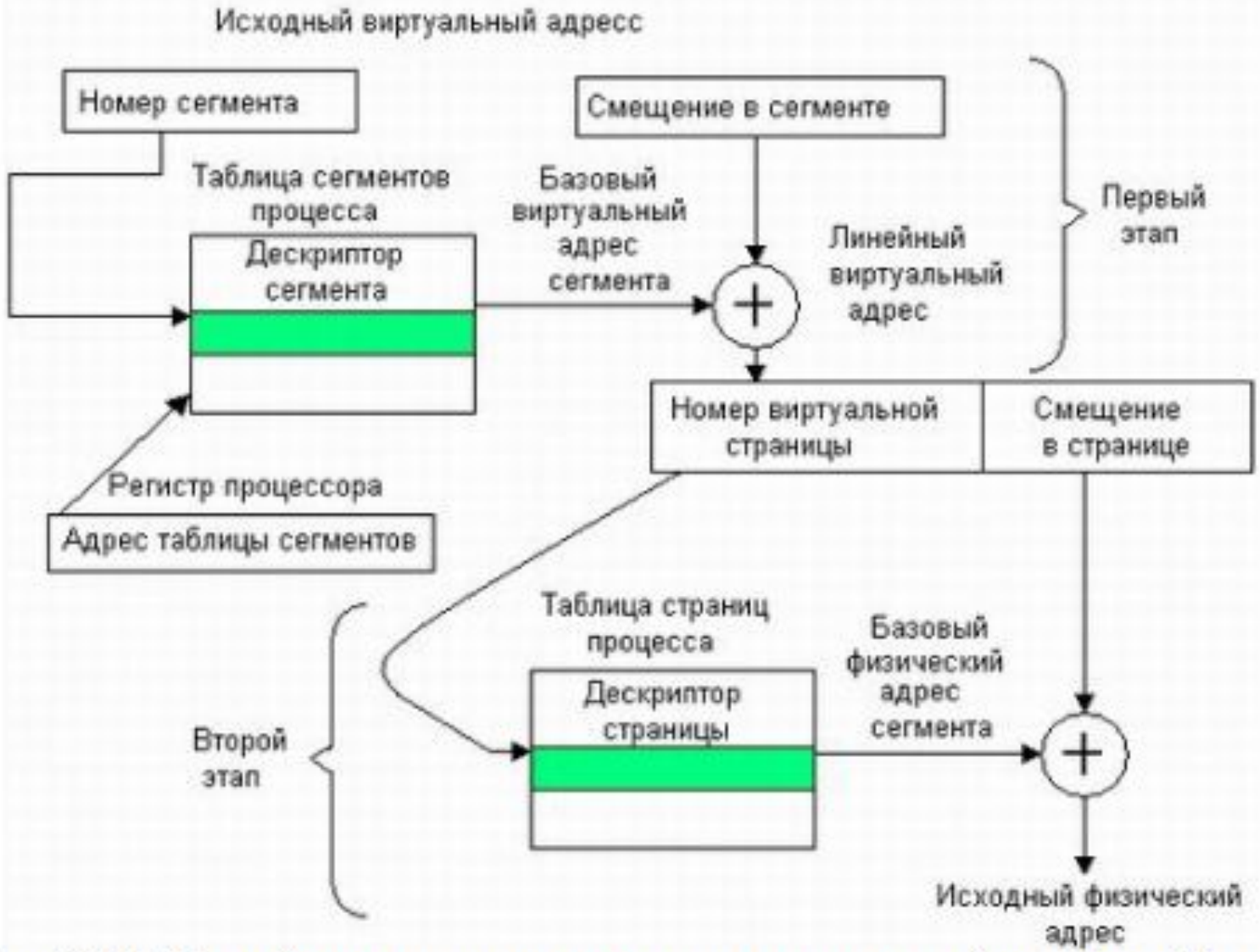


Рис. 3.17. Преобразование виртуального адреса в физический при сегментно -страничной организации памяти