Системное программное обеспечение

Лекция № 3 «Организация памяти»

Организация памяти

Физическая память, к которой процессор имеет доступ по шине адреса называется ***оперативной памятью*** *(или* ***оперативным запоминающим устройством — ОЗУ****). ОЗУ организовано как последовательность ячеек — байтов. Каждому байту соответствует свой уникальный адрес (его номер), называемый физическим.*

Диапазон значений физических адресов зависит от разрядности шины адреса процессора. Для 80486 и Pentium он находится в пределах от 0 до $2^{32} - 1$ *(4 Гбайт). Для процессоров Pentium Pro/II/III/IV этот диапазон шире — от 0 до* $2^{36} - 1$ *(64 Гбайт). Процессор 8086 имел 1 Мбайт памяти при двадцатиразрядной шине адреса – от 0 до* $2^{20} - 1$.

Организация памяти

Процессор аппаратно поддерживает две модели использования оперативной памяти:

- *В **сегментированной модели** программе выделяются непрерывные области памяти (сегменты), а сама программа может обращаться только к данным, которые находятся в этих сегментах*
- ***Страничную модель** можно рассматривать как надстройку над сегментированной моделью. Основное применение этой модели связано с организацией виртуальной памяти, что позволяет операционной системе использовать для работы программ пространство памяти большее, чем объем физической памяти за счет объединения в единое адресное пространство оперативной и внешней памяти*

Организация памяти

*Кстати, другое название физического адреса — **линейный адрес**. Подобная двойственность в названии как раз обусловлена наличием страничной модели организации оперативной памяти. Эти названия являются синонимами только при отключении страничного преобразования адреса (в реальном режиме страничная адресация всегда отключена). В страничной модели линейный и физический адреса имеют разные значения.*

Механизм управления памятью является полностью аппаратным и позволяет обеспечить:

- *компактность хранения адреса в машинной команде*
- *гибкость механизма адресации*
- *защиту адресных пространств задач в многозадачной системе*
- *поддержку виртуальной памяти*

Организация памяти

В семействе процессоров 80x86 выбор метода обращения к памяти определяется режимом работы процессора.

*В **реальном режиме** процессор может обращаться только к первому мегабайту памяти, адреса которого находятся в диапазоне от 00000 до FFFFF в шестнадцатеричном выражении. При этом процессор работает в однопрограммном режиме (т.е. в заданный момент времени он может выполнять только одну программу). Однако при этом он может в любой момент прервать ее выполнение и переключиться на процедуру обработки прерывания, поступившего от одного из периферийных устройств. Любой программе, которую выполняет в этот момент процессор, разрешен доступ без ограничения к любым областям памяти, находящимся в пределах первого мегабайта: к ОЗУ — по чтению и записи, а к ПЗУ, понятно, только по чтению. Реальный режим работы процессора используется*

Организация памяти

*В **защищенном режиме** процессор может одновременно выполнять несколько программ. При этом каждому процессу (т.е. выполняющейся программе) может быть назначено до 4 Гбайт оперативной памяти. Чтобы предотвратить взаимное влияние выполняющихся программ друг на друга им выделяются изолированные участки памяти. В защищенном режиме работают такие ОС, как MS Windows и Linux.*

*В **виртуальном режиме адресации процессора 8086**, последний на самом деле работает в защищенном режиме. Для каждой задачи создается собственная виртуальная машина, которой выделяется изолированная область памяти размером 1 Мбайт, и полностью эмулируется работа процессора 80x86 в реальном режиме адресации. Например, в операционных системах Windows 2000 и XP виртуальная машина процессора 8086 создается каждый раз при запуске*

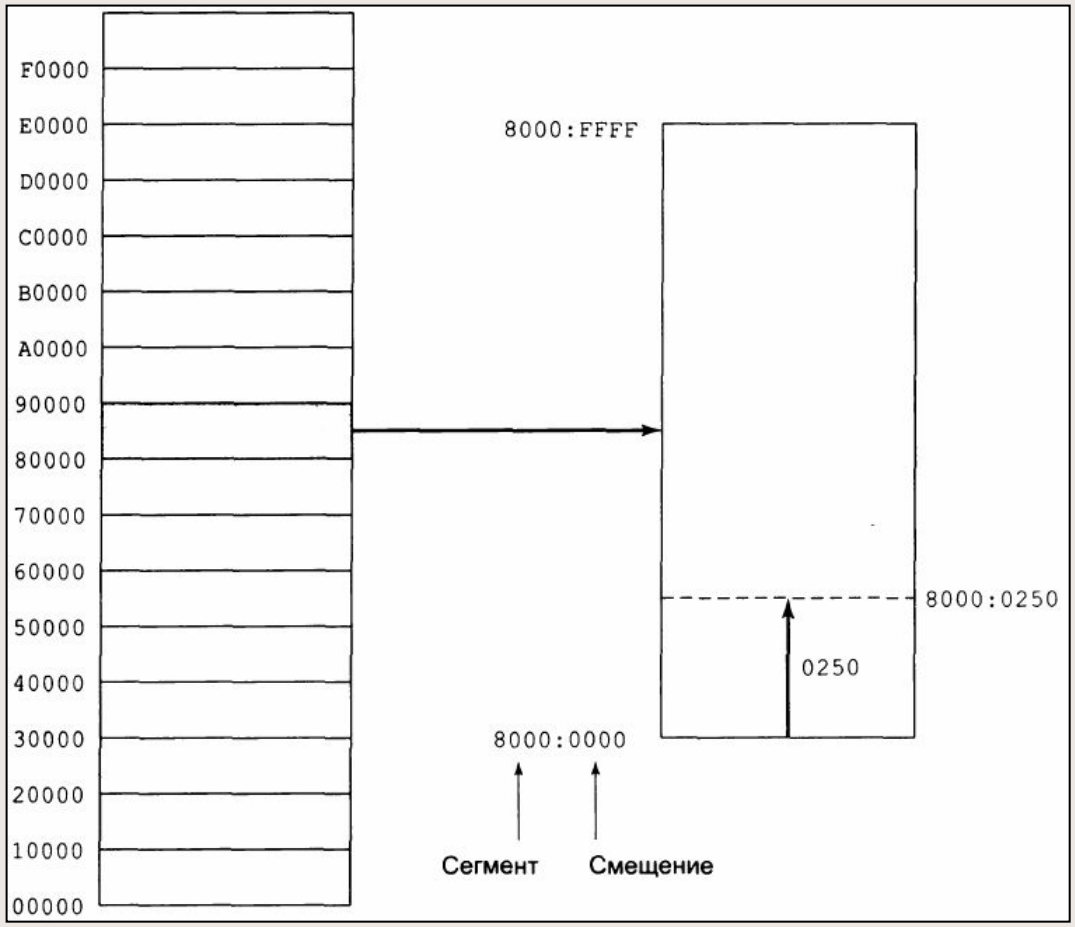
Организация памяти

Реальный режим адресации

Отличительные черты механизма адресации физической памяти в реальном режиме следующие:

- *Диапазон изменения физического адреса — от 0 до 1 Мбайт, поскольку при адресации используется только 20 младших разрядов шины адреса*
- *Максимальный размер памяти, адресуемой с помощью 16–разрядных регистров — 64 Кбайт*
- *Для обращения к конкретному физическому адресу во всей доступной оперативной памяти используется сегментация памяти, т.е. разбиение доступного адресного пространства на сегменты размером 64 Кбайт и использование вместо физического логического адреса в форме <сегмент>:<смещение>, т.е. комбинации адреса начала сегмента и смещение внутри сегмента*
- *16–разрядный адрес начала сегмента помещается в один из шести сегментных регистров (CS, DS, ES, SS, FS или GS)*

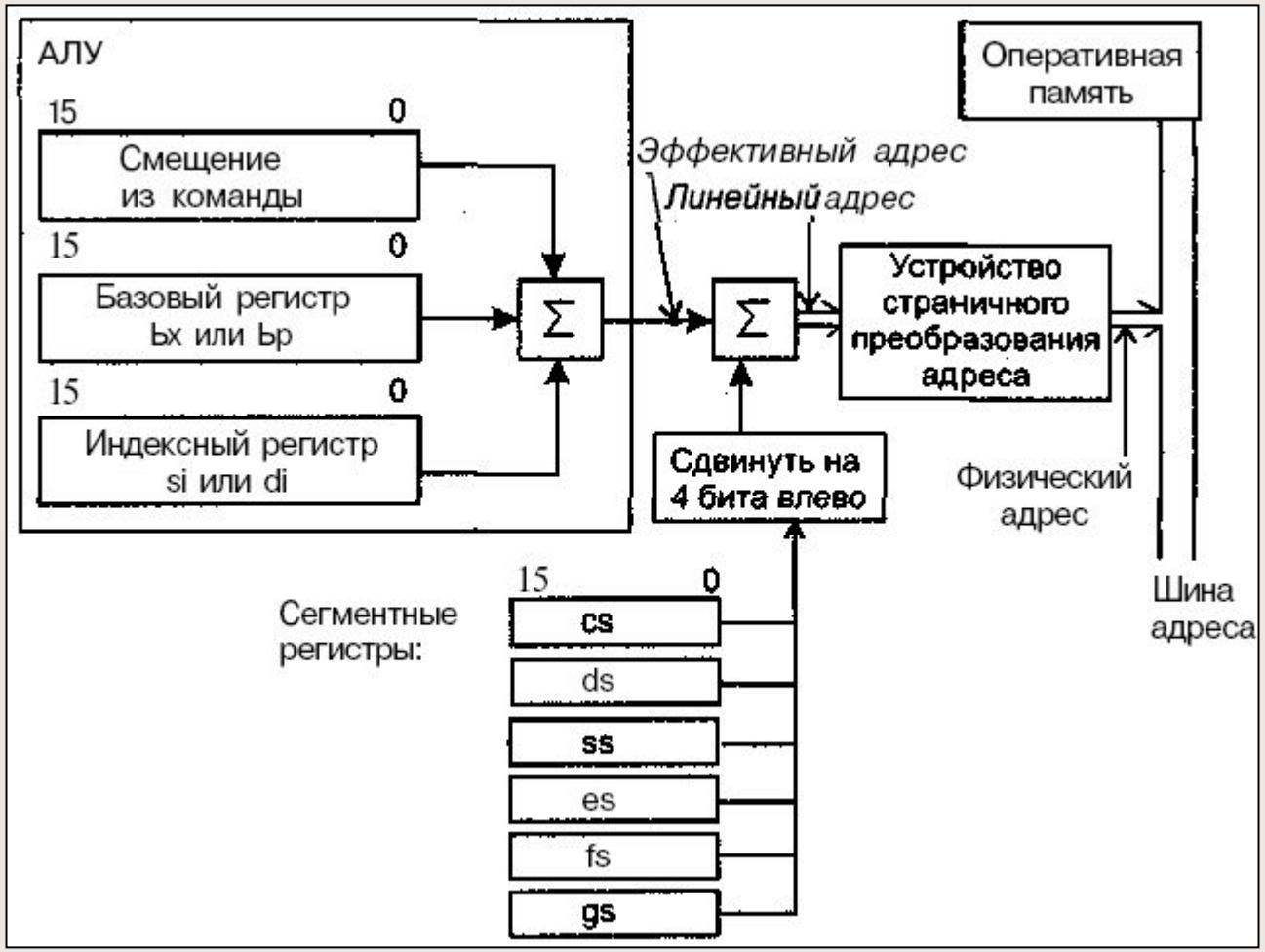
Организация памяти



Младшая шестнадцатеричная цифра в адресе каждого сегмента равна нулю, т.е. адрес любого сегмента всегда будет кратен 16 байтам и границы сегментов располагаются через каждые 16 байт физических адресов. Каждый из этих 16–байтовых фрагментов называется параграфом.

Организация памяти

Адреса, заданные в программах в форме "сегмент–смещение", автоматически преобразуются процессором в 20–разрядные линейные адреса в процессе выполнения команды по следующей схеме:



Организация памяти

Пример: байт, заданный в форме "сегмент–смещение": 8000:0250 в шестнадцатиричной транскрипции.

Логический адрес: 8000:0250

Сегмент: 80000

\+

Смещение: 0250

Физический адрес: 80250

В типичной программе, написанной для процессоров семейства 80x86, как правило, есть три сегмента: кода, данных и стека. При запуске программы их базовые сегментные адреса загружаются в регистры CS, DS и SS, соответственно. В трех оставшихся регистрах ES, FS и GS программа может хранить указатели на дополнительные сегменты.

Организация памяти

Недостатки такой организации памяти:

- *сегменты бесконтрольно размещаются с любого адреса, кратного 16 (так как содержимое сегментного регистра аппаратно смещается на 4 разряда), и, как следствие, программа может обращаться по любым адресам, в том числе и реально не существующим*
- *сегменты имеют максимальный размер 64 Кбайт*
- *сегменты могут перекрываться другими сегментами*

Организация памяти

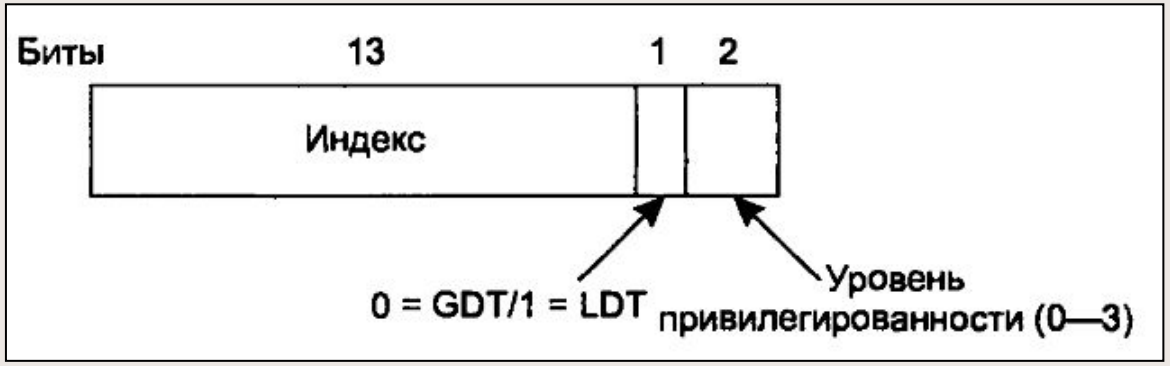
Защищенный режим адресации

При работе в защищенном режиме каждой программе может быть выделен блок памяти размером до 4 Гбайт, адреса которого в шестнадцатеричном представлении могут меняться от 00000000 до FFFFFFFF. При этом говорят, что программе выделяется линейное адресное пространство (linear address space).

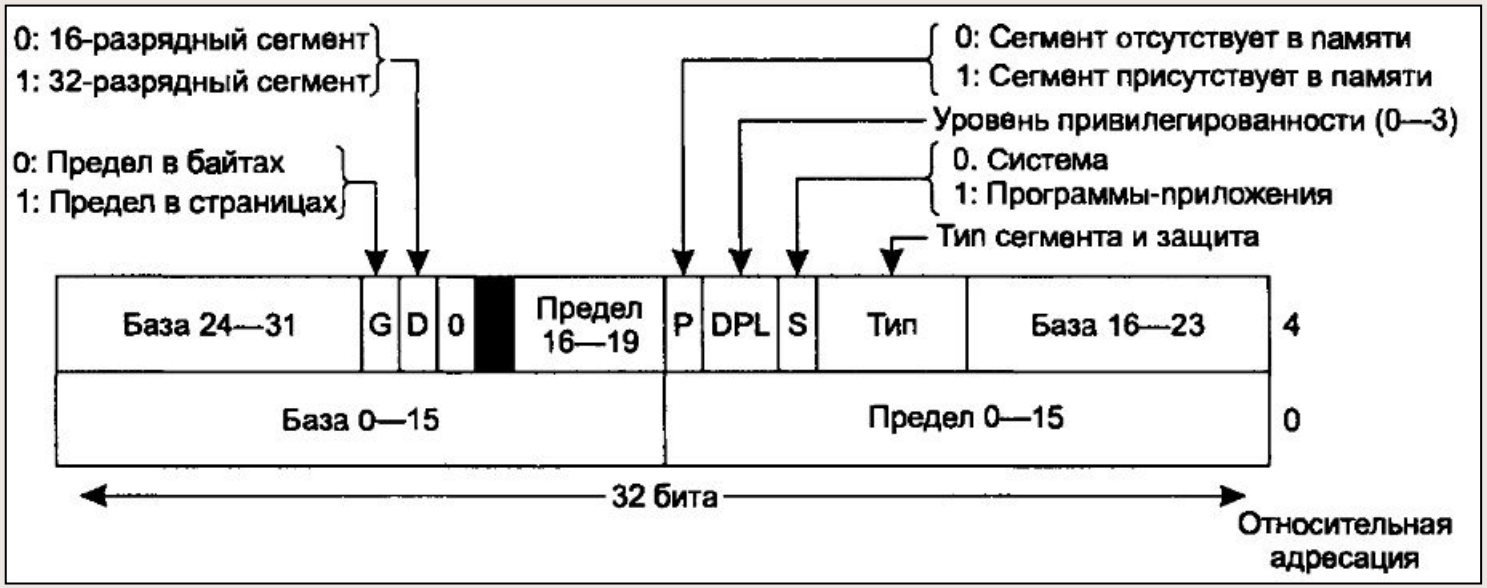
В защищенном режиме в сегментных регистрах (CS, DS, SS, ES, FS, GS) хранятся не 16–разрядные базовые адреса сегментов, а селекторы–указатели на дескрипторы сегмента (segment descriptor), расположенные в одной из системных таблиц дескрипторов (descriptor table). По информации, находящейся в дескрипторе, операционная система определяет линейные адреса сегментов программы. Существует две разновидности таблиц: Global Descriptor Table (глобальная таблица дескрипторов) и Local Descriptor Tables (локальные таблицы дескрипторов).

Организация памяти

Структура селектора дескриптора сегмента:



Дескриптор состоит из 8 байт, в которые входят базовый адрес сегмента, размер и другая информация:



Организация памяти

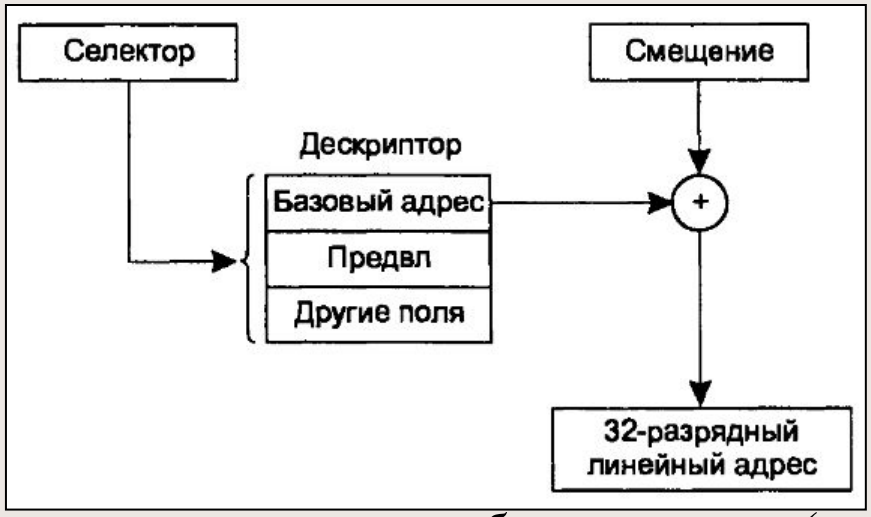
Дескриптор 0 является запрещенным — его можно безопасно загрузить в сегментный регистр, чтобы обозначить, что сегментный регистр в данный момент недоступен, но при попытке его использовать вырабатывается прерывание.

В типичной программе, написанной для защищенного режима, как правило, есть три сегмента: кода, данных и стека, информация о которых хранится в трех перечисленных ниже сегментных регистрах.

- *В регистре CS хранится указатель на дескриптор сегмента кода программы*
- *В регистре DS хранится указатель на дескриптор сегмента данных программы*
- *В регистре SS хранится указатель на дескриптор сегмента стека программы*

Организация памяти

Преобразование пары селектор–смещение в физический адрес осуществляется по следующей схеме:



Если разбиение на страницы блокировано (с помощью бита в глобальном управляющем регистре), линейный адрес интерпретируется как физический адрес и посылается в память для чтения или записи. С другой стороны, если доступна страничная подкачка, линейный адрес интерпретируется как виртуальный адрес и отображается на физический адрес с помощью таблицы страниц.

Организация памяти

В защищенном режиме аппаратно поддерживаются модели памяти:

Flat Model (плоская, сплошная или линейная модель) *– организация памяти, при которой все сегменты отображаются на единственную область линейных адресов. Для этого дескрипторы всех сегментов указывают на один и тот же сегмент памяти, который соответствует всему 32–разрядному физическому адресному пространству компьютера. Для плоской модели должно создаваться, как минимум, два дескриптора, один для ссылок к коду, а другой для ссылок к данным.*

Организация памяти

Дескрипторы хранятся в специальной системной таблице, которая называется таблицей глобальных дескрипторов (Global Descriptor Table, или GDT). Для плоской модели каждый дескриптор имеет базовый адрес, равный 0. Значение поля, определяющего границу сегмента, умножается процессором на шестнадцатеричное число 1000. Сегменты могут покрывать весь 4–х гигабайтный диапазон физических адресов, или только те адреса, которые отображаются на физическую память. Если установить границу сегмента в значение 4 гигабайта, механизм сегментации предотвращает генерацию исключений для ссылок к памяти, выходящих за границу сегмента.

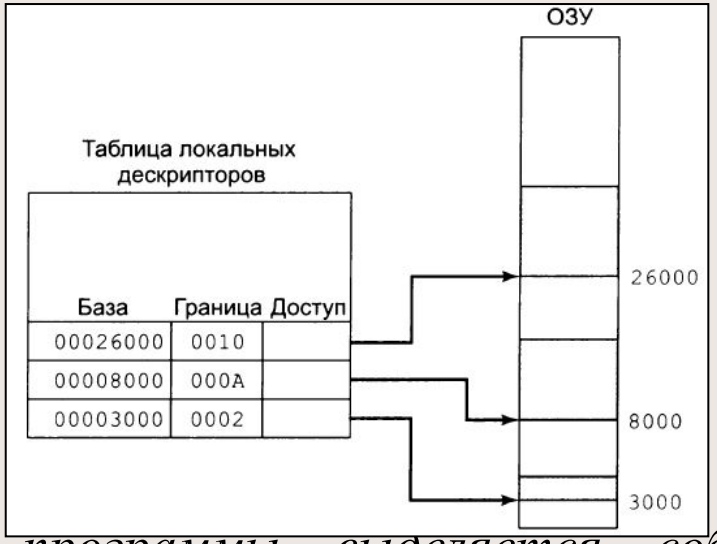
Организация памяти

Данная модель позволяет исключить механизм сегментации из архитектуры системы, так как все операции с памятью обращаются к общему пространству памяти.

С точки зрения программиста, эта модель наиболее проста в использовании, поскольку для хранения адреса любой переменной или команды достаточно одного 32–разрядного целого числа.

Организация памяти

Multisegmented Model (многосегментная модель)



Для каждой программы выделяется собственная таблица сегментных дескрипторов, которая называется таблицей локальных дескрипторов (Local Descriptor Table, или LDT). При этом появляется возможность для каждого процесса создать собственный набор сегментов, которые никак не пересекаются с сегментами других процессов. В результате каждый сегмент находится в изолированном адресном пространстве.

Организация памяти

На рисунке показано, что каждый элемент таблицы локальных дескрипторов определяет различные сегменты памяти. В каждом дескрипторе сегмента указывается его точная длина. Например, сегмент, начинающийся с адреса 3000, имеет длину 2000 байтов в шестнадцатеричном представлении, поскольку значение поля дескриптора, определяющего границу сегмента, равно 0002, а 0002x1000=2000. По аналогии, длина сегмента, начинающегося с адреса 8000, равна A000.

Следует отметить, что Flat Model реализуется как частный случай сегментированной модели, когда программа обращается к сегменту, под который отведено все линейное пространство.

Организация памяти

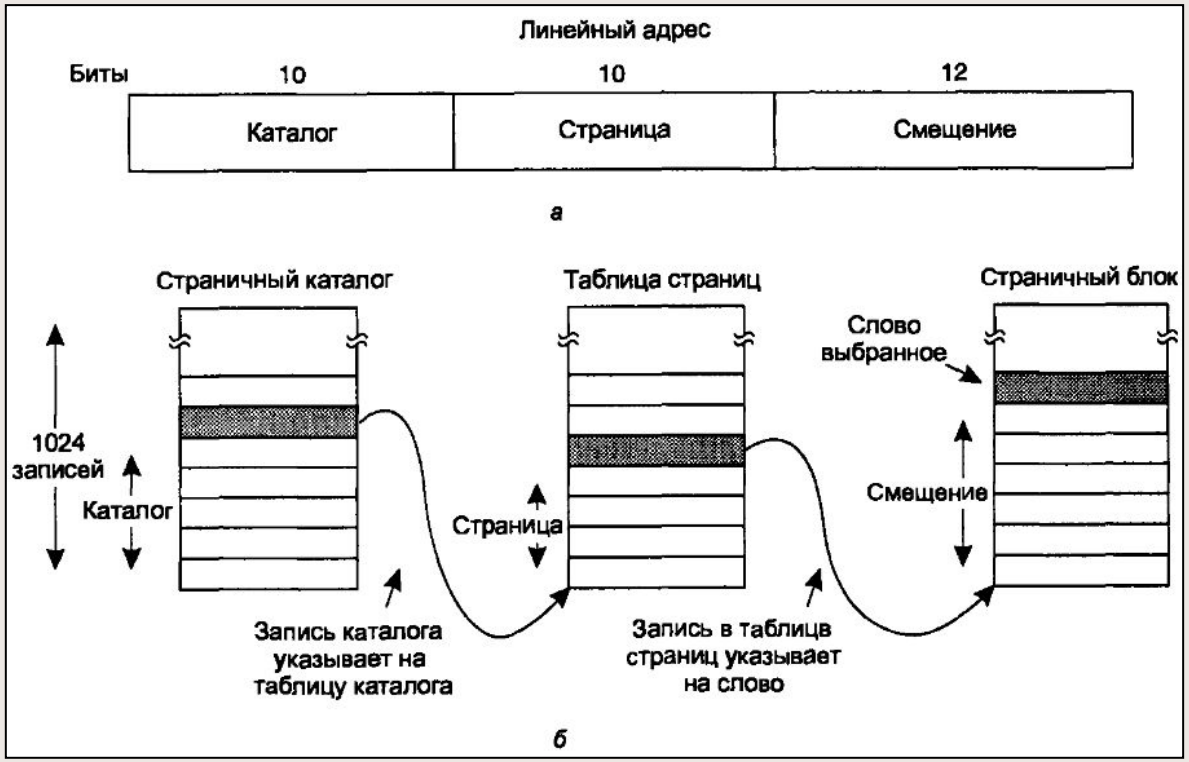
Paging (страничная модель памяти)

Эта модель представляет собой форму управления памятью для моделирования большого несегментированного адресного пространства с использованием части дисковой памяти и фрагментированного адресного пространства. Обеспечивает доступ к структурам данных, имеющим размер больше, чем размер доступного объема памяти, сохраняя их частично в оперативной памяти и частично на диске.

По этой модели линейное адресное пространство делится на блоки одинакового размера (обычно 4 Кбайт), которые называются страницами (page).

Организация памяти

На рисунке представлен линейный адрес, разделенный на три поля: Каталог, Страница и Смещение.



Поле Каталог используется как индекс в страничном каталоге, определяющий расположение указателя на правильную таблицу страниц.

Организация памяти

Затем обрабатывается поле Страница в качестве индекса в таблице страниц с целью найти физический адрес страничного блока. Чтобы получить физический адрес требуемого байта или слова, к адресу страничного блока прибавляется последнее поле Смещение.

В результате можно легко сделать так, чтобы суммарный объем оперативной памяти, используемой во всех выполняющихся на компьютере программах, превышал объем реальной памяти компьютера. Именно поэтому страничная организация памяти очень часто называется виртуальной памятью (virtual memory). Работоспособность системы виртуальной памяти обеспечивает специальная программа, являющаяся частью операционной системы, которая называется диспетчером виртуальной памяти (virtual memory manager).

Организация памяти

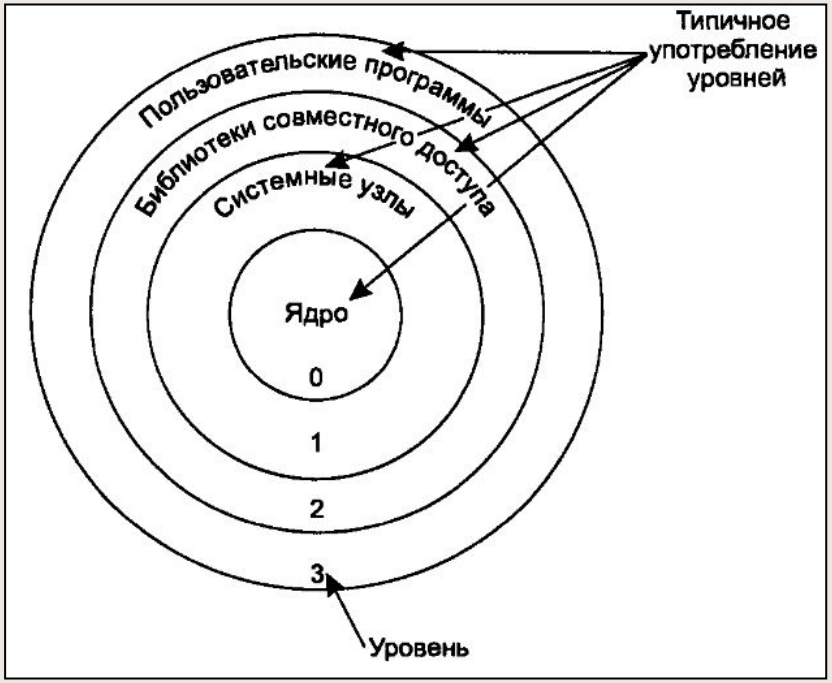
Страничная организация памяти как нельзя лучше решает проблему нехватки памяти. Дело в том, что перед началом выполнения любая программа должна быть загружена в оперативную память, размер которой, всегда ограничен (например, в силу конструктивных особенностей компьютера или цены модуля памяти). Пользователи компьютера обычно загружают в память сразу несколько программ, чтобы в процессе работы иметь возможность переключаться между ними (например, переходить из одного окна в другое). С другой стороны, объемы дисковой памяти намного превышают объемы оперативной памяти компьютера, да и к тому же эта память намного дешевле. Поэтому за счет привлечения дисковой памяти при использовании страничной организации памяти для пользователя создается впечатление, что он располагает ОЗУ неограниченного объема. Разумеется, что за все нужно платить: скорость доступа к дисковой памяти на несколько порядков ниже, чем к оперативной памяти.

Организация памяти

При выполнении программы, участки ее оперативной памяти (или страницы), которые не используются в данный момент, можно безболезненно сохранить на диске. Говорят, что часть задачи вытеснена (swapped) на диск. В оперативной памяти компьютера имеет смысл сохранять только те страницы, к которым процессор активно обращается, например, выполняет некоторый программный код. Если же процессор должен обратиться к странице памяти, которая в настоящий момент вытеснена на диск, происходит системная ошибка (или прерывание) из–за отсутствия страницы (pagefault). Обработкой этой ошибки занимается диспетчер виртуальной памяти операционной системы, который находит на диске страницу, содержащую нужный код или данные, и загружает ее в свободный участок оперативной памяти.

Организация памяти

С виртуальной памятью тесно связана тема защиты. Pentium поддерживает четыре уровня защиты, где уровень 0 является наиболее привилегированным, а уровень 3 — наименее привилегированным. В каждый момент времени работающая программа находится на определенном уровне Каждый сегмент в системе также имеет свой уровень.



Организация памяти

На уровне 0 находится ядро операционной системы, занимающееся обработкой операций ввода/вывода, управлением памятью и другими первоочередными вопросами. На уровне 1 – обработчик системных вызовов. Пользовательские программы этого уровня могут обращаться к процедурам для выполнения системных вызовов, но только к определенному и защищенному списку процедур. Уровень 2 содержит библиотечные процедуры, возможно, совместно используемые несколькими работающими программами. Пользовательские программы вправе вызывать эти процедуры и читать их данные, но не могут их изменять. И, наконец, пользовательские программы работают на уровне 3, который имеет наименьшую степень защиты.