Аппаратные средства телекоммуникационных систем

Лекция 3. Модель памяти процессоров

Типы организации памяти процессора

Лекция 3. Модель памяти процессоров

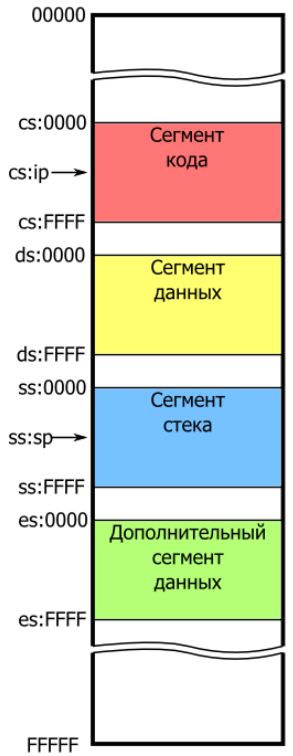
Аппаратные средства телекоммуникационных систем

Модель памяти процессора

- **Модель памяти процессора** – метод организации пространства ОЗУ (доступа к памяти) с аппаратной и/ или программной точки зрения.
- **Плоская модель памяти** — это метод организации адресного пространства оперативной памяти, в котором программная память и память данных находятся в одном адресном пространстве (*Используется в современных ЭВМ*).
  - Для 16-битных процессоров плоская модель памяти позволяет адресовать 64 кБ оперативной памяти; для 32-битных процессоров 4 ГБ, для 64-битных — до 16 эксабайт (для amd64 размер ограничен 256 ТБ).
    - Например адресное пространство для 32 битного режима будет состоять из 2^32 ячеек памяти пронумерованных от 0 и до 2^32-1

Модель памяти процессора

- **Сегментная модель памяти (модель адресации памяти) –** модель в которой память разбита на сегменты, каждый из которых характеризуется своим функционалом
- В общем случае каждый сегмент имеет свою цель: стек, данные, программы.
- Также память может быть разбита на сегменты по уровню доступа (ОС, приложения, драйвера и тп)
- Каждый такой сегмент имеет свои настройки и права доступа



Модель памяти процессора

- **Виртуальная память** — модель памяти ПК, в которой процессор работает с виртуальной памятью, транслируемой в физические адреса устройством управления памятью (MMU).
  - Преимущество – виртуальное расширение ОЗУ за счет файла подкачки.
- **Страничная виртуальная модель памяти –** способ организации виртуального адресного пространства, в котором единицей отображения виртуального адреса в физический является область фиксированного размера (страница).
  - Страницы могут быть не только в ОЗУ, но и сброшены в файл подкачки на жесткий диск.
  - Физический адрес = адрес страницы + смещение внутри нее.
  - Организацией работы страниц занимается операционная система.

Модель памяти процессора. Особые виды памяти

- **Регистровая память** – набор регистров процессора, ячейки памяти в самом процессоре.
  - **Basic program registers** (Основные программные регистры) - это для обслуживания процессора и обработки целочисленных данных.
    Floating Point Unit registers (FPU, X87) – это набор регистров для работы с данными в формате с плавающей точкой.
    MMX и XMM registers - это регистры для систематизированной обработки увеличенного количества операндов.
  - Когда процессор совершает какие-то операции со значением или с памятью, он берет эти значения непосредственно из регистров или из стека.
- **Стек** - специальный раздел оперативной памяти, предназначенный для быстрого безадресного доступа к элементам (по принципу last input first output).

Особенности сегментов памяти

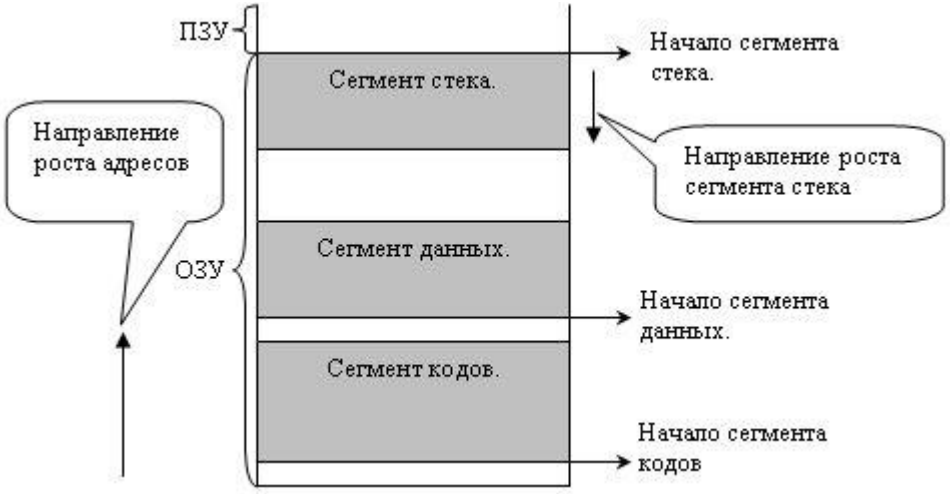
Лекция 3. Модель памяти процессоров

Аппаратные средства телекоммуникационных систем

Модель памяти процессора. Сегменты памяти

Логически память разделена на 3 основных сегмента памяти.

- **CS** - **сегмент кода**, содержит машинные команды (проргамму);
- **DS** - **сегмент данных** – содержит данные, то есть константы и рабочие области, необходимые программе;
- **SS - сегмент стека** – содержит адреса возврата в точку вызова подпрограмм.
- адресу определенному ОС.

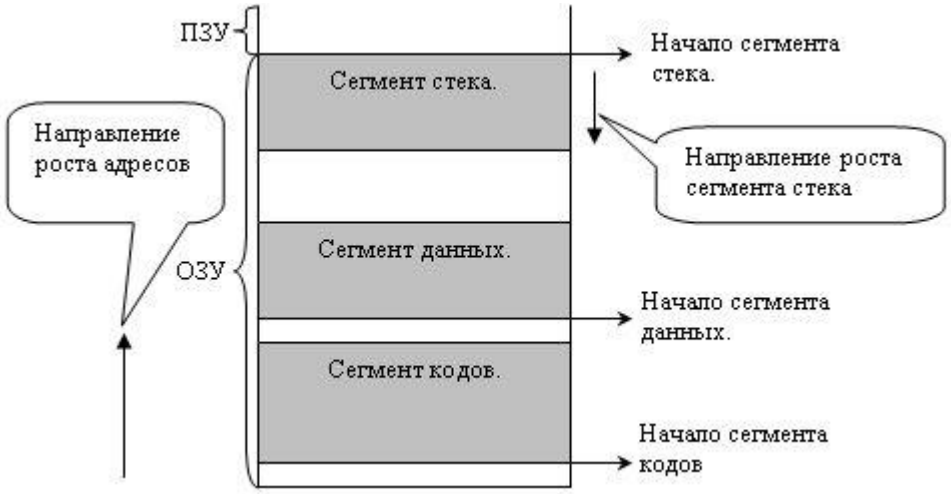


Модель памяти процессора. Сегменты памяти

- При записи команд на языке Ассемблера принято указывать адреса с помощью следующей конструкции:

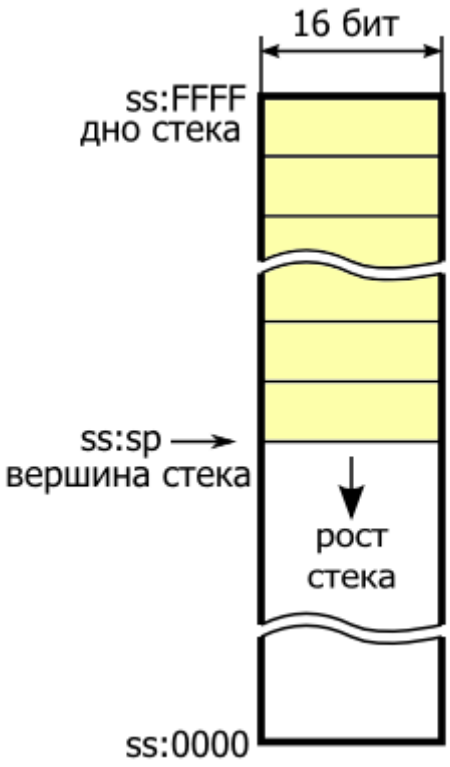
<адрес сегмента>:<смещение> или <сегментный регистр>:<адресное выражение>

- За распределение сегментов, их начальный (базовый) адрес и их размер отвечает дескриптор сегментов – 64 бита памяти, расположенных по адресу определенному ОС.
- В современных системах базовые адреса всех сегментов могут быть одним нулевым адресом (то есть ОС сама распределяет где какой сегмент будет сама).



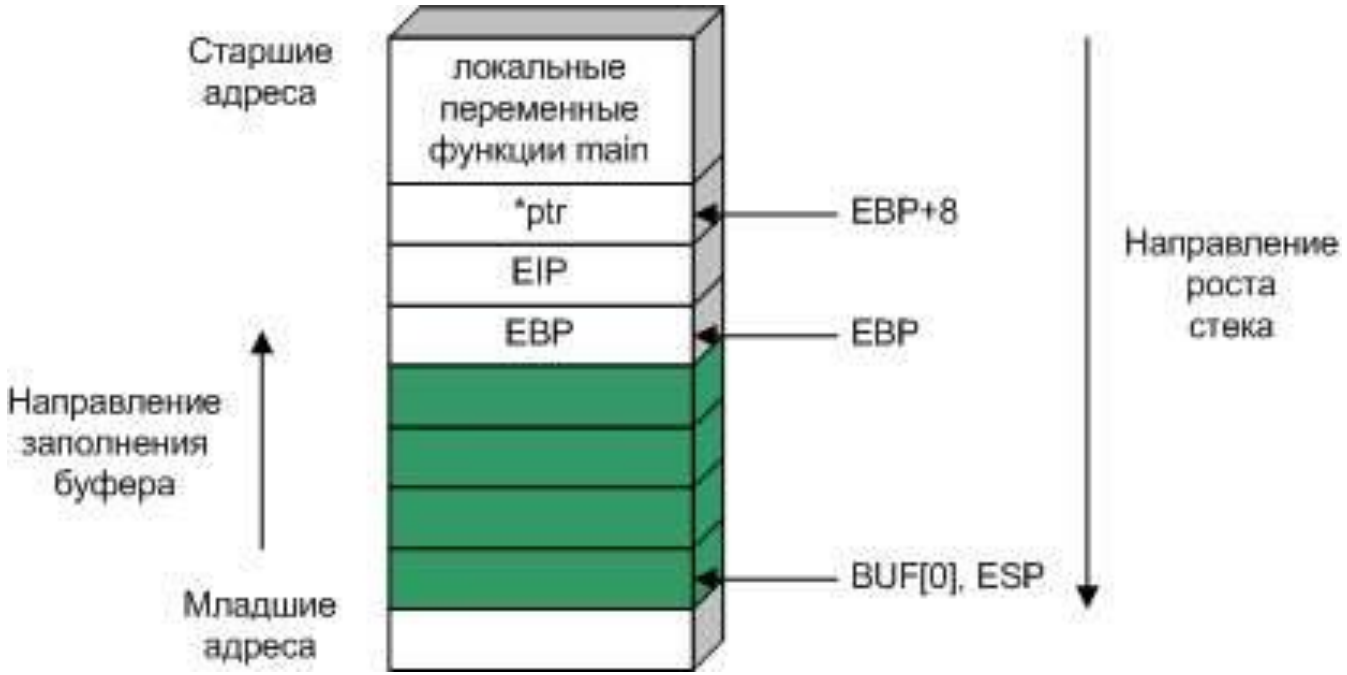
Модель памяти процессора. Стек

- Стек располагается в оперативной памяти в сегменте стека, и поэтому адресуется относительно сегментного регистра SS.
- Ширина стека называется размер элементов, которые можно помещать в него или извлекать.
- Чаще всего ширина стека равна двум байтам 16 бит.
- Регистр SP (указатель стека) содержит адрес последнего добавленного элемента - вершина стека. Противоположный конец стека называется дном.
- Дно стека находится в верхних адресах памяти. При добавлении новых элементов в стек значение регистра SP уменьшается, то есть стек растёт в сторону младших адресов.
- При добавлении значения в стек регистр SP повышается на 2 при выборке значения понижается.



Модель памяти процессора. Стек

- Стек расположен в оперативной памяти, каждый элемент занимает одно слово
- ESP – хранит адрес вершины стека
- EBP – хранит адрес начала стекового фрейма
- SS – регистр, хранит селектор стека
- Стек – растет от старших адресов к младшим (LIFO)



Модель памяти процессора. Стек

- **Стек служит для загрузки каких-то данных в процессе работы, например при вызове прерывания в стек помещают значения счетчика команд и последних операндах и результате работы АЛУ.**
- Стек используется для организации многопоточной системы – при переключении между потоками состояние каждого процесса (потока) сохраняется в стеке.
- Из сказанного очевидно, что стек позволяет организовать рекурсии, деревья, графы и т.д.
- **В современных соглашениях о вызове подпрограмм (напр., fastcall и stdcall) стек используется для передачи большого числа аргументов в подпрограмму. В случае fastcall – также и через регистры (1-4 аргументы для AMD64).**

Особенности страничной организации виртуальной памяти

Лекция 3. Модель памяти процессоров

Аппаратные средства телекоммуникационных систем

Модель памяти процессора. Страничная организация виртуальной памяти

- **Страничная организация памяти** - способ организации виртуальной памяти, при котором единицей отображения (трансляции) виртуальных адресов на физические является регион постоянного размера (т. н. страница).
  - Страничная организация включается/выключается в регистре CR0.
  - **в 32-х битных системах (т.н. защищенный режим)** - доступ к 4 ГБ виртуальной памяти.
  - **в 64-х битных системах (Расширенный режим) -** доступ к 64 ГБ памяти.

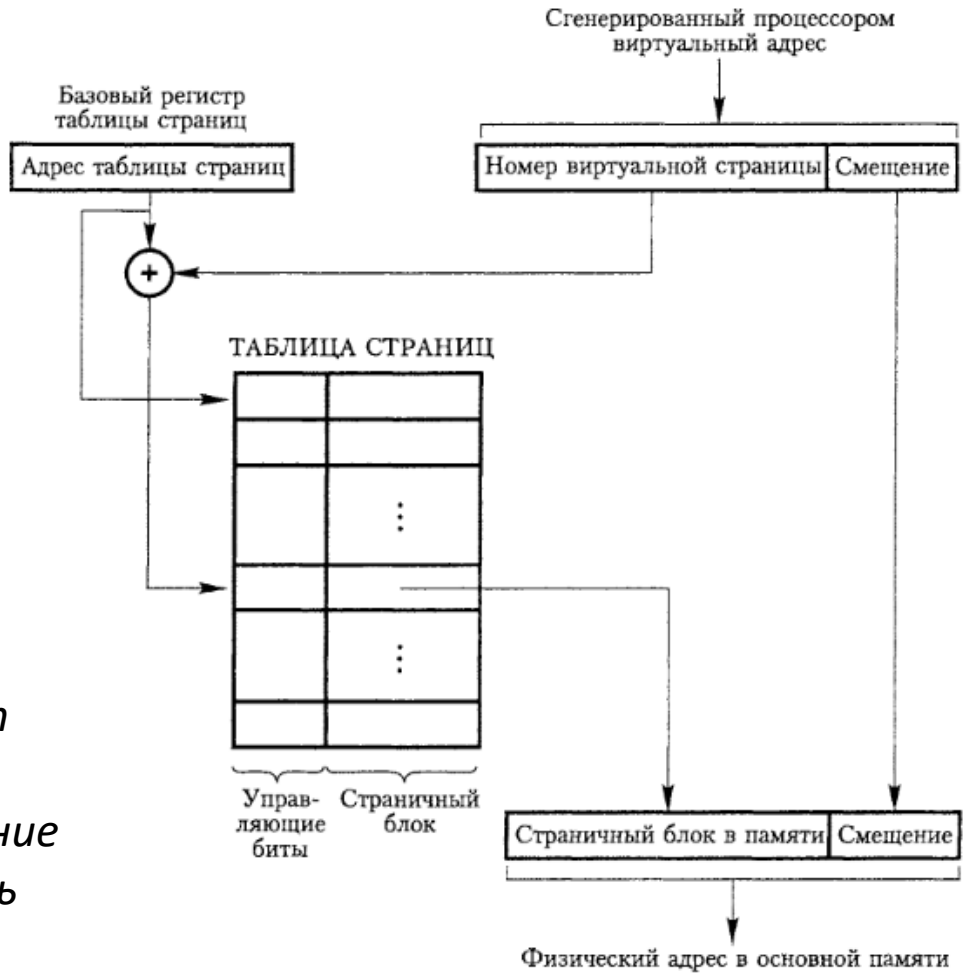
Модель памяти процессора. Страничная организация виртуальной памяти

- Типичные размеры страницы — 4 кБ и 4 Мб, 2 Мб (в PAE), 1 ГБ (long-mode).
  - **Страницы по 4 кБ объединены в таблицу страниц (1024 таблиц), таблицы объединены в каталог страниц(1024 таблицы).**
  - *Страницы по 4 МБ таблицы объединяются в каталог страниц.*
    - Каталог страниц содержится в CS3 - PDBR (Page Directory Base Register).
    - Адрес ячейки = 10(каталог)+10(таблица)+12=32 бита.
    - Режим со страницами по 4МБ – PSE, включается отдельно в CS4.
    - *Адрес ячейки в PAE = 8(указатель каталога)+9(каталог)+9(таблица)+12=36 бит.*

Модель памяти процессора. Страничная организация виртуальной памяти

Виртуальный адрес страницы и физический могут не совпадать.

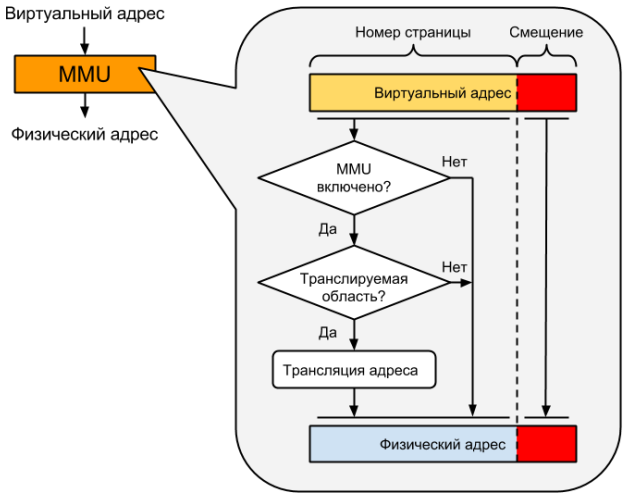
- *у одной страницы могут быть разные виртуальные адреса и один физический.*
- При отсутствии ресурсов в ОЗУ часть страниц могу быть сброшены на жесткий диск пока память в ОЗУ не будет освобождена (файл подкачки).
- *Страницы в файле подкачки будут отмечены как отсутствующие, обращение к ним выдаст исключение #PE, в котором ОС должна вернуть страницы в ОЗУ и перегрузить инструкцию обращения*.



Модель памяти процессора. Виртуальная память

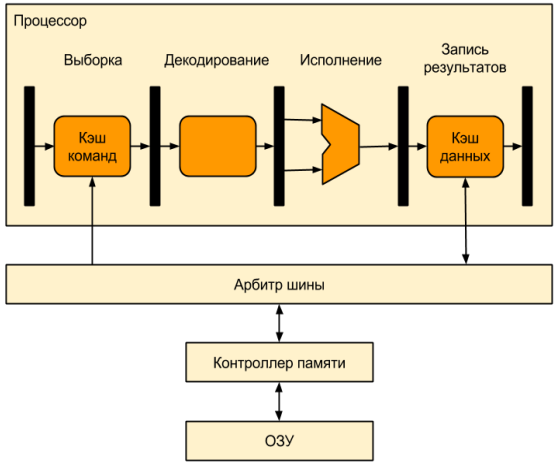
Устройство управления памятью –MMU – транслирует виртуальные адреса в физические.

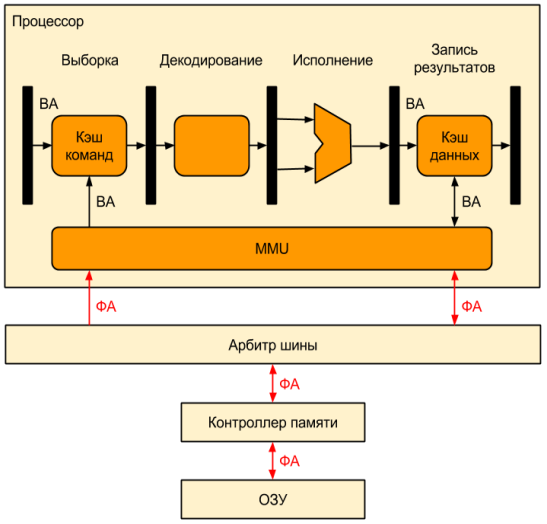
- Трансляция полностью аппаратная.
- Виртуальная память как правило страничная.
- *Трансляция адресов осуществляется через специальную кэш память TLB.*
  - Данная память хранит адреса часто обращаемых виртуальных страниц.



Если адреса нет в TLB, то модуль MMU ищет адрес по всем таблицам страниц каждого процессора.

Для улучшения производительности MMU таблицы страниц могут кэшироваться

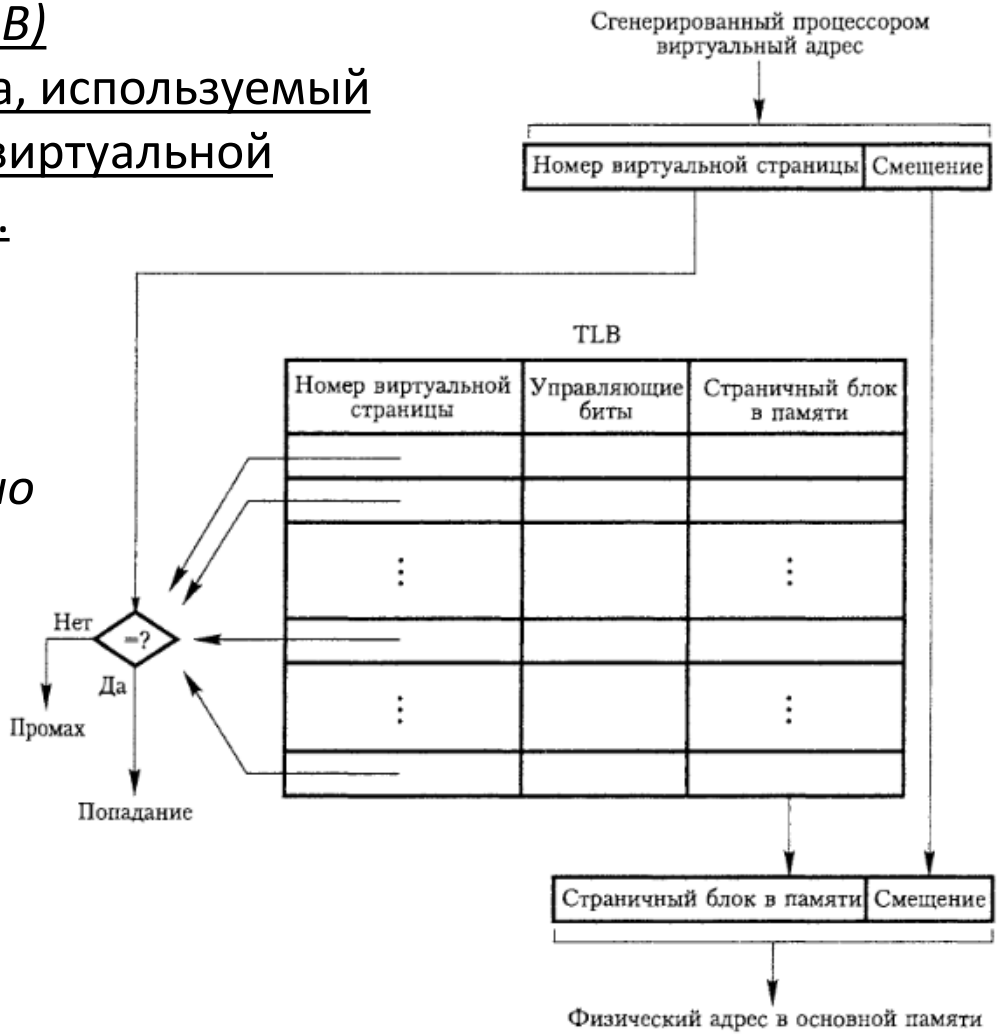




Модель памяти процессора. Страничная организация виртуальной памяти

Буфер ассоциативной трансляции
(англ. *Translation lookaside buffer, TLB)*
*сп*ец. кэш центрального процессора, используемый для ускорения трансляции адреса виртуальной памяти в адрес физической памяти.

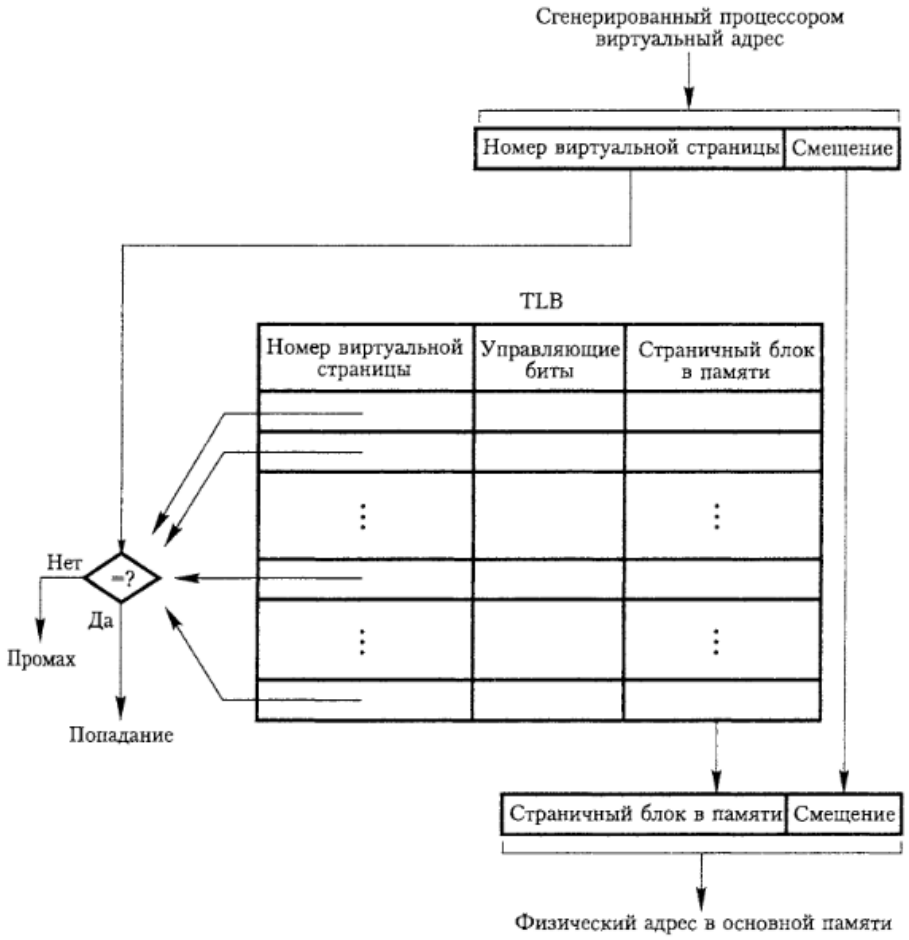
- *TLB вектор содержит сопоставления физических и виртуальных адресов для недавно использовавшихся страниц и атрибуты защиты каждой страницы*



Модель памяти процессора. Страничная организация виртуальной памяти

В результате кеширования исключается повторная трансляция страниц при обращении к одинаковым адресам .

- *Виртуального адреса в TLB нет - для его поиска понадобится несколько обращений к памяти,*
- *Вирт. Адрес есть - обращение будет сразу.*
- *TLB может быть многоуровневый*
- *А также разный для разного размера страниц.*



Модель памяти процессора. Виртуальная память

В случае нескольких ядер каждое ядро имеет свой TLB

Также возможны отдельные TLB для данных и команд.

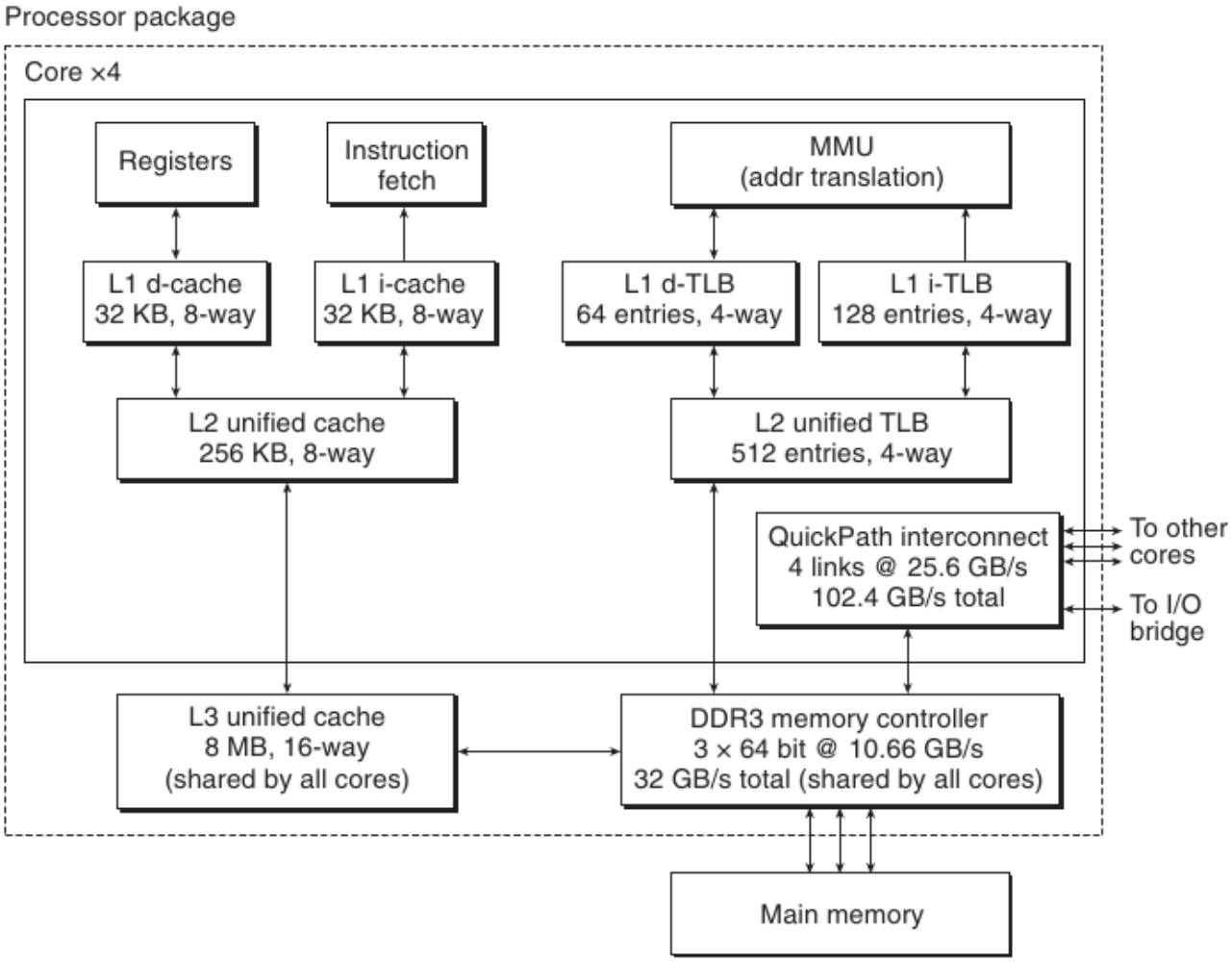


Figure 9.21 **The Core i7 memory system.**

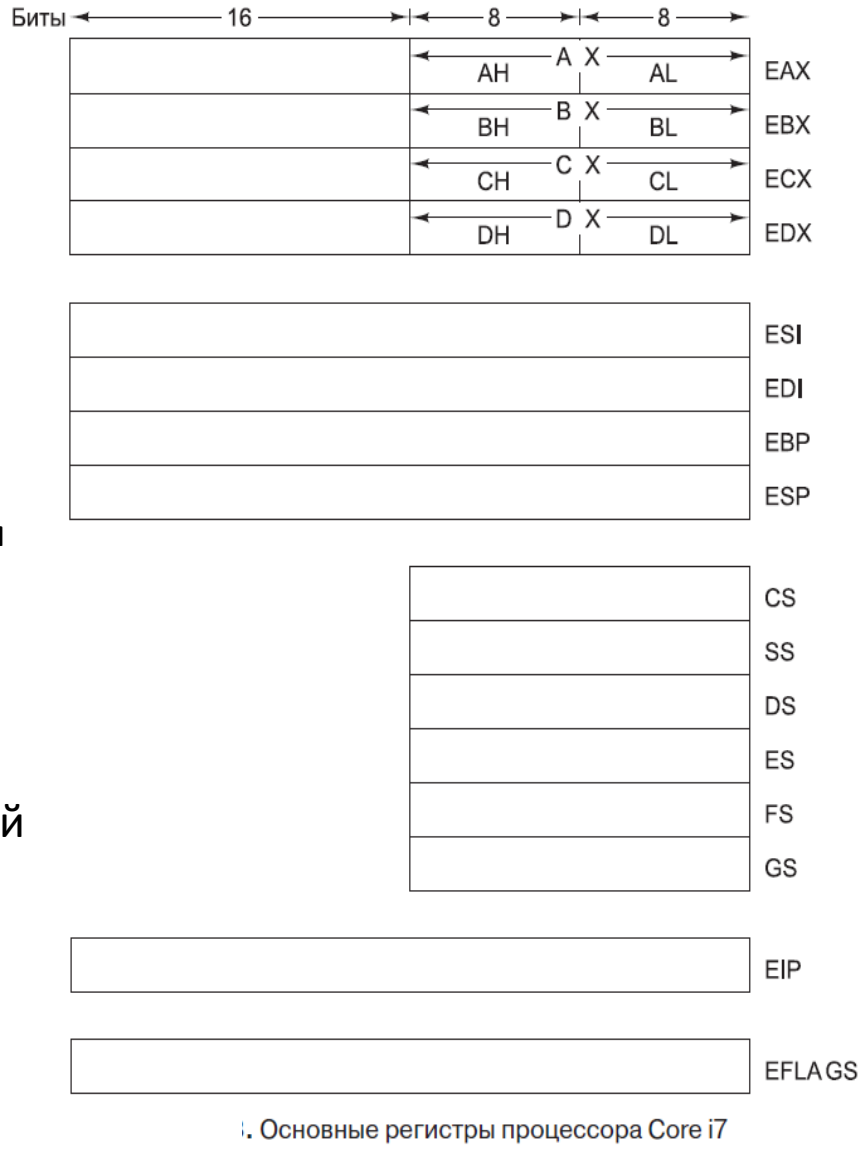
Особенности регистровой памяти

Лекция 3. Модель памяти процессоров

Аппаратные средства телекоммуникационных систем

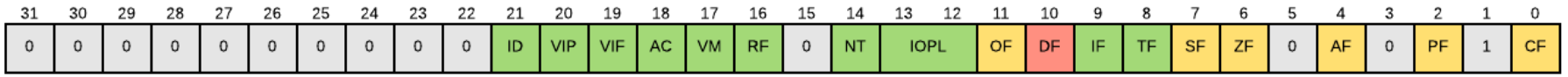
Модель памяти процессора. Регистровая память IA-32 (X86).

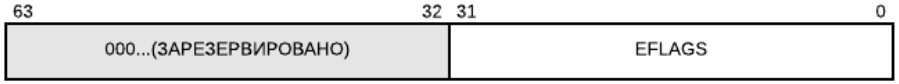
- **EAX, EBX, ECX и EDX – 32 разрядные GPR**
  - (GPR - регистры общего назначения);
  - также допустимы регистры 8 и 16 бит
  - **EAX** — основной арифметический регистр;
    - EBX предназначен для хранения указателей (адресов памяти);
  - **ECX** связан с организацией циклов;
  - **EDX** нужен для умножения и деления
    - вместе с EAX 64-разрядные произведения и делимые.
- **ESI и EDI** - указатели строковых команд:
  - **ESI** указывает на исходную строку,
  - **EDI** — на целевую.
- **EBP** предназначен для хранения указателей **(указатель кадра)**.
- **ESP** — это указатель стека
- **EIP** – счетчик команд
- **EFLAGS** — флаговый регистр.
- **CS-GS** сегментные регистры.



. Основные регистры процессора Core i7

Модель памяти процессора.
Регистровая память. Регистр флагов. Операционные фалги





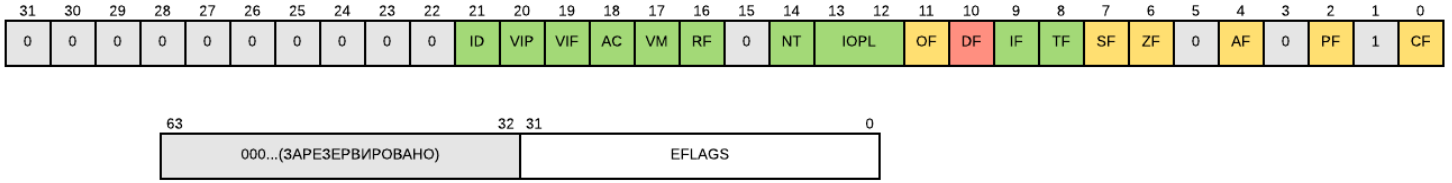
После выполнения очередной команды процессор сохраняет результат выполнения команды в регистре флагов

Регистр флагов служит для индикации процессору о результате выполнения каждой команды или текущем состоянии программы (напр. о прерываниях).

–**EFLAG тоже, что и RFLAG но для 32 бит**

- **Флаг CF, Carry Flag (бит 0)** (флаг переноса для без знаковых чисел)
- **Флаг PF, Parity Flag (бит 2)** 1 если значение результата АЛУ четное.
- **Флаг AF, Auxiliary Carry Flag (бит 4)** перенос для двоично-десятичных чисел
- **Флаг ZF, Zero Flag (бит 6)** 1, если результат последней операции 0
- **Флаг SF, Sign Flag (бит 7)** 1, если результат последней операции <0
- **Флаг OF, Overflow Flag (бит 11)** 1, если переполнение (перенос знаковых чисел)
- **Флаг DF, Direction Flag (бит 10)** 1 авто-декремент (обрабатывать строки от верхнего адреса к нижнему), 0 - наоборот .

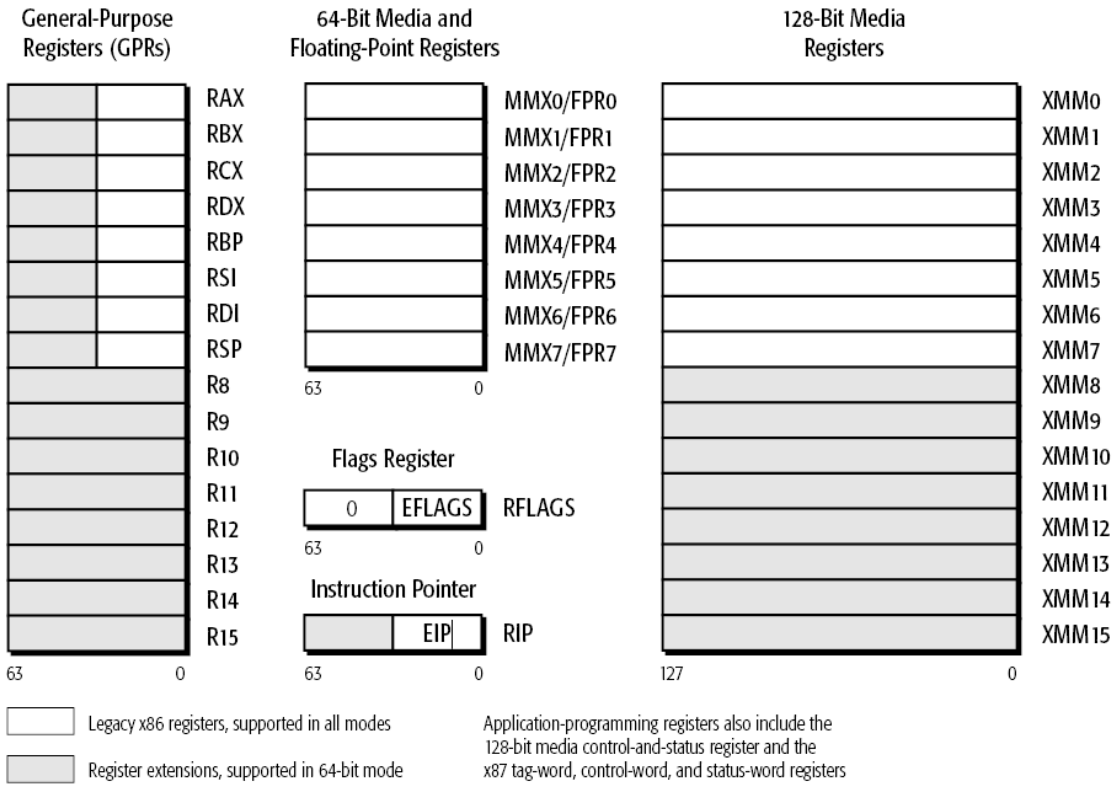
Модель памяти процессора.
Регистровая память. Регистр флагов. Системные флаги



- Системные флаги - **не должны изменяться прикладными программами.**
- **Флаг TF, Trap Flag (бит 8)** 1 для дебага step-to-step.
- **Флаг IF, Interrupt enable flag (бит 9)** 1 если процесс в прерывании
- **Поле IOPL, I/O privilege level field (биты 12 и 13)** уровень приоритета обрабатываемого устройства В-В
- **Флаг NT, Nested task flag (бит 14)** 1 если текущая команда связана с предыдущей.
- **Флаг RF, Resume Flag (бит 16)** 1 исключения в режиме дебага.
- **Флаг VM, Virtual-8086 mode flag (бит 17)** 1 для виртуальной совместимости с 8086.
- **Флаг AC, Alignment check (or access control) flag (бит 18)** режим выравнивания бит
- **Флаг VIF, Virtual interrupt flag (бит 19)** доп. Флаг порываний для предотвращения конфликта прерываний с разными приоритетами.
- **Флаг VIP, Virtual interrupt pending flag (бит 20)** 1 если прерывание ожидает (напр. Окончания другого прерывания).
- **Флаг ID, Identification flag (бит 21)** поддержка режима получения инф. О процессоре.

Модель памяти процессора. Регистровая память AMD64 (X64)

- 16 регистров общего назначения 64-битные (RAX-RDX~=EAX-EDX);
- 8 128-битных XMM регистров (SSE команды);
- 8 64-битных регистров MMX (3D Now команды)
- **специальный режим "Long Mode":**
  - до 64-бит виртуальных адресов;
  - 64-битные счетчик команд (RIP);
  - 64 битный регистр флагов RFLAGS



AMD64 имеет 3 режима доступа к памяти:

- реальный режим,
- защищённый режим
- 64-разрядный режим, или long mode

Модель памяти процессора. Виды команд X86-X64

- **Команды общего назначения. Основные x86 целочисленные команды.**
  - Большинство из них предназначены для загрузки, сохранения, обработки данных, расположенных в регистрах общего назначения или памяти. Некоторые из этих инструкций управляют потоком команд, обеспечивая переход к другому месту в программе.
- **x87 команды. Обрабатывают данные в x87 регистрах (FUP сопроцессор).**
  - Предназначены для работы с плавающей точкой в x87 приложениях.

Модель памяти процессора. Виды команд X86-X64

- **128-битные медиа-команды. SSE, SSE2 и SSE3 (streaming SIMD extension).**
  - Команды предназначенные для загрузки, сохранения, или обработки данных, расположенных в 128-битных XMM регистрах.
  - Команды выполняют операции целочисленные или с плавающей точкой над векторными (упакованными) и скалярными типами данных.
  - Векторные инструкции могут независимо выполнять одну операцию над множеством данных (SIMD) командами.
  - Векторные команды используются для медиа- и научных приложений для обработки блоков данных.
- **64-битные медиа-команды. Multimedia extension (MMX) и 3DNow! команды**.
  - Команды сохраняют, восстанавливают и обрабатывают данные, расположенные в 64-битных MMX регистрах.
  - Команды выполняют операции целочисленные и с плавающей точкой надо векторными (упакованными) и скалярными данными как и XMM.

Режимы работы процессора по принципу организации памяти

Лекция 3. Модель памяти процессоров

Аппаратные средства телекоммуникационных систем

Модель памяти процессора. Режимы работы процессора.

Основные режимы

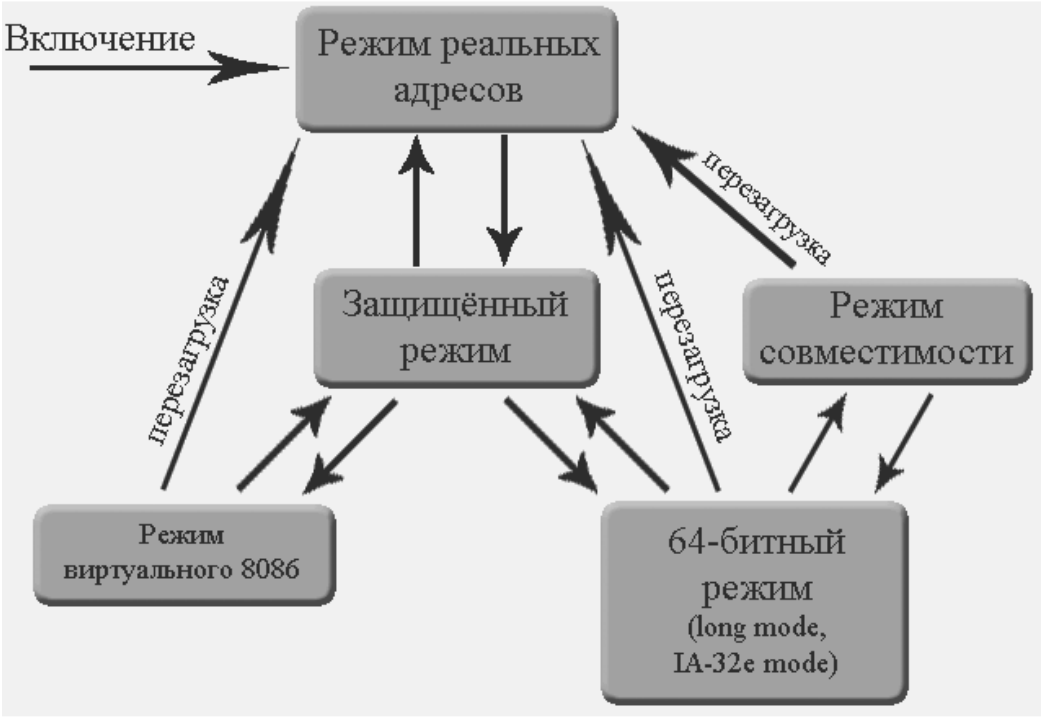
– **Реальный режим** (при включении, 64 кБ RAM) и виртуальны 8086 для совместимости с 16 битными приложениями.

– **Защищенный режим** (32 битный, 4 ГБ RAM) и расширенный (64 ГБ RAM)

– **Длинный режим** (64 битный, 256 ТБ RAM) и режим совместимости.

Режимы отличаются:

- методом и уровнями доступа к памяти,
- Допустимым объемом памяти
- Набором команд
- Размером слова



Модель памяти процессора. Виды адресов X86-X64

- **Физический адрес** – это адрес в системной памяти компьютера, именно тот адрес, который выставляется на шину адреса.
- **Логический адрес** – адрес с указанием сегмента в формате – «сегмент: смещение»
  - сегмент указывается в сегментном регистре (cs, ds, ss) или непосредственно значением (это значение может быть только 16-битным), а адрес – в обычном регистре или непосредственно значением (это значение может быть 16-, 32-, 64-битным в зависимости от режима).
  - способ преобразования логического адреса в физический зависит от режима процессора.

Модель памяти процессора. Виды адресов X86-X64

- **Линейный адрес** – адрес полученный после преобразования логического адреса
  - После преобразования адреса получается абсолютный 20-, 32-, 64-битный адрес (в зависимости от режима); этот адрес называется линейным.
  - В режиме реальных адресов физический адрес сразу выставляется на шину адреса.
- **Виртуальный адрес** – линейный адрес, полученный в 64 совместимом режиме при помощи механизма трансляции. (Механизм задается ОС, при отсутствии механизма виртуальный адрес = линейный).

Модель памяти процессора. Режимы работы x86-x64. Реальный режим

- **Реальный режим** – это режим, в который переходит процессор после включения или перезагрузки.
  - Стандартный 16-разрядный режим,
  - доступно 1 Мб физической памяти и возможности процессора почти не используются
  - все адреса, к которым обращаются программы, являются физическими, т. е. без какого-либо преобразования будут выставлены на шину адреса.
  - Размер слова 2 байта (WORD);
  - Вся память делится на сегменты размером 64 Кб;
    - *физический_адрес = сегмент * 10h + смещение*
    - Смещение 16-бит значение в РОН или const.
    - Первые 1024 байт заняты таблицей порываний (256 адресов подпрограмм прерываний) с нулевого адреса.

Модель памяти процессора. Режимы работы x86-x64. Защищенный режим

- **Защищённый режим (protected mode, или legacy mode (AMD))**– это 32-разрядный режим (X86);
  - доступ к 4-гигабайтному адресному пространству,
    - *при включении механизма трансляции адресов можно получить доступ к 64 Гб памяти.*
  - В защищённый режим можно перейти только из реального режима (CR0:0).
  - Размер слова 4 байта, или двойное слово (*DWORD double word*).
  - Все операнды, которые выступают как адреса, должны быть 32-битными;
  - Защищенный режим использует страничную модель памяти.

Модель памяти процессора. Режимы работы x86-x64. Защищенный режим

- **Защищённый режим (protected mode, или legacy mode (AMD))**– это 32-разрядный режим (X86);
  - **За работу защищенного режима отвечает операционная система (ОС);**
  - Защищённый режим называется так потому, что позволяет защитить данные операционной системы от приложений,
    - Например часть памяти резервируется по привилегированные данные ОС.
    - В режиме возможна многозадачность.
    - Для каждой задачи выделяется отдельная область виртуальной памяти – описываемая дескриптором (адрес начала, размер, особенности доступа).
  - *Если в защищённом режиме происходит нарушение условий защиты, то процессор генерирует специальное прерывание – исключение.*

Модель памяти процессора. Режимы работы x86-x64. Защищенный режим

Особенности защищенного режима:

- Сегментная и страничная адресация памяти. Поддержка динамического размещения процессов в памяти (виртуальной памяти).
- Организация многозадачности - аппаратное переключение контекстов процессов при смене процессов операционной системой.
- Защита памяти и программ:
  - контроль обращения программ к памяти и портам ввода-вывода в соответствии с назначенным им операционной системой «уровнем привилегий»;
  - контроль недопустимости использования в прикладных программах «привилегированных команд», разрешенных только для операционной системы;
  - разделение адресных пространств загруженных процессов
  - контроль попыток прикладных программ выйти за пределы своего сегмента;
  - контроль попыток чтения или записи в запрещенные операционной системой сегменты памяти.

Модель памяти процессора. Режимы работы x86-x64. Защищенный режим

Защита сегментов памяти обеспечивается следующими аппаратно-программными мерами:

- изолирование операционной системой адресных пространств задач с помощью Локальных дескрипторных таблиц;
- контроль попыток программного «выхода за границы сегмента» по величине смещения, выполняемый процессором;
- наложение операционной системой на сегменты или страницы ограничений на чтение и запись и контроль попыток их нарушения со стороны процессора;
- назначение операционной системой сегментам различных «уровней привилегий» и контроль программного обращения по соответствию этому уровню со стороны процессора.

Модель памяти процессора. Режимы работы x86-x64. Защищенный режим

Защищенный режим допускает два распределения адресного пространства между задачами. Эти решения принимает операционная система:

- а) *единое адресное пространство для всех задач (как в реальном режиме).* Для этого операционной системе достаточно создать только Глобальную дескрипторную таблицу (GDT), где содержится описание всех сегментов. Локальные дескрипторные таблицы не создаются. Тогда регистр-селектор локальной дескрипторной таблицы - LDTR должен содержать пустой селектор - 0.
- Такая структура распределения памяти удобна для «перевода» программных систем, написанных для реального режима, в защищенный режим. Однако, здесь могут быть проблемы с защитой сегментов задач в памяти, так как они используют единое адресное пространство. Это скажется на устойчивости операционной системы в целом.
- б) *собственное адресное пространство для каждой задачи.* Операционная система создает отдельную Локальную дескрипторную таблицу (LDT) для каждой задачи. В LDT описываются сегменты памяти, которые будут доступны *только этой задаче.* Необходимое общее адресное пространство обеспечивается через глобальную дескрипторную таблицу GDT.
- Такая реализация требует больших затрат от операционной системы, но обеспечивает максимальную защиту адресных пространств отдельных задач.

Модель памяти процессора. Режимы работы x86-x64. Защищенный режим. Уровни привилегий

- **Уровни привилегий доступа к памяти:**
  - уровень 0: ядро операционной системы;
  - уровень 1: драйверы ОС;
  - уровень 2: интерфейс ОС;
  - уровень 3: прикладные программы.
  - В современных ОС уровни 0 - 2 могут иметь одинаковые привилегии уровня 0, но уровень 3 в защищённом режиме всегда имеет привилегии ниже чем другие.
  - **Программы и данные ограничены внутри своих уровней привилегий.**
- **Каждый уровень привилегий имеет свой сегмент в памяти**
  - размер сегмента может быть от 0 до 4ГБ.
  - Первые 256 байт каждого сегмента резервируются под служебную информацию.
  - Адрес сегмента – слектор (16 бит).
    - логический адрес – адрес внутри сегмента

Модель памяти процессора. Режимы работы x86-x64. Защищенный режим. Сегменты памяти

- *Каждый сегмент и объект в памяти, который управляется процессором описываются в выделенной отдельно области памяти* – **дескрипторе** (8 байт),
  - — Дескрипторы объединяются в таблицы декскрипторов.
  - — Дескриптор описывает адресацию сегмента и уровни доступа.
  - — Существуют
    - — глобальная таблица декскрипторов (GDT);
    - — локальная таблица декскрипторов (LDT);
    - — таблица векторов прерываний (IDT);
    - — Отдельная область памяти – шлюз вызовов (call gates)
    - — Также отдельно может быть выделен сегмент задач TSS (Task state segment)
  - — Механизм дескрипторов позволяет организовать многозадачность в защищенном режиме.
  - — в защищенном режиме регистры селекторы содержат *номер дескриптора сегмента. Например,* во время исполнения текущего процесса регистр CS содержит селектор сегмента кода, а регистр DS - селектор сегмента данных

Модель памяти процессора. Режимы работы x86-x64. Защищенный режим. Сегменты памяти

Адрес каждого дескриптора содержится с специальных регистрах процессора – т.н. сегментных регистрах (доступны только ОС):

- **Регистр GDTR** - 48-разрядный регистр адреса глобальной дескрипторной таблицы (GDT). Содержит полный 32-разрядный адрес начала размещения сегмента глобальной дескрипторной таблицы в памяти и ее размер (предел смещения).
- **Регистр IDTR** - 48-разрядный регистр адреса таблицы дескрипторов прерываний (IDT). Содержит 32- разрядный адрес сегмента таблицы дескрипторов прерываний в памяти и предел смещения в ней.
- **Регистр LDTR** - 16-разрядный регистр - селектор сегмента локальной дескрипторной таблицы (LDT). Используется для определения адреса размещения в памяти локальной дескрипторной таблицы.
- **Регистр TR** - 16-разрядный регистр - селектор сегмента состояния задачи (TSS). Используется при переключении задач.
- Также для управления работой ОС используются системные регистры CR0,CR1,… - в которых содержаться специальные флаги для ОС и регистра DR0,.. отладки.

Модель памяти процессора. Режимы работы x86-x64. Защищенный режим. Дескрипторы

- **Глобальные дескрипторы** – (таблица *GDT, ее адрес в регистре* GDTR).
- предназначена для описания сегментов операционной системы и общих сегментов для всех прикладных процессов, например сегментов межпроцессного взаимодействия (шлюзов); сегментов состояния процессора (TSS);
- Таблица GDT наряду с записями об специальных сегментах содержит запись о самой себе, а также обо всех таблицах LDT.
- GDTR не может меняться в ходе работы ОС (устанавливается при переходе в защищённый режим; LDTR меняется для каждой текущей задачи (LDT), выполняемой ОС.
-

Модель памяти процессора. Режимы работы x86-x64. Защищенный режим. Дескрипторы

- **Локальные дескрипторы (таблица LDT,** *адрес* в LDTR**) -** дескрипторы, создаваемые ОС под каждый пользовательский процесс (задачу).
  - Главное отличие LDT от GDT - в ней нельзя определять дескрипторы системных объектов – объектов, которые использует процессор.
  - Для передачи данных между дескрипторами используется особая область памяти – шлюз, которая описывается локальным дескриптором шлюзов.
  - В GDT можно определить несколько таблиц LDT, но только один является текущим в любой момент времени: обычно он связан с текущей Задачей.
  - LDT создаются операционной системой динамически под конкретные задачи – адрес-сегектор текущая LDT хранится в LDTR.

Модель памяти процессора. Режимы работы x86-x64. Защищенный режим. Дескрипторы

- **Таблица векторов прерываний** (**таблица IDT, размер в IDTR**)- дескрипторы адресов и настроек (шлюз порывания) .
  - —Адрес соответствует подпрограмме прерываний (действиям в ответ на вызов прерывания, напр. обработка нажатия клавиши на клавиатуре).
  - —Прерывания могут быть аппаратные, исключения и программные.
  - —Прерывания находят на 0 уровне привилегий, поэтому обращение к ним через «шлюз».
- **Шлюз вызовов – это область памяти через которую может быть произведен обмен данными между разными процессами** (например, глобальные переменные)
- **TSS (Сегмент состояний задач) –** специальный сегмент, выделены в качестве стека сохранения состояния процессора при переключении на обработку прерываний и исключений.

Модель памяти процессора. Режимы работы x86-x64. Длинный режим

- **long mode («длинный режим», или IA-32e по документации Intel)** – это 64-разрядный режим.
  - По принципу работы он почти полностью сходен с защищённым режимом.
  - В 64-разрядный режим можно перейти только из защищённого режима.
  - Размеры слов - это двойное слово (DWORD), но можно оперировать данными размером в 8 байт (QWORD). Размер адреса всегда 8-байтовый.

Модель памяти процессора. Дополнительные режимы x86-x64

- **Режим системного управления** (System Management Mode) - режим в который процессор переходит при получении специального прерывания SMI.
  - Режим предназначен для выполнения некоторых действий с возможностью их полной изоляции от прикладного программного обеспечения и даже операционной системы. Например использоваться для реализации системы управления энергосбережением компьютера или функций безопасности и контроля доступа.
  - Переход в этот режим возможен только аппаратно.
- **Режим виртуального процессора 8086** – это подрежим защищённого режима для поддержки старых 16-разрядных приложений.
  - Его можно включить для отдельной задачи в многозадачной операционной системе защищённого режима;
- **Режим совместимости для long mode.** В режиме совместимости приложениям доступны 4 Гб памяти и полная поддержка 32-разрядного и 16-разрядного кода;
  - Размер слова - двойное. Размер адреса 32- битный, а размер операнда не может быть 8-байтовым.
  - Режим совместимости можно включить для отдельной задачи в многозадачной 64- битной операционной системе.

Модель памяти процессора. Сравнение работы процессоров в защищенном и длинном режимах

Operating Mode		Operating System Required	Application Recompile Required	Defaults		Register Extensions	Typical
				Address Size (bits)	Operand Size (bits)		GPR Width (bits)
Long Mode	64-Bit Mode	New 64-bit OS	yes	64	32	yes	64
	Compatibility Mode		no	32		no	32
				16	16		16
Legacy Mode	Protected Mode	Legacy 32-bit OS	no	32	32	no	32
				16	16		
	Virtual-8086 Mode			16	16		16
	Real Mode	Legacy 16-bit OS					

Register or Stack	Legacy and Compatibility Modes			64-Bit Mode		
	Name	Number	Size (bits)	Name	Number	Size (bits)
General-Purpose Registers (GPRs)	EAX, EBX, ECX, EDX, EBP, ESI, EDI, ESP	8	32	RAX, RBX, RCX, RDX, RBP, RSI, RDI, RSP, R8–R15	16	64
128-Bit XMM Registers	XMM0–XMM7	8	128	XMM0–XMM15	16	128
64-Bit MMX Registers	MMX0–MMX7	8	64	MMX0–MMX7	8	64
x87 Registers	FPR0–FPR7	8	80	FPR0–FPR7	8	80
Instruction Pointer	EIP	1	32	RIP	1	64
Flags	EFLAGS	1	32	RFLAGS	1	64
Stack	–		16 or 32	–		64

Особенности организации прерываний

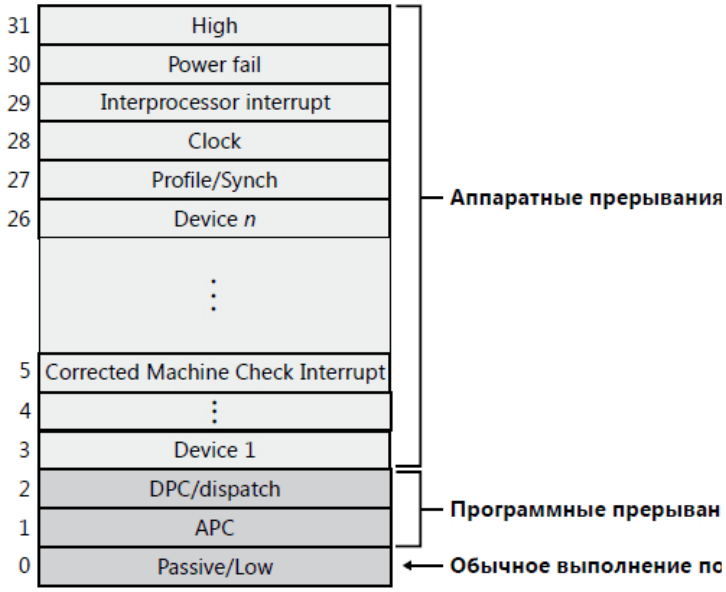
Лекция 3. Модель памяти процессоров

Аппаратные средства телекоммуникационных систем

Модель памяти X86-X64, Прерывания

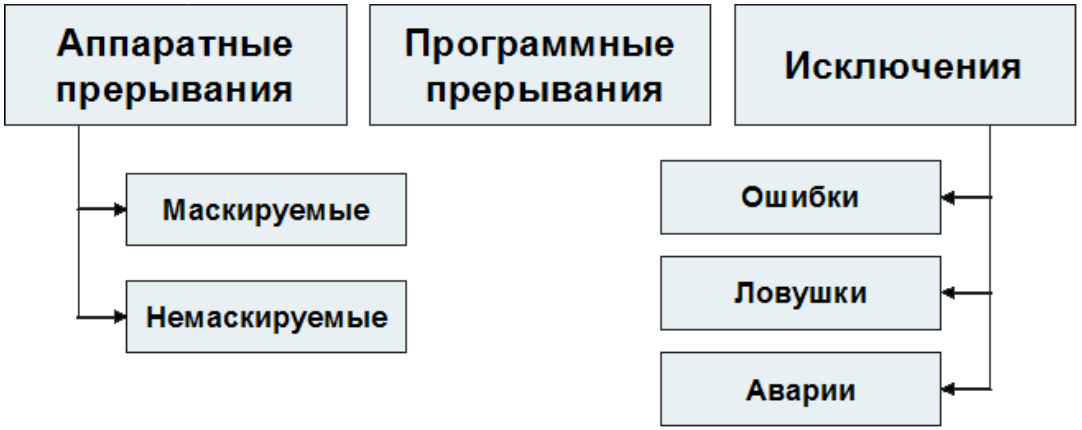
Прерывание (от англ. interrupt) - это *прекращение выполнения текущей команды или текущей последовательности команд для обработки некоторого события специальной программой – обработчиком прерывания, с последующим возвратом к выполнению прерванной программы.*

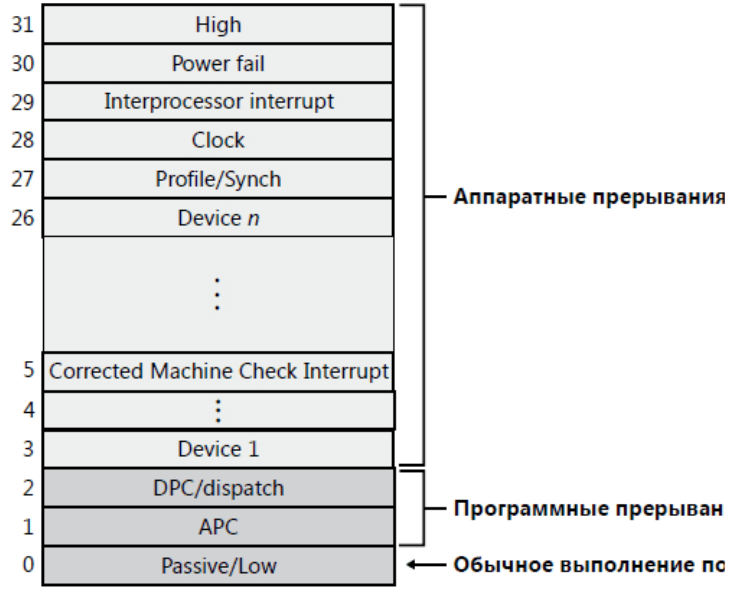
Прерывания имеют наибольший приоритет среди задач процессора.



Модель памяти X86-X64, Прерывания

Прерывание используется для быстрой реакции процессора на особые ситуации, возникающие при выполнении программы и взаимодействии с внешними устройствами.





Модель памяти X86-X64.
Исключения.

Исключения генерируются процессором, когда какая-либо программа пытается нарушать ограничения защиты.

- Повлиять на исключения прикладные программы (работающие даже на нулевом уровне привилегий) не могут, замаскировать – тоже.
- Исключения процессора генерируются вне зависимости от флага IF.

- ***1. Ошибка –*** возникает в ситуации ошибочных действий программы
  - *(подразумевается, что такую ошибку можно исправить).*
  - допускает рестарт команды, которая вызвала исключение после исправления ситуации
  - в стеке обработчика адрес возврата из прерывания указывает на команду, вызвавшую исключение.
  - Примером -исключение отсутствующей страницы-механизм виртуальной памяти.

Модель памяти X86-X64.
Исключения.

Исключения генерируются процессором, когда какая-либо программа пытается нарушать ограничения защиты.

- ***2. Ловушка*** – это исключение, возникающее сразу после выполнения «отлавливаемой» команды.
  - исключение позволяет продолжить выполнение программы со следующей команды.
  - На ловушках строится механизм отладки программ.

- ***3. Авария*** – это исключение, которое не позволяет продолжить выполнение прерванной программы и сигнализирует о серьёзных нарушениях целостности системы.
  - Пример - исключение двойного нарушения (прерывание 8), когда сама попытка обработки одного исключения вызывает другое исключение.

Модель памяти X86-X64.
Исключения.

Исключения – прерывания 0-31

Примеры исключений

Номер вектора	Название	Описание	Тип	Код ошибки	Источник
0	#DE	Ошибка деления	Ошибка	Нет	Команды DIV и IDIV
1	#DB	Отладка	Ошибка или ловушка	Нет	Любая команда или команда INT 1
2	–	Прерывание NMI	Прерывание		Немаскируемое внешнее прерывание
3	#BP	Точка останова	Ловушка	Нет	Команда Int3
4	#OF	Переполнение	Ловушка	Нет	Команда INTO
5	#BR	Превышение предела	Ошибка	Нет	Команда BOUND
6	#UD	Недопустимая команда	Ошибка	Нет	Недопустимая команда или команда UD2
7	#NM	Устройство недоступно	Ошибка	Нет	Команды плавающей точки или команда WAIT/FWAIT
8	#DF	Двойная ошибка	Авария	Всегда ноль	Любая команда

прерывание 2 является единственным немаскируемым прерыванием, возникает, напр. при сбое чтения памяти.

Модель памяти X86-X64. Прерывания. Программные прерывания.

- Прерывания можно генерировать программно командами INT.
  - Команда INT3 генерирует системное прерывание, которое вызывает прерывание 3.
  - Например.
    - Int16h – обработчик ввода-вывода клавиатуры,
    - Int3h – точка остановки.
    - Int17h – принтер
    - Int11h – список оборудования.
    - Int4h - переполнение
  - Команда INTO вызывает прерывание 4, если установлен флаг OF.
  - *Программные прерывания – это наиболее простой из методов вызова привилегированного кода для программ, которые работают на нулевом уровне привилегий.*
  - **В современных операционных системах программные прерывания запрещены для пользователя – доступ к предоставляемому ими функционалу осуществляется через драйвера и операционную систему.**

Прерывания. Аппаратные прерывания

Аппаратные прерывания (англ. hardware interrput) – это сигнал от любого устройства системы для процессора, который по этому сигналу должен обслужить данное устройство.

- **Маскируемые аппаратные прерывания**
  – можно запрещать в соответствующих регистрах (бит IF).
- **Немаскируемые аппаратные прерывания –**
  - обязательные к исполнению (напр. Обработка ошибки)

Аппаратные прерывания имеют свой уровень приоритета

В случае нескольких прерываний с одним уровнем приоритета Они будут поставлены в очередь.