© 2013 МГУ/ВМК/СП

Лекция 22

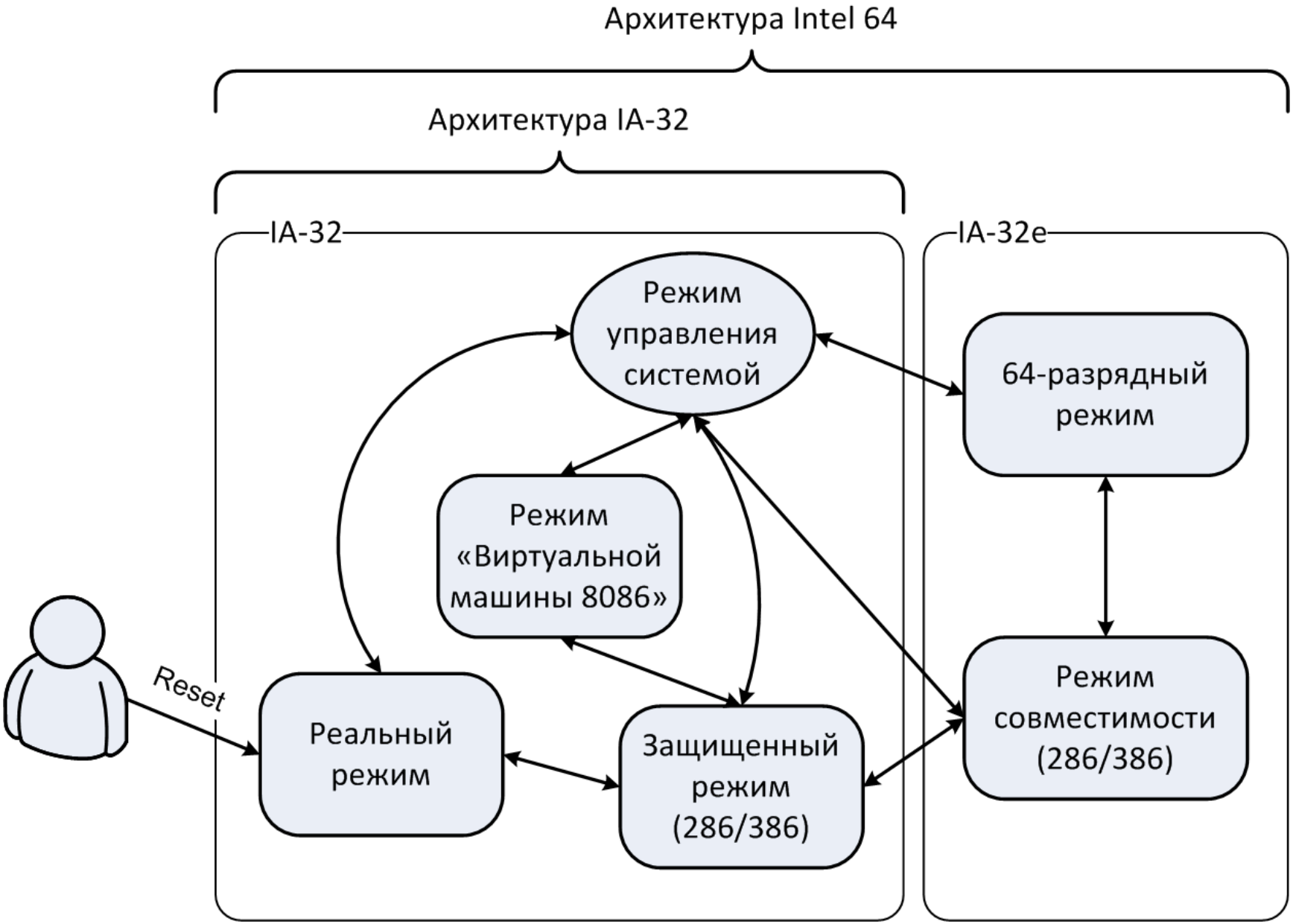
24 апреля

© 2013 МГУ/ВМК/СП

История развития x86

- 4004 – ноябрь 1971. 4-битный микропроцессор. Первый в мире коммерчески доступный однокристальный микропроцессор.
- 8008 – апрель 1972. 8080 – апрель 1974. 8-битные процессоры.
- 8086 – 1978. Размер слова – 16 бит, ширина адресной шины – 20 бит. Адреса вычисляются с использованием сегментных регистров.
- 80186 – 1982. Добавлено несколько новых инструкций.
- 80286 – 1982. Ширина адресной шины – 24 бита, добавлено устройство контроля памяти – MMU. Процессор мог переключаться между двумя режимами – реальным и защищенным.
- 80386 – октябрь 1985. Снят с производства в 2007. Размер слова – 32-разряда. Процессор мог переключаться между тремя режимами – реальным, защищенным, виртуальным. Адресуемая память – 4 ГБ.
- ...
- Intel Xeon E7-8870 - 2011. 2.40 GHz; 2.2×10^9 транзисторов; системная шина QPI: 6.4 ГТ/с; обращение к памяти: 102 ГБ/с; Адресуемая память – 16 ТБ; Кэш L1: 64КБ L2: 256КБ L3: 30МБ

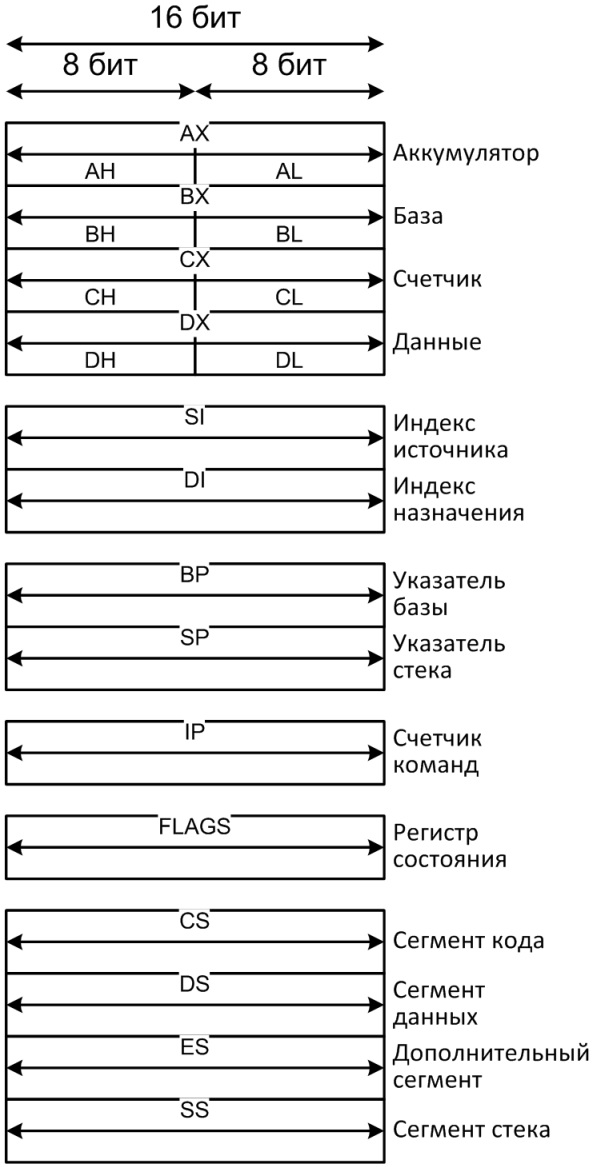
© 2013 МГУ/ВМК/СП



© 2013 МГУ/ВМК/СП

Реальный режим / 8086

- Среда, в которой работает **одна** программа
  - Эта программа управляет всеми ресурсами
- Машинное слово 16 разрядов, адрес 20 разрядов
  - Сегментные регистры
  - Эффективный адрес = (сег_рег) << 4 + смещение
- Доступна вся память
- Периферийные устройства управляются через порты ввода/вывода



© 2013 МГУ/ВМК/СП

Обратная совместимость

- Обратная совместимость – свойство семейства процессоров. На более новом компьютере могут выполняться программы, рассчитанные на более ранние модели.
  - Прямая совместимость
- Линия A20. 8086 → 80286.
  - Контроллер клавиатуры Intel 8042

```
; открываем адресную линию A20
in   al, 92h
or   al, 2
out  92h, al
```

© 2013 МГУ/ВМК/СП

Многозадачная работа компьютера

- Привилегированный режим
  - Разделение машинных команд на две категории
- Механизм защиты памяти
- Таймер
- Механизм прерываний
  - Приоритеты прерываний
    - Одновременно произошедшие прерывания
    - Вложенные прерывания

© 2013 МГУ/ВМК/СП

Защищенный режим / 80386

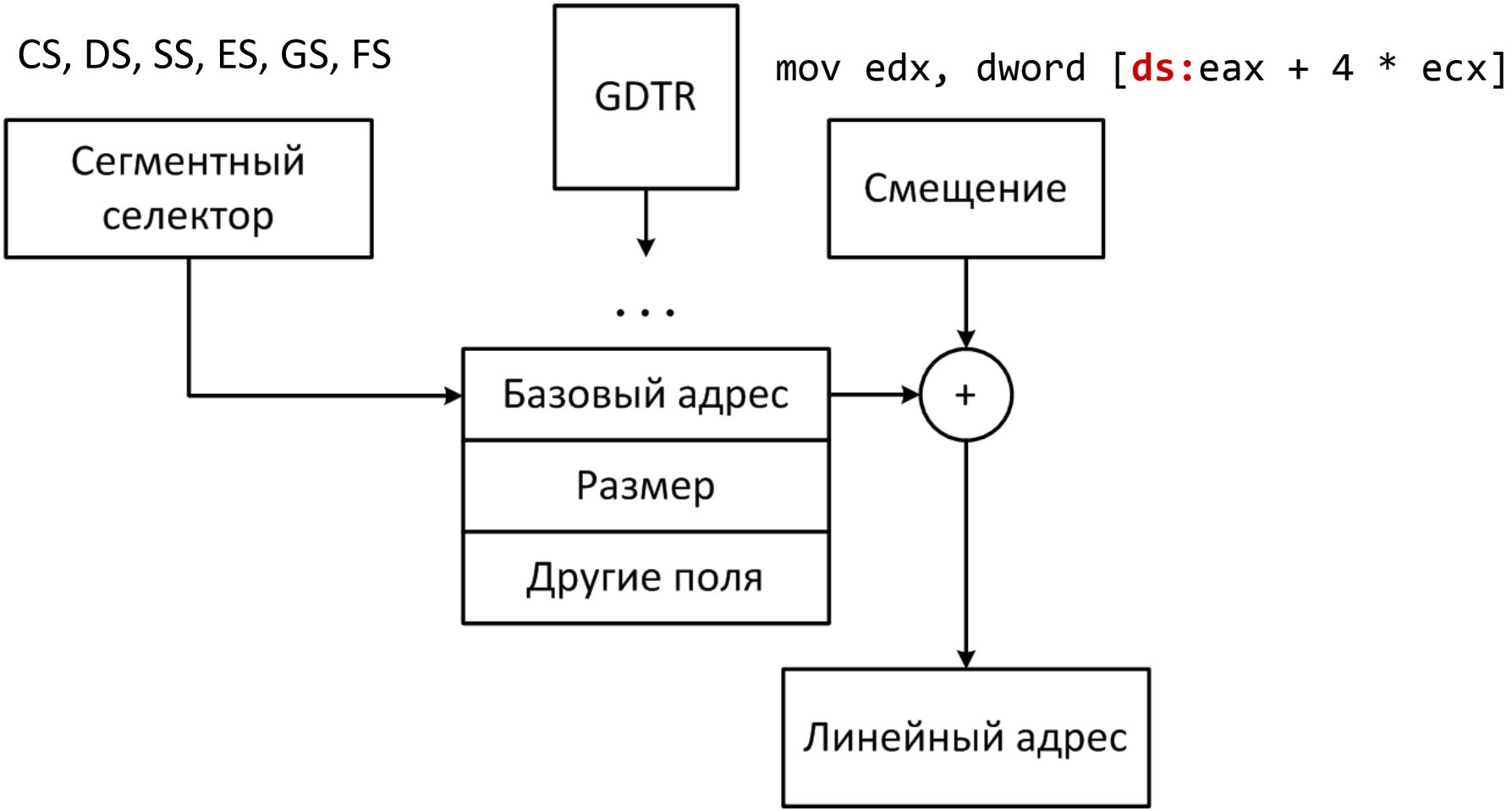
- Машинное слово и адрес 32 бита
- Два дополнительных сегментных регистра gs и fs
- Регистры управления
  - CR0, CR2, CR3
- Аппаратная защита памяти
  - Базы таблиц дескрипторов GDTR, LDTR, IDTR
  - Мультисегментная модель памяти
    - Фактически не используется
  - Многоуровневая трансляция адреса памяти
- Аппаратное переключение задач
  - фактически не используется

© 2013 МГУ/ВМК/СП

Сегментные селектор и дескриптор

© 2013 МГУ/ВМК/СП

Адресация в защищенном режиме



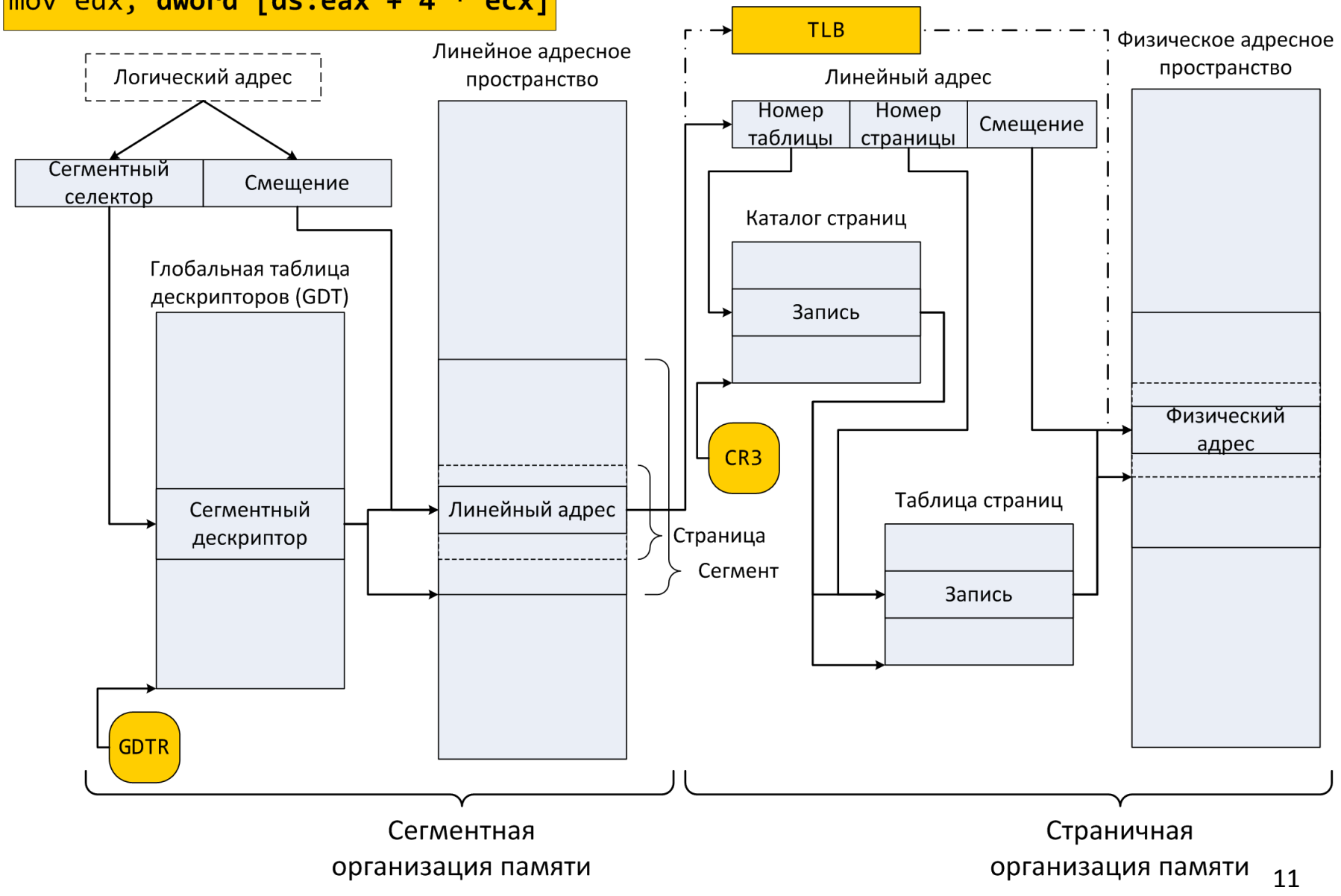
© 2013 МГУ/ВМК/СП

Страничная организация памяти

- Нехватка физической памяти
- Одновременная работа нескольких программ.
- Линейные адреса автоматически преобразуются в физические адреса.
  - Память разделена на фрагменты-страницы одинакового размера.
  - Старшие биты адреса меняются на биты, взятые из соответствующей записи таблицы страниц
- Отображение адресов выполняет MMU

© 2013 МГУ/ВМК/СП

```
mov edx, dword [ds:eax + 4 * ecx]
```



© 2013 МГУ/ВМК/СП

Уровни защиты

- Каждый уровень привилегий обладает своим контекстом
- Стек вызовов – часть контекста
- Переход между уровнями контролируется полями PL в дескрипторах

© 2013 МГУ/ВМК/СП

Обращение к памяти: итоги

Канонический адрес

- Вычисляем смещение / адресный код в машинной команде
- Вычисляем линейный адрес / сегменты

Линейный адрес

- Пересчитываем линейный адрес в физический
  - Если есть запись в TLB берем готовый адрес
  - Если в TLB ничего нет, вычисляем физический адрес, проходя по таблицам трансляции

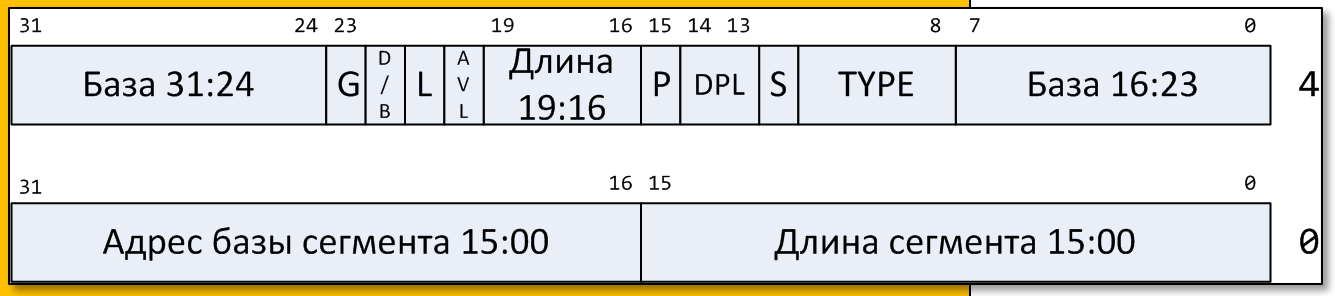
Физический адрес

- Обращаемся в память за данными
  - Берем из кэша, если данные в нем присутствуют

© 2013 МГУ/ВМК/СП

```
gdt:
gdt_null:
   dd   0, 0              ; Нулевой дескриптор
gdt_code:
   dw   0xFFFF   ; Сегмент кода
   dw   0
   db   0
   db   10011010b
   db   11001111b
   db   0
gdt_data:
   dw   0xFFFF   ; Сегмент данных
   dw   0
   db   0
   db   10010010b
   db   11001111b
   db   0
gdt_end:

; Значение, которое мы загрузим в GDTR:
gd_reg:
   dw   gdt_end - gdt - 1
   dd   gdt
```



© 2013 МГУ/ВМК/СП

```
[BITS 16]
[ORG 0x0700] ; начинаем располагать код с этого адреса

;инициализируем сегментные регистры
   cli
   mov  ax, 0
   mov  ds, ax
   mov  es, ax
   mov  ss, ax
   mov  sp, 0x700
   sti

; Конфигурируем контроллер прерываний
...

   call enableA20

; Загрузка регистра GDTR:
   xor  ax, ax
   mov  ds, ax

   lgdt [gd_reg]
```

```
   ; Установка бита PE регистра CR0
   mov  eax, cr0
   or   al, 1
   mov  cr0, eax

   ; Переходим в защищенный режим, в 32-бита
   jmp  0x08:protected
```