# Семинар 1 Защищённый режим (2 лаба)

## Режимы

Компьютере на базе процессора intel работают в трёх режимах

1. Реальный:  
   Это 16-разрядный режим с 20 разрядной шиной адреса.  
   В этом режиме работали процессору 8086.  
   2^20 – позволяет адресовать не более 1ого Мб (1024 Кб).  
   Работали эти компьютеры под DOS (дисковая операционная система)  
   DOS – однозадачная ОС (в один момент времени, в памяти могла находится только одна программа, и выполнялась она с начала до конца)
2. Защищённый (Protected)  
   32-разрядный режим  
   процессоры 3086  
   32ух разрядные регистры, 32ух разрядная шина адреса  
   Эти процессоры поддерживали две независимые… схемы… памяти…
   1. …Виртуально По запросу (виртуально – означает что фактически такой памяти нет, она поддерживается аппаратно – существуют аппартаные схемы которые поддерживают это память)
   2. …По запросу
   3. Сегментирован по запросу (взять лучше от 2ух предыдущих)

В защищённом режиме есть специальный режим V86(virtual) **–** (в нём...) **Как задачи в этом режиме** запускаются операционные системы реального режима.  
Фактически запускается виртуальная машина и в этой среде может выполнятся одна программа реального режима. Intel8086

Этот режим многозадачный, но каждая такая запущенная виртуальная машина является виртуальной машинной 86ой, со всем вытекающими отсюда последствиями (1 программа, 1Мб памяти, операторы 16-разрядные)  
Почему режим называется защищённый (потому что microsfot наглые)  
Винды являются персональным разделением данных. Вся теория операционных систем уже существовала к 4ому поколению. **Адресные пространства процесса должны быть защищены!**Естественно адресные процессы ОС тоже защищены (как только появились многопроцессорные (многопроцессорность))

1. Длинный (режим long)  
   64-разрядные регистры и 64-разрядные операнды  
   Многопроцессорность  
   Виртуальная память только страничная

В этом режиме поддерживается специальный режим (…), в котором могут выполняться 32ух разрядные процессы.

Принято делить регистры на группы (см. тетрадь):

1. РОНы (Регистры общего назначения)
2. Индексные регистры и регистры-указатели
3. Сегментные
4. Регистры системных адресов
5. Управляющие
6. PAE?
7. EIP
8. EFLAGS

(Минимально адресованной ед. памяти является 1 байт.)

Перейдём к GDTR и …

Во 2ой лабе мы пишем управляющую программу.

1ое что делает эта программа – переводит из защ. режима в реал.  
Необходимо написать 2 обработчика прерываний.

После этого, программа должна вернуть в защ. Режим

Можно разбить на 2 шага:

1. Написать программу, которая переводит из защ. в реал. и обратно (Рудаков-Финогеннов – костыль, будем расширять эту программу, надо найти неправильное и исправить)
2. Что-то?

Процессор может выполнять только те программы, которые находятся в памяти.

Операционная система лежит с определённого адреса в таблице (когда комп. выключен).

Минимум действий, для загрузки ОС в память, но этого мало. Необходимо обеспечить адресацию. Все команды выполняются последовательно.

Таблица глобальных дескрипторов содержит дескрипторы сегментов физической памяти.

В SMP архитектуре равноправные процессоры, которые работают с общей памятью. Один процессор назначается главным только для того, чтобы он выполнял обработчик прерываний от системного таймера (это может делать только один).

В системе JDT память будет одна, и таблица глобальных дескрипторов тоже будет одна.

Эта таблица должна находится в физической памяти, причём в памяти ядра системы (см. тетрадь)

В защищённом режиме доступно 4 Гб

Локальные дескр. таблицы описывают виртуальные адресные пространства процессов – каждая такая таблица явл. системной. Будет занимать сегмент. Сегменты физ. памяти описывается…???

LDTR – является селектором к …???

В защищённом режиме используется модель памяти FLAT.

*В реальном режиме – таблица векторов прерываний, а в защищённом режиме – таблица дескрипторов прерываний (вектор — это непосредственно сам адрес прерывания, а дескриптор — это другая структура, позволяющая получить адрес)*

DOS не защищён, но есть уровень привилегий.

Сохранение аппаратного контекста (pusha – сохр. все регистры)

Косвенный вызов – мощны механизм адресации (например, делали меню)  
