# Программирование

Базовая архитектура IBM PC X86

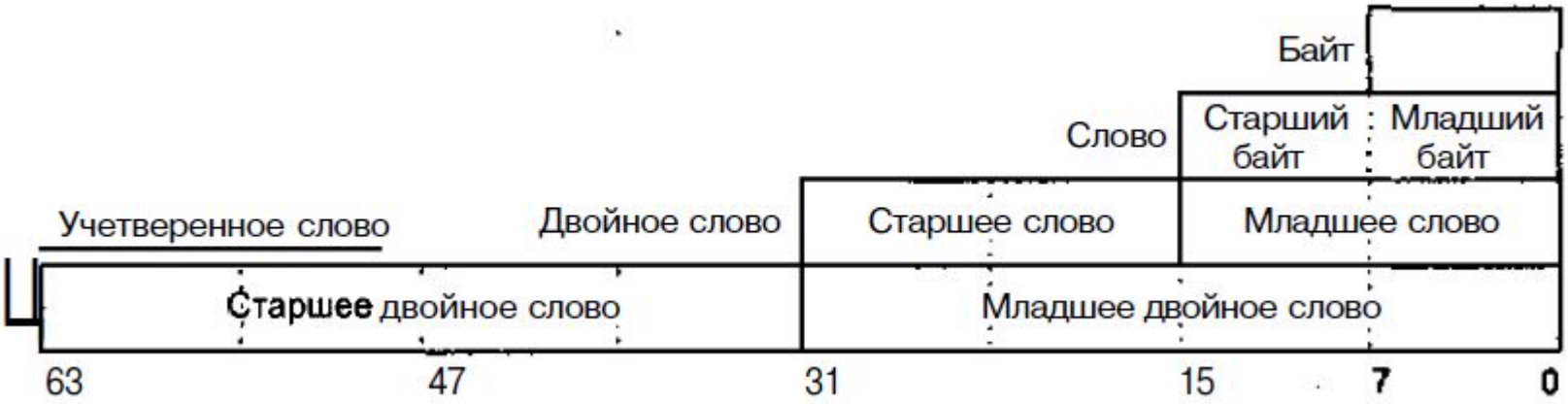
# Архитектура ВМ

**Архитектура – это функциональная организация ВМ без физической реализации.**

1. **Представление данных и их связь с назначением, организация хранения и способы кодирования.**
2. **Адресация – способ определения адреса операнда по информации в адресной части команды.**
3. **Структура команд: части команд и взаимодействие частей.**
4. **Система команд – перечень команд.**
5. **Организация вычислительного процесса.**
6. **Организация ввода-вывода.**
7. **Система прерываний.**

# Базовая архитектура IBM PC X86

**Представление данных. С точки зрения размерности**




- Процессоры Intel имеют важную особенность — младший байт всегда хранится по меньшему адресу.

# Базовая архитектура IBM PC X86

**Представление данных. С точки зрения логической интерпретации**

*Целый тип со знаком* (в дополнительном коде)

- 8-разрядное целое— от-128 до+127;
- 16-разрядное целое — от -32 768 до +32 767;
- 32-разрядное целое — от -2^31 до +2^31 - 1.

*Целый тип без знака*

- байт — от 0 до 255;
- слово — от 0 до 65 535;
- двойное слово— от 0 до 2^32 - 1.

*Указатель на память*

- *ближний тип* — 32-разрядный логический адрес
- *дальний тип* — 48-разрядный логический адрес

# Базовая архитектура IBM PC X86

**Представление данных. С точки зрения логической интерпретации**

- *Цепочка* представляет собой некоторый непрерывный набор байтов, слов или двойных слов максимальной длиной до 4 Гбайт.
- *Битовое поле.* Каждый бит является независимым и может рассматриваться как отдельная переменная. Битовое поле может начинаться с любого бита любого байта и содержать до 32 битов.
- *Неупакованный двоично-десятичный тип* — байтовое представление десятичной цифры от 0 до 9. По одной цифре в каждом бай
- *Упакованный двоично-десятичный тип.* Хранит две десятичных цифр от 0 до 9 в одном байте. Каждая цифра хранится в своем полубайте.

# Базовая архитектура IBM PC X86

**Представление данных. С точки зрения логической интерпретации**

- *Типы данных с плавающей точкой.* Сопроцессор имеет несколько собственных типов данных, несовместимых с типами данных целочисленного устройства.
- *Типы данных MMX-расширения Pentium MMX/II/III/IV.* Данный тип данных появился в процессоре Pentium MMX.
- *Типы данных MMX-расширения Pentium III/IV.* Этот тип данных появился в процессоре Pentium III.

# Базовая архитектура IBM PC X86

**Организация вычислительного процесса**

Вычислительный процесс организован в полном соответствии с принципами фон Неймана.

Для ускорения введено понятие конвейера команд, из которого извлекается следующая команда.

# Базовая архитектура IBM PC X86

## Регистры процессора: регистр флагов

EFLAGS

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ID | VIP | VIF | AC | VM | RF | 0 | NT | IOPL | OF | DF | IF | TF | SF | ZF | 0 | AF | 0 | PF | 1 | CF |

RFLAGS

| 63 … 32 | 31 … 0 |
|---|---|
| 000...(ЗАРЕЗЕРВИРОВАНО) | EFLAGS |

FLAGS

| 15 | 14 | 13 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | NT | IOPL | OF | DF | IF | TF | SF | ZF | 0 | AF | 0 | PF | 1 | CF |

В регистре хранятся данные о состоянии процессора и результатах выполнения некоторых команд.

# Базовая архитектура IBM PC X86

**Регистры процессора: регистр флагов**

FLAGS

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | NT | IOPL | | OF | DF | IF | TF | SF | ZF | 0 | AF | 0 | PF | 1 | CF |

C – carry flag (флаг переноса) – выполнение операции привело к возникновению переноса

P – parity flag (флаг четности) – количество единиц в младшем байте результата чётно

A – auxiliary carry flag (флаг дополнительного переноса) – используется при операциях с двоично-десятичными числами

Z – zero flag (флаг нуля) – результатом операции был ноль

# Базовая архитектура IBM PC X86

**Регистры процессора: регистр флагов**

FLAGS

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | NT | IOPL | | OF | DF | IF | TF | SF | ZF | 0 | AF | 0 | PF | 1 | CF |

S – sign flag (флаг знака) – старший разряд результата имеет значение «1»

T – trap flag (флаг трассировки) – используется программами-отладчиками

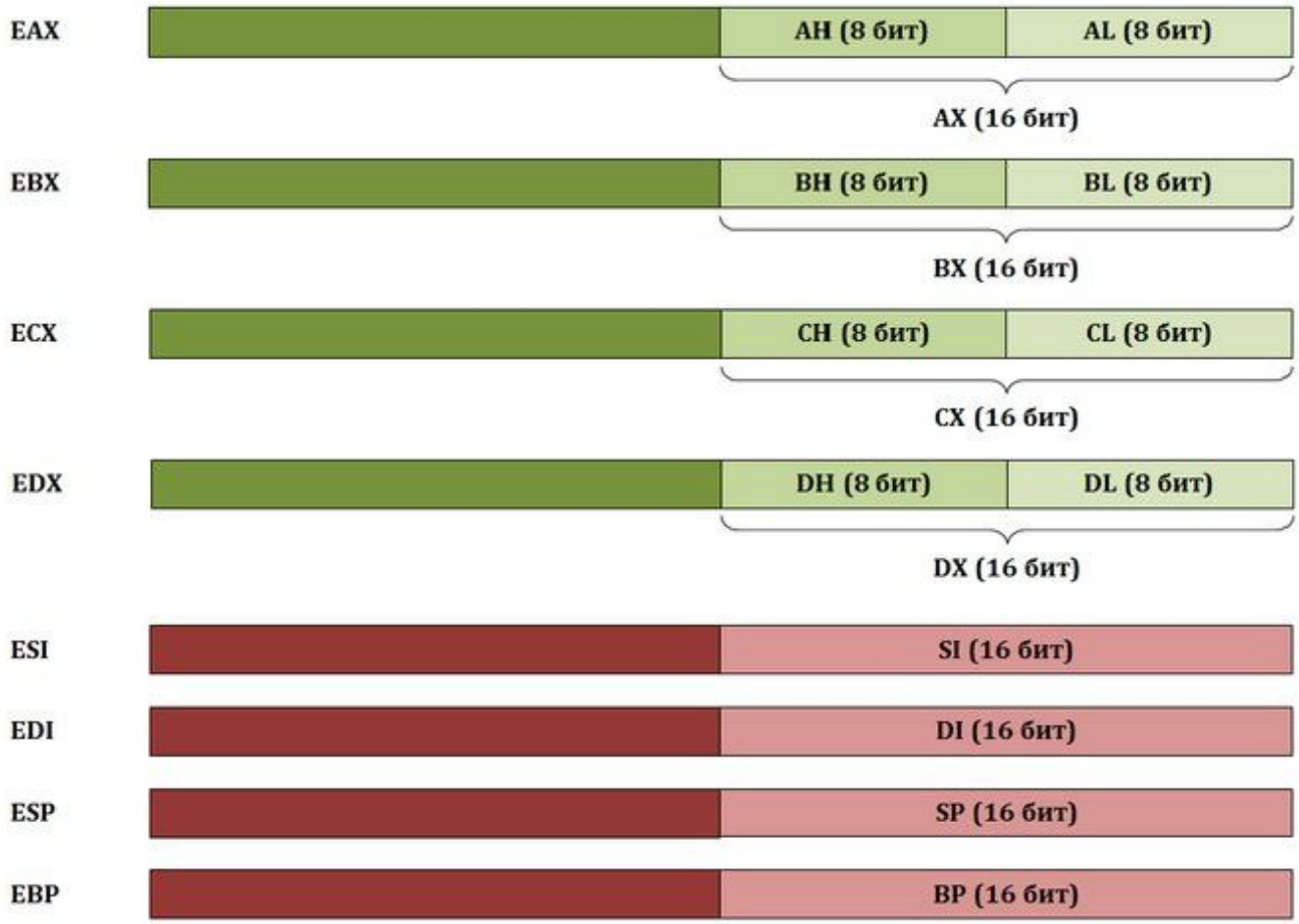
I – interrupt flag (флаг прерывания) – процессор реагирует на прерывания

D – direction flag (флаг направления) – используется командами обработки строк

O – overflow flag (флаг переполнения) – устанавливается при переполнении (результат операции не помещается в регистре

# Базовая архитектура IBM PC X86

## Регистры процессора: РОНы



# Базовая архитектура IBM PC X86

**Регистры процессора: РОНы**

Регистр AX (accumulator, аккумулятор)

Это регистр-накопитель. Наиболее эффективно его использование в арифметических и логических операциях, а также в операциях пересылки, т.к. именно эти операции оптимизированы для использования регистра AX и, как правило, обладают более высоким быстродействием.

# Базовая архитектура IBM PC X86

**Регистры процессора: РОНы**

Регистр BX (base, базовый регистр)

В некоторых операциях этот регистр используется для реализации расширенной адресации.

# Базовая архитектура IBM PC X86

**Регистры процессора: РОНы**

Регистр CX (counter, счётчик)

Обычно этот регистр используется как счётчик, указывающий количество выполнений команды или группы команд (циклические вычисления, сдвиги).

# Базовая архитектура IBM PC X86

**Регистры процессора: РОНы**

Регистр DX (data, регистр данных)

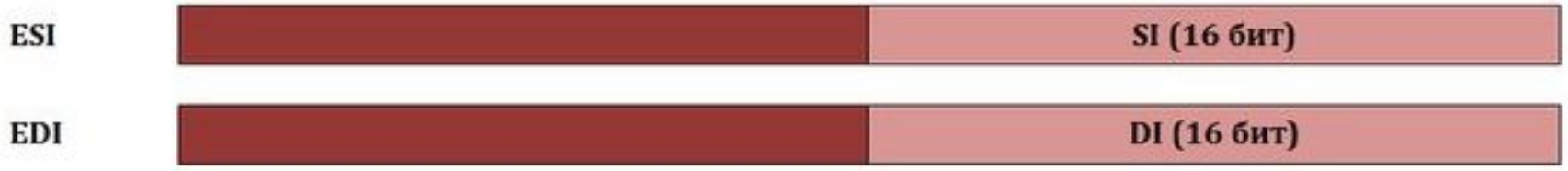
Этот регистр используется в операциях умножения и деления, а также является единственным регистром, в котором может быть указан адрес порта в командах ввода-вывода.

# Базовая архитектура IBM PC X86

**Регистры процессора: РОНы**

Регистры SI, DI

Индексные регистры источника (SI, source index) и приёмника (DI, destination index), содержащие смещения относительно некоторого базового адреса. Обычно используются для выполнения операций над массивами данных.



# Базовая архитектура IBM PC X86

**Регистры процессора: РОНы**

Регистр BP

Базовый регистр, в котором содержится смещение относительно начала сегмента, в качестве которого по умолчанию предполагается сегмент стека. Обычно используется при организации вычислений в стековых структурах.

# Базовая архитектура IBM PC X86

**Регистры процессора: РОНы**

Регистр SP (stack pointer, указатель стека)

В SP содержится смещение относительно начала сегмента стека. При операциях со стеком система сама следит за изменениями содержимого SP в соответствии с выполняемыми операциями. В SP содержится адрес младшего байта данных, который был послан в стек последним.

# Базовая архитектура IBM PC X86

**Регистры процессора: РОНы**

Регистр IP (instruction pointer, счётчик команд)

Регистр содержит адрес команды, следующей за выполняемой в текущий момент, в сегменте памяти, который задаётся регистром CS.

# Базовая архитектура IBM PC X86

**Регистры процессора: сегментные регистры**

CS (code segment) – указывает на сегмент, в котором содержатся команды программы (начальный адрес сегмента кода). Адрес команды – CS:IP.

DS (data segment) – адресует начало сегмента данных.

ES (extra segment) – указывает на дополнительный сегмент данных; используется обычно при строковых операциях при формировании адреса приёмника данных.

SS (stack segment) – адресует сегмент стека.

# Базовая архитектура IBM PC X86

**Адресация**

Разрядность шины адреса: 20 разрядов

Максимальный адрес: $2^{20}$

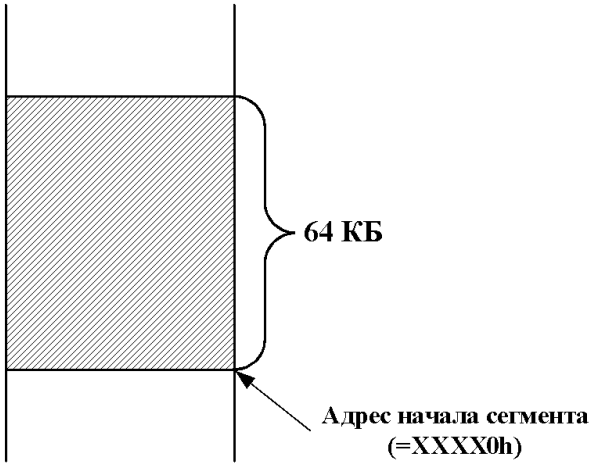
$2^{20}$ байтов = 1 Мбайт

Адрес задаётся с помощью двух 16-разрядных значений – сегмента и смещения – и вычисляется по формуле

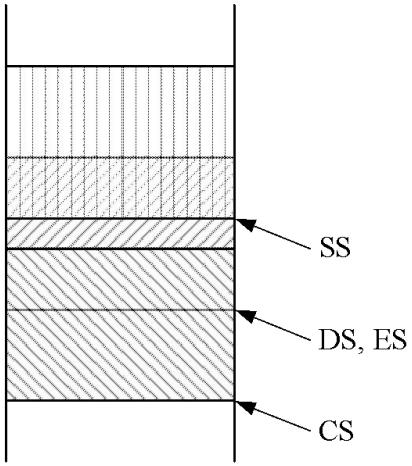
<целевой адрес> = <сегмент>*16 + смещение

# Базовая архитектура IBM PC X86

## Сегменты



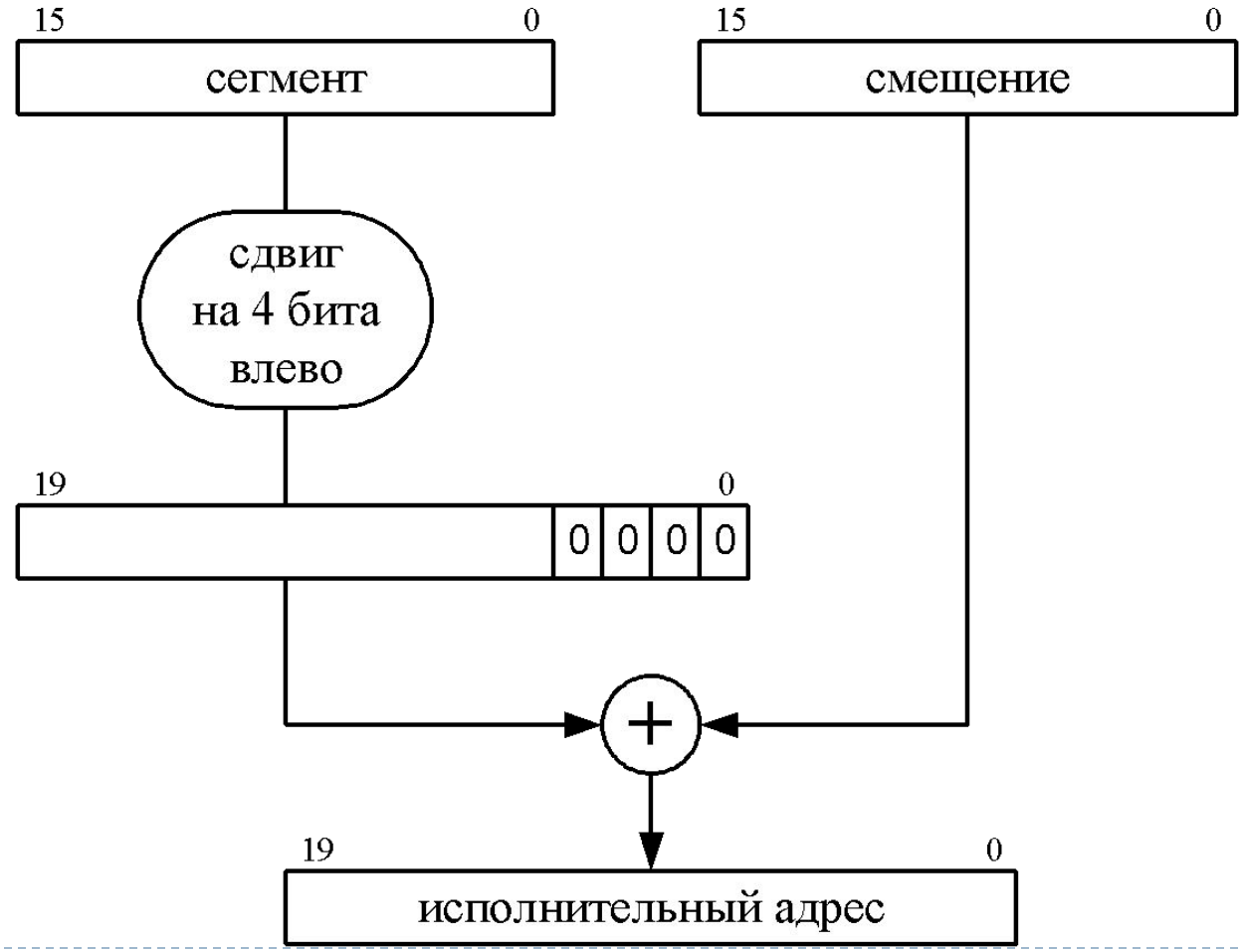

При работе с данными в пределах сегмента изменяется только смещение, адрес начала сегмента не меняется.




Сегменты в памяти могут перекрываться.

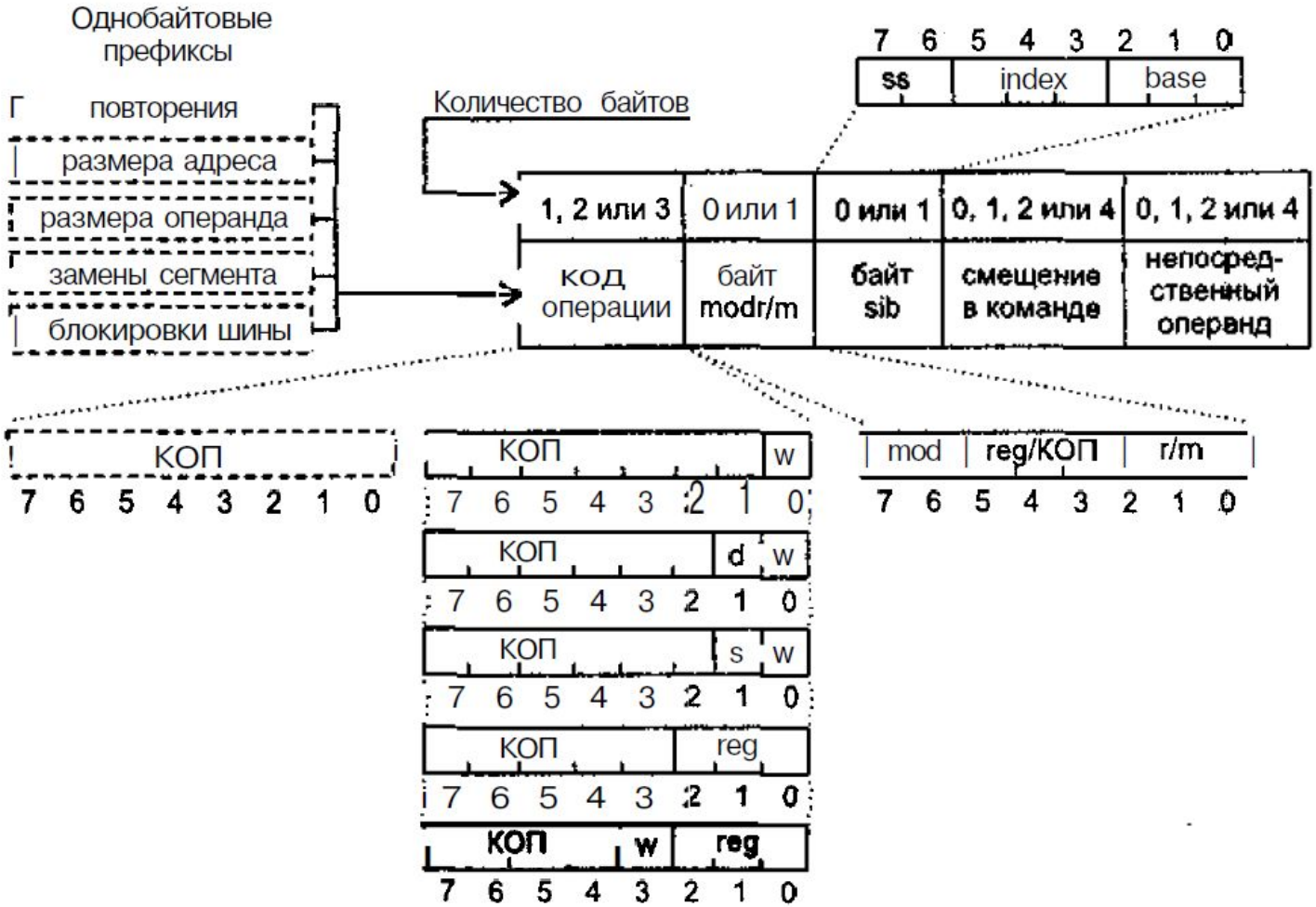
# Базовая архитектура IBM PC X86

**Сегменты: адресация**



# Базовая архитектура IBM PC X86

## Структура команд



# Базовая архитектура IBM PC X86

**Структура команд**

- *Префиксы* — необязательные однобайтные элементы машинной команды. Назначение префиксов — изменить действия, выполняемые командой. Машинная команда может иметь до четырех префиксов одновременно. Порядок их следования при этом может быть любым.
- *Код операции* — обязательный элемент, описывающий операцию, выполняемую командой. Код операции может занимать от одного до трех байт. Для некоторых машинных команд часть битов кода операции может находиться в байте mod r/m.

# Базовая архитектура IBM PC X86

**Структура команд**

- *Байт режима адресации* mod r/m, иногда называемый *постбайтом,* несет информацию об операндах и режиме адресации. Если операнд находится в памяти, то байт mod r/m определяет компоненты (смещение, базовый и индексный регистры), используемые для вычисления его эффективного адреса.
- *Байт масштаба, индекса и базы* (Scale-Index-Base — sib) используется для расширения возможностей адресации операндов. На наличие байта sib в машинной команде указывает сочетание одного из значений 01 или 10 поля mod и значения поля r/m = 100.

# Базовая архитектура IBM PC X86

**Структура команд**

- *Поле смещения в команде* — это 8-, 16- или 32-разрядное целое число со знаком, представляющее собой полностью или частично значение эффективного адреса операнда.
- *Поле непосредственного операнда* — необязательное поле, представляющее собой 8-, 16- или 32-разрядный непосредственный операнд.

# Базовая архитектура IBM PC X86

**Система команд**

1. Команды пересылки:

   а) между регистрами и памятью;

   б) между регистрами и устройствами ввода-вывода.
2. Команды управления.
3. Арифметические и логические команды.
4. Команды манипулирования битами.
5. Команды для обработки строк.
6. Команды для поддержки механизма прерываний.
7. Команды изменения состояния процессора.

# Базовая архитектура IBM PC X86

**Система прерываний**

Предусмотрены прерывания аппаратные и программные.

Всего в системе может быть до 255 прерываний.

Для реализации механизма прерываний выделен 1 КБайт оперативной памяти.

# Базовая архитектура IBM PC X86

**Организация ввода-вывода**

В языках ассемблера отсутствуют готовые процедуры ввода-вывода. Для выполнения этих операций существуют следующие варианты:

- 1. Напрямую обращаться к устройствам ввода-вывода. Этот способ является единственным в случае программирования «голой» машины, т. е. когда полностью отсутствуют готовые процедуры для работы с внешними устройствами. Для реализации первого способа в системе выделено 4 КБайта ОП.
- 2. Использование процедур BIOS, размещенных в ПЗУ и соответственно постоянно присутствующих в ПК (обращение к функциям BIOS). Этот способ используется при программировании в отсутствие операционной системы.
- 3. Обращение к сервисам ОС с запросами на ввод-вывод для соответствующих устройств. Этот вариант является наиболее предпочтительным и часто используемым.