Тема 2

Модели памяти персонального компьютера

Оперативная память

Представляет собой совокупность 8-битных блоков – *байтов*, каждый из которых имеет свой уникальный *физический* (или *линейный*) адрес – целое неотрицательное число. Максимальное значение физического адреса ограничено размерностью (или шириной) шины адреса.

Механизм управления памятью полностью аппаратный.

Программа не задает физический адрес, который передается на шину адреса и используется для доступа к оперативной памяти. Вместо этого она манипулирует с составными частями адреса, которые преобразуются процессором в физический адрес.

Модели организации оперативной памяти

- Сегментная организация памяти:

  программе выделяются непрерывные участки памяти – сегменты, и она может работать с памятью только в пределах выделенных сегментов.

- Страничная организация памяти:

  память рассматривается как совокупность блоков одинакового размера – страниц (4 Кбайта).
  Все страницы, в отличие от сегментов, имеют одинаковые размеры, а разбиение виртуального адресного пространства процесса на страницы выполняется системой автоматически.
  Страничную организацию памяти можно считать надстройкой над сегментной моделью.

Суть сегментации

Сегментация – это механизм, обеспечивающий существование нескольких адресных пространств как в рамках одной задачи, так и в рамках параллельно выполняющихся процессов.

Программа использует в качестве адреса смещение относительно начала одного из предопределенных сегментов. Это смещение иногда называют *эффективным* адресом.

Адреса начала сегмента хранятся в реальном режиме в сегментных регистрах, а в защищенном режиме – в специальных дескрипторах сегмента. В последнем случае сегментные регистры процессора используются для доступа к таблице этих дескрипторов.

Преимущества сегментной модели

- компактность хранения адреса в машинной команде;
- гибкость механизма адресации;
- защиту адресных пространств задач в многозадачной системе.

Типы реализации сегментной организации памяти

- сегментная модель реального режима;
- сегментная модель защищенного режима;
- плоская модель.

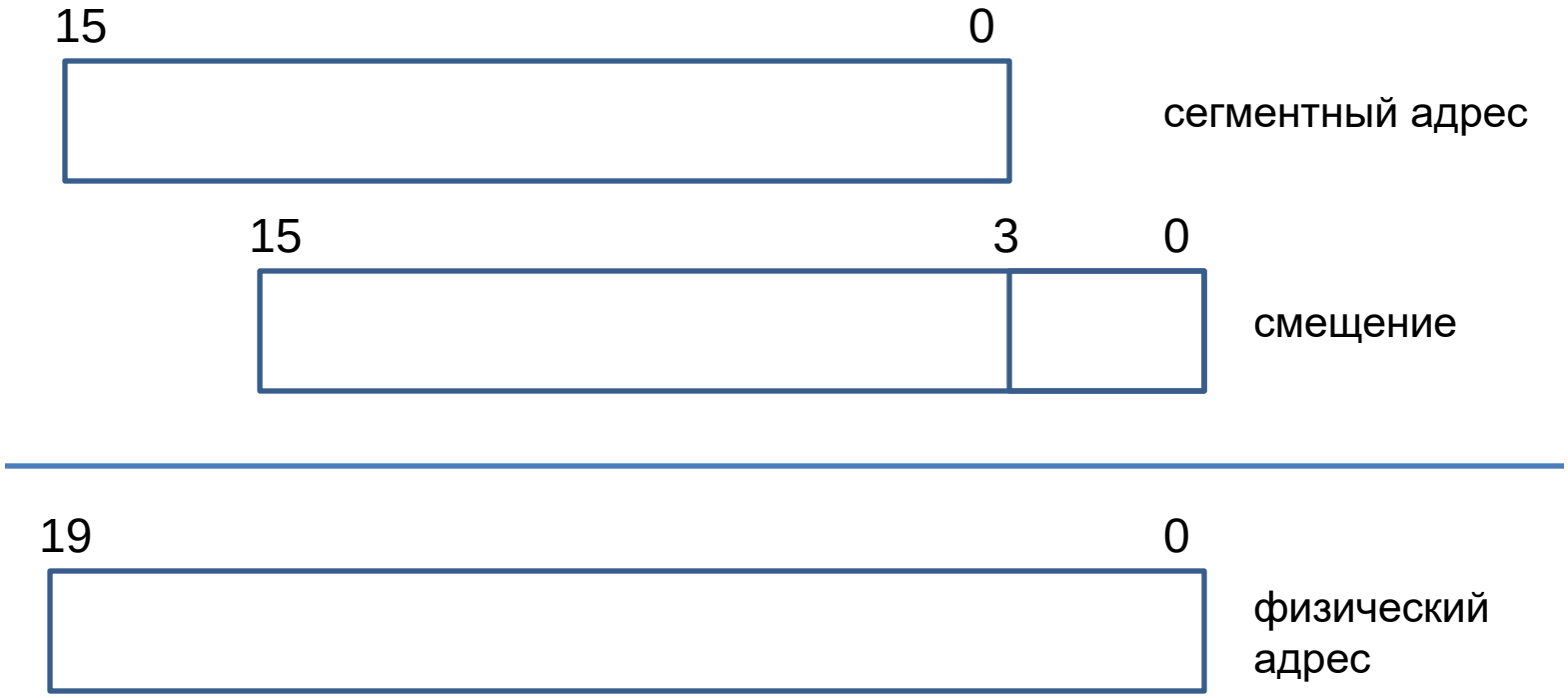
Сегментная модель реального режима

- Размер сегмента не превышает 64 Кбайт;
- Сегменты могут полностью или частично перекрываться;
- Сегменты начинаются с адреса, кратного 16;
- Адрес начала сегмента, деленный на 16, хранится в одном из сегментных регистров **(DS, CS, SS, ES);**
- Смещение имеет длину в 16 бит и может храниться в регистре или памяти.

Схема формирования физического адреса в реальном режиме работы

Формула для формирования физического адреса:

*физический адрес = 16 * сегментный адрес + смещение*



Максимальный адрес равен *10FFFFh*

Способы записи адреса в реальном режиме

сегмент : смещение

- **2145h:0003h**

  адрес начала сегмента – **2145h**, смещение – **0003h**, физический адрес – **21453h**
- **DS:0010h**

  адрес начала сегмента хранится в регистре **DS**, смещение – **0010h**
- **DS:BX**

  адрес начала сегмента хранится в регистре **DS**, смещение – в регистре **BX**

Типы сегментов

- ***Сегмент кода (code segment)***

  В этом сегменте хранятся все выполняемые команды программы

- ***Сегменты данных (data segment)***

  Эти сегменты предназначен для хранения констант и переменных, к которым нужно обеспечить доступ всем процедурам программы

- ***Сегмент стека (stack segment)***

  предназначен для хранения параметров, передаваемых при вызове подпрограмм, а также для временного хранения данных

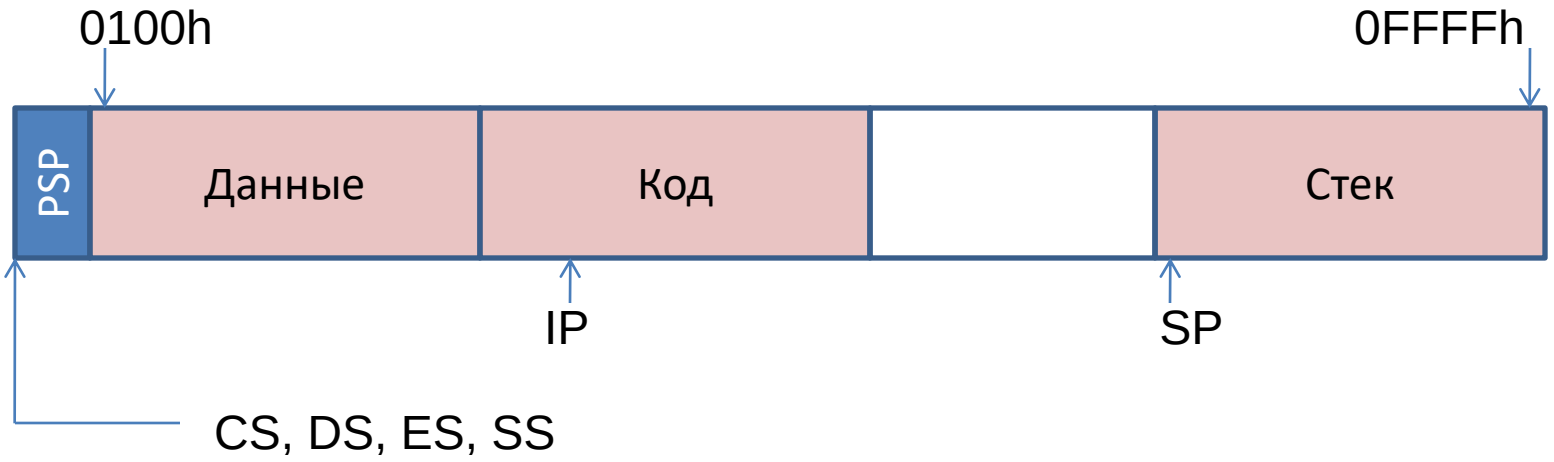
Типы сегментов и сегментные регистры

Тип сегмента	Сегментный регистр
Сегмент кода	**CS**
Основной сегмент данных	**DS**
Дополнительный сегмент данных	**ES**
Сегмент стека	**SS**

Модели памяти

Модель памяти	Адресация кода	Адресация данных	Операционная система	Чередование кода и данных
TINY	NEAR	NEAR	MS-DOS	Допустимо
SMALL	NEAR	NEAR	MS-DOS, Windows	Нет
MEDIUM	FAR	NEAR	MS-DOS, Windows	Нет
COMPACT	NEAR	FAR	MS-DOS, Windows	Нет
LARGE	FAR	FAR	MS-DOS, Windows	Нет
HUGE	FAR	FAR	MS-DOS, Windows	Нет
FLAT	NEAR	NEAR	Windows 2000, Windows NT, Windows XP, Windows 7-10	Допустимо

Модель памяти *tiny*

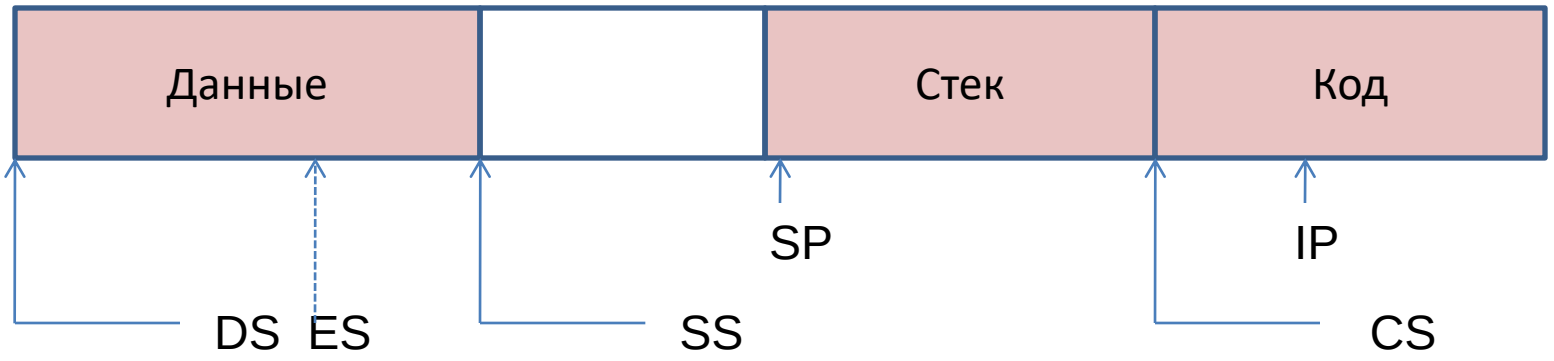


PSP – *префикс программного сегмента* – специфическая структура данных MS DOS.

Все четыре сегментных регистра (CS, DS, SS и ES) устанавливаются на один и тот же адрес, что дает общий размер кода, данных и стека, равный 64К.

Она является плоской моделью памяти, предназначенной для *реального режима.*

Модель памяти *small*



Сегменты кода и данных расположены отдельно друг от друга и не перекрываются, что позволяет иметь 64К кода программы, 64К данных и 64К стека.

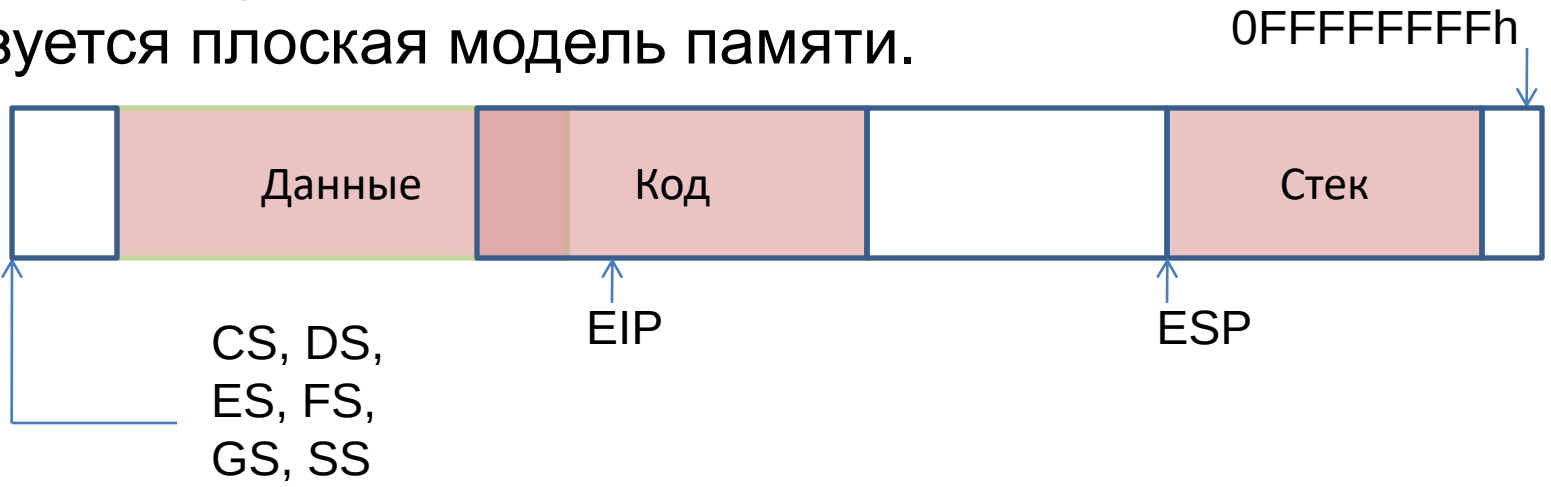
Строгой спецификации относительно размещения сегмента стека и PSP для этой и других моделей памяти нет.

Недостатки сегментной организации памяти

- размер сегмента нигде не хранится, так что существует опасность задать эффективный адрес, выводящий за границы сегмента, либо несуществующий адрес;
- сегменты могут перекрываться, и переход в другой сегмент никак не контролируется;
- размер сегмента (64 Кбайт) зачастую недостаточен для хранения больших массивов данных;
- контроль над правильностью заполнения сегментных регистров практически отсутствует.

Модель памяти *flat*

В абсолютном большинстве современных 32-разрядных ОС (для микропроцессоров Intel) используется плоская модель памяти.



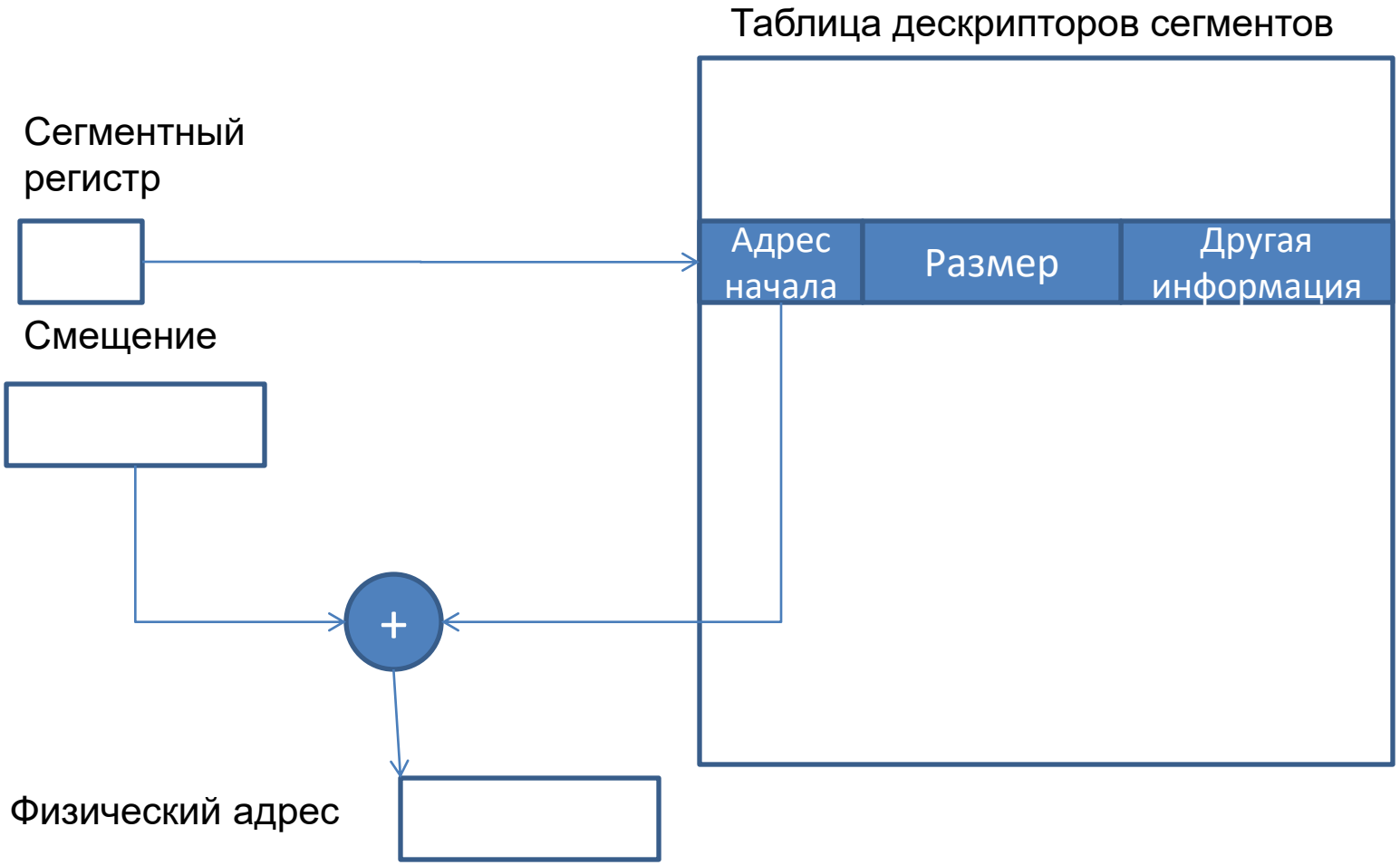
предполагает несегментированную конфигурацию программы, используется только в 32-разрядных операционных системах и позволяет работать с сегментами очень большого размера – 4 Гбайта.

Эта модель подобна модели tiny в том смысле, что данные и код размещены в одном сегменте, только 32-разрядном.

Сегментная модель памяти защищенного режима

- Для каждого сегмента создается специальная управляющая структура – дескриптор сегмента. Различные поля этой структуры хранят сведения о начальном адресе сегмента, его размере и других важных характеристиках.
- Сегментные регистры в защищенном режиме служат для доступа к элементам дескрипторных таблиц.

Сегментная модель памяти защищенного режима



Работа со стеком

Для организации временного хранения данных, а также для удобной реализации механизма подпрограмм в процессоры **Intel** на аппаратном уровне включен специальный механизм работы со стеком.

При работе процессора **Intel** в оперативной памяти выделяется специальный сегмент – **сегмент стека**. Этот сегмент может перекрываться с другими сегментами.

Работа со стеком

Для хранения адреса начала сегмента стека используется регистр **SS.**

Эффективный адрес вершины стека хранится в регистре **ESP**.

Таким образом, пара регистров **SS:ESP** описывает адрес вершины стека.

Работа со стеком

Для работы со стеком могут быть использованы следующие возможности:

- занесение слова в стек по команде **PUSH (PUSHA)**;
- извлечение элемента по команде **POP (POPA)**;
- неявная работа со стеком при вызове подпрограмм и выходе из подпрограмм;
- неявная работа со стеком при обработке прерываний.

Работа со стеком

Кроме того, мы можем работать с элементами, находящимися в стеке, как с обычными данными в памяти.

В частности, мы можем получить доступ к любому элементу стека, а не только к его вершине.

Для этих целей активно используется регистр **`EBP`**.