Адресное пространство, физическая и виртуальная память

Луцив Дмитрий Вадимович

Кафедра системного программирования СПбГУ

Содержание

Основные термины

- **Физический адрес** — номер байта в оперативной памяти, начиная с нулевого
- **Виртуальный (Логический) адрес** — адрес, с которым работает прикладное ПО, например, значение указателя
- **Адресное пространство** — множество логических адресов
- **Адресное преобразование** — вычисление физического адреса по виртуальному

Виды адресации (кроме страничной)

- Прямая адресация
- Банковые расширения
- Совместимые расширения и эволюция ЦП
- Сегментная адресация

Сущность прямой адресации

- **Прямая адресация** — способ адресации, при которой физический адрес равен логическому

Т.е. адресное преобразование тривиально, фактически «указатель» сразу выдаётся на шину адреса

Свойства прямой адресации

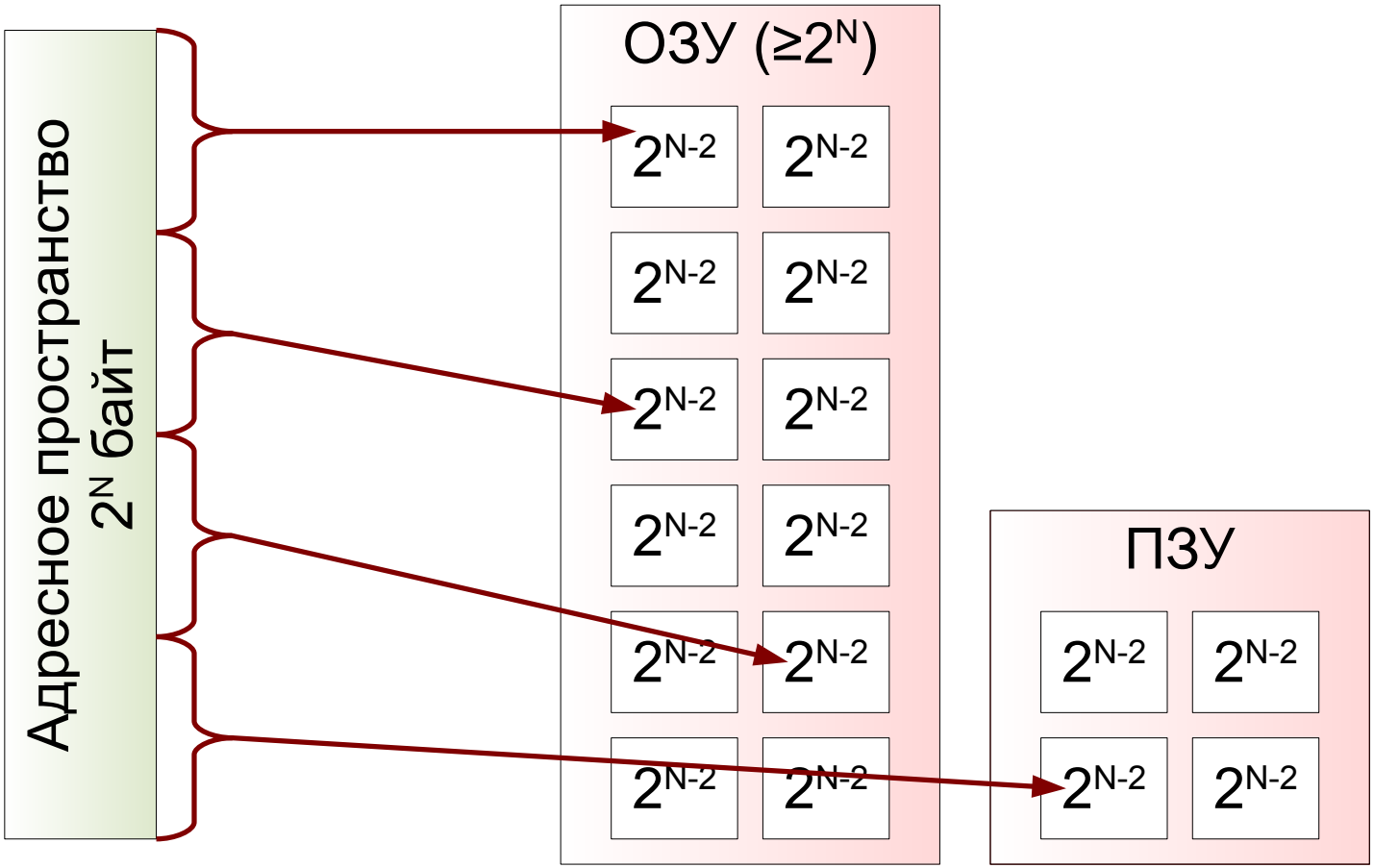
- Типично для восьмибитной шины данных — 16-битные внутренняя арифметика и шина адреса
- Изначально даже возможные 64KiB не использовались. Примеры:
  - Sinclair ZX 80 — ЦП Z80 (64K), но только 1K ОЗУ; Sinclair ZX
  - Spectrum — тоже Z80, но от 16 K ОЗУ

Свойства прямой адресации

- Типично для восьмибитной шины данных — 16-битные внутренняя арифметика и шина адреса
- Изначально даже возможные 64KiB не использовались. Примеры:
  - Sinclair ZX 80 — ЦП Z80 (64K), но только 1K ОЗУ; Sinclair ZX
  - Spectrum — тоже Z80, но от 16 K ОЗУ

Для офисных приложений и игр нужны графика и больше памяти

Сущность банковых расширений



Банковая адресация — способ адресации, при которой адресное пространство разделяется на *банки адресного пространства*, которым в соответствие ставятся *банки оперативной памяти*

Соответствие обычно не произвольное, варианты ограничены

Свойства банковых расширений

Хорошо

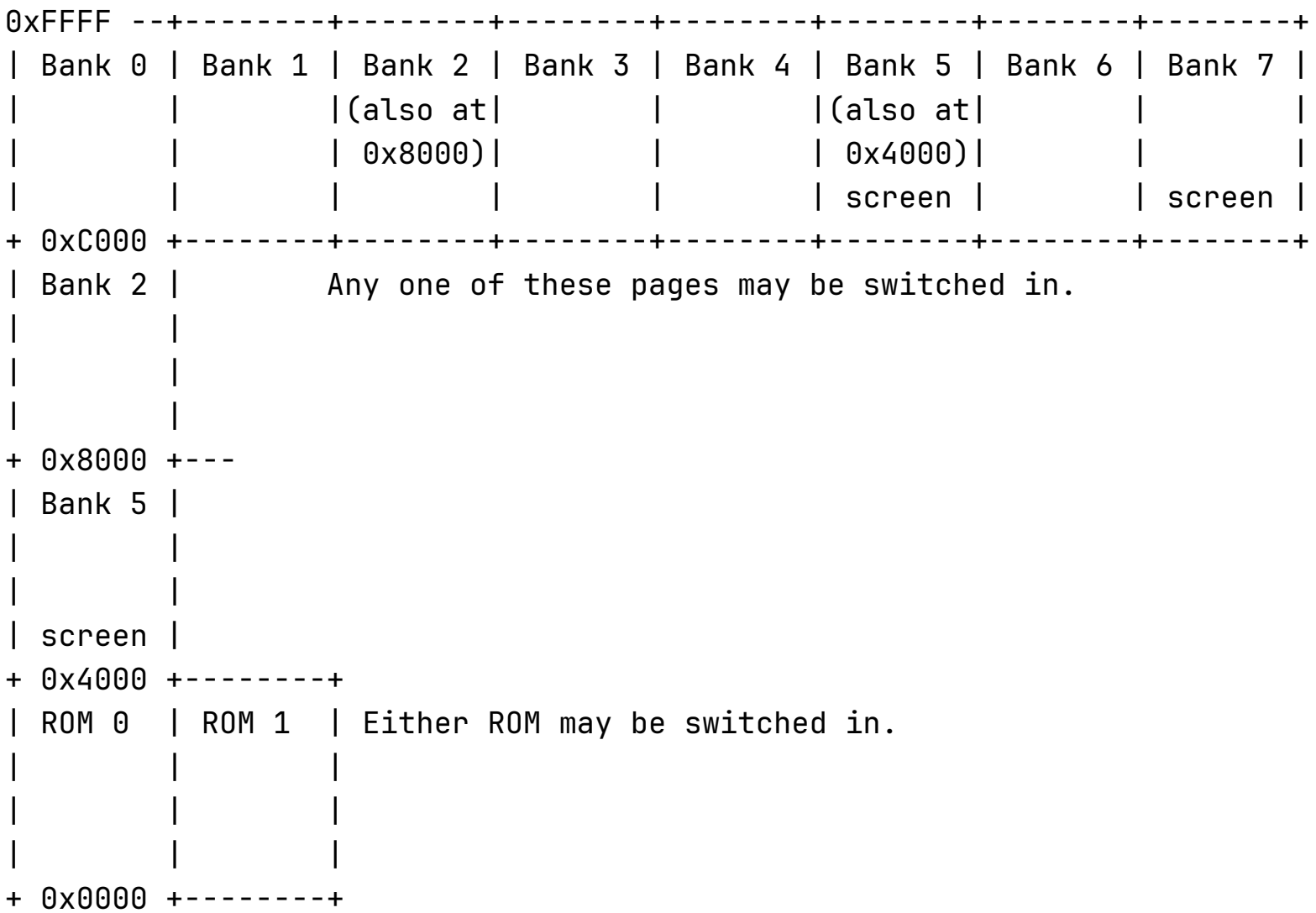
- Процессор тот же, вообще вмешательство в архитектуру минимально
- Дешево стоит
- Память можно расширять произвольно, доступно радиолюбителям
- Сравнительно удобно работать с данными

Плохо

- Работает не быстро
- Программируется через порты; тяжело менять банки кода, особенно для:
  - Вызова процедур
  - Прерываний
  - Переключения задач
- Разные расширения часто несовместимы

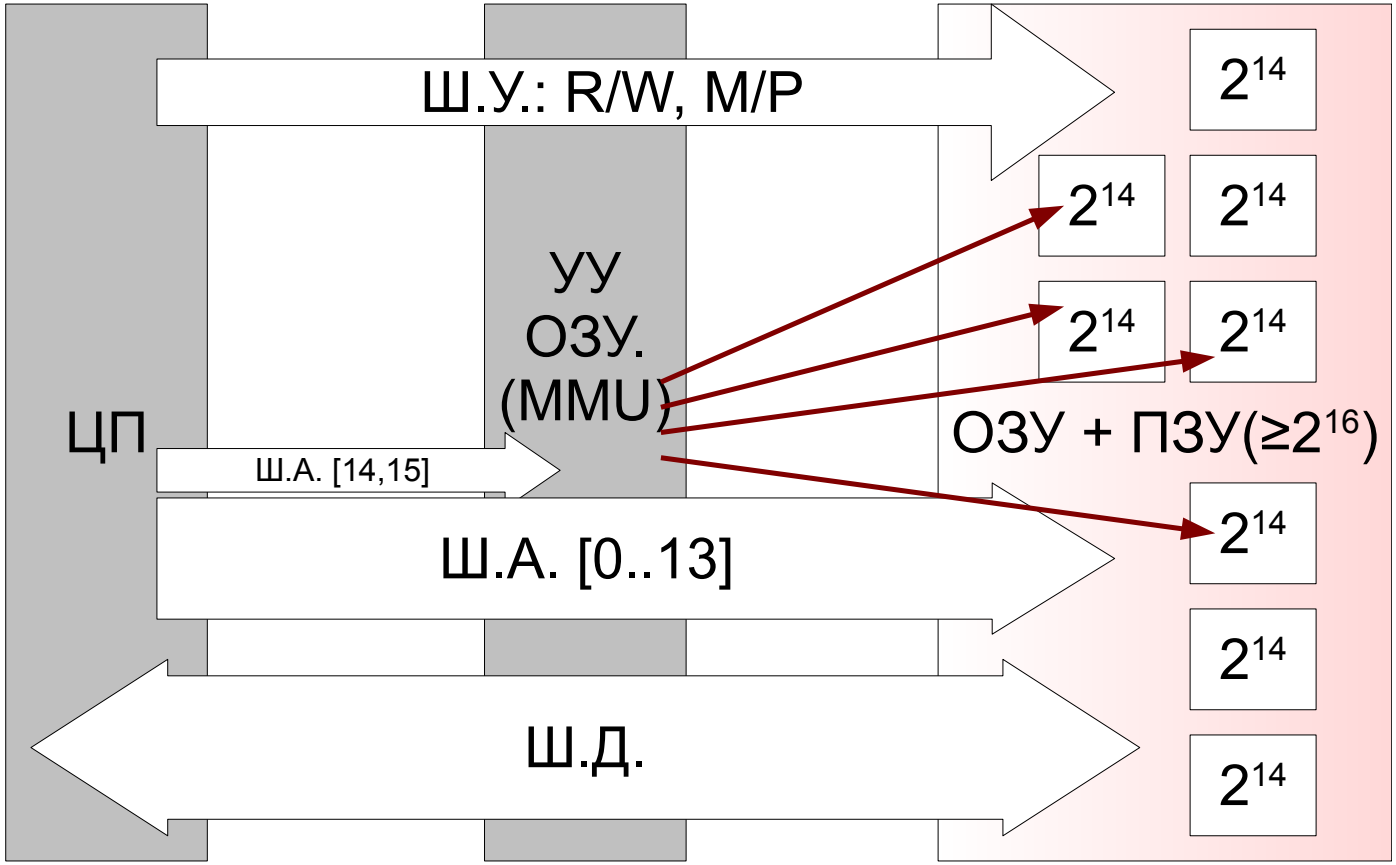
Пример 1: ZX Spectrum 128K

ZX Spectrum 128K



Пример 1: Как это реализовано

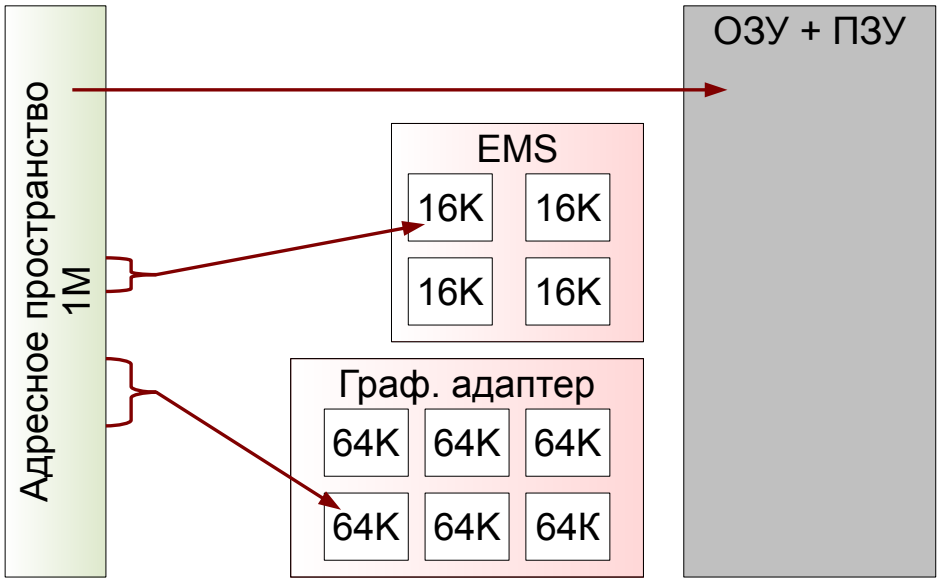
Адресное преобразование вычисляет номер банка ОЗУ по номеру банка (старшим 2 битам) АП. Отображение хранится в регистрах схемы MMU



Адресное преобразование: $A_{ph} = MMU[A_{log;14...15}] + A_{log;0...13}$

Пример 2: Окна в адресном пространстве IBM PC

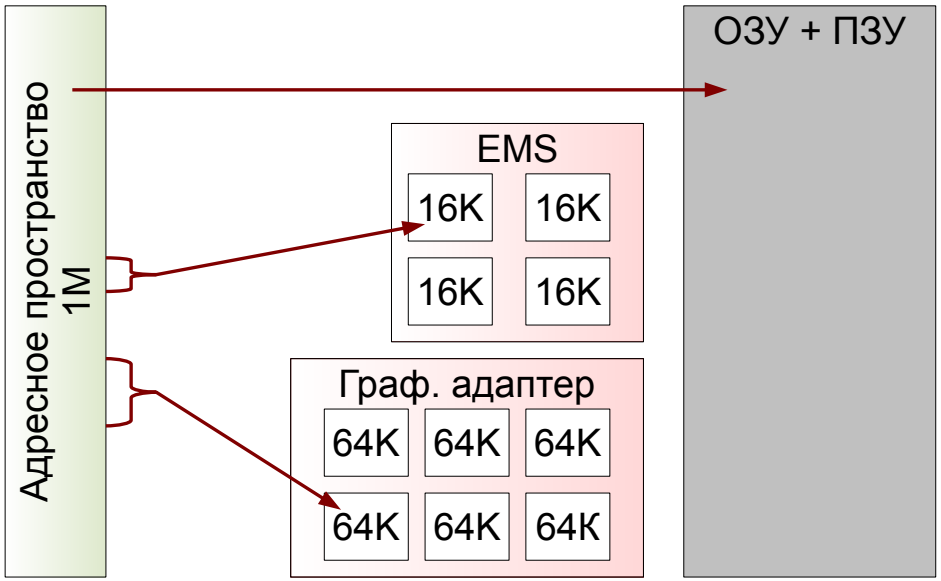
Можно рассматривать, как вариант банковой системы



- Некоторые внешние устройства, например графические адаптеры, поддерживают до сих пор
- Использовалась в EMS — сначала плата расширения, затем встроенная поддержка на материнской плате, затем эмуляция при помощи страничной адресации

Пример 2: Окна в адресном пространстве IBM PC

Можно рассматривать, как вариант банковой системы



- Некоторые внешние устройства, например графические адаптеры, поддерживают до сих пор
- Использовалась в EMS — сначала плата расширения, затем встроенная поддержка на материнской плате, затем эмуляция при помощи страничной адресации
- Использовалась в XMS — сразу чисто программное решение (эмуляция), Intel 80286+

Адресное преобразование: устройства перехватывают обращение к ОЗУ на системной шине

Расширение адресных регистров

Расширение адресных регистров

- Старый машинный код не знает, что у адресного регистра есть старшая половинка
- Все переходы и обращения к данным в старом коде «короткие»
- Т.к. старые приложения не используют всю память, появляется внутренняя фрагментация

Расширение адресных регистров

Расширение адресных регистров

- Старый машинный код не знает, что у адресного регистра есть старшая половинка
- Все переходы и обращения к данным в старом коде «короткие»
- Т.к. старые приложения не используют всю память, появляется внутренняя фрагментация

Пример:

- i8086 – i80286 — 16-битный регистр BP
- i80386 – i80586 — 32-битный регистр EBP (но младшие 16 бит доступны, как BP)
- x86_64 — 64-битный регистр RBP (но доступны EBP и BP)

Смысл сегментной адресации

- Компилятор (точнее, компоновщик) генерирует программу, в бинарном файле которой обозначены данные разных видов — машинный код, статические данные (проинициализированные глобальные переменные) и т.д.
- Операционная система может размещать эти данные в физической памяти по своему усмотрению
- Получается, что нужен уровень косвенности, как при вызове API ядра через прерывания или спец. инструкции, только уже внутри программы.

Реализация сегментной адресации (1): Жёсткие сегменты x86

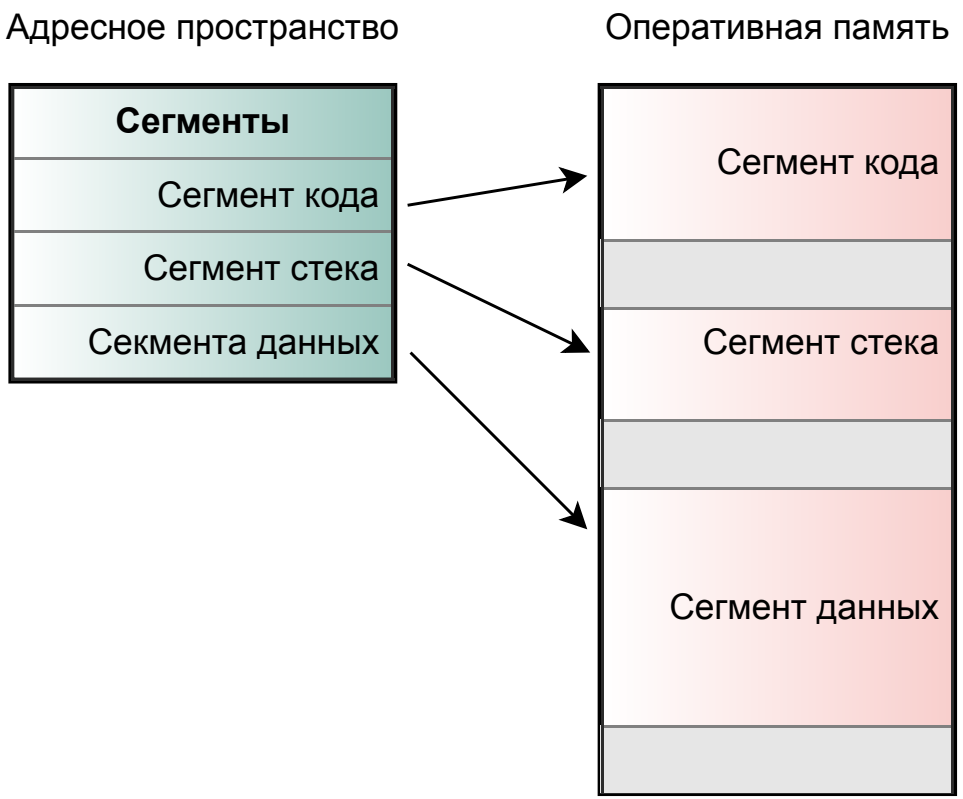
В 16-битном режиме x86 полный указатель состоит из *смещения* (адресный регистр) и *номера сегмента* (хранится в сегментном регистре). Пример для x86 (i8086):

- Адрес исполняемой инструкции `CS:IP` — Code Segment, Instruction Pointer
- Адрес вершины стека `SS:SP` — Stack Segment, Stack Pointer
- Адреса данных:
  - `DS:SI`, `ES:SI`, `DS:BX` и т.д., много сочетаний

В ранних x86 защиты не было, и программа могла работать с разными сегментами * Обычно ОС (чаще всего DOS) выдавала ей по одному сегменту кода, стека и данных * Если 64К для чего-то из этого было мало, можно было использовать другие модели памяти, работая с разными сегментами. Подробнее здесь

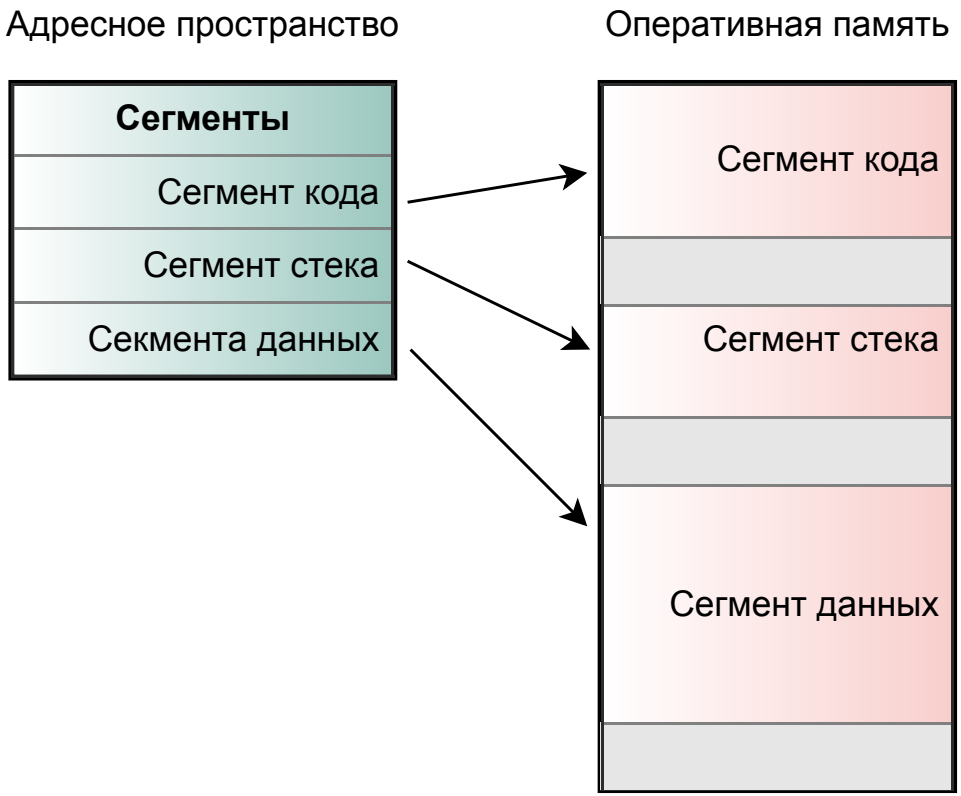
Адресное преобразование: $A_{ph} = S \cdot 16 + O$

Реализация: Управляемые сегменты



- Произвольного размера
- Требуют специальных таблиц размещения (таблицы дескрипторов)
- Могут ассоциироваться с правами на хранение данных и кода и выполнение кода

Реализация: Управляемые сегменты



- Произвольного размера
- Требуют специальных таблиц размещения (таблицы дескрипторов)
- Могут ассоциироваться с правами на хранение данных и кода и выполнение кода

Если включена защита, процесс не может обращаться к чужим сегментам (как минимум должна «попросить разрешения» у ОС)

Адресное преобразование: $A_{ph} = T_{desc}[S] + O$

Виртуальная память и страничная адресация

Что такое виртуальная память

Виртуальная память — механизм работы с адресным пространством, превосходящим по размеру оперативную память или тот её фрагмент, который выделен данному процессу

Виртуальная память может по размеру превосходить физическую, и состоять из различных фрагментов, находящихся ОЗУ (по разным физическим адресам) или даже жесткого диска (за счёт чего и можно наращивать её размер). При этом адресное преобразование скрывает всю эту неоднородность от программы

Предпосылки

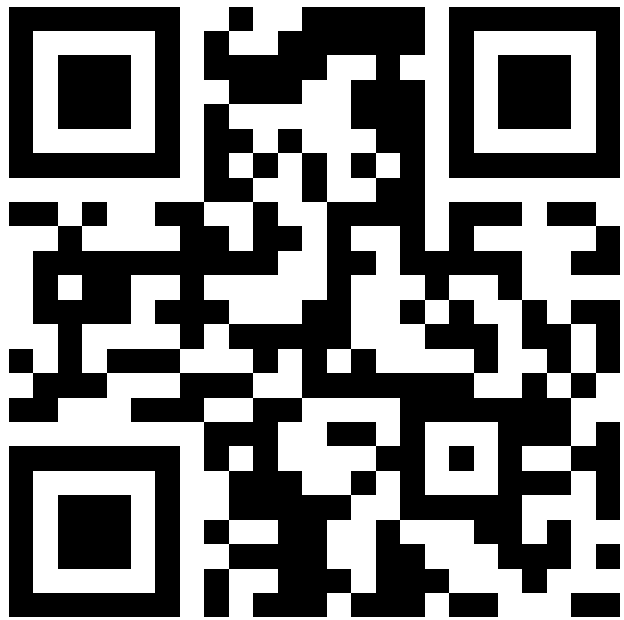
Задачи

- Ранние ЭВМ — научные расчёты, обычно сравнительно небольшой код и объёмные данные. Данные можно обрабатывать порциями. ПО относительно недорого в масштабах системы
- Позже — большее количество программ, больший объём кода. Большая относительная стоимость ПО

Решения

- Типичный способ работать с большим машинным кодом — загружать его кусками — *оверлеями* (overlay). Иногда использовался и для DOS на PC.
- В некоторых архитектурах часть адресного пространства проецировалась на диск (или барабан), чтобы облегчить программисту работы
- ...
- Сделать прозрачную для программиста систему

Вопросы



EDU.DLUCIV.NAME