Виртуальная память – I

Основы информатики.

Компьютерные основы программирования

goo.gl/X7evF

На основе **CMU** 15-213/18-243:
Introduction to Computer Systems

goo.gl/TDDVV

Лекция **13**, **18** Мая, 2015

Лектор:

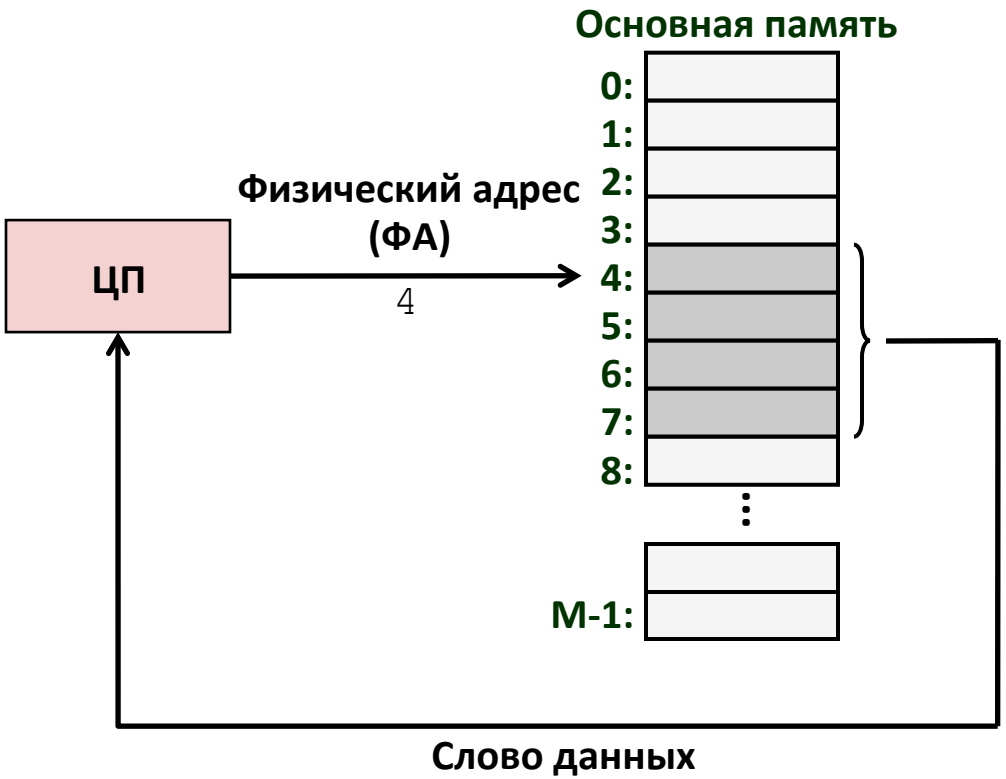
Дмитрий Северов, кафедра информатики 608 КПМ

dseverov@mail.mipt.ru

Виртуальная память – 1: понятия

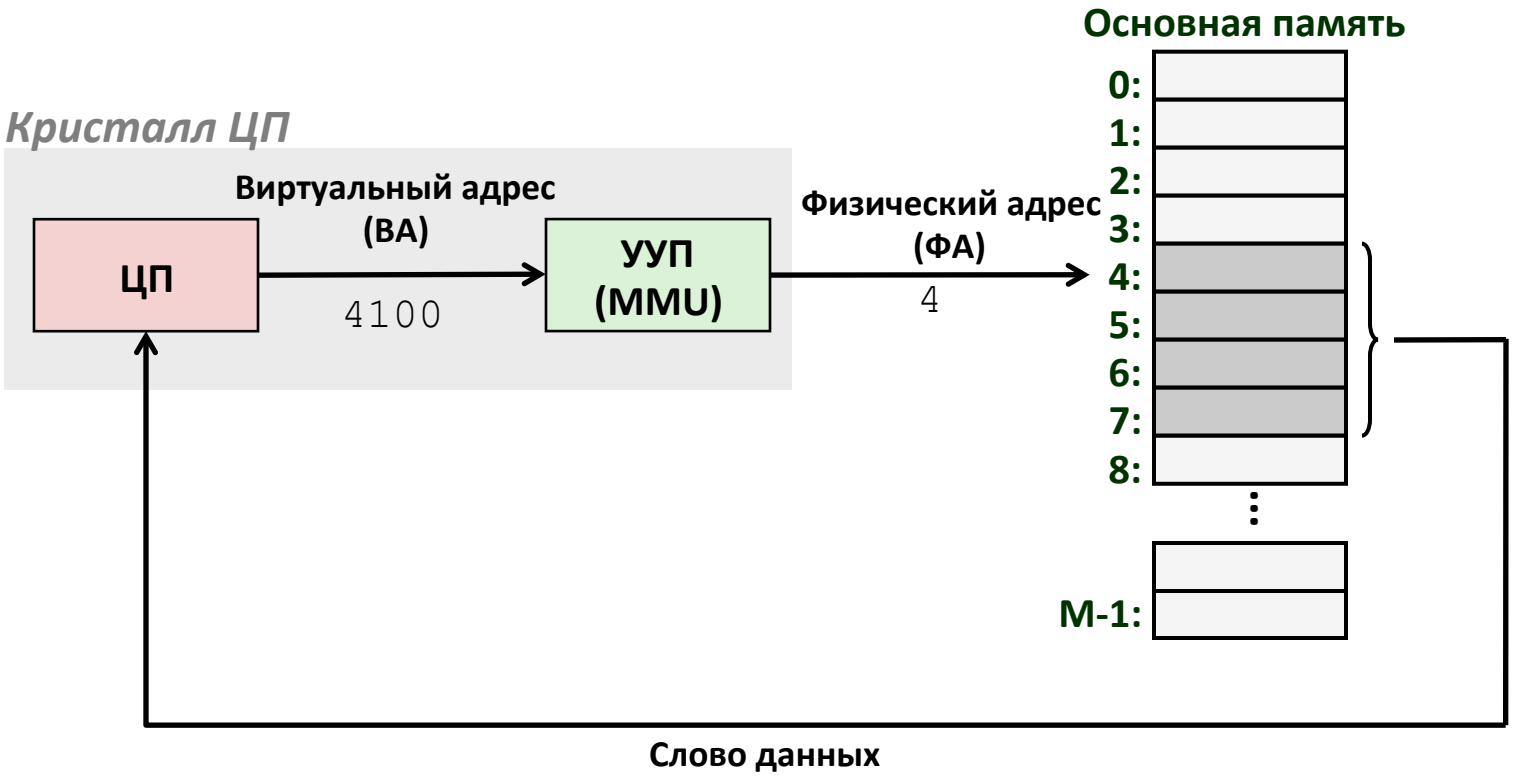
- Пространства адресов
- ВП как средство кеширования
- ВП как средство управления памятью
- ВП как средство защиты памяти
- Трансляция адресов

Система с физической адресацией



- Используется в “простых” системах – микроконтроллерах, встроенных в датчики, регуляторы света, часы

Системы с виртуальной адресацией



- Используется во всех современных серверах, ПК, смартфонах
- Одна из замечательных идей в информатике

Адресные пространства

- **Линейное адресное пространство:** Упорядоченное множество смежных неотрицательных целых адресов: {0, 1, 2, 3 … }

- **Виртуальное адресное пространство:** Множество N = 2^n виртуальных адресов: {0, 1, 2, 3, …, N-1}

- **Физическое адресное пространство:** Множество M = 2^m физических адресов {0, 1, 2, 3, …, M-1}

- **Четкое различием между данными (байтами) и их атрибутами (адресами)**
- **Каждый объект может иметь несколько адресов**
- **Каждый байт в основной памяти имеет:**
  один физический адрес, один (или более) виртуальных адресов

Виртуальная память нужна, чтобы…

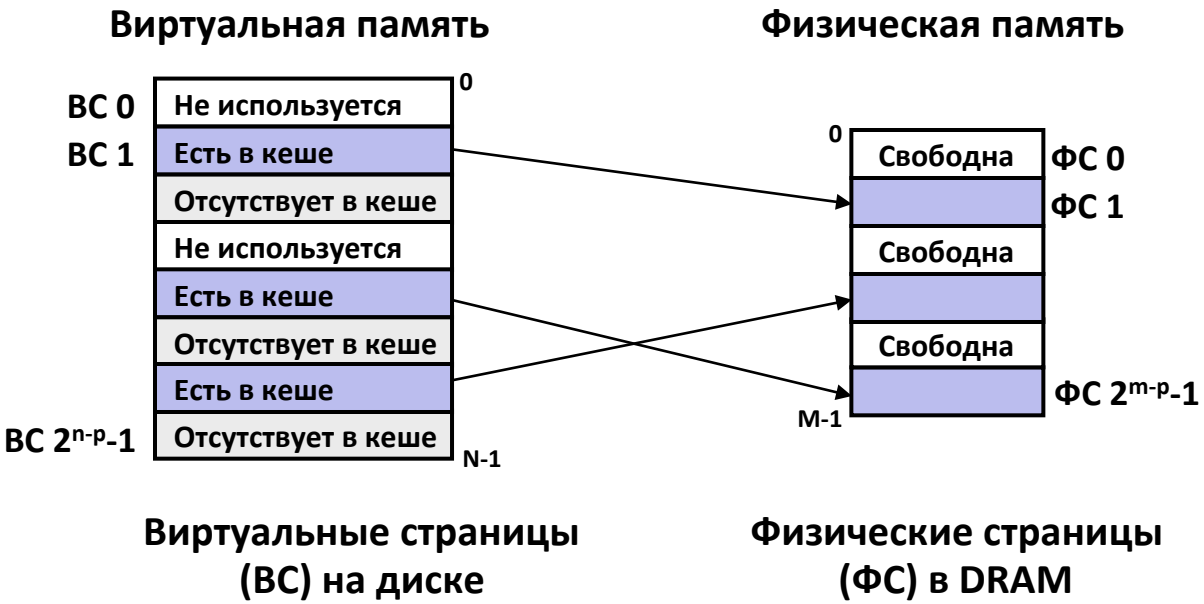
- **использовать физическую память эффективно**
  - DRAM – кеш к частям виртуального пространства
- **упростить управление памятью**
  - Каждый процесс получает типовой экземпляр линейного адресного пространства
- **изолировать адресные пространства**
  - Ни один процесс не может обращаться к памяти другого
  - Программы пользователей не имеют доступа привилегированной информации ядра ОС

Виртуальная память – 1: понятия

- Пространства адресов
- ВП как средство кеширования
- ВП как средство управления памятью
- ВП как средство защиты памяти
- Трансляция адресов

ВП как средство кеширования

- ***Виртуальная память -*** **массив N смежных байт на диске**
- **Содержимое массива на диске кешируется в *физической памяти* (*DRAM cache*)**
  - Блоки этого кеша – *страницы* размером P = 2^p байт



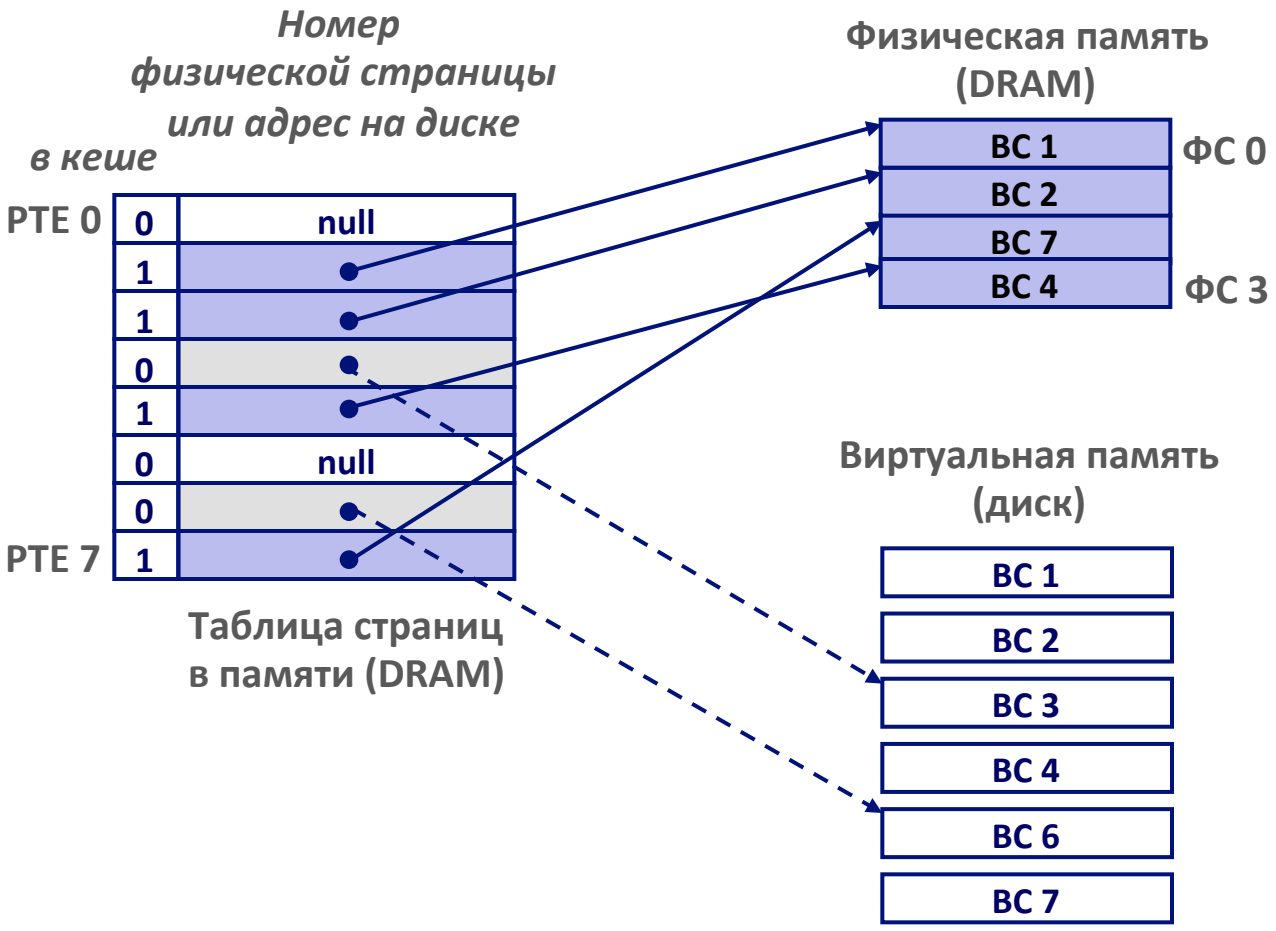
Организация DRAM-кеша

- **Определяется огромной ценой промаха**
  - DRAM на 2 порядка (***100 раз***) медленнее SRAM
  - Диск на 4 порядка (***10 тысяч раз***) медленнее DRAM

- **Следствия**
  - Большой размер страницы (блока): обычно 4-8 KB, иногда 4 MB
  - Полностью ассоциативен
    - любая ВС может быть размещена в любой ФС
    - требует “большую” функцию отображения, в отличие от кешей ЦП
  - Высокоизощрённые, дорогостоящие алгоритмы замены
    - Слишком сложные и неустоявшиеся для аппаратной реализации
  - Write-back, но не write-through

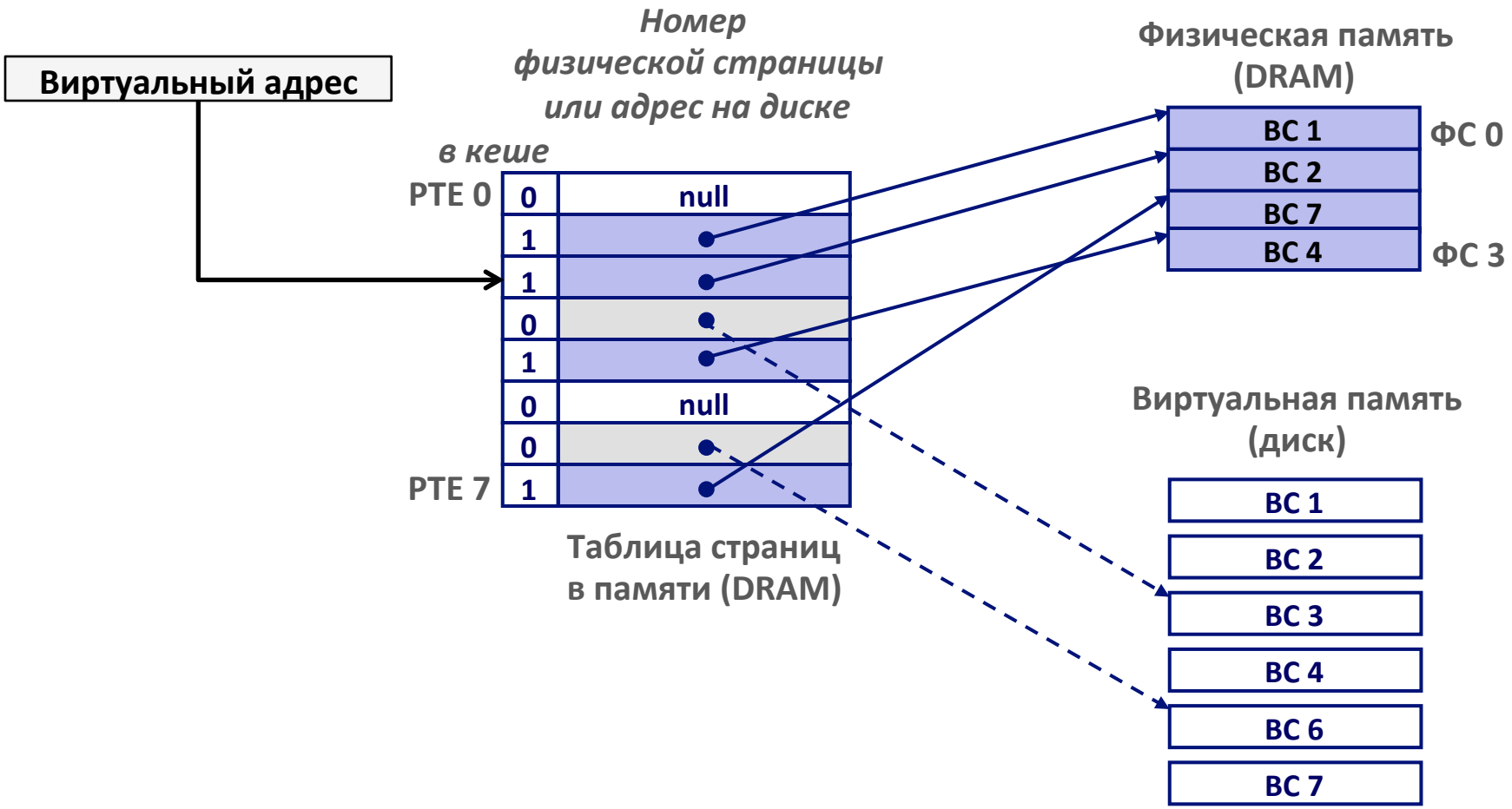
Таблица страниц

- ***Страничная таблица –*** **массив табличных записей (PTE) отображений виртуальных страниц на физические.**
  - Отдельная для каждого процесса структура данных ядра ОС в DRAM



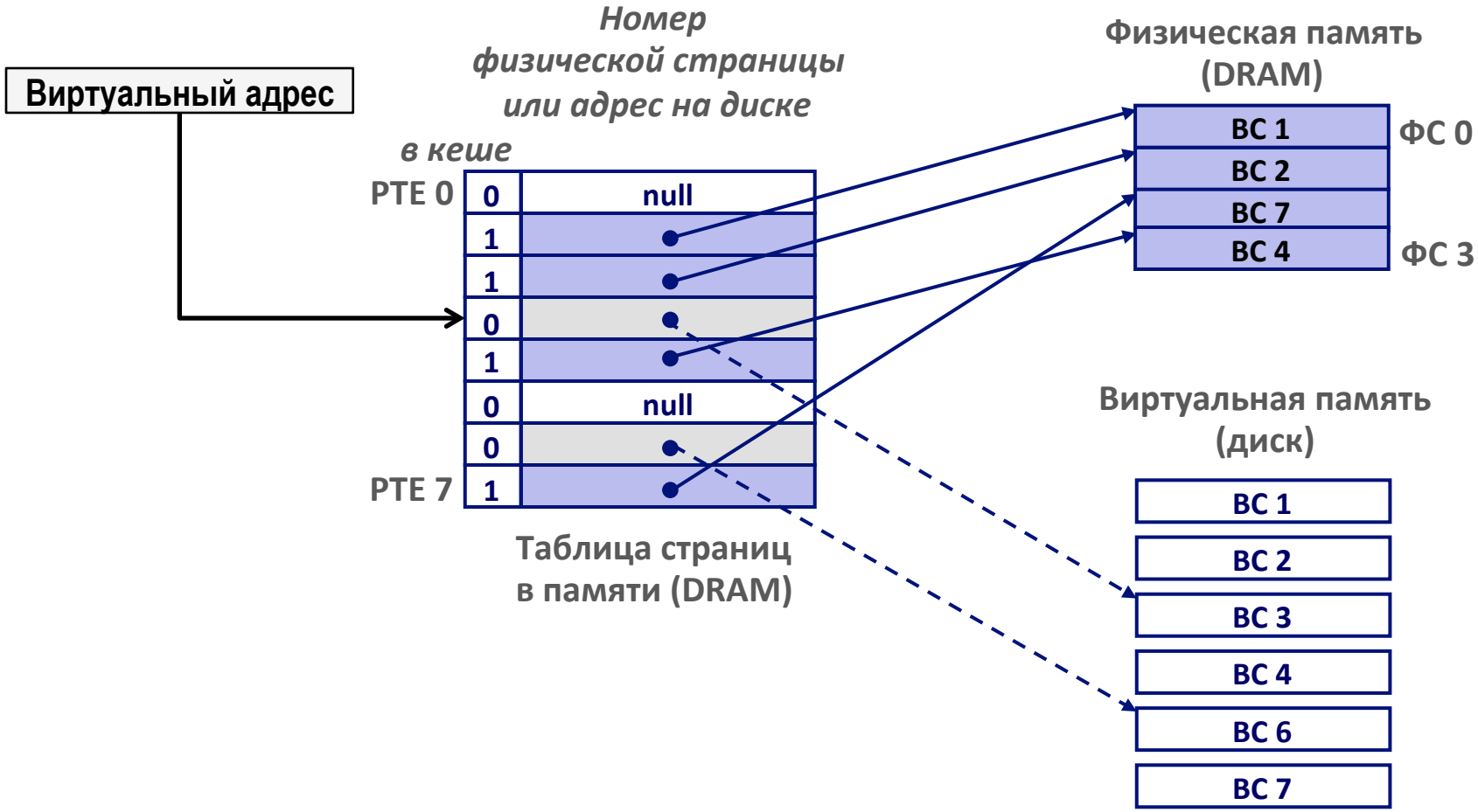
Страничное попадание

- ***Страничное попадание:*** **ссылка на страницу ВП говорит, что страница находится в DRAM (попадание DRAM-кеша)**



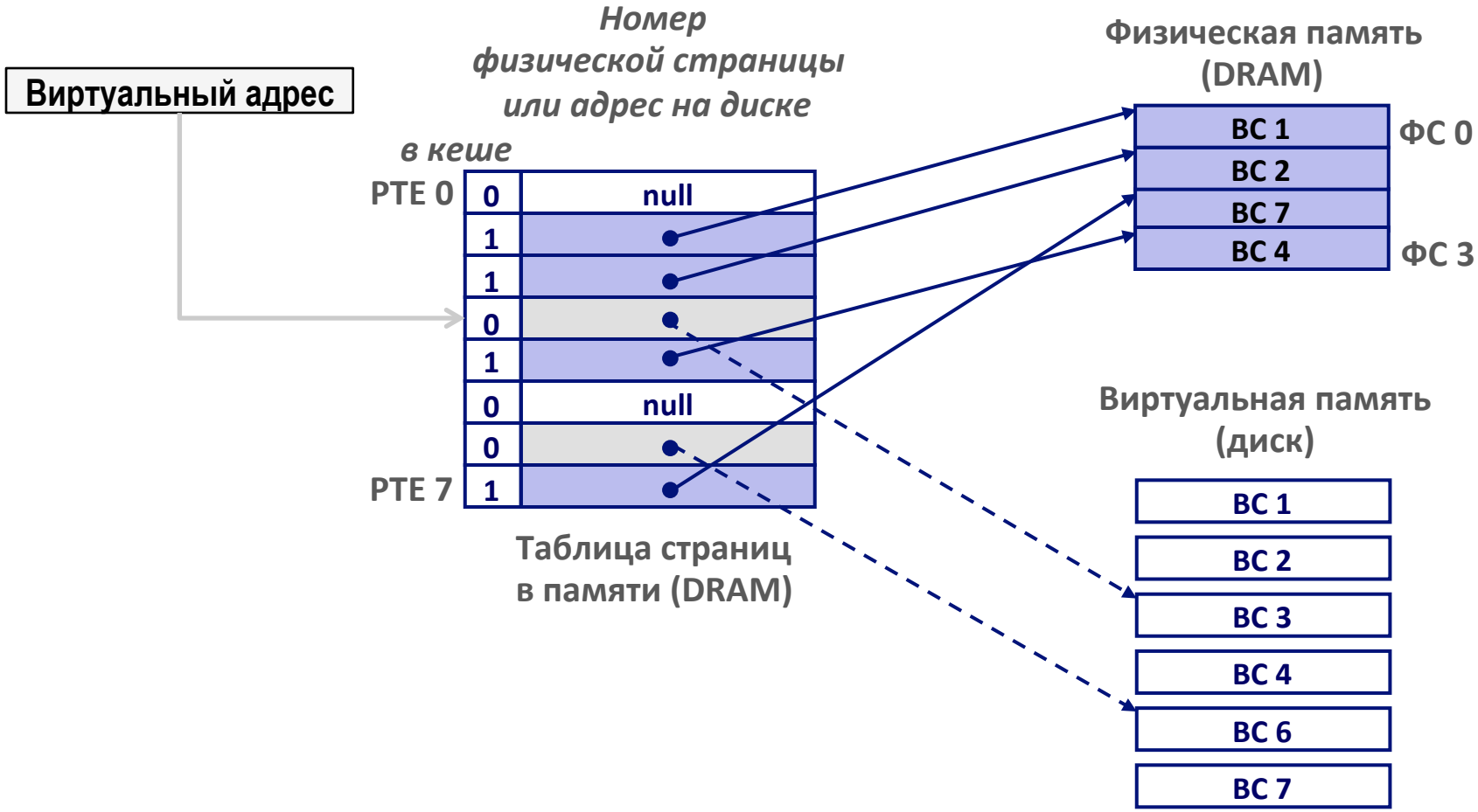
Страничный сбой

- ***Страничный сбой:*** **ссылка на страницу ВП говорит, что страница отсутствует в DRAM (промах DRAM-кеша)**



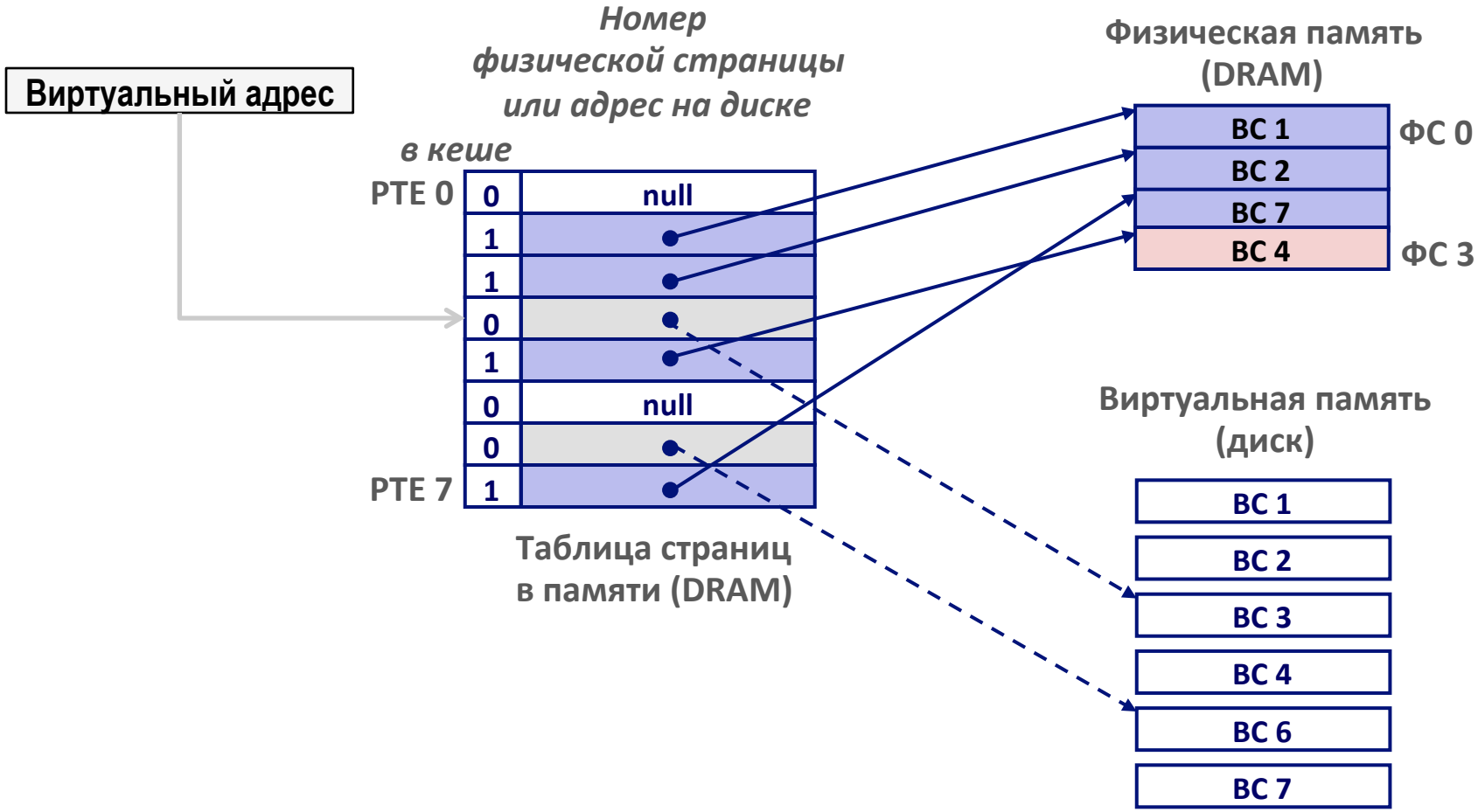
Обработка страничного сбоя – 1

- Страничный промах вызывает страничный сбой (исключение)



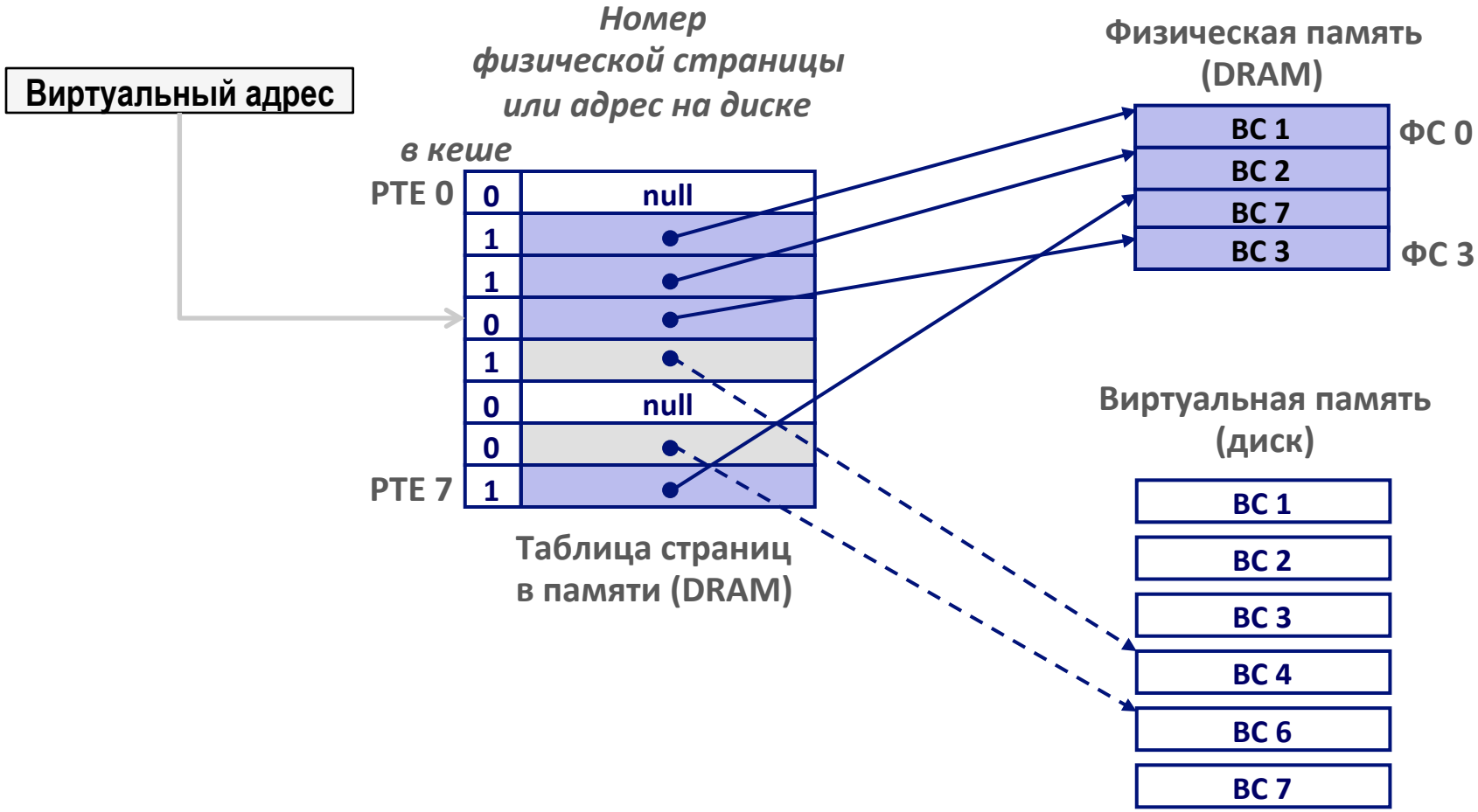
Обработка страничного сбоя – 2

- Страничный промах вызывает страничный сбой (исключение)
- Обработчик страничного сбоя выбирает жертву откачки (здесь ВС 4)



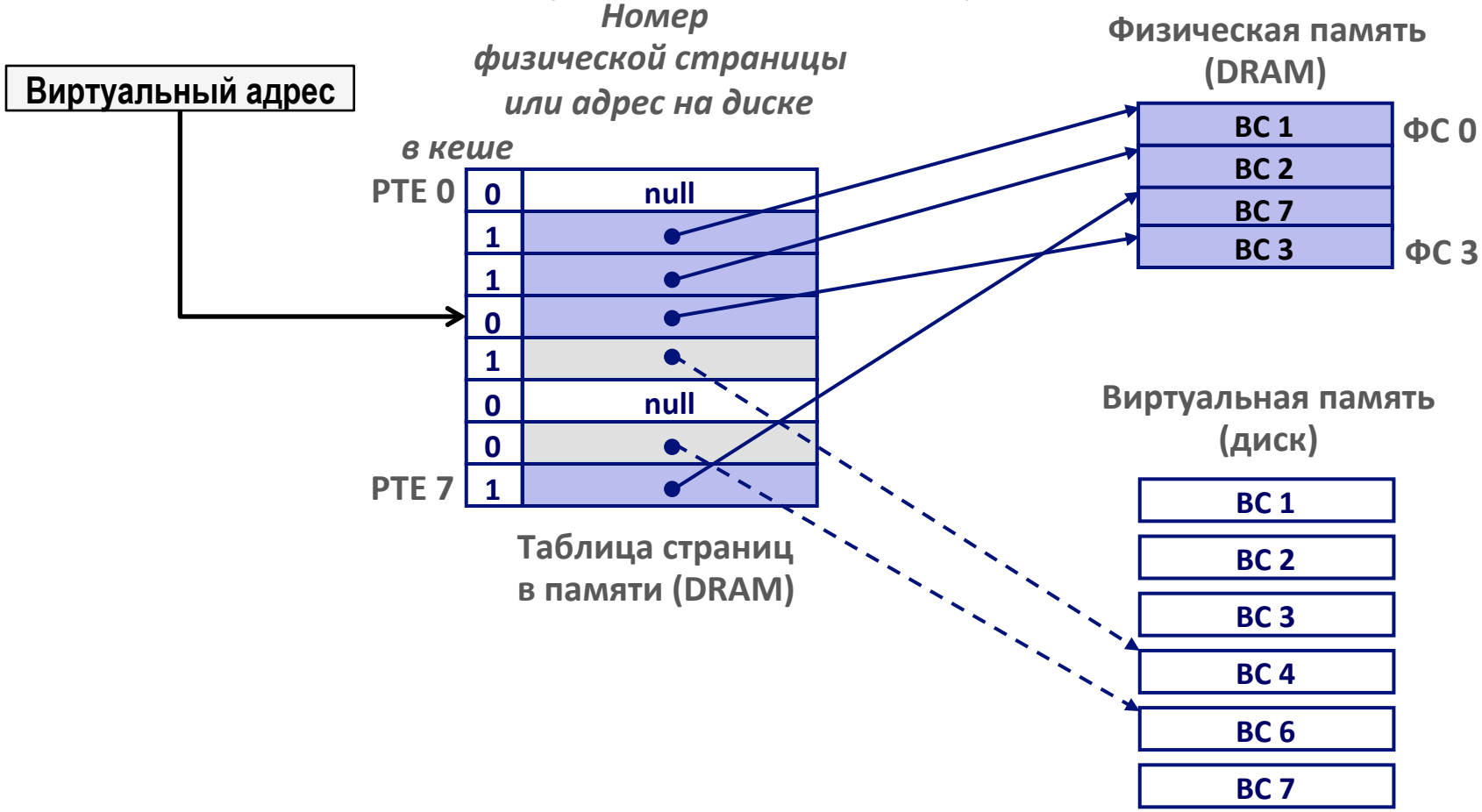
Обработка страничного сбоя - 3

- Страничный промах вызывает страничный сбой (исключение)
- Обработчик страничного сбоя выбирает жертву откачки (здесь ВС 4)
- Обработчик подкачивает с диска в память нужную страницу (здесь ВС 3)



Обработка страничного сбоя - 4

- Страничный промах вызывает страничный сбой (исключение)
- Обработчик страничного сбоя выбирает жертву откачки (здесь ВС 4)
- Обработчик подкачивает с диска в память нужную страницу (здесь ВС 3)
- Сбойная команда повторно выполняется : страничное попадание!



Локальность снова выручает!

- **Виртуальная память работает благодаря локальности**
- **В каждый момент времени, программы стремятся доступаться к набору активных виртуальных страниц – *рабочему набору***
  - Программы с лучшей временной локальностью будут иметь рабочий набор меньшего размера
- **Если размер рабочего набора < размера основной памяти**
  - Хорошая производительность процесса после неизбежных промахов
- **Если сумма размеров рабочих наборов > размера основной памяти**
  - *Пробуксовка (Thrashing):* деградация производительности при непрерывной откачке/подкачке страниц

Виртуальная память – 1: понятия

- Пространства адресов
- ВП как средство кеширования
- ВП как средство управления памятью
- ВП как средство защиты памяти
- Трансляция адресов

ВП как средство управления памятью - 1

- **Ключевая идея: каждому процессу – собственное виртуальное адресное пространство**
  - память представляется простым линейным массивом
  - отображение разбрасывает адреса по физической памяти
    - удачные отображения упрощают распределение и управление