Управление оперативной памятью

Основные задачи

1. Контроль состояния каждой единицы памяти (свободна/распределена)
2. Стратегия распределения памяти (кому, когда и сколько памяти должно быть выделено)
3. Выделение памяти (выбор конкретной области, которая должна быть выделена)
4. Стратегия освобождения памяти (процесс освобождает, ОС “забирает” окончательно или временно)

Управление оперативной памятью

Стратегии и методы управления

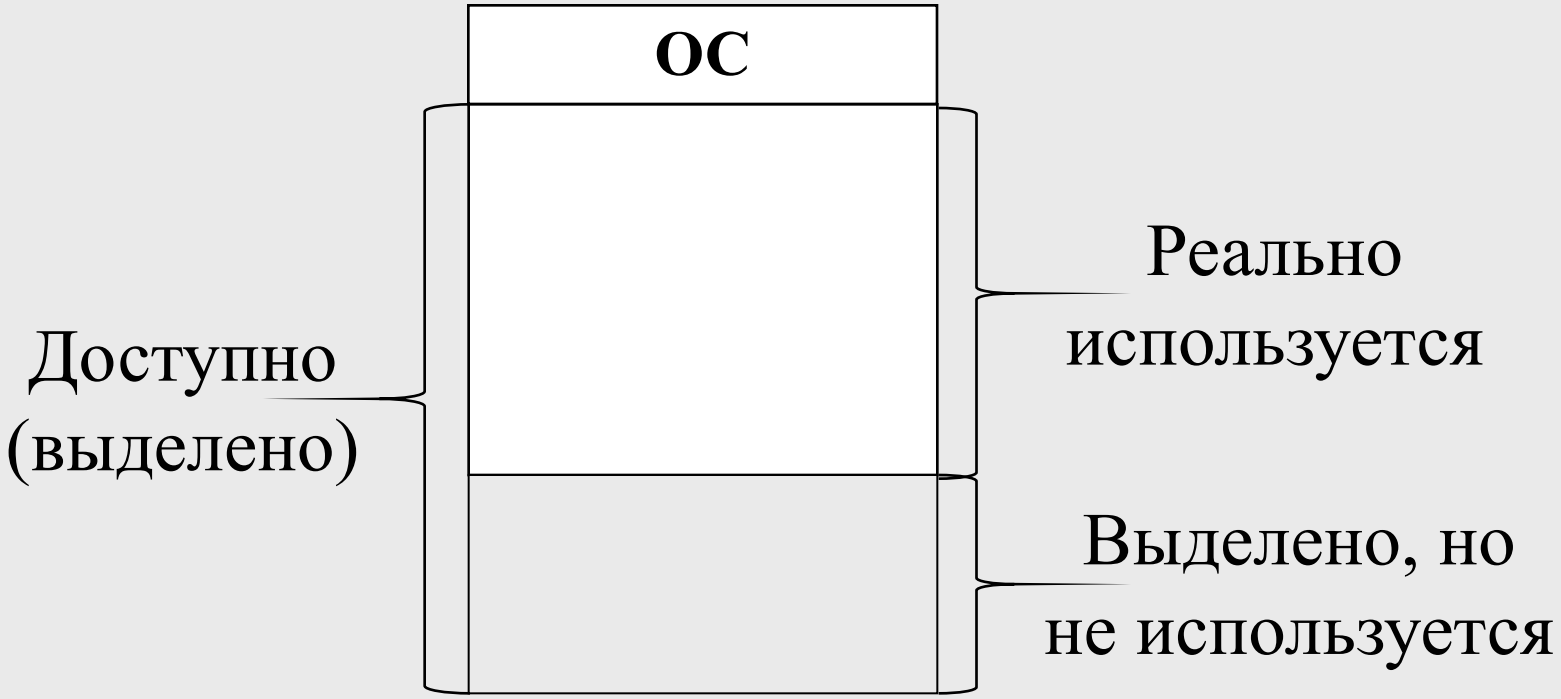
1. Одиночное непрерывное распределение
2. Распределение разделами
3. Распределение перемещаемыми разделами
4. Страничное распределение
5. Сегментное распределение
6. Сегментно-страничное распределение

План рассмотрения стратегий управления

1. Основные концепции
2. Необходимые аппаратные средства
3. Основные алгоритмы
4. Достоинства, недостатки

Одиночное непрерывное распределение

Основные концепции



Одиночное непрерывное распределение

Необходимые аппаратные средства

- Регистр границ + режим ОС / режим пользователя
- Если ЦП в режиме пользователя попытается обратиться в область ОС, то возникает прерывание

Алгоритмы: очевидны.

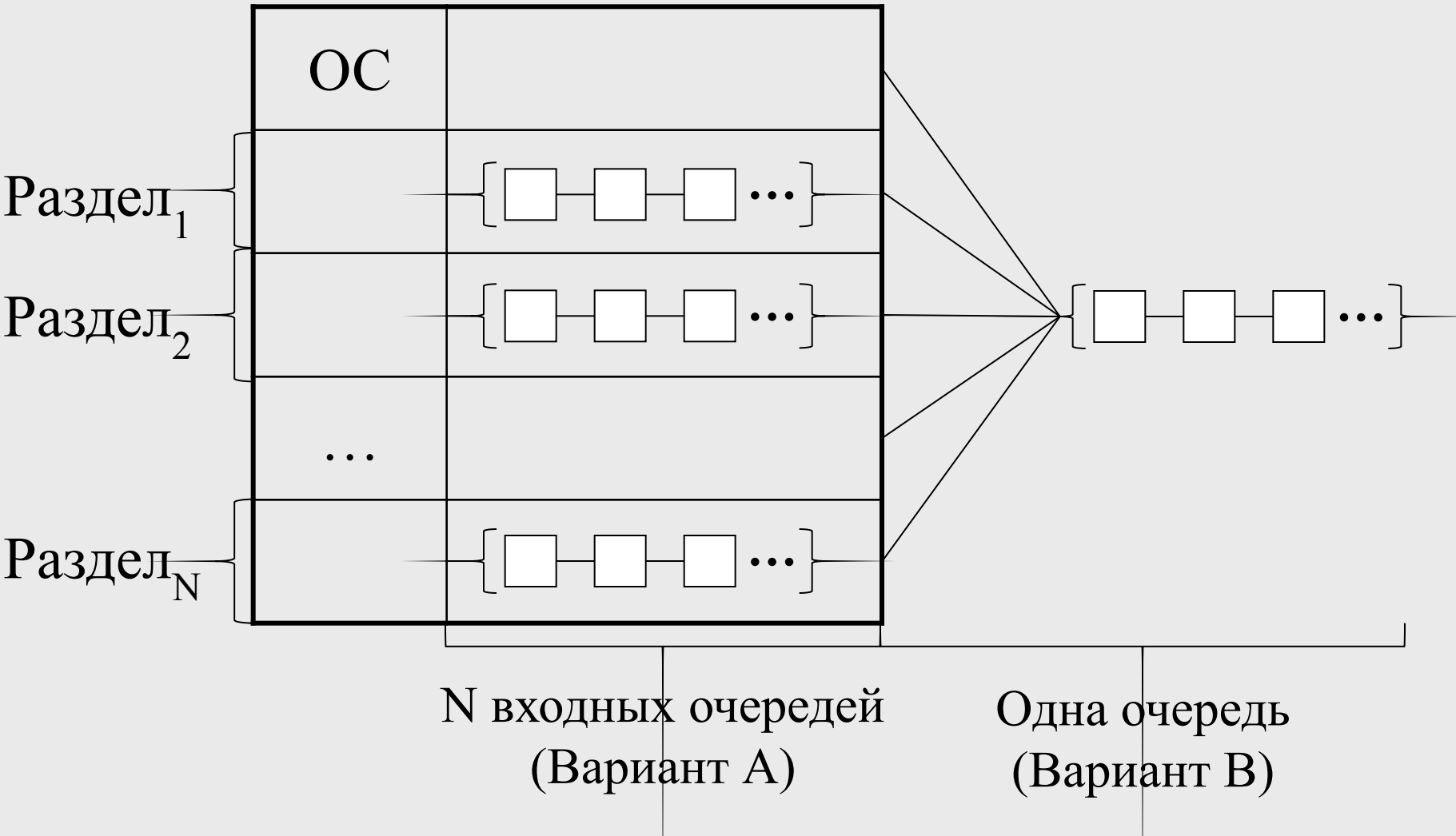
Достоинства: простота.

Недостатки

1. Часть памяти не используется
2. Процессом/заданием память занимается все время выполнения
3. Ограничение на размеры задания

Распределение неперемещаемыми разделами

Основные концепции



Распределение неперемещаемыми разделами

Необходимые аппаратные средства

1. Два регистра границ
2. Ключи защиты (PSW)

Распределение неперемещаемыми разделами

Алгоритмы

Модель статического определения разделов

A. Сортировка входной очереди процессов по отдельным очередям к разделам. Процесс размещается в разделе минимального размера, достаточного для размещения данного процесса. В случае отсутствия процессов в каких-то подочередях — неэффективность использования памяти.

Распределение неперемещаемыми разделами

Алгоритмы

Модель статического определения разделов

Б. Одна входная очередь процессов

1. Освобождение раздела $\Rightarrow$ поиск (в начале очереди) первого процесса, который может разместиться в разделе.
   Проблема: большие разделы $\leftrightarrow$ маленькие процессы
2. Освобождение раздела $\Rightarrow$ поиск процесса максимального размера, не превосходящего размер раздела.
   Проблема: дискриминация «маленьких» процессов
3. Оптимизация варианта 2. Каждый процесс имеет счетчик дискриминации. Если значение счетчика процесса $\geq K$, то обход его в очереди невозможен

Распределение неперемещаемыми разделами

Достоинства

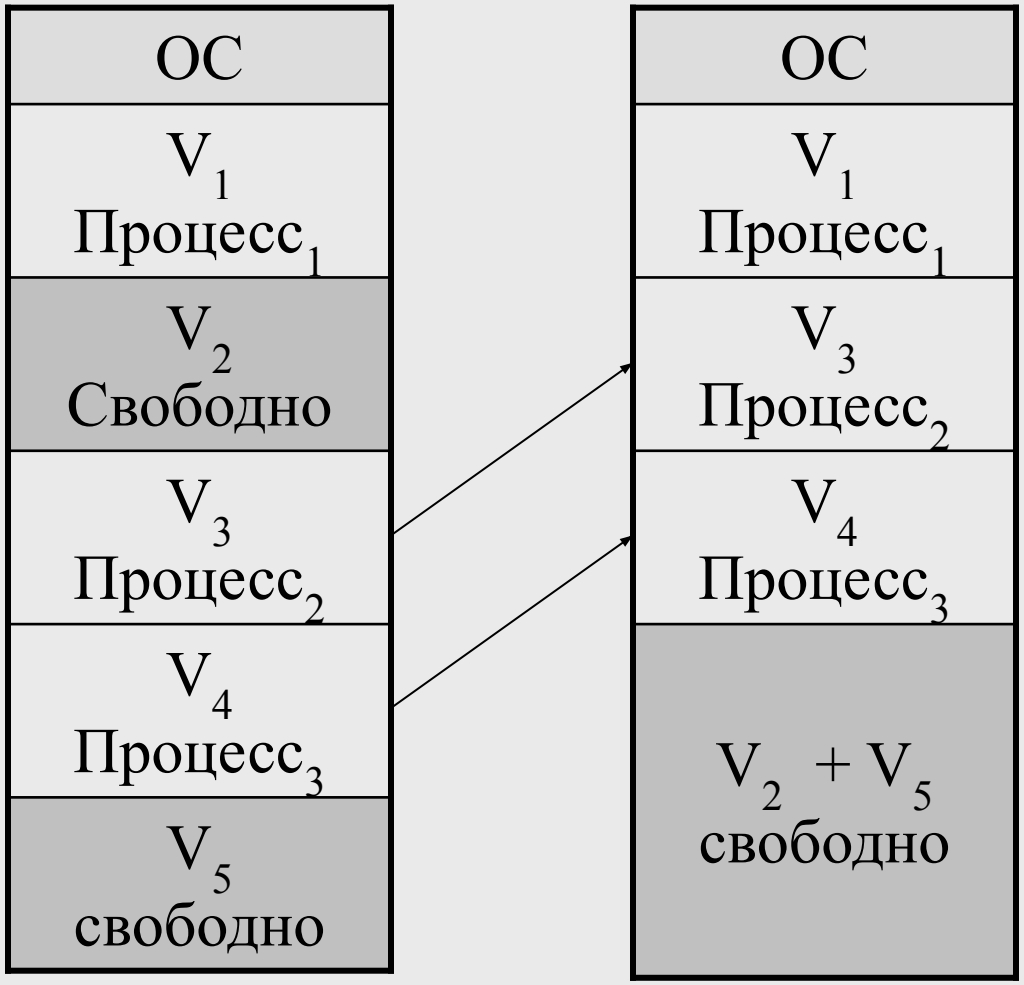
1. Простое средство организации мультипрограммирования
2. Простые средства аппаратной поддержки
3. Простые алгоритмы

Недостатки

1. Фрагментация
2. Ограничение размерами физической памяти
3. Весь процесс размещается в памяти — возможно неэффективное использование

Распределение перемещаемыми разделами

Основные концепции



Виртуальная память

$Процесс_4$

(например, $V = V_2 + ½ V_5$)

Распределение перемещаемыми разделами

Необходимые аппаратные средства

1. Регистры границ + регистр базы
2. Ключи + регистр базы

Алгоритмы:

Распределение перемещаемыми разделами

Достоинства

1. Ликвидация фрагментации

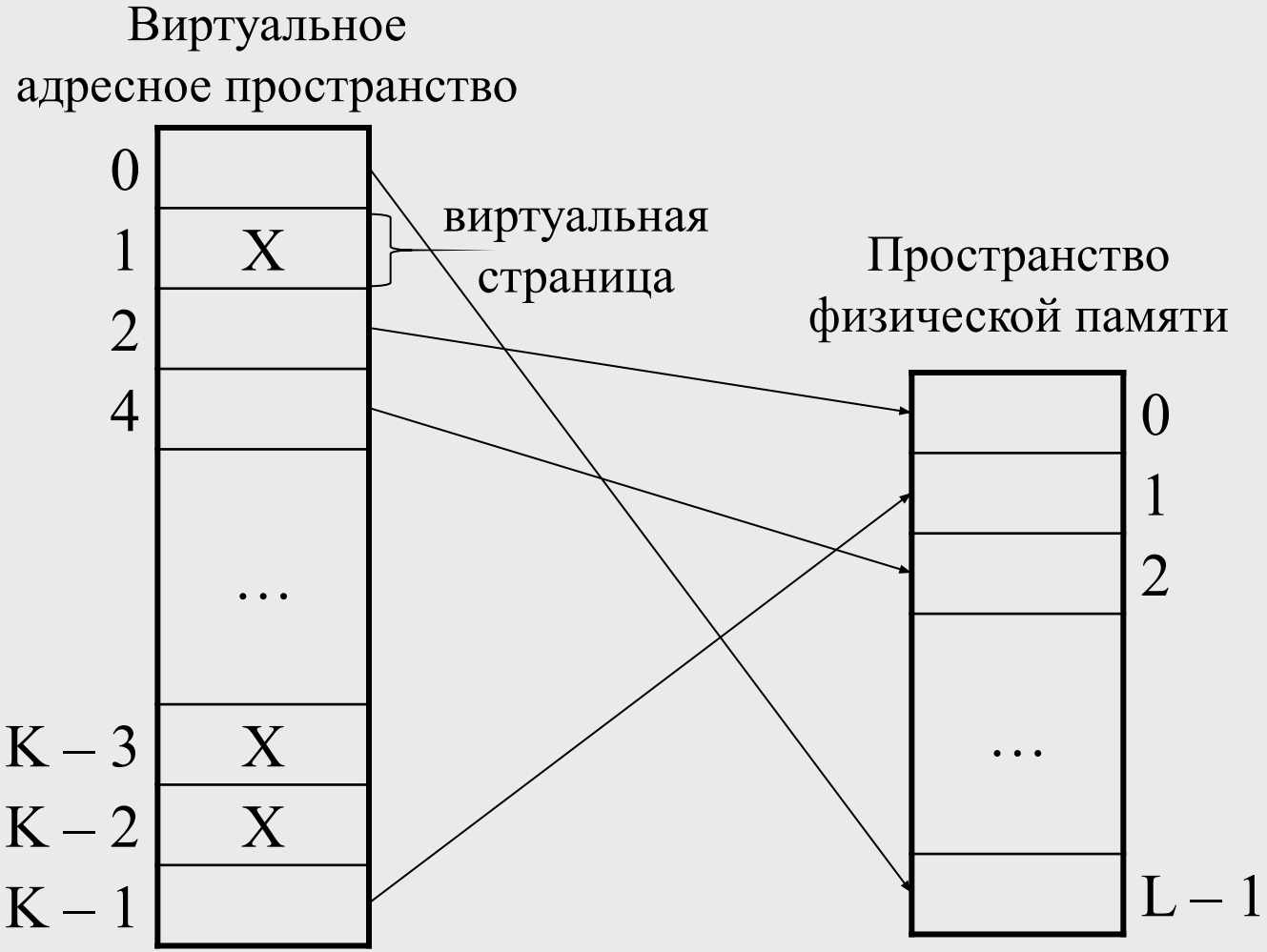
Недостатки

1. Ограничение размером физической памяти
2. Затраты на перекомпоновку

Страничное распределение

Основные концепции

Таблица страниц



Страничное распределение

Основные концепции

Таблица страниц — отображение номеров виртуальных страниц на номера физических.

Проблемы

1. Размер таблицы страниц (количество 4KB страниц при 32-х разрядной адресации — 1 000 000; любой процесс имеет собственную таблицу страниц)
2. Скорость отображения

Страничное распределение

Необходимые аппаратные средства

1. Полностью аппаратная таблица страниц (стоимость, полная перегрузка при смене контекстов, скорость преобразования)
2. Таблица страниц в ОЗУ + Регистр начала таблицы страниц в памяти (простота, управление смены контекстов, медленное преобразование)
3. Гибридные решения

Страничное распределение

Алгоритмы и организация данных

Размер и организация таблицы страниц ???

Модельная структура записи таблицы страниц

ε	δ	γ	β	α	Номер физической страницы

α — присутствие/отсутствие

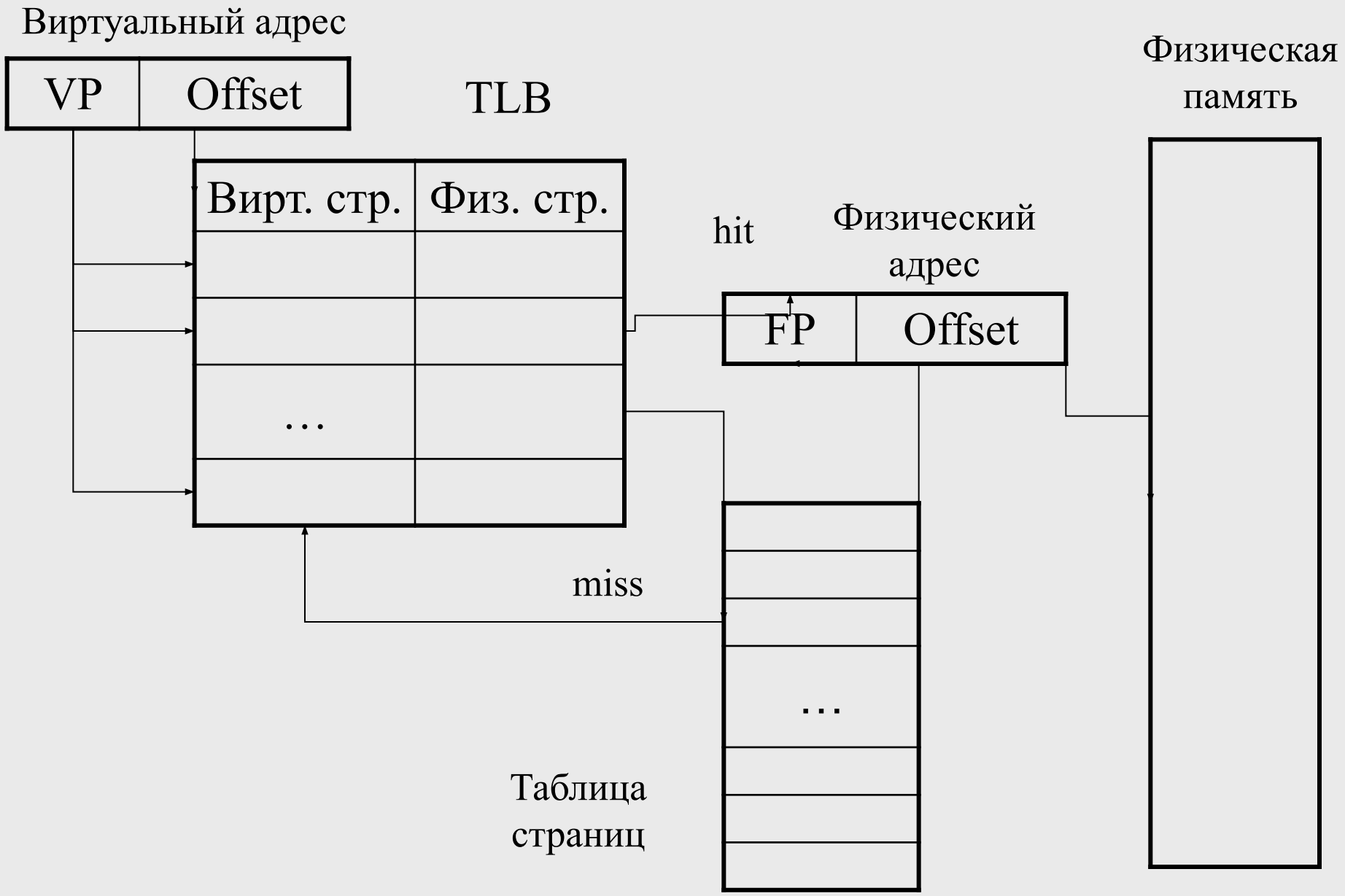
β — защита (чтение, чтение/запись, выполнение)

γ — изменения

δ — обращение (чтение, запись, выполнение)

ε — ……..

TLB (Translation Lookaside Buffer)



Иерархическая организация таблицы страниц

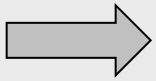
Проблема — размер таблицы страниц.

Объем виртуальной памяти современного компьютера — $2^{32}…2^{64}$ байт

Пример

$\mathbf{V_{вирт.}} = 2^{32}$

$\mathbf{V_{стр.}} = 2^{12}$ (4KB)

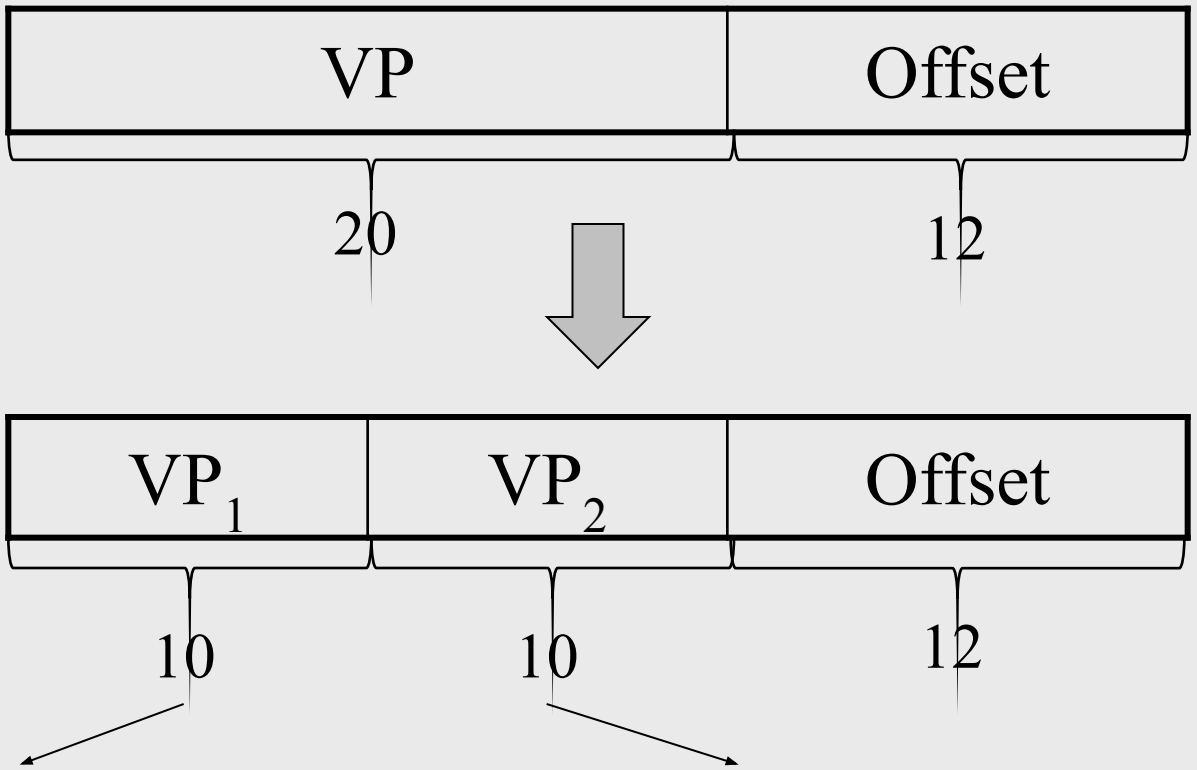


Количество виртуальных страниц — 2^{20} (**<u>много</u>**)

Решение — использование многоуровневых таблиц страниц (2^x, 3^x, 4^x)

Иерархическая организация таблицы страниц

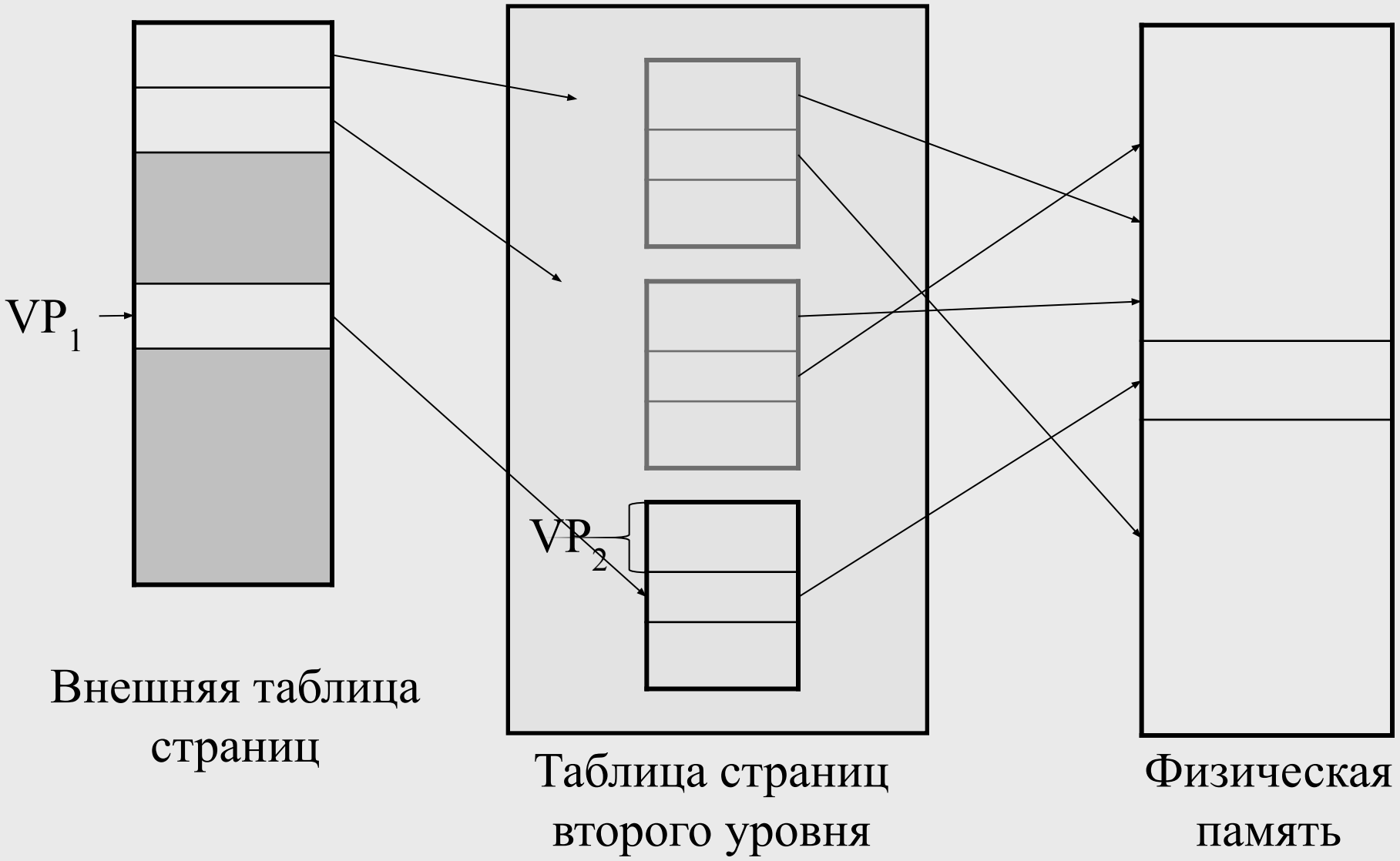
Двухуровневая организация



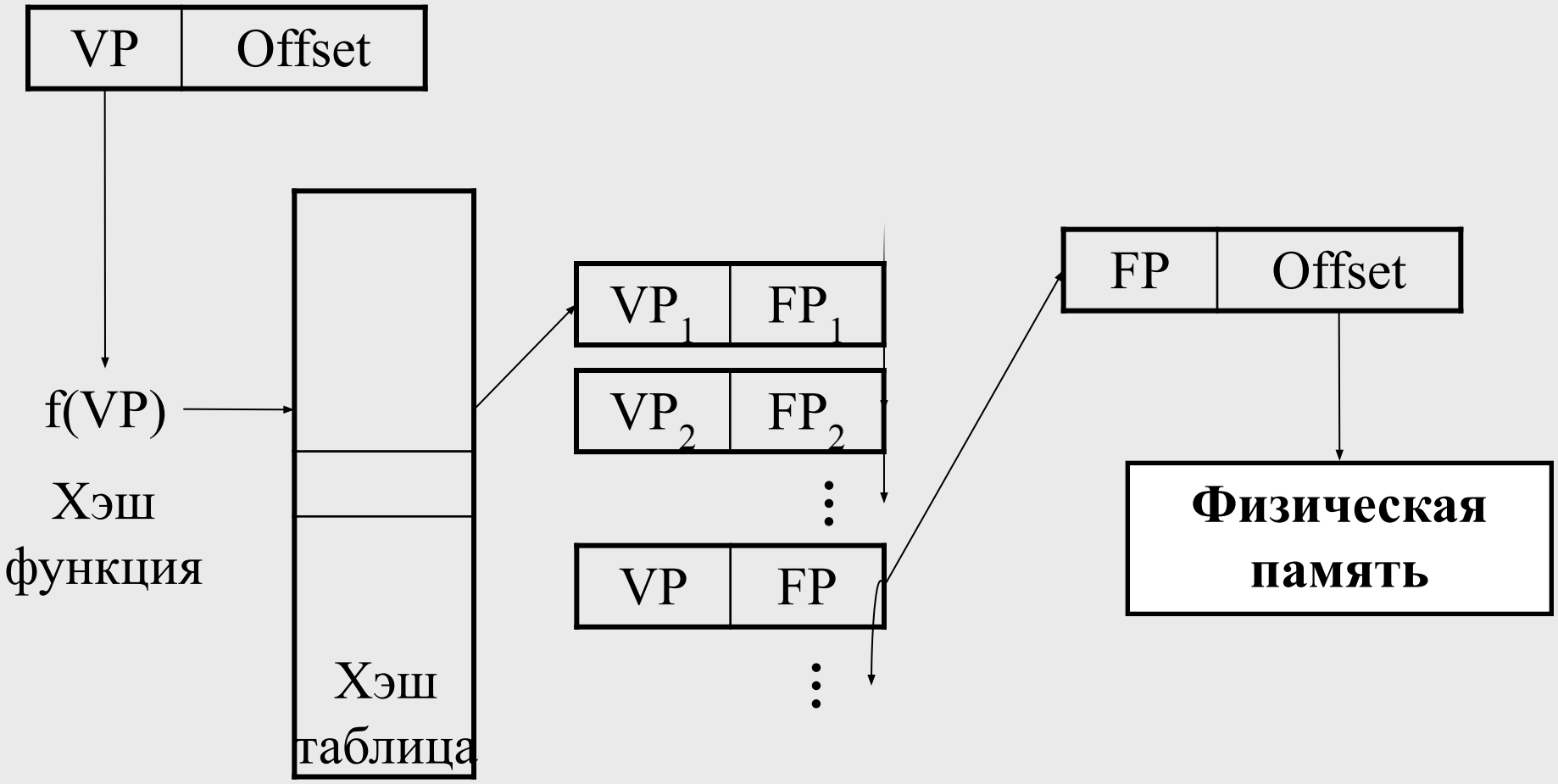
Индекс по «внешней» таблице страниц

Смещение по странице, указанной через VP_1

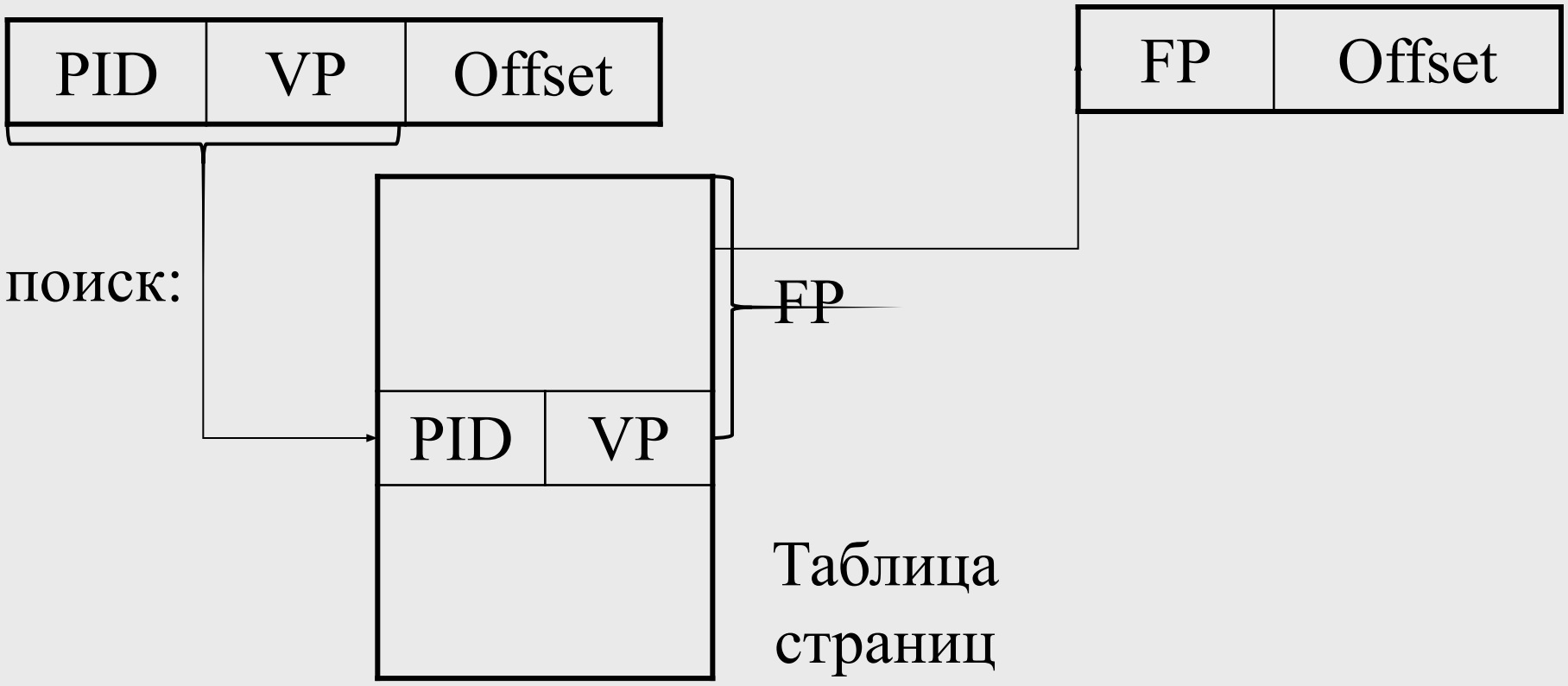
Иерархическая организация таблицы страниц



Использование хэш-таблиц (функция расстановки)



Инвертированные таблицы страниц



Проблема — поиск по таблице (хэширование)

Решение проблемы перезагрузки таблицы страниц при смене обрабатываемых ЦП процессов

Замещение страниц

Проблема загрузки «новой» страницы в память.
Необходимо выбрать страницу для удаления из памяти (с учетом ее модификации и пр.)

Алгоритм NRU (Not Recently Used — не использовавшийся в последнее время)

Используются биты статуса страницы в записях таблицы страниц

R — обращение
M — изменение
} устанавливаются аппаратно

обнуление — программно (ОС)

Замещение страниц

Алгоритм

1. При запуске процесса M и R для всех страниц процесса обнуляются
2. По таймеру происходит обнуление всех битов R
3. При возникновении страничного прерывания ОС делит все страниц на классы:
   - Класс 0: M=0, R=0
   - Класс 1: M=1, R=0
   - Класс 2: M=0, R=1
   - Класс 3: M=1, R=1
4. Случайная выборка страницы для удаления в непустом классе с минимальным номером

Замещение страниц

Стратегия: лучше выгрузить измененную страницу, к которой не было обращений как минимум в течение 1 «тика» таймера, чем часто используемую страницу

Замещение страниц

Алгоритм FIFO

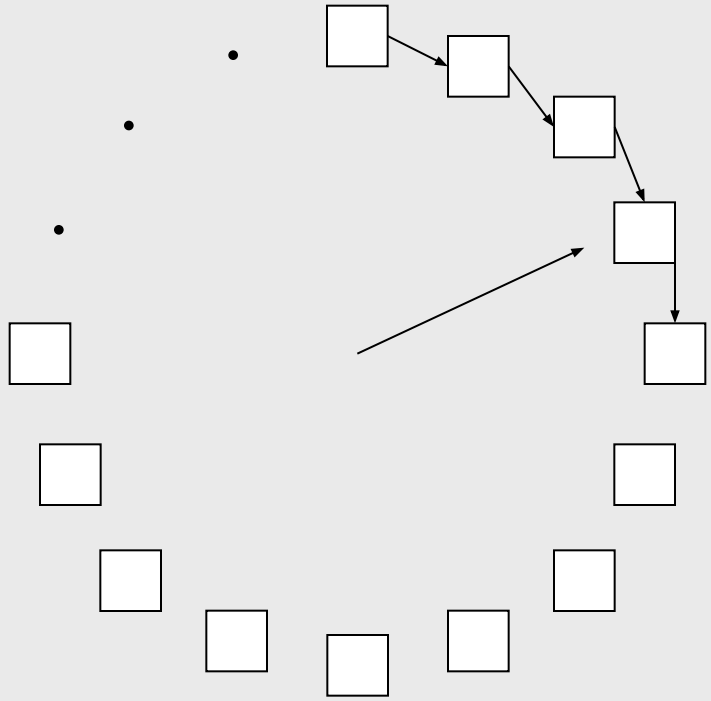
«Первым прибыл — первым удален» — простейший вариант FIFO. Проблема «справедливости»

Модификация алгоритма (алгоритм вторая попытка)

1. Выбирается самая «старая страница». Если R=0, то она заменяется
2. Если R = 1, то R обнуляется, обновляется время загрузки страницы в память (т.е. переносится в конец очереди). На п.1

Замещение страниц

Алгоритм «Часы»



1. Если R = 0, то выгрузка страницы и стрелка на позицию вправо
2. Если R = 1, то R обнуляется, стрелка на позицию вправо и на п.1

Замещение страниц

Алгоритм NFU (Not Frequently Used — редко использовавшаяся страница)

Для каждой физической страницы i – программный счетчик $Count_i$

0. Изначально $Count_i$ – обнуляется для всех i.
1. По таймеру $Count_i = Count_i + R_i$

Выбор страницы с минимальным значением $Count_i$

Недостатки

- «помнит» старую активность
- при большой активности, возможно переполнение счетчика

Замещение страниц

Модификация NFU — алгоритм старения

Модификация:

1. Значение счетчика сдвигается на 1 разряд вправо
2. Значение R добавляется в крайний левый разряд счетчика

Сегментная организация памяти

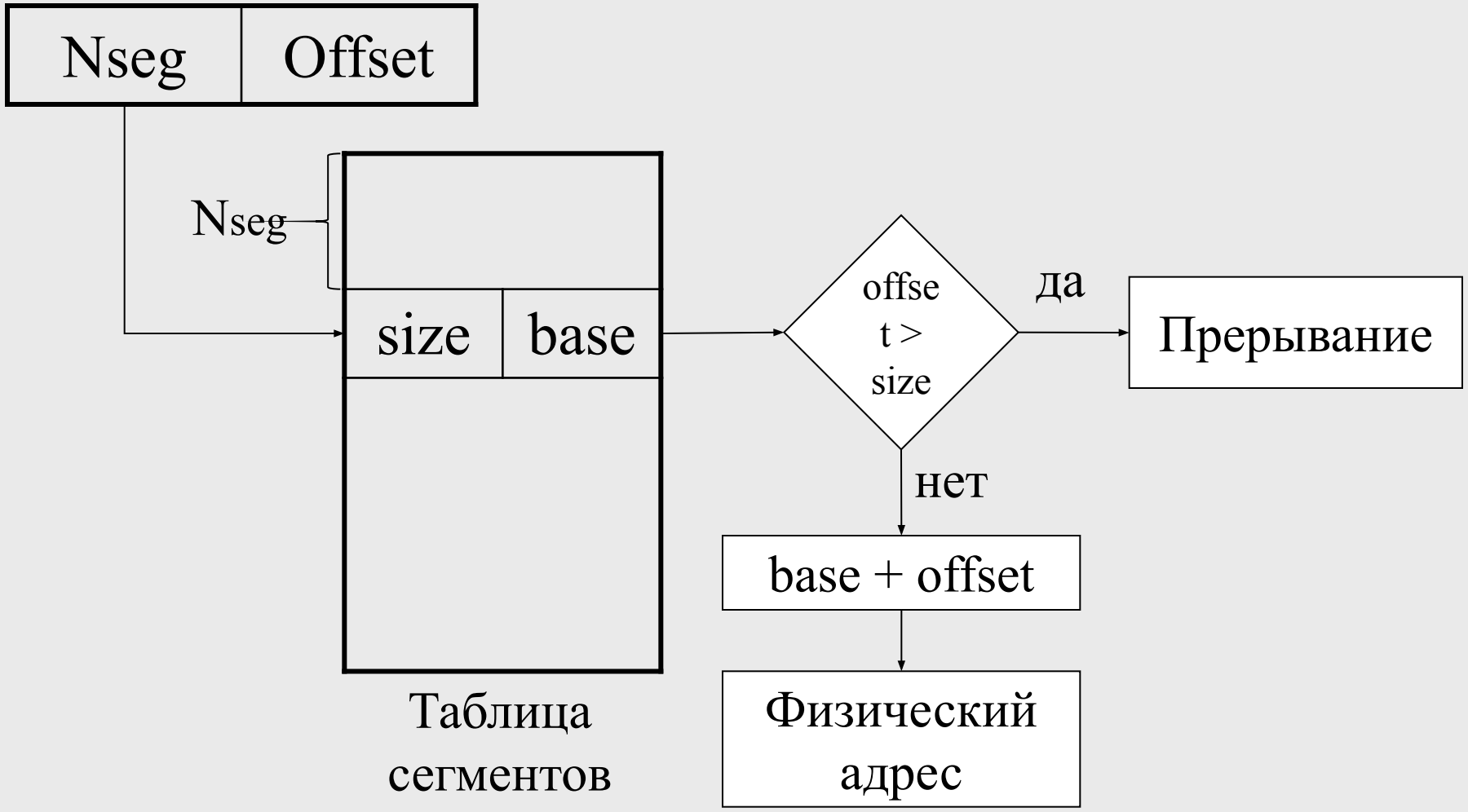
Основные концепции

- Виртуальное адресное пространство представляется в виде совокупности сегментов
- Каждый сегмент имеет свою виртуальную адресацию (от 0 до N – 1)
- Виртуальный адрес: <номер_сегмента, смещение>

Сегментная организация памяти

Необходимые аппаратные средства

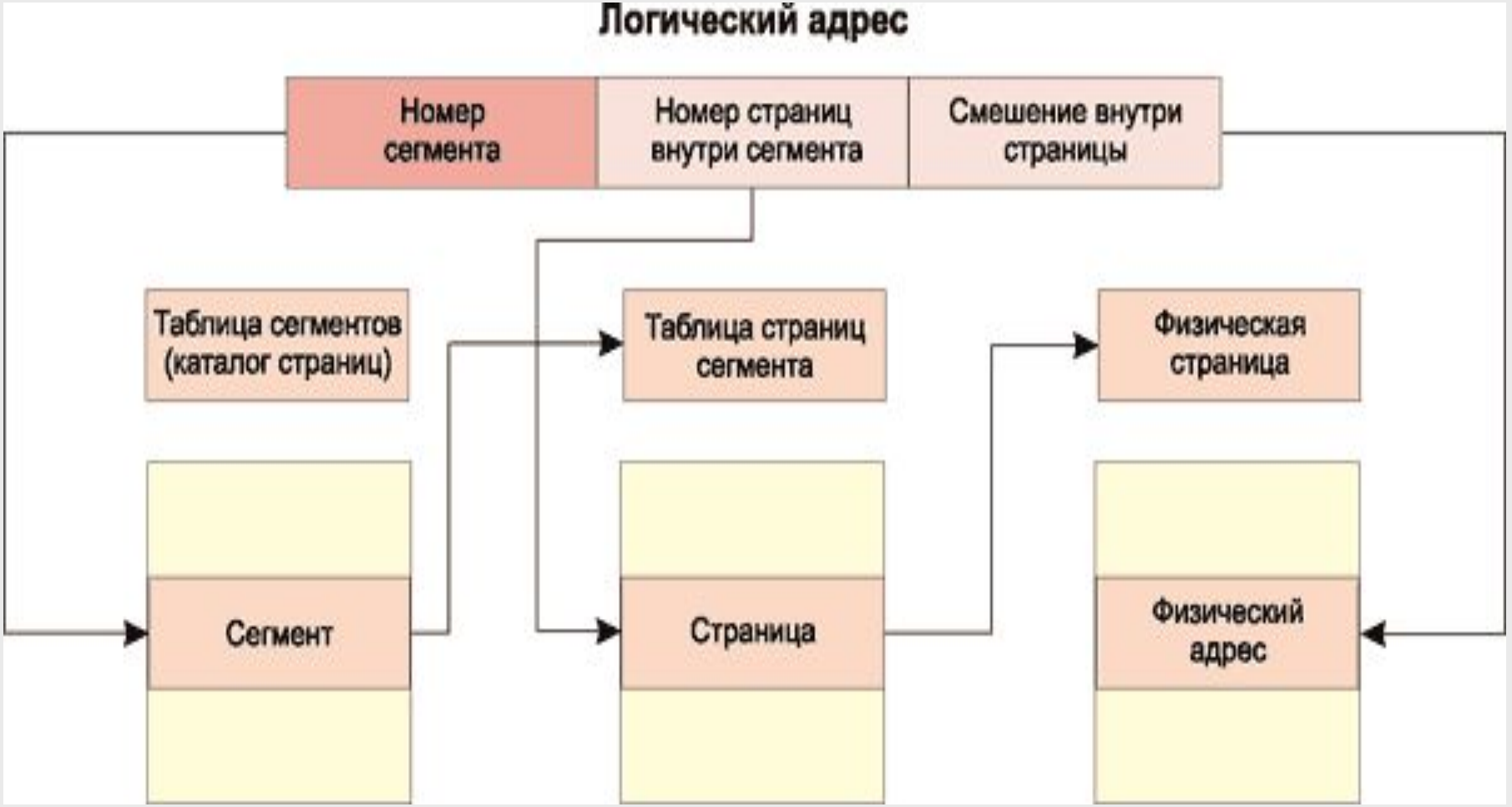
Виртуальный адрес:



Сегментно-страничная организация памяти

Основные концепции:

Nseg	Npage	Offset



Сегментно-страничная организация памяти

Модельный пример (Intel):

Виртуальный адрес:

Таблицы локальных дескрипторов (сегменты доступные для данного процесса) LDT (Local Descriptor Table)

Таблица глобальных дескрипторов (разделяемые между процессами сегменты) GDT (Global Descriptor Table)

Каждая запись LDT и GDT – полная информация о сегменте (адрес базы, размер и т.д.).

Сегментно-страничная организация памяти

Необходимые аппаратные средства