Управление памятью

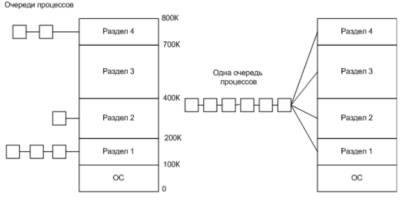
- Память — важнейший ресурс ЭВМ, требующий тщательного управления
- Иерархия памяти
  - регистры
  - кэш-память
  - ОЗУ
  - внешняя память
- Менеджер памяти (модуль управления памятью) — часть ОС, отвечающая за управление памятью
  - Следит, какая часть памяти используется, а какая свободна
  - Выделяет/освобождает память для процессов
  - Управляет обменом память-диск
- Типы менеджеров памяти:
  - Без перемещения (без использования внешней памяти)
  - С перемещением

Однозадачная система без подкачки на диск

- В каждый момент времени работает только одна программа
- Память разделяется между ОС и программой
- При запуске программы ОС копирует её с диска в память и передает ей управление. По окончании загружает другую программу поверх первой.

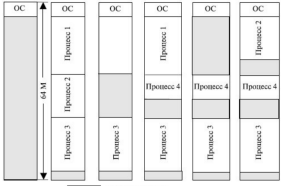
Многозадачная система с фиксированными разделами

- Память разделяется на несколько разделов, возможно разного размера
  - общая очередь процессов ко всем разделам
  - отдельные очереди к разделам
- Недостаток нескольких очередей — возможное простаивание раздела
- Алгоритмы планирования в случае одной очереди:
  - первый пришел — первый обслужен, циклический
  - выбирается задача, которая максимальной займет раздел; выбор осуществляется при освобождении раздела; возможна дискриминация «небольших» задач (решение — выделение специального раздела «небольшого» раздела).
- OS/MFT (Multiprogramming with Fixed number Tasks)



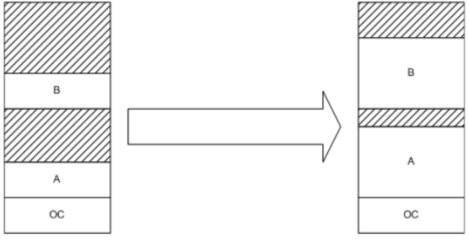
Многозадачная система с динамическим разделами

- Недостатки фиксированных разделов:
  - ограничение на количество задач;
  - неэффективное использование в случае небольших задач.
- Решение: динамическое количество разделов переменного размера
  - изначально вся память пуста (кроме занятой ОС)
  - процессы последовательно загружаются в память, начиная с адреса после ОС
- Недостаток: фрагментация — наличие большого числа небольших разделов, в которые не умещается процесс. Решается с помощью уплотнения (перемещение всех занятых участков, чтобы свободная память образовывала единую область) (трудоемкая задача)



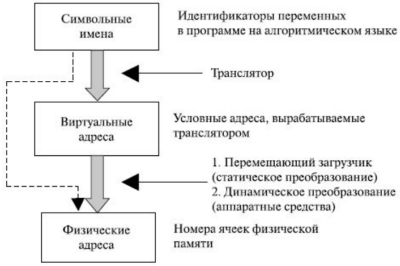
Многозадачная система с динамическим разделами: рост разделов

- Возможна ситуация, когда процессу нужен дополнительный объем памяти
- Возможное решение:
  - заранее выделять больше места
  - расширять раздел, если нет возможности для расширения — перемещать в новый раздел



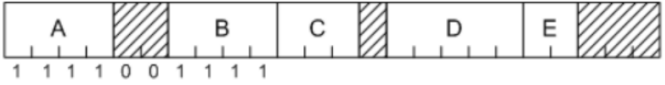
Дополнительные задачи ОС

- Использование сложных моделей размещения процессов в памяти ставит перед ОС дополнительные задачи:
  - настройка адресов
  - защита адресного пространства
- Глобальное решение: оснащение процессора специальными регистрами
  - базовый (указывает на начало адресного пространства процесса)
  - предельный (указывает конец или размер адресного пространства процесса)



Учет свободной памяти: битовые массивы

- Вся память разбивается на блоки
- Выделяется специальный массив, в котором каждый бит означает, что соответствующий по порядку блок занят (значение 1) или свободен (значение 0)
- При выделении памяти происходит поиск последовательности нулей нужной длины
- Недостаток — долгий поиск (искомая последовательность может пересекать границы слов и нужно многократное сравнение слов)

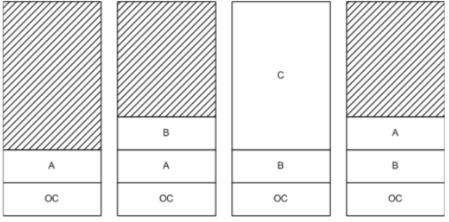


Учет свободной памяти: связные списки

- Создается список, каждый элемент которого содержит
  - признак занят (P) или свободен (H) фрагмент памяти
  - адрес начала фрагмента
  - длина фрагмента
- Алгоритмы выделения блока памяти:
  - первый подходящий участок
  - следующий подходящий участок, стартует не с начала списка, а с того места, на котором остановились в предыдущий раз (немного медленнее, показало моделирование)
  - самый подходящий участок (медленнее, сильная фрагментация маленькими фрагментами)
  - самый неподходящий участок, расчет на наличие больших остатков (моделирование показало, что работает не очень хорошо)

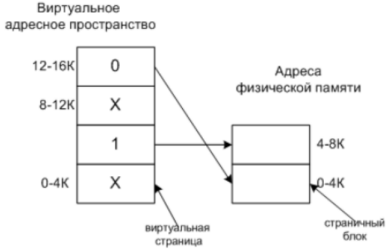
Использование внешней памяти

- Часто возникают ситуации, когда процесс надолго занят операцией ввода/вывода. В этом случае целесообразно выгрузить его во внешнюю память, освободив ОЗУ
- Своппинг (подкачка) — процесс целиком выгружается/загружается на диск (во внешнюю память)
- Виртуальная память — процесс частично может быть загружен в память



Виртуальная память: страничная организация

- Страничная организация памяти — способ реализации виртуальной памяти
- Страницы — части, на которые разбивается пространство виртуальных адресов
- Страничные блоки — единицы физической памяти
- Страницы имеют фиксированный размер. Передача между ОЗУ и диском осуществляется в страницах.
- Страничное прерывание — происходит, если процесс обратился к странице, которая не загружена в ОЗУ

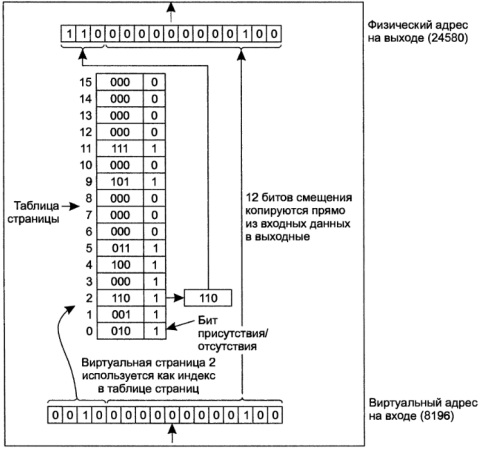


Таблицы страниц

- Таблицы страниц используются для хранения соответствия адресов виртуальной страницы и страничного блока
- Таблица страниц может быть размещена:
  - в ОЗУ (дешево, долго);
  - в аппаратных регистрах (буфер быстрого преобразования, TLB: translation lookaside buffer).

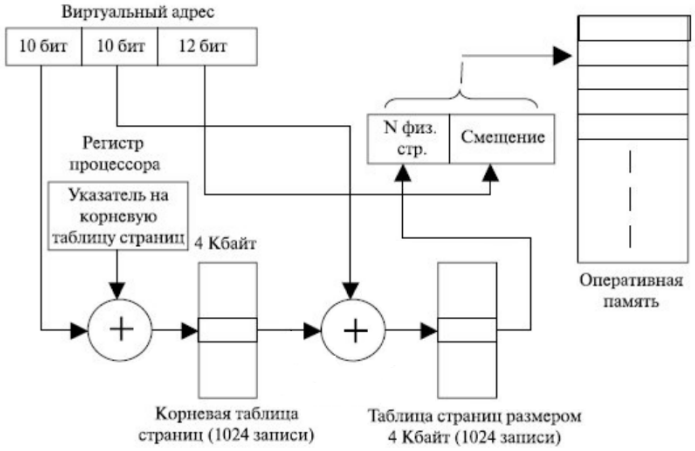
Использование таблицы страниц

Пример использования таблицы страниц в системе из 16 страниц по 4Кб



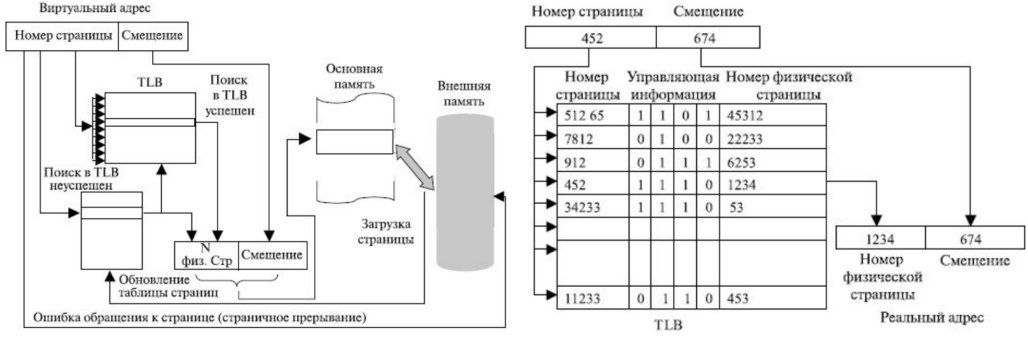
Многоуровневые таблицы страниц

Многоуровневые таблицы используются для решения проблемы большого размера таблицы страниц



TLB

- Буфер быстрого преобразования используется для повышения быстродействия преобразования виртуального адреса в физический
- Используется ассоциативный доступ (поиск одновременно во всех ячейках TLB)



Алгоритмы замещения страниц

- Во время возникновения ошибки отсутствия страниц операционная система должна выбрать страницу для удаления, чтобы освободить место для затребованной страницы
- Произвольный выбор страницы для удаления приводит к существенному снижению производительности
- Оптимальный алгоритм замещения:
  - каждая страница помечается количеством команд, которые должны быть выполнены до обращения к этой странице
  - выгружается страница с максимальной меткой.
- Алгоритм нереализуем, поскольку неизвестно, когда произойдет обращение к странице

Алгоритм NRU

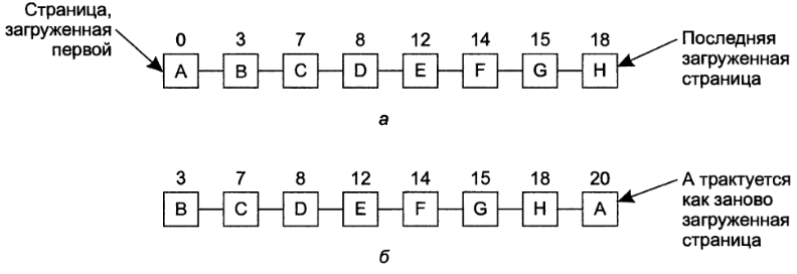
- **NRU** (Not Recently Used) — не использовавшаяся в последнее время
- Идея: если страница не использовалась в последнее время, значит, она не будет использоваться в дальнейшем
- Для каждой страницы устанавливаются два бита
  - $R = 1$, если было чтение из страницы; периодически R обнуляется
  - $M = 1$, если была модификация страницы
- Все страницы делятся на 4 класса:
  - класс 0: не было обращений и изменений $(R = 0, M = 0)$
  - класс 1: обращений не было, страница изменена $(R = 0, M = 1)$
  - класс 2: обращение было, страница не изменена $(R = 1, M = 0)$
  - класс 3: было обращение, страница изменена $(R = 1, M = 1)$
- Выбирается произвольная страница из непустого минимального класса

Алгоритм FIFO

- **FIFO** (First in — first out) — первым пришел, первым ушел
- Идея: если страница «старая», то к ней не было и не будет обращений
- Поддерживается список всех страниц, удаляется первая из этого списка
- Недостаток: можно удалить часто используемую страницу

Алгоритм «второй шанс»

- Идея: если страница «старая», то к ней не было и не будет обращений; дополнительно проверятся бит R
- Если $R = 0$, страница удаляется
- Если $R = 1$, R обнуляется и страница попадает в конец списка (обновляется время ее загрузки)



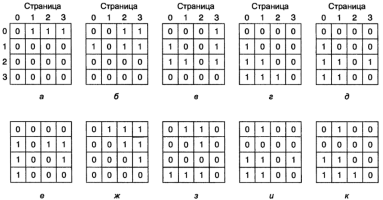
Алгоритм часов

- Алгоритм «второй шанс» не эффективен, требует модификации списка
- Идея: страницы хранятся к кольцевом списке и добавляется указатель на текущую страницу — кандидат для удаления
- По сути, отличается от «второго шанса» реализацией

Алгоритм LRU

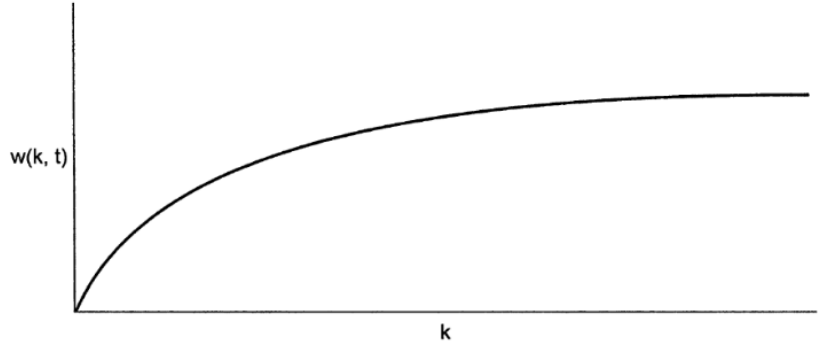
- LRU (Least Recently Used) — дольше всего не использовавшаяся страница
- Идея: страницы, к которым ранее не было обращений, не потребуются в дальнейшем
- Реализуем, но трудоемок:
  - либо поддерживать список страниц в нужном порядке
  - либо использовать аппаратный счетчик, который увеличивается после каждой команды
- Вариант аппаратной реализации: матрица обращения
  - для n страниц используется матрица $n * n$
  - при доступе к страничному блоку k всей k-ой строке присваиваются единицы, а k-му столбцу присваиваются нули
  - строка с наименьшим двоичным значением является дольше всего неиспользуемой

0 1 2 3 2 1 0 3 2 3



Модель рабочего набора

- В момент запуска процесса нужны страницы отсутствуют в памяти
- Через некоторое время в памяти скапливается достаточное количество необходимых процессу страниц и он начинает работать с небольшим количеством страничных прерываний
- Процессы характеризуются локальностью обращений, во время выполнения любой фазы процесс обращается к небольшой части собственных страниц
- Множество страниц, которое процесс использует в данный момент, называется рабочим набором
- Системы замещения страниц пытаются отслеживать рабочий набор процесса и обеспечивают его нахождение в памяти еще о запуска процесса — модель рабочего процесса
- k — количество обращений к памяти. Пусть для каждого момента времени t есть набор, включающий все страницы, которым было k последних обращений к памяти
- $w(k, t)$ — рабочий набор



Модель рабочего набора: wsclock

- Из-за асимптотического поведения $w(k, t)$ содержимое рабочего набора нечувствительно к значению k; существует большое количество значений k, при которых рабочие наборы одинаковы
- Для реализации механизмов рабочего набора операционная система должна отслеживать, какие страницы в нем находятся
- Возможная реализация — использовать механизм «старения» страниц
  - у каждой страницы есть счетчик нахождения в памяти (время последнего использования)
  - старший бит счетчика может означать присутствие страницы в рабочем наборе
  - если за n последовательных тактов обращения к странице не происходит, она убирается из рабочего набора
- **wsclock** — алгоритм «часы» с дополнительной проверкой присутствия страницы в рабочем наборе

Linux 2.6 Алгоритм PFRA, логика утилизации