БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Кафедра информатики

Факультет НиДО

Специальность ИиТП

Контрольная работа № 1

по дисциплине «Операционные системы и среды»

Выполнил студент: Дегтярев А.А.

группа 393551

Зачетная книжка № 902021-26

Минск 2016

**Вариант 2.**

Подсистема памяти. Адресное пространство.

Цель работы – более детальное ознакомление с концепциями, механизмами, алгоритмами и средствами распределения ресурсов вычислительной системы, а также управления вводом-выводом.

Стоит начать с того, что в большинстве современных вычислительных устройств используется несколько разных уровней памяти, устроенной в иерархическую структуру. Сущность необходимости построения иерархической памяти — необходимость обеспечения вычислительной системы (отдельного компьютера или кластера) достаточным объёмом памяти, как оперативной так и постоянной. Учитывая неоднородность периодичности обращения к конкретным записям (внутренним регистрам процессора, кэш-памяти, страницам и файлам) применяются различные технические решения, имеющие отличные характеристики.   


Часто выделяют 4 основных уровня иерархии

* Внутренняя память процессора (регистры, организованные в регистровый файл и кэш процессора).
* ОЗУ системы (RAM) и вспомогательных карт памяти.
* Накопители с «горячим» доступом — или вторичная компьютерная память. Жесткие диски и твердотельные накопители, не требующие длительных (секунды и больше) действий для начала получения данных.
* Накопители, требующие переключения носителей — или третичная память. Сюда относятся магнитные ленты, ленточные и дисковые библиотеки, требующие длительной перемотки либо механического (или ручного) переключения носителей информации.

В большинстве современных ПК используется следующая иерархия памяти:

1. Регистры процессора, организованные в регистровый файл — наиболее быстрый доступ (порядка 1 такта), но размером лишь в несколько сотен или, редко, тысяч байт.
2. Кэш процессора 1го уровня (L1) — время доступа порядка нескольких тактов, размером в десятки килобайт
3. Кэш процессора 2го уровня (L2) — большее время доступа (от 2 до 10 раз медленнее L1), около полумегабайта или более
4. Кэш процессора 3го уровня (L3) — время доступа около сотни тактов, размером в несколько мегабайт (в массовых процессорах используется недавно)
5. ОЗУ системы — время доступа от сотен до, возможно, тысячи тактов, но огромные размеры в несколько гигабайт, вплоть до сотен.
6. Дисковое хранилище — многие миллионы тактов, если данные не были закэшированны или забуферизованны заранее, размеры до нескольких терабайт
7. Третичная память — задержки до нескольких секунд или минут, но практически неограниченные объёмы (ленточные библиотеки).

На программном уровне, считается, что память делится на два уровня, оперативную память и дисковые накопители, хотя в ассемблерных языках и ассемблерно-совместимых (типа C) существует возможность непосредственной работы с регистрами. Получение преимуществ от иерархии памяти требует совместных действий от программиста, аппаратуры и компиляторов.

Программисты отвечают за организацию передачи данных между дисками и памятью (ОЗУ), используя для этого файловый ввод-вывод; Современные ОС также реализуют это как подкачку страниц.

Аппаратное обеспечение отвечает за организацию передачи данных между памятью и кэшами.

Оптимизирующие компиляторы отвечают за генерацию кода, при исполнении которого аппаратура эффективно использует регистры и кэш процессора.

По удаленности / доступности для процессора память можно разделить на:

**Первичная память** (сверхоперативная, СОЗУ) — доступна процессору без какого-либо обращения к внешним устройствам.

регистры процессора (процессорная или регистровая память) — регистры, расположенные непосредственно в АЛУ;

кэш процессора — кэш, используемый процессором для уменьшения среднего времени доступа к компьютерной памяти. Разделяется на несколько уровней, различающихся скоростью и объёмом (например, L1, L2, L3).

**Вторичная память** — доступна процессору путём прямой адресации через шину адреса (адресуемая память). Таким образом доступна оперативная память (память, предназначенная для хранения текущих данных и выполняемых программ) и порты ввода-вывода (специальные адреса, через обращение к которым реализовано взаимодействие с прочей аппаратурой).

**Третичная память** — доступна только путём нетривиальной последовательности действий. Сюда входят все виды внешней памяти — доступной через устройства ввода-вывода. Взаимодействие с третичной памятью ведётся по определённым правилам (протоколам) и требует присутствия в памяти соответствующих программ.

**Виды адресов**

**Реальный** – местоположение данных в реальной памяти

**Виртуальный –** адрес в виртуальный памяти при использование преобразуется в реальный

**Логический** – адрес определяемый программой путем преобразования ключа данных по некоторому алгоритму. Адрес в виртуальной памяти.

**Адрес устройства** – логический адрес состоящий из номера канала и устройства в канале

**Физический** – число идентифицирующее ячейку или область физической памяти. Уникальное имя однозначно определяющее конкретное устройство;

**Адресация памяти**

**Реальная адресация(простая flat)** памяти(в соответствии с физическим расположением данных); Делает адресацию на больших пространствах проблематичным.

**Расширенная (extended)** доступс использования особого формата команды, позволяющего использовать больший диапазон адресов

**Виртуальное адресное пространство** - способ адресации не отражает физического расположения её данных. Каждая программа – ограниченное непрерывное поле адресов. Адреса

**Ассоциативная** – без указания точного местоположения, задается значение определенного поля, идентифицирующее эти данные.

**Особенность исполнение программ**

**Статическая адресация –** соответствен между виртуальными и физическими адресами устанавливается до начала и не меняется в ходе выполнения программы

**Динамическая** – адреса преобразовываются в процессе исполнения программы.

**Кодирование адресов**

**Явное –** путемявного задания адресов

**Неявная –** путем использования нескольких операндов или адресов операндов фиксированных для команды, не требующего явного указания

**Абсолютная** – с абсолютным адресом (адрес на машинном языке индентифицирующий ячейку памяти)

**Символический** – с символическим адресом(выраженный в удобной для программирования форме(имя переменной))

**Вычисление адреса**

**Непосредственный(прямой) –** с использованием прямого адреса( в виде некоторого значения. Все ячейки на одной странице)

**Косвенная –** с косвенным адресом(адрес операнда)

**Регистровая –** задание адресов операндов в регистрах

**Базисная –** вычисление адреса в машинных командах относительно содержимого регистра, указанного в качестве базавого

**Базовая(Относительная) – адрес = базовый адрес + смещение.** Базовый адрес хранится в специальном регистре.

**Индексная** – адрес = базовый адрес + индексный регистр (массивы)

**Стековая –** адресация посредством регистра – указателя стека

**Самоопределяющаяся –** относительно адреса команды в которой он находится

Особенно стоит остановится на **виртуальной памяти –** метод позволяющий выполнять программы требующие большего кол-ва памяти путем перемещения частей программы между основной памятью и вторичным хранилищем; Что принципе сделало возможным существование **многозадачных ОС**. Применение ВП позволяет освободить программиста от необходимости вручную управлять загрузкой частей программы в память и согласовывать использование с другими программами; изолирует исполняемые программы друг от друга в непересекающихся пространствах. Аппаратно поддерживается в большинстве современных процессоров. (Кроме систем критичных к длительности отклика).

В большинстве современных операционных систем виртуальная память организуется с помощью страничной адресации. Оперативная память делится на страницы: области памяти фиксированной длины, которые являются минимальной единицей выделяемой памяти. Исполняемый процессором пользовательский поток обращается к памяти с помощью адреса виртуальной памяти, который делится на номер страницы и смещение внутри страницы. Процессор преобразует номер виртуальной страницы в адрес соответствующей ей физической страницы. Если ему не удалось это сделать, то требуется дозаполнение буфера путём **обращения к таблице страниц** (так называемый Page Walk), что может сделать либо сам процессор, либо операционная система. Если страница была выгружена из оперативной памяти, то операционная система подкачивает страницу с жёсткого диска в ходе обработки события Page fault. (свопинг, подкачка страниц). При запросе на выделение памяти операционная система может «сбросить» на жёсткий диск страницы, к которым давно не было обращений. Критические данные (например, код запущенных и работающих программ, код и память ядра системы) обычно находятся в оперативной памяти.

Механизм организации виртуальной памяти, при котором виртуальное пространство делится на части произвольного размера — сегменты. Этот механизм позволяет, к примеру, разбить данные процесса на логические блоки. Для каждого сегмента, как и для страницы, могут быть назначены права доступа к нему пользователя и его процессов. При загрузке процесса часть сегментов помещается в оперативную память (при этом для каждого из этих сегментов операционная система подыскивает подходящий участок свободной памяти), а часть сегментов размещается в дисковой памяти. Сегменты одной программы могут занимать в оперативной памяти несмежные участки. Во время загрузки система создает таблицу сегментов процесса (аналогичную таблице страниц), в которой для каждого сегмента указывается начальный физический адрес сегмента в оперативной памяти, размер сегмента, правила доступа, признак модификации, признак обращения к данному сегменту за последний интервал времени и некоторая другая информация. Если виртуальные адресные пространства нескольких процессов включают один и тот же сегмент, то в таблицах сегментов этих процессов делаются ссылки на один и тот же участок оперативной памяти, в который данный сегмент загружается в единственном экземпляре. Система с сегментной организацией функционирует аналогично системе со страничной организацией: время от времени происходят прерывания, связанные с отсутствием нужных сегментов в памяти, при необходимости освобождения памяти некоторые сегменты выгружаются, при каждом обращении к оперативной памяти выполняется преобразование виртуального адреса в физический. Кроме того, при обращении к памяти проверяется, разрешен ли доступ требуемого типа к данному сегменту.

Виртуальный адрес при сегментной организации памяти может быть представлен парой (g, s), где g — номер сегмента, а s — смещение в сегменте. Физический адрес получается путём сложения начального физического адреса сегмента, найденного в таблице сегментов по номеру g, и смещения s.

Недостатком данного метода распределения памяти является фрагментация на уровне сегментов и более медленное по сравнению со страничной организацией преобразование адреса.

Основные преимущества использование страничной организации памяти:

* поддержка изоляции процессов и защиты памяти путём создания своего собственного виртуального адресного пространства для каждого процесса
* поддержка изоляции области ядра от кода пользовательского режима
* поддержка памяти «только для чтения» и неисполняемой памяти
* поддержка отгрузки давно не используемых страниц в область подкачки на диске (см. свопинг)
* поддержка отображённых в память файлов, в том числе загрузочных модулей
* поддержка разделяемой между процессами памяти, в том числе с копированием-по-записи для экономии физических страниц
* поддержка системного вызова fork() в ОС семейства UNIX
* поддержка зашиты от исполнения данных( PAE ) семейства Windows