1.Ос-комплекс программ для управления ресурсами ПК и взаимодействия с пользователем

Основное назначение ос: эффективное управление ресурсами системы

Основные ресурсы ОС: Процессор, память, уст ввода\вывода

Функции: 1) взаимодействие между пользователем и системой 2) разделение аппаратных средств между пользователями 3) планировать доступ к общим ресурсам 4) выполнение операций ввода\вывода 5) восстановление в случае сбоя

2.1поколение(1940-1955): Программирование на уровне машинных кодов

2поколение(1955-1960) – первый ассемблер(fortran), появление стандартных библиотек, монопольный режим, ресурсы расходуются неэффективно.

3поколение(1960-1980)- мультизадачность, ос реального времени, первые виртуальные машины и мультипрограммирование.

4поколение(1980-сегодня) – ОС для пк, графические интерфейсы, мультипроцессорные ОС, мобильные и встроенные ОС

Типы ОС: 1)мейнфреймы – большой сервер, со значительным количеством ресурсов, нужен зачастую для научной деятельности 2)Серверные – одновременно обслуживают много людей 3)Персональные – для пк с многозадачным режимом работы. 4) встроенные – в телевизорах, видеомагнитофонах. 5) мобильные – ios,android.

3. По взаимодействию с пользователем: 1) пакетный – формируется пакет шагов, выполняется без пользователя 2)Интерактивный – взаимодействие постоянное с пользователем через интерфейс.

По кол-ву задач: 1) однопользовательский однозадачная – работает только один полдьзователь и использует все ресурсы пк для одной задачи. 2) однопользовательская мультзадачная – также один пользователь, но есть возможность параллельного выполнения нескольких задач. 3) многопользовательская мультизадачная – снабжена многотерминальной средством поддержки.

4. Мультизадачный режим работы: 1) классическое мультипрограммирование - способ организации вычислительного процесса при к-ром на одном процессоре попеременно выполняется сразу несколько программ. 2) Корпоративная мультизадачность – след задача выполняется только после текущей, когда она отдаст процессорное время.3) Квантование по времени - ос сама передает управление от одной задачи другой в случае завершения операций ввода-вывода, возникновения событий в аппаратуре компьютера, истечения таймеров и квантов времени, или же поступлений тех или иных сигналов от одной программы к другой. 4) режим реального времени – в зависимости от поступивших запросов, последний запрос – наивысший приоритет.

5. Архитектурные особенности ОС: 1) монолитное ядро – вся ос работает в пространстве ядра, но для этого нужно увеличить размер ядра и размер ОС. 2) Многоуровневая – делится на несколько уровней и каждый уровень отвечает за свою функциональность. 3) Микроядерные – ядро отвечает только за базовые функции, все остальное выносится в пользовательское пространство, ядро очень компактное. 4) гипервизор – развертывание ПО на физ оборудовании с использованием виртуализации. 5) Системы с ядром смешанного типа.

6.Процесс – совокупность находящихся под управлением ОС:1) последовательность исполняющихся команд 2) соответствующие ресурсы 3) текущее состояние

С целью чтобы ОС распологала всей необходимой информацией, создается специальная информационная структура – дескриптор или блок управления процесса. Состоит:1) ID процесса 2)Тип процесса 3) приоритет процесса 4) переменную состояния 5) защищенную область памяти 6) информацию о ресурсах 7) место для организации общения между процессами 8)параметр времени запуска.

7. Нити в отличие от процессов, работают в контексте одного процесса – родителя. Реализация потоков : 1) в пространстве ядра – Достоинства: 1) планирование работы нескольких потоков одного и того же процесса на нескольких ядрах 2) мультпрограммирование в рамках всех процессов 3) про блокировании одного из потоков можно выбрать другой поток этого же процесса 4) процессы ядра могут быть много поточными. Недостатки: необходимо двухкратное переключение между режимом пользователь-ядро, ядро-пользователь для передачи управления от одного потока другому. 2) В пространстве пользователя: Достоинства - 1) реализуется собственными средствами в ОС 2) высокая произоводительность, нет необходимости переключаться в режим ядра и обратно 3) ядро о потоках управляет однопоточными процессами 4) применение любого вида алгоритма планирования и исполнения потоков. Недостатки: 1) системный вызов блокирует все потоки процесса 2)Не возможность работать в многопроцессорном режиме 3) не возможность создать планировщик по таймеру для поочередному выполненияя потоков.

8. Необходимость планирования возникает из необходимости эффективно управлять ограниченным набором ресурсов и обеспечения комфортных условий решения задач.

Задачи планирования возникли из-за первых систем с режимом мультипрограммной работы.

Моменты планирования: 1)новый процесс переходит в состояние готовности 2) процесс переходит из состояния выполнения в состояние готовности 3)переходит из состояния выполнения в состояние ожидания 4) из состояния ожидания в готовность 5)из выполнения в завершение

При рассмотрении алгоритмов планирования различается три операционных среды:

1) пакетной обработки 2) интерактивную 3) среду реального времени

9.Задачи алгоритмов планирования -1 1) Для всех систем: 1) равнодоступность –всем процессам справедливое доля времени цп 2) принуждение к определенной политке – наблюдение за выполнение установленной политики 3) баланс –поддержка всей загруженности всех составных частей системы. 2) Пакетные системы 1) производительность –макс кол-ва заданий в час 2) оборотное время – минимизация времени между создания задачи и завершением 3) использование ЦП – постоянная загруженность ЦП.

Задачи алгоритмов планирования- 2 1)Интерактивные системы : 1) время отклика – быстрый ответ на запросы 2) пропорциональность – оправдание пользовательских надежд 2) Системы реального времени : 1) соблюдение предельных сроков – предотвращение потери данных 2) предсказуемость – предотвращение ухудшения качеств в мульт-мед системах.

10.Алгоритм планирования FCFS И JSF :

1) FCFS –FIRST COMES FIRST SERVED. Достоинства: Простота как программирования, так и понимания. Все происходит с помощью единого связанного списка, выбор след процесса происходит из извлечения одного процесса из начала очереди, добавления процесса происходит к концу списка. Недостатки : монополизация времени одним из процессов, это не эффективное расходывание ресурсов.

2)JSF- JOB SHORTEST FIRST: Планировщик выбирает самое короткое задание и выполняет его первым. Оптимален только если все задачи доступны одновременно.

11.Алгоритм RR(round robin): каждому процессу выделяется время работы, если он не успевает за этого время выполниться , то ресурсы выделенные ЦП передаются другому процессу с таким же выделенным временем. Эффективность существенно зависит от размера кванта времени.

Приоритетное планирование: каждому процессу присваивается значение приоритетности и запускается тот процесс, который находится в состоянии готовности и имеет наивысший приоритет. С каждым прерыванием процесса его приоритет падает, если его приоритет упадет ниже приоритета следующего произойдет переключение процессов. Приоритеты присваиваются системой в статическом или в динамическом режиме.

12. Гарантированное планирование – если в системе зарегестрированы н пользователей каждый пользователь получет 1\н от мощности процессора. Так же и в однопользовательской системе с н работающими процессами.

Справедливое планирование – если пользователь 1 запустит 9 процессов, а пользователь 2 1 процесс , то 1 получит 90% процессорного времени, а второй только 10%.

Система реального времени – события в ней периодитческие или апериодические, проводится тест по которому вычисляется какое количество процессорного времени выделить тому или иному процессу.

13. Зачем нужно межпроцессорное взаимодействие: 1) Повышение производитесльности и рациональное использование ресурсов 2) совместное использование данных 3) модульный подход проектирования системы 4) удобство и эффективность реализации алгоритмов и механизмов взаимодействия с пользователем. Классификация IPC – 1) По уровню локализации – локальное, удаленное 2) по направлению передачи – одно или двунаправленное 3) по характеру доступа – открытое закрытое. Проблемы межпроцессового взаимодействия: 1) проблема синхронизации: 1) независимые процессы должны быть независимыми 2) обеспечение гарантирование доставки данных 3) Исключение взаимных блокировок при доступе к общим ресурсам 2)Проблема безопастности – среда передачи между процессорами является уязвимым местом.

14. Методы межпроцессового взаимодействия. Каналы – некая сущность в которую можно помещать и извлекать данные, для чего служать два файловых дескриптора ассоциированных с каналом, один для записи другой для чтения. Создание канала – системный вызов PIPE(). Отличчительные свойства канала : 1) к неименованному каналу невозможен доступ по имени, только через файловые дескрипторы. 2) канал не существует вне процесса 3) канал реализует модель последовательного доступа к данным(FIFO). Именнованный канал(FIFO файл) расширяет свою область применения за счет того что к нему можно подключиться любой процесс в любое время, это возможно благодаря наличия у них имен. Подключение происходит посредством системного вызова open (), но есть несколько правил :1) когда открыт файл для чтения, он блокируется пока процесс не откроет его же на запись.2) когда открыт он для записи, он блокирован, пока тот же процесс не откроет его для чтения.

Сигналы – уведомление процесса о наступлении события в системе, их может посылать сама система или другой процесс. Сигналы посылаемые ОС уведомляет о наступлении строго предопределенных ситуаций и каждой такой ситуации представлен свой сигнал. Сигналы предопределенны в файле signal.h .

15. Очереди сообщений представляют собой связный список в адресном пространстве ядра. Сообщения могут посылаться в очередь по порядку и доставаться из очереди несколькими разными путями. Каждая очередь сообщений однозначно определена идентификатором IPC. Очереди сообщений как средство межпроцессной связи позволяют процессам взаимодействовать, обмениваясь данными. Данные передаются между процессами дискретными порциями, называемыми сообщениями. Процессы, использующие этот тип межпроцессной связи, могут выполнять две операции:послать или принять сообщение.

16.Разделяемая память – является самым быстрым средством обмена данными между процессами. В других средствах межпроцессового взаимодействия обмен информациями происходит через ядро. А тут используется обмен информации через общий для процессов сегмент памяти без использования системных вызовов ядра. Сегмент разделяемой памяти подключается в свободную часть виртуального адрессного пространства процесса. Два разных процесса могут иметь разные адресса одной и той же ячейки, подключенной к разделяемой памяти.

17. Сокет - универсальный способ взаимодействия процессов, используется как для локального взаимодействия, так и для обмена данных с удаленными однородными системами. Бывают unix 1) реализуемые между файлами спец типа и предназначены для организации локального обмена между процессами 2) сетевые сокеты, для сетевого взаимодействия между процессами удаленных систем 3) потоковые, реализуют соединения «от точки до точки», надежная передача данными. 4)Ориентированные на быструю передачу данных без установки соединения.

18. Проблемы синхронизации : 1) независимые процессы должны оставаться – независимыми 2)Обеспечить гарантированную доставку данных 3) исключение возможность взаимной блокировок при доступе к общим ресурсам.

Важным понятие в проблеме межпроцессного взаимодействия является состояние состязания – ситуация, в которой два или более процесса считывают и записывают данные одновременно и конченый результат зависит от того, какой из них был первым. Для предотвращения такого состояния и любой другой ситуации, связанной с совместным использованием памяти, файлов и чего-либо ещё, используется взаимное исключение – запрет одновременной записи и чтения разделенных данных более чем одним процессом. Для этого необходимо выполнение 4 условий бесконфликтного доступа к общим данным:

• два процесса не должны одновременно находиться в критических областях;

• в программе не должно быть предположений о скорости и количестве процессоров;

• процесс, находящийся вне критической области, не может блокировать другие процессы;

• невозможна ситуация, в которой процесс вечно ждет попадания в критическую область.

19.1) Запрет прерываний – после входа в критическую область происходит запрет на прерывание и разрешает после выхода из неё. Недостатки – Возможность краша процесса, снижение эффективности приема в мультипроцессорной системе с общей памятью, потеря событий из-за длительного запрета на прерывания.

2)Блокирующие переменные – происходит проверка свободен ли ресурс, если да, пользуемся и занимаем его, если нет ждем и пытаемся. Недостатки – Гонки при попытке доступа к блокирующим переменным нескольких процессов, процесс тратит бесполезно время на опрос ресурса.

3)Строгое чередование – проверка постоянная переменную пока она не примет нужное нам значение, спин блокировка. После каждого процесса переменная меняет значение для преступление след. Процесс.

4)Поддержка критической секции аппаратурой

20.Семафоры – одна из форм IPC, используется для синхронизации доступа нескольких процессов к разделяемым ресурсам.

Классический семафор – вид числовой переменной, над которой определены две неделимые операции – 1) уменьшение ее значения с возможным блокированием процесса 2) увеличение значения с возможным разблокированием из ранее заблокированных процессов.

Процесс работы – 1) с каждым разделяемым ресурсом связывается один семафор из набора 2) положительное – ресурс свободен, отрицательное – занят 3) перед тем как обратиться к ресурсу процесс уменьшает значение семафора, если после значение отрицательно – процесс блокируется, пока семафор не примет значение, при уменьшение которого будет положительным. 4) закончив работу, процесс увеличивает значение семафора и разблокирует один из ранее заблокированных процессов.

Двоичный семафор – мьютекс (когда семафора два – либо 0, либо 1 – доступ всегда только одному процессу одновременно)

Фьютекс – объект синхронизации в пользовательском пространстве

21. иерархия от вверха к низу по времени доступа и чатсота обращения 1) регистры процессора 2) кэш-память процессора 3) оперативная память 4) внешняя память на ссд дисках 5) внешняя память на магнитных дисках 6) медленная память для архивного хранения.

Архитектура компьютера различает физическое адресное пространство (ФАП) и логическое адресное пространство (ЛАП). Физическое адресное пространство представляет собой простой одномерный массив байтов, доступ к которому реализуется аппаратурой памяти по адресу, присутствующему на шине адреса микропроцессорной системы. Логическое адресное пространство организуется самим программистом исходя из конкретных потребностей. Трансляцию логических адресов в физические осуществляет блок управления памятью MMU.

22. С фиксированными разделами - Простейший способ управления оперативной памятью состоит в том, что память разбивается на несколько областей фиксированной величины, называемых разделами. Такое разбиение может быть выполнено вручную оператором во время старта системы или во время ее установки. После этого границы разделов не изменяются.

Очередной новый процесс, поступивший на выполнение, помещается либо в общую очередь, либо в очередь к некоторому разделу.

С разделами переменного размера - Изначально вся память, отводимая для приложений, свободна. Каждому вновь поступающему на выполнение приложению на этапе создания процесса выделяется вся необходимая ему память (если достаточный объем памяти отсутствует, то приложение не принимается на выполнение и процесс для него не создается). После завершения процесса память освобождается, и на это место может быть загружен другой процесс. Таким образом, в произвольный момент времени оперативная память представляет собой случайную последовательность занятых и свободных участков (разделов) произвольного размера.

Оврелеи – решения проблемы нехватки оперативной памяти, по средствам использования структур с перекрытием. Когда разбивают на сегменты вручную на этапе написания программы, необходимые сегменты загружаются с диска.

Свопинг – другое решение, выгрузка ожидающего процесса целиком на диск при загрузке другого процесса, получившего время процессора и готового к выполнению.

Виртуальная память – принципе обмена между оперативной памятью и внешним хранилищем, фрагментами – сегментами или страницами.

23. При реализации схем с динамическим распределением памяти возникает проблема управления блока свободной памяти. Проблема решается двумя способами – использования битовых матриц и связных списков.

Суть связных списков: В системе имеется переменная, в которой хранится адрес первого свободного участка. В начале первого свободного участка записывается его размер и адрес следующего свободного участка. В простейшем случае список свободных блоков никак не упорядочивается. Поиск выполняется перебором списка.

Суть битовых матриц: В методах битовой карты создается "карта" памяти - массив бит, в котором каждый однобитовый элемент соответствует единице доступной памяти и отражает ее состояние: 0 - свободна, 1 - занята. Если считать единицей распределения единицу адресации - байт, то сама карта памяти будет занимать 1/8 часть всей памяти, что делает ее слишком дорогостоящей. Поэтому при применении методов битовой карты обычно единицу распределения делают более крупной, например, 16 байт. Карта, таким образом, отражает состояние каждого 16-байтного кадра. Карта может рассматриваться как строка бит, тогда поиск участка памяти для выделения выполняется как поиск в этой строке подстроки нулей требуемой длины.

24.