**Цвет учебника - виртуальный**

**Лекция 1 06.09.2022 Введение в операционные системы**

**Операционная система** обеспечивает удобство по средством представления расширенной машины и рациональное управление ресурсами компьютера

Управление ресурсами включает:

* Планирование ресурса
* Отслеживание состояния

**Эволюция операционных систем**

1945-1955 Использовались реле и электронные лампы

1955-1965 Появились полупроводниковые элементы. Вместо ламп появились транзисторы. Произошло разделение персонала на программистов и операторов, эксплуатационники и разработчики вычислительных машин. Появились первые алгоритмические языки. Появились первые системы развития обработки, которые автоматизировали запуск одной программы за другой. По итогу пакетной обработки был разработан язык управления заданиями. С помощью, которой программист сообщал системе и оператору, какое задание хочет выполнить. Компьютеры использовались для научных и технических вычислений. Появились языки программирования Fortran и Assembler и первые операционные системы

1965-1980 Переход к интегральным схемам. Создание семейств программно-совместимых машин. Появление IBM 360. Программная совместимость требовала совместимость операционных систем. Важнейшим достижением являлась реализация мультипрограммирование (Одномерное выполнение нескольких программ). Спулинг-способ организации вычислительного процесса в соответствие с которым задания с перфокарт считывались в том темпе, в котором они появлялись в вычислительном центре, а затем после завершения очередного задания новое загружалось в освободившийся раздел. PDP 1. 74 бит- оперативная память этого компьютера. Появился новый тип ОС – системы с разделением времени (система Мультикс)

1980 – наст.вр. Появление больших интегральных схем. Появление персональных компьютеров. Доминирование на рынке двух операционных систем однопользовательский MS DOS и мультипрограммной мультипользовательской Unix. С развитием сетей в 80-годы бурно развивались сетевые операционные системы.

**Лекция 2 13.09.2022 Классификации операционных систем**

Отличительные черты операционных систем:

1. По размеру:
   1. ОС mainframe. В основном ориентированы на обработку множества одновременных заданий. Большинству из которых требуется операция Ввода-вывода. Предлагается три вида обслуживания:
      1. Пакетная обработка
      2. Обработка транзакций
      3. Разделение времени
   2. Серверные операционные системы – работают на серверах, рабочих станциях, и некоторых mainframe. Обслуживают несколько пользователей. И позволяют делить между собой программные ресурсы. Windows-сервер, Apache.
   3. Многопроцессорные операционные системы:
      1. Мультипроцессорные системы – в центре общая память а вокруг 12 центральных процессоров. Операционная система делятся на ассиметричные( операционная система выполняется только на одном из процессоров, распределяя прикладные задачи между остальными) симметричная( полностью децентрализована и использует весь пул процессоров распределяя их между системными и прикладными задачами). Все Windows NT(Next Texnology)
      2. Многомашинные системы и кластеры (схема у которого у каждого есть своя процессор и своя локальная память)
      3. Глобальная и распределенные системы (схема у которой у каждой есть полная система и локальная память и соединяет все это сеть интернет)
   4. Операционные системы для персональных компьютеров – представление удобного интерфейса для одного пользователя.
   5. ОС реального времени – главным параметром является время. Существует еще гибкая система реального времени, в котором допустимы редкие пропуски сроков выполнения операций. Примером гибкой системы реального времени является телевизор.
   6. Встроенные ОС. В компьютерах, в микроволновых печах.
   7. ОС для смарт-карт – сим-карта.
2. В зависимости от выбранной структуры организации различают:
   1. Монолитные. Сначала компилируются отдельные модули, затем компануется ОС. Обычно есть ядро и пространство пользователя. Ядро отвечает за всю работу. Параметры помещаются в строго определенные места регистры или стек. После чего выполняется команда прерывания. Известная как «вызов ядра», которая переключает машину из режима пользователя в режим ядра и передает управление ОС. Структура такой ОС
      1. Главная программа- которая вызывает служебную процедуру
      2. Набор служебных процедур выполняющую служебные вызовы
      3. Набор утилит обслуживающие служебные процедуры
   2. Многоуровневые системы.
   3. Виртуальные машины
   4. Экзоядро – раздрабление машина на реальные ресурсы
   5. Модель клиент-сервер микроядро

**Лекция 3 20.09.2022**

**Управление процессами**

**Процесс или задача -** это абстракция описывающая выполняющуюся программу.

Под системой управления процессами планируе управление процессами то есть распределяет процессорное время между несколькими одновременно существующими в системе процессорами, а также занимается созданием и уничтожением процессов, обеспечивает процессы необходимыми системными ресурсами и поддерживает взаимодействие между процессами. В многозадачной системе реальный процессор переключается с процесса на процесс, но рассматривается набор процессов, идущих параллельно, то есть процессор переключается от программы к программе, и это переключение называется **многозадачностью**.

Каждый процессор имеет счетчик команд. При параллельном вычислении время будет выше чем при пакетной обработке

**Состояние процессов**

В многозадачной многопроцессорной системе процесс может находиться в одном из трех основных состояний: выполнение, готовность и ожидание

**Выполнение** – активное состояние процессора во время которого он обладает всеми необходимыми ресурсами и непосредственно выполняется процессором

**Ожидание –** пассивное состояние процесса, процесс заблокирован, он не может выполняться по своим внутренним причинам, он ждет осуществления некого события.

**Готовность –** пассивное состояние процесса, но в этом случае процесс

**Алгоритмы планирования процессов**

Планирование включает:

Определение момента времени для смены выполняемого процесса

Выбор процесса для выполнения из очереди готовых процессов

Переключение контекстов старого и нового процессов(переключение регистров, страниц памяти)

Алгоритмы планирования можно разделить на 2 группы:

1. Основанные на квантовании (кванты времени на выполнение)
2. Основанные на приоритетах

При квантовании смена активного процесса происходит, если

Приоритет число характеризующее степень привилегированности процесса при использовании ресурсов вычислительной машины. Чем выше приоритет, тем выше привилегии.

Приоритет может назначаться директивно администратором системы, либо

Существуют две разновидности приоритетных алгоритмов:

1. Алгоритмы использующие относительные приоритеты
2. Алгоритмы использующие абсолютные приоритеты

В системах с отосительным приоритетом активный процесс выполняется до тех пор пока сам не покинет процессор, переходя в состояние «ожидание». В системах с абсолютным приоритетом выполнение активного процесса прерывается при еще одном условии – если в очереди готовых процессов появился процесс. В этом случае прерванный процесс переходит в состояние готовности.

**Вытесняющие и невытесняющие алгоритмы планирования**

Существует 2 типа процедур планирования – вытесняющие и невытесняющие. Невытесняющая многозадачность – способ панирования процессов, при котором активный процесс выполняется до тех пор, пока сам не отдаст управление планировщику ОС

Вытесняющая многозадачность – способ при котором решение о переключении процессора с выполнения одного процесса на другой принимается планировщиком ОС

При невытесняюшей многозадачности механизм планирования распределен между системой и прикладными программами.

Преимущество невытесняющихх систем является более высокая скорость переключения с задачи на задачу

Современные системы реального времени делятся на жесткие системы реального времени что означает жесткие сроки

И гибкие системы реального времени в которых нарушения графика нежелательно, но допустимо

**Алгоритмы планирования RMS**

Планирование с приоритетом, пропорциональным частоте. Используется при следующих условиях:

* Каждый периодический процесс должен быть завершен за время его периода
* Ни один процесс не должен зависеть от любого другого процесса
* Каждому процессу требуется одинаковое процессорное время на каждом интервале
* У непериодических процессов нет жестких сроков
* Прерывание процесса происходит мгновенно без накладных расходов

Алгоритм RMS работает, назначая каждому процессу фиксированный приоритет, равный частоте возникновения события процесса.

**Алгоритм планирования EDF**

Процесс с ближайшим сроком завершения выполняется в первую очередь. Приоритеты динамические, поэтому не прерываются процессы.

**Средства синхронизации и взаимодействие процессов**

**Критическая секция** – часть программы в которой осуществляется доступ к разделяемым данным, чтобы исключить эффект гонок по отношению к некоторому ресурсу необходимо обеспечить чтобы в каждый момент в критической секции находился максимум один процесс. Простейший способ взаимного исключения - запрещения прерывания. Другой способ – использование блокирующих переменных. Операция проверки и установки блокирующей переменной должна быть неделимо. Желательно иметь единую команду «проверка - установка» либо реализовывать программные примитивы, которые бы запрещали прерывания на протяжении всей операции проверки-установки.

Можно использовать проверки-установки.

Обещающее средство синхронизации процессов вводит два примитива обозначаемыми P и V, оперируя целыми неотрицательными значениями называемыми семафорами. В частном случае семафор может принимать 0 или 1. Он превращается в блокирующую переменную Mutex.

**Тупики**

Существует проблема синхронизаций - взаимной блокировки называемыми дедлоками, клинчами или тупиками. Проблема тупиков включает предотвращение тупиков, распознавание тупиков. Восстановление системы после тупиков. Для написания корректных программ используется высокоуровневое средство синхронизации называемое монитором. Монитор это набор процедур, переменных и структур данных. Имеет свойство, только один процесс может быть активным по отношению к монитору. Существует несколько языков программирования поддерживающих монитор.

**Потоки**

Современные операционные системы предлагают использовать новый механизм многопоточной обработки мультитрейдинг. Потоки имеют собственный программный счетчик. Стек регистры состояния, разделяют адресные пространства, глобальные переменные, открытые файлы, таймеры, семафоры, статистическую информацию, дочерний процесс. Потоки реализуются в пространстве пользователя или ядре