ТЕМА 1. АРХИТЕКТУРА И ФУНКЦИОНИРОВАНИЕ ПАРАЛЛЕЛЬНЫХ МНОГОПРОЦЕССОРНЫХ СИСТЕМ.

Лекция 3.

МНОГОПРОЦЕССОРНЫЕ ЭВМ С РАЗДЕЛЯЕМОЙ И РАСПРЕДЕЛЕННОЙ ПАМЯТЬЮ.

Первый вопрос.

Варианты организации памяти в многопроцессорных ЭВМ.

- системы **с разделяемой памятью**, у которых имеется одна большая виртуальная память и все процессоры имеют одинаковый доступ к данным и командам, хранящимся в этой памяти;

- системы **с распределенной памятью**, у которых каждый процессор имеет свою локальную оперативную память и к этой памяти у других процессоров нет доступа.

Различие этих двух типов памяти проявляется **в структуре виртуальной памяти**, то есть памяти, как она "видна" процессору.

Физически память обычно делится на части, доступ к которым может быть организован независимо.

Варианты объединения процессоров в случае общей памяти.

Простейший способ создать многопроцессорный вычислительный комплекс с разделяемой памятью - взять несколько процессоров, соединить их с общей шиной и соединить эту шину с оперативной памятью.

При этом между процессорами возникает борьба за доступ к шине и если один процессор принимает команду или передает данные, все остальные процессоры вынуждены будут перейти в режим ожидания. Это приводит к тому, что начиная с некоторого числа процессоров, быстродействие такой системы перестанет увеличиваться при добавлении нового процессора.

Эту проблему частично решает наличие локальной (принадлежащей данному процессору) кэш-памяти. При этом следующая необходимая ему команда с большой вероятностью будет находиться в кэш-памяти. В

Наличие в системе множества микросхем памяти позволяет использовать потенциальный параллелизм, заложенный в такой организации. Для этого микросхемы памяти часто объединяются в банки или модули, содержащие фиксированное число слов, причем только к одному из этих слов банка возможно обращение в каждый момент времени. Чередуемая память разделяется на банки памяти.

Принято соглашение о том, что ячейка памяти с номером *i* находится в банке памяти с номером *i mod n*, где *n* - количество банков памяти. Таким образом, если имеется 8 банков памяти, то первому банку памяти будут принадлежать ячейки памяти с номерами 0, 8, 16, :, второму - 1, 9, 17, ... и т.д.

Запросы к различным банкам памяти могут обрабатываться

"Redundant Arrays of Independent Discs" – избыточный массив независимых дисков.

В 1987 г. три американских исследователя Паттерсон, Гибсон и Катц из Калифорнийского университета Беркли написали статью "A Case for Redundant Arrays of Inexpensive Discs (RAID)".

В статье описывалось, каким образом можно несколько дешевых жестких дисков объединить в одно логическое устройство таким образом, что в результате объединения повышаются емкость и быстродействие системы, а отказ отдельных дисков не приводит к отказу всей системы.

Возможность одновременной работы с несколькими дисками можно

Для организации параллельного доступа

рабочее пространство дисков размечается на зоны определенного размера (блоки) для размещения данных и избыточной информации. Информация, подлежащая записи на диски (запрос на обслуживание), разбивается на такие же по величине блоки, и каждый блок записывается на отдельный диск. При поступлении обирается из нескольких

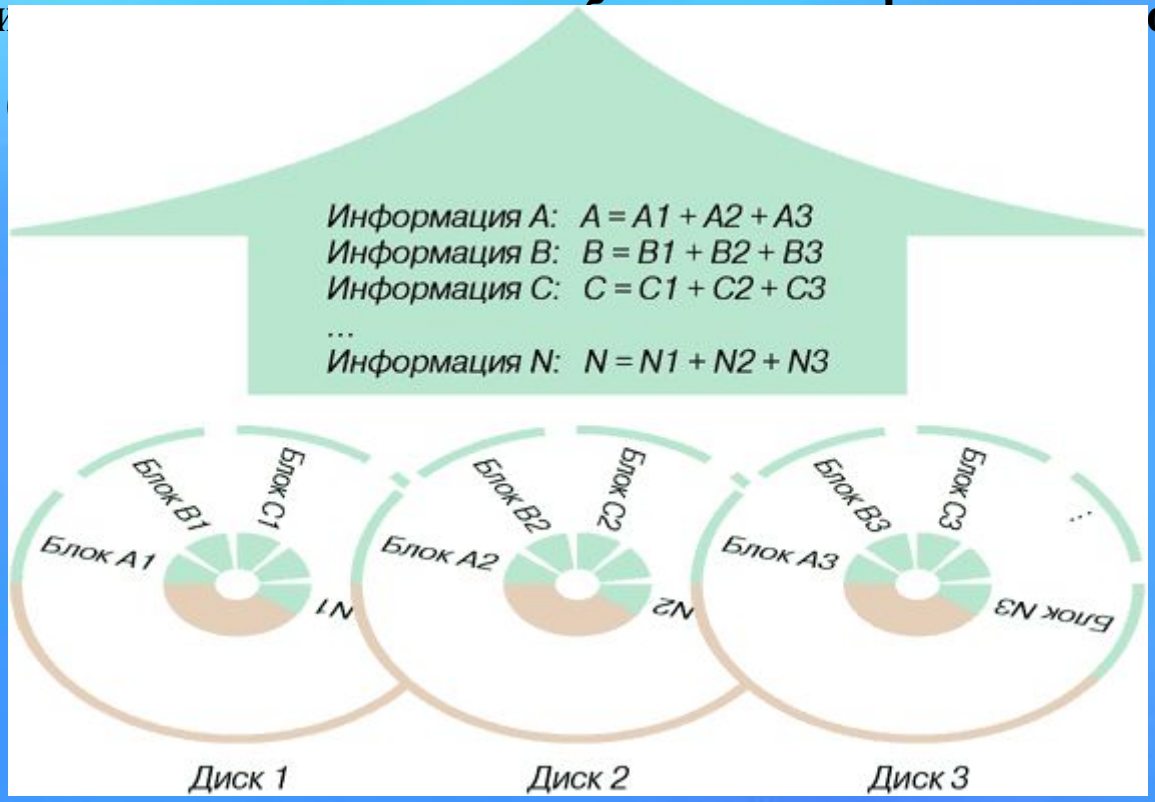

Рис. 1. Массив с параллельным доступом

Для организации независимого доступа

рабочее пространство дисков также размечается на зоны определенного размера (блоки). Однако, в отличие от предыдущего случая, каждый запрос на запись

или ч

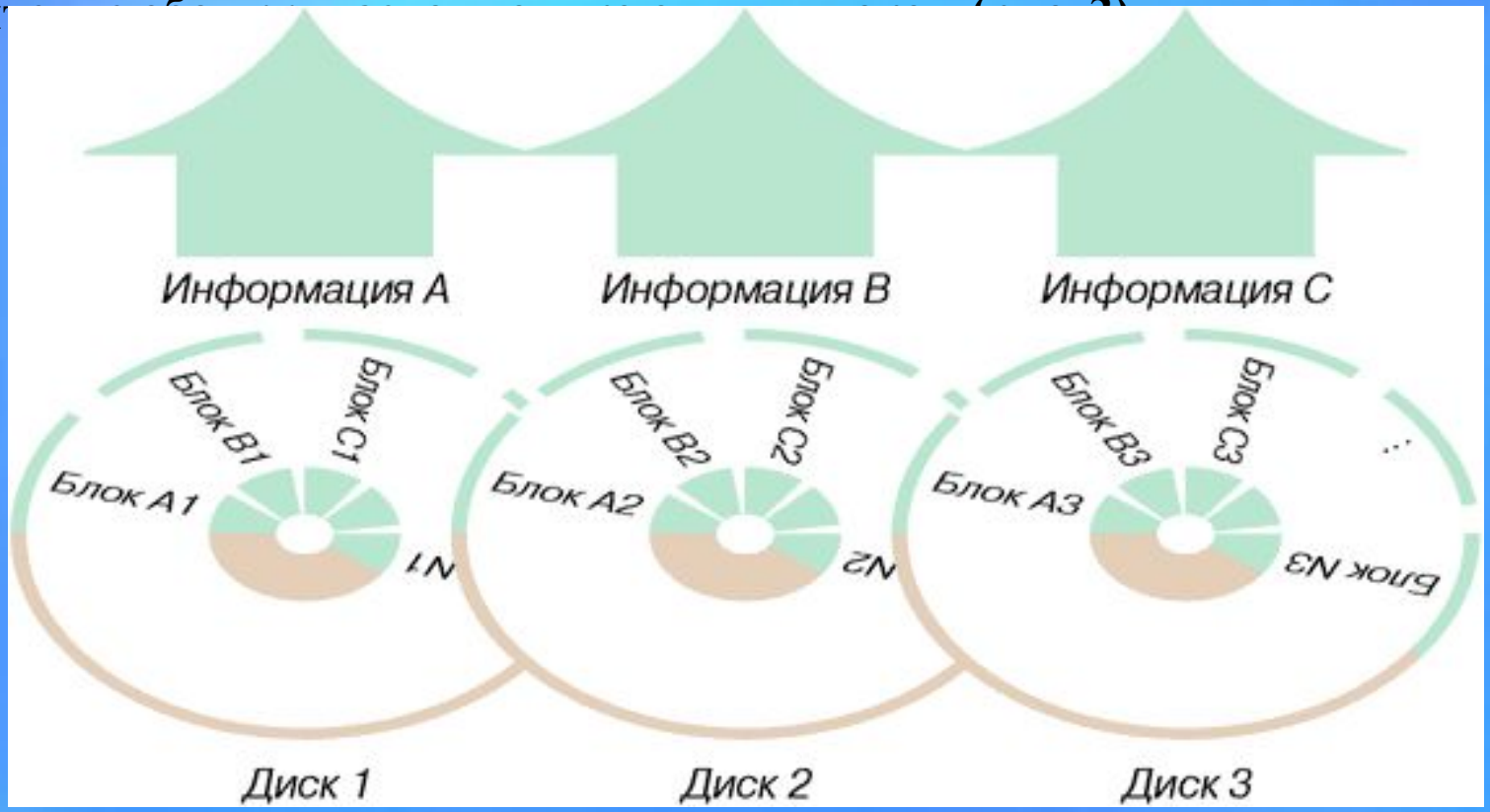

Рис. 2. Массив с независимым доступом

При сравнении RAID различного уровня в первую очередь необходимо сравнивать размер логических блоков. Точнее говоря, не собственно размер, а соотношение размера блока и величины запроса на обслуживание (объем информации, подлежащей записи или считыванию).

Другим фактором, влияющим на производительность, является способ размещения избыточной информации. Избыточная информация может храниться на специально выделенном для этого диске и может распределяться по всем дискам.

И, наконец, в RAID различного уровня применяются различные способы вычисления избыточной информации. Это также влияет на характеристики RAID (надежность, в первую очередь, производительность и стоимость).

Основные способы: полное дублирование информации, применение кодов с коррекцией ошибок (применяется код с коррекцией одиночных ошибок и

Технология I2O в RAID-контроллерах

I 2O (Intelligent Input/Output) – это спецификация, которая определяет стандартную архитектуру интеллектуального ввода-вывода и не зависит от конкретных устройств и операционной системы.

Идея технологии I 2O заключается в том, чтобы за счет применения отдельного процессора ввода-вывода разгрузить центральный процессор. Все низкоуровневые прерывания, поступающие от периферийных устройств, обрабатываются не центральным процессором, а специализированным процессором ввода-вывода (IOP). Спецификация I 2O определяет разбиение драйвера устройства на две части: ОС-зависимого модуля (OSM – Operation System Services Module) и аппаратно-зависимого модуля (HDM – Hardware Device Module). Благодаря этому в значительной мере решается задача устранения зависимости от конкретной операционной системы. Под конкретную операционную систему разрабатывается только драйвер OSM. По

Второй вопрос.
Основные классы параллельных суперкомпьютеров.

Симметричные мультипроцессорные системы (SMP)

(Symmetric Multi-Processing)

Для дальнейшей систематики мультипроцессоров учитывается способ построения общей памяти. Первый возможный вариант – использование единой (централизованной) общей памяти (shared memory) (см. рис. 1.а). Такой подход обеспечивает однородный доступ к памяти (uniform memory access или UMA) и служит основой для построения векторных параллельных процессоров (parallel vector processor или PVP) и симметричных

Рис. 1. Архитектура многопроцессорных систем с общей (разделяемой) памятью: системы с однородным (а) и неоднородным (б) доступом к памяти

Симметричный многопроцессорный (SMP) узел содержит два или более одинаковых процессора, используемых равноправно. Все процессоры имеют одинаковый доступ к вычислительным ресурсам узла. Поскольку процессоры одновременно работают с данными, хранящимися в единой памяти узла, в SMP-архитектурах обязательно должен быть механизм, поддержки когерентности данных.

Когерентность данных означает, что в любой момент времени для каждого элемента данных во всей памяти узла существует только одно его значение несмотря на то, что одновременно могут существовать несколько копий элемента данных, расположенных в разных видах памяти и обрабатываемых разными процессорами.

Механизм когерентности должен следить за тем, чтобы операции с одним и тем же элементом данных выполнялись на разных процессорах

SMP-узлы очень удобны для разработчиков приложений

операционная система почти автоматически масштабирует приложения, давая им возможность использовать наращиваемые ресурсы. Само приложение не должно меняться при добавлении процессоров и не обязано следить за тем, на каких ЦПУ оно работает.

Временная задержка доступа от любого ЦПУ до всех частей памяти и системы ввода-вывода одна и та же. Разработчик оперирует с однородным адресным пространством. Все это приводит к тому, что SMP-архитектуры разных производителей выглядят в основном одинаково: упрощается переносимость программного обеспечения между SMP-системами.

Переносимость программ - одно из основных достоинств SMP-платформ.

Типичные SMP-архитектуры в качестве аппаратной реализации механизма поддержки когерентности используют шину слежения (snoopy bus).

SMP система состоит из нескольких однородных процессоров и массива общей памяти.

Один из часто используемых в SMP архитектурах подходов для формирования масштабируемой, общедоступной системы памяти, состоит в однородной организации доступа к памяти посредством организации масштабируемого канала память-процессоры. Каждая операция доступа к памяти интерпретируется как транзакция по шине процессоры - память. Когерентность кэшей поддерживается аппаратными средствами.

Недостатком данной архитектуры является необходимость организации канала процессоры - память с очень высокой

Вся система работает под управлением единой ОС

(обычно UNIX-подобной, но для Intel-платформ поддерживается Windows NT).

ОС автоматически (в процессе работы) распределяет процессы/нити по процессорам (scheduling), но иногда возможна и явная привязка.

Модель программирования

Программирование в модели общей памяти. (POSIX threads, OpenMP). Для SMP-систем существуют сравнительно эффективные средства автоматического распараллеливания.

Массивно-параллельные системы (MPP)

(Massively Parallel Processing)

Мультикомпьютеры (многопроцессорные системы с распределенной памятью) уже не обеспечивают общего доступа ко всей имеющейся в системах памяти (no-remote memory access или NORMA) (рис. 2). При всей схожести подобной архитектуры с системами с разделяемой общей памятью с неоднородным доступом (рис. 1.б), мультикомпьютеры имеют принципиальное отличие: каждый процессор системы может использовать только свою локальную память, в то время как для доступа к данным, располагаемым на других процессорах, необходимо явно выполнить операции передачи сообщений (message passing operations). Данный подход применяется при построении двух типов многопроцессорных вычислительных систем - массивно-параллельных систем (massively parallel processor или MPP) и кластеров (clusters).

Рис. 2. Архитектура многопроцессорных систем с распределенной

Узлы в архитектуре MPP обычно состоят из одного ЦПУ, небольшой памяти и нескольких устройств ввода-вывода.

В каждом узле работает своя копия ОС, а узлы объединяются между собой специализированным соединением. Взаимосвязи между узлами (и между копиями ОС, принадлежащими каждому узлу) не требуют аппаратно поддерживаемой когерентности, так как каждый узел имеет собственную ОС и, следовательно, свое уникальное адресное пространство физической памяти.

Когерентность реализуется программными средствами, с использованием техники передачи сообщений.

Задержки, которые присущи программной поддержке когерентности на

- **производительность MPP-систем весьма чувствительна к задержкам, определяемым программной реализацией протоколов и аппаратной реализацией среды передачи сообщений (будь то коммутатор, или сеть). Вообще говоря, настройка производительности MPP-систем включает распределение данных для того, чтобы минимизировать трафик между узлами.**
- **MPP-архитектуры привлекательны в первую очередь для разработчиков аппаратных средств, так как в этом случае возникает меньше проблем и ниже стоимость аппаратуры.**
- **Такие системы обеспечивают высокий уровень производительности для приложений с большой интенсивностью вычислений, со статистически разделяемыми данными и с минимальным обменом данными между узлами.**

- **Для большинства коммерческих приложений MPP-системы подходят плохо из-за того, что структура базы данных меняется со временем и слишком велики затраты на перераспределение данных.**

Ключевым различием между одиночным SMP-узлом и MPP-системой является то, что внутри SMP-узла когерентность данных поддерживается исключительно аппаратными средствами. Это действительно быстро, но и дорого. В MPP-системе с таким же числом процессоров когерентность между узлами реализуется программными средствами. Поэтому происходит это более медленно, однако и цена значительно ниже.

MPP система состоит из нескольких однородных вычислительных узлов, включающих один или несколько процессоров, локальную для каждого узла память, коммуникационный процессор или сетевой адаптер. Узлы объединяются через высокоскоростную сеть или коммутатор.

Существуют **два основных варианта**:

- Полноценная ОС работает только на управляющей машине (front-end), на каждом узле работает сильно урезанный вариант ОС, обеспечивающие только работу расположенной в нем ветви параллельного приложения.

  Пример: Cray T3E.

На каждом узле работает полноценная UNIX-подобная ОС (вариант

Литература:

- 1. АРХИТЕКТУРА ЭВМ, УПРАВЛЯЕМОЙ ПОТОКОМ ДАННЫХ, С РАЗДЕЛЬНЫМИ ЗУ ПРОГРАММ И ПОМЕТОК. Экспресс-информация «Вычислительная техника», № 17.М.: ВИНИТИ.- 1983.- с.5 – 10.
- 2. A data flow computer architecture with program and tocen memories.Sowa M.,Murata J.“ IEEE Trans. Comput.” , 1982. 31, # 9, 820 - 824
- 3. Компьютеры на СБИС: В 2–х кн. Кн. 2: Пер. с япон./ Мотоока Т., Хорикоси Х., Сакаути М., и др. – М.: Мир, 1988.- 336 с., ил.
- 4. Т.Ц. Кенчев, К.Л. Боянов. Вычислительная структура с граф – программным управлением. Управляющие системы и машины, № 6, 1978
- 5. Михаил Кузьминский. Процессоры для высокопроизводительных вычислений // Открытые системы. – М., 2007 - Вып.9.