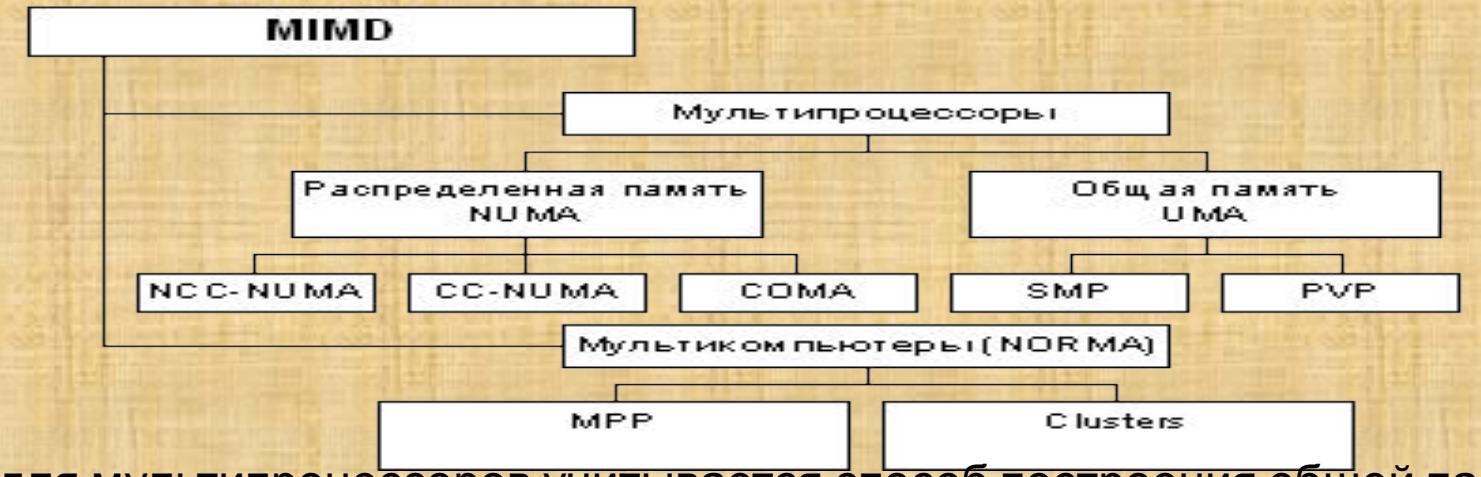
Параллельные архитектуры на основе общей разделяемой памяти: особенности организации, примеры

SMP-симметричные многопроцессорные системы

Основные классы современных параллельных компьютеров.

Различают два важных типа многопроцессорных систем – **мультипроцессоры** или системы с общей разделяемой памятью и **мультикомпьютеры** или системы с распределенной памятью.



Далее для мультипроцессоров учитывается способ построения общей памяти. Возможный подход - использование единой (централизованной) общей памяти. Такой подход обеспечивает однородный доступ к памяти *(uniform memory access or UMA)* и служит основой для построения векторных суперкомпьютеров *(parallel vector processor, PVP)* и симметричных мультипроцессоров *(symmetric multiprocessor or SMP).* Среди примеров первой группы суперкомпьютер *Cray T90,* ко второй группе относятся *IBM eServer p690, Sun Fire E15K, HP Superdome, SGI Origin 300* и др.

Описание архитектуры.

SMP архитектура *(symmetric multiprocessing)* - симметричная многопроцессорная архитектура. Главной особенностью систем с архитектурой *SMP* является наличие общей физической памяти, разделяемой всеми процессорами, причем по отношению к этой памяти все процессоры являются равнозначными и выполняют идентичные функции:

- Все процессы выполняются в одном и том же пространстве физической и виртуальной памяти.
- Любой процессор может выполнять любую нить в системе.
- Любой процессор может обрабатывать любое внешнее прерывание. Каждый процессор обрабатывает те внутренние прерывания, которые возникают в ходе выполнения текущего потока команд.
- Любой процессор может инициализировать операцию ввода-вывода.

Такая взаимозаменяемость означает, что любой процессор потенциально может выполнять любую ожидающую выполнения операцию

Память разделена на блоки, чтобы снизить количество конфликтов при одновременном обращении к памяти нескольких процессоров. Доступ к памяти идет по шине (в более дешевых 1-4-х процессорных системах) или через коммутатор.

Какой процессор играет роль загрузочного, определяется аппаратными средствами или совместно аппаратурой и BIOS. Это сделано для удобства и имеет значение только во время инициализации и выключения. BSP-процессор отвечает за инициализацию системы и за загрузку ОС. AP-процессор активизируется после загрузки ОС.

Симметричность

Симметричность имеет два важных аспекта: симметричность памяти и ввода-вывода. Память симметрична, если все процессоры совместно используют общее пространство памяти и имеют в этом пространстве доступ с одними и теми же адресами. Симметричность памяти предполагает, что все процессоры могут исполнять единственную копию ОС. В таком случае любые существующие системы и прикладные программы будут работать одинаково, независимо от числа установленных в системе процессоров. Требование симметричности ввода-вывода выполняется, если все процессоры имеют возможность доступа к одним и тем же подсистемам ввода-вывода (включая порты и контроллеры прерывания), причем любой процессор может получить прерывание от любого источника. Некоторые многопроцессорные системы, имеющие симметричный доступ к памяти, в то же время являются асимметричными по отношению к прерываниям устройств ввода-вывода, поскольку выделяют один процессор для обработки прерываний. Симметричность ввода-

Сильносвязанная реализация базируется на схеме, согласно которой процессоры совместно используют данные из пула общих ресурсов, прежде всего, из общей памяти. **Слабосвязанные** системы используют механизм обмена сообщениями между процессами для совместного использования ресурсов, когда это необходимо. В некоторых слабосвязанных системах каждый процессор может даже иметь свой собственный контроллер диска и другие подсистемы.

Наиболее целесообразно использовать симметричную сильносвязанную модель. Тогда ОС обеспечивает мощную поддержку симметричной многопроцессорной обработки, так как планировщик в ядре ОС функционирует на уровне нити, поэтому сервер может назначить две нити одного процесса различным процессорам. Это особенно полезно для прикладных программ баз данных, где запросы могут быть расщеплены на нити и распределены между процессорами, что ведет к значительному увеличению производительности.

Приоритетные прерывания

Чтобы полнее воспользоваться преимуществами *SMP* при организации многозадачности, выполнение нитей процесса контролируется с помощью приоритетных прерываний. Приоритетное прерывание позволяет операционной системе поддерживать контроль над программами, какую программу и когда запускать, так что сбившиеся программы не могут поработить систему и вызвать проблемы. При приоритетных прерываниях - постоянный, занимающие микросекунды запуск и остановка нескольких программ - как только возобновляется выполнение нити, ОС может назначить ее другому процессору.

Масштабируемость.

Масштабируемость очень низкая.

Для достижения масштабируемости важно использовать асинхронные операции. При асинхронном вводе/выводе процессу не надо ожидать завершения чтения или записи, прежде чем он приступит к выполнению другой задачи. Каждый процесс создается с использованием единственной нити, которая выступает отдельным блоком при выполнении процессором команд программы. Программы могут запускать новые нити по мере потребности, и ОС назначает и контролирует их без участия высокоуровневой прикладной системы.

SMP-приложения

На сетевых *SMP*-серверах, с которыми одновременно работают множество пользователей, можно и нужно реализовать параллельное выполнение задач. В среде операционных систем со средствами *SMP* все стандартные приложения обычно выполняются без проблем, однако, чтобы реально использовать преимущества многопроцессорной обработки и масштабируемости, сами приложения также должны поддерживать *SMP* и учитывать архитектуру базовой ОС. Пока таких приложений немного, но число их быстро растет. Считается, что *SMP*-системы оптимально подходят для задач с интенсивным использованием процессора, где важное значение имеет малое время отклика для пользователя. К подобным задачам можно отнести базы данных и ПО коллективного пользования.

Достоинства:

- Простота и универсальность для программирования. Архитектура *SMP* не накладывает ограничений на модель программирования, используемую при создании приложения: обычно используется модель параллельных ветвей, когда все процессоры работают абсолютно независимо друг от друга - однако, можно реализовать и модели, использующие межпроцессорный обмен. Использование общей памяти увеличивает скорость такого обмена, пользователь также имеет доступ сразу ко всему объему памяти. Для *SMP*-систем существуют сравнительно эффективные средства автоматического распараллеливания.
- Легкость в эксплуатации. Такие системы мало отличаются от однопроцессорных. Обычно, нет никакой надобности в специальных навыках обслуживания.
- Относительно невысокая цена. Особенно при организации доступа к памяти через общую шину. В этом случае разница в стоимости по сравнению с однопроцессорной машиной будет лишь за счет цены дополнительных процессоров (обычно серийных), а также за счет схемы поддержки когерентности КЭШей (которую можно сделать очень простой).

Недостатки:

- Плохая масштабируемость. Этот важный недостаток *SMP*-системы не позволяет считать их по-настоящему перспективными. Причины плохой масштабируемости состоят в том, что в данный момент шина способна обрабатывать только одну транзакцию, вследствие чего возникают проблемы разрешения конфликтов при одновременном обращении нескольких процессоров к одним и тем же областям общей физической памяти. Вычислительные элементы начинают друг другу мешать. Когда произойдет такой конфликт, зависит от скорости связи и от количества вычислительных элементов. В настоящее время конфликты могут происходить при наличии *8-24*-х процессоров. Кроме того, системная шина имеет ограниченную (хоть и высокую) пропускную способность (ПС) и ограниченное число слотов. Все это с очевидностью препятствует увеличению производительности при увеличении числа процессоров и числа подключаемых пользователей. В реальных системах можно использовать не более *32* процессоров. Для построения масштабируемых систем на базе *SMP* используются кластерные или *NUMA*-архитектуры.

Итоги:

Архитектура	Система состоит из нескольких однородных процессоров и массива общей памяти (обычно из нескольких независимых блоков). Все процессоры имеют доступ к любой точке памяти с одинаковой скоростью. Процессоры подключены к памяти либо с помощью общей шины (базовые 2-4 процессорные SMP-сервера), либо с помощью crossbar-коммутатора (HP 9000).
Примеры	HP 9000 V-class, N-class; SMP-серверa и рабочие станции на базе процессоров Intel (IBM, HP, Compaq, Dell, ALR, Unisys, DG, Fujitsu и др.).
Масштабируемость	Наличие общей памяти сильно упрощает взаимодействие процессоров между собой, однако накладывает сильные ограничения на их число - не более 32 в реальных системах. Для построения масштабируемых систем на базе SMP используются кластерные или NUMA-архитектуры.
Операционная система	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, OpenM... SMP систем существуют сравнительно эффективные