ПОДХОД К РАСШИРЕНИЮ ФУНКЦИОНАЛЬНОСТИ АДАПТЕРА КОММУНИКАЦИОННОЙ СЕТИ EC8430 ДЛЯ ПОДДЕРЖКИ ГЛОБАЛЬНО АДРЕСУЕМОЙ ПАМЯТИ

М.А. Гильмендинов, А.С. Фролов

Ключевые слова: подсистема памяти, коммуникационная сеть, параллельные вычисления

Объемы данных, с которыми оперируют прикладные задачи, постоянно увеличиваются и ограничиваются только физическими ресурсами вычислительной системы. В большинстве случаев использование медленной дисковой памяти крайне не эффективно. В области бизнес-аналитических систем и баз данных активно развиваются технологии под общим названием in-memory computing.

Для эффективного решения таких задач используются либо специализированные компьютеры с аппаратной поддержкой общей памяти либо программные реализации протоколов для кластерных систем с распределенной памятью.

Существуют различные подходы к реализации общей памяти в аппаратных вычислительных системах. Примером систем с аппаратной поддержкой общей памяти являются SGI Altix UV[1] и NumaScale[2]. Обе системы реализуют кэш-когерентную общую память(ccNUMA), но используют различные интерфейсы подключения к коммуникационной сети: SGI Altix UV — Intel QPI и NumaScale — HT. Программные реализации(vSMP[3], Intel Clustered OpenMP) значительно медленнее аппаратных. Основным недостатком таких систем для поддержания общей памяти является ограниченность масштабируемости, связанная с накладными расходами, вызванными обеспечением когерентности памяти вычислительных узлов.

Университетами Валенсии и Гёйдельберга разрабатывается система MEMSCALE[4], которая имеет два принципиальных отличия от систем с общей памятью: во-первых, не поддерживается когерентность общей памяти, во-вторых, задачи в MEMSCALE могут использовать ядра только в пределах вычислительного узла, в то время как память может выделяться на любых вычислительных узлах.

В данной работе рассматривается возможность адаптации и расширения идей MEMSCALE как один из подходов к реализации программно-аппаратного механизма, обеспечивающего динамическое выделение оперативной памяти на удаленных узлах вычислительного кластера, на базе разрабатываемой отечественной коммуникационной сети EC8430[5]. Использование памяти других узлов должно быть прозрачно для приложений - не должна требоваться перекомпиляция, запуск с особыми параметрами, линковка специальных библиотек и т.п.

Реализация предлагаемого подхода состоит в решении следующих задач:

1. Реализация поддержки обработки удаленных команд обращений к памяти в адаптере коммуникационной сети.
2. Модификация менеджера памяти в ядре ОС для отображения виртуальных страниц на область ввода-вывода для перенаправления команд обращений к памяти в коммуникационную сеть.
3. Разработка библиотеки для выделения памяти с возможностью управления распределением памяти на уровне пользователя.

Для реализации аппаратной поддержки в маршрутизатор будет добавлен дополнительный конвейер для обработки запросов на чтение/запись удаленной памяти, поступающих от локального узла через интерфейс PCIe. Также необходимо назначение регистров базового адреса (BAR) PCIe на уровне BIOS для перенаправления на них запросов на получение физических страниц. Адресный флит будет формироваться аппаратно на основе данных о номере узла, физическом адресе и типе операции из PCIe пакета. Для приема и посылки страниц данных будут использоваться команды PCIe чтения и записи, в дальнейшем возможно расширение функциональности: атомарные операции(CAS, FADD), а также входящие в спецификацию PCIe 3.0. Поскольку все обращения в память будут производиться из одного узла, то поддержание когерентности между узлами не требуется.

На уровне ОС требуется обеспечить поддержку выделения памяти на удаленном узле. Для этого будет разработан модуль ядра, который будет перенаправлять запросы на выделение памяти удаленной памяти на PCIe устройство и ставить в соответствие виртуальному диапазону адресов непрерывный диапазон физических адресов соответствующих устройству PCIe для узла-потребителя. На узле-доноре модуль ядра будет отвечать за резервирование непрерывного диапазона физических адресов.

Рассмотрим механизм выделения удаленной памяти: запрос направляется узлу-донору, который выделяет память, а полученный физический адрес, указывающий на начало выделенной физической страницы, дополняет своим логическим номером и возвращает узлу запросившему память. Полученный адрес узел-потребитель использует как адрес начала физической страницы, на которую отображается виртуальная страница. При выдаче команд load или store в эту страницу запрос автоматически перенаправляется в сетевой адаптер, где формируется пакет, который передается соответствующему узлу.

Для взаимодействия с системой динамического расширения оперативной памяти предполагается подключение библиотеки с функциями управления памятью (malloc, free и т.п.), с помощью которой будет выделяться память в глобальном пуле, распределенному по множеству узлов. Библиотеку можно будет подключать динамически, например, через переменные окружения.

С помощью библиотеки, позволяющей вручную указывать программе особенности размещения данных в памяти, также можно будет давать системе рекомендации по миграции памяти.

При выделении памяти будет учитываться топология сети, что позволит снизить нагрузку на сеть, а также минимизировать задержку.

Помимо выделения памяти с учетом топологии будет исследовано изменение скорости работы приложений с автоматической миграцией выделенной памяти между узлами с последующей реализацией в случае ускорения работы. Рассматриваются несколько критериев влияющих на миграцию:

- Топология (незанятый узел, который «ближе» с точки зрения топологии, включая сам узел-потребитель).
- Статистика обращения к памяти.

Также исследуются возможность и необходимость изменения поведения системного менеджера памяти для повышения эффективности выделения памяти и оптимизации сетевого протокола для передачи страниц.

Разработку и апробацию описанного механизма планируется осуществить на 9-узловом макете сети ЕС8430 на базе ПЛИС. Тестирование производительности будет проведено на приложениях интенсивно работающих с памятью большого объема, в том числе базах данных в памяти и графовых задачах.

ЛИТЕРАТУРА:

1. SGI Altix UV: http://www.sgi.com/products/servers/uv/specs.html.
2. Numascale: http://www.numascale.com/numa_technology.html
3. vSMP: http://www.scalemp.com/architecture
4. H. Montaner, F. Silla, H. Fröning, J. Duato: A new degree of freedom for memory allocation in clusters, 2011, http://www.ziti.uni-heidelberg.de/ziti/uploads/ce_group/2011cluster-freedom.pdf
5. А.И. Слуцкин, А.С. Симонов, Д.В. Макагон, Е.Л. Сыромятников Разработка межузловой коммуникационной сети ЕС8430 «ангара» для перспективных российских суперкомпьютеров//Успехи современной радиоэлектроники, 2012, № 1
6. A. Szalay Extreme Data-Intensive Scientific Computing //Computing in Science & Engineering , vol.13, no.6, pp.34-41, Nov.-Dec. 2011
7. In-Memory Database Systems (IMDSs) Beyond the Terabyte Size Boundary: http://www.mcobject.com/130/EmbeddedDatabaseWhitePapers.htm