Коммуникационная сеть МВС-Экспресс

В.К. Левин, Б.Н. Четверушкин, Г.С. Елизаров, В.С. Горбунов,
А.О. Лацис, В.В Корнеев, А.А. Соколов, Д.В. Андрюшин, Ю.А. Климов

Аннотация. Рассматривается подход к построению коммуникационной сети МВС-Экспресс с прямой коммутацией пакетов PCI Express. Предложены средства программирования на базе модели с распределенной общей памятью ВС, создаваемых с использованием этой сети. Приведены оценки эффективности предлагаемых программно-аппаратных решений.

Ключевые слова: модели и средства программирования с распределенной общей памятью, коммуникационная сеть с малой задержкой и высоким темпом выдачи коротких сообщений.

Введение

Среди актуальных направлений использования высокопроизводительных вычислительных систем (ВС) – суперкомпьютеров - выделяется решение задач с малыми пространственной и временной локализациями обращений к памяти. Примерами задач такого рода служат решение уравнений математической физики на адаптивных нерегулярных сетках и обработка больших объемов плохо структурированных данных. Основная проблема, возникающая при решении подобных задач, состоит в крайне высокой трудоемкости программирования, а иногда и нереализуемости по практическим причинам алгоритмов при традиционном способе параллельного программирования на базе модели MPI (модели программирования с передачей сообщений).

Для эффективного решения такого рода задач строят специальные заказные суперкомпьютеры, например, Cray XMT, ключевыми особенностями которых являются: аппаратная поддержка глобально адресуемой разделяемой (общей) памяти, высокая пропускная способность коммуникационной сети при передаче большого количества коротких сообщений, использование специальных массово-мультитредовых процессоров и другие решения. Для повышения продуктивности программирования и эффективности исполнения алгоритмов с интенсивными непрогнозируемыми нерегулярными обращениями к памяти разрабатываются и используются языки класса PGAS (Partitioned Global Address Space) – распределённого разделяемого адресного пространства. Модель программирования PGAS характеризуется тем, что работа осуществляется в едином адресном пространстве, но данные на программном уровне делятся на локальные и глобальные, что позволяет программисту управлять локализацией данных и вычислений. Модель PGAS реализуется как на уровне коммуникационных библиотек типа shmem, gasnet, armci, так и в виде высокоуровневых языков параллельного программирования типа UPC, CAF, а также перспективных Chapel, X10, Fortress.

Коммерчески доступные коммуникационные сети (InfiniBand, GigabitEthernet и другие), применяемые при построении суперкомпьютеров из коммерчески доступных микропроцессорных кристаллов, на сегодняшний день не обладают аппаратной поддержкой общей памяти и лишь частично соответствуют требованиям, предъявляемым к оборудованию, необходимому для эффективного решения рассматриваемых задач. Для применения модели PGAS в таких суперкомпьютерах реализованы и раз-

виваются языки класса PGAS, такие как UPC, CAF, позволяющие работать в общем адресном пространстве на сетях без аппаратной поддержки общей памяти. При этом для выше упомянутых коммерчески доступных коммуникационных сетей ключевой для рассматриваемых алгоритмов режим работы коммуникационной сети – передача большого количества коротких сообщений – характеризуется низкими показателями достигаемой пропускной способности. Причина этого, в том числе, в том, что существующие на сегодняшний день коммерческие сети, используя PCI-Express для подключения к коммерчески доступным микропроцессорным кристаллам, задействуют дополнительный аппаратно-программный уровень – адаптеры, драйверы которых выполняют преобразование пакетных данных, за счет чего существенно вырастает латентность передачи сообщений. Как следствие, выход на близкую к пиковой пропускную способность происходит лишь на сообщениях большого размера.

ФГУП «НИИ «Квант» совместно с ИПМ им. М.В. Келдыша РАН разработал низколатентную коммуникационную сеть «МВС-Экспресс», обладающую высоким темпом выдачи коротких сообщений и применяемую при построении из коммерчески доступных микропроцессорных кристаллов ВС с общим полем памяти [1, 2].

В отличие от большинства коммерчески доступных сетей, сеть МВС-Экспресс построена на технологии прямой коммутации пакетов PCI-Express. Благодаря этому отсутствуют задержки, связанные с преобразованием сетевых пакетов в разных аппаратных средах. Сеть реализует на аппаратном уровне общее поле памяти большого объёма, которое складывается из открытых для общего доступа областей системной памяти каждого из входящих в кластер узлов и целиком доступно каждому из узлов [1-3]. Благодаря этому достигается высокий темп передачи коротких сообщений: для передачи машинного слова из одного узла кластера в другой достаточно выполнить машинную команду записи слова по соответствующему адресу. Уровни скоростей и задержек при передаче данных в такой сети сравнимы с аналогичными показателями для передачи данных внутри вычислительного узла. Темп записи отдельных слов в чужую память заметно выше 10 млн. записей в секунду (70-80 нс на одну запись), латентность передачи коротких сообщений немного превышает 1 микросекунду. При этом скорость передачи данных между узлами лежит в диапазоне от 600 Мбайт до 1.5 Гбайт в секунду, в зависимости от выбранного варианта (gen1 или gen2) PCI Express (при ширине кабеля x4). Длина пакета данных, на которой достигается пиковая пропускная способность, в сравнении с Infiniband сокращается с нескольких килобайт до десятков байт. Таким образом, возникли предпосылки построения суперкомпьютера с архитектурой, близкой к архитектуре ведущих производителей оригинальных суперкомпьютеров с общим полем памяти, но с показателями доступности и низкой удельной стоимости, такими же, как у суперкомпьютеров, построенных на основе кластерных технологий из коммерчески доступных комплектующих.

Разработка сети прямой коммутации пакетов PCI Express, близкая к предлагаемой, одновременно с нами также была выполнена американской фирмой OneStopSystems[1], причем использовались именно те микросхемы и кабели, что и в сети МВС-Экспресс.

Статья имеет следующую структуру. Во втором разделе представлены основы структуры и архитектуры сети МВС-Экспресс. Третий раздел представляет инструментальные программные средства, используемые в ВС, построенных на базе сети МВС-Экспресс. В четвёртом разделе приводятся результаты экспериментов по оценке предложенных аппаратно-программных решений. В заключении отмечены достигнутые результаты и намечены перспективы развития предложенных решений.

1. Основы построения сети МВС-Экспресс

1.1. Структура вычислительных систем на базе МВС-Экспресс

Структура вычислительной системы, создаваемой на базе коммуникационной сети (КС) с прямой коммутацией пакетов PCI Express приведена на Рис.1.

[1] http://www.onestopsystems.com

Коммуникационная сеть состоит из:

- коммутатора;
- интерфейсных плат (ИП), которые подключаются к магистралям PCI Express соответствующих вычислительных модулей (ВМ);
- сетевых кабелей (СП), с помощью которых ИП подключаются к коммутатору.

В существующем на сегодня варианте сети МВС-Экспресс используются сетевые кабели шириной x4, состоящие из 4 линий (lane), каждая из которых, в свою очередь, состоит из двух проводников для передачи в одном направлении и двух проводников для передачи в противоположном.

На Рис. 2 показаны существующий вариант устройств сети МВС-Экспресс на базе PCI Express gen2.

Интерфейсная плата представляет собой карту половинного размера со стандартным ножевым разъёмом PCI Express для присоединения к ВМ и кабельным разъемом для присоединения к коммутатору. В качестве ВМ используется коммерчески доступная многоядерная серверная плата.

Коммутатор осуществляет функции пересылки пакетов PCI Express между ВМ. С логической точки зрения все коммуникации в системе относятся к категории «точка-точка», то есть каждый ВМ соединен с каждым. Коммутатор реализован как отдельная плата, на которой размещена коммерчески доступная микросхема коммутатора пакетов PCI Express на 24 порта x4 и разъемы для подключения 24 кабелей x4.

В настоящее время для изготовления интерфейсных плат и коммутаторов коммерчески доступны микросхемы нескольких производителей, включая PLX Technology. Причём коммутаторы могут быть созданы как с использованием одной микросхемы, так и построены как составные из нескольких микросхем для увеличения количества портов.

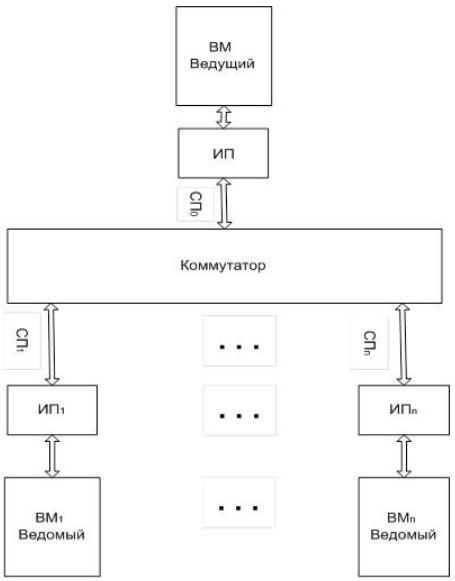

Рис.1. Структура коммуникационной среды МВС-Экспресс

Собственно, приведенная структура характерна для большинства современных коммуникационных сетей. Особенности КС МВС-Экспресс обуславливаются использованием протокола PCI Express для передачи данных не только внутри узла, но и между узлами, что приводит к следующим специфичным свойствам:

- должен быть выделен один ВМ, называемый «ведущим» (в отличие от остальных - «ведомых»), для размещения в его адресном пространстве окон доступа ко всем ведомым узлам;
- в каждом ВМ, как правило, должен быть выделен тред или ядро (в многоядерных ВМ), предназначенные для работы с интерфейсной платой, для поддержки синхронизации и когерентного состояния памяти ВМ.

Рис.2 Интерфейсная плата и плата коммутатора сети МВС-Экспресс

1.2. Топология и типы устройств магистрали PCI Express

Стандарт PCI Express выделяет (Рис.3) три типа устройств: Root Complex, PCI Switch (PCI коммутатор) и Endpoint (Конечная точка, или Устройство).

Root Complex включает выделенный порт или несколько портов PCI Express, один CPU (Центральный процессор), объединённый с RAM (Оперативной памятью) контроллером памяти. Root Complex – это устройство, которое является точкой соприкосновения трех интерфейсов: шины памяти, PCI Express и процессорной шины. Логически Root Complex выполняет две важные функции. Во-первых, именно он распределяет адреса между устройствами при включении компьютера – конфигурирует магистраль. Во-вторых, именно он является «полномочным представителем» процессора и системной памяти в коммуникационной системе PCI Express. Когда процессор выполняет команду записи в некоторое устройство, именно Root Complex является отправителем соответствующего пакета внутри коммуникационной системы. Когда устройство пишет в системную память в режиме DMA, именно Root Complex является адресатом пакетов, отправляемых устройством.

Магистраль PCI Express имеет древовидное строение. Root Complex является корнем дерева, Конечные точки (Устройства) – его листьями. Все остальные, не корневые и не листовые, вершины дерева – это PCI коммутаторы. Каждая листовая вершина дерева имеет небольшое число (не более 6) сплошных адресных областей в адресном пространстве памяти. Число и размеры этих областей определяются свойствами устройства (например, конкретной адаптерной платы формата PCI Express). Конкретное размещение этих областей в адресном пространстве памяти выполняет Root Complex при конфигурировании магистрали (например, при включении компьютера). Одновременно настраиваются коммутаторы. Если конфигурирование прошло успешно, то каждая из запрошенных тем или иным устройством адресных областей приобретает конкретное, уникальное и неизменное, размещение в адресном пространстве памяти, в адресах, не занятых системной памятью или другими устройствами. Коммутаторы, в свою очередь, теперь «знают», какие диапазоны адресов находятся за каждой из нисходящих ветвей, что позволяет им работать прозрачно для отправителя пакетов. Любой отправитель пакета в правильно сконфигурированной магистрали должен лишь снабдить

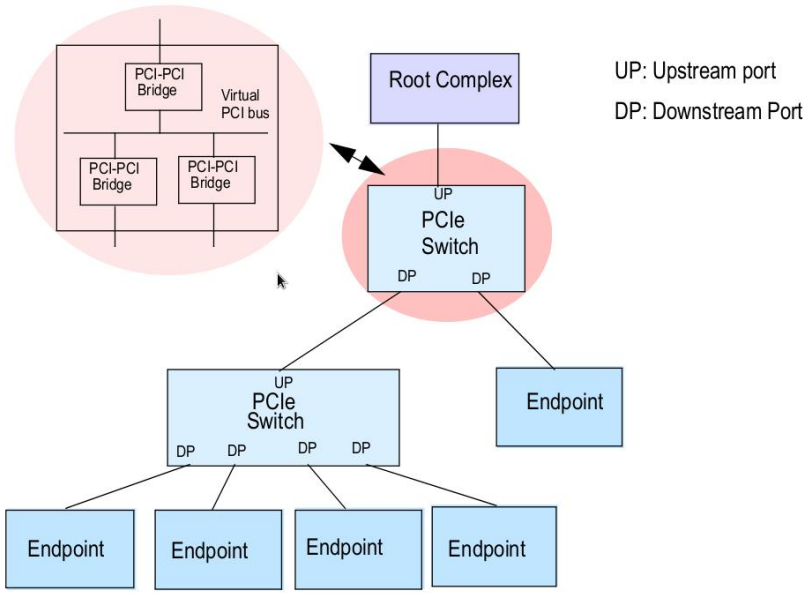

Рис. 3. Топология магистрали PCI Express

пакет адресом получателя и «вбросить» его в сеть, в результате чего пакет будет доставлен. Не только получателем, но и отправителем пакетов может быть любая оконечная точка (устройство), а не только Root Complex. Или, иными словами, любое устройство PCI Express может инициировать обмен DMA как с системной памятью, так и с другим устройством.

Такая организация коммуникационной магистрали логически не отличается от организации традиционной магистрали PCI (не Express). В традиционном исполнении PCI вместо коммутаторов использовались общие шины, а вместо передачи пакетов – адресные шинные транзакции чтения и записи. Для сохранения преемственности терминологии в системе нумерации устройств PCI Express применяется понятие «номер шины», а коммутаторы иногда называют «прозрачными шинными мостами», хотя на самом деле в PCI Express никаких шин и шинных мостов (в схемотехническом смысле этого слова) нет.

Таким образом, PCI Express, как и традиционные, более ранние варианты исполнения магистрали PCI, используют «плоское» (линейное) адресное пространство памяти, в котором размещаются (в виде отдельных адресных областей) как системная память, так и программно доступные ресурсы всех устройств. Любое устройство может взаимодействовать с системной памятью и с любым другим устройством транзакциями чтения и записи, указывая адрес назначения транзакции в едином адресном пространстве памяти. Внутри сети транзакции превращаются в сетевые пакеты, автоматически доставляемые адресату и интерпретируемые им. Древовидная структура магистрали при этом прозрачна не только для программы, инициирующей транзакции в устройствах, но и для каждого из устройств на аппаратном уровне.

Представленная здесь логическая модель магистрали PCI Express, безусловно, не полна и в чем-то не точна, но достаточна для целей дальнейшего изложения.

Важным ограничением приведенной выше модели является наличие единого адресного пространства, в котором единственный на магистрали Root Complex назначает адреса устройствам. Если нам требуется объединить несколько магистралей PCI Express, каждая из которых управляется своим Root Complex-ом, то есть несколько компьютеров, в единую магистраль передачи данных, то мы этого, на первый взгляд, сделать не можем. Проблема эта, однако, имеет простое решение, если в нашем распоряжении имеется конечная точка (устройство) специального вида – непрозрачный мост (Non-Transparent Bridge или NTB).

Это устройство специально предназначено для подключения двумя своими сторонами к разным магистралям PCI Express, например, к разным компьютерам. Физически оно обычно представляет собой две платы, связанные кабелем, но возможен и вариант с одной платой и кабелем для подключения непосредственно к разъему PCI Express. Если непрозрачный мост интегрирован в материнскую плату, достаточно просто кабеля.

В терминах древовидной топологии PCI Express непрозрачный мост – это два листа от разных деревьев, склеенные между собой. С точки зрения каждого из Root Complex-ов непрозрачный мост – это конечная точка («приклеенный» к данному листу лист от другого дерева Root Comlex-у не виден, с его точки зрения магистраль в этой точке заканчивается). Поэтому устройство называется «непрозрачным». Работа такого устройства заключается в том, чтобы просто пропускать через «склейку» в соседнее дерево все адресованные ему пакеты (то есть обращенные к нему транзакции). Поэтому устройство называется «мостом».

Адрес, указанный в транзакции, которая обращена к непрозрачному мосту, вряд ли имеет смысл в адресном пространстве по другую сторону моста – ведь там над распределением адресов независимо поработал другой Root Complex, распоряжающийся другим адресным пространством. Чтобы пропускание транзакций через непрозрачный мост имело смысл, необходимо, чтобы адреса транзакций (значения поля «адрес» в кодирующих транзакции пакетах) при проходе через мост систематически модифицировались, например, заменой старших разрядов адреса. Конкретный характер модификации (что именно на что именно заменять) может быть задан только явно, путем программной настройки соответствующих ре-

гистров моста. Необходимость такой явной настройки перед использованием – еще одно основание называть мост «непрозрачным».

Таким образом, использование NTB позволяет решить проблему объединения нескольких вычислительных модулей, каждый из которых снабжен независимой магистралью PCI Express (режим Мульти-Root).

На Рис. 4 изображены Root Complex 1 и 2, соединённые между собой через NTB. Задача NTB в данном случае заключается в том, чтобы разграничить адресные пространства Address Domain 1 и Address Domain 2. Таким образом, при автоконфигурации своей магистрали Root Complex 1 производит инициализацию локальных PCI устройств до NTB, т.е. он видит NTB как конечное устройство (Endpoint или Internal Endpoint). Root Complex 2, со своей стороны, также инициализирует свои локальные PCI устройства, также до NTB и, соответственно, также видит NTB как конечное устройство (Endpoint или External Endpoint).

Сама по себе возможность соединения двух магистралей PCI Express еще ничего не говорит о том, как такой объединенной магистралью пользоваться – все зависит от того, какие области адресов реализованы в NTB, и от того, как именно преобразуются адреса транзакций, обращенных к этим областям при проходе через NTB. На основании Рис. 4 рассмотрим вариант использования такого соединения для реализации области оперативной памяти, доступной обоим объединяемым компьютерам. Пусть на стороне, обращенной к Root Complex 1, NTB реализует область адресов размером в 1МБ. С точки зрения программы, выполняющейся на компьютере 1 (Рис. 4 слева), эта область логически не отличается от дополнительного блока оперативной памяти размером 1МБ, в ячейки которой можно писать данные обычными командами обращения к оперативной памяти. Если теперь выделить в системной памяти компьютера 2 (Рис. 4 справа) некоторую область системной памяти, размером в 1МБ, и настроить преобразователь адресов в NTB со стороны компьютера 1 именно на этот мегабайт в адресном пространстве компьютера 2, то задача решена. Всякая запись, выполненная в компьютере 1 в адресную область его стороны

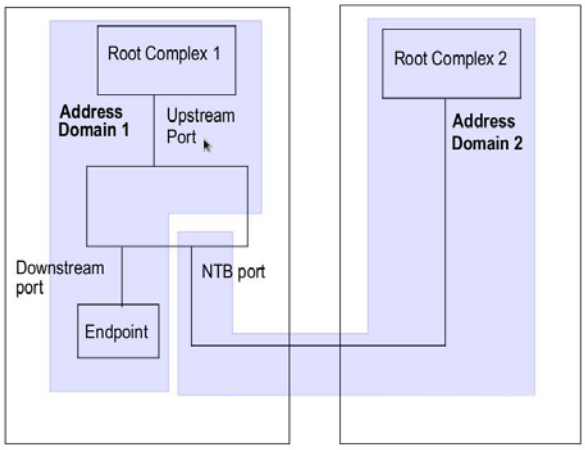

Рис. 4. Мульти-Root с применением NT Bridge

NTB, будет автоматически превращаться в запись по адресу системной памяти в компьютере 2. Запись будет происходить с тем же смещением относительно начала отображаемой области (в ту же самую ячейку в пределах выделенного мегабайта). Магистраль PCI Express компьютера 2 будет при этом видеть, что ее сторона NTB пишет в системную память компьютера 2 в режиме DMA по соответствующим адресам.

Поскольку у NTB две стороны, на каждой из которых имеются свои адресные области, а у каждой из них – свой, независимо настраиваемый программным образом преобразователь адресов, ничто не мешает нам поступить симметрично, чтобы компьютер 2 также «видел», через адресную область своей стороны NTB, некоторый выделенный мегабайт системной памяти компьютера 1. Мы реализовали упомянутый выше принцип построения сети МВС-Экспресс (правда, пока всего для двух компьютеров): каждый узел сети предоставляет для стороннего доступа некоторый кусок своей системной памяти, имея при этом доступ ко всем таким кускам, предоставленным другими компьютерами сети, по некоторым адресам в своем адресном пространстве памяти.

В настоящее время на рынке присутствуют микросхемы массового выпуска, специально предназначенные, в том числе, для использования в качестве PCI Express NTB. Обычно их используют совместно с конфигурационной flash-памятью, содержимое которой указывает, какие именно адресные области реализует NTB с каждой из сторон. Например, можно создать такую

конфигурационную «прошивку» flash-памяти, чтобы со стороны компьютера 1 (Рис. 4) NTB имел 2 адресных области, размерами 16 и 1024 мегабайт, соответственно, а со стороны компьютера 2 – одну, размером 16МБ.

Каждая сконфигурированная таким образом адресная область на каждой стороне NTB автоматически будет снабжена собственным независимым преобразователем адресов, и все эти преобразователи надо будет настроить программно до первого обращения программы к соответствующим адресным областям.

Легко видеть, что описанная схема связи между двумя компьютерами тривиально обобщается на схему, в которой один компьютер – «ведущий» - связан с каждым из многих «ведомых». Ведущим будем называть тот компьютер, который не оснащен собственным NTB. На Рис. 4 это компьютер 2, расположенный справа. Снабдив Root Complex 2 PCI-коммутатором, мы можем подключить к нему несколько (по числу нисходящих каналов коммутатора) компьютеров аналогично компьютеру 1, каждый из которых снабжен своим NTB и, тем самым, связан с ведущим рассмотренным только что образом. Вопрос о том, как в этой схеме подключения организовать связь между ведомыми, рассмотрим в следующем разделе.

1.3. Сеть МВС-Экспресс

Сеть «МВС-Экспресс» реализует коммуникационную среду, основанную на стандарте PCI Express (PCIe). Корнем дерева является особый узел – Root Complex (с точки зрения PCIe), называемый Ведущим. Листья дерева это устройства, подключённые к шине. В случае сети «МВС-Экспресс» таковыми являются Интерфейсные платы (ИП) всех остальных вычислительных модулей ВМ, называемых Ведомыми, которые подключены через коммутатор. Каждая ИП представляет собой NTB, сконфигурированный и настроенный некоторым специальным образом.

Коммутатор (Рис. 5) осуществляет функции пересылки пакетов между ВМ согласно адресу получателя. С логической точки зрения все коммуникации в системе относятся к категории «точка-точка» и соединяют все устройства напрямую. Коммутатор имеет 1 порт Upstream

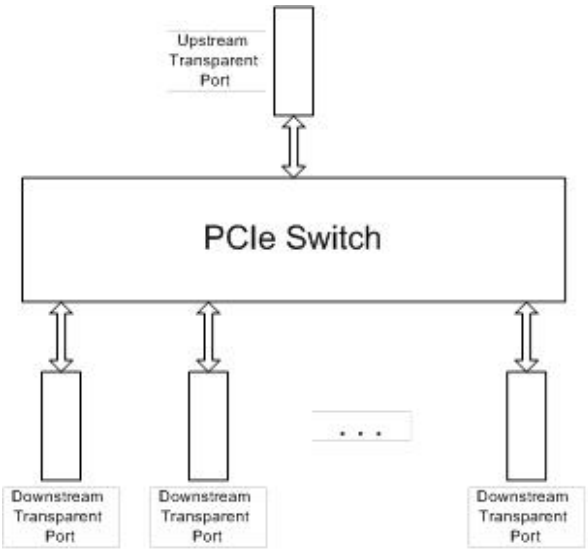

Рис. 5. Общая блок-схема коммутатора МВС-Экспресс

для подключения к Ведущему ВУ и N-1 Downstream портов для подключения к ведомым ВУ.

Вычислительный модуль (ВМ) представляет собой сервер, который имеет на своей материнской плате магистраль PCIe. Таким образом, реализована та самая схема соединения одного ведущего со многими ведомыми, которая была упомянута в конце предыдущего раздела. Ниже будет конкретно показано, как в этой схеме подключения реализовать связь между ведомыми, а не только каждого из ведомых с ведущим.

Интерфейсная плата имеет два порта – Upstream Port, используемые для подключения к PCIe ВМ, и Downstream Port, используемых для подключения к коммутатору PCIe через кабель. В состав интерфейсной платы входит микросхема, которая может работать в двух режимах: либо как NTB, либо как вырожденный PCIe-коммутатор «1 в 1». Режим работы задается конфигурационной прошивкой. Режим вырожденного коммутатора, или прозрачного порта, используется только для подключения Ведущего (главного Root Complex) к коммутатору. Он необходим только из соображений конструктивного исполнения – если бы в стандартных серверах был предусмотрен кабельный разъем PCIe, то ведущий узел вообще не нуждался бы в интерфейсной плате. Во всех остальных узлах (Ведомых) для подключения к коммутатору используется режим непрозрачного порта (NTB).

Организация логики работы интерфейсной платы осуществляется специальной конфигурационной прошивкой. Как уже говорилось в предыдущем разделе, ее задача – обеспечить в момент начального конфигурирования магистрали наличие на обеих сторонах NTB адресных областей (правильно называемых memory BAR) необходимого размера и в необходимом количестве. Для подробного рассмотрения необходимых настроек преобразования адресов введем и уточним некоторые понятия.

Локальная порция — часть системной памяти по известному фиксированному адресу, которая выделяется при загрузке ядра операционной системы (ОС) на ВМ. Память эта по умолчанию не используется ОС, но может быть сделана доступной приложению при помощи специального драйвера. Именно локальная порция каждого ВМ предоставляется им для стороннего доступа.

Адресные области на разных сторонах NTB, о которых шла речь выше, в терминах PCI привязаны к базовым адресным регистрам (BAR) и различаются по номерам этих регистров (от 0 до 5). В МВС-Экспресс используются области, привязанные к регистрам 2 и 4, и мы далее будем называть их BAR2 и BAR4.

Теперь рассмотрим настройки преобразования адресов.

BAR4 на каждой из сторон каждого NTB конфигурируется с размером, равным размеру локальной порции, и преобразование адресов при обращении к каждой из этих областей настраивается на локальную порцию узла сети, находящегося по другую сторону NTB. В результате получается схема связи ведущего со всеми ведомыми и каждого из ведомых с ведущим, рассмотренная в конце предыдущего раздела. В адресном пространстве ведущего узла имеется несколько (по числу ведомых) устройств NTB. Обращения к BAR4 каждого из этих NTB автоматически перенаправляются в локальную порцию соответствующего ведомого узла, как это было описано в предыдущем разделе. В полном соответствии с принципом построения сети МВС-Экспресс, ведущий узел видит в своем адресном пространстве локальные порции всех остальных узлов по вполне определенным адресам. В свою очередь, каждый ведомый настраивает преобразование адресов для BAR4 на своей стороне NTB на локальную порцию ведущего, и, тем самым, видит локальную порцию ведущего через BAR4.

BAR2 используется только на стороне NTB, обращенной к ведомому узлу. Таким образом в адресном пространстве каждого ведомого узла, помимо BAR4, имеется еще одна адресная область – BAR2. Ее размер значительно больше, чем размер локальной порции. В каких-то частях BAR2 ведомый узел должен видеть локальные порции остальных ведомых. Добиться этого очень просто.

В адресном пространстве ведущего узла выбирается минимальный выровненный согласно правилам выравнивания BAR диапазон адресов, перекрывающий все окна доступа к ведомым – область перекрестного доступа. Все BAR4 всех NTB ведущего узла, обеспечивающие в совокупности доступ ведущего ко всем ведомым, должны входить в эту область. Теперь достаточно нацелить преобразование адресов BAR2 в каждом из ведомых узлов на область перекрестного доступа – и задача решена. Внутри BAR2 каждого из ведомых узлов теперь есть поддиапазон адресов, обращения к которому автоматически перенаправляются к некоторому другому ведомому. В каждом BAR2 такие поддиапазоны есть для всех других ведомых, что и требовалось получить.

При этом физически обмен данными при перекрестном доступе выполняется не через ведущий узел. Ведущий узел предоставляет ведомым адресное пространство, а построенные аппаратно при начальном запуске PCI Express маршрутные таблицы обеспечивают взаимодействие по кратчайшему пути.

На Рис. 6 представлена обобщенная схема отображения адресных пространств. LP – локальная порция, W(1), …., W(N-1) – окна доступа к ведомым (External Endpoint), все они перекрыты областью перекрестного доступа. Каждое из них отображено на LP соответствующего ведомого. W0 – окно доступа к LP ведущего (Internal Endpoint), отображено на LP ведущего. Окно перекрестного доступа (External Endpoint) отображено на область перекрестного доступа в ведущем. Выбирая в этом окне соответствующее место, один ведомый может увидеть LP другого через область перекрестного доступа.

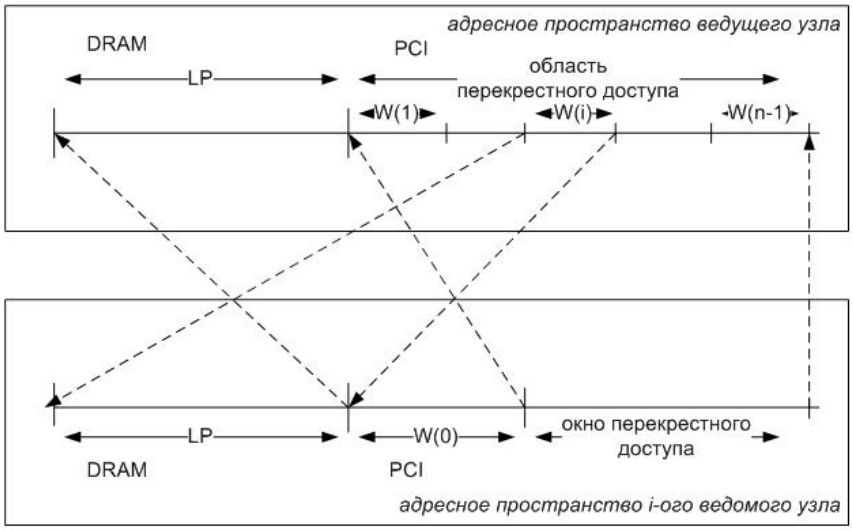

Рис. 6. Обобщенная схема отображения адресных пространств ВМ

Таким образом, любой ВМ может осуществлять прямой доступ к памяти всех ВМ (используемый режим доступа – Master DMA). Также необходимо отметить, что в используемой в настоящее время реализации базового программного обеспечения чтение из чужой памяти осуществляется при помощи программного запроса встречной записи.

1.4. Архитектура МВС-Экспресс

Вычислительный модуль представляет собой сервер, который имеет на своей материнской плате магистраль PCIe.

Как было отмечено в предыдущем разделе, при загрузке ядра операционной системы (ОС) программно-аппаратные средства коммуникационной сети МВС-Экспресс создают в каждом ВМ выделенную область системной памяти, называемую локальной порцией разделяемой памяти ВМ.

Программной настройкой интерфейсных плат сети МВС-Экспресс локальные порции всех ВМ отображаются без пересечения в непрерывную область адресного пространства устройств ввода-вывода шины PCI, и в каждом ВМ известна область адресов, соответствующая локальной порции каждого другого ВМ. Выполнение команды записи по выбранному адресу адресного пространства устройств ввода-вывода шины PCI приведёт к записи в ячейку локальной порции памяти, соответствующую выбранному адресу. Тем самым создаётся разделяемая память объёма, равного сумме объёмов локальных порций всех ВМ.

Таким образом, любой ВМ может осуществлять прямой доступ к памяти всех ВМ (используемый режим доступа к памяти в удаленных ВМ – Master DMA).

Собственно, в такой организации отображения локальных порций памяти ВМ в непрерывную область адресного пространства устройств ввода-вывода шины PCI заключается ограничение количества N объединяемых ВМ: адресное пространство памяти ВМ (за вычетом системной памяти, установленной в ВМ) должно быть достаточным для отображения всех N локальных порций разделяемой памяти.

Если используется ВМ с адресным пространством 2^{32}, то при объединении 128 ВМ размер разделяемой всеми ВМ памяти не превосходит 32 Мбайта, что по современным меркам очень мало.

Поэтому при использовании предлагаемой технологии создания коммуникационной среды МВС-Экспресс желательно применять аппаратные средства, поддерживающие работу PCI Express в рамках адресного пространства 2^{64}. В современных BIOS 64-разрядный ввод-вывод поддержан не полностью соответствующим спецификации PCI Express (в связи с отсутствием практической потребности в традиционных серверах). Поэтому для сети МВС-Экспресс пришлось разработать технику ручной перена-

стройки магистрали PCI Express при помощи специальных скриптов.

Следует также отметить, что реальный объем физического адресного пространства в современных серверах общего назначения далек от 2^{64} и зачастую не превышает 2^{40}. На первый взгляд, это много, но с учетом объемов системной памяти современных серверов, а также не рассматриваемых в этой работе средств масштабирования сети МВС-Экспресс с использованием нескольких коммутаторов, даже этого объема адресного пространства оказывается недостаточно.

Поэтому предоставляется альтернатива:

- создавать параллельные программы, использующие ограниченный объём разделяемой памяти, равный размеру локальной порции;
- создать программных агентов, перемещающих данные из локальной порции, используемой как входной буфер, в область памяти, трактуемую как локальная память распределённого разделяемого адресного пространства всех ВМ.

В последнем случае, в силу того, что для коммуникаций доступна лишь небольшая область памяти узла – локальная порция, использование всей доступной памяти узла реализуется с помощью фонового агента, который прослушивает эту область на предмет наличия запросов, и при их наличии – он, фоновый агент, их выполняет. Для работы фонового агента коммуникационной среды резервируется одно процессорное ядро, которое не доступно для пользовательских задач.

2. Программное обеспечение сети МВС-Экспресс

2.1. Библиотека SHMEM - Экспресс

В сети МВС-Экспресс в качестве базового средства разработки приложений была выбрана библиотека shmem, реализующая модель PGAS и парадигму программирования в стиле односторонних коммуникаций. Общая память реализуется с помощью механизма co-array, как показано на Рис. 7.

Система shmem не стандартизована, и ее реализации для различного оборудования отличаются в незначительных деталях.

Модель программирования shmem подразумевает взаимодействие независимых процессов, каждый – со своей локальной памятью, занумерованных по порядку от нуля, и в этом она похожа на модель программирования MPI. Отличие от MPI состоит в том, что обмены данными между процессами являются односторонними. Чтобы данные были переданы из процесса А в процесс Б, требуется не согласованная активность обоих процессов, как в MPI, а лишь «желание» одного из участников. Например, процесс А может «насильно» послать данные процессу Б без какой-либо ответной активности с его стороны. Процесс Б, в свою очередь, может «насильно» прочитать данные из процесса А. В англоязычных описаниях этой модели ее принято называть «модель put/get», в отличие от принятой в MPI «модели send/receive».

Среди функций библиотеки shmem можно выделить 3 основные группы. Во-первых, это функции организации распределённых массивов. Во-вторых, это функции обращения к удалённой памяти – запись, чтение и атомарные операции. В-третьих, это функции барьерной синхронизации (общей и по списку) и блокировки процессов на время ожидания завершения всех выданных обращений к удалённой памяти.

Работа в дисциплине единой разделяемой памяти ставит перед программистом две проблемы, которых нет при использовании, например, MPI.

Во-первых, передача данных в «чужую» область памяти перестает сопровождаться принудительной синхронизацией с процессом – «хозяином» этой области памяти. Следовательно, отдельные предписания, предназначенные специально для синхронизации процессов, становятся совершенно необходимыми. В модели программирования shmem основным видом синхронизации является барьер. В отличие от большинства реализаций shmem, shmem-экспресс позволяет выполнить барьер как по всем процессам, так и по совершенно произвольному подмножеству процессов.

Во-вторых, выполнение одностороннего обмена подразумевает, что процессу – инициатору обмена известны значения адресов, по которым интересующие его данные расположены в «чужой» памяти. В shmem для решения этой

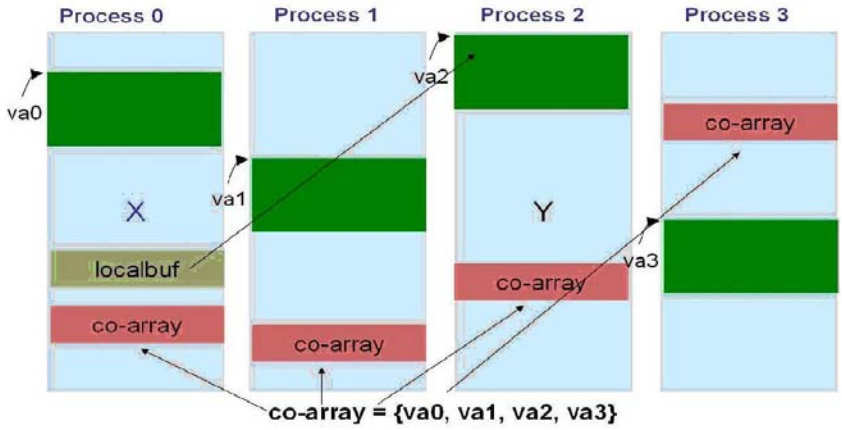

Рис. 7. Адресация памяти в shmem

проблемы применяется частный, но очень простой и потому эффективный прием. Прежде всего, предполагается, что двоичный исполняемый файл для всех процессов – один и тот же. Это автоматически означает, что для многих категорий переменных и массивов адреса одноименных программных объектов в разных процессах совпадают. В частности, в программах на C таковыми являются все переменные и массивы, объявленные статически. Для таких переменных и массивов односторонние обмены можно задавать в стиле: «мой массив A положить в массив B процесса C». Зная адрес своего массива B, процесс – инициатор обмена тем самым знает и его адрес в любом другом процессе и может пользоваться своим адресом в качестве чужого. С другой стороны, к переменным и массивам, объявленным в теле функции без квалификатора «static», операции односторонних обменов данными в стиле shmem просто неприменимы. Адреса этих объектов не только не совпадают в разных процессах, но и не определены статически. Адреса областей памяти, возвращаемые при обращениях к malloc(), занимают промежуточное положение: они определены начиная с момента обращения к malloc() и не изменяются в процессе выполнения программы, но в разных процессах, вообще говоря, отличаются. В большинстве реализаций shmem предусмотрена специальная коллективная операция, выделяющая такие области памяти во всех процессах с гарантированно одинаковыми адресами. Данный подход поддерживается в shmem-экспресс, но не является основным. В качестве основного в shmem-экспресс используется другой подход, основанный на понятии распределенных массивов (co-arrays). В рамках этого подхода адреса массивов, которые программисту удобно считать «одним и тем же массивом в разных процессах», не делаются принудительно одинаковыми при выделении памяти, а просто рассылаются всем процессам для общего сведения.

Для организации распределённого массива каждый процесс должен сначала выделить локально память (адрес локальной памяти – va, ее размер – size), которая будет его локальной частью распределённого массива. Затем с помощью коллективной операции shmem_coarray_all(va, size, coarray) процессы обмениваются указателями на части массивов, получая список указателей в массивах coarray (количество элементов массива coarray равно количеству shmem-процессов). Эти массивы одинаковы на всех узлах. На Рис. 7 показаны расположения локальной памяти (адреса va0, va1, va2 и va3) и массива coarray в виртуальной памяти каждого процесса.

Чтобы процессу X записать в память процесса Y массив localbuf размера size, процессу X достаточно вызвать библиотечную функцию shmem_put(coarray[procY], localbuf, size, procY). Необходимо отметить, что операция записи в shmem-экспресс является не блокирующей. Обеспечение фактического выполнения операции отправки данных с инициирующей стороны возможно вызовом функциий shmem_fence(void). Однако гарантировать запись данных в память

процесса получателя возможно лишь в результате барьерной синхронизации. Для выполнения операции удаленного чтения требуется выполнить операцию shmem_get(coarray[procY], localbuf, size, procY). Гарантией выполнения операции чтения служит барьерная синхронизация.

Технология shmem была разработана и успешно применяется для коммуникационных сетей с очень низкими показателями латентности и задержки записи. Коммуникационная сеть МВС-экспресс является именно такой сетью. Использование для этих сетей программистской модели shmem (а не MPI) не случайно. Оно продиктовано необходимостью дать программисту возможности ускорения и упрощения параллельных программ, которые характерны именно для этого оборудования.

В традиционных коммуникационных сетях, на которые ориентирована модель программирования MPI, запуск обмена, даже совсем короткого, стоит очень дорого. Он эквивалентен времени передачи (в процессе выполнения уже запущенного обмена), как минимум, тысяч байт данных. Технология shmem эффективна в сетях, среднее время передачи отдельного слова удаленному процессу в которых сопоставимо по порядку величины с временем локального присваивания. Это позволяет передавать сложно организованные, разбросанные по памяти мелкими порциями данные не путём запаковки их в одно большое сообщение, как в MPI, а путем выдачи серии из большого количества отдельных односторонних обменов, за которыми следует один акт общей синхронизации. Необходимо отметить, что это касается только односторонней записи, но не чтения. Запуск одностороннего чтения, даже в высокореактивных сетях, стоит примерно столько же, сколько в сетях, традиционно использующих MPI. Несмотря на то, что операции удаленного чтения реализованы в библиотеке shmem-экспресс, их использование, для достижения выкокой производительности, необходимо максимально ограничить.

В библиотеке shmem-экспресс доступны два вида барьерной синхронизации: синхронизация всех процессов с помощью фунции shmem_barrier_all() и выборочная синхронизация процессов по списку shmem_barrier(int nodes, int *barlist, int barbuf), что выгодно отличает ее от других реализаций shmem. Дополнительно реализованы атомарные операции модификации данных.

Кроме того, планируется расширять библиотеку shmem. Одним из самых важных расширений является предоставление программисту возможности использовать многоярусность памяти процессов, выделяя группу процессов, выполняющихся на одном многоядерном процессоре, а также группу процессов на одном многопроцессорном узле, группу процессов на макроузле и так далее. Учёт локализации процессов позволит более эффективно отображать алгоритмы на архитектуру вычислительных систем, простота же библиотеки односторонних коммуникаций shmem делает архитектурно-ориентированное программирование продуктивным.

Интересным расширением представляется также возможность удалённого вызова процедуры – механизм RPC (Remote Procedure Call). Этот механизм в некоторых случаях может сократить объём коммуникаций, когда вместо подкачки большого объёма данных сами вычисления передаются в узел, где эти данные располагаются.

2.2. Высокоуровневые языки

С целью повышения продуктивности программирования вычислительных систем, создаваемых с применением сети МВС-Экспресс, было выполнено портирование языков UPC, CAF, а также библиотеки armci на библиотеку shmem (Рис. 8).

Библиотека MPI является де-факто стандартом при разработке приложений для кластерных суперкомпьютеров, в связи, с чем была выполнена реализация этой библиотеки для использования с коммуникационной сетью МВС-Экспресс.

3. Экспериментальная оценка эффективности сети МВС-Экспресс

Ключевой тест для оценки возможностей решения, применённого в сети МВС-Экспресс, Message Rate. С помощью этого теста измеря-

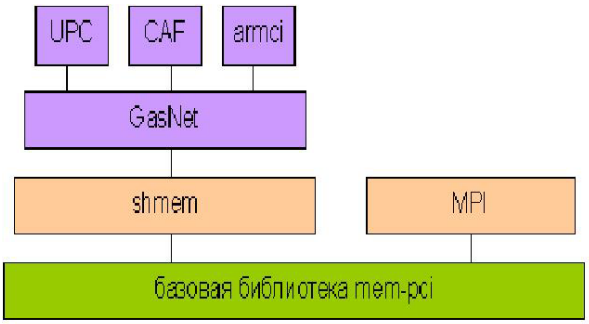

Рис.8 . Средства разработки, доступные при использовании сети МВС-Экспресс

ется темп выдачи сообщений в сеть в режиме, когда один из двух процессов, расположенных на разных узлах, посылает на потоке данные второму процессу. Хотя название теста буквально означает темп выдачи сообщений, результаты приводятся и как в виде количества переданных сообщений в единицу времени, так и в виде достигнутой пропускной способности.

При реализации теста с использованием библиотеки shmem в модели односторонних коммуникаций операции обмена выполняются только посылающим процессом, при использовании MPI в модели двусторонних коммуникаций – и посылающим, и принимающим процессами соответственно.

На Рис. 9 видно, что при меньшей пиковой пропускной способности по сравнению с InfiniBand, реальная пропускная способность сети МВС-Экспресс gen 2 на коротких сообщениях заметно выше, чем у InfiniBand (тестирование выполнялось на суперкомпьютере К-100 [2] ИПМ им. М.В. Келдыша РАН).

Приведённые результаты соответствуют случаю, когда на каждом из 2-х узлов запускается по одному процессу. Существуют модификации теста, в которых для передачи данных используются все доступные ядра узлов, и в этом случае эффективная пропускная способность для Infiniband вырастает, хотя также остаётся на коротких сообщениях ниже пропускной способности МВС-Экспресс.

Измерения темпа выдачи сообщений на тесте Message Rate для Infiniband Mellanox показывают аналогичную картину в сравнении в МВС-Экспресс. Стоит заметить, что в некоторых реализациях MPI (например, в MVAPICH) для повышения результативности на тесте Message Rate разработчики предусмотрели специальные механизмы программной агрегации сообщений (упаковка нескольких MPI сообщений в один сетевой пакет), и в этом случае результаты измерений оказываются значительно выше. Однако в этом случае сетевые пакеты становятся значительно больше, и поэтому такие программные решения неадекватно представляют аппаратные возможности сети при передаче коротких сообщений.

Для оценки эффективности использования сети МВС-Экспресс на приложениях, реализованных с помощью традиционной библиотеки MPI, было выполнено тестирование на базе пакета NAS Parallel benchmark.

В измерениях участвовали четыре узла кластера ПТК [4]. Сравнение выполнялось с коммуникационной сетью QDR Infiniband (Qlogic), с использованием библиотеки Intel MPI 4.0.1 в комплекте с компилятором icc 12.0.

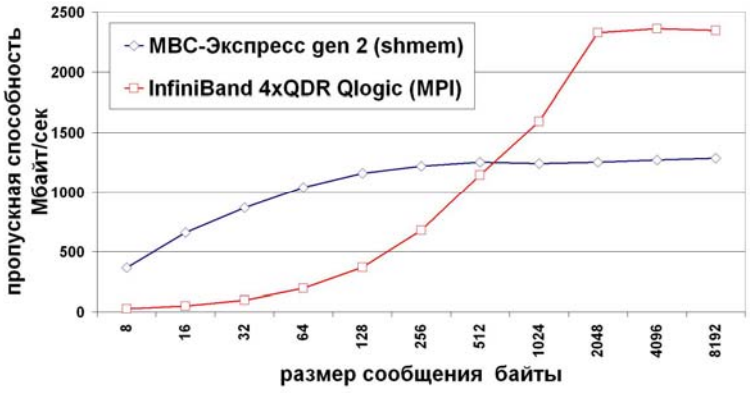

Рис. 9. Пропускная способность на тесте Message Rate, сравнение сетей МВС-Экспресс и InfiniBand Qlogic QDR 4x

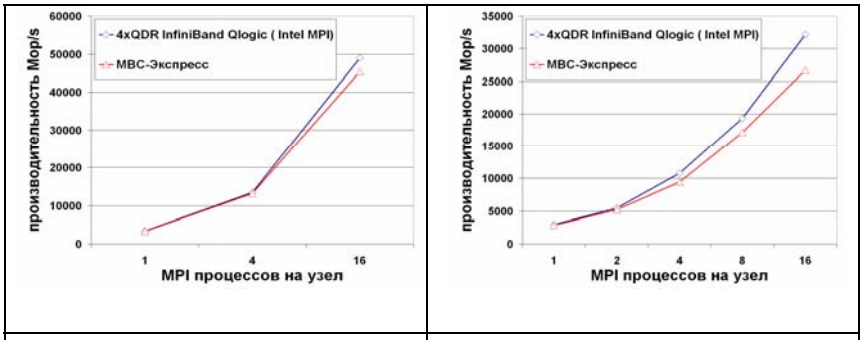

Рис. Результаты сравнительного тестирования сетей МВС-Экспресс и InfiniBand, NAS Parallel benchmark, MPI

Результаты измерений показали, что на тестах, характеризующихся преобладанием больших пересылок (тесты FT Рис. 10) и IS пакета NPB), сеть МВС-Экспресс за счёт меньшей пропускной способности проигрывает сети Infiniband. Отставание составляет 10-15% производительности. При этом на тестах, характеризующихся большой гранулированностью взаимодействий (например, тест SP пакета NPB), при меньшей пропускной способности сети МВС-Экспресс меньшая латентность позволяет практически сравнять производительности, достигаемые для сети МВС-Экспресс и QDR Infiniband.

Также представляется перспективным активно прорабатываемый в настоящее время подход совместного применения двух сетей для выполнения пересылок различного типа с целью эффективного использования обеих сетей.

Заключение

Сеть МВС-Экспресс является комплексным программно-аппаратным решением проблемы построения суперкомпьютера с архитектурой, близкой к архитектуре суперкомпьютеров с общим полем памяти, но с показателями доступности и низкой удельной стоимости, такими же, как у суперкомпьютеров, построенных на основе кластерных технологий из коммерчески доступных комплектующих.

На данный момент совместными усилиями ФГУП «НИИ «Квант» и ИПМ им. М.В. Келдыша отработано и эксплуатируется второе поколение сети МВС-Экспресс, реализованное на базе PCI-Express 2.0. Разрабатывается третье поколение сети на базе PCI-Express 3.0. Ведутся работы по совершенствованию программного обеспечения сети МВС-Экспресс.

Сеть МВС-Экспресс использовалась при построении кластера К-100 в ИПМ им. М.В. Келдыша [2], а также в суперкомпьютерном программно-технологическом комплексе (ПТК) в Санкт-Петербургском государственном политехническом университете [4].

Активно ведутся исследования, направленные на эффективное совместное использование коммуникационных сетей МВС-Экспресс и InfiniBand, как на уже работающих установках, так и при разработке перспективных архитектур иерархических суперкомпьютеров экзафлопсного уровня.

Литература

1. Горбунов В.С., Лацис А.О., Иванов А.Н. О построении суперкомпьютеров на основе интерфейса PCI-EXPRESS // Материалы международной научно-технической конференции "Суперкомпьютерные технологии: разработка, программирование, применение" (СКТ-2010), 27 сентября — 2 октября 2010, Дивноморское, Россия.
2. Давыдов А. А., Лацис А. О., Луцкий А. Е., Смольянов Ю. П., Четверушкин Б. Н., Шильников Е. В. Многопроцессорная вычислительная система гибридной архитектуры МВС-Экспресс. Доклады Академии наук. 2010. т. 434. №4. с. 459-463.
3. Дбар С.А., Лацис А.О., Храмцов М.Ю. Вычислительная система МВС-Экспресс. Руководство программиста. http://www.kiam.ru/MVS/documents/k100/shmemprogman.html
4. Речинский А., Горбунов В.С., Эйсымонт Л.К., Суперкомпьютер с глобально адресуемой памятью // Открытые системы, №7, 2011.

Владимир Константинович Левин. Научный руководитель ФГУП «НИИ «Квант». Окончил МЭИ в 1950 году. Доктор технических наук, профессор, Академик и член бюро Отделения нанотехнологий и информационных технологий РАН. Лауреат Ленинской премии, Государственных премий СССР и РФ. Автор более 100 печатных работ. Область научных интересов: информационные и вычислительные системы. E-mail: levin@rdi-kvant.ru

Борис Николаевич Четверушкин. Директор ИПМ им. М.В. Келдыша РАН. Окончил Московский физико-технический институт в 1966 году. Доктор физико-математических наук, профессор, Академик РАН, Заслуженный деятель науки РФ. Член президиума РАН. Автор более 360 печатных работ и четырех монографий. Область научных интересов: математическое моделирование, прикладная математика, параллельные вычисления. E-mail: office@keldysh.ru

Георгий Сергеевич Елизаров. Директор ФГУП «НИИ «Квант. Доктор технических наук. Автор более 80 печатных работ. Область научных интересов: информационные и вычислительные системы. E-mail: elizarov@rdi-kvant.ru

Виктор Станиславович Горбунов Заместитель директора по научной работе ФГУП «НИИ «Квант». Окончил МЭИ в 1986 году. Кандидат технических наук. Автор более 50 печатных работ. Область научных интересов: высокопроизводительные вычислительные системы. E-mail: gorbunov@rdi-kvant.ru

Алексей Оттович Лацис. Заведующий сектором ИПМ им. М.В. Келдыша РАН. Окончил МГУ им. М. В. Ломоносова в 1980 году. Доктор физико-математических наук. Автор 50 печатных работ и одной монографии. Область научных интересов: архитектура и системное программное обеспечение суперкомпьютеров. E-mail: lacis@kiam.ru

Виктор Владимирович Корнеев. Заместитель директора по научной работе ФГУП «НИИ «Квант». Окончил Алтайский политехнический институт им. И.И. Ползунова в 1970 году. Доктор технических наук, профессор. Автор 126 печатных работ и 5 монографий. Область научных интересов: компьютеры и программное обеспечение. E-mail: korv@rdi-kvant.ru

Алексей Алексеевич Соколов. Начальник отдела ФГУП «НИИ «Квант». Окончил Московский физико-технический институт в 2004 году. Автор 5 печатных работ. Область научных интересов: высокопроизводительные вычислительные системы. E-mail: sokolov@rdi-kvant.ru

Дмитрий Владимирович Андрюшин. Начальник лаборатории ФГУП «НИИ «Квант». E-mail: dv_andryushin@mail.ru
Окончил МГУ им. М. В. Ломоносова в 2004 году. Автор 2 печатных работ. Область научных интересов: вычислительная техника и программирование. E-mail: andryushin@rdi-kvant.ru

Юрий Андреевич Климов. Старший научный сотрудник ИПМ им. М.В. Келдыша РАН. Окончил МГУ им. М. В. Ломоносова в 2002 году. Кандидат физико-математических наук. Автор 40 печатных работ. Область научных интересов: функциональное программирование, оптимизация и преобразование программ, языки программирования, суперкомпьютеры. E-mail: yuklimov@keldysh.ru