УДК 519.6

КОММУНИКАЦИОННАЯ СЕТЬ МВС-ЭКСПРЕСС

Г. С. Елизаров[1], В. С. Горбунов[1], В. К. Левин[1], А. О. Лацис[2], В. В. Корнеев[1], А. А. Соколов[1], Д. В. Андрюшин[1], Ю. А. Климов[2]

Рассматривается подход к построению коммуникационной сети МВС-Экспресс с прямой коммутацией пакетов PCI Express. Предложены средства программирования на базе модели с распределенной общей памятью для 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.

1. Введение. Среди актуальных направлений использования высокопроизводительных вычислительных систем (ВС) выделяется решение задач с малыми пространственной и временно́й 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) — распределенного 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 и CAF, а также перспективных Chapel, X10, Fortress.

Коммерчески доступные коммуникационные сети (InfiniBand, GigabitEthernet и др.), 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 развиваются языки класса PGAS, такие как UPC и CAF, позволяющие работать в общем адресном пространстве на сетях без аппаратной поддержки общей памяти. При этом для 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, заключается в том, что существующие на сегодняшний день коммерческие сети, применяя 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

[1] ФГУП “НИИ “Квант”, 4-й Лихачевский переулок, д. 15, 125438, Москва; Г. С. Елизаров, директор, e-mail: elizarov@rdi-kvant.ru; В. С. Горбунов, зам. директора, e-mail: gorbunov@rdi-kvant.ru; В. К. Левин, академик РАН, науч. руководитель, e-mail: levin@rdi-kvant.ru; В. В. Корнеев, профессор, зам. директора, e-mail: korv@rdi-kvant.ru; А. А. Соколов, нач. лаборатории, e-mail: sokolov@rdi-kvant.ru; Д. В. Андрюшин, нач. лаборатории, e-mail: andryushin@rdi-kvant.ru

[2] ИПМ им. М.В. Келдыша РАН, Миусская пл., д. 4, 125047, Москва; А. О. Лацис, зав. сектором, e-mail: lacis@kiam.ru; Ю. А. Климов, ст. науч. сотр., e-mail: yuklimov@keldysh.ru

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].

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 объема, которое складывается из открытых для общего доступа областей системной памяти каждого из входящих в кластер узлов и целиком доступно каждому из узлов [1, 2]. 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 миллионов 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.

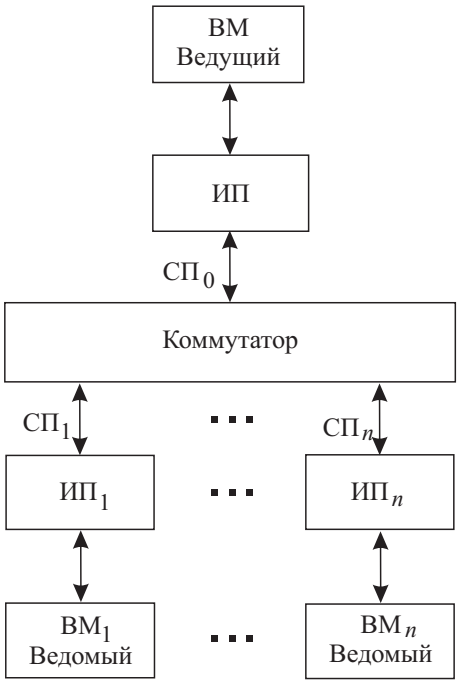

Рис. 1. Структура коммуникационной среды МВС-Экспресс

Разработка сети прямой коммутации пакетов PCI Express, близкая к предлагаемой, одновременно с нами выполнена американской фирмой OneStopSystems (http://www.onestopsystems.com), 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 четвертом разделе приводятся результаты экспериментов по оценке предложенных аппаратно-программных решений. В заключении отмечены достигнутые результаты и намечены перспективы развития предложенных решений.

2. Основы построения сети МВС-Экспресс.

2.1. 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.

Коммуникационная сеть состоит из коммутатора, интерфейсных плат (ИП), которые подключаются к магистралям PCI Express соответствующих вычислительных модулей (ВМ), и 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 разъемом 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”, т.е. 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 коммерчески доступны микросхемы нескольких производителей, включая PLX Technology, для изготовления интерфейсных плат и коммутаторов, причем коммутаторы могут быть как созданы с использованием одной микросхемы, так и построены как составные из нескольких микросхем для увеличения количества портов.

Собственно, приведенная структура характерна для большинства современных коммуникационных сетей. Особенности КС МВС-Экспресс обусловливаются 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), предназначенный для работы с интерфейсной платой, для поддержки синхронизации и когерентного состояния памяти ВМ.

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

2.2. Архитектура МВС-Экспресс. Вычислительный модуль представляет собой сервер, который имеет на своей материнской плате магистраль PCIe. При загрузке ядра операционной системы (ОС) программно-аппаратные средства коммуникационной сети МВС-Экспресс создают в каждом ВМ по известному фиксированному адресу выделенную область системной памяти, называемую локальной порцией разделяемой памяти ВМ. Эта память по умолчанию не используется ОС, но может быть сделана доступной приложению, исполняемому на данном ВМ, при помощи специального драйвера. Именно локальная порция каждого ВМ предоставляется этим ВМ для записи в нее из всех остальных 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 приведет к записи в ячейку локальной порции памяти, соответствующую выбранному адресу. Тем самым создается разделяемая память объема, равного сумме объемов 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). Кроме того, необходимо отметить, что в используемой в настоящее время реализации базового программного обеспечения чтение из чужой памяти осуществляется при помощи программного запроса встречной записи.

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 (Basic Input/Output System) 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 перенастройки 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 настоящей статье средств масштабирования сети МВС-Экспресс с использованием нескольких коммутаторов, даже этого объема адресного пространства оказывается недостаточно.

В этой связи предоставляется альтернатива:

— создавать параллельные программы, использующие ограниченный объем 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 распределенного 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 выполняет эти запросы. Для работы фонового агента коммуникационной среды резервируется одно процессорное ядро, которое не доступно для пользовательских задач.

3. Программное обеспечение сети МВС-Экспресс.

3.1. 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. 3.

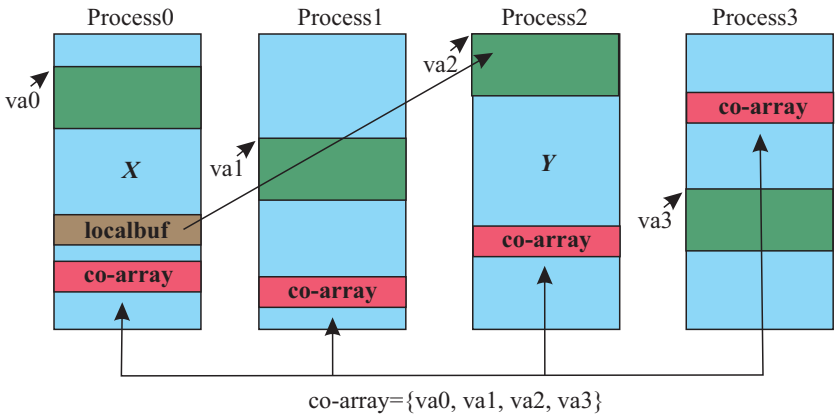

Рис. 3. Адресация памяти в shmem

Система shmem не стандартизована; 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 три основные группы. Во-первых, это функции организации распределенных массивов. Во-вторых, это функции обращения к удаленной 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 удаленной 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 проблемы применяется частный, но очень простой, а потому и 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 Си таковыми являются все переменные и массивы, объявленные статически. Для таких переменных и массивов односторонние обмены можно задавать по схеме “мой массив А положить в массив В процесса С”. 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 не применимы. 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 распределенного массива каждый процесс должен сначала выделить локально память (адрес локальной памяти — `va`, ее размер — `size`), которая будет его локальной частью распределенного 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. 3 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 функций `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 путем 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 высокой производительности необходимо максимально ограничить.

В библиотеке shmem-экспресс доступны два вида барьерной синхронизации: синхронизация всех процессов с помощью функции `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 т.д. Учет локализации процессов позволит более эффективно отображать алгоритмы на архитектуру вычислительных систем, простота же библиотеки односторонних коммуникаций shmem делает архитектурно-ориентированное программи-

рование продуктивным.

Кроме того, интересным расширением представляется возможность удаленного вызова процедуры — механизм RPC (Remote Procedure Call). Этот механизм в некоторых случаях может сократить объем коммуникаций, когда вместо подкачки большого объема данных сами вычисления передаются в узел, где эти данные располагаются.

3.2. Высокоуровневые языки. С целью повышения продуктивности программирования вычислительных систем, создаваемых с применением сети МВС-Экспресс, выполнено портирование языков UPC и CAF, а также библиотеки armci на библиотеку shmem.

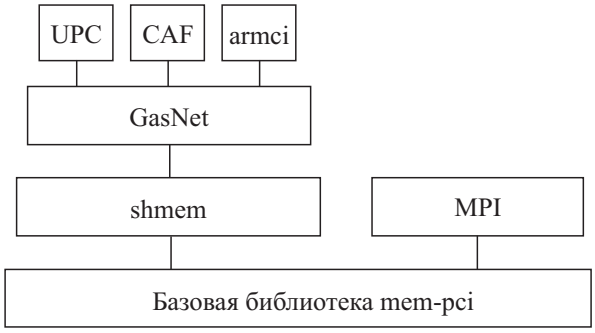

Рис. 4. Средства разработки, доступные при использовании сети МВС-Экспресс

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 (рис. 4).

4. Экспериментальная оценка эффективности сети МВС-Экспресс. Ключевой тест для оценки возможностей решения, примененного в сети МВС-Экспресс, — это тест Message Rate. С помощью этого теста измеряется 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 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.

Рис. 5. Пропускная способность на тесте Message Rate, сравнение сетей МВС-Экспресс и InfiniBand Qlogic QDR 4x:
1 — МВС-Экспресс gen 2 (shmem);
2 — InfiniBand 4xQDR Qlogic (MPI)

На рис. 5 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). Приведенные результаты соответствуют случаю, когда на каждом из двух узлов запускается по одному процессу. Существуют модификации теста, в которых для передачи данных используются все доступные ядра узлов; в этом случае эффективная пропускная способность для Infiniband вырастает, хотя также остается 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. Следует 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); в этом случае результат измерений оказываются значительно выше. Однако в этом случае сетевые пакеты становятся значительно больше, поэтому такие программные решения не адекватно 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 с использованием пакета NAS Parallel benchmark. В измерениях использовались четыре узла кластера ПТК [3]. Сравнение выполнялось с коммуникационной сетью QDR Infiniband (Qlogic) с использованием библиотеки Intel MPI 4.0.1 в

комплекте с компилятором icc 12.0. Результаты измерений показали, что на тестах, характеризующихся преобладанием больших пересылок (тесты FT (рис. 6) и IS пакета NPB), сеть МВС-Экспресс проигрывает сети Infiniband за счет меньшей пропускной способности. Отставание составляет 10-15% производительности. Несмотря на это отставание, на реальных задачах, например это можно увидеть на тесте SP пакета NPB, при меньшей пропускной способности сети меньшая латентность МВС-Экспресс позволяет практически сравнять производительности, достигаемые при использовании сетей МВС-Экспресс и QDR Infiniband.

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

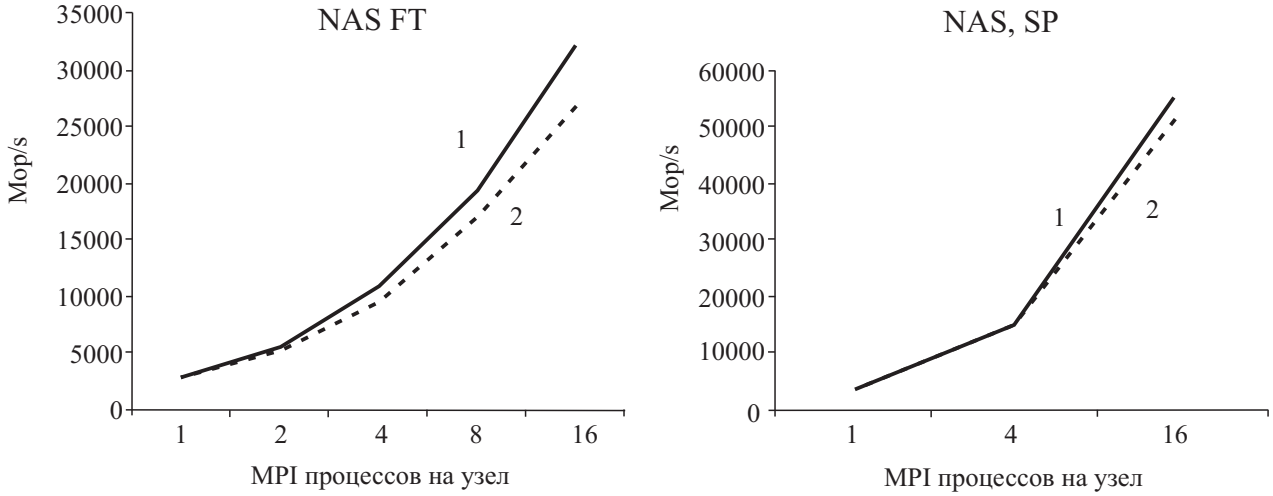

Рис. 6. Результаты сравнительного тестирования сетей МВС-Экспресс и Infiniband, NAS Parallel benchmark, MPI: 1 — Intel MPI (infiniband); 2 — МВС-mpi (МВС-Экспресс)

5. Выводы. 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 [3].

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.

СПИСОК ЛИТЕРАТУРЫ

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). 27.09.2010–02.10.2010. Дивноморское, 2010.
2. *Дбар С.А., Лацис А.О., Храмцов М.Ю.* Вычислительная система МВС-Экспресс. Руководство программиста (http://www.kiam.ru/MVS/documents/k100/shmemprogman.html).
3. *Речинский А., Горбунов В., Эйсымонт Л.* Суперкомпьютер с глобально адресуемой памятью // Открытые системы. 2011. № 7. 26–30.

Поступила в редакцию
15.08.2012