УДК 519.6

РЕАЛИЗАЦИЯ ИНТЕРФЕЙСА ПЕРЕДАЧИ СООБЩЕНИЙ (MPI) ДЛЯ ОТЕЧЕСТВЕННОЙ СИСТЕМЫ МЕЖПРОЦЕССОРНОГО ОБМЕНА (СМПО)

А. М. Варгин, Д. В. Ежов, Ю. Л. Кирьянов
(РФЯЦ-ВНИИЭФ)

Описываются принципы реализации библиотеки функций обмена сообщениями, предназначенной для работы с системой межпроцессорного обмена, разработанной и функционирующей в РФЯЦ-ВНИИЭФ. Библиотека в полном объеме отвечает требованиям международного стандарта MPI-1.

Введение

Важнейшей частью любой высокопроизводительной параллельной вычислительной системы является коммуникационная система. Она определяет скорость коммуникационных обменов между процессорными элементами и, в конечном итоге, эффективность всей системы в целом. При разработке коммуникационных систем применяются новейшие достижения в области проектирования как аппаратного, так и программного обеспечения (ПО).

В РФЯЦ-ВНИИЭФ разработана собственная система межпроцессорного обмена (СМПО) [1, 2], использующая оригинальный, не совместимый с широко распространенными коммуникационными системами аппаратный протокол передачи сообщений. Поэтому использование существующего коммуникационного программного обеспечения (КПО) для собственной системы межпроцессорного обмена без существенных доработок невозможно. И, как следствие этого, — необходимость создания собственного КПО для реализации интерфейса передачи сообщений между пользовательскими процессами на базе СМПО.

В качестве базового пакета для реализации интерфейса передачи сообщений рассматривались три системы с открытым программным кодом:

- MPI/МП-3;
- LAM-MPI;
- MPICH версии 1.2.

MPI/МП-3 — базовая реализация интерфейса передачи сообщений для мультипроцессорной системы МП-3 [3, 4]. Этот вариант не взят за основу в силу своей значительной привязанности к аппаратуре МП-3. В этом варианте реализована лишь одна модель низкоуровневой передачи пакетов — модель передачи с подтверждением (аналог Rendezvous в MPICH), накладные расходы для передачи коротких сообщений в которой выше, чем в других моделях. Кроме того, при выборе данного варианта пришлось бы полностью пересмотреть концепцию реализации глобальных обменов, так как в МП-3 они осуществлялись через общую шину.

LAM-MPI — система для разработки и выполнения MPI-программ для гетерогенных UNIX-сред. Преимущество этой системы в том, что она включает реализацию MPI-2, основной недостаток — что данный продукт разрабатывался для использования с протоколом TCP/IP, а это влечет за собой как большие накладные расходы (следовательно, большую латентность и низкую скорость передачи), так и плохую переносимость на другие протоколы.

MPICH версии 1.2 — реализация стандарта MPI-1, разработанная Арагонской национальной лабораторией, на сегодня одна из самых используемых систем. Благодаря реализованной технологии ADI2 (Abstract Device Interface) имеется возможность создания собственных устройств с их полной функциональностью. Для технологии ADI2 в MPICH существует интерфейс Chameleon (канальный интерфейс), который предоставляет разработчику свои интерфейсные функции и прототипы функций (в базовом варианте bsend, brecv, nprobe), что обеспечивает простоту пере-

носа и создания собственных устройств. В состав MPICH входит система профилирования, графического отображения информации, трассировки, отладки, а также большой набор внутренних тестов. В MPICH-1.2 включены расширения стандарта MPI2 — поддержка многопотоковых приложений и параллельный ввод-вывод (MPI-IO).

Пакет MPICH поддерживает три вида низкоуровневых передач сообщений:

Inlining — используется для сообщений небольшой длины. Для данного типа характерно уменьшение накладных расходов при передаче за счет встраивания тела основного сообщения в управляющее;

Eagern — механизм передачи без подтверждения, т. е. посылается сначала управляющее сообщение, затем основное;

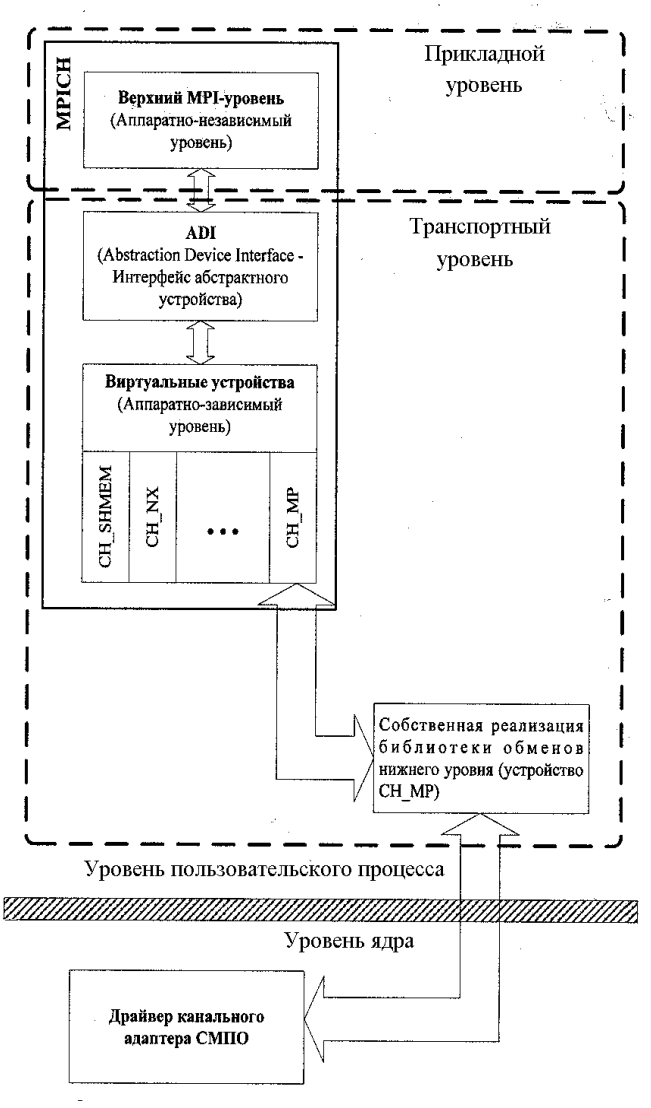
Rendezvous — механизм передачи с подтверждением. Используется для сообщений большой длины. Особенность данного типа передачи в том, что реально она начинается лишь тогда, когда на принимающей стороне выставлен соответствующий запрос и об этом сообщено передающей стороне.

После анализа функциональных возможностей, объема вновь разрабатываемого и модифицируемого кода, документированности и т. д. в качестве базового пакета для реализации MPI для СМПО был выбран MPICH.

В MPICH библиотека обмена сообщениями реализована с использованием так называемого виртуального устройства, осуществляющего связь между аппаратно-зависимой и аппаратно-независимой частями пакета. Поскольку пакет допускает создание новых виртуальных устройств, помимо тех, которые есть в базовом варианте, при реализации библиотеки была полностью сохранена структура пакета MPICH, что, в свою очередь, обеспечило реализацию стандарта MPI.

Библиотека включает в себя низкоуровневые функции приема/передачи, функции интерфейса с драйвером канального адаптера СМПО, функции интерфейса с системой маршрутизации и систему очередей для обработки поступающих пакетов и формирования из них сообщения.

Реализация библиотеки выполнена на языке C. Структура пакета MPICH, его взаимодействия с библиотекой функций и драйвером канального адаптера показаны на рисунке.



Структура взаимодействия составных частей пакета

Уровни ПО обменов

Для обеспечения механизма передачи сообщений было разработано следующее ПО вычислительного модуля (ВМ):

- драйвер канального PCI-адаптера СМПО;
- система маршрутизации коммуникационной системы;
- библиотека нижнего уровня передачи сообщений на базе пакета MPICH (устройство CH_MP).

В данной статье основное внимание уделено описанию библиотеки нижнего уровня передачи сообщений.

Во время разработки библиотеки нижнего уровня были реализованы структуры и функции для обеспечения работы системы очередей и

транспортного механизма. Рассмотрим эти компоненты более подробно.

Система очередей

Система очередей представляет собой локальный пул принятых процессом пакетов данных и набор управляющих функций. Каждый пакет состоит из заголовка и собственно данных. В заголовке содержится вся необходимая информация для однозначной идентификации сообщения (ранг отправителя, ранг получателя, идентификатор сообщения, тип, длина, количество пакетов в сообщении, контрольная сумма сообщения) и пакета (порядковый номер пакета в сообщении, длина, контрольная сумма пакета). Данные в очереди могут находиться сколь угодно долго, пока с верхнего уровня не поступит запрос на их извлечение.

Для работы с очередями предусмотрены функции инициализации, закрытия и добавления пакета.

Механизм очередей работает следующим образом. Если в буфере драйвера обнаруживается пакет, адресованный данному процессу, он извлекается и добавляется в свободный элемент очереди. Если этот пакет является последним в сообщении, оно помечается как полностью принятое путем размещения его идентификатора (части заголовка последнего пакета, неизменной для всех пакетов сообщения) во вспомогательной очереди. Затем увеличивается на единицу значение переменной, отражающей количество готовых сообщений. Необходимость во вспомогательной очереди обусловлена тем, что количество пакетов, составляющих сообщение, может варьироваться в зависимости от длины передаваемого сообщения. Длина пакета не может превышать значения величины, определяемой аппаратурой как длина буфера для передачи по аппаратным каналам.

При добавлении пакета происходит контроль корректности передачи данных путем вычисления контрольной суммы принятого пакета и сравнения ее с контрольной суммой, содержащейся в заголовке, вычисленной на передающей стороне. Если эти суммы не совпадают, выполняется аварийное завершение программы с печатью соответствующей диагностики.

При вызове функции приема происходит обращение к очереди, содержащей идентификаторы готовых сообщений, и при обнаружении нужного происходит его извлечение из основной очереди. Далее, в качестве дополнительного контроля выполняется подсчет контрольной суммы для всего сообщения и сравнение ее со значением, содержащимся в заголовке. Если они совпадают, происходит передача сообщения на верхний уровень с последующим его удалением из очереди. Если же не совпадают, то программа аварийно завершается с печатью соответствующей диагностики.

Взаимодействие с драйвером

Интерфейсом между канальным адаптером СМПО и операционной системой является драйвер, работающий на уровне ядра. Драйвер предоставляет возможность работать с данным аппаратным обеспечением как с обычным символьным устройством. Для взаимодействия с драйвером канального PCI-адаптера в данной реализации системы очередей предусмотрены функции открытия/закрытия устройства, приема/передачи данных.

Для ускорения отправки данных при открытии устройства выполняется отображение буфера драйвера в память пользовательского процесса, что позволяет избежать лишних операций копирования *память-память*.

Работа с общей памятью

Для передачи сообщений между процессами, выполняющимися на одном вычислительном модуле, используется механизм общей памяти. В этом случае не происходит обращения к драйверу канального PCI-адаптера СМПО, а выполняется непосредственное взаимодействие между процессами. Это позволяет разгрузить аппаратуру и ускорить обмен.

Для работы с общей памятью предусмотрены функции инициализации, закрытия, отправки/приема данных.

При отправке сообщения происходит анализ соответствия рангов процесса-отправителя, процесса-получателя и физических номеров вычислительных модулей, на которых они выполняются. В случае совпадения этих номеров обмен выполняется через общую память, иначе — через драйвер канального PCI-адаптера СМПО. Синхронизация работы процессов с общей областью памяти происходит с помощью механизма семафоров.

Транспортный механизм MPI

В состав транспортного механизма входят сервисные функции, функции подготовки и инициализации обменов, а также функция опроса в неблокирующем режиме очереди принятых сообщений. Непосредственно обмен выполняется либо обращением к драйверу, либо через общую память, как было описано выше. Подготовка обмена включает в себя разбиение сообщения на пакеты, дополнение их маршрутным заголовком, созданным на основе анализа таблицы соответствия рангов процессов (отправителя и получателя) и физических номеров вычислительных модулей, на которых эти процессы выполняются. Функция опроса проверяет наличие запрашиваемого сообщения в очереди.

Сервисные функции включают в себя функции генерации таблицы соответствия MPI-рангов процессов и физических номеров узлов, функцию печати диагностики, функцию аварийного завершения.

Механизм запуска заданий

На языке сценариев `Perl` была написана программа для генерирования скриптов запуска и завершения задания на вычислительных модулях.

Файл запуска задания передает значения переменных среды для каждого запускаемого процесса, определяющие рабочую директорию, количество потоков, количество процессов, ранг процесса, логическое имя символьного устройства СМПО, путь к конфигурационному файлу. Значения переменных среды передаются с помощью программы `envarg`, которая является авторской модификацией программы `env`.

Конфигурационный файл содержит таблицу соответствия MPI-рангов процессов, логических имен символьных устройств СМПО и физических имен вычислительных модулей. Основным преимуществом конфигурационного файла является то, что таблицу соответствия можно построить таким образом, что процессы, обменивающиеся между собой наиболее интенсивно, будут физически располагаться наиболее близко друг к другу, используя для коммуникации наименьшее количество маршрутов. Это позволяет разгрузить коммуникационную систему за счет уменьшения количества транзитных передач и увеличить скорость обменов. Другими словами, с помощью конфигурационного файла можно настроить конкретное задание для конкретной топологии.

Скрипт завершения задания содержит набор команд `kill` для каждого процесса, составляющего задание, на каждом вычислительном модуле.

Заключение

В результате данной работы для отечественной СМПО было создано надежно работающее ПО. Испытания показали, что накладные расходы при работе библиотеки (анализ сообщения, разбиение его на пакеты, добавление заголовка и посылка запроса драйверу канального адаптера) составляют $\sim 0{,}01\,\%$ от общих накладных расходов на отправку/прием сообщения (накладные расходы библиотеки очередей, драйвера канального PCI-адаптера СМПО и самой системы межпроцессорного обмена).

Направление дальнейших работ над библиотекой связано с оптимизацией отдельных ее функций с целью более полного использования возможностей, заложенных в аппаратуре СМПО.

Список литературы

1. *Попов В. С., Степаненко С. А., Холостов А. А.* Архитектура аппаратных средств системы межпроцессорного обмена мультипроцессорной системы МП-Х // Вопросы атомной науки и техники. Сер. Математическое моделирование физических процессов. 2002. Вып. 4. С. 61—64.
2. *Байков Э. Г., Басалов В. Г., Варгин А. М. и др.* Система межпроцессорных обменов для мультипроцессорных сред // Межд. семинар "Супервычисления и мат. моделирование". Саров, 6—10 октября 2003 г.
3. MPI: A Message Passing Interface standart // Message Passing Interface Forum. March 22, 1994.
4. *Сапронов С. И., Шумилин В. В.* Реализация функций MPI для мультипроцессорной системы МП-3 // Вопросы атомной науки и техники. Сер. Математическое моделирование физических процессов. 2000. Вып. 3. С. 62—67.

Статья поступила в редакцию 16.03.04.