МРІ-модель организации параллельных вычислений в кластерах гетерогенного типа

Г.Т. Мачарадзе

АО НПЦ «ЭЛВИС», г. Москва, gmacharadze@elvees.com

Аннотация — Построение кластеров на базе гетерогенных вычислительных систем требует применения специальных программных средств, обеспечивающих эффективное выполнение прикладного программного обеспечения. В работе рассматривается программная модель, в основе которой лежат MPI, OpenCL, драйвер ОС Linux, а также служебных процессов mpi-run. Служебный процесс в рамках вычислительного узла осуществляет порождение вычислительных MPI-процессов, используя OpenCL. Взаимодействие между процессами осуществляется путём передачи сообщений: процессы в рамках одного вычислительного узла – через общую память, процессы разных вычислительных узлов – через механизм сетевых сокетов. Экспериментальное исследование модели осуществлялось на российском гетерогенном СнК RoboDeus с использованием шестнадцати высокопроизводительных ядер ELcore-50, разработанных АО НПЦ «ЭЛВИС». Полученные результаты функций BLAS, а также теста HPL при решении задачи с плотными матрицами в формате плавающей точкой двойной точности подтвердили эффективность предложенной MPI-модели.

Ключевые слова — кластерные вычислительные системы, параллельные вычисления, гетерогенные вычислительные системы, проблемно-ориентированные ядра, процессор 1892BM248, RoboDeus, MPI, OpenCL, HPL.

I. ВВЕДЕНИЕ

Высокопроизводительные вычисления (HPC) выполняются на вычислительных кластерах, состоящих из большого числа вычислительных узлов, объединенных высокопроизводительной вычислительной сетью и представляющих единую вычислительную среду. Основным стандартом программирования вычислительных кластеров, построенных на основе гомогенных вычислительных узлов, является интерфейс передачи сообщений (Message Passing Interface, MPI) [1]. В основе стандарта MPI лежит передача сообщений между параллельными процессами, исполняющимися на различных вычислительных узлах кластера.

В последнее время ввиду ослабления закона Мура наблюдается резкий переход к гетерогенным вычислительным архитектурам, отличающиеся высокой энергоэффективностью и производительностью при решении определенного класса задач. Это достигается за счет применения

наряду с процессором общего назначения (центральный) проблемно-ориентированных ядер, предназначенных для высокоэффективной обработки данных. Как правило, подобные системы реализуются в виде системы на кристалле (СнК), то есть память, вычислительные ядра, а также внешние интерфейсы располагаются на одном кремниевом кристалле. Такой способ размещения позволяет минимизировать задержки прохождения сигналов между различными элементами СнК.

Подобные гетерогенные СнК являются асимметричными, то есть центральный процессор CPU выполняет роль ведущего, а остальные – ведомые, управление которыми осуществляется со стороны ведущего. Таким образом, использование только MPI аналогично гомогенным вычислительным кластерам не применимо к кластерам гетерогенного типа. Целью данной работы является адаптация MPI-модели к гетерогенным вычислительным кластерам на базе многоуровневой модели программирования с учетом особенностей гетерогенных СнК.

II. ОБЪЕКТ ИССЛЕДОВАНИЯ

В качестве объекта исследования в данной работе используется гетерогенная система на кристалле 1892BM248 RoboDeus, разработанная российским дизайн-центром АО НПЦ «ЭЛВИС» [2]. СнК RoboDeus предназначена для решения задач компьютерного зрения. Гетерогенные кластеры, построенные на RoboDeus, эффективны в задачах видеоаналитики, анализа больших данных (спутниковой информации), моделирования поведения сложных систем (HPC), задачах прогнозирования и др. Эффективность решения таких задач обусловлена наличием 16 сигнальных процессоров (DSP) ELcore-50 с суммарной производительностью 4 TFlops при вычислениях с одинарной точностью (FP32). Помимо DSP-ядер в состав СнК входят (таблица 1) центральный восьмиядерный процессор MIPS64r6 I6500, графический ускоритель GPU PowerVR Dutton производительностью 600 GFlops FP32, процессор обработки изображений ISP PowerVR Felix, процессор вывода изображений PDP Chimera, а также аппаратный блок кодирования и декодирования, поддерживающие стандарты HEVC, H.264, VP8, VP9, MPEG4, MPEG2.

Все устройства в рамках СнК RoboDeus объединены высокопроизводительной накристалльной сетью (NoC,

Network on Chip) и имеют доступ к общей накристалльной памяти. Эффективность обработки с использованием DSP достигается за счет наличия скалярного и векторного сопроцессоров, использующие единый регистровый файл.

Векторная расширение ELcore vector extension (EVX) дополняет базовую систему инструкций процесса ELcore-50 для обеспечения высокоэффективной параллельной векторной обработки, включая тензорные инструкции умножения матриц и фильтрации (рис 1).

Одновременное исполнение скалярных и векторных инструкций достигается за счет использования VLIW архитектуры (Very Long Instruction Word) – архитектуры с широким командным словом, вследствие чего ядро ELcore-50 поддерживает одновременное исполнение до 8 инструкций в каждом такте. Применение VLIW-архитектуры помимо повышения производительности также обеспечивает синхронизацию между скалярным и векторным каналами.

Таблица 1

Состав СнК RoboDeus

№	Наименование	Краткое описание
1	CPU	Центральный процессор MIPS64r6 I6500, 8 ядер
2	DSP	16 сигнальных процессоров ELcore-50, 4 Tflops fp32
3	GPU	Графический процессор PowerVR Dutton, 600 Gflops fp32
4	ISP	Процессор обработки изображений PowerVR Felix, 4K@60 FPS
5	Encoder/decoder	16 потоков 1080p@60 FPS; кодирование HEVC, H.264; декодирование HEVC, H.264, VP8, VP9, MPEG4, MPEG2
6	Pixel Display Pipeline (PDP)	Процессор вывода кадров на дисплей Chimera, 4K@60 FPS

Рис. 1. Структура ядра ELcore-50

III. MPI-МОДЕЛЬ ДЛЯ КЛАСТЕРОВ ГЕТЕРОГЕННОГО ТИПА

Для реализации MPI-модели была выбрана многоуровневая модель программирования, включающая драйвер ОС Linux, библиотеки OpenCL [3] и MPI. Диаграмма компонентов приведена на рис. 2.

Рис. 2. Диаграмма компонентов реализации MPI-модель

Операционная система отвечает за взаимодействие с аппаратным обеспечением СнК RoboDeus. Драйвер elcore50, являясь частью операционной системы, реализуют механизм постановки вычислительных задач на процессорные ядра, поддержку системных вызовов

для ELcore-50 и предоставляют программный интерфейс (API, application programming interface) высокоуровневому ПО. В драйвере реализован механизм zero copy [4] для избежания дополнительного копирования данных между CPU и DSP.

OpenCL [3] (Open Computing Language) – стандарт для гетерогенных вычислений. Стандартная технология OpenCL поддерживает параллелизм уровня задач и данных, явные механизмы управления памятью и единое адресное пространство, что делает процесс разработки параллельных приложений более гибким. Универсальность достигается за счёт унификации различного класса устройств. Библиотека OpenCL реализована поверх драйвера elcore50 и предоставляет высокоуровневый механизм разработки параллельных приложений на DSP ELcore-50. Результаты профилирования реализации OpenCL [5] (таблица 2) обеспечивает минимальное значение вносимой задержки.

Таблица 2

Результаты профилирования запуска задачи с использованием реализации технологии OpenCL

Название этапа	Время выполнения этапа, мкс
Подсчёт числа аргументов	2
Настройка аргументов для задания	72
Настройка ELF-секций задания	632
Создание объекта для виртуальной памяти	60
Настройка кэш-памяти	< 1
Установка регистров	1
Копирование программы	< 1
Дополнительные расходы	15
Итого:	783

Библиотеки `openmpi` и `elcore_mpi` (рис. 2) реализуют программный интерфейс MPI для CPU и DSP соответственно.

Компонент `elcore-runtime` – является библиотекой среды выполнения на ELcore-50. Библиотека содержит необходимые функции для полноценной разработки на ELcore-50: код инициализации; реализацию системных вызовов; API для работы с DMA.

Программа MPI состоит из автономных процессов, выполняющих свой код в стиле MIMD на CPU и ядрах

ELcore-50. Коды, выполняемые каждым процессом, могут отличаться. Запуск вычислительных процессов в рамках SnK (хоста) осуществляется служебным процессом `mpi-run`, используя API OpenCL.

Взаимодействие между процессами осуществляется через вызовы примитивов связи MPI. В зависимости от типа рассматриваемых процессов применяются различные механизмы при реализации интерфейса MPI.

В рамках рассматриваемой MPI-модели существует два типа вычислительных процессов (рис. 3) [6]:

- процессы, выполняемые на одном хосте – локальные. Обмен сообщениями осуществляется через общую накристалльную память.
- процессы, выполняемые на разных хостах – глобальные. Обмен сообщениями осуществляется через вычислительную сеть, используя механизм сокетов.

Рис. 3. Способы взаимодействия процессов в рамках кластера

Для каждого хоста в рамках вычислительного кластера используется общая циклическая очередь для локальных и глобальных (входящих и исходящих) сообщений.

Очередь делится на две части: кэшируемую для данных и некэшируемую для заголовков (рис. 4). Использование некэшируемой памяти обусловлено необходимостью уменьшения времени на синхронизацию процессов при одновременной записи сообщений.

Рис. 4. Структура MPI-очереди

Запись в неэкэшируемую очередь заголовков ведётся несколькими процессами и требует синхронизации. При чтении синхронизации не требует, при этом допускается копирование содержимого очереди в локальную кэшируемую память процесса.

Структура заголовка имеет следующий вид:

- commId - уникальный номер коммуникатора (берётся из коммуникатора).
- srcRank - номер узла-отправителя в коммуникаторе.
- dstRank - номер узла-получателя в коммуникаторе.
- dstHost - номер хоста-получателя нужный для маршрутизации (соответствует одному из экземпляров служебного процесса mpi-run).
- dstMask - маска процессов-получателей на хосте.
- srcHost - номер хоста-отправителя (соответствует одному из экземпляров служебного процесса mpi-run).
- recvAck - подтверждение приёма.
- msgFunc - код функции, отправившей сообщение.
- msgSize - размер сообщения.
- msgData - смещение в кэшируемом буфере данных.
- msgTag – тег, используемый в некоторых функций.

При отправке сообщений между вычислительными процессами одного и того же хоста:

- 1) Процесс-отправить вызывает MPI-функцию, которая создаёт запись в неэкэшируемой очереди заголовков, а также размещает данные в кэшируемой очереди.
- 2) Принимающий процесс вызывает парную MPI-функцию, которая анализирует состояние неэкэшируемой очереди. При появлении сообщений, данные из кэшируемой очереди копируются в адресное пространство принимающего процесса.
- 3) Принимающий-процесс сбрасывает соответствующий бит в маске процессов-получателей dstMask. Сообщение из очереди удаляется после того, как значение маски dstMask станет равно нулю.

При отправке сообщений между вычислительными процессами разных хостов:

- 1) Процесс-отправить вызывает MPI-функцию, которая создаёт запись в неэкэшируемой очереди заголовков, а также размещает данные в кэшируемой очереди.
- 2) Служебный процесс mpi-run обнаруживает в очереди сообщение, предназначенное другому хосту (поле dstHost). Данное сообщение через

механизм сетевых сокетов передаётся служебному процессу соответствующего хоста.

- 3) Служебный процесс принимающего хоста копирует данное сообщение в свою очередь, после чего вычислительные процессы получают к нему доступ

Таким образом, предложенная многоуровневая программная модель реализует различные подходы для реализации коммуникаций между локальными и глобальными вычислительными процессами, что позволяет более эффективно использовать вычислительные ресурсы.

IV. РЕЗУЛЬТАТЫ ЭКСПЕРИМЕНТАЛЬНЫХ ИССЛЕДОВАНИЙ

Оценка эффективности разработанной MPI-модели осуществлялась на функциях линейной алгебры библиотеки BLAS (Basic Linear Algebra Subroutines) [7] тесте HPL (High-Performance Linpack) [8], который ориентирован на работу с плотными матрицами в формате плавающей точки двойной точности (FP64) и активно использующий функции BLAS (таблица 3) [5]. Тест HPL направлен на получение наивысшей производительности на вычислительных системах.

Таблица 3

Доля времени функций BLAS при выполнении теста HPL

N	1024	2048	4096	8192
Доля времени функции DGEMM – матричное умножение, %	11	37	60	80
Доля времени функции DTRSM – решение СЛАУ с треугольной матрицей, %	4	6	6	5

Экспериментальное исследование проводилось путем эмуляция вычислительного кластера с использованием одной СнК RoboDeus. В рамках данного исследования запускалось несколько служебных процессов mpi-run, каждый из которых порождал группу вычислительных процессов. Вычислительные процессы в рамках своей группы общаются через общую память. Вычислительные процессы из разных групп взаимодействуют через друг с другом через соответствующие служебные процессы, используя механизм сокетов.

Результаты запуска подпрограмм линейной алгебры третьего уровня на ядрах ELcore50 приведены на рисунках 4-5.

Текущие результаты запуска теста HPL приведены в таблице 4. Использование стандартного теста, на основе которого строится международный рейтинг суперкомпьютеров ТОП-500 [9], в качестве эталона позволяет проводить исследование и достоверный

сравнительный анализ вычислительных устройств и систем мирового уровня.

Рис. 4. Результаты запуска подпрограмм литейной алгебры библиотеки BLAS (функция DGEMM) на ядрах ELcore50

Рис. 5. Результаты запуска подпрограмм литейной алгебры библиотеки BLAS (функция DTRSM) на ядрах ELcore50

Таблица 4

Результаты запуска теста HPL на процессоре RoboDeus (1892BM248)*

Размерность, N	Результаты теста HPL при частоте процессора 750 МГц, Gflops
35000	296.50
45000	330.01
55000	343.05
60000	329.75

* Тест запускался при NB=256, P=1, Q=16.

V. ЗАКЛЮЧЕНИЕ

В работе продемонстрировано применение MPI-модели в качестве программной модели для гетерогенных СнК серии «Мультикор» для решения задач высокопроизводительной обработки. Результаты показали, что архитектура СнК с использованием ядер ELcore позволяет эффективно выполнять высокопараллельные вычисления, включая тензорные операции матричного умножения и фильтрации.

Новыми результатами проведённых исследований являются способ реализации MPI-модели на гетерогенных вычислительных ядрах с использованием технологии MPI, OpenCL, а также использование служебного процесса mpi-run для запуска и коммуникации процессов.

Полученные численные результаты по производительности на подпрограммах линейной алгебры третьего уровня библиотеки BLAS показывают высокую эффективность при параллельной реализации математических функций на одном и шестнадцати ядрах процессора RoboDeus. Другим значимым численным результатом являются результаты теста HPL при работе с плотными матрицами (таблица 3).

Таким образом, полученные результаты на подпрограммах высокопроизводительной обработки на процессоре RoboDeus показывают высокую эффективность аппаратной реализации микроархитектуры ELcore-ядер и работы подсистемы памяти.

ЛИТЕРАТУРА

- [1] URL: <https://computing.llnl.gov/tutorials/mpi/> (дата обращения: 19.07.2021).
- [2] URL: <https://www.multicore.ru> (дата обращения: 19.07.2021)
- [3] URL: <https://www.khronos.org/opencl/> (дата обращения: 19.07.2021)
- [4] URL: <https://www.kernel.org/doc/html/v4.16/driver-api/dma-buf.html> (дата обращения: 19.07.2021)
- [5] Янакова Е.С., Мачарадзе Г.Т., Костулин Н.В., Тюрин А.А. Решение больших СЛАУ на гетерогенных СнК серии «Мультикор» // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС). 2020. Выпуск 1. С. 70-75. doi:10.31114/2078-7707-2020-1-70-75
- [6] URL: https://www.lrz.de/services/compute/courses/x_lecturenotes/Parallel_Programming_Languages_Workshop/MPI.pdf (дата обращения: 19.07.2021)
- [7] URL: <https://www.netlib.org/blas/> (дата обращения: 19.07.2021)
- [8] URL: <https://www.netlib.org/benchmark/hpl> (дата обращения: 19.07.2021).
- [9] URL: <https://www.top500.org> (дата обращения: 19.07.2021).

MPI-based Software Model for Parallel Computing in Heterogeneous Clusters

G.T. Macharadze

JSC R&D Center «ELVEES», Moscow, gmacharadze@elvees.com

Abstract — The construction of clusters based on heterogeneous computing systems requires the use of special software tools that ensure the effective execution of application software. The paper considers a software model based on MPI, OpenCL, the Linux OS driver, as well as mpi-run service processes. The service process within the computing node generates MPI computing processes using OpenCL. Interaction between processes is carried out by sending messages: processes within one computing node – through shared memory, processes of different computing nodes – through the network socket mechanism. The experimental study of the model was carried out on the Russian heterogeneous SoC RoboDeus using sixteen high-performance ELcore-50 cores developed by JSC ELVEES research center. The obtained results of the BLAS functions, as well as the HPL test for solving the problem with dense matrices in the double-precision floating-point format, confirmed the effectiveness of the proposed MPI model.

Keywords — cluster computing systems, parallel computing, heterogeneous computing systems, problem-oriented cores, processor 1892VM248, RoboDeus, MPI, OpenCL, BLAS, HPL.

REFERENCES

- [1] MPI. <https://computing.llnl.gov/tutorials/mpi/> (access data: 19.07.2021).
- [2] URL: <https://www.multicore.ru> (access data: 19.07.2021)
- [3] URL: <https://www.khronos.org/opencl/> (access data: 19.07.2021)
- [4] URL: <https://www.kernel.org/doc/html/v4.16/driver-api/dma-buf.html> (access data: 19.07.2021)
- [5] Yanakova E.S., Macharadze G.T., Kostulin N.V., Tiurin A.A. Solving large SLE on heterogeneous SoC of «Multicore» series // Problems of Perspective Micro- and Nanoelectronic Systems Development - 2020. Issue 1. P. 70-75. doi:10.31114/2078-7707-2020-1-70-75
- [6] URL: <https://www.netlib.org/blas/> (access data 19.07.2021)
- [7] URL: <https://www.netlib.org/benchmark/hpl> (access data: 19.07.2021).
- [8] URL: https://www.lrz.de/services/compute/courses/x_lecturenotes/Parallel_Programming_Languages_Workshop/MPI.pdf (access data: 19.07.2021)
- [9] URL: <https://www.top500.org> (access data: 19.07.2021).