УДК 004.2/.7 EDN: XQZRQE

АРХИТЕКТУРА ВЫЧИСЛИТЕЛЬНОГО КОМПЛЕКСА «ТЕРАГРАФ»

А.Ю. Попов alexpopov@bmstu.ru
С.В. Ибрагимов sibragimov@bmstu.ru
Е.Н. Дубровин dubrovinen@bmstu.ru
М.П. Калитвенцев
М.А. Гейне
Ц. Ли
Г.А. Парамазян
Т.М. Курохтин
Д.А. Залимханов

МГТУ им. Н.Э. Баумана, Москва, Российская Федерация

Аннотация

Применение графовых моделей является одним из наиболее перспективных направлений развития систем аналитики данных и искусственного интеллекта. Исследования в этой области затрагивают не только классические графы, но и такие сложные виды, как гипер- и метаграфы. Масштаб и сложность возникающих задач обработки графовых моделей делает актуальной разработку специализированных аппаратно-программных средств, повышающих эффективность существующих вычислительных комплексов для такого рода нагрузки. Рассмотрены архитектура и особенности функционирования вычислительного комплекса и облачной платформы «Тераграф», предназначенной для обработки графов большой размерности. Основным принципом, заложенным в архитектуру вычислительного комплекса, является применение многоуровневой ассоциативной подсистемы памяти, что существенно повлияло на методологию разработки и особенности функционирования программ. Приведено систематическое описание структуры аппаратного и программного обеспечения вычислительного комплекса. Представлены основные технические решения, которые позволили реализовать ассоциативную память большого размера на основе адресной памяти DDR4 DRAM. Ассоциативно-адресная трансляция для доступа к такому накопителю реализована на основе блока обработки трасс

Ключевые слова

Вычислительная система, граф, множество, набор команд дискретной математики, ассоциативная обработка, ускоритель вычислений, суперкомпьютер

микропроцессора «Леонард Эйлер». Представлено дальнейшее усовершенствование технических средств подсистемы сетевого взаимодействия, основанной на двунаправленной кольцевой топологии и высокоскоростных линиях 100Gb Ethernet

Поступила 27.05.2024
Принята 26.06.2024

Введение. Графовые модели хорошо подходят для наиболее полного представления систем и явлений, обладающих многообразием внутренних связей. Графы позволяют сохранять и анализировать не только атрибутную часть информации об объектах (вершин), но и контекстную — на основе ребер. Это особенно важно для развития таких областей применения вычислительных технологий, как искусственный интеллект, биология, медицина, социология, проектирование микросхем и др. [1]. Тем не менее актуальность и то внимание, которое уделяется исследованию графов и алгоритмам их анализа, пока не повлекли за собой каких-либо изменений в наборах команд микропроцессоров или составе ускорителей вычислений.

Архитектура и состав вычислительных систем ориентированы в первую очередь на универсальную вычислительную нагрузку, т. е. на арифметико-логическую обработку чисел или векторов. Ввиду наличия программных библиотек и возможностей современных компиляторов весь многообразный математический аппарат сводится к последовательности примитивных арифметических операций, например сложения. Однако аппарат дискретной математики предполагает скорее ассоциативный способ доступа к информации, чем адресный. Это ярко проявляется в системах управления базами данных, абстрагирующих потребителя этих технологий от особенностей хранения данных в памяти, или при использовании структур данных в программировании. Ассоциативность (или инцидентность) является неотъемлемым свойством вершин и ребер графа [2]. В связи с этим устройство для эффективного решения задач на графах должно реализовывать ассоциативный способ доступа. В этом случае парадокс заключается в том, что существующие средства вычислительной техники используют принцип адресности, в то время как на его основе приходится представлять ассоциативные модели.

Еще одной характерной особенностью алгоритмов обработки графов является существенное превышение времени доступа к данным над временем арифметико-логической обработки [3–5]. В частности, запрос данных микропроцессором и их пересылка занимает много времени: латентность доступа к данным, отсутствующим в кэш-памяти, часто занимает более 1000 тактов [6]. В результате только около 10 % времени работы

алгоритмов обработки графов занимает непосредственная арифметико-логическая обработка данных.

Еще одна особенность современной вычислительный техники — пакетный и крупноблочный режим обмена данными. Современные шинные интерфейсы и конвейеры микропроцессоров спроектированы в расчете на обработку информации, расположенную в памяти близко друг к другу, либо поступающей большими блоками [6]. Вследствие этого сглаживается разрыв в производительности процессора, памяти и других устройств, достигается большая пропускная способность каналов памяти, шин ввода-вывода, периферийных устройств, накопителей и т. д. В случае обработки графов эта особенность является недостатком. Дело в том, что работа с множествами вершин и ребер не является крупноблочной и групповой, а элементы извлекаются индивидуально [7]. Таким образом, наблюдается парадокс, при котором архитектура современных вычислительных систем и организация вычислительного процесса неадекватна принципам обработки графов.

Графические процессоры (GPU) также используются для обработки графов и показывают высокую эффективность, в особенности на тех задачах, где графовые модели обладают регулярностью [8]. Благодаря архитектуре SIMT (Single Instruction and Multiple Threads) GPU позволяют запускать множество потоков обработки и обладают более высокой пропускной способностью подсистемы памяти по сравнению с универсальными микропроцессорами, что позволяет им выбирать большие блоки данных и запускать множество параллельных нитей вычислений. Однако при нерегулярной структуре графов шаблоны доступа к памяти также нерегулярны, вследствие чего производительность существенно снижается. На эффективность GPU при обработке графов негативно влияют короткие циклы алгоритмов обработки и частые ветвления, которые не позволяют планировать длинную последовательность вычислений [9]. Таким образом, GPU также не соответствуют характеру графовых алгоритмов, что связано с их первоначальным назначением: реализацией стадий графического конвейера.

Исследование различных подходов к решению указанных проблем показало, что в настоящее время актуальной является разработка программных и аппаратных средств для реализации алгоритмов обработки динамически изменяемых графов. Один из подходов, которые применяют для этой цели, основан на оптимизации алгоритмов на графах [10]. Применение структуры данных, обеспечивающих низкую вычислительную сложность алгоритмов, является важным элементом такого подхода [11]. Однако

их разработка и внедрение не решают ключевых проблем повышения эффективности микропроцессоров и памяти при обработке графов.

С позиции применения аппаратного обеспечения можно выделить такие направления, как перенос обрабатывающих устройств в память [12, 13], ускорение обмена данных между микропроцессором и ускорителем вычислений, повышение эффективности коммутации ядер для множественного доступа к памяти [7], реализация параллельных вычислительных структур и распределение вершин графа в них [7, 14], реализация большого числа многопоточных ядер одновременно с увеличением числа независимых каналов памяти [15–17].

Большинство из перечисленных подходов направлены на повышение эффективности существующих универсальных ЭВМ. Вместе с тем следует расширить поиск принципов организации систем обработки графов в более широких пределах системной абстракции, начиная с особенностей реализации исполнительных устройств, оптимизации наборов машинных инструкций, принципов организации памяти, особенностей взаимодействия устройств в системе, конвейеризации, масштабирования, виртуализации и т. д. Очевидно, что проблем намного больше, чем используемых способов реализации алгоритмов на графах. В настоящей работе представлена уточненная концепция вычислительной системы [17–20], оперирующей графами большой размерности.

Структурно-иерархический подход к разработке вычислительного комплекса «Тераграф». Одной из существенных проблем при создании вычислительного комплекса, выполняющего хранение и обработку графов, является применение системного подхода к его проектированию, т. е. определение наиболее рационального варианта разделения программного и аппаратного обеспечения по уровням системной иерархии. Это означает, что на ранних этапах разработки должны быть установлены число и функциональное назначение элементов системы для каждого уровня системной иерархии. Выявлены восемь уровней (рис. 1, *а*).

1. Память ключей и значений. Множества вершин и ребер организованы в виде ключей и значений, для их хранения используются доступные в настоящий момент полупроводниковые запоминающие устройства (ЗУ) на основе DDR4 DRAM. Поскольку к одному носителю должны иметь доступ несколько устройств, необходимо определить способы разделения единого адресного пространства ЗУ между ними и способы упорядочивания запросов. Доступ к памяти осуществляется в пакетном режиме, что предполагает формирование пакетов ключей и значений. Кроме ключей и значений, для представления вершин и ребер графов в памяти также размещается

индексная информация, которая позволяет организовать эффективные алгоритмы доступа и обработки.

2. Параллельный доступ к памяти. Существенная проблема многопоточной обработки — полоса пропускания памяти. Для ее увеличения память подразделяется на независимые ЗУ, а доступ осуществляется с использованием аппаратных очередей и устройства арбитража.

3. Ассоциативно-адресная трансляция. Извлечение информации из множеств предполагает ассоциативный способ доступа, а физические носители информации используют адресный способ. Проблема преобразования ключа в адрес его хранения является сложной технической задачей, которая может быть решена различными способами. В настоящий момент выбор наиболее рационального варианта основан на применении B+-деревьев в качестве индексных структур. Вместе с тем поиск наилучших вариантов продолжается.

4. Команды дискретной математики. На этом уровне необходимо определить состав аппаратных средств и эффективно реализовать с их помощью набор примитивных операций над множествами, лежащих в основе набора команд дискретной математики DISC (Discrete Mathematics Instuction Set). Это предполагает создание специализированного микропроцессорного устройства и микрокода устройств управления.

5. Гетерогенное ядро. Обработка множеств или их совокупности основана не только на командах дискретной математики, но и на арифметико-логической обработке. Применение универсальных микропроцессоров также необходимо для создания привычной программной модели. Поэтому универсальный микропроцессор с архитектурой riscv64 (Computing Processing Element, CPE) и микропроцессор lnh64 с набором команд дискретной математики DISC (Structure Processing Element, SPE) объединены в единое гетерогенное ядро (комплекс из двух различных микропроцессорных ядер).

6. Сетевое взаимодействие. Ресурсы и производительность гетерогенных ядер ограничены, поэтому требуется реализовать средства и механизмы их масштабирования, а также предусмотреть возможность взаимодействия гетерогенных ядер между собой с использованием высокоскоростной сети передачи данных, обладающей высокой пропускной способностью и низкой латентностью. Это позволит использовать для решения прикладных задач большое число параллельно работающих гетерогенных ядер одновременно (горизонтальное масштабирование) или организовать последовательную обработку потока данных на нескольких гетерогенных ядрах (конвейеризация).

7. Вычислительный узел. Множество гетерогенных ядер должно быть объединено в вычислительный узел, в рамках которого осуществляется несколько действий (инициализация, запуск и управление выполнением задач, обеспечение доступа к аппаратным и программным ресурсам, отладка программ, сетевое взаимодействие с удаленным пользователем и т. д.).

8. Вычислительный комплекс. Несколько узлов могут быть соединены вместе, вследствие чего достигается большее горизонтальное масштабирование, становится возможным увеличить число гетерогенных ядер и объем доступной памяти для хранения множеств. На уровне вычислительного комплекса применяется облачный подход IaaS (Infrastructure as a Service), решающий задачи управления ресурсами, что в полной мере поддерживается разработанными системными библиотеками и аппаратными ресурсами.

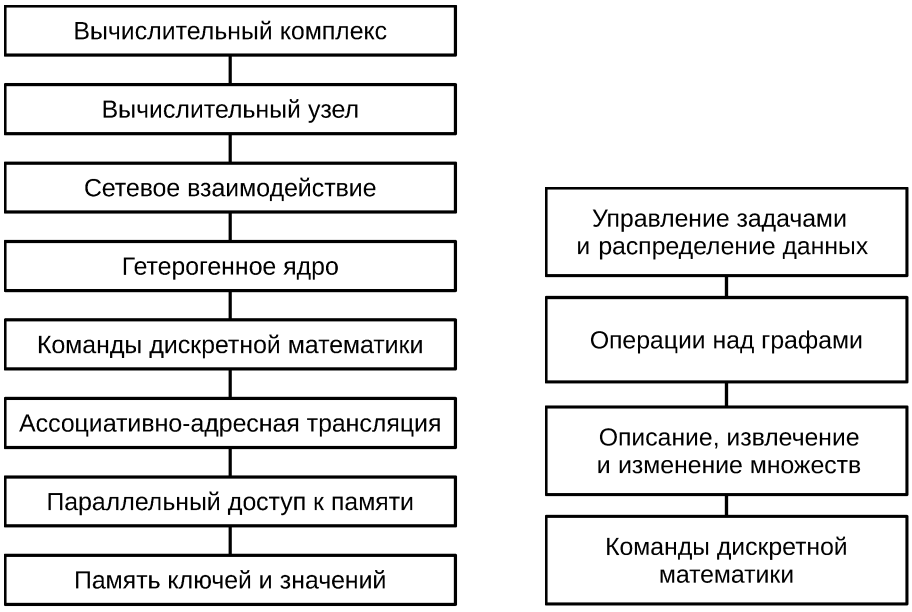


Рис. 1. Иерархическая структура вычислительного комплекса «Тераграф»: *а* — уровни системной иерархии абстракции; *б* — уровни программных средств

Следует отметить положительный технический эффект, который достигается в результате предложенной структуризации (рис. 1, *а*): формируется набор требований к набору функций каждого уровня, вследствие чего становятся возможными дальнейшая модернизация структуры и реализация каждого уровня в отдельности. Достигается возможность планирования НИОКР.

Возможность применения программных библиотек и языков программирования C/C++ поддерживается для гетерогенного ядра и вычислительного узла. Для них предложены четыре уровня функциональной абстракции программных средств (рис. 1, *б*).

1. Команды дискретной математики — базовый набор операций DISC над множествами, состоящий из 21 машинной инструкции высокого уровня. Команды используются на уровне гетерогенного ядра.

2. Описание, извлечение и изменение множеств — языковые конструкции расширяют возможности обработки множеств и позволяют построить более сложную и итерационную обработку. Например, возможно создавать композитные ключи и значения, выполнять последовательный обход элементов множества, т. е. итерировать по ключам.

3. Операции над графами — определяются программно вызываемые функции для описания и обработки графов.

4. Управление задачами и распределение данных — для обеспечения масштабируемости и конвейеризации необходимо использовать программные средства для контроля и управления обрабатывающими ресурсами комплекса.

Указанное разделение программно доступных функций и объединение их в библиотеки с иерархической организацией решает несколько важных задач: обеспечивает относительную независимость их дальнейшей разработки и модернизации; позволяет аналогичным образом представлять прикладные задачи и формировать абстрактные модели уровней системной иерархии. Одним из результатов структурного анализа аппаратной и программной составляющих вычислительного комплекса должен стать набор описаний обобщенных моделей системного и функционального типов, составляющих открытый стандарт ассоциативных вычислений, подобно стандарту OpenCL. В работе предпринимается попытка не только представить архитектуру и принципы функционирования вычислительной системы для обработки множеств и графов на основе команд дискретной математики, но и сформировать концепцию для стандартизации уровней абстракции такой гетерогенной ассоциативной системы.

Далее будут кратко изложены принятые технические решения для представленных на рис. 1, *б* уровней абстракции.

Память ключей и значений. Для повышения эффективности команд дискретной математики необходимо выбрать вариант представления множеств, который будет обеспечивать отображение пространства ключей в адресное пространство памяти с произвольным доступом. В существующей версии системы используется внутреннее представление множеств

в виде леса B+-деревьев. В результате становится возможной параллельная обработка нескольких вершин дерева и нескольких деревьев как на их верхних (индексных) уровнях, так и на нижнем информационном уровне, хранящем непосредственно ключи и значения. Любая операция DISC начинается с поиска информации в индексной части B+-дерева и заканчивается обработкой вершин нижнего уровня. Если необходимо перестроение структуры B+-дерева вследствие добавления или вставки ключей и значений, то это может потребовать изменения информационной и индексных частей.

Каждому процессору с набором команд дискретной математики выделяется некоторый фиксированный объем в памяти ключей и значений (Local Structure Memory), размер которой суммируется из регионов для индексной и информационной частей. В текущей версии индексный регион занимает 512 МБ, а информационный — 2ГБ, что обеспечивает возможность хранения 117 млн 64-битных ключей и значений.

Параллельный доступ к памяти. В текущей версии микропроцессора «Леонард Эйлер» до шести микропроцессорных ядер объединены в группы, причем каждая группа подключена к отдельной шине памяти ключей и значений (Local Structure Memory типа DDR4 DRAM) (рис. 2). Проведено нагрузочное тестирование, которое продемонстрировало достаточный запас пропускной способности каналов памяти при таком числе микропроцессорных ядер. Результаты приведены в [17].

Однако может возникнуть обратная ситуация: неполное использование каналов памяти. Поэтому важной задачей является увеличение числа потоков, исполняемых в микропроцессорных ядрах riscv64 CPE, lnh64 SPE. Это предполагается достичь применением кооперативной или вытесняющей многозадачности (сопрограмм и потоков) в CPE, а также более глубокой конвейерности и многонитевости в ядрах SPE.

Ассоциативно-адресная трансляция. Ассоциативно-адресная трансляция осуществляется в результате аппаратной реализации механизмов обработки B+-деревьев. Последовательность вершин дерева, составляющая путь для аппаратного поиска ключа от корня до листа, представляет собой трассу, которая извлекается из индексного региона и буферизируется во внутренней памяти SPE. После обработки трассы следует загрузка вершины нижнего уровня из памяти ключей и значений в микропроцессорное ядро SPE. Для этого требуется транслировать номер вершины в двоичный код выборки, который служит для обращения во внутреннюю память структур.

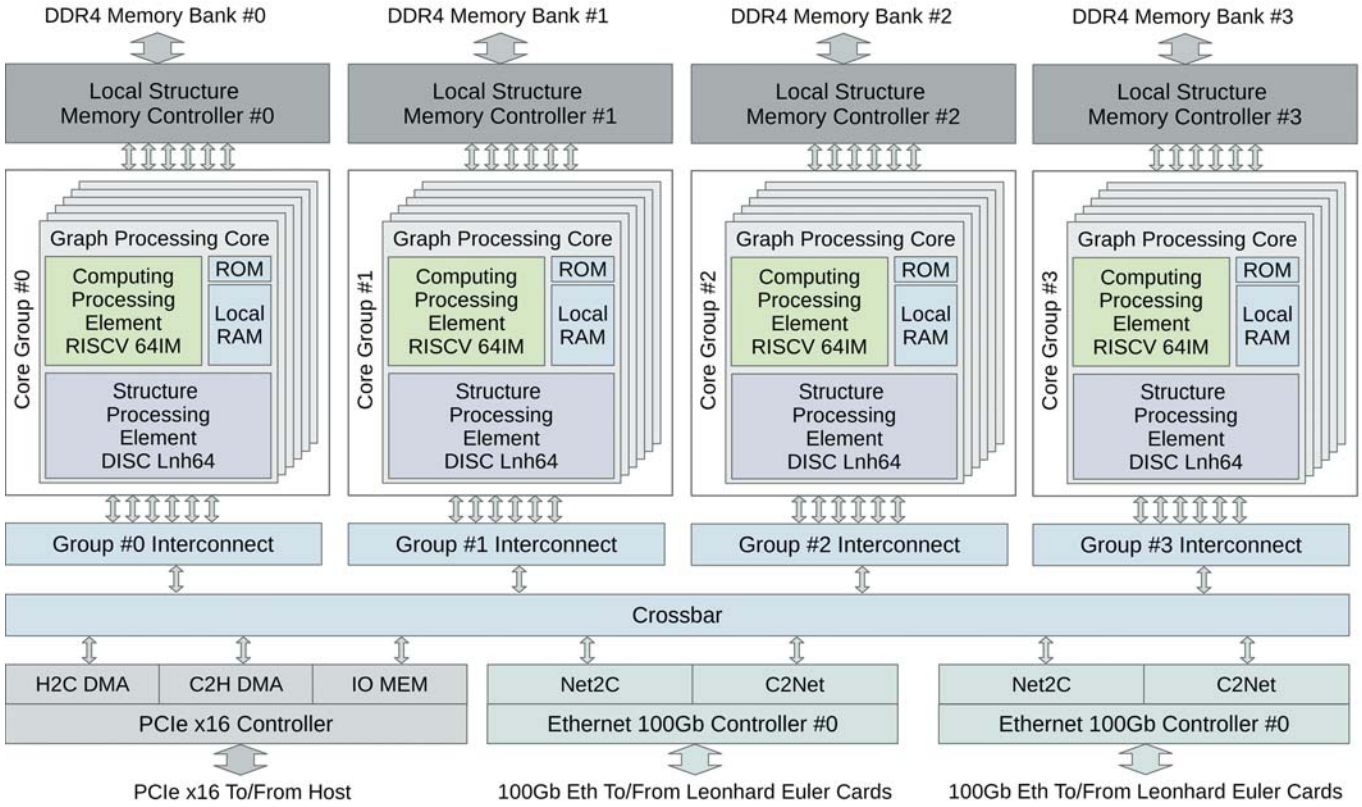


Рис. 2. Структура микропроцессора «Леонард Эйлер»

Механизм ассоциативно-адресной трансляции в предлагаемой системе реализован аппаратно и ниже уровня набора команд, что является существенным достижением. Это решение открывает множество возможностей по созданию ассоциативных вычислительных систем.

Команды дискретной математики. Формирование набора команд — ключевой вопрос, решение которого в большой степени определяет дальнейшие технические характеристики вычислительных систем. Набор команд дискретной математики DISC, состоящий из таких операций, как поиск (SRCH), вставка (INS), удаление (DEL), поиск ближайшего меньшего или большего (NSM,NGR), минимума или максимума (MIN, MAX), объединение множеств (OR), пересечение множеств (AND), разность множеств (NOT), команды формирования подмножеств (срезы по условиям LS, LSEQ, GR, GREQ, GRLS) и др., приведен в [17, 18]. Всего предложено использовать 21 машинную инструкцию высокого уровня [17, 19]. Операндами для указанных команд являются 64-битные ключи и номера множеств (структур), в которых они хранятся. Набор команд DISC сформирован из основных теоретико-множественных операций и кванторов, вследствие чего он обеспечивает покрытие операций над множествами в алгоритмах дискретной оптимизации и, в частности, в алгоритмах на графах.

Применение микропроцессора «Леонард Эйлер» для решения практических задач [3, 4, 17–20] выявило, что для эффективного использования аппаратного обеспечения вычислительной системы «Терапраф» необходимо разработать лингвистические средства, которые в достаточной мере выразительны для внедрения набора команд DISC в широко применяемые языки программирования (C/C++, *Python* и др.). Очевидно, что набор команд DISC имеет сходство с операциями, выполняемыми в реляционных СУБД, для которых в языке SQL имеются аналогичные операции поиска, вставки, удаления, перехода к следующим и предыдущим записям и т. д. В связи с этим разработана технология кодогенерации, использующая синтаксис языка SQL для описания, извлечения и изменения множеств. В результате использования средств кодогенерации программист может применять конструкции языка SQL — CREATE TABLE и SELECT, которые преобразуются в код на языке C++ и могут быть скомпилированы в бинарный исполняемый файл.

Гетерогенное ядро обработки графов. Как отмечено выше, обработка графов в системе «Терапраф» выполняется в гетерогенных ядрах, состоящих из микропроцессоров двух типов (CPE и SPE, рис. 3). При этом CPE является универсальным ядром riscv64 с арифметическим набором команд, в то время как lnh64 SPE реализует набор команд DISC.

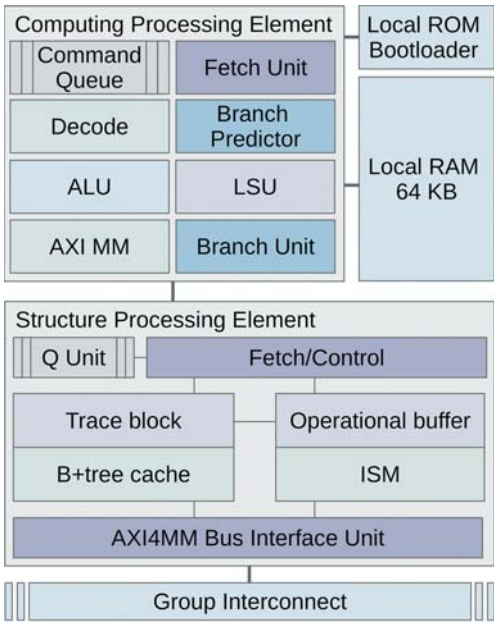


Рис. 3. Микроархитектура ядра обработки графов

Каждый вычислительный элемент CPE состоит из традиционного набора исполнительных устройств, характерных для конвейерного микропроцессора с упорядоченным исполнением: очереди команд (Command Queue), блока выборки (Fetch Unit), блока декодирования команд (Decode), модуля предсказания переходов (Branch Predictor), арифметико-логического устройства (ALU), устройства доступа в память (Load/Store Unit), интерфейса шин (шины AXIMM и шины доступа к ускорителю MM), блока ветвлений (Branch Unit). Вычислительный элемент связан шиной памяти с ПЗУ, в который записан начальный загрузчик. Для размещения программ и данных каждый CPE имеет оперативную память размером 64 КБ.

Процессор обработки структур SPE с микроархитектурой lnh64 представляет собой управляемое микропроцессорное устройство DISC, предназначенное для хранения и обработки больших множеств дискретной информации. Он состоит из модуля очередей (QUnit), блока выборки и контроля (Fetch/Control), памяти структур (Internal Structure Memory, ISM), операционного буфера (Operational buffer), блока обработки трасс (Trace Block), ассоциативного блока хранения трасс (B+tree cache) и модулей интерфейса шин AXI4MM и MM.

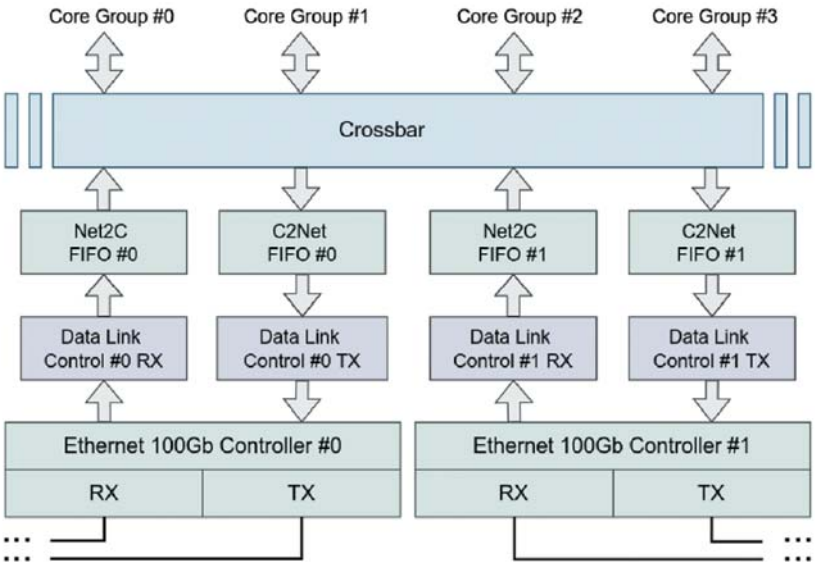
Под управлением поступающих от CPE команд SPE выполняет сохранение ключей и значений в многоуровневой подсистеме памяти, выполняет поиск, изменение и выдачу информации другим устройствам системы.

Сетевое взаимодействие. Задача, выполняемая средствами уровня сетевого взаимодействия, заключается в обеспечении обмена данными гетерогенными ядрами без участия внешнего посредника (хост-подсистемы). Каждое ядро должно иметь возможность записи в адресное пространство любого другого ядра вне зависимости от того, насколько эти ядра разделены физически (находятся в различных группах, на разных картах в пределах одного узла, в пределах двух различных узлов). Такой принцип позволяет средствам более высоких уровней (вычислительного узла, вычислительного комплекса) управлять вычислительными ресурсами ядер произвольным образом, в зависимости от выполняемой задачи. Становятся возможными применение конвейеризации и виртуализации ассоциативной системы, по аналогии с виртуализацией, применяемой по отношению к адресному пространству оперативной памяти в микропроцессорных системах.

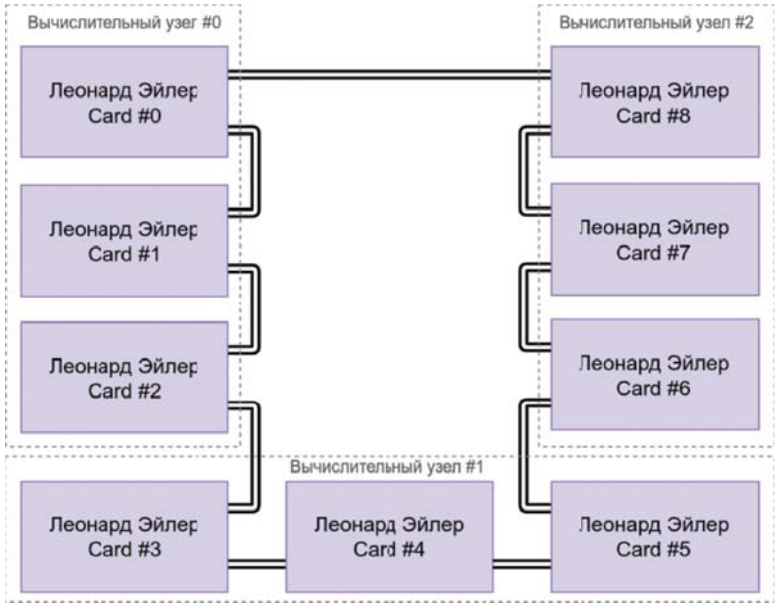
Сеть связи ядер можно условно подразделить на три уровня: 1) уровень отдельных ядер внутри группы; 2) уровень групп ядер внутри карты; 3) уровень карт в рамках комплекса. На уровне 1 происходит обмен данными между ядрами одной и той же группы за счет использования общей памяти. В настоящий момент в разработке находится техническое решение, которое позволит осуществлять этот процесс с использованием коммутации сетевых интерфейсов ядер. Уровень 2 групп ядер объединяет потоки данных от всех групп GPC в рамках одной вычислительной карты. Осуществляется как обмен данными между соседними группами, так и пересылка данных на выходные интерфейсы для отправки группам, находящимся в пределах других карт. Структурная схема сети на этом уровне показана на рис. 4.

Для связи с прочими картами используют контроллеры 100Gb Ethernet. В непосредственной близости от них находятся блоки контроля доставки, работающие по специальному протоколу канального уровня. Функциональность блоков контроля доставки — обеспечение передачи данных через 100Gb Ethernet к другой карте без потерь. Принимаемые и передаваемые данные буферизуются в очередях FIFO, которые в свою очередь с использованием потоковых интерфейсов подключены к блоку коммутации. Последний осуществляет перенаправление фрагментов данных к определенной группе ядер либо к внешним подключениям посредством разрешения адресов. Каждое ядро имеет собственный адрес в едином для всех ядер адресном пространстве.

В рамках комплекса уровень 3 представлен набором активных в настоящий момент вычислительных карт, связанных через высокоскоростные линии 100Gb Ethernet и образующих топологию «двунаправленное кольцо» (рис. 4, *б*). Предпочтение такой топологии отдано ввиду относительно



а



б

Рис. 4. Сеть вычислительного комплекса «Терапраф»:
а — структура сети на уровне групп ядер; *б* — общий вид кольцевой топологии

небольшого числа узлов сети на этом уровне, а также ввиду отсутствия необходимости вводить центральный узел, контролирующий коммутацию карт. Вследствие наличия двух контроллеров 100Gb Ethernet для каждой карты данные могут передаваться в двух направлениях. При условии, что

направление передачи определяется положением передающего и принимающего узлов в топологии, это позволяет уменьшить среднее время доставки по сравнению с аналогичной однонаправленной топологией и снижает нагрузку на линии передачи. Задача определения направления и выбора интерфейса для передачи ложится на коммутатор уровня групп ядер (Crossbar), так как в его распоряжении находятся данные сопоставления адресов ядер-получателей и физических интерфейсов передачи.

Вычислительный узел. Этот узел представляет собой функционально законченную вычислительную систему и может использоваться самостоятельно. Вместе с тем для обеспечения большего масштабирования в проекте реализуется вычислительная система, состоящая из трех однотипных узлов. Рассмотрим структуру одного узла, состоящего из хост-подсистемы, подсистемы хранения графов, подсистемы коммутации узлов и подсистемы обработки графов (рис. 5).

Хост-подсистема выполняет функции управления запуском вычислительных задач, поддержки сетевых подключений, обработки и балансировки нагрузки. В хост-подсистему входят несколько многоядерных ЦПУ и оперативная память, на которых функционирует программный код управления. Эта подсистема организует взаимодействие подсистемы обработки графов с подсистемой хранения и обеспечивает сетевое взаимодействие узла с внешними ресурсами. К функциям хост-подсистемы относятся:

– на стадии инициализации: настройка сетевой подсистемы, подсистем хранения и обработки графов;

– на стадии создания/изменения графов в локальной памяти DISC: реализация очередей запросов на вставку/изменения, балансировка запросов к DISC-системам, выделение и освобождение структур, контроль выполнения операций изменения;

– на стадии запуска алгоритмов оптимизации: буферизация запросов оптимизации, инициализация процедур обработки, их запуск и контроль исполнения;

– на стадии визуализации графов: буферизация запросов на визуализацию, настройка процедур формирования представлений графов для пользовательских процессов, запуск формирования представлений и контроль результатов, буферизация и передача представлений или изменений в представлениях пользовательским процессам.

Перечисленные функции реализованы в программном ядре хост-подсистемы (Host Software Kernel) — программном обеспечении, взаимодействующим с подсистемой обработки графов через шину PCIe и работающем

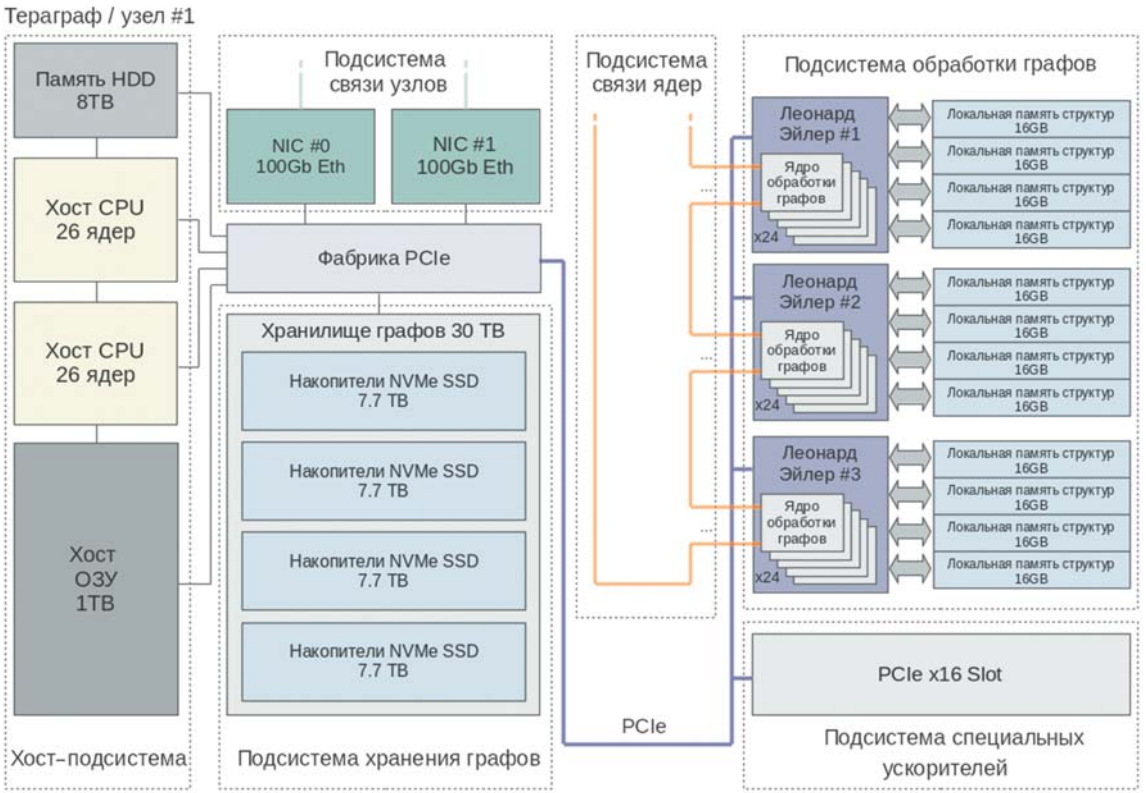


Рис. 5. Структура гетерогенного узла обработки графов

в составе драйвера ускорительной карты. Со стороны подсистемы обработки графов реализовано до 24 гетерогенных DISC-ядер. Каждое ядро содержит обрабатывающие элементы SPE и CPE, на которых функционирует так называемое программное ядро ускорителя (Software Kernel) — специальная программа, передаваемая в бинарном виде в локальное ОЗУ CPE в процессе инициализации и взаимодействующая с SPE.

В подсистему хранения графов одного узла входит основное хранилище объемом 30 ТБ, состоящее из твердотельных накопителей. Хранилище организовано в виде ассоциативного key-value-хранилища с возможностью программного доступа через API.

Подсистема связи узлов представляет собой два сетевых модуля взаимодействия через интерфейсы 100Gb Ethernet, позволяющие организовать соединение каждого вычислительного узла с другим узлом комплекса. Шина PCIe обеспечивает высокопроизводительное взаимодействие хост-подсистемы с 24-ядерными процессорами «Леонард Эйлер» и с подсистемой хранения графов.

Подсистема обработки графов состоит из трех или четырех карт (в зависимости от версии) многоядерных процессоров «Леонард Эйлер», в каждом из них содержится по три или четыре группы гетерогенных ядер (Core Groups, CG). Таким образом, многоядерный процессор «Леонард Эйлер» версии 4 может содержать до 24 гетерогенных обрабатывающих DISC-ядра, а весь комплекс «Тераграф» может содержать до 288 гетерогенных обрабатывающих DISC-ядра.

Группа гетерогенных ядер содержит разделяемые между гетерогенными ядрами контроллеры памяти, которые обеспечивают взаимодействие между DISC-ядрами и память ключей и значений (локальная память структур, LSM) типа DDR4 DRAM (см. рис. 5).

Вычислительный комплекс «Тераграф» и облачная платформа «Тераграф Cloud». Объединение узлов в единый комплекс обеспечивается с помощью сетей нескольких типов: горизонтальные связи на уровне хост-подсистем обеспечиваются на основе прямых соединений подсистемы связи узлов (см. рис. 5); на уровне гетерогенных ядер обмен данными обеспечивается с помощью контроллеров сетевого взаимодействия. Кроме того, организована независимая от сетей передачи данных управляющая сеть, которая служит для управления и мониторинга.

Конструктив комплекса состоит из восьми элементов (рис. 6): 1) трех однотипных узлов обработки графов; 2) консоли управления комплексом; 3) сети связи гетерогенных ядер (выделена оранжевым цветом); 4) сети связи узлов обработки графов (выделена зеленым цветом); 5) источников

бесперебойного питания; 6) подсистемы обработки графов, содержащей три однотипные карты микропроцессора «Леонард Эйлер»; 7) подсистемы хранения графов, реализованной на основе твердотельных накопителей; 8) хост-подсистемы, основанной на микропроцессорах общего назначения, оперативной памяти и накопителей на жестких магнитных дисках.

Рис. 6. Вычислительный комплекс «Тераграф»

В зависимости от решаемой задачи имеющиеся аппаратные ресурсы комплекса и хранилища данных могут быть программно перераспределены между задачами обработки множеств и графов, вычислительными задачами общего назначения, задачами машинного обучения, а также задачами обработки на специализированных ускорителях вычислений на базе ПЛИС FPGA. Комплекс сопровождается программными библиотеками системного и прикладного уровней. Для доступа к аппаратным ресурсам используется контейнеризация прикладных задач. В результате обеспечивается возможность разделения ресурсов между пользователями и их запущенными задачами, упрощается создание прикладных программ и достигается высокий уровень быстродействия комплекса.

Облачная платформа «Тераграф Cloud» (TeraCloud) будет объединять в единый взаимосвязанный кластер три рабочих узла вычислительного кластера: 1) «Тераграф» с микропроцессорами «Леонард Эйлер»; 2) вычислительный сервер Nvidia DGX2 с графическими ускорителями Nvidia Tesla V100; 3) мастер-узел для функционирования единого управляющего портала. Потребителям облачной платформы «Тераграф Cloud» предоставляются следующие виртуальные ресурсы:

– личный кабинет пользователя с отображением состояния доступных ресурсов;

– персональное файловое хранилище для долговременного размещения необходимых данных;

– база Docker-образов с предустановленным программным обеспечением и зависимостями;

– управляющая панель для запуска и останова контейнеров;

– гетерогенные ядра обработки графов GPC, подключаемые к запущенным контейнерам на время проведения расчетов;

– графические ускорительные ядра GPU Nvidia Tesla V100 для проведения расчетов, связанных с работой алгоритмов машинного обучения и искусственного интеллекта;

– выделенное ассоциативное хранилище Tarantool или аналогичное для реализации ассоциативного хранилища графов большой размерности;

– предварительно настроенный образ с Jupyter Notebook для разработки программ на языке *Python* и C/C++;

– предварительно настроенный образ с расширяемой средой разработки MS VSCode, поддерживающей большинство существующих языков разработки (C, C++, *Python*, *JavaScript*, *Java*, *Rust*, *Go* и др.).

Облачная платформа «Тераграф Cloud» позволит предоставить одновременный доступ к ресурсам комплекса «Тераграф», ускорительным картам «Леонард Эйлер» и Nvidia Tesla большому числу пользователей, прежде всего, студентам и преподавателям МГТУ им. Н.Э. Баумана.

Заключение. Отличительной особенностью представленного вычислительного комплекса «Тераграф» является глубокая аппаратная поддержка ассоциативной обработки. Это выражается в том, что на всех уровнях программного обеспечения комплекса хранение данных осуществляется в ассоциативной памяти большой размерности, а ввиду наличия микропроцессорного ядра с набором команд дискретной математики DISC фактически решена проблема ассоциативно адресной трансляции.

На основе результатов работ, в которых продемонстрированы основополагающие принципы работы микропроцессора «Леонард Эйлер» с набором команд дискретной математики и результаты оценки производительности [17, 18], проведена дальнейшая систематизация принципов организации системы. Представлен структурно-иерархический подход к формированию архитектуры комплекса и его программного обеспечения, основанный на ассоциативных принципах обработки. Показано, что ассоциативно-адресная трансляция может быть успешно реализована аппаратно на более низком уровне по отношению к машинным инструкциям микропроцессорного DISC-ядра. В связи с этим становится возможным создание и дальнейшее независимое совершенствование механизмов отображения ассоциативного пространства ключей, доступного программным средствам, в физические адреса на уровне памяти ключей и значений.

Для каждого уровня структурной иерархии показаны его назначение, состав и функциональные особенности. Кратко рассмотрена концепция распределения и изоляции ресурсов для обеспечения многопользовательского доступа в рамках облачного подхода. В настоящее время завершается разработка облачной платформы «Терапраф Cloud» и набора системных и прикладных библиотек комплекса.

ЛИТЕРАТУРА

[1] Majeed A., Rauf I. Graph theory: a comprehensive survey about graph theory applications in computer science and social networks. *Inventions*, 2020, vol. 5, iss. 1, art. 10. DOI: https://doi.org/10.3390/inventions5010010

[2] Mondal B., De K. Overview applications of graph theory in real field. *IJSRCSEIT*, 2017, vol. 2, no. 5, pp. 751–759.

[3] Попов А.Ю. О реализации алгоритма Форда-Фалкерсона в вычислительной системе с многими потоками команд и одним потоком данных. *Наука и образование: научное издание*, 2014, № 9. EDN: TDPOMV

[4] Попов А.Ю. Исследование вариантов реализации алгоритмов Крускала и Прима в вычислительной системе с многими потоками команд и одним потоком данных. *Наука и образование: научное издание*, 2015, № 11. EDN: VDRHNX

[5] Ibrahim A.H., Abdelhalim M.B., Hussein H., et al. Analysis of x86 instruction set usage for Windows 7 applications. *2nd Int. Conf. on Computer Technology and Development*, 2010, pp. 511–516. DOI: https://doi.org/10.1109/ICCTD.2010.5645851

[6] Qian Z., Wei J., Xiang Y., et al. A performance evaluation of DRAM access for in-memory databases. *IEEE Access*, 2021, vol. 9, pp. 146454–146470.
DOI: https://doi.org/10.1109/ACCESS.2021.3123379

[7] Aananthakrishnan S., Ahmed N.K., Cave V., et al. PIUMA: programmable integrated unified memory architecture. *arXiv:2010.06277*.
DOI: https://doi.org/10.48550/arXiv.2010.06277

[8] Aasawat T.K., Reza T., Ripeanu M. How well do CPU, GPU and hybrid graph processing frameworks perform? *IEEE IPDPSW*, 2018, pp. 458–466.
DOI: https://doi.org/10.1109/IPDPSW.2018.00082

[9] Zou Y., Lin M. GridGAS: an I/O-efficient heterogeneous FPGA+CPU computing platform for very large-scale graph analytics. *FPT*, 2018, pp. 246–249.
DOI: https://doi.org/10.1109/FPT.2018.00045

[10] Lumsdaine A., Gregor D.P., Hendrickson B., et al. Challenges in parallel graph processing. *Parallel Process. Lett.*, 2007, vol. 17, no. 1, pp. 5–20.
DOI: https://doi.org/10.1142/S0129626407002843

[11] Cormen T.H., Leiserson C.E., Rivest R.L., et al. Introduction to algorithms. MIT Press, 2009.

[12] Ahn J., Hong S., Yoo S., et al. A scalable processing-in-memory accelerator for parallel graph processing. *ACM/IEEE 42nd ISCA*, 2015, pp. 105–117.
DOI: https://doi.org/10.1145/2749469.2750386

[13] Angizi S., Fan D. GraphiDe: a graph processing accelerator leveraging In-DRAM-computing. *GLSVLSI '19*, 2019, pp. 45–50.
DOI: https://doi.org/10.1145/3299874.3317984

[14] Dysart T., Kogge P., Deneroff M., et al. Highly scalable near memory processing with migrating threads on the emu system architecture. *IA3*, 2016, pp. 2–9.
DOI: https://doi.org/10.1109/IA3.2016.007

[15] Gui C., Zheng L., He B., et al. A survey on graph processing accelerators: challenges and opportunities. *J. Comput. Sci. Technol.*, 2019, vol. 34, no. 2, pp. 339–371.
DOI: https://doi.org/10.1007/s11390-019-1914-z

[16] Hennessy J., Patterson D. Domain specific architectures. In: *Computer architecture*. Elsevier, 2017, pp. 540–606.

[17] Popov A., Ibragimov S., Dubrovin E. Teragraph heterogeneous system for ultra-large graph processing. In: Voevodin V., Sobolev S., Yakobovskiy M., Shagaliev R. (eds). *Supercomputing. RuSCDays 2022. Lecture Notes in Computer Science*, vol. 13708. Cham, Springer, 2022, pp. 574–590. DOI: https://doi.org/10.1007/978-3-031-22941-1_42

[18] Popov A. An introduction to the MISD technology. *Proc. Annual Hawaii Int. Conf. on System Sciences*, 2017, pp. 1003–1012.

[19] Попов А.Ю. Принципы организации гетерогенной вычислительной системы с набором команд дискретной математики. *Информационные технологии*, 2020, т. 26, № 2, с. 67–79. DOI: https://doi.org/10.17587/it.26.67-79

[20] Попов А.Ю. Применение гетерогенной вычислительной системы с набором команд дискретной математики для решения задач на графах. *Информационные технологии*, 2019, т. 25, № 11, с. 682–690. DOI: https://doi.org/10.17587/it.25.682-690

Попов Алексей Юрьевич — д-р техн. наук, доцент кафедры «Компьютерные системы и сети» МГТУ им. Н.Э. Баумана (Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5, стр. 1).

Ибрагимов Станислав Вадимович — старший преподаватель кафедры «Компьютерные системы и сети» МГТУ им. Н.Э. Баумана (Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5, стр. 1).

Дубровин Егор Николаевич — ассистент кафедры «Компьютерные системы и сети» МГТУ им. Н.Э. Баумана (Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5, стр. 1).

Калитвенцев Максим Павлович — инженер НИЧ НУК ИУ МГТУ им. Н.Э. Баумана (Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5, стр. 1).

Гейне Михаил Антонович — инженер НИЧ НУК ИУ МГТУ им. Н.Э. Баумана (Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5, стр. 1).

Ли Цзяцзянь — инженер НИЧ НУК ИУ МГТУ им. Н.Э. Баумана (Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5, стр. 1).

Парамазян Гор Арменович — инженер НИЧ НУК ИУ МГТУ им. Н.Э. Баумана (Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5, стр. 1).

Курохтин Тимофей Михайлович — инженер НИЧ НУК ИУ МГТУ им. Н.Э. Баумана (Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5, стр. 1).

Залимханов Динислам Алиевич — инженер НИЧ НУК ИУ МГТУ им. Н.Э. Баумана (Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5, стр. 1).

Просьба ссылаться на эту статью следующим образом:

Попов А.Ю., Ибрагимов С.В., Дубровин Е.Н. и др. Архитектура вычислительного комплекса «Терапраф». *Вестник МГТУ им. Н.Э. Баумана. Сер. Приборостроение*, 2025, № 1 (150), с. 113–136. EDN: XQZRQE

TERAGRAPH COMPUTER ARCHITECTURE

A.Yu. Popov alexpopov@bmstu.ru
S.V. Ibragimov sibragimov@bmstu.ru
E.N. Dubrovin dubrovinen@bmstu.ru
M.P. Kalitventsev
M.A. Geine
J. Li
G.A. Paramazyan
T.M. Kurokhtin
D.A. Zalimkhanov

BMSTU, Moscow, Russian Federation

Abstract

The use of graph models is one of the most promising areas for the development of data analytics and artificial intelligence systems. Research in this field affects not only classical graphs, but also such complex types as hyper- and metagraphs. The scale and complexity of the emerging graph model processing tasks makes it urgent to develop specialized hardware and software tools that increase the efficiency of existing computing systems for this kind of workload. The article discusses the architecture and features of the *Teragraf* computing complex and cloud platform, designed for processing large-dimensional graphs. The main principle embed-

Keywords

Computing system, graph, dataset, discrete mathematics instructions set, associative processing, computing accelerator, supercomputer

ded in the architecture of the computing complex is the use of a multi-level associative memory subsystem, which significantly influenced the methodology of development and the features of the functioning of programs. A systematic description of the structure of the hardware and software of the computing complex is given. The main technical solutions that have made it possible to implement large-size associative memory based on addressable DDR4 DRAM memory are presented. The associative address translation for accessing such a drive is implemented on the basis of the trace processing unit of the *Leonard Euler* microprocessor. It is proposed to further improve the technical means of the subsystem of network interaction based on a bidirectional ring topology and high-speed 100Gb Ethernet lines

Received 27.05.2024
Accepted 26.06.2024

REFERENCES

[1] Majeed A., Rauf I. Graph theory: a comprehensive survey about graph theory applications in computer science and social networks. *Inventions*, 2020, vol. 5, iss. 1, art. 10. DOI: https://doi.org/10.3390/inventions5010010

[2] Mondal B., De K. Overview applications of graph theory in real field. *IJSRCSEIT*, 2017, vol. 2, no. 5, pp. 751–759.

[3] Popov A.Yu. On the implementation of the Ford-Fulkerson algorithm on the Multiple Instruction and Single Data computer system. *Nauka i obrazovanie: nauchnoe izdanie* [Science and Education of the BMSTU], 2014, no. 9 (in Russ.). EDN: TDPOMV

[4] Popov A.Yu. The study of Kruskal's and Prim's algorithms on the Multiple Instruction and Single Data stream computer system. *Nauka i obrazovanie: nauchnoe izdanie* [Science and Education of the MSTU], 2015, no. 11 (in Russ.). EDN: VDRHNX

[5] Ibrahim A.H., Abdelhalim M.B., Hussein H., et al. Analysis of x86 instruction set usage for Windows 7 applications. *2nd Int. Conf. on Computer Technology and Development*, 2010, pp. 511–516. DOI: https://doi.org/10.1109/ICCTD.2010.5645851

[6] Qian Z., Wei J., Xiang Y., et al. A performance evaluation of DRAM access for in-memory databases. *IEEE Access*, 2021, vol. 9, pp. 146454–146470.
DOI: https://doi.org/10.1109/ACCESS.2021.3123379

[7] Aananthakrishnan S., Ahmed N.K., Cave V., et al. PIUMA: programmable integrated unified memory architecture. *arXiv:2010.06277*.
DOI: https://doi.org/10.48550/arXiv.2010.06277

[8] Aasawat T.K., Reza T., Ripeanu M. How well do CPU, GPU and hybrid graph processing frameworks perform? *IEEE IPDPSW*, 2018, pp. 458–466.
DOI: https://doi.org/10.1109/IPDPSW.2018.00082

[9] Zou Y., Lin M. GridGAS: an I/O-efficient heterogeneous FPGA+CPU computing platform for very large-scale graph analytics. *FPT*, 2018, pp. 246–249.
DOI: https://doi.org/10.1109/FPT.2018.00045

[10] Lumsdaine A., Gregor D.P., Hendrickson B., et al. Challenges in parallel graph processing. *Parallel Process. Lett*., 2007, vol. 17, no. 1, pp. 5–20.
DOI: https://doi.org/10.1142/S0129626407002843

[11] Cormen T.H., Leiserson C.E., Rivest R.L., et al. Introduction to algorithms. MIT Press, 2009.

[12] Ahn J., Hong S., Yoo S., et al. A scalable processing-in-memory accelerator for parallel graph processing. *ACM/IEEE 42nd ISCA*, 2015, pp. 105–117.
DOI: https://doi.org/10.1145/2749469.2750386

[13] Angizi S., Fan D. GraphiDe: a graph processing accelerator leveraging In-DRAM-computing. *GLSVLSI '19*, 2019, pp. 45–50.
DOI: https://doi.org/10.1145/3299874.3317984

[14] Dysart T., Kogge P., Deneroff M., et al. Highly scalable near memory processing with migrating threads on the emu system architecture. *IA3*, 2016, pp. 2–9.
DOI: https://doi.org/10.1109/IA3.2016.007

[15] Gui C., Zheng L., He B., et al. A survey on graph processing accelerators: challenges and opportunities. *J. Comput. Sci. Technol.*, 2019, vol. 34, no. 2, pp. 339–371.
DOI: https://doi.org/10.1007/s11390-019-1914-z

[16] Hennessy J., Patterson D. Domain specific architectures. In: *Computer architecture*. Elsevier, 2017, pp. 540–606.

[17] Popov A., Ibragimov S., Dubrovin E. Teragraph heterogeneous system for ultra-large graph processing. In: Voevodin V., Sobolev S., Yakobovskiy M., Shagaliev R. (eds). *Supercomputing. RuSCDays 2022. Lecture Notes in Computer Science*, vol. 13708. Cham, Springer, 2022, pp. 574–590. DOI: https://doi.org/10.1007/978-3-031-22941-1_42

[18] Popov A. An introduction to the MISD technology. *Proc. Annual Hawaii Int. Conf. on System Sciences*, 2017, pp. 1003–1012.

[19] Popov A.Yu. Principles of the organization of a heterogeneous computing system with a set of commands of discrete mathematics. *Informatsionnye tekhnologii* [Information Technologies], 2020, vol. 26, no. 2, pp. 67–79 (in Russ.).
DOI: https://doi.org/10.17587/it.26.67-79

[20] Popov A.Yu. Application of a heterogeneous computing system with a discrete mathematics instruction set to solve large-scale graphs problems. *Informatsionnye tekhnologii* [Information Technologies], 2019, vol. 25, no. 11, pp. 682–690 (in Russ.).
DOI: https://doi.org/10.17587/it.25.682-690

Popov A.Yu. — Dr. Sc. (Eng.), Assoc. Professor, Department of Computer Systems and Networks, BMSTU (2-ya Baumanskaya ul. 5, str. 1, Moscow, 105005 Russian Federation).

Ibragimov S.V. — Senior Lecturer, Department of Computer Systems and Networks, BMSTU (2-ya Baumanskaya ul. 5, str. 1, Moscow, 105005 Russian Federation).

Dubrovin E.N. — Assistant, Department of Computer Systems and Networks, BMSTU (2-ya Baumanskaya ul. 5, str. 1, Moscow, 105005 Russian Federation).

Kalitventsev M.P. — Engineer, Research Institute, Scientific and Educational Complex "Informatics and Control Systems", BMSTU (2-ya Baumanskaya ul. 5, str. 1, Moscow, 105005 Russian Federation).

Geine M.A. — Engineer, Research Institute, Scientific and Educational Complex "Informatics and Control Systems", BMSTU (2-ya Baumanskaya ul. 5, str. 1, Moscow, 105005 Russian Federation).

Li J. — Engineer, Research Institute, Scientific and Educational Complex "Informatics and Control Systems", BMSTU (2-ya Baumanskaya ul. 5, str. 1, Moscow, 105005 Russian Federation).

Paramazyan G.A. — Engineer, Research Institute, Scientific and Educational Complex "Informatics and Control Systems", BMSTU (2-ya Baumanskaya ul. 5, str. 1, Moscow, 105005 Russian Federation).

Kurokhtin T.M. — Engineer, Research Institute, Scientific and Educational Complex "Informatics and Control Systems", BMSTU (2-ya Baumanskaya ul. 5, str. 1, Moscow, 105005 Russian Federation).

Zalimkhanov D.A. — Engineer, Research Institute, Scientific and Educational Complex "Informatics and Control Systems", BMSTU (2-ya Baumanskaya ul. 5, str. 1, Moscow, 105005 Russian Federation).

Please cite this article in English as:

Popov A.Yu., Ibragimov S.V., Dubrovin E.N., et al. *Teragraph* computer architecture. *Herald of the Bauman Moscow State Technical University, Series Instrument Engineering*, 2025, no. 1 (150), pp. 113–136 (in Russ.). EDN: XQZRQE