ISSN 2079-3316 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ №4(27), 2015, с. 3–28
УДК 004.27

Д. Н. Змеев, Н. Н. Левченко, А. С. Окунев, А. Л. Стемпковский

Принципы организации системы ввода/вывода параллельной потоковой вычислительной системы

Аннотация. В статье рассматриваются основные принципы работы блока ввода/вывода данных параллельной потоковой вычислительной системы «Буран», приводится анализ влияния различных алгоритмов ввода данных на эффективность решения задач. Описываются средства, позволяющие уменьшать необходимый размер ассоциативной памяти ключей и анализируются результаты экспериментов на программной модели параллельной потоковой вычислительной системы.

Ключевые слова и фразы: блок ввода/вывода, алгоритмы ввода данных, ассоциативная память, параллельная потоковая вычислительная система.

Введение

С начала прошлого десятилетия в области высокопроизводительных вычислений всё большую остроту приобретает проблема распараллеливания вычислений. Как видно из рис. 1, именно в это время производители процессоров перешли к многоядерной архитектуре. Фактически была достигнута предельная производительность одного ядра, что связано не только с пиковой частотой, но и с ограничениями по отводу тепла с единицы площади. В результате пользователи получили более производительные многоядерные системы, но с программами, которые остались последовательными, поэтому их пришлось адаптировать к «новым» реалиям.

Несмотря на то, что закон Мура продолжает действовать (верхняя кривая Transistors на рис. 1), ограничение на рассеиваемую мощность (Typical Power) с единицы площади кристалла препятствует дальнейшему росту тактовых частот процессоров (Frequency) и поэтому растет число ядер на одном кристалле (Number Cores).

Однако процесс распараллеливания программ по вычислительным ядрам оказался не так прост, а когда стали создавать высокопроизводительные вычислительные системы кластерного типа с числом ядер,

© Д. Н. Змеев, Н. Н. Левченко, А. С. Окунев, А. Л. Стемпковский, 2015
© Институт проблем проектирования в микроэлектронике РАН, 2015
© Программные системы: теория и приложения, 2015

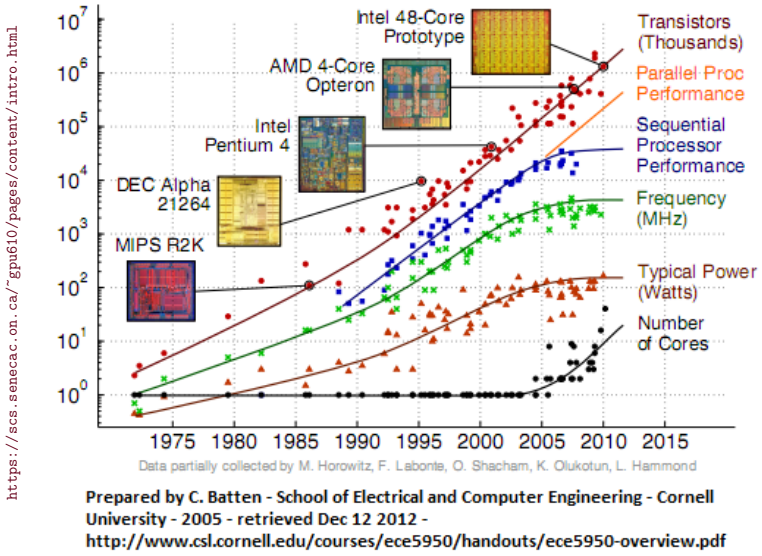


Рис. 1. Использование передовой элементной базы (закон Мура)

исчисляемых десятками и сотнями тысяч, распараллеливание еще более усложнилось. В результате, на сегодняшний день растет число алгоритмов решения задач, эффективно распараллеливающихся всего лишь на десятки, в лучшем случае — на сотни вычислительных ядер. В первую очередь это связано с использованием традиционной модели вычислений (фон Неймана), лежащей в основе современных высокопроизводительных вычислительных систем кластерного типа, и которые создаются на основе коммерчески доступных многоядерных кристаллов. Реальная производительность таких систем, насчитывающих десятки и сотни тысяч вычислительных ядер, падает вплоть до долей процентов от пиковой на ряде актуальных задач.

Одним из путей решения проблемы повышений реальной производительности и степени масштабируемости вычислений является переход к новым моделям организации вычислений. Это, в частности, подтверждается мнением известного международного эксперта в области создания суперкомпьютеров, одного из авторов списка TOP500, Джека Донгарры: «...модель dataflow позволит сделать систему не такой чувствительной к задержкам доступа к памяти и синхронизации

и «выжать» максимум параллелизма из алгоритмов, потому что там можно явно прописать зависимости по данным» [1]. Далее он говорит, что это единственный путь для будущих систем со сверхбольшим количеством процессоров.

Одной из таких моделей является потоковая модель вычислений с динамически формируемым контекстом, разработкой которой и реализацией её в архитектуре параллельной потоковой вычислительной системы (ППВС) «Буран» занимаются в Институте проблем проектирования в микроэлектронике РАН [2–5].

При использовании вычислительных систем под системой ввода/вывода традиционно понимается преобразование данных либо из формата, в котором они находятся во внешней памяти, либо из формата различных внешних устройств (датчики, системы видеонаблюдения, средства ручного ввода и др.) в формат прямоадресуемой памяти (фактически, ОЗУ), непосредственно с которой работает вычислительная система.

В предлагаемой авторами архитектуре ОЗУ в обычном понимании отсутствует, то есть пользователь не может получить какое-либо данное или изменить его, обратившись по адресу в оперативном пространстве задачи. В ППВС «Буран» все оперативные данные циркулируют в виде особых структур — токенов (см. раздел 1), что требует более «глубокого» преобразования входных данных по сравнению с традиционными системами.

Тем самым процесс ввода токенов в вычислительную систему в предлагаемой архитектуре специфичен и требует разработки нетрадиционных аппаратно-программных средств. Фактически система ввода/вывода представляет собой прежде всего преобразователь данных из обычного представления в формат токенов.

Кроме того, особенностью ППВС является то, что работа вычислительной системы может начинаться до поступления полного набора входных данных, а также то, что архитектура реализована с использованием развитой системы распределения вычислений и наличием аппаратной ассоциативной памяти в каждом вычислительном ядре [6].

Другое основное назначение системы ввода — управление активностью выполнения всей задачей. Поскольку при реализации принципа начала работы по готовности данных может образовываться большое количество готовых к исполнению пакетов (отсюда и рост общего параллелизма), этими «активностями» можно управлять. Причем, в

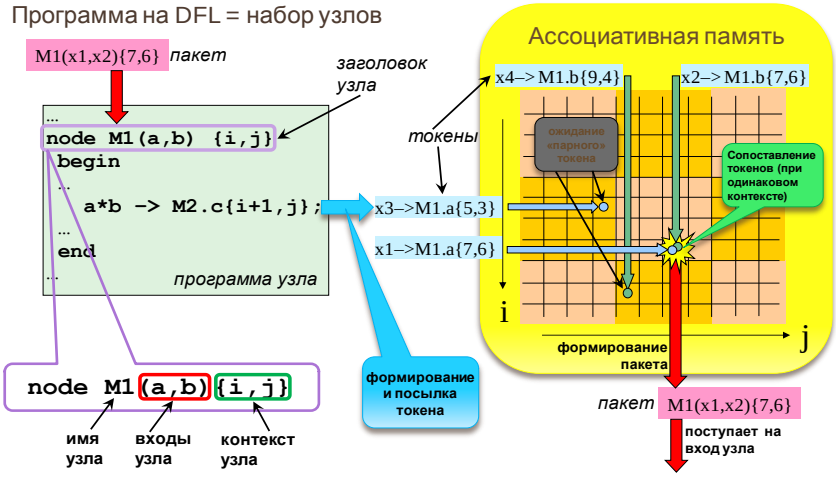


Рис. 2. Потоковая модель вычислений с динамически формируемым контекстом

предлагаемой архитектуре эти средства реализованы как программно, так и аппаратно. С помощью этих средств управления возможно распределение вычислений по пространству (по вычислительным ядрам) и во времени (по этапам).

Для эффективной реализации этих особенностей и служит оригинальная система ввода/вывода (СВВ) данных ППВС «Буран».

1. Потоковая модель вычислений с динамически формируемым контекстом

В основе потоковой модели вычислений с динамически формируемым контекстом [7] лежит активация неделимых вычислительных квантов по готовности данных (рис. 2). Вычислительный квант — это программный узел, который, будучи активированным, дорабатывает до конца без приостановки вычислительного процесса и ожидания каких-либо дополнительных внешних данных.

Для программирования создан параллельный язык DFL, который существенно снижает семантический разрыв между программным и аппаратным обеспечением.

Программа для ППВС представляет собой набор описаний программных узлов, каждый из которых состоит из заголовка узла и

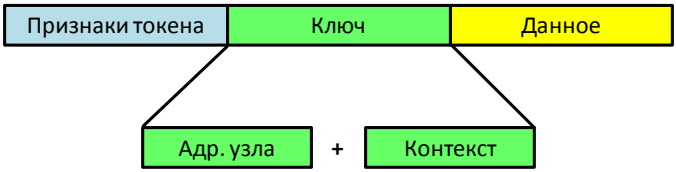


Рис. 3. Структура токена

программного кода. Заголовок программного узла содержит имя узла, список входов и список атрибутов узла — контекст. Активация программного узла происходит только тогда, когда на все входы конкретного узла (одного узла с определенным именем и контекстом) поступят все требуемые элементы данных — токены. Токен представляет собой структуру (рис. 3), содержащую непосредственно данное, набор служебных полей и ключ, который однозначно определяет местоположение этого данного в виртуальном адресном пространстве задачи.

При активации программного узла выполняется код программы над пришедшими на вход данными в пакете структурой данных, возникающей в результате обработки токенов в процессоре сопоставлений. Программа вычисляет новые величины (исключительно на основе значений входов и атрибутов ключа) и посылает их на другие программные узлы (входы узлов), причем атрибуты ключа адресата вычисляются непосредственно в этой же программе.

Память в потоковой модели вычислений служит только для временного содержания токенов. Как только токены приходят на все входы своего узла, происходит образование пакета и активация программы узла. При образовании пакета, как правило, участвующие в его образовании токены удаляются из ассоциативной памяти. Исключением является применение механизма кратности токена, реализация которого позволяет токену участвовать в образовании числа пакетов, равного значению его кратности. При исчерпании кратности токен удаляется из памяти. Различные программные узлы взаимодействуют между собой исключительно через отправку токенов, которые в свою очередь активируют новые программные узлы.

Следует отметить, что в узле-отправителе определяется как передаваемое значение, так и адрес получателя, то есть данная модель вычислений работает в парадигме «раздачи». В этом состоит принципиальное отличие нашего подхода от традиционного, когда вычис-

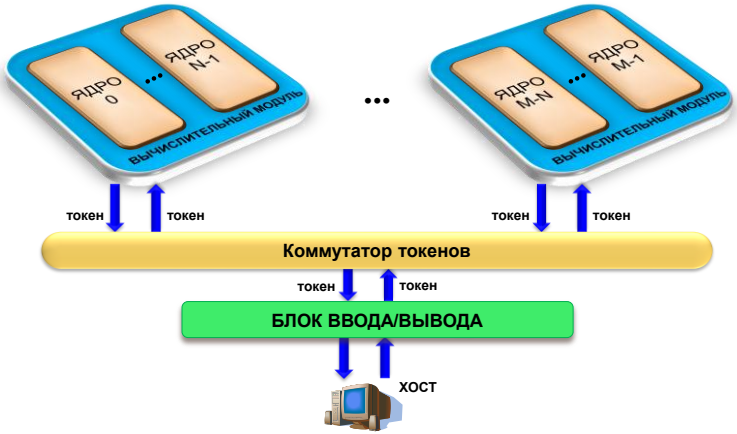


Рис. 4. Базовая архитектура параллельной потоковой вычислительной системы

лительный процесс сам запрашивает нужные ему данные из памяти или у других процессов, то есть работает в парадигме «сбора» [8].

2. Архитектура параллельной потоковой вычислительной системы

Архитектура параллельной потоковой вычислительной системы (рис. 4) реализует потоковую модель вычислений с динамически формируемым контекстом [9].

На рис. 5 приведена архитектура вычислительного модуля ППВС, состоящего из вычислительных ядер. В общих чертах работа данной системы заключается в следующем.

Выполнение задачи начинается с поступления данных в блок ввода/вывода, в котором происходит первичное преобразование их в токены, вычисляется значение хэш-функций. Далее токены поступают в коммутатор токенов, затем — на внутренний коммутатор токенов (ВКТ) соответствующего вычислительного модуля (номер вычислительного ядра принадлежит данному вычислительному модулю), осуществляющего их доставку на вход процессора сопоставления (ПС) [10–13]. Процессор сопоставления выявляет данные с одинаковым контекстом, относящиеся к одной и той же программе узла.

Если нужное для сопоставления данное на момент прихода токена отсутствует, то пришедший токен будет ожидать в ПС поступления

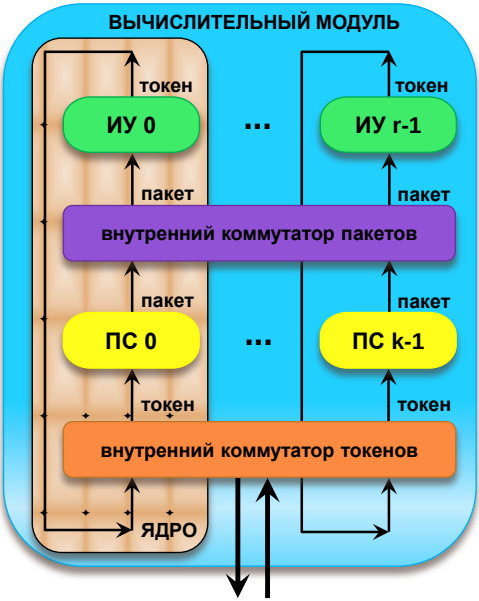


Рис. 5. Архитектура вычислительного модуля ппвс

«парного» ему токена. При совпадении контекстов происходит формирование (по определенным правилам) структуры данных (пакета), которая выдается на внутренний коммутатор пакетов. Наиболее естественным образом активация вычислительных квантов осуществляется через ассоциативную память, в которой происходит синхронизация вычислительного процесса по данным.

Внутренний коммутатор пакетов играет в структуре вычислительного модуля роль аппаратного балансера, поскольку он обеспечивает поступление пакета на любое свободное исполнительное устройство (ИУ) в пределах вычислительного модуля, тем самым обеспечивая выравнивание возможной неравномерности генерации пакетов различными процессорами сопоставления.

В исполнительном устройстве [14–16] происходит выполнение программы узла, которая может содержать как команды обработки поступивших на узел данных, так и команды изменения вычислительного контекста узла, идентифицирующего данные в виртуальном пространстве задачи. Следовательно, в программе узла можно выделить команды обработки данных и команды обработки контекста.

В основном эти команды не имеют взаимной зависимости по данным и, тем самым, могут быть выполнены в параллель. Сформированные в результате выполнения программы узла токены выдаются из ИУ на блок хэш-функций.

В блоке формирования хэш-функций происходит вычисление функции распределения по пространству. Распределение вычислений по пространству (другими словами — локализация по вычислительным ядрам) позволяет добиться как равномерной загрузки вычислениями ядер системы, так и равномерной нагрузки на коммуникационную сеть. Распределение виртуальных узлов осуществляется посредством задания функции распределения, присоединяемой к DFL-программе, причем функция не является частью программы, то есть в отлаженную программу не требуется вносить каких-либо корректив при изменении распределения вычислений. Аргументом функции распределения является ключ токена, а значением — номер вычислительного ядра. Затем токены, содержащие адрес узла, в который они должны поступить и где над ними должна выполниться соответствующая программа, поступают на внутренний коммутатор токенов.

Во внутреннем коммутаторе токенов происходит анализ, «принадлежит» ли токен текущему вычислительному модулю. Если токен «принадлежит» этому модулю, то он передается на требуемое вычислительное ядро в пределах модуля, в противном случае токен передается в другой вычислительный модуль. В этом случае токен выдается в системный коммутатор токенов для доставки его в соответствующее ядро.

Вычислительная система хорошо масштабируема благодаря тому, что и между вычислительными модулями, представляющими собой группу ядер, и внутри их действует один и тот же протокол взаимодействия, осуществляющий доставку токенов к ядрам.

При увеличении числа ядер в ППВС падение реальной производительности на задачах со сложно организованными данными происходит существенно медленнее, чем при решении подобных задач на вычислительных системах с классической архитектурой. Это достигается, во-первых, за счет использования принципа потока данных, когда готовые к выполнению данные активируют выполнение программы узла; во-вторых, активируемому узлу для своего полного выполнения не требуется никаких дополнительных данных; в-третьих, узлы выполняются полностью независимо друг от друга и, в-четвертых,

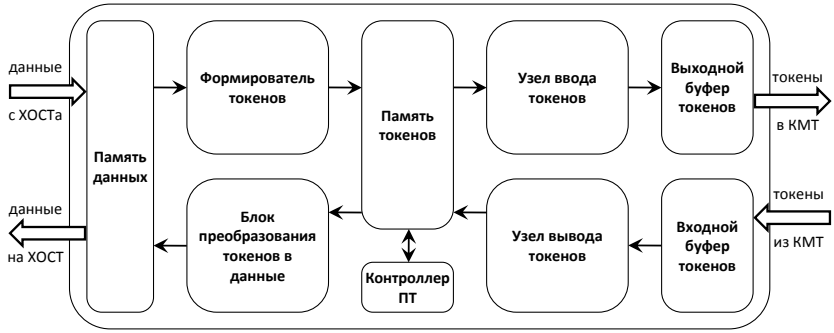


Рис. 6. Структурная схема блока ввода/вывода данных

благодаря правильному выбору хеш-функции (функции распределения вычислений по группам ядер), сокращающей число межъядерных передач токенов.

Благодаря такой модели вычислений, реализуемой в архитектуре ППВС, выполнение программы и ее создание не зависят от конкретной конфигурации вычислительной системы, то есть программа будет работать как на одном ядре, так и на любом другом числе ядер без повторной компиляции.

3. Структура блока ввода/вывода данных

Основной функцией блока ввода/вывода (БВВ) данных является преобразование поступающих на вход БВВ данных в токены, выработка хэш-функции, а также обратное преобразование выдаваемых на хост токенов в данные.

Блок ввода/вывода данных конструктивно состоит из следующих основных элементов: формирователя токенов (ФТ), памяти токенов (ПТ), узла ввода токенов (УВТ), узла вывода токенов и блока преобразования токенов в данные (рис. 6).

Данные с хост-машины или с датчиков поступают в виде массива данных в память данных БВВ. Помимо данных, в БВВ поступает информация о режиме формирования токенов, в соответствии с которой в ФТ по заранее определенным шаблонам происходит преобразование данных в токены — определяется разрядность и положение структурных единиц токенов (поля контекста, поля маски, поля данных и других). Структурная схема формирователя токенов приведена на рис. 7. В ФТ также происходит формирование номеров ядер и этапов

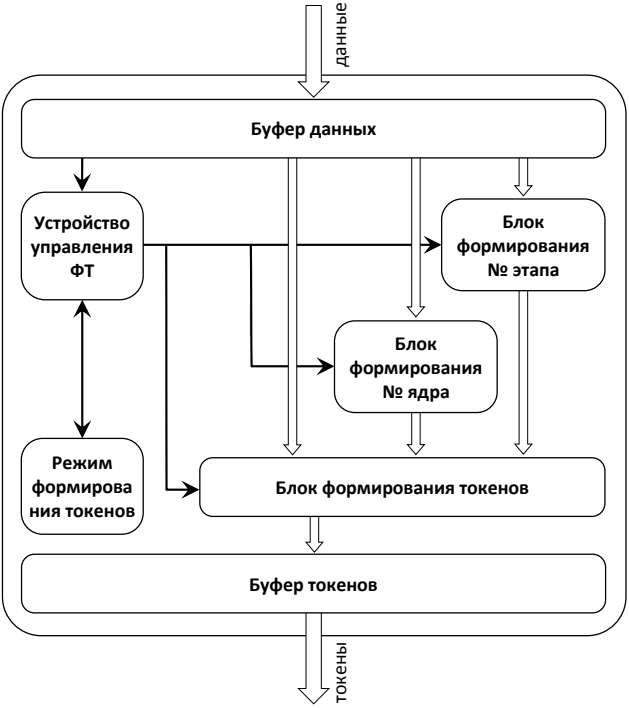


Рис. 7. Структурная схема формирователя токенов

в соответствии с функциями распределения по пространству и во времени [17].

Затем токены поступают на узел ввода токенов, в котором в соответствии с заданным режимом ввода токенов в систему определяется очередность их поступления в коммутатор токенов вычислительной системы. Предварительно происходит анализ на основе номера этапа токена — к «активным» или «пассивным» этапам он относится, то есть следует ли его задержать в БВВ до того момента, пока этап этого токена станет «активным».

После этого токены поступают в выходной буфер токенов, а затем — в коммутатор токенов вычислительной системы.

Обратный процесс существенно проще. Токены, выдаваемые на хост-машину, поступают из коммутатора токенов во входной буфер, откуда передаются в узел вывода токенов. В этом узле происходит

сортировка токенов (поскольку токены выдаются по мере их готовности) по номеру задач/подзадач, к которым принадлежат токены, и запись токенов в соответствующие области памяти токенов БВВ.

Далее блок преобразования токенов в данные извлекает токены из памяти и в соответствии с заданным режимом производит преобразование их в выходной массив данных, который выдается на хост-машину.

4. Основные режимы работы системы ввода/вывода данных

Авторами в работе [18] был кратко описан блок ввода данных и основные режимы его работы. Ниже приводится более детальное описание доступных режимов ввода данных в систему и вывода данных на хост-машину.

Система ввода/вывода выполняет следующие функции:

- формирование токенов;
- ввод токенов в систему;
- вывод и преобразование токенов.

При формировании токенов из данных СВВ поддерживает несколько режимов: «сквозной», работа с шаблоном, работа со счетчиками и др.

Первый режим обеспечивает транзитную передачу полностью сформированного токена с хост-машины. Он используется для ввода в вычислительную систему «одиночных» данных, а также данных нерегулярной структуры, для которых затруднительно определить правило формирования токенов.

Режим «работа с шаблоном» обеспечивает формирование токенов в соответствии с заранее заданным шаблоном, который предварительно загружается в формирователь токенов. Шаблон содержит в себе поля ключа и признаков токена, а также непосредственно правило формирования токена и изменения соответствующих полей ключа.

Режим «работа со счетчиками» используется при передаче в вычислительную систему и последующем формировании токенов из n-мерных массивов данных. В этом случае в поля «ключа» входят индексы передаваемого элемента массива. В формирователе токенов предварительно передаются маски счетчиков, которые определяют размерность и положение индексов в полях ключа, а также верхние и нижние границы счетчиков. В формирователе токенов происходит аппаратная проверка взаимного наложения масок. Счетчики имеют

иерархическую вложенность: внутренний увеличивается с каждым следующим токеном в соответствии с заданным правилом формирования токенов, а внешние — при достижении внутренними счетчиками «верхних» границ.

Вторая функция СВВ — ввод токенов в систему — поддерживает следующие режимы: ввод разреженных «данных», ввод «активных этапов» и др.

Режим ввода разреженных «данных» обеспечивает посылку в вычислительную систему только токенов с ненулевыми данными, что позволяет реализовать одну из особенностей параллельной потоковой вычислительной системы — отсутствие необходимости хранения «лишних» данных, над которыми не производится никакой работы. При этом возможно наложение на поле «данных» маски, причем байты, закрытые маской, считаются нулевыми.

Режим ввода «активных этапов» обеспечивает передачу в вычислительную систему только тех данных, которые относятся к активным этапам. Токены остальных этапов задерживаются в памяти токенов. Тем самым значительно уменьшается объем ассоциативной памяти, который требуется для выполнения программы. Изменяя размер этапа, можно регулировать степень параллелизма выполнения задачи. Одна из возможных реализаций заключается в подсчете количества отправленных и принятых токенов. После отправки заданного числа токенов узел ввода токенов приостанавливает ввод до получения результатов их обработки, затем отправляет следующую «порцию» токенов.

Третья функция СВВ — вывод и преобразование токенов, которые сформировались в результате выполнения задачи, — поддерживает следующие режимы: преобразование токенов по формату, накопление массива данных, выдача единичного данного, выдача массива данных.

5. Алгоритмы ввода токенов в коммутатор токенов

Одним из основных преимуществ архитектуры ППВС является то, что работа вычислительной системы может начинаться до поступления полного набора входных данных. В этой связи БВВ реализует различные алгоритмы ввода данных, которые учитывают эту специфику [18]. Эти алгоритмы работают как в формирователе токенов, так и в узле ввода токенов, и позволяют формировать токены из передаваемых данных, а также выстраивать очередность поступления

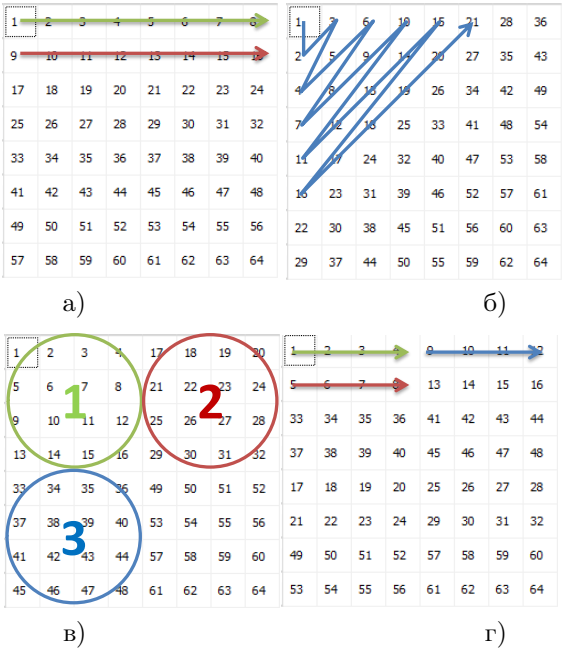


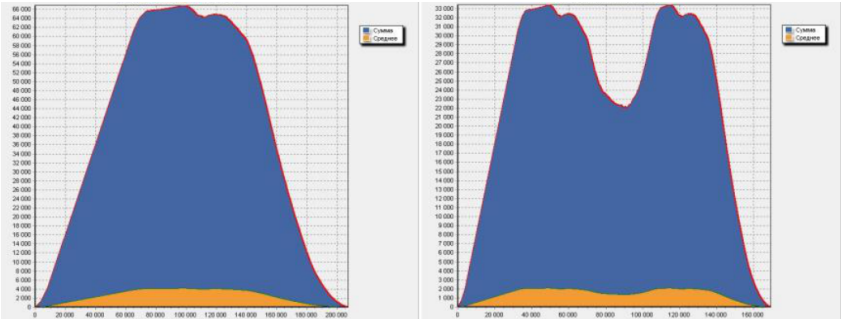
Рис. 8. Схематическое представление алгоритмов ввода данных

токенов в вычислительную систему наиболее эффективным способом для каждой конкретной задачи в зависимости от ее алгоритма обработки данных.

Перечислим основные алгоритмы ввода данных:

- последовательный ввод по строкам;
- последовательный ввод по столбцам;
- поблочно-квадратный ввод;
- поблочно-квадратный половинчатый ввод;
- ввод колонками;
- ввод планками;
- блочно-точечный ввод;
- диагональный ввод;
- ввод векторов для БПФ.

На рис. 8 схематично изображены некоторые из них. Ввод данных (токенов) производится в порядке возрастания нумерации, в



а) б)

Рис. 9. Суммарная загрузка памяти ключей на задаче «Быстрое преобразование Фурье»

направлении изображенных стрелок, т.е. в соответствии с номерами, изображенными на рисунках.

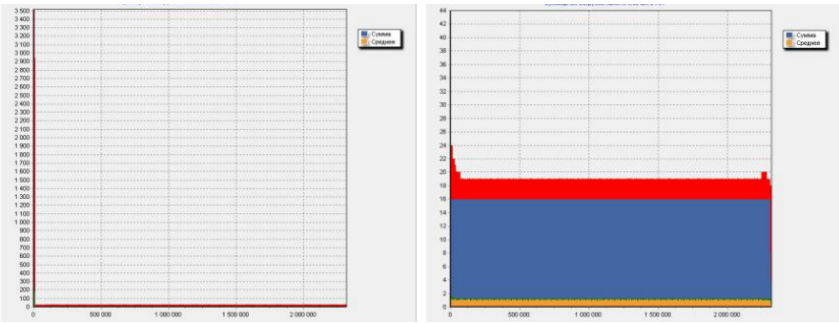
На рис. 8.а ввод данных осуществляется построчно в порядке возрастания номеров элементов массива. На рис. 8.б представлен диагональный ввод данных. На рис. 8.в ввод данных осуществляется поблочно, каждый из блоков является квадратом. В приведенном примере матрица разбивается на четыре части, которые вводятся в систему последовательно по строкам внутри блока. На рис. 8.г ввод данных осуществляется поблочно, но, в отличие от предыдущего алгоритма, на первом этапе вводится верхняя половина блока, а на втором этапе, который начинается после ввода всех строк верхней половины блока, вводятся уже строки нижней половины блока данных.

Такие алгоритмы позволяют запускать обработку данных на всех имеющихся вычислительных ядрах практически одновременно, а не последовательно, по мере заполнения данными ядер, что позволяет повысить реальную производительность вычислительной системы и снизить требуемый объем ассоциативной памяти ключей[1].

6. Исследование эффективности ввода данных на задачах

Исследования проводились на задачах «Быстрое преобразование Фурье», «Сложение векторов», «Перемножение матриц» на программ-

[1]Ассоциативная память ключей обеспечивает сопоставление («объединение») данных с одинаковым ключом фактически с темпом поступления данных на ее вход



а) б)

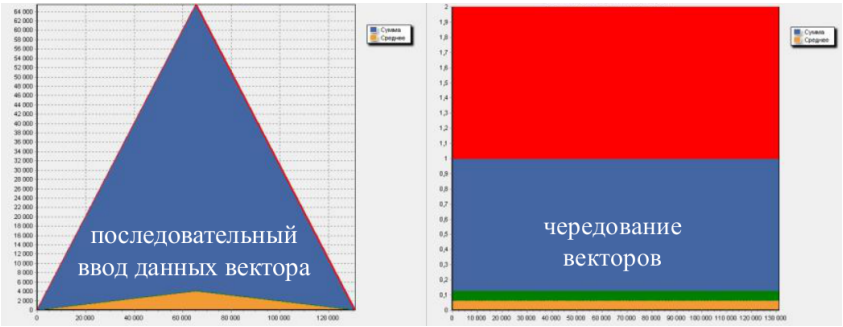
Рис. 10. Суммарная загрузка памяти ключей на задаче «Перемножение матриц»

ной блочно-регистровой модели [**19**, **20**] параллельной потоковой вычислительной системы.

На рис. 9 приведена суммарная загрузка ассоциативной памяти ключей в устройстве сопоставления на задаче «Быстрое преобразование Фурье» [**20**] для последовательного алгоритма ввода данных (рис. 9.а) и алгоритма ввода данных методом чередования полувекторов (рис. 9.б). Требуемый объем ассоциативной памяти ключей при применении второго метода ввода данных уменьшен в два раза (с 66000 ячеек ассоциативной памяти ключей до 33000). При этом время выполнения задачи уменьшилось практически на 20% по сравнению с алгоритмом последовательного ввода данных.

На задаче «Перемножение матриц» (рис. 10) при вводе двух матриц методом чередования (рис. 10.б) максимальный объем ассоциативной памяти составляет всего 44 ячейки, что почти на два порядка лучше показателя по объему памяти ключей при последовательном вводе (рис. 10.а) этих матриц (максимально необходимый объем ассоциативной памяти для этого метода ввода данных составил 3500 ячеек).

Следует также отметить алгоритм «Сложение векторов» (рис. 11), на котором удалось добиться требуемого объема ассоциативной памяти ключей всего в одну ячейку при переменном вводе данных (рис. 11.б) каждого из векторов, иначе пришлось бы хранить один из векторов (рис. 11.а) в процессоре сопоставления до момента начала поступления элементов второго вектора.



а) б)

Рис. 11. Суммарная загрузка памяти ключей на задаче «Сложение векторов»

Кроме того, с использованием разработанных методов на поведенческой модели получено ускорение прохождения задачи «Сложение векторов» на 16%, а для задачи «Быстрое преобразование Фурье» на 18%. Это свидетельствует о том, что при помощи разработанных методов ввода данных в систему не только уменьшается требуемый объем ассоциативной памяти ключей, но и растет производительность системы.

Заключение

В данной статье рассмотрены принципы построения блока ввода данных для параллельной потоковой вычислительной системы и особенности функционирования данного блока. Кроме функций, описанных в статье, некоторые узлы БВВ используются для построения блока откачки/подкачки данных, являющегося частью иерархической ассоциативной памяти вычислительной системы.

Влияние различных алгоритмов ввода данных на эффективность прохождения задач на ППВС и на необходимый размер ассоциативной памяти ключей процессора сопоставлений связано с возможностью начала вычислений, не дожидаясь загрузки всего объема входных данных программы.

Анализ полученных экспериментальных результатов показал, что суммарная загрузка ассоциативной памяти ключей в процессоре сопоставлений на различных задачах при применении разных алгоритмов

ввода существенно уменьшается. На некоторых задачах было получено существенное ускорение прохождения этих задач без использования управляющих алгоритмов ввода данных.

На основе проведенных исследований можно сделать вывод, что система, реализующая описанную модель вычислений, позволит создать масштабируемую суперкомпьютерную систему с высокой реальной производительностью [**21**]. При этом аппаратура сможет сама выявлять скрытый параллелизм [**22**, **23**] задачи, то есть параллелизм на уровне данных, готовых к обработке, что упрощает как параллельное программирование, так и создание компиляторов [**24**]. Эксперименты на программной модели ППВС также подтверждают, что система ввода/вывода, описанная в статье, играет важную роль в организации эффективной работы параллельной потоковой вычислительной системы.

Разработанные аппаратно-программные средства ввода/вывода данных и проведенные исследования актуальны для разрабатываемых новых суперкомпьютерных систем эксафлопсного класса [**25**, **26**].

Список литературы

[1] И. Лёвшин. «Свежий взгляд из-за океана (Беседа с Джеком Донгаррой о тенденциях и реалиях)», *Суперкомпьютеры*, 2013, №2(14), с. 6–8. ↑5

[2] А. Л. Стемпковский, Н. Н. Левченко, А. С. Окунев, В. В. Цветков. «Параллельная потоковая вычислительная система — дальнейшее развитие архитектуры и структурной организации вычислительной системы с автоматическим распределением ресурсов», *Информационные технологии*, 2008, №10, с. 2–7. ↑5

[3] Н. Н. Левченко, А. С. Окунев, А. Л. Стемпковский, «Использование модели вычислений с управлением потоком данных и реализующей ее архитектуры для систем эксафлопсного уровня производительности», *Всероссийская научно-техническая конференция «Проблемы разработки перспективных микро- и наноэлектронных систем-2012»*, Сборник трудов, МЭС-2012, ИППМ РАН, М., 2012, с. 459–462, URL: www.mes-conference.ru/data/year2012/pdf/D95.pdf ↑5

[4] Ю. Н. Никольская, Д. Н. Змеев, Н. Н. Левченко, А. С. Окунев, «Взаимодействие между процессами в параллельной потоковой вычислительной системе», *Материалы Международной научно-технической конференции «Многопроцессорные вычислительные и управляющие системы»*. Т. 1, МВУС-2009 (Дивноморское, Россия, 2009), ТТИ ЮФУ, Таганрог, с. 103–106. ↑5

[5] А. Л. Стемпковский, Н. Н. Левченко, А. С. Окунев, «Архитектура высокопроизводительной вычислительной системы с высокой реальной производительностью», *Материалы Международной научно-технической конференции «Суперкомпьютерные технологии: разработка, программирование, применение»*. Т. 1, СКТ-2010, Таганрог–М., 2010, с. 153–157. ↑[5]

[6] Д. Н. Змеев, Н. Н. Левченко, А. С. Окунев, А. В. Климов, «Способы регулирования вычислений в параллельной потоковой вычислительной системе», *Проблемы разработки перспективных микро- и наноэлектронных систем-2014*, Сборник трудов. Т. IV, ред. А. Л. Стемпковский, ИППМ РАН, М., 2014, с. 79–82. ↑[5]

[7] А. В. Климов, Н. Н. Левченко, А. С. Окунев. «Преимущества потоковой модели вычислений в условиях неоднородных сетей», *Информационные технологии и вычислительные системы*, 2012, №2, с. 36–45. ↑[6]

[8] А. В. Климов, Н. Н. Левченко, А. С. Окунев, А. Л. Стемпковский. «Суперкомпьютеры, иерархия памяти и потоковая модель вычислений», *Программные системы: теория и приложения*, **5**:1(19) (2014), с. 15–36, URL: `http://psta.psiras.ru/read/psta2014_1_15-36.pdf` ↑[8]

[9] А. В. Климов, Н. Н. Левченко, А. С. Окунев, А. Л. Стемпковский, «Использование архитектуры потока данных для создания сверхвысокопроизводительных вычислительных систем», *Материалы Второй Всероссийской научно-технической конференции «Суперкомпьютерные технологии»*, СКТ-2012 (24–29 сентября 2012 года, с. Дивноморское, Геленджикский район), 2012, с. 64–68. ↑[8]

[10] Д. Е. Яхонтов, Н. Н. Левченко, А. С. Окунев, «Принципы работы блока специальных операций модуля ассоциативной памяти параллельной потоковой вычислительной системы ППВС», *Материалы Международной научно-технической конференции «Суперкомпьютерные технологии: разработка, программирование, применение»*. Т. 1, СКТ-2010, Таганрог–М., 2010, с. 166–170. ↑[8]

[11] Н. Н. Левченко, А. С. Окунев, Д. Е. Яхонтов, «Методы снижения энергопотребления АЗУ ППВС», *Материалы Второй Всероссийской научно-технической конференции «Суперкомпьютерные технологии»*, СКТ-2012 (24–29 сентября 2012 года, с. Дивноморское, Геленджикский район), 2012, с. 73–78. ↑[8]

[12] Д. Н. Змеев, Н. Н. Левченко, А. С. Окунев, Д. Е. Яхонтов, «Исследование последовательности выбора полей токена при поиске в ассоциативном процессоре ППВС», *Материалы Второй Всероссийской научно-технической конференции «Суперкомпьютерные технологии»*, СКТ-2012 (24–29 сентября 2012 года, с. Дивноморское, Геленджикский район), 2012, с. 48–53. ↑[8]

[13] Н. Н. Левченко, А. С. Окунев, Д. Е. Яхонтов, «Исследование работы процессора сопоставления параллельной потоковой вычислительной системы «Буран»», *Всероссийская научнотехническая конференция «Проблемы разработки перспективных микро- и наноэлектронных систем-2012»*, Сборник трудов, МЭС-2012, ИППМ РАН, М., 2012, с. 467–470, URL: http://www.mes-conference.ru/data/year2012/pdf/D118.pdf ↑[8]

[14] Н. Н. Левченко, А. С. Окунев, «Исследование применения многовходовых узлов в параллельной потоковой вычислительной системе для задач фильтрации», *Материалы Пятой Международной молодежной научно-технической конференции «Высокопроизводительные вычислительные системы»*, ВПВС-2008 (Таганрог, Россия, 2008), с. 367–370. ↑[9]

[15] Н. Н. Левченко, А. С. Окунев, «Анализ различных вариантов реализаций многовходовых узлов в параллельной потоковой вычислительной системе», *Материалы Международной научно-технической конференции «Многопроцессорные вычислительные и управляющие системы»*. Т. 1, МВУС-2009 (Дивноморское, Россия, 2009), ТТИ ЮФУ, Таганрог, с. 75–77. ↑[9]

[16] Н. Н. Левченко, А. С. Окунев, «Об одном подходе к применению векторного функционального устройства в ППВС», *Материалы Международной научно-технической конференции «Суперкомпьютерные технологии: разработка, программирование, применение»*. Т. 1, СКТ-2010, Таганрог–М., 2010, с. 124–126. ↑[9]

[17] А. В. Климов, Н. Н. Левченко, А. С. Окунев, А. Л. Стемпковский. «Автоматическое распараллеливание для гибридной системы с потоковым ускорителем», *Информационные технологии и вычислительные системы*, 2011, №2, с. 3–11. ↑[12]

[18] Д. Н. Змеев, Н. Н. Левченко, А. С. Окунев, А. Л. Стемпковский, «Исследование принципов работы блока ввода данных для параллельной потоковой вычислительной системы», *Проблемы разработки перспективных микро- и наноэлектронных систем-2014*, Сборник трудов. Т. IV, ред. А. Л. Стемпковский, ИППМ РАН, М., 2014, с. 83–86, URL: http://www.mes-conference.ru/data/year2014/pdf/D164.pdf ↑[13,14]

[19] Н. Н. Левченко, А. С. Окунев, Д. Н. Змеев, «Расширение возможностей поведенческой блочно-регистровой модели параллельной потоковой вычислительной системы», *Материалы Пятой Международной молодежной научно-технической конференции «Высокопроизводительные вычислительные системы»*, ВПВС-2008 (Таганрог, Россия, 2008), с. 371–374. ↑[17]

[20] Д. Н. Змеев, Н. Н. Левченко, А. С. Окунев, Л. С. Ходош, «Средства визуализации процесса прохождения задачи в программной модели ППВС», *Материалы Международной научнотехнической конференции «Многопроцессорные вычислительные и управляющие системы»*. Т. 1, МВУС-2009 (Дивноморское, Россия, 2009), ТТИ ЮФУ, Таганрог, с. 49–52. ↑17

[21] А. В. Климов, Н. Н. Левченко, А. С. Окунев, «Модель вычислений с управлением потоком данных как средство решения проблем больших распределенных систем», *Материалы Второй Всероссийской научно-технической конференции «Суперкомпьютерные технологии»*, СКТ-2012 (24–29 сентября 2012 года, с. Дивноморское, Геленджикский район), 2012, с. 303–307. ↑19

[22] А. В. Климов, Н. Н. Левченко, А. С. Окунев, «Аппаратно-программные средства преодоления неравномерной загрузки отдельных устройств в параллельной потоковой вычислительной системе», *Материалы Международной научнотехнической конференции «Многопроцессорные вычислительные и управляющие системы» в составе мультиконференции «Проблемы информационно-компьютерных технологий и мехатроники»*. Т. 1, Издательство ТТИ ЮФУ, Таганрог, 2007, с. 92–95. ↑19

[23] А. Л. Стемпковский, А. В. Климов, Н. Н. Левченко, А. С. Окунев. «Методы адаптации параллельной потоковой вычислительной системы под задачи отдельных классов», *Информационные технологии и вычислительные системы*, 2009, №3, с. 12–21. ↑19

[24] Арк. В. Климов, Н. Н. Левченко, А. С. Окунев, А. Л. Стемпковский, «Автоматическое распараллеливание последовательных программ для гибридной системы с ускорителем на основе потока данных», *Параллельные вычислительные технологии*, Труды международной научной конференции, ПаВТ'2011 (Москва, 28 марта–1 апреля 2011 г.), с. 211–218, URL: http://omega.sp.susu.ac.ru/books/conference/PaVT2011 ↑19

[25] T. Gelle. "Supercomputing's exaflop target", *Communications of the ACM*, **54**:8 (2011), pp. 16–18. ↑19

[26] Y. Chen, "Towards scalable I/O architecture for exascale systems", *Proceedings of the 2011 ACM international workshop on Many task computing on grids and supercomputers*, MTAGS '11, pp. 43–48. ↑19

Рекомендовал к публикации *к.ф.-м.н. С. А. Романенко*

Об авторах:

Дмитрий Николаевич Змеев

Научный сотрудник Института проблем проектирования в микроэлектронике РАН. Область научных интересов: параллельное программирование, потоковая модель вычислений, моделирование потоковой архитектуры, программные модели, RTL-проектирование.

e-mail: zmejevdn@ippm.ru

Николай Николаевич Левченко

К.т.н., заведующий отделом Высокопроизводительных микроэлектронных вычислительных систем Института проблем проектирования в микроэлектронике РАН. Область научных интересов: параллельные вычисления, параллельное программирование, модель вычислений управляемых потоком данных, использование ассоциативной памяти в высокопроизводительных вычислительных системах.

e-mail: nick@ippm.ru

Анатолий Семенович Окунев

К.т.н., ведущий научный сотрудник Института проблем проектирования в микроэлектронике РАН. Область научных интересов: архитектуры высокопроизводительных вычислительных систем, модель вычислений с управлением потоком данных, применение различных САПР для реализации архитектурных решений, применение принципов ассоциативности при разработке различных аппаратных узлов.

e-mail: oku@ippm.ru

Александр Леонидович Стемпковский

Академик, доктор технических наук, директор Института проблем проектирования в микроэлектронике РАН. Область научных интересов: фундаментальные проблемы построения систем автоматизации проектирования СБИС, методология проектирования сложных микроэлектронных устройств, архитектуры высокопроизводительных микроэлектронных вычислительных систем.

e-mail: ippm@ippm.ru

Пример ссылки на эту публикацию:

Д. Н. Змеев, Н. Н. Левченко, А. С. Окунев, А. Л. Стемпковский. «Принципы организации системы ввода/вывода параллельной потоковой вычислительной системы», *Программные системы: теория и приложения*, 2015, **6**:4(27), с. 3–28.

URL: http://psta.psiras.ru/read/psta2015_4_3-28.pdf

Dmitriy Zmejev, Nikolay Levchenko, Anatoliy Okunev, Aleksandr Stempkovsky. *The principles of organization of the I/O system of the parallel dataflow computing system.*

Abstract. This article discusses the basic principles of I/O block of the parallel dataflow computing system "Buran", analyses the effect of different algorithms for data entry on the efficiency of task computation. Describes a means to reduce the required size of content-addressable memory of keys and analyzes the results of experiments on the programming model of the parallel dataflow computing system. (*In Russian*).

Key Words and Phrases: I/O block, algorithms of data entry, content-addressable memory, parallel dataflow computing system.

References

[1] I. Lëvshin. "Svezhiy vzglyad iz-za okeana (Beseda s Dzhekom Dongarroy o tendentsiyakh i realiyakh)", *Superkomp'yutery*, 2013, no.2(14), pp. 6–8 (in Russian).

[2] A. L. Stempkovskiy, N. N. Levchenko, A. S. Okunev, V. V. Tsvetkov. "Parallel stream computing system — further development of architecture and structural organization of computing system with automatic resource allocation", *Informatsionnyye tekhnologii*, 2008, no.10, pp. 2–7 (in Russian).

[3] N. N. Levchenko, A. S. Okunev, A. L. Stempkovskiy, "The usage of dataflow computing model and architecture realizing these for exaflops performance system", *All-Russia Science & Technology Conference "Problems of Advanced Micro- and Nanoelectronic Systems Development"*, MES-2012, IPPM RAN, M., 2012, pp. 459–462 (in Russian), URL: http://www.mes-conference.ru/data/year2012/pdf/D95.pdf

[4] Yu. N. Nikol'skaya, D. N. Zmeyev, N. N. Levchenko, A. S. Okunev, "The interaction between the processes in a parallel computer system stream", *Materialy Mezhdunarodnoy nauchno-tekhnicheskoy konferentsii "Mnogoprotsessornyye vychislitel'nyye i upravlyayushchiye sistemy"*. V. 1, MVUS-2009 (Divnomorskoye, Rossiya, 2009), TTI YuFU, Taganrog, pp. 103–106 (in Russian).

[5] A. L. Stempkovskiy, N. N. Levchenko, A. S. Okunev, "The architecture of high-performance computing systems with high real performance", *Materialy Mezhdunarodnoy nauchno-tekhnicheskoy konferentsii "Superkomp'yuternyye tekhnologii: razrabotka, programmirovaniye, primeneniye"*. V. 1, SKT-2010, Taganrog–M., 2010, pp. 153–157 (in Russian).

[6] D. N. Zmeyev, N. N. Levchenko, A. S. Okunev, A. V. Klimov, "Methods of regulation of computation in parallel dataflow computating system", *All-Russia Science & Technology Conference "Problems of Advanced Micro- and Nanoelectronic Systems Development"*. V. IV, MES-2014, ed. A. L. Stempkovskiy, IPPM RAN, M., 2014, pp. 79–82 (in Russian).

[7] A. V. Klimov, N. N. Levchenko, A. S. Okunev. "Benefits of dataflow computation model within nonuniform networks", *Informatsionnyye tekhnologii i vychislitel'nyye sistemy*, 2012, no.2, pp. 36–45 (in Russian).

© D. N. Zmejev, N. N. Levchenko, A. S. Okunev, A. L. Stempkovsky, 2015
© Institute for Design Problems in Microelectronics, 2015
© Program systems: Theory and Applications, 2015

[8] A. V. Klimov, N. N. Levchenko, A. S. Okunev, A. L. Stempkovskiy. “Supercomputers, memory hierarchy and dataflow computation model”, *Programmnye Sistemy: Teoriya i Prilozheniya*, **5**:1 (2014), pp. 15–36 (in Russian), URL: http://psta.psiras.ru/read/psta2014_1_15-36.pdf

[9] A. V. Klimov, N. N. Levchenko, A. S. Okunev, A. L. Stempkovskiy, “Using the architecture of the data stream to build computer systems supercomputer”, *Materialy Vtoroy Vserossiyskoy nauchno-tekhnicheskoy konferentsii “Superkomp’yuternyye tekhnologii”*, SKT-2012 (24–29 sentyabrya 2012 goda, s. Divnomorskoye, Gelendzhikskiy rayon), 2012, pp. 64–68 (in Russian).

[10] D. Ye. Yakhontov, N. N. Levchenko, A. S. Okunev, “The principles of operation of the unit special operations unit parallel associative memory Stream computing system HAPP”, *Materialy Mezhdunarodnoy nauchno-tekhnicheskoy konferentsii “Superkomp’yuternyye tekhnologii: razrabotka, programmirovaniye, primeneniye”*. V. 1, SKT-2010, Taganrog–M., 2010, pp. 166–170 (in Russian).

[11] N. N. Levchenko, A. S. Okunev, D. Ye. Yakhontov, “Methods to reduce AMU energy HAPP”, *Materialy Vtoroy Vserossiyskoy nauchno-tekhnicheskoy konferentsii “Superkomp’yuternyye tekhnologii”*, SKT-2012 (24–29 sentyabrya 2012 goda, s. Divnomorskoye, Gelendzhikskiy rayon), 2012, pp. 73–78 (in Russian).

[12] D. N. Zmeyev, N. N. Levchenko, A. S. Okunev, D. Ye. Yakhontov, “Research field selection sequence token when searching associative processor HAPP”, *Materialy Vtoroy Vserossiyskoy nauchno-tekhnicheskoy konferentsii “Superkomp’yuternyye tekhnologii”*, SKT-2012 (24–29 sentyabrya 2012 goda, s. Divnomorskoye, Gelendzhikskiy rayon), 2012, pp. 48–53 (in Russian).

[13] N. N. Levchenko, A. S. Okunev, D. Ye. Yakhontov, “Study of mapping processor for dataflow parallel computing system ”Buran””, *All-Russia Science & Technology Conference ”Problems of Advanced Micro- and Nanoelectronic Systems Development”*, MES-2012, IPPM RAN, M., 2012, pp. 467–470 (in Russian), URL: http://www.mes-conference.ru/data/year2012/pdf/D118.pdf

[14] N. N. Levchenko, A. S. Okunev, “Study of the use of multinodes in parallel computing system for streaming filtration problems”, *Materialy Pyatoy Mezhdunarodnoy molodezhnoy nauchno-tekhnicheskoy konferentsii “Vysokoproizvoditel’nyye vychislitel’nyye sistemy”*, VPVS-2008 (Taganrog, Rossiya, 2008), pp. 367–370 (in Russian).

[15] N. N. Levchenko, A. S. Okunev, “Analysis of the various embodiments of multi-streaming nodes in parallel computer system”, *Materialy Mezhdunarodnoy nauchno-tekhnicheskoy konferentsii “Mnogoprotsessornyye vychislitel’nyye i upravlyayushchiye sistemy”*. V. 1, MVUS-2009 (Divnomorskoye, Rossiya, 2009), TTI YuFU, Taganrog, pp. 75–77 (in Russian).

[16] N. N. Levchenko, A. S. Okunev, “An approach to the use of vector functional device HAPP”, *Materialy Mezhdunarodnoy nauchno-tekhnicheskoy konferentsii “Superkomp’yuternyye tekhnologii: razrabotka, programmirovaniye, primeneniye”*. V. 1, SKT-2010, Taganrog–M., 2010, pp. 124–126 (in Russian).

[17] A. V. Klimov, N. N. Levchenko, A. S. Okunev, A. L. Stempkovskiy. “Automatic parallelization for hybrid system with dataflow accelerator”, *Informatsionnyye tekhnologii i vychislitel’nyye sistemy*, 2011, no.2, pp. 3–11 (in Russian).

[18] D. N. Zmeyev, N. N. Levchenko, A. S. Okunev, A. L. Stempkovskiy, “Research the principles of operation of the input block for the parallel dataflow

computing system", *All-Russia Science & Technology Conference "Problems of Advanced Micro- and Nanoelectronic Systems Development"*. V. IV, MES-2014, ed. A. L. Stempkovskiy, IPPM RAN, M., 2014, pp. 83–86 (in Russian), URL: http://www.mes-conference.ru/data/year2014/pdf/D164.pdf

[19] N. N. Levchenko, A. S. Okunev, D. N. Zmeyev, "Empowering behavioral model of block-register parallel stream computing system", *Materialy Pyatoy Mezhdunarodnoy molodezhnoy nauchno-tekhnicheskoy konferentsii "Vysokoproizvoditel'nyye vychislitel'nyye sistemy"*, VPVS-2008 (Taganrog, Rossiya, 2008), pp. 371–374 (in Russian).

[20] D. N. Zmeyev, N. N. Levchenko, A. S. Okunev, L. S. Khodosh, "Visualization of the passage of the problem in the software model HAPP", *Materialy Mezhdunarodnoy nauchnotekhnicheskoy konferentsii "Mnogoprotsessornyye vychislitel'nyye i upravlyayushchiye sistemy"*. V. 1, MVUS-2009 (Divnomorskoye, Rossiya, 2009), TTI YuFU, Taganrog, pp. 49–52 (in Russian).

[21] A. V. Klimov, N. N. Levchenko, A. S. Okunev, "Model calculations with flow control as a means of solving the problems of large distributed systems", *Materialy Vtoroy Vserossiyskoy nauchno-tekhnicheskoy konferentsii "Superkomp'yuternyye tekhnologii"*, SKT-2012 (24–29 sentyabrya 2012 goda, s. Divnomorskoye, Gelendzhikskiy rayon), 2012, pp. 303–307 (in Russian).

[22] A. V. Klimov, N. N. Levchenko, A. S. Okunev, "Hardware and software means to overcome the uneven loading of individual devices stream in a parallel computer system", *Materialy Mezhdunarodnoy nauchnotekhnicheskoy konferentsii "Mnogoprotsessornyye vychislitel'nyye i upravlyayushchiye sistemy" v sostave mul'tikonferentsii "Problemy informatsionno-komp'yuternykh tekhnologiy i mekhatroniki"*. V. 1, Izdatel'stvo TTI YuFU, Taganrog, 2007, pp. 92–95 (in Russian).

[23] A. L. Stempkovskiy, A. V. Klimov, N. N. Levchenko, A. S. Okunev. "Methods of adapting parallel dataflow computer system to various task types", *Informatsionnyye tekhnologii i vychislitel'nyye sistemy*, 2009, no.3, pp. 12–21 (in Russian).

[24] Ark. V. Klimov, N. N. Levchenko, A. S. Okunev, A. L. Stempkovskiy, "Automatic parallelization of sequential programs hybrid system for an accelerator-based data stream", *Parallel'nyye vychislitel'nyye tekhnologii*, Trudy mezhdunarodnoy nauchnoy konferentsii, PaVT'2011 (Moskva, 28 marta–1 aprelya 2011 g.), pp. 211–218 (in Russian), URL: http://omega.sp.susu.ac.ru/books/conference/PaVT2011"Supercomputing's exaflop target", *Communications of the ACM*, **54**:8 (2011), pp. 16–18

[25] Y. Chen, "Towards scalable I/O architecture for exascale systems", *Proceedings of the 2011 ACM international workshop on Many task computing on grids and supercomputers*, MTAGS '11, pp. 43–48.

Sample citation of this publication:

Dmitriy Zmejev, Nikolay Levchenko, Anatoliy Okunev, Aleksandr Stempkovsky. "The principles of organization of the I/O system of the parallel dataflow computing system", *Program systems: theory and applications*, 2015, **6**:4(27), pp. 3–28. (*In Russian*).

URL: http://psta.psiras.ru/read/psta2015_4_3-28.pdf