УДК 004.27 DOI: 10.31114/2078-7707-2022-3-129-134

Реализация функционала мажоритарного элемента в ППВС на основе процессора сопоставлений

Н.Н. Левченко, Д.Н. Змеев

Институт проблем проектирования в микроэлектронике РАН, г. Москва, nick@ippm.ru

***Аннотация* — При создании высокопроизводительных вычислительных систем много усилий уделяется вопросам надежности. Децентрализованная архитектура параллельной потоковой вычислительной системы и распределение ресурсов, поддержанное на уровне аппаратуры, позволяют без дополнительных аппаратных затрат реализовать т.н. метод программного мажорирования. В статье рассматриваются варианты реализации функционала мажоритарного элемента в параллельной потоковой вычислительной системе на основе процессора сопоставлений с использованием особенностей его функционирования. Проанализированы преимущества и недостатки вариантов реализации функционала мажоритарного элемента.**

***Ключевые слова* — параллельная потоковая вычислительная система, мажоритарный элемент, процессор сопоставлений, ассоциативная память.**

I. Введение

Одной из проблем стоящих перед разработчиками суперкомпьютерных вычислительных систем, объединяющих в своём составе десятки и сотни тысяч вычислительных ядер, является проблема надежности [1]. В вычислительных системах, функционирующих в традиционной модели вычислений, разработано большое число различных методов и средств, направленных на обеспечение их надежности, как на уровне всей системы, так и на уровне отдельных микросхем [2]. Для оригинальных моделей вычислений такие средства контроля и обеспечения надежности не всегда применимы и эффективны, приходится их адаптировать к специфике архитектуры вычислительной системы.

К оригинальным моделям вычислений разработчиков высокопроизводительных вычислительных систем вынуждает переходить проблема распараллеливания вычислений. К таким моделям относится перспективная потоковая модель вычислений с динамически формируемым контекстом [3]. Данную модель вычислений реализует параллельная потоковая вычислительная система (ППВС) «Буран», которой на аппаратном уровне имманентно присущ параллелизм вычислений.

Одним из основных методов повышения надежности является модульное резервирование (мажорирование). Этот метод хорошо ложится на децентрализованную архитектуру ППВС и поддержанное на уровне аппаратуры распределение вычислений.

Целью данной работы является анализ вариантов реализации в ППВС одного из блоков модульного резервирования – мажоритарного элемента на основе ключевого блока вычислительной системы – процессора сопоставлений.

II. Модель вычислений и архитектура ППВС

Потоковая модель вычислений с динамически формируемым контекстом характеризуется активацией вычислительных квантов по готовности данных. Вычислительным квантом называется программа относительно небольшого размера (программный узел), которая активируется, как только на ее вход поступят все данные. Между собой эти кванты взаимодействуют при помощи сообщений – структуры данных, содержащих операнд, служебные поля и контекст. Контекст однозначно определяет положение операнда в виртуальном адресном пространстве задачи.

Именно сопоставление сообщений и является основой как потоковой модели вычислений с динамически формируемым контекстом, так и стандартной модели вычислений с управлением потоком данных (dataflow). Сопоставлением называется процедура, в результате которой по заранее определенным правилам производится поиск сообщений с «совпадающими» признаками. В результате положительного поиска (при совпадении) образуется пакет – структура данных, готовая к обработке. Для его обработки не требуется запрос каких-либо дополнительных данных, а соответственно отсутствует необходимость прерывания вычислительного процесса. Обработка пакета происходит в соответствии с последовательностью команд программного узла, адрес которого присутствует в самом пакете. В результате обработки пакета образуются новые сообщения, которые либо посылаются на другие узлы, либо в качестве результата вычислений выдаются на ХОСТ-машину. Адреса узлов и контекст новых сообщений вычисляются непосредственно в программе узла, т.е. параллельная программа в целом выполняется в парадигме «раздачи».

Потоковая модель вычислений с динамически формируемым контекстом реализуется в архитектуре параллельной потоковой вычислительной системы (ППВС) «Буран» [4-5].

ППВС является многоядерной масштабируемой вычислительной системой. Между вычислительными ядрами в системе передаются единицы информации в

виде токенов. Токеном называется структура данных, в состав которой входит предаваемый операнд, ключ с контекстом и адресом программного узла, маска и ряд других специальных признаков. Коммутация между ядрами осуществляется на основе значения номера ядра, вырабатываемого блоком хэширования на основе настраиваемой функции распределения вычислений, входным параметром которой является контекст токена. Группа вычислительных ядер в рамках одного кристалла называется вычислительным модулем.

В состав вычислительного модуля входят следующие узлы и блоки: процессор сопоставлений (ПС), исполнительное устройство (ИУ), блок хэш-функций, внутренний коммутатор токенов и коммутатор пакетов. Причем, число ПС и ИУ может варьироваться и не обязательно должно быть равным друг другу.

Принципы программирования и архитектура параллельной потоковой вычислительной системы ориентированы на высокопараллельное выполнение задач. Это является изначальным свойством потоковой модели вычислений с динамически формируемым контекстом. На рис. 1 представлена функциональная схема вычислительного модуля ППВС.

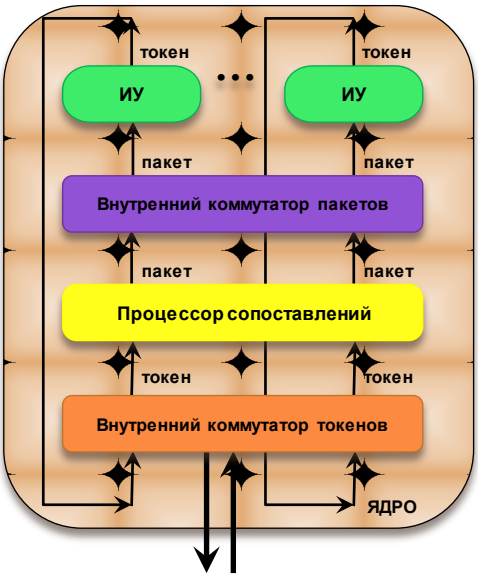

Рис. 1. Функциональная схема вычислительного модуля ППВС

Архитектура вычислительного модуля представляет собой, фактически, набор процессоров сопоставлений связанных с набором исполнительных устройств посредством внутреннего коммутатора пакетов и внутреннего коммутатора токенов. Токены по процессорам сопоставлений распределяются согласно номеру процессора сопоставлений, который вычисляется согласно заранее выбранной хэш-функции [6] и содержится в одном из полей токена.

Процессор сопоставлений является фактически главным устройством управления вычислительным процессом в ППВС, поддерживающим сравнение ключей токенов задачи, обеспечивающим все принципы модели вычислений с управлением потоком данных и предотвращающим проблемы переполнения или недозагрузки аппаратных ресурсов в процессе вычислений.

Токен, поступивший на вход ПС, называется верхним, а который ожидает сопоставления в ПС – нижним. По приходу верхнего токена запускается процедура поиска «совпадающих» токенов. После взаимодействия с нижними токенами, верхний может быть удален, либо записан в ПС (тогда он становится нижним), запись происходит и при отсутствии откликов. После взаимодействия с верхним токеном, нижний может быть оставлен без изменений, перезаписан с другими значениями полей либо удален из ПС. При взаимодействии верхнего и нижнего токенов образуются пакеты.

Пакеты, в свою очередь, передаются во внутренний коммутатор пакетов, связывающий процессоры сопоставлений с группой исполнительных устройств. Внутренний коммутатор пакетов фактически выполняет функцию аппаратного балансёра, передавая пакеты на любое свободное в данный момент исполнительное устройство вычислительного модуля. Это реализуется благодаря тому, что исполнительные устройства в вычислительной системе обезличены, а каждое из ИУ содержит параллельную программу всей задачи.

III. Варианты реализации мажоритарного элемента

Благодаря специфике работы процессора сопоставлений и входящей в его состав ассоциативной памяти [7], функционал мажоритарного элемента может быть реализован непосредственно в процессоре сопоставлений без дополнительных аппаратных затрат.

Ассоциативная память ключей (АПК) входит в состав процессора сопоставления. АПК предназначена для хранения ключей с масками, фиксации записанных ключей в регистре занятости, формирования адресов свободных ячеек для записи ключей, а также для поиска совпадения с ключами, хранящимися в ней. В отличие от традиционной прямоадресуемой памяти, в которой доступ к данному происходит по адресу, ассоциативная память функционирует по содержимому и доступ ко всем ячейкам памяти осуществляется параллельно. Подавая на вход ассоциативной памяти данное, в случае обнаружения «совпадающего» данного, на выход выдается адрес ячеек памяти, в которых это данное содержится.

Особенностью ассоциативной памяти ключей, используемой в параллельной потоковой вычислительной системе, является наличие как внешней, так и внутренней маски (т.н. тернарная ассоциативная память). Маска позволяет проводить сравнение токенов только по области ключа, не закрытой ею, остальная часть содержимого ключа игнорируется и считается совпавшей вне зависимости от содержимого [8, 9].

Реализация мажоритарного элемента при помощи процессора сопоставлений возможна следующими вариантами:

- использование полей контекста;
- использование стандартных токенов;
- использование спецтокенов tMJR;
- использование многовходовых токенов.

В общем случае, на вход ПС в результате выполнения программы должно поступить три токена, анализируя которые, будет определен «правильный» результат (два из трёх одинаковых значений операнда).

A. Использование полей контекста

Специфика функционирования ассоциативной памяти предполагает одновременное сравнение входного данного со всеми данными, которые уже находятся в ассоциативной памяти. В ассоциативной памяти ключей процессора сопоставлений хранятся и проверяются ключи токена. Таким образом, если проверяемый операнд будет находиться в поле ключа токена, то ассоциативная память фактически будет выполнять функции мажоритарного элемента.

В этом случае первый из трёх токенов пришедший в процессор сопоставлений после процедуры «Поиск» останется в ПС дожидаться прихода остальных токенов. Второй пришедший токен сопоставится с первым токеном. Положительное сопоставление этих токенов (сформирован т.н. «отклик») означает, что операнды совпали и можно продолжать вычисления (формируется соответствующий токен с операндом в поле данных). Если отклики отсутствуют, то второй токен также записывается в ПС, где будет находиться в ожидании оставшегося токена. Как только третий токен поступит в ПС, то будет выполнена стандартная процедура «Поиск», в результате которой операнд пришедшего токена будет сравниваться с операндами первых двух токенов. При положительном сопоставлении будет сформирован токен для продолжения вычислений.

Описанный вариант имеет ряд недостатков – это необходимость сборки «мусора», который может остаться в ПС, а также он не дает ответа на вопрос – что делать, если операнды трёх токенов не совпадают между собой. Под «мусором» понимается один из токенов (при положительном сопоставлении, или все три при отрицательном), остающийся в ПС. Неопределенность же возникает, когда все три токена приходят с разными операндами в поле контекста. Она связана с тем, что эти токены просто не провзаимодействуют (не сопоставятся) друг с другом и, соответственно, не будет сформирован сигнал «АВОСТ».

Одним из решений описанных проблем является добавление в контекст кроме операнда, еще и координат мажоритарного элемента в виртуальном адресном пространстве задачи. В этом случае процедура «Поиск» для таких токенов разбивается на два этапа. На первом этапе происходит сопоставление по операнду, а на втором этапе – по координатам. Соответственно, в случае, когда все операнды различны, на первом этапе отклики будут отсутствовать, а на втором – будет сформировано два отклика, что свидетельствует о приходе всех трёх токенов и отсутствии совпадений. Следовательно, во-первых, может быть сформирован сигнал «АВОСТ», а во-вторых, все лишние токены могут быть стёрты.

К преимуществам данного варианта относится:

- эффективное использование аппаратных ресурсов необходимых для сравнения операндов;
- возможность проведения сравнения, не дожидаясь прихода всех операндов.

К недостаткам относится:

- увеличение размера ключа, который должен храниться в АПК, либо резервирование отдельной АПК небольшого размера для мажоритарного элемента.

Помимо первого варианта, который опирается на специфику работы АП, имеется возможность реализовать аналогичный функционал, но без нахождения операнда в одном из полей ключа токенов. В следующих трех вариантах токены в полях ключа содержат только координаты мажоритарного элемента в виртуальном адресном пространстве задачи, а операнд находится в поле «Данное». Сравнение операндов происходит уже после того, как произошло сопоставление токенов в ассоциативной памяти ключей.

B. Использование стандартных токенов

Стандартными называются токены, которые не обладают дополнительным функционалом и не изменяют хода вычислительного процесса. При отсутствии откликов верхний токен записывается в ПС. В общем случае, при взаимодействии двух стандартных токенов формируется пакет с двумя операндами (данными).

При поступлении первого токена из трёх (имеющих одинаковый контекст) в процессор сопоставлений, после процедуры «Поиск», в результате которой отклики отсутствуют, происходит его запись и дальнейшее ожидание в ПС следующих токенов.

Следующий пришедший токен после положительного сопоставления формирует с ожидающим в ПС токеном пакет и происходит удаление обоих, участвующих в сопоставлении токенов: пришедшего и ожидавшего в процессоре сопоставлений. При отрицательном сопоставлении происходит его запись и дальнейшее ожидание в ПС последнего токена. Пакет направляется на исполнение на любое свободное исполнительное устройство. В ИУ происходит сравнение двух операндов, и если они совпадают, то значение, соответствующее большинству входных значений мажоритарного элемента, определено и можно продолжать вычисления, сформировав соответствующий токен. Третье сравнение в этом случае производить не нужно, но для того, чтобы удалить из процессора сопоставлений избыточный токен, необходимо сформировать токен «стирание». При несовпадении двух операндов

следует сформировать два токена и послать их в ПС на сравнение с оставшимся токеном.

Для гарантированного обеспечения такого механизма работы, начальные три токена посылаются как «универсальные», т.е. они обеспечивают сопоставление как между «универсальными» токенами, так и между т.н. «левыми» и «правыми», т.е. токенами направленными либо на первый вход узла, либо на второй. Токены, которые образуются при несовпадении первых двух операндов, следует посылать уже как «левые», что обеспечит их сопоставление с оставшимся третьим токеном, но не между собой; поскольку два «левых», как и два «правых» токена между собой не взаимодействуют.

Если эти два токена поступят в ПС раньше третьего, то они будут дожидаться его прихода в процессоре сопоставлений. Третий пришедший токен может провзаимодействовать либо с токеном «стирание» (и тогда он просто уничтожится), либо с одним из двух «левых» токенов. В результате будет также образован пакет и выполнен на ИУ. Как и в предыдущем случае, операнды могут совпасть и тогда значение, соответствующее большинству входных значений мажоритарного элемента, определено. Помимо токена продолжающего вычисления, также будет сформирован и токен «стирание» для удаления из памяти оставшегося токена. При несовпадении операндов будет сформирован токен с третьим операндом и отправлен в ПС, но уже как «правый», чтобы обеспечить сравнение с оставшимся токеном.

На последнем этапе в ПС сопоставятся два токена («левый» и «правый»), будет сформирован пакет и выполнен на ИУ. Если операнды совпадут, то значение, соответствующее большинству входных значений мажоритарного элемента, определено, и вычисления будут продолжены, в противном случае будет сформирован сигнал «АВОСТ», т.к. все три операнда оказались разными.

К преимуществам данного варианта относится:

- использование стандартных токенов;
- проведение сравнения, не дожидаясь прихода третьего токена.

К недостаткам относится:

- увеличение времени сравнения операндов для «худшего» случая.

C. Использование спецтокенов «tMJR»

Третьим вариантом реализации является использование специального типа токенов («tMJR»). Такие токены при поступлении в процессор сопоставлений выполняют поиск только с поступившими ранее токенами того же типа (рис. 2).

Вначале все три токена приходят в ПС как «универсальные». Когда поступает первый из серии токенов, то в результате сопоставления отклики отсутствуют, а ключ этого токена помещается в ассоциативную память ключей. Второй и третий токены из серии при сопоставлении должны вызвать отклики. Если два первых токена совпадают, то возможно либо образование нового токена (согласно специфике работы данного типа токенов – при полностью аппаратной реализации алгоритма), либо образование пакета, при обработке которого могут образовываться новые токены. Помимо этого, формируется токен «стирание», который остается в ассоциативной памяти ключей и ожидает прихода третьего токена, после чего они уничтожаются.

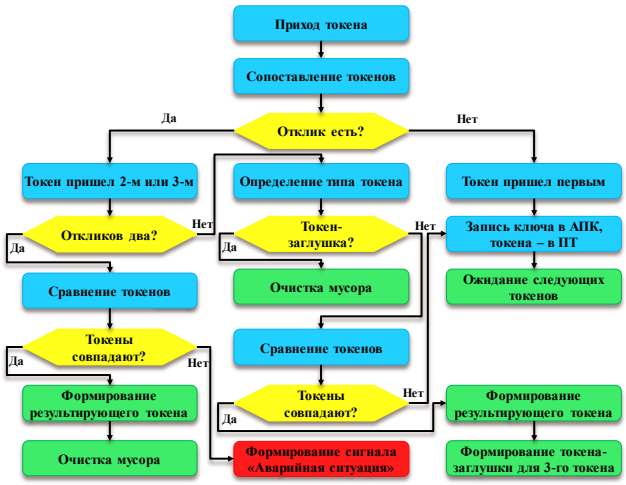

Рис. 2. Алгоритм работы процессора сопоставлений с токенами «tMJR»

В случае, когда два первых токена не совпадают, токены преобразуются в «левый» и «правый» соответственно, и дожидаются прихода третьего токена в ПС. Третий токен взаимодействует с двумя оставшимися токенами и если значение, соответствующее большинству входных значений мажоритарного элемента, определено, то формируется токен с ним для продолжения вычислений, а мусор удаляется. При отсутствии совпадения, т.е. когда все три операнда токенов разные, формируется сигнал «АВОСТ».

К преимуществам данного варианта относится:

- исполнительные устройства не задействуются в процессе работы мажоритарного элемента;
- сравнения выполняются, не дожидаясь поступления третьего токена.

К недостаткам относится:

- увеличенное время процесса мажорирования для «худшего» случая;
- введение нового типа токенов.

В отличие от предыдущего варианта, все сравнения выполняются непосредственно в процессоре сопоставлений без использования исполнительных устройств.

D. Использование многовходовых токенов

Реализовать мажоритарный элемент можно также при помощи многохводовых токенов [10]. В отличие от стандартных двухвходовых токенов, на использование которых настроен функционал ассоциативной па-

мяти ключей, многовходовые токены позволяют активировать и выполнять программные узлы с числом операндов больше двух.

Для выбора операнда организуется дополнительный трехвходовой программный узел. На каждый из входов данного узла поступает токен с признаком «Безразличный», что позволяет упростить и сделать одинаковой программы узлов формирования таких токенов. Как только все три токена поступают в процессор сопоставлений формируется пакет, который передается на исполнение в любое свободное ИУ. В программе узла происходит сравнение всех трёх данных. При обнаружении двух или трех одинаковых данных происходит посылка токена для дальнейшего исполнения задачи. В противном случае формируется сигнал «АВОСТ».

К преимуществам данного варианта относится:

- снижение требований к объему ассоциативной памяти ключей процессора сопоставлений;
- уменьшение числа пакетов, требуемых для проведения полного сравнения операндов.

Недостатками данного варианта являются:

- для начала работы необходим приход в ПС всех трёх токенов (два первых токена ожидают в процессоре сопоставлений, а третий приходит на его вход), тогда как в предыдущем варианте, работа мажоритарного элемента могла быть выполнена и после поступления первых двух одинаковых токенов;
- вариант может быть использован только в случае, когда ключи всех трёх токенов не будут содержать ошибки, в противном случае формирование пакета не состоится.

IV. Заключение

При создании высокопроизводительных вычислительных систем много усилий уделяется вопросам надежности. Параллельная потоковая вычислительная система, базирующаяся на потоковой модели вычислений с динамически формируемым контекстом, не является исключением.

Децентрализованная архитектура вычислительной системы и распределение ресурсов, поддержанное на уровне аппаратуры, позволяют без дополнительных аппаратных затрат реализовать т.н. метод программного мажорирования. В статье рассматриваются варианты реализации одного из составляющих этого метода – мажоритарного элемента.

Аппаратные возможности ПС и специфика работы одного из его блоков (ассоциативной памяти), позволяют на ее базе эффективно реализовывать различные варианты функционала мажоритарного элемента.

Литература

[1] Горбунов В.В., Елизаров Г.А., Эйсымонт Л.К. Экзафлопсные суперкомпьютеры: достижения и перспективы // Открытые системы. 2013. № 7. С. 10-15.

[2] Гаврилов С.В., Иванова Г.А., Рыжова Д.И., Стемпковский А.Л. Методы повышения надежности комбинационных микроэлектронных схем на основе мультиинтервального анализа быстродействия // Системы высокой доступности. 2015. Т. 11. № 4. С. 69-76.

[3] Левченко Н.Н., Окунев А.С., Стемпковский А.Л. Преимущества потоковой модели вычислений // Проблемы разработки перспективных микро- и наноэлектронных систем. 2018. Выпуск 3. С. 24-30.

[4] Стемпковский А.Л., Левченко Н.Н., Окунев А.С., Цветков В.В. Параллельная потоковая вычислительная система – дальнейшее развитие архитектуры и структурной организации вычислительной системы с автоматическим распределением ресурсов // ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ. 2008. №10. С. 2-7.

[5] Zmeev D.N., Klimov A.V., Levchenko N.N., Okunev A.S., Stempkovskii A.L. Features of the Architecture Implementing the Dataflow Computational Model and Its Application in the Creation of Microelectronic High-Performance Computing Systems // Russian Microelectronics. 2019. V. 48. № 5. Pp. 292-298. DOI: 10.1134/S1063739719050111.

[6] Змеев Д.Н., Климов А.В., Левченко Н.Н. Средства распределения вычислений в ППВС «Буран» и варианты реализации блока выработки хэш-функций // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС). 2016. № 2. С. 107-113.

[7] Левченко Н.Н., Окунев А.С., Змеев Д.Н., Климов А.В. Архитектура ассоциативной памяти ключей и методы предотвращения ее переполнения в параллельной потоковой вычислительной системе «Буран» // Вестник Рязанского государственного радиотехнического университета. 2018. № 65. С. 63-73.

[8] Змеев Д.Н., Кузьмин Е.Н., Левченко Н.Н., Окунев А.С. Тенденции развития архитектур ассоциативной памяти и ее применение в параллельной потоковой вычислительной системе // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС). 2016. № 2. С. 114-119.

[9] Egor Kuzmin, Nikolay Levchenko, Anatoly Okunev. Associative Processors: Application, Operation, Implementation Problems // Proceedings of IEEE East-West Design & Test Symposium (EWDTS'2019), Batumi, Georgia, Sept 13 - 16, 2019. Pp. 424-429.

[10] Левченко Н.Н., Окунев А.С., Шурчков И.О., Яхонтов Д.Е. Варианты реализации контроллера памяти параллельной потоковой вычислительной системы для работы с векторными и многовходовыми узлами // Всероссийская научно-техническая конференция «Проблемы разработки перспективных микро- и наноэлектронных систем – 2012» (МЭС-2012): сборник трудов. – М.: ИППМ РАН, 2012. С. 463-466.

Implementation of Majority Function Based on Matching Processor in the Parallel Dataflow Computing System "Buran"

N.N. Levchenko, D.N. Zmejev

Institute for Design Problems in Microelectronics of Russian Academy of Sciences, Moscow, nick@ippm.ru

***Abstract* — The reliability is one of the problems facing the developers of supercomputers with tens and hundreds of thousands of computational cores. Systems based on non-traditional computing models must also provide the required level of reliability. One of such systems is a promising data-flow computing model with a dynamically formed context. This computing model is implemented by the parallel data-flow computing system (PDCS) "Buran", to which the parallelism of computations immanently inherent at the hardware level. One of the main methods for improving reliability is modular redundancy. This method fits well with the decentralized PDCS architecture and the distribution of computations supported at the hardware level. The dataflow computing model with a dynamically formed context is characterized by the activation of computational quanta by data readiness. A computational quantum is a relatively small program that is activated as soon as all the data is received at its input. These quanta interact with each other using messages - data structures containing an operand, service fields and context. The context uniquely defines the position of the operand in the virtual address space of the task. Due to the specifics of the operation of the matching processor and the content-addressable memory included in it, the majority function (modular redundancy) can be implemented directly in the matching processor without additional hardware costs. The implementation of the majority function using the matching processor is possible in the following ways: use of context fields, use of standard tokens, use of tMJR special tokens and use of multi-input tokens. The article discusses these options, the advantages and disadvantages are given.**

***Keywords* — parallel dataflow computing system, majority function, matching processor, content-addressable memory.**

REFERENCES

[1] Gorbunov V.V., Elizarov G.A., Jejsymont L.K. Jekzaflopsnye superkomp'jutery: dostizhenija i perspektivy (Exascale supercomputers: achievements and prospects) // Otkrytye sistemy. 2013. № 7. S. 10-15.

[2] Gavrilov S.V., Ivanova G.A., Ryzhova D.I., Stempkovskij A.L. Metody povyshenija nadezhnosti kombinacionnyh mikrojelektronnyh shem na osnove mul'tiinterval'nogo analiza bystrodejstvija (Methods for improving the reliability of combinational microelectronic circuits based on multi-interval performance analysis) // Sistemy vysokoj dostupnosti. 2015. T. 11. № 4. S. 69-76.

[3] Levchenko N.N., Okunev A.S., Stempkovsky A.L. Advantages of Dataflow Computing Model // Problems of Perspective Micro- and Nanoelectronic Systems Development - 2018. Issue 3. P. 24-30. doi:10.31114/2078-7707-2018-3-24-30

[4] Stempkovskij A.L., Levchenko N.N., Okunev A.S., Cvetkov V.V. Parallel'naja potokovaja vychislitel'naja sistema – dal'nejshee razvitie arhitektury i strukturnoj organizacii vychislitel'noj sistemy s avtomaticheskim raspredeleniem resursov (Parallel dataflow computing system - the further development of architecture and the structural organization of the computing system with automatic distribution of resources) // INFORMACIONNYE TEHNOLOGII. 2008. №10. S. 2-7.

[5] Zmeev D.N., Klimov A.V., Levchenko N.N., Okunev A.S., Stempkovskii A.L. Features of the Architecture Implementing the Dataflow Computational Model and Its Application in the Creation of Microelectronic High-Performance Computing Systems // Russian Microelectronics. 2019. V. 48. № 5. Pp. 292-298.

[6] Zmejev D.N., Klimov A.V., Levchenko N.N. The means for computation distribution in the PDCS "Buran" and the implementation variants of a block of hash-functions // Problems of Perspective Micro- and Nanoelectronic Systems Development - 2016. Proceedings / edited by A. Stempkovsky, Moscow, IPPM RAS, 2016. Part 2. P. 107-113.

[7] Levchenko N.N., Okunev A.S., Zmeev D.N., Klimov A.V. Arhitektura associativnoj pamjati kljuchej i metody predotvrashhenija ee perepolnenija v parallel'noj potokovoj vychislitel'noj sisteme «Buran» (Building variants of content addressable memory of keys for parallel dataflow computing system "Buran") // Vestnik Rjazanskogo gosudarstvennogo radiotehnicheskogo universiteta. 2018. № 65. S. 63-73.

[8] Zmejev D.N., Kuzmin E.N., Levchenko N.N., Okunev A.S. Trends in development of content addressable memory architectures and its application in the parallel dataflow computing system // Problems of Perspective Micro- and Nanoelectronic Systems Development - 2016. Proceedings / edited by A. Stempkovsky, Moscow, IPPM RAS, 2016. Part 2. P. 114-119.

[9] Egor Kuzmin, Nikolay Levchenko, Anatoly Okunev. Associative Processors: Application, Operation, Implementation Problems // Proceedings of IEEE East-West Design & Test Symposium (EWDTS'2019), Batumi, Georgia, Sept 13 - 16, 2019. Pp. 424-429.

[10] Levchenko N.N., Okunev A.S., Yakhontov D.E., Shurchkov I.O. Variants of realization of controller for parallel dataflow computing system to work with vector and multioperand nodes // Problems of Perspective Micro- and Nanoelectronic Systems Development - 2012. Proceedings / edited by A. Stempkovsky, Moscow, IPPM RAS, 2012. P. 463-466.