УДК 004.27 DOI: 10.31114/2078-7707-2021-3-82-93

Автоматизированное определение оптимальной конфигурации параллельной потоковой вычислительной системы для решения конкретной задачи

А.Л. Стемпковский, С.Г. Бобков, Д.Н. Змеев, Н.Н. Левченко, Арк.В. Климов,

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

***Аннотация* — В статье рассматривается проблема нахождения оптимальной конфигурации архитектуры параллельной потоковой вычислительной системы (ППВС) для решения конкретных задач. Определены параметры изменения архитектуры ППВС и критерии оценки конфигурации вычислительной системы. Определены диапазоны изменения параметров и их влияние на поведение ППВС при решении задач. Описаны особенности вычисления критериев, а также их роль в оценке эффективности работы ППВС. На основании этих параметров и критериев разработан алгоритм автоматизированного определения оптимальной конфигурации ППВС для решения конкретной задачи, а также обозначены перспективы дальнейшей автоматизации этого процесса.**

***Ключевые слова* — параллельная потоковая вычислительная система, параметры оптимизации, критерии эффективности, потоковая модель вычислений, оптимальная архитектура.**

I. Введение

Современные суперкомпьютеры (многопроцессорные многоядерные вычислительные системы кластерного типа) основываются на массовых микропроцессорах и ускорителях вычислений GPGPU. Такие системы на задачах, входящих в тестовый пакет LINPACK, демонстрируют высокую реальную производительность. При решении же актуальных задач, в которых присутствует работа со сложноорганизованными данными, их реальная производительность не превышает 3-5%, о чем свидетельствует результаты теста HPCG [1].

Одной из причин низкой реальной производительности вычислительных систем является то, что традиционные многопроцессорные системы предоставляют пользователю малое число параметров (например, объем оперативной памяти, число процессоров, число графических ускорителей вычислений, тип коммуникационной среды), настраивая которые можно адаптировать систему под требования своей конкретной задачи. Еще одной причиной является используемая в традиционных системах модель вычислений, которая сталкивается с определенными трудностями при масштабировании многопроцессорных систем и программ для них.

Существуют различные варианты повышения реальной производительности многопроцессорных вычислительных систем на текущей элементной базе. Основные из них:

- адаптация вычислительных систем под конкретные задачи;
- переход к новым моделям вычислений и реализующим их архитектурам;
- эффективное распараллеливание вычислений.

В ИППМ РАН ведется работа над универсальной вычислительной системой, реализующей потоковую модель вычислений с динамически формируемым контекстом. В данной вычислительной системе в той или иной степени воплощены все вышеперечисленные варианты, а сама ее концепция предоставляет пользователям большое число параметров, которые можно настраивать исходя из особенностей решаемой задачи.

Данная статья посвящена аспектам определения оптимальной конфигурации вычислительной системы для решения конкретной задачи.

II. Потоковая модель вычислений и архитектура, реализующая ее

В основе классической потоковой модели вычислений (dataflow) лежит активация вычислений по готовности данных. В этом заключается ее кардинальное отличие от традиционной фон-неймановской модели вычислений. Традиционной модель вычислений называется так потому, что практически все современные микропроцессоры в настоящее время реализуют фон-неймановскую модель вычислений. Классические потоковые вычислительные системы начали разрабатываться в 80-90-х годах прошлого века [2-4]. Работы велись в США, Японии, Великобритании [5-7].

Несмотря на то, что некоторые из этих разработок были воплощены в жизнь, работа над dataflow-системами были свернуты [8]. Причинами этого стали недостаточное развитие элементной базы и фактическое отсутствие спроса на высокопараллельную вычислительную технику (в то время казалось, что возможности традиционных микропроцессоров и динамика их развития могут удовлетворить «любые» запросы пользователей). Тем не менее, в последнее время разработчики аппаратуры всё чаще обращаются к возможностям потоковой модели вычислений, проводятся конференции, публикуются статьи как по самой модели вычислений [9, 10], так и по аппаратуре, требуемой для эффективной ее реализации [11]. Ярким примером этого стало то, что в 2018 году фирма Intel, являющаяся апологетом

традиционной фон-неймановской модели вычислений, заявила о создании процессора (сопроцессора) с реальным dataflow управлением («really data flow engines») [12].

В нашей стране модель вычислений с управлением потоком данных также используют в своих работах некоторые научные коллективы [13-15].

В ИППМ РАН ведется работа над потоковой моделью вычислений с динамически формируемым контекстом [16, 17]. В общем виде модель вычислений можно представить в виде сопоставляющего устройства (рабочего пространства токенов) бесконечного размера, на вход которого приходят данные (токены, представляющие собой структуру данных, содержащую сам операнд и некоторый набор признаков), которые могут быть как внешними (начальными данными задачи), так и внутренними (образованными в результате выполнения задачи). Внутри сопоставляющего устройства происходит поиск данных с «совпадающими» признаками. При обнаружении таких данных активируется выполнение программы, связанной с этими данными, а сами данные удаляются из сопоставляющего устройства. Выполнение программы обработки данных осуществляется на вычислителе. В результате выполнения программы образуются новые данные, которые либо подаются на вход сопоставляющего устройства, либо выдаются в качестве результата выполнения программы.

Потоковая модель вычислений с динамически формируемым контекстом воплощена в языке параллельного программирования – DFL [18]. Программа на языке DFL представляет собой виртуальный граф вычислений в виде набора программных узлов, соединенных ребрами, которые обозначают пути обмена данными (токенами) между узлами согласно программе. Токен содержит операнд, ключ (контекст), который однозначно определяет положение операнда в виртуальном адресном пространстве задачи, и ряд служебных полей. Программный узел состоит из заголовка узла (включающего имя программного узла, список входов и контекст) и программного кода (набора операций, последовательно исполняемых на вычислителе). Активация конкретного экземпляра программного узла происходит только при условии, что на все его входы поступили токены с «одинаковым» именем узла и контекстом. Результатом активации программного узла является его исполнение на вычислителе, которое не прерывается на подкачку дополнительных данных, поскольку оперирует исключительно входными данными, контекстом и константами. Между собой программные узлы обмениваются только токенами, которые образуются как результат исполнения программного кода. При такой организации вычислительного процесса исключена вероятность его искажения и самоблокирования (при правильно составленной программе), так как программа узла обрабатывает только данные, поступившие к ней на вход. А поскольку отсутствует повторное использование цепей графа, т.е. данные в виртуальный узел поступают только один раз, что эквивалентно принципу, заложенному в языках однократного присваивания, то при параллельно выполняемой обработке данных исключается сама возможность использования «устаревших» данных.

Потоковая модель вычислений с динамически формируемым контекстом характеризуется тем, что атрибуты ключа токенов, формируемых в ходе выполнения программы узла, вычисляются непосредственно в этом же коде согласно программе узла. Это означает, что данная модель вычислений функционирует в парадигме «раздачи», когда тот, кто формирует новое значение, знает кому оно потребуется и обеспечивает его рассылку получателям. Традиционная модель вычислений функционирует в парадигме «сбора», при которой производитель новых данных ничего не знает об их потребителе.

Основными структурными единицами параллельной программы, на уровне которых можно говорить о параллелизме в потоковой модели вычислений, являются программные узлы. Каждый активированный экземпляр программного узла не зависит от других активированных экземпляров и тем самым может выполняться параллельно с ними.

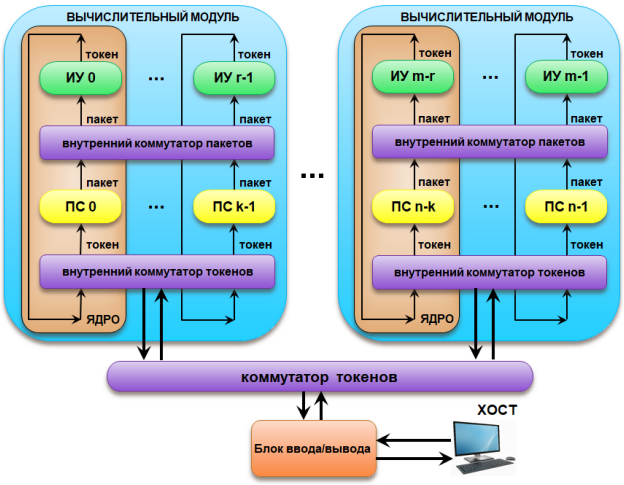

Рис. 1. Архитектура параллельной потоковой вычислительной системы

Потоковая модель вычислений реализована в архитектуре параллельной потоковой вычислительной системы (ППВС) «Буран» [19, 20]. ППВС представляет собой параллельную многопроцессорную вычислительную систему (рис. 1), состоящую из вычислительных модулей, объединенных общей коммуникационной сетью, и блока ввода/вывода. По коммуникационной сети передаются исключительно токены. Вычислительный модуль является процессором (в рамках одного кристалла), который может объединять несколько вычислительных ядер, связанных между собой локальным коммутатором токенов и локальным коммутатором пакетов. Коммутация между вычислительными ядрами выполняется по номеру ядра, которое вырабатывается хэш-функцией на основе контекста токена.

В состав вычислительного ядра входят процессор сопоставлений и исполнительное устройство. Поскольку аппаратно реализовать сопоставляющее

устройство большого размера не представляется возможным, то оно разбивается на множество параллельно работающих процессоров сопоставлений. Благодаря тому, что потоковая модель вычислений позволяет провести такое разбиение, предоставляется возможность неограниченного наращивания числа ядер и, тем самым, емкости рабочего пространства [21].

Процессор сопоставлений (ПС) выполняет функцию организатора вычислительного процесса. Ключевым узлом процессора сопоставлений является ассоциативная память, которая наиболее эффективным образом реализует механизм аппаратного сопоставления токенов. Ассоциативная память отличается от оперативной памяти, которая используется в традиционных вычислительных системах, адресацией по содержимому, возможностью записи на свободное место, а также одновременным просмотром всех ячеек в режиме «Поиск» [22].

Процессор сопоставлений из поступающих на его вход токенов формирует готовую к исполнению структуру данных – пакет, который поступает в локальный коммутатор пакетов, выполняющий роль аппаратного балансера. Локальный коммутатор пакетов направляет готовые к исполнению пакеты на любое из свободных исполнительных устройств вычислительного модуля.

Исполнительные устройства (ИУ) в архитектуре ППВС обезличены. Пакет всегда обрабатывается от начала и до конца на одном ИУ. Программа узла никогда не бывает статически приписана к какому-либо конкретному ИУ. ИУ является универсальным процессором традиционной архитектуры, способным произвести любое преобразование информации над входными данными в пределах доступных ему аппаратных ресурсов. Результатом обработки пакета является формирование новых токенов, которые передаются на локальный коммутатор токенов, где определяется, на какой вычислительный модуль направлен токен – внешний или внутренний.

Более подробно архитектура ППВС описана в работах [20, 23].

III. ПАРАМЕТРЫ ПОТОКОВОЙ ВЫЧИСЛИТЕЛЬНОЙ СИСТЕМЫ

Процесс разработки потоковой вычислительной системы, реализующей новую модель вычислений, связан с изучением поведения системы при решении различных задач, что сводится к анализу изменения результатов моделирования работы вычислительной системы во время решения задач при изменении значений множества параметров этой системы.

Для потоковых вычислительных систем, основанных на концептуально иной (а с учетом свойств и отличий от традиционной, можно даже сказать «противоположной») модели вычислений, правила поведения, присущие традиционным системам, не применимы. Для потоковых систем требуется определить свой набор параметров и изучить их влияние на поведение вычислительной системы.

Параметры условно можно разделить на две группы – глобальные и локальные. Изменение глобальных параметров оказывает влияние на всю вычислительную систему (включая её архитектуру), а изменение локальных – только на отдельные узлы и блоки системы.

К глобальным относятся следующие:

- число вычислительных модулей;
- число процессоров сопоставлений;
- число исполнительных устройств;
- топология коммуникационной среды;
- и др.

К локальным параметрам можно отнести:

- объем памяти процессора сопоставлений;
- число функциональных блоков в процессоре сопоставлений;
- число функциональных блоков в исполнительном устройстве;
- тип функции распределения вычислений;
- время выполнения операций функциональными блоками;
- задержки отдельных узлов и блоков;
- и др.

Опишем более подробно основные из них.

A. Число вычислительных модулей

Данный параметр отвечает за общее число вычислительных модулей (ВМ) в конкретной конфигурации вычислительной системы. Изменение данного параметра влияет на общее число процессоров сопоставлений и исполнительных устройств в системе, а также на структуру коммуникационной среды (к которой подключены вычислительные модули). Данный параметр оказывает существенное влияние на конечную стоимость вычислительной системы.

Параметр может принимать значение больше или равным единице. При этом рекомендуется, чтобы значение было равно степени числа два. Это связано как со структурой коммуникационной сети, так и с распределением вычислений по модулям, которые при такой размерности демонстрируют свою наибольшую эффективность. В противном случае, распределение данных может быть, как неравномерным, так и вызывать чрезмерный обмен данными между вычислительными модулями системы, что может привести:

- к увеличению времени выполнения задачи, вызванному как неравномерным распределением вычислительной нагрузки по имеющимся аппаратным ресурсам, так и очередями в коммуникационной среде;
- к блокировке вычислительного процесса (например, все токены будут направлены только в один вычислительный модуль).

Максимальное (эффективное для конкретной задачи) значение данного параметра зависит от:

- алгоритма задачи и реализующей ее программы;
- размерности входных данных;
- верхней границы стоимости вычислительной системы.

B. Число процессоров сопоставлений

Данный параметр отвечает за число процессоров сопоставлений, размещенных в одном вычислительном модуле. Как уже отмечалось – вычислительный модуль представляет собой кристалл, т.е. от степени интеграции микросхемы зависит число процессоров сопоставлений в одном вычислительном модуле. Верно и обратное утверждение – выяснив необходимое число ПС в одном вычислительном модуле, становится известна требуемая степень интеграции микросхемы. Общее число ПС в конфигурации вычислительной системы равно произведению значений параметров «Число вычислительных модулей» и «Число процессоров сопоставлений».

Увеличение числа ПС повышает параллелизм, но ведет к росту накладных расходов на коммуникации, а также к уменьшению объема памяти ключей/токенов в одном ПС.

Значение параметра больше или равно единице. Верхняя граница ограничена физическими возможностями размещения заданного числа ПС на кристалле. Также, как и в случае с параметром «Число вычислительных модулей» (по описанным выше причинам), желательно, чтобы значение этого параметра было равным степени двойки.

На практике, при проведении исследований верхняя граница параметра ограничивалась 16-ю, поскольку это максимальное число ПС, которое, учитывая его сложность, можно разместить на одном кристалле при существующих технологических нормах производства.

C. Число исполнительных устройств

Данный параметр отвечает за число исполнительных устройств, размещенных в одном вычислительном модуле.

Изменяя этот и предыдущий параметры появляется возможность моделировать и исследовать различную архитектуру вычислительного модуля от базовой (где одному процессору сопоставлений соответствует одно исполнительное устройство), стандартной (при которой на один ПС приходится больше одного ИУ), и до утилитарной (при которой в системе и вовсе отсутствуют ИУ). Последний вариант применяется при полной адаптации вычислительной системы под конкретную задачу, когда весь необходимый для ее решения функционал может быть реализован в процессоре сопоставлений через механизм специальных токенов [24].

Поскольку концепция потоковой модели вычислений с динамически формируемым контекстом и параллельного языка DFL предполагает, что программный узел относительно прост и небольшого размера, то для его выполнения достаточно простого однопоточного процессора без механизмов суперскаляра, мультитрединга (многопоточности) и т.п. Не последнюю роль в этом играет и внутренний коммутатор пакетов, выполняющий роль распределителя пакетов по свободным ИУ. Поэтому вместо одного сложного многопоточного процессора, можно выбрать несколько простых ИУ. Увеличение числа ИУ на один ПС преследует сразу несколько целей. Во-первых, это повышение темпа обработки пакетов (особенно если программа узла, обрабатывающая этот пакет, содержит несколько десятков команд), а, во-вторых, повышение темпа генерации новых токенов, возникающих в ходе обработки пакетов.

Изменение данного параметра оказывает влияние не только на время выполнения задачи, но и, в зависимости от алгоритма решения задачи, на требуемый размер ассоциативной памяти ПС.

D. Топология коммуникационной среды

Данный параметр определяет топологию коммутатора токенов, к которому подключены вычислительные модули. Выбор значения параметра осуществляется между такими топологиями как «Дерево», «Решетка», «3D-Куб» и «3D-Тор». Наиболее универсальной топологией по результатам проведенных исследований на задачах различных классов является «3D-Тор».

Этот параметр необходимо рассматривать в тесной взаимосвязи с параметром «Тип функции распределения вычислений», поскольку максимального эффекта от локализации вычислений (минимизации дальности и числа передач данных между вычислительными модулями) можно достичь только когда функция распределения (и её аргументы) учитывает особенности конкретной конфигурации коммутационной сети, которая формируется в зависимости от числа вычислительных модулей в системе.

E. Объем памяти процессора сопоставлений

Параметр определяет максимальное число токенов, которые могут находиться в ПС. Значение параметра напрямую влияет на объем памяти ключей, а объем памяти токенов определяется как произведение числа токенов на размер структуры токена (в байтах) и повышающий коэффициент M. Коэффициент M зависит от того, используется в алгоритме потоковой программы многовходовые и векторные токены или нет. Если они не используются, то соотношение объема памяти задается исходя из правила «один токен – одно данное». В случае использования многовходовых токенов – на одну запись в памяти ключей приходится $(T+N*8)$ байт в памяти токенов, где T – размер структуры токена без данного, а N – число входов (данных) программного узла, к которому относятся эти многовходовые токены.

Изменение значения параметра влияет на время прохождения задачи и на максимальное число ПС в вычис-

лительном модуле. Влияние на время прохождения задачи проявляется в том случае, если в процессе работы память ключей или память токенов окажутся переполнены и будет активирован механизм по откачке/подкачке данных. На откачку и на последующую подкачку ранее откаченных данных (после того как в процессе дальнейшей работы ПС память освободится) тратится время, что влияет и на общее время выполнения задачи.

Влияние на максимальное число ПС в вычислительном модуле обусловлено размерами и технологической нормой производства кристалла, в границах которого размещается вычислительный модуль. Т.е. чем больше значения этих параметров, тем больше объем доступной памяти в процессоре сопоставлений, и тем больше область, занимаемая одним процессором сопоставлений, на кристалле.

Нулевое значение для данного параметра указывает на бесконечную память ПС и используется для исследования прохождения задачи на программной модели, что позволяет получить данные о максимальном объеме памяти, необходимом для решения конкретной задачи, а также анализировать отношение этого объема к размеру начальных данных.

F. Число функциональных блоков в ИУ

Данный параметр позволяет задавать количество функциональных блоков (входящих в состав ALU, FPU и блока специальных операций) в исполнительном устройстве, вплоть до полного исключения из конфигурации системы отдельных функциональных блоков.

Изменение значения параметра для каждого типа функционального блока влияет не только на время выполнения программы узла за счет параллельной обработки однотипных операций (при их наличии в программном узле). Грамотный выбор комбинации функциональных блоков позволяет еще и сократить число исполнительных устройств в вычислительном модуле без увеличения времени выполнения задачи, что в конечном итоге позволяет повысить эффективность вычислительного модуля.

G. Тип функции распределения вычислений

Для эффективного функционирования любой многопроцессорной вычислительной системы необходимо обеспечение равномерной нагрузки на процессоры и коммуникационную сеть системы. ППВС в этом вопросе не является исключением. Распределение вычислений в ППВС обеспечивается при помощи функции распределения вычислений (хэш-функции). Данная функция для каждого токена (при его создании), циркулирующего в вычислительной системе, в зависимости от его ключа вырабатывает значение, которое определяет, в какой процессор сопоставлений он должен попасть при помощи коммуникационной сети.

В ППВС имеется возможность задавать свою функцию распределения вплоть до отдельного программного узла, что обеспечивает широкие возможности по настройке вычислительной системы под задачу [25].

Также в ППВС уже реализованы следующие универсальные функции распределения: Std, Field и Zip

Хэш-функция Std является универсальным инструментом, гарантирующим равномерное распределение вычислений для любой задачи. Ее недостатком является интенсивный обмен токенами между вычислительными модулями, создающий высокую коммуникационную нагрузку. Данная функция используется на стадии отладки параллельных программ.

Хэш-функция Field производит распределение вычислений по конкретному полю ключа токена. Эффективна при выполнении сеточных и итерационных задач, в которых присутствует явно выраженное поле ключа, на изменении которого и построен алгоритм.

Хэш-функция Zip осуществляет распределение вычислений сразу по нескольким полям ключа токена. Эффективна при выполнении многомерных сеточных задач, в которых несколько полей контекста задают координаты ячеек по измерениям, а обмены производятся в основном между ячейками с близкими координатами. Позволяет разбивать адресное пространство задачи на множество прямоугольных блоков одинакового размера. Распределение данных, при котором на каждый модуль приходится минимум один такой блок, существенно сокращает передачу данных между вычислительными модулями (а значит и общее время выполнения задачи) за счет локализации вычислений в границах блока.

В целом данный параметр оказывает существенное влияние на равномерность распределения вычислений в параллельной потоковой вычислительной системе, что влияет на общее время выполнения задачи.

Исследование поведения ППВС с конфигурацией, заданной конкретными значениями представленных параметров, проводится на программной блочно-регистровой модели (ПБРМ) [26]. Помимо указанных параметров, модель поддерживает и множество других, используемых для более «тонкой» настройки системы. ПБРМ позволяет оценить влияние всех этих параметров на характер работы конкретной конфигурации вычислительной системы.

IV. Критерии потоковой вычислительной системы

Результатом работы ПБРМ является набор показателей, включающий в себя временные диаграммы работы узлов и блоков, изменение различных количественных показателей во времени (например, количество токенов в памяти ключей ПС), изменение качественных показателей во времени (например, загруженность ИУ в процентах), а также общесистемные числовые значения (например, время работы системы в условных тактах).

Автоматизированное определения оптимальной конфигурации архитектуры ППВС для решения конкретной задачи опирается на набор критериев, позволя-

ющий оценить эффективность функционирования вычислительной системы. Все критерии можно разделить на две группы – промежуточные и итоговые.

A. Промежуточные критерии

Промежуточные критерии применяются в качестве средств оценки и настройки параметров (таких как, «Число процессоров сопоставлений», «Число исполнительных устройств», «Тип функции распределения вычислений» и др.) вычислительной системы. Непосредственного участия в выборе оптимальной конфигурации они не принимают.

1) Средняя загруженность исполнительных устройств

Значение критерия рассчитывается как сумма средней загруженности каждого отдельного ИУ, деленная на суммарное число ИУ в системе, и исчисляется в процентах. Средняя загруженность отдельного ИУ определяется по результатам работы ПБРМ как отношение числа тактов выполнения полезных действий ИУ к общему времени моделирования задачи. Данный критерий позволяет оценить эффективность использования вычислительных ресурсов исполняемой задачей. Чем выше величина данного критерия, тем лучше.

2) Максимальная загрузка памяти ключей

Значение критерия определяется как максимальное среди всех процессоров сопоставлений значение локального максимального числа токенов, находившихся в памяти ключей ПС во время прохождения задачи.

Критерий исчисляется в токенах и позволяет оценить размер памяти ключей и памяти токенов, необходимый для бесперебойного (без остановок на откачку/подкачку данных для предотвращения переполнения памяти) выполнения задачи. Чем меньше значение данного критерия, тем лучше.

Альтернативой этому критерию является средняя максимальная загрузка памяти ключей ПС, рассчитываемая как сумма максимальных загрузок каждого ПС, деленная на число ПС в вычислительной системе. Этот критерий может быть использован для оценки влияния алгоритма задачи на загрузку памяти – например, если при увеличении числа процессоров сопоставлений в N раз, средняя максимальная загрузка уменьшается менее чем в N раз, то следует обратить внимание на эффективность алгоритма решения задачи.

В целом же, альтернативный критерий является менее информативным для оценки степени эффективности конфигурации вычислительной системы при решении задачи.

3) Равномерность распределения данных

Распределение данных, поддержанное аппаратно, между вычислительными модулями это одна из ключевых особенностей потоковой модели вычислений. Данный критерий показывает степень равномерности этого распределения. Расчет значения осуществляется методом среднеквадратичного отклонения для числа токенов, поступивших на вход процессоров сопоставлений.

$$\sigma = \sqrt{\frac{1}{N} * \sum_{i=1}^{N} (MP_i - \overline{MP})^2}$$

$$\overline{MP} = \frac{1}{N} * \sum_{i=1}^{N} MP_i \ ,$$

где σ – среднеквадратичное отклонение от среднего числа токенов, поступивших на вход ПС, N – общее число ПС в системе, MP_i – число токенов, поступивших на вход i-го ПС, $\overline{MP}$ – среднее число токенов, поступивших на вход ПС.

Значение критерия интерпретируется следующим образом. Чем ближе оно к нулю, тем более равномерно поступление токенов на входы ПС, а, следовательно, и более равномерно распределение вычислений. При анализе критерия важно оценивать динамику изменения значения при регулировке таких параметров как «Число вычислительных модулей», «Число процессоров сопоставлений» и «Тип функции распределения вычислений». Так, если при изменении функции распределения, значение критерия увеличивается, то это говорит о том, что выбранная функция распределения является менее эффективной для решения конкретно этой задачи. Если же рост значения фиксируется при увеличении суммарного числа ПС, то это может говорить об ошибках в алгоритме задачи в части разделения адресного пространства задачи. Чем меньше величина данного критерия, тем лучше.

B. Итоговые критерии

Итоговые критерии применяются для оценки эффективности вычислительной системы, и именно на их основании происходит выбор оптимальной конфигурации вычислительной системы для решения конкретной задачи.

1) Время прохождения задачи

Один из ключевых критериев, определяющий время прохождения задачи, моделируемой на конкретной конфигурации вычислительной системы. Измеряется в условных тактах. Анализ данного критерия позволяет оценивать эффективность вычислительной системы и степень масштабируемости задачи – через сравнение с аналогичными показателями, полученными в результате моделирования конфигурации с другими значениями параметров.

2) Стоимость системы

Критерий используется для оценки стоимости вычислительной системы в абстрактных единицах. Реальный подсчет стоимости многопроцессорной вычислительной системы возможен только на этапе производства. Но даже в этом случае, детальный подсчет стоимости для десятков и сотен различных конфигураций вычислительной системы (при различных значениях параметров) сопряжен с целым рядом трудностей.

Для подсчета значения критерия каждому узлу или блоку присваивается некий коэффициент сложности, а итоговая стоимость рассчитывается как сумма произведений (коэффициент сложности на число устройств). Стоимость отдельного блока рассчитывается исходя из

сложности и количества его узлов, а стоимость системы – из стоимости и количества ее блоков.

Данный критерий в первую очередь является ограничителем роста сложности вычислительной системы, когда наблюдается диспропорция между увеличением ее стоимости и эффективностью работы (например, если при увеличении числа вычислительных модулей в два раза, время моделирования сократилось на 10%, а стоимость системы увеличилось на 90%). Чем меньше значение данного критерия, тем лучше.

3) Энергопотребление

Критерий используется для оценки потребляемой вычислительной системой энергии в условных единицах. Подсчет энергопотребления конфигурации вычислительной системы основывается на значениях энергопотребления отдельных узлов и блоков. Значение данного критерия учитывается при определении конфигурации вычислительного модуля.

4) Занимаемая площадь

Данный критерий применяется для оценки площади, требуемой для размещения и обслуживания конкретной конфигурации вычислительной системы. Значение критерия исчисляется в условных единицах и рассчитывается с помощью коэффициентов размера, присваиваемым отдельным узлам и блокам вычислительной системы, включая систему охлаждения.

V. Определение оптимальной конфигурации архитектуры ППВС

Определение оптимальной конфигурации для конкретной задачи в общем случае требует проведения множества экспериментов. Важной задачей является разработка алгоритма определения оптимальной конфигурации архитектуры ППВС с учетом подходов, которые позволяют снизить их суммарное число (рис. 2).

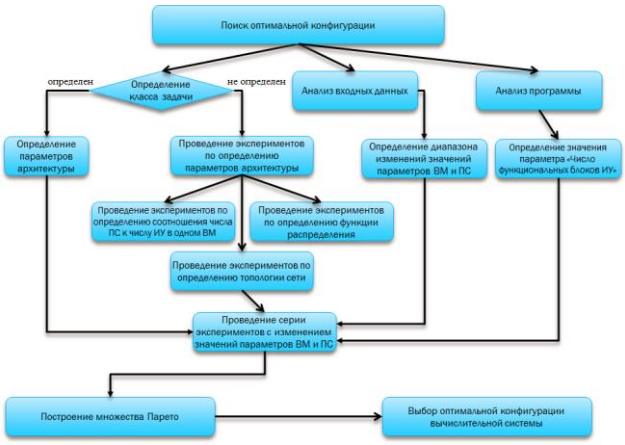

Рис. 2. Алгоритм определения оптимальной конфигурации ППВС для решения конкретной задачи

В ПБРМ [27] имеются ограничения (обусловленные разрядностью программной модели) на размер начальных данных и суммарное число ПС в моделируемой системе. Вследствие этого, первым этапом является выбор размерности задачи и начальных данных с учетом ограничений ПБРМ.

Такие параметры как «Тип коммуникационной среды», «Тип функции распределения» и соотношение числа ИУ к ПС в одном ВМ слабо взаимосвязаны с другими параметрами, что позволяет определить их значения с зафиксированными остальными параметрами, и, тем самым, сократить суммарное число проводимых экспериментов. Параметры «Тип коммуникационной среды» и «Тип функции распределения» в большей степени определяются классом задачи, т.е. зная класс задачи, становится возможным определить значения этих параметров. Класс задачи определяется разработчиком программы, при условии, что он обладает достаточной компетенцией и может задать этот класс явно. В противном случае, эти параметры определяются путем проведения исследований прохождения программы с различными значениями этих параметров. Определение оптимального значения параметра «Функция распределения вычислений» и его аргументов основывается на анализе промежуточных критериев «Равномерность распределения данных» и «Средняя загруженность исполнительных устройств». Критерий «Средняя загруженность исполнительных устройств» также применяется для определения параметра «Число исполнительных устройств». В настоящее время исследуется возможность использования нейронных сетей [28] для автоматического определения класса задачи по тексту реализующей её программы.

Параллельно осуществляется анализ начальных данных и анализ параллельной программы. Анализ начальных данных позволяет определить диапазон изменений значений параметров «Число вычислительных модулей» и «Число процессоров сопоставлений», т.е. верхнюю границу общего числа процессоров сопоставлений. Анализ параллельной программы определяет значения параметра «Число функциональных блоков ИУ», которые вычисляются исходя из максимального числа и типа независящих между собой (по данным) операций в каждой из программ узла. Эти значения вычисляются без проведения экспериментов.

Следующим этапом является определение значений параметров архитектуры, чье влияние на прохождение задачи планируется исследовать. С учетом размерности задачи определяется диапазон и шаг изменения значений этих параметров. Подготовленный таким образом шаблон эксперимента преобразуется в индивидуальные наборы для моделирования, где каждый набор – это конфигурация архитектуры с уже учтенными значениями параметров и списком начальных данных.

Ранее проведенные исследования различных конфигураций системы на ПБРМ позволили изучить поведение системы на задачах различных классов. Время прохождения задачи уменьшается с ростом числа вычислительных ядер. Однако начиная с некоторого значения числа ПС в системе, время прохождения задачи выходит на т.н. «плато», которое характеризуется тем, что увеличение числа ПС практически не приводит к уменьшению времени выполнения задачи. Это связано с недостатком данных для одного вычислительного ядра. Для поиска «плато» значение параметра «Число процес-

соров сопоставлений» приравнивается единице и изменяется значение параметра «Число вычислительных модулей». Таким образом, как только эксперименты показывают, что задача вышла на своё «плато», то это означает, что достигнут максимум суммарного числа ПС к текущей размерности задачи. Исходя из этих данных можно определить оптимальное соотношение размерности исходных данных к суммарному числу ПС в системе.

Далее, сгенерированные наборы передаются в ПБРМ, где выполняется моделирование поведения конкретной конфигурации архитектуры ППВС при выполнении программы. Результатом являются статистические данные по каждому из наборов, на основании которых рассчитываются значения критериев. После прохождения каждой из конфигураций системы имеется возможность автоматически проводить анализ такого критерия как «Время прохождения задачи» для определения точки выхода на «плато» и прекращения экспериментов с увеличением параметра «Число вычислительных модулей».

Значение параметра «Объем памяти процессора сопоставлений» при проведении экспериментов устанавливается равным бесконечности, что позволяет вычислить значение промежуточного критерия «Максимальная загрузка памяти ключей» для каждой из конфигураций вычислительной системы.

Следующим этапом проводится серия экспериментов для выяснения значения параметра «Число процессоров сопоставлений» в одном ВМ, оставляя неизменным общее число ПС в системе.

По результатам проведенной серии экспериментов вычисляются итоговые критерии для каждой конфигурации вычислительной системы. Необходимо найти минимум целевой функции, зависящей от итоговых критериев. Если была обнаружена конфигурация, совокупность итоговых критериев которой минимальна, то она и будет оптимальной. В противном случае на основании итоговых критериев строится пространство Парето, которое позволяет исключить из рассмотрения результаты с худшими комбинациями критериев.

Для определения пространства Парето строится график с множеством осей (по числу выбранных критериев), на нем располагаются все результаты моделирования с учетом рассчитанных значений итоговых критериев, формируя тем самым область каждой конфигурации. Итоговая область образуется наложением областей для каждого отдельного результата и тем самым формируется пространство, в котором располагаются результаты с худшими значениями критериев. Результатом является сокращенный список конфигураций вычислительной системы с указанием того какие значения параметров привели к каждой из них.

После определения оптимальной конфигурации проводится экстраполяция полученных данных в соответствии с реальной размерностью задачи, определенной пользователем.

VI. Заключение

Повышение реальной производительности микропроцессоров [29] и вычислительных систем является актуальной задачей. Один из вариантов решения этой проблемы – переход к новым моделям вычислений и реализующим их архитектурам. Одной из таких систем является параллельная потоковая вычислительная система [30-32]. Важной предпосылкой для достижения высокой реальной производительности является выбор оптимальной конфигурации вычислительной системы под задачу пользователя, причем этот выбор по возможности должен быть автоматизирован.

Автоматизированное определение оптимальной конфигурации параллельной потоковой вычислительной системы требует набора параметров, которые пользователь или разработчик может настраивать, а также критериев, на основании которых производится выбор оптимальной конфигурации системы.

Разработанный алгоритм поиска оптимальной конфигурации параллельной потоковой вычислительной системы под конкретную задачу позволяет сократить общее число проводимых экспериментов. Реализация данного алгоритма в программном комплексе проектирования потоковых систем позволит существенно снизить общее время исследования потоковой вычислительной системы, а также обеспечить оперативное реконфигурирование базовой архитектуры ППВС в спецвычислитель (с использованием модулярной арифметики в качестве функциональных устройств [33]) для решения конкретной задачи или группы задач.

В дальнейшем, важным является возможность реализовать часть блоков архитектуры ППВС в программируемом варианте в виде ПЛИС (разработке которых в нашей стране уделяется большое внимание [34-39]), что позволит критически важные для эффективности вычислений блоки настраивать непосредственно под алгоритм задачи на аппаратном уровне, а не использовать универсальные схемы.

Также на отдельных этапах реализации алгоритма рассматривается возможность использования нейросетей (которые сейчас активно применяются в различных областях науки и техники [40]) с целью выявления оптимальных значений параметров конфигурации вычислительной системы для решения задач различных классов.

Литература

[1] URL: https://top500.org/lists/hpcg/2021/06/ (дата обращения: 25.08.2021)

[2] Lee B., Hurson A.R. Dataflow Architectures and Multithreading // Computer. Aug. 1994. V. 27. № 8. P. 27-39. doi:10.1109/2.303620

[3] Nikhil R.S., Papadopoulos G.M., Arvind. T: a multithreaded massively parallel architecture / Proceedings of the 19th annual international symposium on Computer architecture (ISCA '92). ACM, New York, NY, USA, 1992. P. 156-167. doi:10.1145/139669.139715

[4] Arvind, Brobst S. The evolution of dataflow architectures: from static dataflow to P-RISC // International Journal of

High Speed Computing. 1993. V. 5. № 2. P. 125-153. doi:10.1142/S0129053393000074
[5] Gurd J.R., Kirkham C.C., Watson I. The Manchester Prototype Dataflow Computer // Communications of the ACM. Jan. 1987. V. 28. №1 P. 34-52. doi: 10.1145/2465.2468
[6] Grafe V.G., Hoch J.E. The Epsilon-2 Multiprocessor System // Journal of Parallel and Distributed Computing. Dec. 1990. V. 10. №4. P. 309-318. doi:10.1016/0743-7315(90)90032-K
[7] Yuba T., Shimada T., Yamaguchi Y., Hiraki K., Sakai S. Dataflow computer development in Japan / Proceedings of the 4th International Conference on Supercomputing (ICS `90). Amsterdam, The Netherlands, June 1990. P. 140-147. doi:10.1145/77726.255151
[8] Silc J., Robic B., Ungerer T. Asynchrony in parallel computing: From dataflow to multithreading // Parallel and Distributed Computing Practices. 1998. V. 1. № 1. P. 3- 30.
[9] Ertel S., Adam J., Castrillon J. Supporting Fine-grained Dataflow Parallelism in Big Data Systems / Proceedings of the 9th International Workshop on Programming Models and Applications for Multicores and Manycores (PMAM'18), Quan Chen, Zhiyi Huang, and Pavan Balaji (Eds.). ACM, New York, NY, USA. 2018. P. 41-50. doi:10.1145/3178442.3178447
[10] Nowatzki T., Gangadhar V., Ardalani N., Sankaralingam K. Stream-Dataflow Acceleration / Proceedings of the 44th Annual International Symposium on Computer Architecture (ISCA '17). ACM, New York, USA. 2017. P. 416-429. doi:10.1145/3079856.3080255
[11] Yantir H.E., Eltawil A.M., Kurdahi F.J. A Two-Dimensional Associative Processor // IEEE Transactions on Very Large Scale Integration (VLSI) Systems. Sept. 2018. V. 26. № 9. P. 1659-1670. doi:10.1109/TVLSI.2018.2827262
[12] URL: https://www.nextplatform.com/2018/08/30/intels-exascale-dataflow-engine-drops-x86-and-von-neuman/ (дата обращения: 25.08.2021)
[13] Хилько Д.В., Степченков Ю.А., Шикунов Ю.И., Орлов Г.А. Развитие средств капсульного программирования потоковой рекуррентной архитектуры // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС). 2018. Вып. 3. С. 2-9. doi:10.31114/2078-7707-2018-3-2-9
[14] Степченков Ю.А., Хилько Д.В., Шикунов Ю.И., Орлов Г.А. Специализированные преобразователи тегов для рекуррентного обработчика сигналов // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС). 2020. Вып. 2. С. 73-80. doi:10.31114/2078-7707-2020-2-73-80
[15] Левин И.И., Дордопуло А.И., Каляев И.А., Доронченко Ю.И., Раскладкин М.К. Современные и перспективные высокопроизводительные вычислительные системы с реконфигурируемой архитектурой // Вестник ЮУрГУ. Серия «Вычислительная математика и информатика». 2015. Том 4. №3. С. 24–39. doi:10.14529/cmse150303
[16] Левченко Н.Н., Окунев А.С., Стемпковский А.Л. Преимущества потоковой модели вычислений // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС). 2018. Вып. 3. С. 24-30. doi:10.31114/2078-7707-2018-3-24-30
[17] Zmejev D.N., A.V. Klimov, N.N. Levchenko. The Change of Computation Paradigm and Programming Model – The Future of New Supercomputers // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС). 2017. Вып. 2. С. 40-45.
[18] Климов А.В., Левченко Н.Н. Механизм ветвей в потоковом метаязыке UPL (METAL) и методы его реализации в ППВС «Буран» // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС). 2018. Вып. 3. С. 31-37. doi:10.31114/2078-7707-2018-3-31-37
[19] 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 // Журнал «ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ». 2008. №10. С. 2–7.
[20] 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. P. 292-298. doi:10.1134/S1063739719050111
[21] Змеев Д.Н., Левченко Н.Н., Окунев А.С., Стемпковский А.Л. Влияние особенностей модели вычислений и архитектуры на надежность параллельной потоковой вычислительной системы // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС). 2020. Вып. 1. С. 64-69. doi:10.31114/2078-7707-2020-1-64-69
[22] Levchenko N.N., Okunev A.S., Zmejev D.N., Klimov A.V., Stempkovsky A.L. Organization of Computations in Conditions of Limited Amount of Content Addressable Memory in the Parallel Dataflow Computing System "Buran" // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС).2019. Вып 2. С. 52-57. doi:10.31114/2078-7707-2019-2-52-57
[23] Бобков, С.Г., Левченко Н.Н., Окунев А.С. Параллельный потоковый процессор на новых архитектурных принципах для решения широкого круга задач // Наноиндустрия. 2020. Т. 13. № S5-2(102). С. 285-296. doi:10.22184/1993-8578.2020.13.5s.285.296
[24] Змеев Д.Н., Окунев А.С. Разработка и исследование алгоритма задачи перемножения разреженных матриц для параллельной потоковой вычислительной системы «Буран» // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС). 2018. Вып. 3. С. 16-23. doi:10.31114/2078-7707-2018-3-16-23
[25] Zmejev D.N., Klimov A.V., Levchenko N.N., Okunev A.S. Hash Unit as One of Computations Control Elements in Parallel Dataflow Computing System // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС). 2017. Вып.2. С. 46-51.
[26] Змеев Д.Н. Средства проектирования высокопроизводительных потоковых вычислительных систем // Проблемы разработки перспективных микро- и наноэлектронных систем - 2016. Сборник трудов / Под общ. ред. академика РАН А.Л. Стемпковского. М.: ИППМ РАН, 2016. Часть II. С. 159-163.
[27] Levchenko N.N., Okunev A.S., Zmejev D.N. Development Tools for High-Performance Computing Systems Using Associative Environment for Computing Process Organization / Proceedings of IEEE EAST-WEST DESIGN & TEST SYMPOSIUM (EWDTS'2016). Yerevan, Armenia, October 14-17, 2016. P. 359-362. doi:10.1109/EWDTS.2016.7807630
[28] Соловьев Р.А., Кустов А.Г., Рухлов В.С. Методика реализации нейронной сети для распознания рукописных цифр в FPGA на основе вычислений с фиксированной точкой // Проблемы разработки

перспективных микро- и наноэлектронных систем (МЭС). 2018. Вып. 3. С. 126-131. doi:10.31114/2078-7707-2018-3-126-131
[29] Бобков С.Г. Пути и методы повышения производительности микропроцессоров // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС). 2020. Вып. 4. С. 127-133. doi:10.31114/2078-7707-2020-4-127-133
[30] Климов А.В., Левченко Н.Н., Окунев А.С., Стемпковский А. Л. Суперкомпьютеры, иерархия памяти и потоковая модель вычислений // Программные системы: теория и приложения: электрон. научн. журн. 2014. Т. 5. № 1(19). С. 15–36.
[31] Змеев Д.Н., Климов А.В., Левченко Н.Н., Окунев А.С., Стемпковский А. Л. Потоковая модель вычислений как парадигма программирования будущего // Информатика и её применения. 2015. Т. 9. Выпуск 4. С. 29-36. doi:10.14357/1992264150403
[32] Левченко Н.Н., Окунев А.С., Стемпковский А.Л. 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. М.: ИППМ РАН. 2012. С. 459-462.
[33] Стемпковский А.Л., Амербаев В.М., Соловьев Р.А. Принципы рекурсивных модулярных вычислений // Информационные технологии. 2013. № 2. С. 22-27.
[34] Фролова П.И., Чочаев Р., Иванова Г.А., Гаврилов С.В. Алгоритм размещения с оптимизацией быстродействия на основе матриц задержек для реконфигурируемых систем на кристалле // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС). 2020. Вып. 1. С. 2-7. doi:10.31114/2078-7707-2020-1-2-7
[35] Рухлов В.С., Соловьев Р.А., Кустов А.Г. Программно-аппаратные решения повышения сбоеустойчивости комбинационных схем в базисе ПЛИС с учётом межсоединений и блоков ввода-вывода // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС). 2020. Вып. 1. С. 113-118. doi:10.31114/2078-7707-2020-1-113-118
[36] Михмель А.С., Мкртычан И.А., Тельпухов Д.В. Разработка моделей специальных логических элементов для анализа быстродействия реконфигурируемых систем на кристалле // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС). 2020. Вып. 3. С. 73-78. doi:10.31114/2078-7707-2020-3-73-78
[37] Заплетина М.А., Железников Д.А., Гаврилов С.В. Иерархический подход к трассировке реконфигурируемой системы на кристалле островного типа // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС). 2020. Вып. 3. С. 16-21. doi:10.31114/2078-7707-2020-3-16-21
[38] Гаврилов С.В., Хватов В.М., Железников Д.А., Гарбулина Т.В. Метод статического временного анализа с учетом трассировочных ресурсов для схем на базе реконфигурируемых систем на кристалле // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС). 2020. Вып. 2. С. 2-8. doi:10.31114/2078-7707-2020-2-2-8
[39] Жуков Д.В., Железников Д.А., Заплетина М.А. Применение SAT подхода к трассировке блоков коммутации для реконфигурируемых систем на кристалле // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС). 2020. Вып. 1. С. 26-32. doi:10.31114/2078-7707-2020-1-26-32
[40] Solovyev R.A., Stempkovsky A.L., Telpukhov D.V. Study of Fault Tolerance Methods for Hardware Implementations of Convolutional Neural Networks // Optical Memory and Neural Networksю 2019. V. 28. P. 82–88. doi:10.3103/S1060992X19020103

Automated Optimal Configuration Determination of the Parallel Dataflow Computing System for Solving a Specific Problem

A.L. Stempkovskiy, S.G. Bobkov, D.N. Zmejev, N.N. Levchenko, A.V. Klimov,

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

***Abstract* — One of the most important trends in modern multiprocessor computing is the increase in energy efficiency of computing, which is closely related to the high real performance of computing systems. One of the reasons for the low real performance is the small number of parameters (in traditional multiprocessor systems) provided to the user, adjusting which he can adapt the system to the requirements of his specific task. The article is devoted to the problem of determining the optimal configuration of a universal parallel dataflow computing system (PDCS) for solving a specific task. The article briefly describes the dataflow computation model and the original PDCS architecture that implements this model. The criteria by which it is proposed to evaluate the PDCS architecture are given; the parameters and their influence on these criteria are described as well. It is proposed to consider the parameters as global, the change of which affects the architecture of the computing system as a whole, and as local, which affect the behavior of individual nodes and blocks of the system. The global parameters are: the number of computational modules, the number of matching processors, the number of execution units, the type of distribution function of computations, and others. The local parameters are: the number of functional blocks in the execution unit, the memory size of the matching processor, the number of functional blocks in the matching processor, and others. The criteria are: the task execution time, the average workload of the execution units, the cost of the system, the uniformity of data distribution, and the**

maximum load of the memory of keys of the matching processor. An algorithm for the automated determination of the optimal configuration of a computing system for solving a specific task is presented. The algorithm is based on the described parameters and criteria, as well as the use of a behavioral block-register model. Plans for further improvement of the algorithm are outlined. The implementation of this algorithm in the software package for the design of dataflow computing systems will significantly reduce the total research time of the PDCS and ensure the operational transformation of the basic PDCS architecture into a special computer for solving a specific task or a group of tasks.

***Keywords* — parallel dataflow computing system, optimization parameters, performance criteria, dataflow computational model, optimal architecture.**

REFERENCES

[1] URL: https://top500.org/lists/hpcg/2021/06/ (access date: 25.08.2021)

[2] Lee B., Hurson A.R. Dataflow Architectures and Multithreading // Computer. Aug. 1994. V. 27. № 8. P. 27-39.

[3] Nikhil R.S., Papadopoulos G.M., Arvind. T: a multithreaded massively parallel architecture / Proceedings of the 19th annual international symposium on Computer architecture (ISCA '92). ACM, New York, NY, USA, 1992. P. 156-167.

[4] Arvind, Brobst S. The evolution of dataflow architectures: from static dataflow to P-RISC // International Journal of High Speed Computing. 1993. V. 5. № 2. P. 125-153.

[5] Gurd J.R., Kirkham C.C., Watson I. The Manchester Prototype Dataflow Computer // Communications of the ACM. Jan. 1987. V. 28. №1 P. 34-52.

[6] Grafe V.G., Hoch J.E. The Epsilon-2 Multiprocessor System // Journal of Parallel and Distributed Computing. Dec. 1990. V. 10. №4. P. 309-318.

[7] Yuba T., Shimada T., Yamaguchi Y., Hiraki K., Sakai S. Dataflow computer development in Japan / Proceedings of the 4th International Conference on Supercomputing (ICS `90). Amsterdam, The Netherlands, June 1990. P. 140-147.

[8] Silc J., Robic B., Ungerer T. Asynchrony in parallel computing: From dataflow to multithreading // Parallel and Distributed Computing Practices. 1998. V. 1. № 1. P. 3- 30.

[9] Ertel S., Adam J., Castrillon J. Supporting Fine-grained Dataflow Parallelism in Big Data Systems / Proceedings of the 9th International Workshop on Programming Models and Applications for Multicores and Manycores (PMAM'18), Quan Chen, Zhiyi Huang, and Pavan Balaji (Eds.). ACM, New York, NY, USA. 2018. P. 41-50.

[10] Nowatzki T., Gangadhar V., Ardalani N., Sankaralingam K. Stream-Dataflow Acceleration / Proceedings of the 44th Annual International Symposium on Computer Architecture (ISCA '17). ACM, New York, USA. 2017. P. 416-429.

[11] Yantir H.E., Eltawil A.M., Kurdahi F.J. A Two-Dimensional Associative Processor // IEEE Transactions on Very Large Scale Integration (VLSI) Systems. Sept. 2018. V. 26. № 9. P. 1659-1670.

[12] URL: https://www.nextplatform.com/2018/08/30/intels-exascale-dataflow-engine-drops-x86-and-von-neuman/ (access date: 25.08.2021)

[13] Khilko D.V., Stepchenkov Yu.A., Shikunov Yu.I., Orlov G.A. Development of Capsule Programming Means for Recurrent Data-flow Architecture // Problems of Perspective Micro- and Nanoelectronic Systems Development - 2018. Issue 3. P. 2-9. doi:10.31114/2078-7707-2018-3-2-9

[14] Stepchenkov Yu.A., Khilko D.V., Shikunov Yu.I., Orlov G.A. Specialized tag transformer for recurrent signal processor // Problems of Perspective Micro- and Nanoelectronic Systems Development - 2020. Issue 2. P. 73-80. doi:10.31114/2078-7707-2020-2-73-80

[15] Levin I.I., Dordopulo A.I., Kaljaev I.A., Doronchenko Ju.I., Raskladkin M.K. Sovremennye i perspektivnye vysokoproizvoditel'nye vychislitel'nye sistemy s rekonfiguriruemoj arhitekturoj (Modern and next-generation high-performance computer systems with reconfigurable architecture) // Vestnik JuUrGU. Serija «Vychislitel'naja matematika i informatika». 2015. Tom 4. №3. S. 24–39.

[16] 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

[17] Zmejev D.N., A.V. Klimov, N.N. Levchenko. The Change of Computation Paradigm and Programming Model – The Future of New Supercomputers // Problems of Perspective Micro- and Nanoelectronic Systems Development. 2017. Issue 2. P. 40-45.

[18] Klimov A.V., Levchenko N.N. Branches in the Dataflow Metalanguage UPL (METAL) and Methods of their Implementation in the PDCS “Buran” // Problems of Perspective Micro- and Nanoelectronic Systems Development - 2018. Issue 3. P. 31-37. doi:10.31114/2078-7707-2018-3-31-37

[19]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 (Dataflow computing system - further development of the architecture and structural organization of the computing system with automatic resource allocation) // Zhurnal «INFORMACIONNYE TEHNOLOGII». 2008. №10. S. 2–7.

[20] 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. P. 292-298.

[21] Zmejev D.N., Levchenko N.N., Okunev A.S., Stempkovsky A.L. Impact of features of the computing model and architecture on the reliability of the parallel dataflow computing system // Problems of Perspective Micro- and Nanoelectronic Systems Development - 2020. Issue 1. P. 64-69. doi:10.31114/2078-7707-2020-1-64-69

[22] Levchenko N.N., Okunev A.S., Zmejev D.N., Klimov A.V., Stempkovsky A.L. Organization of Computations in Conditions of Limited Amount of Content Addressable Memory in the Parallel Dataflow Computing System "Buran" // Problems of Perspective Micro- and Nanoelectronic Systems Development – 2019. Issue 2. Pp. 52-57.

[23] Bobkov, S.G., Levchenko N.N., Okunev A.S. Parallel'nyj potokovyj processor na novyh arhitekturnyh principah dlja reshenija shirokogo kruga zadach (Parallel dataflow processor based on new architectural principles for solving a wide range of tasks) // Nanoindustrija. 2020. T. 13. № S5-2(102). S. 285-296.

[24] Zmejev D.N., Okunev A.S. Development and Investigation of Algorithm of Sparse Matrices Multiplication Task for the Parallel Dataflow Computing System "Buran" // Problems of Perspective Micro- and Nanoelectronic Systems

Development - 2018. Issue 3. P. 16-23. doi:10.31114/2078-7707-2018-3-16-23

[25] Zmejev D.N., Klimov A.V., Levchenko N.N., Okunev A.S. Hash Unit as One of Computations Control Elements in Parallel Dataflow Computing System // Problems of Perspective Micro- and Nanoelectronic Systems Development . 2017. Issue II. P. 46-51.

[26] Zmejev D.N. Design tools of high-performance dataflow computing systems // Problems of Perspective Micro- and Nanoelectronic Systems Development - 2016. Proceedings / edited by A. Stempkovsky, Moscow, IPPM RAS, 2016. Part 2. P. 159-163.

[27] Levchenko N.N., Okunev A.S., Zmejev D.N. Development Tools for High-Performance Computing Systems Using Associative Environment for Computing Process Organization / Proceedings of IEEE EAST-WEST DESIGN & TEST SYMPOSIUM (EWDTS'2016). Yerevan, Armenia, October 14-17, 2016. P. 359-362.

[28] Solovyev R.A., Kustov A.G., Rukhlov V.S. The Technique for Implementing a Neural Network for Recognizing Handwritten Digits in FPGAs Based on Fixed Point Calculations // Problems of Perspective Micro- and Nanoelectronic Systems Development - 2018. Issue 3. P. 126-131. doi:10.31114/2078-7707-2018-3-126-131

[29] Bobkov S.G. Directions and methods for improving the performance of microprocessors // Problems of Perspective Micro- and Nanoelectronic Systems Development - 2020. Issue 4. P. 127-133. doi:10.31114/2078-7707-2020-4-127-133

[30] Klimov A.V., Levchenko N.N., Okunev A.S., Stempkovskij A.L. Superkomp'jutery, ierarhija pamjati i potokovaja model' vychislenij (Supercomputers, memory hierarchy and dataflow computation model) // Programmnye sistemy: teorija i prilozhenija: jelektron. nauchn. zhurn. 2014. T. 5. № 1(19). S. 15–36.

[31] Zmeev D.N., Klimov A.V., Levchenko N.N., Okunev A.S., Stempkovskij A.L. Potokovaja model' vychislenij kak paradigma programmirovanija budushhego (Dataflow computing model as a paradigm of future mainstream of software development) // Informatika i ejo primenenija. 2015. T. 9. Vypusk 4. S. 29-36.

[32] Levchenko N.N., Okunev A.S., Stempkovsky A.L. The usage of dataflow computing model and architecture realizing these for exaflops performance system // Problems of Perspective Micro- and Nanoelectronic Systems Development - 2012. Proceedings / edited by A. Stempkovsky, Moscow, IPPM RAS, 2012. P. 459-462.

[33] Stempkovskij A.L., Amerbaev V.M., Solov'ev R.A. Principy rekursivnyh moduljarnyh vychislenij (Principles of recursive modular computing) // Informacionnye tehnologii. 2013. № 2. S. 22-27.

[34] Frolova P.I., Chochaev R., Ivanova G.A., Gavrilov S.V. Timing-driven placement algorithm based on delay matrix model for reconfigurable system-on-chip // Problems of Perspective Micro- and Nanoelectronic Systems Development - 2020. Issue 1. P. 2-7. doi:10.31114/2078-7707-2020-1-2-7

[35] Rukhlov V.S., Solovyev R.A., Kustov A.G. Hardware and software solutions to increase the reliability of combinational logic in the FPGA basis without taking into account interconnections and the I/O blocks // Problems of Perspective Micro- and Nanoelectronic Systems Development - 2020. Issue 1. P. 113-118. doi:10.31114/2078-7707-2020-1-113-118

[36] Mikhmel A.S., Mkrtchan I.A., Telpukhov D.V. Development of Special Logic Element Models for Timing Analysis of Reconfigurable System-on-a-Chip // Problems of Perspective Micro- and Nanoelectronic Systems Development - 2020. Issue 3. P. 73-78. doi:10.31114/2078-7707-2020-3-73-78

[37] Zapletina M.A., Zheleznikov D.A., Gavrilov S.V. The Hierarchical Approach to Island Style Reconfigurable System-on-a-chip Routing // Problems of Perspective Micro- and Nanoelectronic Systems Development - 2020. Issue 3. P. 16-21. doi:10.31114/2078-7707-2020-3-16-21

[38] Gavrilov S.V., Khvatov V.M., Zheleznikov D.A., Garbulina T.V. Static Timing Analysis Method with Routing Resources Estimation for Reconfigurable System-on-a-Chip // Problems of Perspective Micro- and Nanoelectronic Systems Development - 2020. Issue 2. P. 2-8. doi:10.31114/2078-7707-2020-2-2-8

[39] Zhukov D.V., Zheleznikov D.A., Zapletina M.A. Application of SAT Approach to Switch Blocks Routing for Reconfigurable System-on-a-chip // Problems of Perspective Micro- and Nanoelectronic Systems Development - 2020. Issue 1. P. 26-32. doi:10.31114/2078-7707-2020-1-26-32

[40] Solovyev R.A., Stempkovsky A.L., Telpukhov D.V. Study of Fault Tolerance Methods for Hardware Implementations of Convolutional Neural Networks // Optical Memory and Neural Networksю 2019. V. 28. P. 82–88.