УДК 004.27

Специализация архитектуры многоядерной параллельной потоковой вычислительной системы для решения задачи быстрого преобразования Фурье

Н.Н. Левченко, А.С. Окунев

Учреждение Российской академии наук Институт проблем проектирования в микроэлектронике РАН, nick@ippm.ru, oku@ippm.ru

***Аннотация* — Архитектура параллельной потоковой вычислительной системы позволяет создавать специализированные аппаратные решения под конкретные задачи. В статье приводятся сравнительные результаты исследования варианта решения задачи быстрого преобразования Фурье на модели адаптированного под задачу многоядерного потокового процессора. Приведены оценки эффективности прохождения задачи при различных конфигурациях вычислительной системы, в том числе при использовании новых типов токенов и переносе вычислений в ассоциативную память.**

***Ключевые слова* — параллельная потоковая вычислительная система, быстрое преобразование Фурье, специализация архитектуры, управление данными.**

I. Введение

Одним из основных методов увеличения производительности вычислительных средств является метод распараллеливания вычислительных процессов. Вычислительная производительность одного ядра на используемой элементной базе близка к пределу. В связи с этим, на аппаратном уровне основным методом повышения производительности вычислительных систем в настоящее время является увеличение числа процессоров в системе. Однако, существенным фактом является то обстоятельство, что для большинства актуальных задач с увеличением числа процессоров вычислительной системы падает реальная производительность каждого процессора, особенно, если в задаче присутствует активная работа с глобальными данными. Для разного рода задач это падение различно и зависит от возможности локализации данных, с которыми она работает, что подтверждается результатами, полученными на тестовых пакетах измерения производительности. К основным причинам падения реальной производительности многопроцессорных вычислительных систем, можно отнести следующие:

- высокие накладные расходы на организацию параллельных вычислений, пропорциональные числу процессоров в системе;
- низкую пропускную способность подсистем оперативной памяти на реальных задачах (в случае нелинейной организации данных);
- необходимость синхронизации вычислительных процессов по данным.

Вследствие вышесказанного, представляет несомненный интерес разработка принципов построения вычислительной системы, архитектура которой основывается на нетрадиционной модели вычислений, отличной от общепринятой – фон-неймановской.

Работы по созданию систем, использующих принцип "потока данных" интенсивно проводились в Великобритании, США и Японии в конце 80-х и в 90-х годах прошлого века. Однако, в этих разработках не был решен целый ряд принципиальных вопросов, сдерживающих развитие данного направления. Можно назвать современные проекты, в модели вычислений которых используется принцип потока данных. Это проект WaveScalar [1] (макет создан в январе 2006 года в Вашингтонском университете). Еще один проект – вычислительная система TRIPS [2].

Настоящая работа имеет целью описание перспективной архитектуры, позволяющей достичь высокой реальной производительности на актуальных задачах, и продемонстрировать возможность адаптации такой архитектуры под конкретную задачу – на примере задачи быстрого преобразования Фурье (БПФ), а также показать возможность сохранения высокой реальной производительности при масштабировании параллельной потоковой вычислительной системы (ППВС), реализующей такую архитектуру.

II. Описание архитектуры ППВС

A. Гибридная модель вычислений

Активизация вычислительных квантов в модели вычислений с управлением потоком данных осуществляется по готовности данных - токенов. Под вычислительным квантом понимается небольшая программа (исполняемая в традиционной парадигме), которая выполняется без привлечения дополнительной информации, то есть все необходимые данные готовы на момент старта исполнения. Различные вычислительные кванты между собой не взаимодействуют.

При пересылке токена из одного вычислительного ядра в другое определяется адрес получателя (фактически, это часть ключа, хранящаяся в одном из полей токена), кроме того, в токен записывается значение операнда (данное). Этим наша модель вычислений отличается от традиционного подхода, в рамках которого вычислительный процесс запрашивает нужные ему данные.

Таким образом, в ППВС к моменту активации кванта вычислений все требуемые данные находятся при нем локально, что позволяет избавиться от длительных задержек ожидания активными процессами запрошенных данных из памяти (в отличие от традиционной модели вычислений).

Единицами информации, обрабатываемыми в вычислительной системе, являются токен и пакет. Токены представляют собой структуру, которая содержит передаваемый операнд, ключ, а также служебные признаки и атрибуты. Пакет представляет собой структуру, содержащую операнды, готовые к обработке на исполнительных устройствах системы.

В токене поле ключа содержит такую информацию, которая однозначно определяет положение токена или пакета в виртуальном адресном пространстве задачи.

Архитектура ППВС (рис. 1) представляет собой многоядерную систему, вычислительные ядра которой объединены коммуникационной сетью.

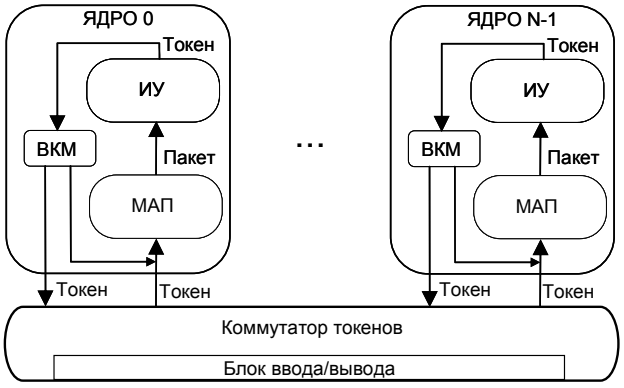

Рис. 1. Структурная схема ППВС (МАП – модуль ассоциативной памяти, ИУ – исполнительное устройство)

Если говорить о принципах работы, нужно отметить, что описание параллельного алгоритма выполняемой задачи состоит из набора вычислительных узлов. Узлы соответствуют вершинам потокового графа программы. Каждый узел – это набор команд (программа), выполняемых над операндами, поступающими в пакете.

Токены, формируемые в исполнительном устройстве, содержат адрес вычислительного ядра, в которое они должны поступить и где будет выполняться соответствующая программа над операндами.

Б. Функциональная схема ядра

Вычислительное ядро системы состоит из следующих блоков (рис. 2) – модуля ассоциативной памяти (МАП), исполнительного устройства (ИУ), блока хеш-функции локализации вычислений (БХЛ), блока хеш-функции формирования подмножества (БХП), внутреннего коммутатора (ВКМ) и блока регулирования параллелизма (БРП). Память откачки/подкачки, основная память и программная память не входят в структуру ядра.

Работа системы начинается с инициализации памяти команд ИУ. Затем осуществляется запуск задачи, который начинается с поступления токенов в коммутатор токенов, и далее на вход соответствующего вычислительного ядра. ВКМ позволяет принимать токены как из коммуникационной среды, так и из ИУ данного ядра, из него токены поступают на вход БХП. В БХП происходит формирование номера подмножества токенов задачи, что используется при регулировании параллелизма вычислений.

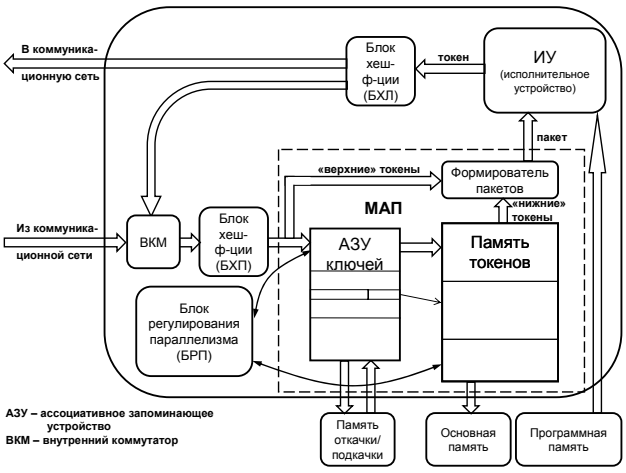

Рис. 2. Структурная схема базового вычислительного ядра

В модуле АП происходит сопоставление токенов по ключу и маске. Если нужный для сопоставления операнд на момент прихода токена отсутствует, то токен будет ожидать в АП поступления токена с соответствующим ключом и маской. При «совпадении» ключей токена происходит формирование пакета, который выдается на исполнение в ИУ.

Блок регулирования параллелизма позволяет задавать необходимый уровень параллельности вычислений, который зависит как от аппаратуры (размер памяти ключей, число ядер в системе и т.п.), входных данных, так и от свойств самого алгоритма.

Исполнительное устройство выполняет программу над операндами, поступающими в пакете. Результатом выполнения такой программы является генерация новых токенов, которые через БХЛ поступают либо

на вход ВКМ, либо в коммуникационную среду для передачи на другие вычислительные ядра или выдачи на хост.

B. Масштабируемость архитектуры ППВС

Поскольку архитектура ППВС реализует модель вычислений с управлением потоком данных с динамически формируемым контекстом, то распределение вычислений по вычислительным ядрам системы не привязано непосредственно к программе, то есть программа будет работать и на одном ядре, и на произвольном числе ядер без повторной компиляции или изменения программы.

Таким образом, архитектура ППВС масштабируема, и при увеличении числа ядер в системе падение реальной производительности на задачах со сложноорганизованными 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, из-за того, что активируемому узлу для своего полного выполнения не требуется дополнительных данных; в-третьих, тем, что узлы выполняются независимо друг от друга и, в-четвертых, благодаря правильному выбору хеш-функции (функции распределения вычислений по группам ядер), сокращающей число межъядерных передач токенов.

III. Реализация алгоритма БПФ на ППВС с использованием специализированного вычислительного ядра

Одним из наиболее распространенных алгоритмов БПФ является алгоритм, построенный по методу Cooley-Tukey (Кули-Таки) [3], который обладает рядом хороших технологических свойств. Структура алгоритма и его базовые операции не зависят от числа отсчетов (меняется только число прогонов базовой операции "бабочка"). Алгоритм легко каскадируется, поскольку коэффициенты БПФ для 2*N* отсчетов могут быть определены преобразованием коэффициентов двух БПФ по *N* отсчетов, полученных "прореживанием" исходных 2*N* отсчетов.

Прореживание по частоте разбивает компоненты входного вектора на два подмножества, содержащие соответственно первые *N*/2 компонент и вторые *N*/2 компонент.

Прореживание по времени разбивает множество компонент входного вектора на два подмножества: множество компонент с четными индексами и множество компонент с нечетными индексами. Множество компонент выходного вектора разбивается при этом на множество первых *N*/2 компонент и множество вторых *N*/2 компонент.

Основным компонентом БПФ алгоритма Кули-Таки по основанию 2 является операция «бабочка». Алгоритм Кули-Таки по основанию 2 был выбран в связи с тем, что в базовой архитектуре вычислительного ядра пакеты образуются в результате сопоставления двух токенов.

Перед началом вычислений (в алгоритме БПФ, основанном на прореживании по времени) производится перестановка элементов последовательности таким образом, что новым номером элемента в последовательности становится зеркальное отражение битовой записи его старого номера. Все поворачивающие множители вычисляются заранее и хранятся в памяти констант исполнительного устройства. В отличие от стандартной (универсальной) структуры ядра, при специализации – «аппаратной» реализации алгоритма БПФ – на поведенческой модели в структуру ядра и систему команд были внесены следующие изменения:

- в систему команд ассоциативной памяти добавлен новый тип токена – «токен БПФ»;
- в модуль ассоциативной памяти добавлен блок быстрого преобразования Фурье, который выполняет операцию «бабочка», формирует токены для следующих итераций, а также отслеживает момент окончания программы (по анализу одного из полей токена);
- создана хеш-функция (функция распределения вычислений), предназначенная для уменьшения числа пересылок между ядрами вычислительной системы;
- исключен блок исполнительного устройства.

Старт выполнения задачи БПФ осуществляется путем посылки токенов с инвертированными индексами и кодом операции «БПФ» в токене.

Стоит отметить, что, наряду с вариантом исключения блока ИУ из состава ядра, имеется возможность организации параллельной работы специализированного ядра с универсальным ИУ. Это позволит в многозадачном режиме повысить эффективность ядра за счет параллельного функционирования универсального ИУ и функционального устройства включенного в состав модуля АП.

Важным является тот факт, что экстрагирование неявного параллелизма из программы происходит на разных этапах выполнения (по мере готовности операндов), при этом разные этапы (итерации) могут выполняться одновременно.

IV. Результаты проведенных экспериментов

Исследования быстродействия алгоритма БПФ проводились на инструментальном комплексе базовой архитектуры ППВС, в состав которого входит поведенческая блочно-регистровая модель, параллельный язык высокого уровня, ассемблер.

На рис. 3 приведен график зависимости коэффициента ускорения работы вычислительной системы от числа вычислительных ядер в конфигурации системы. Под коэффициентом ускорения понимается отношение числа тактов выполнения программы на одном ядре ко времени выполнения программы на многих ядрах. Коэффициент реального ускорения на ППВС незначительно отличается от идеального ускорения. Причем на данной задаче, судя по полученным результатам, имеется тенденция и к дальнейшему увеличению коэффициента ускорения при увеличении числа вычислительных ядер в конфигурации ППВС.

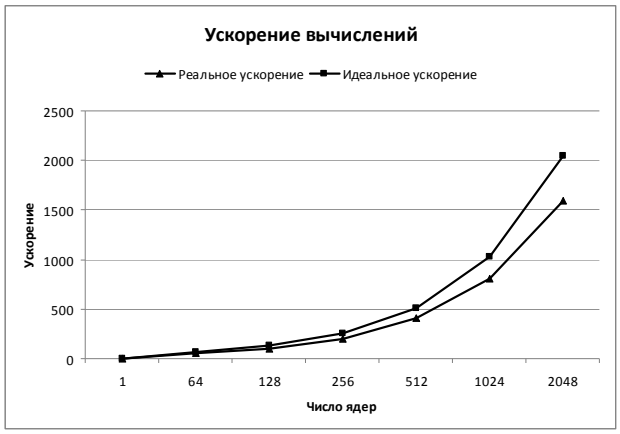

Рис. 3. Зависимость коэффициента ускорения от числа вычислительных ядер в конфигурации системы

На рис. 4 приведена зависимость времени выполнения программы БПФ от числа вычислительных ядер при использовании универсального исполнительного устройства и базового модуля памяти сопоставлений.

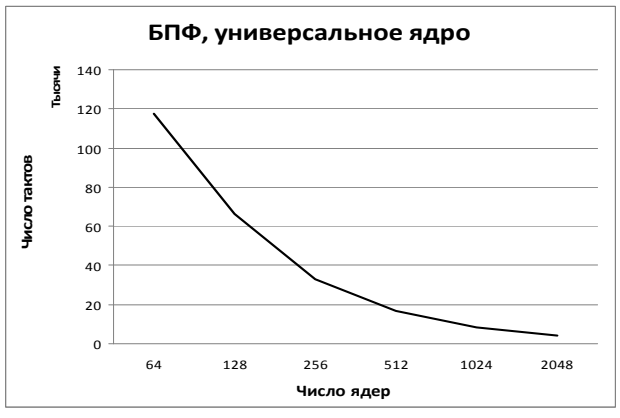

Рис. 4. Зависимость времени выполнения программы БПФ от числа вычислительных ядер при использовании универсального ядра

На рис. 5 приведена зависимость выполнения программы БПФ на ППВС с использованием специализированного вычислительного ядра. Сравнение результатов прохождения задачи на специализированном и универсальном ядрах показало, что при конфигурации вычислительной системы в 2048 ядер, задача выполняется на специализированном ядре примерно за 400 тактов, что на порядок быстрее по сравнению с использованием универсального ядра. Такое ускорение достигается за счет уменьшения накладных расходов на организацию вычислений.

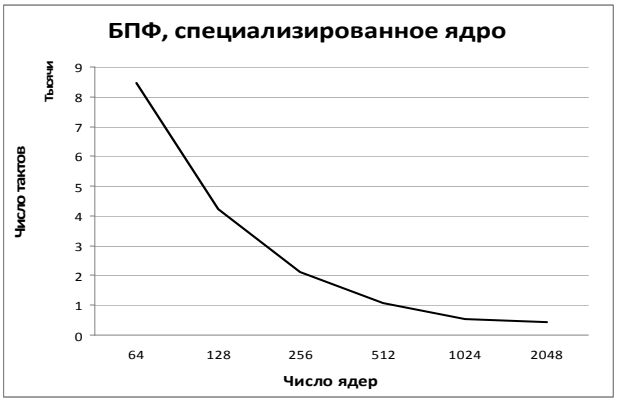

Рис. 5. Зависимость времени выполнения программы БПФ от числа вычислительных ядер при использовании специализированного вычислительного ядра

V. Заключение

Исследована архитектура параллельной потоковой вычислительной системы, на базе которой имеется возможность создавать специализированные решения под конкретные задачи. В качестве примера приведен вариант специализации ППВС для решения задачи БПФ, а также результаты сравнения ее прохождения на поведенческой модели, как на универсальной архитектуре, так и на специализированной. Показано, что такая специализация позволяет существенно сократить время выполнения задачи.

Применение такого подхода, основанного на использовании архитектуры ППВС, позволит создавать специализированные однокристальные системы, которые имеют высокую производительность.

Литература

[1] Steven Swanson and others. The WaveScalar architecture // ACM Transactions on Computer Systems (TOCS) Volume 25 Issue 2, May 2007.

[2] Mark Gebhart and others. An Evaluation of the TRIPS Computer System // Proceeding of the 14th international conference on Architectural support for programming languages and operating systems, Washington, DC, USA — March 07 - 11, 2009, Pages 1-12.

[3] Блейхут Р. Быстрые алгоритмы цифровой обработки сигналов: Пер. с англ. – М.: Мир, 1989. – 448 с.