ВОПРОСЫ ПРОГРАММИРУЕМОСТИ МНОГОЯДЕРНОЙ ВЫЧИСЛИТЕЛЬНОЙ АРХИТЕКТУРЫ С ЕДИНЫМ ПОТОКОМ ДЛЯ ЭФФЕКТИВНОЙ РЕАЛИЗАЦИИ РЕКУРРЕНТНЫХ ВЫЧИСЛЕНИЙ

Д.В. Хилько, Ю.А. Степченков

Учреждение российской академии наук Институт Проблем Информатики Российской Академии Наук *(*ИПИ РАН*)*

Аннотация: В статье рассматриваются ключевые аспекты нового подхода к организации архитектур потоковых систем. Представлена модель реализации новой архитектуры, рассмотрены проблемы её программирования. Определены направления разработки специализированных средств программирования новой архитектуры.

Ключевые слова: многоядерность, рекуррентность, параллелизм, потоковая архитектура

Введение

В ИПИ РАН ведутся работы по созданию нетрадиционной рекуррентной архитектуры, предназначенной для реализации параллельных вычислений ограниченной размерности в области сигнальной обработки. Для экспериментальной апробации предлагаемой архитектуры разрабатывается рекуррентный обработчик сигналов (РОС), исполняемый в гибридном, двухуровневом варианте с ведущим фон-неймановским процессором на управляющем (верхнем) уровне (УУ) и рядом потоковых процессоров на нижнем уровне – рекуррентном операционном устройстве (РОУ)[1].

Архитектура РОУ радикально отличается по основным моментам не только от классической архитектуры фон Неймана, но и от других нетрадиционных параллельных архитектур. Для существующих традиционных и нетрадиционных компьютерных архитектур характерно наличие двух потоков: активного потока инструкций и пассивного потока данных. В РОУ оба потока сливаются в один общий поток самодостаточных

данных, в котором, помимо собственно обрабатываемых данных, содержится и необходимая для их обработки управляющая и служебная информация (в существующих архитектурах кодируемая в форме инструкций).

Наглядное представление о сравнительных качествах рассматриваемых архитектур дает их сопоставление [2]:

- по организации памяти: *Control-Flow/Static (CF/S), Data-Flow/Static (DF/S), Data-Flow/Dynamic (DF/D)* (рис. 1);
- по количеству шагов, необходимых для выполнения инструкции (рис. 2).

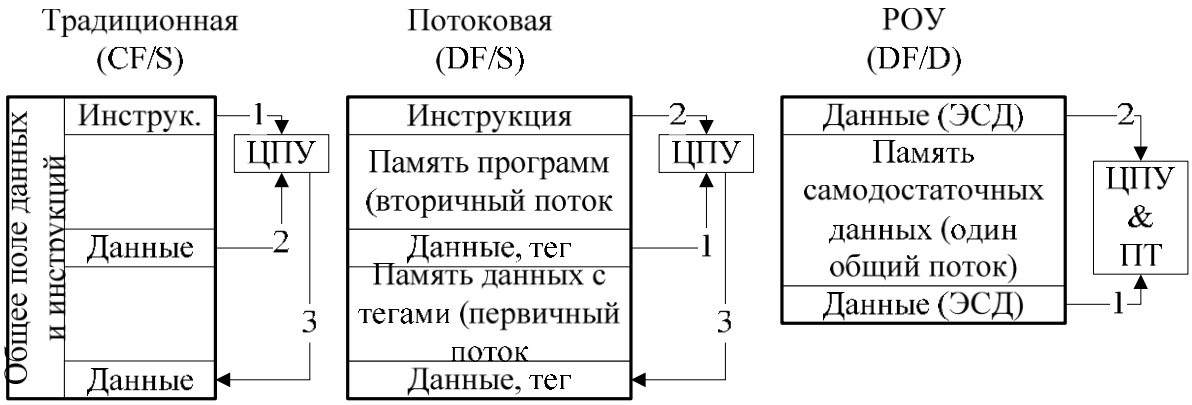

Рисунок 1. Принципиальные отличия сравниваемых архитектур.

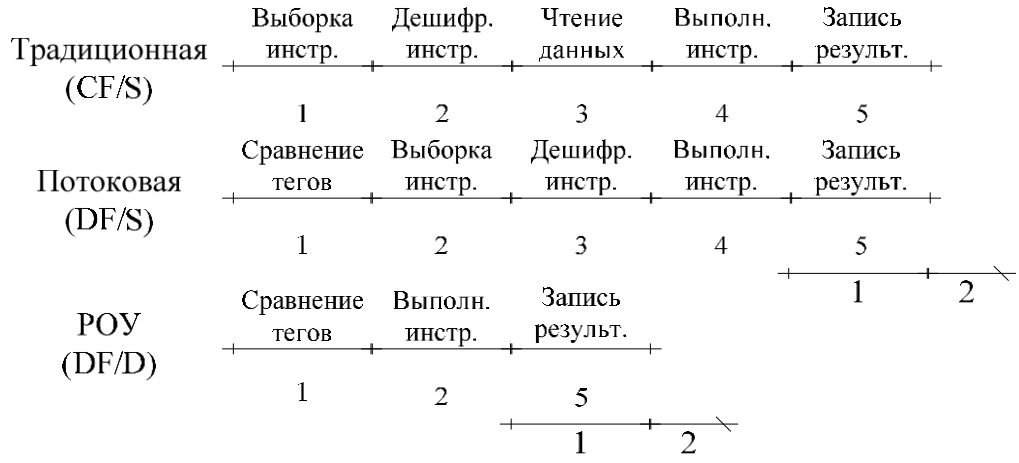

Рисунок 2. Выполнение инструкций в сравниваемых архитектурах.

РОС относится к классу потоковых архитектур, причем, в отличие от них, тегируемые данные являются самодостаточными (рекуррентно разворачивающимися). Теги содержат некоторую начальную сжатую информацию, обеспечивающую выполнение требуемой процедуры. Каждый

следующий шаг процедуры рекуррентно самоопределяется, в том числе с учетом результата предыдущего шага.

В разрабатываемой архитектуре в состав центрального процессорного устройства (ЦПУ) включено автономное устройство преобразования тегов (ПТ), которое обладает возможностью саморазвертки рекуррентного вычислительного процесса. Устройство ПТ представляет собой относительно простую комбинационную схему, содержащую средства настройки на предметную область. В памяти есть только набор операндов и начальных значений их функциональных полей, которые динамически подвергаются рекуррентной саморазвертке устройством ПТ. Данное представление программы называется капсула. Более подробное сравнение предлагаемой архитектуры и существующих архитектур приводится в работе [2].

В текущем исполнении модель РОС имеет четыре процессорных ядра (ПЯ), способных функционировать параллельно. Таким образом, можно говорить о параллелизме на уровне ПЯ. Однако, это не единственный уровень параллелизма, задействованный в РОС. Специфика исполнения архитектуры позволяет выделить достаточно глубокий конвейер, каждая из ступеней которого может функционировать независимо. Также можно выделить параллелизм на самом низком уровне – вычислительных блоках и регистрах. В настоящем исполнении модель РОС поддерживает несколько разновидностей суперскалярных режимов вычислений [3].

На рис. 3 приведена структура двухуровневой архитектуры РОС, более подробное описание которой содержится в работе [1].

Возможные подходы программирования в РОС

Специфика архитектуры РОС находит свое отражение в способе его программирования. Интеграция потоков команд и данных ведет к аналогии с классами и объектами в объектно-ориентированном подходе. Следуя принципу инкапсуляции – программы в среде РОС представляют собой капсулы. Каждая капсула – это рекуррентно свернутый алгоритм. Она содержит набор самоопределяющихся данных (операндов). В свою очередь,

функциональные поля операндов подвергаются рекуррентному и функциональному преобразованиям в ходе развертки. Данный подход позволяет добиться эффективной реализации рекуррентных вычислений в параллельных архитектурах, где эта проблема проявляется особенно остро.

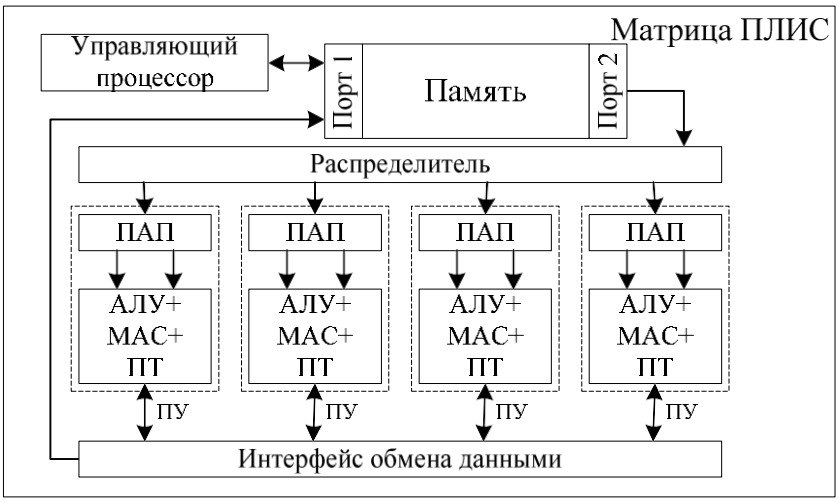

Рисунок 3. Структура РОС.

Из всего сказанного следует, что существующие подходы и языки программирования не могут быть в полной мере применены для эффективного программирования РОС. Это наталкивает на мысль о заимствовании сильных сторон того или иного подхода в программировании РОС.

В результате анализа существующих подходов программирования [4], в качестве отправной точки, был выбран функциональный язык параллельного и потокового программирования Sisal [5], который является развитием языка VAL. Результатом компиляции Sisal-программы является не только оптимизированный под параметры параллельной системы объектный код, но и мультиграф вычислительного процесса.

Другим интересным языком является К-язык [6], в основе которого лежат исчисления предикатов и λ-исчисления. Элементы этого языка наиболее точно соответствуют элементам РОС.

Графодинамическое представление алгоритмов

Развитие точных методов в программировании привело к

возникновению различных формальных моделей программ, в том числе и моделей параллельных программ. Среди них можно выделить: операторные схемы программ, сети Петри, UCLA-графы и др.[6]. Эти модели позволяют представить алгоритм в виде графовых структур, демонстрирующих последовательные и параллельные участки как вычислительного, так и управляющего процессов.

Исполняемая на РОУ программа представляется в виде капсулы. Выявление параллелизма и построение графа вычислительного процесса являются необходимыми, но не достаточными условиями написания капсулы. Необходимо также корректно отобразить управляющие процессы, с учетом специфичности архитектуры РОС.

Развитием идеи моделей параллельных программ в рамках данной работы – стала разработка внутреннего, графодинамического представления алгоритмов. На рис. 4 приведены фрагменты капсулы и графа вычисления алгоритма быстрого преобразования Фурье (БПФ), являющегося одним из основных в области цифровой обработки сигналов. Более подробно о реализации БПФ в РОС можно узнать в работе [3].

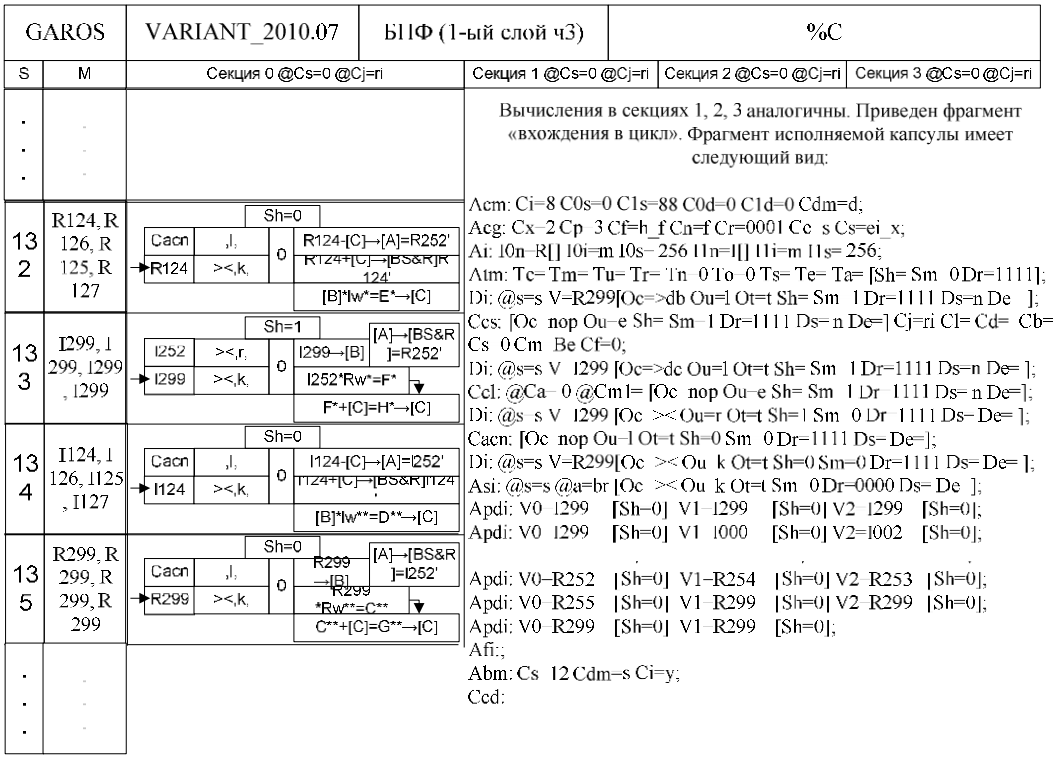

Рисунок 4. Фрагмент капсулы вычисления БПФ.

Представление алгоритма в виде подобного графа позволяет задать корректные настроечные параметры для устройства ПТ (в составе РОУ) и осуществить рекуррентную свертку в капсулу. Формальное описание графодинамического представления выходит за рамки данной статьи и не приводится.

Заключение

Результаты исследования проблем программирования РОС:

- подход к программированию РОС является комбинацией сильных сторон существующих парадигм;
- графодинамическое представление алгоритмов должно быть неотъемлемой частью процесса разработки программ в среде РОС;
- язык программирования РОС будет развитием К-языка;

Таким образом, необходимо выработать технологию разработки программного обеспечения в среде РОС, охватывающую весь жизненный цикл. Причем, эта технология будет подразумевать использование специализированных языков программирования и компиляторов для них. Эти языки и компиляторы должны охватывать три основных этапа: выявление параллелизма и представление алгоритма в виде графа; трансляция графа в графодинамическое представление; затем трансляция в капсульную форму.

Разрабатываемый инструментарий войдет в состав системы капсульного программирования и отладки СКАТ [7].

Литература

1. Степченков Ю.А., Петрухин В.С. Особенности гибридного варианта реализации на ПЛИС рекуррентного обработчика сигналов // Системы и средства информатики: Доп. Вып. – М.: ИПИ РАН, 2008. – С. 118-129.

2. Степченков Ю.А., Петрухин В.С., Филин А.В. Рекуррентное операционное устройство для процессоров обработки сигналов // Системы и средства информатики. Вып. 11. М.: Наука, 2001. – С. 283-315.

3. Степченков Ю.А., Волчек В.Н., Петрухин В.С., Прокофьев А.А., Зеленов Р.А. Механизмы обеспечения поддержки алгоритмов цифровой обработки речевых сигналов в рекуррентном обработчике сигналов // Системы и средства информатики. Вып. 20, №1. – М.: ТОРУС ПРЕСС, 2010. – С. 31-47.

4. Степченков Ю.А., Петрухин В.С., Хилько Д.В. Выбор языковых средств представления параллельных алгоритмов для рекуррентного обработчика сигналов // Системы и средства информатики: Доп. Вып. – М.: ИПИ РАН, 2008. – С. 149-158.

5. Gaudiot J.-L., BohmW., Najjar W., DeBoni T., Feo J., Miller P. The Sisal Model of Functional Programming and its Implementation // Proceedings of the Second Aizu International Symposiumon Parallel Algorithms/Architectures Synthesis (pAs ’97), Aizu-Wakamatsu, Japan, March 17-21, 1997.

6. Питерсон Дж. Теория сетей Петри и моделирование систем: Пер. с англ. – М.: Мир, 1984. – 264 с., ил.

7. Зеленов Р.А., Степченков Ю.А., Волчек В.Н., Хилько Д.В., Шнейдер А.Ю., Прокофьев А.А. Система капсульного программирования и отладки // Системы и средства информатики. Вып. 20, №1. – М.: ТОРУС ПРЕСС, 2010. – С. 25-30.