УДК 004.272.43

Методы и алгоритмы автоматической расстановки задержек в вычислительных структурах с обратными связями

А. А. Гуленок[1], А. В. Бовкун[2], В. А. Гудков[3]

[1]andrei_gulenok@mail.ru, [2]simans2002@mail.ru, [3]slava_gudkov@mail.ru
[1,2,3] Научно-исследовательский институт многопроцессорных вычислительных систем Южного федерального университета (НИИ МВС ЮФУ)

Поступило в редакцию 17.10.2012

Аннотация. Предлагаются методы и алгоритмы синхронизации информационных потоков в вычислительных структурах без обратных связей или с одной или несколькими обратными связями, состоящих из конвейерных блоков, данные на которые поступают с одинаковой тактовой частотой.

Ключевые слова. *Реконфигурируемая вычислительная система; ПЛИС; синхронизация; обратная связь*

В последнее время широкое распространение получили программируемые логические интегральные схемы (ПЛИС), которые широко применяются в производстве ЖК-дисплеев, медиацентров, сетевого оборудования, автомобилестроении и других областях [1].

Как правило, кристаллы ПЛИС встраиваются в общую схему оборудования для решения отдельных подзадач из области цифровой обработки сигналов, сетевых технологий и др., а их использование обусловлено возможностью изменения заданной области без изменения конструкции изделия.

СОСТОЯНИЕ ВОПРОСА

В настоящее время производительность кристаллов ПЛИС растет гораздо быстрее производительности традиционных микропроцессоров [2]. Полученные теоретические и практические результаты показывают, что существует большой класс задач, для которых целесообразно использовать кристаллы ПЛИС не только в качестве сопроцессора, решая на них отдельные специализированные подзадачи, но и в качестве основного вычислительного ресурса. Такие задачи названы потоковыми [3] и характеризуются обработкой больших объемов данных по одному и тому же алгоритму.

Для решения большинства потоковых задач необходим ресурс более чем одного кристалла ПЛИС.

На данный момент в мире развивается направление по созданию класса вычислительных устройств, названных реконфигурируемыми вычислительными системами (РВС), представляющими собой совокупность нескольких кристаллов ПЛИС на одной печатной плате с возможностью объединения нескольких печатных плат в единый вычислительный контур.

Для эффективного использования РВС в НИИ МВС ЮФУ разрабатываются и развиваются инструментальные средства программирования подобных систем [4], которые обеспечивают быструю разработку эффективных параллельных программ для реконфигурируемых вычислительных систем на языке высокого уровня.

Одной из важнейших задач синтеза эффективных параллельных программ является задача автоматической расстановки синхронизирующих задержек в конвейерных вычислительных структурах, обеспечивающих одновременность поступления операндов на все конвейерные блоки. Данная задача заключается в обеспечении одновременности поступления входных операндов для каждого конвейерного блока [5].

Исследования выполнены при поддержке научно-исследовательских работ по договору № 1301-Ю от 01.04.2011 г. во исполнение государственного контракта № 07.514.12.4001 от 24.12.2010 г. Министерства образования и науки РФ.

ПОСТАНОВКА ЗАДАЧИ

Параллельный алгоритм прикладной программы можно представить в виде информационного графа, где вершины соответствуют вычислительным и интерфейсным блокам, реализующим математические и схемотехнические операции, а дуги – информационным зависимостям между ними. В данной работе рассматриваются алгоритмы синхронизации информационных потоков в вычислительных структурах, состоящих из конвейерных блоков, данные на которые поступают с одинаковой тактовой частотой.

Множество вершин информационного графа $G = (U, V)$ структурной программы состоит из трех подмножеств: $U = I \cup O \cup W$, где I – множество вершин, соответствующих источникам потоков операндов, O – множество вершин, соответствующих приемникам потоков операндов, W – множество вершин, соответствующих операционным блокам вычислительной структуры прикладной задачи.

Задача синхронизации информационных потоков в вычислительных структурах, состоящих из конвейерных блоков, заключается в обеспечении одновременности поступления операндов на входы конвейерных блоков. При этом должна быть сохранена реальная производительность РВС при решении различных прикладных задач с конвейерной вычислительной структурой, содержащих контуры.

СУЩЕСТВУЮЩИЕ МЕТОДЫ И АЛГОРИТМЫ ДЛЯ АВТОМАТИЧЕСКОЙ СИНХРОНИЗАЦИИ

В ранее разработанных инструментальных средствах программирования РВС [4] используются следующие метод и алгоритм автоматической расстановки задержек. Одновременность поступления операндов обеспечивается за счет добавления дополнительных задержек в те потоки входных операндов, которые опережают самый «медленный» поток (поток с наибольшей суммарной латентностью, рассчитанной от источников операндов).

Для вычислительных структур без обратных связей данный метод реализуется с помощью следующего алгоритма, состоящего из двух этапов: расчета максимальной латентности информационных потоков на каждом конвейерном блоке, считая от источников операндов, и выравнивания потоков операндов на каждом конвейерном блоке относительно потока с максимальной латентностью.

На первом этапе сначала для всех вершин, соответствующих источникам операндов, параметру максимальной латентности от источников операндов присваивается значение «0»: $\forall x \in I$, $S_x = 0$. Для всех остальных вершин графа параметру максимальной латентности от источников операндов присваивается значение «-1»: $\forall y \in (O \cup W)$ S_y =-1, где S_x и S_y – значение максимальной латентности информационных потоков, считая от источников операндов, для вершин x и y соответственно. Далее выполняется обход в ширину информационного графа параллельной программы, начиная с вершин – источников операндов. Если в графе существует такая вершина z, для которой S_z=-1 и S_w >-1, $\forall w \in P_z$, где P_z – множество предшественников вершины z, то для вершины z значение максимальной латентности информационных потоков от источников операндов рассчитывается, как максимум, в качестве суммы S_w+L_w среди всех предшественников, где L_w – латентность конвейерного блока, представленного вершиной w. Обход в ширину заканчивается, когда для всех вершин информационного графа рассчитаны значения максимальной латентности от источников операндов.

Если представить информационный граф в ярусно-параллельной форме [6], то очевидно, что для любой вершины x на любом ярусе значение S_x будет определено (примет значение, отличное от «-1»), если определены значения для вершин на всех предыдущих ярусах (в том числе и для всех предшественников вершины x). А так как для всех вершин на первом уровне (множества источников операндов) значение S задается равным нулю, то перед началом обхода графа всем вершинам информационного графа будет присвоено значение S, отличное от «-1».

На втором этапе для каждой вершины x информационного графа рассчитывается число дополнительных задержек, которые необходимо добавить на входные потоки операндов, по формуле

$$\Delta l_i = S_x - l_i, \qquad (1)$$

где S_x определяет максимальную латентность среди всех потоков операндов для блока x; l_i – латентность i-го потока операндов конвейерного блока x; Δl_i определяет число задержек, которое необходимо поставить на i-й поток операндов.

Описанный метод обеспечивает выравнивание потоков операндов на каждом конвейерном блоке параллельной прикладной программы. При этом количество аппаратного ресурса, задействованного под синхронизацию информационных потоков, может оказаться соизмеримым с количеством аппаратного ресурса, отводимого под функциональные блоки вычислительной структуры. Для вычислительных структур без обратных связей существуют методы снижения суммарного числа синхронизирующих задержек, которые подробно рассмотрены в [7]. Данные методы позволяют повысить удельную производительность реконфигурируемой вычислительной системы (отношение реальной производительности к суммарному занятому аппаратному ресурсу). При этом реальная производительность РВС при решении прикладной задачи, вычислительная структура которой сформирована из конвейерных блоков и не содержит обратных связей, зависит лишь от числа функциональных устройств и рабочей тактовой частоты, так как данные на конвейерные функциональные блоки вычислительной структуры и результаты с них поступают каждый такт.

ПРЕДЛАГАЕМЫЕ МЕТОДЫ И АЛГОРИТМЫ

При появлении обратной связи в вычислительной структуре прикладной задачи может оказаться, что данные на конвейерные блоки необходимо будет подавать с некоторой «скважностью» через некоторое количество тактов [5]. Связано это с тем, что при наличии обратной связи, как это показано на рис. 1, на вход конвейерного блока нельзя подавать новое данное, пока не будет готов предыдущий результат. При наличии обратных связей в вычислительной структуре задачи реальная производительность РВС вычисляется как произведение числа функциональных блоков на значение рабочей тактовой частоты и деленное на минимальную скважность.

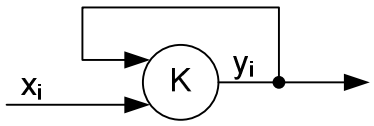

Рис. 1. Пример конвейерного блока с обратной связью

В данном примере минимальная скважность будет равняться латентности конвейерного блока. В случае если конвейерный блок является составным (состоит из нескольких последовательно включенных блоков), то его латентность определяется суммой латентности последовательно соединенных конвейерных блоков, и в данном случае скважность зависит от рассчитанной суммарной латентности. Любая задержка, попавшая между последовательно включенными конвейерными блоками с обратной связью, увеличит их суммарную латентность, а следовательно, увеличит минимальную скважность и снизит реальную производительность системы. Поэтому при расстановке синхронизирующих задержек важно не допускать попадания задержек между последовательно соединенными конвейерными блоками с обратной связью.

Дуги информационных графов, соответствующих обратным связям, не участвуют в расчете максимальной латентности от источников операндов и в расчете величины дополнительных задержек для информационных связей, синхронизирующих входные операнды. Контуром в информационном графе, соответствующим обратной связи конвейерной вычислительной структуры, будем называть такой путь в информационном графе, где первая и последняя вершина совпадают и соответствуют блоку-приемнику обратной связи.

На рис. 2 приведен пример информационного графа параллельной программы, состоящий из источников потоков операндов $X4$, $X5$, $X6$, приемника операндов $X7$ и операционных конвейерных блоков $X1$, $X2$ и $X3$, для которых определены латентности блоков L_i, i =1..7, рассчитаны максимальные латентности от источников операндов S_i и необходимые задержки Δl_i, рассчитанные по формуле (1).

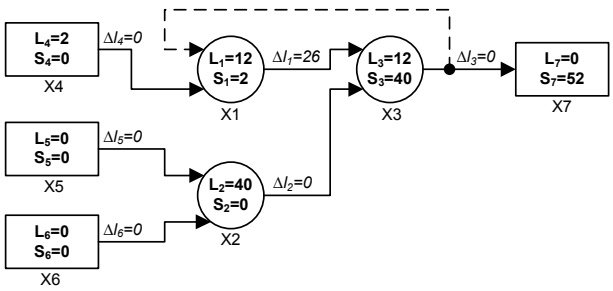

Рис. 2. Пример информационного графа с обратной связью

Данный граф содержит контур {*X*1, *X*3, *X*1} и при расчете задержек по формуле (1) необходимо на связь между вершинами *X*1 и *X*3 добавить задержку в 26 тактов, что приведет к увеличению суммарной латентности в контуре, а следовательно, и к снижению темпа поступления входных операндов для контура {*X*1, *X*3, *X*1}.

Из графа (рис. 2) видно, что если для первого элемента контура – вершины *X*1 □ значение максимальной латентности от источников операндов S_1 увеличить на суммарное число синхронизирующих задержек, попавших между вершинами контура, и пересчитать задержки операндов для элемента X_1 и всех последующих элементов, то синхронизирующие задержки из контура будут удалены (рис. 3).

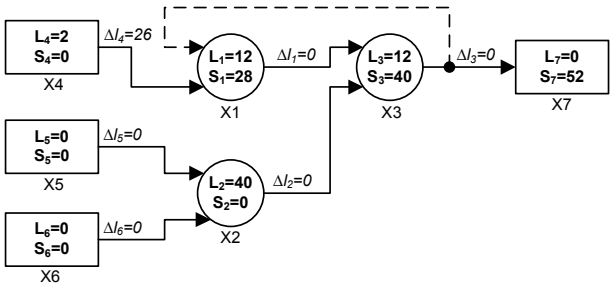

Рис. 3. Информационный граф с обратной связью после оптимизации задержек

Такой метод позволяет гарантированно удалять задержки из единственного контура в информационном графе. В том случае, если информационный граф содержит несколько контуров, может понадобиться многократное применение данного метода, что значительно увеличит время работы алгоритма синхронизации. Для решения задачи синхронизации в конвейерной вычислительной структуре с одной или несколькими обратными связями предлагается единый метод расчета задержек для всех случаев (с одним и с несколькими контурами): любой подграф $H = (U_H, V_H)$ информационного графа $G = (U, V)$, формируемый множеством вершин одного контура, рассматривать в некотором смысле как «единую» вершину, т. е. для любой вершины x, являющейся последователем хотя бы одной вершины из множества U_H, не определять значение максимальной латентности от источников операндов S_x, пока есть хоть одна вершина из множества U_H, для которой не определена максимальная латентность от источников операндов.

Алгоритм синхронизации информационных потоков в конвейерных вычислительных структурах с одной или несколькими обратными связями выглядит следующим образом. На первом этапе из информационного графа выделяется множество непересекающихся подграфов *UC*, множество вершин каждого из которых соответствует вершинам одного контура. Далее по очереди выбираются подграфы из множества *UC* и для каждой вершины x выбранного подграфа $H = (U_H, V_H)$ рассчитываются значения максимальной латентности информационных потоков S_x относительно первых вершин контуров. При этом в расчете участвуют только те дуги, которые соединяют вершины подграфа H. В итоге полученные для подграфов значения $|S_x - S_y|$, где x и y – смежные вершины одно и того же контура, определяют, насколько должны различаться значения рассчитываемых максимальных латентностей в исходном информационном графе, чтобы между вершинами x и y не была помещена дополнительная синхронизирующая задержка.

После того как для всех вершин подграфа $H = (U_H, V_H)$ из множества *UC* определены максимальные латентности информационных потоков, для всех вершин $x \in U_H$ рассчитывается значение $D_x = S_y - S_x$, где y – вершина из подграфа H с максимальным значением рассчитанной латентности информационных потоков в данном контуре. На рис. 4 представлен пример расчета значений S и D для контура {*X*1, *X*2, *X*3, *X*4, *X*1}.

После данного этапа запоминаются значения D, рассчитанные для всех вершин, входящих в какой-либо контур. Далее, как и для ранее описанного алгоритма, $\forall x \in I$, $S_x = 0$ и $\forall y \in (O \cup W)$, $S_y = -1$, где S_x и S_y – значения максимальной латентности информационных потоков, считая от источников операндов, для вершин x и y соответственно.

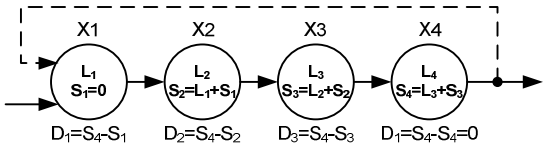

Рис. 4. Пример расчета задержек внутри контура

На следующем этапе алгоритма расчета задержек в конвейерных вычислительных структурах с обратными связями выполняется обход в ширину, аналогичный обходу в алгоритме синхронизации вычислительных структур без обратных связей, но который имеет два ключевых отличия. Первое отличие: если вершина x

является последователем некоторой вершины y и вершина y входит в множество вершин какого-либо контура, то значение S_x не будет рассчитано, пока не рассчитаны значения максимальной латентности информационных потоков для всех вершин данного контура. Второе отличие: как только для всех вершин подграфа $H = (U_H, V_H)$, описывающего некоторый контур, рассчитаны значения максимальной латентности информационных потоков от источников операндов, то для всех вершин $x \in U_H$ пересчитываются данные значения следующим образом: $S_x = S_y - D_x$, где y – вершина из подграфа H с максимальным значением рассчитанной латентности информационных потоков в данном контуре. В результате введения данных отличий в алгоритм обхода графа разница значений $S_z - S_x$ между двумя последовательно соединенными вершинами x и z (x предшественник z) будет равняться латентности конвейерного блока, соответствующей вершине x, и значение Δl для дуги между вершинами x и z, рассчитанное по формуле (1), будет равно нулю.

Описанные метод и алгоритм синхронизации информационных потоков в конвейерных вычислительных структурах с одной или несколькими обратными связями позволяет выравнивать потоки операндов для каждого блока, при этом исключает расстановку задержек между блоками, представленными одним контуром в информационном графе задачи, тем самым не увеличивается минимальная скважность для потоков операндов, что позволяет сохранять реальную производительность реконфигурируемой системы при автоматической расстановке синхронизирующих задержек в вычислительной структуре прикладной задачи. Тем не менее, суммарная величина задержек может оказаться немного больше, чем суммарная величина задержек, которые были бы расставлены алгоритмом синхронизации информационных потоков для вычислительных структур без обратных связей.

Увеличение числа задержек зависит индивидуально от решаемой задачи и не может быть оценено теоретически для какого-либо класса задач.

ПРАКТИЧЕСКИЕ ИССЛЕДОВАНИЯ

Предложенный авторами алгоритм был включен в 2012 г. в инструментальные средства программирования РВС, а именно – в синтезатор масштабируемых вычислительных структур Fire!Consructor. Данный алгоритм используется синтезатором на этапе автоматической расстановки задержек, выравнивающих потоки операндов в конвейерных вычислительных структурах.

Работа синтезатора со старым и новым алгоритмом расстановки задержек была проверена на более чем 200 тестовых примерах и реальных прикладных задачах из области математической физики, линейной алгебры и цифровой обработки сигналов. Увеличение суммарной величины синхронизирующих задержек не превышало 4–5 % для вычислительных структур с обратными связями по сравнению с суммарной величиной задержек, расставленных алгоритмом синхронизации потоков операндов, не учитывающим обратные связи и составляющим не более 2 % процентов от суммарного аппаратного ресурса, занимаемого функциональными блоками вычислительной структуры задачи.

При использовании алгоритма синхронизации вычислительных структур с обратными связями минимальная скважность для потоков операндов не увеличивалась после расстановки задержек.

ВЫВОДЫ

В статье авторы предлагают метод и алгоритм синхронизации, учитывающие обратные связи при автоматической расстановке задержек, выравнивающих потоки операндов для конвейерных вычислительных структур. Предложенные метод и алгоритм по сравнению с существующими исключают расстановку задержек внутри контуров, тем самым сохраняются темп поступления данных, а следовательно, и реальная производительность РВС при решении различных прикладных задач с конвейерными вычислительными структурами.

СПИСОК ЛИТЕРАТУРЫ

1. **ПЛИС Xilinx**. Итоги 2006 года и тенденции развития // Компоненты и технологии. 2006. №12. СПб.: Изд-во «Файнстрит», 168 с.

2. Состояние рынка и расширение сферы применения ПЛИС. Компоненты и технологии №5 2004.

3. Реконфигурируемые мультиконвейерные вычислительные структуры / И. А. Каляев [и др.]. Ростов н/Д: Изд-во ЮНЦ РАН, 2008. 320 с.

4. Средства программирования реконфигурируемых многопроцессорных вычислительных систем // Известия ТРТУ. Тематический выпуск «Интеллектуальные и многопроцессорные системы» / В. А. Гудков [и др.]. Таганрог: Изд-во ТРТУ, 2006. - № 16 (71). Спец. выпуск. С. 16–20.

5. **Раскладкин М. К.** Методы и средства автоматизированного сопряжения функциональных узлов и блоков в

приложениях для реконфигурируемых вычислителей: дис. … канд. техн. наук. Таганрог, 2010. 228 с.

6. **Воеводин В. В., Воеводин Вл. В.** Параллельные вычисления. СПб.: БХВ-Петербург, 2002. 608 с.

7. **Доронченко Ю. И.** Организация эффективных вычислений для реконфигурируемых вычислительных систем на основе ПЛИС // Известия ТРТУ. Тематический выпуск «Интеллектуальные и многопроцессорные системы». Таганрог: Изд-во ТРТУ, 2006. № 16. С. 11–16.

ОБ АВТОРАХ

Гуленок Андрей Александрович, ст. науч. сотр., инж. по программному обеспечению (ТРТУ, 2005). Канд. техн. наук (НИИ МВС ЮФУ, 2011). Иссл. в обл. параллельного программирования на РВС.

Бовкун Александр Викторовмч, мл. науч. сотр., инж. по программному обеспечению (ТРТУ, 2001). Иссл. в обл. параллельного программирования на РВС.

Гудков Вячеслав Александрович, ст. науч. сотр., инж. по программному обеспечению (ТРТУ, 2005). Канд. техн. наук (НИИ МВС ЮФУ, 2010). Иссл. в обл. параллельного программирования на РВС.

METADATA

Title: Methods and algorithms of automatic placing of delays in computing structures with no feedbacks.

Authors: A. A. Gulenok, A.V. Bovkun, V.A. Gudkov

Affiliation:
[1] Scientific Research Institute of Multiprocessor Computer Systems at Southern Federal University (SRI MCS SFU), Russia.
[2] Scientific Research Institute of Multiprocessor Computer Systems at Southern Federal University (SRI MCS SFU), Russia.
[3] Scientific Research Institute of Multiprocessor Computer Systems at Southern Federal University (SRI MCS SFU), Russia.

Email: [3]Slava_Gudkov@mail.ru.

Language: Russian.

Source: Vestnik UGATU (Scientific journal of Ufa State Aviation Technical University), 2013, Vol. 17, No. 2 (55), pp. 125-130. ISSN 2225-2789 (Online), ISSN 1992-6502 (Print).

Abstract: The paper covers methods and algorithms of synchronization of data streams in computing structures with no feedbacks, or with one or several feedbacks. The computing structures consist of pipeline blocks which receive data with the same clock frequency.

Key words: reconfigurable computer system; FPGA; synchronization; feedback.

References (English Transliteration):

1. Xilinx FPGAs. “Results of 2006 and development trends” in *Components and Technologies*, St. Petersburg, FineStreet Publishing, 2006, No. 12, 168 pp.
2. “Market conditions and expansion of FPGA use” in *Components and Technologies*, St. Petersburg, FineStreet Publishing, 2004, No. 5.
3. Kalyaev I.A., Levin I.I., Semernikov E.A., Smoilov V.I. “Reconfigurable multipipeline computing structures”. – Rostov-On-Don, SSC RAS Publishing, 2008, 320 pp.
4. Gudkov V.A., Gulenok A.A., Dordopulo A.I., Slasten L.M. “Software tools of reconfigurable multiprocessor computer systems” *in TSURE Proceedings “Intelligent and multiprocessor systems”*, Taganrog, TSURE Publishing, 2006, No. 16 (71), Special issue, pp. 16-20.
5. Raskladkin M.K. “Methods and tools of computer-aided interfacing of functional nodes in applications for reconfigurable computers” in the PhD thesis, defended on 19.11.2010, Taganrog, 2010, 228 pp.
6. Voevodin V.V., Voevodin Vl.V. “Parallel calculations”, St. Petersburg, BHV-Petersburg, 2002, 608 pp.
7. Doronchenko Y.I. “Organization of effective calculations for FPGA-based reconfigurable computer systems”, *in TSURE Proceedings “Intelligent and multiprocessor systems”*, Taganrog, TSURE Publishing, 2006, No. 16 (71), Special issue, pp. 11-16.

About authors:

1. Gulenok, Andrey Aleksandrovitch, junior scientific officer, software engineer (TSURE, 2005). PhD, Dept. of Intelligent and Multiprocessor Systems. (SRI MCS SFU, 2011). Research in parallel programming of RCS.
2. Bovkun, Alexander Victorovitch, junior scientific officer, software engineer (TSURE, 2001). Research in parallel programming of RCS.
3. Gudkov, Vyacheslav Aleksandrovitch, senior scientific officer, software engineer (TSURE, 2005). PhD, Dept. of Intelligent and Multiprocessor Systems. (SRI MCS SFU, 2011). Research in parallel programming of RCS.