Разработка комплекса средств для организации вычислений на FPGA-системах с помощью инструментального средства CAME&L

М. Вуденберг[2], Л.А. Наумов[1, 2]
[1] Санкт-Петербургский государственный университет информационных технологий, механики и оптики; [2] University of Amsterdam
levnaumov@mail.ru

Была выполнена работа [1] по созданию программного обеспечения для организации вычислительных экспериментов на FPGA-системах [2] ("Field Programmable Gate Array" – "программируемая вентильная матрица") при помощи инструментального средства CAME&L [3–5]. Тем самым было расширено множество вычислительных архитектур, на которых можно проектировать и выполнять вычисления с помощью упомянутого инструментального средства.

Идея использования FPGA-процессоров для выполнения вычислений при помощи клеточных автоматов не нова [6, 7], так как применение этих систем позволяет существенно повысить эффективность вычислений. В работе [7] показано, что это при использовании современных FPGA-процессоров, вычисления могут производиться в тысячи раз быстрее, чем на отдельно-стоящем персональном компьютере.

Для приведения компонента правил [4, 8, 9], описывающего вычислительный эксперимент для инструментального средства CAME&L, к виду, пригодному для выполнения на FPGA-системе, было разработано три программы, обрабатывающие исходный код одна за другой. Перечислим их, чтобы далее ссылаться на них по номерам.

1. Препроцессор.
2. Транслятор.
3. Синтаксический анализатор шаблона.

Опишем процедуру преобразования и компиляции исходного кода. На входе препроцессора (программа №1) имеется исходный код класса, реализующего компонент правил. В задачи этой программы входит преобразование конструкций языка C++ и специфических выражений библиотеки CADLib (таких как, обращение к клетке, к её соседям и т.п.), а также вычислительного ядра правил в форму, пригодную для взаимодействия с FPGA-системой на основании имеющегося у неё шаблона. Большинство инструкций заменяется вызовами коммуникационных функций и функций преобразования данных к виду, пригодному для работы с FPGA-процессорами. Специфические параметры, например, тип окрестности, описываются директивами `#pragma` [10], анализируемыми препроцессором. Результат работы этой программы, с одной стороны, подаётся на вход транслятору (программа № 2), с другой собирается обычным компилятором (как правило, используется Microsoft Visual C++ 6) в библиотеку управляющего компонента правил, который будет взаимодействовать с FPGA-системой и координировать эксперимент.

Транслятор (средство № 2) состоит из двух частей. Первая, созданная с помощью инструментального средства LEX [11], переводит исходный код на языке программирования C++, порождённый препроцессором (программа № 1), в набор лексем языка C. Вторая, которая была разработана при помощи средства YACC [11], транслирует этот набор на язык VHDL [12] ("VHSIC Hardware Description Language", где "VHSIC" – "Very-High-Speed Integrated Circuit", таким образом, название можно перевести как "язык описания аппаратного обеспечения для сверхвысокоскоростных интегральных схем"), являющийся одним из самых распространённых языков высокого уровня для программирования FPGA-

систем. Необходимо отметить, что здесь имеет место не тривиальный перевод, а трансляция с элементами семантического анализа. Так, например, обращение к функции `rand()` [10] преобразуется в существенный фрагмент кода на языке VHDL, реализующий генерацию псевдослучайных чисел. Одной из самых существенных задач транслятора, при этом, является приведение разрядности переменной для хранения состояния клетки, заявленной в исходном коде компонента правил, в соответствие с характеристиками разрядности клетки используемого аппаратного обеспечения.
Полученный на выходе транслятора (программа № 2) шаблон будущего кода (набор инструкций на языке VHDL), обрабатывается синтаксическим анализатором шаблона (программа № 3). Имея в своём распоряжении подготовленный заранее типичный шаблон обмена информацией и взаимодействия с памятью, а также, обладая значениями фундаментальных параметров эксперимента, как, например, размерность решётки, синтаксический анализатор шаблона собирает программу на языке VHDL окончательно, сопоставляя, в частности, номера портов ввода-вывода всем клеткам решётки FPGA-системы для дальнейшего взаимодействия со средой выполнения клеточных автоматов. Итоговая программа поступает на вход компилятору с языка VHDL, а результат его работы закачивается в FPGA-систему, после чего она готова к проведению вычислительного эксперимента под управлением инструментального средства CAME&L с загруженным в него управляющим компонентом правил.

Работа выполнена при поддержке Российского фонда фундаментальных исследований по гранту № 05-07-90086 «Разработка среды и библиотеки "CAME&L" для организации параллельных и распределённых вычислений на основе клеточных автоматов».

Литература

1. *Woudenberg M.* Using FPGAs to Speed Up Cellular Automata Computations / Master's thesis for University of Amsterdam. 2006.
2. *Lee S.* Advanced Digital Logic Design Using VHDL, State Machines, and Synthesis for FPGA's. Thomson-Engineering. 2005.
3. Сайт «CAMEL Laboratory» – http://camellab.spb.ru.
4. *Naumov L.* CAME&L – Cellular Automata Modeling Environment & Library // Cellular Automata. Sixth International Conference on Cellular Automata for Research and Industry, ACRI-2004. Springer–Verlag. 2004.
5. *Наумов Л.* CAME&L – Среда моделирования и библиотека разработчика клеточных автоматов // Труды XI Всероссийской научно-методической конференции Телематика'2004. Том 1. СПб.: СПбГУ ИТМО. 2004.
6. *Shackleford B., Tanaka M., Carter R.J., Snider G.* FPGA Implementation of Neighborhood-of-Four Cellular Automata Random Number Generators / Proceedings of FPGA'2002. 2002.
7. *Halbach M., Hoffmann R.* Implementing Cellular Automata in FPGA Logic / Proceedings of the 18th International Parallel and Distributed Processing Symposium (IPDPS'04). 2004.
8. *Наумов Л.* Решение задач с помощью клеточных автоматов посредством программного обеспечения CAMEL (Часть I) // Информационно управляющие системы. 2005. № 5.
9. *Наумов Л.* Решение задач с помощью клеточных автоматов посредством программного обеспечения CAMEL (Часть II) // Информационно управляющие системы. 2005. № 6.
10. *Страуструп Б.* Язык программирования C++. СПб: Невский диалект. 1999.
11. *Керниган Б., Пайк Р.* UNIX. Программное окружение. СПб.: Символ-Плюс. 2003.
12. *Pedroni V.* Circuit Design with VHDL. The MIT Press. 2004.