УДК 621.398+004.4'4
ББК 32.965
М 23

Мандрик Татьяна Анатольевна
Аспирант кафедры вычислительной техники института компьютерных технологий и информационной безопасности Южного федерального университета, Таганрог, тел. (8634) 371656, e-mail: ugrevatova@sfedu.ru
Поленов Максим Юрьевич
Доцент, кандидат технических наук, доцент кафедры вычислительной техники института компьютерных технологий и информационной безопасности Южного федерального университета, Таганрог, тел. (8634) 371656, e-mail: mypolenov@sfedu.ru

Высокоуровневые инструменты трансляции моделей для разработки ПЛИС*

(Рецензирована)

***Аннотация**. Цель работы – сравнение высокоуровневых инструментов, используемых для разработки современных ПЛИС: HDL coder от компании Mathworks, HLS Compiler от Intel и HLS от компании Xilinx. Для достижения поставленной цели был рассмотрен основной функционал данных пакетов. В результате исследования было показано, что в настоящее время наиболее эффективным средством высокоуровневой системы для разработки ПЛИС является HLS Compiler, производимый компанией Intel. В качестве дополнительного инструментария предлагается также использование собственной разработки – среды многоязыковой трансляции Мультитранслятора.*

***Ключевые слова**: инструментальные средства, разработка ПЛИС, трансляция моделей, языки HDL.*

Mandrik Tatyana Anatolyevna
Post-graduate student of the Department of Computer Engineering, Institute of Computer Technology and Information Security, Southern Federal University, Taganrog, ph. (8634) 371656, e-mail: ugrevatova@sfedu.ru
Polenov Maxim Yuryevich
Associate Professor, Candidate of Engineering Sciences, Associate Professor of the Department of Computer Engineering, Institute of Computer Technology and Information Security, Southern Federal University, Taganrog, ph. (8634) 371656, e-mail: mypolenov@sfedu.ru

High-level models' translation tools for FPGA design

***Abstract.** The purpose of the work is to compare high-level tools used to develop modern FPGAs: HDL coder from Mathworks, HLS Compiler from Intel, and HLS from Xilinx. To achieve this goal, the main functionality of these packages was considered. As a result of the study, it is shown that the HLS Compiler, manufactured by Intel, is currently the most effective tools of a high-level system for FPGA development. As an additional tool, it is also proposed to use the author's own development – the multilanguage compilation environment – the Multitranslator.*

***Keywords**: instrumental tools, FPGA design, models' translation, HDL languages.*

Введение

Разработка и создание различных схемных решений на основе программируемых логических интегральных схем (ПЛИС) – достаточно длительный и трудоемкий процесс даже для профессиональных инженеров, работающих в области схемотехники. Под ПЛИС в данной статье рассматриваются в основном FPGA (Field Programmable Gate Array), работа над которыми ведется с применением языков описания аппаратуры (ЯОА) – hardware description language (HDL), например, таких, как VHDL и Verilog. Необходимо также отметить, что на данный момент существует большое количество программных решений, предназначенных для трансляции программных моделей устройств с языков высокого уровня (чаще всего с Си) на языки описания аппаратуры (VHDL, Verilog). Эти оттранслированные модели используются в дальнейшем при компиляции непосредственно для проектов и программирования устройств.

В последние годы производители FPGA и различные компании активно развивают методы разработки, отличающиеся от привычных подходов использованием высокоуровневых

* *Работа выполнена при финансовой поддержке РФФИ (проект № 19-07-00936).*

средств разработки. Обычно при разработке ПЛИС в качестве основного инструмента используют язык описания аппаратуры VHDL или Verilog, но растущая популярность новых подходов и инструментальных программных средств вызывает достаточно большой интерес разработчиков. Таким образом, можно сделать вывод, что изучение данных инструментов и проведение их сравнения являются достаточно актуальными.

В данной работе рассмотрены три наиболее широко распространенные инструментальные средства для проектирования ПЛИС: HDL coder от компании Mathworks, HLS Compiler от Intel и Vivado HLS от компании Xilinx.

Инструментальные средства, используемые для проектирования ПЛИС

HDL Coder [1] от компании Mathworks предоставляет пользовательский интерфейс для автоматизации и управления рабочим процессом проектирования FPGA таких фирм, как Xilinx, Microsemi и Intel. Данный программный продукт расширяет возможности для разработки аппаратных блоков и систем в среде моделирования MATLAB. Исходная модель при этом создается в подсистеме Simulink, а компонент HDL Coder обеспечивает генерацию кода модели на языке VHDL или Verilog с побитовым и цикловым соответствием.

Разработчик при этом имеет возможность контролировать архитектуру системы в процессе проектирования и синтеза HDL-проекта, находить и устранять критические или нежелательные пути распространения сигнала, а также проводить предварительную оценку объема используемых ресурсов для исполняемого алгоритма. HDL Coder обеспечивает однозначное соответствие между моделью MATLAB/Simulink и сгенерированным VHDL или Verilog кодом, кроме того, в пакете присутствует функция верификации проектов с высокой плотностью размещения на кристалле согласно стандарту DO-254.

Также HDL Coder позволяет сформировать Verilog или VHDL код модели, портируемый на платформы различных производителей, из функций MATLAB, моделей Simulink и диаграмм состояний Stateflow. Полученный код может быть использован для программирования FPGA или прототипирования заказных микросхем ASIC [1].

HLS Compiler [2] от Intel – это инструмент синтеза высокого уровня (High Level Synthesis – HLS), который принимает в качестве входных данных язык untimed C++ и генерирует код уровня передачи регистров производственного качества (Register Transfer Level – RTL), оптимизированный для FPGA корпорации Intel. Этот инструмент ускоряет время проверки над RTL-уровне путем повышения уровня абстракции для проектирования аппаратного обеспечения FPGA, поскольку модели, разработанные на C++, обычно проверяются на порядок быстрее, чем RTL-описание.

Компилятор HLS от Intel включен в программную установку среды проектирования Intel Quartus Prime Design. В настоящее время выпущен новый релиз Intel HLS Compiler v19.3 [2].

Особенностями нового релиза являются:

- новая функция HLS float, позволяющая создать шаблон определенного типа на основе требуемой ширины мантиссы и экспоненты, а затем использовать его в качестве обычного типа float в коде HLS;
- новый вид планировщика задач на основе диаграммы Ганта;
- LSU Control – возможность более гибкой настройки между производительностью и занимаемой площадью [3].

Vivado HLS [4] от компании Xilinx транслирует код, написанный на языках высокого уровня (C++ или SystemC), в описания на HDL, которые можно использовать для дальнейшего синтеза и реализации. Использование HLS позволяет проводить проектирование FPGA разработчику, который не специализируется на HDL: для создания работающего модуля (или даже проекта) не обязательно детально знать, что такое, например, «тактовая частота» и многие другие низкоуровневые конструкции. Естественно, пользователи с таким уровнем подготовки редко создают проект на FPGA полностью. Но реализовать на FPGA, хотя бы на уровне прототипа, отдельный модуль, исполняющий знакомый специалисту (программисту, ин-

женеру по обработке сигналов, математику и др.) алгоритм, становится для него вполне реализуемой задачей и без постоянного и активного привлечения высококвалифицированного ПЛИС-разработчика. Также использование Vivado HLS удобно, когда важно добиться работы на FPGA поведенческой модели, не беспокоясь об объеме занятых ресурсов кристалла.

В итоге Vivado HLS от Xilinx позволяет:

- разрабатывать проекты на С-уровне;
- проводить верификацию также на С-уровне;
- создавать код, работающий на любом семействе FPGA [4, 5].

Можно сказать, что высокоуровневые инструменты, используемые для разработки ПЛИС, традиционно применяются со следующими целями:

- ускорить разработку проекта:
  - за счет повторного применения готового кода моделей на языках высокого уровня;
  - за счет использования всех возможностей и достоинств языков высокого уровня;
  - за счет сокращения временных затрат на компиляцию и верификацию кода;
- возможность создавать код, который подойдет для реализации проектов на различных семействах FPGA;
- снизить порог вхождения в разработку для FPGA. Уход от низкоуровневых понятий (таких, например, как «тактовая частота») даст возможность писать код для FPGA инженеру, не знакомому с языками описания аппаратуры.

Сравнение инструментальных средств

В настоящее время наибольшего успеха в области HLS достигли компании, обладающие существенными финансовыми и человеческими ресурсами, в то время как инициативные разработки несколько отстают.

В данной работе рассматриваются такие продукты, как HDL coder, HLS Compiler и Vivado HLS. Ранее сравнение первых двух пакетов было рассмотрено в [6]. Здесь необходимо отметить, что инструментальные средства фирм Intel и Xilinx имеют различную структуру, что делает затруднительным проведение однозначного сравнения результатов. В данном случае хотелось бы отметить, что Xilinx HLS и Intel HLS поддерживают компиляторы C/C++ и на техническом уровне они схожи.

Для сравнения возможностей программных продуктов была рассмотрена реализация выполнения алгоритма быстрого преобразования Фурье (БПФ), который широко используется при построении систем цифровой обработки сигналов. Рассмотрим БПФ для входных данных разной длины.

Таблица использованных ресурсов и быстродействия для каждого программного продукта представлена на рисунке 1.

Размер БПФ	HLS Compiler		HDL Coder	
	Задержка (тыс.циклов)	Кол-во блоков	Задержка (тыс.циклов)	Кол-во блоков
512	3,2	2760	2,7	2007
1024	7,4	3216	6,1	2241
2048	18,1	3989	16,2	2547
4096	28,9	4625	31,4	2946
8192	60,1	5433	70,5	3879
16384	131,8	6328	167,9	4748

Рис. 1. Использованные ресурсы

Для оценки проектов используем следующих показатели:

- задержка – количество тактовых циклов сигнала синхронизации, необходимых для получения результата на выходе устройства;

- количество логических блоков: таблицы поиска и синхронные триггеры.

Одним из основных преимуществ пакетов Intel HLS и Xilink HLS по сравнению с HDL coder является тот факт, что HLS compiler реализует в ПЛИС синхронный проект, хотя и потребляет большее количество ресурсов. Сгенерированная архитектура демонстрирует, что Intel HLS и Xilink HLS направлены на достижение максимально возможной производительности, но не экономят ресурсы. Объемное потребление регистров и блочной памяти объясняется большим количеством дополнительных модулей. Несомненно, простота и эффективность получаемого результата при использовании HDL coder являются достаточно весомыми факторами, однако при стандартных настройках не удается получить проект, который будет обладать достаточно высокой производительностью.

Для того чтобы наглядно сравнить результаты выполнения алгоритма, данные, приведенные в таблице на рисунке 1, представлены в виде двух графиков (рис. 2 и рис. 3).

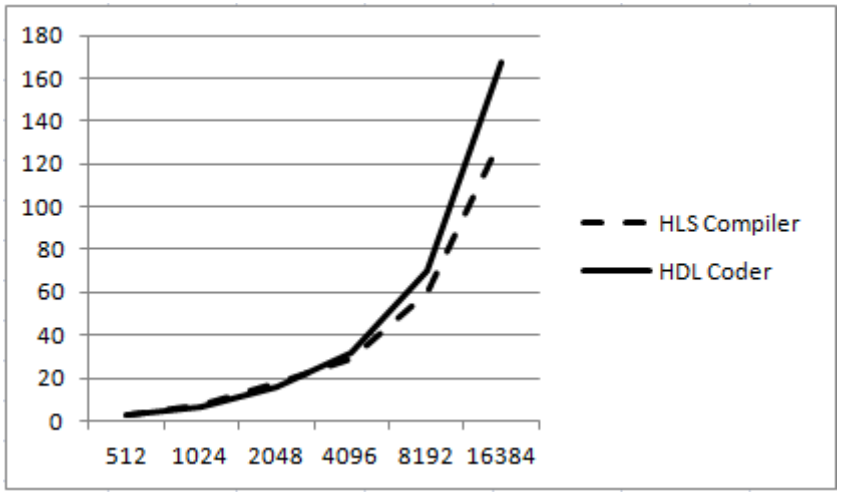

Рис. 2. Количество тактовых циклов

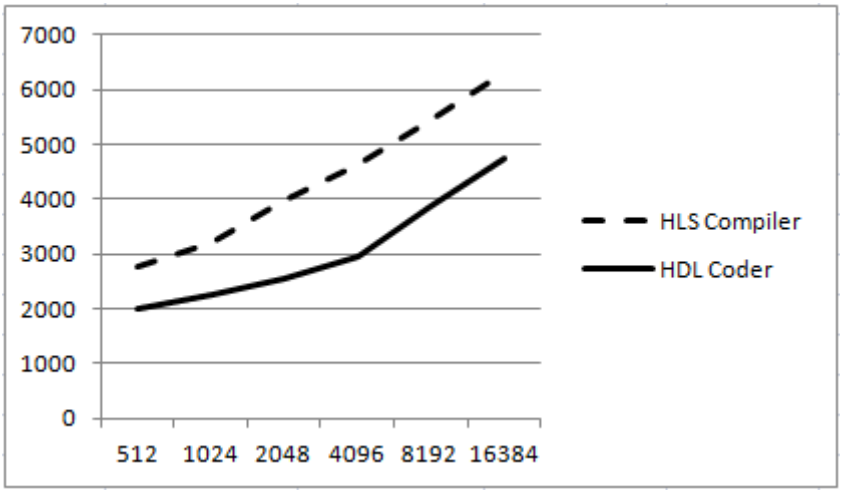

Рис. 3. Количество логических блоков

Из графика на рисунке 2 видно, что при увеличении размерности БПФ HLS Compiler генерирует проект с меньшей задержкой, а HDL Coder показывает хорошие результаты только для задач небольшой размерности. Расход ресурсов значительно больше у HLS Compiler, для решения этой проблемы производитель предлагает использование специальных библиотек, оптимизированных функций и различных программных директив.

На взгляд авторов, в настоящее время наиболее совершенной является технология и пакет HLS Compiler, которые были разработаны компанией Intel. Данный программный продукт позволяет получить работающий проект, не предъявляя особых требований к исходному коду на языке C/C++. Тогда как для использования HDL coder входная модель должна быть сразу адаптирована к компиляции проекта на языки описания аппаратуры.

Заключение

Таким образом, можно с уверенностью сказать, что все существующие высокоуровневые средства не являются идеальными. Они могут использоваться для задач проектирования, требующих максимально быстрого решения (когда нет необходимости экономить ресурсы или получать пиковую производительность), или же для больших задач, которые можно разделить на достаточно простые подзадачи, легко реализуемые с помощью HLS. Но эти инструменты не являются полной заменой низкоуровных проектов.

При выборе какого-то конкретного средства возникает необходимость написания кода модели на том языке высокого уровня, с которым работает компилятор (чаще всего с C/C++). В настоящее время процесс создания программных моделей в большей степени основан на использовании библиотек готовых моделей. Можно применять существующие модели для решения задач различной сложности. Таким образом возникают ситуации, когда реализованные и отлаженные модели компонент необходимо конвертировать из одной среды в другую (например, из среды Matlab на язык C++).

Для реализации такой конвертации (трансляции) программных моделей можно использовать различные программные средства. Одним из таких средств является среда многоязыковой трансляции моделей – Мультитранслятор, разработанная на кафедре вычислительной техники ЮФУ [7]. В основу Мультитранслятора заложен подход по созданию и использованию трансляционных модулей для конвертации моделей с исходного на целевой язык [8], при этом дополнительно используется экспертная система, которая помогает разрешить возникающие при переводе моделей неопределенности [9]. Многоязыковая трансляция может применяться на каждом этапе проектирования, например для высокоуровневого синтеза. Мультитранслятор позволяет сделать процесс проектирования систем более гибким, так как дает возможность разработчикам создавать собственные правила трансляции и редактировать существующие [10, 11].

В заключение хотелось бы подчеркнуть, что на данном этапе развития высокоуровневых средств проектирования пользователю, прежде всего, при выборе этих средств следует ориентироваться на собственные конкретные цели проектирования.

Примечания:

1. HDL Coder. 2020. URL: https://exponenta.ru/products/hdl-coder#description
2. Intel HLS Compiler. 2020. URL: https://www.intel.ru/content/www/ru/ru/software/programmable/quartus-prime/hls-compiler.html
3. An Analytical Model of Memory-Bound Applications Compiled with High Level Synthesis / M.A. Davila-Guzman [et al.] // 2020 IEEE 28th Annual International Symposium on Field-Programmable Custom Computing Machines (arXiv:2003.13054).
4. Vivado High-Level Synthesis. 2020. URL: https://www.xilinx.com/products/design-tools/vivado/integration/esl-design.html
5. Разработка IP-блока с помощью инструментов высокоуровнего синтеза: HLS. Ч. 1. 2020. URL: http://fpga-sys-tems.ru/publ/xilinx/xilinx_hls/razrabotka_ip_bloka_s_pomosh-hju_instrumentov_vysokourovnego_sinteza_hls_chast_1/21-1-0-69
6. Угреватова Т.А., Поленов М.Ю. Сравнение высокоуровневых инструментов трансляции программных моделей для ПЛИС // Информационные технологии, системный анализ и управление (ИТСАУ-2019): сб. трудов XVII Всерос. науч. конф. молодых ученых, аспирантов и студентов. Т 1.

References:

1. HDL Coder. 2020. URL: https://exponenta.ru/products/hdl-coder#description
2. Intel HLS Compiler. 2020. URL: https://www.intel.ru/content/www/ru/ru/software/programmable/quartus-prime/hls-compiler.html
3. An Analytical Model of Memory-Bound Applications Compiled with High Level Synthesis / M.A. Davila-Guzman [et al.] // 2020 IEEE 28th Annual International Symposium on Field-Programmable Custom Computing Machines (arXiv:2003.13054).
4. Vivado High-Level Synthesis. 2020. URL: https://www.xilinx.com/products/design-tools/vivado/integration/esl-design.html
5. Developing an IP block using high-level synthesis tools: HLS. Pt. 1. 2020. URL: http://fpga-sys-tems.ru/publ/xilinx/xilinx_hls/razrabotka_ip_bloka_s_pomosh-hju_instrumentov_vysokourovnego_sinteza_hls_chast_1/21-1-0-69
6. Ugrevatova T.A., Polenov M.Yu. Comparison of high-level tools for translating programming models for FPGAs // Information technologies, system analysis and management (ITSAU-2019): Proceedings of the 17th Russian Scientific Conference. Vol. 1. Rostov-on-Don: Publishing House of SFedU, 2019. P. 6–10.

Ростов н/Д: Изд-во ЮФУ, 2019. С. 6–10.

7. Поленов М.Ю. Организация распределенных инструментальных средств поддержки многократно используемых моделей // Известия ЮФУ. Технические науки. 2013. № 7. С. 201–207.
8. Поленов М.Ю., Лапшин В.С., Гушанский С.М. Реализация модуля конвертации моделей для среды MATLAB // Известия ЮФУ. Технические науки. 2017. № 6. С. 212–223.
9. Поленов М.Ю., Курмалеев А.О. Использование интеллектуальных средств при организации трансляции моделей // Информатизация и связь. 2016. № 3. С. 23–27.
10. Чернухин Ю.В., Поленов М.Ю., Булгаков Д.В. Использование мультиязыковой трансляции при конверсии моделей, представленных на языках описания аппаратуры // Искусственный интеллект. 2009. № 4. С. 462–471.
11. Чернухин Ю.В., Поленов М.Ю., Булгаков Д.В. Использование многоязыковой трансляции при RTL-синтезе СБИС на языках описания аппаратуры // Известия ЮФУ. Технические науки. 2010. № 7. С. 220–226.

7. Polenov M.Yu. Organization of distributed tools for supporting reusable models // News of the Southern Federal University. Engineering Science. 2013. No. 7. P. 201–207.
8. Polenov M.Yu., Lapshin V.S., Gushansky S.M. Implementation of the Model Conversion Module for the MATLAB Environment // News of the Southern Federal University. Engineering Sciences. 2017. No. 6. P. 212–223.
9. Polenov M.Yu., Kurmaleev A.O. Intellectual Tools Usage for Models Translation // Informatization and Communication. 2016. No. 3. P. 23–27.
10. Chernukhin Yu.V., Polenov M.Yu., Bulgakov D.V. Multilanguage Translation Usage at Conversion of Models Presented on Hardware Description Languages // Artificial Intelligence. 2009. No. 4. P. 462–471.
11. Chernukhin Yu.V., Polenov M.Yu., Bulgakov D.V. Usage of the multilanguage translation in VLSI RTL-synthesis on hardware description languages // News of the Southern Federal University. Engineering Science. 2010. No. 6. P. 220–226.