УДК 004.272.23

СРАВНИТЕЛЬНЫЙ АНАЛИЗ ЭФФЕКТИВНОСТИ ВЫСОКОУРОВНЕВЫХ СРЕДСТВ РАЗРАБОТКИ ДЛЯ ПЛИС

Андреев А. Е., Малолетков В. А., Оболонин М. А., Фролов Г. О., Чернояров Д. А.

ГОУ ВПО «Волгоградский государственный технический университет», Волгоград, Россия (400005, Волгоград, пр. им. Ленина, 28), e-mail: andan2005@yandex.ru

Проведен сравнительный анализ систем высокоуровнего синтеза(HLS) LegUp, Impulse C и Автокод HDL. HLS-системы сравнивались по функциональным возможностям, простоте освоения, удобству использования и по ряду количественных характеристик синтезированных ими модулей.
Анализ синтезированных модулей проводился по следующим критериям: количество израсходованных логических вентилей, умножителей и ячеек памяти, максимально возможная рабочая тактовая частота, производительность и энергопотребление синтезированных ими тестовых модулей вычисления циклического кода (CRC), БПФ (FFT) и криптоалгоритма AES.
Исследование показало, что на данный момент времени открытые HLS-системы, такие как LegUp, способны конкурировать с широко известными коммерческими, например Impulse C, если и не в производительности генерируемых модулей, то, как минимум, в удобстве использования и простоте освоения, но на сегодняшний день ни одна из протестированных систем не способна составить конкуренцию описанным на HDL схемам по производительности.

Ключевые слова: Verilog, FPGA, LegUp, Impulse C, Автокод HDL.

COMPARATIVE ANALYSIS OF MODERN HIGH LEVEL SYNTHESIS TOOLS FOR FPGA

Andreev A. E., Maloletkov V. A., Obolonin M. A., Frolov G. O., Chernoyarov D. A.

[1]Volgograd State Technical University, Volgograd, Russia (400005, Volgograd, Lenin avenue, 28), e-mail:andan2005@yandex.ru

This paper contains a comparative analysis of three high level synthesis systems(HLSs): LegUp, Impulse C and Autocode HDL. Those systems were compared by their functionality, operability, simplicity of mastering and number of hardware metrics of units, synthesized by this HLSs.
We used set of metrics that include: number of logic elements(LE), number of memory bits used (bits), number of used multipliers(muls), maximum unit clock frequency(max freq), thermal power dissipation and unit performance for CRC, FFT and AES computing units.
Our research showed that state-of-the-art open source HLSs like LegUp capable of competing with commercial like Impulse C and Autocode HDL. We established that open Source HLSs like Legup couldn't be a concurrents to commercical HLSs in performance aspect, but they can be more user-friendly. But potential users of LegUp should know that this HLSs only available on Linux. We also ascertained that all tested HLSs couldn't be a performance competitors to HDL-based schematics.

Key words: Verilog, FPGA, LegUp, Impulse C, Autocode HDL.

Введение

Создание схемных решений для работы с ПЛИС (программируемые логические интегральные схемы, под которыми в данной работы в основном подразумеваются FPGA) с применением таких языков описания оборудования, как VHDL и Verilog, зачастую требует значительных временных затрат даже от профессиональных инженеров-схемотехников. Но существует немало программных решений, предназначенных для трансляции программного кода непосредственно с языков высокого уровня (чаще всего – C) в языки описания аппаратуры (VHDL, Verilog) для последующей компиляции непосредственно в данные для программирования устройств.

В ноябре 2012 года Altera объявила о скором внедрении поддержки стандарта OpenCL для ряда своих продуктов. На данный момент заявлена поддержка семейств Stratix IV–V. В настоящий момент поддерживаются платы Terasic DE-4 и Nallatech PCIe-385 . В будущем это позволит эффективно использовать FPGA Altera в составе гибридных вычислительных кластеров. В связи с тем, что официально будут поддерживаться только модели плат на базе самых передовых ПЛИС Altera, актуальным становится вопрос рассмотрения других решений для синтеза схем, в том числе и открытых (opensource), для оценки перспективности их использования в научно-исследовательских проектах.

В данной статье мы проводим сравнительный анализ трех существующих высокоуровневых средств разработки для FPGA (LegUp, Impulse C и Автокод HDL).

Обзор сравниваемых HLS-средств

LegUp представляет собой свободный каркас ПО с открытым исходным кодом для синтеза схемных решений для FPGA из программ на языке C, разрабатываемый группой ученых из университета Торонто [6]. Последняя на момент написания статьи версия каркаса 3.0 обеспечивает частичную поддержку ANSI C, включая многомерные массивы, структуры, адресную арифметику и арифметику с плавающей точкой. Динамическое выделение памяти и рекурсия не поддерживаются.

LegUp предоставляет транслятор байт-кода LLVM (lowlevelvirtualmachine) в Verilog и использует clang для компиляции кода на C в байт-код LLVM.

Необходимо также отметить, что каркас работает только под Linux, не имеет графического интерфейса и поддерживает только семейства FPGA Cyclone 2 и Stratix 4.

ImpulseC – это основанный на языке C набор библиотек и средств для написания программ для ПЛИС. Модель программирования ImpulseC – это модифицированная версия модели взаимодействующих последовательных процессов. Основными понятиями модели ImpulseC являются процессы и потоки. Процессы – это независимые, параллельно выполняющиеся компоненты приложения [4]. Коммуникации между процессами осуществляются в основном через буферизованные потоки данных, которые реализованы непосредственно в аппаратных средствах. Помимо потоков поддерживаются другие способы коммуникации – сигналы, общая память, семафоры и регистры.

Среда разработки CoDeveloper для Impulse C предоставляется в нескольких вариантах: для FPGA компании Xilinx и компании Altera, существуют версии для Windows и Linux.

Главным недостатком Impulse C является высокая стоимость, но имеется возможность получить временную лицензию для ознакомления с продуктом.

Наибольший интерес среди российских разработок в области высокоуровневых средств синтеза схем вызывает решение Автокод HDL, создаваемое в институте прикладной

математики им. М. В. Келдыша РАН [2, 3]. По мнению его авторов, написание программы на языках VHDL и Verilog представляет собой некоторое подобие ручного кодирования команд процессора, а программа на автокоде – ассемблер для этого процессора. Спроектированная версия языка Автокод HDL позволяет описывать разрабатываемые схемы с помощью описания операторов, выполняющихся на каждом такте, при этом синхронность исполнения тактов обеспечивается самим языком [2].

В текущей версии языка отсутствуют некоторые базовые логические операторы, а также нет возможности создавать пользовательские типы данных.

Авторы рассчитывают, что после завершения работы над «ассемблером» для ПЛИС, следующим шагом станет появление более высокоуровневых языков, подобных С/Фортран, которые будут поддерживать традиционные подходы к программированию.

Методология сравнительного анализа

Существует немало приемов и методик для оценки эффективности высокоуровневых средств разработки (например, работы [9], [10] и [11]). В частности в работе [11] авторами предложена методика, позволяющая получать интегральные оценки по интересующим метрикам для HLSs. Эту методику, за исключением алгоритма вычисления результирующих оценок для HLSs, мы использовали в нашей работе. Не меньшее значение имеет и подбор алгоритмов, используемых в качестве тестовых. Проблеме бенчмаркинга систем высокоуровнего синтеза посвящена работа [12], в ней предложен набор тестовых алгоритмов из разных областей информатики (преимущественно ЦОС, криптография и архивация), которые, по мнению авторов, имеют хорошие перспективы для реализации на ПЛИС. В данной работе тестирование HLSs мы проводили на алгоритмах FFT(БПФ) (64-х точечное, целочисленное, отсчеты и коэффициенты в комплексной форме, разрядность мнимой части = разрядности вещественной части = 16 бит), шифрования AES (длина блока = длине ключа = 128 бит), CRC16 (разрядность входных данных 8 бит) и функции-заглушке (dummyfunction), возвращающей входные значения назад без изменений (размерность передаваемых данных 8 бит), для оценки накладных расходов от использования того или иного средства разработки. Схемные решения, синтезированные при помощи описанных выше HLSs, мы сравнивали между собой с помощью следующих критериев:

- максимальная тактовая частота, на которой может работать схема,
- пропускная способность схемы,
- рассеиваемая мощность ПЛИС с запрограммированной схемой,
- количество задействованных в схеме логических элементов,
- время, затраченное на разработку схемы.

Данные собирались при помощи Quartus II 9.1 WebEdition. Проекты собирались для ПЛИС EP2C7oF89618 семейства Cyclone 2, т.к. только ПЛИС этого семейства поддерживаются как Quartus II 9.1 WebEdition, так и всеми исследуемыми HLS. Мы также задействовали в сравнительном анализе схемы, описанные на Verilog.

После проведения всех испытаний каждой HLSs по каждому атрибуту выставлялась оценка по следующему правилу [11]:

$$S_{l,f} = \frac{1}{N_a} \sum_{a=1}^{N_a} \frac{v_{a,l,f} - v_{a,l_0,f}^{min}}{v_{a,l_1,f}^{max} - v_{a,l_0,f}^{min}}$$

где:

N_a – количество тестовых задач;

$v_{a,l,f}$ – значение критерия f для HLSs l при тестовой задаче a;

$v_{a,l_0,f}^{min}$, $v_{a,l_1,f}^{max}$ – минимальное и максимальное значение критерия f для тестовой задачи a.

Таким же способом осуществлялась оценка необходимого для начала работы с HLSs времени, за исключением того, что для этой оценки $N_a = 1$.

Результаты тестирования HLSs представлены в таблице 1. Вычисленные для них по приведенному выше правилу интегральные оценки – на рисунке 1.

Качественный сравнительный анализ

Наиболее «низкоуровневым» из рассматриваемых нами средств разработки является Автокод HDL, поэтому временные затраты на синтез схем ожидаемо больше, по сравнению с ImpulseC и LegUp, но все же меньше, чем при разработке на Verilog.

ImpulseC является наиболее гибким инструментом среди рассматриваемых нами, но это коммерческий проект, кроме того для освоения парадигмы программирования, предлагаемой разработчиками данной системы, потребуется дополнительное время.

LegUp, с его уровнем поддержки ANSI C и набором скриптов, автоматизирующих процесс генерации прошивок, пожалуй, является наиболее удобным и простым для программиста, незнакомого с языками описания аппаратуры. К тому же, это единственная HLSs из рассмотренных, которая распространяется не только свободно, но и с открытым исходным кодом.

Таблица 1. Результаты тестирования

	Пропускная способность (Мб/с)	Максимальная тактовая частота (МГц)	Кол-во логических элементов	Затраты на разработку (ч)	Рассеиваемая мощность(мВ)	
Автокод HDL	11,9	143	135	4	36	CRC16
Impulse C	12,5	167	190	1	195	
Verilog	250,9	258	44	5	200	
LegUp(PH)	34,8	146	186	0,5	195	

Автокод HDL	66,3	70	25412	20	221	AES
Impulse C	1	69	2842	4	195	
Verilog	89,3	112	45325	25	239	
LegUp(PH)	2,7	102	3301	2	199	
Автокод HDL	263	67	51071	16	220	FFT
Impulse C	122	142	4245	3	197	
Verilog	242	99	29073	30	201	
LegUp(PH)	5,5	56	5588	1,5	197	
Автокод HDL	476	500	8	0,5	30	Dummyfunction
Impulse C	181	333	19	0,4	31,9	
Verilog	476	500	8	0,5	30	
LegUp(PH)	238	500	2	0,25	33	

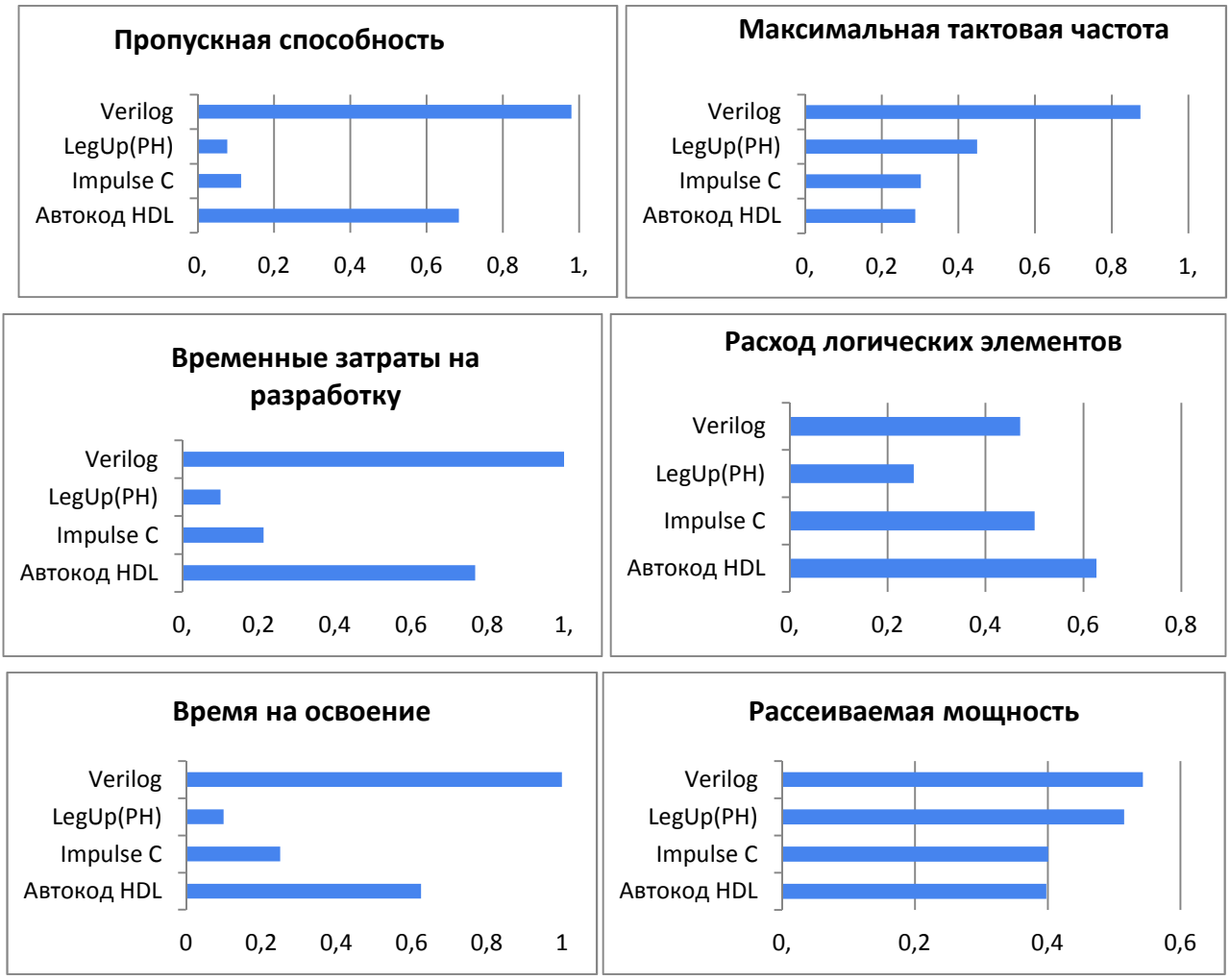

Рисунок 1. Интегральные оценки для HLSs по результатам анализа

Количественный сравнительный анализ

Чтобы оценить накладные расходы, создаваемые интерфейсом модуля, генерируемого исследуемыми HLS и определить максимально достижимую тактовую частоту модулей, мы создавали тестовые схемы, возвращающие входные данные разрядностью 8 бит без изменений. В этом тесте как Verilog, так и Автокод, и Legup сгенерировали регистр, но в

схеме, синтезированной Legup, присутствуют вспомогательные схемы, из-за которых она выдает данные один раз в 2 такта, а не в 1, что объясняет вдвое меньшую пропускную способность. Схема, синтезированная при помощи Impulse C, даже для такого простого примера включает в себя аппаратную реализацию базовых абстракций языка – входных и выходных потоков, которые заметно снижают общую производительность системы.

Следующим проведенным нами тестом была реализация алгоритма CRC-16. Вычислительная сложность данного алгоритма не высока и при последовательной подаче байтов входных данных вычисления 16-ти битного значения CRC16 занимает 1 такт при ручной реализации на Verilog. Схемы, сгенерированные участвующими в тестировании HLSs, тратят на вычисление значительно больших тактов, и не могут сравниться с эталонной, ни по пропускной способности, ни по занимаемой площади кристалла, ни по рассеиваемой мощности.

64-х точечный алгоритм комплексного БПФ, использованный нами в следующем тесте, позволяет оценить возможности HLSs по конвейеризации схем. Специфика алгоритма такова, что значительное время работы схемы тратится на прием и передачу входных и выходных данных. 64 отсчета / коэффициента занимают 64*2*2 = 256 байт и принимаются / выдаются в течение 64-х тактов, сам же этап вычислений может быть проведен за 60–100 тактов. При организации 3-х ступенчатого конвейера можно получить пиковую пропускную способность 1 коэффициента за такт. Результаты, полученные в этом тесте, показывают, что оптимальными средствами для организации вычислительных конвейеров обладает Автокод, а Legup не особенно подходит для организации высокопроизводительных схем.

Последний тест с наибольшей вычислительной сложностью – блочное шифрование по алгоритму AES со 128-ми битным ключом и блоком. Объем передаваемых данных для операции шифрования блока (128+128 бит на вход и 128 бит на выход) не слишком высок и позволяет организовать схемное решение с входными / выходными шинами соответствующей разрядности. Также алгоритм включает повторяющиеся действия над данными в процессе шифрования (10 раундов), что обеспечивает возможность конвейеризации. Этот тест оказался самым тяжелым для всех HLSs, кроме Автокода, отставшего по производительности от эталона на Verilog всего на 33 %. Хуже всего в этом тесте показал себя Legup, очевидно, из-за того, что перенос модели памяти программы на С непосредственно на ПЛИС малоэффективно, и это особенно заметно на задачах, требующих интенсивных вычислений, таких как AES.

Заключение

Проведённое небольшое исследование позволяет утверждать, что свободные системы высокоуровнего синтеза показывают результаты, сравнимые с коммерческими системами.

Среди рассмотренных нами HLSs наиболее производительные модули можно создавать при помощи Автокод HDL, что, однако, достигается за счет максимального, среди всех рассмотренных HLSs, расхода логических элементов. LegUp, в свою очередь, обеспечивает минимальное расходование ресурсов кристалла, но производительность синтезируемых им модулей самая низкая. При этом, время, затрачиваемое программистом на разработку схемы, как и время, необходимое на освоение системы, при использовании Legup минимально, т.к. его входной язык практически полностью соответствует стандарту C99.

Список литературы

1 Андреев А. Е., Силкин И. М., Шафран Ю. В. Высокоуровневые средства разработки для FPGA // Современные научные исследования и инновации. – Июнь, 2012 [Электронный ресурс]. – URL: http://web.snauka.ru/issues/2012/06/14365(дата обращения 06.12.12).

2 Технологии разработки прикладного ПО для реконфигурируемых вычислительных структур [Электронный ресурс] / С. С. Андреев [и др.]. – 2010. – URL :http://sdat.ispras.ru/archive/2010/20100318_Latsis.pdf.

3 Руководство программиста Автокод HDL [Электронный ресурс] / С. С. Андреев [и др.]. – 2012. – URL: http://www.kiam.ru/MVS/research/avtokod/index.html.

4 Языки высокого уровня ImpulseC, Mitrion-C и Handel-C [Электронный ресурс]. – URL :http://fpga.parallel.ru/lang.html (дата обращения: 30.11.12).

5 LegUp: High-Level Synthesis for FPGA-Based Processor/Accelerator Systems [Электронныйресурс] / A. Canis [и др.] // University of Toronto, Altera Toronto. – 2011. – URL: http://legup.eecg.utoronto.ca/fpga60-legup.pdf (датаобращения 08.12.12).

6 Impact of FPGA Architecture on Resource Sharing in High-Level Synthesis [Электронныйресурс] / Stefan Hadjis [и др.] // University of Toronto, Altera Toronto. – 2011. – URL: http://legup.eecg.utoronto.ca/p111-hadjis.pdf (дата обращения 05.12.12).

7 Impulse Tutorial: Generating HDL from C-Language [Электронныйресурс]. – URL: http://impulseaccelerated.com/Tutorials/Basic/Tutorial_Basic_HW_Gen.pdf (дата обращения: 02.12.12).

8 Impulse Tutorial: Generating an Altera FPGA Netlist from C-Language [Электронныйресурс]. – URL: http://impulseaccelerated.com/Tutorials/Altera/Tutorial_QuartusII_ Generic.pdf (дата обращения: 03.12.12).

9 Holland B., Vacas M., Aggarwal V., De Ville R., Troxel I., and George A. D., "Survey of C-based Application Mapping Tools for Reconfigurable Computing" [Электронныйресурс]. – URL: https://wiki.ittc.ku.edu/eecs700/images/b/b8/Holland.pdf (дата обращения: 24.12.12).

10 El-Araby E., Taher M., Abouellail M., El-Ghazawi T., and Newby G. B., “Comparative analysis of High LevelProgramming for Reconfigurable Computers: Methodology and Empirical Study” [Электронныйресурс]. – URL: http://www.ann.ece.ufl.edu/courses/eel6935_10spr/student _talks/Comparative_analysis_of_High_Level.pptx (дата обращения: 23.12.12).
11 El-Araby E., Merchant S. and El-Ghazawi T. “A Framework for Evaluating High-Level Design Methodologies for High-Performance Reconfigurable Computers” [Электронный ресурс]. – URL : http://www.chrec.org/pubs/TPDS10_G4.pdf (дата обращения: 25.12.12).
12 Hara Y. and others “CHStone: A Benchmark Program Suite for Practical C-Based High-Level Synthesis” [Электронныйресурс]. – URL: http://ir.nul.nagoya-u.ac.jp/jspui/bitstream/2237/ 12085/1/iscas2008-300_1027.pdf (дата обращения: 25.12.12).

Рецензенты:

Камаев Валерий Анатольевич, д-р техн. наук, профессор, заведующий кафедрой САПР и ПКВолгГТУ, г. Волгоград.

Муха Юрий Петрович, д-р техн. наук, профессор, заведующий кафедрой «Вычислительная техника» ВолгГТУ, г. Волгоград.