УДК [004.31+004.75]:681.518 DOI: 10.31114/2078-7707-2022-4-50-55

Аппаратная реализация приложения обработки изображений транспортных объектов на основе технологии «система на кристалле»

Ш.С. Фахми[1,2], И.Г. Малыгин[1,3], О.А. Королев[1]

[1]ФГБУН Институт проблем транспорта им. Н.С. Соломенко Российской академии наук, г. Санкт-Петербург, korolev@iptran.ru

[2]Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В. И. Ульянова, г. Санкт-Петербург, shakeebf@mail.ru

[3]ФГБОУ ВО «Санкт-Петербургский университет ГПС МЧС России», г. Санкт-Петербург, malygin_com@mail.ru

***Аннотация* — Растущая сложность современных систем обработки видеоинформации и ограниченные сроки их проектирования требуют применения новых инструментов разработки, используемых в современных системах автоматизированного проектирования и основанные на программируемых логических интегральных схемах (ПЛИС). В настоящее время в дополнение к традиционным языкам описания аппаратного обеспечения (hardware description languages – HDL) всё большую популярность набирают инструменты высокоуровневого синтеза (High Level Synthesis – HLS) микроархитектур, преимущество которых состоит в повышении уровня абстракции при разработке различных систем обработки сигналов, в частности изображений в области транспорта. Несмотря на простоту проектирования и тестирования в сравнении HDL, при использовании HLS необходимо учитывать недостатки, проявляющиеся при генерации описания аппаратного обеспечения, свойственные данному подходу. В статье на примере проектирования известного фильтра Кэнни, позволяющего выделить границы транспортных объектов на изображениях при решении задач обнаружения и распознавания, представлено сравнительное исследование двух методик синтеза цифровых устройств обработки изображений: HLS и HDL. Приведены результаты сравнения двух реализаций HDL и HLS, полученные на основании оценки вычислительных ресурсов и времени выполнения операций. Поскольку фильтр Кэнни, как и многие другие фильтры выделения контуров, является сверточным, данная работа позволяет оценить сильные и слабые стороны любого оператора в области обработки изображений. Исследование показало, что с учетом затрачиваемых ресурсов и времени отклика реализация HLS продемонстрировала более высокие показатели с точки зрения задействования ресурсов и времени отклика в сравнении с подходом HDL.**

***Ключевые слова* — ПЛИС, фильтр Кэнни, HDL, HLS, конфигурация, система на кристалле.**

I. Введение

Развитие интеллектуальных транспортных видеосистем на кристалле [1] требует в первую очередь внедрение субмикронных технологий проектирования устройств анализа видеоинформации. При этом основными функциями для обнаружения и отслеживания объектов в реальном времени, в частности транспортных средств, являются сегментация изображений, выделение контуров и распознавание. Кроме того, сама обработка изображений в режиме реального времени представляет собой сложный процесс и обычно требует высокой вычислительной мощности. Следовательно, алгоритмы обработки видео и изображений в реальном времени неспособны работать на устройствах с низкой вычислительной мощностью при разумной частоте кадров съёмки. В этом случае оптимальное решение – это применение программируемых логических интегральных схем (ПЛИС) [2].

ПЛИС находятся в промежуточном положении между специализированными и универсальными процессорами, учитывая их способность перенастраивать свою архитектуру в соответствии с задачами и высокую энергоэффективность [3,4]. За последнее десятилетие было предпринято множество усилий по снижению энергопотребления больших вычислительных систем и в этом плане ПЛИС укрепляются в качестве жизнеспособной альтернативы для достижения энергоэффективности конечного продукта.

Однако ПЛИС не получили массового распространения, как это первоначально ожидалось производителями. В течение более чем одного десятилетия приложения для ПЛИС разрабатывались исключительно с использованием языков описания оборудования (HDL). К сожалению, HDL имеют много недостатков: они многословны и тем самым подвержены ошибкам, требуют глубоких знаний в

области цифровой схемотехники, а следовательно, на проектирование с использованием HDL затрачивается больше времени.

В результате сообщество ПЛИС изучило альтернативные инструменты для повышения уровня абстракции, которые позволили бы снизить затраты на программирование и ускорить время выхода разработки на рынок [4]. С начала 2000-х годов несколько производителей ПЛИС начали предлагать инструменты высокоуровневого синтеза (HLS) для разработки систем. В подходе HLS программисты кодируют приложения ПЛИС, используя языки высокого уровня, такие как C, C++ или SystemC. Затем с помощью инструментов конвертации выполняется генерация соответствующего кода HDL [5].

Важно отметить, что инженеры, используя инструменты HLS, работают на более высоком уровне абстракции и имеют возможность повторно использовать ранее созданные конструкции, при этом обладание специальными знаниями в области аппаратного обеспечения не является обязательным. Этот альтернативный подход за счет повышения производительности и снижения стоимости разработки позволил отрасли сократить время выхода готового продукта на рынок.

Несмотря на то, что инструменты HLS обладают несомненными преимуществами при разработке описаний оборудования, у них также есть слабое место. Поскольку языки высокого уровня были разработаны для программных приложений, они имеют некоторые недостатки при описании аппаратного обеспечения, которые могут негативно влиять на использование ресурсов и времени отклика конечных устройств аппаратного обеспечения.

В этом контексте важно знать преимущества и недостатки различных языков и подходов для синтеза оптимальных описаний аппаратного обеспечения. В данной статье представлено сравнительное исследование двух решений при реализации фильтра Кэнни на основе технологии «система на кристалле» (СнК) в среде САПР Vivado, обеспечивающей как HDL, так и HLS-проектирование устройств обработки видеоинформации. Проектирование включает следующие основные процедуры:

1) создание общедоступных HDL- и HLS-проектов фильтра Кэнни доступными средствами САПР Vivado. Поскольку фильтр Кэнни является сверточным оператором, данные проекты могут быть легко адаптированы для создания других фильтров обработки изображений;

2) тщательное сравнение обоих решений с точки зрения использования вычислительных ресурсов, времени выполнения и усилий по созданию соответствующих программ.

Таким образом, можно определить сильные и слабые стороны каждого проекта в аспекте синтеза цифровых устройств обработки изображении на основе ПЛИС.

II. Способы аппаратной реализации на ПЛИС

Verilog и VHDL являются основными и наиболее известными языками описания схем и средствами HDL-синтеза аппаратных систем. Оба языка были разработаны в 80-х годах и с тех пор несколько раз обновлялись [6]. При этом важнейшим условием синтеза с использованием выше указанных языков является наличие у разработчика высочайшего уровня программирования и опыта схемотехнического проектирования, что ограничивает использование ПЛИС инженерами-аппаратчиками. Следовательно, в результате процессы разработки и отладки получаются низкоуровневые медленные.

Недостатки HDL привели к разработке новых инструментов для описания аппаратного обеспечения в начале 2000-х годов, таких как Vivado HLS (Xilinx), Catapult C (Mentor Graphics) и Intel OpenCL SDK (Intel). Эти инструменты на основе языков высокого уровня (HLL) повышают уровень абстракции и расширяют возможности ПЛИС для инженеров, специализирующихся на создании встроенного программного обеспечения. К сожалению, языки на основе C/C++ имеют некоторые недостатки при описании аппаратного обеспечения.

Во-первых, поскольку в HLS отсутствует определение аппаратной синхронизации, то либо разработчиком, либо в инструментарии HLS должна быть указана модель параллелизма.

Во-вторых, в HLL отсутствует определение точной битовой ширины сигнала, поскольку в них используются только ограниченные типы данных, такие как bool, int и/или long.

В-третьих, HLL не имеют абстракций аппаратных интерфейсов и в отличие от модели распределенной памяти ПЛИС предполагают плоскую модель памяти, доступ к которой можно получить через указатели. Поэтому инструменты HLS должны предоставлять расширения для HLL с помощью библиотек и наборов директив, чтобы преодолеть эти недостатки. В других случаях инструменты HLL накладывают ограничения на HLS, например, не поддерживают динамическое распределение памяти.

III. Фильтр Кэнни (Canny)

Поиск границ объектов на изображениях, в том числе транспортных, с помощью фильтра Кэнни, позволяющего работать в условиях зашумленности сигнала изображения, является одним из наиболее широко используемых алгоритмов их обнаружения. Повышенная сложность алгоритма требует использования высоких тактовых частот и значительных затрат на энергопотребление при условии использования классических микропроцессорных архитектур. Особенно, когда необходимо соответствовать ограничениям,

сопутствующим обработке изображений в реальном времени.

Фильтр Кэнни - это многоступенчатый детектор. Этапы выделения границ фильтром Кэнни показаны на рис. 1.

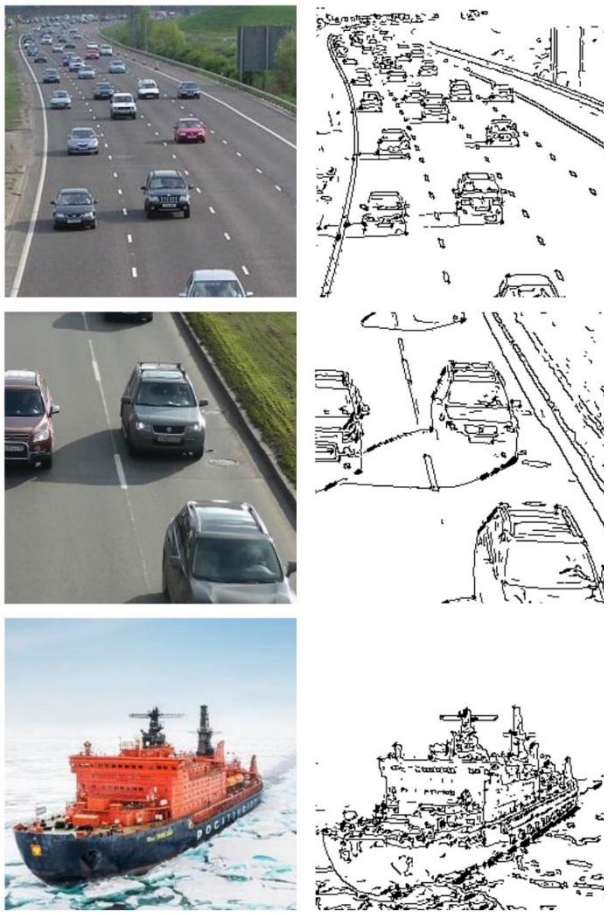

Рис. 1. Тестовые изображения выделения контуров фильтром Кэнни

HLS-реализация фильтра Кэнни на базе технологии «система на кристалле» на платформе САПР Vivado требует знания алгоритма выделения контуров и наличия опыта в решении проблем при выполнении процесса обнаружения границ объекта с помощью HDL- и HLS-инструментов.

Алгоритм выделения контуров Кэнни состоит из четырех этапов:

1. Этап сглаживания, на котором происходит удаление шума посредством фильтра Гаусса. Пусть исходное изображение будет задано яркостью x (m, n). Гауссово сглаживание с радиусом r считается по формуле (1):

$$y(m,n) = \frac{1}{2\pi r^2}\sum_{u,v} e^{\frac{-(u^2+v^2)}{2r^2}} x(m+u, v+v), \quad (1)$$

пределы суммирования по u и v можно выбирать как $\pm\sigma$. т.е. соответствующие нескольким радиусам r, что в итоге позволяет оценить сложность алгоритма как $O(r^2)$ операций на пиксель. Используя известное свойство сепарабельности гауссова сглаживания, можно сократить количество операций, добившись ускорения процесса обработки изображения. А именно, выполнить фильтрацию по оси x для каждой строки и полученное изображение отфильтровать по каждому столбцу y, получив тот же результат со сложностью $O(r)$ операций на пиксель.

2. Поиск градиентов. На данном этапе применяется дискретный дифференцированный оператор Собеля, с помощью которого выполняется расчёт приближенного значения градиента в каждой точке изображения. По итогам вычисления происходит формирование либо вектора градиента яркости в каждой точке изображения, либо его норма. С целью обнаружения изменений интенсивности в горизонтальном (d_x) и вертикальном (d_y) направлениях выполняется расчёт величины M края (см. формулу 2) и направление градиента θ и (см. формулу 3) соответственно:

$$M = \sqrt{d_x^2 + d_y^2} \quad (2)$$

$$\theta = tan^{-1}\left(\frac{d_y}{d_x}\right) \quad (3)$$

3. Этап подавления не максимумов. Для выявления пикселей границ объекта вычисляются локальные максимумы по яркости, при этом удаляются пиксели, локальный максимум которых достигается не в направлении вектора градиента. В итоге подавления не максимумов на изображении останутся только тонкие линии границ объектов. Отметим, что локальный максимум вдоль направления градиента обнаруживается посредством вычисления значений M и θ.

4. Двойная пороговая фильтрация. Алгоритм выполнения данной операции заключается в следующем: в случае, если яркость пикселя превышает установленное значения верхней границы, то присваивается максимально возможное значение яркости, если ниже – пиксель подавляется. Такой подход является эффективным инструментом устранения полос на изображении.

IV. Аппаратная реализация приложения обработки изображений

Чтобы сравнить подходы HDL и HLS, было проведено исследование по обнаружению объектов путём выделения контуров на изображениях формата RGB.

В состав архитектуры системы входит: ARM-процессор, блоки памяти, блок прямого доступа к памяти (DMA), блок модуля контроллера SDRAM с удвоенной скоростью передачи данных (DDR) для систем на основе ПЛИС (DDR memory controller) и блок приложения выделения контуров фильтром Кэнни на ПЛИС.

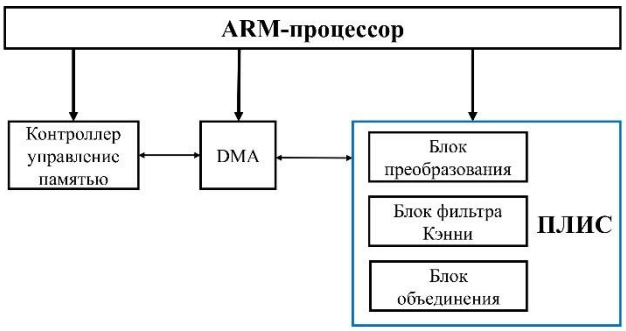

Рис. 2. Структурная схема проекта приложения обработки изображений

Рассмотрим структурную схему проекта приложения обработки изображений (см. рис. 2). Блок DMA обращается к памяти microSD для считывания изображения и отправляет поток пикселей в процессор для обработки. После выделения контуров объектов процессор возвращает поток пикселей в блок DMA для дальнейшего хранения в SD-памяти (microSD). Процессор отвечает за настройку и управление системой. Все модули проекта были интегрированы в среду САПР Vivado.

Блок выделения контуров состоит из 3 блоков, которые были синтезированы и проверены с использованием подходов HDL и HLS:

1) блок преобразования исходного изображения формата RGB в черно/белое изображение, содержащее оттенки серого формата GRAY (RGB→GRAY), используя метод, основанный на среднем арифметическом трех компонентов R-красного, G-зеленого и B-синего;

2) блок фильтра Кэнни для выделения контуров объектов на изображениях (см. рис. 1);

3) блок объединения четырех 8-битных целочисленных переменных для формирования 32-х битного слова.

A. HLS-версия

Для реализации фильтра Кэнни в HLS были использованы две структуры памяти, называемые линейным буфером и скользящим окном.

Каждый буфер строк соответствует целой строке изображения для сохранения рабочего набора скользящего окна Кэнни. Скользящее окно содержит пиксели изображения, которые будут свернуты. Фильтр требует наличия трех линейных буферов, расположенных один над другим. Когда принимается новый пиксель, линейные буферы выполняют вертикальный поворот перед сохранением нового пикселя. Когда два верхних строчных буфера заполнены, а три пикселя находятся в нижнем строчном буфере, процесс свертки выполняется с использованием пикселей скользящего окна и масок Кэнни. Подробное описание фильтра Кэнни можно найти в работах [1,7].

B. Сравнение расход ресурсов

В табл. 1 представлены результаты использования ресурсов для двух реализаций фильтра Кэнни. Значения в столбцах отражают процент задействованных в ходе обработки изображения логических таблиц (Lut), регистров (Reg), мультиплексоров (MUX) и блоков оперативной памяти (RAM) соответственно. В частности, для HLS-реализации требуется более чем в 5,5 раз больше логических таблиц LUT и в 4,6 раз больше регистров, чем при использовании HDL-версии.

Таблица 1

Расход ресурсов

Тип реализации	**Расход ресурсов**			
	Lut	**Reg**	**MUX**	**RAM**
HDL	0,92	0,7	0	1,9
HLS	5,2	3,1	0,1	1,89

В табл. 2 представлены результаты сравнения производительности HDL и HLS подходов при реализации фильтра Кэнни. При этом, при обработке для каждого тестового изображения рассчитываются два временных показателя:

1) время работы блоков обработки и выделения контуров, обозначенное в таблице словом Кэнни. Нужно отметить, что данные значения вычисляются только для столбца Кэнни каждого изображения;

2) время ввода-вывода (I/O) системы, отмеченное в таблице как I/O. Время ввода-вывода примерно одинаково для обеих версий, поскольку они совместно используют эту операцию.

Таблица 2

Сравнение производительности

Изображение	**Задача**	**Скорость выполнения (мс)**		
		HDL	**HLS**	**Ускорение**
Интенсивный транспортный поток	Кэнни	6.9	53.3	7,7
	I/O	923	955	-
Автомобильная дорога с малой интенсивностью движения	Кэнни	7,8	49,2	6,3
	I/O	1078	1028	-
Ледокол	Кэнни	34,5	63,4	1,8
	I/O	3993	3964	-

Кроме того, в последнем столбце приведено соотношение времени выполнения HLS- и HDL-реализаций фильтра Кэнни, обозначенное как ускорение.

Как видно из представленных в табл. 2 результатов, фильтр Кэнни при HDL-реализации во всех случаях работает быстрее, чем его HLS версия. Наибольшая

разница в производительности проявилась при обработке изображения интенсивного транспортного потока, представленного на рис. 1, достигнув ускорения более чем в 7 раз. При этом с увеличением размеров транспортных объектов на изображении разница в производительности сокращается, достигнув минимума в 1.8 раза при обработке изображения ледокола.

Реализация HDL превзошла версию HLS как по использованию ресурсов, так и по производительности. Это связано с особенностями процесса автоматизации в среде Vivado при переводе кода из HLL в HDL, при котором не всегда учитываются особенности оборудования при синтезе оптимальных описаний аппаратного обеспечения. В итоге результирующий RTL (register transfer level), описывающий последовательность логических операций, применяемых для передачи данных от одного регистра к другому, требует большего количества конечных автоматов и цифровых сигналов, нуждаясь в повышенном потреблении ресурсов и увеличивая время отклика.

Заключение

В данной работе было проведено сравнительное исследование подходов HLS и HDL при программировании ПЛИС. Взяв в качестве примера фильтр Кэнни, предложено оптимизированное решение на основе СнК для обнаружения выделения контуров транспортных объектов на изображениях, а также проведены сравнения между HDL и HLS с точки зрения использования ресурсов и времени выполнения. Поскольку фильтр Кэнни, как и многие другие фильтры выделения контуров, является сверточным фильтром, это предоставляет возможность объективного сравнение сильных и слабых сторон каждого фильтра при выполнении обработки транспортных изображений.

Результаты показывают, что HDL-реализация фильтра Кэнни превосходит его HLS-версию, при рассмотрении их функционирования с позиции объема затрачиваемых ресурсов, т.е. использования логических таблиц (Lut), регистров (Reg), мультиплексоров (MUX) и блоков оперативной памяти (RAM), а также времени, необходимого для обработки изображений. Согласно этим результатам, подход HDL является оптимальным с точки зрения затрачиваемых ресурсов, времени отклика.

Литература

[1] Фахми Ш.С. Выделение контуров изображений морских судов / Ш.С. Фахми, Н.В. Шаталова, Я.А. Селиверстов, Е.С. Калинина, А.В. Иванов // Морские интеллектуальные технологии. 2019. Т.3, № 3(45). С. 132-143.

[2] Escobar F.A., Chang X., Valderrama C. Suitability Analysis of FPGAs for Heterogeneous Platforms in HPC // IEEE Transactions on Parallel and Distributed Systems. 2016. Vol. 27, № 2. pp. 600-612.

[3] Бухтеев А., Немудров В. Системы на кристалле. новые тенденции // Электроника: Наука, технология, бизнес. 2004. № 3 (53). С. 52-57.

[4] Шагурин И.В., Шалтырев В. Проектирование систем на кристалле на базе FPGA компании Xilinx // ChipNews. Инженерная микроэлектроника. 2005. № 10. С. 54-58.

[5] Martin G., Smith G. High-level synthesis: Past, present, and future // IEEE Design Test of Computers. 2009. Vol. 26, № 4. pp. 18-25.

[6] Harris D.M., Harris S.L. Digital Design and Computer Architecture (Second Edition). Morgan Kaufmann, 2013. 690 p.

[7] Chaithra.N.M., Reddy K.V.R.Implementation of Canny Edge Detection Algorithm on FPGA and displaying Image through VGA Interface // International Journal of Engineering and Advanced Technology. 2013. Vol. 2, Issue-6. pp. 243-247.

Hardware Implementation of Image Processing Application based on "System on Chip" Technology

S.S. Fahmi[1,2], I.G. Malygin[1,3], O.A. Korolev[1]

[1]Institute of transport problems N. S. Solomenko of the Russian Academy of Sciences, Saint Petersburg, korolev@iptran.ru

[2]Saint Petersburg Electrotechnical University "LETI", Saint Petersburg, shakeebf@mail.ru

[3]FSBEI HE “Saint-Petersburg State Fire Service University of EMERCOM of Russia”, Saint Petersburg, malygin_com@mail.ru

***Abstract* — Growing complexity of modern video information processing systems and the limited time frame for their design require the use of new development tools used in modern computer-aided design systems and based on programmable logic integrated circuits (FPGAs). Currently, in addition to the traditional hardware description languages (hardware description languages - HDL), high-level synthesis tools (High Level Synthesis - HLS) of microarchitectures are gaining more and more popularity, the advantage of which is to increase the level of abstraction in the development of**

various signal processing systems, in particular images in the field of transport. Despite the ease of design and testing compared to HDL, when using HLS, it is necessary to take into account the shortcomings that appear in the generation of hardware descriptions inherent in this approach. In the article, the example of designing the well-known Canny filter, which makes it possible to highlight the boundaries of transport objects in images when solving detection and recognition problems, a comparative study of two methods for synthesizing digital image processing devices: HLS and HDL is presented. The results of a comparison of two implementations of HDL and HLS, obtained on the basis of an estimate of computational resources and operation execution time, are presented. Since the Canny filter is used, like many other edge detection filters, is convolutional, this work allows evaluation of the strengths and weaknesses of any operator in the field of image processing. The study showed that, given the resources involved and response time, the HLS implementation performed better in terms of resource utilization and response time than the HDL approach.

***Keywords* — FPGA, Canny filter, HDL, HLS, configuration, system on chip.**

REFERENCES

[1] Fakhmi Sh.S. Vydeleniye konturov izobrazheniy morskikh sudov [Selection of contours of images of sea vessels] / Sh.S. Fakhmi, N.V. Shatalova, YA.A. Seliverstov, Ye.S. Kalinina, A.V. Ivanov // Morskiye intellektual'nyye tekhnologii. 2019. T.3, № 3(45). S. 132-143.

[2] Escobar F.A., Chang X., Valderrama C. Suitability Analysis of FPGAs for Heterogeneous Platforms in HPC // IEEE Transactions on Parallel and Distributed Systems. 2016. Vol. 27, № 2. pp. 600-612

[3] Bukhteyev A., Nemudrov V. Sistemy na kristalle. novyye tendentsii [Systems on a crystal. new trends] // Elektronika: Nauka, tekhnologiya, biznes. 2004. № 3 (53). S. 52-57.

[4] Shagurin I.B., Shaltyrev V. Proyektirovaniye sistem na kristalle na baze FPGA kompanii Xilinx [Designing systems on a chip based on Xilinx FPGA] // ChipNews. Inzhenernaya mikroelektronika. 2005. № 10. S. 54-58.

[5] Martin G., Smith G. High-level synthesis: Past, present, and future // IEEE Design Test of Computers. 2009. Vol. 26, № 4. pp. 18-25.

[6] Harris D.M., Harris S.L. Digital Design and Computer Architecture (Second Edition). Morgan Kaufmann, 2013. 690 p.

[7] Chaithra.N.M., Reddy K.V.R.Implementation of Canny Edge Detection Algorithm on FPGA and displaying Image through VGA Interface // International Journal of Engineering and Advanced Technology. 2013. Vol. 2, Issue-6. pp. 243-247.