ВОЗМОЖНОСТИ ПРИМЕНЕНИЯ СВОБОДНОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ РАЗРАБОТКИ СИСТЕМ НА КРИСТАЛЛЕ

А. И. Егоров, С. Н. Коянкин, А. Г. Кузякин

ФГУП «РФЯЦ-ВНИИЭФ», г. Саров Нижегородской обл.

Введение

Раньше, при меньшей степени интеграции элементов в интегральной микросхеме, одна микросхема реализовывала, как правило, функционал одного блока устройства. Сегодня технический прогресс шагнул далеко вперед, мировые фабрики осваивает технологический процесс 4 нм. При сегодняшней сверхбольшой степени интеграции появилась возможность объединения множества сложных функциональных (СФ) блоков на одном кристалле в целую систему [1]. Следствием этого должно быть понижение стоимости разработки устройства в целом, улучшение его массо-габаритных характеристик, повышение надёжности к воздействию внешних факторов. Таким образом, разработка систем на кристалле (СнК) – наше настоящее и будущее в проектировании электронных устройств.

Преимущества свободного программного обеспечения

Коммерческие (проприетарные) решения для проектирования СнК по большей части иностранного производства и находятся под запретом использования, в связи с чем в этой области возникает необходимость импортозамещения. На помощь в этом должно прийти свободное программное обеспечение.

Благодаря доступности исходных кодов свободных программ для изучения и модификации, имеется возможность воспользоваться наработанным опытом сообщества и применять существующие решения СПО для своих нужд, не тратя время на разработку аналогичных по функциональности программ, а только на изучение того, как работает уже разработанная, протестированная и отлаженная программа, и её внедрение в свой маршрут проектирования.

Структура СнК

Сегодня уже практически ни одно сложное устройство, имеющее в составе цифровую часть, невозможно представить без встроенного контроллера или процессорного ядра. Так, и в центре типовой СнК (рис. 1) находится процессор, размещённый на одном кристалле с периферийными устройствами, специализированными блоками, различными интерфейсами для связи с внешними объектами и внутренней памятью [2].

Так как СнК по структуре – набор СФ-блоков, представляемых в виде цифровых моделей, то для проектирования её составных частей возможно применить стандартный маршрут проектирования СБИС (рис. 2) [3].

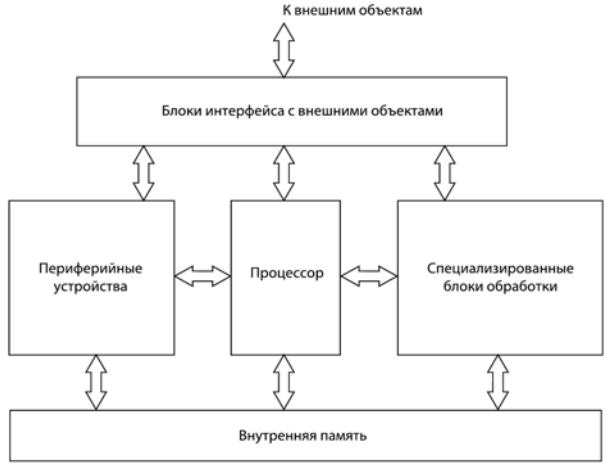

Рис. 1 Структура типовой СнК

Рис. 2. Стандартный маршрут проектирования СБИС

Маршрут разработки СФ-блока СнК

Разработка цифровой модели любого СФ-блока начинается с изучения спецификаций. Далее, разрабатывается функциональная модель на высокоуровневых языках описания аппаратуры VHDL или Verilog. После того, как модель готова, необходимо проверить её на соответствие заданным спецификациям. Данный этап называется верификацией и заключается в моделировании разработанного функционала. Для этого разрабатывается тестирующая программа – testbench, генерирующая набор тестовых векторов, подающих входные воздействия на схему и сравнивающая выходные реакции с эталонными. Верификация производится на каждом этапе проектирования, после каждого изменения или преобразования схемы. Таким образом, процесс разработки поэтапный и нелинейный. После верификации функциональной модели производится синтез схемы в технологическом базисе целевой ПЛИС, либо завода изготовителя заказной или полузаказной микросхемы на основе базового матричного кристалла (БМК). После обратной верификации и подтверждения того, что синтезированная схема эквивалентна разработанной функциональной модели, осуществляется этап размещения и топологического синтеза. Затем, снова верификация, и наконец – производство опытных образцов и их входной контроль.

Разрабатывать модель можно в любом текстовом редакторе. Свободных текстовых редакторов представлено великое множество. Существуют, например, программы, такие как Eclipse, Geany, которые помимо текстового редактора предоставляют целые среды разработки программного обеспечения, обладая такими функциями как: подсветка исходного кода с учетом синтаксиса используемого языка программирования, автозавершение слов, менеджер проектов, поддержка плагинов.

Свободные среды моделирования также представлены разнообразные, в зависимости от выбранного языка описания. Так, в случае разработки на VHDL – это GHDL или FreeHDL, Verilog – Icarus Verilog. В случае применения языка C++ при моделировании – это Verilator, в случае использования интерпретатора Python – PyHDL.

В нашем маршруте разработка моделей ведётся на языке Verilog, и поэтому для верификации разработанной модели, её описание преобразуется в машинный код с помощью свободного компилятора языка Verilog – Icarus Verilog. Выгружаемый в процессе моделирования файл временных диаграмм .vcd можно открыть свободным просмотрщиком временных диаграмм цифровых сигналов GTKWave. Кроме того, тестирующую программу testbench можно разработать таким образом, что в ходе моделирования в случае отклонений будут выдаваться сообщения о дефектах на соответствующих временных участках. Такое построение тестирующей программы существенно сокращает время поиска неисправности и отладки модели.

Преобразование функциональной модели в структурную электрическую схему проводится свободной программой логического синтеза Yosys.

Дальнейшая работа с СнК

После того, как готовы разработанные специализированные СФ-блоки, необходимо собрать и сконфигурировать СнК в целом. Для этого в нашем маршруте применяется свободный генератор-конфигуртотар СнК FuseSoC.

FuseSoC – перспективный генератор описаний СнК, который предоставляет ядра с различными системами команд, например OpenRISC 1000 и RISC-V. Он также управляет доступными ядрами периферии и позволяет легко настраивать и генерировать разрабатываемую СнК. По своей сути он является менеджером пакетов и набором инструментов для сборки HDL-кода. Основная цель FuseSoC – увеличение объёма повторного использования СФ-блоков, при этом он является вспомогательным средством для создания, построения и моделирования систем на кристалле [4].

После того, как СнК сконфигурирована, сгенерирована, верифицирована её аппаратная часть, становится возможным применять то программное обеспечение, для которого и была разработана СнК. Теперь его можно загрузить во внутреннюю или внешнюю память СнК и заставить работать. Для этого существует отдельный маршрут.

Работа с ПО для СнК начинается также с текстового редактора исходного кода. Для каждого процессора существует свой toolchain – набор инструментов, который по цепочке, максимально автоматизированно (с помощью make-файла или подобного механизма) преобразует компилятором (например, GCC) исходный код программы на языке высокого уровня, например C++, в машинный код, понимаемый процессором для исполнения. Помимо компилятора, в работе необходим отладчик (GDB) для отладки компилируемого кода, эмулятор (QEMU) для моделирования работы программы до ее загрузки в микросхему, и непосредственно загрузчик (OpenOCD) машинного кода в память СнК. После этого, с помощью GDB доступна отладка в микросхеме.

Заключение

В результате проделанной работы по изучению возможностей применения СПО для разработки СнК были изучены практически все доступные решения в части проектирования СБИС. Были отобраны самые стабильные и полнофункциональные средства, из которых сформирован маршрут проектирования СФ-блоков СнК.

Маршрут был отработан на тестовой сконфигурированной СнК, включающей открытое программное ядро, основанное на архитектуре RISC-V, системную шину Wishbone и интерфейс UART на периферии.

Сформированный маршрут позволил быть независимыми от применяемой в разрабатываемой СнК архитектуры процессора, от фирмы-разработчика заказной микросхемы и системы автоматизированного проектирования. Маршрут привёл к ускорению процесса переноса проекта при смене производителя микросхемы. Средства свободного программного обеспечения, сформировавшие маршрут, показали, что могут применяться при разработке СнК специализированных вычислителей из состава перспективных изделий систем автоматики.

Литература

1. Немудров В. Г., Мартин Г. Системы-на-кристалле. Проектирование и развитие. М.: Техносфера, 2004.

2. Фельдман В. М. Направления развития архитектуры отечественных микропроцессорных линий Эльбрус и МЦСТ-R. ЗАО МЦСТ. Тезисы доклада. 28.10.2010.

3. Проектирование СБИС типа «Система на кристалле». Маршрут проектирования. Синтез схемы. Часть 1 [Электронный ресурс] // «Russian Electronics»: [сайт]. URL: https://russianelectronics.ru/developer-r/ review/ 2189 .

4. FuseSoC is a package manager and a set of build tools for FPGA/ASIC development [Электронный ресурс] // «GitHub»: [сайт]. URL: https://github.com/olofk/fusesoc.