2. Taizo Suzuki and Masaaki Ikehara, Realization of lossless-to-lossy image coding compatible with JPEG standard by direct-lifting of DCT-IDCT, Proceedings of 2010 IEEE 17th International Conference on Image Processing September 26-29, 2010, Hong Kong.

3. *Taizo Suzuki, Masaaki Ikehara,* Integer DCT Based on Direct-Lifting of DCT-IDCT for Lossless-to-Lossy Image Coding IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 19, NO. 11, NOVEMBER 2010, PP. 2958-2965.

4. *Петровский Ал.А.,. Станкевич А.В, Петровский А.А* .“Быстрое прототипирование систем мультимедиа от прототипа”, Глава 6, стр.173-207, Минск “Бестпринт”, 2011.

5. *Ключеня В.В., Петровский А.А.* “Выбор оптимальной реализации структурного решения процессора ДКП на распределенной арифметике”, Доклады БГУИР 2010 №7, С.66-75.

УДК 004.052

СОЗДАНИЕ СРЕДЫ ВЕРИФИКАЦИИ VHDL МОДЕЛИ МИКРОСХЕМЫ БЕСКОНТАКТНОЙ ИДЕНТИФИКАЦИИ

Ю.Ю. ЛАНКЕВИЧ

Белорусский государственный университет информатики и радиоэлектроники
ул. П. Бровки, 6, г. Минск, 220013, Республика Беларусь
yurafreedom18@gmail.com

Разработаны некоторые компоненты среды верификации в рамках OVM (Open Verification Methodology). Выявлены особенности использования данной методологии на конкретном примере тестового окружения для устройства бесконтактной идентификации.

Ключевые слова: Open Verification Methodology, функциональная верификация, тестовое окружение.

Современные системы автоматизированного проектирования цифровых устройств на базе программируемых логических интегральных схем (ПЛИС) и заказных цифровых сверхбольших интегральных схем (СБИС) позволяют провести сквозной процесс проектирования – от описаний алгоритмов функционирования устройств до получения файлов конфигурирования ПЛИС либо до описаний топологии заказных СБИС. На первый план при проектировании в настоящее время выступают проблемы верификации исходных спецификаций, представленных на высокоуровневых языках проектирования, например, на языке VHDL. Для такой верификации разрабатываются различные подходы, в том числе и функциональная верификация.

Цель работы состояла в разработке методики функциональной верификации исходной VHDL модели микросхемы радиочастотной бесконтактной идентификации Atmel ATA5577C [1] в рамках открытой методологии верификации (Open Verification Methodology – OVM). OVM представляет собой библиотеку классов и макросов на языке SystemVerilog, используемых для построения тестового окружения [2]. Методика функциональной верификации включает написание тестового плана, выполнение функционального покрытия и покрытия кода. С помощью средств OVM было разработано тестовое окружение VHDL модели микросхемы.

Atmel ATA5577C является микросхемой для бесконтактной радиочастотной идентификации с функцией чтения/записи, используемой в метках с частой функционирования 125–134 кГц. Микросхема содержит электрически стираемое перепрограммируемое постоянное запоминающее устройство объемом 343 бита, состоящее из 11 блоков по 33 бита (32 бита данных и один бит блокировки). Два блока являются конфигурационными и содержат более 20 параметров, задающих режим работы микросхемы. Задавая различные значения параметров можно выбрать четыре кодировки для входных команд, 11 (в базовом режиме) либо девять (в расширенном режиме) различных типа кодирования выходных данных, передаваемых меткой, восемь (в базовом режиме) либо 64 (в расширенном режиме) скорости передачи данных, число передаваемых блоков памяти, четыре режима ускоренного приема команд и др. Микросхема поддерживает команды для чтения/записи с паролем и без, обеспечения процедуры антиколлизии и др., всего семь команд.

При верификации проверялись все возможные команды, поданные с помощью всех возможных протоколов передачи данных и все возможные варианты кодирования ответного сообщения.

Подход написания тестового окружения, с использованием OVM, основан на объектно-ориентированном программировании. В иерархии тестового окружения используются модули и классы. В рамках данного подхода, были описаны отдельные компоненты, отвечающие за определённые функции, такие как: формирование тестовых сигналов (sequencer), взаимодействие с тестируемым устройством (driver), отслеживание состояния интерфейсов тестируемого устройства (monitor), сбор функционального покрытия (coverage), сравнение поведения тестируемой VHDL модели и эталонной модели (scoreboard). Описано также взаимодействие между компонентами на различных уровнях иерархии тестового окружения [2].

На рисунке 1 представлена структура тестового окружения. Модуль top верхнего уровня включает модель тестируемого устройства (DUT) и объект класса test, который, в свою очередь, на более низких уровнях иерархии включает в себя экземпляры таких классов, как environment, scoreboard, agent и другие.

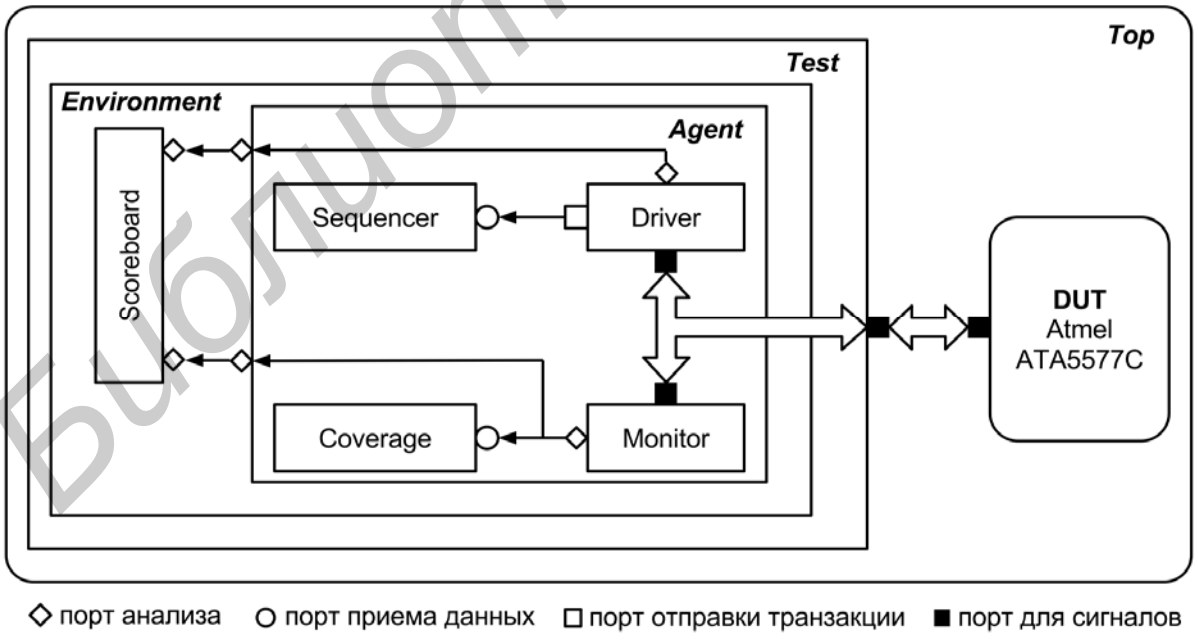

Рис. 1. Структура тестового окружения