го из *Кн* приводил к невозможности полноценного функционирования целой группы *ОН*. Вместе с удалением из схемы *Кн* такая необходимость пропала, что существенно облегчает разработку алгоритмов самомаршрутизации сигналов в нейронной сети. Это достижение в значительной мере оправдывает увеличение аппаратных затрат, поэтому двухслойная структура является более перспективной для дальнейших разработок.

Авторы считают, что в данной работе новыми являются следующие положения и результаты: модернизирована существовавшая структурная схема трёхслойной нейронной сети, в которой удалось устранить третий слой за счёт увеличения количества связей между вторым скрытым и выходным слоями, но при этом значительно увеличены возможности по обеспечению отказоустойчивости, что облегчает разработку алгоритмов самомаршрутизации сигналов, что в свою очередь, в будущем может существенно уменьшить аппаратные затраты на реализацию нейронной сети с двумя скрытыми слоями по сравнению с трёхслойной структурой.

В настоящее время решаются следующие задачи: проблема отказоустойчивости коммутаторов с возможностью использовать *ОН*, подсоединенных к отказавшему *Км*, разработка и моделирование внутренней структуры *ОН* как части, отвечающей за маршрутизацию сигналов, так и измерительной и диагностической частей, разработка распределенной системы управления работой *Км* и *ОН*.

Литература

1. *Матушкин Н.Н., Южаков А.А.* Измерительные преобразователи на основе потоковой динамической архитектуры // Известия вузов. Приборостроение. 1994. № 1. С. 16-21.

2. *Посягин А.И., Южаков А.А.* Разработка аналого-цифрового преобразователя на основе нейронной сети // Электротехника. 2012. № 11. С. 18-24.

3. *Посягин А.И., Южаков А.А.* Самомаршрутизирующийся аналого-цифровой преобразователь на основе двухслойной нейронной сети // Нейрокомпьютеры: разработка, применение. 2013. № 11. С. 076-081

Two-layer neural network of self-routing analog-to-digital converter review

***Anton Igorevich Posyagin**, Assistant, Chair of Automatics and Telemechanics*
***Alexandr Anatolyevich Yuzhakov**, Doctor of Technical Sciences, Professor*
Head of Chair of Automatics and Telemechanics
Perm National Research Polytechnic University

The paper deals with upgraded neural network scheme of self-routing analog-to-digital converter. The analysis of the results of two-layer and three-layer neural network is produced and perspectives of self-routing analog-to-digital converter based on neural network are discussed.

Keywords: analog-to-digital converter, self-routing, neural network, fault-tolerance.

УДК 681.396.6

UVM EXPRESS – УПРОЩЕННАЯ МЕТОДИКА ВНЕДРЕНИЯ UVM

***Иван Витальевич Селиванов**, млад. науч.сотр.*
Тел.: (495) 129 8736, e-mail: shubin@magn.ru
***Николай Юрьевич Шубин**, канд.физ.-мат.наук, зав. сект.*
Тел.: (495) 129 87 36, e-mail: shubin@magn.ru
Научно-исследовательский институт системных исследований
Российской академии наук НИИСИ РАН
http://www.niisi.ru

UVM - это прогрессивная методика функциональной верификации проекта, позволяющая стандартизировать, упростить и ускорить этот процесс. Однако для ее полноценного внедрения требуется значительное время, что зачастую служит причиной для отказа от ее использования. В данной статье рассматривается методика пошагового внедрения UVM, дающая положительные результаты, начиная с первого этапа.

Ключевые слова: SystemVerilog, универсальная методология верификации, тестбенч, функциональное покрытие

Введение

Согласно закону Мура, количество транзисторов, размещаемых на кристалле ИС, удваивается каждые 2 года. Но сегодня мы можем видеть, что растет не просто объем проекта, увеличивается его сложность. По статистике, более половины современных чипов содержит как минимум два встроенных процессора; для подавляющего большинства актуальна проблема контроля энергопотребления - отключение неиспользованных блоков, множество разных синхросигналов, сложные протоколы взаимодействия между блоками, и т. д. В проектах также растет процент чужого кода - покупных блоков. Как итог, больше половины общего времени разработки занимает не написание кода, а его верификация. Количество людей, занимающихся верификацией проекта, уже сравнялось с количеством его разработчиков, и тем не менее, не более трети проектов работают с первого раза. 2/3 проектов не укладываются в отведенные сроки.

И.В. Селиванов

Н.Ю. Шубин

Для решения создавшейся проблемы потребовались новые стандарты. Был разработан язык SystemVerilog, затем написана библиотека OVM, и на ее основе фирма Accelera разработала Универсальную Методологию Верификации (UVM). Об актуальности данного подхода говорит то, что SystemVerilog - это самый быстроразвивающийся язык за всю историю САПР. 89% больших проектов используют его для написания тестов.

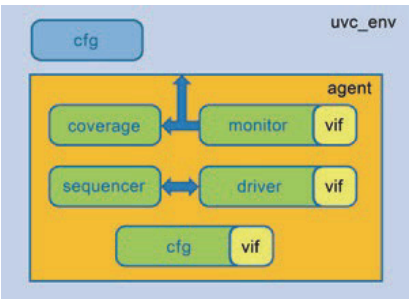


Рис. 1. Структура универсального компонента

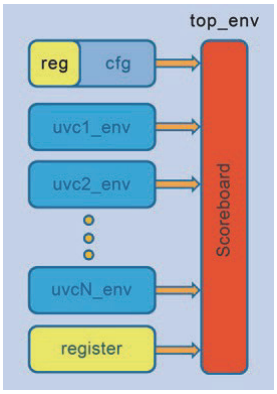


Рис. 2. Среда верхнего уровня

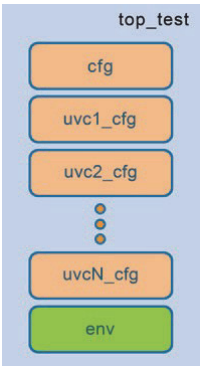


Рис. 3. Структура теста

Ключевые технологии, применяющиеся в UVM - это объектно-ориентированное программирование и моделирование на уровне транзакций. Они обеспечивают этому подходу высокую гибкость, увеличивая уровень абстракции и позволяя повторное использование блоков в режиме plug & play. Но они являются абсолютно новыми для разработчиков, а также для большинства специалистов по верификации. Это, возможно, главная причина, по которой считается, что UVM пригоден только для больших проектов заказных микросхем. Но опыт показывает, что это не так. UVM обеспечивает лучшие предпосылки для того, чтобы быть внедрённым независимо от сложности и аппаратной реализации проекта. Область его применения не ограничивается заказными мик-

росхемами, но также может распространяться и на ПЛИС. Вопрос в том, как внедрить UVM максимально эффективно.

Библиотеку UVM можно рассматривать как большой, но очень хорошо структурированный конструктор для генерации тестовых схем (тестбенчей), а также для реализации последовательностей и тестов. Руководство пользователя UVM и такие документы, как UVM Cookbook, содержат правила, рекомендации и примеры использования отдельных компонентов в библиотеке UVM.

Архитектура UVM тестбенча

Стандартный UVM тестбенч состоит из набора универсальных компонентов, по одному для каждого функционального интерфейса тестируемого устройства. Все компоненты состоят из агента и его конфигурации. Каждый агент состоит из 5 объектов - драйвер, генератор последовательностей, монитор, блок оценки покрытия и конфигурация (см. рис. 1).

Все универсальные компоненты объединяются в среду верхнего уровня, которая также может содержать информационное табло - некоторую структуру для быстрой оценки результатов верификации, а также модель регистра (см. рис. 2).

Тест содержит в себе среду верхнего уровня, а также все конфигурации (см. рис. 3). Он передает объекты конфигурации, такие как виртуальные интерфейсы, по иерархии проекта непосредственно в те блоки, где они нужны.

При описании такой структуры на SystemVerilog может возникнуть множество синтаксических ошибок, а также ошибок в реализации моделирования на уровне транзакций. Это снижает интерес к данному подходу, а также усложняет прогнозирование затрат усилий и времени на написание UVM тестбенча. Все это может являться причиной для отказа от внедрения UVM, особенно для проектов на ПЛИС. Устранение этого барьера сделает UVM доступным для более широкого ряда компаний и проектов.

В данной статье описывается подход, помогающий внедрить UVM для любых проектов, больших или маленьких, и любых реализаций, ПЛИС или ASIC.

UVM Express - это набор техник, стилей написания кода и применений UVM, направленных на увеличение продуктивности функциональной верификации. Техники включают в себя повышение уровня абстракции тестов, написание тестов с использованием функциональных моделей шин, добавление функционального покрытия и генерации псевдослучайных последовательностей импульсов.

UVM это библиотека классов, которая позволяет использовать возможности для верификации, которые предоставляет SystemVerilog. Библиотека классов UVM содержит мощные конструкции для верификации, включая модуль синхронизации, систему отчетов, конфигурационную базу данных, специальный объектный класс для переопределения типов и возможность писать тесты в виде последовательностей (или вызовов функций). Некоторые команды, занимающиеся верификацией, практически без проблем переходят на UVM, но обычно это случается когда уже есть опыт в VMM, AVM или OVM. Чтобы добиться успеха с SystemVerilog и UVM, нужно инвестировать время и деньги в обучение. Часто процесс внедрения можно сократить, наняв консультантов.

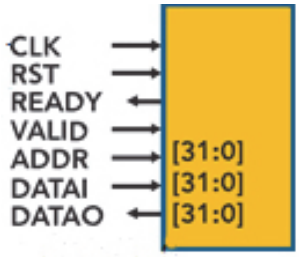
У большинства команд, однако, время и бюджет ограничены, и это не позволяет полностью освоить UVM, или освоить его в приемлемые сроки. В таких командах обычно не хватает людей и финансирования, и они слишком загружены. Это как раз те люди, которым должен помочь UVM, но цена внедрения слишком высока.

UVM Express позволяет сделать первый шаг по внедрению UVM. UVM Express - это способ создать свою среду для тестирования, повысить уровень абстракции, проверить качество тестов и способ подумать над написанием своих тестов. Каждый шаг методологии UVM Express это часть инфраструктуры для верификации, которую можно использовать многократно. Этапы UVM Express поз-

воляют последовательно освоить методологию UVM, при этом на каждом этапе получая результаты и увеличивая продуктивность верификации. UVM Express не заменяет UVM, но обеспечивает его постепенное внедрение. Это тот же UVM, просто структурированный таким способом, который обеспечивает постепенное внедрение и экономическое обоснование на каждом этапе.

UVM Express: написание тестов при помощи функциональной модели шин

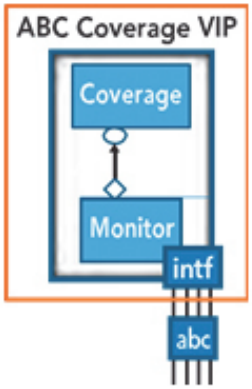
Первый уровень UVM Express это функциональная модель шины. Функциональная модель шины управляет сигналами и предоставляет тестбенчу интерфейс для доступа к ним. Например, модель может содержать связи шины ‘abc’, и метод ‘read(address, data)’, которая при вызове будет выполнять чтение из шины и возвращать прочитанные данные (см. код ниже).



На выполнение метода ‘read’ требуется время. Все тесты пишутся в терминах функциональных моделей шин, и сопутствующих функций, которые они предоставляют. Запрещается прямое взаимодействие с пинами.

UVM Express: Функциональное покрытие

Второй уровень UVM Express добавляет агента, оценивающего функциональное покрытие к тестам с функциональными моделями шин. Этот агент «сидит» на шине, отслеживая покрытие, создавая транзакции и публикуя их для подписчиков. Типичный подписчик это сборщик покрытия. Каждая публикуемая транзакция учитывается конструкциями covergroup языка SystemVerilog в агенте. Обычно сборщик покрытия пишется как UVM агент и помещается в пакет (сборку). Пользователям этого UVM агента не нужно напрямую взаимодействовать с UVM, или понимать, как устроен агент. Им нужно только понимать опции для конфигурирования данного агента.



UVM агент просто создается и подсоединяется к шине. BFM это функциональная модель шины. Это способ, которым тест взаимодействует с сигналами. Функциональная модель шины реализуется в видеинтерфейса SystemVerilog (см. пример кода ниже), имеющего сигналы (связи) и методы для манипулирования сигналами. Методы функциональной модели шины представляют собой типы транзакций, поддерживаемых
интерфейсом, например, read(), write(), burst_read() и burst_write(). Чтобы создать тесты, вызываются методы функциональной модели, но все обращения непосредственно к сигналам происходят только внутри модели шины и скрыты от пользователя.

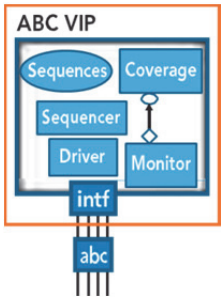
```
interface abc_if(input wire CLK);
reg RST;
reg VALID; reg READY;

reg RW;
reg BURST;
reg [31:0] ADDR; reg [31:0] DATAI; wire[31:0]
DATAO; task      reset();
task read( bit[31:0] addr, output bit[31:0] data);
task write( bit[31:0] addr, input  bit[31:0] data);
task burst_read( bit[31:0] addr, output bit[31:0] data[4]);
task burst_write( bit[31:0] addr, input  bit[31:0] data[4]);
task monitor(output bit transaction_ok,
output bit rw,
output bit[31:0] addr, output bit[31:0] data[4], output
bit burst); endinterface
```

Функциональное покрытие - это способ оценить, насколько полно протестировано соответствие работы устройства его спецификации. Например, при использовании конструкции covergroup из SystemVerilog, мы можем доказать, что функционал устрой-

ства был задействован полностью, и что во всех случаях оно выдавало правильные результаты. Память может быть прочитана из минимального и максимального адресов, а также из нескольких адресов между ними. Каждая операция чтения должна возвращать корректные значения. Чтение из неверного адреса должно быть обработано корректно. Все входные (адреса для чтения) и выходные (считанные данные) учитываются, и могут быть использованы для описания правильного функционирования.

UVM Express: псевдослучайные импульсы

Третий уровень UVM Express заменяет управляемый пользователем тест на псевдослучайные входные воздействия. Эти два метода могут дополнять друг друга, но на данном, третьем этапе UVM Express, это не предусмотрено. На этом уровне добавляется UVM агент, который может генерировать воздействия и оценивать покрытие. Этот агент используется так же, как и агент из второго уровня - он создается и подключается. После того, как он был создан, подключен и запущен, он начинает запускать в шине транзакции – это генератор траффика. Транзакции генерируются при помощи встроенной в UVM агент функции рандомизации – обычно, последовательности и элементы последовательностей с параметрами. Пользователю такого UVM агента не нужно напрямую взаимодействовать с UVM или с агентом. UVM агент является самодостаточным инструментом для генерации импульсов. Пользователь должен знать, какие у агента есть настройки, но ему не обязательно понимать подробности низкоуровневого взаимодействия. Под псевдослучайными импульсами в данном случае понимаются импульсы, описанные при помощи параметров SystemVerilog.



Параметры рандомизируются, чтобы получить новый набор импульсов. Каждая рандомизация начинается с исходного значения. Серия импульсов может начинаться с конкретного значения, и затем продолжаться. После завершения последовательности, можно взять другой параметр с таким же или другими исходными значениями, чтобы получить другую последовательность импульсов.

Заключение

Авторы считают, что в данной работе новыми являются следующие положения и результаты: впервые рассмотрена методика пошагового внедрения методологии UVM, позволяющая начать использование этой прогрессивной технологии без существенных временных и финансовых затрат на обучение специалистов. Это позволит популяризировать UVM среди людей, не являвшихся ранее его целевой аудиторией - небольших групп разработчиков устройств на ПЛИС.

Литература

1. Making it Easy to Deploy the UVM, by Dr. Christoph Sühnel, frobas GmbH. URL: https://verificationacademy.com/verifica-tion-horizons/june-2013-volume-9-issue-2#article9

2. UVM Cookbook. URL: https://verificationacademy.com/cookbook/uvm

UVM Express – simplified technique for UVM introduction

***Ivan Vitalyevich Selivanov**, Junior researcher*
***Nikolay Yurevich Shubin** , **C**andidate of Physical and Mathematical Sciences, Manager of sector*
Research Institute of System Researches of Russian Academy of Sciences

UVM is an innovative methodology of functional verification allowing to standardize, simplify and speed up the verification process. However its full implementing takes a significant amount of time. This is what prevents its wider use. This article describes a step-by-step methodology of implementing the UVM which produces value since the first step.

Keywords: SystemVerilog, universal verification methodology (UVM), testbench, functional coverage.