Применение средств Mentor Graphics 0-In и Mentor Graphics Questa для верификации проекта с использованием формальных методов

Алексей Рабоволюк (Москва)

В статье рассмотрен метод расширенной функциональной верификации проекта, основанный на формальной модели. Данный метод позволяет «выловить» такие ошибки, которые традиционными методами обнаружить очень сложно, а в некоторых случаях и невозможно. В статье приводится пример очень простого, но не поддающегося верификации стандартными методами проекта. Также упоминаются библиотеки формальных моделей, используя которые, разработчик может обнаружить такие ошибки, о возможности существования которых он и не подозревал.

В настоящее время размер цифровых проектов заказных СБИС постоянно растёт, и сейчас он уже достиг таких величин, при которых для верификации необходимо разработать тестбенч, сопоставимый по размерам и сложности с самим проектом, а иногда и многократно превосходящий его.

В таких условиях очень высока вероятность того, что тестбенч всё же «пропустит» хотя бы одну ошибку, которая выявляется только после физической реализации проекта, что приведёт к многомиллионным убыткам. Однако усложнять тестбенч до бесконечности тоже не представляется возможным ввиду ограниченности времени превращения идеи в рыночный продукт (time-to-market). Таким образом, разработчик тестбенча должен точно определить момент, когда необходимо завершить верификацию, и это решение ложится на разработчика огромным грузом ответственности. Недоработанный тестбенч может пропустить ошибку, что приведёт к большим дополнительным расходам и как следствие – к удорожанию продукта и, соответственно, потере его конкурентоспособности, а с другой стороны, затягивание с разработкой тестбенча приведёт к увеличению time-to-market и убытки будут не меньше, чем в первом случае. Полностью отказаться от человеческого фактора невозможно, но можно существенно упростить задачу, применяя при верификации формальные методы и оценивая завершённость тестбенча на основе некоторых количественных характеристик проекта.

Созданный тестбенч должен ответить на два вопроса: «Это работает?» и «Всё ли протестировано?» [1]. Ответ на первый вопрос получить относительно просто. В тестбенче можно реализовать те же функции, что и в основном проекте (далее основой проект будет обозначаться аббревиатурой DUT – Design Under Test), и, сравнивая результаты, можно точно определить, правильно ли работает проект. Ответ на второй вопрос получить гораздо сложнее, и он всецело зависит от таланта разработчика.

Одним из стандартных инструментов, помогающих оценить степень завершённости тестбенча, является анализ полноты покрытия кода. Среда верификации Questa предоставляет для этого широчайшие возможности, позволяя анализировать полноту покрытия по множеству параметров и различными методами. Анализ полноты покрытия позволяет также проследить распределение задействованности фрагментов кода. Например, может возникнуть такая ситуация, при которой используется в работе только 20% кода, в то время как остальные 80% «простаивают».

Существует несколько типов анализаторов полноты покрытия, реализованных в среде Questa:

- Code coverage: branch, condition, expression, statement, and toggle;
- Finite State Machine (FSM) coverage;
- SystemVerilog covergroup coverage;
- SVA and PSL directive coverage;
- Assertion data (pass, fail, attempt counts);
- User-defined data;
- Test item data.

Некоторые анализаторы используются уже давно и хорошо известны разработчикам. Например, Code coverage подсчитывает, сколько раз в процессе моделирования была выполнена каждая строка исходника. Но некоторые, например, SVA and PSL directive coverage, Assertion data (pass, fail, attempt counts), SystemVerilog covergroup coverage, являются относительно новыми и заслуживают отдельного внимания.

Относительно новым подходом к верификации является создание инженером формальной модели проекта одновременно с разработкой самого проекта. Формальная модель описывается в виде ассертов (утверждений) и может быть реализована несколькими способами: в виде комментариев в исходном коде, с помощью конструкций SystemVerilog или при помощи отдельного файла на языке PSL.

Пример реализации на SystemVerilog [2]:

```
property abc(a,b,c);
disable iff (a==2) not @clk (b
##1 c);
endproperty
env_prop: assert property
(abc(rst,in1,in2)) pass_stat else
fail_stat
```

Этими ассертами задаются правила работы проекта, и если в процессе моделирования было нарушено хотя бы одно правило, то это значит, что в проекте присутствует ошибка. Однако ассерты могут выступать не только в виде «маркеров ошибок», но и в виде формальных признаков протестированности проекта. Рассмотрим простой пример. При разработке калькулятора с двумя действиями – «плюс» и «минус» – необходимо протестировать две функциональности проекта: сложение и вычитание. Для отладки данного проекта необходимо проверить правильность работы сначала одного действия, а затем другого при всех крайних значениях входных данных и при нескольких промежуточных значениях. Традиционным подходом в данном случае является создание тестера, генерирующего тестовый вектор, который бы сначала «включал» функцию сложения и подавал все необходимые варианты входных данных, а затем то же самое делал при вычитании. Однако существует вероятность того, что при создании тестового вектора разработчик пропустит какое-либо пограничное значение. Гораздо эффективнее было бы использовать другой подход: параллельно с разработкой проекта задавать и признаки протестированности функций при помощи всё тех же ассертов (в данном примере, разработав сложную систему ассертов, отлавливающую ситуацию «протестированности» всех пограничных значений входных данных для функции сложения). Эту же систему ассертов можно использовать и для контроля протестированности вычитания как библиотечный элемент. В этом случае этой системе ассертов присваивается некое имя, и среда Questa позволяет обращаться посредством этого имени к информации о завершённости тестирования каждой из функций. Теперь остаётся только «включить» функцию сложения и подавать на вход значения, создаваемые генератором случайных чисел. Как только будет отловлен признак завершённости тестирования функции сложения, надо «включить» вычитание и опять подавать случайные последовательности на вход. Даже в данном очень простом примере видно, что разработать тестер, генерирующий случайные последовательности, намного проще, чем держать в голове все возможные комбинации пограничных значений и последовательно забивать их в тестер. И чем сложнее проект, тем разница в сложности тестера будет больше.

В Questa реализован механизм мониторинга ассертов, т.е. они все сведены в набор таблиц, в которых для каждого ассерта отображается различная статистическая информация.

Однако даже стопроцентное покрытие не гарантирует отсутствие ошибок. Допустим для примера, что ошибка возникает только в случае специфического сочетания внутренних состояний и при полном заполнении определённого блока FIFO (очереди). Чтобы проверить эту ситуацию с помощью моделирования, надо, во-первых, провести все циклы, необходимые для заполнения FIFO; во-вторых, как-то предусмотреть создание необходимого сочетания внутренних состояний. Вероятность того, что подобное сочетание возникнет в результате случайной или псевдослучайной генерации тестов, очень низкая. Чтобы возникло нужное сочетание, может потребоваться слишком большой набор тестов. В данном случае поможет новая стратегия верификации, предлагаемая средой 0-In.

Чтобы понять, каким именно образом 0-In сможет отловить описанную выше ошибку, обратим внимание на цель, стоящую перед разработчиком тестбенча. Традиционный подход заключается в том, чтобы найти такую последовательность входных сигналов, которая вызовет несоответствие результатов работы DUT результатам эталонной модели. При этом остаётся риск существования внутренней ошибки, не приводящей в данном конкретном тестбенче к несоответствию результатов моделей. Решением может служить постановка дополнительной задачи: найти такую комбинацию входных воздействий, которая, не нарушая утверждений для входных портов, приводила бы к нарушению хотя бы одного утверждения внутри проекта. Идеальным решением дополнительной задачи является полный перебор всех возможных входных значений. Предположим, проект содержит два бита на входе и один двухбитный регистр внутри, т.е. проект может находиться в одном из четырёх состояний. Полный перебор такого проекта будет выглядеть следующим образом: нужно перебрать все возможные варианты входных сигналов и посмотреть, не нарушается ли какое-либо утверждение на первом такте. Всего получится четыре проверки правильности работы проекта на первом такте. Далее необходимо для каждого значения входных сигналов на первом такте перебрать все возможные варианты входных сигналов на втором такте. Количество проверок возрастает до $2 \times 4 = 16$. На сколько-нибудь большом проекте скорость возрастания количества проверок на каждом следующем такте возрастает гораздо быстрее. Таким образом, прямой перебор ограничен некоторым количеством тактов, при котором время моделирования останется в «разумных пределах». Назовём это число тактов «глубиной поиска». Выполнять такой перебор на некую глубину может среда 0-In. В рассмотренном примере проект имеет четыре состояния (двухбитный регистр), следовательно, при глубине поиска, равной 4, можно на 100% гарантировать правильность работы проекта при условии правильно созданной формальной модели. На реальном проекте состояний гораздо больше, и прямой перебор с приемлемой глубиной поиска не позволит достичь абсолютного большинства состояний. Решением является объединение усилий генератора тестов, созданного разработчиком, и прямого перебора с приемлемой глубиной поиска. Результатом такого объединения будет существенное повышение качества верификации по сравнению с использованием только одного генератора тестов.

В итоге получается следующая стратегия верификации: генератор тестов, подавая на каждом следующем такте новые входные значения, «проводит» проект по тем состояниям, по которым задумал разработчик, а 0-In при этом на каждом такте приостанавливает «основное» моделирование и выполняет как бы «виртуальное моделирование», заключающееся в полном пере-

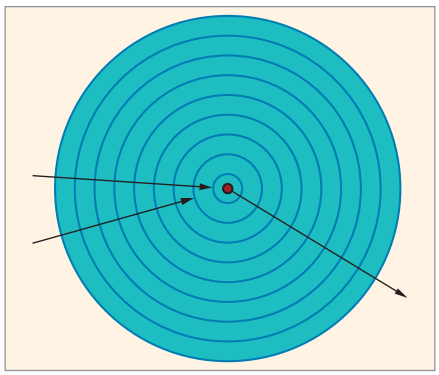

Рис. 1. Множество состояний кубика Рубика

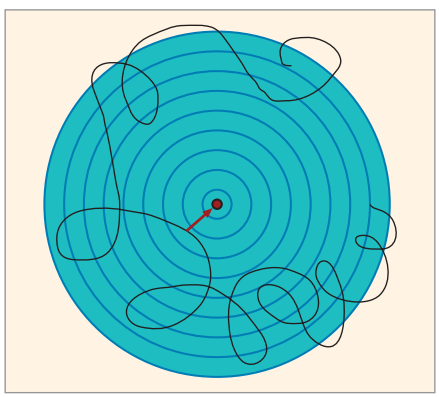

Рис. 2. Количество оставшихся поворотов до сборки

боре допустимых входных данных на некоторую глубину. Целью этого «виртуального» моделирования является поиск такой последовательности комбинаций входных сигналов (т.е. поиск такого теста), которая, не нарушая ни одного ассерта для входных портов, нарушала бы внутренний ассерт. Назовём найденную последовательность комбинаций входных сигналов, не нарушающих ограничений (ассертов) на вход, но приводящую к нарушению конкретного ограничения (ассерта) внутри проекта, термином counterexample. Если в процессе «виртуального» моделирования найден counterexample, то в зависимости от стратегии верификации, например, может произойти остановка моделирования и вывод отчёта о найденном counterexample. В противном случае моделирование будет продолжено с того состояния, в котором было приостановлено «основное» моделирование.

Рассмотрим пример: кубик Рубика, у которого на каждой стороне по четыре квадрата и каждый квадрат может принимать одно из шести значений цвета (по числу сторон кубика):

```
`define WHITE 0
`define BLUE 1
`define RED 2
`define GREEN 3
`define ORANGE 4
`define YELLOW 5
```

На входе кубика присутствуют две шины: face – выборка стороны кубика и move – направление, в которое эту сторону нужно вращать:

```
input [2:0] face;
input [1:0] move;
```

Задача: создать тестбенч, который бы собирал этот кубик из любого состояния. Если не знать, по каким принципам собирается кубик Рубика, то решение только одно – подавать на вход случайные значения в надежде, что через некоторое время он сам соберётся:

```
face = $random(seed_count) %
3'd6;
move = ($random(seed_count) %
2'd3) + 1;
```

Представим множество состояний кубика в следующем виде: в центре точка – единственное искомое состояние, при котором он собран. Вокруг центра – окружность, символизирующая множество состояний кубика, при которых до сборки остаётся сделать один поворот. Далее – окружность состояний, в которых до сборки остаётся два поворота и т.д. (рис. 1).

Тогда в нашем примере траектория движения кубика будет примерно такой, как на рисунке 2.

Время ожидания того момента, когда траектория попадёт в центр, может оказаться неприемлемо большим. Решение – на каждом такте работы тестбенча выполнять прямой перебор с определённой глубиной. Для этого потребуется формальными методами описать искомое состояние.

Например, такая формальная модель, описанная на языке Verilog совместно с внутренним языком описания формальной модели, 0-In, приведена на сайте журнала.

В результате получится, что если тестер случайно подвёл проект на расстояние, равное глубине поиска (показано красной стрелкой), – 0-In обязательно найдёт искомое состояние. При данном подходе время поиска является приемлемым.

На практике процесс верификации выглядит следующим образом: анализируя все внутренние ассерты поочерёдно, 0-In пытается решить две задачи – или найти counterexample, или доказать ассерт, т.е. доказать, что принципиально не существует такого теста, который бы, не нарушая ассертов на входных портах, приводил бы к нарушению анализируемого ассерта. Таким образом, в процессе верификации у каждого ассерта есть один из трёх признаков:

- найден counterexample – в этом случае требуется остановить верификации и устранить ошибку;
- доказан – значит, может быть исключён из дальнейшей верификации, повышая производительность системы верификации;
- неизвестно – т.е. пока не удалось ни найти counterexample, ни доказать ассерт. В идеале при условии правильно созданной формальной модели стопроцентная доказанность ассертов гарантирует отсутствие функциональных ошибок. В реальном проекте это невозможно, и разработчик должен сам определить, когда ему завершить разработку тестбенча, руководствуясь выбранной стратегией.

При работе со стандартными элементами (стандартные шины, FIFO…) можно воспользоваться уже готовыми библиотеками ассертов. Так, среда Questa уже содержит две такие библиотеки: Questa™ Verification Library Checkers и Questa™ Verification Library Monitors. Например, работая с памятью DDR SDRAM или с шиной PCI Express, можно воспользоваться соответствующими элементами из библиотеки Monitors. При этом в процессе верификации могут быть выловлены такие ошибки, о возможности существования которых инженер даже и не подозревал.

Вывод. Среда верификации 0-In совместно с системой Questa предлагает новую стратегию верификации, которая подразумевает дополнительные затраты времени разработчика на создание формальной модели, при этом существенно повышая качество верификации и существенно сокращая time-to-market.

Литература

1. Advanced Verification Methodology. http://www.megratec.ru/data/conference/cookbook-2.0.pdf.
2. Questa™ SV/AFV User's Manual, Software Version 6.2c.

Новости мира News of the World Новости мира

Wi-Fi вытеснит мобильные телефоны

Когда многочисленные сотовые операторы вкладывали огромные деньги (только в Великобритании сумма зашкаливает за £22,47 млрд.) в специальное оборудование и лицензии на использование сетей третьего поколения (3G), то были практически уверены, что на ближайшие 20 лет они будут продавать Интернет и разговоры всем своим абонентам. Однако, по мнению некоторых пессимистов, такие инвестиции могут не оправдать себя из-за повсеместного распространения Wi-Fi.

Основатель Truphone (сервис, пересылающий мобильные звонки через Интернет, значительно уменьшая их стоимость) Джеймс Тагг (James Tagg) утверждает, что на сегодняшний момент в Великобритании существуют более 3 млн. точек доступа к Wi-Fi, что делает эту беспроводную сеть чрезвычайно доступной для пользователей. Самым привлекательным в системе звонков, предлагаемой Truphone, является её условная «бесплатность» для производителей, которым не нужно платить национальным сетям за передачу звонка.

Использование мобильного телефона для звонков посредством VoIP популярно не только в европейских странах, но и в США, где Wi-Fi распространён не меньше. Единственным препятствием на пути повсеместного распространения такого рода сервисов является сложность настройки, из-за чего он пока не доступен всем пользователям.

www.3dnews.ru

LG сократит производство плазменных панелей

Корейская компания LG Electronics решила сократить объёмы производства плазменных панелей. При этом будут закрыты три старые производственные линии, ещё две линии продолжат работу в привычном режиме. Это приведёт к сокращению месячных объёмов производства с 430 до 360 тыс. панелей и позволит увеличить годовые сбережения до $22...32 млн.

Как известно, на сегодняшний день рынок панелей поделен на два основных сектора: плазменный и жидкокристаллический. При этом в сегменте с диагональю до 40...42 дюйма лидерами являются жидкокристаллические панели. При больших диагоналях более выгодным оказывается производство плазменных панелей. С совершенствованием и удешевлением технологий производства ЖК-панелей «плазма» понемногу уступает часть рынка. Согласно исследовательским данным DisplaySearch, продажи плазменных панелей в первом квартале этого года впервые после активного роста упали на 1% по сравнению с аналогичным периодом 2006 г.

pcworld.com

SEMI: рынок кремниевых подложек немного подрос, по итогам Q1 2007

Согласно данным квартального отчёта Silicon Manufacturers Group (SMG), входящей в состав Всемирной организации производителей полупроводниковой продукции (SEMI), мировые поставки кремниевых подложек для производства микросхем пусть незначительно, но выросли в первом квартале 2007 г. по сравнению с уровнем четвёртого квартала 2006 г.

«Несмотря на то что начало года традиционно считается слабым временем для производителей кремниевых подложек, мы наблюдаем неизменный или даже слегка увеличенный спрос на них, по сравнению с четвёртым кварталом прошлого года, – заявил Волкер Бретч (Volker Braetsch), председатель SEMI SMG и вице-президент Siltronic AG. – Особенно отличился Азиатский регион, фактически вытащивший отрасль на фоне общего снижения спроса на кремниевые подложки в других регионах».

Общая площадь отгруженных кремниевых подложек в первом квартале 2007 г. составила 1,36 млн. м^2, что представляет собой 1-% рост по сравнению с 1,34 млн. м^2, показанными отраслью в четвёртом квартале прошлого года. По сравнению же с аналогичным периодом прошлого года с 1,22 млн м^2 отгруженных кремниевых пластин, рост выражен значительно больше и составляет вполне приличные 11%.

digitimes.com

Тайваньские полупроводниковые кузницы в плюсе по итогам апреля

Вселяя надежды в сердца аналитиков на скорое восстановление после нескольких месяцев упадка, ведущий независимый производитель полупроводниковых продуктов, тайваньская компания Taiwan Semiconductor Manufacturing (TSMC), объявила о положительных результатах работы в апреле 2007 г.

Согласно опубликованным данным, выручка компании за апрель 2007 г. составила 677 млн. долл. США, что представляет собой 2,7-процентный рост по сравнению с аналогичным показателем предыдущего месяца, который всё же на 17,1% хуже уровня прошлого апреля. Выручка компании за первые четыре месяца текущего года, составившая 2,58 млрд. долл. США, также на 17,8% меньше продемонстрированной в 2006 г.

Есть признаки улучшения дел и у основного конкурента – United Microelectronics (UMC), опубликовавшей на днях отчёт о 7,7-процентном увеличении объёма выручки, полученной в апреле и достигшей отметки в 244 млн. долл. США. Однако, как и в случае с TSMC, компания всё ещё не отстаёт от темпов развития 2006 г.а, не дотягивая до показателей прошлого апреля на 3,92%.

reed-electronics.com

Фотовспышки станут движущей силой рынка LED

Согласно данным агентства Strategy Analytics, спрос на светодиодные источники (LED) для встроенных фотовспышек будет подогреваться со стороны производителей переносных портативных устройств в течение 2007 – 2011 гг. Это вызовет устойчивый рост объёма производства LED-источников примерно на 23% в год, даже несмотря на тот факт, что доходность данного сегмента рынка начнёт постепенно снижаться. Отмечается, однако, что до 61% всех светодиодных источников в 2011 г. будет использовано в производстве модулей задней подсветки для жидкокристаллических экранов.

«К 2011 г. около 80% всех переносных устройств будут оснащены камерами, – утверждает Асиф Анвар (Asif Anwar), руководитель группы мониторинга полупроводниковой продукции Strategy Analytics. – Это в свою очередь увеличит спрос на LED-источники для производства встроенных вспышек, который, возможно останется единственным сегментом для увеличения доходов производителей LED. Рост будет ещё заметнее, если удельный вес 2-мегапиксельных и выше моделей интегрированных фотокамер будет и дальше увеличиваться, а он, по оценкам аналитиков, может достигнуть 68% от всех установленных на мобильных устройствах камер».

Однако общий уровень доходов уменьшится с 1,30 млрд. долл. США в 2006 г. до 984 млн. в 2011 по причине неуклонно снижающейся средней цены реализации (ASP) на светодиодные источники, – считают аналитики агентства.

eetimes.com