Использование JTAG-тестирования и программирования на производстве

Тема этой статьи — применение периферийного сканирования на производстве. Казалось бы — чего тут писать, бери и применяй. Но, как выясняется, тема эта обширна и несправедливо обижена вниманием. По разным причинам. Одна из них — это то, что в статьях и на семинарах речь всегда начинается с того, как работает периферийное сканирование, как происходит тестирование и как разрабатывать тесты. На все это уходит столько энергии, что про применение JTAG-технологий обычно говорится пара слов. Вторая причина в том, что производство электроники в России развивается! И несколько лет назад никого не интересовала автоматизация запуска тестов, интеграция ICT-теста и периферийного сканирования на одном рабочем месте и прочее. А теперь многие предприятия интересуются этими вещами: возросла серийность и, одновременно, опыт, который неизменно приводит к культуре производства.

Алексей Иванов

Вариант 1. Автономная станция

Если ваша продукция не является крупносерийной, то вас может вполне устроить автономная станция периферийного сканирования. Это — отдельный ПК с установленным ПО для запуска тестов и приложений для программирования, плюс контроллер периферийного сканирования с дополнительными модулями, которые нужны для тестирования внешних цифровых и аналоговых разъемов. Автономная станция — самый логичный вариант, не требующий трудовых и денежных затрат на интеграцию в другие установки, создание для этого оснастки и пр. Поэтому большинство предприятий, использующих периферийное сканирование в России, применяют для производственного тестирования либо автономную станцию, либо станцию с ПО для разработки тестов. (Такое ПО обычно содержит встроенный секвенсор для запуска тестов и операций программирования.)

Что же представляет собой автономная станция в программном плане? Начнем с того, что, как многие читатели уже знают, существует ПО для разработки (или, точнее, генерации) тестов и приложений для программирования флэш и ПЛИС. У JTAG Technologies — это пакет ProVision. Принцип работы всем известен: конвертируется схематика тестируемой платы из САПР и при необходимости BOM. Затем подключается библиотека моделей компонентов (от резисторов до логики и памяти) и компонентам схематики придается функциональность. Благодаря этому, система ProVision «знает» всю информацию о тестируемом изделии: связи всех пинов (паяных соединений, если угодно), информацию о связующей логике (какие резисторы являются подтягивающими, какие — проходными, как сигнал меняется, проходя через логику, и т. д.).

Используя эти свои «знания», система проектирования генерирует приложения для тестирования и программирования. Их много, так как для каждой отдельной микросхемы ОЗУ, ПЗУ, логики и прочего генерируется отдельное приложение. В результате получаются разрозненные приложения, которые можно запускать по отдельности. Но это хорошо для отладки сгенерированных тестов, когда один и тот же тест нужно запустить несколько раз, проверяя, не возникают ли ошибки, подстраивая частоту TCK.

Когда же нам нужно выполнить основную задачу периферийного сканирования, а именно провести тотальный тест изделия и программирование всех его компонентов нажатием одной кнопки, нужно сформировать из всех созданных ранее приложений так называемую тестовую последовательность. JTAG ProVision содержит встроенный секвенсор для создания таких последовательностей. То есть скомпоновать приложения можно еще на станции разработки тестов. Производственное же программное обеспечение, которое называется ProVision Platform, представляет собой подобие JTAG ProVision, где из функций есть только вышеупомянутый секвенсор.

На рис. 1 показано окно секвенсора, которое пользователь видит, используя для «прогона» тестов и приложений для программирования станцию разработки ProVision или автономную производственную систему.

Разберем интерфейс более подробно: это поможет нам понять возможности станции периферийного сканирования на производстве. На рисунке можно видеть саму последовательность и кнопки запуска, паузы, продолжения и отмены выполнения. Первичный статус с цветовой индикацией (зеленый/красный) показывается в само́й последовательности. Но при наличии конкретных дефектов этого недостаточно.

Поэтому для каждого из приложений можно выбрать опцию: показывать результаты диагностики в итоговом отчете. Итоговый отчет формируется в реальном времени в процессе тестирования и программирования платы, фрагмент его можно увидеть в нижней части рис. 1. Такой отчет формируется каждый раз при запуске последовательности, и при желании можно посмотреть архив, который сохраняется в определенных папках. Также отчеты о тестировании можно открыть прямо в программе: их видно в левой части рис. 1.

В правой части рис. 1 можно увидеть интересные объекты: IF, GOTO, label, pause и др. Эти элементы можно использовать при создании последовательностей. Судя по названиям элементов, из них можно, например, сделать ветвящийся алгоритм прохождения операций. Приведем пример, когда это может потребоваться. Предположим, что мы тестируем изделие, и после определенных тестов требуется также осуществить операции программирования. Но на стенд тестирования могут попадать платы как с пустыми микросхемами флэш-памяти, так и с уже прошитыми (возможно, на повторную проверку после ремонта или эксплуатации). Так как большинство изделий поступает с «чистыми» ПЗУ, то нет смысла у каждого изделия предварительно стирать флэш. Чтобы у уже прошитых изделий обнулялось содержимое памяти, можно построить разветвляющийся алгоритм.

Сначала в последовательности пойдет проверка ПЗУ на отсутствие записанных данных. Если в ПЗУ есть записанные данные, то следующий шаг — операция стирания содержимого микросхемы. Если флэш окажется «чистой», то алгоритм «перепрыгнет» через операцию стирания, что сэкономит нам время. Пример алгоритма, где реализованы действия, похожие на те, что описаны выше, показан на рис. 2.

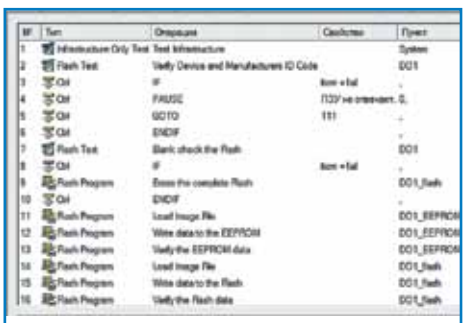

Рис. 2. Разветвляющийся алгоритм для операций программирования флэш-памяти

С помощью элементов из правой части экрана можно создавать и всплывающие информационные окна для оператора. Например, если очередь дошла до приложения, которое зажигает несколько светодиодов, работу которых нужно проверить визуально, перед этим приложением в последовательности можно запустить предупреждение об этом. Иначе оператор может забыть посмотреть на светодиоды. Задачи такого типа решаются с помощью функции Pause/Display Message.

В панели управления сразу под самой последовательностью (рис. 1) есть окно, в которое можно при необходимости вводить серийный номер тестируемой платы. Этот серийный номер затем будет отражаться в html-отчете с описанием дефектов.

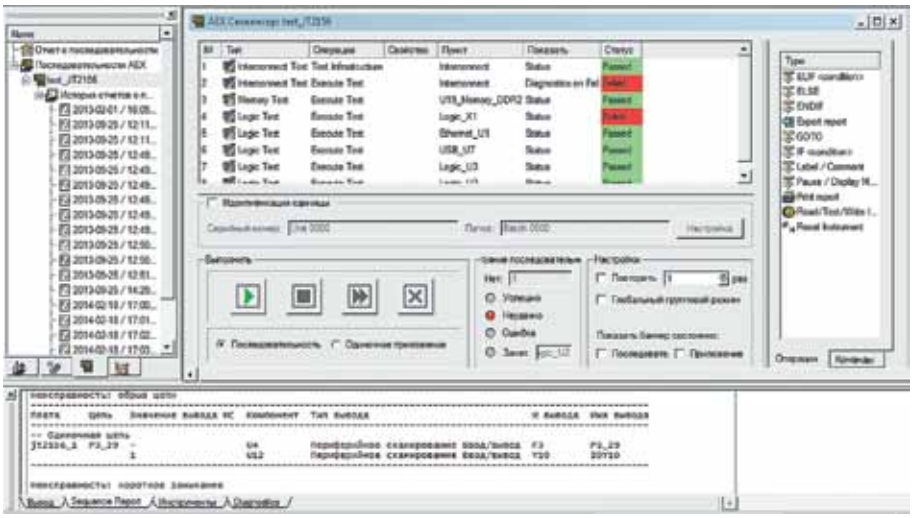

Рис. 1. Окно секвенсора автономной станции запуска тестов от JTAG Technologies

Также можно использовать командную строку (вкладка «Команды» в правой части экрана на рис. 1), что дает, по сути, возможность запускать в тестовой последовательности любые внешние приложения, не связанные с JTAG ProVision. Например, может возникнуть необходимость включить какой-то дополнительный прибор (скажем, источник питания).

Вариант 2. Интеграция в функциональные тестеры

Прежде всего, давайте разберемся, что же мы будем подразумевать под функциональными тестерами. Понятие это очень емкое. Функциональными тестерами можно назвать специализированную оснастку, стенды и ПО, созданные предприятием для какого-то конкретного изделия. Можно назвать функциональным тестером и стойку, например 19-дюймовую, в которую входят разнообразные покупные приборы, работа которых во время проверки изделия контролируется каким-то общим ПО. Это ПО создается либо силами предприятия с помощью общеизвестных сред программирования (например, C++, Visual Basic и т. п.), либо используется готовый программный продукт типа LabVIEW, TestStand и им подобные.

Существуют в промышленности и универсальные платформы функционального теста, например на базе шасси PXI. Многие компании выпускают под одной маркой множество видов измерительных модулей PXI, из которых, как из конструктора, можно собрать любой функциональный тестер.

В развитых в технологическом плане странах практикуется и другой подход к созданию функциональных тестеров. Существуют компании, специализирующиеся на построении тестовых комплексов для конкретных изделий. Обычно такая фирма изучает требования заказчика, комплектует стойку всем необходимым оборудованием и пишет ПО для прогона всех проверок, создает контактные приспособления.

А вообще, функциональный тест, если его описывать в других категориях, — это проверка работоспособности изделия и выполнения заданных функций, а также обеспечения заявленных в документации на изделие характеристик. И каким бы подробным не оказался функциональный тест, он не указывает на дефекты (их местоположение, тип и т. п.). Для этого предназначен другой тип контроля — структурный тест, к которому относится внутрисхемное тестирование и периферийное сканирование. Именно это является основной причиной желания интеграции функциональных тестеров и периферийного сканирования. Структурный тест необходим, но не всегда бывает удобно организовывать для него отдельное рабочее место и операции, особенно если производство — крупносерийное. Но, как следует из предыдущих абзацев, различия в самих функциональных тестерах делают различными и механизмы интеграции для каждого конкретного случая.

Предположим, что наши приложения, как для тестирования, так и программирования, разработаны и отлажены в среде JTAG ProVision. Помимо запуска их с помощью автономной станции, как было показано в начале статьи, существует множество других вариантов.

Самый простой пример, это если у вас на производстве для функционального контроля используется оборудование National Instruments. Компания JTAG Technologies создала пакеты интеграции JTAG-тестов в программное обеспечение LabView и TestStand. Нужно отметить, что это не просто драйверы работы JTAG-контроллера под управлением данного ПО, а наборы библиотек, позволяющих использовать процедуры, разработанные в JTAG ProVision, и результаты их выполнения в общих алгоритмах функционального тестирования. При установке на компьютер пакета интеграции устанавливаются и наборы виртуальных инструментов (VI, Virtual Instruments). С помощью этих виртуальных инструментов пользователь имеет

Рис. 3. Контроллер периферийного сканирования JT37x7 DataBlaster в разных исполнениях:
а) JT37x7/TSI; б) JT37x7/PXI

Рис. 4. Контроллер JT37x7/RMI

возможность из LabView контролировать установки контроллера периферийного сканирования (напряжения, часто́ты TCK и т. д.), выполнять в этой среде тесты, приложения для программирования флэш и ПЛИС, обрабатывать и выводить результаты тестирования и диагностические сообщения о местоположении дефектов.

Что это дает? Очевидно, что используя, скажем, LabView в более широких целях, мы имеем возможность создать одну тестовую программу, управляющую целым комплексом, который может состоять из различного оборудования, начиная от анализатора цепей и заканчивая контроллером периферийного сканирования. Поэтому для тестируемого изделия можно провести весь необходимый арсенал проверок: от измерения КСВН радиочасти до тестирования цифровой части и программирования флэш-памяти.

Аналогичным образом работает и интеграция в другие программные средства. Существуют драйверы для работы приложений ProVision в программах, написанных на C++ и Visual Basic, в TestStand от National Instruments и других средах.

Многие наверняка обратили внимание на то, что контроллеры периферийного сканирования часто выпускаются в виде карт PXI и PXIe (рис. 3б). Очевидно, что делается это для того, чтобы использовать их в составе функциональных тестеров на базе PXI-шасси. Однако это не означает, что интеграция приложений периферийного сканирования в LabView или TestStand будет работать только с такими контроллерами. Эти программы могут работать с любыми контроллерами, в том числе и с теми, что используются в автономной станции (например, настольными).

Выбор контроллера, используемого при интеграции, зависит от многих факторов. Но здесь можно учесть некоторые моменты. Даже если у вас применяется шасси с приборами PXI для функционального тестирования, и вы задумываетесь о запуске тестов ProVision в LabView или TestStand, то JTAG-контроллер PXI — хотя и элегантное, но не всегда лучшее решение. Недостаток такого решения в том, что вы остаетесь «привязаны» к стойке. Если выбрать аналогичный контроллер, но в настольном исполнении (рис. 3а), то вы сможете использовать его, и подключив к стойке PXI, и вместе с обычным ПК, где установлен JTAG ProVision. А вот контроллер в форм-факторе PXI будет трудно подключить к любому ПК.

А если, к примеру, шасси PXI монтируется в 19-дюймовую стойку, то самым логичным вариантом будет использование контроллера JT37x7/RMI (рис. 4), имеющего высоту 1U. Преимущество же контроллера в формате PXI — это удобное решение для мобильных стендов диагностики, где не очень удобно носить с места на место (или даже к заказчику) несколько отдельных приборов.

Пакеты интеграции тестов и приложений для программирования в функциональные тестеры обозначаются общей аббревиатурой PIP (Production Integration Packages). Они представляют собой программные пакеты, содержащие драйверы, утилиты или виртуальные инструменты. Для этих пакетов отсутствуют аппаратные средства интеграции, как для ICT-тестеров или установок с «летающими» щупами, где JTAG-сигналы нужно проводить еще и через сами тестеры.

Еще один совет. Не следует гадать, что же выбрать: интеграцию в функциональный тестер или отдельную, автономную станцию для запуска приложений периферийного сканирования? Задача интеграции, что называется, сама вас «найдет». Обычно интеграция — это необходимость, которую диктуют условия производства. Если такой задачи не стоит, то смело выбирайте автономную станцию.

Заключение

В следующей части статьи мы расскажем об интеграции периферийного сканирования во внутрисхемные тестеры и установки с «летающими» щупами, а также о расширенных опциях программно-аппаратных средств периферийного сканирования, полезных для производственных нужд.