Введение

Отладка современных FPGA проектов – это сложная задача. Когда вашему продукту потребуется развитие для увеличения его сложности, увеличивается время, затрачиваемое вами на верификацию проекта. Чтобы быстро продвигать ваш продукт на рынок, необходимо уменьшить время на верификацию. Чтобы облегчить напряженность между временем и рынком, используется инструмент верификации, отличающийся высокой функциональностью и простотой в использовании.

Программа Quartus II представляет инструмент отладки в системе для верификации в реальном времени вашего проекта. Каждый инструмент отладки в чипе использует комбинацию из доступных ресурсов памяти, логики и разводки для содействия процессу отладки. Инструмент создает видимые отводы (или "пробник") сигналов в вашем проекте для логики отладки. Логика отладки компилируется вместе с вашим проектом и загружается в FPGA или CPLD для анализа. Поскольку различные проекты имеют различные требования и ограничения, такие как число запасных выводов или величину остаточных ресурсов памяти в физическом чипе, вы можете выбрать инструмент из доступных инструментов отладки, которые отвечают специфическим требованиям вашего проекта.

Этот раздел содержит краткий перечень доступных инструментов отладки в чипе и описывает критерий выбора наилучшего инструмента для вашего проекта.

Экосистема отладки в чипе

Таблица IV-1 обобщает инструменты комплекта инструментов верификации в системе, которые описаны в этом разделе настольной книги Quartus II.

Таблица IV–1. Доступные инструменты комплекта инструментов для верификации в системе (Часть 1 из 2)

Инструмент	Описание	Типичные обстоятельства для использования
Логический анализатор SignalTap® II	Этот встроенный логический анализатор использует ресурсы для выборки тестовых узлов и вывода информации в программу Quartus II для отображения и анализа.	Вы имеете запас памяти на чипе и хотите запустить функциональную верификацию вашего проекта.
SignalProbe	Этот инструмент последовательно направляет внутренние сигналы на выводы I/O, наряду с тем, что сохраняет результаты вашей последней размещения и разводки проекта.	Вы имеете запас выводов I/O и хотели бы проверить работу небольшим тестом на контрольном выводе, используя внешний логический анализатор или осциллограф.
Интерфейс логического анализатора (LAI)	Этот инструмент переключает большой набор сигналов на небольшое число запасных выводов I/O. LAI позволяет вам выбрать, какие сигналы переключить на выводы I/O через JTAG.	Вы ограничены в памяти на чипе и имеете большой набор внутренних шин, которые вы хотите верифицировать, используя внешний логический анализатор. Поставщики логических анализаторов, такие как Tektronics и Agilent, имеют простую интеграцию с инструментом.
Редактор содержимого памяти в системе	Этот инструмент показывает и позволяет вам редактировать память на чипе.	Вы хотели бы посмотреть и редактировать содержимое кэша команд или кэша данных приложения процессора Nios® II.

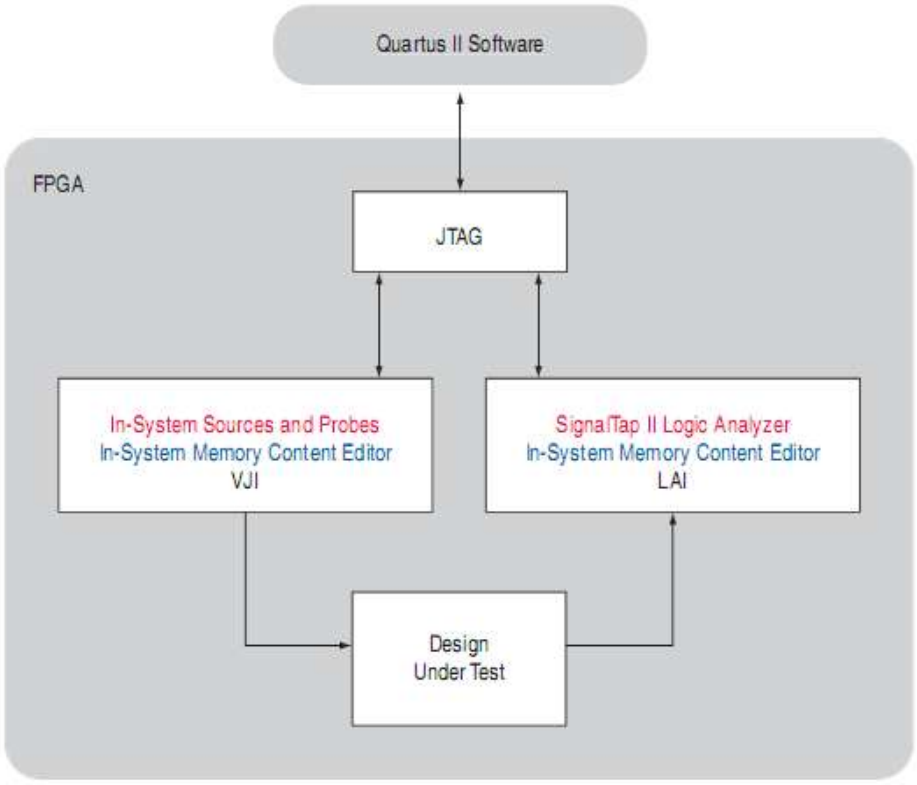
Таблица IV–1. Доступные инструменты комплекта инструментов для верификации в системе (Часть 2 из 2)

Инструмент	Описание	Типичные обстоятельства для использования
Исходники и пробники в системе	Это средство позволяет вам самым простым путём подвести и выбрать логические значения к и от внутренних узлов с помощью интерфейса JTAG.	Вы хотите создать прототип лицевой панели с виртуальными кнопками для вашего проекта FPGA.
Виртуальный интерфейс JTAG	Эта мегафункция открывает интерфейс JTAG так, чтобы вы смогли разработать своё собственное приложение.	Вы хотите сгенерировать большой набор тестовых векторов, чтобы послать их в ваше устройство по интерфейсу порта JTAG для функциональной верификации вашего проекта, запущенного в чипе.

За исключением SignalProbe, каждый инструмент отладчика в чипе использует порт JTAG для контроля и обратного чтения данных от отлаживаемой логики и тестовых сигналов. Ресурс JTAG является общим для всех инструментов отладки в чипе. Программа Quartus II компилирует логику внутри вашего проекта автоматически, распознавая информацию о данных и контроле, а также отладочные логические блоки, когда требуется ресурс JTAG. Это арбитражная логика, также называемая инфраструктурой отладки на уровне системы (SLD), показывается в иерархии проекта, когда вы компилируете проект, как **sld_hub:sld_hub_inst**. Логика SLD позволяет вам размещать различные отладочные блоки в вашем проекте и запускать их одновременно.

Чтобы максимизировать отладочные ограничения, программа Quartus II позволяет вам комбинировать инструменты отладки для полного использования и анализа логики во время теста. Все описанные в таблице IV–1 инструменты имеют основные встроенные средства отладки; это означает, что все инструменты позволяют вам прочитать обратно информацию от узлов проекта, которые подключены к отладочной логике. Из набора отладочных средств: Логический анализатор SignalTap II, LAI и средства SignalProbe – основные отладочные инструменты, оптимизированы для сигналов пробников в вашем RTL списке соединений. Исходники и пробники в системе, виртуальный интерфейс JTAG и редактор содержимого памяти в системе – дополнительные инструменты, позволяющие обратное чтение данных от точек отладки, позволяя вам задавать входные значения для вашего проекта во время прогона. Подытожим, набор инструментов отладки в системе формируют отладочную экосистему. Набор инструментов может генерировать возбуждение и получать реакцию от логики во время теста, предлагая полное решение по отладке (рисунок IV-1).

Figure IV–1. Quartus II Debugging Ecosystem *(Note 1)*



Note to Figure IV–1:

(1) The set of debugging tools offer end-to-end debugging coverage.

Инструменты в цепочке инструментов предлагают различные преимущества и различные компромиссы. Чтобы понять выбор критерия между различными инструментами, следующие разделы анализируют инструменты согласно их типовым приложениям.

Первый раздел "Инструменты анализа для RTL узлов", сравнивает логический анализатор SignalTap II, SignalProbe и LAI. Эти три инструмента логически сгруппированы потому, что они предназначены для отладочных узлов вашего RTL списка соединений на скорости работы системы.

В следующем разделе "Инструменты, оказывающие воздействие" на странице 13-8, сравниваются редактор содержимого памяти в системе, мегафункция интерфейса виртуального JTAG и Исходники и пробники в системе. Эти инструменты логически сгруппированы потому, что они предназначены для транзакций чтения и записи с помощью порта JTAG.

Инструменты анализа для RTL узлов

Встроенный логический анализатор SignalTap II, средство SignalProbe и LAI разрабатываются специально для проб и отладки RTL сигналов на скорости работы системы. Это основные инструменты анализа, которые позволяют вам отводить и анализировать любые разведённые узлы в FPGA или CPLD. Три этих инструмента удовлетворяют ряду требований. Если вы имеете запас ресурсов логики и памяти, логический анализатор SignalTap II вам полностью подходит для быстрой функциональной верификации вашего проекта, запущенного в реальном устройстве.

С другой стороны, если вы стеснены в ресурсах логики и памяти, вам требуется больше сосредоточиться на пробниках, глубоко ассоциированных с внешними логическими анализаторами, такими как LAI и средства SignalProbe, позволяющими вам легко наблюдать внутренние сигналы проекта, используя внешнее оборудование.

Одно из главнейших критериев выбора этих трёх инструментов являются доступные ресурсы, оставшиеся в вашем чипе после размещения вашего проекта, и число доступных свободных выводов. Это даст результат для оценки ваших предпочтительных настроек отладки на ранних стадиях процесса планирования проекта, обеспечивающих вашу печатную плату, ваш проект Quartus II и вашу разработку всеми установками для поддержки соответствующих опций. Раннее планирование может уменьшить временные затраты на отладку и устранение необходимых последующих изменений при подгонке вашей предпочтительной методики отладки. Следующие два раздела содержат информацию, помогающие вам выбрать соответствующий инструмент путём сравнения инструментов, согласно использованными вами ресурсами и задействованными выводами.

Логический анализатор SignalTap II не поддерживает CPLD, поскольку они не имеют доступных на чипе ресурсов памяти.

Использование ресурсов

Некоторые инструменты отладки, которым требуется использовать JTAG соединение, требуется сначала упомянуть инфраструктуру логики SLD, для подключения к JTAG интерфейсу и разрешения конфликтов между установленными модулями отладки. Эта служебная логика использует около 200 ЛЭ, небольшую долю доступных в большинстве поддерживаемых чипов. Служебная логика распределяется между всеми доступными модулями отладки в вашем проекте. Логический анализатор SignalTap II и LAI используют JTAG подключение.

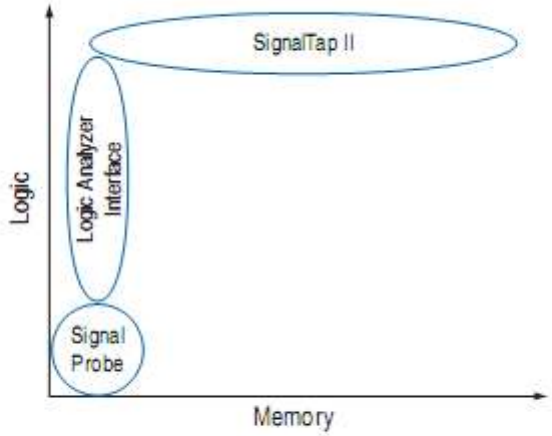
SignalProbe требуется очень мало ресурсов чипа. Поскольку ему не требуется соединение по JTAG, SignalProbe не использует ресурсы логики и памяти – он использует только ресурсы разводки для вывода внутреннего сигнала на отладочную тестовую точку.

LAI требуется небольшое количество логики для размещения функции мультиплексора между тестовыми сигналами, дополнительно к инфраструктуре логики SLD. Поскольку выборки данных не сохраняются в чипе, LAI не использует ресурсы памяти.

Логическому анализатору SignalTap II требуются ресурсы и логики, и памяти. Численность используемых ресурсов логики зависит от численности отводов сигналов и сложности триггерной логики. Поэтому, величина логических ресурсов, которые обычно использует логический анализатор SignalTap II, - это небольшой процент большинства проектов. Базовая конфигурация состоит из логики разрешения конфликтов SLD и одного узла с базовой триггерной логикой, состоящей примерно из 300 - 400 логических элементов (ЛЭ). Каждый дополнительный узел увеличивает базовую конфигурацию на 11 ЛЭ. Оцените ресурсы логики, памяти и прочие важные факторы для вашего проекта. Использование памяти значительное, оно определяется тем, как вы конфигурируете ваш модуль логического анализатора SignalTap II для захвата данных и глубины проб, которые вам потребуются для отладки. Логический анализатор SignalTap II имеет преимущество, которое состоит в том, что ему не требуются внешние устройства, поскольку вся триггерная логика и хранение данных расположены на чипе.

На рисунке IV–2 показан концептуальный график использования ресурсов этими тремя инструментами по отношению друг к другу.

Figure IV–2. Resource Usage per Debugging Tool *(Note 1)*

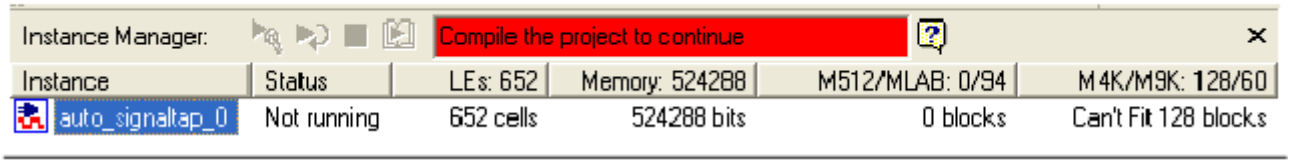


Note to Figure IV–2:

(1) Though resource usage is highly dependent on the design, this graph provides a rough guideline for tool selection.

Средство оценки ресурсов для логического анализатора SignalTap II и LAI позволяет вам быстро решить, имеется ли достаточное число ресурсов чипа перед компиляцией инструмента вместе с проектом. На рисунке IV–3 показано средство оценки ресурсов для логического анализатора SignalTap II и Интерфейса логического анализатора (LAI).

Figure IV–3. Resource Estimator



Использование выводов

Соотношение количества использованных выводов на чипе для сигналов, выведенных с помощью средства SignalProbe, один к одному. Поскольку это средство быстро расходует свободные выводы, обычным применением для этого средства является разводка контрольных сигналов для резервирования отладочных выводов для отладки.

Соотношение количества выводов на чипе для сигналов, выведенных с помощью LAI, много сигналов на один вывод. Оно может выводить до 256 сигналов на каждый отладочный вывод в зависимости от доступных ресурсов. Контроль активного сигнала, который подводится к резервируемому I/O выводу, выполняется через JTAG порт. LAI идеально подходит для разводки шин данных для отладки на тестовых выводах.

Кроме тестовых выводов JTAG, логический анализатор SignalTap II более не использует дополнительных выводов. Все данные буферизируются с помощью памяти на чипе и передаются в графическую оболочку (GUI) SignalTap II посредством тестового порта JTAG.

Улучшенное удобство в применении

Встроенный логический анализатор SignalTap II, средство SignalProbe и LAI могут быть добавлены к существующему проекту с минимальным эффектом. С помощью средства поиска узлов, вы можете найти сигналы для разводки их к модулю отладки без совершения каких-либо изменений в вашем HDL коде. SignalProbe вставляет сигналы только из вашей базы данных пост-компоновка. Логический анализатор SignalTap II и LAI поддерживают вставку сигналов из списков соединений пре-синтез и пост-компоновка. Все три инструмента позволяют вам быстро находить и конфигурировать ваши установки отладки. Дополнительно, средства инкрементной компиляции и разводки Quartus II позволяют вам быстро получать новый файл программирования, повышая продуктивность и позволяя быстрые ограничения в отладке.

LAI и логический анализатор SignalTap II поддерживают инкрементную компиляцию. С помощью инкрементной компиляции, вы можете добавлять модули логического анализатора SignalTap II или LAI инкрементно в ваш уже размещённый и разведённый проект. Это особенно полезно с точки зрения сохранения вашего времени и оптимизации площади уже существующего проекта, а также уменьшении времени компиляции при неизбежных некоторых изменения во время процесса отладки. Инкрементная компиляция сберегает вам до 70% времени компиляции во время полной компиляции.

SignalProbe использует средство инкрементной разводки. Средство инкрементной разводки запускается только на стадии компиляции Компоновщика. Оно также позволяет вам компилировать, не затрагивая ваш проект, за исключением разводки новых узлов. SignalProbe сберегает вам до 90% времени компиляции во время полной компиляции.

Продуктивным удобством в применении является то, что все инструменты отладки на чипе поддерживают скриптирование из набора quartus_stp Tcl. Для логического анализатора SignalTap II и для LAI, скриптирование позволяет пользователю определить автоматизацию набора данных во время отладки в лаборатории.В дополнение, сервер JTAG позволяет вам отлаживать проект, который уже запущен в чипе, удалённо подключенном к ПК. Это позволит вам применять установку вашего устройства в лаборатории, загружать в него только новый .sof файл, и выполнять некоторый анализ непосредственно с вашего рабочего стола.

В таблице IV-2 собраны основные отладочные средства этих инструментов и даны рекомендации о том, как лучше воспользоваться представленными средствами.

Таблица IV–2. Примерные инструменты отладки в чипе для основных средств отладки (Часть 1 из 2)

Свойство	Signal Probe	Интерфейс логического анализатора (LAI)	Логический анализатор SignalTap II	Описание
Большая глубина выборок	не доступно	да	возможно, но не приводит к наилучшим результатам	Внешний логический анализатор, используемый вместе с LAI, имеет больший буфер для хранения захваченных данных, чем логический анализатор SignalTap II. Ни какие данные не захватываются или удерживаются в SignalProbe.
Простота и скорость получения результата отладки	да	да	возможно, но не приводит к наилучшим результатам	Внешнее оборудование, такое как осциллограф и смешивающий сигнал осциллограф (MSO), позволяет вам использовать LAI или SignalProbe вместе с LAI, чтобы получить доступ к временному режиму, позволяющему вам во время отладки комбинировать потоки данных.

Таблица IV–2. Примерные инструменты отладки в чипе для основных средств отладки (Часть 2 из 2)

Свойство	Signal Probe	Интерфейс логического анализатора (LAI)	Логический анализатор SignalTap II	Описание
Минимальное влияние на логику проекта	да	да(2)	да(2)	LAI добавляет минимум логики к проекту, потребляя небольшое количество ресурсов. Логический анализатор SignalTap II имеет малое влияние на проект, поскольку он расположен в отдельном разделе проекта. SignalProbe инкрементно разводит узлы к выводам, не затрагивая проект как таковой.
Сокращение времени компиляции и перекомпиляции	да	да(2)	да(2)	SignalProbe инкрементно разводит сигналы до резервированных заранее выводов, требуется очень малое время на перекомпиляцию, чтобы сделать изменения в секции исходных сигналов. SignalTap II и LAI имеют преимущество перекомпоновки только своих разделов, чтобы сократить время перекомпиляции.
Переключательная способность	не доступно	не доступно	да	Логический анализатор SignalTap II предлагает переключательную способность, доступную только коммерческим логическим анализаторам.
Использование I/O	возможно, но не приводит к наилучшим результатам		да	Логическому анализатору SignalTap II не требуется дополнительных выходных выводов. LAI и SignalProbe требуется назначить I/O выводы.
Скорость захвата	не доступно	возможно, но не приводит к наилучшим результатам	да	SignalTap II может обрабатывать данные на скоростях более 200 МГц. Такие же скорости обработки достижимы LAI совместно с внешними логическими анализаторами, хотя целостность сигналов может быть потеряна.
Не требуется подключение JTAG	да	возможно, но не приводит к наилучшим результатам		Для проектов FPGA с логическим анализатором SignalTap II или LAI требуется активное подключение по JTAG хост-машины с запущенным ПО Quartus II. SignalProbe не требуется хост-машина для целей отладки.
Не требуется внешних устройств	возможно, но не приводит к наилучшим результатам		да	Логика SignalTap II находится полностью внутри программируемого устройства FPGA. Дополнительные внешние устройства, кроме подключение по JTAG хост-машины с запущенным ПО Quartus II или только ПО SignalTap II. SignalProbe и LAI требуется использовать внешние устройства отладки, такие как, мультиметры, осциллографы или логические анализаторы.

(2)Когда используется инкрементная компиляция

Инструменты, оказывающие воздействие

Редактор содержимого памяти в системе, исходники и пробники в системе, а также виртуальный интерфейс JTAG – каждый из них позволяет вам использовать интерфейс JTAG в качестве основного порта связи. Все три инструмента могут быть использованы для достижения одних и тех же результатов, это значит, что в определённых приложениях использовать один инструмент проще, чем другие. Исходники и пробники в системе идеальны для переключения контрольных сигналов. Редактор содержимого памяти в системе применяется для ввода больших объёмов тестовых данных. В заключении, мегафункция виртуального JTAG предназначена для продвинутых пользователей, которые желают разрабатывать собственные приложения JTAG.

Исходники и пробники в системе

Исходники и пробники в системе – это простейший способ использовать ресурсы JTAG для чтения и записи в ваш проект. Вы можете начать с того, что вставите мегафункцию в ваш HDL код. Мегафункция содержит исходные порты и порты пробников, чтобы передавать значения во внутрь и получать значения сигналов, подключенных к портам, соответственно. Детали обмена по JTAG скрыты мегафункцией. Во время работы, GUI показывает каждый исходный порт и порт пробника для блока, и позволяет вам читать с каждого порта пробника и управлять портом исходника. GUI делает этот инструмент идеальным для переключения набора контрольных сигналов во время процесса отладки.

Хорошим применением исходников и пробников в системе - это вытеснение GUI кнопок-тумблеров и светодиодов, задействованных в фазе разработки проекта. Кроме того, исходники и пробники в системе поддерживают команды скриптирования для чтения и записи с использованием quartus_stp. Если вы используете Tk-инструмент, вы можете создать собственный графический интерфейс – средство, создание виртуальной лицевой панели прибора на фазе прототипирования проекта.

Редактор содержимого памяти в системе

Редактор содержимого памяти в системе позволяет вам быстро видеть и модифицировать содержимое памяти с помощью GUI интерфейса или с помощью команд Tcl скриптирования. Редактор содержимого памяти в системе работает при включении однопортовых RAM блоков в двухпортовые RAM блоки. Один порт подключается к вашему тактовому домену и сигналам данных, а другой порт подключается к JTAG тактам и сигналам данных для редактирования или наблюдения.

Поскольку вы можете вы можете просто модифицировать большие наборы данных, редактор содержимого памяти в системе лучше применять для генерации тестовых векторов для вашего проекта. Например, вы вставляете свободный блок памяти, подключаете выходной порт к тестовой логике (используете те же самые тактовые сигналы, что и ваша тестовая логика) и создаёте связывающую логику для генерации адреса и контроля памяти. Во время работы, вы можете модифицировать содержимое памяти, используя скрипт или GUI редактора содержимого памяти, и выполнить пакет транзакций данных в модифицированном RAM блоке синхронно с тестируемой логикой.

Мегафункция интерфейса виртуального JTAG

Мегафункция интерфейса виртуального JTAG – эта наивысший уровень манипуляции ресурсами JTAG. Эта мегафункция позволяет вам создавать свою собственную цепь сканирования за счёт использования всех контрольных сигналов JTAG, конфигурирования ваших регистров инструкций JTAG (IR) и регистров данных JTAG (DR). Во время работы, вы контролируете цепи IR/DR с помощью Tcl API. Это средство предназначено для пользователей, которые имеют полное представление об интерфейсе JTAG, и хотят точного контроля количества и типом используемых ресурсов.

Заключение

Пакет инструментов отладки в чипе Quartus II позволяет вам быстро решать ограничения отладки, путём предоставления вам набора функциональных инструментов анализа, а также инструментов, использующих порт JTAG в качестве основного интерфейса коммуникаций. Программа Quartus II значительно расширяет границы применения приложения, предоставляя вам комплексное Tcl/Tk API. С помощью Tcl/Tk API вы не только повысите уровень автоматизации всех инструментов анализа, но и сможете создать виртуальную лицевую панель легко и быстро на стадии прототипирования.

В дополнение, все инструменты отладки в чипе имеют плотную интеграцию с большинством производственных средств программы Quartus II. Средства инкрементной компиляции и инкрементной разводки позволяют вам быстро получать свежий файл программирования. Средство межпробника позволяет вам быстро находить и идентифицировать узлы. Логический анализатор SignalTap II, который используется совместно с временным анализатором TimeQuest, - лучший в своём классе пакет временной верификации, который позволяет вам временную и функциональную верификацию.

В этом томе подробно описан каждый инструмент отладки в чипе. Он состоит из следующих глав:

- Глава 13, Быстрая отладка проекта с помощью SignalProbe
- Глава 14, Отладка проекта с помощью встроенного логического анализатора SignalTap II
- Глава 15, Отладка в системе с помощью внешних логических анализаторов
- Глава 16, Обновление памяти и констант в системе
- Глава 17, Отладка в системе с помощью исходников и пробников в системе