Добавление сигналов с помощью плагина

Вместо того, чтобы добавлять отдельные или сгруппированные сигналы с помощью Поиска узлов, вы можете добавлять группы соответствующих сигналов особого типа IP с помощью плагина. Встроенный логический анализатор SignalTap II содержит один плагин, устанавливаемый для процессора Nios II. Кроме быстрого добавления сигнала, плагин также предлагает набор других средств, таких как предпроектная мнемоническая таблица, использующаяся для создания триггеров и наблюдения за данными, которая также используется для деассемблирования кода в захваченных данных.

Плагин Nios II, например, создаёт одну мнемоническую таблицу во вкладке **Установки** и две таблицы во вкладке **Данные**:

■ Инструкции Nios II (вкладка Установки) – Охватывает все сигналы, необходимые для защёлкивания по выбранному адресу инструкции.

■ Элементы адреса Nios II (вкладка Данные) – Показывает адрес выполненных инструкций в шестнадцатеричном формате или в имени символа программирования, если это опционально определено в файле формата исполнения и связей (**.elf**).

■ Деассемблирование Nios II (вкладка Данные) – Показывает деассемблированный код по соответствующему адресу.

За дополнительной информацией о других средствах, предоставляемых плагином, обратитесь к главе "Определение триггеров" на странице 14-33 и к главе "Наблюдение, анализ и использование захваченных данных" на странице 14-66.

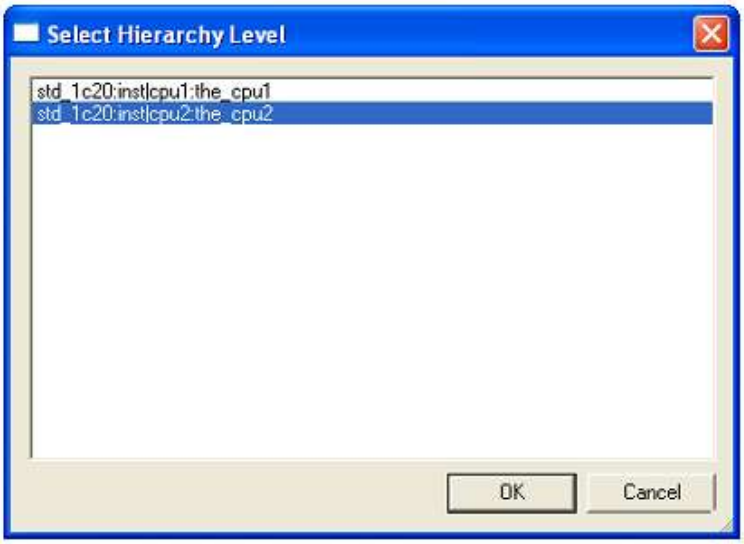
Для добавления сигнала в **.stp** файл с помощью плагина, выполните следующие шаги после запуска Анализа и Выработки вашего проекта:

1. Правый клик на список узлов. В подменю **Добавить узлы с помощью плагина**, кликните на название плагина, который вы хотите использовать, например, на имеющийся плагин, называемый Nios II.

Если IP для выбранного плагина не существует в вашем проекте, сообщение проинформирует вас о том, что вы не сможете использовать выбранный плагин.

2. В диалоговом окне **Выбор уровня иерархии** показывается IP иерархия вашего проекта (рисунок 14-9). Выберите IP, содержащий сигналы, которые наблюдать с помощью плагина и кликните **ОК**.

Figure 14–9. IP Hierarchy Selection



3. Если все сигналы доступны в плагине, раскроется диалоговое окно, зависящее от выбранного плагина, с набором доступных опций для плагина. Для плагина Nios II вы можете дополнительно выбрать **.elf** файл, содержащий программные символы из вашей программы проекта Nios II IDE. Установите требуемые опции и кликните **ОК**.

Чтобы сделать доступными все нужные сигналы, в настройках Анализа и Синтеза Quartus II включите опцию **Создавать отладочные узлы для IP ядер**. Все сигналы, присутствующие в плагине будут добавлены в список узлов.

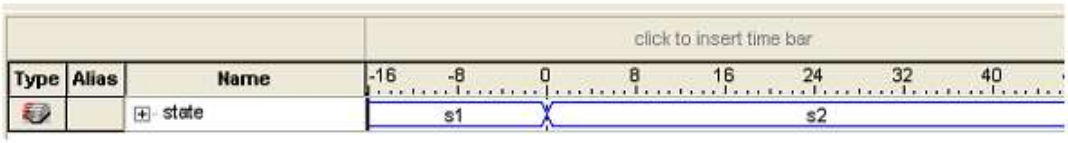
Добавление регистров состояний конечного автомата

Поиск сигналов для отладки конечных автоматов (FSM) может быть трудоёмким. Поиск узлов в списке соединений пост-компоновка может быть бесполезен, потому что сигналы кодов FSM могут быть изменены или оптимизированы во время синтеза и размещения и разводки. Если вы хотите найти все необходимые узлы в списке соединений пост-компоновка или вы используете узлы из списка соединений пре-синтеза, требуется дополнительный шаг для поиска и маршрутизации FSM значений сигналов от определённых вами имён состояний в вашем HDL коде.

Начиная с 8,0 версии программы Quartus II, SignalTap II GUI находит конечные автоматы в вашем скомпилированном проекте. Конфигурация SignalTap II автоматически проводит сигналы состояний FSM, например состояния кодирования, через процесс компиляции. Правым кликом в диалоговом окне SignalTap II вы можете добавить все сигналы состояний FSM в ваш встроенный логический анализатор одной командой. Для каждого конечного автомата, добавленного в вашу конфигурацию SignalTap II, средство отладки FSM добавляет мнемоническую таблицу, чтобы развести значения сигналов согласно нумерации в вашем исходном коде. Мнемонические таблицы позволяют вам запросто визуализировать переходы конечного автомата на дисплее временных диаграмм. Средство отладки FSM поддерживает добавление сигналов из списков соединений пре-синтез и пост-компоновка.

На рисунке 14-10 показан дисплей временных диаграмм с декодированными значениями конечного автомата, добавленного средством отладки FSM.

Figure 14–10. Decoded FSM Mnemonics



За рекомендациями написания кода для определения FSM на Verilog и VHDL, обратитесь к главе "Рекомендованные стили программирования HDL" в томе 1 Настольной книги Quartus II.

Для добавления сигналов пре-синтеза в файл конфигурации, выполните следующие шаги после запуска Анализа и Выработки вашего проекта:

1. Создайте новый **.stp** файл или используйте существующий **.stp** файл. Некоторые **.stp** файлы, которые менеджер плагинов MegaWizard создаёт из реализаций, не поддерживаются этим средством.
2. На вкладке установок SignalTap II правым кликом в любом месте списка узлов выберите **Добавить узлы конечного автомата**. Раскроется диалоговое окно **Добавить узлы конечного автомата**. В этом диалоговом окне находится список всех FSM, найденных в вашем проекте. В SignalTap II GUI для детектирования сигналов пре-синтеза конечных автоматов, выполните Анализ и Выработку.
3. В раскрывшемся меню **Список соединений** выберите **пре-синтез.**
4. Выберите нужный конечный автомат.
5. Кликните **ОК**. Этим вы добавите узлы FSM в файл конфигурации. А мнемоническая таблица автоматически применится к группе сигналов конечного автомата.

Для добавления сигналов пост-компоновки в файл конфигурации, выполните следующие шаги после выполнения полной компиляции вашего проекта:

1. Установите тип раздела проекта для конечного автомата, который вы будете отлаживать, пост-компоновка.
2. Разрешите **.stp** файл для проекта Quartus II, используя страницу встроенного логического анализатора SignalTap II во вкладке **Настройки**. Вы можете просто создать новый **.stp** файл или использовать имеющийся **.stp** файл.

Для детектирования сигналов пост-компоновки конечного автомата в SignalTap II GUI, выполните полную компиляцию вашего проекта.

3. На вкладке установок SignalTap II правым кликом в любом месте списка узлов выберите **Добавить узлы конечного автомата**. Раскроется диалоговое окно **Добавить узлы конечного автомата**. В этом диалоговом окне находится список всех FSM, найденных в вашем проекте.
4. В раскрывшемся меню **Список соединений** выберите **пост-компоновка.**
5. Выберите нужный конечный автомат.
6. Кликните **ОК**. Этим вы добавите узлы FSM в файл конфигурации. А мнемоническая таблица автоматически применится к группе сигналов конечного автомата.

Изменение и восстановление мнемонических таблиц для конечных автоматов

Когда вы добавите сигналы состояний конечного автомата с помощью средства отладки FSM, SignalTap II GUI создаёт мнемоническую таблицу, используя формат *<StateSignalName>_table*, где StateSignalName – это имя сигналов состояний, которое вы декларировали в вашем RTL.

Вы можете редактировать некоторые мнемонические таблицы, используя диалоговое окно **Установки мнемонической таблицы**.

Если вы хотите восстановить изменённую мнемоническую таблицу, правым кликом в любом месте в окне списка узлов выберите **Воссоздать мнемоники конечного автомата**. По умолчанию, восстановление мнемонической таблицы перезаписывает существующую мнемоническую таблицу, которую вы изменяли. Если вы хотите иметь восстановленную мнемоническую таблицу FSM в виде новой записи, снимите отметку в опции **Перезаписать существующую мнемоническую таблицу** в диалоговом окне **Воссоздание мнемоники конечного автомата.**

Если вы хотите добавить или удалить сигнал из группы сигналов состояний FSM во вкладке установок, удалите группу изменённых регистров и заново добавьте FSM сигналы.

За дополнительной информацией об использовании мнемоник обратитесь к главе "Создание мнемоник для наборов разрядов" на странице 14-69.

Дополнительные возможности анализа

Графическая оболочка конфигурации SignalTap II распознаёт конечные автоматы в вашем проекте только, если вы используете встроенный синтез Quartus II (QIS). Средство отладки конечных автоматов не сможет вывести сигналы FSM или декодировать состояния, если вы использовали инструментарий сторонних разработчиков.

Если вы добавляете сигналы пост-компоновки FSM, то средство отладки FSM может не вывести большинство сигналов оптимизированных во время процесса компиляции. Если разрешены две следующие оптимизации, то средство отладки FSM SignalTap II не сможет представить список мнемонических таблиц для конечных автоматов проекта:

■ Если вы включили физический синтез, регистры состояний могут попасть в балансирование ресурсов (восстановление синхронизации регистров) для достижения f_{max}. Средство отладки FSM может не найти в списке пост-компоновки регистров состояний после выполнения восстановления синхронизации регистров.

■ Средство отладки FSM может не найти сигналов состояний, которые были упакованы в блоки RAM и DSP во время оптимизации QIS и Компоновщика.

Вы можете сделать доступным для использования средства отладки FSM, добавляя сигналы состояния пре-синтеза.

Определение глубины замеров

Глубина замеров определяет численность замеров, собранных и сохранённых для каждого сигнала в буфере собранных данных. Чтобы установить глубину замеров, выберите нужное число замеров и сохраните в списке **Глубина замеров**. Диапазон замеров от 0 до 128K.

Если ресурсы памяти чипа ограничены, то вы не сможете успешно скомпилировать ваш проект с выбранным размером буфера замеров. Уменьшайте глубину замеров, чтобы уменьшить использование ресурсов.

Сбор данных в RAM определённого типа

Когда вы используете встроенный логический анализатор SignalTap II с несколькими чипами, вы обладаете опцией выбора типа RAM, в которой будут сохраняться собранные данные. Выбор памяти позволяете вам сохранять определённые блоки памяти для вашего проекта и размещать в других частях памяти собранные данные SignalTap II. Например, если в проекте размещено большое буферизирующее приложение, например системный кэш, оно идеально подходит для размещения в M-RAM блоках, так чтобы оставить M512 или M4K блоки для сохранения данных SignalTap II.

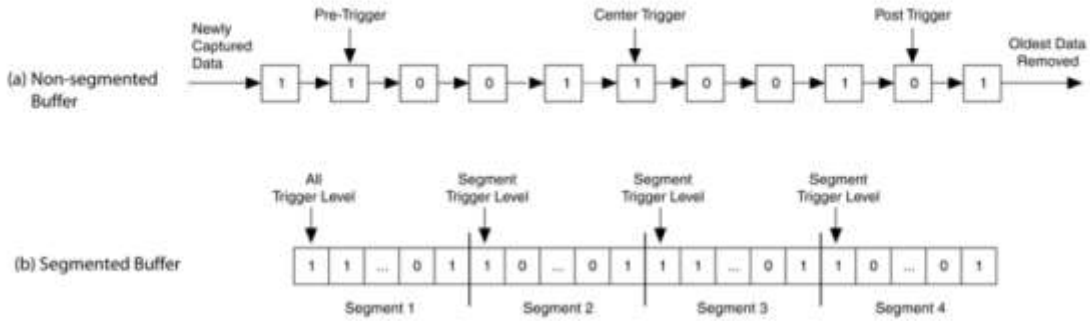
Для выбора типа RAM, используемого для буфера SignalTap II, выберите её из списка типов памяти. Используйте это средство, когда собранные данные (как написано в отчёте счётчика ресурсов SignalTap II) не больше, чем доступная память выбранного типа в FPGA.

Выбор режима для буфера сбора данных

Средство выбора типа для буфера сбора данных во встроенном логическом анализаторе SignalTap II предоставляет вам выбор, как организовать буфер собранных данных, и потенциально может уменьшить количество памяти, требуемой для сбора данных в SignalTap II. Существуют два типа буферов во встроенном логическом анализаторе SignalTap II - несегментный буфер и сегментный буфер. Несегментный буфер – это когда встроенный логический анализатор SignalTap II использует всё пространство памяти как один FIFO, постоянно заполняя буфер, пока встроенный логический анализатор находится в определенном состоянии триггера. Сегментный буфер – это когда пространство памяти поделено на определённое количество отдельных буферов. Каждый буфер работает как самостоятельный FIFO со своими состояниями триггера. Только один буфер может быть активным во время измерений. Встроенный логический анализатор SignalTap II обращается к следующему сегменту по достижению определённого состояния или состояний триггера активного сегмента.

Когда используется несегментный буфер, вы можете использовать средство квалификации памяти для определения, какие отсчёты должны записываться в буфер сбора данных. Средство квалификации памяти поможет вам максимально использовать доступное пространство памяти для сегментных и несегментных буферов. На рисунке 14-11 показаны различия между буферами двух типов.

Figure 14–11. Buffer Type Comparison in the SignalTap II Embedded Logic Analyzer *(Note 1)*



Примечания к рисунку 14-11:

(1) Несегментный и сегментный буферы могут использовать предопределённую позицию триггера (Пре-триггер, Центральный триггер, Пост-триггер) или определить другую позицию, используя вкладку **Основные состояния переключений**. Обратитесь к «Определение позиции триггеров» на странице 14-48 за подробной информацией.

(2) Каждый сегмент представляет собой FIFO и работает как несегментный буфер, показанный в (a).

За дополнительной информацией о средстве квалификации памяти обратитесь к главе «Использование средства квалификации памяти» на странице 14-25.

Несегментный буфер

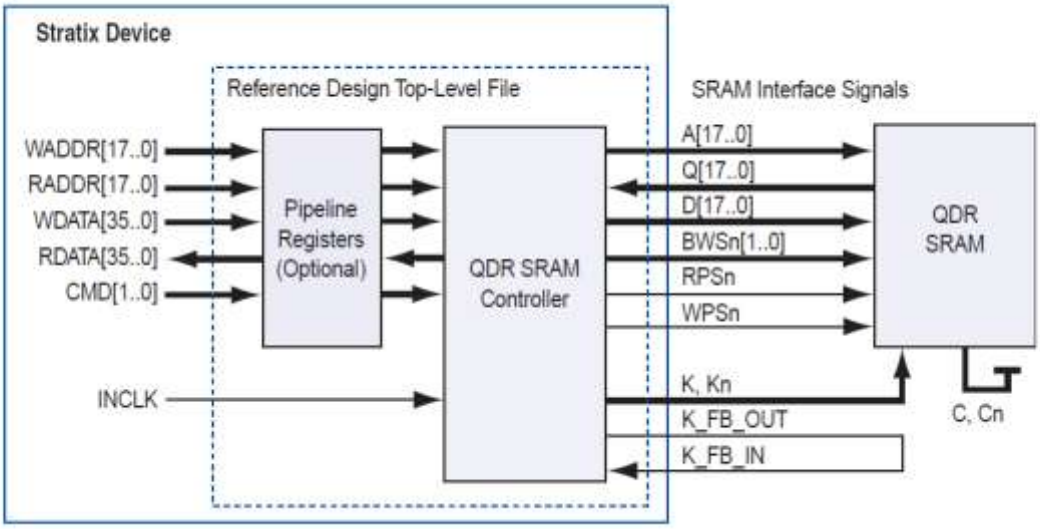
Несегментный буфер (также называемый круговой буфер), показанный на рисунке 14-11(a), - это тип буфера по-умолчанию, используемый встроенным логическим анализатором SignalTap II. Пока запущен встроенный логический анализатор, данные сохраняются в буфере, пока не переполнят его, а потом новые данные заменяют старые данные. Это продолжается до тех пор, пока не выполняется определённое событие – не устанавливается определённое состояние триггера. Когда это происходит, логический анализатор продолжает собирать данные после события триггера, основываясь на настройках позиции триггера в панели **Конфигурация сигнала** в **.stp** файле, пока не наполнится буфер. Выбирайте настройки из списка для определения, какие данные нужнее перед (**позиция пост-триггер**), после (**позиция пре-триггер**) или в центральной позиции данных (**позиция центр триггер)**. Иначе используйте процесс основных состояний переключений для определения другой позиции триггера внутри буфера собранных данных.

Обратитесь к «Определение позиции триггеров» на странице 14-48 за подробной информацией.

Сегментный буфер

Сегментный буфер делает проще отладку систем, которые имеют относительно редкие периодичные события. Память данных разделяется на ровные сегменты, с набором состояний триггера, определённого для каждого сегмента. Каждый сегмент функционирует как несегментный буфер. На рисунке 14-12 показан пример этого типа системы буферов.

Figure 14–12. Example System that Generates Recurring Events



Встроенный логический анализатор SignalTap II верифицирует функциональность проекта, показанного на рисунке 14-12, чтобы проверить, корректность записи данных в контроллер SRAM. Буфер собранных данных встроенного логического анализатора SignalTap II позволяет вам наблюдать порт *RDATA*, когда в порт *RADDR* посылается H'0F0F0F0F. Вы можете наблюдать различные транзакции чтения из чипа SRAM без повторного запуска встроенного логического анализатора SignalTap II. Средство буфера сохранённых данных позволяет вам сегментировать память так, чтобы вы могли снимать показания в нужный момент в любое время, без опустошения локализованной памяти. Количество циклов, в которые можно снимать показания, зависит от количества сегментов, определённых в настройках **Данные**.

Чтобы разрешить и сконфигурировать буфер захвата, выберите **Сегментированный** в редакторе SignalTap II, затем выберите количество используемых сегментов. В этом примере выбрано шестьдесят четыре 64-отсчётных сегмента, которые позволят вам захватывать 64 цикла чтения, когда сигнал *RADDR* равен H'0F0F0F0F.

За дополнительной информацией о режиме для буфера сбора данных, обратитесь к главе «Настройки режима для буфера сбора данных» в разделе Помощи Quartus II.