JTAG-тестирование

(часть 3)

Иосиф Каршенбойм (Санкт-Петербург)

Данная статья представляет собой обзор материалов, посвященных тестированию плат и устройств по интерфейсу JTAG. В третьей части статьи продолжается рассмотрение команд тестирования, выдаваемых по интерфейсу JTAG, и структуры BSDL-файла, а также даётся описание «привязки» битов регистра сканирования к выводам микросхемы.

Переходы, необходимые при работе с данными

Возвращаемся к описанию диаграмм переходов. Теперь мы уже точно знаем, как выдать команду и какая команда нам нужна. В этом разделе мы рассмотрим, как надо выдавать и получать данные из микросхемы, как биты данных будут управлять работой вывода микросхемы и какие данные будут читаться с конкретного вывода микросхемы.

Диаграмма переходов, которые необходимо выполнить для того, чтобы работать с данными (см. рис. 13), также обычно начинается с состояния Run_Test/Idle, затем наступают состояния Select_DR, CAPTURE_DIR, Shift_DR. В состоянии Shift_DR происходит приём и передача данных. После работы с данными необходимо перевести автомат через состояния Exit1_DR, Update_DR снова в Run_Test/Idle. По окончании работы с данными необходимо выполнить условие, оговоренное выше, а именно – передать бит TMS, указывающий на завершение данного режима, вместе с последним битом данных.

Есть возможность осуществить и другие переходы управляющего автомата, но рассмотрение этих вопросов выходит за рамки статьи. Необходимо подчеркнуть, что мы не пользуемся всеми состояниями автомата, для того чтобы произвести загрузку команды и обмен данными.

Теперь ещё раз посмотрим на JTAG в картинках. На рис. 14 приведена вторая часть программы Boundary Scan Coach. На рисунке показана типовая структура узлов JTAG-сканирования, встроенных в микросхему.

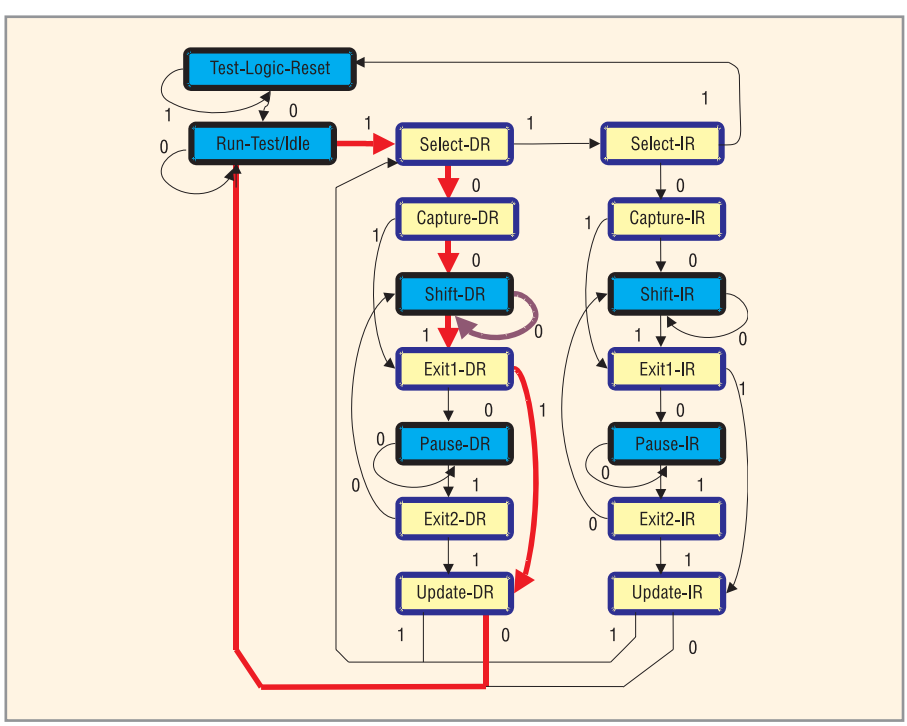

Рис. 13. Диаграмма переходов, которые необходимо выполнить для того, чтобы работать с данными

Функциональная часть микросхемы – Core Logic – показана коричневым цветом. JTAG-регистры – зелёным цветом, выводы микросхемы – серым цветом. Голубым цветом выделен контроллер JTAG-порта. Красным цветом показан путь прохождения данных по регистру граничного сканирования. Но кроме этого, в нижней части рисунка показан фрагмент, представляющий собой интерфейс мастера. Пользователь может установить данные в регистр граничного сканирования и посмотреть на результат, полученный после записи команд и данных.

Возвращаемся к BSDL-файлу

Теперь мы уже знаем, что BSDL-файл содержит описания команд, их коды и другую информацию, необходимую для работы с микросхемой. Также мы знаем, что и как выдавать в микросхему. Но как это связано с выводами микросхемы? Как читать информацию со входов микросхемы и где будет находиться нужный нам бит информации с конкретного входа?

Первая часть BSDL-файла, представляющая для нас сейчас интерес, – это порты.

Термины in bit, out bit, inout bit, buffer вполне понятны, они описывают порты, которые подключены к регистру сканирования как входы, выходы или как двунаправленные входы-выходы. Термины linkage bit, linkage bit_vector описывают порты, которые не подключены к регистру сканирования. Это те входы или выходы микросхемы, которые не могут быть проверены в режиме граничного сканирования.

Далее приведён пример описания портов микросхемы.

```
port( TDI, TMS, TCK: in bit;
port( TDI, TMS, TCK: in bit;
TDO: out bit; IN1, IN2: in bit;
OUT1: out bit; OUT2: buffer bit;
OUT3: out bit_vector (1 to 8);
OUT4: out bit_vector (4 downto
1);
```

```
BIDIR1, BIDIR2, BIDIR3: inout
bit;
GND, VCC: linkage bit);
```

Иногда производители вместо вывода микросхемы указывают PAD – технологическую контактную площадку на чипе. Далее они дают «перекодировку», указывая, какой PAD какому выводу соответствует. Иногда PAD указывается в примечании, например, вот так:

```
IO_A5:  inout bit; --   PAD6
IO_A6:  inout bit; --   PAD10
IO_A10: inout bit; --   PAD18
```

Здесь IO – тип вывода, A5 – контакт микросхемы, PAD6 – технологическая контактная площадка чипа, с которой сделана разварка проводника на вывод A5 корпуса кристалла.

Про параметр INSTRUCTION_LENGTH мы уже говорили. Вот, например, для микросхемы XC2V250 фирмы Xilinx в корпусе FG456 можно прочитать:

```
attribute BOUNDARY_LENGTH of
XC2V250_FG456 : entity is 732.
```

Таким образом, длина регистра сканирования – 732 бит, и это в 1,6 раза больше, чем число выводов. Но ведь 456 выводов – это не только сигнальные цепи, это же ещё и питание и «земля». Следовательно, мы имеем на каждый вывод 2 или 3 бита в регистре сканирования! Для BF537 в корпусе mBGA со 182 выводами мы имеем более скромные цифры – длина регистра сканирования составляет 261 бит, и это только в 1,4 раза больше, чем число выводов. Давайте посмотрим на то, как выполнен регистр сканирования. Он представляет собой последовательную цепочку из триггеров, в которые записывается информация. Если микросхема имеет вывод, который предназначен для ввода информации, то для такого вывода есть только один триггер в регистре сканирования. Аналогично и с выходами. Если нужно передать информацию из регистра сканирования на выход микросхемы, то для этого нужен только один триггер в регистре сканирования. Теперь рассмотрим двунаправленные выводы, которые могут принимать и передавать информацию. Для таких выводов двух триггеров недостаточно. Им ещё нужен третий триггер, который «сообщает» ячейке направление передачи информации. Как обычно, двунаправленная ячейка может либо принимать информацию в регистр, либо выдавать информацию из регистра. Разница этих режимов в том, что в первом случае ячейка принимает информацию, которая поступает на данный вывод «извне». А во втором случае ячейка принимает информацию, которая записана в триггере регистра и выводится этой ячейкой «наружу» Но информация, которая выводится из триггера через ячейку «наружу», может отличаться от той информации, которая есть на выводе, например, если вывод закорочен на землю, на шину питания или на другой вывод, имеющий большую мощность. В таком случае при граничном сканировании мы прочитаем информацию непосредственно с вывода микросхемы и сможем решить – та ли это информация, которую мы ожидали увидеть.

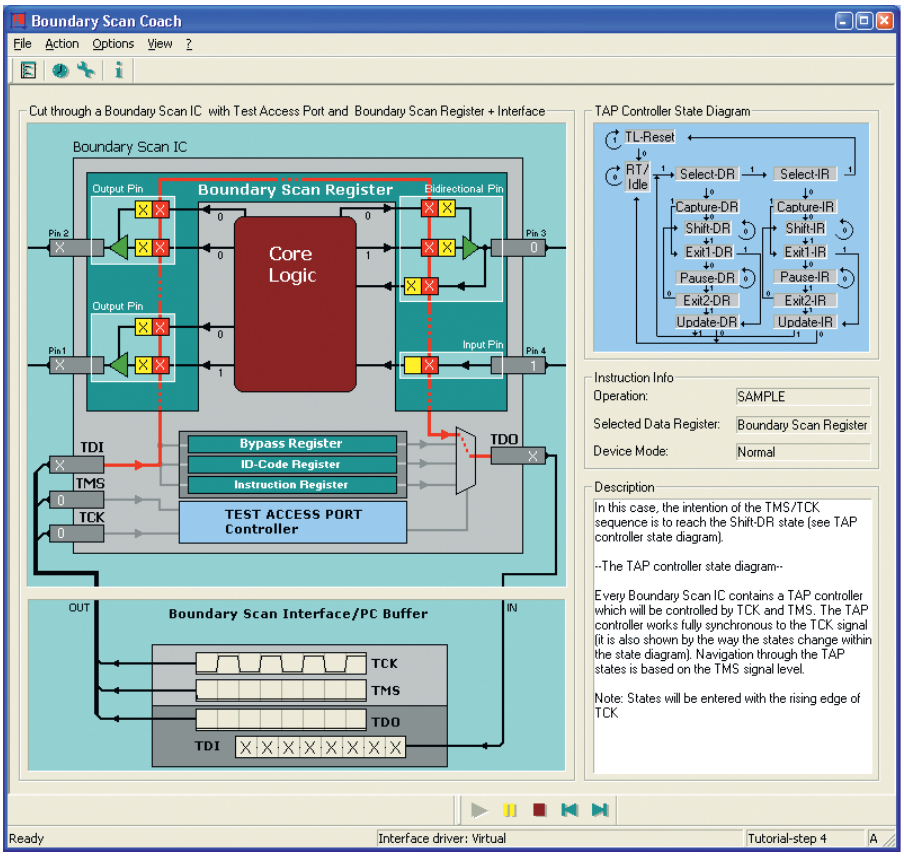

Рис. 14. Окно программы Boundary Scan Coach. Часть 2

Из сказанного выше можно сделать вывод, что «чемпионами» в длине регистра граничного сканирования всегда будут выступать микросхемы FPGA, имеющие регулярную структуру с независимыми по управлению состоянием выводами. А что же у микроконтроллеров? И в частности, у нашего «подопытного кролика»? А у микроконтроллера BlackFin – полный набор вариантов. Есть порты флагов, имеющие раздельное управление на каждый вывод, есть шины, имеющие один сигнал управления на всю шину, есть однонаправленные сигналы, а также есть и сигналы, не охваченные регистром сканирования.

Предостережение для начинающих

Необходимо обратить внимание на следующее. Фирмы – разработчики средств тестирования всегда пишут о том, что JTAG-тестирование – это передовая технология, где нет нерешённых проблем. Да, для фирм, использующих данную технологию десятилетиями, возможно, это и так. Но для тех, кто только начинает работать с портом JTAG, необходимы совершенно другие «правила игры».

Без сомнения, эта технология значительно облегчает жизнь пользователя, но только при одном условии: порт JTAG и всё что с ним связа-

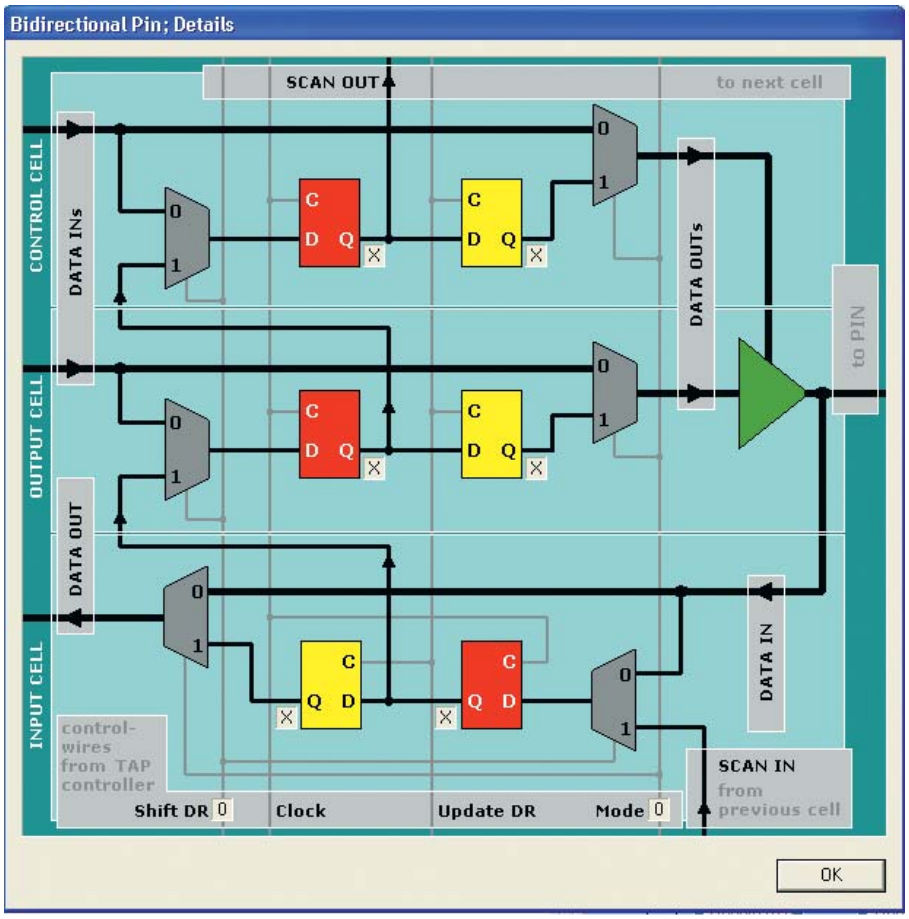

Рис. 15. Окно линзы, показывающее состояние выводов в программе Boundary Scan Coach
В этом окне показаны две типовые ячейки BC_1 и BC_2

но, а именно аппаратные адаптеры, кабели, питание и пр., – всё это должно работать безотказно! Всегда помните о том, что хотя ваше тестирующее оборудование исправно и проверено, но сама цепь JTAG, по которой передаются тестовые сигналы, находится на проверяемой плате, и эта цепь тоже может быть источником ошибок. Если цепочка бит, сдвигаемая в порт, достигает «в длину» десятков тысяч бит, то при сбое 1 бит на 1000 бит вы никогда не получите правильного результата! Мало того, из-за сдвига неверной последовательности вы можете случайным образом воздействовать на внутренние регистры микросхемы или на её выводы. А это, в свою очередь, может привести к отказу самой микросхемы или к отказу той периферии, которая управляется данной микросхемой. Поэтому необходимо принимать определённые меры безопасности при работе с портом JTAG. Ниже, при анализе битов регистра сканировани, обратите особое внимание на то, как описывается безопасное состояние и в какое состояние биты должны при сканировании устанавливаться по умолчанию.

Привязка регистра сканирования к выводам

В этой части будет дано описание «привязки» битов регистра сканирования к выводам микросхемы. Давайте обратимся к разделу BOUNDARY_CELLS файла BSDL_BF536_7_182.bsd.

```
attribute BOUNDARY_CELLS of
ADSP_BF537:  entity is
"BC_1, BC_2, BC_3, BC_4";
-- BC_1: output, control; BC_2:
input;
-- BC_3: internal; BC_4: clock;
attribute BOUNDARY_LENGTH of
ADSP_BF537: entity is 261;

attribute BOUNDARY_REGISTER of
ADSP_BF537: entity is
--num cell port function safe
(ccell disval rslt)
```

В разделе BOUNDARY_CELLS перечислены типы ячеек регистра сканирования. В соответствии с IEEE 1149.1, возможно применять в микросхемах типы ячеек от BC_1 до BC_9. Мы рассмотрим только те ячейки, которые будут применяться в файле для микроконтроллера, так как именно эти ячейки являются наибо-

Таблица 6. Функции выводов микросхемы

№	Название функции	Описание	Примечание
1	INPUT	Ячейка – вход микросхемы	Эта функция показывает, что данная ячейка имеет функцию ввода информации (контроля и ввода информации при режиме INTEST) и соединена с входом микросхемы. Эта функция может быть соединена только с выводом типа in или inout
2	OUTPUT2	Ячейка – выход микросхемы	Эта ячейка имеет 2 состояния (симметричное или несимметричное по выходным уровням сигнала) и может быть подключена к выходу, имеющему тип out, buffer, или inout
3	OUTPUT3	Ячейка – выход микросхемы	Эта ячейка имеет 3 состояния (симметричное или несимметричное по выходным уровням сигнала) и может быть подключена к выходу, имеющему тип out или inout
4	CONTROL	Ячейка используется как элемент управления для других ячеек микросхемы	Ячейка, которая управляет третьим состоянием или направлением передачи. Эта ячейка не подключается к выводам микросхемы
5	CONTROLR	Ячейка используется как элемент управления для других ячеек микросхемы	Ячейка, которая управляет третьим состоянием или направлением передачи. Эта ячейка не подключается к выводам микросхемы. Ячейка сбрасывается в состояние запрета выводов при прохождении автоматом управления TAP-контроллера состояния Test-Logic-Reset
6	INTERNAL		Ячейка не связана с сигналами микросхемы и принимает постоянные значения «1», «0» или «X»
7	CLOCK	Ячейка на входе clock	Функция, которая показывает, что данная ячейка подключена к системной тактовой частоте и это позволяет работать в режиме INTEST
8	BIDIR	Двунаправленная ячейка для двунаправленных входов-выходов	Эта ячейка двунаправленная и может быть подключена к выходу, имеющему тип inout
9	OBSERVE_ONLY	Дополнительная ячейка для приёма входных сигналов	Ячейка имеет возможность принимать сигнал как ячейка INPUT, но не поддерживает режим работы INTEST. Или же она может использоваться как дополнительная ячейка для ввода сигналов

Таблица 7. Выводы микросхемы в «запрещённом» состоянии

№	Название состояния	Описание	Примечание
1	Z	Высокоимпедансное состояние	
2	WEAK0	Внешний подпор в низкий уровень	Используются для ассимметричных выходов, таких как TTL открытый коллектор или ECL открытый эмиттер
3	WEAK1	Внешний подпор в высокий уровень	
4	PULL0	Внутренний подпор в низкий уровень	Используются для ассимметричных выходов, таких как TTL открытый коллектор или ECL открытый эмиттер при использовании внутреннего подпора
5	PULL1	Внутренний подпор в высокий уровень	
6	KEEPER	Сохранение предыдущего состояния	Вывод «запоминает» состояние

лее распространёнными «строительными кубиками».

Ячейка BC_1 «отвечает» за сигналы output и control. Это значит, что она имеет возможность переводить свой вывод в третье состояние под управлением сигнала control. Ячейка BC_2 «отвечает» за сигнал input и используется для ввода сигналов в регистр сканирования. Ячейки BC_3 и BC_4 используется для ввода служебных сигналов и сигналов тактирования.

Если вывод микросхемы двунаправленный, то это значит, что к нему подключена ячейка, содержащая в себе фактически две типовые части – ячейки BC_1 и BC_2. Пример такого узла приведен на рис. 15, где представлено одно из окон программы Boundary Scan Coach. Верхняя часть рисунка представляет собой эквивалент ячейки BC_1, а нижняя – BC_2. Из схемы этого узла видно, что с микросхемы снимается та информация, которая записана в регистре сканирования, а вводится информация непосредственно с вывода микросхемы. В том случае, если на этот вывод «встречно» подаётся другое значение сигнала, в регистр сканирования или попадает выходная или входная информация, или информация будет не определена. В данном случае всё будет зависеть от того, какой из источников сигнала – сигнал от ячейки или внешний сигнал – будет мощнее. Если один из источников сигнала будет достаточно мощным и «пересилит» другой так, что суммарный сигнал попадёт в границы допустимых по стандарту работы порта уровней, то он и определит значение, читаемое в регистре сканирования. А в том случае, если этого не произойдёт и мощности источников сигнала будут примерно одинаковые, результирующий сигнал будет не определён и в регистр сканирования может быть записано недостоверное значение.

В разделе BOUNDARY_LENGTH указано общее количество ячеек. В разделе BOUNDARY_REGISTER приведено описание подключения ячеек к выводам. В строчке комментариев приводится уточнение: num cell port function safe (ccell disval rslt). В описании регистра должны быть указаны следующие параметры:

- num – это номер ячейки (ячейка с номером «0» подключена к TDO);
- cell – это название ячейки;
- port – это тип порта ячейки;
- function – это функции выводов микросхемы;
- safe – это безопасное состояние вывода;
- ccell – это ячейка управления для ячейки вывода;
- disval – это состояние, в котором выход будет запрещен;
- rslt – это результат выдачи сигнала запрета выхода.

Теперь необходимо дать небольшие комментарии по поводу параметров и их значений. Что касается функций выводов, то они перечислены в табл. 6.

Безопасное состояние вывода в BSDL-файле называется safe bit или safe. Это то состояние, которое подаётся на выходную ячейку в режиме EXTEST или присутствует на выходе в режиме INTEST. В этом состоянии микросхема имеет минимальный ток драйвера вывода. Значения безопасного состояния вывода должны быть записаны как «0» или «1». Также возможно значение «X».

Сигнал запрета подаётся на ячейку управления, которая в BSDL-файле называется CONTROL cell или ccell.

Состояние, в котором находится сигнал запрета в BSDL-файле, называется disable value или disval. Сигнал может принимать значения «0» или «1».

Результатом установки сигнала запрета является перевод выходного буфера в одно из безопасных для работы данной микросхемы состояний.

Если сигнал запрета установлен, то вывод микросхемы может находиться в одном из состояний, перечисленных в табл. 7. Такое состояние в BSDL-файле называется disable result или rslt. Для примера рассмотрим шины и сигналы на выводах микроконтроллера BlackFin.

Продолжение следует

Литература

1. http://www.analog.com.
2. www.eltech.spb.ru.
3. Boundary Scan Coach. GOEPEL Electronic. http://www.goepel.com.
4. http://www.pld.ttu.ee/applets.
5. http://www.universalscan.com.
6. *Платунов А.Е., Постников Н.П., Чистяков А.Г.* Механизмы граничного сканирования в неоднородных микропроцессорных системах. Chip News. http://lmt.cs.ifmo.ru/article_chip_news.html.
7. *Рустинов В., Городецкий А.* Разделяй и властвуй – принцип граничного сканирования. Chip News. http://chip-news.gaw.ru/html.cgi/arhiv/01_06/stat-3.htm.
8. *Каршенбойм И.* Виртуальные кнопки и светодиоды, или Неизвестное обо всём известном JTAG_сканировании. Компоненты и технологии. 2005. № 6.
9. http://www.national.com/appinfo/scan/index.html.
10. IEEE 1149.1 (JTAG) Boundary-Scan Testing for Stratix II Devices. Altera. Ch. 9. www.altera.com.
11. *Kuznetsov D.* JTAG Boundary-Scan Test – introduction. http://www.orc.ru/~dkuzn/j_intro.htm.
12. http://jtagtools.sourceforge.net/download.html.
13. IEEE Standard Test Access Port and Boundary-Scan Architecture. IEEE Std 1149.1-2001.
14. Boundary-Scan Test and In-System Programming Software. Corelis. http://www.corelis.com/products/Test_Software.htm.
15. Serial Vector Format Specification. ASSET InterTech. Texas Instruments. www.asset-intertech.com/support/svf.pdf.
16. www.amontec.com. JTAG Interface: Common Pinouts amt_ann003 (v1.1). Application Note.
17. EIA/JEP106, JEDEC Publication 106, Standard Manufacturer's Identification Code.
18. *Каршенбойм И.* Микропроцессор своими руками/4. Как отладить встроенный в FPGA микроконтроллер? Компоненты и технологии. 2006. № 11.