электронное научно-техническое издание
НАУКА и ОБРАЗОВАНИЕ
Эл № ФС 77 - 30569. Государственная регистрация №0421100025. ISSN 1994-0408

Модель блока поддержки технологии граничного сканирования для задач тестирования цифровых схем

77-30569/403744

04, апрель 2012

Деменкова Т. А., Николаев С. А.

УДК 004.414.2

МГТУ МИРЭА
demenkova@mirea.ru

Введение

С развитием электроники наблюдается тенденция увеличения степени интеграции и миниатюризации электронных компонентов и печатных плат. Возрастает возможность ошибок проектирования и производства и усложняется процесс верификации устройств как на этапе проектирования, так и на этапе проверки качества произведенных устройств. Ошибки проектирования и производства компонентов и устройств ведут к потерям экономического характера. На их поиск и решение тратится огромное количество времени, что приводит к значительному увеличению финансирования на проектирование устройств, увеличению стоимости готовых изделий и снижению их конкурентоспособности. Все вышеизложенное заставляет производителей все больше внимания уделять тестопригодности разрабатываемых изделий [1].

Одним из эффективных решений задач тестопригодного проектирования является применение технологии граничного сканирования. Она была разработана в 80-х годах группой Joint Test Automation Group (JTAG – объединенной группой по автоматизации тестирования) и впервые стандартизована в 1990 году (IEEE Std. 1149.1 Test Access Port and Boundary-Scan Architecture). Последняя версия стандарта была опубликована в 2001 году [2].

Постановка задачи

До появления стандарта существовали различные подходы к обеспечению тестирования. Одним из таких подходов было внутрисхемное тестирование (In-Circuit testing - ICT), обеспечивающее доступ к цепям печатных плат с помощью пробников (контактных иголок). Внутрисхемное тестирование требует наличие на платах специальных контактных площадок. В связи со все возрастающей плотностью

компоновки и использованием многослойных печатных плат, а также проблемами отладки, связанными с уникальностью каждого нового типа изделия, осуществление такого подхода сталкивается со все большими проблемами. Одним из главных преимуществ граничного сканирования является обеспечение стандартного механизма решения проблем тестирования для разных задач электронной промышленности [3].

Со времени опубликования стандарта технология граничного сканирования (Boundary Scan - BS) утвердила себя как незаменимый инструмент при тестировании устройств с ограниченным доступом к выводам ИС. Широкое применение многослойных печатных плат с ИС в корпусах, изготовленных по технологиям BGA, COB и QFP, дало новый мощный импульс развитию и повсеместному применению этой технологии. Граничное сканирование используется также как средство доступа к внутренним регистрам микросхем для наблюдения за их состоянием в процессе отладки электронных плат. Исключительно широко технология BS применяется также для внутрисхемного программирования (On-Board Programming - OBP), одним из направлений которого является внутрисистемное программирование (ISP – In System Programming) [4].

В работе поставлена задача синтеза модели легко интегрируемого в различные компоненты блока, совместимого со стандартом IEEE Std. 1149.1-2001 и обеспечивающего таким устройствам доступ к возможностям граничного сканирования.

Архитектура тестового окружения

Блок поддержки граничного сканирования разработан в виде программного IP-ядра, специфицированного в качестве RTL-модели на языке описания аппаратуры Verilog HDL, стандартизированного как IEEE Std. 1364-2001 [5]. Блок обеспечивает полную совместимость со стандартом IEEE Std. 1149.1 Test Access Port and Boundary-Scan Architecture [2].

Базовая архитектура тестовой логики включает в себя следующие элементы (рис.1):

- Порт тестирования TAP (Test Access Port);
- TAP контроллер;
- Регистр команд;
- Набор тестовых регистров данных.

Операцией тестирования руководит TAP контроллер, работа которого осуществляется с помощью сигналов, подаваемых по линиям TMS, TCK и, опционально, TRST. Под управлением контроллера через порт TAP в регистр команд последовательно сдвигается код команды для исполнения. Команда декодируется, и, в зависимости от команды, декодером выбирается нужный регистр данных для подключения его между выводами TDI и TDO порта TAP. Выработка управляющих сигналов для регистров данных осуществляется TAP контроллером и декодером команд. В зависимости от команды, тестовые данные могут быть сдвинуты в

соответствующий регистр через вывод TDI, пропущены через регистр обхода (bypass) на вывод TDO. Результаты тестирования и значения регистра идентификации также могут быть выдвинуты на вывод TDO под управлением TAP контроллера и декодера команд.

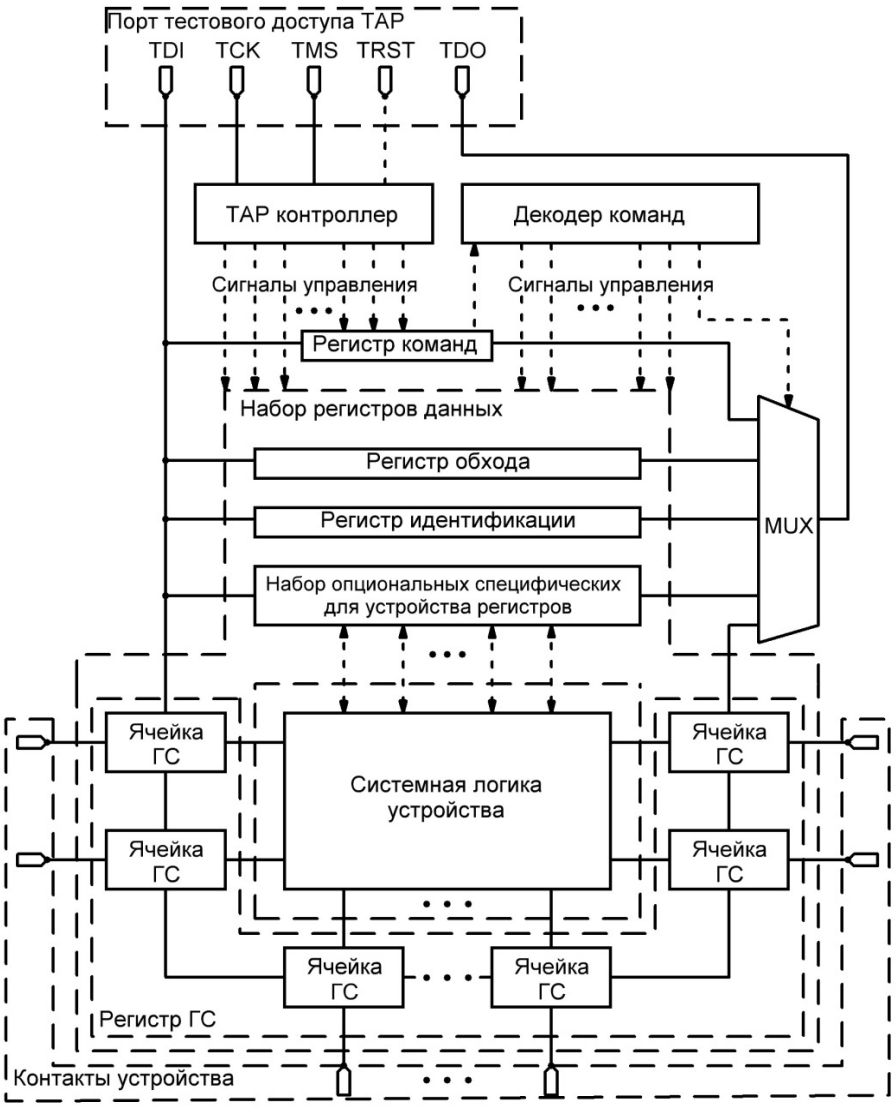

Рис. 1. Базовая архитектура тестовой логики

Порт тестирования TAP состоит из четырех обязательных и одной опциональной последовательных линий:

- TCK (Test Clock Input)– входная линия синхросигнала, управляющего всей тестирующей логикой;
- TMS (Test Mode Select Input) – входная линия сигнала управления операцией тестирования;
- TDI (Test Data Input) – входная линия сигнала тестовых данных;

- TDO (Test Data Output) – выходная линия сигнала тестовых данных;
- TRST (Test Reset Input) – опциональная входная линия ассинхронного сброса контроллера TAP.

TAP контроллер представляет собой синхронный конечный автомат, реагирующий на изменения сигналов TCK и TMS и контролирующий последовательность операций схемы тестирования.

Машина состояний TAP контроллера приведена на рис. 2.

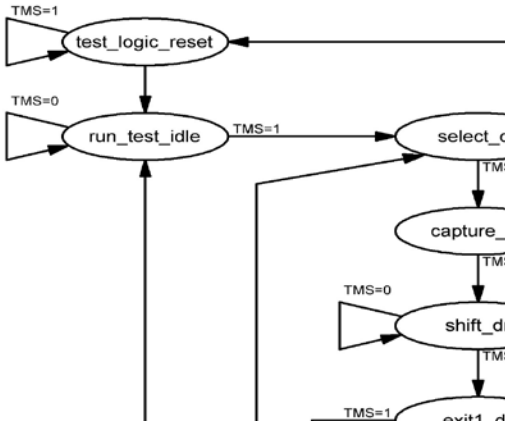

Рис. 2. Машина состояний TAP контроллера

Машину состояний условно можно разделить на две части: для доступа к регистру команд (SelectIR) и для доступа к регистру данных (SelectDR). Первая часть реализует три основные состояния, в которых происходит работа с регистром команд: CaptureIR, ShiftIR, UpdateIR. В состоянии CaptureIR на теневой сдвиговый регистр загружается константа, состоящая из двоичной последовательности 01. В состоянии ShiftIR это значение выдвигается на выходную линию TDO, при этом новое значение задвигается с линии TDI. В состоянии UpdateIR новое значение с теневого регистра фиксируется на регистре команд через параллельный выход теневого регистра. Вторая часть реализует три основные состояния, в которых происходит работа с регистром данных: CaptureDR, ShiftDR, UpdateDR. CaptureIR, ShiftIR, UpdateIR. В состоянии CaptureIR на теневой сдвиговый регистр загружаются значения диагностируемых сигналов. В состоянии ShiftIR это значение выдвигается на выходную линию TDO, при

этом новое значение задвигается с линии TDI. В состоянии UpdateIR новое значение с теневого регистра фиксируется на регистре данных через параллельный выход теневого регистра.

Остальные состояния ветвей доступа диаграммы являются либо временными, для перехода между основными состояниями (Exit1IR, Exit1DR, Exit2IR, Exit2DR), либо позволяют временно прервать работу тестирующей логики с последующим ее возобновлением (PauseIR, PauseDR). В состоянии Test-Logic-Reset тестовая логика отключена. Это состояние инициализации контроллера, в которое переходит контроллер после получения сигнала сброса на опциональной линии TRST.

Ниже перечислен набор тестовых регистров данных, которые либо должны быть обязательно включены в тестовую логику, либо могут быть включены опционально:

- Регистр граничного сканирования (Boundary-Scan Register, BSR);
- Регистр обхода (Bypass Register);
- Идентификационный регистр (Idcode Register);
- Специфические регистры конкретного устройства.

Минимально набор должен включать регистр граничного сканирования и регистр обхода bypass. Регистр граничного сканирования представляет собой сдвиговый регистр, состоящий из специальных ячеек граничного сканирования (Boundary-Scan Cell, BSC), подключаемых между внешними выводами устройства и его системной логикой. Ячейки могут пропускать через себя сигналы (и могут сохранять значения пропускаемых сигналов) или могут прекращать пропуск сигналов и устанавливать тестовые значения, полученные через порт TAP на внешние выходы или на системную логику устройства. Регистр обхода представляет собой однобитовый сдвиговый регистр, позволяющий создать кратчайший путь между выводами TDI и TDO порта TAP. Идентификационный регистр представляет собой сдвиговый регистр, содержащий 32-х битный двоичный код, позволяющий определить производителя устройства, его серийный номер и версию. Также для устройств, программируемых пользователем, в идентификационный регистр возможна запись пользовательского 32-х битного идентификационного кода. Специфические регистры могут быть включены разработчиком устройства для поддержки специфических режимов тестирования, не определенных стандартом.

Команды, которые могут выполняться тестирующей логикой, условно можно разделить на несколько групп:

- Минимальный обязательный набор команд тестирования: BYPASS, EXTEST, SAMPLE, PRELOAD;
- Опциональный набор команд регистра идентификации: IDCODE, USERCODE;
- Опциональный набор команд тестирования: INTEST, RUNBIST, CLAMP, HIGHZ.

Команда BYPASS подключает между линиями TDI и TDO регистр обхода (bypass register) и не влияет на работу системной логики устройства. Команда EXTEST подключает между линиями TDI и TDO регистр граничного сканирования и позволяет

проверить исправность внешних соединений устройства, системная логика при этом отключается от внешних соединений. Команда PRELOAD позволяет последовательно загрузить данные для тестирования в регистр граничного сканирования и захватить их. Команда SAMPLE позволяет зафиксировать в регистре граничного сканирования данные с внешних линий устройства, тем самым получив их значения в рабочем режиме для последующего анализа.

Команды регистра идентификации позволяют получить доступ к регистру идентификации, если он включен в тестовую логику. Команда IDCODE вызывает загрузку в регистр идентификации предусмотренного производителем идентификационного кода и последовательно выдвигает его с линии TDO. Команда USERCODE позволяет загрузить в идентификационный регистр определенный пользователем идентификационный код. Данная инструкция требуется, только если компонент программируется пользователем.

Команда INTEST подключает между линиями TDI и TDO регистр граничного сканирования и позволяет проверить исправность работы системной логики путем воздействия на нее сигналами, находящимися в регистре граничного сканирования. Результат тестирования затем фиксируется в регистре граничного сканирования. Команда RUNBIST позволяет активизировать встроенную в устройство собственную схему тестирования и не нуждается в предварительной загрузке внешних данных на регистр граничного сканирования. Команда CLAMP подключает между линиями TDI и TDO регистр обхода (bypass register) и устанавливает на выходные линии устройства сигналы, определяемые регистром граничного сканирования. Команда HIGHZ подключает между линиями TDI и TDO регистр обхода (bypass register) и устанавливает выходные линии устройства в Z состояние (высокого импеданса).

Блок поддержки механизма граничного сканирования

RTL модель блока разработана с помощью языка описания аппаратуры Verilog. На рис.3 представлен фрагмент кода описания машины состояний TAP контроллера для разработанного модуля.

```
139   // 2. RUN_TEST_IDLE
140   always @(posedge trst_in or posedge tck_in)
141     begin
142       if(trst_in)
143         run_test_idle <= 1'b0;
144       else if (tms_rst1 & tms_rst2 & tms_rst3 & tms_rst4 & tms_rst5)
145         run_test_idle <= 1'b0;
146       else if (~tms_in & (run_test_idle | test_logic_reset| update_dr | update_ir))
147         run_test_idle <= 1'b1;
148       else
149         run_test_idle <= 1'b0;
150     end
151
152   // 3. SELECT_DR
153   always @ (posedge trst_in or posedge tck_in)
154     begin
155       if(trst_in)
156         select_dr <= 1'b0;
157       else if (tms_rst1 & tms_rst2 & tms_rst3 & tms_rst4 & tms_rst5)
158         select_dr <= 1'b0;
159       else if (tms_in & (run_test_idle | update_dr | update_ir))
160         select_dr <= 1'b1;
161       else
162         select_dr <= 1'b0;
163     end
164
165   // 4. SELECT_IR
166   always @ (posedge trst_in or posedge tck_in)
167     begin
168       if(trst_in)
169         select_ir <= 1'b0;
170       else if (tms_rst1 & tms_rst2 & tms_rst3 & tms_rst4 & tms_rst5)
171         select_ir <= 1'b0;
172       else if (tms_in & select_dr)
```

Рис. 3. Фрагмент кода описания машины состояний TAP контроллера

Так как стандартом определены не только обязательные команды (BYPASS, SAMPLE, PRELOAD, EXTEST), но и опциональные команды (INTEST, IDCODE, USERCODE, RUNBIST, CLAMP, HIGHZ), а также из-за отсутствия четких требований к двоичным кодам команд со стороны стандарта, было принято решение реализовать параметризуемую модель. Параметризация реализуется при помощи механизма макроопределений языка Verilog. В качестве параметров выступает список опциональных команд, а также двоичные коды команд. Данное решение позволяет разработчику конечного устройства привести блок поддержки граничного сканирования в соответствие с требованиями его проекта, варьировать объем поддерживаемых опциональных команд, двоичные коды, задающие эти команды. При отсутствии необходимости в некоторых опциональных командах может быть достигнута экономия ресурсов компонентов, выражающаяся в уменьшении избыточности, вносимой в проект тестовой логикой.

В общем случае (при поддержке всех описанных в стандарте команд), структурная схема блока имеет следующий вид (рис. 4).

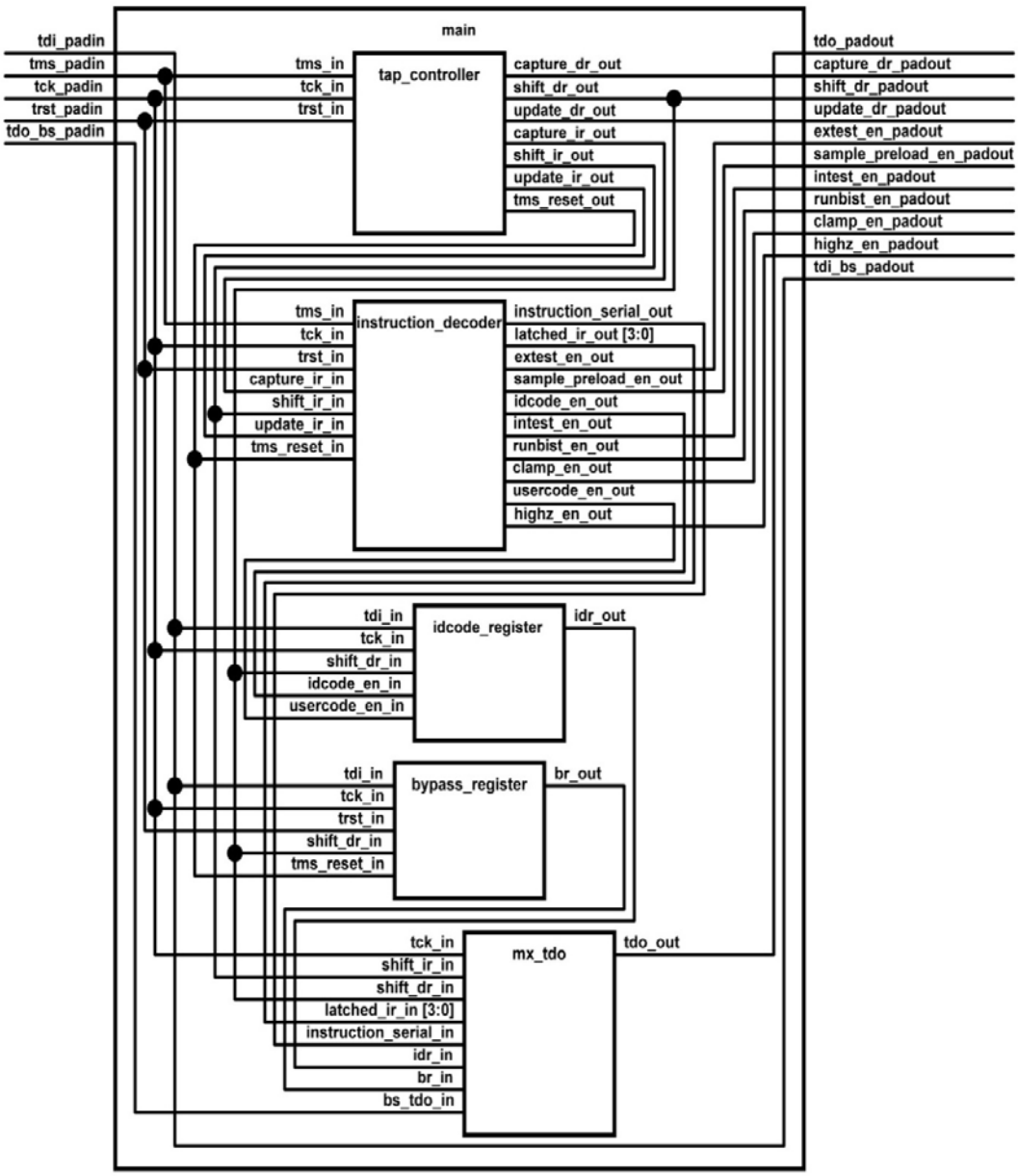

Рис. 4. Структурная схема блока поддержки граничного сканирования

Структурная схема блока поддержки граничного сканирования включает в себя пять модулей:

- tap_controller,
- instruction decoder,
- idcode_register,
- bypass_register,
- mx_tdo.

Модуль tap_controller реализует контроллер управления портом тестирования и содержит синхронный конечный автомат, диаграмма которого представлена на рис. 2 .

Модуль в ответ на сигналы, приходящие по линиям порта TAP (TCK, TMS, TRST), вырабатывает шесть сигналов, соответствующих основным рабочим состояниям машины состояний: capture_dr, shift_dr, update_dr, capture_ir, shift_ir, update_ir. Также модуль вырабатывает сигнал tms_reset, соответствующий пяти высоким состояниям сигнала TMS по восходящему фронту TCK.

Модуль instruction_decoder реализует 4-х разрядный регистр команд и декодирующую логику. Под управлением модуля tap_controller (сигналы capture_ir, shift_ir, update_ir, tms_reset) и сигнала порта TAP (TRST) с линии порта TDI в регистр команд последовательно сдвигается двоичный код команды для исполнения и фиксируется в нем на время исполнения команды. Исходя из значения кода команды, модуль вырабатывает сигналы разрешения выполнения команд (extest_en, sample_preload_en, idcode_en, intest_en, runbist_en, clamp_en, usercode_en, clamp_en) и подает их на соответствующие регистры данных. Модуль имеет последовательный выход instruction_serial для содержимого регистра команд, который подается на вход данных мультиплексора mx_tdo выходных данных и 4-х разрядный параллельный выход для зафиксированной исполняющейся команды, которая подается на управляющий вход мультиплексора mx_tdo.

Модуль idcode_register реализует идентификационный регистр. Модуль управляется сигналами порта TAP (TCK), модуля tap_controller (shift_dr) и модуля instruction_decoder (сигналы разрешения команд idcode_en, usercode_en). В зависимости от выполняемой команды, модуль может параллельно загрузить на сдвиговый 32-х разрядный регистр идентификационный код производителя и последовательно выдвинуть его на выход idr, подключенный к мультиплексору выходных данных (команда IDCODE), или сохранить на регистре-защелке предварительно загруженный на этот сдвиговый регистр пользовательский 32-х разрядный идентификационный код (команда USERCODE).

Модуль bypass_register реализует одноразрядный сдвиговый регистр обхода. Модуль управляется сигналами порта TAP (TCK, TRST) и модуля tap_controllera (shift_dr, tms_reset) и служит для пропуска тестовых данных с линии TDI на линию TDO по кратчайшему пути.

Модуль mx_tdo служит для мультиплексирования выходов регистров данных и регистра команд (instruction_serial, idr, br, bs_tdo) на выходную линию порта TAP TDO. На управляющие входы мультиплексора подается фиксированная для исполнения команда (через 4-х разрядный вход latched_ir). Данные подаются на выход мультиплексора только в состоянии TAP контроллера ShiftDR или ShiftIR. Во всех остальных состояниях модуль устанавливает выход в Z-состояние, как того требует стандарт.

Результаты моделирования

Моделирование описанного выше блока было произведено в системе моделирования и верификации ModelSim компании Mentor Graphics. Ниже (рис. 5 и

рис. 6) приведены временные диаграммы, поясняющие последовательность выполнения команд тестирования при помощи технологии граничного сканирования. Вертикальные линии на диаграммах приведены для указания перехода конечного автомата TAP контроллера между отдельными состояниями в процессе операции тестирования.

Входные данные фиксируются со входов порта TAP по восходящему фронту синхросигнала (tck_padin). Вывод данных осуществляется по нисходящему фронту синхросигнала.

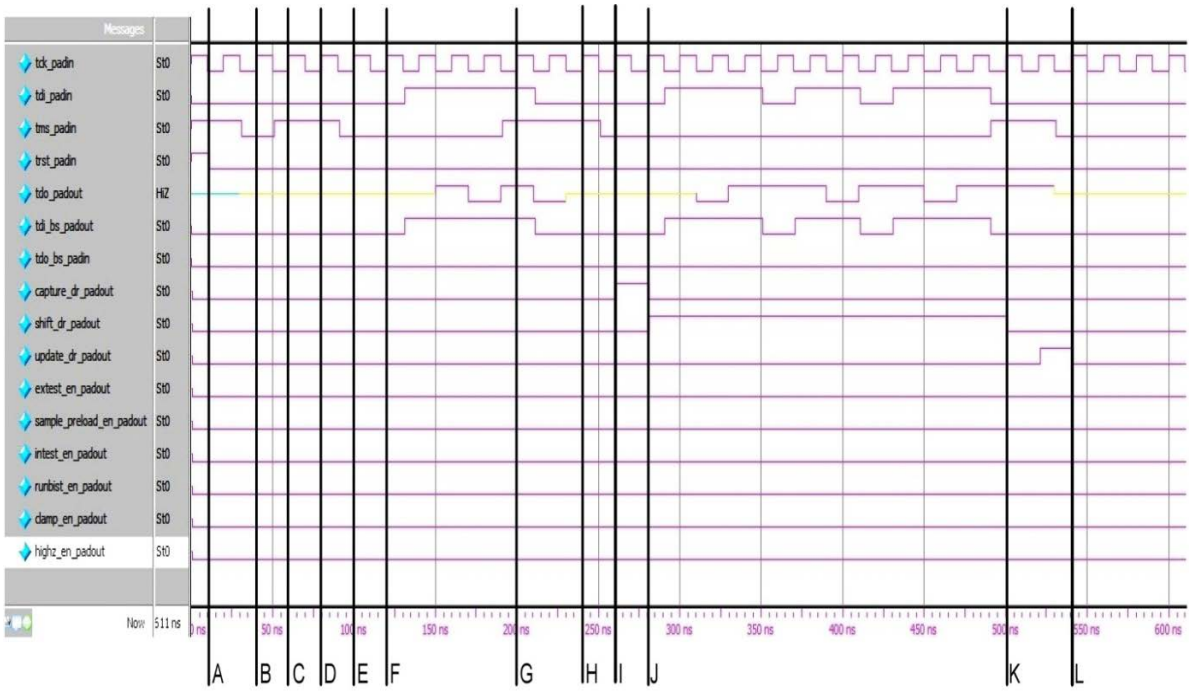

Рис. 5. Временная диаграмма выполнения команды BYPASS

Временная диаграмма выполнения команды BYPASS представлена на рис. 5. Ниже приведены пояснения к ней:

- В точке A TAP контроллер находится в состоянии Test-Logic\Reset.
- В точке B на линии TMS фиксируется низкий уровень сигнала и контроллер переходит в состояние Run-Test\Idle.
- В точке C по высокому уровню TMS контроллер переходит в состояние SelectDR.
- В точке D по высокому уровню TMS контроллер переходит в состояние SelectIR.
- В точке E по низкому уровню TMS контроллер переходит в состояние CaptureIR. На сдвиговый регистр команд загружается двоичная последовательность 0101 (как того требует стандарт).
- В точке F по низкому уровню TMS контроллер переходит в состояние ShiftIR. Новый двоичный код команды последовательно загружается на регистр команд (1111 –

команда BYPASS) по восходящему фронту TCK. По нисходящему фронту TCK с регистра команд на выход TDO выдвигается двоичная последовательность 0101.

- В точке G на вход TMS последовательно поступают две логические единицы. TAP контроллер переходит через состояние Exit1IR в состояние UpdateIR и команда, сдвинутая на регистр команд, фиксируется для исполнения. Между выводами порта TAP TDI и TDO подключается одноразрядный сдвиговый регистр обхода.
- В точке H по высокому уровню сигнала на входе TMS TAP контроллер переходит в состояние SelectDR.
- В точке I по низкому уровню сигнала на входе TMS TAP контроллер переходит в состояние CaptureDR.
- В точке J по низкому уровню сигнала на входе TMS TAP контроллер переходит в состояние ShiftDR. По восходящему фронту TCK тестовая последовательность (01110110111) с линии TDI задвигается в регистр обхода, по нисходящему фронту TCK через такт она выдвигается на линию TDO.
- В точке K на вход TMS последовательно поступают две логические единицы. TAP контроллер переходит через состояние Exit1DR в состояние UpdateDR.
- В точке L по низкому уровню сигнала на входе TMS TAP контроллер переходит в состояние Run-Test\Idle и остается в нем, пока с линии TMS по восходящему фронту TCK не поступит логическая 1.

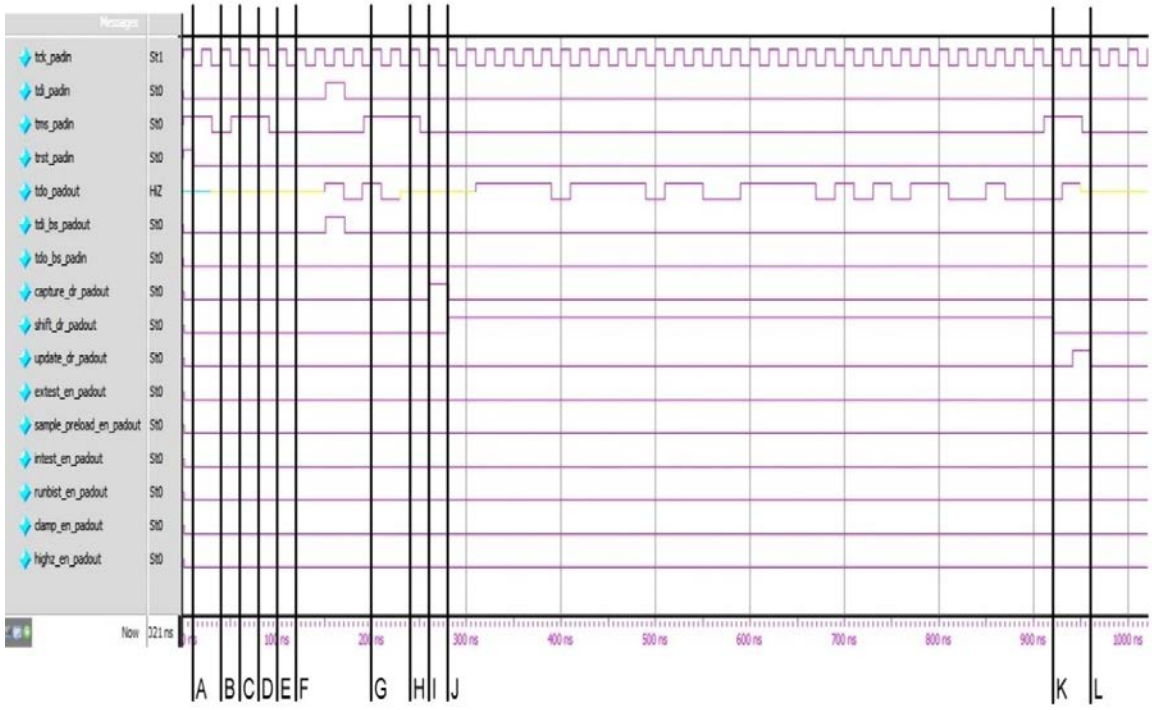

Рис. 6. Временная диаграмма выполнения команды IDCODE

На рис. 6 приведена временная диаграмма выполнения команды IDCODE. Выполнение команды IDCODE аналогично выполнению команды BYPASS за исключением:

- В точке F загружается двоичный код команды IDCODE (1000).
- В точке G между TDI и TDO подключается 32-х разрядный регистр идентификации.
- В точке I на сдвиговый регистр идентификации загружается двоичный 32-х разрядный идентификационный код производителя (для примера взят код 8'h89ABCDEF).
- В точке J на линию TDO последовательно выдвигается двоичный идентификационный код.

Методика конфигурирования блока

Ниже приведена методика конфигурирования блока под конкретные требования разработчика:

1. Провести анализ разрабатываемого проекта на возможность и необходимость использования опциональных команд тестирования. Определить двоичные коды команд тестирования.
2. Отредактировать состав макроопределений блока в соответствии с полученными данными. Все макроопределения вынесены в отдельный файл конфигурации блока и доступны для редактирования (рис. 7).
3.

```
1  ////////////////////////////////////////////////////
2
3  // 32-х битный идентификационный код
4  // Определяет версию, номер партии, производителя
5  `define IDCODE_VALUE  32'h89abcdef
6
7  // IR length - длина регистра команд
8  `define IR_LEN 4
9
10 //Макроопределения
11 //Удалить соответствующую строку для
12 //исключения поддержки команды из блока
13 `define IDCODE_ENABLE
14 `define INTEST_ENABLE
15 `define RUNBIST_ENABLE
16 `define CLAMP_ENABLE
17 `define USERCODE_ENABLE
18 `define HIGHZ_ENABLE
19
20 //Двоичные коды инструкций
21 `define EXTEST          4'b0000
22 `define SAMPLE_PRELOAD  4'b0001
23 `define BYPASS          4'b1111
24
25 `define IDCODE          4'b0010
26 `define INTEST          4'b0011
27 `define RUNBIST         4'b0100
28 `define CLAMP           4'b0110
29 `define USERCODE        4'b0111
30 `define HIGHZ           4'b1000
31
32 ////////////////////////////////////////////////////
```

Рис. 7. Файл конфигурации блока

4. В зависимости от требований, предъявляемых к проекту, разработчик может включить в состав блока дополнительные специфические регистры данных и поддержку команд, не описанные в стандарте.
5. При проектировании конечного устройства необходимо предусмотреть избыточность, вносимую блоком и предназначенную для поддержки граничного сканирования. К избыточности относятся: дополнительные выводы для организации порта тестирования, ресурсы для организации самого блока поддержки тестирования, ячейки граничного сканирования для построения регистра BSR и связи между этими компонентами.
6. Включить блок в иерархию проекта конечного устройства, к выводам устройства подключить соответствующие их типу наборы ячеек граничного сканирования

Заключение

Рассмотренная в рамках данной статьи модель многофункционального блока поддержки механизма граничного сканирования может быть легко встроена в проект разработчиками при проектировании широкого спектра устройств для обеспечения их тестопригодности. Поддержка блоком всех описанных в стандарте команд расширяет возможности его применения в самых различных проектах цифровых устройств.

Предложенный и реализованный механизм выбора поддерживаемых блоком опциональных команд стандарта, возможность изменения двоичного кода, определяющего команды при помощи макроопределений, позволяет гибко сконфигурировать блок под конкретные требования разработчика и избавиться на этапе синтеза (имплементации) от лишних для конкретной реализации элементов.

Литература

1. Городецкий А., Курилан Л. Тестопригодное проектирование схем для граничного сканирования. // Производство электроники. 2008, №1. URL: http://www.elproduct.ru/ (дата обращения: 05.12.2011).
2. IEEE Standard Test Access Port and Boundary-Scan Architecture, IEEE Computer Society, IEEE, New York, NY, IEEE Std 1149.1 - 2001.
3. Parker, Kenneth P. The Boundary-Scan Handbook Second Edition: Analog and Digital. - New York: Kluwer Academic Publishers, 2002. – 288 p.
4. Рустинов В., Городецкий А. «Разделяй и властвуй» - принцип граничного сканирования. // ChipNews 2001, №6. URL: http://www.chipinfo.ru/literature/chipnews/200106/4.html (дата обращения: 05.12.2011).
5. IEEE Standard Verilog Hardware Description Language, IEEE Computer Society, IEEE, New York, NY, IEEE Std 1364 - 2001.

electronic scientific and technical periodical
SCIENCE and EDUCATION
EL № FS 77 - 30569. №0421100025. ISSN 1994-0408

Model of support element for boundary scanning technology for the problems of digital system testing

77-30569/403744

 # 04, April 2012
Demenkova T.A., Nikolaev S.A.

MSTU MIREA
demenkova@mirea.ru

The authors consider the task of creating a model of the element easily integrated into various devices ensuring access to performance capabilities of boundary scanning. The multifunctional support element for boundary scanning was developed as IP-kernel specified as RTL-model in the Verilog Hardware Description Language. The authors describe the test logic architecture and consider the results of simulation and configuration technique of the element according to the developer's specific requirements. The proposed and implemented selection mechanism for optional CCS, the possibility of change of the binary code determining the commands through macro definitions allows to eliminate extra elements for any particular implementation at the corresponding stage.

Publications with keywords: boundary scan, modeling of the digital circuits, testing
Publications with words: boundary scan, modeling of the digital circuits, testing

References

1. Gorodetskii A., Kurilan L. Testoprigodnoe proektirovanie skhem dlia granichnogo skanirovaniia [Suitable for testing the design of schemes for boundary scan]. *Proizvodstvo elektroniki*, 2008, no.1. Available at: http://www.elproduct.ru/, accessed 05.12.2011.

2. *IEEE Standard Test Access Port and Boundary-Scan Architecture*, IEEE Computer Society, IEEE, New York, NY, IEEE Std 1149.1 - 2001.

3. Parker Kenneth P. *The Boundary-Scan Handbook Second Edition: Analog and Digital*. New York, Kluwer Academic Publishers, 2002. 288 p.

4. Rustinov V., Gorodetskii A. «Razdeliai i vlastvui» - printsip granichnogo skanirovaniia ["Divide and rule" - the principle of boundary-scan]. *ChipNews*, 2001, no.6. Available at: http://www.chipinfo.ru/literature/chipnews/200106/4.html , accessed 05.12.2011.

5. *IEEE Standard Verilog Hardware Description Language*, IEEE Computer Society, IEEE, New York, NY, IEEE Std 1364 - 2001.