ЧИТАЕМ ДНК ПЛИС

Автор: KeisN13

Рецензенты:

Aspect

Intekus

Материал подготовлен при поддержке:

КТЦ «Инлайнгруп» – дистрибьютор фирмы Xilinx (<u>www.plis.ru</u>)

ATPCenterXilinx – Сертифицированный тренинг-центр Xilinx (<u>www.plis2.ru</u>)

Оглавление

Аннотация	
Введение	3
Идентификатор и DNA кристалла	4
Модуль DNA_PORT	5
Считывание DNA	5
Режимы работы DNA_PORT	7
Разработка модуля для считывания DNA	
Сборка проекта в IP Integrator	
Моделирование	
Проверка «в железе»	
Заключение	
Домашнее задание	
Библиографический список	
Список тренингов	

Аннотация

В этой статье рассмотрен аппаратный блок DNA_PORT ПЛИС Xilinx, который хранит уникальный номер FPGA, способный помочь Вам защитить Ваши проекты от копирования и вести учёт версий и отдельных копий прошивки. Использованный демонстрационный проект собран с помощью IP Integrator, который, начиная с версии Vivado 2017.1, поддерживает схемотехническое соединение модулей, написанных на SystemVerilog, Verilog и VHDL. Приведены полное описание модуля DNA_PORT, пошаговая инструкция по сборке проекта, результаты его поведенческого моделирования и аппаратной проверки, выполненной с помощью Logic Analyzer.

Введение

Не секрет, что в FPGA существуют некоторые аппаратные модули/регистры, которые могут быть однократно запрограммированы пользователем. Такие регистры в FPGA компании Xilinx носят название eFUSE. Среди этих eFUSE регистров есть и такой, который программируется непосредственно на фабрике при изготовлении и хранит уникальный номер конкретного кристалла. Широко известно, что конфигурация молекулы ДНК (DNA) является практически уникальным идентификатором любого живого организма. Потому, по аналогии, этот регистр своих FPGA компания Xilinx назвала «DNA». А аппаратный модуль, позволяющий считать DNA кристалла, был назван DNA_PORT.

Уникальность аппаратного блока DNA_PORT в том, что считанные им данные доступны не только через интерфейс JTAG, но и непосредственно для проекта, залитого в FPGA. Остальные eFUSE-регистры такой возможности не имеют: их содержимое может быть считано исключительно через JTAG.

Поскольку DNA – это уникальный номер FPGA (повтор возможен, но только у 32 микросхем во всем семействе, и при этом не внутри подсемейства), то он может быть применён для различного круга задач: от защиты проекта от копирования на другие кристаллы до помощи в понимании текущей версии прошивки.

3

Цель статьи – дать общее представление о модуле DNA_PORT. Задачи, которые поставлены в этой статье:

- 1. Разработать модуль для чтения уникального DNA-номера FPGA.
- 2. Выполнить моделирование.
- 3. Провести имплементацию проекта.
- 4. Проверить корректность работы модуля в логическом анализаторе на реальном кристалле, установленном на имеющейся в нашем распоряжении макетной плате Arty Board.

При описании будем ориентироваться на 7-ое семейство FPGA Xilinx, поскольку на Arty Board установлена ПЛИС Artix-7, хотя и в других семействах тоже имеется DNA_PORT. За основу будут взяты следующие руководства пользователя [1, 2, 3].

Идентификатор и DNA кристалла

В кристаллах 7-ого семейства компании Xilinx имеется специализированный аппаратный блок, хранящий уникальный 64-битный идентификатор (FUSE_DNA). Он может быть использован проектом для получения 57-битного значения DNA Идентификатор В энергонезависимой кристалла. хранится памяти И программируется однократно при производстве кристалла, а далее является неизменным. Этот номер является уникальным для каждого экземпляра FPGA 7-ого семейства. Однако внутри семейства – но не внутри подсемейства – возможно появление 32 экземпляров FPGA с одинаковым номером. Это означает, что внутри подсемейства Artix-7 не может быть двух FPGA, у которых совпадут DNA, но могут быть совпадения между, например, экземплярами Kintex-7 и Artix-7. Чтение полного 64-битного номера FPGA (FUSE_DNA) возможно только с помощью JTAG. При этом 57-битный DNA-номер, доступный для чтения прошивкой при помощи модуля DNA_PORT, расположен с 63 по 7 биты FUSE_DNA.

Модуль DNA_PORT

Аппаратный модуль DNA_PORT позволяет получить доступ к DNA-номеру кристалла. Различные варианты графического отображения соответствующего блока приведены на рисунке 1.



Рисунок1 – различные представления модуля DNA_PORT

Как видим, аппаратный модуль DNA_PORT имеет всего 5 портов:

- DIN порт ввода пользовательских данных. Пользователи могут дополнять считываемый номер DNA своими данными, если это необходимо. См. далее схемы включения DNA_PORT.
- READ порт команды чтения данных из постоянного регистра во временный, из которого данные могут быть считаны пользователем.
- SHIFT порт команды выполнения сдвига DNA-номера. Временный регистр, доступный пользователю для чтения, представляет собой простой сдвиговый регистр.
- СLК вход тактового сигнала.
- DOUT выход данных со сдвигового регистра.

Считывание DNA

На рисунке 2 показано устройство и функционал модуля DNA_PORT.



Рисунок2 – функционал модуля DNA_PORT

Для того чтобы выполнить считывание DNA кристалла, необходимо выставить значение READ в «1» на один такт CLK, после чего DNA будет передан во временный сдвиговый регистр, который доступен для считывания пользователю. Для выполнения операции сдвига в сдвиговом регистре необходимо управлять входом SHIFT. Ниже приведена таблица с операциями, доступными для модуля DNA_PORT.

Operation	DIN	READ	SHIFT	CLK	Shift Register	DOUT
HOLD	х	0	0	х	Hold previous value	Hold previous value
READ	х	1	х	Ŷ	Parallel load with 57-bit DNA	Bit 56 of Identifier
SHIFT	DIN	0	1	Ŷ	Shift DIN into bit 0, shift contents of Shift Register toward DOUT	Bit 56 of Shift Register

Notes:

X = Don't care

 \uparrow = Rising clock edge

Рисунок3 – список допустимых операций модуля DNA_PORT

Одним достаточно важным ограничением является тактовая частота работы модуля DNA_PORT. Для того чтобы найти диапазон допустимых значений тактовой частоты работы модуля, необходимо обратиться к [3]. Согласно таблице на рисунке 4, максимальная рабочая частота чтения DNA для нашего кристалла на Arty Board составляет 100 МГц. Учтите эти ограничения, если вы будете использовать DNA_PORT: для других семейств значения могут отличаться.

		Speed Grade					
Symbol	Description		1.0V		0.95V	0.9V	Units
		-3	-2/-2LE	-1	-1LI	-2LE	
Device DNA	Access Port						
F _{DNACK}	DNA access port (DNA_PORT)	100.00	100.00	100.00	100.00	70.00	MHz, Max

Рисунок4 – значения максимальной рабочей частоты модуля DNA_PORT 6

Режимы работы DNA_PORT

Считываемое с помощью DNA_PORT уникальное значение DNA может быть расширено пользователем. Для этого используется одна из следующих трёх схем включения данного модуля:

1. Наиболее часто используемая простейшая схема включения: на вход DIN подаётся константа «0» или «1»:



Рисунок5 – схема включения модуля DNA_PORT с дополнением константой

2. Расширенная схема включения, в которой вход DIN и выход DOUT образуют обратную связь, неограниченно расширяя тем самым считываемое значение DNA его же циклическими копиями:



Рисунок6 – схема включения модуля DNA_PORT в режиме зацикливания

3. Расширенная схема включения, в которой считываемый номер DNA расширяется произвольными пользовательскими данными:



Рисунок7 – включение модуля DNA_PORT в режиме дополнения пользовательскими данными

Разработка модуля для считывания DNA

Поскольку аппаратный модуль DNA_PORT позволяет получить доступ к DNA, нам необходимо разработать модуль, который бы управлял им. Согласно таблице на рисунке 3, логика функционирования разрабатываемого модуля оказывается достаточно несложной, так что разумнее всего реализовать его в виде конечного автомата, последовательно проходящего следующие состояния:

- 1. Ожидание сигнала старта для считывания DNA.
- 2. Установка сигнала READ на один такт для записи DNA во временный сдвиговый регистр.
- 3. Управление сигналом SHIFT для сдвига значений и их записи.
- 4. Запись текущего значения в другой сдвиговый регистр для параллельной выдачи результата.
- 5. Установка сигнала DONE, сообщающего о завершении чтения DNA.

Модуль, реализующий конечный автомат и параллельную выдачу значения DNA, приведён в листинге 1. Реализовывать конечный автомат и последующее подключение модулей мы будем для схемы на рисунке 5.

СКАЧАТЬ ФАЙЛ ЛИСТИНГА

Листинг 1

8

Но для полноценной работы нам необходимы ещё 2 модуля:

- 1. Экземпляр непосредственно самого DNA_PORT.
- 2. Блок формирования и управления тактовой частотой.

Для добавления в код экземпляра модуля необходимо воспользоваться шаблонами для вставки, которые можно найти либо в [Ошибка! Источник ссылки не найден.], либо в подборке языковых шаблонов (Language Templates) Vivado:

🚴 dna_reader - [C:/Projects/FPG/	A-Systems/dna_reader/project/dna_reader/dr	na_reader.x
<u>File Edit Flow Tools</u>	<u>W</u> indow Layout <u>V</u> iew <u>H</u> elp	Q- Qui
🕞 🛧 🖈 🗉 🗍	Report	•
Flow Navigator ₹ ♦	Create and Package New IP Create Interface Definition	
 Settings Add Sources Language Template IP Catalog 	Enable <u>P</u> artial Reconfiguration <u>R</u> un Tcl Script Property Editor Associate EL <u>F</u> Files Generate Memory Config <u>u</u> ration File	Ctrl+J
 IP INTEGRATOR Create Block Desig Open Block Design 	Xilinx Icl Store Custom Commands	•
Generate Block Des	Language <u>T</u> emplates Settings	

Рисунок8 –Доступ к шаблонам Vivado

t a language template	
nplates	Preview
L ≍ ≑ <u></u> 2+	19
	20 Library UNISIM;
arch: Qr dna_port (14 matches) 21 use UNISIM.Vcomponents.all;
	22 and compared to the balant this line and pasts into the architecture body and
> 🗁 Kintex UltraScale	
> 📄 Kintex UltraScale+	25 DNA PORT: Device DNA Access Port
> Finter-7	26 Artix-7
	27 Xilinx HDL Language Template, version 2017.3
> Virtex UltraScale	28
> Virtex UltraScale+	29 DNA_PORT_inst : DNA_PORT
> 🗁 Virtex-7	30 generic map (
VHDL	31 SIM_DNH_VALOE => X*0000000000000000000 Specifies a sample 5/-Dit DNA Value for simul
Device Primitive Instantiation	33 port map (
✓ ☐ Artix-7	34 DOUT => DOUT, 1-bit output: DNA output data.
× Config/BSCAN Components	35 CLK => CLK, 1-bit input: Clock input.
	36 DIN => DIN, 1-bit input: User data input pin.
Device DNA Access Port (DNA_PORT)	37 READ => READ, 1-bit input: Active high load DNA, active low read input.
> 📄 Kintex UltraScale	38 SHIFT => SHIFT 1-bit input: Active high shift enable input.
> E Kintex UltraScale+	39);
> 🗁 Kintex-7	40 '
Notex Litre Ceele	

Рисунок9 – Шаблоны модуля DNA_PORT

Этот шаблон вставляется в модуль, который называется dna_port_wrapper. Его код приведён в листинге 2:

СКАЧАТЬ ФАЙЛ ЛИСТИНГА 2

Листинг 2

Сборка проекта в IP Integrator

Начиная с версии Vivado 2017.1 в IP Integrator (который мы использовали для сборки процессорной системы на базе <u>Microblaze</u>) появилась возможность выполнять подключение не только блоков IP, но и произвольных модулей, описанных на RTL-уровне. Для того, чтобы это сделать, создадим новый Block Design:

FlowNavigator \rightarrow IP INTEGRATOR \rightarrow Create Block Design:



Рисунок10 – Меню для создания Block Design

Вводим имя Block Design: «dna_reader_top»

À Create Block Design		×
Please specify name	of block design.	4
<u>D</u> esign name:	dna_reader_top	\otimes
Directory:	🛜 <local project="" to=""></local>	~
Specify source set:	🗅 Design Sources	~
?	ОК	Cancel

Рисунок11 –Задание имени для BlockDesign

Для добавления модулей на рабочее поле, необходимо во вкладке Sources выбрать RTL-модуль, который мы хотим добавить, кликнуть правой кнопкой мыши и выбрать Add Module to Block Design:

www.FPGA-Systems.ru



Рисунок12 – Вызов меню добавления RTL модуля в BlockDesign

После этого на рабочем поле Diagram появится выбранный модуль, с надписью «RTL» посередине:



Рисунок13 –Экземпляр RTL-модуля dna_reader_fsm

Таким же образом добавляем модуль dna_port_wrapper:



РисунокОшибка! Закладка не определена. – экземпляр модуля dna_port_wrapper

Теперь добавляем модуль управления тактовой частотой и синхронизацией. Кликаем правой кнопкой в пустом месте на поле Diagram, нажимаем Add IP, в поле поиска вводим Clocking Wizard. Должен появиться следующий IP:



Рисунок14 – IP-блок управления тактовой частотой и синхронизацией

Выполним настройку тактовых частот:

- 1. Дважды кликаем по символу clk_wiz_0.
- 2. Устанавливаем входную тактовую частоту 100 МГц.

👃 Re-customize IP		×
Clocking Wizard (5.4)		4
1 Documentation 🛛 📄 IP Location		
IP Symbol Resource	Component Name clk_wiz_0	
Show disabled ports	Clocking Options Output Clocks MMCM Settings Summary MMCM PLL 	^
	Clocking Features Jitter Optimization	
	✓ Frequency Synthesis	
	Phase Alignment Spread Spectrum Minimize Output Jitter	
	Dynamic Reconfig Dynamic Phase Shift OMaximize Input Jitter filter	ing
	Safe Clock Startup	
- reset clk_out1 - clk_in1 locked -	Dynamic Reconfig Interface Options Phase Duty Cycle Config Write DRP registers AXI4Lite DRP	
	Input Clock Information	
	Input Clock Port Name Input Frequency(MHz) Jitter	Options Input Jitter
	Primary Clk_in1 [100.000 10.000 UI	▼ 0.010
	Secondary Clk_in2 100.000 60.000 - 120.000	0.010
		, , , , , , , , , , , , , , , , , , ,
	C	>
	[OK Cancel

Рисунок15 – Настройка входной тактовой частоты

3. Устанавливаем выходную тактовую частоту во вкладке Output Clocks: clk_out1- 50МГц.

👃 Re-customize IP							
Clocking Wizard (5.4) Documentation IP Location							4
IP Symbol Resource	Component Name	clk_wiz_0					
Show disabled ports	Clocking Options	Output Clock	s MMCM Settings	Summary			
	The phase is calc	ulated relative to	the active input clock				Î
	Output Clock	Port Name	Output Freq (MHz) Requested	Actual	Phase (degrees) Requested	Actual	Duty Cycle Requested
- reset clk_out1 -	✓ clk_out1	clk_out1	50 🛞	100.000	0.000	0.000	50.000
clk_in1 locked	Clk_out2	clk_out2	100.000	N/A	0.000	N/A	50.000
	clk_out3	clk_out3	100.000	N/A	0.000	N/A	50.000
	Clk_out4	clk_out4	100.000	N/A	0.000	N/A	50.000
		alls outE	100.000	NIA	0.000	ΝΙ/Α	50.000 ×

Рисунок16 – Настройка выходной тактовой частоты

4. Пролистываем вниз и снимаем галочку для отключения входа reset

À Re-customize IP		×
Clocking Wizard (5.4)		4
🚯 Documentation 🛛 🕞 IP Location		
IP Symbol Resource	Component Name clk_wiz_0	
Show disabled ports	Clocking Options Output Clocks MMCM Settings Summary clk_out7 1	^
- clk_in1 - locked -	Enable Optional Inputs / Outputs for MMCM/PLL Reset Type reset power_down input_clk_stopped I locked clkfbstopped	
	ок с	ancel

Рисунок17 – Отключение входа reset

5. Нажимаем ОК.

Поскольку мы реализуем включение по схеме с рисунка 5, необходимо добавить IP блок Constant и установить его значение в «0»:

À Re-customize IP		×	
Constant (1.1)			🎤 🗹 C Y
			on
🜖 Documentation 🛛 🖨 IP Location			
Show disabled ports	Component Name xlconstant_0		
			xlconstant_0
	Const Val		dout[0:0]
			Constant
dout[0:0]			

Рисунок18 – Установка значения IP блока constantв 0

Выполним подключение внешних выводов и соединение блоков. Сделаем вход clk_in1 модуля clk_wiz_0 внешним:

- 1. Кликаем правой кнопкой по символу clk_in1.
- 2. Выбираем Make External.



Рисунок19 – Объявление порта clk_in1внешним

После этого должен появиться порт clk_in1_0:



Рисунок20 – Созданный внешний порт для clk_in1

Переименуйте clk_in1_0 в clk_in1. Это можно сделать в окне External Port Properties, доступном по нажатию на clk_in1_0.



Рисунок21 – Переименование порта

Выполните аналогичные шаги для выводов istart, odna_ready и odna[63:0] модуля dna_reader_fsm_0, чтобы получился результат, как на рисунке 22 (для оптимизации рабочего пространства нажмите кнопку Regenerate Layout, доступную на панели инструментов):



Рисунок22 – Объявление внешними нескольких портов проекта

Теперь мы можем выполнить подключение модулей. Окончательный вариант выглядит, как показано на рисунке 23:



Рисунок23 – схема соединения модулей

Для выполнения DRC в Block Design, нажмите на кнопку Validate Design. Если всё было выполнено корректно, появится сообщение, как на рисунке 24. Если же появились ошибки, внимательно прочитайте текст пояснений и постарайтесь исправить ошибки самостоятельно.



Рисунок24 – проверка на Block Design на ошибки



Сохраните проект, нажав на кнопу Save:

Рисунок25 – сохранение результатов

Обратите внимание, что модули RTL пока что не появляются в иерархии Block Design. Необходимо создать обёртку для Block Design «dna_reader_top», кликнув по нему правой кнопкой мыши и выбрав Create HDL Wrapper:



Рисунок26 – Создание обёртки для Block Design

После этого дерево иерархии проекта обновится.

www.FPGA-Systems.ru

BLOCK DESIGN - dna_reader_top *	
Sources × Design Signals	? _ 🗆 🖒
Q 素 ♦ + ? ● 0	٥
Design Sources (2)	
> 🐠 dna_port_wrapper(structural) (dna_p	ort_wrapper.vhd) (
> <a>> dna_reader_top_wrapper(STRUCTURE	E) (dna_reader_top
> 🗁 Constraints	
> 📄 Simulation Sources (2)	
	> · · · · · · · · · · · · · · · · · · ·
Hierarchy IP Sources Libraries Compi	le Order

Рисунок27 – Окно иерархии проекта

Теперь пометим модуль dna_reader_top_wrapper в качестве модуля верхнего уровня – сделаем его собственно «топовым». Для этого кликаем по нему правой кнопкой мыши и выбираем Set as Top:

Sources ×	Design	Sig	nals ? _ 🗆 🖸	Diagram
Q ₹ \$	+	?-	• 0 ¢	ପ୍ ପ୍
🗸 📄 Design S	ources (2		Source Node Properties	Ctrl+E
> 🐠 📥 dna	_port_w		Open File	Alt+O
> 🐨 dna_	reader_to nts		Replace File	
> 📄 Simulatio	n Source		Copy File Into Project	
			Copy All Files Into Project	Alt+I
	_	×	Remove File from Project	Delete
Hierarchy IF	P Sources		Enable File	Alt+Equals
			Disable File	Alt+Minus
Source File Pro	perties		Move to Simulation Sources	
谢 dna_reader_	top_wrap		Move to Design Sources	
			Hierarchy Update	•
Enabled		С	Refresh Hierarchy	
Location:	C:/Pro		IP Hierarchy	•
Type:	VHDI		Set as Top	
			Out Ola hall hadred.	

Рисунок28 – Задание модуля верхнего уровня

Теперь иерархия проекта выглядит следующим образом:



Рисунок29 – окно иерархии проекта

Как видите, для каждого RTL-модуля создаётся обёртка. Вы можете менять содержимое RTL-модулей, то есть редактировать их. Только не забывайте при этом делать обновление моделей в Block Design. Если содержимое RTL-модуля было изменено, появится соответствующее сообщение.

Моделирование

Наш проект собран, и теперь мы можем выполнить его моделирование (симуляцию). Если Вы обратили внимание, то при моделировании Вы можете указать значение, которое будет выдавать DNA_PORT. Для его модели оно является параметром.

```
SIM_DNA_VALUE => X"1D01234567890a2" -- Specifies a sample 57-bit DNA value for simulation
```

Рисунок30 – Значение DNA для моделирования

Код модуля тестирования (тестбенча) приведён в листинге 3: <u>СКАЧАТЬ ФАЙЛ ЛИСТИНГА 3</u>

Листинг 3. Код тестбенча



Результаты моделирования приведены на рисунке 31.

Рисунок31 – Результаты моделирования проекта

Проверка «в железе»

После удачного моделирования необходимо выполнить проверку модуля на аппаратном уровне – «в железе». Делать мы это будем с помощью логического анализатора Logic Analyzer. Но прежде, чем к этому приступить, нам необходимо сказать Vivado, какие цепи мы хотим наблюдать и отлаживать в логическом анализаторе. Нас будут интересовать istart, odna_ready и odna[63:0]. Чтобы их добавить в логический анализатор:

- 1. Выберите цепь.
- 2. Нажмите на ней правой кнопкой мыши.
- 3. Выберите из появившегося меню пункт Debug.



Рисунок32 – Отмечаем цепи для отладки

Повторите эти действия для odna_ready и istart. Сохраните текущие изменения.

Однако, перед выполнением синтеза нам предстоит создать ещё один модуль верхнего уровня. Причина, по которой необходимо это сделать: сейчас у нас внешними (то есть теми, которые будут подключены к ножкам ПЛИС) являются 67 выводов модуля: iclk, istart, odna_ready и odna[63:0]. После синтеза и имплементации они должны быть физически подключены к ножкам ПЛИС – в противном случае мы не сможем получить файл прошивки. Но на Arty Board нет 67 свободных портов – да и они нам не нужны. Нам нужны только iclk и istart – поэтому мы создаём новый модуль, в котором наружу подключаем только эти два вывода. Листинг модуля top приведён ниже:

СКАЧАТЬ ФАЙЛ ЛИСТИНГА 4

Листниг 4. Код модуля top

Поскольку мы отметили сигналы для отладки, то они не будут выборшены в ходе оптимизации во время синтеза, и мы сможем наблюдать значения на них.

Запускаем синтез:



Рисунок33 – Запуск синтеза проекта



После окончания синтеза откройте синтезированный проект:

Рисунок34 – Открытие результатов синтеза

Обратите внимание: несмотря на то, что цепи от выводов odna[63:0] и odna_ready не подключены к внешним портам ввода/вывода, они не были исключены из проекта в ходе оптимизации, поскольку были отмечены для отладки в логическом анализаторе.



Рисунок35 – Netlist проекта

Теперь выполним настройку логического анализатора:

1. Откройте окно отмеченных для отладки цепей (Window \rightarrow Debug):



Рисунок36 – Открытие окна со списком цепей для отладки

2. Запустите мастер настройки логического анализатора, нажав кнопку Setup Debug:

Tcl Console Messages Log Reports Design Runs Debug ×			
Q 素 ≑ ★ + ≓			
Name Set Up Debug	Driver Cell	Driver Pin	Probe Type
✓ ☐ Unassigned Debug Nets (66)			
> Jro inst_dna_reader/dna_reader_top_i/dna_reader_fsm_0_odna (64)	Multiple	Multiple	
∫ [©] inst_dna_reader/dna_reader_top_i/istart_0_1	IBUF	0	
∫ [©] inst_dna_reader/dna_reader_top_i/dna_reader_fsm_0_odna_ready	FDRE	Q	

Рисунок37 – Окно со списком цепей для отладки

3. После появления мастера настройки нажмите Next. Перед вами появится окно настройки соответствия цепей и тактовых доменов, в котором Вы

можете указать, по какому тактовому сигналу будет производиться выборка сигнала. Поскольку тактовый домен у нас один, оставляем всё без изменений. Нажимаем Next:

≽ Set Up Debug	×
Nets to Debug The nets below will be debugged with ILA cores. To add nets click "Find Nets to Ad windows, then drag them to the list or click "Add Selected Nets".	d". You can also select nets in the Netlist or other
Q 素 € M2 M1 + −	0
Name	Clock Domain
> fir* inst_dna_reader/dna_reader_top_i/dna_reader_fsm_0_odna (64)	inst_dna_reader/dna_reader_top_i/clk_wiz_0/inst/clk_out1
「	inst_dna_reader/dna_reader_top_i/clk_wiz_0/inst/clk_out1
「	inst_dna_reader/dna_reader_top_i/clk_wiz_0/inst/clk_out1
Find Nets to Add	Nets to debug: 66
?	Back Next > Finish Cancel

Рисунок38 – Окно соответствия цепей и тактового домена

4. В этом окне указывается длина выборки – сколько отсчётов каждого сигнала нужно записать. Чем больше отсчётов, тем больше данных мы можем получить за один запуск логического анализатора, но и тем больше встроенной RAM ПЛИС логическому анализатору потребуется. Установите значение в «4096».

≽ Set Up Debug	×
ILA Core Options Choose features for the ILA debug cores.	
Sample of data depth: 4096 V Input pipe stages: 0 V Trigger and Storage Settings	
Capture control Advanced trigger	
?	< Back Next > Finish Cancel

Рисунок39 – Установка длины выборки

- 5. Нажмите Next и затем Finish.
- Сохраните сделанные изменения. Поскольку файл ограничений мы не создавали, Vivado предложит создать его и сохранить изменения в этот файл. Нажмите кнопку сохранения и затем ОК.



Рисунок40 – Созданный автоматически файл проектных ограничений

Настройка логического анализатора закончена. Теперь назначим ножки для iclk и istart.

Для назначения ножек откройте окно I/O Ports (Window \rightarrow I/O Ports)

www.FPGA-Systems.ru

1	<u>W</u> indow		Layout	View	<u>H</u> elp	Q- Quie
	Σ	Pr	oject Su <u>m</u> r	mary		4
		<u>S</u> 0	urces			-1
	ę	IP	<u>C</u> atalog			
		Pr	op <u>e</u> rties		Ctrl+	
		Se	lection			
		Ne	etlist			
		De	evi <u>c</u> e Cons	traints		
		Ph	ysical Con	straints		
		Tir	ming <u>C</u> ons	traints		
		<u>P</u> a	ckage Pin	S		
		I/ <u>C</u>	Ports			
		CI	ock Region	10		np

Рисунок41 – Вызов окна с настроек портов ввода/вывода

Установите значения для iclk и istart в соответствии с рисунком 42 и сохраните изменения.

Tcl Console Messages Log	Reports Design Run	ns Debug	I/O Ports ×						
Q 素 ≑ K + H									
Name	Direction	Neg Diff Pair	Package Pin	Fixed	Bank	I/O Std	Vcco	Vref	Driv
✓ i All ports (2)									
CLK.CLK_IN1_21659 (1)	IN			\checkmark	35	LVCMOS33* ·	3.300		
 Scalar ports (1) 									
Ick	IN		E3 ~	\checkmark	35	LVCMOS33*	3.300		
 Scalar ports (1) 									
☑> istart	IN		D9 ~	\checkmark	16	LVCMOS33*	3.300		
<									

Рисунок42 – Настройки внешних портов для Arty Board

Теперь мы можем запустить имплементацию проекта и генерацию файла прошивки (bitstream). Нажимаем Generate Bitstream и ждём окончания выполнения операции.



Рисунок43 – Запуск имплементации и генерации файла прошивки

После окончания генерации файла прошивки FPGA открываем Hardware Manager:



Рисунок44 – Открытие Hardware Manager

Подключите Arty Board к компьютеру.

Нажимаем Open target \rightarrow Auto Connect:

HARDWARE MANAGER - unconnect	ted		
🚯 No hardware target is open. Op	en tar	get	
Hardwaro	ø	Auto Connect	
Hardware		Recent Targets	Þ
$ \mathcal{Q} \cong \oplus \mathbb{P} $		Available Targets on Server	ŀ
		Open New Target	

Рисунок45 – Обнаружение подключённых устройств

После сканирования кристалл, установленный на Arty, появится в списке подключённых устройств.



Рисунок46 – Список обнаруженных устройств

Для прошивки нашего кристалла нажимаем правой кнопкой мыши по обнаруженному устройству и выбираем Program Device:

Hardware	?	_ 🗆 🖾 🗙			
Q ¥ ♦ Ø ▶	> 🔳	0			
Name		Status			
 Iocalhost (1) 		Connected			
✓ ✓ ✓ ✓ Ø xilinx_tcf/Digilent/21	03197897	Open			
✓	Hardware De	evice Properties	Ctri+E		
· ADC (3	Program Dev	vice			
	Verify Device				
Hardware Device Prope	Refresh Device				
🛑 xc7a35t_0	Add Configur	ation Memory Device	vice		

Рисунок47 – вызов окна программирования кристалла

В появившемся окне необходимо указать bit-фал прошивки FPGA и список цепей, которые мы отметили для отладки. Как правило, эти поля заполняются автоматически, но на всякий случай проверьте, что подключаются именно нужные файлы.

À Program Device	×
Select a bitstream prog that corresponds to the	ramming file and download it to your hardware device. You can optionally select a debug probes file debug cores contained in the bitstream programming file.
Bitstream file:	C:/Projects/FPGA-Systems/dna_reader/project/dna_reader/dna_reader.runs/impl_1/top.bit
Debu <u>a</u> probes file:	C:/Projects/FPGA-Systems/dna_reader/project/dna_reader/dna_reader.runs/impl_1/top.ltx
Enable end of s	tartup check
?	Program Cancel

Рисунок48 – Окно выбора файла прошивки и файла со списком цепей для отладки

После окончания прошивки Vivado изменит представление на приспособленное для работы с логическим анализатором или отладки с помощью Logic Analyzer:

www.FPGA-Systems.ru

Hardware ?	_ 🗆 🖒 X	hw_ila_1 ? 🗆 🖾	×
Q ≚ ≑ ∅ ▶ ≫ ■	٠	Waveform - hw_ila_1 ? _ U ×	
Name > ii localhost (1) > iiinx_tct/Digilent/2103197897 > iiinx_tc	Status Connected Open Programmed	Status: Idle 4,096 ILA Status: Idle 4,096 Name Value > M dna_reader_fsm_0_odna[63:0] 0 Ik statu_0_1 10,000	
Debug Probe Properties ? Source: NETLIST Type: ILA Probe type: Data and Trigger	× ×		r
Width: 1 Display Name Long name: inst_dna_read Short name: istart_0_1 Custom name:	der/dna_reader_t	Settings - hw_ila Status - hw_ilx ? Image: Settings - hw_ila ? <td< th=""><th></th></td<>	

Рисунок49 – Представление Vivado в режиме логического анализатора

Нам необходимо получить данные с цепей при определённом условии: когда сигнал запуска автомата istart станет равным «1». Это называется триггером, то есть условием, по которому будет начата запись. Пока условие не выполнено, запись не начнётся и логический анализатор будет находится в режиме ожидания выполнения соответствующего условия.

Для добавления условия срабатывания (т.е. триггера условия), нажмите на крестик в окне Trigger Setup, и выберите сигнал istart, затем нажмите ОК.

Add Probes	Setup - hw_ila_1
Search: Q-	1
<pre>be dna_reader_fsm_0_odna[63:0] be dna_reader_fsm_0_odna_ready be istart_0_1 </pre>	the 🕂 button to add probes.

Рисунок50 – Список доступных цепей, выбор триггера

После этого istart появится в окне триггеров. 33

_ □
2[0]
E

Рисунок51 – Добавленная в определение триггера цепь

Теперь необходимо настроить условие для этого триггера. Сейчас поле Value имеет значение «х». Это означает, что запись состояний наблюдаемых цепей будет начинаться вне зависимости от состояния сигнала istart. Измените значение поля Value сигнала istart на «1».



Рисунок52 – Настройка триггера (условия срабатывания)

У логического анализатора много настроек и много возможностей. В качестве триггеров можно использовать уровни сигналов и их фронты различной полярности, задавать срабатывание по цепочкам следующих друг за другом условий или логическим выражениям с ними. Есть даже возможность задания конечного автомата условий, имеющего свой собственный синтаксис и описание. Но это «задание со звёздочкой».

Здесь же мы рассмотрим ещё одну полезную функцию. Предположим, мы хотим посмотреть сигнал не только после срабатывания условия, но и за несколько отсчётов до этого момента – т. е. состояние линии до срабатывания условия. Сделать это можно, указав соответствующее количество отсчётов до момента срабатывания в поле Trigger positioning window. Сейчас это значение установлено в 128, что означает, что нам будет доступно состояние цепи за 128 отсчётов до

срабатывания условия плюс оставшиеся отсчёты после срабатывания. Никакой магии «обращения времени назад» при этом нет: логический анализатор просто с самого момента запуска схемы постоянно пишет значения сигнала в кольцевой буфер, а по срабатыванию триггера – через фиксированное время останавливает запись и выдаёт пользователю результат.

Settings - hw_ila_1 × Status -	hw_ila_1	?	_ □	Trigger Setu	ıp - hw_ila_
Trigger Mode Settings				Q +	- D
Trigger mode: BASIC ON				Name	Operator
Ingger mode. DASIC_ON				istart_0_1	==
Capture Mode Settings			_		
Capture mode:	ALWAYS	\sim			
Number of windows:	1	[1 - 4096]			
Window data depth:	4096	✓ [1 - 4096]			
Trigger position in window:	128	[0 - 4095]			
General Settings					

Рисунок53 – Окно настроек логического анализатора

Для запуска логического анализатора нажмите на копку запуска, показанную на рисунке 54.

		3	? _ 🗆 ×
🕒 🔍 G		•F 4 • 12 2r +F [@ +F]+	•
		4,096	
Value	0	L	°
< >	<		
	Value		Image: Second

Рисунок54 – Кнопка запуска логического анализатора

Логический анализатор окажется в режиме ожидания до тех пор, пока istart не перейдёт в «1».

Мы подключили istart к кнопке btn0, установленной на Arty. Нажмите её. Если всё до этого момента было выполнено корректно, на экране должна появится временная диаграмма.



Рисунок55 – Считанные значения цепей после срабатывания триггера

Значение odna[63:0] на этой временной диаграмме, естественно, у Вас будет другим, поскольку DNA уникальна для каждого кристалла (с учётом ограничений, о которых мы говорили).

Как мы можем быть уверены, что это значение DNA является корректным, и мы считали его правильно? Если помните, DNA можно также прочитать, используя JTAG.

На самом деле все уже было прочитано Vivado в тот момент, когда мы выполняли поиск устройств по нажатию кнопки Autoconnect. Значение DNA может быть найдено следующим образом (см. рисунок):

- 1. Выберите подключённое устройство.
- 2. Перейдите во вкладку Properties.
- 3. Найдите поле Registers.
- 4. Затем EFUSE.
- 5. Потом DNA_PORT.

www.FPGA-Systems.ru



Рисунок56 – Значение DNA, считанное через JTAG

А теперь сравните значение поля DNA_PORT с тем, что мы получили в логическом анализаторе. Если совпало, то ПОЗДРАВЛЯЮ ВАС! На этом можно закончить. Если же что-то не получилось – ещё раз внимательно всё проверьте или повторите. Если и это не помогает – оставьте комментарий или задайте вопрос на <u>www.fpga-systems.ru</u>. Мы обязательно Вам поможем.

Заключение

Каждая FPGA уникальна. Уникальны как её номер, так и её свойства. Нужно ли Вам использовать DNA в своих проектах, решать Вам. Но, не сомневайтесь, этой статьи не было бы, если бы DNA никто не использовал. Кто-то применяет её для защиты проектов, кто-то – для контроля версий... А для чего DNA FPGA может пригодиться Вам? Напишите об этом в комментариях к данной статье на <u>www.fpga-systems.ru</u>

Не забудьте сделать и домашнее задание. Удачи!

Домашнее задание

- 1. Выполните реализацию других схем включения DNA_PORT.
- Настройте триггер логического анализатора на сигнал odna_ready за 1234 отсчёта до его срабатывания. (Учтите, что для перезапуска потребуется заново прошить FPGA, поскольку автомат управления не имеет условия возврата к начальному состоянию, из которого происходит его запуск по istart).
- 3. * (повышенной сложности) Получить DNA можно и через JTAG. Попробуйте получить DNA с использованием Tcl-команд.
- 4. ** (высокой сложности) Считайте все значения EFUSE-регистров с помощью Tcl-скрипта и сформируйте результаты в удобочитаемый отчёт.

Библиографический список

- 1.<u>UG470</u> 7 Series FPGAs Configuration.
- <u>UG953</u> Vivado Design Suite7 Series FPGA and Zynq-7000 All Programmable SoC Libraries Guide.
- 3. <u>DS181</u> Artix-7 FPGAs Data Sheet: DC and AC Switching Characteristics.
- 4. <u>UG835</u> Vivado Design Suite Tcl Command Reference Guide.
- 5. <u>UG894</u> Using Tcl Scripting.
- 6. <u>Vivado на сайте Xilinx.</u>

7. <u>Описание</u> Arty Board на сайте Digilent.

8.<u>UG908</u> Programming and Debugging.

Список тренингов

Список тренингов по проектированию на FPGA в сертифицированном тренинг-центре компании Xilinx:

- 1. <u>Проектирование на FPGA в Vivado Design Suite #1</u>
- 2. <u>Проектирование на FPGA в Vivado Design Suite #2</u>
- 3. <u>Проектирование на FPGA в Vivado Design Suite #3</u>
- 4. <u>Проектирование на FPGA в Vivado Design Suite #4</u>
- 5. <u>Проектирование на FPGA 7 серии</u>
- 6. Полный перечень курсов

Скачать файлы проекта