

Создание пользовательского IP-ядра в Qsys/Platform Designer

Автор: andrewkushchenko

Рецензенты: Intekus

KeisN13







Оглавление

Аннотация	3
Введение	
Создание ІР-ядра	4
Заключение	14



Аннотация

PayPal

В статье рассмотрен способ создания пользовательских IP-ядер в Intel Qsys/Intel Platform Designer. И Intel Qsys, и Intel Platform Designer являются компонентами среды Intel Quartus. Разница между Qsys и Platform Designer невелика: во втором незначительно расширен функционал. Qsys используется в более старых версиях Quartus, более новые – поставляются с Platform Designer. У меня на машине установлен Quartus Prime Lite 17.1, который идет уже с Platform Designer, так что именно он и будет использован для примера.

Введение

Концепция добавления пользовательских IP-ядер в Platform Designer проста: описать все интерфейсы модуля. Это необходимо для правильного подключения различных модулей друг к другу. Platform Designer поддерживает достаточно широкий спектр интерфейсов, которые можно видеть на рисунке 1.



Рисунок 1 – Список поддерживаемых интерфейсов

Для примера возьмем простой модуль, написанный на VHDL, имеющий 3 интерфейса: Avalon-STin, Avalon-STout и Avalon-MMcsr. Модуль должен принимать поток данных по интерфейсу Avalon-STin, выполнять над данными некоторые преобразования (это сейчас для нас не важно) и выдавать результат по интерфейсу Avalon-STout. A Avalon-MMcsr будет при этом задавать параметры работы модуля.

В листинге представлен код модуля

```
<sup>Яндекс</sup>
Деньги ڬ
```

```
library ieee;
use ieee.std_logic_1164.all;
entity stream_converter is
    generic (
       CONST_A : integer := 16#00#;
        CONST B : integer := 16#F5#
    );
    port (
       clk
              : in std logic;
       reset : in std_logic;
        stream in data : in std logic vector(31 downto 0);
        stream_in_sop : in std_logic;
        stream_in_eop : in std_logic;
        stream_in_valid : in std_logic;
        stream_in_ready : out std_logic;
        stream_out_data : out std_logic_vector(31 downto 0);
                         : out std logic;
        stream out sop
                         : out std_logic;
        stream_out_eop
        stream_out_valid : out std_logic;
        stream_out_ready : in std_logic;
                       : in std_logic_vector(5 downto 0);
       cp_address
       cp_write
                        : in std_logic;
                       : in std_logic_vector(31 downto 0);
        cp_writedata
                        : in std_logic;
       cp_read
        cp_readdata
                        : out std_logic_vector(31 downto 0);
        cp_waitrequest : out std_logic
    );
end entity;
architecture rtl of stream_converter is
begin
end architecture;
```

Создание ІР-ядра

После того, как модуль создан, можно создавать IP-ядро в Platform Designer. Для этого откроем Platform Designer, и на панели IP Catalog нажмем кнопку New.



) A 🕢

You Tube



Platform Designer - unsaved.qsys* (C:\ File Edit System Generate View Tools	intelFPG Help	A_lite\	17.1\unsav	red.qsys)				-	• ×
P Catalog 2 _ C _ C		System	Contents	X Address Map X	Interconnect Requirements 🛛				- ರೆ ರ
🔍 🗙 💱		X 4	Sy:	stem: unsaved					
Project	11 1	Use	Conn	Name	Description	Export	Clock	Base	
New Component				⊡ clk_0	Clock Source				
Library Basic Functions	×		P-	dk_in	Clock Input	clk	exported		
+-DSP			P-	ck_in_reset	Reset Input	reset			
Interface Protocols			×	clk div sout	Clock Output	Double-click to export	dk_0		
Low Power			~~~~	ck_reset	Reset Output	Double-click to export			
Memory Interfaces and Controllers									
Processors and Peripherals Osys Interconnect	_								
Tri-State Components	-								
University Program									
	T								
New Edit + Add									
	-								
👯 Hieri 🛛 Device F 🖾 📃 📑 🗖	-								
unsaved [unsaved.gsys*]	1								
🗉 🖿 dk									
🖲 🖿 reset									
⊞-∎∎ dk_0									
		<							>
		rh~ 41	t · 🌱 🛒	Current filter:					
	S E	Messag	ies 🛛						- d =
	TVD	e	Path	Message					
	<								>
0 Errore 0 Warninge								Generate HDI	Einich

PayPal

Рисунок 2 – Главное окно Platform Designer

После нажатия кнопки мы увидим окно создания нового ІР-ядра

L Component	Editor - new <u>B</u> eta <u>V</u> iew	/_compone	ent_hw.tcl*						
Component Type	e 🖾 Blo	ck Symbol	8 Files	83	Parameters	83	Signals & Interfaces	X	- 🗗
 About Comp 	onent Type								
Name:	new_compor	ient							
Display name:	new_compor	ient							
Version:	1.0								
Group:									 ~
Description:									
Created by:									
Icon:									
Documentation:	Title		URL						
	+ -								
Messages 🐰	1								_
To Do: Add H	DL files on the	e Files tab	or add signa	ls on t	he Signals tab				
10 00. Add 1	DE TRES OFF CH	er nes taby t	or daa signe	13 011 0	ne orginalo tabi				
			<u>H</u> elp		Prev	<u>N</u> ext	▶ <u>F</u> inish		

Рисунок 3 – Окно создания нового IP-ядра

На самой первой вкладке необходимо заполнить информацию об IP-ядре: имя, версию, группу в IP-каталоге и т. д. Мы зададим только поля Name и Display name. В большинстве случаев

W	You	0	^{Яндекс} Деньги 🗳	PayPal	www.FPGA-Systems.ru

этого будет достаточно. В оба поля напишем название модуля «Stream_converter». Имя может быть любым и не обязательно соответствовать названию модуля.

Далее перейдем на вкладку Files и добавим исходные файлы к IP-ядру.

👗 C File T	omponent Edit emplates Beta	or - stream_c	onverter_hw.tcl*							×
Com	ponent Type	Block Sym	ibol 🛛 Files	8 Param	neters 🛛	Signals & Interfaces	8		_	
► A	About Files									
Synt	thesis Files									
Thes	e files describe t	his component's	s implementation, a	nd will be cre	ated when a	Quartus synthesis mo	del is gen	erated.		
The p	parameters and s	ignals found in	the top-level modu	le will be use	d for this co	mponent's parameters	and signal	s.		
	Output Path		Source File				Туре	2	Attributes	
•	test_compon	ent.vhd	D:/work/Custo	mQSYSComp	onent/test_	component.vhd	VHD	L	Top-level Fi	ile
×										
	Add File	Remove File	Analyze Synthes	sis Files C	reate Synth	esis File from Signals				
Too-	evel Module:	naluza filos ta	coloct modulo)							
		naryze nies to	select module) V							
Veri	log Simulation	Files	/erilog simulation m	odel is gener	ated				63	
	Output Path		Source File	ouch bigenen	Тур	e		Attributes		
×										
	(No files)									
Ŧ										
	Add File	Remove File	Copy from Synth	nesis Files						
Mess	sages 🛛								_	
🖵 Т	o Do: The top-lev	vel module doe	s not contain any si	gnals.						
			Help		Next	▶ <u>F</u> inish				

Рисунок 4 – Добавление исходных файлов к ІР-ядру

Можно добавить все файлы, который входят в состав модуля, но я предпочитаю добавлять только обертку верхнего уровня модуля, т. к. все добавленные файлы будут копироваться в каталог сгенерированных файлов, что не всегда удобно.

После добавления всех необходимых файлов жмем кнопку Analyze Synthesis Files и ждем, пока Platform Designer проверит синтаксис добавленных файлов.

Теперь на вкладке Block Symbol мы можем уже увидеть все интерфейсы, которые Platform Designer смог определить автоматически.

www.FPGA-Systems.ru



PayPa

еньги

Рисунок 5 – Block Symbol

Как видим, Platform Designer корректно определил интерфейс Avalon-MM (ControlPort), а вот оба stream интерфейса определились неправильно. Теперь на вкладке Signals & Interfaces необходимо подправить некорректно определённые интерфейсы.





PayPal

деньги 🕑

You Tube

Рисунок 6 – Signals & Interfaces

Тут мы видим, что Platform Designer распознал 2 stream интерфейса, как 1 memory-mapped. Для исправления ситуации создадим 2 stream интерфейса вручную, нажав на кнопку <<addinterface>>, и перетянем туда соответствующие сигналы. Т.к. у нас один Avalon-ST интерфейс приемный, а второй передающий, то необходимо создать один интерфейс Avalon Streaming Sink, а второй Avalon Streaming Source. Пустой автосгенерированный интерфейс можно удалить.





PayPal

деньги 🕑

You Tube Рисунок 7 – Создание новых интерфейсов

Как видим, теперь у нас и правда 2 Avalon-ST интерфейса, но все сигналы помечены красным, и в логе будет множество ошибок, т. к. сигналы имеют неправильное назначение. В каждом интерфейсе необходимо задать правильные функции для всех сигналов, чтобы в Platform Designer корректно подключать между собой разные модули.





PayPal

деньги 🖆

You Tube f

Рисунок 8 – Назначение функций сигналам

После назначения функций осталась ещё одна ошибка и пара предупреждений.





PavPa

еньги

Рисунок 10 – Ошибки назначения сброса

Необходимо для каждого интерфейса типа Avalon-MM и Avalon-ST назначить сигнал сброса. Обязательным ещё является тактовый сигнал, но он определился автоматически. Задавать тактовый сигнал и сигнал сброса для интерфейсов необходимо для того, чтобы Platform Designer мог корректно подключать интерфейсы из разных клоковыми доменов. Например, наше IP-ядро может работать на системной частоте 100 МГц, но интерфейс программирования может быть подключен к PCIe контроллеру, который работает на частоте 125 МГц (его интерфейсная часть со стороны FPGA). В случае с доступом к регистрам модуля мы можем просто соединить наше IP-ядро и PCIe-контроллер, не думая о CDC (Clock Domain Crossing). Platform Designer автоматически вставит переход между частотами. Но это не очень правильный путь, т.к. подобные преобразования потребляют дополнительные ресурсы, и в большом проекте Platform Designer может автоматически поставить десяток переходников между клоковыми доменами и, например, преобразователей разрядности шины. Но оптимизация системы в Platform Designer – это отдельная общирная тема, её мы рассмотрим в другой раз.



После последних исправлений у нас больше нет никаких предупреждений, и остается только нажать кнопку Finish для завершения создания IP-ядра.



Рисунок 11-Детальная информация интерфейса Avalon-MM

На этом этапе дополнительно можно настроить параметры интерфейсов. В мануале на шину Avalon описано множество её параметров. Например, адресация словная или байтовая, задержка чтения и т. д. В большинстве случаев стандартных настроек достаточно, так что создание IP-ядра можно завершать.

Теперь в окне IP catalog у нас появился новое IP-ядро.

B	You Tube	Ø		^{Яндекс} ДЕНЬГИ 🗳	PayPal ⁻		www.FPGA-Systems.ru
				IP Catalog Project Project Stream Library Sasic Funce Solution Processors Processors Processors Procesors Processors Procesors Processors P	Component Component converter tions Protocols terfaces and Co and Peripheral connect components Program	- C C	

Рисунок 12 – Новый компонент в окне IP-catalog

Теперь его можно использовать в своей системе. Добавим в систему наше IP-ядро и ещё несколько библиотечных для демонстрации. Поставим JTAG to Avalon Master Bridge, чтобы иметь доступ к регистрам нашего модуля через JTAG и добавим Clocked Video Input/Output для имитации приема и передачи видео. Как видно, интерфейсы нашего модуля легко подключаются к интерфейсам IP-ядер от самого Intel. Таким образом, можно создать библиотеку из необходимого набора пользовательских IP-ядер и достаточно быстро собирать большие системы.

IP Catalog 🛛 🗕 🗖	• 🗖 🏗	System	Contents 🛛 Address	Map 🛛 Interconnect Rev Path: alt_vip_cti_0.clocked	juirements 🙁				- 5
Numiest	× +	Use	Connections	Name	Description	Export	Clock	Base	End
roject	^ 🙀				Clock Source	Capore	Circle	Cosc	Lind
 stream converter 				dk in	Clock Josuit	clle	ownexted		
ibrary				dkjiri dkjiri streti	Coock Input	CIR	exported		
Basic Functions				ok_in_reset	Reset input	reset			
Arithmetic				OK .	Clock Output	Double-click to export	ak_u		
Bridges and Adaptors				dk_reset	Reset Output	Double-click to export			
Clock		\checkmark		□ 0 master_0	JTAG to Avalon Master Bridge				
Interrupt			(†	dk	Clock Input	Double-click to export	clk_0		
Memory Mapped	T		$ \bullet \rightarrow \rightarrow$	ck_reset	Reset Input	Double-click to export			
Reset				master	Avalon Memory Mapped Master	Double-click to export	[clk]		
I Streaming				master_reset	Reset Output	Double-click to export			
Elocks; PLLs and Resets				□ stream converter 0	stream converter				
E-Configuration and Programming			↓	dock	Clock Input	Double-click to export	clk 0		
DMA				reset	Reset Input	Double-click to export	[clock]		
On Chip Memory				~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~	Avalon Memory Manned Slave	Double, click to export	[clock]		0.000
 Simulation; Debug and Verification 	~ I			φ audea abaanina aiala	Avalor Menory Mapped Slave	Double-click to export	[clock]		0x00
	- 1			avaion_streaming_sink	Avaion Streaming Sink	Double-click to export	[CIOCK]		
New Edit + Add.				avaion_streaming_so	Avaion Streaming Source	Double-click to export	[CIOCK]		
		\leq		alt_vip_cti_0	Clocked Video Input				
	=1			is_dk_rst	Clock Input	Double-click to export	clk_0		
, Hierarch; 🛛 Device Family 🖾 🗕 🖻				is_dk_rst_reset	Reset Input	Double-click to export	[is_clk_rst]		
unsaved [unsaved gavs*]				dout	Avalon Streaming Source	Double-click to export	[is_clk_rst]		
alt vio cti 0 clocked video			P=	clocked_video	Conduit	alt_vip_cti_0_clocked_v.			
alt_vip_itc_0_clocked_video				alt vip itc 0	Clocked Video Output				
► dk			$ \bullet \rightarrow \rangle$	is dk rst	Clock Input	Double-click to export	clk 0		
reset				is dk rst reset	Reset Input	Double-click to export	lis dk rsti		
a∏t alt vip cti 0				dia.	Auslee Streaming Siels	Double click to export	Do elle soll		
docked video				un de de al colora	Avalori Streaming Sink	Double-click to export	[IS_CIK_ISU]		
dout		<		Concent Vineo	Conduit	air vin ire ii clocked v			
		al de							
🐵 🖿 is_dk_rst_reset		14 M	🗧 🔻 💘 Current filte	r:					
alt_vip_itc_0	X	Message	es 🛙						
ie-∎ din									
is_dk_rst	19	pe	Path			Message			
is_dk_rst_reset		7	1 Warning						
₽.ª.	4	▲	unsaved.alt_vip_cti	_0	1	The vid_datavalid signal indicates acti	ve picture.		
u master_0									
stream_converter_0									
Connections									

Рисунок 13 – Собранная тестовая система



Заключение

PavP

Аналогичным образом можно использовать шину AXI, а если новый модуль имеет какие-то нестандартные интерфейсы, или просто те, которые необходимо экспортировать на физические пины микросхемы, то можно выбрать в качестве типа интерфейса conduit, который не регламентирует набор сигналов в интерфейсе.

В примере выше мы создавали IP-ядро с помощью графической оболочки, и в результате у нас сформировался файл stream_converter_hw.tcl, который описывает все наши настройки в текстовом виде. Для того, чтобы в новом проекте было доступно уже созданное IP-ядро, необходимо в настройках (Tools ->Options) добавить путь к файлу stream_converter_hw.tcl.

Как вы уже поняли, мы могли бы вместо создания IP ядра в графической среде все сделать в текстовом редакторе, т. к. файл stream_converter_hw.tcl является просто скриптом на языке TCL. Но я предпочитаю создавать IP-ядра в графической среде по нескольким причинам. Во-первых, не надо помнить свойства всех интерфейсов (т. к. в реальной работе создавать IP-ядра для Platform Designer приходится нечасто, то все свойства интерфейсов быстро забываются). А во-вторых, PlatformDesigner в реальном времени указывает нам на ошибки.

Понравилась статья? Не забудьте поддержать автора

PayPal[®]