Qsys and IP Core Integration

Prof. David Lariviere

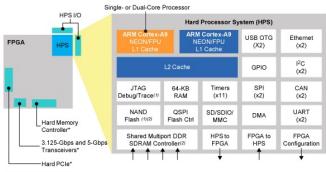
Columbia University

Spring 2014

Overview

What are IP Cores? Altera Design Tools for using and integrating IP Cores Overview of various IP Core Interconnect Fabrics

Cyclone V SoC - Mix of Hard and Soft IP Cores



*Optional Configuration

IP Core

"In electronic design a semiconductor intellectual property core, IP core, or IP block is a reusable unit of logic, cell, or chip layout design that is the intellectual property of one party... The term is derived from the licensing of the patent and/or source code copyright that exist in the design. IP cores can be used as building blocks within ASIC chip designs or FPGA logic designs."

Source: Wikipedia: Semiconductor Intellectual Property Core

IP Core Types - Hard vs. Soft

"Hard" IP: Dedicated ASIC blocks within the FPGA targeting specific functionality

"Soft" IP: Implemented using the general purpose configurable fabric of the FPGA

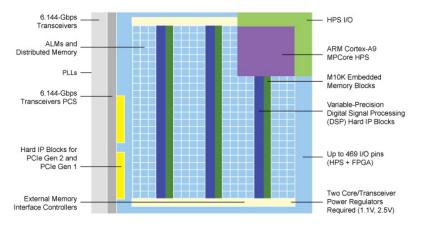
CPUs: either hard (ARM SoC) or soft-core (Nios II)

Highspeed I/O: Hard IP Blocks for High Speed Transceivers (PCI Express, 10Gb Ethernet, etc)

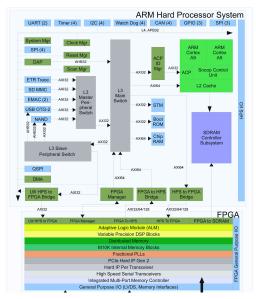
Memory Controllers: (onboard SRAMs, external DDR3, etc)

Clock and Reset signal generation: (PLLs)

Cyclone V SoC - FPGA layout

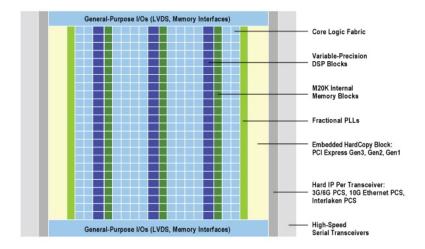


Cyclone V SoC - HPS Layout

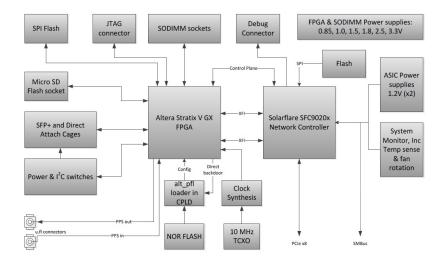


Source: NARD, LLC.

Stratix V - FPGA Layout

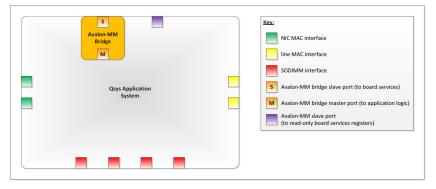


Stratix V - Solarflare AoE PCB Layout



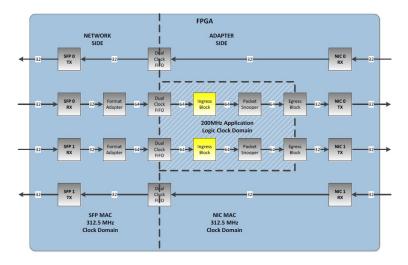
Source: Solarflare FDK

Stratix V - Solarflare AoE Qsys Layout



Source: Solarflare FDK

Stratix V - Solarflare AoE Qsys Example



Source: Solarflare FDK

Bridges

Purpose: modify how data is transported between components.

Allows for: different clock domains, protocol conversion (AXI <-> Avalon), pipelining, bus width adaptation, etc

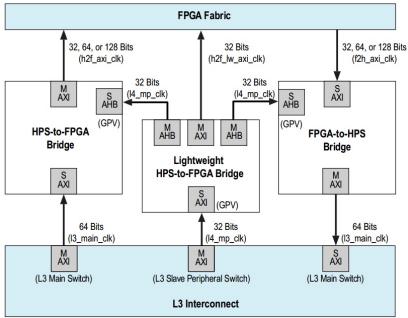
Bridge Types:

SOC HPS <-> FPGA Bridge

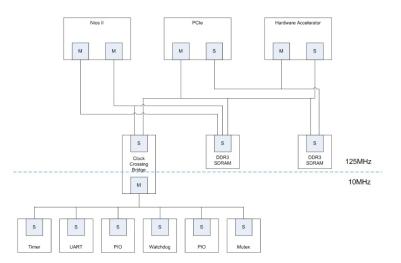
Avalon MM Clock Crossing Bridge

Avalon MM Pipeline Bridge

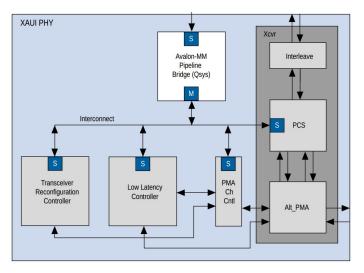
Cyclone V SoC: FPGA <-> HPS Bridge



Clock Crossing Bridge Example



Pipeline Bridge Example



Altera Tools for IP Cores

Megawizard

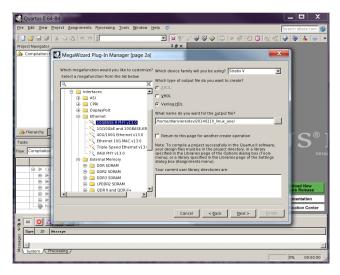
SOPC Builder: System-on-Chip (SoC); old, replaced by Qsys

Qsys: Network-on-Chip (NoC).

Megawizard

Quartus II 64-Bit	The second s	
Eile Edit View Project Assignments Processing	Iools Window Help 🦃	Search altera.com
Project Navigator	Run Sim <u>u</u> lation Tool	\$\$\$© >₹800 ≈<\$ <mark>\$}\$\$</mark> \$
A Compilation Hierarchy	DimeQuest Timing Analyzer	
	Advisors	
	 Chip Planner Design Partition Planner Netlist Viewers 	
	SignalTap II Logic Analyzer In-System Memory Content Editor Logic Analyzer Interface Editor In-System Sources and Probes Editor	
A Hierarchy 📑 Files 🥜 Design Units 🤌	SignalProbe Pins	IADTIC®
Tasks	🧼 [TAG Chain Debugger	JARTUS®
Flow: Compilation	77 Transceiver Toolkit	Vers
Task	External Memory Interface Toolkit	
Compile Design B Analysis & Synthesis Fitter (Place & Route) D Assembler (Generate programmin	MegaWizard Plug-In Manager Nios II Software Build Tools for Eclipse	O Download New
TimeQuest Timing Analysis	/ Tcl Scripts	Software Release
EDA Netlist Writer	Customize	Ocumentation
Program Device (Open Programmer)	Options	Notification Center
× All 🔞 🛆 🔥 🛷 << Search>>	License Setup	
	💱 Install Devices	
. Type ID Ressage		
essages		F
System (Processing)		
Starts MegaWizard Plug-In Manager		0% 00:00:00

Megawizard - Example 10Gb Ethernet PHY



Megawizard IP Cores

Arithmetic: Addition, Subtraction, Multiplication, Division, Multiply-(add|accumulate), ECC

Floating Point:

Gate Functions: Shift Registers, Decoders, Multiplexers

I/O Functions: PLL, temp sensor, remote update, various high speed transceiver related

Memory: (Single|Dual)-port RAM or ROMs, (Single|Dual)-clock FIFOs, (RAM) Shift registers

DSP: FFT, ECC, FIR, etc (large suite specifically for graphics as well)

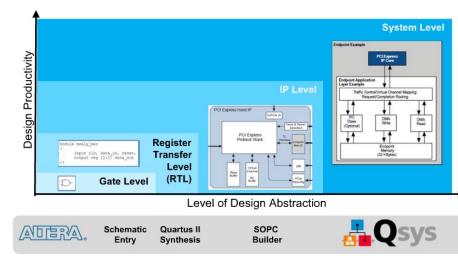
Note: availability of specific megafunctions is FPGA specific (may only be available on Stratix V but no Cyclone V SoC, etc)

Qsys is Altera's system integration tool for building Network-on-Chip (NoC) designs connecting multiple IP cores.

It replaces SOPC Builder (previous version of the tool).

You utilize Qsys to construct a system of IP components (and even system of systems), and Qsys will automatically generate the interconnect, add required adaptation, warn on misconfigurations, etc.

Qsys - Raising Level of Abstraction



Qsys UI

omponent Library	Syste	em Cont	ents Address Map	Clock Settings Project Set	tings System Inspector	HDL Example Generation	
×	+	Use	Connections	Name	Export	Description	Cloc
	×			ext_clk		Clock Source	
New component			D-		ext_clk_clk_in	Clock Input	
E System			D-	clk_in_reset	ext_clk_clk_in_reset		
my_subsystem	X			clk	Glick to export		ext_clk
ibrary		-		clk_reset	System	Reset Output	
 Clock Source 	-	~		pipeline_bridge	Contents	Avision-MM Pipeline Bridge	
Reset Bridge				clk		Clock Input	ext_clk
Component				reset	Unck to export		[clk]
Bridges and Adapters	52			s0	pipeline_bridge_s0		[clk]
Custo	1.011	V		m0	Click to export	Avalon Memory Mapped Master DDR3 SDRAM Controller with UniPHY	[clk]
Debug Components Digital Signal Processing		M		ddr3_sdram	Click to export	Conduit	
Interface Protocols				clock_sink	Click to export		ext_clk
Memories and Memory Contro				clock sink reset	Click to export		[clock si
+-DMA			×	clock source	Click to export		ddr3 sdr
				half_clock_source	Click to export		ddr3_sdr
				Other	Click to export	Conduit	0010_001
New Edit				afi cal debug	Glick.to export	Conduit	
		•					F
ssages							
		Descrip	otion			Path	
3 Errors			_				
7 Warnings			Messad	ges: System	Validation		
2 Info Messages			moood	,,,	- andanon		

System Level Design - Why use Qsys

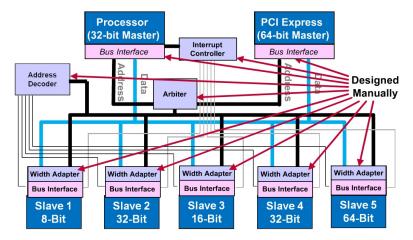
Avoids manually developing custom interconnect fabrics and signaling.

Instead of cycle-to-cycle coordination between every individual IP core, focus on 'transaction' level designs.

Design IP without knowing exactly when data will transfer and instead only focus on how (once it does).

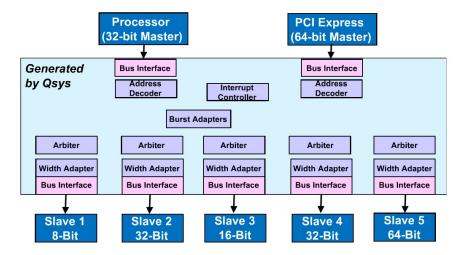
(Only valid if you design your individual components to one of the standardized interfaces)

Old (Manual) Method of Design



Large Amount of Engineering Overhead!

Qsys-based Method of Design



Interface Types

Memory-mapped Interfaces:

Avalon MM (Altera) AXI (ARM, supported by Qsys now for SoC)

Streaming Interfaces: Avalon ST:

Avalon ST source port: outputs streaming data

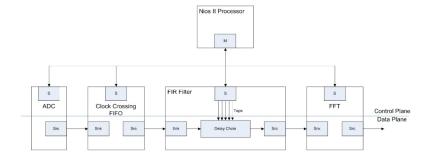
Avalon ST sink port: receives incoming streaming data

Control Plane: Memory mapped registers typically used for configuring devices, querying status, initiating transactions, etc (low bandwidth)

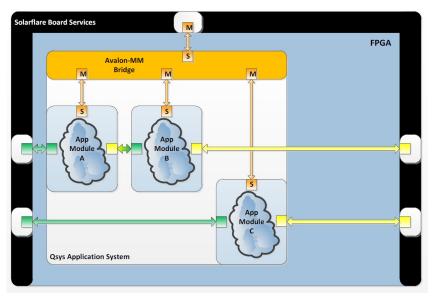
Data Plane: Streaming directed graphs for actually moving and processing large amounts of data (audio/video, network packets, etc); high bandwidth

A single IP core can have both MM and ST interfaces (including multiple of each).

Control and Data Planes Example



Control and Data Planes Example - Solarflare AoE



Qsys signal types

Clock

Reset

Interrupt

Avalon MM signals

Avalon ST signals

Tristate

Conduit (avoid unless truly not one of the above)

Why explicitly label signal types?

...vs. simply making everything a wire/conduit

Allows Qsys to protect you from yourself!

Ensure matching between signal types ("clock out" -> "clock in", etc)

Detect and automatically insert dual clock crossing domains (only if it knows which clock domains IPs are in)

Automatically convert data widths, formats, error flags (convert 32 bit master into four 8-bit slave reads, etc)

Automatically synchronize and OR-gate multiple resets

Automatically insert pipeline stages to improve fmax

Avalon MM Master Signals

Signal type	Width	Direction	Required	Description
address	1-64	Output	Y	Byte address corresponding to slave for transfer request (discussed later)
waitrequest waitrequest_n	1	Input	Y	Forces master to stall transfer until deasserted; other Avalon-MM interface signals must be held constant
read read_n	1	Output	N	Indicates master issuing read request
readdata	8, 16, 32, 64, 128, 256, 512, 1024	Input	N	Data returned from read request
write write_n	1	Output	N	Indicates master issuing write request
writedata	8, 16, 32, 64, 128, 256, 512, 1024	Output	N	Data to be sent for write request
byteenable byteenable_n	2, 4, 8, 16, 32, 64, 128	Output	N	Specifies valid byte lane(s) for readdata or writedata (width = data width / 8)
lock lock_n	1	Output	N	Once master is granted access to shared slave, locks arbiter to master until deasserted

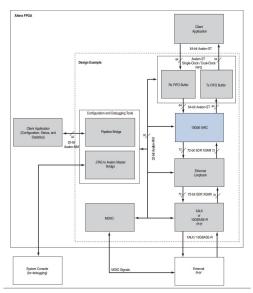
Avalon MM Slave Signals

Signal type	Width	Direction	Required	Description
address	1-64	Input	N	Word address of slave for transfer request (discussed later)
waitrequest waitrequest_n	1	Output	N	Allows slave to stall transfer until deasserted (other Avalon-MM interface signals must be held constant)
read read_n	1	Input	N	Indicates slave should respond to read request
readdata	8, 16, 32, 64, 128, 256, 512, 1024	Output	N	Data provided to Qsys interconnect in response to read request
write write_n	1	Input	N	Indicates slave should respond to write request
writedata	8, 16, 32, 64, 128, 256, 512, 1024	Input	N	Data from the Qsys interconnect for a write request
byteenable byteenable_n	2, 4, 8, 16, 32, 64, 128	Input	N	Specifies valid byte lane for readdata or writedata (width = data width / 8)
begintransfer begintransfer_n	1	Input	N	Asserts at the beginning (first cycle) of any transfer

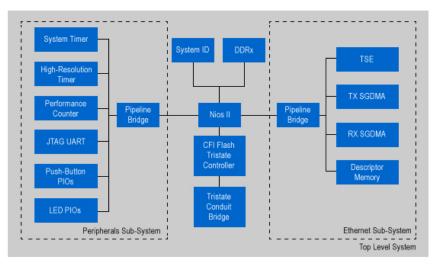
Avalon ST Signals

Signal type	Width	Direction	Description	
Fundamental signals				
ready	1	$\text{Sink} \to \text{Source}$	Indicates the sink can accept data	
valid	1	$\textbf{Source} \rightarrow \textbf{Sink}$	Qualifies all source to sink signals	
data	1-4096	$\textbf{Source} \rightarrow \textbf{Sink}$	Payload of the information being transmitted	
channel	1-128	Source \rightarrow Sink	Channel number for data being transferred (if multiple channels supported)	
error	1-255	Source \rightarrow Sink	Bit mask marks errors affecting the data being transferred	
Packet transfer signals				
startofpacket	1	$\textbf{Source} \rightarrow \textbf{Sink}$	Marks the beginning of the packet	
endofpacket	1	$\textbf{Source} \rightarrow \textbf{Sink}$	Marks the end of the packet	
empty	1-8	Source \rightarrow Sink	Indicates the number of symbols that are empty during cycles that contain the end of a packet	

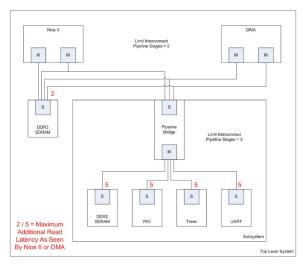
Example Qsys Layout - 10Gb Reference Design



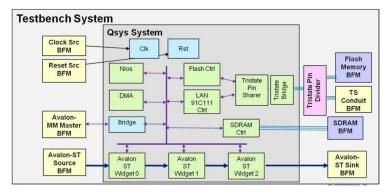
Advanced - Qsys Hierarchical Designs



Advanced - Qsys Automatic Pipelining



Advanced - Qsys Testbench Generation



Altera online training lectures: (HIGHLY recommended; many of these slides are taken directly from them)

http://www.altera.com/education/training/curriculum/trncurriculum.html

Introduction to Qsys

Advanced System Design Using Qsys

Custom IP Development Using Avalon and AXI Interfaces