Multiplier v12.0

LogiCORE IP Product Guide

Vivado Design Suite

PG108 November 18, 2015

Table of Contents

IP Facts

Chapter 1: Overview

Chapter 2: Product Specification

Chapter 3: Designing with the Core

Chapter 4: Design Flow Steps

Appendix A: Migrating and Upgrading

Appendix B: Debugging

Appendix C: Additional Resources and Legal Notices

IP Facts

Introduction

The Xilinx® LogiCORE™ IP Multiplier implements high-performance, optimized multipliers. Resource and performance trade-off options are available to tailor the core to a particular application.

Features

- Generates fixed-point parallel multipliers and constant-coefficient multipliers for two's complement signed or unsigned data
- Supports inputs ranging from 1 to 64 bits wide and outputs ranging from 1 to 128 bits wide with any portion of the full product selectable
- Configurable latency for all multiplier variants
- Supports symmetric rounding to infinity when using the DSP Slice.

LogiCORE IP Facts Table	
Core Specifics	
Supported Device Family[1]	UltraScale+™ Families UltraScale™ Architecture Zynq®-7000 All Programmable SoC 7 Series
Supported User Interfaces	N/A
Resources	Performance and Resource Utilization web page
Provided with Core	
Design Files	Encrypted RTL
Example Design	Not Provided
Test Bench	Not Provided
Constraints File	Not Provided
Simulation Model	Encrypted VHDL
Supported S/W Driver	N/A
Tested Design Flows[2]	
Design Entry	Vivado® Design Suite System Generator for DSP
Simulation	For supported simulators, see the Xilinx Design Tools: Release Notes Guide.
Synthesis	Vivado Synthesis
Support	
Provided by Xilinx Xilinx Support web page	

Notes:

1. For a complete listing of supported devices, see the Vivado IP catalog.
2. For the supported versions of the tools, see the Xilinx Design Tools: Release Notes Guide.

Overview

The Multiplier core can be configured in either of the following architectures:

- **Parallel**: The multiplier accepts inputs on buses A and B and generates the product of these two values. Various implementations are offered to allow a trade-off between slice logic, dedicated multiplier resources and maximum achievable clock frequency.
- **Constant-Coefficient**: The multiplier accepts data on the A input bus and multiplies it by a user-defined constant value. The multiplier can be constructed from distributed memory, block memories in conjunction with slice logic, or from embedded multipliers.

Feature Summary

The Multiplier core allows the designer fine control of the resources used to construct a fixed-point multiplier. DSP Slices, slice logic, or a combination can be used, and the structure optimized for performance or resources. Similarly, constant-coefficient multipliers can be implemented using a number of different logic resource options.

The number of pipeline stages can be selected by the designer to suit the latency and performance requirements.

The symmetric rounding feature of the DSP Slice is available for use in parallel multipliers.

Applications

The Multiplier core can be used in all applications where integer or fixed-point multiplication is required. It can also be used as a building block in construction of an efficient floating-point multiplier.

Licensing and Ordering Information

This Xilinx® LogiCORE™ IP module is provided at no additional cost with the Xilinx Vivado® Design Suite under the terms of the Xilinx End User License. Information about this and other Xilinx LogiCORE IP modules is available at the Xilinx Intellectual Property page. For information about pricing and availability of other Xilinx LogiCORE IP modules and tools, contact your local Xilinx sales representative.

Product Specification

This chapter provides access to the resource utilization and port descriptions for this core.

Resource Utilization

For full details about performance and resource utilization, visit the Performance and Resource Utilization web page.

Port Descriptions

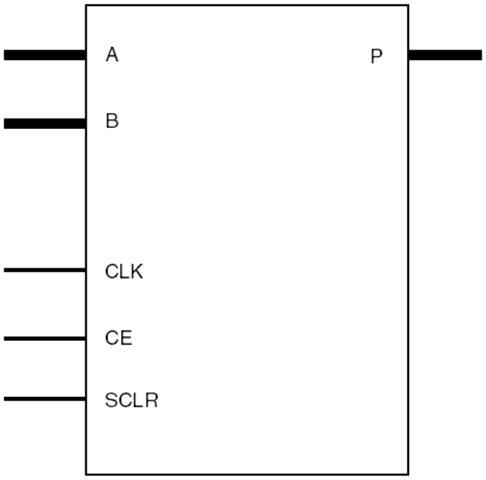

Figure 2-1: **Core Schematic Symbol**

Figure 2-1 and Table 2-1 illustrate and define the schematic symbol signal names. All control inputs are active-High.

Table 2-1: **Core Signal Pinout**

Signal	Direction	Description
A[N-1:0]	Input	A operand input bus, N bits wide
B[M-1:0]	Input	B operand input bus, M bits wide (parallel multipliers only)
CLK	Input	Rising-edge clock input

Table 2-1: **Core Signal Pinout *(Cont'd)***

Signal	Direction	Description
CE	Input	Active-High Clock Enable
SCLR	Input	Active-High Synchronous Clear (SCLR/CE priority is configurable)
P[X:Y]	Output	Product Output – bit X down to bit Y

Designing with the Core

This chapter includes guidelines and additional information to facilitate designing with the core.

Clocking

The core has a single clock pin, CLK, which is rising-edge triggered. If selected, the active-High clock enable pin, CE, stalls all core processing when de-asserted.

Resets

The core has a single, active-High synchronous reset, SCLR. Asserting SCLR for a single cycle resets all registers in the core. The priority of SCLR and CE pins can be selected in the IP catalog for the core in the Vivado Integrated Design Environment (IDE).

Design Flow Steps

This chapter describes customizing and generating the core, constraining the core, and the simulation, synthesis and implementation steps that are specific to this IP core. More detailed information about the standard Vivado® design flows and the IP integrator can be found in the following Vivado Design Suite user guides:

- *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [Ref 1]
- *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 2]
- *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 3]
- *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 4]

Customizing and Generating the Core

This section includes information about using Xilinx tools to customize and generate the core in the Vivado® Design Suite.

If you are customizing and generating the core in the Vivado IP integrator, see the *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [Ref 1] for detailed information. IP integrator might auto-compute certain configuration values when validating or generating the design. To check whether the values do change, see the description of the parameter in this chapter. To view the parameter value you can run the `validate_bd_design` command in the Tcl console.

You can customize the IP for use in your design by specifying values for the various parameters associated with the IP core using the following steps:

1. Select the IP from the IP catalog.
2. Double-click the selected IP or select the Customize IP command from the toolbar or right-click menu.

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 2] and the *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 3].

Core Parameters

The Multiplier core has several pages with fields to set parameter values for the particular instantiation required. This section provides a description of each Vivado IDE field.

- **Component Name**: The name of the core component to be instantiated. The name must begin with a letter and be composed of the following characters: a to z, 0 to 9, and "_".
- **Multiplier Type**: Select between parallel and constant-coefficient multiplier options.
- **Input Options**: Select the required operand widths and whether the operands represent two's complement signed or unsigned data.
  - **A Data Type**: Selects signed or unsigned input for operand A. In IP Integrator, this parameter is auto-updated.
  - **A Width**: Determines the width, in bits, of the A operand. In IP Integrator, this parameter is auto-updated.
  - **B Data Type**: Selects signed or unsigned input for operand B. In IP Integrator, this parameter is auto-updated.
  - **B Width**: Determines the width, in bits, of the B operand. In IP Integrator, this parameter is auto-updated.
- **Parallel Multiplier Options**: These options are visible only when the multiplier type chosen is Parallel Multiplier.
  - **Multiplier Construction**: Allows the choice of LUTs or dedicated multiplier primitives to be selected for the core implementation.
  - **Optimization Options**:
    - **DSP48E1 Slice:** Speed or area optimization can be selected for multiplier sizes up to 47x47. Speed optimization makes full use of multiplier primitives to provide the highest performance implementation. Area optimization uses a mixture of slice logic and dedicated multiplier primitives to reduce DSP Slice-based multiplier utilization, while still providing reasonable performance. For sizes above 47x47, only optimization for speed is allowed.
    - **LUT-based multipliers:** Area optimization allows reduced latency and LUT utilization, at the expense of achievable clock frequency. The area optimization is most effective when both input operands are unsigned, and both input operands are smaller than 16 bits.
- **Constant-Coefficient Multiplier Options**: These options are visible only when the multiplier type chosen is Constant-Coefficient Multiplier.
  - **Coefficient**: Enter the integer value of the coefficient within the limits of the range shown. Positive and negative coefficients are supported. The input type (signed or unsigned) for the constant (B) port is automatically configured by the Vivado IDE

based on the integer constant entered. You can select whether the A port is signed or unsigned.

- **Memory Options**: Select if the multiplier should be implemented with distributed memory, block memory, or using DSP Slices.

- **Output Product Range**: The Vivado IDE automatically configures the output product width to represent the full product, based on the widths of the input operands.
  - **Use Custom Output Width**: The number of product bits can be customized if only a portion of the full product is required for an application by setting the MSB and LSB range.
  - **Use Symmetric Rounding**: For DSP Slice-based parallel multipliers, the product can be symmetrically rounded towards infinity if required. This is the same behavior as the MATLAB® software *round* function. The multiplier must fit on exactly one DSP Slice, and the LSB of the product must lie within the full-range product width.
- **Pipelining and Control Signals**:
  - **Pipeline Stages**: Select the level of pipelining for the multiplier instance. The label on the right provides feedback on the optimum number of pipeline stages for maximum performance. The core assumes that all inputs are registered.
    - Pipeline Stages = 0 implies that the core is combinatorial.
    - Pipeline Stages = 1 implies that only the core output is registered.
    - Pipeline Stages > 1 cause registers to be inserted between input and output up to the optimum pipeline stages value. Adding more registers improves achievable clock speed while increasing latency.
    - Pipeline Stages set to a value greater than the optimum value fully-pipelines the core and causes SRL16-based shift registers to be added at the output to implement the extra latency.
  - **Clock Enable**: Select if all registers in the design have a clock enable control.
  - **Synchronous Clear**: Select if all registers in the design have a synchronous reset control.
  - **SCLR/CE Priority:** When both SCLR and CE pins are present, the priority of SCLR and CE can be selected. The fewest resources are used, and best performance is achieved, when SCLR overrides CE.
- **Resource Estimates Tab**: Clicking the resource estimates tab below the Vivado IDE symbol displays an estimate of the FPGA resources used for a particular multiplier instance. The values update instantaneously with changes in the Vivado IDE, allowing trade-offs in implementation to be evaluated immediately.

Using the Multiplier IP Core

The Vivado IDE performs error-checking on all input parameters. Resource estimation and optimum latency information are also available.

User Parameters

Table 4-1 shows the relationship between the fields in the Vivado IDE and the User Parameters (which can be viewed in the Tcl Console).

Table 4-1: **Vivado IDE Parameter to User Parameter Relationship**

Vivado IDE Parameter/Value[1]	User Parameter/Value[1]	Default Value
Multiplier Type	multtype	Parallel_multiplier
Data Type (under A)	portatype	Signed
Width (under A)	portawidth	18
Data Type (under B)	portbtype	Signed
Width (under B)	portbwidth	18
Constant Value (Integer)	constvalue	129
Memory Type	ccmimp	Distributed_Memory
Multiplier Construction	multiplier_construction	Use_LUTs
Optimization Options	optgoal	Speed
Area Optimized	Area	
Speed Optimized	Speed	
Use Custom Output Width	use_custom_output_width	False
Output MSB	outputwidthhigh	35
Output LSB	outputwidthlow	0
Use Symmetric Rounding	userounding	False
Pipeline Stages	pipestages	1
Clock Enable	clockenable	False
Synchronous Clear	syncclear	False
Synchronous Controls and Clock Enable (CE) Priority	sclrcepriority	SCLR_Overrides_CE

1. Parameter values are listed in the table where the Viado IDE parameter value differs from the user parameter value. Such values are shown in this table as indented below the associated parameter.

Output Generation

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 2].

System Generator for DSP

This section describes each tab of the System Generator for DSP Vivado IDE and details the parameters that differ from the Vivado IDE. The Multiplier core can be found in the Xilinx® Blockset in the Math section. The block is called 'Mult.' See the System Generator for DSP help page for the 'Mult' block for more information on parameters not mentioned here.

Tab 1: Basic

The Basic tab is used to specify the data types and control pins in a similar way to pages 1 and 3 of the Vivado IDE.

- **Precision:** Selecting "User Defined" allows Signed or Unsigned options to be selected. Both ports must be of the same type. Otherwise, System Generator for DSP automatically sets the input width parameters based on the signal properties of the "a" and "b" input ports.
- **Optional Port:** "Provide enable port" specifies whether the core has a clock enable pin (the equivalent of selecting the CE option in the Vivado IDE).
- **Latency:** Specify the latency required for the multiplier. This is equivalent to the Pipeline Stages setting in the Vivado IDE.

Tab 2: Advanced

The Advanced tab has no equivalent parameters on the Vivado IDE. The option to override with doubles applies to System Generator for DSP only.

Tab 3: Implementation

The Implementation tab is used to specify the optimization options in a similar way to page 2 of the Vivado IDE.

- **Use embedded multipliers**: Specifies if DSP Slices should be used to construct the multiplier. If this is unchecked, LUTs are used instead.
- **Optimize for speed/area:** Specifies if the multiplier, when built with DSP Slices, should be optimized for speed (using more dedicated multiplier resources) or area (using a combination of dedicated multipliers and slice resources, where appropriate).
- **Test for optimum pipelining:** Verifies if the specified latency is the optimal selection for the hardware multiplier which will be created. Latency values that pass this test imply that the core produced would be optimized for speed of operation.

Constraining the Core

This section contains information about constraining the core in the Vivado Design Suite.

Required Constraints

This section is not applicable for this IP core.

Device, Package, and Speed Grade Selections

This section is not applicable for this IP core.

Clock Frequencies

This section is not applicable for this IP core.

Clock Management

This section is not applicable for this IP core.

Clock Placement

This section is not applicable for this IP core.

Banking

This section is not applicable for this IP core.

Transceiver Placement

This section is not applicable for this IP core.

I/O Standard and Placement

This section is not applicable for this IP core.

Simulation

For comprehensive information about Vivado simulation components, as well as information about using supported third-party tools, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 4].

Synthesis and Implementation

For details about synthesis and implementation, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 2].

Appendix A

Migrating and Upgrading

This appendix contains information about migrating a design from the ISE® Design Suite to the Vivado® Design Suite, and for upgrading to a more recent version of the IP core. For customers upgrading in the Vivado Design Suite, important details (where applicable) about any port changes and other impact to user logic are included.

Migrating to the Vivado Design Suite

For information about migrating to the Vivado Design Suite, see the *ISE to Vivado Design Suite Migration Guide* (UG911) [Ref 5].

Updating from Multiplier v9.0 and Later

The Vivado IP update feature can be used to update an existing Multiplier XCO file to version 12.0 of the Multiplier core. The core can then be regenerated to create a new netlist. For more information on this feature, see the *ISE to Vivado Design Suite Migration Guide* (UG911) [Ref 5].

Latency Changes

The latency used for the previous multiplier core is reused when regenerating the core as v12.0 However, some cases might offer reduced latency in v12.0 compared to previous versions. To verify that the latency used is the optimal figure, the updated IP can be opened in Vivado and the latency can be compared with the optimum latency value.

Updating from Versions Prior to Multiplier v9.0

It is not currently possible to automatically update versions of the Multiplier core prior to v9.0. Xilinx recommends that customers use the Multiplier v12.0 Vivado IDE to customize a new core. Some features and configurations are unavailable in Multiplier v12.0. Also, some port names might differ between versions.

Upgrading in the Vivado Design Suite

This section provides information about any changes to the user logic or port designations that take place when you upgrade to a more current version of this IP core in the Vivado Design Suite.

Parameter Changes

There are no changes in parameter values or ranges compared to Multiplier v11.2.

Port Changes

There are no changes in ports or pin polarities compared to Multiplier v11.2.

Other Changes

Simulation

Starting with Multiplier v12.0 (2013.3 version), behavioral simulation models have been replaced with IEEE P1735 Encrypted VHDL. The resulting model is bit and cycle accurate with the final netlist. For more information on simulation, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 4].

Appendix B

Debugging

This appendix includes details about resources available on the Xilinx Support website and debugging tools.

Finding Help on Xilinx.com

To help in the design and debug process when using the Multiplier core, the Xilinx Support web page contains key resources such as product documentation, release notes, answer records, information about known issues, and links for obtaining further product support.

Documentation

This product guide is the main document associated with the Multiplier core. This guide, along with documentation related to all products that aid in the design process, can be found on the Xilinx Support web page or by using the Xilinx® Documentation Navigator.

Download the Xilinx Documentation Navigator from the Downloads page. For more information about this tool and the features available, open the online help after installation.

Answer Records

Answer Records include information about commonly encountered problems, helpful information on how to resolve these problems, and any known issues with a Xilinx product. Answer Records are created and maintained daily ensuring that users have access to the most accurate information available.

Answer Records for this core can be located by using the Search Support box on the main Xilinx support web page. To maximize your search results, use keywords such as:

- Product name
- Tool message(s)
- Summary of the issue encountered

A filter search is available after results are returned to further target the results.

Master Answer Record for the Multiplier

AR: 54506

Technical Support

Xilinx provides technical support in the Xilinx Support web page for this LogiCORE™ IP product when used as described in the product documentation. Xilinx cannot guarantee timing, functionality, or support if you do any of the following:

- Implement the solution in devices that are not defined in the documentation.
- Customize the solution beyond that allowed in the product documentation.
- Change any section of the design labeled DO NOT MODIFY.

To contact Xilinx Technical Support, navigate to the Xilinx Support web page.

Debug Tools

There are many tools available to address Multiplier design issues. It is important to know which tools are useful for debugging various situations.

Vivado Design Suite Debug Feature

The Vivado® Design Suite debug feature inserts logic analyzer and virtual I/O cores directly into your design. The debug feature also allows you to set trigger conditions to capture application and integrated block port signals in hardware. Captured signals can then be analyzed. This feature in the Vivado IDE is used for logic debugging and validation of a design running in Xilinx devices.

The Vivado logic analyzer is used with the logic debug IP cores, including:

- ILA 2.0 (and later versions)
- VIO 2.0 (and later versions)

See the *Vivado Design Suite User Guide: Programming and Debugging* (UG908) [Ref 6].

Simulation Debug

The simulation debug flow for Mentor Graphics Questa Simulator (QuestaSim) is illustrated in Figure B-1. A similar approach can be used with other simulators.

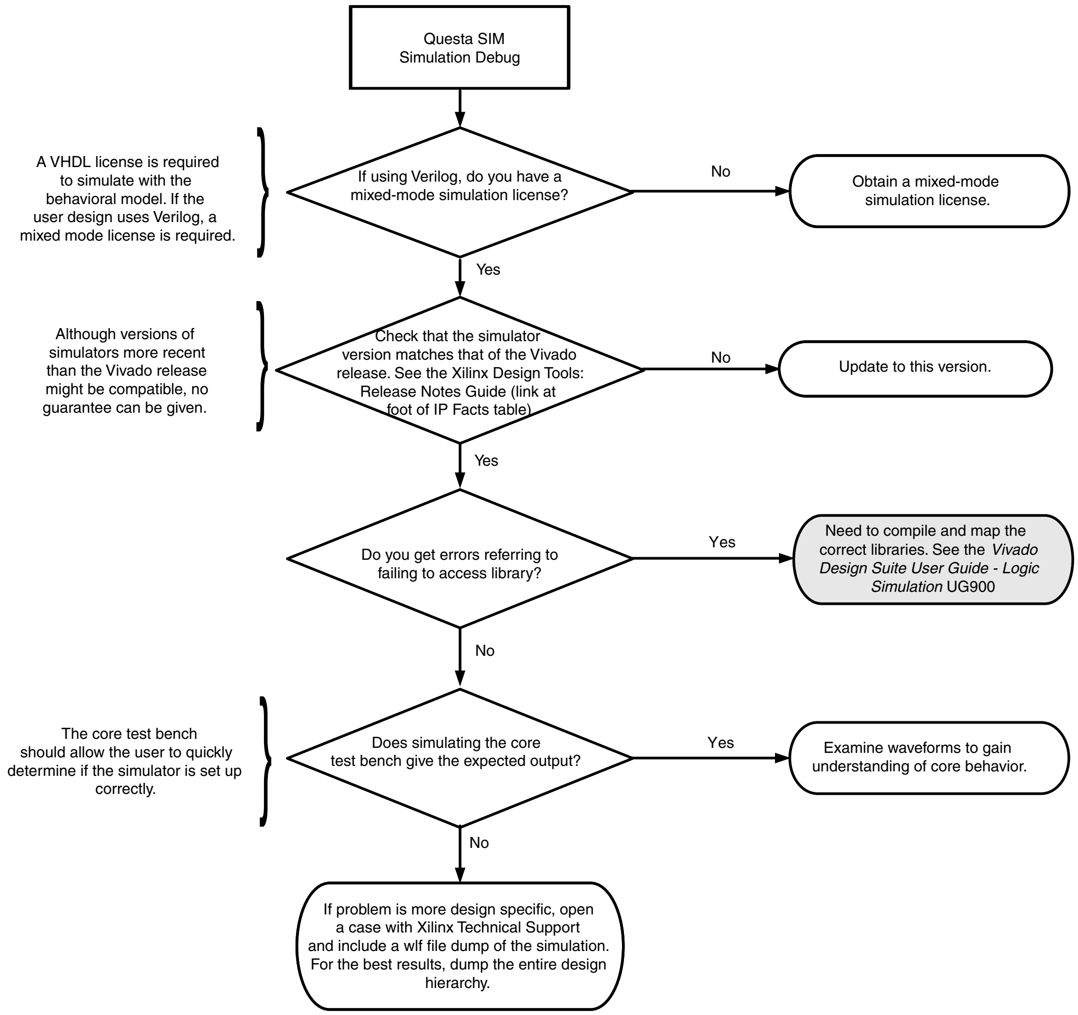
Questa SIM Simulation Debug
If using Verilog, do you have a mixed-mode simulation license?
A VHDL license is required to simulate with the behavioral model. If the user design uses Verilog, a mixed mode license is required.
No
Obtain a mixed-mode simulation license.
Yes
Check that the simulator version matches that of the Vivado release. See the Xilinx Design Tools: Release Notes Guide (link at foot of IP Facts table)
Although versions of simulators more recent than the Vivado release might be compatible, no guarantee can be given.
No
Update to this version.
Yes
Do you get errors referring to failing to access library?
Yes
Need to compile and map the correct libraries. See the Vivado Design Suite User Guide - Logic Simulation UG900
No
Does simulating the core test bench give the expected output?
The core test bench should allow the user to quickly determine if the simulator is set up correctly.
Yes
Examine waveforms to gain understanding of core behavior.
No
If problem is more design specific, open a case with Xilinx Technical Support and include a wlf file dump of the simulation. For the best results, dump the entire design hierarchy.

Figure B-1: **QuestaSim Debug Flow Diagram**

Additional Resources and Legal Notices

Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see Xilinx Support.

References

These documents provide supplemental material useful with this product guide:

1. *Vivado® Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994)
2. *Vivado Design Suite User Guide: Designing with IP* (UG896)
3. *Vivado Design Suite User Guide: Getting Started* (UG910)
4. *Vivado Design Suite User Guide: Logic Simulation* (UG900)
5. *ISE® to Vivado Design Suite Migration Guide* (UG911)
6. *Vivado Design Suite User Guide: Programming and Debugging* (UG908)

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
11/18/2015	12.0	Added support for UltraScale+ families.
04/02/2014	12.0	• Added link to resource utilization figures • Added User parameter table (Table 4-1)
12/18/2013	12.0	• Added UltraScale™ architecture support information. • Added Simulation, Synthesis, Example Design and Test Bench chapters. • Updated Migrating appendix.
10/02/2013	12.0	Minor updates to IP Facts table and Migrating appendix. Document version number advanced to match the core version number.
03/20/2013	1.0	Initial release as a Product Guide; replaces DS255. Added support for Artix®-7 and Zynq®-7000. No other documentation changes.

Please Read: Important Legal Notices

The information disclosed to you hereunder (the "Materials") is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available "AS IS" and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of Xilinx's limited warranty, please refer to Xilinx's Terms of Sale which can be viewed at http://www.xilinx.com/legal.htm#tos; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in such critical applications, please refer to Xilinx's Terms of Sale which can be viewed at http://www.xilinx.com/legal.htm#tos.