LogiCORE IP Multiplier v11.2

DS255 March 1, 2011 **Product Specification**

Introduction

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

Features

- Drop-in module for Virtex®-7 and Kintex™-7, Virtex-6, Virtex-5, Virtex-4, Spartan®-6, Spartan-3/XA, Spartan-3E/XA, Spartan-3A/3AN/3A DSP/XA FPGAs
- 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
- Resource estimation in the Xilinx CORE Generator™ graphical user interface (GUI)
- Supports symmetric rounding to infinity for Virtex and Kintex device multipliers when using the XtremeDSP™ slice
- For use with Xilinx CORE Generator and Xilinx System Generator for DSP 13.1

LogiCORE IP Facts Table					
Core Specifics					
Supported Device Family[1]	Virtex-7 and Kintex-7 Virtex-6, Virtex-5, Virtex-4, Spartan-6, Spartan-3/XA, Spartan-3E/XA, Spartan-3A/3AN/3A DSP/XA				
Supported User Interfaces	Not Applicable				
	Resources[2]				**Frequency**
Configuration	**LUTs**	**FFs**	**DSP Slices**	**Block RAMs**	**Max. Freq.**
9x9, Use LUTs, Speed optimized	115	110	0	0	450 MHz
25x18, Use Mults, Speed optimized	0	0	1	0	450 MHz
Provided with Core					
Documentation	Product Specification				
Design Files	Netlist				
Example Design	Not Provided				
Test Bench	Not Provided				
Constraints File	Not Applicable				
Simulation Model	VHDL behavioral model in the xilinxcorelib library VHDL UniSim structural model Verilog UniSim structural model				
Tested Design Tools					
Design Entry Tools	CORE Generator 13.1 System Generator for DSP 13.1				
Simulation	Mentor Graphics ModelSim 6.6d Cadence Incisive Enterprise Simulator (IES) 10.2 Synopsys VCS and VCS MX 2010.06 ISIM 13.1				
Synthesis Tools	N/A				
Support					
Provided by Xilinx, Inc.					

1. For a complete listing of supported devices, see the release notes for this core.
2. Resources listed here are for Virtex-6 devices. For more complete device performance numbers, see Performance and Resource Utilization, page 7.

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.

Important: Multiplier v11.2 is not backward compatible with version 8.0 (and earlier) of the Multiplier core. See Performance and Resource Utilization for more information.

Pinout

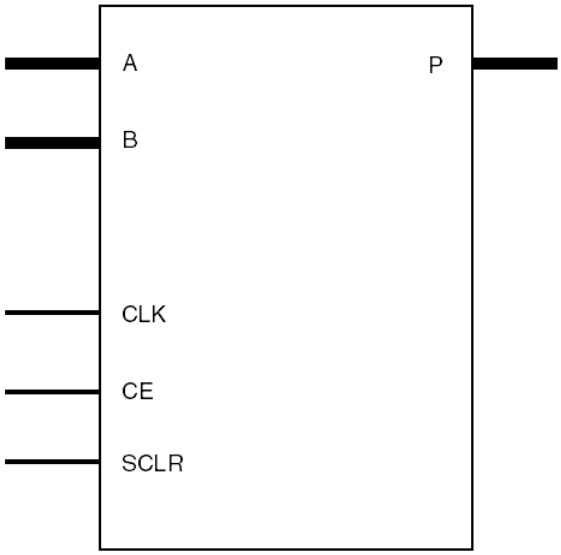

Figure 1: **Core Schematic Symbol**

Figure 1 and Table 1 illustrate and define the schematic symbol signal names. All control inputs are active high. Should an active low input be required for a specific control pin, an inverter must be placed in the path to the pin and is absorbed appropriately during synthesis and/or mapping.

Table 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
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

CORE Generator Graphical User Interface Parameters

The Multiplier core GUI has several pages with fields to set parameter values for the particular instantiation required. This section provides a description of each GUI 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.
- **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**:
    - **XtremeDSP slice and MULT18X18 multipliers:** 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 MULT18X18/XtremeDSP slice utilization, while still providing reasonable performance. For sizes above 47x47, only optimization for speed is allowed.
    - **LUT-based multipliers:** For FPGA devices with LUT6s, area and speed optimization is offered. 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. Speed optimization implements the same architecture for both LUT4 and LUT6 based FPGA devices.
- **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 GUI based on the integer constant entered. The user may 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 embedded multiplier blocks.
- **Output Product Range**: The GUI 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 XtremeDSP 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 XtremeDSP slice (maximum signed operand widths of 25x18 for Kintex-7, Virtex-7, Virtex-6 and Virtex-5, and 18x18 for Virtex-4 devices), 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 GUI symbol displays an estimate of the FPGA resources used for a particular multiplier instance. The values update instantaneously with changes in the GUI, allowing trade-offs in implementation to be evaluated immediately.

Using the Multiplier IP Core

The CORE Generator GUI performs error-checking on all input parameters. Resource estimation and optimum latency information are also available.

Several files are produced when a core is generated, and customized instantiation templates for Verilog and VHDL design flows are provided in the .veo and .vho files, respectively. For detailed instructions, see the CORE Generator software documentation.

Simulation Models

The core has a number of options for simulation models:

- VHDL behavioral model in the xilinxcorelib library
- VHDL UniSim-based structural simulation model
- Verilog UniSim-based structural simulation model

The models required may be selected in the CORE Generator project options.

Xilinx recommends that simulations utilizing UniSim-based structural models be run using a resolution of 1 ps. Some Xilinx library components require a 1 ps resolution to work properly in either functional or timing simulation. The UniSim-based structural simulation models may produce incorrect results if simulated with a resolution other than 1 ps. See the "Register Transfer Level (RTL) Simulation Using Xilinx Libraries" section in *Chapter 6 of the Synthesis and Simulation Design Guide* for more information. This document is part of the ISE® Software Manuals set available at www.xilinx.com/support/software_manuals.htm.

XCO Parameters

Table 2 defines valid entries for the XCO parameters. Parameters are not case sensitive. Default values are displayed in bold.

Xilinx strongly suggests that XCO parameters not be manually edited in the XCO file; instead, use the CORE Generator GUI to configure the core and perform range and parameter value checking.

Table 2: **XCO Parameters**

XCO Parameter	Valid Values
component_name	ASCII text using characters: a..z, 0..9 and '_' ; starting with a letter
PortAWidth	1 - 64 (Unsigned data) 2 - 64 (Signed data) Default value is **18**
PortAType	**Signed**, Unsigned
PortBWidth	1 - 64 (Unsigned data) 2 - 64 (Signed data) Default value is **18**
PortBType	**Signed**, Unsigned
Use_Custom_Output_Width	**false**, true (must be set to 'true' to vary OutputWidthHigh/Low from full-precision)
OutputWidthHigh	PortAWidth+PortBWidth-1 (default value is **35**)
OutputWidthLow	0 to PortAWidth+PortBWidth-1 (default value is **0**)
MultType	**Parallel_Multiplier**, Constant_Coefficient_Multiplier
Multiplier_Construction	**Use_LUTs**, Use_Mults
OptGoal	**Speed**, Area
ConstValue	Integer constant representing any binary value up to 64 bits (**129**)
CcmImp	**Distributed_Memory**, Block_Memory, Dedicated_Multiplier
PipeStages	Integer in the range 0 to 30 (default value is **1**)
UseRounding	**false**, true
RoundPoint	Integer value less than OutputWidthHigh where binary point lies (**0**)
ClockEnable	**false**, true
SyncClear	**false**, true
SclrCePriority	**SCLR_Overrides_CE**, CE_Overrides_SCLR

Migrating to Multiplier v11.2 from Earlier Versions

Updating from Multiplier v9.0 and Later

The CORE Generator core update feature may be used to update an existing Multiplier XCO file to version 11.2 of the Multiplier core. The core may then be regenerated to create a new netlist. See the CORE Generator documentation for more information on this feature.

Port Changes

There are no differences in port naming conventions, polarities, priorities or widths between versions.

Latency Changes

The latency used for the previous multiplier core is reused when regenerating the core as v11.2. However, some cases may offer reduced latency in v11.2 compared to previous versions. To verify that the latency used is the optimal figure, the updated XCO file may be loaded into CORE Generator and the latency on page 3 of the GUI 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 v11.2 GUI to customize a new core. Some features and configurations are unavailable in Multiplier v11.2. Also, some port names may differ between versions.

System Generator for DSP Graphical User Interface

This section describes each tab of the System Generator for DSP GUI and details the parameters that differ from the CORE Generator GUI. The Multiplier core may 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 CORE Generator GUI.

- **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 will have a clock enable pin (the equivalent of selecting the CE option in the CORE Generator GUI).
- **Latency:** Specify the latency required for the multiplier. This is equivalent to the Pipeline Stages setting in the CORE Generator GUI.

Tab 2: Advanced

The Advanced tab has no equivalent parameters on the CORE Generator GUI. 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 CORE Generator GUI.

- **Use embedded multipliers:** Specifies if embedded multipliers/XtremeDSP slices should be used to construct the multiplier. If this is unchecked, LUTs will be used instead.
- **Optimize for speed/area:** Specifies if the multiplier, when built with embedded multipliers or XtremeDSP slices, should be optimized for speed (using more dedicated multiplier resources) or area (using a combination of dedicated multipliers and slice resources, where appropriate). LUT-based multipliers targeting Kintex-7, Virtex-7, Virtex-6, Virtex-5 and Spartan-6 devices may also be optimized for performance or resources.
- **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.

Performance and Resource Utilization

Tables 3 through 6 provide performance and resource usage information for a number of different multiplier configurations. Configurations not supported for a particular FPGA device are grayed-out.

In each case, the Core Latency value shown represents the fully-pipelined latency of that core configuration. Reduced resource utilization may be possible with a reduction in latency, at the expense of maximum achievable clock frequency.

The maximum clock frequency results were obtained by double-registering input and output ports to reduce dependence on I/O placement. The inner level of registers used a separate clock signal to measure the path from the input registers to the first output register through the core.

The resource usage results do not include the preceding "characterization" registers and represent the true logic used by the core to implement a single multiplier. LUT counts include LUTs used as route-throughs and shift registers.

The map options used were: "`map -pr b -ol high`"

The par options used were: "`par -ol high`"

Clock frequency does not take clock jitter into account and should be derated by an amount appropriate to the clock source jitter specification.

The maximum achievable clock frequency and the resource counts may also be affected by other tool options, additional logic in the FPGA device, using a different version of Xilinx tools, and other factors.

The Virtex-6 FPGA test cases in Table 3 used an XC6VLX75T-FF784 (-1 speed grade) device and ISE speed file version "ADVANCED 1.01g 2009-07-27."

Table 3: **Virtex-6 FPGA Family Performance and Resource Utilization**

Multiplier Configuration	Data Type	Core Latency (Cycles)	Maximum Clock Frequency (MHz)	LUT/FF Pairs	LUT6s	FFs	XtremeDSP Slices
9x9 Use LUTs Optimize for Area	Unsigned	3	417	89	89	108	0
9x9 Use LUTs Optimize for Speed	Unsigned	4	450	116	115	110	0
12x12 Use LUTs Optimize for Area	Unsigned	3	373	158	158	180	0
12x12 Use LUTs Optimize for Speed	Unsigned	4	450	176	176	179	0
18x18 Use Mults Optimize for Speed	Signed	3	450	0	0	0	1
20x20 Use Mults Optimize for Speed	Signed	4	450	14	3	17	2
20x20 Use Mults Optimize for Area	Signed	3	450	23	23	24	1
24x24 Use Mults Optimize for Speed	Unsigned	4	450	12	5	17	2
24x24 Use Mults Optimize for Area	Unsigned	5	450	202	200	236	1
25x18 Use Mults Optimize for Speed	Signed	3	450	0	0	0	1
25x18 Use Mults Optimize for Area	Signed	3	450	0	0	0	1
35x35 Use Mults Optimize for Speed	Signed	6	450	37	33	69	4
53x53 Use Mults Optimize for Speed	Unsigned	12	450	150	133	276	10

The Virtex-5 FPGA test cases in Table 4 used an XC5VLX50-FF1153 (-1 speed grade) device and ISE speed file version "PRODUCTION 1.65 2009-07-27, STEPPING level 0."

Table 4: **Virtex-5 FPGA Family Performance and Resource Utilization**

Multiplier Configuration	Data Type	Core Latency (Cycles)	Maximum Clock Frequency (MHz)	LUT/FF Pairs	LUT6s	FFs	XtremeDSP Slices
9x9 Use LUTs Optimize for Area	Unsigned	3	340	108	90	108	0
9x9 Use LUTs Optimize for Speed	Unsigned	4	450	123	109	110	0
12x12 Use LUTs Optimize for Area	Unsigned	3	328	180	168	180	0
12x12 Use LUTs Optimize for Speed	Unsigned	4	423	185	167	179	0
18x18 Use Mults Optimize for Speed	Signed	3	450	0	0	0	1
20x20 Use Mults Optimize for Speed	Signed	4	450	17	0	17	2
20x20 Use Mults Optimize for Area	Signed	3	416	24	24	24	1
24x24 Use Mults Optimize for Speed	Unsigned	4	450	17	0	17	2
24x24 Use Mults Optimize for Area	Unsigned	5	395	239	170	236	1
25x18 Use Mults Optimize for Speed	Signed	3	450	0	0	0	1
25x18 Use Mults Optimize for Area	Signed	3	450	0	0	0	1
35x35 Use Mults Optimize for Speed	Signed	6	450	87	35	87	4
53x53 Use Mults Optimize for Speed	Unsigned	12	450	280	229	280	10

The Spartan-6 FPGA test cases in Table 5 used an XC6SLX45T-FGG484 (-2 speed grade) device and ISE speed file version "ADVANCED 1.01e 2009-07-27."

Table 5: **Spartan-6 Family Performance and Resource Utilization**

Multiplier Configuration	Data Type	Core Latency (Cycles)	Maximum Clock Frequency (MHz)	LUT/FF Pairs	LUT6s	FFs	XtremeDSP Slices
9x9 Use LUTs Optimize for Area	Unsigned	3	214	91	89	99	0
9x9 Use LUTs Optimize for Speed	Unsigned	4	250	110	109	105	0
12x12 Use LUTs Optimize for Area	Unsigned	3	212	158	158	179	0
12x12 Use LUTs Optimize for Speed	Unsigned	4	250	177	177	167	0
18x18 Use Mults Optimize for Speed	Signed	3	250	0	0	0	1
20x20 Use Mults Optimize for Speed	Signed	8	250	44	43	77	4
20x20 Use Mults Optimize for Area	Signed	4	250	67	67	67	1
24x24 Use Mults Optimize for Speed	Unsigned	8	250	47	46	86	4
24x24 Use Mults Optimize for Area	Unsigned	6	236	358	355	396	1
25x18 Use Mults Optimize for Speed	Signed	5	250	13	13	25	2
25x18 Use Mults Optimize for Area	Signed	5	250	161	158	187	1
35x35 Use Mults Optimize for Speed	Signed	8	250	58	58	107	4
53x53 Use Mults Optimize for Speed	Unsigned	24	238	300	297	504	16

The Spartan-3A DSP FPGA test cases in Table 6 used an XC3SD3400A-FG676 (-4 speed grade) device and ISE speed file version "PRODUCTION 1.33 2009-07-27."

Table 6: **Spartan-3A DSP Family Performance and Resource Utilization**

Multiplier Configuration	Data Type	Core Latency (Cycles)	Maximum Clock Frequency (MHz)	Slices	LUT4s	FFs	XtremeDSP Slices
9x9 Use LUTs Optimize for Area	Unsigned						
9x9 Use LUTs Optimize for Speed	Unsigned	4	215	67	113	104	0
12x12 Use LUTs Optimize for Area	Unsigned						
12x12 Use LUTs Optimize for Speed	Unsigned	4	199	93	179	165	0
18x18 Use Mults Optimize for Speed	Signed	3	250	0	0	0	1
20x20 Use Mults Optimize for Speed	Signed	8	250	56	72	75	4
20x20 Use Mults Optimize for Area	Signed	4	195	36	67	67	1
24x24 Use Mults Optimize for Speed	Unsigned	8	250	55	65	72	4
24x24 Use Mults Optimize for Area	Unsigned	6	166	218	359	397	1
25x18 Use Mults Optimize for Speed	Signed	5	250	21	17	25	2
25x18 Use Mults Optimize for Area	Signed	5	193	101	135	188	1
35x35 Use Mults Optimize for Speed	Signed	8	250	70	87	105	4
53x53 Use Mults Optimize for Speed	Unsigned	24	245	298	492	454	16

Support

Xilinx provides technical support for this LogiCORE product when used as described in the product documentation. Xilinx cannot guarantee timing, functionality, or support of product if implemented in devices that are not defined in the documentation, if customized beyond that allowed in the product documentation, or if changes are made to any section of the design labeled *DO NOT MODIFY*.

Refer to the IP Release Notes Guide (XTP025) for further information on this core. There is a link to all the DSP IP and then to each core. For each core, there is a master Answer Record that contains the Release Notes and Known Issues list for each core. The following information is listed for each version of the core:

- New Features
- Bug Fixes
- Known Issues

Ordering Information

This LogiCORE IP module is included at no additional cost with the Xilinx ISE Design Suite software and is provided under the terms of the Xilinx End User License Agreement. Use the CORE Generator software included with the ISE Design Suite to generate the core. For more information, please visit the core page.

Please contact your local Xilinx sales representative for pricing and availability of additional Xilinx LogiCORE modules and software. Information about additional Xilinx LogiCORE modules is available on the Xilinx IP Center.

Revision History

Date	Version	Description of Revisions
30/28/03	1.0	Revision History added to document.
02/27/04	1.1	Updated copyright, release date, UI screen shots and template.
05/21/04	1.2	Updated with Virtex-4 and CORE Generator system v6.2i.
04/28/05	1.3	Updated document to indicate support for Xilinx software v7.1i and Spartan-3E device.
01/18/06	2.0	Completely revised layout, removed UI screen shots, added VHDL generic information and direct instantiation usage details.
07/13/06	3.0	Updated with new generic information and performance data. Removed table of XCO parameters. Removed timing diagrams since handshaking control signals were deprecated.
02/15/07	3.1	Updated for ISE 9.1i support, Spartan-3A/3AN device, and updated the performance data. Added section on GUI parameters.
04/02/07	3.5	Added support for Spartan-3A DSP devices.
04/25/08	3.6	Updated for ISE v10.1 support and updated the performance data table. Removed section on direct instantiation.
04/24/09	4.0	Update for ISE 11.1 support.
09/16/09	4.1	Update for ISE 11.3 support.
03/01/11	4.2	Support added for Virtex-7 and Kintex-7. ISE Design Suite 13.1

Notice of Disclaimer