LatticeCORE

Divider IP Core User's Guide

June 2012
ipug108_01.0

Table of Contents

Chapter 1. Introduction 3
- Quick Facts 3
- Features 3
- Release Information 3
- Device Support 3

Chapter 2. Functional Description 4
- Key Concepts 4
- Block Diagram 4
- Primary I/O 4
- Timing Specifications 5

Chapter 3. Parameter Settings 7
- Implementation 8

Chapter 4. IP Core Generation 9
- Licensing the IP Core 9
- Getting Started 9
- Configuring the Divider IP Core in IPexpress 9
  - IPexpress-Created Files and Top Level Directory Structure 12
  - Instantiating the Core 13
  - Running Functional Simulation 13
  - Synthesizing and Implementing the Core in a Top-Level Design 13
- Hardware Evaluation 14
- Updating/Regenerating the IP Core 14
  - Regenerating an IP Core in Diamond 14

Chapter 5. Support Resources 16
- Lattice Technical Support 16
  - Online Forums 16
  - Telephone Support Hotline 16
  - E-mail Support 16
  - Local Support 16
  - Internet 16
- References 16
- .Revision History 16

Appendix A. Resource Utilization 17
- LatticeECP3 Devices 17
  - Ordering Part Number 17
- LatticeECP2M Devices 17
  - Ordering Part Number 17
- LatticeECP2 Devices 17
  - Ordering Part Number 18
- LatticeXP2 Devices 18
  - Ordering Part Number 18

© 2012 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal. All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

Introduction

The Divider IP core is a one-clock divider which completes one integer division every clock. It supports signed or unsigned inputs and provides configurable output latency.

Quick Facts

Table 1-1. Divider IP Core Quick Facts

		Divider IP Core Configuration		
Core Requirements	FPGA Families Supported	LatticeECP3™	LatticeECP2/M	LatticeXP2™
	Minimum Device Required	LFE3-17EA	LFE2-6E	LFXP2-5E
	Targeted Device	LFE3-35EA-8FN672C	LFE2-35E-7F672C	LFXP2-30E-7F672C
Resource Utilization	Configuration	20-bit numerator, 10-bit denominator, 20 output latency		
	Registers	828	828	886
	LUTs	311	311	368
	Slices	446	446	484
Resource Utilization	Configuration	24-bit numerator, 12-bit denominator, 12 output latency		
	Registers	586	586	619
	LUTs	409	409	442
	Slices	409	409	431
Resource Utilization	Configuration	32-bit numerator, 32-bit denominator, 32 output latency		
	Registers	3127	3123	3137
	LUTs	1459	1459	1458
	Slices	1791	1788	1791
Design Tool Support	Lattice Implementation	Lattice Diamond® 2.0		
	Synthesis	Synopsys® Synplify™ Pro for Lattice F-2012.03L		
	Simulation	Aldec® Active-HDL™ 9.1 Lattice Edition Mentor Graphics® ModelSim™ SE 6.3F		

Features

- Supports signed or unsigned numerator and denominator
- Supports numerator and denominator data width 4-64
- Supports forced positive remainder
- Supports configurable output latency
- Optional clock enable and data valid ports

Release Information

- Divider IP Core version 1.0
  - Last updated May, 2012

Device Support

- , LatticeECP3, LatticeECP2M™, LatticeECP2™ and LatticeXP2 FPGA Families

Key Concepts

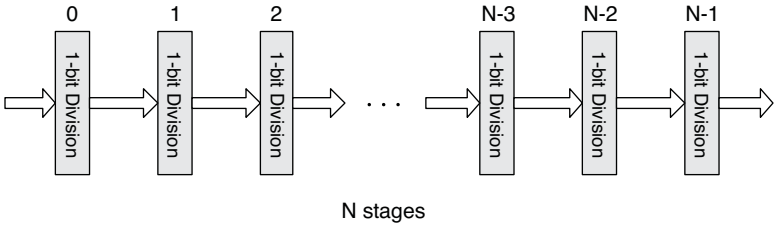
The Divider IP core implements integer division with the formula:

Numerator = Denominator * Quotient + Remainder

Numerator and Denominator can be signed or unsigned integers. When either the Numerator or Denominator is a signed integer, Quotient and Remainder will also be in signed integer format. When the Remainder is configured to be “Always Positive Remainder”, the Remainder will be a positive signed integer value.

Block Diagram

Figure 2-1. Divider IP Core Block Diagram



The Divider IP core uses a non-restoring division algorithm to implement the integer division operation.

There are N stages of 1-bit division in an integer division operation, where N is the width of the quotient.

Each stage generates a 1-bit quotient and partial-remainder. In the last stage, the final quotient and remainder are generated. 1-bit division uses an adder-subtracter to compare the partial remainder and denominator to get a new partial remainder. Quotient-digit selection is based on the sign of the partial remainder. In the last stage, the partial remainder is corrected to get the final remainder.

The Divider IP core supports configurable output latency. The latency can be any number of clock cycles from 1 to N. When latency is set to the value M, M stages of output registers are uniformly distributed into the N stages of 1-bit division operation. The final division stage always has output registers.

Primary I/O

Figure 2-2. Divider IP Core I/O Diagram

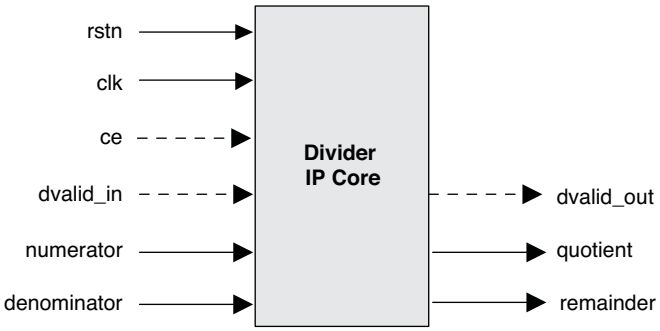


Table 2-1. Primary I/O Descriptions

Port	Size	I/O	Description
General I/Os			
rstn	1	I	Asynchronous active-low reset signal
clk	1	I	Input clock
ce	1	I	Clock enable, active high
dvalid_in	1	I	Optional input data valid signal, active-high
numerator	4-64	I	Input numerator value
denominator	4-64	I	Input denominator value
dvalid_out	1	O	Optional output data valid signal, active high
quotient	4-64	O	Output quotient
remainder	4-64	O	Output remainder

Timing Specifications

The Divider IP core is a one-clock divider. It can accept a numerator and denominator every clock cycle and generate a quotient and remainder every clock cycle.

When the input numerator and denominator are in an unsigned format, the output quotient and remainder are in an unsigned format. When either the numerator or denominator is in a signed format, the output quotient and remainder are always in a signed format.

Figure 2-3. Unsigned Integer Division with a Latency of 8

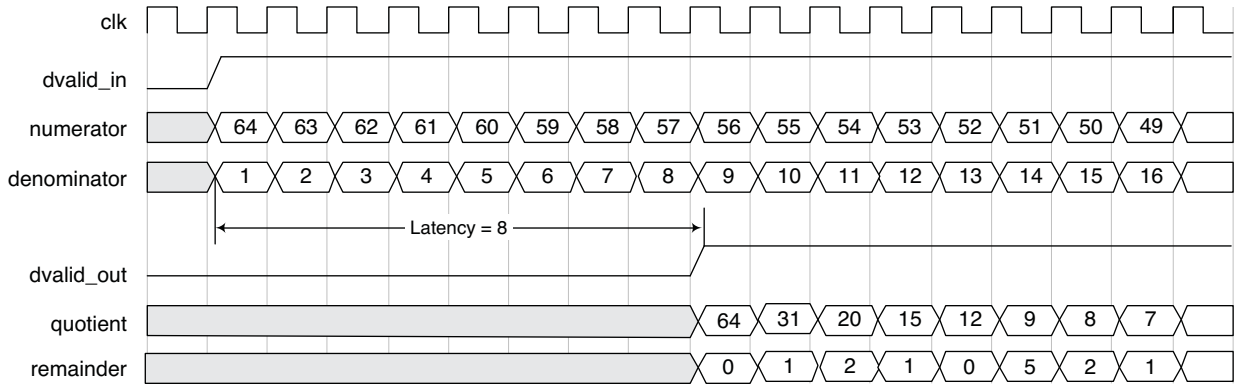


Figure 2-4. Signed Integer Division with a Latency of 8

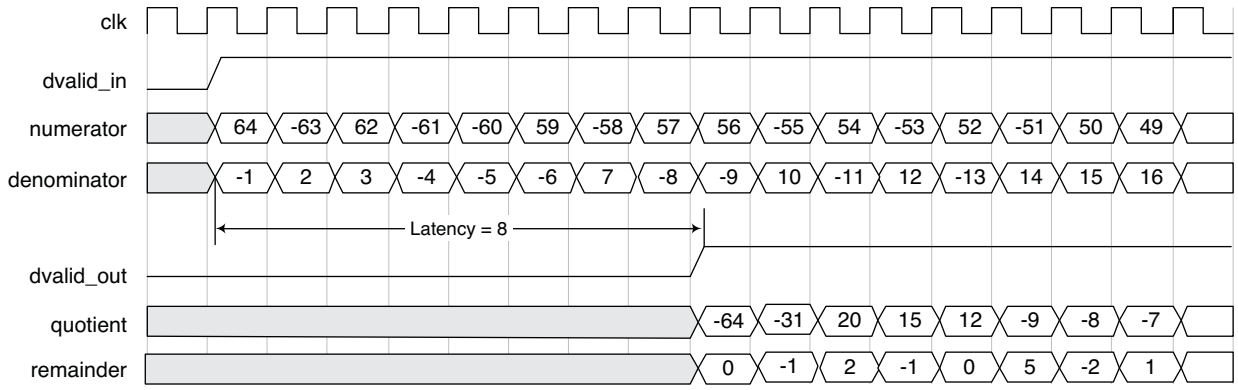
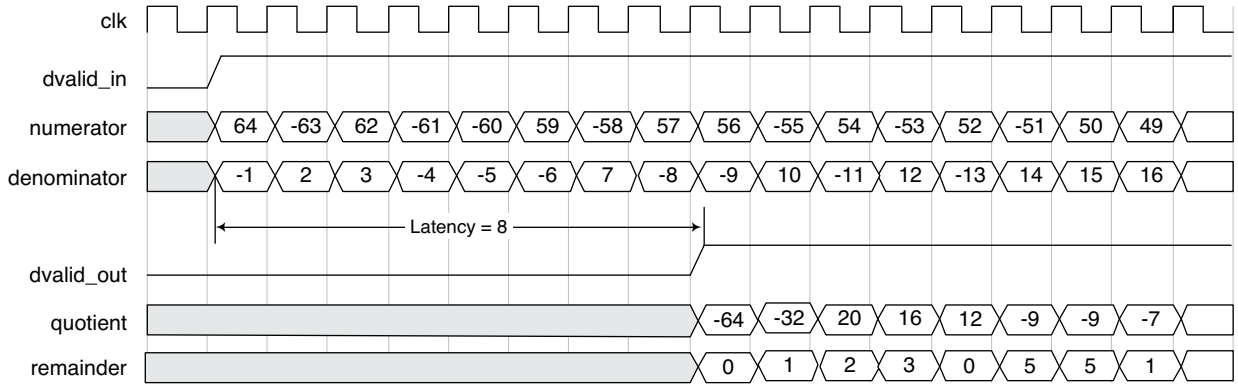


Figure 2-5. Signed Integer Division with a Positive Remainder



Parameter Settings

This section describes how to generate the Lattice Divider IP core using the Diamond IPexpress™ tool. Refer to "IP Core Generation" on page 9 for a description of how to generate the IP core.

The Divider configuration GUI is accessed via the IPexpress tool and provides an interface for setting the desired parameters and invoking the IP Core Generator. Since the values of some parameters affect the size of the core, the maximum value for these parameters may be limited by the size of the target device. Table 3-1 provides a list of user configurable parameters for the Divider IP core.

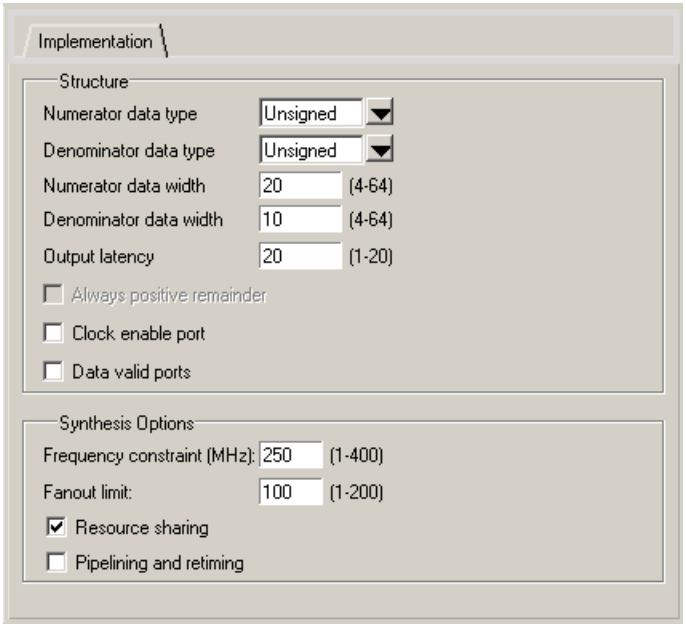
Table 3-1. Divider IP Core Parameters

Parameter	Range/Options	Default
Structure		
Numerator data type	Unsigned, Signed	Unsigned
Denominator data type	Unsigned, Signed	Unsigned
Numerator data width	4-64	20
Denominator data width	4-64	10
Output latency	1-64	20
Always positive remainder	Yes, No	No
Clock enable port	Yes, No	No
Data valid ports	Yes, No	No
Synthesis Options		
Frequency constraint (MHz)	1-400	250
Fanout limit	1-200	100
Resource sharing	Yes, No	Yes
Pipelining and retiming	Yes, No	No

Implementation

The Implementation tab provides settings for Divider structure and synthesis options.

Figure 3-1. Implementation Tab



- **Numerator data type** specifies the data type of input Numerator.
- **Denominator data type** specifies the data type of input Denominator.
- **Numerator data width** specifies the data width of input Numerator.
- **Denominator data width** specifies the data width of input Denominator.
- **Output latency** specifies the output latency of the Divider core. Its value can be between 1 and Numerator data width.
- The **Clock enable port** check box specifies whether the core has a clock enable port.
- The **Data valid ports** check box specifies whether the core has an input data valid and output data valid ports.
- **Frequency constraint** sets the required clock frequency in MHz. This option applies to all of the clocks in the core.
- **Fanout limit** sets the fanout limit value for the synthesis tool.
- **Resource sharing** and **Pipelining and retiming**, if enabled, are synthesis directives that are used in the core generation. Users can adjust these options to get better timing results.

This section provides information on how to generate the Divider IP core using the Diamond IPexpress tool, and how to include the core in a top-level design.

Licensing the IP Core

An IP core- and device-specific license is required to enable full, unrestricted use of the Divider IP core in a complete, top-level design. Instructions on how to obtain licenses for Lattice IP cores are provided on the Lattice website: www.latticesemi.com/products/intellectualproperty/aboutip/index.cfm.

Users may download and generate the Divider IP core and fully evaluate the core through functional simulation and implementation (synthesis, map, place and route) without an IP license. The core supports Lattice's IP hardware evaluation capability, which makes it possible to create versions of the IP core that operate in hardware for a limited time (approximately four hours) without requiring an IP license. See "Hardware Evaluation" on page 14 for further details. However, a license is required to enable timing simulation, to open the design in the Diamond design software and to generate bitstreams that do not include the hardware evaluation timeout limitation.

Getting Started

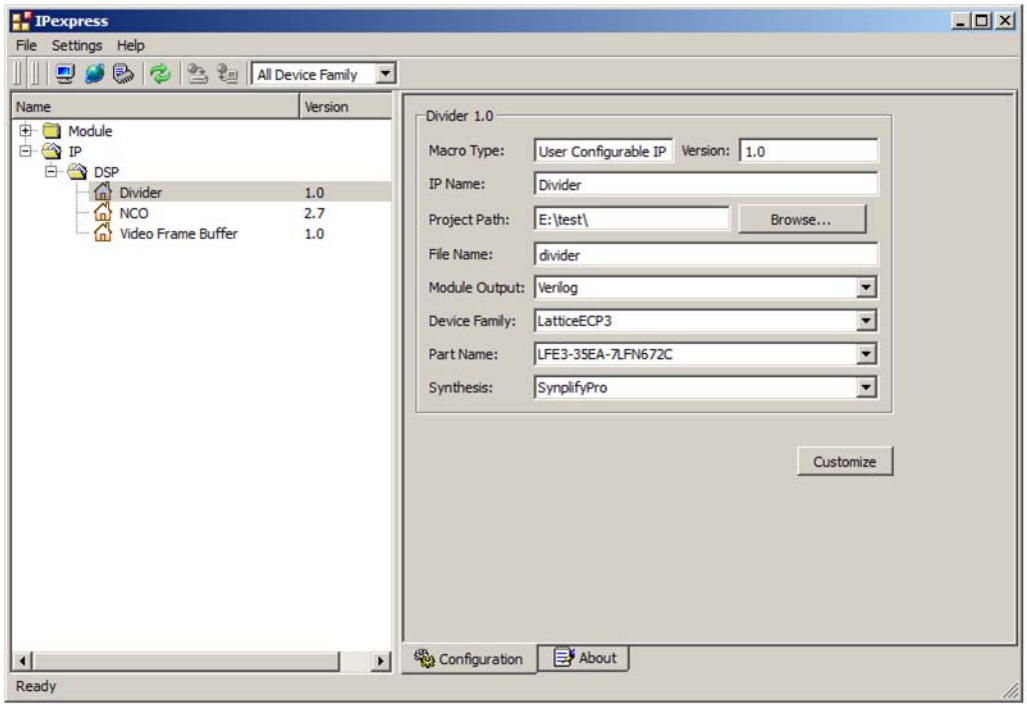
The Divider IP core is available for download from the Lattice IP Server using the IPexpress tool. The IP files are automatically installed using ispUPDATE technology in any user-specified directory. After the IP core has been installed, the IP core is available in the IPexpress GUI dialog box shown in Figure 4-1. To generate a specific IP core configuration, the user specifies:

- **Project Path** – Path to the directory where the generated IP files will be located.
- **File Name** – "username" designation given to the generated IP core and corresponding folders and files.
- **Module Output** – Verilog or VHDL.
- **Device Family** – Device family to which IP is to be targeted. Only families that support the particular IP core are listed.
- **Part Name** – Specific targeted part within the selected device family.

Configuring the Divider IP Core in IPexpress

The Divider IP core configuration GUI is accessed via the Diamond IPexpress tool, and provides an interface for setting the desired parameters and invoking the IP core generator. The start-up IPexpress page allows the user to select the IP core to be generated, the project directory, the user designated module name, the design entry type, and the target device. The "File Name" will be used as the username in the IP core generation. The Divider IP core is found in IPexpress under **IP > DSP**, as shown in Figure 4-1.

Figure 4-1. IPexpress Dialog Box

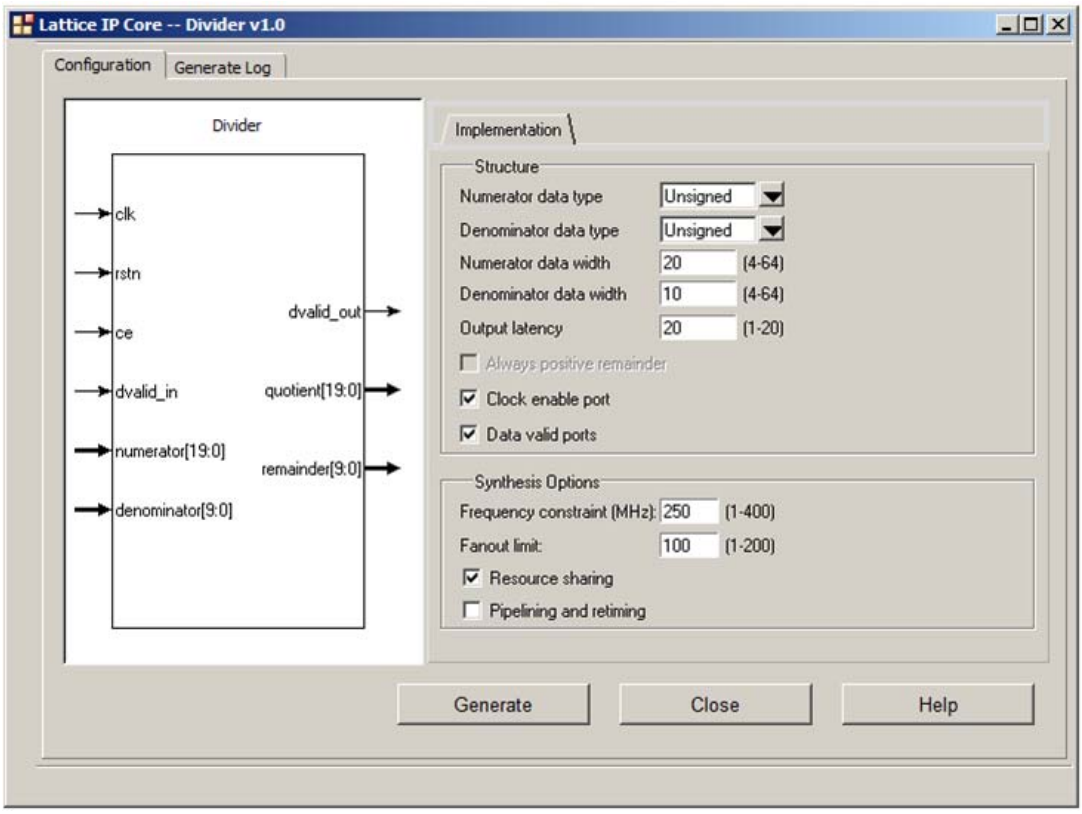


Important Note: File Name cannot be set to "divider_core," as this name has been used in the internal design of the IP core.

Note that if the IPexpress tool is called from within an existing project, Project Path, Module Output, Device Family and Part Name default to the specified project parameters. Refer to the IPexpress tool online help for further information.

To create a custom configuration, the click the **Customize** button in the IPexpress tool dialog box to display the Divider IP core Configuration GUI, as shown in Figure 4-2. From this dialog box, the user can select the IP parameter options specific to their application.

Figure 4-2. Configuration GUI



IPexpress-Created Files and Top Level Directory Structure

When the user clicks the **Generate** button in the IP Configuration dialog box, the IP core and supporting files are generated in the specified Project Path directory. The directory structure of the generated files is shown in Figure 4-3. This example shows the directory structure generated for the Divider IP core for a LatticeECP3 FPGA.

Figure 4-3. Lattice Divider IP Core Directory Structure

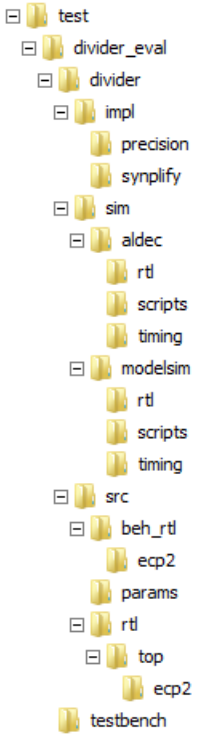


Table 4-1 provides a list of key files and directories created by the IPexpress tool and how they are used. The IPexpress tool creates several files that are used throughout the design cycle. The names of most of the created files are customized to the user's module name specified in the IPexpress tool.

Table 4-1. File List

File	Description
<username>.lpc	This file contains the IPexpress tool options used to recreate or modify the core in the IPexpress tool.
<username>.ipx	The IPX file holds references to all of the elements of an IP core or module after it is generated from the IPexpress tool. The file is used to bring in the appropriate files during the design implementation and analysis. It is also used to re-load parameter settings into the IP core/module generation GUI when an IP core/module is being regenerated.
<username>.ngo	This file provides the synthesized IP core.
<username>_bb.v	This file provides the synthesis black box for the user's synthesis.
<username>_inst.v	This file provides an instance template for Divider IP core.
<username>_beh.v	This file provides the front-end simulation library for Divider IP core.

Table 4-2 provides a list of key additional files providing IP core generation status information and command line generation capability are generated in the user's project directory.

Table 4-2. Additional Files

File	Description
<username>_generate.tcl	This file is created when the GUI **Generate** button is pushed. This file may be run from the command line.
<username>_generate.log	This is the synthesis and map log file.
<username>_gen.log	This is the IPexpress IP generation log file.

Instantiating the Core

The generated Divider IP core package includes black-box (<username>_bb.v) and instance (<username>_inst.v) templates that can be used to instantiate the IP core in a top-level design. An example RTL top-level reference source file that can be used as an instantiation template for the IP core is provided in **\<project_dir>\divider_eval\<username>\src\rtl\top**. Users may also use this top-level reference as the starting template for the top level of their complete design.

Running Functional Simulation

Simulation support for the Divider IP core is provided for the Aldec Active-HDL (Verilog and VHDL) simulator and the Mentor Graphics ModelSim simulator. The functional simulation includes a configuration-specific behavioral model of the Divider IP core. The test bench sources stimulus to the core and monitors output from the core. The generated IP core package includes the configuration-specific behavior model (<username>_beh.v) for functional simulation in the "Project Path" root directory. The simulation script supporting ModelSim evaluation simulation is provided in **\<project_dir>\divider_eval\<username>\sim\modelsim\scripts**. The simulation script supporting Active-HDL evaluation simulation is provided in **\<project_dir>\divider_eval\<username>\sim\aldec\scripts**. Both Modelsim and Active-HDL simulation is supported via test bench files provided in **\<project_dir>\divider_eval\testbench**. Models required for simulation are provided in the corresponding \models folder.

Users may run the Active-HDL evaluation simulation by following the steps below:

1. Open Active-HDL.
2. Under the **Tools** tab, select **Execute Macro**.
3. Browse to folder **\<project_dir>\divider_eval\<username>\sim\aldec\scripts** and execute one of the "do" scripts shown.

Users may run the ModelSim evaluation simulation by following the steps below:

1. Open ModelSim.
2. Under the **File** tab, select **Change Directory** and choose the folder **<project_dir>\divider_eval\<username>\sim\modelsim\scripts**.
3. Under the **Tools** tab, select **Execute Macro** and execute **..\scripts\<username>_rtl_se.do**.

*Note: When the simulation is complete, a pop-up window will appear asking "Are you sure you want to finish?" Choose **No** to analyze the results. Choosing **Yes** closes ModelSim.*

Synthesizing and Implementing the Core in a Top-Level Design

Synthesis support for the Divider IP core is provided for Mentor Graphics Precision RTL or Synopsys Synplify. The Divider IP core itself is synthesized and is provided in NGO format when the IP core is generated in IPexpress. Users may synthesize the core in their own top-level design by instantiating the core in their top-level design as described previously and then synthesizing the entire design with either Synplify or Precision RTL synthesis.

The top-level file **<username>_eval_top.v** provided in **\<project_dir>\divider_eval\<username>\src\top** supports the ability to implement the Divider IP core in isolation. Push-button implementation of this top-level design with either Synplify or Precision RTL synthesis is supported via the project files **<username>_eval.ldf** located in the **\<project_dir>\divider_eval\<username>\impl\synplify** and **\<project_dir>\divider_eval\<username>\impl\precision directories**, respectively.

To use this project file in the Diamond design software:

1. Choose **File > Open > Project**.
2. Browse to **\<project_dir>\divider_eval\<username>\impl\(synplify or precision)** in the **Open Project** dialog box.
3. Select and **open <username>.ldf**. At this point, all of the files needed to support top-level synthesis and implementation will be imported to the project.
4. Select the **Process** tab in the left-hand GUI window.
5. Implement the complete design via the standard Diamond GUI flow.

Hardware Evaluation

The Divider IP core supports the Lattice IP hardware evaluation capability, which makes it possible to create versions of the IP core that operate in hardware for a limited period of time (approximately four hours) without requiring the purchase of an IP license. It may also be used to evaluate the core in hardware in user-defined designs.

Choose **Project > Active Strategy > Translate Design Settings**. The hardware evaluation capability may be enabled or disabled in the Strategy dialog box. It is enabled by default.

Updating/Regenerating the IP Core

By regenerating an IP core with the IPexpress tool, the user can modify any of its settings including device type, design entry method, and any of the options specific to the IP core. Regenerating can be done to modify an existing IP core or to create a new but similar one.

Regenerating an IP Core in Diamond

To regenerate an IP core in Diamond:

1. In IPexpress, click the **Regenerate** button.
2. In the Regenerate view of IPexpress, choose the IPX source file of the module or IP you wish to regenerate.
3. IPexpress shows the current settings for the module or IP in the Source box. Make your new settings in the **Target** box.
4. If you want to generate a new set of files in a new location, set the new location in the **IPX Target File** box. The base of the file name will be the base of all the new file names. The IPX Target File must end with an .ipx extension.
5. Click **Regenerate**. The module's dialog box opens showing the current option settings.
6. In the dialog box, choose the desired options. To get information about the options, click **Help**. Also, check the **About** tab in IPexpress for links to technical notes and user's guides. IP cores may come with additional information. As the options change, the schematic diagram of the module changes to show the I/O and the device resources the module will need.
7. To import the module into your project, if it's not already there, select **Import IPX to Diamond Project** (this is not available in stand-alone mode).

8. Click **Generate**.

9. Check the **Generate Log** tab to check for warnings and error messages.

10. Click **Close**.

The IPexpress package file (.ipx) supported by Diamond contains references to all of the elements of the generated IP core required to support simulation, synthesis and implementation. The IP core may be included in a user's design by importing the .ipx file to the associated Diamond project. To change the option settings of a module or IP core that is already in a design project, double-click the module's .ipx file in the File List view. This opens IPexpress and the module's dialog box showing the current option settings. Then go to step 6 above.

Support Resources

This chapter contains information about Lattice Technical Support, additional references, and document revision history.

Lattice Technical Support

There are a number of ways to receive technical support.

Online Forums

The first place to look is Lattice Forums (www.latticesemi.com/support/forums.cfm). Lattice Forums contain a wealth of knowledge and are actively monitored by Lattice Applications Engineers.

Telephone Support Hotline

Receive direct technical support for all Lattice products by calling Lattice Applications from 5:30 a.m. to 6 p.m. Pacific Time.

- For USA & Canada: 1-800-LATTICE (528-8423)
- For other locations: +1 503 268 8001

In Asia, call Lattice Applications from 8:30 a.m. to 5:30 p.m. Beijing Time (CST), +0800 UTC. Chinese and English language only.

- For Asia: +86 21 52989090

E-mail Support

- techsupport@latticesemi.com
- techsupport-asia@latticesemi.com

Local Support

Contact your nearest Lattice sales office.

Internet

www.latticesemi.com

References

- HB1009, LatticeECP3 Family Handbook
- HB1003, LatticeECP2/M Family Handbook
- HB1004, LatticeXP2 Family Handbook

.Revision History

Date	Document Version	IP Core Version	Change Summary
June 2012	01.0	1.0	Initial release.

Resource Utilization

This appendix gives resource utilization information for Lattice FPGAs using the Divider IP core.

IPexpress is the Lattice IP configuration utility, and is included as a standard feature of the Diamond design tools. Details regarding the usage of IPexpress can be found in the IPexpress and Diamond help system. For more information on the Diamond design tools, visit the Lattice web site at www.latticesemi.com/software.

LatticeECP3 Devices

Table A-1. Performance and Resource Utilization[1]

Numerator Data Type	Numerator Data Width	Denominator Data Type	Denominator Data Width	Output Latency	Registers	LUTs	Slices	f_{MAX}
Unsigned	20	Unsigned	10	20	828	311	446	316
Unsigned	24	Unsigned	12	12	586	409	409	218
Signed	32	Signed	32	32	3127	1459	1791	248

1. Performance and utilization data are generated targeting a LFE3-35EA-8FN672C device using Lattice Diamond 2.0 and Synplify Pro F-2012.03L software. Performance may vary when using a different software version or targeting a different device density or speed grade within the LatticeECP3 family.

Ordering Part Number

The Ordering Part Number (OPN) for the Divider IP core targeting LatticeECP3 devices is DIVIDE-E3-U.

LatticeECP2M Devices

Table A-2. Performance and Resource Utilization[1]

Numerator Data Type	Numerator Data Width	Denominator Data Type	Denominator Data Width	Output Latency	Registers	LUTs	Slices	f_{MAX}
Unsigned	20	Unsigned	10	20	828	311	446	305
Unsigned	24	Unsigned	12	12	586	409	409	210
Signed	32	Signed	32	32	3123	1459	1788	232

1. Performance and utilization data are generated targeting a LFE2M35E-7F672C device using Lattice Diamond 2.0 and Synplify Pro F-2012.03L software. Performance may vary when using a different software version or targeting a different device density or speed grade within the LatticeECP2M family.

Ordering Part Number

The Ordering Part Number (OPN) for Divider IP core targeting LatticeECP2M devices is DIVIDE-PM-U.

LatticeECP2 Devices

Table A-3. Performance and Resource Utilization[1]

Numerator Data Type	Numerator Data Width	Denominator Data Type	Denominator Data Width	Output Latency	Registers	LUTs	Slices	f_{MAX}
Unsigned	20	Unsigned	10	20	828	311	446	301
Unsigned	24	Unsigned	12	12	586	409	409	211
Signed	32	Signed	32	32	3123	1459	1788	217

1. Performance and utilization data are generated targeting a LFE2-35E-7F672C device using Lattice Diamond 2.0 and Synplify Pro F-2012.03L software. Performance may vary when using a different software version or targeting a different device density or speed grade within the LatticeECP2 family.

Ordering Part Number

The Ordering Part Number (OPN) for Divider IP core targeting LatticeECP2 devices is DIVIDE-P2-U.

LatticeXP2 Devices

Table A-4. Performance and Resource Utilization[1]

Numerator Data Type	Numerator Data Width	Denominator Data Type	Denominator Data Width	Output Latency	Registers	LUTs	Slices	f_{MAX}
Unsigned	20	Unsigned	10	20	828	311	446	281
Unsigned	24	Unsigned	12	12	586	409	409	211
Signed	32	Signed	32	32	3123	1459	1788	202

1. Performance and utilization data are generated targeting a LFE2M35E-7F672C device using Lattice Diamond 2.0 and Synplify Pro F-2012.03L software. Performance may vary when using a different software version or targeting a different device density or speed grade within the LatticeECP2M family.

Ordering Part Number

The Ordering Part Number (OPN) for Divider IP core targeting LatticeXP2 devices is DIVIDE-X2-U.

LifeElectronics
Живое партнерство

ООО «ЛайфЭлектроникс» **“LifeElectronics” LLC**

ИНН 7805602321 КПП 780501001 Р/С 40702810122510004610 ФАКБ "АБСОЛЮТ БАНК" (ЗАО) в г.Санкт-Петербурге К/С 30101810900000000703 БИК 044030703

Компания «Life Electronics» занимается поставками электронных компонентов импортного и отечественного производства от производителей и со складов крупных дистрибьюторов Европы, Америки и Азии.

С конца 2013 года компания активно расширяет линейку поставок компонентов по направлению коаксиальный кабель, кварцевые генераторы и конденсаторы (керамические, пленочные, электролитические), за счёт заключения дистрибьюторских договоров

Мы предлагаем:

- *Конкурентоспособные цены и скидки постоянным клиентам.*
- *Специальные условия для постоянных клиентов.*
- *Подбор аналогов.*
- *Поставку компонентов в любых объемах, удовлетворяющих вашим потребностям.*

- *Приемлемые сроки поставки, возможна ускоренная поставка.*
- *Доставку товара в любую точку России и стран СНГ.*
- *Комплексную поставку.*
- *Работу по проектам и поставку образцов.*
- *Формирование склада под заказчика.*

- *Сертификаты соответствия на поставляемую продукцию (по желанию клиента).*
- *Тестирование поставляемой продукции.*
- *Поставку компонентов, требующих военную и космическую приемку.*
- *Входной контроль качества.*
- *Наличие сертификата ISO.*

В составе нашей компании организован Конструкторский отдел, призванный помогать разработчикам, и инженерам.

Конструкторский отдел помогает осуществить:

- *Регистрацию проекта у производителя компонентов.*
- *Техническую поддержку проекта.*
- *Защиту от снятия компонента с производства.*
- *Оценку стоимости проекта по компонентам.*
- *Изготовление тестовой платы монтаж и пусконаладочные работы.*

Тел: +7 (812) 336 43 04 (многоканальный)
Email: org@lifeelectronics.ru

www.lifeelectronics.ru