Interleaver/De-interleaver IP Core User's Guide

December 2010
IPUG61_02.7

Table of Contents

Chapter 1. Introduction 4
Quick Facts 4
Features 8
Chapter 2. Functional Description 10
Block Diagrams 10
Convolutional Interleaver/de-interleaver 11
Rectangular Interleaver/De-interleaver 12
Latency 13
Signal Descriptions 13
Timing Diagrams 16
Chapter 3. Parameter Settings 17
Type and Mode Tab 18
Type 18
Mode 18
Convolutional Parameters Tab 19
Interleaver/Deinterleaver 19
Rectangular Parameters Tab 20
Interleaver/Deinterleaver 20
Block Size Type 20
Permutations 21
Convolutional Optional Pins Tab 21
Rectangular Optional Pins Tab 22
Chapter 4. IP Core Generation 24
Licensing the IP Core 24
Getting Started 24
IPexpress-Created Files and Top Level Directory Structure 26
Permutation Pattern Input File Format 28
Instantiating the Core 28
Running Functional Simulation 28
Synthesizing and Implementing the Core in a Top-Level Design 29
Hardware Evaluation 30
Enabling Hardware Evaluation in Diamond 30
Enabling Hardware Evaluation in ispLEVER 30
Updating/Regenerating the IP Core 30
Regenerating an IP Core in Diamond 30
Regenerating an IP Core in ispLEVER 31
Chapter 5. Support Resources 32
Lattice Technical Support 32
Online Forums 32
Telephone Support Hotline 32
E-mail Support 32
Local Support 32
Internet 32
References 32
LatticeEC/ECP 32
LatticeECP2/M 32
LatticeECP3 32
LatticeSC/M 32
LatticeXP 33

© 2010 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.

LatticeXP2 33
Revision History 33
Appendix A. Resource Utilization 34
LatticeECP3 FPGAs 34
Ordering Part Number 34
LatticeECP and LatticeEC FPGAs 34
Ordering Part Number 34
LatticeECP2 Devices 35
Ordering Part Number 35
LatticeECP2M Devices 35
Ordering Part Number 35
LatticeXP Devices 35
Ordering Part Number 35
LatticeXP2 Devices 36
Ordering Part Number 36
LatticeSC/M Devices 36
Ordering Part Number 36

Interleaving is a technique commonly used in communication systems to overcome correlated channel noise such as burst error or fading. The interleaver rearranges input data such that consecutive data are spaced apart. At the receiver end, the interleaved data is arranged back into the original sequence by the de-interleaver. As a result of interleaving, correlated noise introduced in the transmission channel appears to be statistically independent at the receiver and thus allows better error correction.

The Lattice Interleaver/de-interleaver IP core supports rectangular block type and convolutional architectures. Rectangular interleaving arranges the input data row-wise in a matrix. The interleaved data is obtained by reading the columns of the matrix. Convolutional interleaving feeds the input data to a number of branches, each of which has a shift register with pre-defined length. The output data is taken from the branch outputs. Lattice's Convolutional Interleaver/de-interleaver IP Cores are compliant with ATSC and DVB standards, while the Rectangular Interleaver/de-interleaver is compliant with IEEE 802.16a standard.

Quick Facts

Table 1-1 through Table 1-9 give quick facts about the Interleaver/de-interleaver IP core for LatticeEC™, LatticeECP™, LattceECP2™, LatticeECP2M™, LatticeECP3™, LatticeSC™, LatticeSCM™, LatticeXP™, and LatticeXP2™ devices, respectively.

Table 1-1. Interleaver/De-interleaver IP Core for LatticeEC Devices Quick Facts

		Interleaver/de-interleaver IP Configuration			
		Convolutional Interleaver	**Convolutional De-interleaver**	**Rectangular Interleaver**	**Rectangular De-interleaver**
Core Requirements	FPGA Families Supported	LatticeEC			
	Minimal Device Needed	LFEC3E			
Resource Utilization	Targeted Device	LFEC20E-5F672C			
	LUTs	200	200	100	100
	sysMEM EBRs	2	2	4	4
	Registers	200			
Design Tool Support	Lattice Implementation	Lattice Diamond™ 1.0 or ispLEVER® 8.1			
	Synthesis	Synopsys® Synplify™ Pro for Lattice D-2009.12L-1			
	Simulation	Aldec® Active-HDL™ 8.2 Lattice Edition			
		Mentor Graphics ModelSim™ SE 6.3F			

Table 1-2. Interleaver/De-interleaver IP Core for LatticeECP Devices Quick Facts

		Interleaver/de-interleaver IP Configuration			
		Convolutional Interleaver	**Convolutional De-interleaver**	**Rectangular Interleaver**	**Rectangular De-interleaver**
Core Requirements	FPGA Families Supported	LatticeECP			
	Minimal Device Needed	LFECP6E			
Resource Utilization	Targeted Device	LFECP20E-5F672C			
	LUTs	200	200	100	100
	sysMEM EBRs	2	2	4	4
	Registers	200			
Design Tool Support	Lattice Implementation	Lattice Diamond 1.0 or ispLEVER 8.1			
	Synthesis	Synopsys Synplify Pro for Lattice D-2009.12L-1			
	Simulation	Aldec Active-HDL 8.2 Lattice Edition			
		Mentor Graphics ModelSim SE 6.3F			

Table 1-3. Interleaver/De-interleaver IP Core for LatticeECP2 Devices Quick Facts

		Interleaver/de-interleaver IP Configuration			
		Convolutional Interleaver	**Convolutional De-interleaver**	**Rectangular Interleaver**	**Rectangular De-interleaver**
Core Requirements	FPGA Families Supported	LatticeECP2			
	Minimal Device Needed	LFE2-6E			
Resource Utilization	Targeted Device	LFE2-50E-7F672C			
	LUTs	200	200	100	100
	sysMEM EBRs	1	1	2	2
	Registers	200			
Design Tool Support	Lattice Implementation	Lattice Diamond 1.0 or ispLEVER 8.1			
	Synthesis	Synopsys Synplify Pro for Lattice D-2009.12L-1			
	Simulation	Aldec Active-HDL 8.2 Lattice Edition			
		Mentor Graphics ModelSim SE 6.3F			

Table 1-4. Interleaver/De-interleaver IP Core for LatticeECP2M Devices Quick Facts

		Interleaver/de-interleaver IP Configuration			
		Convolutional Interleaver	**Convolutional De-interleaver**	**Rectangular Interleaver**	**Rectangular De-interleaver**
Core Requirements	FPGA Families Supported	LatticeECP2M			
	Minimal Device Needed	LFE2M20E			
Resource Utilization	Targeted Device	LFECP2M35E-7F672C			
	LUTs	200	200	100	100
	sysMEM EBRs	1	1	2	2
	Registers	200			
Design Tool Support	Lattice Implementation	Lattice Diamond 1.0 or ispLEVER 8.1			
	Synthesis	Synopsys Synplify Pro for Lattice D-2009.12L-1			
	Simulation	Aldec Active-HDL 8.2 Lattice Edition			
		Mentor Graphics ModelSim SE 6.3F			

.

Table 1-5. Interleaver/De-interleaver IP Core for LatticeECP3 Devices Quick Facts

		Interleaver/De-interleaver IP Configuration			
		Convolutional Interleaver	**Convolutional De-interleaver**	**Rectangular Interleaver**	**Rectangular De-interleaver**
Core Requirements	FPGA Families Supported	Lattice ECP3			
	Minimal Device Needed	LFE3-17			
Resource Utilization	Targeted Device	LFSC3GA25E-7F900C			
	LUTs	150	150	100	100
	sysMEM EBRs	1	1	2	2
	Registers	200	200	150	150
Design Tool Support	Lattice Implementation	Lattice Diamond 1.0 or ispLEVER 8.1			
	Synthesis	Synopsys Synplify Pro for Lattice D-2009.12L-1			
	Simulation	Aldec Active-HDL 8.2 Lattice Edition			
		Mentor Graphics ModelSim SE 6.3F			

Table 1-6. Interleaver/De-interleaver IP Core for LatticeSC Devices Quick Facts

		Interleaver/de-interleaver IP Configuration			
		Convolutional Interleaver	**Convolutional De-interleaver**	**Rectangular Interleaver**	**Rectangular De-interleaver**
Core Requirements	FPGA Families Supported	LatticeSC			
	Minimal Device Needed	LFSC3GA15E			
Resource Utilization	Targeted Device	LFSC3GA25E-7F900C			
	LUTs	200	200	100	100
	sysMEM EBRs	1	1	2	2
	Registers	200			
Design Tool Support	Lattice Implementation	Lattice Diamond 1.0 or ispLEVER 8.1			
	Synthesis	Synopsys Synplify Pro for Lattice D-2009.12L-1			
	Simulation	Aldec Active-HDL 8.2 Lattice Edition			
		Mentor Graphics ModelSim SE 6.3F			

Table 1-7. Interleaver/De-interleaver IP Core for LatticeSCM Devices Quick Facts

		Interleaver/de-interleaver IP Configuration			
		Convolutional Interleaver	**Convolutional De-interleaver**	**Rectangular Interleaver**	**Rectangular De-interleaver**
Core Requirements	FPGA Families Supported	LatticeSCM			
	Minimal Device Needed	LFSCM3GA15EP1			
Resource Utilization	Targeted Device	LFSCM3GA25EP1-7F900C			
	LUTs	200	200	100	100
	sysMEM EBRs	1	1	2	2
	Registers	200			
Design Tool Support	Lattice Implementation	Lattice Diamond 1.0 or ispLEVER 8.1			
	Synthesis	Synopsys Synplify Pro for Lattice D-2009.12L-1			
	Simulation	Aldec Active-HDL 8.2 Lattice Edition			
		Mentor Graphics ModelSim SE 6.3F			

Table 1-8. Interleaver/De-interleaver IP Core for LatticeXP Devices Quick Facts

		Interleaver/de-interleaver IP Configuration			
		Convolutional Interleaver	**Convolutional De-interleaver**	**Rectangular Interleaver**	**Rectangular De-interleaver**
Core Requirements	FPGA Families Supported	LatticeXP			
	Minimal Device Needed	LFXP3C			
Resource Utilization	Targeted Device	LFXP20C-5F484C			
	LUTs	200	200	100	100
	sysMEM EBRs	2	2	4	4
	Registers	200			
Design Tool Support	Lattice Implementation	Lattice Diamond 1.0 or ispLEVER 8.1			
	Synthesis	Synopsys Synplify Pro for Lattice D-2009.12L-1			
	Simulation	Aldec Active-HDL 8.2 Lattice Edition			
		Mentor Graphics ModelSim SE 6.3F			

Table 1-9. Interleaver/De-interleaver IP Core for LatticeXP2 Devices Quick Facts

		Interleaver/de-interleaver IP Configuration			
		Convolutional Interleaver	**Convolutional De-interleaver**	**Rectangular Interleaver**	**Rectangular De-interleaver**
Core Requirements	FPGA Families Supported	Lattice XP2			
	Minimal Device Needed	LFXP2-5E			
Resource Utilization	Targeted Device	LFXP2-30E-7FT256CES			
	LUTs	200	200	100	100
	sysMEM EBRs	1	1	2	2
	Registers	200			
Design Tool Support	Lattice Implementation	Lattice Diamond 1.0 or ispLEVER 8.1			
	Synthesis	Synopsys Synplify Pro for Lattice D-2009.12L-1			
	Simulation	Aldec Active-HDL 8.2 Lattice Edition			
		Mentor Graphics ModelSim SE 6.3F			

Features

- High performance and area efficient symbol interleaver/de-interleaver
- Supports multiple standards, such as DVB, ATSC and IEEE 802.16
- Convolutional and rectangular block type architectures available
- Fully synchronous design using a single clock
- Symbol size from 1 to 256 bits
- Full handshake capability for input and output interfaces
- Rectangular block type features
  - Variable block size
  - Variable number of rows
  - Variable number of columns
  - Row permutations

  - Column permutations

- Convolutional type features
  - User-configurable number of branches
  - User-configurable branch length

The functionality of the Convolutional and Rectangular Interleaver/de-interleaver cores are described in this chapter. Figure 2-1 shows a convolutional interleaver/de-interleaver block diagram. Figure 2-2 shows a rectangular interleaver/de-interleaver block diagram.

Block Diagrams

Figure 2-1. Convolutional Interleaver/De-interleaver Block Diagram

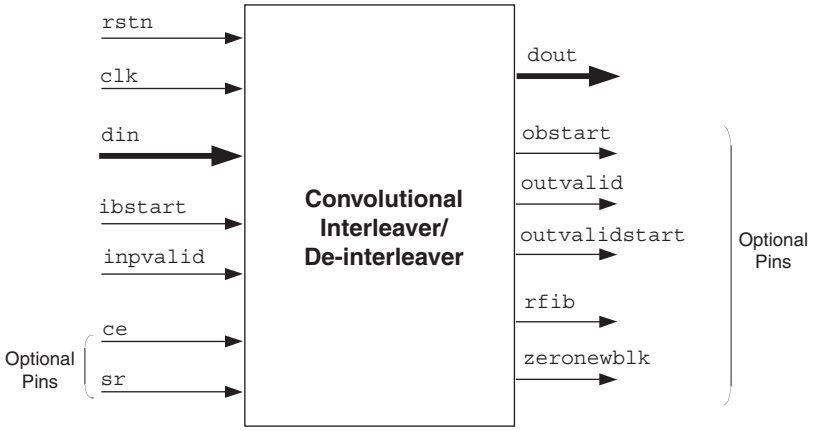


Figure 2-2. Rectangular Interleaver/De-interleaver Block Diagram

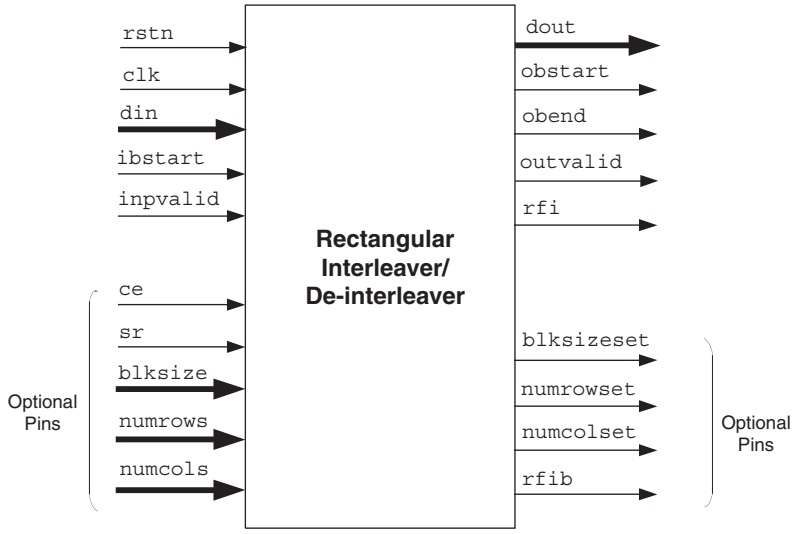
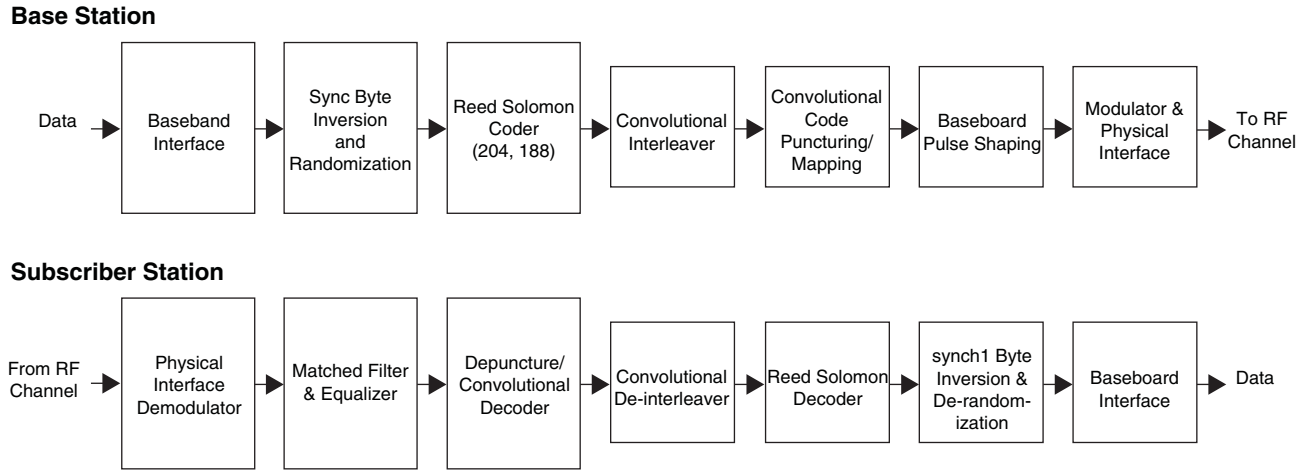


Figure 2-3 shows the role of interleaving and de-interleaving in a broadband wireless access system.

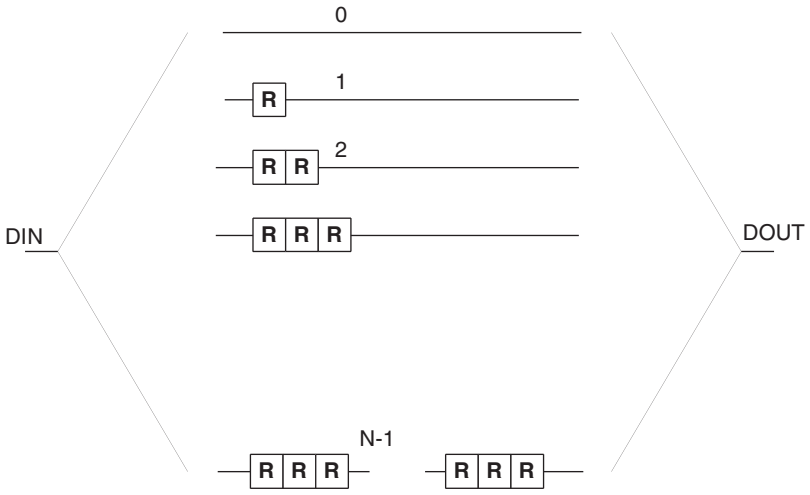
Figure 2-3. Broadband Wireless Access System



Convolutional Interleaver/de-interleaver

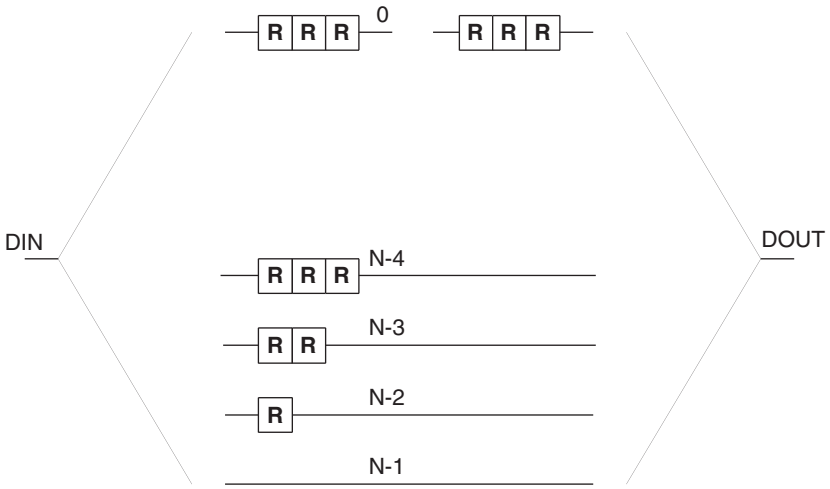
The convolutional interleaver consists of a bank of N branches. The value of N is set by the `number of branches` parameter. The interleaver pushes input data into the first shift register of a branch and reads the data from the output of the last register of the branch. The first branch has no delay, and every succeeding branch thereafter has a delay increase of R up to (N-1)*R, where R is the depth of the interleaver. The value of R is set by the `branch_length` parameter. The input and output are both connected to a commutator that synchronously rotates to each branch for each input symbol starting at branch zero. After switching to the final branch, the commutator rotates back to branch zero and repeats the process.

Figure 2-4. Convolutional Interleaver Functional Diagram



The de-interleaver is constructed similar to the interleaver, but its branches are arranged opposite to those of the interleaver. The first branch has a delay of (N-1)*R and the last branch has no delay.

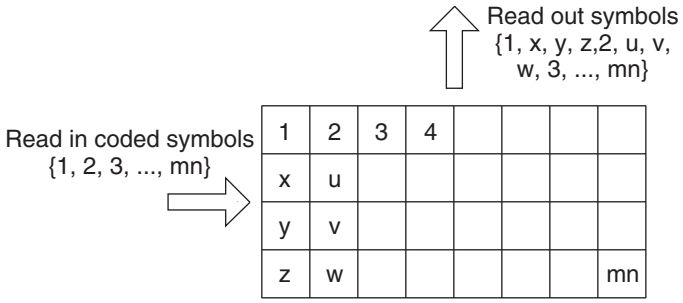
Figure 2-5. Convolutional De-Interleaver Functional Diagram



Rectangular Interleaver/De-interleaver

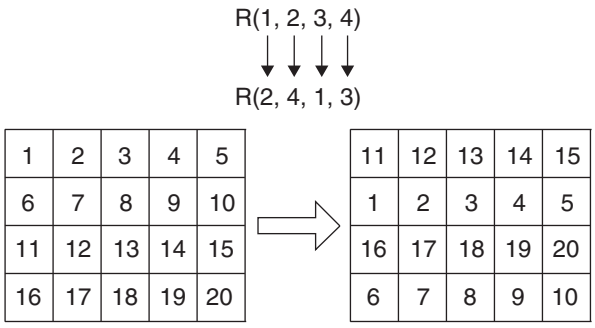
The rectangular interleaver receives the encoded data and formats it into a rectangular array of *m* rows and *n* columns. Typically, each row of the array composes a code word of length n. Symbols are read in row-wise and written out column-wise as indicated in Figure 2-6.

Figure 2-6. Rectangular Interleaver Matrix



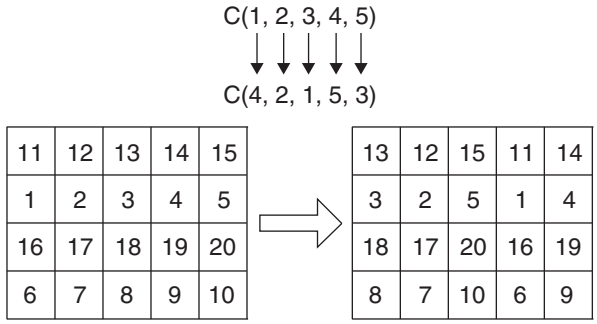
Inter-row and inter-column permutation formats may also be incorporated, but the matrix must be completely filled to do that. An example is shown in Figure 2-7 with a 4 x 5 array.

Figure 2-7. Row Permutation



The notation R (2,4,1,3) for row permutation means row 1 is changed to row 2, row 2 to row 4, row 3 to row 1 and row 4 to row 3.

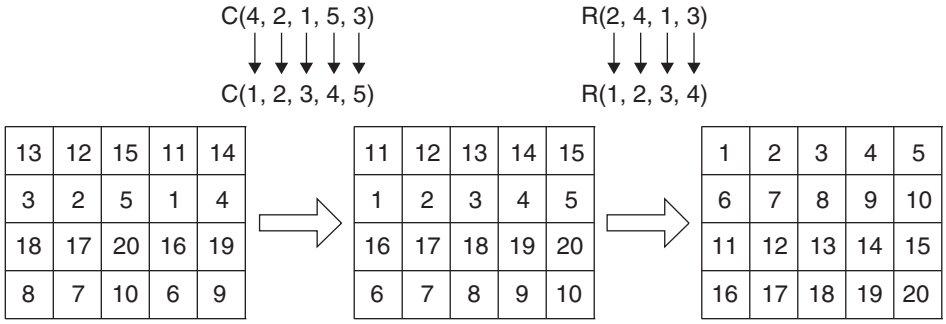
Figure 2-8. Column Permutation



The notation C(4,2,1,5,3) used to specify column permutation means column 1 is changed to column 4, column 2 to column 2, column 3 to column 1, column 4 to column 5 and column 5 to column 3.

Block de-interleaving is also a permutation formatting that is opposite of the permutation done for interleaving. This insures that the original data is correctly restored from the interleaver's output data. De-interleaving of the last generated array from Figure 2-8 is illustrated in Figure 2-9.

Figure 2-9. De-interleaving



In the last array the rows are read out left-to-right and top-to-bottom to give the output data {1, 2, 3,...,20}. The block interleaver reads in one block of symbols and then outputs the same block with symbols rearranged. No new inputs can be read until the previous block's interleaved symbols are output.

Latency

Latency for convolutional core, defined as the number of clock cycles between the sampling of the first input data and the availability of the first interleaved data at the output port, is six clock cycles.

Latency for rectangular core, defined as the number of clock cycles between the sampling of the last input data in the block and the availability of the first interleaved data at the output port, is five clock cycles.

Signal Descriptions

Table 2-1 and Table 2-2 list the input and output signals for the Interleaver/de-interleaver IP cores.

Table 2-1. Convolutional Interleaver/de-interleaver Signal Description

Port Name	I/O Type	Width	Signal Description
clk	Input	1	System Clock
rstn	Input	1	Asynchronous Reset. When this signal is asserted, all the core's registers are cleared. This is an active low signal.

Table 2-1. Convolutional Interleaver/de-interleaver Signal Description (Continued)

Port Name	I/O Type	Width	Signal Description
ibstart	Input	1	This signal must be asserted when the first data is being presented on input. It must be asserted only if rfib is high or the current output stream is aborted. ibstart also initializes the commutator arms to branch zero. The signal inpvalid must be high for this signal to be effective.
inpvalid	Input	1	Qualifies input data din to be a valid new data symbol.
din	Input	1-256	Input Data. The input symbols present on this port are interleaved/de-interleaved.
dout	Output	1-256	Output Data. The interleaved/de-interleaved symbols are output on this port.
Optional Signals			
ce	Input	1	Clock Enable. This signal has the highest priority after rstn. The core operation freezes if this signal is low. Use of this port increases the size of the core. It should only be selected if necessary.
sr	Input	1	Synchronous Reset. When this signal is high, all the registers in the core are cleared synchronously. Use of this port increases the size of the core. It should only be selected if necessary.
obstart	Output	1	This signal indicates that the first interleaved data is present on the dout output port after an output latency of few cycles.
outvalid	Output	1	This signal indicates that a valid data is present on the dout output port. However, some of the output data may be from the initial values stored in the branch registers, as it takes a few cycles for the branch registers to get filled up with input data from the din port.
outvalidstart	Output	1	This signal indicates that a valid data is present on the dout output port and it corresponds to a valid data from the din port.
rfib	Output	1	This signal is asserted for one cycle as soon as the interleaver is ready to accept a new data. After rfib goes high, a new data stream and ibstart can be applied at the input.
zeronewblk	Output	1	This signal is available in the convolutional interleaver mode only. This signal is similar to rfib but with the following differences. The signal zeronewblk is only asserted when the input data source has filled all the storage elements in the interleaver. It indicates that the input source can again start feeding the data. This signal can be used for Block Boundary Synchronization. This signal is also asserted high at synchronous and asynchronous resets.

Table 2-2. Rectangular Interleaver /De-interleaver Signal Description

Port Name	I/O Type	Width	Signal Description
clk	Input	1	System Clock
rstn	Input	1	Asynchronous Reset. This is an active low signal. When this signal is asserted, all the registers in the core are cleared.
ibstart	Input	1	This signal signifies the first data on input din. It must be asserted only after rfib goes high. Otherwise the core will terminate the processing of the current block and start processing the new block. The signals inpvalid and rfi must be high for ibstart to be effective.
inpvalid	Input	1	Qualifies the input data din as a valid data symbol. If rfi is low, inpvalid will be ignored except for the case when the last input data symbol is placed at the din port.
din	Input	1-256	Input Data. The input symbols present on this port are interleaved/de-interleaved.
dout	Output	1-256	Output Data. The interleaved/de-interleaved symbols are output on this port.
obstart	Output	1	Assertion of this signal indicates that first data is present on the dout output port.

Table 2-2. Rectangular Interleaver /De-interleaver Signal Description (Continued)

Port Name	I/O Type	Width	Signal Description
outvalid	Output	1	Assertion of this signal indicates that valid data is present on the dout output port.
obend	Output	1	Assertion of this signal indicates that the last output data is present on the dout output port.
rfi	Output	1	Ready for Input. This signal is asserted when the core is ready to accept data. It is de-asserted one clock cycle before the last input data is read.
Optional Signals			
blksize	Input	4-16	Block size value is supplied through this port for variable block size configurations. The core reads the value on this port when ibstart is high.
numcols	Input	2-9	Number of columns in the current block. The number of columns value is provided through this port for a variable number of columns configuration. The core reads the value on this port when ibstart is high.
numrows	Input	3-9	Number of rows in current block. The number of rows value is provided through this port for a variable number of rows configuration. The core reads the value on this port when ibstart is high.
ce	Input	1	Clock Enable. This signal has the highest priority after rstn. The core operation freezes if this signal is low. This port increases the size of the core and should only be selected if necessary
sr	Input	1	Synchronous Reset. When this signal is high, all the registers in the core are cleared synchronously. This port increases the size of the core and should only be selected if necessary.
numcolset	Output	1	This signal is high if the number of columns value provided on the numcols port is valid. If the value on numcols is not valid, numcolset is low. It is active three clock cycles after the number of columns value is sampled. If invalid, the signal rfi (and the optional rfib signal, if selected) will be asserted in the next clock cycle.
numrowset	Output	1	This signal is high if the number of rows value provided on the numrows port is valid. If the value on numrows is not valid, numrowset is low. It is active three clock cycles after the number of rows value is sampled. If invalid, the signal rfi (and the optional rfib signal, if selected) will be asserted in the next clock cycle.
blksizeset	Output	1	This signal is high if the block size value provided on blksize port is valid. If the block size value on blksize is not valid, blksizeset is low. It is active three clock cycles after the block size value is sampled. If invalid, the signal rfi (and the optional rfib signal, if selected) will be asserted in the next clock cycle.
rfib	Output	1	This signal is asserted for one cycle as soon as the interleaver is ready to accept a new data block. After rfib goes high, a new data block and ibstart can be applied at the input.

Timing Diagrams

Figure 2-10. Convolutional Interleaver Interface Timing

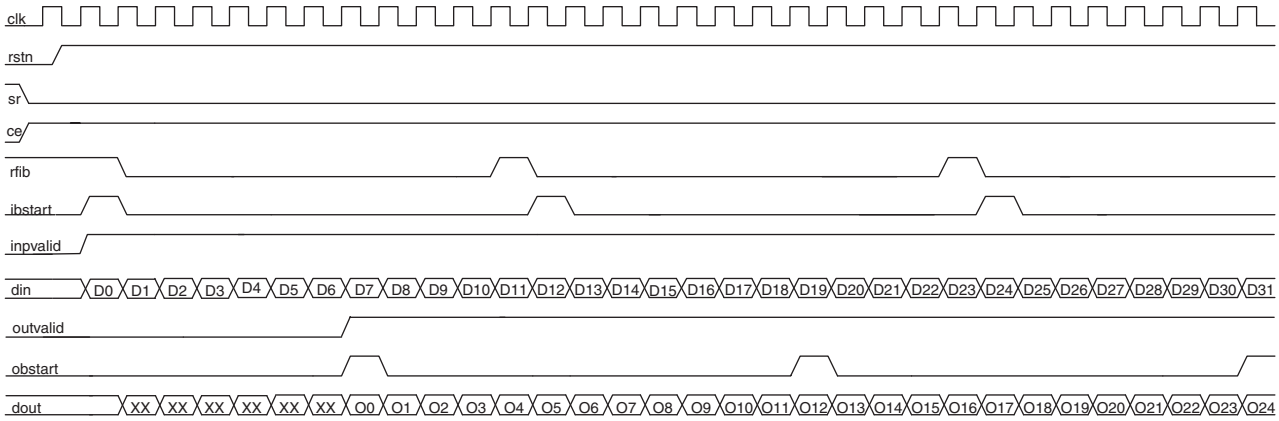
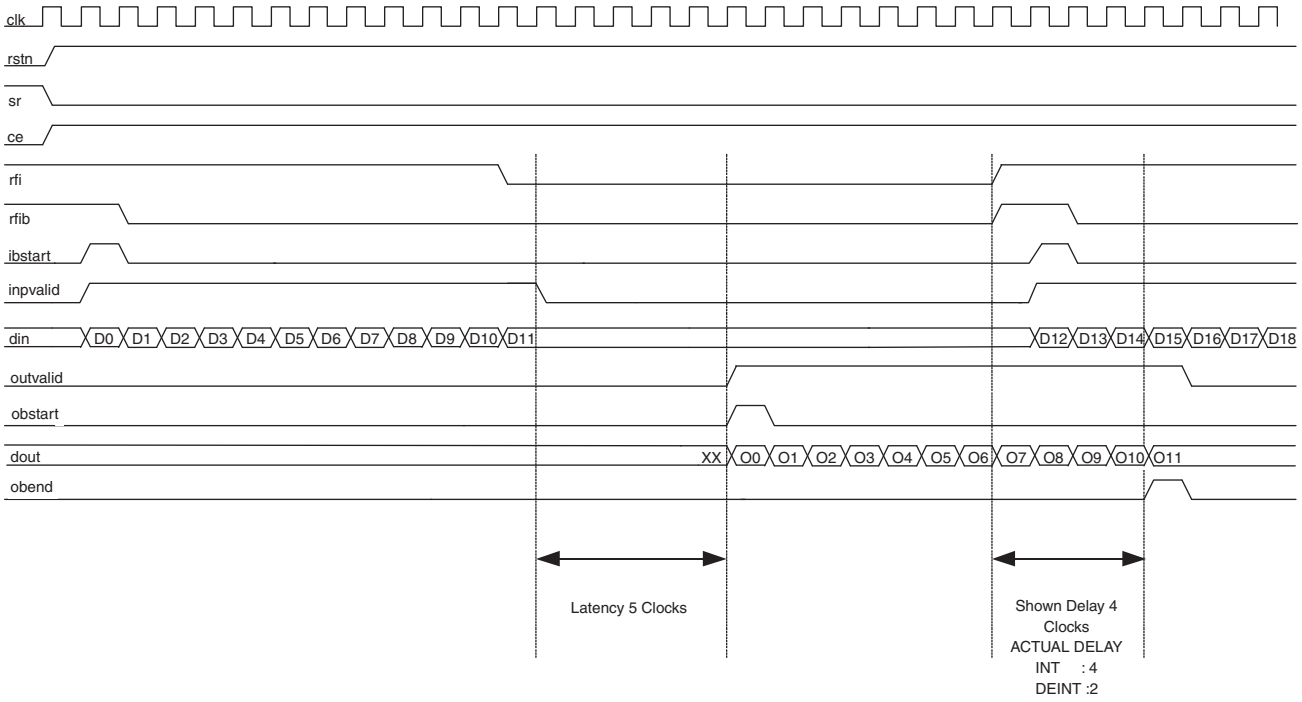


Figure 2-11. Rectangular Interleaver Interface Timing



Parameter Settings

The IPexpress™ tool is used to create IP and architectural modules in the Diamond or ispLEVER software. Refer to "IP Core Generation" on page 24 for a description of how to generate the IP. The Interleaver/de-interleaver IP core can be customized to suit a specific application by adjusting parameters prior to core generation. Since the values of some parameters affect the size of the resultant core, the maximum value for these parameters may be limited by the size of the target device.

Table 3-1 and Table 3-2 provide the list of user configurable parameters for the Interleaver/de-interleaver IP core. The parameter settings are specified using the Interleaver/de-interleaver IP core Configuration GUI in IPexpress.

Table 3-1. Convolutional Interleaver/De-interleaver Parameter Descriptions

Parameter	Range/Options	Default
Type	Convolutional/Rectangular	
Mode	Interleaver/de-interleaver	Interleaver
Symbol Width	1-256	8
Number of Branches	ECP/EC: 7-256	12
Branch Length	1-780	17

Table 3-2. Rectangular Interleaver De-interleaver Parameter Descriptions

Parameter	Range/Options	Default
Mode	Interleaver or De-interleaver	Interleaver
Symbol Width	1-256	8
Block Size Type	Constant Row*Col Variable	Constant
Block Size	ECP/EC: 9-65536	4096
Number of Columns	2-256	256
Number of Rows	4-256	16
Row Permutations	Yes or No	No
Column Permutations	Yes or No	No
Row Type	Constant/Variable	Constant
Column Type	Constant/Variable	Constant
Row Width	3-9	5
Column Width	2-9	9
Block Width	ECP/EC: 4-16	13

Type and Mode Tab

Figure 3-1 shows the Type and Mode tab of the Interleaver/de-interleaver IP core configuration dialog box. The user can select Convolutional or Rectangular type and Interleaver or De-interleaver mode.

- If Convolutional type is selected, when the user clicks **Next**, the Convolutional tab displays.
- If Rectangular type is selected, when the user clicks **Next**, the Rectangular tab displays.

Figure 3-1. Type and Mode Tab

Type

This parameter allows the user to choose either Convolutional or Rectangular type.

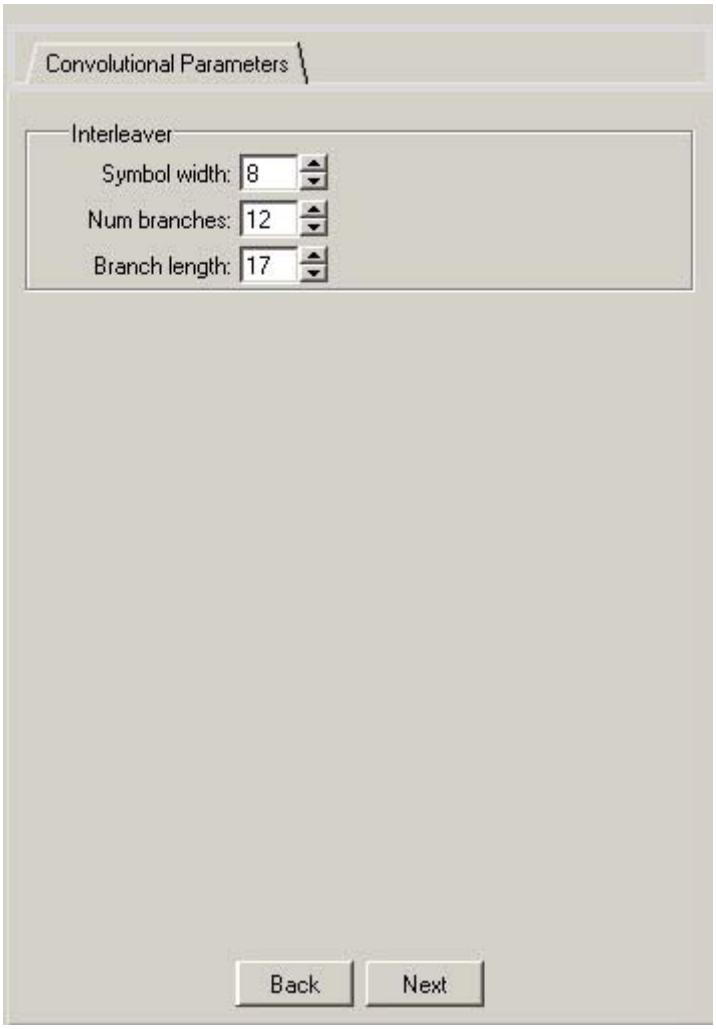
Mode

This parameter determines allows the user to choose either Interleaver or De-interleaver mode.

Convolutional Parameters Tab

If Convolutional was selected in the Type and Mode tab, the Convolutional Parameters tab is displayed when the **Next** button is clicked. Figure 3-2 shows the Convolutional Parameters tab of the Interleaver/de-interleaver IP core configuration dialog box.

Figure 3-2. Convolutional Parameters Tab



Interleaver/Deinterleaver

The following options are available in the Convolutional Parameters tab for both Interleaver or De-interleaver modes.

Symbol Width
This options sets the data width of the input symbols.

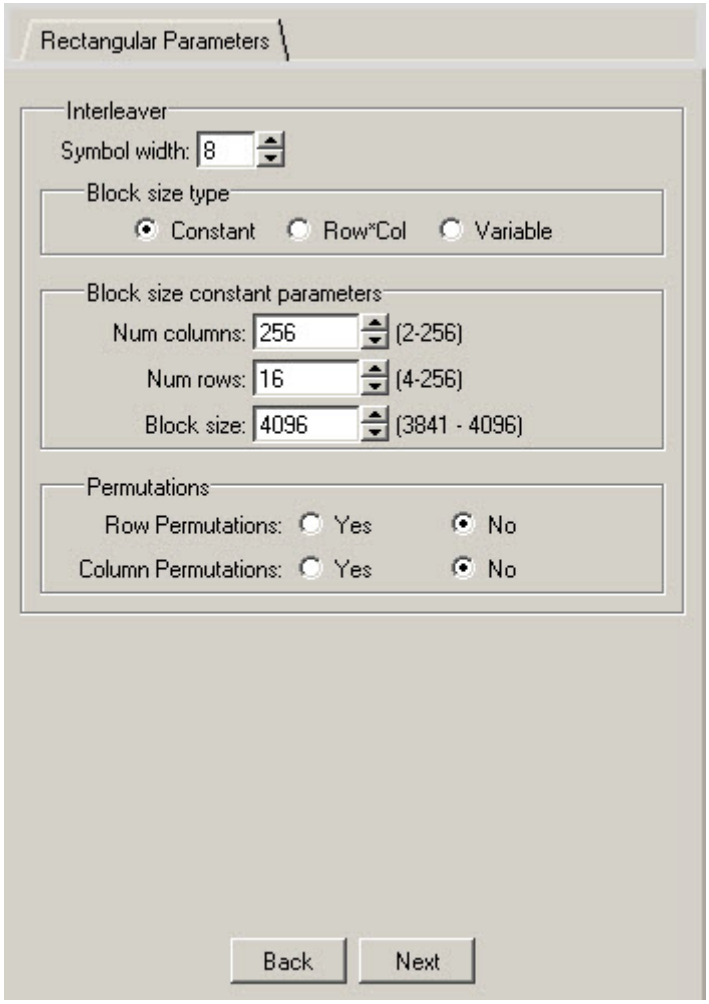
Num Branches
This options sets the number of branches of the commutator arm.

Branch Lengths
This options sets the difference in storage elements for consecutive branches.

Rectangular Parameters Tab

If Rectangular was selected in the Type and Mode tab, the Rectangular Parameters tab is displayed when the **Next** button is clicked. Figure 3-3 shows the Rectangular Parameters tab of the Interleaver/de-interleaver IP core configuration dialog box.

Figure 3-3. Convolutional Parameters Tab



Interleaver/Deinterleaver

The following options are available in the Rectangular Parameters tab for both Interleaver or De-interleaver modes.

Symbol Width

This parameter specifies the data width of the input symbols.

Block Size Type

This parameter specifies the block size input to the core. There are three options for giving block size value.

- Constant: In this configuration the block size value is constant and assigned a value during core configuration.
- Row*Col: In this configuration, block size value is computed form the Row and Column values.
- Variable: In this configuration, block size value is given through an input port.

Block Size Constant Parameters

Block Size

This parameter is only available if Block Size Type = Constant. Block size input to the core: The block size value should be such that the last symbol in the block should come on the last row. Therefore block size value should be greater than (Col*(Row-1)) and less than or equal to (Col*Row). When block size type is constant, number of rows and number of columns is also constant. Inter-row permutations and inter-column permutations are supported when block size value is Col*Row. Col is number of columns and Row is number of rows.

Number of Columns

This parameter specifies the number of columns used in the core. Inter-row permutations and inter-column permutations are supported when block size value is Col*Row.

Number of Rows

This parameter specifies the number of rows used in the core. Inter-row permutations and inter-column permutations are supported when block size value is Col*Row.

Permutations

Row Permutations

Rows can be permutated if required.Inter-row permutations and inter-column permutations are supported when block size value is Col*Row.

Column Permutations

Columns can be permutated if required. Inter-row permutations and inter-column permutations are supported when block size value is Col*Row.

Row Type

There are two options for giving number of rows.

- Constant: In this configuration, number of rows value is constant and assigned a value during core configuration.
- Variable: In this configuration, number of rows value is given through an input port. Row permutations are not supported when Row type is variable.

Column Type

There are two options for giving number of columns.

- Constant: In this configuration, number of columns value is constant and assigned a value during core configuration.
- Variable: In this configuration, number of columns value is given through the input port. Column permutations are not supported when Column type is variable.

Row Width

This parameter is only available if Row Type = Variable. Width for number of rows port (numrows).

Column Width

This parameter is only available if Column Type = Variable. Width for number of columns port (numcols)

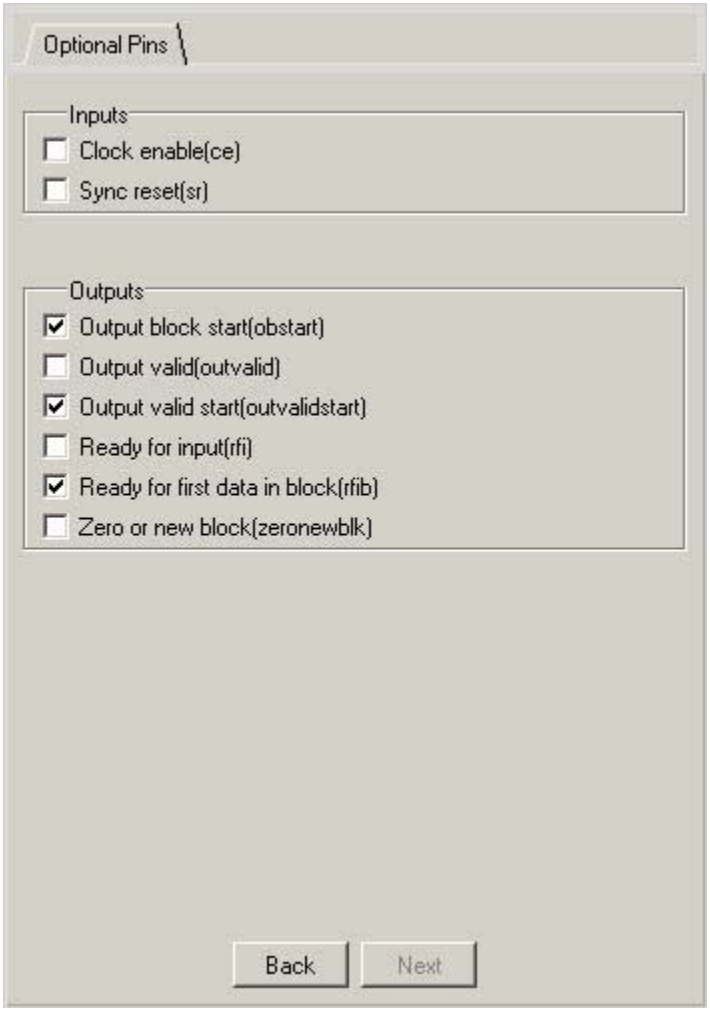
Block Width

This parameter is only available if Block Size Type = Variable. Block Width required for the block size value. For variable block size type, either rows or columns or both can be selected to be variable. Both rows and columns cannot be constant. Row and column permutations are not supported when block size type is variable.

Convolutional Optional Pins Tab

If Convolutional was selected in the Type and Mode tab, the Convolutional Optional Pins tab is displayed when the **Next** button is clicked twice. Figure 3-4 shows the Convolutional Parameters tab of the Interleaver/de-interleaver IP core configuration dialog box.

Figure 3-4. Convolutional Optional Pins Tab

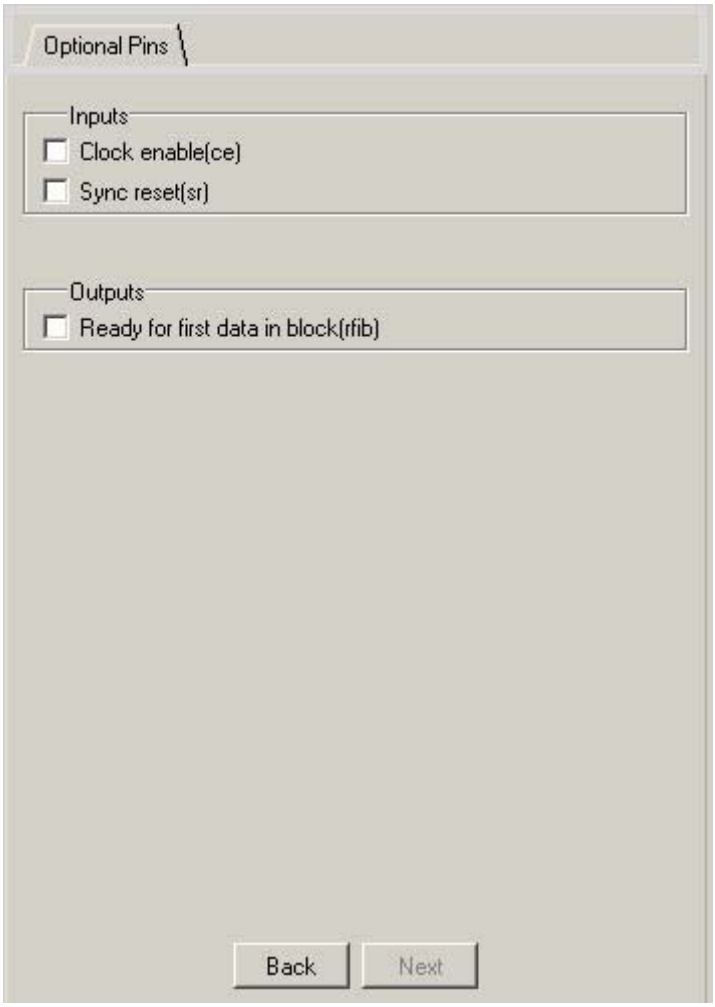


In this tab, all ports are optional in convolutional mode, users can select or de-select them to interface with other modules.

Rectangular Optional Pins Tab

If Rectangular was selected in the Type and Mode tab, the Rectangular Optional Pins tab is displayed when the **Next** button is clicked twice. Figure 3-5 shows the Convolutional Parameters tab of the Interleaver/de-interleaver IP core configuration dialog box.

Figure 3-5. Rectangular Optional Pins Tab



In this tab, all ports are optional in rectangular mode, users can select or de-select them to interface with other modules.

IP Core Generation

This chapter provides information on how to generate the Lattice Interleaver/de-interleaver IP core using the Diamond or ispLEVER software 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 Interleaver/de-interleaver IP core in a complete, top-level design. Instructions on how to obtain licenses for Lattice IP cores are given at:

http://www.latticesemi.com/products/intellectualproperty/aboutip/isplevercoreonlinepurchas.cfm

Users may download and generate the Interleaver/de-interleaver IP core and fully evaluate the core through functional simulation and implementation (synthesis, map, place and route) without an IP license. The Interleaver/de-interleaver IP core also 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 "To use this project file in Diamond:" on page 29 for further details. However, a license is required to enable timing simulation, to open the design in the Diamond or ispLEVER EPIC tool, and to generate bitstreams that do not include the hardware evaluation timeout limitation.

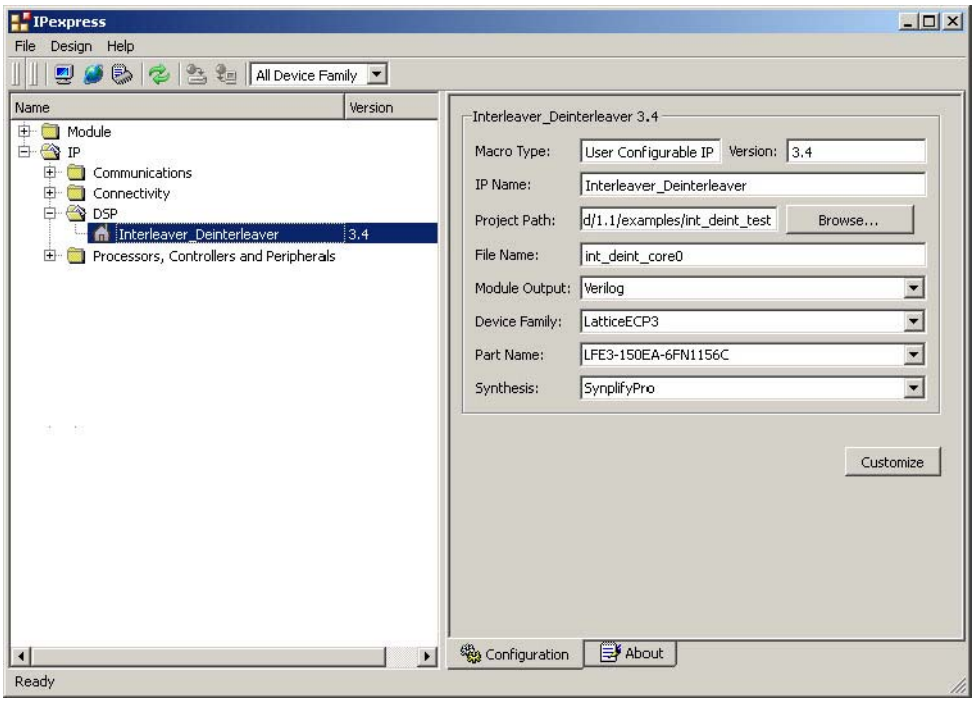
Getting Started

The Interleaver/de-interleaver 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 customer-specified directory. After the IP core has been installed, the IP core will be available in the IPexpress GUI dialog box shown in Figure 4-1.

The IPexpress tool GUI dialog box for the Interleaver/de-interleaver IP core is 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 loaded.
- **File Name** – "username" designation given to the generated IP core and corresponding folders and files.
- **(Diamond) Module Output** – Verilog or VHDL.
- **(ispLEVER) Design Entry Type** – Verilog HDL or VHDL
- **Device Family** – Device family to which IP is to be targeted (e.g. LatticeECP2M, LatticeECP3, etc.). Only families that support the particular IP core are listed.
- **Part Name** – Specific targeted part within the selected device family.

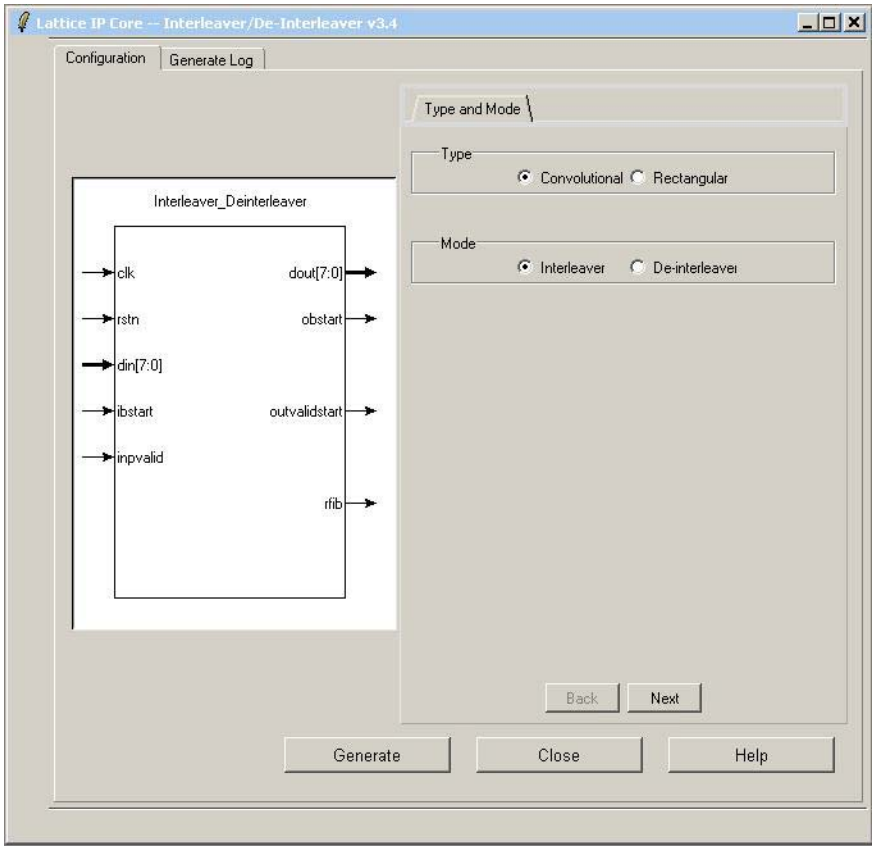
Figure 4-1. IPexpress Dialog Box (Diamond Version)



Note that if the IPexpress tool is called from within an existing project, Project Path, Module Output (Design Entry in ispLEVER), 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 user clicks the **Customize** button in the IPexpress tool dialog box to display the Interleaver/de-interleaver 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. Refer to "Parameter Settings" on page 17 for more information on the Interleaver/de-interleaver IP core parameter settings.

Figure 4-2. Configuration GUI (Diamond Version)



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.

Figure 4-3. LatticeECP3 Interleaver/de-interleaver IP Core Directory Structure

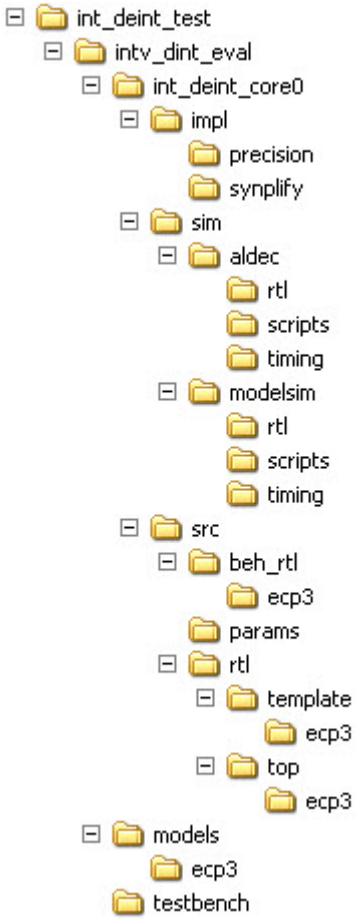


Table 4-1 provides a list of key files and directories created by the IPexpress tool and how they are used. The IPexpress tool creates several files that are used throughout the design cycle. The names of most of the created files are customized to the user's module name specified in the IPexpress tool.

Table 4-1. File List

File	Description
<username>_inst.v	This file provides an instance template for the IP.
<username>.v	This file provides the Interleaver/de-interleaver core for simulation.
<username>_beh.v	This file provides a behavioral simulation model for the Interleaver/de-interleaver core.
<username>_bb.v	This file provides the synthesis black box for the user's synthesis.
<username>.ngo	The ngo files provide the synthesized IP core.
<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 or Module after it is generated from the IPexpress tool (Diamond version only). 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/Module generation GUI when an IP/Module is being re-generated.
<username>_top.[v,vhd]	This file provides a module which instantiates the Interleaver/de-interleaver core. This file can be easily modified for the user's instance of the Interleaver/de-interleaver core. This file is located in the `intv_dint_eval/<username>_/src/rtl/top` directory.

Table 4-1. File List (Continued)

File	Description
<username>_generate.tcl	Created when GUI “Generate” button is pushed, invokes generation, may be run from command line.
<username>_generate.log	IPexpress scripts log file.
<username>_gen.log	IPexpress IP generation log file

Permutation Pattern Input File Format

For the rectangular core, inter-row and inter-column permutations are supported. The permutation patterns are given through a configuration file. This file should have “.cfg” extension. The Interleaver/de-interleaver GUI requires this file during core configuration. The following example explains the contents of the configuration file:

radix = 10;

row_permute_array = 5,3,4,1,2,0;
col_permute_array = 4,2,3,0,1;

The first line in the configuration file indicates the radix of the values in the row and column permutation arrays. The values can be entered in binary, decimal or hexadecimal formats. For binary, decimal and hexadecimal numbers, radix values of 2, 10 and 16 respectively must be entered.

In the above example, number of rows are 6 and number of columns are 5. Row and column permutations are entirely independent of each other. Therefore, any of these permutation combinations can be selected: row only, column only, both or none.

For the above example, inter-row permutations are done as follows:

row number 5 is placed at row number 0.
row number 3 is placed at row number 1.
row number 4 is placed at row number 2.
row number 1 is placed at row number 3.
row number 2 is placed at row number 4.
row number 0 is placed at row number 5.

For the above example, inter-column permutations are done as follows:

column number 4 is placed at column number 0.
column number 2 is placed at column number 1.
column number 3 is placed at column number 2.
column number 0 is placed at column number 3.
column number 1 is placed at column number 4.

Instantiating the Core

The generated Interleaver/de-interleaver IP core package includes black-box (<username>_bb.v) and instance (<username>_inst.v) templates that can be used to instantiate the 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>\intv_dint_eval\<username>\src\rtl\top. Users may also use this top-level reference as the starting template for the top-level for their complete design.

Running Functional Simulation

Simulation support for the Interleaver/de-interleaver IP core is provided for Aldec Active-HDL (Verilog and VHDL) simulator, Mentor Graphics ModelSim simulator. The functional simulation includes a configuration-specific behavioral model of the Interleaver/de-interleaver 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 scripts supporting ModelSim evaluation simulation is provided in \<*project_dir*>\intv_dint_eval\<*username*>\sim\modelsim\scripts. The simulation script supporting Aldec evaluation simulation is provided in \<*project_dir*>\intv_dint_eval\<*username*>\sim\aldec\scripts. Both Modelsim and Aldec simulation is supported via test bench files provided in \<*project_dir*>\intv_dint_eval\testbench. Models required for simulation are provided in the corresponding \models folder. Users may run the Aldec evaluation simulation by doing the following:

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

Users may run the Modelsim evaluation simulation by doing the following:

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

Note: *When the simulation completes, a pop-up window will appear asking “Are you sure you want to finish?” Answer “No” to analyze the results (answering “Yes” closes ModelSim).*

Synthesizing and Implementing the Core in a Top-Level Design

Synthesis support for the Interleaver/de-interleaver IP core is provided for Mentor Graphics Precision or Synopsys Synplify. The Interleaver/de-interleaver IP core itself is synthesized and is provided in NGO format when the core is generated in IPexpress. Users may synthesize the core in their own top-level design by instantiating the core in their top-level as described previously and then synthesizing the entire design with either Synplify or Precision RTL Synthesis. The following text describes the evaluation implementation flow for Windows platforms. The flow for Linux and UNIX platforms is described in the Readme file included with the IP core. The top-level files <username>_top.v are provided in \<*project_dir*>\intv_dint_eval\<*username*>\src\rtl\top. Push-button implementation of the reference design is supported via Diamond or ispLEVER project files, <username>.syn, located in the following directory: \<*project_dir*>\intv_dint_eval\<*username*>\impl\(synplify or precision).

To use this project file in Diamond:

1. Choose **File > Open > Project**.
2. Browse to \<*project_dir*>\intv_dint_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.

To use this project file in ispLEVER:

1. Choose **File > Open Project.**

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

Hardware Evaluation

The Interleaver/de-interleaver IP core supports 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 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.

Enabling Hardware Evaluation in Diamond

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

Enabling Hardware Evaluation in ispLEVER

In the Processes for Current Source pane, right-click the **Build Database** process and choose **Properties** from the dropdown menu. The hardware evaluation capability may be enabled/disabled in the Properties dialog box. It is enabled by default.

Updating/Regenerating the IP Core

By regenerating an IP core with the IPexpress tool, you 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 T**arget** 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 guides. IP 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** (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 holds 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 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.

Regenerating an IP Core in ispLEVER

To regenerate an IP core in ispLEVER:

1. In the IPexpress tool, choose **Tools > Regenerate IP/Module**.
2. In the Select a Parameter File dialog box, choose the Lattice Parameter Configuration (.lpc) file of the IP core you wish to regenerate, and click **Open**.
3. The Select Target Core Version, Design Entry, and Device dialog box shows the current settings for the IP core in the Source Value box. Make your new settings in the Target Value box.
4. If you want to generate a new set of files in a new location, set the location in the LPC Target File box. The base of the .lpc file name will be the base of all the new file names. The LPC Target File must end with an .lpc extension.
5. Click **Next**. The IP core's dialog box opens showing the current option settings.
6. In the dialog box, choose desired options. To get information about the options, click **Help**. Also, check the About tab in the IPexpress tool for links to technical notes and user guides. The IP core might come with additional information. As the options change, the schematic diagram of the IP core changes to show the I/O and the device resources the IP core will need.
7. Click **Generate**.
8. Click the **Generate Log** tab to check for warnings and error messages.

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 (http://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

LatticeEC/ECP

- HB1000, *LatticeEC/ECP Family Handbook*

LatticeECP2/M

- HB1003, *LatticeECP2/M Family Handbook*

LatticeECP3

- HB1009, *LatticeECP3 Family Handbook*

LatticeSC/M

- DS1004, *LatticeSC/M Family Data Sheet*

LatticeXP

- HB1001, *LatticeXP Family Handbook*

LatticeXP2

- DS1009, *Lattice XP2 Datasheet*

Revision History

Date	Document Version	IP Version	Change Summary
September 2006	02.1	3.0	Added LatticeECP2, LatticeSC, and LatticeXP FPGA family support; added IPexpress user-configurable support.
December 2006	02.2	3.1	Updated appendix tables.
			Added LatticeECP2M FPGA family support.
June 2007	02.3	3.2	Updated appendices. Added support for LatticeXP2 FPGA family.
August 2007	02.4	3.2	Updated Convolutional Interleaver/de-interleaver Signal Description table.
			Updated Rectangular Interleaver /De-interleaver Signal Description table.
			Updated Convolutional Interleaver/de-interleaver Block Diagram.
			Updated Rectangular Interleaver De-interleaver Block Diagram.
			Updated Convolutional Interleaver Interface Timing Diagram.
			Updated Rectangular Interleaver Interface Timing Diagram.
August 2008	02.5	3.3	Added Quick Facts table.
			Updated appendices.
July 2010	2.6	3.3	Divided document into chapters. Added table of contents.
			Added Quick Facts tables in Chapter 1, “Introduction.”
			Added new content in Chapter 4, “IP Core Generation.”
December 2010	2.7	3.4	Added support for Diamond software throughout.
			Added support for LatticeECP3 family.

Appendix A:

Resource Utilization

This appendix gives resource utilization information for Lattice FPGAs using the Interleaver/de-interleaver IP core.

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

LatticeECP3 FPGAs

Table A-1. Performance and Utilization[1]

IPexpress User-Configurable Mode	Slices	LUTs	Registers	I/Os	sysMEM™ EBRs	f_{MAX} (MHz)
Convolutional Interleaver DVB	86	122	159	23	1	336
Convolutional De-Interleaver DVB	89	133	164	23	1	340
Rectangular Interleaver 802.16	54	64	101	24	2	340
Rectangular De-Interleaver 802.16	72	82	132	24	2	338

1. Performance and utilization data are generated using an LFE3-95E-8FN672CES device with Lattice's Diamond 1.0 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 Interleaver/de-interleaver targeting LatticeECP3 devices is INTV-DINT-E3-U3.

LatticeECP and LatticeEC FPGAs

Table A-2. Performance and Utilization[1]

IPexpress User-Configurable Mode	Slices	LUTs	Registers	I/Os	sysMEM EBRs	f_{MAX} (MHz)
Convolutional Interleaver DVB	92	128	159	23	2	267
Convolutional De-Interleaver DVB	95	140	164	23	2	235
Rectangular Interleaver 802.16	61	81	101	24	4	258
Rectangular De-Interleaver 802.16	81	94	132	24	4	230

1. Performance and utilization data are generated using an LFECP20E-5F672C device with Lattice's Diamond 1.0 software. Performance may vary when using a different software version or targeting a different device density or speed grade within the LatticeECP/EC family.

Ordering Part Number

The Ordering Part Number (OPN) for the Interleaver/de-interleaver targeting LatticeECP/EC devices is INTV-DINT-E2-U3.

LatticeECP2 Devices

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

IPexpress User-Configurable Mode	Slices	LUTs	Registers	I/Os	sysMEM EBRs	f_{MAX} (MHz)
Convolutional Interleaver DVB	86	121	159	23	1	357
Convolutional De-Interleaver DVB	88	132	164	23	1	370
Rectangular Interleaver 802.16	52	75	101	24	2	370
Rectangular De-Interleaver 802.16	70	103	132	24	2	370

1. Performance and utilization data are generated using an LFE2-50E-7F672C device with Lattice's Diamond 1.0 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 the Interleaver/de-interleaver targeting LatticeECP2 devices is INTV-DINT-P2-U3.

LatticeECP2M Devices

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

IPexpress User-Configurable Mode	Slices	LUTs	Registers	I/Os	sysMEM EBRs	f_{MAX} (MHz)
Convolutional Interleaver DVB	86	121	159	23	1	329
Convolutional De-Interleaver DVB	88	132	164	23	1	370
Rectangular Interleaver 802.16	52	75	101	24	2	353
Rectangular De-Interleaver 802.16	70	103	132	24	2	370

1. Performance and utilization data are generated using an LFE2M35E-7F484C device with Lattice's. Diamond 1.0 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 the Interleaver/de-interleaver targeting LatticeECP2M devices is INTV-DINT-PM-U3.

LatticeXP Devices

Table A-5. Performance and Resource Utilization[1]

IPexpress User-Configurable Mode	Slices	LUTs	Registers	I/Os	sysMEM EBRs	f_{MAX} (MHz)
Convolutional Interleaver DVB	92	128	159	23	2	211
Convolutional De-Interleaver DVB	95	140	164	23	1	194
Rectangular Interleaver 802.16	61	81	101	24	4	191
Rectangular De-Interleaver 802.16	81	94	132	24	4	233

1. Performance and utilization data are generated using an LFXP20E-5F484C device with Lattice's Diamond 1.0 software. Performance may vary when using a different software version or targeting a different device density or speed grade within the LatticeXP family.

Ordering Part Number

The Ordering Part Number (OPN) for the Interleaver/de-interleaver targeting LatticeXP devices is INTV-DINT-XM-U3.

LatticeXP2 Devices

Table A-6. Performance and Resource Utilization[1]

IPexpress User-Configurable Mode	Slices	LUTs	Registers	I/Os	sysMEM EBRs	f_{MAX} (MHz)
Convolutional Interleaver DVB	86	121	159	23	1	314
Convolutional De-Interleaver DVB	88	132	164	23	1	299
Rectangular Interleaver 802.16	52	75	101	24	2	314
Rectangular De-Interleaver 802.16	70	103	132	24	2	314

1. Performance and utilization data are generated using an LFXP2-30E-7F484C device with Lattice's Diamond 1.0 software. Performance may vary when using a different software version or targeting a different device density or speed grade within the LatticeXP2 family.

Ordering Part Number

The Ordering Part Number (OPN) for the Interleaver/de-interleaver targeting LatticeXP2 devices is INTV-DINT-X2-U3.

LatticeSC/M Devices

Table A-7. Performance and Resource Utilization[1]

IPexpress User-Configurable Mode	Slices	LUTs	Registers	I/Os	sysMEM EBRs	f_{MAX} (MHz)
Convolutional Interleaver DVB	107	139	159	23	1	375
Convolutional De-Interleaver DVB	110	154	164	23	1	375
Rectangular Interleaver 802.16	56	84	101	24	2	375
Rectangular De-Interleaver 802.16	77	117	132	24	2	375

1. Performance and utilization data are generated using an LFSC3GA25E-7F900C device with Lattice's Diamond 1.0 software. Performance may vary when using a different software version or targeting a different device density or speed grade within the LatticeSC/M families.

Ordering Part Number

The Ordering Part Number (OPN) for the Interleaver/de-interleaver targeting LatticeSC/M devices is INTV-DINT-XM-U3.

OCEAN CHIPS
Океан Электроники
Поставка электронных компонентов

Компания «Океан Электроники» предлагает заключение долгосрочных отношений при поставках импортных электронных компонентов на взаимовыгодных условиях!

Наши преимущества:

- Поставка оригинальных импортных электронных компонентов напрямую с производств Америки, Европы и Азии, а так же с крупнейших складов мира;
- Широкая линейка поставок активных и пассивных импортных электронных компонентов (более 30 млн. наименований);
- Поставка сложных, дефицитных, либо снятых с производства позиций;
- Оперативные сроки поставки под заказ (от 5 рабочих дней);
- Экспресс доставка в любую точку России;
- Помощь Конструкторского Отдела и консультации квалифицированных инженеров;
- Техническая поддержка проекта, помощь в подборе аналогов, поставка прототипов;
- Поставка электронных компонентов под контролем ВП;
- Система менеджмента качества сертифицирована по Международному стандарту ISO 9001;
- При необходимости вся продукция военного и аэрокосмического назначения проходит испытания и сертификацию в лаборатории (по согласованию с заказчиком);
- Поставка специализированных компонентов военного и аэрокосмического уровня качества (Xilinx, Altera, Analog Devices, Intersil, Interpoint, Microsemi, Actel, Aeroflex, Peregrine, VPT, Syfer, Eurofarad, Texas Instruments, MS Kennedy, Miteq, Cobham, E2V, MA-COM, Hittite, Mini-Circuits, General Dynamics и др.);

Компания «Океан Электроники» является официальным дистрибьютором и эксклюзивным представителем в России одного из крупнейших производителей разъемов военного и аэрокосмического назначения «**JONHON**», а так же официальным дистрибьютором и эксклюзивным представителем в России производителя высокотехнологичных и надежных решений для передачи СВЧ сигналов «**FORSTAR**».

AVIC JONHON

«JONHON» (основан в 1970 г.)

Разъемы специального, военного и аэрокосмического назначения:

(Применяются в военной, авиационной, аэрокосмической, морской, железнодорожной, горно- и нефтедобывающей отраслях промышленности)

«FORSTAR» (основан в 1998 г.)

ВЧ соединители, коаксиальные кабели, кабельные сборки и микроволновые компоненты:

(Применяются в телекоммуникациях гражданского и специального назначения, в средствах связи, РЛС, а так же военной, авиационной и аэрокосмической отраслях промышленности).

FORSTAR®

Телефон: 8 (812) 309-75-97 (многоканальный)
Факс: 8 (812) 320-03-32
Электронная почта: ocean@oceanchips.ru
Web: http://oceanchips.ru/
Адрес: 198099, г. Санкт-Петербург, ул. Калинина, д. 2, корп. 4, лит. А