FIR Filter IP Core User's Guide

May 2011
ipug79_01.3

Table of Contents

The Lattice FIR (Finite Impulse Response) Filter IP core is a widely configurable, multi-channel FIR filter, implemented using high performance sysDSP™ blocks available in Lattice devices. In addition to single rate filters, the IP core also supports a range of polyphase decimation and interpolation filters. The utilization versus throughput trade-off can be controlled by specifying the multiplier multiplexing factor used for implementing the filter. For example, setting the multiplier multiplexing factor to the maximum value supported by the GUI results in the best resource utilization, whereas setting the multiplier multiplexing factor to 1 results in the best throughput. The FIR Filter IP core supports up to 256 channels, with each having up to 2048 taps.

The input data, coefficient and output data widths are configurable over a wide range. The IP core uses full internal precision while allowing variable output precision with several choices for saturation and rounding. The coefficients of the filter can be specified at generation time and/or re-loadable during run-time through input ports.

The FIR Filter IP core can also be generated using the Lattice FIR Filter Simulink® Model. For information on the Simulink flow, refer to the FPGA Design with ispLEVER tutorial.

Quick Facts

Table 1-1 through **Table 1-3** give quick facts about the FIR Filter IP core for LatticeECP™, LattceECP2™, LatticeECP2M™, LatticeXP2™, and LatticeECP3™ devices

Table 1-1. FIR Filter IP Core for LatticeECP Devices Quick Facts

		FIR Filter IP Configuration		
		4 Channels 64 taps 1 multiplier	**1 Channels 32 taps 32 multiplier**	**1 Channels 32 taps 8 multiplier**
Core Requirements	FPGA Families Supported	Lattice ECP		
	Minimal Device Needed	LFECP6E	LFECP33E	LFECP6E
Resource Utilization	Targeted Device	LFECP33E-5F672C		
	LUTs	150	300	700
	sysMEM EBRs	2	0	0
	Registers	250	850	650
	MULT18X18	1	32	8
Design Tool Support	Lattice Implementation	Lattice Diamond™ 1.2 or ispLEVER® 8.1 SP1		
	Synthesis	Synopsys© Synplify© Pro for Lattice D-2010.03L-SP1		
	Simulation	Aldec© Active-HDL© 8.2 Lattice Edition II		
		Mentor Graphics© ModelSim© SE 6.3F		

Table 1-2. FIR Filter IP Core for LatticeECP2, Lattice ECP2M, and LatticeXP2 Devices Quick Facts

		FIR IP Configuration		
		4 Channels 64 taps 1 multiplier	**1 Channels 32 taps 32 multiplier**	**1 Channels 32 taps 8 multiplier**
Core Requirements	FPGA Families Supported	Lattice ECP2/ECP2M/XP2		
	Minimal Device Needed	LFE2-6E LFE2M20E LFXP2-5E	LFE2-35E LFE2M35E LFXP2-40E	LFE2-6E LFE2M20E LFXP2-8E
Resource Utilization	Targeted Device	LFE2-50E-7F672C LFE2M50E-7F672C LFXP2-40E-7F672C		
	LUTs	150	300	550
	sysMEM EBRs	2	0	0
	Registers	250	850	650
	MULT18X18	1	32	8
Design Tool Support	Lattice Implementation	Lattice Diamond 1.2 or ispLEVER 8.1 SP1		
	Synthesis	Synopsys Synplify Pro for Lattice D-2010.03L-SP1		
	Simulation	Aldec Active-HDL 8.2 Lattice Edition II		
		Mentor Graphics ModelSim SE 6.3F		

Table 1-3. FIR Filter IP Core for LatticeECP3 Devices Quick Facts

		FIR IP Configuration		
		4 Channels 64 taps 1 multiplier	**1 Channels 32 taps 32 multiplier**	**1 Channels 32 taps 8 multiplier**
Core Requirements	FPGA Families Supported	Lattice ECP3		
	Minimal Device Needed	LFE3-35EA		
Resource Utilization	Targeted Device	LFE3-70E-8FN672CES		
	LUTs	150	100	650
	sysMEM EBRs	2	0	0
	Registers	250	1000	800
	DSP Slice	1	16	5
Design Tool Support	Lattice Implementation	Lattice Diamond 1.2 or ispLEVER 8.1 SP1		
	Synthesis	Synopsys Synplify Pro for Lattice D-2010.03L-SP1		
	Simulation	Aldec Active-HDL 8.2 Lattice Edition II		
		Mentor Graphics ModelSim SE 6.3F		

Features

- Variable number of taps up to 2048
- Input and coefficients widths of 4 to 32 bits
- Multi-channel support for up to 256 channels

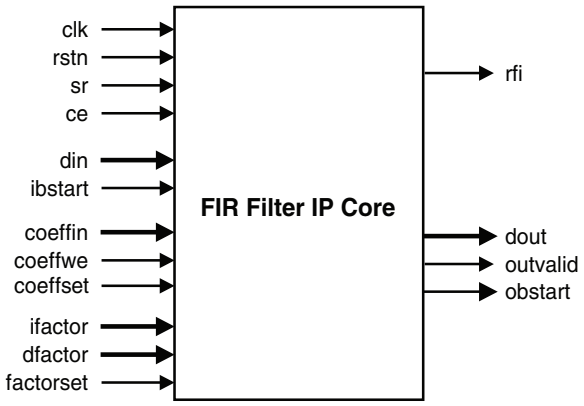
- Decimation and Interpolation ratios from 2 to 256
- Support for half-band filter
- Configurable parallelism from fully parallel to serial
- Signed or unsigned data and coefficients
- Coefficients symmetry and negative symmetry optimization
- Re-loadable coefficients support
- Full precision arithmetic
- Selectable output width and precision
- Selectable overflow: wrap-around or saturation
- Selectable rounding: truncation, round towards zero, round away from zero, round to nearest and convergent rounding
- Width and precision specified using fixed point notations
- Handshake signals to facilitate smooth interfacing

This chapter provides a functional description of the FIR Filter IP core.

Interface Diagram

The top-level interface diagram for the FIR Filter IP core is shown in **Figure 2-1**.

Figure 2-1. Top-Level Interface for the FIR Filter IP Core



FIR Filter Architecture

FIR filter operation on data samples can be described as a sum-of-products operation. For an N-tap FIR filter, the current input sample and (N-1) previous input samples are multiplied by N filter coefficients and the resulting N products are added to give one output sample as shown below.

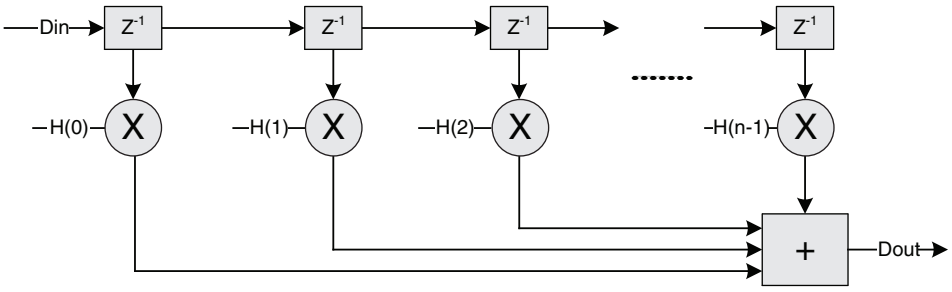
$$y_n = \sum_{i=0}^{N-1} x_{n-i} h_i = x_n h_0 + x_{n-1} h_1 + \ldots + x_{n-N+1} h_{N-1} \quad (1)$$

In the above equation hn, n=0,1,…,N-1 is the impulse response, x_n, n=0,1,…,×, is the input and y_n, n=0,1,…,×, is the output. The number of delay elements (N-1) represents the order of the filter. The number of input data samples (current and previous) used in the calculation of one output sample represents the number of filter taps (N).

Direct-form Implementation

In the direct-form implementation shown in **Figure 2-2**, the input samples will be shifted into a shift register queue and each shift register is connected to a multiplier. The products from the multipliers are summed to get the FIR filter's output sample.

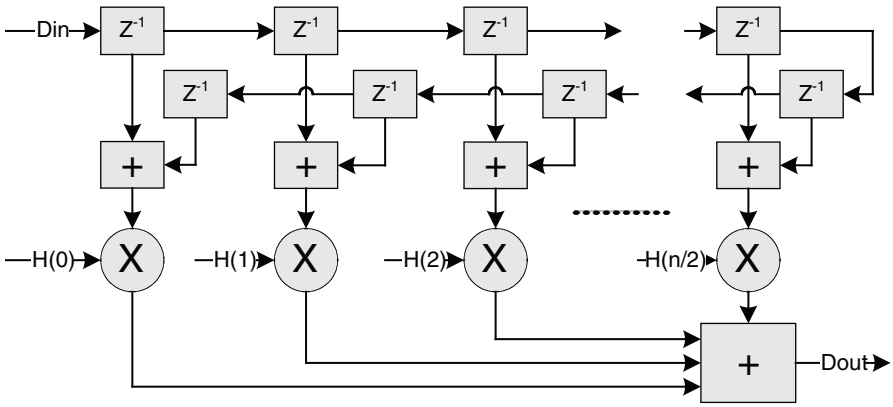
Figure 2-2. Direct-form FIR Filter



Symmetric Implementation

The impulse response for most FIR filters is symmetric. This symmetry can generally be exploited to reduce the arithmetic requirements and produce area-efficient filter realizations. It is possible to use only one half of the multipliers for symmetric coefficients compared to that used for a similar filter with non-symmetric coefficients. An implementation for symmetric coefficients is shown in **Figure 2-3**.

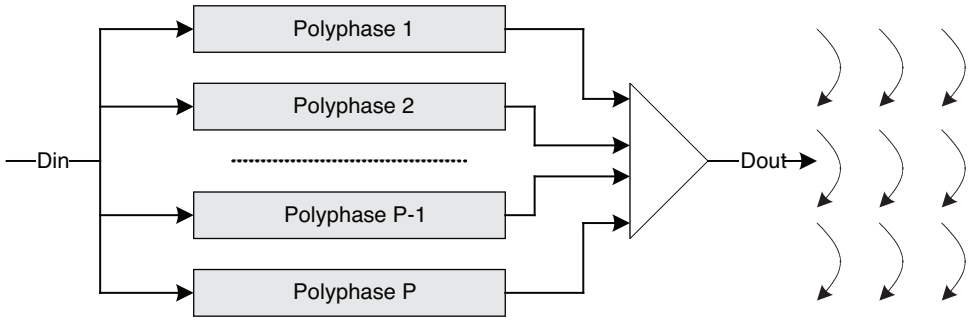
Figure 2-3. Symmetric Coefficients FIR Filter Implementation



Polyphase Interpolation FIR Filter

The polyphase interpolation filter option implements the computationally efficient 1-to-P interpolation filter shown below where P is an integer greater than 1. **Figure 2-4** shows a polyphase interpolator, where each branch is referred to as a polyphase.

Figure 2-4. Polyphase Interpolator

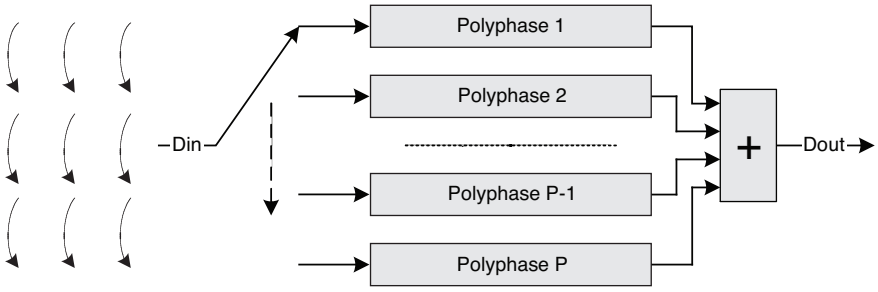


In this structure, the input data will be loaded into each poly-phase at the same time and the output data of each polyphase will be unloaded as an output sample of the FIR. The number of polyphases is equal to the interpolation factor. The coefficients are assigned to all polyphases evenly.

Polyphase Decimation FIR Filter

The polyphase decimation filter option implements the computationally efficient P-to-1 decimation filter shown in **Figure 2-5**, where P is an integer greater than 1.

Figure 2-5. Polyphase Decimator



In this structure, the input sample is loaded sequentially into each of the polyphases with only one polyphase fed at a time. When all the polyphases are loaded with a sample, the result from the polyphases are summed and unloaded as the FIR filter's output. In this scheme, P input samples generate one output sample, where P is the decimation factor.

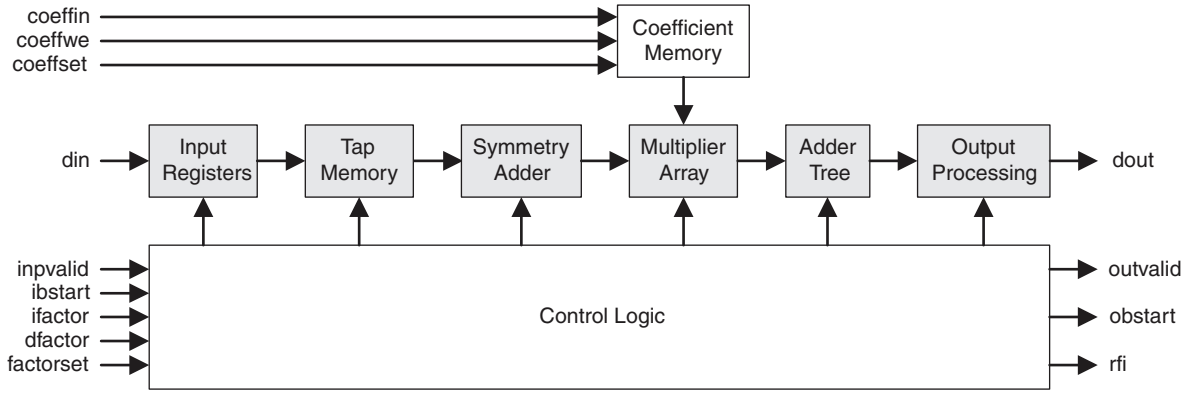
Multi-channel FIR Filters

It is very common to see FIR filters used in multi-channel processing scenarios. The maximum possible throughput of a FIR filter implementation is often much higher than the throughput required for a single channel being processed. For such applications, it is desirable to use the same resources in a time multiplexed way to realize multi-channel FIR filters. Except in fully parallel implementations, where enough multipliers are used to perform all the necessary computations in one clock cycle, the FIR filter uses independent tap and coefficient memories to feed each multiplier. Hence, multi-channel implementations result in lower memory usage compared to multiple instantiations of FIR filters. For cases, where all the channels use the same coefficient set, using a multi-channel FIR filter has the clear advantage of requiring smaller coefficient memories.

Implementation Details

Figure 2-6 shows the functional block diagram of the FIR Filter IP core.

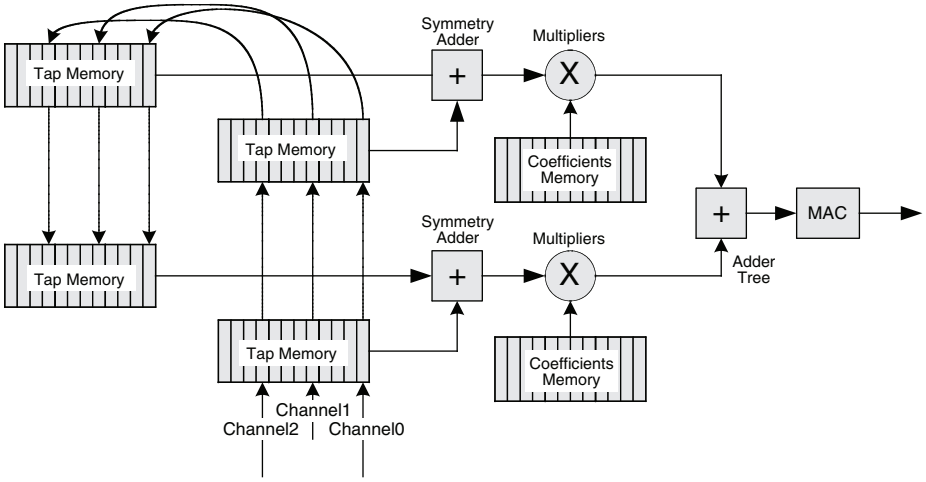
Figure 2-6. Functional Block Diagram of the FIR Filter IP Core



The data and coefficients are stored in different memories shown as tap memory and coefficients memory in the above diagram. The symmetry adder is used if the coefficients are symmetric. The multiplier array contains one or more multipliers depending on the user specification. The adder tree performs the sum of products. Depending on the configuration, the adder tree, or a part of it, is implemented inside DSP blocks. The output processing block performs the output width reduction and precision control. This block contains logic to support different types of rounding and overflow. The block labeled “Control Logic” manages the scheduling of data and arithmetic operations based on the type of filter (interpolation, decimation or multi-channel) and multiplier multiplexing.

The tap and coefficient memories are managed differently for different configurations of the FIR filter. **Figure 2-7** shows the memory assignments for a 16-tap, 3-channel, symmetric FIR filter with two multipliers.

Figure 2-7. Tap and Coefficient Memory Management for a Sample FIR Filter



In the diagram, there are two tap memories and a coefficient memory for each multiplier. The depth of each memory is ceil(taps/multiplier)*channel, where the operator, ceil(x), returns the next higher integer if the argument x is fractional. The memory depth is 12 in this example.

Configuring the FIR Filter IP Core

Architecture Options

The options for number of channels, number of taps and filter type are independent and directly specified in the **Architecture** tab of the IP core GUI (see "Parameter Settings" on page 18 for details). If a polyphase decimator or interpolator is required, the decimation or interpolation factor can be directly specified in the GUI. The decimation or interpolation factor can also be specified through input ports during operation by selecting the corresponding **Variable** option. If the **Variable decimation** (or **Variable interpolation**) factor option is selected, the decimation (or interpolation) factor can be varied from two to **Decimation factor** (or **Interpolation factor**) through the input port.

Coefficients Specification

The coefficients of the filter are specified using a coefficients file. The coefficients file is a text file with one coefficient per line. If the coefficients are symmetric, the check box **Symmetric Coefficients** must be checked so the IP core uses symmetry adders to reduce the number of multipliers used. If the **Symmetric Coefficients** box is checked, only one-half of the coefficients are read from the coefficient file. For an n-tap symmetric coefficients filter, the number of coefficients read from the coefficients file is equal to ceil(n/2). For multi-channel filters, the coefficients for channel 0 are specified first, followed by those for channel 1, and so on. For multi-channel filters, there is an option to specify whether the coefficients are different for each channel or the same (common) for all the channels. If the coefficients are common, only one set of coefficients needs to be specified in the coefficients file. The coefficient values in the file can be in any radix (decimal, hexadecimal or binary) selected by the user. A unary negative operator is used only if the coefficients are specified in decimal radix. For hexadecimal and binary radices, the numbers must be represented in twos complement form. An example coefficients file in decimal format for an 11-tap, 16-bit coefficients set is given below. In this example, the coefficients binary point is 0.

-556
-706
-857
-419
1424
5309
11275
18547
25649
30848
32758

An example coefficients file in floating point format for the above case when the Coefficients binary point position is 8, is given below. The coefficients will be quantized to conform to the 16.8 fractional number, in which 16 is the full width of the coefficients and 8 is the width of the fractional part.

-2.1719
-2.7578
-3.3477
-1.6367
5.5625
20.7383
44.043
72.45
100.0191
120.5
127.96

If the check box **Reloadable Coefficients** is checked, the coefficients can be reloaded to the FIR filter during the operation of the core. With this option, the desired coefficients must be loaded before the operation of the filter. The coefficients must be loaded in a specific order that is determined by the program supplied with the IP core. The IP core can also optionally do the reordering internally, albeit using more resources. If this option is desired, the check box **Reorder Coefficients Inside** can be checked. With this option, the coefficients can be loaded in the normal sequential order to the core.

Multiplier Multiplexing Factor

The throughput and the resource utilization can be controlled by assigning a proper value to the **Multiplier Multiplexing Factor** parameter. Full parallel operation (one output data per clock cycle) can be achieved by setting the **Multiplier Multiplexing Factor** to 1. If the **Multiplier Multiplexing Factor** is set to the maximum value displayed in the GUI, full series operation is supported and it takes up to n clocks to compute one output data sample, where n is the number of taps for a non-symmetric FIR filter and half the number of taps for a symmetric FIR filter. The maximum value of the **Multiplier Multiplexing Factor** for different configurations of an n-tap FIR filter is given in Table 1.

Table 1. Maximum Multiplier Multiplexing Factor for Different Configurations[1]

FIR Type	Single Rate	Interpolator with Factor=i	Decimator with Factor=d
Non-symmetric	n	Ceil(n/i)	Ceil(n/d)
Symmetric	Ceil(n/2)	Ceil(n/2i)	Ceil(n/2d)
Half-band	floor((n+1)/4)+1	floor((n+1)/4)	floor((n+1)/8)+1

1. The operator floor (x) returns the next lower integer, if x is a fractional value.

I/O Specification Options

The controls in the I/O Specifications GUI tab are used to define the various widths and precision methods in the data path. The width and binary point positions of the input data and coefficients can be defined independently. From the input data width, coefficient width and the number of taps, the full precision output width and true location of the output binary point automatically get fixed. The full precision output is converted to user specified output width by dropping some least significant (LS) and some most significant (MS) bits and by performing the specified rounding and overflow processing. The output is specified by the output width and the output binary point position parameter.

Rounding

The following five options are supported for rounding:

- **None** – Discards all bits to the right of the output least significant bit and leaves the output uncorrected.
- **Rounding up** – Rounds to nearest more positive number.
- **Rounding away from zero** – Rounds away from zero if the fractional part is exactly one-half.
- **Rounding towards zero** – Rounds towards zero if the fractional part is exactly one-half.
- **Convergent rounding** – Rounds to the nearest even value if the fractional part is exactly one-half.

Implementation Options

Memory Type

The FIR Filter IP core uses memories for storing delay tap data, coefficients and for some configurations, input or output data. The number of memory units used depends on several parameters including data width, number of taps, filter type, number of channels and coefficient symmetry. In most cases, each multiplier requires one data memory unit and one coefficient memory unit. Interpolation or decimation filters may additionally use input or output buffers. The memory type GUI option can be used to specify whether EBR or distributed memory is used for data, coefficient, input and output storage. The option called “Auto” leaves that choice to the IP generator tool, which uses EBR if the memory is deeper than 128 locations and distributed memory otherwise.

Signal Descriptions

A description of the Input/Output (I/O) ports for the FIR Filter IP core is provided in **Table 2-1**. The top-level interface diagram for the FIR Filter IP core is shown in **Figure 2-1**.

Table 2-1. Top-Level Port Definitions

Port	Bits	I/O	Description
General I/Os			
clk	1	I	System clock for data and control inputs and outputs.
rstn	1	I	System wide asynchronous active-low reset signal.
din	Input data width	I	Input data.
inpvalid	1	I	Input valid signal. The input data is read-in only when inpvalid is high.
dout	Output width	O	Output data
outvalid	1	O	Output data qualifier. Output data dout is valid only when this signal is high.
rfi	1	O	Ready for input. This output, when high, indicates that the IP core is ready to receive the next input data. A valid data may be applied at din only if rfi was high during the previous clock cycle.
When Reloadable coefficients is selected			
coeffin	See note 1	I	Coefficients input. The coefficients have to be loaded through this port in a specific order. Refer to the section “Interfacing with the FIR Filter IP core” for details.
coeffwe	1	I	When asserted, the value on bus coeffin will be written into coefficient memories.

Table 2-1. Top-Level Port Definitions

Port	Bits	I/O	Description
coeffset	1	I	This input is used to signal the filter to use the recently loaded coefficient set. This signal must be pulsed high for one clock cycle after the loading the entire coefficient set using coeffin and coeffwe.
When Number of channels is greater than 1			
ibstart	1	I	Input block start. For multi-channel configurations, this input identifies channel 0 of the input.
obstart	1	O	Output block start. For multi-channel configurations, this output identifies channel 0.
When Variable interpolation factor or Variable decimation factor is checked			
ifactor	Ceil(Log_2(Interpolation factor+1))	I	Interpolation factor value
dfactor	Ceil(Log_2(Decimation factor+1))	I	Decimation factor value
factorset	1	I	Sets the interpolation factor or the decimation factor.
Optional I/Os			
ce	1	I	Clock Enable. While this signal is de-asserted, the core will ignore all other synchronous inputs and maintain its current state
sr	1	I	Synchronous Reset. When asserted for at least one clock cycle, all the registers in the IP core are initialized to reset state.

1. a. Width for signed type and symmetric interpolation is Coefficients width +1.
   b. Width for unsigned and symmetric interpolation is Coefficients width +2.
   c. Width for all other cases is Coefficients width.

Interfacing with the FIR Filter IP core

Multiple Channels

For multi-channel implementations, two ports, ibstart and obstart, are available in the IP core to synchronize the channel numbers. The input ibstart is used to identify channel 0 data applied at the inputs. The output obstart goes high simultaneously with channel 0 output data.

Variable Interpolation/ Decimation Factor

When the interpolation (or decimation) factor is variable, the ports ifactor (or dfactor) and factorset are added to the IP core. The interpolation (or decimation) factor applied on the port ifactor (or dfactor) is set when the strobe signal factorset is high. When the interpolation (or decimation) factor changes, the output rfi goes low for a few cycles. When it becomes high again, the filter performs as an interpolating (or decimating) filter corresponding to the new factor value.

Reloadable Coefficients

When **Reloadable Coefficients** is selected, the two added ports, *coeffin* and *coeffwe*, are used to reload the coefficients. All the coefficients need to be loaded in one batch, while keeping the signal *coeffwe* high during the entire duration of loading. After all the coefficients are loaded, the input signal *coeffset* must be pulsed high for one clock cycle for the new coefficients to take effect.

There are two ways in which coefficients can be applied for reloading the coefficients memory, as specified by the **Reorder Coefficients Inside** parameter.

When **Reorder Coefficients Inside** is not selected, the coefficients have to be applied in a particular sequence for reloading the coefficients memory. The raw coefficients, as specified in the coefficients file, can be converted to the reloadable sequence by using the coefficients generation program coeff_gen.exe (for Windows) available under the "gui" folder in the IP installation directory (for example, under the `C:\LatticeCore\fir_core_v4.2\gui`

folder). The names of the coefficient generation program for UNIX and Linux are *coeff_gen_s* and *coeff_gen_l* respectively. For Windows, the program is invoked as follows:

```
coeff_gen.exe  <IP_file_name>.lpc
```

This command converts the coefficients in the input file[1], as referred by the *coefffile=* parameter in the lpc file, to the loadable coefficients sequence file called *coeff.mem*. Note that the output file may contain more coefficients than there originally were due to inserted zero coefficients. All the coefficients in the output file, including the zeros, have to be applied sequentially through the *coeffin* port. To obtain the sequence of application of coefficients, edit the input coefficients file with sequential numbers 1,2,… and run the *coeff_gen.exe* command. In the reloadable coefficients mode, the core will not be ready for operation (the *rfi* output will not be high) until the coefficients are loaded and *coeffset* is asserted high.

When the parameter **Reorder Coefficients Inside** is selected, the coefficients will be reordered inside the IP core without requiring manual reordering described previously. With this option, reordering logic is added to the IP core and the user can apply the coefficients in the normal sequence.

With this capability, if the parameter **Symmetric Coefficients** is selected, only half of the coefficients provided will be used. For example, if the raw coefficient input sequence is: "1 2 3 4 5 6 5 4 3 2 1," the coefficients that will be used will be "1 2 3 4 5 6."

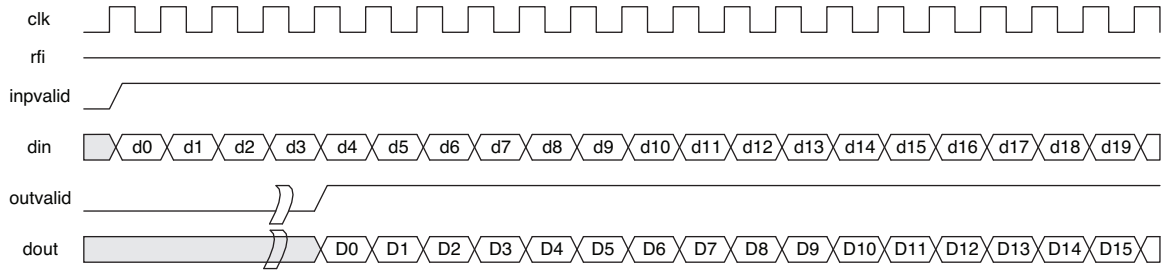
Similarly, if **Half Band** is selected with the this capability, all of the input coefficients in even locations, except the last one, will be discarded. For example, if the raw coefficient input sequence is: "1 0 2 0 3 0 4 0 5 6 5 0 4 0 3 0 2 0 1," the coefficients that will be used will be "1 2 3 4 5 6."

Timing Specifications

Timing diagrams for the FIR Filter IP core are given in **Figure 2-8** through **Figure 2-17**. Note that there are different timing specifications for certain FIR filter applications using LatticeECP3 devices. **Figures 2-8** through **2-11** apply to all FIR applications. **Figures 2-12** through **2-14** apply to all FIR applications using LatticeECP, LatticeECP2/S, LatticeECP2M/S and LatticeXP2 devices and the following applications using LatticeECP3 devices: negative symmetry, half band, factor variable interpolation and decimation and applications using 36x36 multipliers. Figures **2-15** through **2-17** apply to all other LatticeECP3 applications.

Timing Specifications Applicable to All Devices

Figure 2-8. Single Channel, Single Rate FIR Filter with Continuous Inputs



1. If the parameter *varcoeff=* in the lpc file is set to "yes", change it to "no" before generating the new coefficients file.

Figure 2-9. Single Channel, Single Rate FIR Filter with Gaps in Input

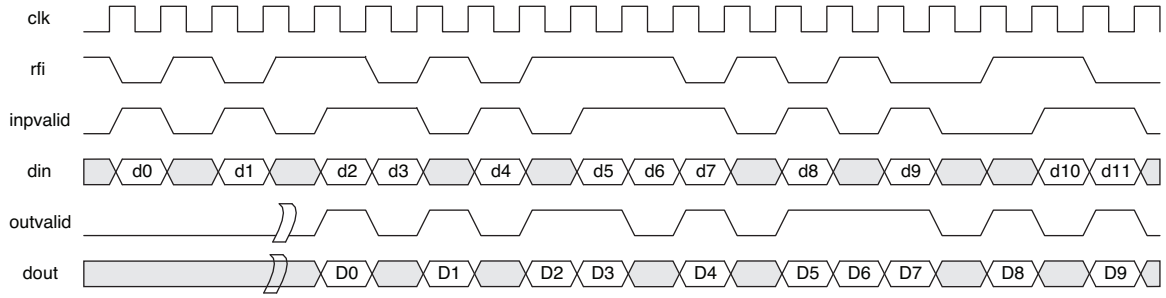


Figure 2-10. Factorset Signals

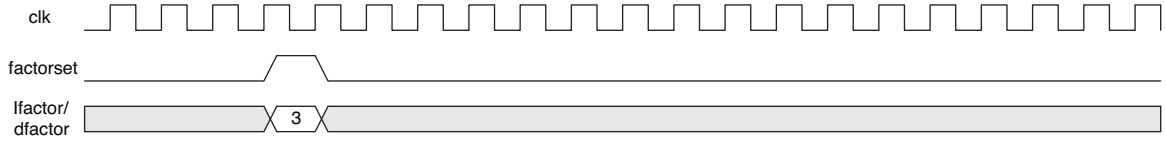
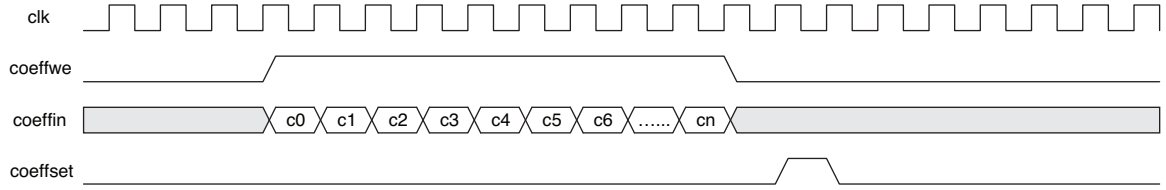


Figure 2-11. Coefficient Reloading



Timing Specifications Applicable to All LatticeECP, LatticeECP2/S, LatticeECP2M/S, LatticeXP2 and Certain LatticeECP3 Implementations

As indicated previously, **Figures 2-12** through **2-14** apply to all FIR applications using LatticeECP, LatticeECP2/S, LatticeECP2M/s and LatticeXP2 devices and the following applications using LatticeECP3 devices: negative symmetry, half band, factor variable interpolation and decimation and applications using 36x36 multipliers.

Figure 2-12. Multi-Channel Single Rate FIR Filter (3 Channels)

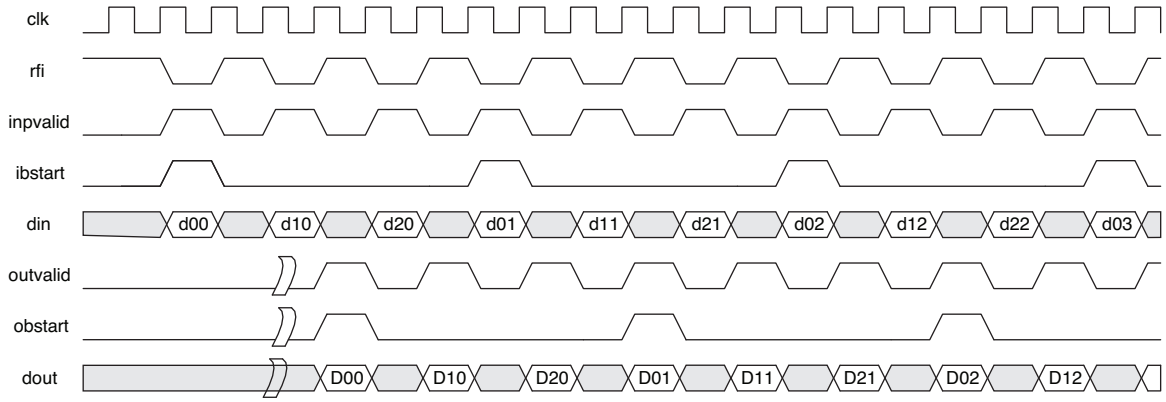


Figure 2-13. Multi-Channel (3 Channels) Interpolator (Factor of 3)

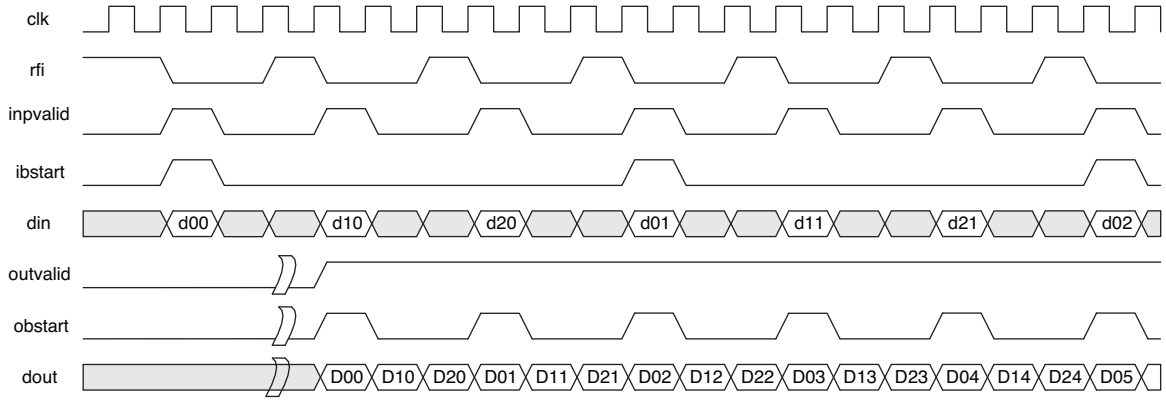
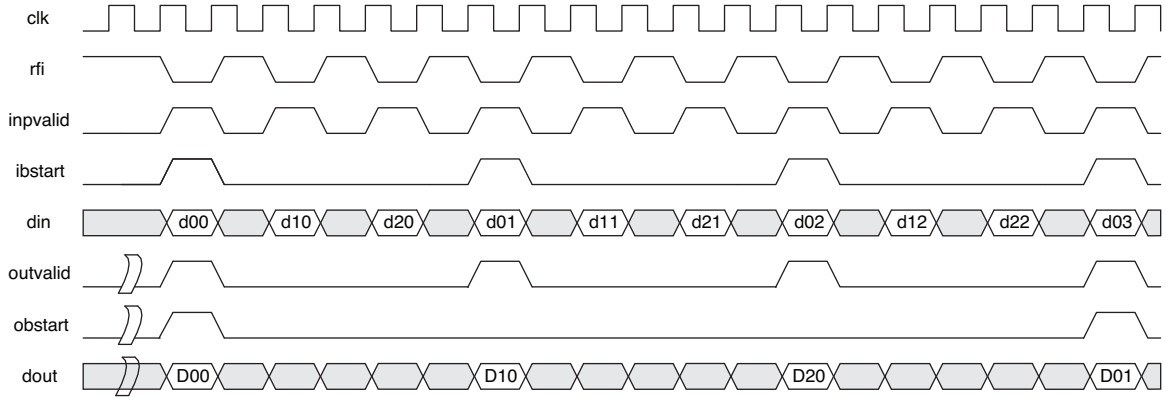


Figure 2-14. Multi-Channel (3 Channels) Decimator (Factor of 3)



Timing Specifications Applicable to Certain LatticeECP3 Implementations

As indicated previously, **Figures 2-15** through **2-17** apply to all LatticeECP3 applications other than those specifically listed in the previous section.

Figure 2-15. Multi-Channel Single Rate FIR Filter (3 Channels)

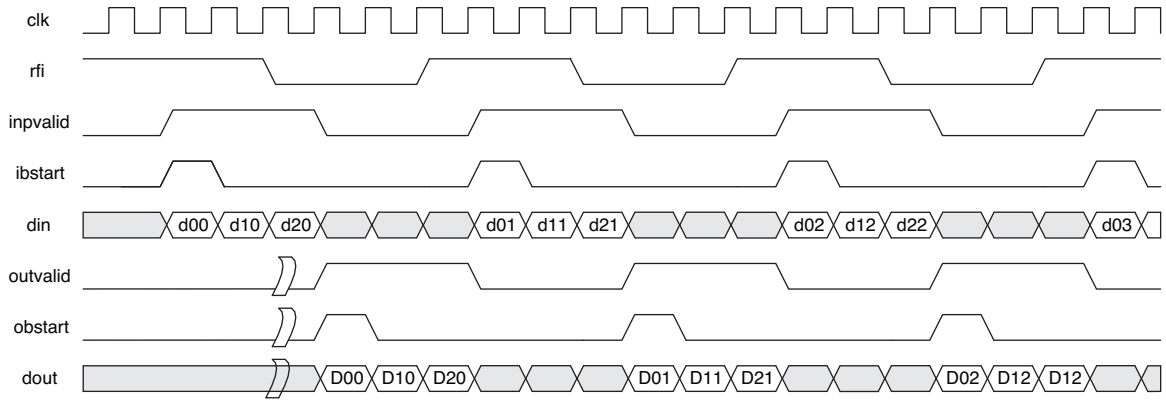


Figure 2-16. Multi-Channel (3 Channels) Interpolator (Factor of 3)

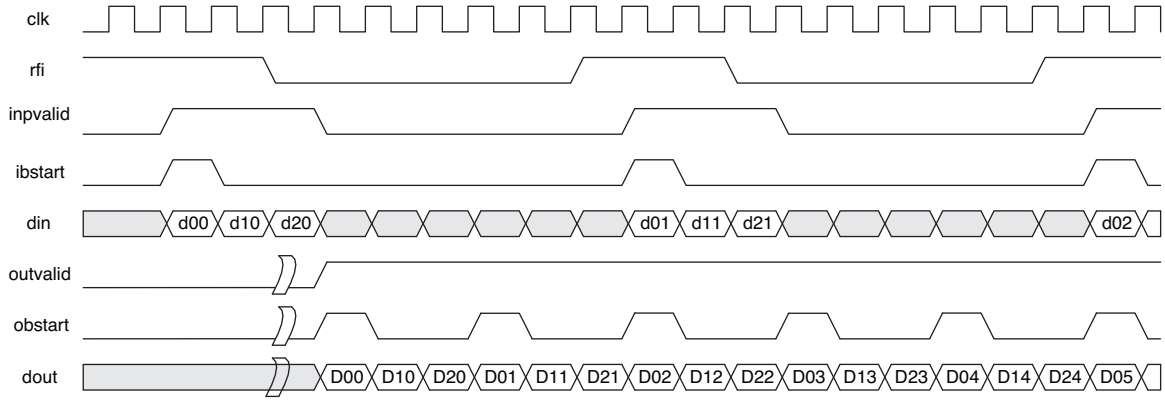
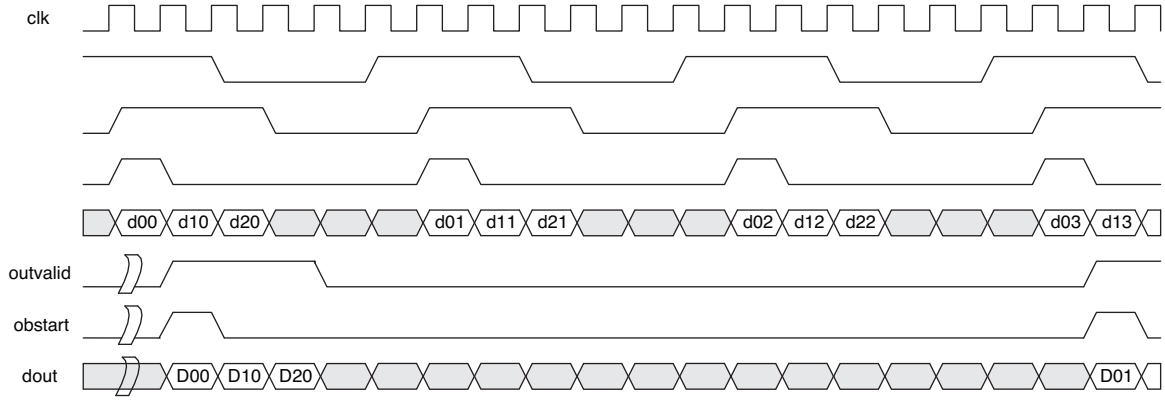


Figure 2-17. Multi-Channel (3 Channels) Decimator (Factor of 3)



Parameter Settings

The IPexpress tool is used to create IP and architectural modules in the Diamond or ispLEVER software. Refer to "IP Core Generation and Evaluation" on page 26 for a description on how to generate the IP.

Table 3-1 provides the list of user configurable parameters for the FIR Filter IP core The parameter settings are specified using the FIR Filter IP core Configuration GUI in IPexpress. The numerous FIR Filter IP core parameter options are partitioned across multiple GUI tabs as described in this chapter.

Table 3-1. Parameter Specifications for the FIR Filter IP Core

Parameter	Range/Options	Default
Filter Specifications		
Number of channels	1 to 256	4
Number of taps	1 to 2048	64
Filter type	Single rate, Interpolator, Decimator	Single rate
Interpolation factor	2 to 256	2
Variable interpolation factor	Yes, No	No
Decimation factor	2 to 256	2
Variable decimation factor	Yes, No	No
Coefficients Specifications		
Reloadable coefficients	Yes, No	Yes
Reorder coefficients inside	Yes, No	No
coefficients set	Common, One per channel	Common
Symmetric coefficients	Yes, No	No
Negative symmetry	Yes, No	No
Half band	Yes, No	No
Coefficient radix	Decimal, Hex, Binary	Decimal
Coefficients file	Type or Browse	—
Advance Options		
Multiplier Multiplexing Factor	Note 1, Note 2	Note 2
Number of sysDSP blocks in a row	5^3	Note 3
I/O Specifications		
Input data type	Signed, Unsigned	Signed
Input data width	4 to 32	16
Input data binary point position	-2 to Input data width + 2	0
Coefficients type	Signed, Unsigned	Signed
Coefficients width	4 to 32	16
Coefficients binary point position	-2 to Coefficients width + 2	0
Output width	4 to Max Output Width	38
Output binary points	(4+Input data binary point position + coefficient binary point position - Max output width) to (Output width + Input data binary point position + Coefficient binary point position - 4)	0
Precision Control		
Overflow	Saturation, Wrap-around	Saturation
Rounding	None, Round-up, Round away from zero, Round towards zero, Convergent rounding	None

Table 3-1. Parameter Specifications for the FIR Filter IP Core (Continued)

Parameter	Range/Options	Default
Memory Type		
Data memory type	EBR, Distributed, Auto	EBR
Coefficient memory type	EBR, Distributed, Auto	EBR
Input buffer type	EBR, Distributed, Auto	EBR
Output buffer type	EBR, Distributed, Auto	EBR
Optimization	Area, Speed	Area
Optional Ports		
ce	Yes, No	No
sr	Yes, No	No
Synthesis Options		
Frequency constraint	1-400	300

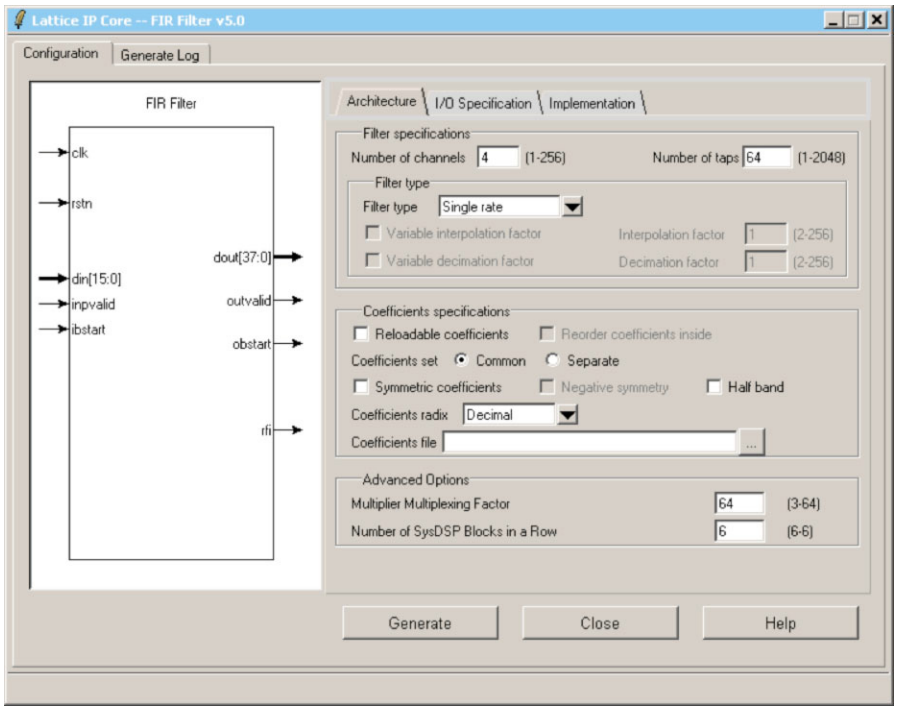
1. The **Multiplier Multiplexing Factor** is limited by the number of DSP blocks in a device (A) and the actual number of DSP blocks a design needs (B). When A>B, the **Multiplier Multiplexing Factor** is set to 1; otherwise the value will be greater than 1.
2. See "Multiplier Multiplexing Factor" on page 21 for details.
3. Maximum number of DSP blocks available in a row in the selected device.

The default values shown in the following pages are those used for the FIR Filter reference design. IP core options for each tab are discussed in further detail.

Architecture Tab

Figure 3-1 shows the contents of the Architecture tab.

Figure 3-1. Architecture Tab of the FIR Filter IP Core GUI



Number of Channels

This option allows the user to specify the number of channels.

Number of Taps

This option allows the user to specify the number of taps.

Filter Type

This option allows the user to specify whether the filter is single rate, interpolator or decimator.

Interpolation Factor

This option allows the user to specify the value of the fixed interpolation factor. When FIR type is interpolation, the value should be 2 to 256, otherwise, it will be set to 1 automatically.

Variable Interpolation Factor

This option allows the user to specify whether the interpolation factor is fixed at the time of IP generation or variable during run-time. If this is checked, the interpolation factor is set through the input port *ifactor* when *factorset* is high.

Decimation Factor

This option allows the user to specify the value of the fixed decimation factor. When FIR type is decimation, the value should be 2 to 256, otherwise, it will be set to 1 automatically.

Variable Decimation Factor

This option allows the user to specify whether the decimation factor is fixed at the time of IP generation or variable during run-time. If this is checked, the decimation factor is set through the input port *dfactor* when *factorset* is high.

Reloadable Coefficients

This option allows the user to specify whether the coefficients are fixed or reloadable. If checked, the coefficients can be reloaded during core operation using the input port *coeffin*.

Reorder Coefficients Inside

When coefficients are reloadable, they need to be entered in a particular order. The reordering can be done using the program supplied along with the IP core. However, the core also provides for optional hardware reordering at the expense of additional hardware resources. If this option is selected, the coefficients can be entered in the normal sequence to the core and the core will internally reorder hem as required. This option is not available when Filter type is interpolator and Symmetric coefficients is enabled.

Coefficients set

This option allows the user to specify whether the same coefficient set is used for all channels or an independent coefficient set is used for each channel.

Symmetric Coefficients

This option allows the user to specify whether the coefficients are symmetric. If this is checked only one half of the number of coefficients (if number of taps is odd, the half value is rounded to the next higher integer) is read from the initialization file.

Negative Symmetry

If this is checked, the coefficients are considered to be negative symmetric. That is the second half of the coefficients are made equal to the negative of the corresponding first-half coefficients.

Half Band

This option allows the user to specify whether a half band filter is realized. If this is checked only one half of the number of coefficients (if the number of taps is odd, the half value is rounded to the next higher integer) is read from the initialization file.

Coefficient Radix

This option allows the user to specify the radix for the coefficients in the coefficients file. For decimal radix, the negative values have a preceding unary minus sign. For hexadecimal (Hex) and binary radices, the negative values must be written in 2's complement form using exactly as many digits as specified by the coefficients width parameter. The floating point coefficients are specified in the form <nn…n>.<dd…d>, where the digits 'n' denote the integer part and the digits 'd', the decimal part. The values of the floating point coefficients must be consistent with the Coefficients width and Coefficients binary point position parameters. For example, if <nn...n>.<dd...d>is 8.4 and Coefficients type is unsigned, the value of the coefficients should be between 0 and 11111111.1111 (255.9375).

Coefficients File

This option allows the user to specify the name and location of the coefficients file. If the coefficients file is not specified, the filter is initialized with a default coefficient set.

Multiplier Multiplexing Factor

This option allows the user to specify the **Multiplier Multiplexing Factor**. This parameter should be set to 1 for full parallel applications and to the maximum value supported in the GUI for full series applications.

Number of sysDSP Blocks in a Row

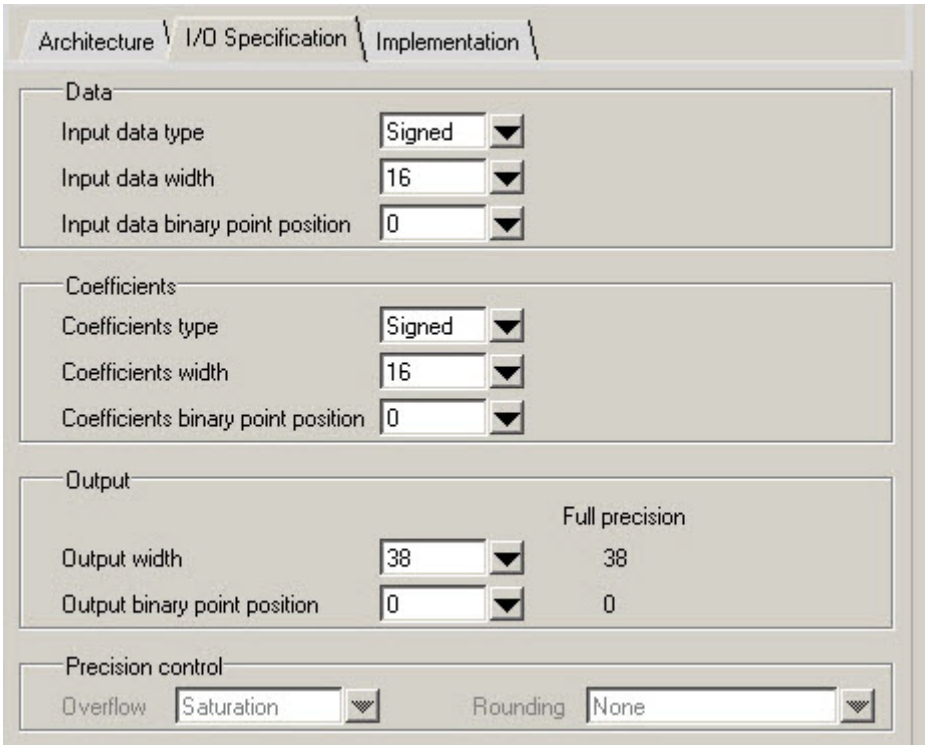
This parameter allows the user to specify the maximum number of DSP multipliers to be use in a DSP row to achieve optimal performance. For example, if the targeted device has 20 multipliers in a DSP row and the design requires 22 multipliers, the user can select to use all 20 multipliers in one row and two multipliers in another row, or fewer than 20 multipliers in each row (e.g. 8), which may yield better performance. Multipliers spread across a maximum of three DSP rows may be used in a single FIR instance.

This parameter is only valid on LatticeECP3 devices.

I/O Specification Tab

Figure 3-2 shows the contents of the I/O Specification tab.

Figure 3-2. I/O Specification Tab of the FIR Filter IP Core GUI



Input Data Type

This option allows the user to specify the input data type as signed or unsigned. If the type is signed, the data is interpreted as a two's complement number.

Input Data Width

This option allows the user to specify input data width.

Input Data Binary Point Position

This option allows the user to specify the location of the binary point in the input data. This number specifies the bit position of the binary point from the LSB of the input data. If the number is zero, the point is right after LSB, if positive, it is to the left of LSB and if negative, it is to the right of LSB.

Coefficients Type

This option allows the user to specifythe coefficients type as signed or unsigned. If the type is signed, the coefficient data is interpreted as a 2's complement number.

Coefficients Width

This option allows the user to specify the coefficients width.

Coefficients Binary Point Position

This option allows the user to specify the location of the binary point in the coefficients. This number specifies the bit position of the binary point from the LSB of the coefficients. If the number is zero, the point is right after LSB, if positive, it is to the left of LSB and if negative, it is to the right of LSB.

Output Width

This option allows the user to specify the output data width. The maximum full precision output width is defined by Max Output Width = Input data width + Coefficients width +ceil(Log2(Number of taps/Interpolation factor)). The

core's output is usually a part of the full precision output equal to the Output width and extracted based on the different binary point position parameters.

The format for the internal full precision output is displayed as static text next to the Output width control in the GUI. The format is displayed as W.F, where W is the full precision output width and F is the location of the binary point from the LSB of the full precision output, counted to the left. For example, if W.F is 16.4, then the output value will be yyyyyyyyyyyy.yyyy in binary radix (e.g. 110010010010.0101).

Output Binary Points

This option allows the user to specify the bit position of the binary point from the LSB of the actual core output. If the number is zero, the point is right after LSB, if positive, it is to the left of LSB and if negative, it is to the right of LSB. This number, together with the parameter Output width determines how the actual core output is extracted from the true full precision output. The precision control parameters Overflow and Rounding are applied respectively when MSBs and LSBs are discarded from the true full precision output.

Overflow

This option allows the user to specify what kind of overflow control is to be used. This parameter is available whenever there is a need to drop some of the MSBs from the true output. If the selection is “Saturation”, the output value is clipped to the maximum, if positive or minimum, if negative, while discarding the MSBs. If the selection is “Wrap-around”, the MSBs are simply discarded without making any correction.

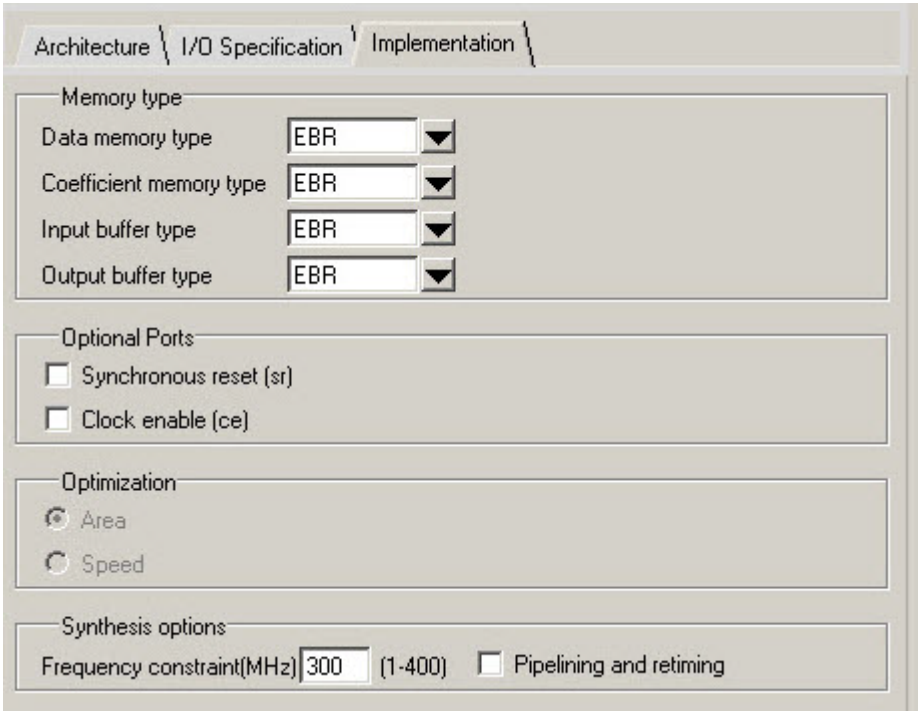
Rounding

This option allows the user to specify the rounding method when there is a need to drop one or more LSBs from the true output.

Implementation Tab

Figure 3-3 shows the contents of the I/O Specification tab.

Figure 3-3. Implementation Tab of the FIR Filter IP Core GUI



Data Memory Type

This option allows the user to specify select the type of memory that is used for storing the data. If the selection is “EBR”, Lattice Embedded Block RAM memories are used for storing the data. If the selection is “Distributed”, look-up-table based distributed memories are used for storing data. If “Auto” is selected, EBR memories are used for memory sizes deeper than 128 locations and distributed memories are used for all other memories.

Coefficient Memory Type

This option allows the user to specify the type of memory that is used for storing the coefficients. If the selection is “EBR”, EBR memories are used for storing the coefficients. If the selection is “Distributed”, distributed memories are used for storing coefficients. If “Auto” is selected, EBR memories are used for memory sizes deeper than 128 locations and distributed memories are used for all other memories.

Input Buffer Type

This option allows the user to specify the memory type for the input buffer.

Output Buffer Type

This option allows the user to specify the memory type for the output buffer.

Optimization

This option specifies the optimization method. If “Area” is selected, the core is optimized for lower resource utilization. If “Speed” is selected, the core is optimized for higher performance, but with slightly higher resource utilization.

Synchronous Reset (sr)

This option allows the user to specify if a synchronous reset port is needed in the IP. Synchronous reset signal resets all the registers in the FIR filter IP core.

Clock Enable (ce)

This option allows the user to specify if a clock enable port is needed in the IP. Clock enable control can be used for power saving when the core is not being used. Use of clock enable port increases the resource utilization and may affect the performance due to the increased routing congestion.

IP Core Generation and Evaluation

This chapter provides information on how to generate the Lattice FIR Filter IP core using the ispLEVER software IPexpress tool included in the Diamond or ispLEVER software, 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 FIR Filter 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 FIR Filter IP core and fully evaluate the core through functional simulation and implementation (synthesis, map, place and route) without an IP license. The FIR Filter 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 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.

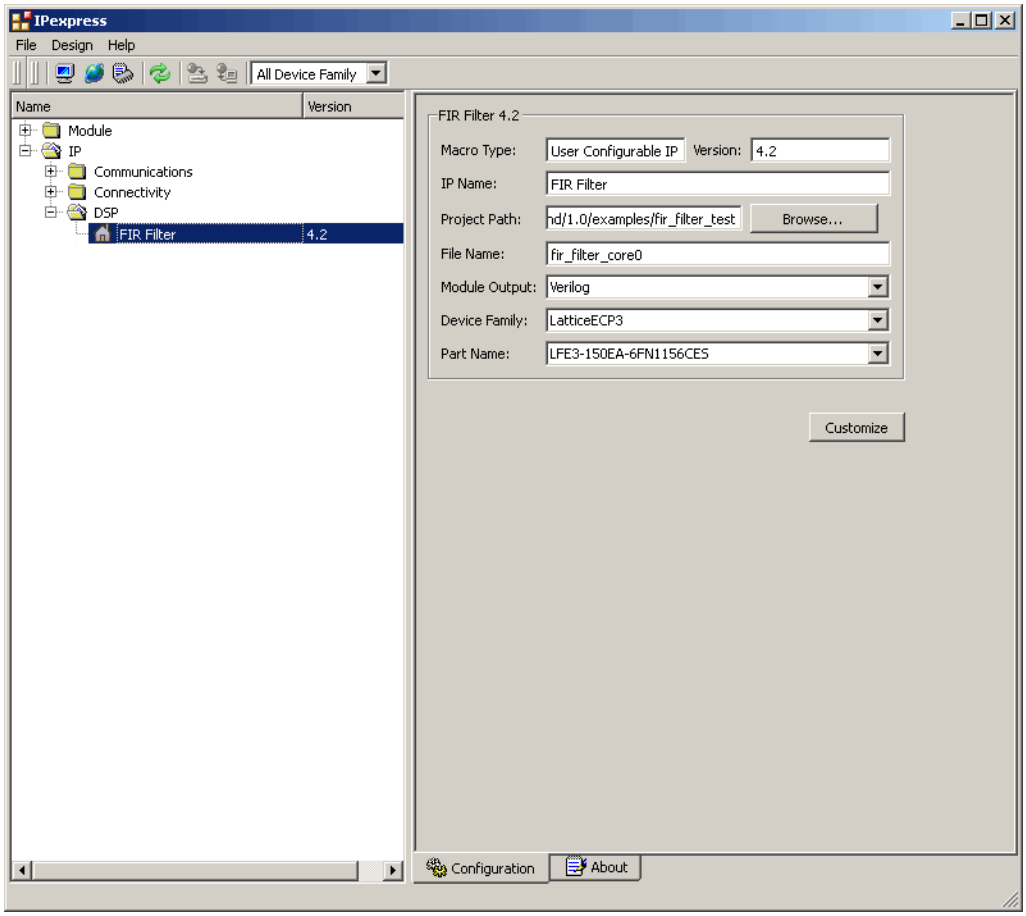
Getting Started

The FIR Filter IP core is available for download from Lattice's 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 FIR Filter IP core 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 located.
- **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. Lattice ECP2M, LatticeECP3, etc.). Only families that support the particular IP core are listed.
- **Part Name** – Specific targeted part within the selected device family.

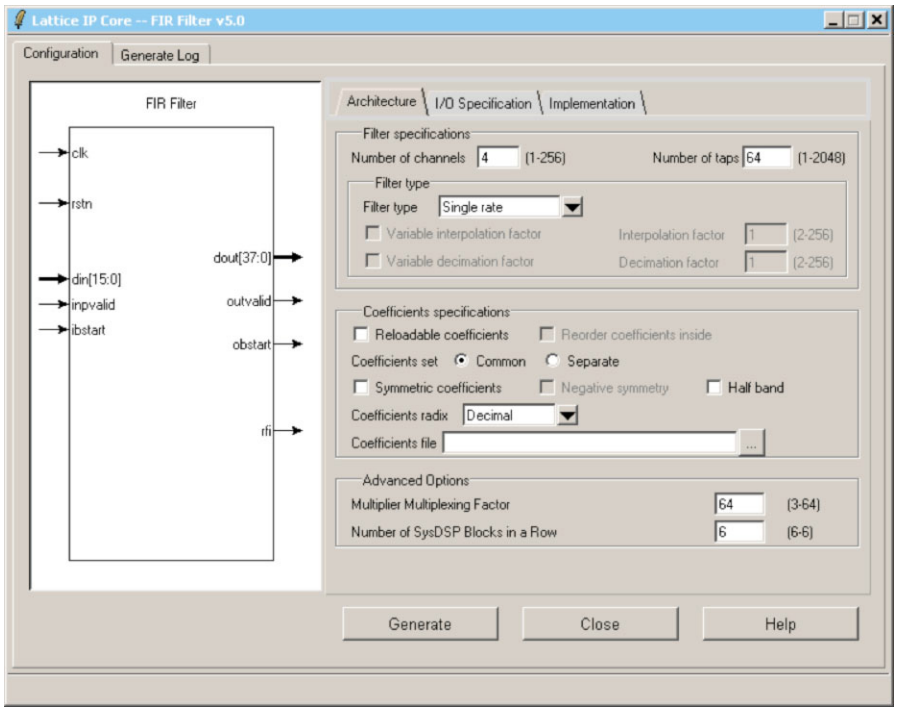
Figure 4-1. IPexpress Dialog Box (Diamond Version)



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 FIR Filter 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 18 for more information on the FIR Filer IP core parameter settings.

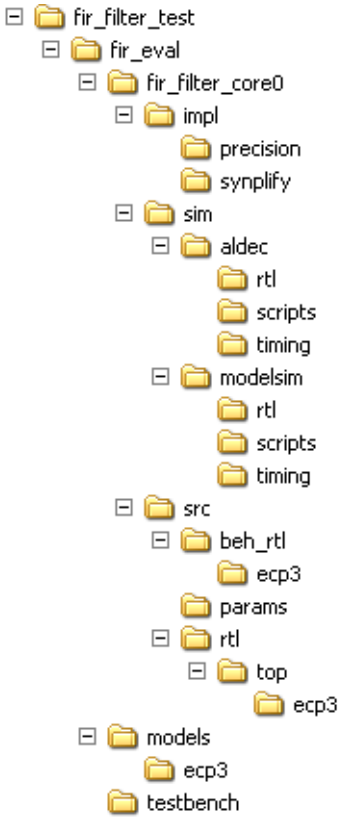
Figure 4-2. Configuration Dialog Box (Diamond Version)



IPexpress-Created Files and Top Level Directory Structure

When the user clicks the **Generate** button, 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. FIR Filter IP Core Generated Directory Structure



The design flow for IP created with the IPexpress tool uses a post-synthesized module (NGO) for synthesis and a protected model for simulation. The post-synthesized module is customized and created during the IPexpress tool generation.

Table 4-1 provides a list of key files created by the IPexpress tool. The names of most of the created files are customized to the user's module name specified in the IPexpress tool. The files shown in Table 4-1 are all of the files necessary to implement and verify the FIR Filter IP core in a top-level design.

Table 4-1. File List

File	Description
<*username*>_inst.v	This file provides an instance template for the IP.
<*username*>.v	This file provides a wrapper for the FIR core for simulation.
<*username*>_beh.v	This file provides a behavioral simulation model for the FIR 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	IPexpress package file (Diamond only). This is a container that 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 this file to the associated Diamond project.

Table 4-1. File List (Continued)

File	Description
pmi_*.ngo	One or more files implementing synthesized memory modules used in the IP core.
*.rom	This file provides filter coefficient memory initialization data.

The following additional files providing IP core generation status information are also generated in the "Project Path" directory:

- **<*username*>_generate.tcl –** A TCL scripts which can regenerate the IP from command line.
- **<*username*>_generate.log** – ispLEVER synthesis and map log file.
- **<*username*>_gen.log** – IPexpress IP generation log file.

Instantiating the Core

The generated FIR Filter 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*>\fir_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 FIR Filter 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 FIR Filter 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*>\fir_eval\<*username*>\sim\modelsim\scripts. The simulation script supporting Aldec evaluation simulation is provided in \<*project_dir*>\fir_eval\<*username*>\sim\aldec\scripts. Both Modelsim and Aldec simulation is supported via test bench files provided in \<*project_dir*>\fir_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*>\fir_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*>\fir_eval\<*username*>\sim\modelsim\scripts.
3. Under the Tools tab, select **Execute Macro** and execute the ModelSim "do" script shown.

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. Chosing* ***Yes*** *closes ModelSim.*

Synthesizing and Implementing the Core in a Top-Level Design

The FIR Filter IP core itself is synthesized and provided in NGO format when the core is generated through IPexpress. You may combine the core in your own top-level design by instantiating the core in your top-level file as described in "Instantiating the Core" on page 30 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 file <*userame*>_top.v is provided in \<*project_dir*>\fir_eval\<*username*>\src\rtl\top. Push-button implementation of the reference design is supported via the project file <*username*>.ldf (Diamond) or .syn (ispLEVER) located in \<*project_dir*>\fir_eval\<*username*>\impl\(synplify or precision).

To use this project file in Diamond:

1. Choose **File > Open > Project**.
2. Browse to \<*project_dir*>\fir_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*>\fir_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 FIR Filter IP 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 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

LatticeECP

- HB1000, *LatticeECP/EC Family Handbook*

LatticeECP2/M

- HB1003, *LatticeECP2/M Family Handbook*

LatticeECP3

- HB1009, *LatticeECP3 Family Handbook*

LatticeXP2

- DS1009, *Lattice XP2 Datasheet*

Revision History

Date	Document Version	IP Version	Change Summary
September 2008	01.0	3.1	Initial release.
April 2009	01.1	4.0	Added support for LatticeECP3 FPGA family.
			Updated appendices for ispLEVER 7.2 SP1.
June 2010	01.2	4.2	Added support for Diamond software throughout.
			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.”
May 2011	01.3	5.0	Added Multiplier Multiplexing Factor.
			Added support for multipliers in multiple DSP rows.
			Changed interface timing for certain configurations in LatticeECP3 devices.

Resource Utilization

This appendix gives resource utilization information for Lattice FPGAs using the PCI Express IP core. The IP configurations shown in this chapter were generated using the IPexpress software tool. 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 and ispLEVER help systems. For more information on the Diamond or ispLEVER design tools, visit the Lattice web site at: www.latticesemi.com/software.

LatticeECP Devices

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

IPexpress User-Configurable Mode	Slices	LUTs	Registers	18x18 Multipliers	sysMEM EBRs	f_{MAX} (MHz)
4 channels, 64 taps, multiplier multiplexing 64	130	113	242	1	2	232
1 channel, 32 taps, multiplier multiplexing 1	413	264	806	32	0	179
1 channel, 32 taps, multiplier multiplexing 4	445	655	625	8	0	180

1. Performance and utilization characteristics are generated targeting an LFECP33E-5F672C device using Lattice Diamond 1.2 and Synplify Pro D-2010.03L-SP1 software. Performance may vary when using a different software version or targeting a different device density or speed grade within the LatticeECP family.

Ordering Part Number

The Ordering Part Number (OPN) for the FIR Filter IP Core targeting LatticeECP devices is FIR-COMP-E2-U4.

LatticeECP2 and LatticeECP2S Devices

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

IPexpress User-Configurable Mode	Slices	LUTs	Registers	18x18 Multipliers	sysMEM EBRs	f_{MAX} (MHz)
4 channels, 64 taps, multiplier multiplexing 64	122	130	242	1	2	345
1 channel, 32 taps, multiplier multiplexing 1	415	268	806	32	0	288
1 channel, 32 taps, multiplier multiplexing 4	382	530	626	8	0	293

1. Performance and utilization characteristics are generated targeting an LFE2-50E-7F672C device using Lattice Diamond 1.2 and Synplify Pro D-2010.03L-SP1 software. Performance may vary when using this IP core in a different density, speed or grade within the LatticeECP2/S family or in a different software version.

Ordering Part Number

The Ordering Part Number (OPN) for the FIR Filter IP Core targeting LatticeECP2/S devices is FIR-COMP-P2-U4.

LatticeECP2M and LatticeECP2MS Devices

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

IPexpress User-Configurable Mode	Slices	LUTs	Registers	18x18 Multipliers	sysMEM EBRs	f_{MAX} (MHz)
4 channels, 64 taps, multiplier multiplexing 64	122	130	242	1	2	370
1 channel, 32 taps, multiplier multiplexing 1	415	268	806	32	0	293
1 channel, 32 taps, multiplier multiplexing 4	382	530	626	8	0	298

1. Performance and utilization characteristics are generated targeting an LFE2M50E-7F672C device using Lattice Diamond 1.2 and Synplify Pro D-2010.03L-SP1 software. Performance may vary wWhen using this IP core in a different density, speed, or grade within the LatticeECP2M/S family or in a different software version., performance may vary.

Ordering Part Number

The Ordering Part Number (OPN) for the FIR Filter IP Core targeting LatticeECP2M/S devices is FIR-COMP-PM-U4.

LatticeECP3 Devices

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

IPexpress User-Configurable Mode	Slices	LUTs	Registers	DSP Slices	sysMEM EBRs	f_{MAX} (MHz)
4 channels, 64 taps, multiplier multiplexing 64	136	159	250	2	2	302
1 channel, 32 taps, multiplier multiplexing 1	493	54	985	16	0	294
1 channel, 32 taps, multiplier multiplexing 4	474	647	794	5	0	213

1. Performance and utilization characteristics are generated targeting an LFE3-70E-8FN672CES device using Lattice Diamond 1.2 and Synplify Pro D-2010.03L-SP1 software. Performance may vary when using this IP core in a different density, speed or grade within the LatticeECP3 family or in a different software version.

Ordering Part Number

The Ordering Part Number (OPN) for the FIR Filter IP Core targeting LatticeECP3 devices is FIR-COMP-E3-U4.

LatticeXP2 Devices

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

IPexpress User-Configurable Mode	Slices	LUTs	Registers	18x18 Multipliers	sysMEM EBRs	f_{MAX} (MHz)
4 channels, 64 taps, multiplier multiplexing 64	122	130	242	1	2	314
1 channel, 32 taps, multiplier multiplexing 1	415	268	806	32	0	283
1 channel, 32 taps, multiplier multiplexing 4	382	530	626	8	0	261

1. Performance and utilization characteristics are generated targeting an LFXP2-40E-7F672C device using Lattice Diamond 1.2 and Synplify Pro D-2010.03L-SP1 software. Performance may vary when using this IP core in a different density, speed or grade within the LatticeXP2 family or in a different software version.

Ordering Part Number

The Ordering Part Number (OPN) for the FIR Filter IP Core targeting LatticeXP2 devices is FIR-COMP-X2-U4.