Lattice Semiconductor Corporation

LatticeCORE™

2D FIR Filter IP Core User's Guide

January 2011
IPUG89_01.0

Table of Contents

Introduction

The 2D FIR Filter IP core performs real-time 2D convolution of windowed portions of incoming video frames with coefficient matrices held in internal memory. Its flexible architecture supports a wide variety of filtering operations on LatticeECP2™, LatticeECP2M™, LatticeECP3™ and LatticeXP2™ devices. The highly parameterized design takes advantage of the embedded DSP blocks available in Lattice FPGAs. A simple I/O handshake makes the core suitable for either streaming or bursty input video data. Coefficients may be set at compile time, or updated in system via a simple memory interface.

Quick Facts

Table 1-1 through Table 1-4 give quick facts about the 2D FIR Filter IP core for LattceECP2, LatticeECP2M, LatticeECP3, and LatticeXP2 devices.

Table 1-1. 2D FIR Filter Quick Facts for LatticeECP2

		2D FIR IP configurations		
		5x5 single-rate, 704x480, CIRCULAR, non-separable	**5x5 single-rate, 704x480, None, separable**	**5x5 single-rate, 704x480, XANDY, separable**
Core Requirements	FPGA Family Support	LatticeECP2		
	Minimal Device Needed	LFE2-6E-5T144C		
Resource Utilization	Target device	LFE2-50E-6F484C		
	Data Path Width	8		
	LUTs	2000	580	680
	sysMEM EBRs	2		
	sysDSP Blocks	3	3	2
	Registers	920	460	410
Design Tool Support	Lattice Implementation	Lattice Diamond™ 1.1 or ispLEVER® 8.1SP1		
	Synthesis	Synopsys® Synplify™ Pro for D-2010.03L-SP1		
	Simulation	Aldec® Active-HDL® 8.2 Lattice Edition		
		Mentor Graphics® ModelSim® SE 6.3F		

Table 1-2. 2D FIR Filter Quick Facts for LatticeECP2M

		2D FIR IP configurations		
		5x5 single-rate, 704x480, CIRCULAR, non-separable	5x5 single-rate, 704x480, None, separable	5x5 single-rate, 704x480, XANDY, separable
Core Requirements	FPGA Family Support	LatticeECP2M		
	Minimal Device Needed	LFE2M20E-5F256C		
Resource Utilization	Target device	LFE2M50E-6F484C		
	Data Path Width	8		
	LUTs	2000	580	680
	sysMEM EBRs	2		
	sysDSP Blocks	3	3	2
	Registers	920	460	410
Design Tool Support	Lattice Implementation	Lattice Diamond 1.1 or ispLEVER 8.1SP1		
	Synthesis	Synopsys Synplify Pro for D-2010.03L-SP1		
	Simulation	Aldec Active-HDL 8.2 Lattice Edition		
		Mentor Graphics ModelSim SE 6.3F		

Table 1-3. 2D FIR Filter Quick Facts for LatticeECP3

		2D FIR IP configurations		
		5x5 single-rate, 704x480, CIRCULAR, non-separable	5x5 single-rate, 704x480, None, separable	5x5 single-rate, 704x480, XANDY, separable
Core Requirements	FPGA Family Support	LatticeECP3		
	Minimal Device Needed	LFE3-17EA-6FTN256CES		
Resource Utilization	Target device	LFE3-17EA-7FN484C		
	Data Path Width	8		
	LUTs	2160	630	630
	sysMEM EBRs	2		
	sysDSP Blocks	3	3	2
	Registers	930	460	410
Design Tool Support	Lattice Implementation	Lattice Diamond 1.1 or ispLEVER 8.1SP1		
	Synthesis	Synopsys Synplify Pro for D-2010.03L-SP1		
	Simulation	Aldec Active-HDL 8.2 Lattice Edition		
		Mentor Graphics ModelSim SE 6.3F		

Table 1-4. 2D FIR Filter Quick Facts for LatticeXP2

		2D FIR IP configurations		
		5x5 single-rate, 704x480, CIRCULAR, non-separable	**5x5 single-rate, 704x480, None, separable**	**5x5 single-rate, 704x480, XANDY, separable**
Core Requirements	FPGA Family Support	LatticeXP2		
	Minimal Device Needed	LFXP2-5E-5M132C		
Resource Utilization	Target device	LFXP2-40E-6F484C		
	Data Path Width	8		
	LUTs	2000	580	680
	sysMEM EBRs	2		
	sysDSP Blocks	3	2	2
	Registers	920	460	410
Design Tool Support	Lattice Implementation	Lattice Diamond 1.1 or ispLEVER 8.1SP1		
	Synthesis	Synopsys Synplify Pro for D-2010.03L-SP1		
	Simulation	Aldec Active-HDL 8.2 Lattice Edition		
		Mentor Graphics ModelSim SE 6.3F		

Features

- Single color plane
- Single-rate, interpolating, and decimating filter configurations
- Input frame size set at compile-time
- Static or dynamic zoom and pan
- User-specified 2D convolution kernel
- Separable and non-separable kernel support
- Kernel symmetry optimization
- Updatable coefficients

Functional Description

Key Concepts

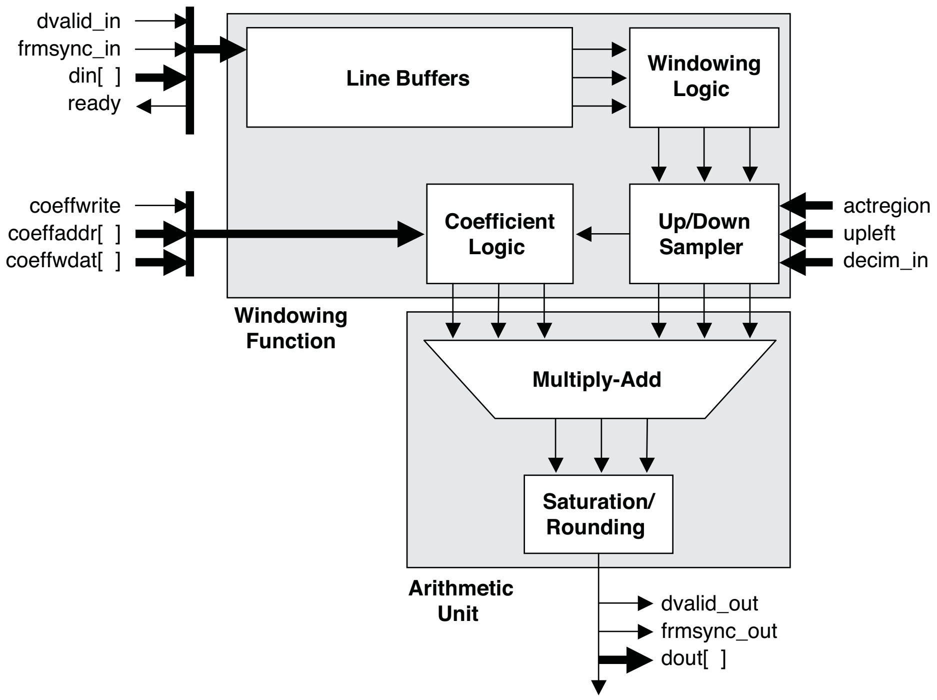
Two-dimensional video filtering is the process of producing pixels in an output frame using the values of pixels in an input frame by calculating each new pixel value as a weighted sum of nearby original pixel values. The number of original pixel values and their weights depends on the filtering algorithm employed. The set of weights is referred to as the "filter kernel" or "coefficient set". Coefficient values depend on the type of filtering operation required.

Interpolating filters insert new pixels in between the pixels in the incoming frame and calculate their values using original pixel values. In general, including more original pixels in the calculation results in a higher quality result, but requires more FPGA resources. Conversely, decimating filters drop unneeded input pixels. Output pixel values are calculated using all the original pixel values in the convolution window. The Lattice 2D FIR Filter IP core allows different interpolation and decimation factors for the horizontal and vertical dimensions.

Block Diagram

The high-level architecture of the 2D FIR Filter IP core is shown in Figure 2-1.

Figure 2-1. 2D FIR Filter IP Core Block Diagram



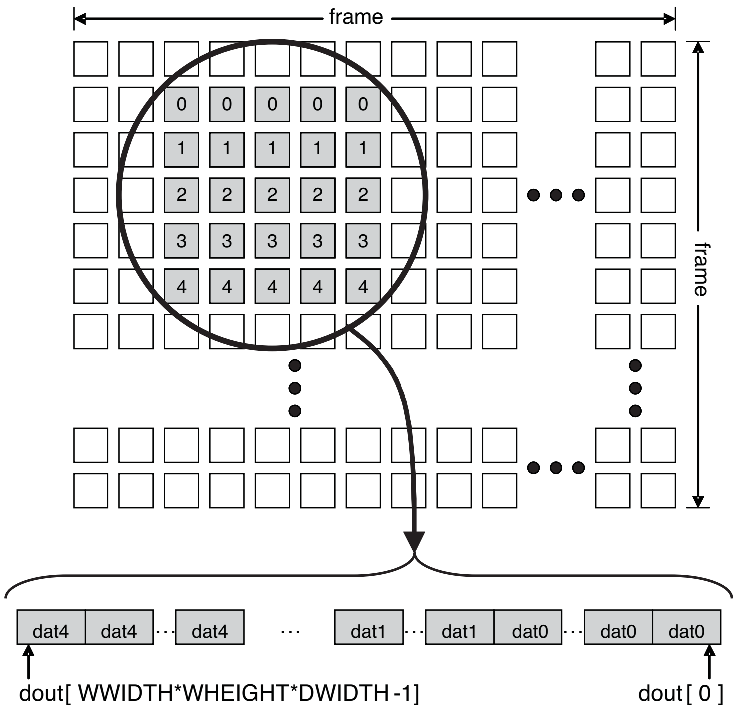
Input data is stored in line buffers, then passed to windowing logic for edge mode handling and data alignment. Next, the up/down sampler decides which pixels, new and existing, will appear in the output frame, and fetches the appropriate coefficients from memory (more details in the section "Coefficients" on page 8). Optional control inputs allow real-time specification of the portion of the input frame used to generate output pixels (referred to as the "active region"), as well the down-sampling (decimation) factor. Coefficient values may also be updated dynamically. The combination of updatable active region, decimation factor, and coefficient values provides dynamic zoom and pan capability.

Windowed data and coefficients are sent to the arithmetic unit, which multiplies the data values by their corresponding coefficients and sums the multiplication results. Depending on the configuration of the core, the multiply-add operations may be distributed over several clock cycles, saving arithmetic resources.

Coefficients

The 2D-FIR Filter works by walking a window of fixed size (specified at compile time) through the incoming video frame, illustrated in Figure 2-2.

Figure 2-2. Windowing Function

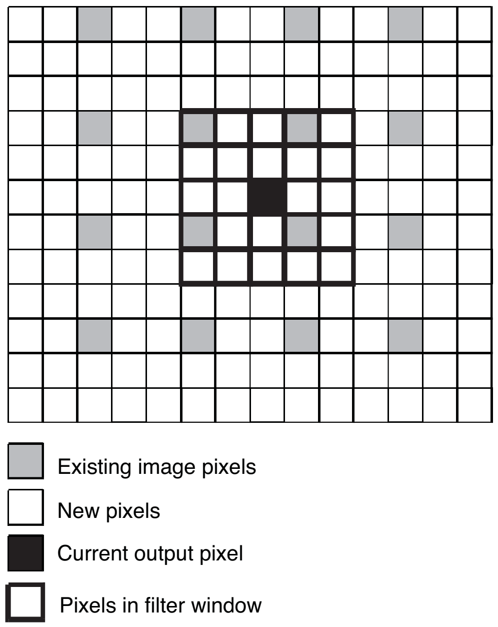


Each pixel value is multiplied by a corresponding coefficient. The memory location of the coefficient for a given pixel is determined by the window dimensions, the interpolation or decimation values, and, perhaps, the symmetry and separability of the coefficient set, as well as the minimum interval between input samples.

Interpolation and Decimation

Interpolation is the process of expanding the size of an image by inserting new pixels between existing ones and calculating their values using the values of nearby existing pixels. The concept is illustrated in Figure 2-3.

Figure 2-3. Interpolation Example



In the example illustrated in Figure 2-3, nine output pixels are generated for every input pixel (the interpolation factor is 3). The example 5-by-5 filter window would normally require 25 multiply-accumulate operations to compute the value of a single output pixel, but because there are only four existing pixels in the window, only four multiply-accumulates are needed. The windowing module, then, outputs the four values needed for calculating a particular output pixel, along with their corresponding filter coefficient values. The subset of the kernel coefficients needed is dependent on the position, or phase, of the output pixel, making this a "polyphase" interpolator.

All coefficients for each phase of a polyphase interpolator occupy a single row in coefficient memory. For the configuration illustrated, there will be nine coefficient memory rows, each with four coefficient values. **The coefficient address is the concatenation of the row and column addresses.**

Decimation is the inverse operation. All input pixel values are shifted into the windowing module's line buffers, but output data and coefficient values are generated only every DECIMY rows and DECIMX columns. For decimating filters, optimization is possible in the follow-on arithmetic units, since processing is allowed to span multiple clock cycles. The number of clock cycles available for processing is DECIMY*DECIMX. The coefficients for each clock cycle occupy a single memory row. A scaler down-sampling by 3 in both X and Y will have 9 rows of coefficient memory.

A further optimization is possible for decimating filters using the MULTICYCLE parameter, configured in the IPexpress™ GUI by the "Minimum input interval" setting. Setting this value to the minimum number of clock cycles between input pixel values allows the output pixel calculation to be spread over more clock cycles, further reducing arithmetic resource requirements. The number of clock cycles available for filter calculations is DECIMY*DECIMX*MULTICYCLE, and the coefficient memory will have that number of memory rows.

Coefficient Generation

The coeffgen.tcl script in the $ip_install/scripts directory will map a coefficient set to appropriate memory locations based on a number of input parameters. Two files are needed: a parameter file and a coefficient file. The format for the parameter file is as follows:

```
windowwidth=5
windowheight=5
separable=0
symmetry=0
x_eq_y=0
interpx=2
interpy=2
decimxmin=1
decimymin=1
multicycle=1
coefffile=path_to_coefficient_file
coefftype=Floating point
ctype=UNSIGNED
cwidth=8
cpoints=8
filter_type=GAUSSIAN
param=1.5
```

windowwidth and windowheight define the window dimensions. separable is 1 for separable kernels, 0 otherwise. symmetry values are: 0 for NONE; 1 for XONLY; 2 for YONLY; 3 for XANDY.

interpx and interpy are the horizontal and vertical interpolation factors (1 means no interpolation). decimxmin and decimymin are the minimum decimation factors (for dynamically alterable decimation factors, the minimum values determine the memory mapping; for fixed decimation, these are simply the decimation values). multicycle is the minimum number of clock cycles between input pixels.coefffile is the path to the coefficient input file. coefftype is the data type of the coefficients in the input file (currently, only Floating point is supported). ctype is SIGNED or UNSIGNED. cwidth is the number of bits in the output coefficients. cpoints is the number of bits to the right of the binary point in the output coefficients. filter_type may be either CUSTOM, GAUSSIAN, LANCZOS, or BICUBIC. CUSTOM means the values are read from an input file; otherwise, they are generated by the script. The meaning of param depends on the filter type. The meaning of param depends on the filter type, as discussed below:

- GAUSSIAN kernel values (in each dimension) are calculated as follows:

$$G(x) = \frac{e^{-\frac{X^2}{2\sigma^2}}}{\sqrt{2\sigma^2}}$$

- LANCZOS kernel values (in each dimension) are calculated as follows:

$$L(x) = sinc(x)sinc(x/a)$$

- BICUBIC kernel values are set as follows:

$$W(x) = \begin{cases} (a+2)|x|^3 - (a+3)|x|^2 + 1 & \text{for } |x| \le 1 \\ a|x|^3 - 5a|x|^2 + 8a|x| - 4a & \text{for } 1 < |x| < 2 \\ 0 & \text{otherwise} \end{cases}$$

The parameter param sets the value of *a*. Common values are -.5 and -.75.

For custom coefficient sets, the input coefficient file is specified using a coefficients file. The coefficients file is a text file with one coefficient per line. Also,the coefficient sets file must contain all the coefficients, even if the coefficient sets is symmetric.

For non-separable filter, the coefficient sets must be arranged by zigzag order.

An example coefficient sets file in decimal format, 5x5 non-separable filter is given below:

```
0.0481
0.0936
0.1169
0.0936
0.0481
0.0936
0.1824
0.2278
0.1824
0.0936
0.1169
0.2278
0.2844
0.2278
0.1169
0.0936
0.1824
0.2278
0.1824
0.0936
0.0481
0.0936
0.1169
0.0936
0.0481
```

For separable filters, the horizontal vector is first, followed by the vertical vector.

```
0.2193
0.4271
0.5333
0.4271
0.2193
0.2193
0.4271
0.5333
0.4271
0.2193
```

The command line for executing coeffgen.tcl is

```
>> tclsh coeffgen.tcl –lpc lpcFile –txt txtFile –map mapFile
```

where:

- lpcFile is the path to the parameter file,
- txtFile is the path to an output text file containing the filter coefficients, and

- mapFile is the path to an output file containing the coefficient memory map.

The lpcFile argument is required. The txtFile is in a format appropriate for use as the “Coefficients file” input in the 2D-FIR Filter IPexpress GUI for coefficient memory initialization. The IP generation scripts provide the memory mapping function. The mapFile contains row addresses and coefficient values after memory mapping is performed. The intended use is as a guide for in-system loading of coefficients via the coefficient update bus.

Dynamic Zoom and Pan

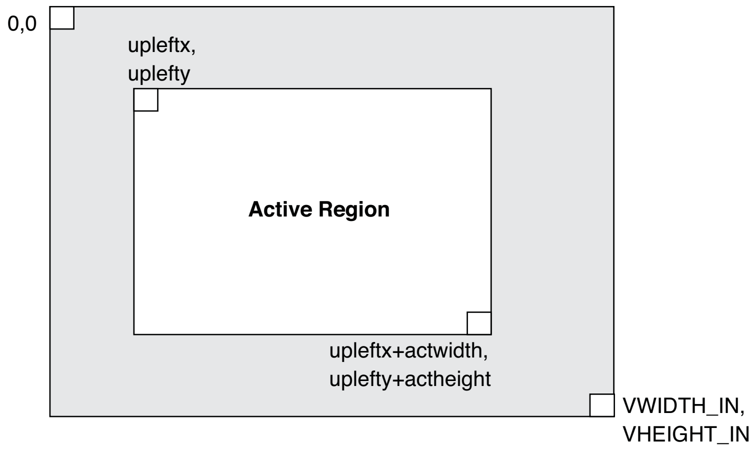
The 2D FIR Filter may be configured to allow the user to dynamically alter the decimation factors, as well as the coordinates of the active region of the input frame. The combination enables a dynamic zoom and pan capability.

With dynamic decimation enabled at core generation time, two optional ports become available: decimx_in and decimy_in. The values on these ports determine the 2D FIR Filter’s decimation factors. The effective decimation factor will be 1 greater than the port value. The maximum decimation factors selected in the 2D FIR Filter’s IPexpress GUI determine the size of the ports.

For decimating filters, the minimum decimation factors determine the number of multipliers that will be available in the hardware. For example, a 5x5 non-symmetric, non-separable kernel must perform 25 multiplies per output. However, if the decimation factors will always be at least 2, there will be at least 4 clock cycles available to calculate an output pixel value. The number of multipliers required will be ceil (25/4), or 7. For interpolating filters, the number of multipliers is a function of the interpolation factors and window size.

The active region concept is illustrated in Figure 2-4.

Figure 2-4. Active Region



The upleftx and uplefty ports set the coordinates of the first pixel in the input frame that will have a corresponding pixel in the output frame. The actwidth and actheight ports determine the region of pixels in the input frame that will have corresponding pixels in the output frame. For example, if the scaling factors in X and Y are 3/2, in order to maintain a constant frame size from input to output, the active region must be 2/3 the width and height of the input frame. The interpolation/decimation factors and the active region dimensions define the zoom characteristics of the scaler. The upleft coordinates provide the pan capability. All three sets of inputs – upleft, active region, and decimation factors – are synchronized internally and delivered to the core logic at the appropriate time to avoid anomalies when moving from frame to frame.

Primary I/O

Table 2-1. Primary I/O

Port	Size	I/O	Description
Global Signals			
clk	1	I	System clock
rstn	1	I	System wide asynchronous active-low reset signal
ce	1	I	Active high clock enable (optional)
sr	1	I	Active high synchronous reset (optional)
Video Input			
ready	1	O	Core is ready for input
dvalid_in	1	I	Input valid
frmsync_in	1	I	Current pixel is at row 0, column 0
din	4 - 24	I	Pixel data in
Video Output			
dvalid_out	1	O	Output valid
frmsync_out	1	O	Current output pixel is at row 0, column 0
dout	4 - 24	O	Pixel data out

Interface Descriptions

Video Input/Output

The 2D FIR Filter uses a simple handshake to pass pixel data into the core. The core asserts its ready output when it is ready to receive data. When the driving module has data to give the core, it drives the core's dvalid_in port to a '1' synchronously with the rising edge of the clk signal, providing the input pixel data on port din. The frmsync_in input should be driven to a '1' during the clock cycle when the first pixel of the first row in the incoming video frame is active.

Correspondingly, dvalid_out is active when valid output pixel data is available on dout, and frmsync_out marks the first pixel, first row of the output video frame.

Note: Output data for interpolating or decimating filter configurations can be quite bursty. Within a row, output pixels emerge in clusters whose size is determined by the interpolation and decimation factors. The spacing of the clusters may depend on the input data rate. During interpolated rows, the core de-asserts its ready signal, since it uses data already in the line buffers to calculate output pixel rows. Also, during interpolated rows, output data emerges at as high a rate as possible (for given interpolation values), not limited by the input data rate. When using the 2D-FIR Filter IP core in a system with continuously streaming data, FIFOs may be required to remove the burstiness from the data flow.

Coefficient Update

The optional coefficient update port provides a write-only capability for modifying coefficient values. The port is synchronous with the rising edge of clk. To write a coefficient, set coeffwrite to '1', coeffaddr to the concatenated row/column address of the coefficient to be written, and set coeffwdat to the value to be written.

If the core was configured with double coefficient memories, all write operations affect only the standby memory, and the active coefficient values are not affected. To make the standby coefficient set active, set the coeffswap input to a '1'. The coeffswap_pending output will be a '1' on the next clock cycle, and remain in that state until the active/standby swap occurs (sometime during the next frame).

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 19 for a description of how to generate the IP.

The 2D FIR 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 provides the list of user configurable parameters for the 2D FIR IP core.

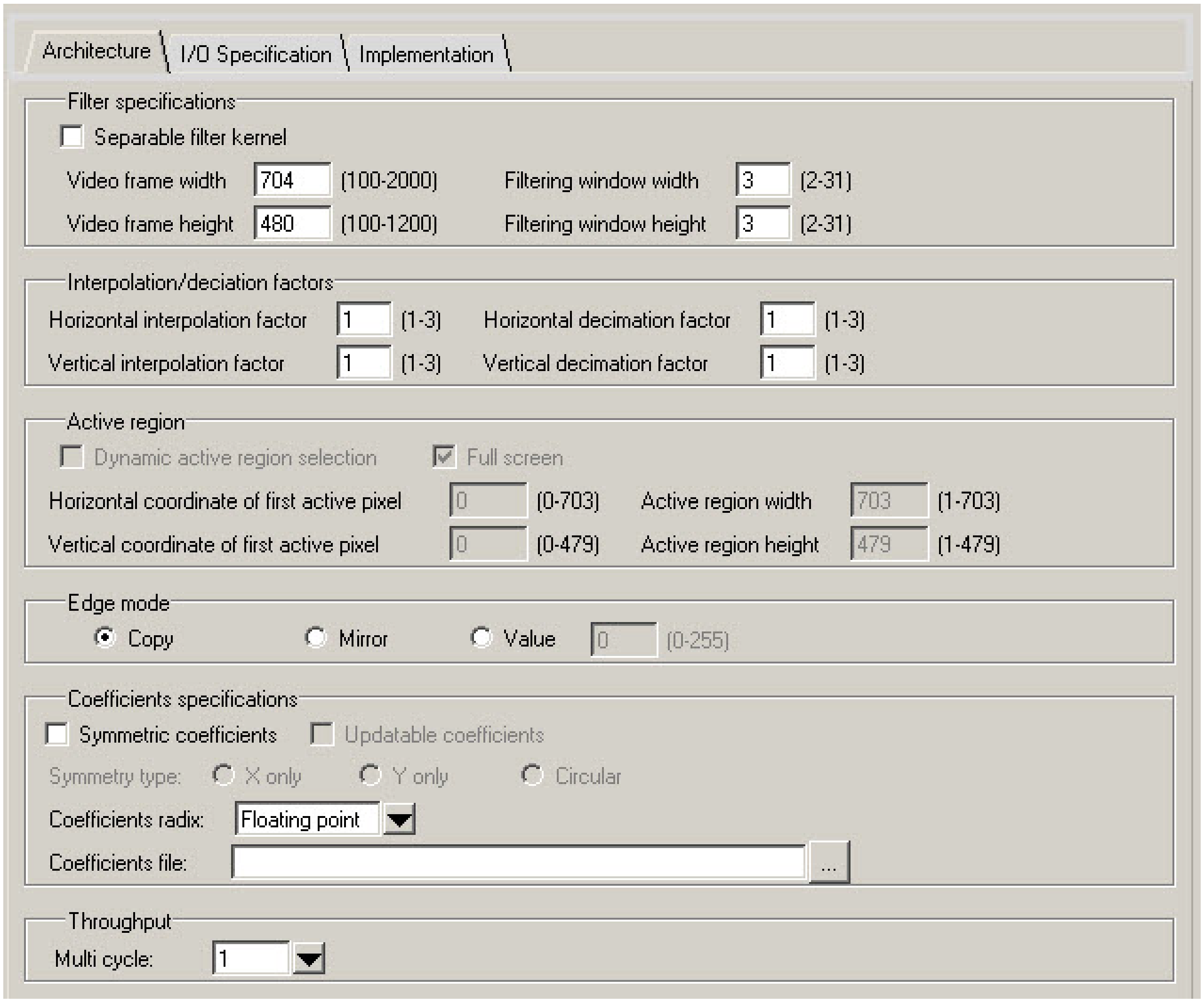
Table 3-1. 2D FIR Filter IP Core Parameters.

Parameter	Range	Default
Data Input Width	4-24	8
Video Frame Width	100-2000	704
Video Frame Height	100-1200	480
Coefficients Width	4-24	8
Coefficients Type	SIGNED or UNSIGNED	UNSIGNED
Filtering Window Width	1-31	3
Filtering Window Height	1-31	3
Edge Mode	VALUE, COPY, MIRROR	COPY
Edge Value	0 - (1<<DWIDTH)-1	
Horizontal Interpolation Factor	1 - WWIDTH	1
Vertical Interpolation Factor	1 - WHEIGHT	1
Horizontal Decimation Factor	1 - WWIDTH	1
Vertical Decimation Factor	1 - WHEIGHT	1
Multi Cycle	1-31	1
Separable Filter Kernel	0 or 1	0
Symmetry Type	NONE, XONLY, YONLY, XANDY/CIRCULAR	NONE
Updatable Coefficients	0 or 1	0
Coefficients Type	REG, ROM, RAM	REG

Architecture Tab

The Architecture tab, shown in Figure 3-1, provides settings for video frame and window parameters, coefficients, scaling factors, and windowing function implementation options.

Figure 3-1. Architecture Tab



Filter specifications

☐ Separable filter kernel

Video frame width	704	(100-2000)	Filtering window width	3	(2-31)
Video frame height	480	(100-1200)	Filtering window height	3	(2-31)

Interpolation/deciation factors

Horizontal interpolation factor	1	(1-3)	Horizontal decimation factor	1	(1-3)
Vertical interpolation factor	1	(1-3)	Vertical decimation factor	1	(1-3)

Active region

☐ Dynamic active region selection ☑ Full screen

Horizontal coordinate of first active pixel	0	(0-703)	Active region width	703	(1-703)
Vertical coordinate of first active pixel	0	(0-479)	Active region height	479	(1-479)

Edge mode

[✓ Copy] Mirror Value 0 (0-255)

Coefficients specifications

☐ Symmetric coefficients ☐ Updatable coefficients

Symmetry type: X only Y only Circular

Coefficients radix: Floating point

Coefficients file:

Throughput

Multi cycle: 1

Filter Specifications

This section provides settings that define the input frame size, and whether the filter kernel is separable.

Separable Filter Kernel
This checkbox determines whether the filter kernel is separable.

Video Frame Width
This parameter defines the input video frame size width.

Video Frame Height
This parameter defines the the input video frame size height.

Filtering window width
This parameter defines the size of the filtering window width.

Filtering window height
This parameter defines the size of the filtering window height.

Interpolation/Decimation Factors

Horizontal interpolation factor
This parameter provide the settings horizontal interpolation.

Vertical interpolation factor
This parameter provide the settings vertical interpolation.

Horizontal and **Vertical decimation** provide the decimation factors for fixed decimation.

Active Region

Dynamic active region
This checkbox enables/disables the dynamic active region feature.

Full screen
This checkbox automatically sets the active region to the full frame size. Unchecking it allows a user-defined active region.

Edge Mode

This section selects between Copy (use the value of the pixels at the edge); Mirror (use the values of pixels the same distance from the edge); and Value (use a fixed value for pixels straddling the boundary).

Coefficients Specifications

Symmetric coefficients
This checkbox enables selection of the coefficient symmetry type.

Updatable coefficients
This checkbox allows in-system setting of coefficient memory.

Symetry Type
if symmetry is set, optimization reduces the number of multiplications that need to be performed in the hardware. Horizontal coefficient is symmetric if Xonly is set. Vertical coefficient is symmetric if Yonly is set. Both horizontal and vertical coefficients are symmetric if Circular is set for non-separable filter or XandY is set for separable filter.

Coefficients radix
The coefficient values in the file can be in any radix (decimal, hexadecimal or binary) selected by the user. For hexadecimal and binary radices, the numbers must be represented in twos complement form.

For floating point radix, the core will quantize the coefficient sets to integer according to the coefficient width and binary point. For decimal, hex and binary radix, user must provide the quantized coefficients value.

For hex format, the quantized coefficients values must not be greater than the width of coefficients. Such as, if the signed coefficient width is 9, then the hex value can only be 0xx or 1xx. The GUI does not accept Fxx.

For binary format, the length of coefficients value must not be greater than the width of coefficients also.

Coefficients file
This parameter provides coefficient initialization values.

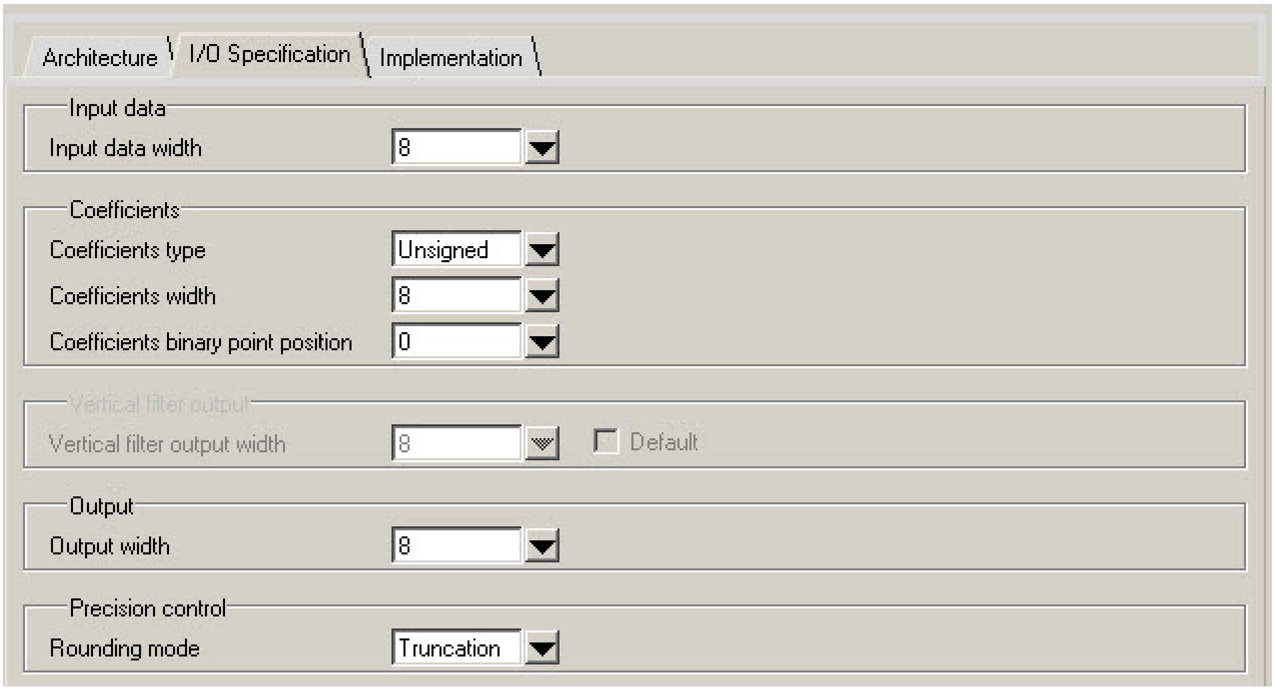
Throughput

Multi cycle
This parameter is the minimum number of clock cycles between input samples. For certain configurations, resource optimization is possible when this number is greater than 1. This does not apply for interpolating filters.

I/O Specification Tab

The I/O Specification tab, shown in Figure 3-2, provides settings for pixel data and coefficient widths, coefficient data type, coefficient binary point, and rounding mode.

Figure 3-2. I/O Specification Tab



Input data
Input data width: 8

Coefficients
Coefficients type: Unsigned
Coefficients width: 8
Coefficients binary point position: 0

Vertical filter output
Vertical filter output width: 8 ☐ Default

Output
Output width: 8

Precision control
Rounding mode: Truncation

Input data

Input data width
This parameter sets the width of the incoming pixel values.

Coefficients

Coefficients type
This parameter may be set to either UNSIGNED or SIGNED.

Coefficients width
This parameter sets the bit width of the coefficients.

Coefficients binary point position
This parameter determines the number of bits in the fractional part of the coefficient.

Rounding mode
This parameter selects between Truncation, Normal (away from zero), and Convergent rounding.

Vertical Filter Output

Vertical filter output width
This parameter sets the size of the vertical filter output for separable filters. Setting the output to values greater than the input data width increases precision at the expense of area.

Output

Output width
This parameter sets the output data width.

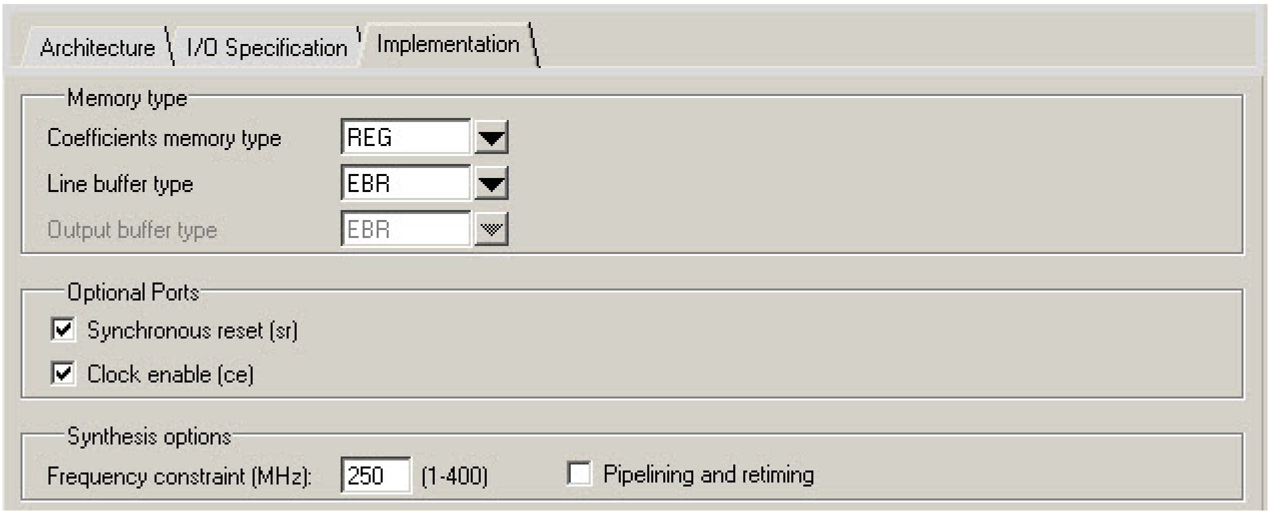
Precision Control

Rounding mode
This parameter selects between Truncation, Normal, and Convergent Rounding.

Implementation Tab

The Implementation tab, shown in Figure 3-3, provides settings for optional ports and synthesis constraints.

Figure 3-3. Implementation Tab



Memory type
Coefficients memory type: REG
Line buffer type: EBR
Output buffer type: EBR

Optional Ports
[✓] Synchronous reset (sr)
[✓] Clock enable (ce)

Synthesis options
Frequency constraint (MHz): 250 (1-400)
[ ] Pipelining and retiming

Memory Type

Coefficient memory type
This parameter selects between REG (register), RAM, and ROM.

Input buffer type
This parameter selects EBR or distributed RAM for the line buffers.

Output buffer type
This parameter selects EBR or distributed RAM for the output buffer (only for decimation filters).

Optional Ports
Checkboxes are provided for optional **synchronous reset** and **clock enable** ports.

Synthesis Options
Frequency constraint
This parameter sets the required clock frequency in MHz.

Pipelining and retiming
This parameter turns these options on in the core generation script. The two options are only valid in the core generation.

IP Core Generation

This chapter provides information on how to generate the 2D FIR Filter 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 2D 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 2D FIR Filter IP core and fully evaluate the core through functional simulation and implementation (synthesis, map, place and route) without an IP license. The 2D 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 "Hardware Evaluation" on page 24 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.

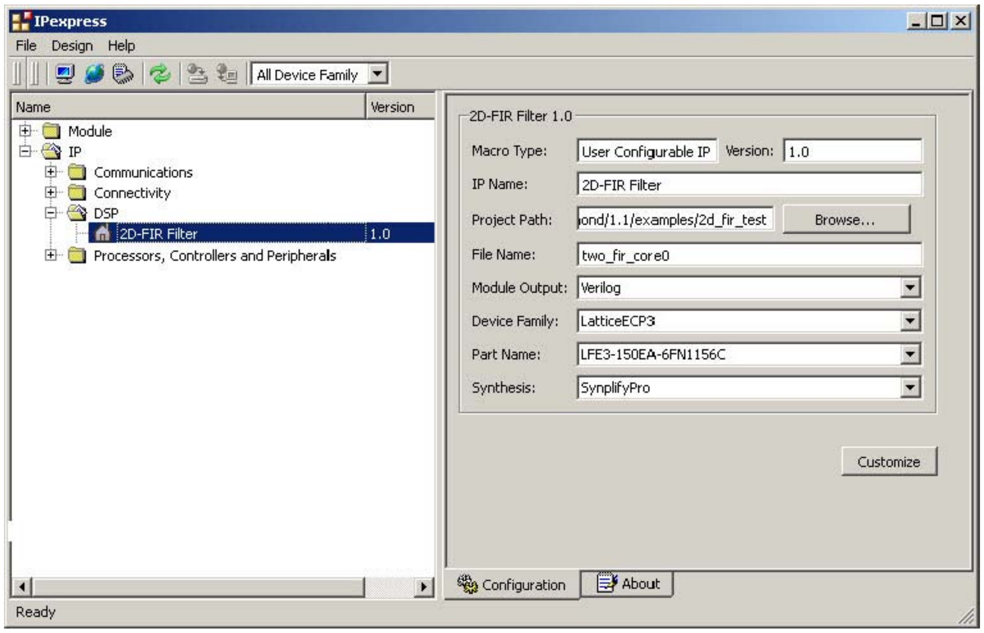
Getting Started

The 2D FIR Filter 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 2D FIR Filter 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 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)



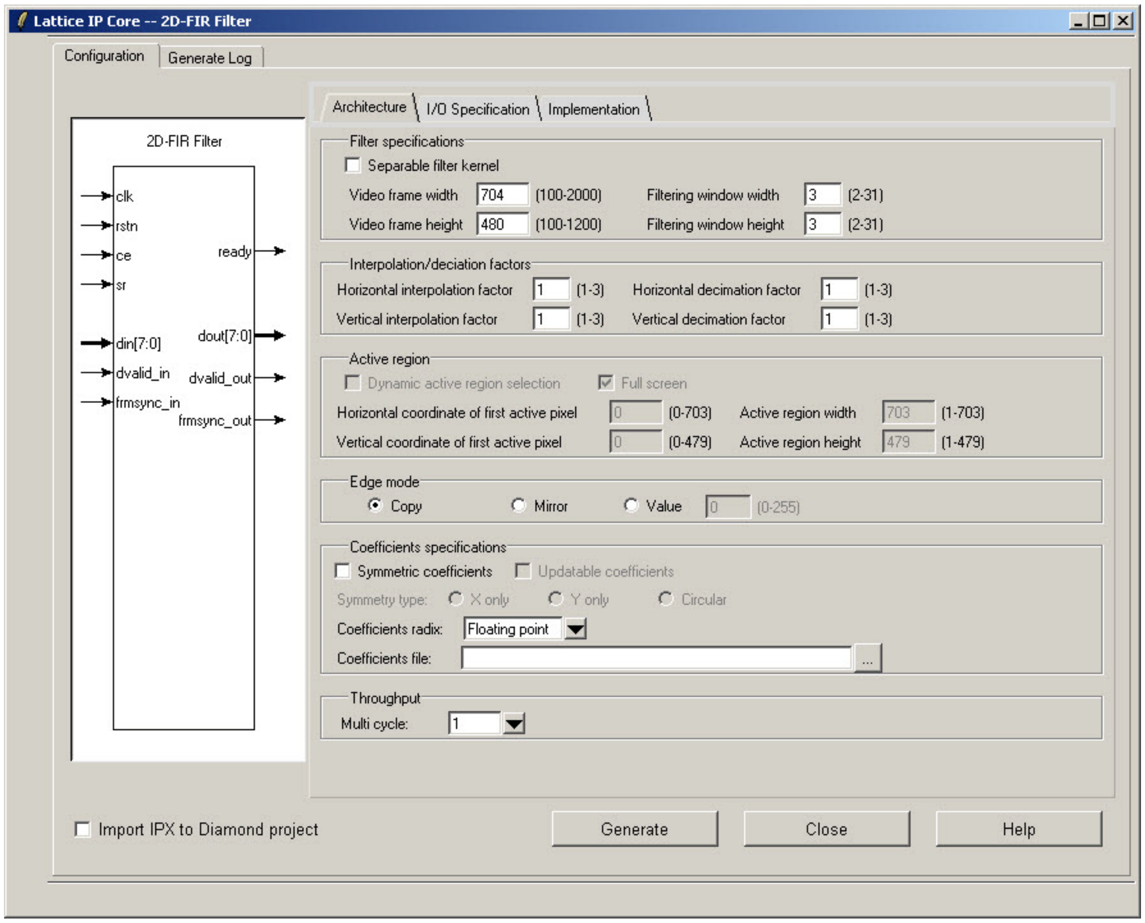
2D-FIR Filter 1.0

Macro Type: User Configurable IP Version: 1.0
IP Name: 2D-FIR Filter
Project Path: ond/1.1/examples/2d_fir_test
File Name: two_fir_core0
Module Output: Verilog
Device Family: LatticeECP3
Part Name: LFE3-150EA-6FN1156C
Synthesis: SynplifyPro

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 2D 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 136 for more information on the 2D FIR Filter IP core parameter settings.

Figure 4-2. Configuration GUI (Diamond Version)



Lattice IP Core -- 2D-FIR Filter

Configuration | Generate Log

Architecture | I/O Specification | Implementation

Filter specifications
- [ ] Separable filter kernel
- Video frame width: 704 (100-2000) — Filtering window width: 3 (2-31)
- Video frame height: 480 (100-1200) — Filtering window height: 3 (2-31)

Interpolation/deciation factors
- Horizontal interpolation factor: 1 (1-3) — Horizontal decimation factor: 1 (1-3)
- Vertical interpolation factor: 1 (1-3) — Vertical decimation factor: 1 (1-3)

Active region
- [ ] Dynamic active region selection — [✓] Full screen
- Horizontal coordinate of first active pixel: 0 (0-703) — Active region width: 703 (1-703)
- Vertical coordinate of first active pixel: 0 (0-479) — Active region height: 479 (1-479)

Edge mode
- [● Copy] Mirror Value 0 (0-255)

Coefficients specifications
- [ ] Symmetric coefficients — [ ] Updatable coefficients
- Symmetry type: X only, Y only, Circular
- Coefficients radix: Floating point
- Coefficients file:

Throughput
- Multi cycle: 1

[ ] Import IPX to Diamond project

IPexpress-Created Files and Top Level Directory Structure

When the user clicks the **Generate** button in the IP Configuration dialog box, the IP core and supporting files are generated in the specified “Project Path” directory. The directory structure of the generated files is shown in Figure 4-3. This example shows the directory structure generated with the 2D FIR Filter IP for LatticeECP3 device.

Figure 4-3. 2D FIR Filter IP Core Directory Structure

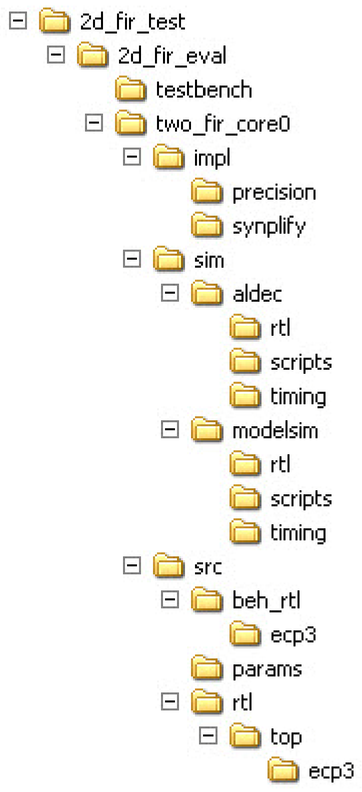


Table 4-1 provides a list of key files and directories created by the IPexpress tool and how they are used. The IPexpress tool creates several files that are used throughout the design cycle. The names of most of the created files are customized to the user's module name specified in the IPexpress tool.

Table 4-1. File List

File	Description
<*username*>.lpc	This file contains the IPexpress tool options used to recreate or modify the core in the IPexpress tool.
<*username*>.ipx	The IPX file holds references to all of the elements of an IP 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*>.ngo	This file provides the synthesized IP core.
<*username*>_bb.v/.vhd	This file provides the synthesis black box for the user's synthesis.
<*username*>_inst.v/.vhd	This file provides an instance template for the 2D FIR Filter IP core.
<*username*>_beh.v/.vhd	This file provides the front-end simulation library for the 2D FIR Filter IP core.

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

Table 4-2. Additional Files

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

Table 4-2. Additional Files (Continued)

<*username*>_gen.log	This is the IPexpress IP generation log file

Instantiating the Core

The generated 2D 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>\2d_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 2D 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 2D 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>\2d_fir_eval\<username>\sim\modelsim\scripts`. The simulation script supporting Aldec evaluation simulation is provided in `\<project_dir>\2d_fir_eval\<username>\sim\aldec\scripts`. Both Modelsim and Aldec simulation is supported via test bench files provided in `\<project_dir>\2d_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>\2d_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>\2d_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

Synthesis support for the 2D FIR Filter IP core is provided for Mentor Graphics Precision or Synopsys Synplify. The 2D FIR Filter 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 top-level file <*username*>_eval_top.v provided in
`\<project_dir>\2d_fir_eval\<username>\src\top`
supports the ability to implement the 2D FIR Filter core in isolation. Push-button implementation of this top-level design with either Synplify or Precision RTL Synthesis is supported via the project files
<username>_eval.ldf (Diamond) or .syn (ispLEVER) located in the
`\<project_dir>\2d_fir_eval\<username>\impl\synplify`
and the
`\<project_dir>\2d_fir_eval\<username>\impl\precision` directories, respectively.

To use this project file in Diamond:

1. Choose **File > Open > Project**.
2. Browse to `\<project_dir>\2d_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>\2d_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 2D 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

Lattice Technical Support

There are a number of ways to receive technical support as listed below.

Online Forums

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

Telephone Support Hotline

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

- For USA and 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

LatticeECP2/M

- HB1003, *LatticeECP2/M Family Handbook*

LatticeECP3

- HB1009, *LatticeECP3 Family Handbook*

LatticeXP2

- DS1009, *Lattice XP2 Datasheet*

Revision History

Date	Document Version	IP Core Version	Change Summary
January 2011	01.0		Initial release.

Resource Utilization

This appendix gives resource utilization information for Lattice FPGAs using the 2D FIR Filter 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.

LatticeECP2 Devices

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

IP Core Configuration	config1	config2	config3
Fmax Requirment(MHz)	125	170	179
Fmax Preference(MHz)	200+100	200+100	200+100
Fmax(MHz)	148	201	211
Start point	1	3	1
Core Utilization			
Reg. #	919	460	401
Total LUT4 #	2000	573	672
SLICES #	1368	422	431
IOB #	23	23	23
EBR #	2	2	2
MULT18X18	9	10	6

1. Performance and utilization data are generated targeting an LFE2-50E-6F484C device using Lattice Diamond 1.1 and Synplify Pro for Lattice 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 LatticECP2 family.

Ordering Part Number

The Ordering Part Number (OPN) for the 2D Edge Detector IP core on LatticeECP2/M devices is 2D-FIR-P2-U1.

LatticeECP2M Devices

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

IP Core Configuration	config1	config2	config3
Fmax Requirment(MHz)	124	169	177
Fmax Preference(MHz)	200+100	200+100	200+100
Fmax(MHz)	147	199	209
Start point	2	1	1
Core Utilization			
Reg. #	919	460	401
Total LUT4 #	2000	573	672
SLICES #	1368	422	431
IOB #	23	23	23
EBR #	2	2	2
MULT18X18	9	10	6

1. Performance and utilization data are generated targeting an LFE2M50E-6F484C device using Lattice Diamond 1.1 and Synplify Pro for Lattice 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 LatticeECP2M family.

Ordering Part Number

The Ordering Part Number (OPN) for the 2D Edge Detector IP core on LatticeECP2/M devices is 2D-FIR-PM-U1.

LatticeECP3 Devices

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

IP Core Configuration	config1	config2	config3
Fmax Requirment(MHz)	109	172	170
Fmax Preference(MHz)	200+100	200+100	200+100
Fmax(MHz)	129	203	201
Start point	2	1	2
Core Utilization			
Reg. #	924	460	404
Total LUT4 #	2155	626	622
SLICES #	1412	429	407
IOB #	23	23	23
EBR #	2	2	2
MULT18X18	9	10	6

1. Performance and utilization data are generated targeting an LFE3-17EA-7FN484C device using Lattice Diamond 1.1 and Synplify Pro for Lattice 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 LatticeECP3 family.

Ordering Part Number

The Ordering Part Number (OPN) for the 2D Edge Detector IP core on LatticeECP3 devices is 2D-FIR-E3-U1.

LatticeXP2 Devices

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

IP Core Configuration	config1	config2	config3
Fmax Requirment(MHz)	98	135	143
Fmax Preference(MHz)	200+100	200+100	200+100
Fmax(MHz)	116	159	169
Start point	3	2	2
Core Utilization			
Reg. #	919	460	401
Total LUT4 #	2000	573	672
SLICES #	1368	422	431
IOB #	23	23	23
EBR #	2	2	2
MULT18X18	9	10	6

1. Performance and utilization data are generated targeting an LFXP2-40E-6F484C device using Lattice Diamond 1.1 and Synplify Pro for Lattice 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 LatticeXP2 family.

Ordering Part Number

The Ordering Part Number (OPN) for the 2D Edge Detector IP core on LatticeXP2 devices is 2D-FIR-X2-U1.