2D Scaler IP Core - Lattice Radiant Software

User Guide

FPGA-IPUG-02133-1.1

June 2021

Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS and with all faults, and all risk associated with such information is entirely with Buyer. Buyer shall not rely on any data and performance specifications or parameters provided herein. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. No Lattice products should be used in conjunction with mission- or safety-critical or any other application in which the failure of Lattice's product could create a situation where personal injury, death, severe property or environmental damage may occur. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
RTL	Register Transfer Level
FPGA	Field Programmable Gate Array
FIFO	First In First Out
RAM	Random Access Memory
EBR	Embedded Block RAM
CPU	Central Processing Unit

1. Introduction

The 2D Scaler IP Core converts input video frames of one size to output video frames of a different size. Its flexible architecture supports a wide variety of scaling algorithms. The highly configurable design takes advantage of the embedded DSP blocks available in Lattice FPGAs. A simple I/O handshake makes the core suitable for either streaming video or burst input video data. In-system input and output frame sizes updating is possible on a frame basis.

1.1. Quick Facts

Table 1.1 presents a summary of the 2D Scaler IP Core.

Table 1.1. Quick Facts

IP Requirements	Supported FPGA Families	CrossLink™-NX, Certus™-NX, CertusPro™-NX
Resource Utilization	Targeted Devices	LIFCL-40, LIFCL-17, LFD2NX-40, LFD2NX-17, LFCPNX-100
	Supported User Interface	Native interface, see Signal Description section.
	Resources	See Table A.1 and Table A.2.
Design Tool Support	Lattice Implementation	IP Core v1.x.x – Lattice Radiant™ software 2.1 or later
	Synthesis	Lattice Synthesis Engine
		Synopsys® Synplify Pro® for Lattice
	Simulation	For a list of supported simulators, see the Lattice Radiant software user guide.

1.2. Features

The key features of 2D Scaler IP Core include:

- Single-color, YCbCr 4:2:2, YCbCr 4:4:4 and RGB video formats
- Serial and parallel processing
- Dynamic parameter updating
- Multi-scaling algorithms
- Configurable number of filter taps for Lanczos coefficient set
- Configurable number of phases for Bicubic and Lanczos coefficient sets
- Configurable pixel data width
- Configurable coefficient width
- Configurable parameter bus width and separate parameter bus clock
- Selectable memory type for line buffer and coefficient memories
- Option for sharing vertical and horizontal filter coefficient memories

1.3. Conventions

1.3.1. Nomenclature

The nomenclature used in this document is based on Verilog HDL.

1.3.2. Signal Names

Signal names that end with:

- _*n* are active low
- _*i* are input signals
- _*o* are output signals
- _io are bi-directional input/output signals

1.4. Attributes

The names of attributes in this document are formatted in title case and italicized (*Attribute Name*).

2. Functional Description

2.1. Overview

Video scaling is the process of calculating values for the pixels in an output frame of dimensions Xout-by-Yout from the values of pixels in an input frame of dimensions Xin-by-Yin. Scaling up by an integer multiple involves inserting new pixels between the original pixels in the input frame and 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 scaling algorithm employed. In general, including more original pixels in the calculation results in a higher quality result (but requires more FPGA resources). Conversely, down-scaling by an integer multiple involves dropping unneeded input pixels. Typically, the drop operation is preceded by a two-dimensional low-pass filter to avoid a jagged appearance in the output frame. The low-pass filtering operation is itself a weighted sum of nearby input pixels. The set of weights is referred to as the filter kernel or coefficient set.

Figure 2.1 shows the functional diagram of the 2D Scaler IP Core.

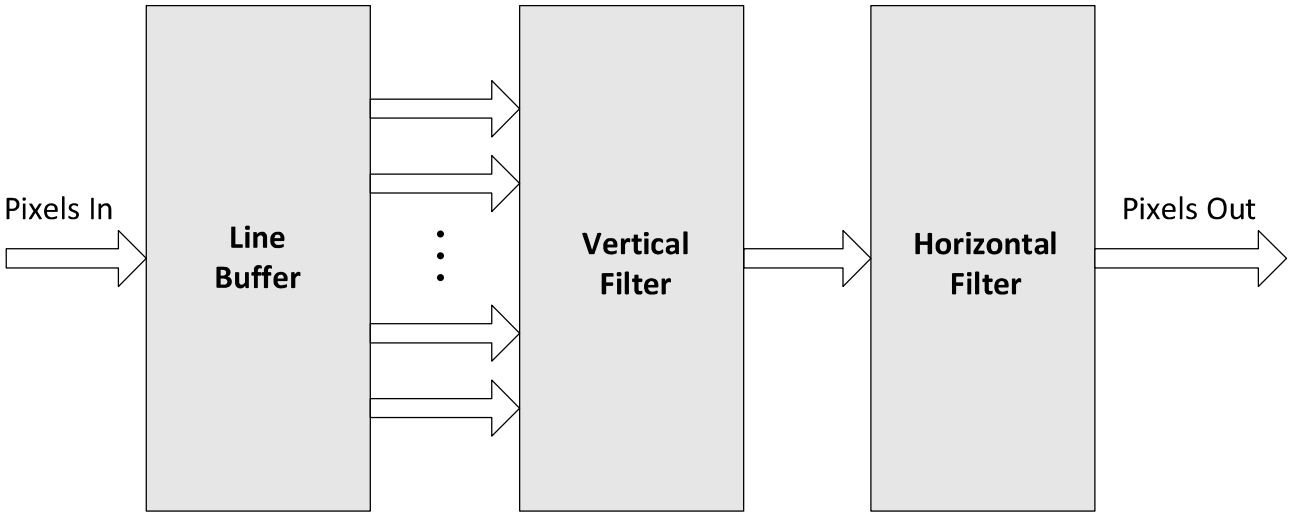


Figure 2.1. 2D Scaler IP Core Functional Diagram

2D Scaler IP Core allows different scaling factors for the horizontal and vertical dimensions. It uses a separable filter architecture which, as depicted by the block diagram, performs the vertical and horizontal scaling in two steps. The input pixel data is first stored in the line buffer. The size of the line buffer is dictated by the number of the vertical filter taps and the maximum input frame width. Pixel data are read out of the line buffer and passed to the vertical filter column by column. Likewise, the vertical filter coefficients are read out of the coefficient memories and passed to the vertical filter for processing along with the pixel data. The row outputs from the vertical filter are then passed to the horizontal filter to generate the output pixels. Output precision control is then performed on the final output pixel value.

It supports in-system re-programming of the input and output frame sizes via a parameter bus. If the IP Core is configured for dynamic parameter updating, then the maximum input and output frame resolutions need to be specified so that line buffer and various counters can be configured appropriately. Also the parameter bus can be configured to run on a separate clock. By default, the parameter bus runs on the input pixel sample clock.

When processing YCbCr 4:2:2 video format, the core averages neighboring pixels' Cb and Cr vectors to construct YCbCr 4:4:4 format for scaling.

Figure 2.2 shows the top-level diagram of 2D Scaler IP Core.

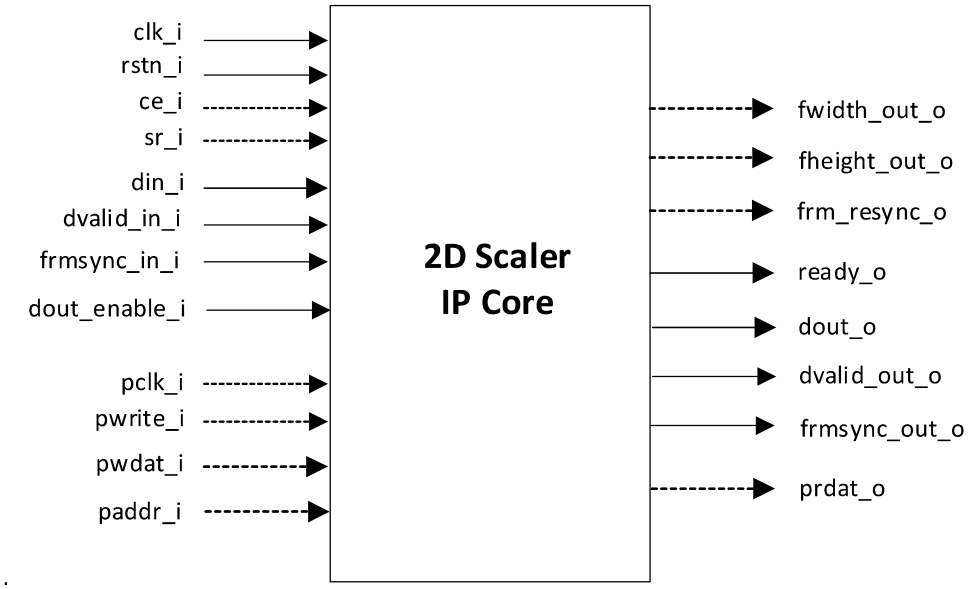
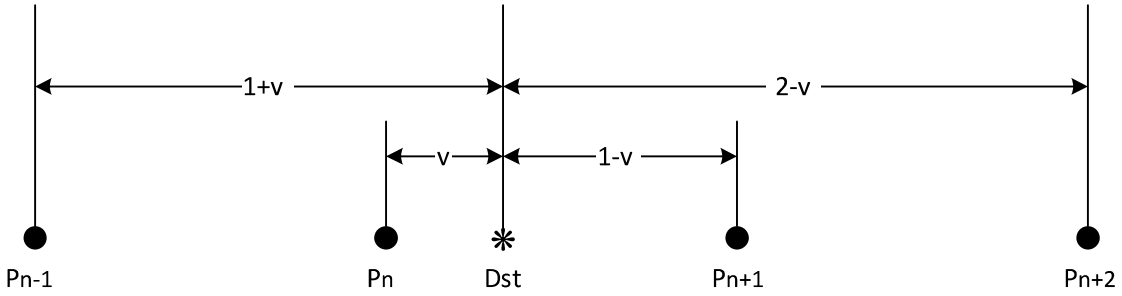


Figure 2.2. 2D Scaler IP Core Block Diagram

2.2. Algorithm and Supported Filter Kernels

Video scaling is a process of generating pixels that do not exist in the original image. In order to compute an output pixel from a set of fixed input pixels, it is necessary to map the output pixel to the input pixel grid and estimate the location of the output pixel relative to the input pixels. The algorithm approximates the output pixel value by using a filter with coefficients weighted accordingly.

A 4-tap Bicubic filter uses four adjacent input pixels to interpolate an output pixel, for example. The spaces between the adjacent pixels are divided into a configurable number of positions called phases so that the output pixel is mapped into one of these positions. The weights of the coefficients are determined by the positions or phases and how close they are to the output pixel. The higher the number of phases, the more accurate is the interpolated output pixel.



$Dst = Pn\text{-}1 * coeff(1+v) + Pn * coeff(v) + Pn+1 * coeff(1-v) + Pn+2 * coeff(2-v)$

● = Source Pixel
❋ = Destination Pixel

Figure 2.3. Scaling Process with 4-Tap Bicubic Filter

The 2D Scaler IP Core supports five adaptive scaling kernels: 1-tap nearest neighbor, 2-tap bilinear, 4-tap bicubic, and configurable multi-tap Lanczos filters. The range for the sizes of the Lanczos filters is from 4 taps to 12 taps. Filter coefficients are generated at compile time when the kernel is configured.

2.3. Interface Descriptions

2.3.1. Video Input/Output

The 2D Scaler IP Core 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_i port to a *1* synchronously with the rising edge of the clk_i signal, providing the input pixel data on port din_i. The frmsync_in_i 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_o is active when valid output pixel data is available on dout_o, and frmsync_out_o marks the first pixel, first row of the output video frame. When the input signal dout_enable_i is asserted, the core outputs video pixels. When dout_enable_i is de-asserted, the core stops generating output pixels after some pipeline delay. The maximum pipeline delay is not greater than 26 clock cycles.

2.3.2. Parameter Register Read/Write Interface

The 2D Scaler IP Core uses a simple register read/write interface to update the input and output frame size control registers for dynamic scaling. The parameter bus interface can be configured to run on a separate clock. It operates at the input pixel clock by default.

When pwrite_i is high, pwdat_i and paddr_i should contain valid data. The contents of all parameter registers are transferred to the core's internal storage when UPDATE is asserted. If a parameter has not been written to before the assertion of UPDATE, its old value is transferred into the internal storage. The prdat_o contains the register read data corresponding to the address value on the paddr_i in the previous clock cycle.

When *Parameter bus width* equals 32, paddr_i[1:0] should be fixed to 0. When *Parameter bus width* equals to 16, paddr_i [0] should be fixed to 0. The *Parameter bus width* is determined by the system CPU data width.

2.3.3. Control Signals and Timing

2.3.3.1. Line and Frame Flushing

The 2D Scaler IP Core can accept pixels line by line and output the pixels line by line. There are several clock cycles required at the end of every line for the core to flush the current output line. Figure 2.4 shows the line flushing cycles. The core cannot accept new input pixels during this time and pulls its ready_o output signal low.

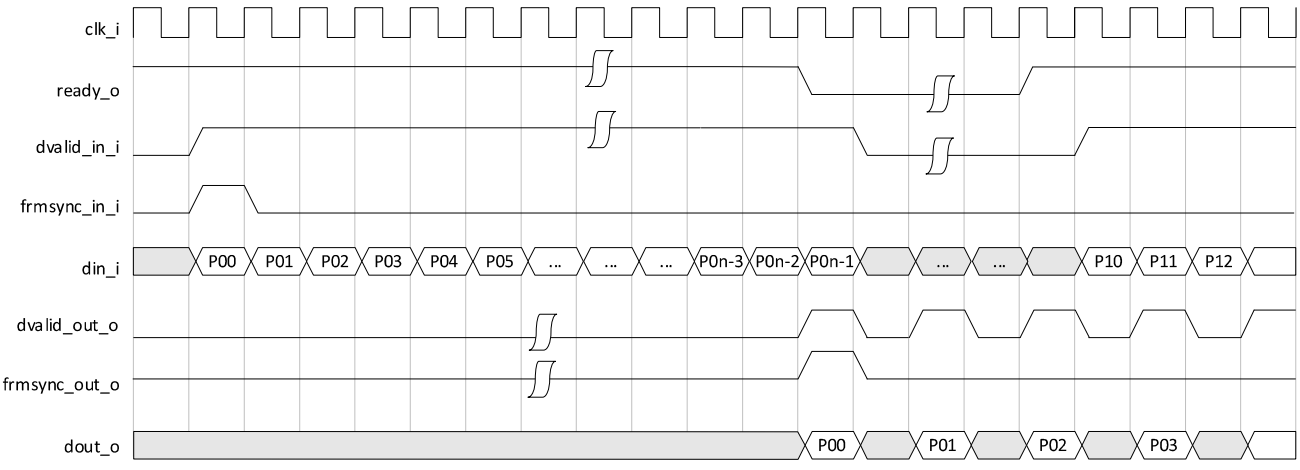


Figure 2.4. Timing Diagram for Line Flushing Cycles

Similarly, there is a frame flushing period at the end of each input frame for the core to complete the current output frame. The number of clock cycles required equals ((number of vertical filter taps) / 2) × (input frame width + line flushing cycle). Figure 2.5 shows the frame flushing cycle.

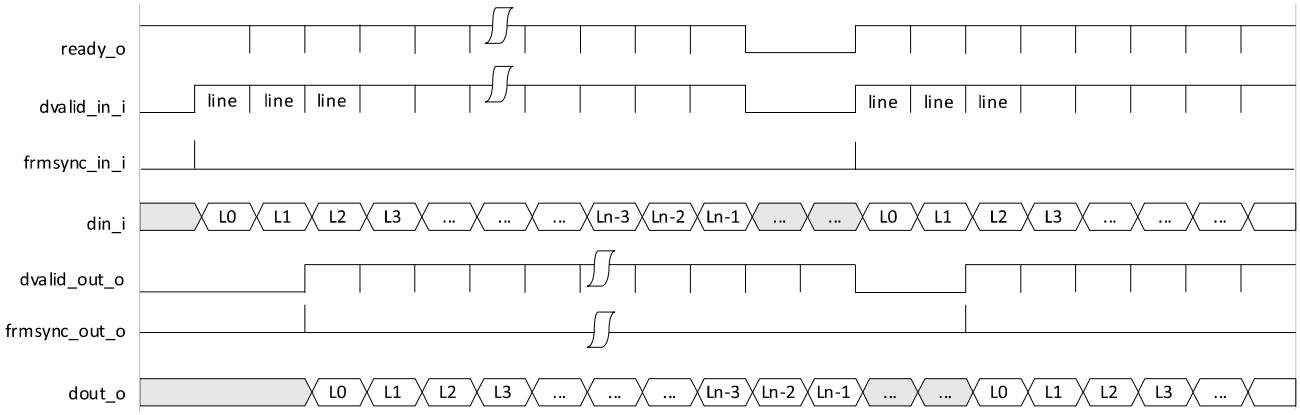


Figure 2.5. Timing Diagram for Frame Flushing Cycles

2.3.3.2. Video Input/Output Timing

The 2D Scaler IP Core supports single color, YCbCr 4:2:2, YCbCr 4:4:4 and RGB video formats. For YCbCr 4:4:4 or RGB video formats, the three planes are interleaved for serial scaling and combined on the din_i and dout_o ports for parallel scaling. Figure 2.6 and Figure 2.7 show the timing of RGB serial scaling and parallel scaling.

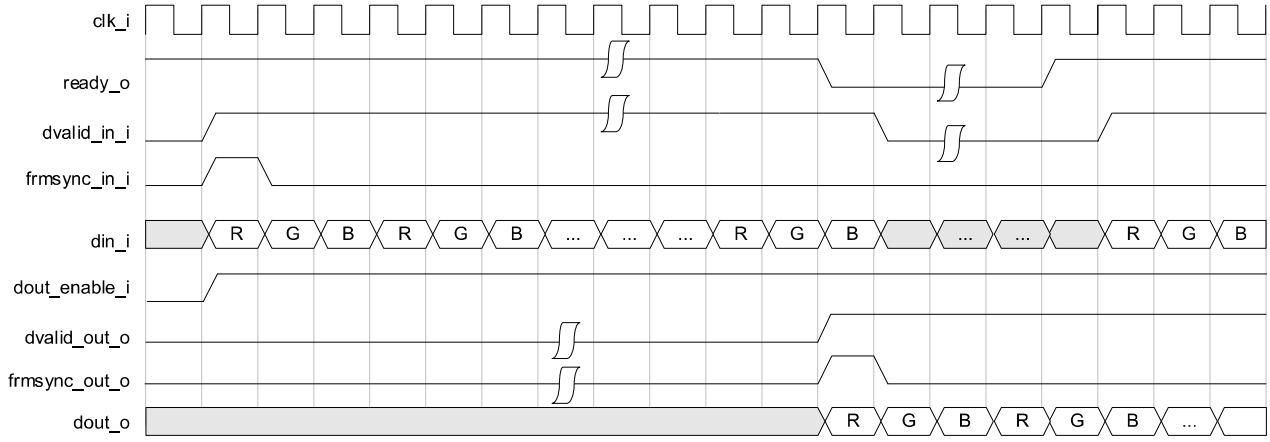


Figure 2.6. RGB Serial Scaling

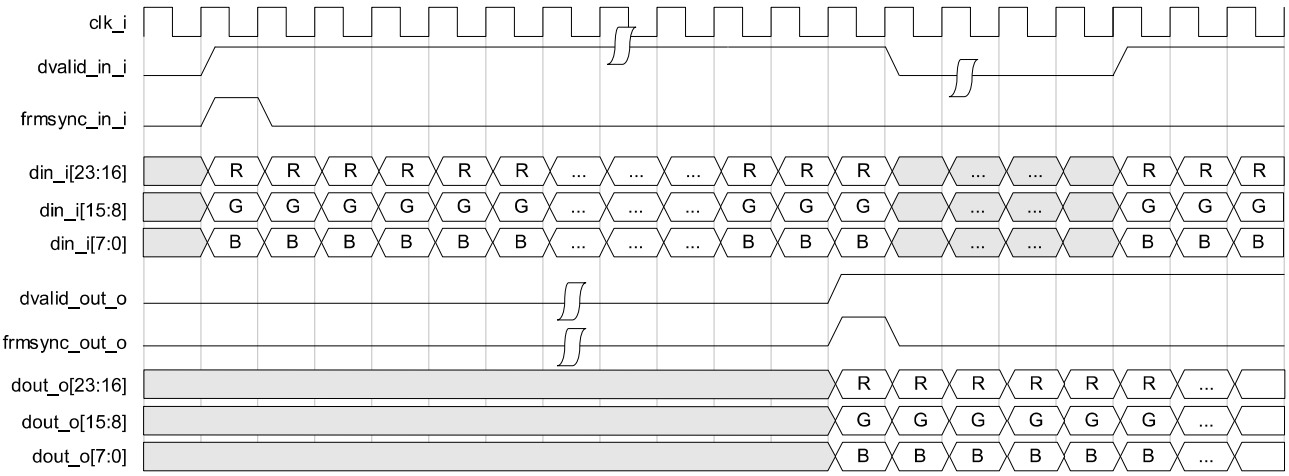


Figure 2.7. RGB Parallel Scaling (8-bit Pixel)

For YCbCr 4:2:2 video serial scaling, the input and output sequence should be Cb, Y, Cr, Y, …. For parallel scaling, the Y plane occupies the upper bits of the din_i and dout_o ports, and the Cb and Cr planes occupy the lower bits. Cb and Cr planes are interleaved in the lower half, and Cb comes before Cr. Figure 2.8 and Figure 2.9 show the timing of YCbCr 4:2:2 serial scaling and parallel scaling.

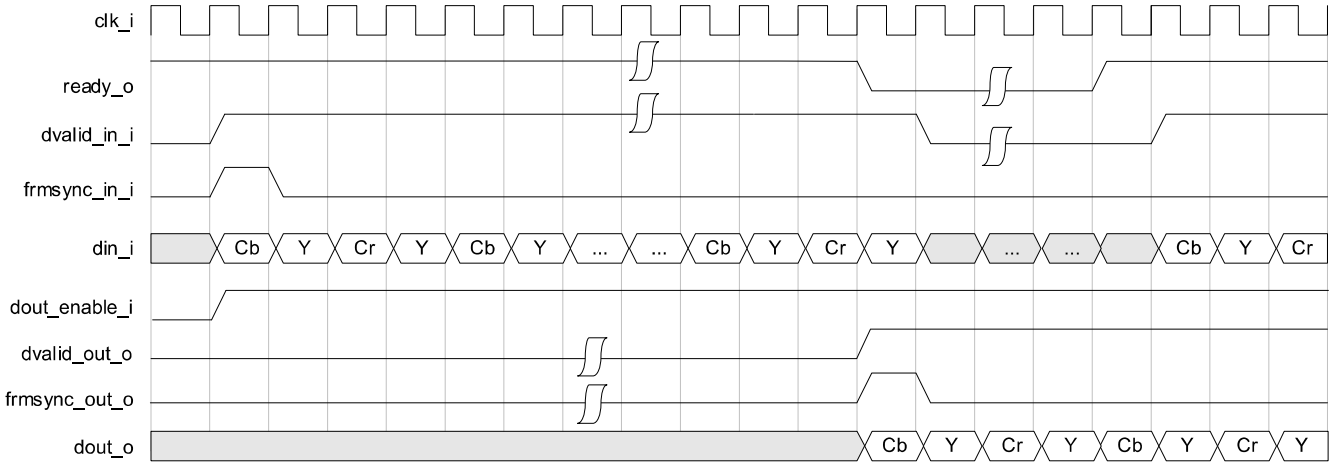


Figure 2.8. YCbCr 4:2:2 Serial Scaling

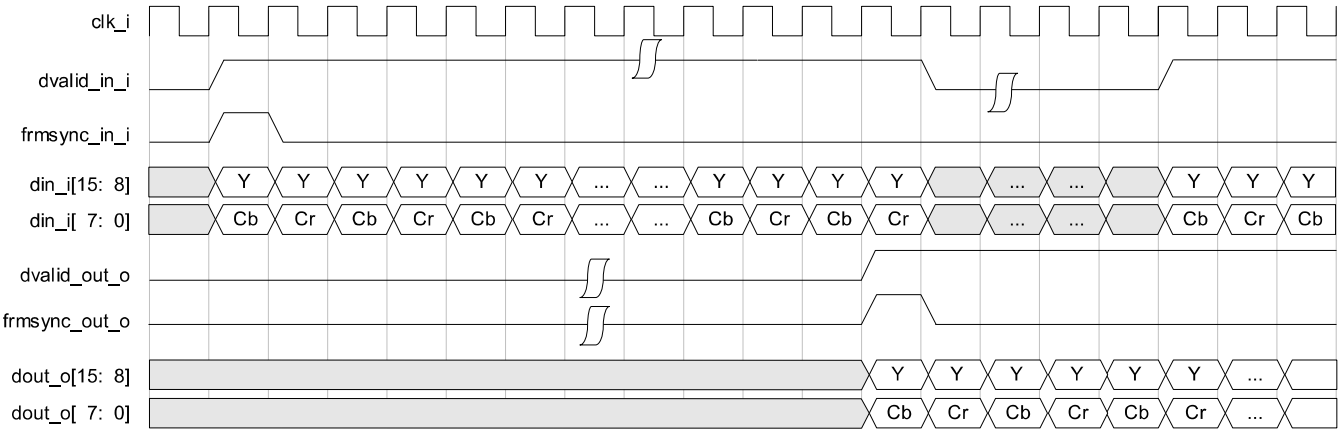


Figure 2.9. YCbCr 4:2:2 Parallel Scaling (8-bit Pixel)

Figure 2.10 shows the dout_enable_i control timing. When the dout_enable_i is de-asserted, the core stops outputting data after the pipeline delay. Similarly, when the dout_enable_i is asserted, the core begins outputting data after a certain pipeline delay. The asserting and de-asserting of dout_enable_i can be used to generate horizontal blank and vertical blank depending on the output video format.

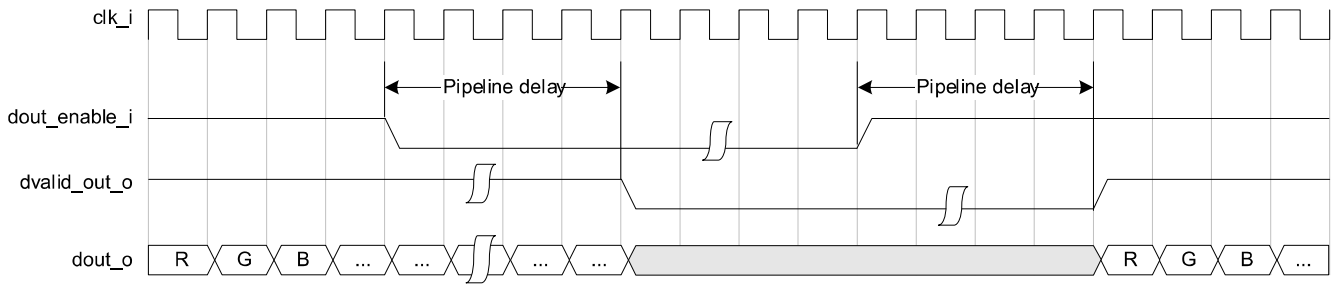


Figure 2.10. dout_enable Control Timing

2.3.3.3. Dynamic Parameter Updating

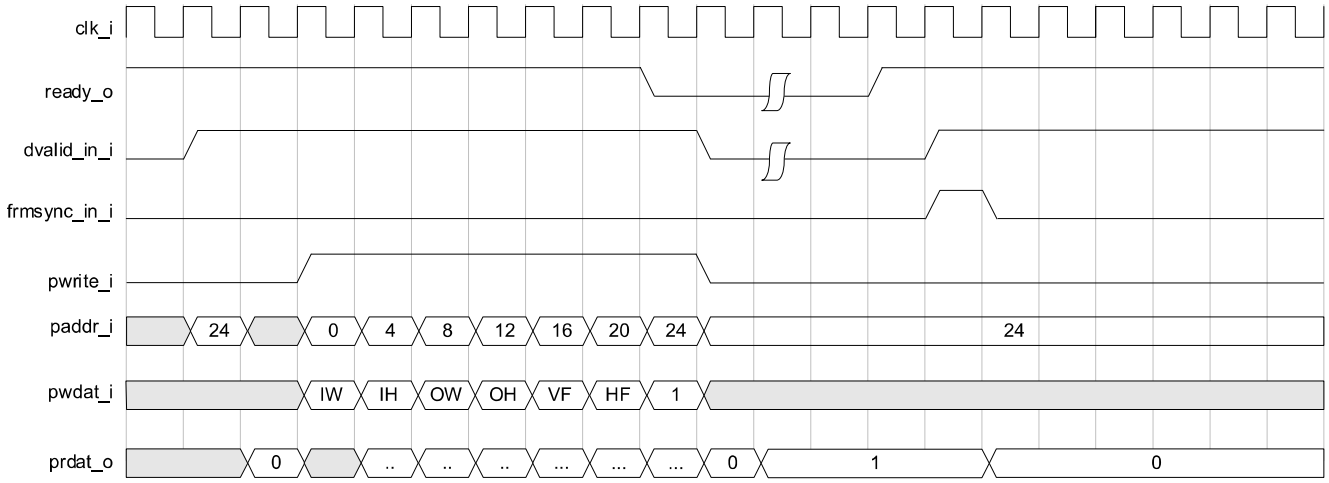


Figure 2.11. Timing Diagram for Dynamic Parameter Updating (Parameter Bus Width = 32)

As shown in Figure 2.11, IW is the input frame width, IH is the input frame height, OW is the output frame width, and OH is the output frame height. VF is the vertical scaling factor and HF the horizontal scaling factor. The parameter registers are writable only when UPDATE register is 0. When the UPDATE register bit is set to 1, the driving block should start the parameter updating at the next active frmsync_in_i and reset the UPDATE register to 0.

2.4. Signal Description

Table 2.1 lists top-level input and output signals and their description.

Table 2.1. 2D Scaler IP Core Signal Description

Port Name	Direction	Bits	Description
Global Signals			
clk_i	Input	1	System clock
rstn_i	Input	1	System asynchronous active-low reset signal
ce_i[1]	Input	1	Active high clock enable.
sr_i[2]	Input	1	Active high synchronous reset.
Video Input			
dvalid_in_i	Input	1	Input valid
frmsync_in_i	Input	1	Input frame sync signal, indicating current input pixel is at row 0, column 0
din_i	Input	2* *Input pixel width* when *Video format* == YCbCr4:2:2 and *Parallel processing*== Checked 3* *Input pixel width* when *Video format* == YCbCr4:4:4 or RGB and *Parallel processing* == Checked *Input pixel width* when *Parallel processing* == Unchecked or *Video format* == Single	Pixel data in
ready_o	Output	1	Signals that core is ready to accept input video frame
Video Output			
dout_enable_i	Input	1	Active high input from down-stream module to enable output data.
dvalid_out_o	Output	1	Signals that the output valid
frmsync_out_o	Output	1	Output frame sync signal, indicating current output pixel is at row 0, column 0
dout_o	Output	2* *Output pixel width* when *Video format* == YCbCr4:2:2 and *Parallel processing* == Checked 3* *Output pixel width* when *Video format* == YCbCr4:4:4 or RGB and *Parallel processing* ==Checked *Output pixel width* when *Parallel processing* == Unchecked or *Video format* == Single	Pixel data out
frm_resync_o[3]	Output	1	Input frame re-sync flag signal
fwidth_out_o[4]	Output	*Output frame width*	Current output frame width
fheight_out_o[4]	Output	*Output frame height*	Current output frame height
Parameter Bus (optional)			
pclk_i[5]	Input	1	Configurable parameter bus clock
pwrite_i[6]	Input	1	Active-high parameter bus write enable
paddr_i[6]	Input	5	Parameter bus address
pwdat_i[6]	Input	*Parameter bus width*	Parameter bus write data
prdat_o[6]	Output	*Parameter bus width*	Parameter bus read data

Notes:

1. Available only when *Clock enable* is Checked.
2. Available when *Synchronous reset* is Checked.
3. Available when *Input frame re-sync flag* is Checked.
4. Available when *Output frame size ports* is Checked
5. Available when *Separate parameter bus clock* is Checked
6. Available only when *Dynamic parameter updating* is Checked.

2.5. Attribute Summary

Table 2.2 provides the list of user selectable and compile time configurable parameters for 2D Scaler IP Core. The parameter settings are specified using the 2D Scaler IP Core user interface in Lattice Radiant software.

Table 2.2. Attributes Table

Attribute	Selectable Values	Default	Dependency on Other Attributes
Architecture			
Frame Dimensions			
Video format	Single color, YCbCr4:2:2, YcbCr4:4:4, and RGB	YCbCr4:2:2	—
Parallel processing	Checked, Unchecked	Checked	Editable when *Video format* is not equal to Single color
Dynamic parameter updating	Checked, Unchecked	Checked	—
(Max.) Input frame width	32-4096	720	—
(Max.) Input frame height	32-4096	576	—
(Max.) Output frame width	32-4096	1280	—
(Max.) Output frame height	32-4096	720	—
Filter Physical Characteristics			
Scaler kernel	Nearest, Bilinear, Bicubic, Lanczos	Lanczos	—
Number of vertical filter taps	4-12	4	Editable only when *Scaler kernel* == Lanczos Fixed to 4 when *Scaler kernel* == Bicubic Fixed to 2 when *Scaler kernel* == Bilinear Fixed to 1 when *Scaler kernel* == Nearest
Number of horizontal filter taps	4-12	4	Editable only when *Scaler kernel* == Lanczos Fixed to 4 when *Scaler kernel* == Bicubic Fixed to 2 when *Scaler kernel* == Bilinear Fixed to 1 when *Scaler kernel* == Nearest
Number of vertical filter phases	16, 32, 64, 128, 256, 512	16	Editable only when *Scaler kernel* == Lanczos *Scaler kernel* == Bicubic Fixed to 16 when *Scaler kernel* == Bilinear or *Scaler kernel* == Nearest
Number of horizontal filter phases	16, 32, 64, 128, 256, 512	16	Editable only when *Scaler kernel* == Lanczos *Scaler kernel* == Bicubic Fixed to 16 when *Scaler kernel* == Bilinear or *Scaler kernel* == Nearest
I/O Specification			
Data Width			
Input pixel width	8-16	8	—
Coefficient width	6-18	9	—
Coefficient type	Signed, Unsigned	Signed	Editable when *Scaler Kernel* != Nearest
Output pixel width	8-16	8	Editable when *Scaler Kernel* != Nearest

Parameter bus			
Parameter bus width	8, 16, 32	32	Editable when *Dynamic parameter updating* == Checked
Separate parameter bus clock	Checked, Unchecked	Unchecked	Editable when *Dynamic parameter updating* is Checked
Optional Ports			
Synchronous reset(sr_i)	Checked, Unchecked	Unchecked	—
Clock enable(ce_i)	Checked, Unchecked	Unchecked	—
Input frame re-sync flag(frm_resync_o)	Checked, Unchecked	Unchecked	—
Output frame size ports(fheight_out_o,fwidth_out_o)	Checked, Unchecked	Unchecked	Editable when *Dynamic parameter updating* == Checked
Precision Control			
Rounding mode	Truncation, Normal, Convergent	Normal	—
Implementation			
Memory Type			
Line buffer type	EBR, Distributed	EBR	—
Vertical coefficient memory type	EBR, Distributed	Distributed	Editable only when *Scaler kernel* == Lanczos *Scaler kernel* == Bicubic
Horizontal coefficient memory type	EBR, Distributed	Distributed	Editable only when *Scaler kernel* == Lanczos *Scaler kernel* == Bicubic
Share vertical and horizontal coefficient memories	Checked, Unchecked	Unchecked	Editable when *Vertical coefficient memory type* and *Horizontal coefficient memory type* is set to EBR
Multiplier type			
Multiplier type	DSP, LUT	DSP	—

Table 2.3. Attributes Description

Attribute	Description
Frame Dimensions	
Video format	Format of video stream.
Parallel processing	Determines whether the core processes video color planes in parallel or not.
Dynamic parameter updating	Determines whether the core supports parameters updating at runtime. When enabled, maximum *Input Frame Width*, maximum *Input Frame Height*, maximum *Output Frame Width* and maximum *Output Frame Height* specify the largest input and output frame sizes the core needs to support. Refer to the Dynamic Parameter Updating section for more information.
(Max.) Input frame width	Width of input video frame
(Max.) Input frame height	Height of input video frame
(Max.) Output frame width	Width of output video frame
(Max.) Output frame height	Height of output video frame
Filter Physical Characteristics	
Scaler kernel	Selects the scaling algorithm of the core.
Number of vertical filter taps	Represents the number of multipliers that may be used by the core for the vertical filter
Number of horizontal filter taps	Represents the number of multipliers that may be used by the core for the horizontal filter
Number of vertical filter phases	Sets the number of phases of the vertical filter

Attribute	Description
Number of horizontal filter phases	Sets the number of phases of the vertical filter
I/O Specification	
Data width	
Input pixel width	Sets the bit width of the incoming pixel values
Coefficient width	Sets the bit width of the coefficient
Coefficient type	Determine whether coefficient is signed or unsigned
Output pixel width	Sets the output pixel bit width
Parameter bus	
Parameter bus width	Sets the bus width of the parameter register's read/write interface
Separate parameter bus clock	Determines whether the core uses a separate clock to run the parameter update interface.
Optional ports	
Synchronous reset (sr_i)	Determines whether the core has a synchronous reset port.
Clock enable (ce_i)	Determines whether the core has a clock enable port.
Input frame re-sync flag	Determines whether the core has a re-sync port.
Output frame size ports	Determines whether the core provides output frame size ports
Precision Control	
Rounding mode	Selects rounding mode for the output pixel value. Normal: Rounds away from zero if the fractional part is exactly one-half. Truncation: Discards all bits to the right of the output's least significant bit. Convergent: Rounds to the nearest integer if the fractional part is exactly one-half.
Implementation	
Memory type	
Line buffer type	Selects memory type for the line buffer implementation
Vertical coefficient memory type	Selects memory type for the vertical coefficient memory
Horizontal coefficient memory type	Selects memory type for the horizontal coefficient memory
Share vertical and horizontal coefficient memories	Determines whether the core uses one memory for both the vertical and horizontal coefficients
Multiplier type	
Multiplier type	Selects the multiplier type to be used on scaling

2.6. Register Description

The 2D Scaler IP Core supports in-system input and output frame sizes updating via a register read/write interface called parameter ports. The parameter registers are listed below on Table 2.4.

Table 2.4. Dynamic Parameter Updating Register Description

Address	Register Name	Size	Access Type	Description
0x0000	FRMWIDTH	32	R/W	Input frame width register – The FRMWIDTH value must be the input frame width minus 1. The minimum value is 31, and the maximum value is the maximum input frame width specified on the IP user interface minus 1. The default value is the maximum value. Input frame width must be an even number for YCbCr4:2:2 for- mat (that is FRMWIDTH must be odd).
0x0004	FRMHEIGHT	32	R/W	Input frame height register – The FRMHEIGHT must be the input frame height minus 1. The minimum value is 31, and the maximum value is the maximum input frame height specified on the IP user interface minus 1. The default value is the maximum value.
0x0008	OUTWIDTH	32	R/W	Output frame width register – The OUTWIDTH must be the output frame width minus 1. The minimum value is 31, and the maximum value is the maximum output frame width specified on the IP user interface minus 1. The default value is the maximum value. Output frame width must be an even number for YCbCr4:2:2 format (that is OUTWIDTH must be odd).
0x000C	OUTHEIGHT	32	R/W	Output frame height register – The OUTHEIGHT must be the output frame height minus 1. The minimum value is 31, and the maximum value is the maximum output frame height specified on the IP user interface minus 1. The default value is the maximum value.
0x0010	VSFACTOR	32	R/W	Vertical scaling factor register – VSFACTOR = ((FRMHEIGHT+1) × (1<<VFCBPWIDTH))/(OUTHEIGHT+1). Where VFCBPWIDTH is the maximum value of log2 (maximum output frame height) and log2 (number of vertical filter phases).
0x0014	HSFACTOR	32	R/W	Horizontal scaling factor register – HSFACTOR = ((FRMWIDTH+1) × (1<< HFCBPWIDTH))/(OUTWIDTH+1). Where HFCBPWIDTH is the maximum value of log2 (maximum output frame width) and log2 (number of the horizontal filter phases).
0x0018	UPDATE	32	R/W	Update parameter enable register – When the writing of the parameter registers is complete, you should set this register to 1 to enable the parameter's status. The default value is 0. When the core updates the new parameters, it resets this register to 0.

The parameter registers can be written to only when the UPDATE register bit is 0. When the UPDATE bit is set to 1, the six parameters inside the core are updated with the new values when the frmsync_in signal is active, indicating a new input frame is arriving. After updating its internal parameters, the core resets the UPDATE bit to 0 to indicate that the parameter registers are now empty and can take on new values.

When dynamic parameter updating is enabled, you need to configure the largest input and output frame sizes the system expects to handle. This is so that the line buffer and various counters within the core can be configured properly. The core uses the default values for the input and output frame sizes to start until the driving block updates the parameters in the subsequent frames. The default values are the maximum input frame size and the maximum output frame size.

For most cases, VFCBPWIDTH equals to log2 (output frame height) for fixed scaler and log2(maximum output frame height) for dynamic scaler. HFCBPWIDTH equals to log2 (output frame width) for fixed scaler and log2 (maximum output frame width) for dynamic scaler. When the number of vertical phases is greater than the maximum output frame height,

the VFCBPWIDTH equals to log2 (number of the vertical filter phases). When the number of horizontal phases is greater than the maximum output frame width, the HFCBPWIDTH equals to log2 (number of horizontal filter phases).

For the nearest neighbor and bilinear kernels, when the VFCBPWIDTH is smaller than the vertical coefficient bit width, its value should be replaced by the vertical coefficient bit width. Similarly, when the HFBPWIDTH is smaller than the horizontal coefficient bit width, its value should be replaced by the horizontal coefficient bit width.

The values of VFCBPWIDTH and HFCBPWIDTH can be found in the generated parameter value file: \<*project_dir*>*scaler_eval*\<*username*>*rtl**params**params.v*.

3. IP Generation and Evaluation

This section provides information on how to generate the 2D Scaler IP Core using the Lattice Radiant software and how to run simulation and synthesis. For more details on the Lattice Radiant software, refer to the Lattice Radiant software user guide.

3.1. Licensing the IP

An IP core-specific license string is required to enable full use of the 2D Scaler IP Core in a complete, top-level design. You can fully evaluate the IP Core through functional simulation and implementation (synthesis, map, place and route) without an IP license string. This IP Core supports Lattice's IP hardware evaluation capability, which makes it possible to create versions of the IP Core, which operate in hardware for a limited time (approximately four hours) without requiring an IP license string. See Hardware Evaluation section for further details. However, a license string is required to enable timing simulation and to generate bitstream file that does not include the hardware evaluation timeout limitation.

3.2. Generation and Synthesis

The Lattice Radiant Software allows you to customize and generate modules and IPs and integrate them into the device's architecture. The procedure for generating the 2D Scaler IP Core in Lattice Radiant software is described below.

To generate the 2D Scaler IP Core:

1. Create a new Lattice Radiant software project or open an existing project.
2. In the **IP Catalog** tab, double-click **Scaler** under the **IP, DSP** category. The **Module/IP Block Wizard** opens as shown in Figure 3.1. Enter values in the **Component name** and the **Create in** fields and click **Next**.

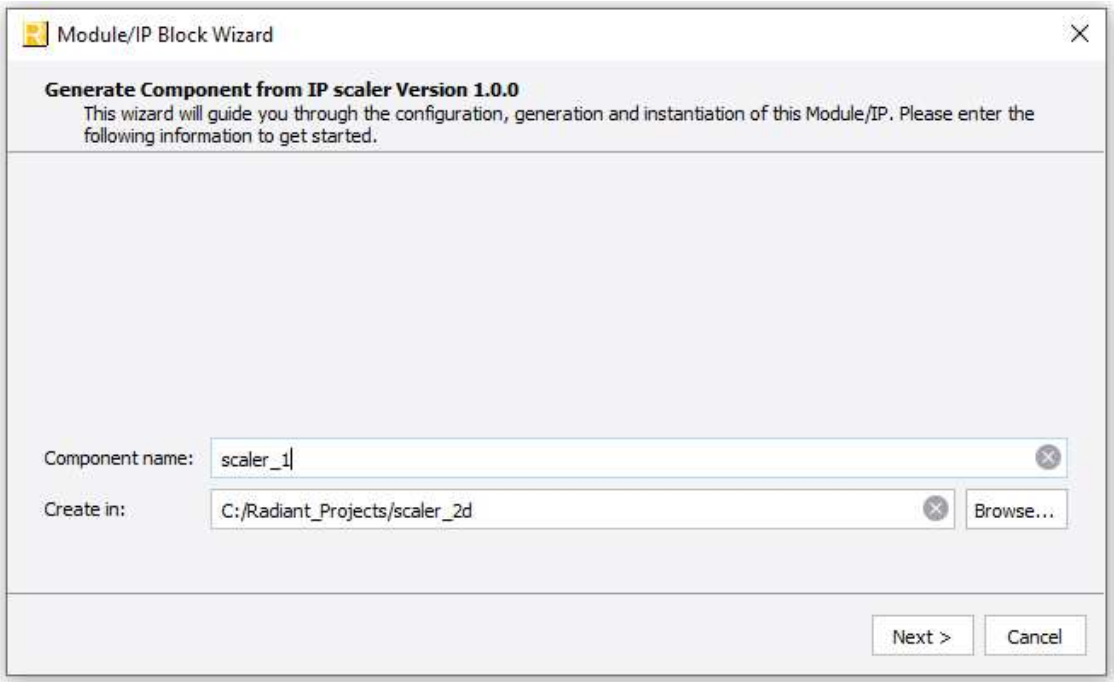


Figure 3.1. Module/IP Block Wizard

3. In the module's dialog box of the **Module/IP Block Wizard** window, customize the selected 2D Scaler IP Core using drop-down menus and check boxes. As a sample configuration, see Figure 3.2 .For configuration options, see the Attribute Summary section.

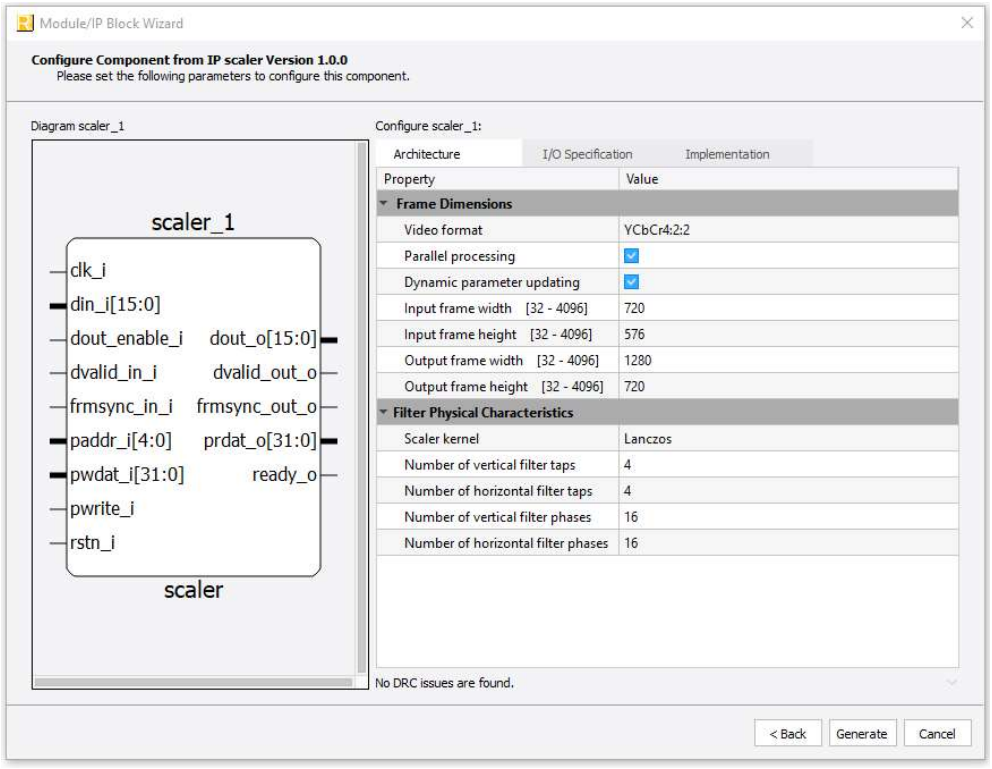


Figure 3.2. Configure User Interface of 2D Scaler IP Core

4. Click **Generate**. The **Check Generating Result** dialog box opens, showing design block messages and results as shown in Figure 3.3

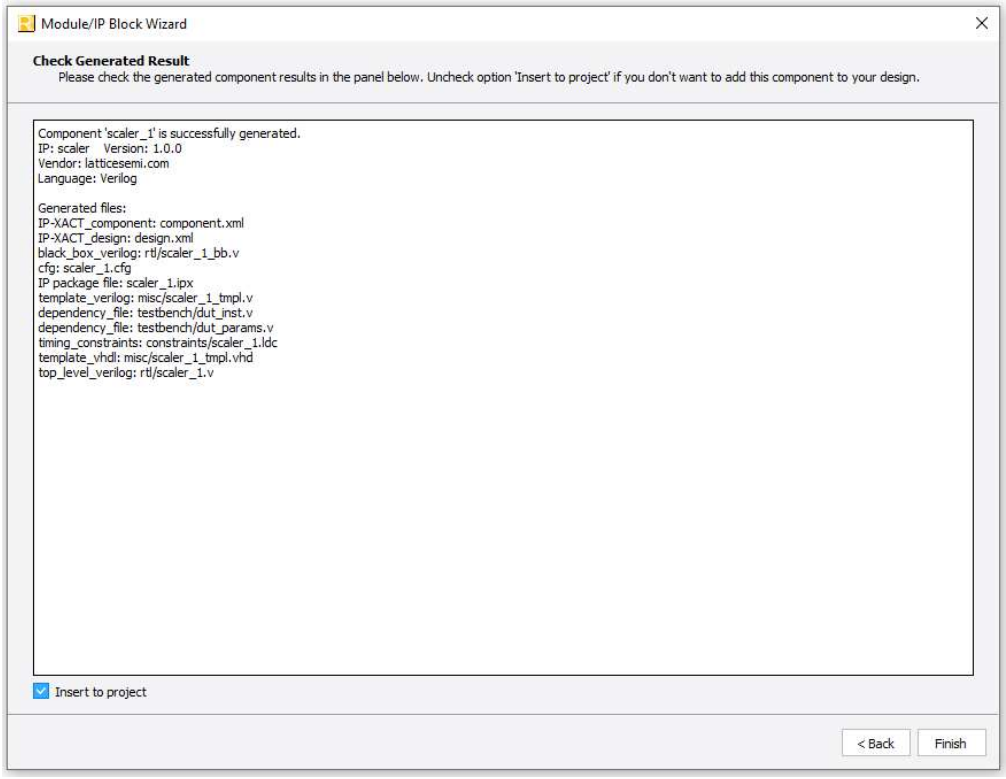


Figure 3.3. Check Generating Result

5. Click the **Finish** button. All the generated files are placed under the directory paths in the **Create in** and the **Component name** fields shown in Figure 3.1.

The generated 2D Scaler IP Core package includes the black box (<Component name>_bb.v) and instance templates (<Component name>_tmpl.v/vhd) that can be used to instantiate the core in a top-level design. An example RTL top-level reference source file (<Component name>.v) that can be used as an instantiation template for the IP core is also provided. You may also use this top-level reference as the starting template for the top-level for their complete design. The generated files are listed in Table 3.1.

Table 3.1. Generated File List

Attribute	Description
<Component name>.ipx	Contains the information on the files associated to the generated IP
<Component name>.cfg	Contains the parameter values used in IP configuration
component.xml	Contains the ipxact:component information of the IP
design.xml	Documents the configuration parameters of the IP in IP-XACT 2014 format
rtl/<Component name>.v	Provides an example RTL top file that instantiates the IP core
rtl/<Component name>_bb.v	Provides the synthesis black box
misc/<Component name>_tmpl.v misc /<Component name>_tmpl.vhd	Provide instance templates for the IP core

3.3. Running Functional Simulation

After the IP is generated, running functional simulation can be performed using different available simulators. The default simulator already has pre-compiled libraries ready for simulation. Choosing a non-default simulator, however, may require additional steps.

To run functional simulation using default simulator:

1. Click the  button located on the **Toolbar** to initiate the **Simulation Wizard** shown in Figure 3.4.

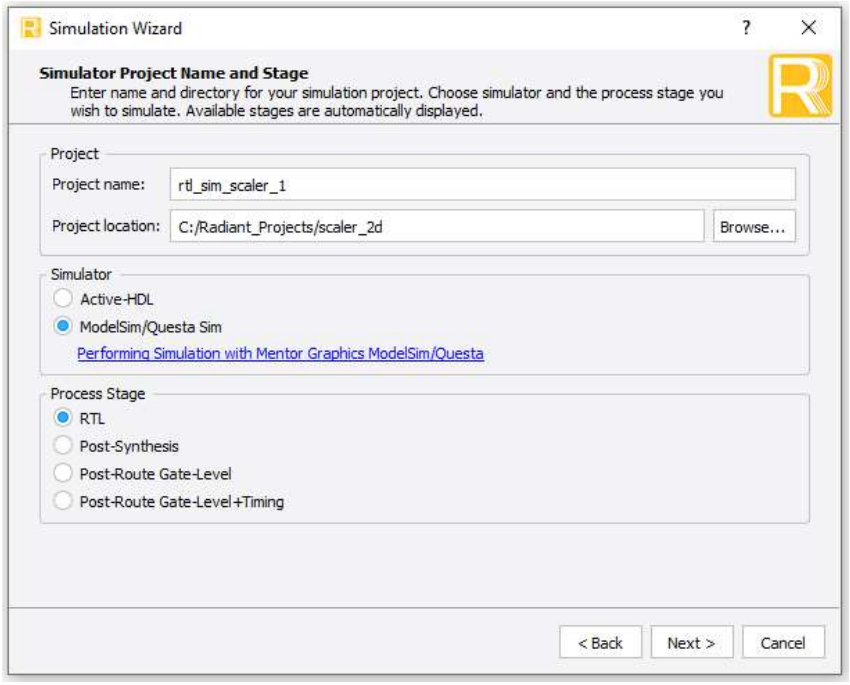


Figure 3.4. Simulation Wizard

2. Click **Next** to open the **Add and Reorder Source** window as shown in Figure 3.5.

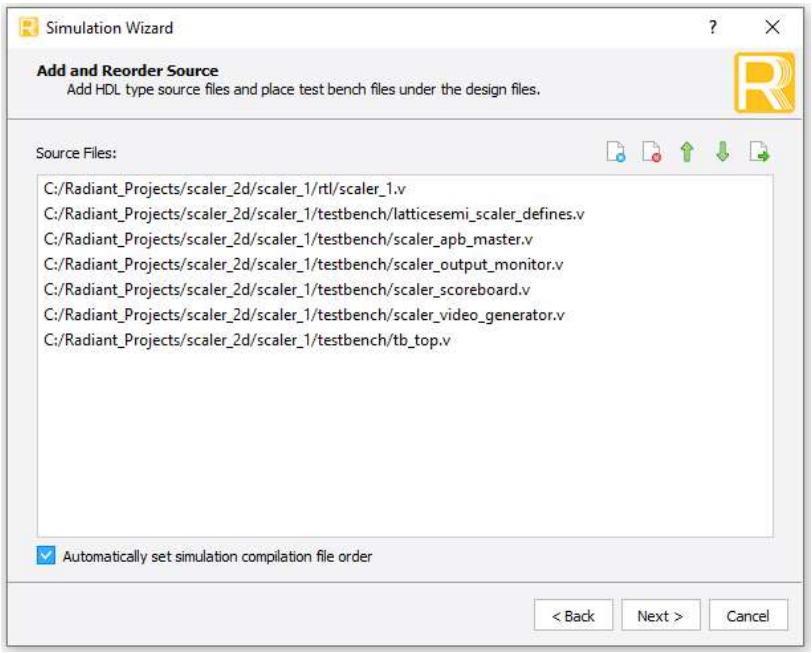


Figure 3.5. Adding and Reordering Source

3. Click **Next**. The **Summary** window is shown.
4. Click **Finish** to run the simulation.

Note: It is necessary to follow the procedure above until it is fully automated in the Lattice Radiant software suite.

The results of the simulation in our example are provided in Figure 3.6.

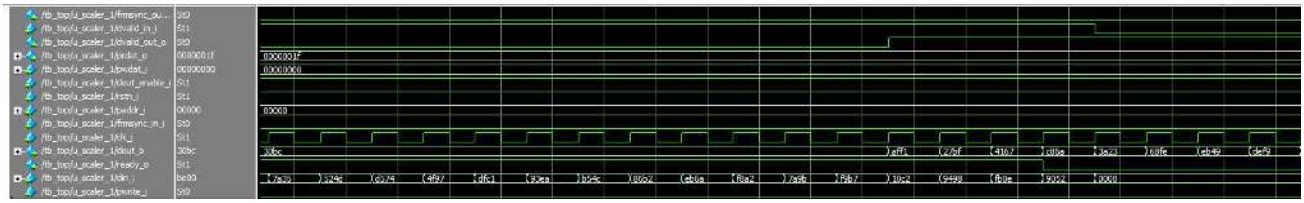


Figure 3.6. Simulation Waveform

3.4. Hardware Evaluation

The 2D Scaler IP Core supports Lattice's IP hardware evaluation capability when used with CrossLink-NX and Certus-NX devices. This 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. The hardware evaluation capability may be enabled/disabled in the Strategy dialog box. It is enabled by default. To change this setting, go to Project > Active Strategy > LSE/Synplify Pro Settings.

4. Ordering Part Number

The Ordering Part Number (OPN) for this IP Core are the following:

- SCALER-CNX-U – 2D Scaler for CrossLink-NX – Single Design License
- SCALER-CNX-UT – 2D Scaler for CrossLink-NX – Site License
- SCALER-CTNX-U – 2D Scaler for Certus-NX – Single Design License
- SCALER-CTNX-UT – 2D Scaler for Certus-NX – Site License
- SCALER-CPNX-U – 2D Scaler for CertusPro-NX – Single Design License
- SCALER-CPNX-UT – 2D Scaler for CertusPro-NX – Site License

Appendix A. Resource Utilization

2D Scaler IP Core resource utilization are shown on Table A.1 using LIFCL-40-9BG400I device and Table A.2 using LFD2NX-40-9BG256I with Lattice Synthesis Engine of Lattice Radiant software 2.1. Default configuration is used and some attributes are changed from the default value to show the effect on the resource utilization.

Table A.1. Resource Utilization using LIFCL-40-9BG400I

Configuration	Clk Fmax (MHz)[1]	Registers	LUTs[2]	EBRs	DSPs
YCbCr422, 720x480 to 1280x720, Parallel Processing and Dynamic parameter updating disabled, Others = Default	200 MHz	1153	1216	4	4
YCbCr422, 1280x720 to 720x480, Parallel Processing enabled and Dynamic parameter updating disabled, Others = Default	200 MHz	1471	1339	8	8
RGB, 1280x720 to 1920x1080, Parallel Processing enabled and Dynamic parameter updating enabled, Others = Default	200 MHz	1913	1986	11	12

Table A.2. Resource Utilization using LFD2NX-40-9BG256I

Configuration	Clk Fmax (MHz)[1]	Registers	LUTs[2]	EBRs	DSPs
YCbCr422, 720x480 to 1280x720, Parallel Processing and Dynamic parameter updating disabled, Others = Default	200 MHz	1153	1216	4	4
YCbCr422, 1280x720 to 720x480, Parallel Processing enabled and Dynamic parameter updating disabled, Others = Default	200 MHz	1471	1339	8	8
RGB, 1280x720 to 1920x1080, Parallel Processing enabled and Dynamic parameter updating enabled, Others = Default	200 MHz	1913	1986	11	12

Notes:

1. Fmax is generated when the FPGA design only contains 2D Scaler IP Core and the target frequency is 200 MHz. These values may be reduced when user logic is added to the FPGA design.
2. The *distributed RAM* utilization is accounted for in the total LUT4s utilization. The actual LUT4 utilization is distribution among *logic, distributed RAM*, and *ripple logic*.

References

- CrossLink-NX FPGA Web Page at www.latticesemi.com
- Certus-NX FPGA-Web Page at www.latticesemi.com
- CertusPro-NX FPGA Web Page at www.latticesemi.com

Technical Support Assistance

Submit a technical support case through www.latticesemi.com/techsupport.

Revision History

Document Revision 1.1, Lattice Radiant SW Version 3.0, June 2021

Section	Change Summary
All	Minor adjustment in formatting.
Introduction	Updated content, including Table 1.1 to add CertusPro-NX support.
Ordering Part Number	Added part number for CertusPro-NX.
References	Updated this section to add CertusPro-NX web page.

Document Revision 1.0, Lattice Radiant SW Version 2.1, December 2020

Section	Change Summary
All	Initial release.

www.latticesemi.com