Deinterlacer IP Core - Lattice Radiant Software

User Guide

FPGA-IPUG-02135-1.1

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
BOBE	Bob deinterlacing algorithm using Even fields
BOBO	Bob deinterlacing algorithm using Odd fields
FIFO	First In First Out
LMMI	Lattice Memory Mapped Interface
RGB	Red Green Blue
SAD	Sum of Absolute Differences

1. Introduction

The Lattice Semiconductor Deinterlacer IP core converts interlaced video into progressive video format using bob, intra motion and inter motion adaptive deinterlacing algorithms to reduce interline flickers and jagged edges. The Deinterlacer IP core supports image sizes up to 4k × 4k with YCbCr 4:2:2, 4:4:4 and RGB video formats. The Deinterlacer IP core supports dynamic parameter updating through a parameter bus, which can be configured to operate on a different clock from the core. Simple frame rate conversion is employed to support different input and output frame rates.

1.1. Quick Facts

Table 1.1 presents a summary of the Deinterlacer IP core.

Table 1.1. Quick Facts

IP Requirements	Supported FPGA Families	CrossLink™-NX, Certus™-NX, CertusPro™-NX
Resource Utilization	Targeted Devices	LIFCL-17, LIFCL-40, LFD2NX-17, LFD2NX-40, LFCPNX-100
	Supported User Interface	Native interface, see Signal Descriptions
	Resources	See Table A.1
Design Tool Support	Lattice Implementation	IP Core v1.x.x - Lattice Radiant® Software 2.2 or later
	Synthesis	Lattice Synthesis Engine (LSE)
		Synopsys® Synplify Pro® for Lattice
	Simulation	For a list of supported simulators, see the Lattice Radiant Software User Guide.

1.2. Features

Key features of the Deinterlacer IP core include:

- Single color, YCbCr 4:2:2, YcbCr 4:4:4, and RGB video formats
- Serial and parallel deinterlacing
- Weave, bob, intra and inter motion adaptive deinterlacing algorithms
- Frame rate conversion
- Configurable initial field
- Provides configurable thresholds for inter motion adaptive deinterlacing algorithm
- Dynamic parameter update of frame size, initial field and bypass mode
- Configurable parameter bus width
- Configurable parameter bus clock
- Configurable memory bus width and base address
- Configurable memory burst length and burst count
- Configurable internal FIFO type and depth
- 8, 10 or 12-bit color depth per plane
- Configurable line buffer type

1.3. Conventions

1.3.1. Nomenclature

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

2. Functional Description

2.1. Overview

In interlaced video, a frame is divided into two fields with each field containing every other horizontal line in a frame. One field is transmitted at a time and thus uses only half the bandwidth. Most modern displays support progressive scan, and to display interlaced video on progressive scan displays, deinterlacing, a method of combining the two fields into a frame, is applied to the video signal.

Each frame of interlaced video is composed of two fields that are captured at different moments in time. As such, there are flickers and jagged edges in the combined frame. A good deinterlacing algorithm reduces these artifacts as much as possible and yields a good video quality in the process.

The Deinterlacer IP core provides several deinterlacing algorithms for different video quality and resource requirements: weave, bob, intra and inter motion adaptive deinterlacing algorithms.

Figure 2.1 shows the block diagram of the Deinterlacer IP core. The core consists of three modules: frame buffer, line buffer, and deinterlacer engine. The frame buffer module manages memory write/read and combines interlaced fields into progressive frames. The line buffer module and deinterlacing engine process the combined frames to reduce artifacts.

In the Deinterlacer IP core, several clock sources are involved. When frame rate conversion is enabled, there are two clocks in the video data path: input pixel sample clock and output pixel sample clock. The frame buffer module handles the rate conversion, and the line buffer and deinterlacing engine operate at the output pixel clock rate. When frame rate conversion is disabled, all the modules in the video data path operate at input pixel clock rate. When dynamic parameter updating is selected, the parameter bus can be configured to run on a separate clock. By default, the parameter bus runs on the input pixel sample clock. The memory interface always operates on a separate clock.

The input data must be in interlaced video format. An interlaced frame is composed of two fields. The first field in the interlaced frames can be configured to be top field or bottom field. This parameter can be configured at run-time.

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

The deinterlacer core does not use multipliers.

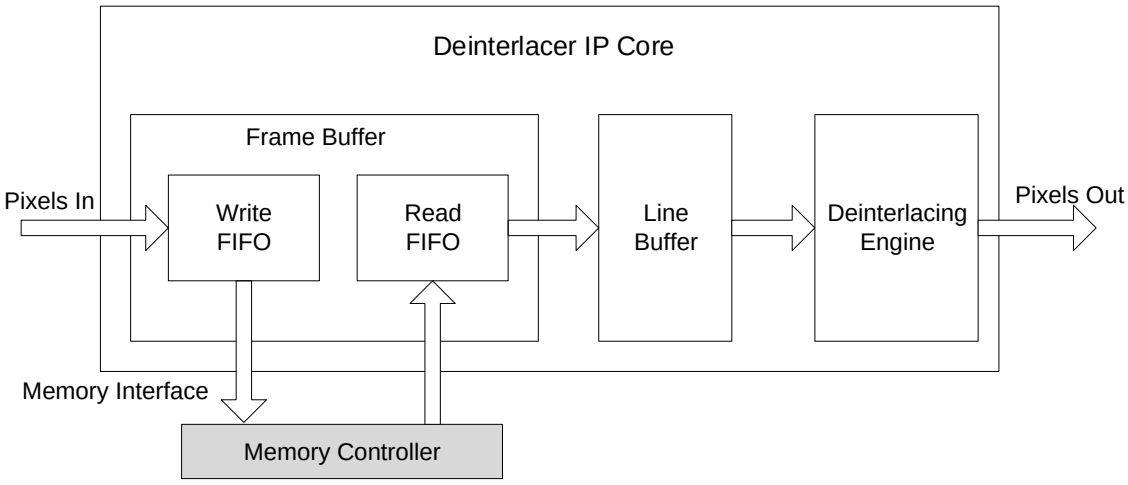

Figure 2.1. Deinterlacer IP Core Block Diagram

2.1.1. Deinterlacing Algorithms

The Deinterlacer IP core supports weave, bob, intra and inter motion adaptive deinterlacing algorithms.

2.1.1.1. Weave

The weave algorithm combines the two interlaced fields together. There is no line buffer module or deinterlacing engine instantiated in the core.

Weaving is fine when there is no change in the image between fields. Any change, however, results in artifacts known as *combing*. This results when pixels in one frame do not line up with pixels in the other frames, thereby forming jagged edges.

2.1.1.2. Bob

Bob algorithm uses a single field to generate a frame by averaging up lines and down lines to generate new lines. Either even or odd fields can be selected for progressive frame generation.

2.1.1.3. Intra Motion Adaptive Deinterlacing

The intra motion adaptive deinterlacing algorithm enhances the quality of the combined frames by using a spatial predictor with advanced ELA (Edge-based Line Average) algorithm to predict the center pixel. The intra Motion Adaptive Filter uses an enhanced multi-stage median filter with robustness control to perform motion adaptive deinterlacing.

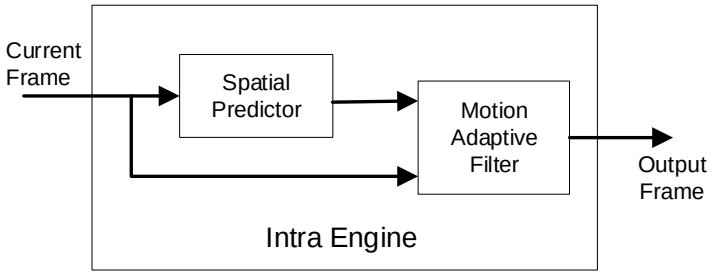

Figure 2.2. Intra Deinterlacing Engine Structure

2.1.1.4. Inter motion adaptive deinterlacing

The inter motion adaptive deinterlacing algorithm uses two combined frames (four fields) to generate one progressive frame The Motion Detector uses a 3x3 pixel window SAD (Sum of Absolute Differences) algorithm to determine the motion of the center pixel. The SAD value is divided by two thresholds (TH1 and TH2) into three-pixel motion regions: still pixel, slow motion pixel, and fast motion pixel. Each region has a separate filter with robustness control to generate the output pixel. The final output pixel value is determined by the detector value and the threshold values.

The two threshold values can be updated at run time.

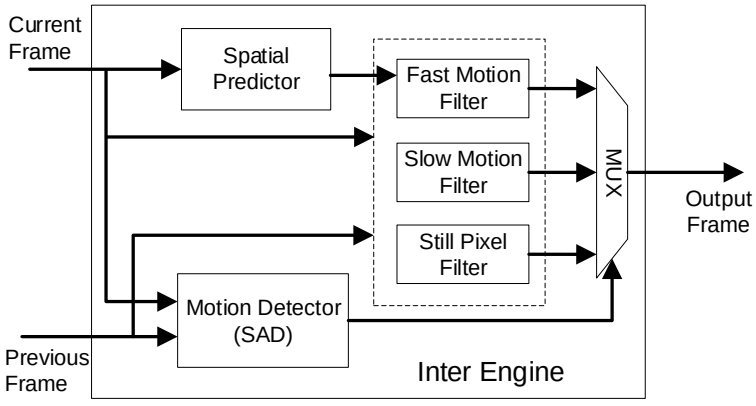

Figure 2.3. Inter Deinterlacing Engine Structure

2.1.2. Frame Rate Conversion

The Deinterlacer IP core provides a simple frame rate conversion by copying or dropping frames.

When frame rate conversion is enabled, the core's output data path runs at the output pixel clock rate, and the core needs one more external frame memory space. The output frame rate is controlled by the output pixel sample clock and dout_enable signal. When dout_enable signal is high, the core outputs pixels continuously. When there is no new interlaced video frame, the core outputs the last progressive frame repeatedly.

When frame rate conversion is disabled, the core's output data path runs on the input pixel sample clock. The output frame is generated directly from the input interlaced video stream. If there is no new input video frame, the core stops outputting data.

2.1.3. Dynamic Parameter Configuration Bus

The Deinterlacer IP core provides a parameter bus port for internal parameter update at run-time. The parameters are double-buffered so that the core's operation is not impacted when they are changed. The new values are buffered and transferred to the working registers when the core is ready to accept a new configuration. The UPDATE register is used to indicate when the new values are consumed and when the buffers can accept new data. Refer to section 3.3 for the list of registers.

2.1.4. Memory Bandwidth and Size

The Deinterlacer IP core stores and retrieves pixels to/from external memory using memory burst write and read commands. When DDR2 is used for external memory, one burst operation, (burst_length × burst_count/2) × memory_data_width bit, cannot exceed the size of a single video line. A single video line is transferred though multiple burst write/read transactions internally.

For weave, bob and intra algorithms, the deinterlacer core needs a two-frame memory storage space; if frame rate conversion is active, a three-frame storage space is required. When the output frame rate is the same as the input frame rate, the required memory bandwidth is twice the input data rate. When the output frame rate is doubled of the input frame rate, the required memory bandwidth is tripled the input data rate.

For the inter algorithm, the deinterlacer core uses two interlaced frame to generate one progressive frame; it needs a 3frame storage space. If frame rate conversion is enabled, the core needs a four-frame memory space. When the output frame rate is the same as the input frame rate, the required memory bandwidth is tripled the input data rate. When the output frame rate is doubled the input frame rate, the required memory bandwidth is 5 times the data rate of input video stream.

For example, for 8-bit YCbCr 4:2:2 pixel, parallel deinterlacing, if the input pixel sample clock is 74.25MHz and output pixel sample clock is 148.5MHz, the required bandwidth for the intra algorithm is 2 × 8 × (74.25+148.5) = 3564 bit MHz. The required bandwidth for the inter algorithm is 2 × 8 × (74.25+2 × 148.5) = 5940 bit MHz. If the memory data width is 32, the require memory clock is (3564/32) = 111.375 MHz for intra algorithm and (5940/32) = 185.625 MHz for inter algorithms.

The total external memory size the core requires can be viewed on the Deinterlacer IP user interface.

2.2. Signal Descriptions

Table 2.1 lists the top-level input and output signals for the Deinterlacer IP core.

Table 2.1. Deinterlacer Input and Output Ports

Signal	Direction	Width (Bits)	Description
General I/O			
rstn	IN	1	Asynchronous active-low reset signal.
sr	IN	1	Synchronous reset signal (optional)
iclk	IN	1	Input pixel sample clock
frmsync_in	IN	1	New input video frame indicator, active-high.
dvalid_in	IN	1	Input video data valid signal, active-high.
din	IN	8/10/12	Input video data in interlaced frame format.
ready	OUT	1	When high, indicating the deinterlacer core can accept more input data.
oclk	IN	1	Output sample clock, present when frame rate conversion is active.
dout_enable	IN	1	Input from down-stream module to enable progressive output data, active high.
dout	OUT	8/10/12	Output video pixel in progressive frame format.
dvalid_out	OUT	1	Output video pixel valid signal, active high
frmsync_out	OUT	1	New output frame indicator, active high.
Memory Interface			
mem_clk	IN	1	Memory write/read clock.
cmd_ready	IN	1	Input from memory controller indicating it's ready to accept a new command, active high.
mem_addr	OUT	32	Memory read/write address.
read_cmd	OUT	1	Memory read command, active-high.
read_data	IN	8/16/32/64/128	Read data output from memory.
read_data_valid	IN	1	Read data valid indicator from memory controller, active high
write_cmd	OUT	1	Memory write command, active-high.
write_data	OUT	8/16/32/64/128	Write data to memory.
write_data_ready	IN	1	Input from memory controller indicating that it's ready to accept new write data, active high.
Optional I/O			
fwidth_out	OUT	16	Current output frame width, only for dynamic mode.
fheight_out	OUT	16	Current output frame height, only for dynamic mode.
pclk	IN	1	Parameter bus clock, configurable.
pwrite	IN	1	Parameter bus write enable, active-high.
paddr	IN	5	Parameter bus address.
pwdat	Output	8/16/32	Parameter bus write data.
prdat	OUT	8/16/32	Parameter bus read data.

2.2.1. Video Input/Output

The Deinterlacer IP core uses a simple handshaking method to pass pixel data into and out of the core. The core asserts its ready output when it is ready to receive data. When the driving module has data to pass to the deinterlacer, it asserts the core's dvalid_in port and at the same time placing the input video data on the din port. The frmsync_in input should be driven to a '1' during the clock cycle when the very first pixel of the interlaced frame is active.

Similarly, dvalid_out is active when valid output pixel data is available on dout, and frmsync_out marks the first pixel in the output progressive frames.

The ports fwidth_out and fheight_out indicate the current output frame's width and height respectively, which are valid only when frmsync_out is asserted.

When the input signal dout_enable is asserted, the core outputs progressive video pixels. When dout_enable is de-asserted, the core stops generating output pixels after some delay, which is less than 16 output pixel sample clock cycles.

2.2.2. Memory Interface

The Deinterlacer IP core implements a flexible memory interface which operates on a separate clock from the main core.

The deinterlacer assumes a memory byte addressing scheme. When connecting the memory controller, fine tune the combination of row/bank/column/ address to get the best throughput.

For example, when DDR2 data width is 16, row size is 14, column size is 10, bank size is 8, burst length is 8 and burst count is 1, which means memory controller data width is 32, row address is 14 bits, column address is 10 bits, bank address is log2(bank size)=3 bits and log2(burst length * burst count)= 3 bits. Normally memory controller address is

ddr_addr = {row_addr[13:0], bank_addr[2:0], col_addr[9:0]}

In order to get the best throughput, a connection between the deinterlacer core and memory controller can be

core_mem_addr = {row_addr[13:0], col_addr[9:3], bank_addr[2:0], col_addr[2:0],1'b0}.

As the memory controller data width is 32, core_mem_addr[1:0] is always zero.

The core has internal control logic to arbitrate between memory write and read operations, ensuring the write_cmd and read_cmd not being asserted at the same time.

When two clock cycles after the write_data_ready is asserted, data becomes available on the write_data port. The cmd_ready can be asserted once every 2 clock cycles, and it should have at least one-cycle interval.

2.2.3. Parameter Configuration Bus

The Deinterlacer IP core implements a simple register read/write interface for run-time parameter updates. The parameter bus interface can be configured to run on a separate clock. It operates at the input pixel clock rate by default.

When pwrite is high, pwdat and paddr must contain valid data. The contents of all parameter registers are transferred to the core's internal storage when UPDATE is asserted. If a parameter hadn't been written to before the assertion of UPDATE, its old value is transferred into the internal storage.

prdat contains register read data corresponding to the address value placed on the paddr in the previous clock cycle.

When the parameter bus data width is equal to 32, paddr[1:0] should be fixed to 0. When parameter bus data width equals to 16, paddr[0] should be fixed to 0.

The parameter bus data width should be configured based on the system CPU's data width.

2.2.4. Timing Description

2.2.4.1. Video Input/Output Timing

The Deinterlacer IP core supports single color, YCbCr 4:2:2, YCbCr 4:4:4 or RGB video format.

For YCbCr 4:4:4 or RGB video format, the three planes are interleaved for serial deinterlacing and combined on the *din* and *dout* ports for parallel deinterlacing. Figure 2.4 and Figure 2.5 show the timing of RGB serial deinterlacing and parallel deinterlacing.

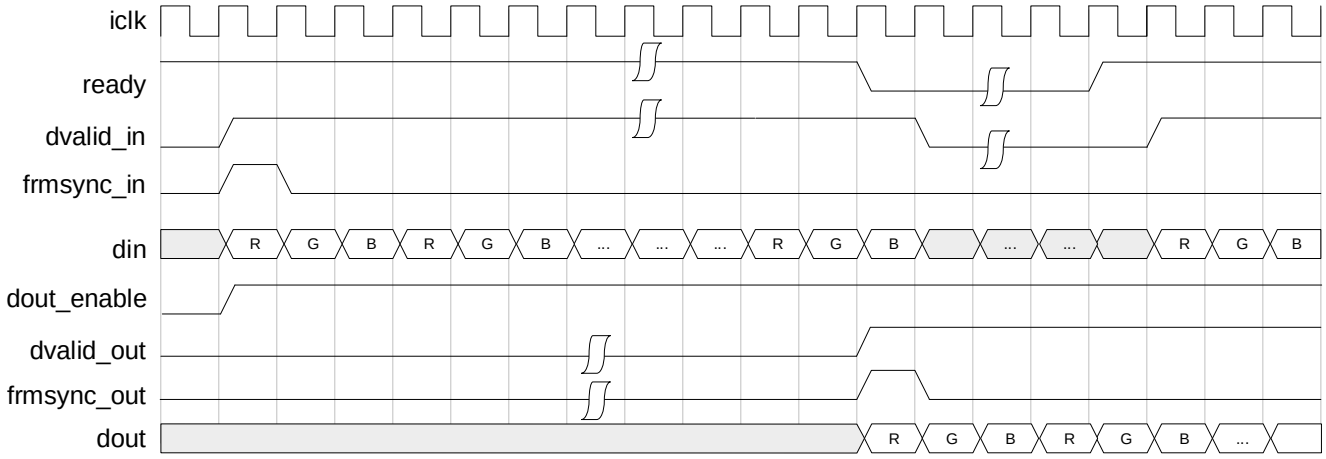

Figure 2.4. RGB Serial Deinterlacing

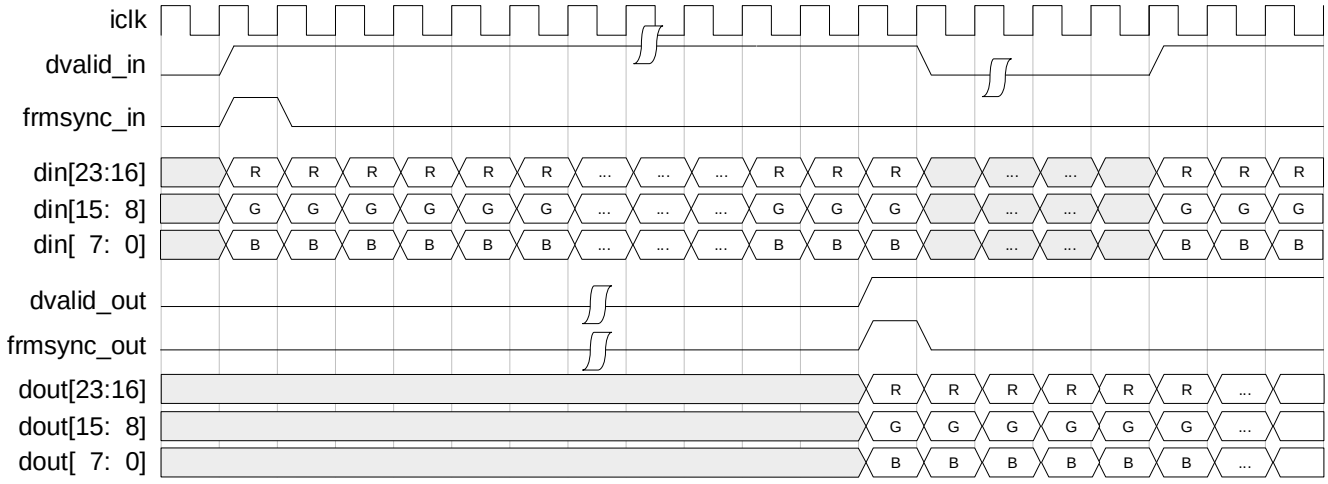

Figure 2.5. RGB Parallel Deinterlacing (8-Bit Pixel)

For YCbCr 4:2:2 video serial deinterlacing, the input and output sequence should be Cb, Y, Cr, Y, …. For parallel deinterlacing, the Y plane occupies the upper bits of the din and dout ports, and the Cb and Cr planes the lower bits. Cb and Cr planes are interleaved in the lower half, and Cb comes before Cr. Figure 2.6 and Figure 2.7 show the timing of YCbCr 4:2:2 serial deinterlacing and parallel deinterlacing.

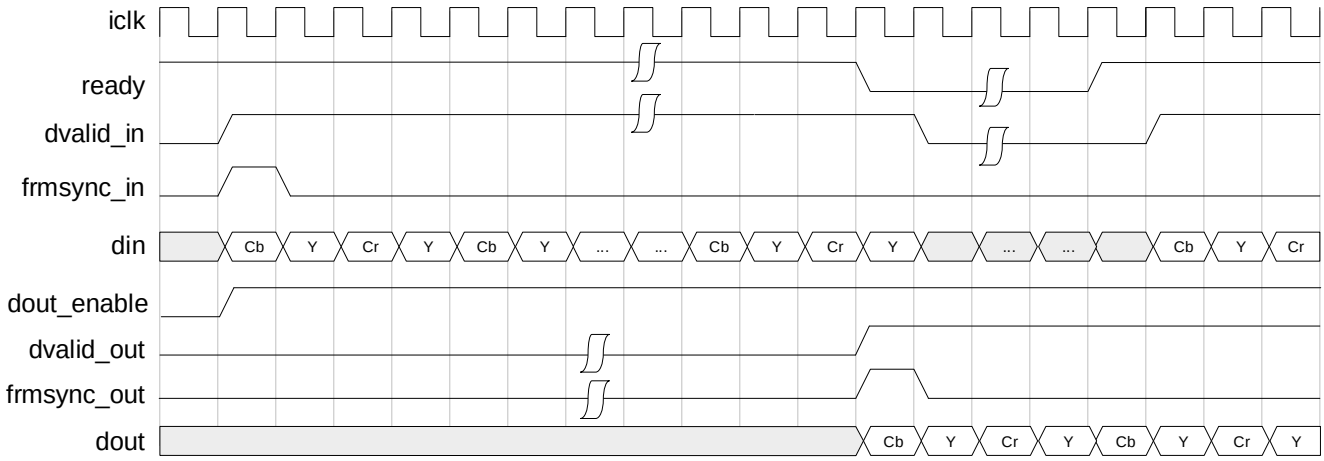

Figure 2.6. YCbCr 4:2:2 Serial Deinterlacing

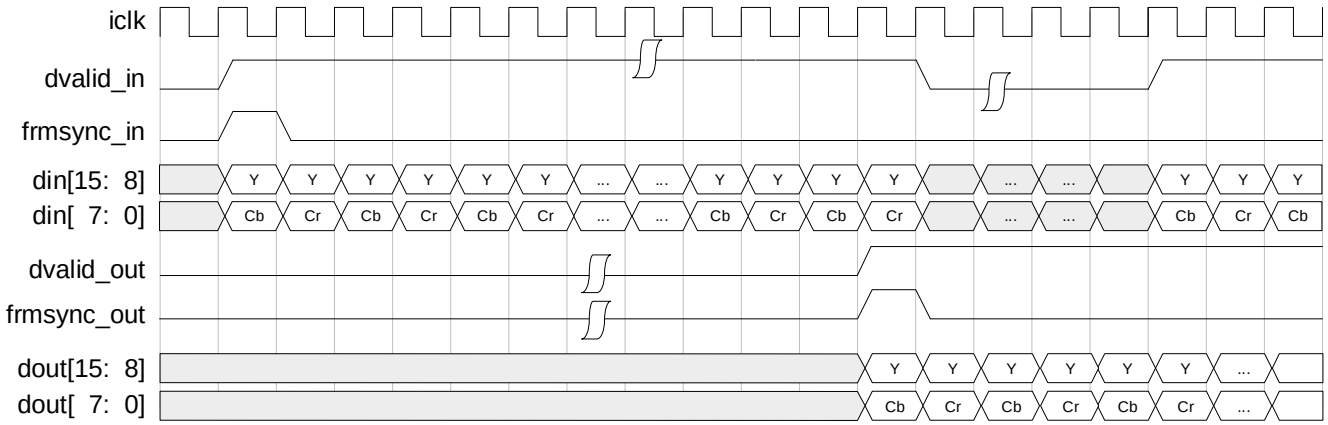

Figure 2.7. YCbCr 4:2:2 Parallel Deinterlacing (8-Bit Pixel)

Figure 2.8 shows the dout_enable control timing. When the dout_enable is de-asserted, the core stops outputting data after a certain delay. Similarly, when the dout_enable is asserted, the core begins outputting data after a certain delay. The number of delay clocks is different for different deinterlacing algorithms and the maximum value is less than 16 clocks. The asserting and de-asserting of dout_enable can be used to generate horizontal blank and vertical blank depending on the output video format.

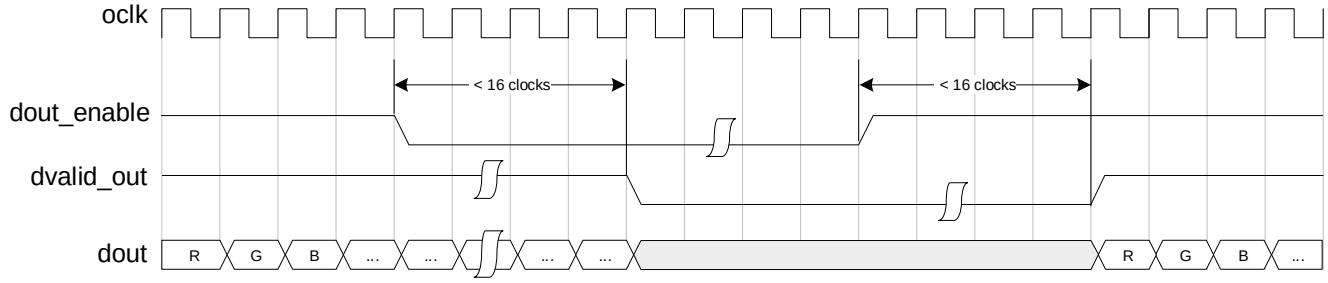

Figure 2.8. dout_enable Control Timing

2.2.4.2. Video Frame Timing

Figure 2.9 shows the frame timing when frame rate conversion is disabled. After accepting one input interlaced frame, the core starts generating progressive frames. The output frame is driven by the input frame. When a new input frame arrives before current output frame is finished, the current output frame is dropped, and a new output frame is started. After the last input frame, the deinterlacer stops generating output frames.

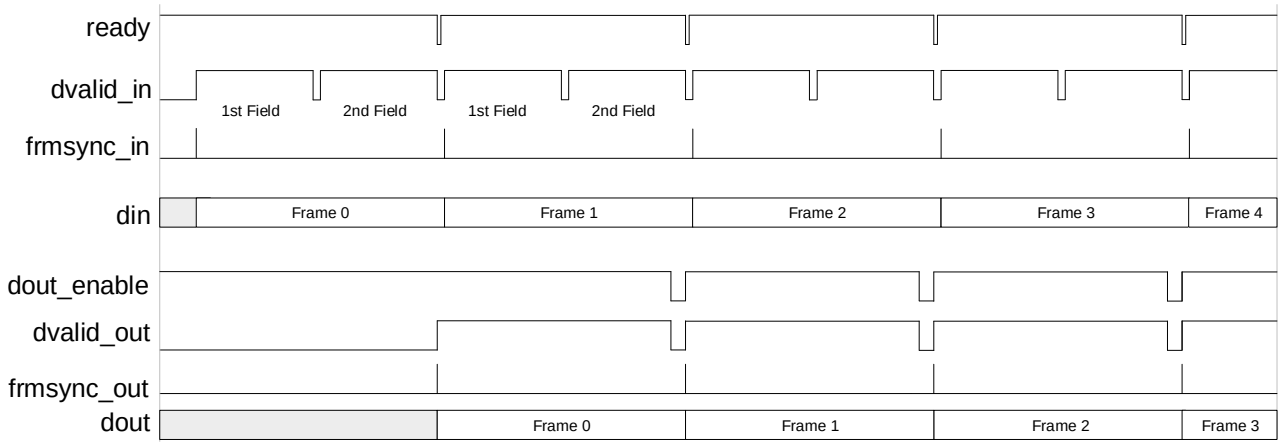

Figure 2.9. Output Frame Rate is Same as Input Frame Rate

Figure 2.10 shows the frame timing when the output frame rate is twice the input frame rate. Output frames run on a separate output pixel clock. The output frame rate is determined by the output pixel clock and dout_enable signal. The core generates output frames based on the oldest pending input frame(s). When a new input frame is received, the core pushes out the oldest frame and exports the second oldest frame. When there is no new frame input, the core exports the stored frames and then keeps exporting the last frame repeatedly. When the external memory is fully occupied by the unconsumed frames (no more space for new frames), the core overwrites the latest input frame.

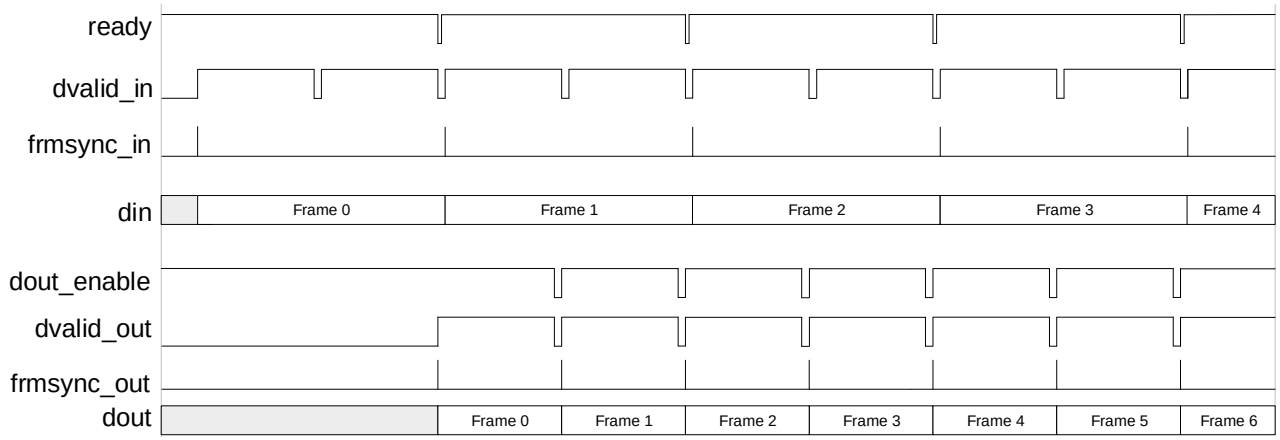

Figure 2.10. Output Frame Rate is Twice Input Frame Rate

2.2.4.3. Memory Interface Timing

Figure 2.11 shows the timing of memory write operation. When the internal write FIFO is half full, it triggers memory write operation cycles. Memory write operation continues until the write FIFO is empty.

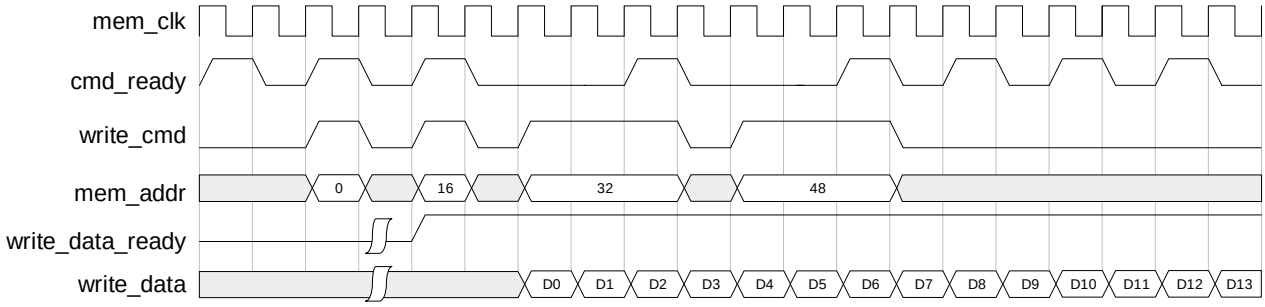

Figure 2.11. Timing Diagram for Memory Write

Figure 2.12 shows the timing of memory read operation. When the internal read FIFO is less than half full, and there is no ongoing write operation, memory read burst operation is started. The length of read burst operation and once read burst makes the read FIFO half full.

Memory write and read operation are in bursts. The memory write and read operations switch at the end of the burst cycle.

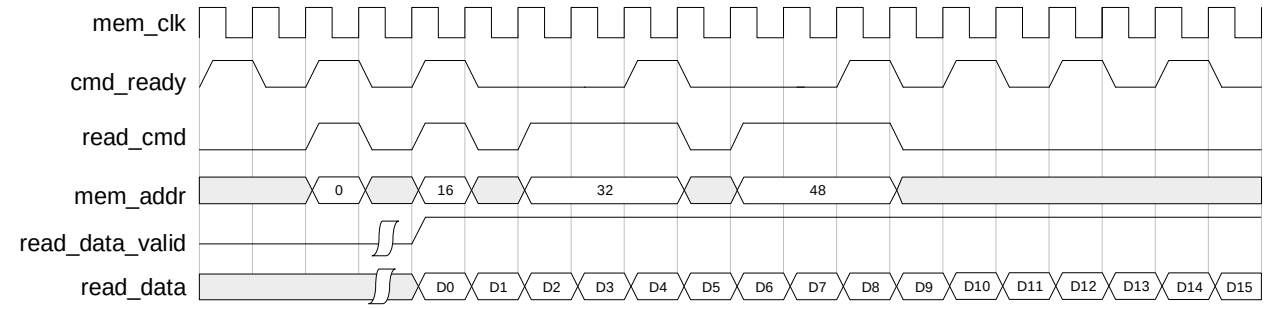

Figure 2.12. Timing Diagram for Memory Read

2.2.4.4. Parameter Configuration Bus Timing

Figure 2.13 shows the timing of dynamic parameters updating. FW and FH are the video frame width and height. IFLD is the INITFLD register value, BP is the BYPASS register value and KP is the KEEP register value.

The parameter bus can be configured to operate on a separate clock. By default, it operates at input pixel clock rate. The parameter registers are writable only when UPDATE register is 0. When the UPDATE register bit is set to 1, the core updates its internal parameter registers with the new values at the next active frmsync_in and reset the UPDATE register bit.

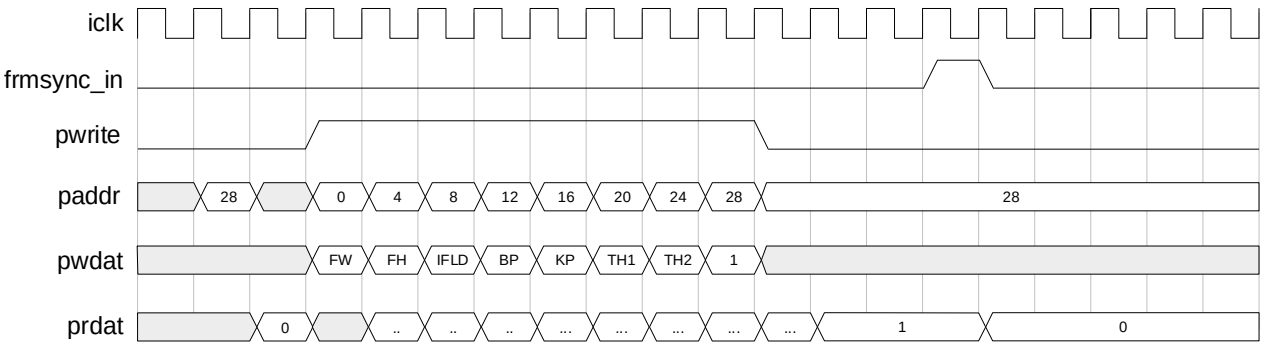

Figure 2.13. Timing Diagram for Dynamic Parameter Configuration Bus (Parameter Bus Width = 32)

2.3. Attribute Summary

Table 2.2 presents a list of user configurable parameters. The parameter settings are specified using the Global Phase Locked Loop Module Configuration user interface in the Lattice Radiant software.

Table 2.2. Attributes Summary

User Interface Configuration. Tab	Attribute	Selectable Values	Default
Architecture	Video Format	Single Color, YbCbCr422, YbCbCr444, RGB	YbCbCr422
	Video Frame Width	64 – 4096	720
	Video Frame Height	64 – 4096	480
	First Field	Top Field, Bottom Field	Top Field
	Parallel Processing	Checked, Unchecked	Checked
	Frame Rate Conversion	Checked, Unchecked	Checked
	Duplicate frames in frame rate conversion mode	Checked, Unchecked	Checked
	Deinterlacing Algorithm	WEAVE, BOBE, BOBO, INTRA, INTER	INTER
	Default SAD thresholds value	Checked, Unchecked	Unchecked
	SAD threshold 1 (For still pixel)	1 – 65535	90
	SAD threshold 1 (For slow motion pixel)	1 – 65535	180
I/O Specification	Input pixel width	8, 10, 12	8
	Memory bus width	8, 16, 32, 64, 128	32
	Memory base address (hex)	0x00000000 – 0xFFFFFFFF	0
	Memory address width	Calculated	22
	Memory size needed (bytes)	Calculated	2764800
	Synchronous reset (sr)	Checked, Unchecked	Unchecked
	Input frame re-sync flag	Checked, Unchecked	Unchecked
	Dynamic parameter bus	Checked, Unchecked	Unchecked
Implementation	Line buffer type	EBR, Distributed	EBR
	Write FIFO type	EBR, Distributed	EBR
	Read FIFO type	EBR, Distributed	EBR
	Write FIFO depth	32, 64, 128, 256, 512	256
	Read FIFO depth	32, 64, 128, 256, 512	256
	DDR memory burst length	2, 4, 8	8
	Command burst count	1, 2, 4, 8	1

2.4. Register Description

Table 2.3 lists GPLL fuses with their descriptions. Initial values of these configuration bits can be configured through the user interface. Most are Run-time configurable through optional LMMI.

Table 2.3. Parameter Register Maps

Address	Registers	Size	W/R	Description
0x00	FRMWIDTH	32	w/r	Frame width register. The FRMWIDTH value must be (frame width – 1). The minimum value is 63, and the maximum value is the max frame width specified on the IP user interface minus 1. The default value is the maximum value. Frame width must be an even number (that is, FRMWIDTH must be odd).
0x04	FRMHEIGHT	32	w/r	Frame height register. The FRMHEIGHT must be (frame height – 1). The minimum value is 63, and the maximum value is the max frame height specified on the IP user interface minus 1. The default value is the maximum value. Frame height must be an even number (that is, FRMHEIGHT must be odd).
0x08	INITFLD	32	w/r	First field register. Specifies the first field in the interlaced input frames. 0 for top field and 1 for bottom field. The default value is the corresponding parameter specified on the IP user interface.
0x0C	BYPASS	32	w/r	BYPASS mode register. The value can be 0 or 1. When BYPASS register is 1, the deinterlacer core passes the interlaced video through unaltered. The default value is 0.
0x10	KEEP	32	w/r	KEEP mode register. The value can be 0 or 1. When KEEP register is 1, the core is locked. The core outputs the same frame repeatedly if frame rate conversion is active; otherwise it generates no output frames. The default value is 0.
0x14	INTER_TH1	32	w/r	The SAD threshold for still pixel for Inter algorithm. The maximum value is 65535.N/A for other algorithms. The default value is the corresponding parameter specified on the IP user interface.
0x18	INTER_TH2	32	w/r	The SAD threshold for slow motion pixel for Inter algorithm. The maximum value is 65535. N/A for other algorithms. The default value is the corresponding parameter specified on the IP user interface. The INTER_TH2 must be greater than INTER_TH1 value.
0x1C	UPDATE	32	w/r	Update parameters enable register. The value can be 0 or 1. Setting this bit to 1 triggers the update of parameters. The core resets it to 0 after the parameters have been updated. The default value is 0.

All the parameter registers can be written to only when the UPDATE register bit is 0. The parameters KEEP, INTER_TH1 and INTER_TH2 takes effect at the next frame, regardless of the UPDATE value. When the UPDATE bit is set to 1, the parameters FRMWIDTH, FRMHEIGHT, INTIFLD and BYPASS takes effect 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 indicate that the parameter registers are now empty and can take on new values.

3. IP Generation

This chapter provides information on how to generate and synthesize the IP using Lattice Radiant Software, as well as how to run the simulation. For more on Radiant Software, please refer to the *Lattice Radiant Software User Guide* and relevant Lattice tutorials.

3.1. Generation and Synthesis

Lattice Radiant Software allows you to generate and customize modules and IPs and integrate them into the device architecture.

To generate IP in Lattice Radiant software:

1. In the Module/IP Block Wizard create a new Lattice Radiant Software project.
2. In the dialog box of the Module/IP Block Wizard window, configure the IP according to custom specifications using drop-down menus and check boxes. As a sample configuration, see Figure 3.1.
3. For configuration options, see Table 2.2.

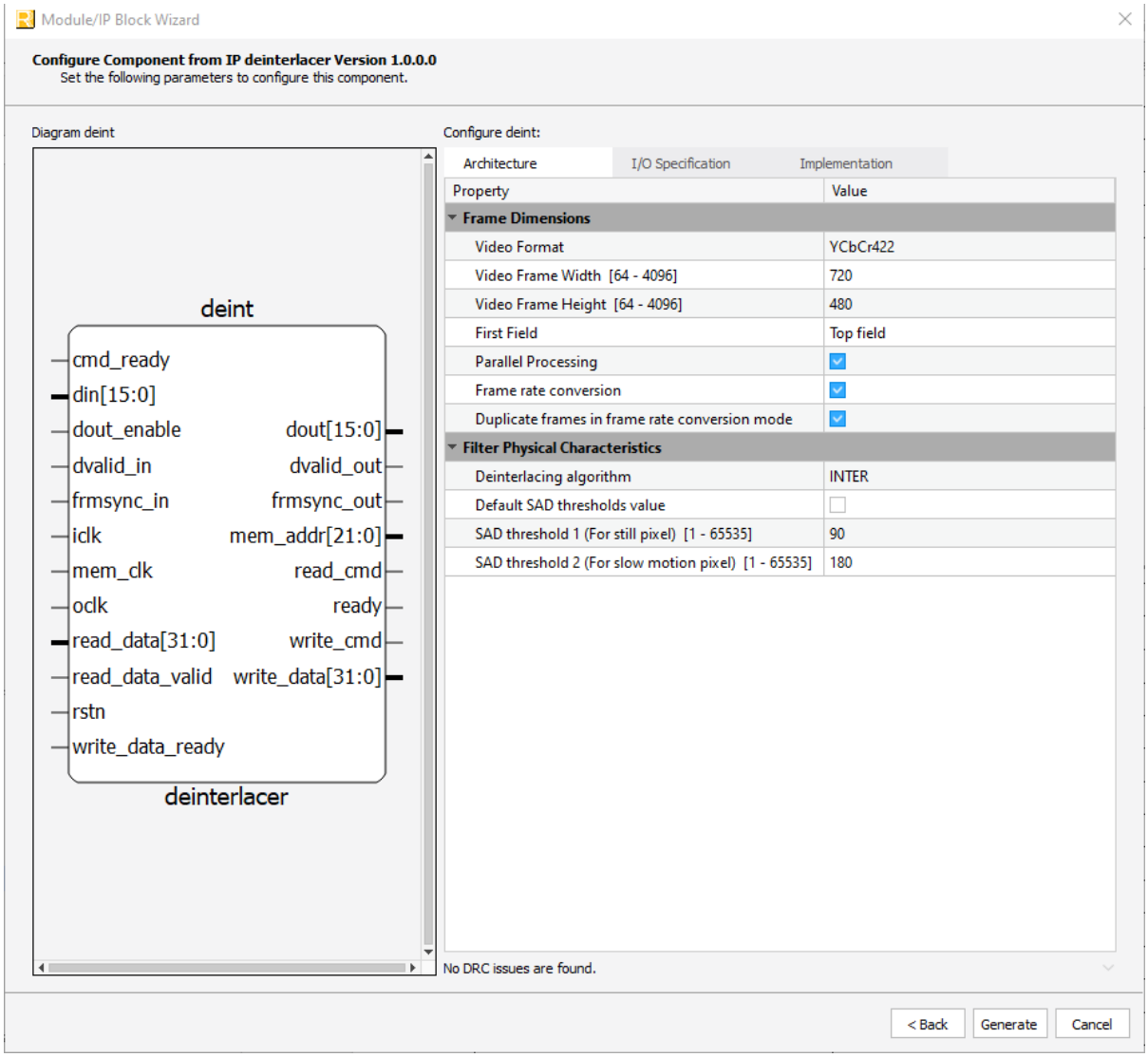

Figure 3.1. Configuring Deinterlacer IP

4. Click **Generate**. The **Check Generating Result** dialog box opens, showing design block messages and results as shown in Figure 3.2.

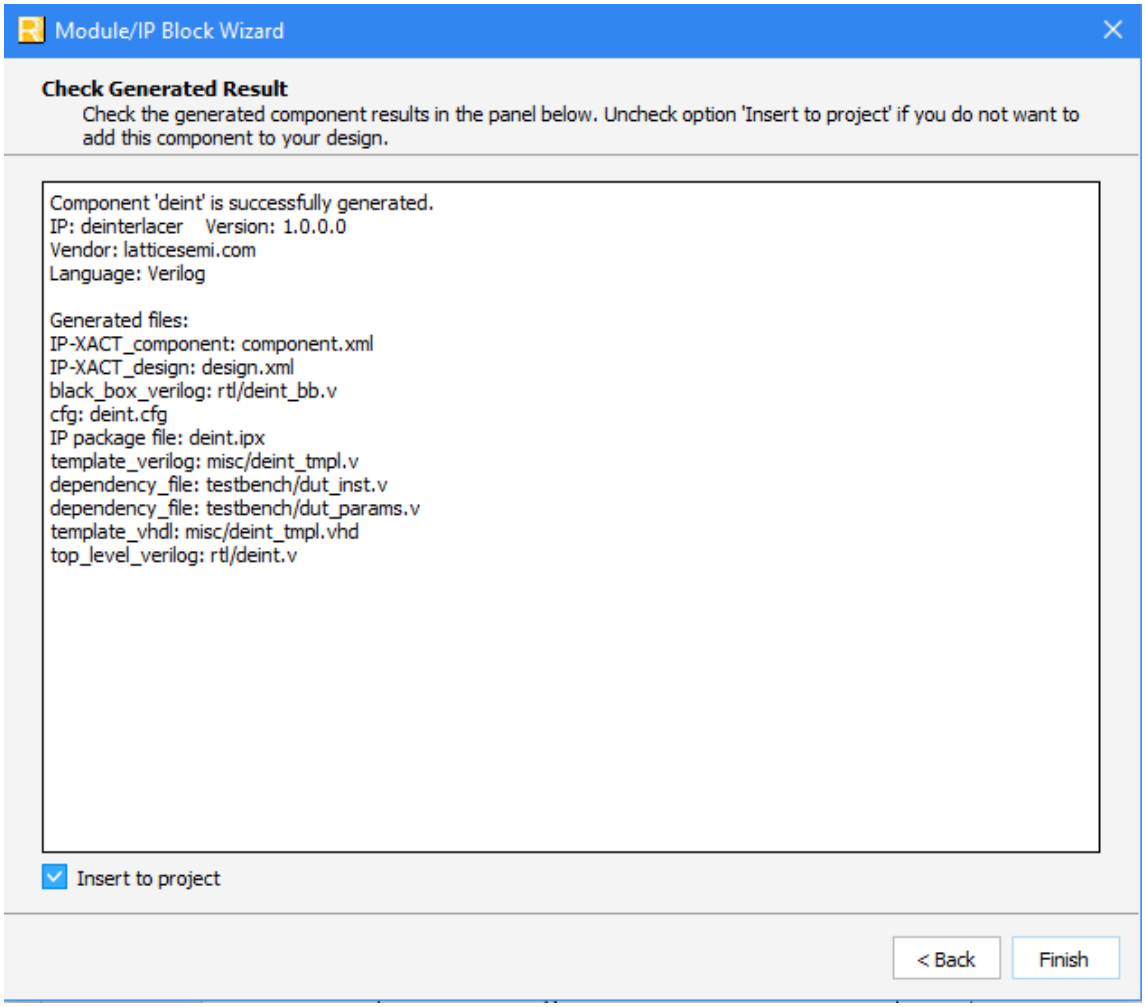

Figure 3.2. Check Generating Result

5. Click the **Finish** button to generate the Verilog file.
6. Upon generating your desired design, you can synthesize it by pressing **Synthesize Design** located in the top left corner of the screen, as shown in Figure 3.3.

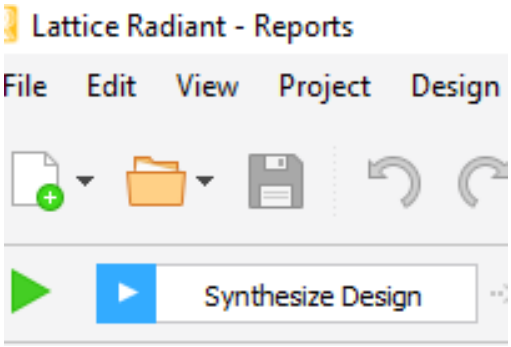

Figure 3.3. Synthesizing Design

3.2. Functional Simulation

To run the simulation, perform the following steps

1. In the generated IP directory, go to testbench folder and check the generated params.v file. The first 2 lines should contain a define that points to the generated stimulus and golden data. Make sure that the path is correct.

   `define GOLDEN_DAT "<IP path>/testbench/golden.dat"

   `define STIMULUS_DAT "<IP path>/testbench/stimulus.dat"

2. Press  button located on the Toolbar on the top of your screen to initiate Simulation Wizard, as shown in Figure 3.4.

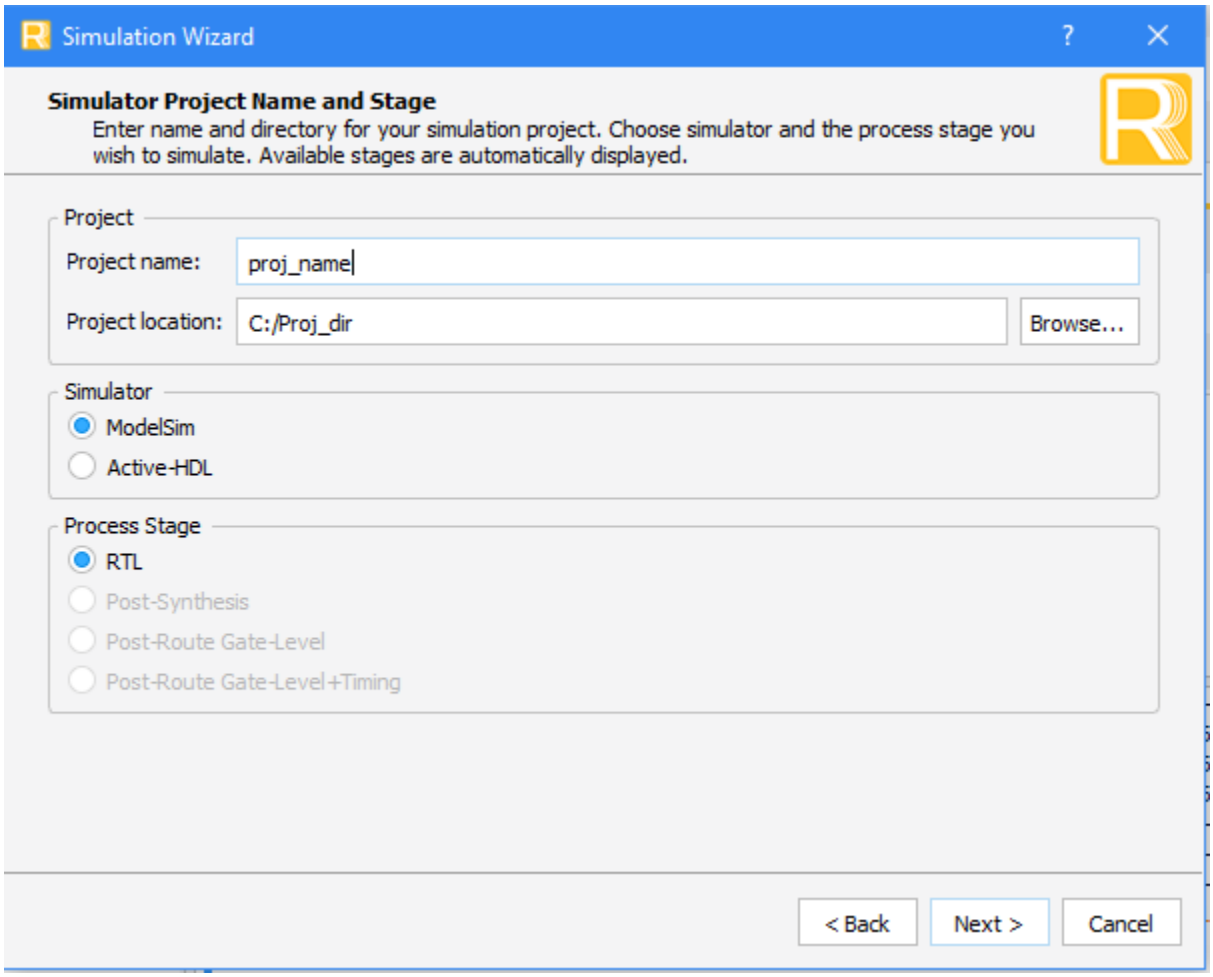

Figure 3.4. Simulation Wizard

3. Click **Next** three times and **Finish** to run simulation.

4. Ordering Part Number

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

- DLACE-CPNX-U – Deinterlacer for CertusPro-NX - Single Design License
- DLACE-CPNX-UT – Deinterlacer for CertusPro-NX - Site License

Appendix A. Resource Utilization

Table A.1. Resource Utilization

IP Configuration	SLICE	LUT	EBR	Fmax (MHz)
Default (YCbCr422)	3764	4857	7	200
Single Color, other options same as Default	2620	3369	5	200
RGB, other options same as Default	5717	7163	9	200

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

Revision 1.1, June 2021

Section	Change Summary
Introduction	Updated Table 1.1. Quick Facts. • Added CertusPro-NX product family. • Added LFCPNX-100 device. • Revised Lattice Implementation.
Ordering Part Number	Added this section.
References	Added CertusPro-NX web page

Revision 1.0, December 2020

Section	Change Summary
All	Initial release

www.latticesemi.com