Debayer IP

IP Version: 1.4.0

User Guide

FPGA-IPUG-02203-1.3

December 2025

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, with all faults, and all associated risk is the responsibility entirely of the Buyer. The information provided herein is for informational purposes only and may contain technical inaccuracies or omissions, and may be otherwise rendered inaccurate for many reasons, and Lattice assumes no obligation to update or otherwise correct or revise this information. 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. LATTICE PRODUCTS AND SERVICES ARE NOT DESIGNED, MANUFACTURED, OR TESTED FOR USE IN LIFE OR SAFETY CRITICAL SYSTEMS, HAZARDOUS ENVIRONMENTS, OR ANY OTHER ENVIRONMENTS REQUIRING FAIL-SAFE PERFORMANCE, INCLUDING ANY APPLICATION IN WHICH THE FAILURE OF THE PRODUCT OR SERVICE COULD LEAD TO DEATH, PERSONAL INJURY, SEVERE PROPERTY DAMAGE OR ENVIRONMENTAL HARM (COLLECTIVELY, "HIGH-RISK USES"). FURTHER, BUYER MUST TAKE PRUDENT STEPS TO PROTECT AGAINST PRODUCT AND SERVICE FAILURES, INCLUDING PROVIDING APPROPRIATE REDUNDANCIES, FAIL-SAFE FEATURES, AND/OR SHUT-DOWN MECHANISMS. LATTICE EXPRESSLY DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY OF FITNESS OF THE PRODUCTS OR SERVICES FOR HIGH-RISK USES. 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.

Inclusive Language

This document was created consistent with Lattice Semiconductor’s inclusive language policy. In some cases, the language in underlying tools and other items may not yet have been updated. Please refer to Lattice’s inclusive language FAQ 6878 for a cross reference of terms. Note in some cases such as register names and state names it has been necessary to continue to utilize older terminology for compatibility.

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
AXI	Advanced extensible Interface
BPP	Bits Per Pixel Color Component
CFA	Color Filter Array
CSR	Configuration and Status Register
DSP	Digital Signal Processor
EBR	Embedded Block RAM
FIFO	First-In First-Out
HDL	Hardware Description Language
IP	Intellectual Property
LUT	Look-Up Table
PPC	Pixels Per Clock
RGB	Red Green Blue
RTL	Register Transfer Level
RX	Receiver
TX	Transmitter

1. Introduction

The Debayer IP provides AXI4-Stream interfaces for the input and the output video streams. The IP can process up to 16 bits per pixel (BPP) and 4 pixels per clock (PPC). An optional AXI4-Lite interface is provided to dynamically configure the parameters and to read out the status of the IP.

This user guide provides a description of the Debayer IP. CMOS color image sensors do not capture all the three-color components for each pixel, but only one of the three color components for any pixel. Since green is the dominant component that closely captures the luminescence compared to red or blue components, half of the pixels in a sensor capture green component while a fourth of the sensor pixels are used to capture each blue or red component. The selective color component sensing is achieved by having a special color filter over the photo sensors, called as Bayer filter, named after Doctor Bayer, a scientist in Kodak. The process of interpolating and recreating the missing color components not captured by the sensor is called as Debayering, or De-Mosaicing or Color Filter Array (CFA) Interpolation. After debayering, each pixel is represented by all the three color components.

1.1. Quick Facts

Table 1.1 presents a summary of the Debayer IP.

Table 1.1. Quick Facts

IP Requirements	Supported Devices	CrossLink™-NX, Certus™-NX (LFD2NX-17, LFD2NX-40, LFD2NX-25, LFD2NX-28), CertusPro™-NX, Lattice Avant™, Certus-N2
	IP Changes[1]	For a list of changes to the IP, refer to the Debayer IP Release Notes (FPGA-RN-02054).
Resource Utilization	Resources	See Appendix A. Resource Utilization.
Design Tool Support	Lattice Implementation	IP v1.4.0 – Lattice Radiant™ software 2025.2
	Synthesis	Lattice Synthesis Engine
		Synopsys® Synplify Pro® for Lattice
	Simulation	For a list of supported simulators, see the Lattice Radiant software user guide.

Note:

1. In some instances, the IP may be updated without changes to the user guide. This user guide may reflect an earlier IP version but remains fully compatible with the later IP version. Refer to the IP Release Notes for the latest updates.

1.2. Features

The following are the key features of the Debayer IP:

- Supports 6, 8, 10, 12, and 16 bits per pixel
- Supports 1, 2, and 4 pixels per clock
- Supports AXI4-Stream Protocol to receive and send pixel information
- Supports AXI4-Lite Protocol to configure and control the IP

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. Algorithms for Debayer

There are several algorithms used for Debayering today including the following popular ones:

- Super pixel
- Nearest Neighborhood
- Bilinear interpolation
- Bicubic interpolation
- Variable Number of Gradients (VNG)
- Smooth hue transition interpolation

Bilinear interpolation is both simplest to implement and has a reasonably good performance. VNG group of algorithms alleviate the smoothed edges problem seen in bilinear interpolation. The method involves finding the gradient of the pixels in all directions and determining high gradient direction, so they can be avoided in calculating the interpolated values. A typical VNG algorithm uses a 5 × 5 kernel and interpolates all the missing color components in one stage.

2.2. Block Diagram

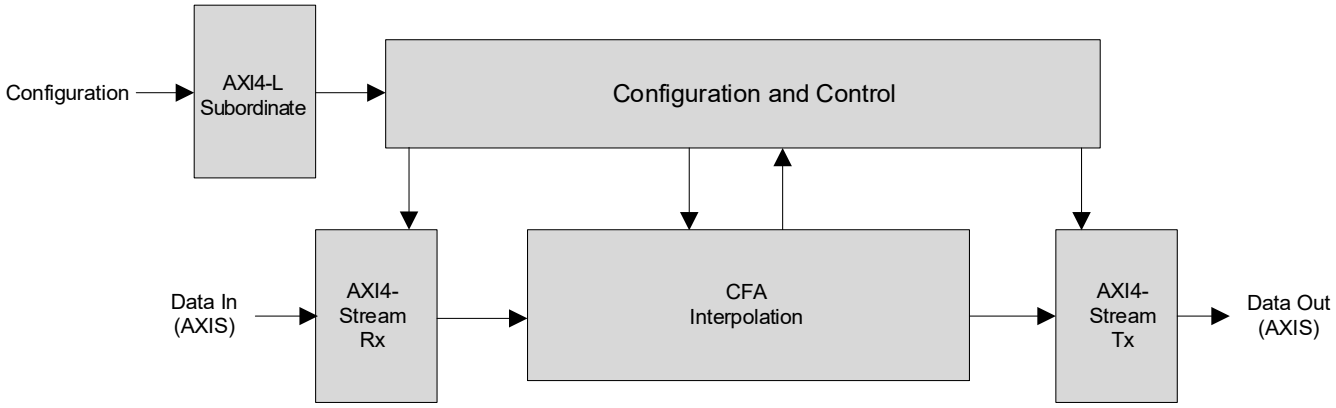

Figure 2.1. Debayer IP Architectural Diagram

The data comes in through the AXI-Stream interface, gets processed and goes out through AXI-Stream.

The following are the essential elements of the Debayer hardware architecture:

- AXI-Stream is used for video data streaming in and out. AXI-Stream receiver receives the video input from the sensor in Bayer domain.
- The input/output data includes pixel data as well as video sync signals. The video sync signals are passed through the user-defined TUSER bus as sideband signals.
- The IP also provides a pass through mode to dynamically bypass Debayer when switched to monochrome sensor.
- The central function of Debayer is converting the Bayer input to an RGB output. This operation typically uses the 5 × 5 pixel array surrounding the pixel to be interpolated.
- An optional AXI-Lite interface is used for the configuration path to write and read configuration and status registers (CSR).

2.3. AXI-Stream Receiver

This block is the protocol receiver for AXI-Stream. The control inputs (sync signals) are sent in through the user-defined sideband bus, TUSER. The data width of AXI-Stream has to be a multiple of 8, with powers of two recommended by the specifications. The number of bytes in TDATA is determined based on pixel width and number pixels per clock and the pixel data is packed across these bytes. The receiver processes the appropriate data bytes in the bus based on the strobe input.

On the AXI-Stream handshaking, note that the Transmitter is not permitted to wait until TREADY is asserted before asserting TVALID. This implies that the Transmitter must independently assert TVALID without reference to TREADY. Also, the standard requires that TVALID, once asserted, must remain asserted until the handshake occurs.

The AXI-Stream handshake signals and the user defined TUSER bus are described below:

- The user defined sideband signal TUSER is used to specify the frame boundaries and active data windows. Specifically, rx_tuser_i[0] indicates start of frame, with this bit asserting high during the first pixel of a frame. The input rx_tuser_i[1] specifies the active part of a line, with a value of 1 indicating the active part and 0, the blanking part. The input rx_tuser_i[2] indicates end of frame. This signal must be high during the last active pixel of a frame.
- The input rx_tlast_i indicates end of a Total line. This must be asserted during the last pixel of horizontal blanking.
- The rx_tready_o signal is always set to *1*, except when the Configuration and Status Registers (CSR) are being updated, at which time it is set to *0*.
- The rx_tstrobe_i signal defines the valid bytes in a data beat, that is, the valid bytes in rx_tdata_i.
- The width of rx_tdata_i depends on the parameters BPP and PPC. The width of this bus is equal to the next higher multiple of 8 of the quantity (PPC × BPP).

The timing diagrams for the AXI-Stream receiver for different PPC and BPP values are shown in Figure 2.2, Figure 2.3, and Figure 2.4.

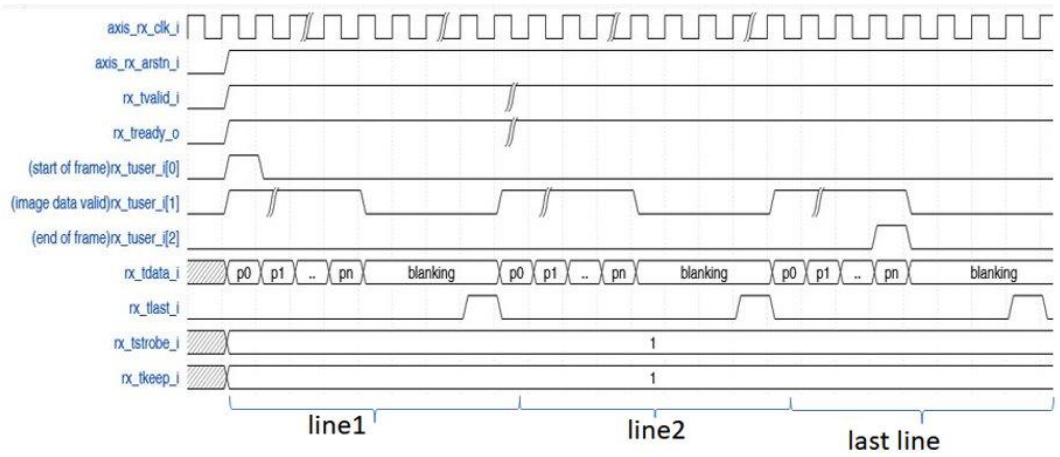

Figure 2.2. AXI-Stream Receiver Signals for PPC=1 and BPP=8 (Bayer Pattern)

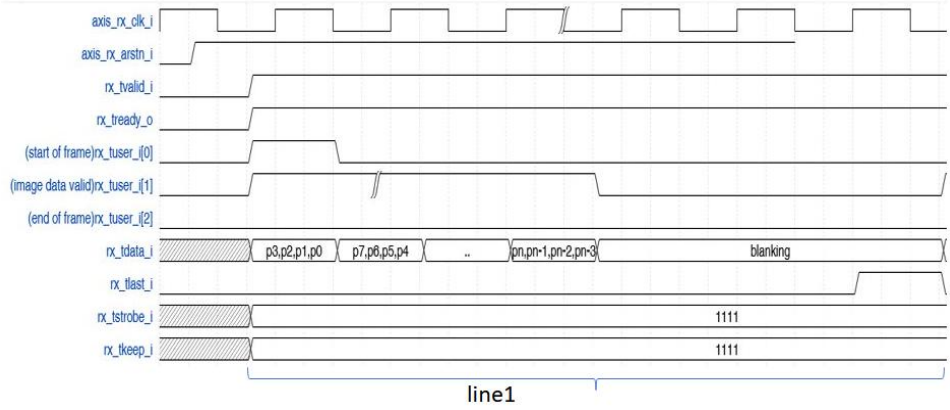

Figure 2.3. AXI-Stream Receiver Signals for PPC=4 and BPP=8 (Bayer Pattern)

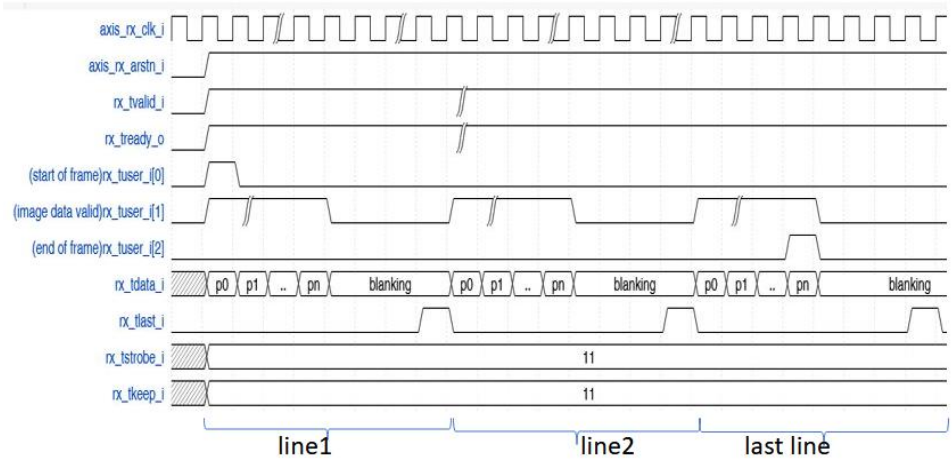

Figure 2.4. AXI-Stream Receiver Signals for PPC=1 and BPP=16 (Bayer Pattern)

The organization of the color component data in the TDATA bus is described here. The mapping of bits for the case of PPC=1 and BPP=6 is shown in Figure 2.5. The mappings for other cases are shown in Figure 2.6 to Figure 2.10.

AXI-Stream bus width must be divisible by 8. Pixel data of (PPC × BPP) bits is mapped to the lower part of the bus, with the remaining upper bits assigned to zeros.

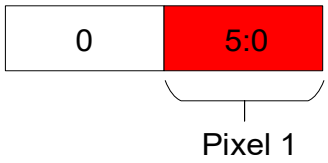

Figure 2.5. Bit Allocation for BPP=6 and PPC=1 for RGGB Input

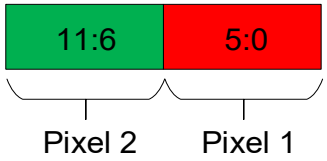

Figure 2.6. Bit Allocation of BPP=6 and PPC=2 for RGGB Input

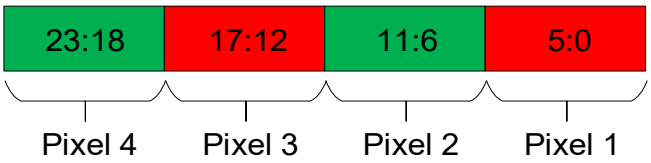

Figure 2.7. Bit Allocation for BPP=6 and PPC=4 for RGGB Input

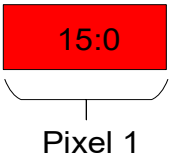

Figure 2.8. Bit Allocation for BPP=16 and PPC=1 for RGGB Input

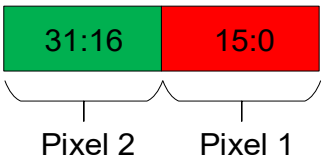

Figure 2.9. Bit Allocation for BPP=16 and PPC=2 for RGGB Input

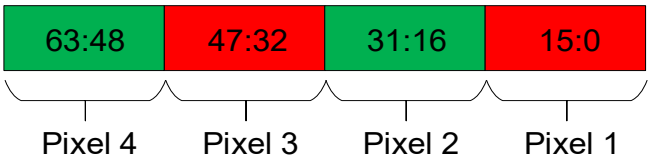

Figure 2.10. Bit Allocation for BPP=16 and PPC=4 for RGGB Input

2.4. RGB Data Packing

The RGB output of the CFA interpolation block is into single multi-byte bus before sending out to AXI-Stream transmitter. The timing diagram for RGB packing for 1 PPC, 8 BPP configuration is shown in Figure 2.11.

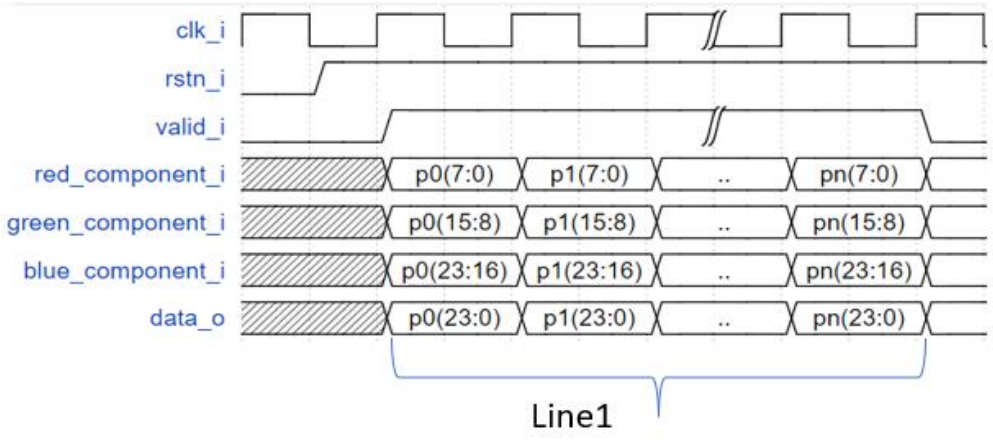

Figure 2.11. Packing of Color Components in RGB Pattern for BPP=8 and PPC=1

2.5. AXI-Stream Transmitter

This module maps the video data back to bytes for transmission via AXI-Stream to the next stage pipeline.

The transmitter sends out the data in RGB format. Here, the red is located in the lower bits followed by green and blue. The user defined bus tx_tuser_o is used to transmit sync signals. The AXI-Stream transmitter sends out data only when tx_tready_i is high. If tx_tready_i goes low, the data is stored in FIFO up to parameter *Tx buffer depth* configured through the IP user interface. If tx_tready_i is deasserted for more than *Tx buffer depth* cycles, the frame is discarded.

The use of AXI-Stream handshake signals and user-defined TUSER bus are described below:

- The user defined sideband signal TUSER is used to specify the frame boundaries and data validity. Specifically, tx_tuser_o[0] indicates start of frame, with this bit asserting high during the first pixel of a frame. The output tx_tuser_o[1] specifies the active part of a line, with a value of 1 indicating the active part and 0, the blanking part. The output tx_tuser_o[2] indicates end of frame. This signal is high during the last active pixel of a frame.
- The output tx_tlast_o indicates end of a total line. This is asserted during the last pixel of total line.
- The tx_tstrobe_o signal defines the valid bytes in a data beat, that is, the valid bytes in tx_tdata_o.

- The width of tx_tdata_o depends on the parameters PPC and BPP. The width of this bus is equal to the next higher multiple of 8 of the quantity (PPC × BPP × 3).
- The timing diagram of AXI-Stream Transmitter are shown in Figure 2.12 and Figure 2.13.

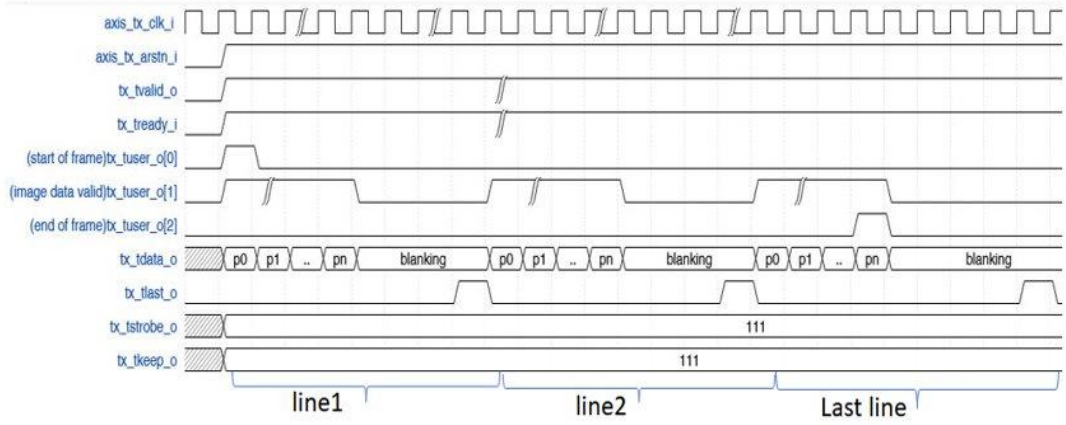

Figure 2.12. AXI-Stream Transmitter with PPC=1 and BPP =8

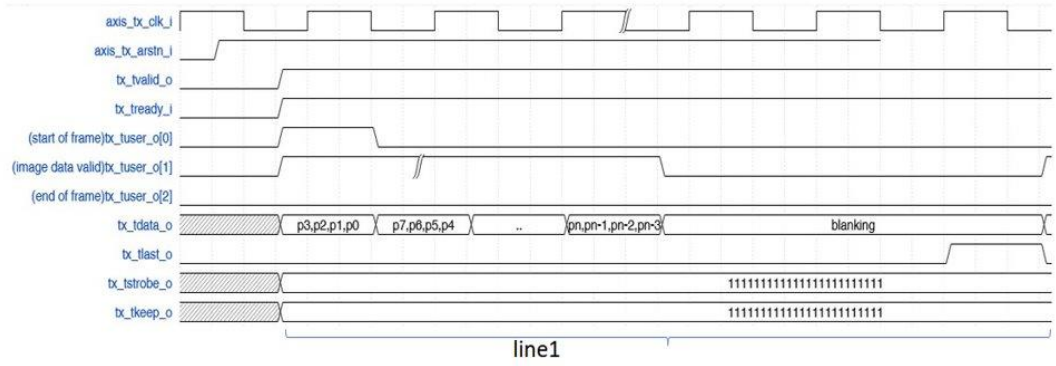

Figure 2.13. AXI-Stream Transmitter with PPC=1 and BPP = 16 for RGB Pattern

The organization of the color component data in TDATA bus depends on the PPC and BPP parameter values. In general, the red component is mapped to the lower bits, green to the middle bits and blue to the upper bits.

AXI-Stream bus width must be divisible by 8. Pixel data of (BPP × PPC × 3) bits is mapped to the lower part of the bus, with the remaining upper bits assigned to zeros. The mapping of bits for the case of PPC=1 and BPP=6 is shown in Figure 2.14. The mappings for other cases are shown in Figure 2.15 to Figure 2.19.

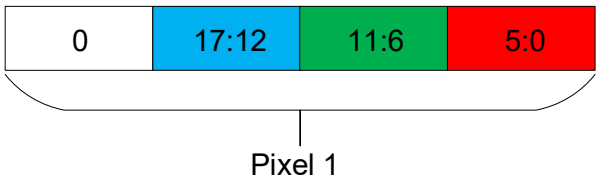

Figure 2.14. Bit Allocation for BPP=6 and PPC=1

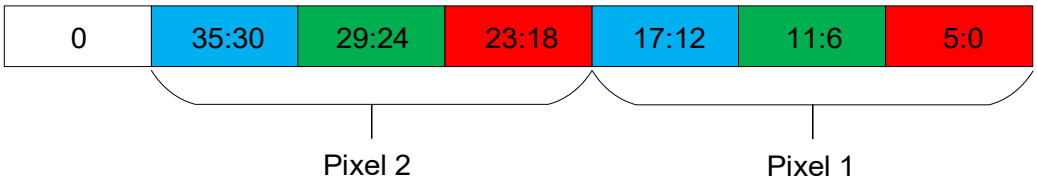

Figure 2.15. Bit Allocation of BPP=6 and PPC=2

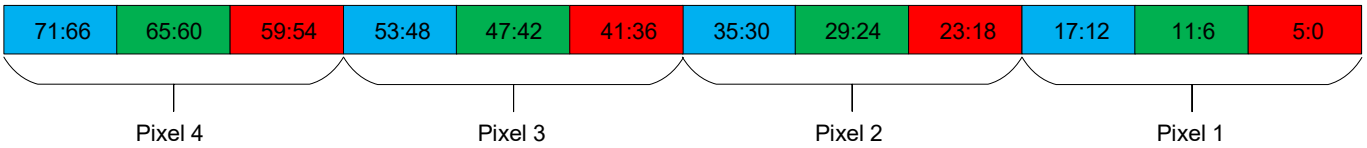

Figure 2.16. Bit Allocation for BPP=6 and PPC=4

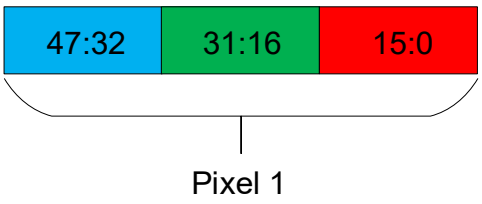

Figure 2.17. Bit Allocation for BPP=16 and PPC=1

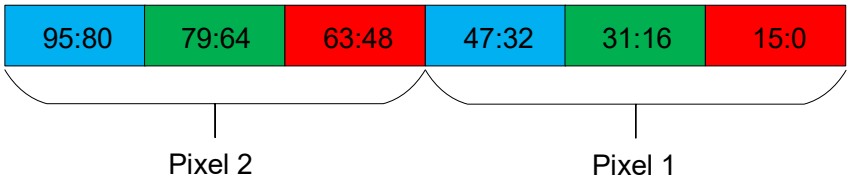

Figure 2.18. Bit Allocation for BPP=16 and PPC=2

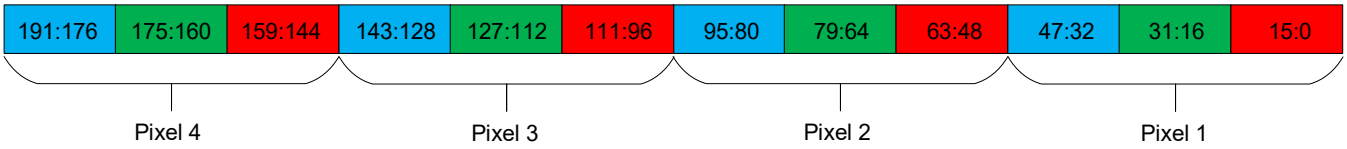

Figure 2.19. Bit Allocation for BPP=16 and PPC=4

2.6. Configuration and Control

This block is used to dynamically configure the IP parameters. This block contains the AXI-Lite subordinate interface and a number of registers to store the dynamically configurable parameters. The parameters that are supported for dynamic configuration are:

- Enable/Disable Debayer
- Bayer Pattern
- Configuration Update

For details on the configuration register, refer to Table 4.1.

2.7. AXI-Lite Subordinate

2.7.1. Description

AXI-Lite consists of five channels namely write address, write data, write response, read address, and read data. The following are the key signals in each channel.

Table 2.1. AXI-Lite Channels and Signals

Channel	Key Signals
Write address	aw_valid_i, aw_address_i, aw_ready_o
Write data	w_valid_i, w_data_i, w_ready_o
Write response	b_valid_o, b_response_o, b_ready_i
Read address	ar_valid_i, ar_address_i, ar_ready_o
Read data	r_data_o, r_valid_o, r_response_o, r_ready_i

2.7.2. Write Operation

This is the sequence for a write operation:

- The manager asserts aw_valid_i along with address.
- The subordinate asserts the aw_ready_i and captures the address.
- The manager asserts w_valid_i along with write data.
- The subordinate asserts w_ready_i and captures the data.
- The subordinate then sends an OKAY response and asserts b_valid_o.
- Manager asserting b_ready_i completes the transaction.

2.7.3. Read Operation

This is the sequence for a read operation:

- The manager asserts ar_valid_i signal along with the address.
- The subordinate captures the address by asserting the ar_ready_i.
- The subordinate then sends the r_data_o along with r_valid_o and r_response_o.
- The r_ready_i signal assertion from the manager completes the transaction.

3. Signal Description

The widths used for the signal buses in the interface table are defined in Table 3.1. The input-output interface signals for Debayer IP are given in Table 3.2.

Table 3.1. Width Parameter Description

Width Parameter Name	Description
AXI_STREAM_IN_DATA_WIDTH	Width of AXI-Stream bus is equal to next higher multiple of 8 of the quantity (PPC × BPP).
AXI_STREAM_OUT_DATA_WIDTH	Width of AXI-Stream bus is equal to next higher multiple of 8 of the quantity (PPC × BPP × 3).
AXI_LITE_ADDR_WIDTH	2
AXI_LITE_DATA_WIDTH	32

Table 3.2. Debayer IP Signal Description

Port Name	I/O	Size	Description
Clock and Reset			
axi_lite_rst_n_i	I	1	AXI-Lite reset (Active low)
axis_rx_arstn_i	I	1	AXI-Stream RX Reset (Active low)
axis_tx_arstn_i	I	1	AXI-Stream TX Reset (Active low)
axi_lite_clk_i	I	1	AXI-Lite clock
axis_rx_clk_i	I	1	AXI-Stream RX Clock
axis_tx_clk_i	I	1	AXI-Stream TX Clock
AXI-Stream Rx			
rx_tdata_i	I	[AXI_STREAM_IN_DATA_WIDTH-1:0]	TDATA is the primary payload that is used to provide the data that is passing across the interface.
rx_tuser_i	I	3	Sideband information transmitted alongside the data stream. rx_tuser_i[0] - start of frame rx_tuser_i[1] - active image data rx_tuser_i[2] - end of frame
rx_tlast_i	I	1	TLAST indicates the end of line for each horizontal line.
rx_tvalid_i	I	1	TVALID indicates that the Tx is driving a valid transfer. A transfer takes place when both TVALID and TREADY are asserted.
rx_tstrobe_i	I	[AXI_STREAM_IN_DATA_WIDTH-1:0]/8	TSTROBE is the byte qualifier that indicates whether the content of the associated byte of is processed as a data byte or a position byte.
rx_tkeep_i	I	[AXI_STREAM_IN_DATA_WIDTH-1:0]/8	TKEEP is the byte qualifier that indicates whether the content of the associated byte of TDATA is processed as part of the data stream. Associated bytes that have the TKEEP byte qualifier deasserted are null bytes and can be removed from the data stream.
rx_tready_o	O	1	TREADY indicated that the Rx can accept a transfer in the current cycle.
AXI-Stream Tx			
tx_tvalid_o	O	1	TVALID indicates that theTransmitter is driving a valid transfer. A transfer takes place when both TVALID and TREADY are asserted.
tx_tready_i	I	1	TREADY indicated that the rx can accept a transfer in the current cycle.
tx_tdata_o	O	[AXI_STREAM_OUT_DATA_WIDTH-1:0]	TDATA is the primary payload that is used to provide the data that is passing across the interface.
tx_tuser_o	O	3	Sideband information transmitted alongside the data stream. rx_tuser_i[0] – start of frame rx_tuser_i[1] – valid image data rx_tuser_i[2] – end of frame

Port Name	I/O	Size	Description
tx_tlast_o	O	1	TLAST indicates the end of line for each horizontal line.
tx_tstrobe_o	O	[AXI_STREAM_OUT_DATA_WIDTH-1:0]/8	TSTROBE is the byte qualifier that indicates whether the content of the associated byte of is processed as a data byte or a position byte.
tx_tkeep_o	O	[AXI_STREAM_OUT_DATA_WIDTH-1:0]/8	TKEEP is the byte qualifier that indicates whether the content of the associated byte of TDATA is processed as part of the data stream. Associated bytes that have the TKEEP byte qualifier deasserted are null bytes and can be removed from the data stream.
AXI-Lite Subordinate			
aw_valid_i	I	1	Write address valid. This signal indicates that the channel is signaling valid write address and control information.
aw_address_i	I	[AXI_LITE_ADDR_WIDTH-1 :0]	Write address
aw_ready_o	O	1	Write address ready. This signal indicates that the subordinate is ready to accept an address and associated control signals.
aw_prot_i	I	[2:0]	Protection type. This signal indicates the privilege and security level of the transaction, and whether the transaction is a data access or an instruction access (not used).
w_valid_i	I	1	Write valid. This signal indicates valid write data.
w_data_i	I	[AXI_LITE_DATA_WIDTH-1 :0]	Write data
w_strobe_i	I	[AXI_LITE_DATA_WIDTH-1 :0]	Write strobes. This signal indicates which byte lanes hold valid data. There is one write strobe bit for each eight bits of the write data bus (not used).
w_ready_o	O	1	Write ready. This signal indicates that the subordinate can accept the write data
b_valid_o	O	1	Write response valid. This signal indicates that the channel is signaling a valid write response
b_response_o	O	2	Write response. This signal indicates the status of the write transaction
b_ready_i	I	1	Response ready. This signal indicates that the manager can accept a write response
ar_valid_i	I	1	Read address valid. This signal indicates that the channel is signaling valid read address
ar_address_i	I	[AXI_LITE_ADDR_WIDTH-1 :0]	Read address
ar_prot_i	I	[2:0]	Protection type. This signal indicates the privilege and security level of the transaction, and whether the transaction is a data access or an instruction access (not used).
ar_ready_o	O	1	Read address ready. This signal indicates that the subordinate is ready to accept an address
r_data_o	O	[AXI_LITE_DATA_WIDTH-1 :0]	Read data.
r_valid_o	O	1	Read valid. This signal indicates that the channel is signaling the required read data
r_response_o	O	2	Read response. This signal indicates the status of the read transfer
r_ready_i	I	1	Read ready. This signal indicates that the manager can accept the read data and response Information.

4. Attributes Summary

Figure 4.1 shows the sample user interface for the Debayer IP and the attributes set through the user interface are described in Table 4.1.

Table 4.1. Attributes Table

Attribute	Selectable Values	Default	Dependency on Other Attributes	Description
AXI-Lite				
Bits per pixel	6, 8, 10, 12, 16	8	—	Number of bits per color component of a pixel
Pixels per clock	1, 2, 4	1	—	Number of pixels streamed in or out in a clock
Enable Dynamic Configuration	Enabled, Disabled	Enabled	—	Whether Configuration and Status Registers (CSR) and AXI-Lite interface are required
Partial Resolution	Enabled, Disabled	Disabled	—	Partial strobe support for last pixel of a line for PPC2 and PPC4, when resolution is not divisible by PPC.
Horizontal Size	[512–4096]	1920	—	Horizontal size of a line
FIFO				
Tx buffer depth	128, 256, 512, 1024	1024	—	This value describes the FIFO depth. If tx_tready_i is deasserted for more than *TX buffer depth* cycles, the entire frame gets skipped. If tx_tready_i is asserted before the Tx buffer depth cycles, then the data flow is not be interrupted.
Blanking buffer depth	128, 512, 1024, 2048, 4096, 8192, 10240	8192	—	This value describes the embedded data buffer depth. This value must be set to at least twice the total horizontal size consisting of active line size and horizontal blanking, rounded up to the nearest power of 2.
CSR				
Bayer Pattern	RGGB, BGGR, GRBG, GBRG	RGGB	—	Different Bayer patterns are selectable
Debayer Enable	Enabled, Disabled	Enabled	If the “Enable Dynamic Configuration” is turned on Debayer Enable is editable, otherwise it is always turned on.	Enable (Debayer operation) Disable (the output pixel data is the same as the input data)

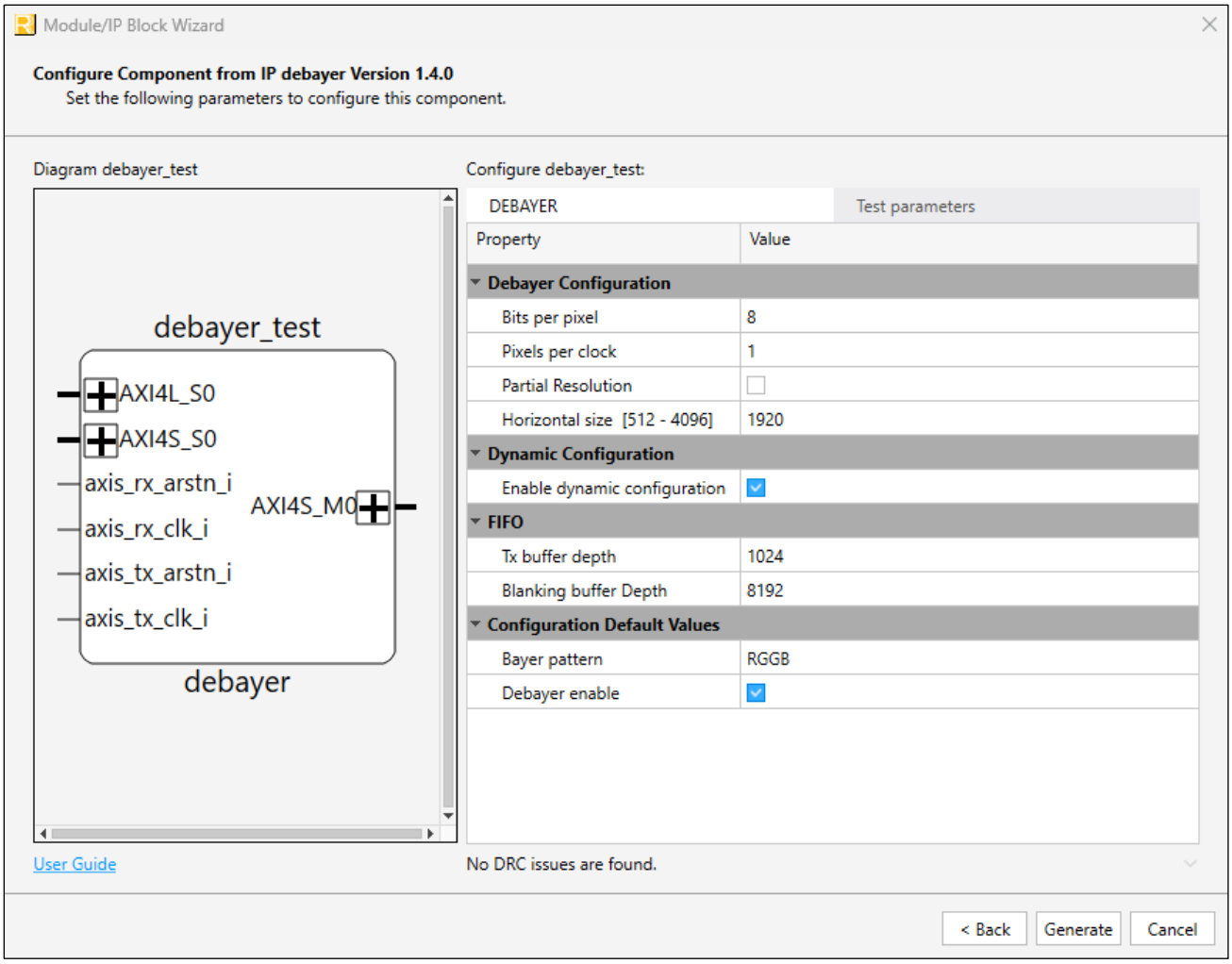

Figure 4.1. Configuration User Interface for Debayer IP

5. Register Description

The registers used in the CSR module of the Debayer IP are shown in Table 5.1. The CSR registers are accessible through the AXI-Lite interface.

Table 5.1. Summary of Configuration and Status Registers

Offset	Register Name	Access	Default Value	Register Description		
				Bit	Field	Field Description
Attribute Registers						
0×00	BAYER_PATTERN	RW	00	[1:0]	bayer_pattern	00 – RGGB pattern 01 – BGGR pattern 10 – GRBG pattern 11 – GBRG pattern
				[31:2]	RSVD	Reserved bits
0×1	DEBAYER_ENABLE	RW	0	0	debayer_enable	0 – enable (Debayer operation) 1 – disable (the output pixel data is the same as the input data)
				[31:1]	RSVD	Reserved bits
0×2	CONFIGURATION_UPDATE	RW	0	0	config_update	The user must make config_update as 1 after updating all the other registers. The IP clears this register after reading configuration values. If the user updates the configuration registers in middle of an active frame, the updated values will be read in during blanking period for the next frame.
				[31:1]	RSVD	Reserved bits

6. IP Generation, Simulation, and Validation

This section provides information on how to generate the Debayer IP 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.

Note: The screenshots provided are for reference only. Details may vary depending on the version of the IP or software being used. If there have been no significant changes to the GUI, a screenshot may reflect an earlier version of the IP.

6.1. Licensing the IP

The Debayer IP is provided at no additional cost with the Lattice Radiant software.

6.2. Generation and Synthesis

The Lattice Radiant software allows the user to customize and generate modules and IPs and integrate them into the device's architecture. The procedure for generating the Debayer IP in Lattice Radiant software is described below.

To generate the Debayer IP:

1. Create a new Lattice Radiant software project or open an existing project.
2. In the **IP Catalog** tab, double-click on Debayer under **Audio_Video_and_Image_Processing** category. The **Module/IP Block Wizard** opens as shown in Figure 6.1. Enter values in the **Component name** and the **Create in** fields and click **Next**.
3. Go to *RadiantIPLocal**debayer**testbench**tb_settings.v*. Update the file path parameter accordingly. Make sure the generated user interface matches the testbench parameters.

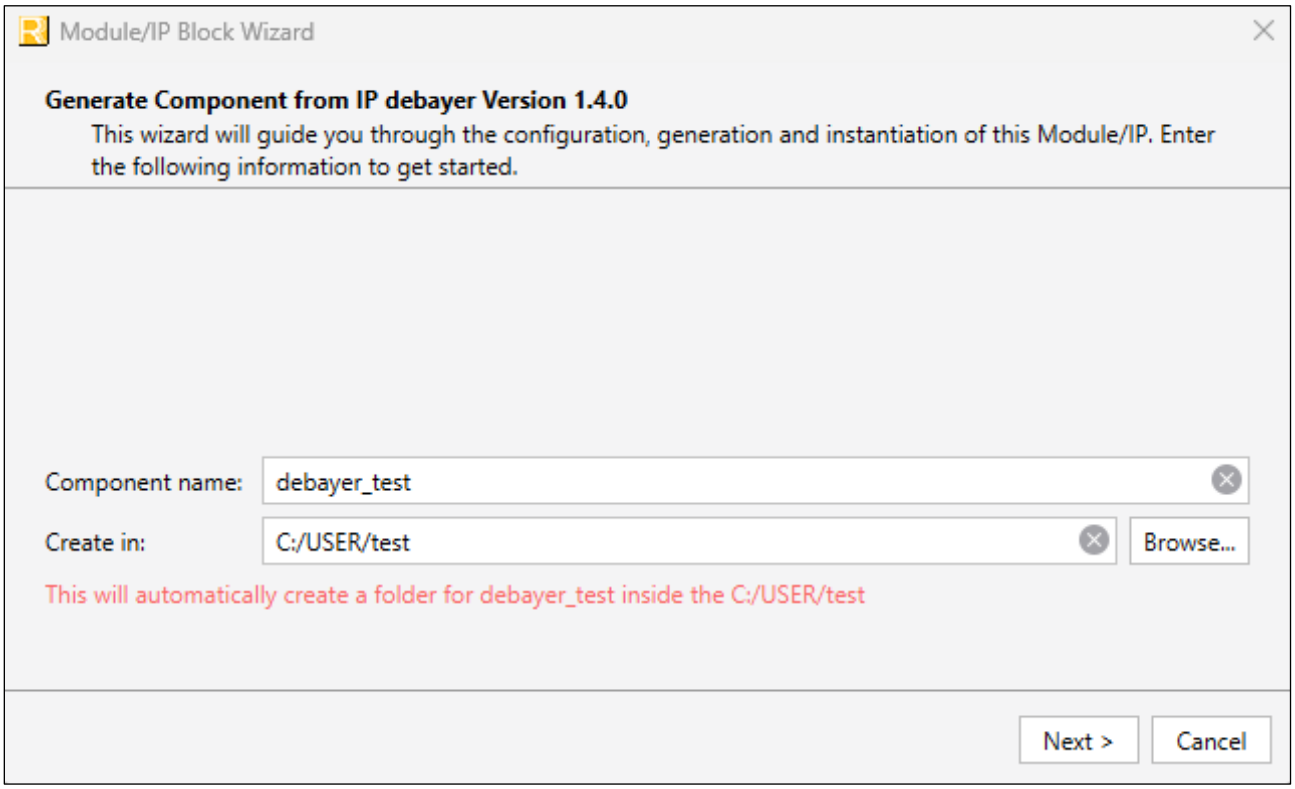

Figure 6.1. Module/IP Block Wizard

4. In the module's dialog box of the **Module/IP Block Wizard** window, customize the selected Debayer IP using drop-down menus and check boxes. Figure 4.1 shows the sample configuration. For configuration options, see Table 3.2.
5. Click **Generate**. The **Check Generated Result** dialog box opens, showing design block messages and results as shown in Figure 6.2.

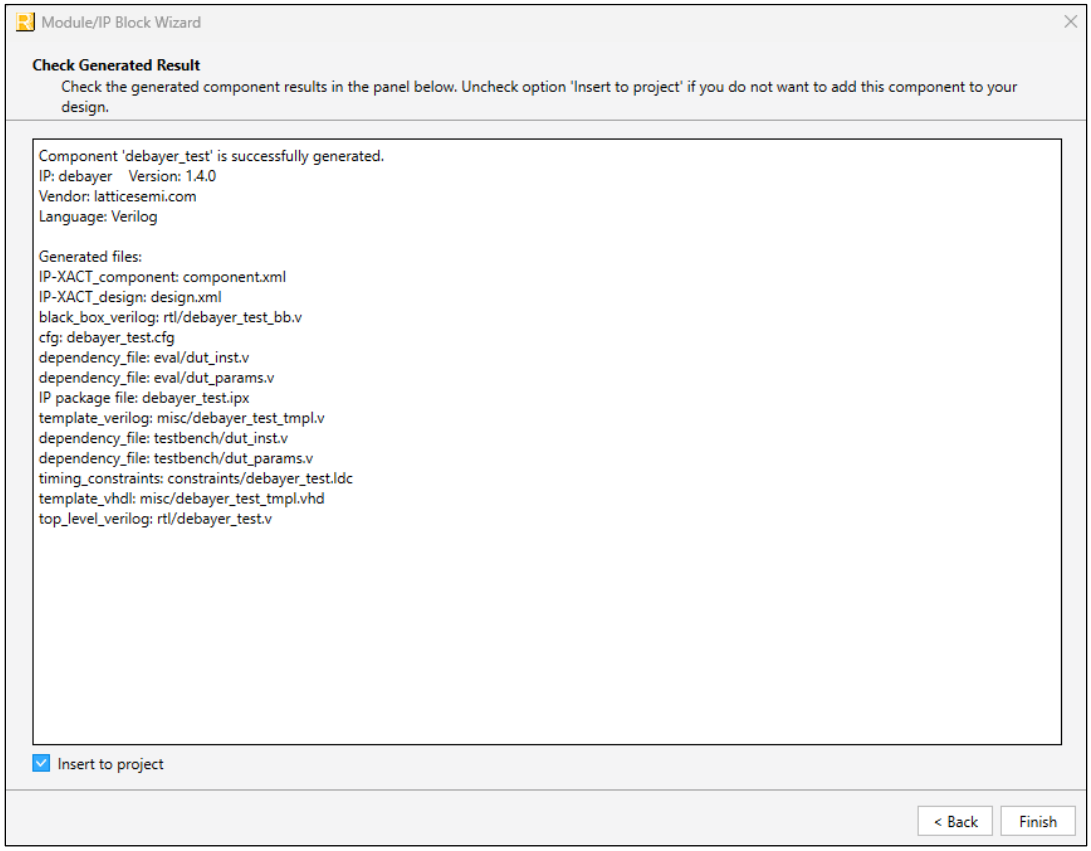

Figure 6.2. Check Generated Result

6. 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 6.1.

The generated Debayer IP package includes the closed-box (<Component name>_bb.v). 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. The user 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 6.1.

Table 6.1. Generated File List

Attribute	Description
<Component name>.ipx	This file contains the information on the files associated to the generated IP.
<Component name>.cfg	This file 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	This file provides an example RTL top file that instantiates the IP core.
rtl/<Component name>_bb.v	This file provides the synthesis closed-box.
eval/<component_name>.pdc	This file contains post-synthesis constraints for evaluating the IP implementation flow. This may include virtual I/O and system level clock constraints.
eval/*	This folder contains other files that may be used to evaluate the IP, including *sty (implementation strategy file) or files that are used by the testbench for simulation

6.3. Running Functional Simulation

Running functional simulation can be performed after the IP is generated. The following steps can be performed.

1. Click the  icon located on the **Toolbar** to initiate the **Simulation Wizard** shown in Figure 6.3.

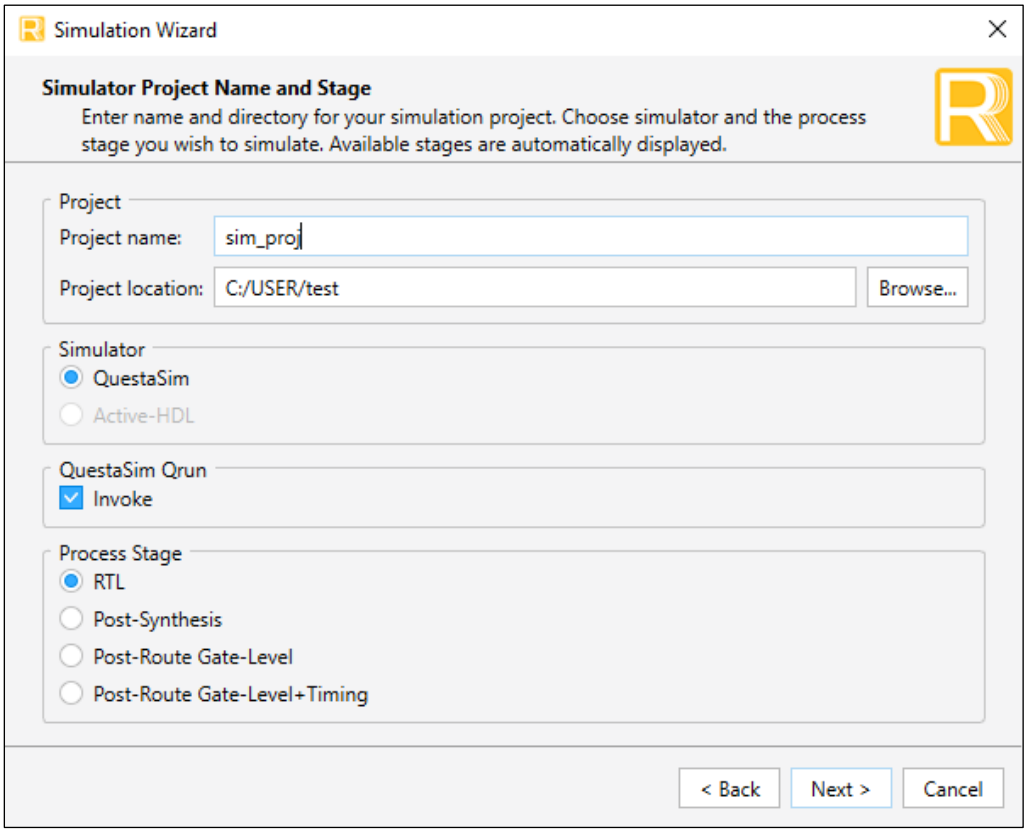

Figure 6.3. Simulation Wizard

2. Once done, it should show up in the simulation wizard, enter a test name and proceed to the next step.
3. Click **Next** to open the **Add and Reorder Source** window as shown in Figure 6.4.

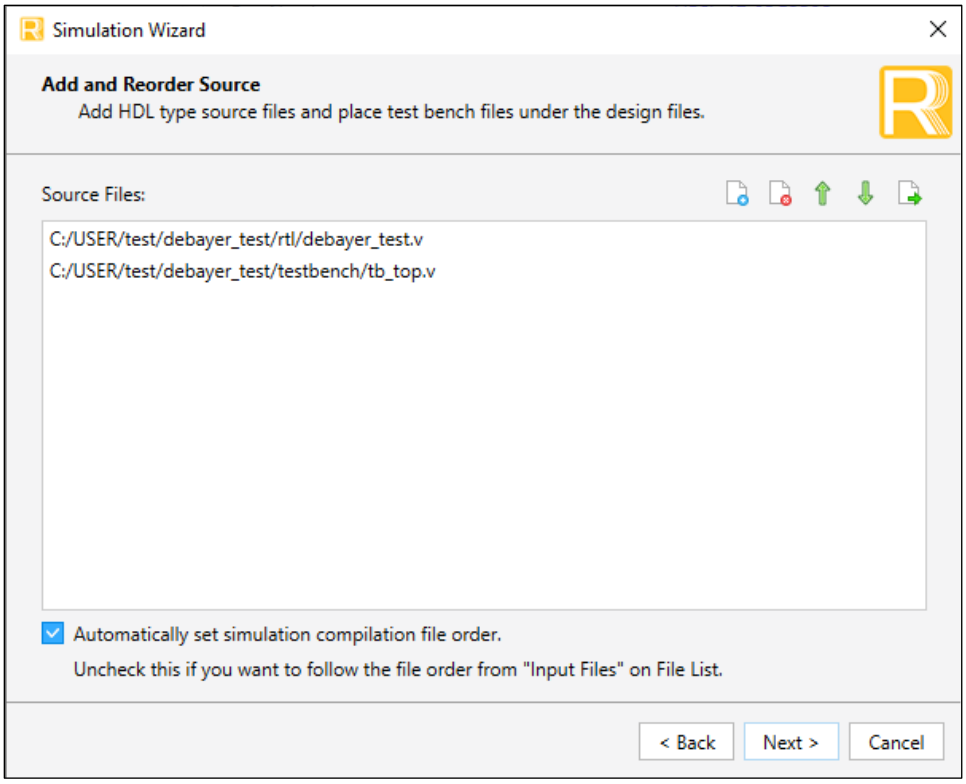

Figure 6.4. Adding and Reordering Source

4. Change the simulation time to 0 ns, means run -all.

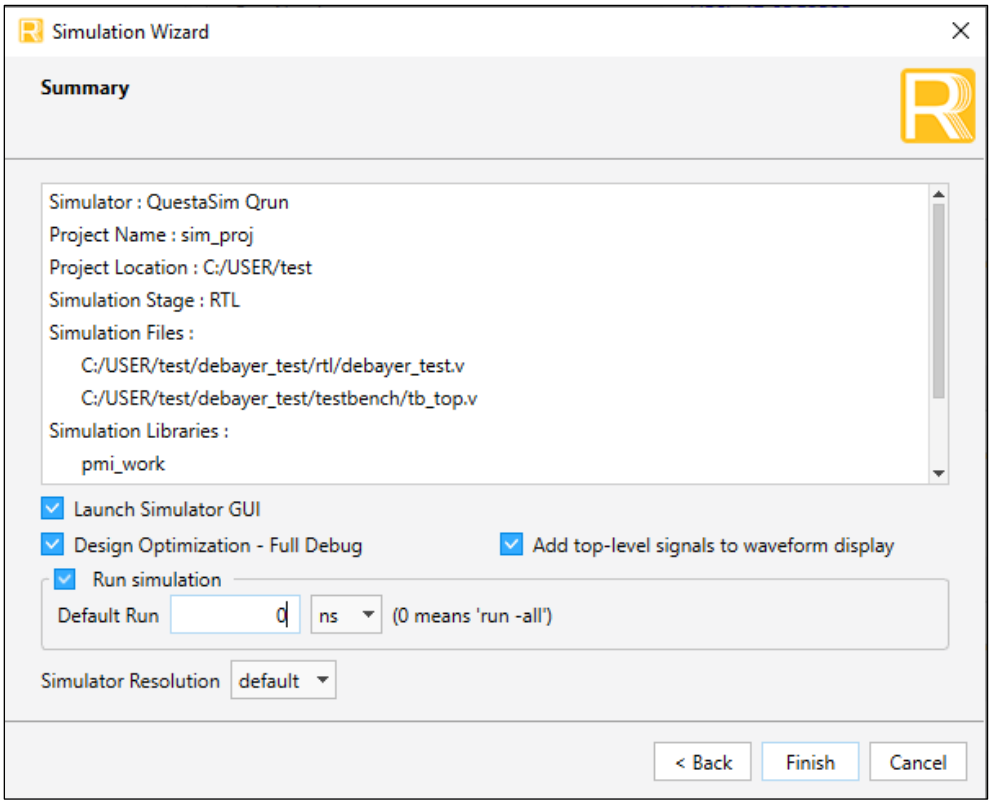

Figure 6.5. Run Time Setting to 0 ns

5. Click **Next**. The Summary window is shown. Click **Finish** to run the simulation. The result of the simulation in the example is provided in Figure 6.6.

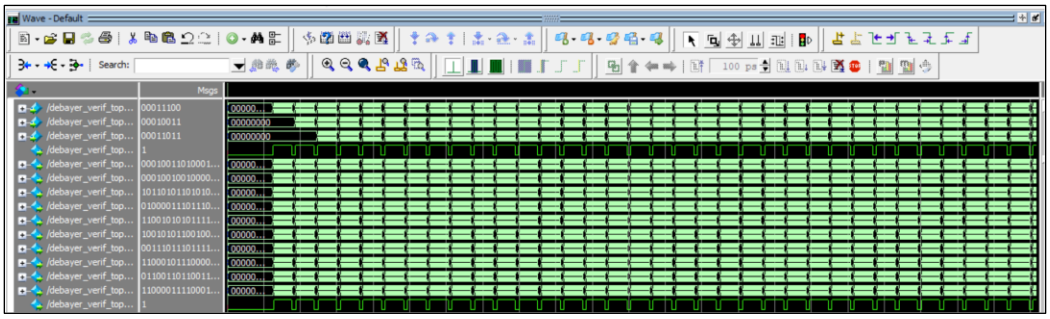

Figure 6.6. Simulation Waveform

Note: The testbench also includes data comparator/checker. The data check completed indicates correctness of the test.

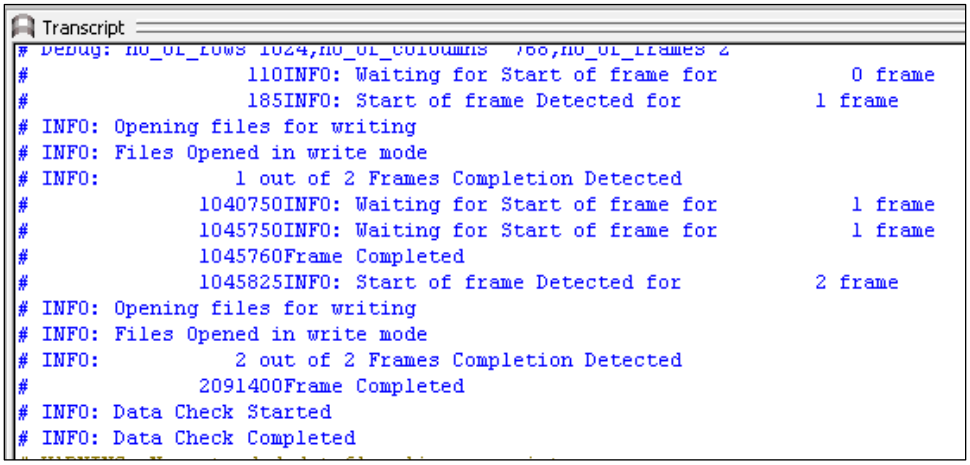

Figure 6.7. Data Check Passed

Appendix A. Resource Utilization

The following tables show the resource utilization of the Debayer IP for a few selected configurations. The results are based on Synplify Pro synthesis tool and the Lattice Radiant software version 2025.2.

Table A.1. Resource Utilization using the LAV-AT-E70ES1-3LFG1156I Device

Configuration	Clock RX_max (MHz)	Clock TX_max (MHz)	Registers	LUTs	DSP	sysMEM EBRs
8 bits per pixel 1 pixel per clock	250	250	3810	3986	3	8
8 bits per pixel 2 pixels per clock	250	250	5505	6079	6	11
8 bits per pixel 4 pixels per clock	250	250	8915	9826	12	16
16 bits per pixel 1 pixel per clock	250	250	6734	6536	6	11
16 bits per pixel 2 pixels per clock	219.73	243.78	10050	10499	12	16
16 bits per pixel 4 pixels per clock	249.50	250	16816	17910	24	27

Notes:

1. Fmax is generated using multiple iterations of Place and Route.
2. Fmax is generated when the FPGA design only contains Debayer IP core. These values may be reduced when user logic is added to the FPGA design.
3. 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*.

Table A.2. Resource Utilization using the LIFCL-40-9BG400I Device

Configuration	Clock RX_max (MHz)	Clock TX_max (MHz)	Registers	LUTs	DSP	sysMEM EBRs
8 bits per pixel 1 pixel per clock	193.65	190.11	3777	4070	3	11
8 bits per pixel 2 pixels per clock	200	186.36	5005	6134	6	16
8 bits per pixel 4 pixels per clock	178.35	181.65	8229	10074	12	27
16 bits per pixel 1 pixel per clock	177.37	171.62	6287	6537	6	21
16 bits per pixel 2 pixels per clock	174.13	159.41	9619	10427	12	32
16 bits per pixel 4 pixels Per clock[1]	163.19	154.70	16085	17688	24	53

Notes:

1. Fmax is generated using multiple iterations of Place and Route.
2. Fmax is generated when the FPGA design only contains Debayer IP core. These values may be reduced when user logic is added to the FPGA design.
3. 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*.

Table A.3. Resource Utilization using the LFD2NX-40-9BG256I Device

Configuration	Clock RX_max (MHz)	Clock TX_max (MHz)	Registers	LUTs	DSP	sysMEM EBRs
8 bits per pixel 1 pixel per clock	200	191.72	3777	4070	3	11
8 bits per pixel 2 pixels per clock	196.43	188.25	5005	6134	6	16
8 bits per pixel 4 pixels per clock	177.81	180.15	8229	10074	12	27
16 bits per pixel 1 pixel per clock	173.13	168.35	6287	6537	6	21
16 bits per pixel 2 pixels per clock	165.51	182.85	9619	10427	12	32
16 bits per pixel 4 pixels per clock	165.62	161.81	16085	17688	24	53

Notes:

1. Fmax is generated using multiple iterations of Place and Route.
2. Fmax is generated when the FPGA design only contains Debayer IP core. These values may be reduced when user logic is added to the FPGA design.
3. 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*.

Table A.4. Resource Utilization using the LFCPNX-100-8BBG484I Device

Configuration	Clock RX_max (MHz)	Clock TX_max (MHz)	Registers	LUTs	DSP	sysMEM EBRs
8 bits per pixel 1 pixel per clock	177.37	182.28	3763	4057	3	11
8 bits per pixel 2 pixels per clock	183.22	175.01	4988	6131	6	16
8 bits per pixel 4 pixels per clock	157.01	160.28	8214	10045	12	27
16 bits per pixel 1 pixel per clock	161.55	157.85	6271	6537	6	21
16 bits per pixel 2 pixels per clock	155.14	143.12	9600	10423	12	32
16 bits per pixel 4 pixels per clock[1]	147.78	149.25	16077	17652	24	53

Notes:

1. Fmax is generated using multiple iterations of Place and Route.
2. Fmax is generated when the FPGA design only contains Debayer IP core. These values may be reduced when user logic is added to the FPGA design.
3. 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

- Debayer IP Release Notes (FPGA-RN-02054)
- Lattice Radiant Timing Constraints Methodology (FPGA-AN-02059)
- CrossLink-NX web page
- Certus-NX web page
- Certus-N2 web page
- CertusPro-NX web page
- Avant-E web page
- Avant-G web page
- Avant-X web page
- Lattice Radiant Software web page
- Lattice Insights for Lattice Semiconductor training courses and learning plans

Technical Support Assistance

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

For frequently asked questions, refer to the Lattice Answer Database at https://www.latticesemi.com/Support/AnswerDatabase.

Revision History

Note: In some instances, the IP may be updated without changes to the user guide. The user guide may reflect an earlier IP version but remains fully compatible with the later IP version. Refer to the IP Release Notes for the latest updates.

Revision 1.3, IP v1.4.0, December 2025

Section	Change Summary
All	• Added a note on IP version in Quick Facts and *Revision History* sections. • Performed minor formatting and editorial edits.
Acronyms in This Document	Updated list of acronyms.

Introduction	• Updated the description in the Introduction section.
	• Updated Table 1.1. Quick Facts as follows:
	  • Added Certus-N2 devices.
	  • Added IP version.
	  • Removed earlier IP versions.

Attributes Summary	• Updated description for the *Blanking buffer depth* attributes in Table 4.1. Attributes Table. • Updated Figure 4.1. Configuration User Interface for Debayer IP.

IP Generation, Simulation, and Validation	• Added a note on IP version in GUI in the IP Generation, Simulation, and Validation section.
	• Updated the Licensing the IP section.
	• Updated the following figures:
	  • Figure 6.1. Module/IP Block Wizard
	  • Figure 6.2. Check Generated Result

Resource Utilization	Updated resource utilization for the latest software version.
References	Updated references.

Revision 1.2, IP v1.3.0, July 2025

Section	Change Summary
Disclaimers	Updated disclaimers.
Inclusive Language	Added inclusive language boilerplate.
Acronyms in This Document	Added definition for CFA.

Introduction	• Updated the description in the Introduction section.
	• Updated Table 1.1. Quick Facts as follows:
	  • Renamed *Supported FPGA Family* to *Supported Devices*.
	  • Removed the *Targeted Devices* row.
	  • Added IP version.
	  • Added IP changes.

IP Generation, Simulation, and Validation	• Changed *black box* to *closed-box* in the Generation and Synthesis section.
	• Updated the following figures:
	  • Figure 6.1. Module/IP Block Wizard
	  • Figure 6.2. Check Generating Result
	  • Figure 6.3. Simulation Wizard
	  • Figure 6.4. Adding and Reordering Source
	  • Figure 6.5. Run Time Setting to 0 ns

Resource Utilization	Updated resource utilization for the latest software version.
Reference	Updated references.

Revision 1.1, December 2022

Section	Change Summary
Acronyms in This Document	Updated the definition of Acronym BPP from *Bits Per Pixel* to *Bits Per Pixel Color Component*.
Introduction	Updated Table 1.1. Quick Facts for the following: • Added Lattice Avant to Supported FPGA Family. • Added LAV-AT-500E to Targeted Devices. • Updated IP version from IP v1.0.0 – Lattice Radiant™ software 3.2 or later to IP v1.1.0 – Lattice Radiant™ software 2022.1 or later.
Functional Description	• Updated bullet information from The input rx_tlast_i indicates the end of a line. This must be asserted during the last pixel of each line. to The input rx_tlast_i indicates end of a Total line. This must be asserted during the last pixel of horizontal blanking. in Section 2.3. • Updated bullet information from *The output tx_tlast_o indicates end of a line. This is asserted during the last pixel of each line.* to *The output tx_tlast_o indicates end of a total line. This is asserted during the last pixel of total line.* in Section 2.5. • Updated Figure 2.2. AXI-Stream Receiver Signals for PPC=1 and BPP=8 (Bayer Pattern), Figure 2.3. AXI-Stream Receiver Signals for PPC=4 and BPP=8 (Bayer Pattern), Figure 2.4. AXI-Stream Receiver Signals for PPC=1 and BPP=16 (Bayer Pattern), Figure 2.12. AXI-Stream Transmitter with PPC=1 and BPP =8, and Figure 2.13. AXI-Stream Transmitter with PPC=1 and BPP = 16 for RGB Pattern.
Signal Description	Updated Table 3.1. Width Parameter Description and Table 3.2. Debayer IP Signal Description.
Attribute Summary	Updated Table 4.1. Attributes Table and Figure 4.1. Configuration User Interface for Debayer IP.
Register Description	Updated Table 5.1. Summary of Configuration and Status Registers.
IP Generation, Simulation, and Validation	• Updated Figure 6.1. Module/IP Block Wizard and Figure 6.2. Check Generating Result. • Deleted Figure 6.4. Choose Appropriate Project Folder and Figure 6.5. Simulation Wizard. • Deleted step *Click Browse and choose your project folder. This is the folder with having user specified name. The user IP folder must be selected in this step for the simulation to run correctly*.
Appendix A. Resource Utilization	Updated Table A. 1. Resource Utilization using LAV-AT-500E-2LFG676I Avant, Table A. 2. Resource Utilization using LIFCL, and Table A. 3. Resource Utilization using Certus NX and added Table A. 4. Resource Utilization using LFCPNX-100-8BBG484I CertusPro-NX.
Reference	Added web page for Lattice Avant.

Revision 1.0, September 2022

Section	Change Summary
All	Initial release.

www.latticesemi.com