Color Correction Matrix (CCM) IP

IP Version: v1.3.1

User Guide

FPGA-IPUG-02214-1.3

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
CCM	Color Correction Matrix
CSR	Configuration and Status Register
GUI	Graphical User Interface
PPC	Pixels Per Clock
RGB	Red Green Blue
RX	Receiver
TX	Transmitter

1. Introduction

The measured RGB values of image sensors are different from the true RGB values of the image. This difference is mostly attributable to the characteristics of the optical filter overlay in the sensor. To obtain the correct colors, the pixels need to be mapped from sensor RGB color space to standard RGB color space. This linear mapping of the color components is achieved using a 3x3 matrix, called color correction matrix (CCM). This mapping is also referred to as RGB blending. Perfect mapping sometimes requires a translation after linear mapping, making the color correction a matrix multiplication followed by vector addition.

Figure 1.1 shows a depiction of linear color mapping using a CCM.

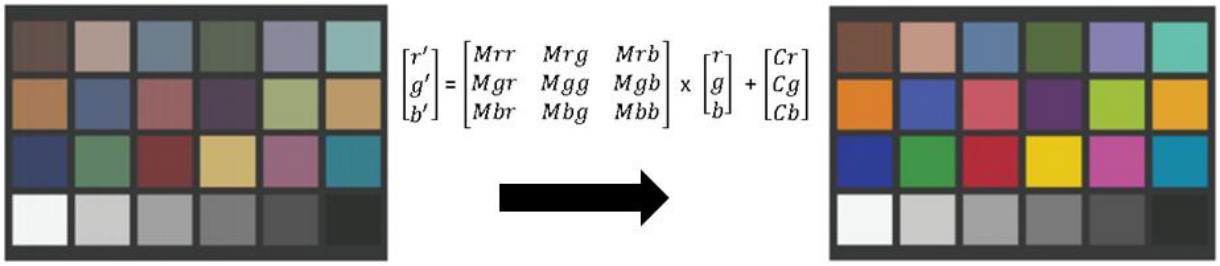

Figure 1.1 Depiction of Linear Mapping using CCM

CCM is a pixel-level operation not requiring any line buffers. The operation is done in the RGB domain. To determine the optimal color correction matrix, it is desirable to have a hardware-based calibration methodology. In this method, the sensor will be made to capture a standard color test pattern and the mean square error for a given CCM is evaluated. It is possible to determine the correct coefficients by solving the overdetermined set of equations using a least squares method in software.

1.1. Quick Facts

Table 1.1 presents a summary of the CCM IP.

Table 1.1. Quick Facts

IP Requirements	Supported Devices	CertusPro™-NX, Certus™-NX (LFD2NX-17, LFD2NX-40, LFD2NX-25, and LFD2NX-28), Certus™-N2, CrossLink™-NX, and Lattice Avant™.
	IP Changes[1]	For a list of changes to the IP, refer to the Color Correction Matrix (CCM) IP Release Notes (FPGA-RN-02030).
Resource Utilization	Resources	See Appendix A. Resource Utilization.
Design Tool Support	Lattice Implementation	IP v1.3.1 – 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 CCM 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 Color Correction Matrix

Color correction is a straightforward operation that computes the new pixels as a linear mapping of the original pixels. The mapping matrix, called CCM is a 3x3 matrix. In some use cases, there will be a constant added to the mapping equation. A generic color correction is given by Equation (1) below.

$$P\tilde{} = M \cdot P + C \quad (1)$$

Where,

P is the original pixel $[R\ G\ B]^T$

P~ is the corrected pixel, $[R\tilde{}\ G\tilde{}\ B\tilde{}]^T$

M is the CCM, whose coefficients are shown in Equation (2).

$$M = \begin{bmatrix} Mrr & Mrg & Mrb \\ Mgr & Mgg & Mgb \\ Mbr & Mbg & Mbb \end{bmatrix} \quad (2)$$

C is a constant translation vector

$$C = [Cr \quad Cg \quad Cb]^T \quad (3)$$

As shown above, the implementation is a simple matrix multiplication followed by vector addition. The vector addition process is not always used and hence that is provided as a configurable option. This is configurable via GUI as well as through configuration and status registers (CSR).

2.2. Block Diagram

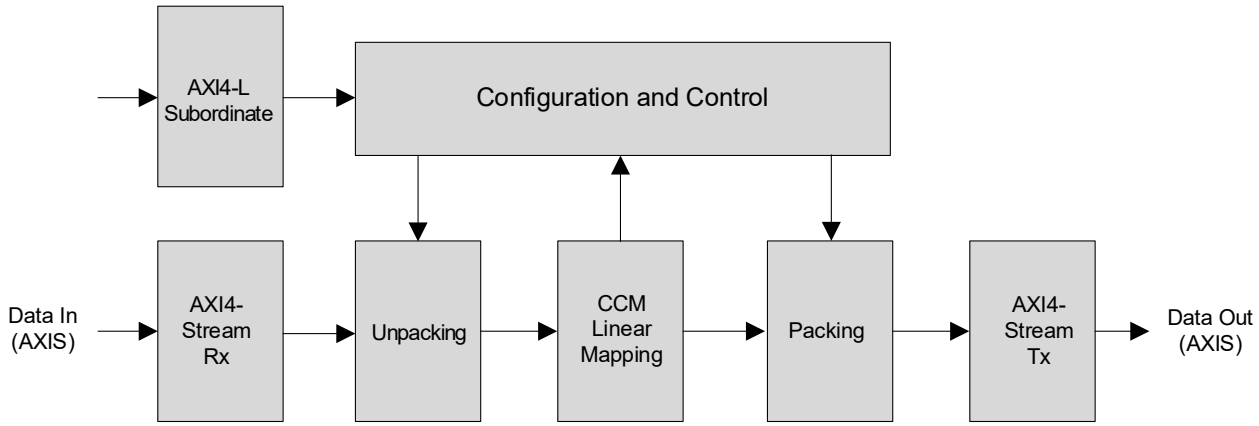

Figure 2.1. CCM IP Architectural Diagram

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 CCM hardware architecture:

- AXI-Stream is used for video data streaming in and out.
- Unpacking of AXI-Stream data to pixel data based on the parameters.
- The input/output data includes pixel data as well as video sync signals. The pixel data is through TDATA bus and the video sync signals are passed through the user-defined TUSER bus as sideband signals.
- Packing is the process of lining up the pixel values and timing information into the TDATA and TUSER busses of the AXI-Stream transmitter.
- AXI-Lite is used for the configuration path to write or read configuration and status registers (CSR).

2.3. AXI-Stream Receiver

The AXI-Stream Rx 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 the number of 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, it is to be noted 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 the start of the 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 the end of the frame. This signal must be high during the last active pixel of a frame.
- The input rx_tlast_i indicates the end of a total line. This signal must be asserted during the last pixel of the entire line.
- 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*3).

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.

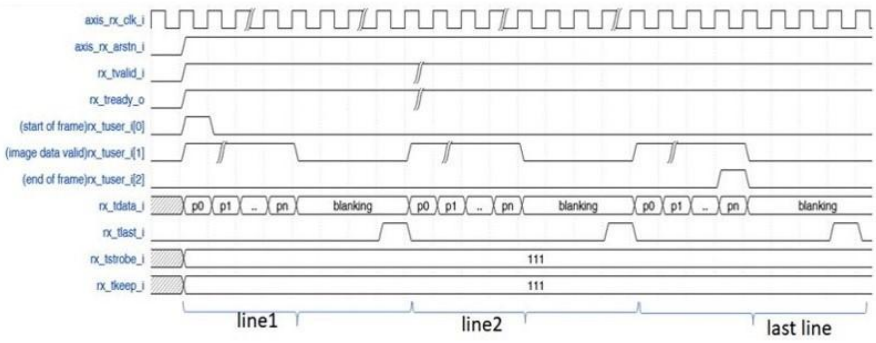

Figure 2.2. AXI-Stream Receiver Signals for PPC =1 and BPP=8 (RGB Format)

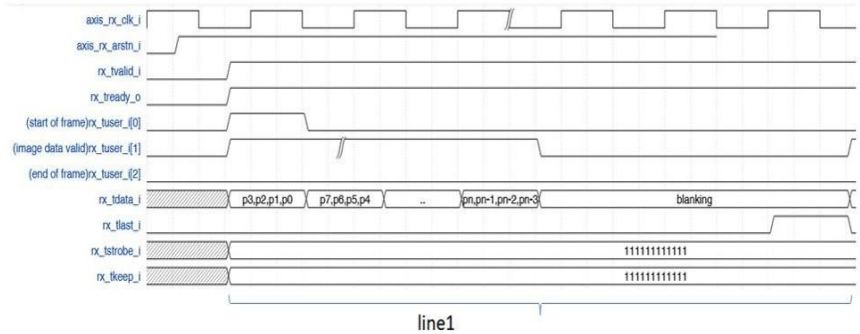

Figure 2.3. AXI-Stream Receiver Signals for PPC=4 and BPP=8 (RGB Format)

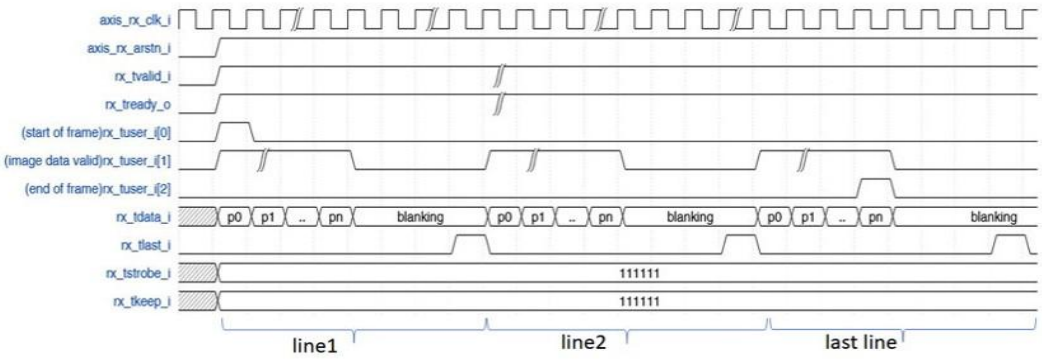

Figure 2.4. AXI-Stream Receiver Signals for PPC =1 and BPP=16 (RGB Format)

The organization of the color component data in the TDATA bus is described here. In general, the red component is mapped to the lower bits, green to the middle bits, and blue to the upper bits, each component mapped from MSB to LSB.

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*3) bits is mapped to the lower part of the bus, with the remaining upper bits assigned to zeros.

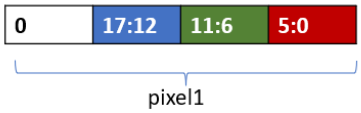

Figure 2.5. Bit Allocation for BPP=6 and PPC=1 for RGB Input

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

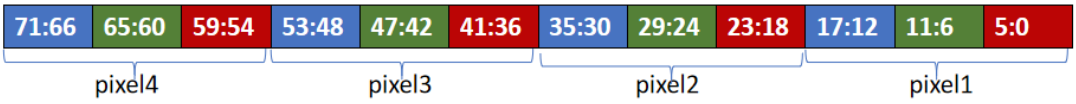

Figure 2.7. Bit Allocation for BPP=6 and PPC=4 for RGB Input

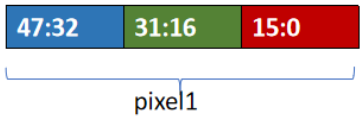

Figure 2.8. Bit Allocation for BPP=16 and PPC=1 for RGB Input

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

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

2.4. Data Unpacker

The Data Unpacker module unpacks the data from the AXI-Stream receiver into individual RGB components. The data unpacking depends on the video type as well as on the Bayer pattern, for Bayer video types. This module provides output in RGB format with data valid signals defining the valid data. The timing diagrams for the unpacker for different configurations are shown in Figure 2.11. As seen in the figure, the data from rx_tdata_i is unpacked to red_component, green_component, and blue_component along with a valid signal.

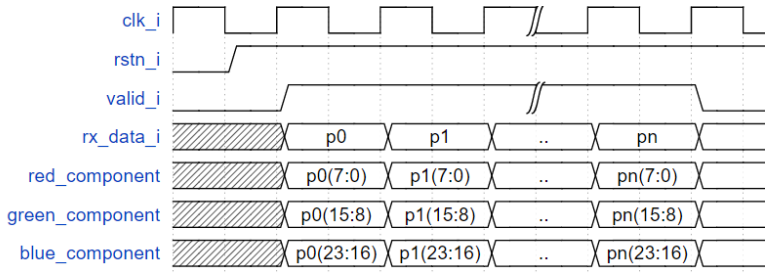

Figure 2.11. Unpacking of Pixels into Different Components for RGB Pattern

2.5. Data Packer

The Data Packer module performs the reverse of the unpacking operation. Here, the pixel data is packed into multiple bytes based on the currently set parameters. The timing diagrams for the packer for different configurations are shown in Figure 2.12.

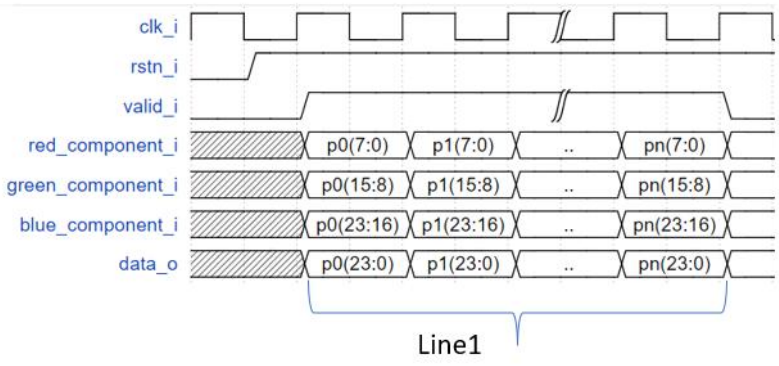

Figure 2.12. Packing of Color Components in RGB Pattern for BPP=8 and PPC=1

2.6. AXI-Stream Transmitter

The AXI Stream Transmitter module maps the video data back to bytes for transmission through AXI-Stream to the next stage pipeline.

The transmitter sends out the data in RGB format. Here, 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. AXI-Stream transmitter sends out data only when tx_tready_i is high. If tx_tready_i goes low, 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 the 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 the start of the 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 the end of the frame. This signal will be high during the last active pixel of a frame.
- The output tx_tlast_o indicates the end of a total line. This is asserted during the last pixel of the entire 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 for AXI-Stream Transmitter is shown in Figure 2.13 and Figure 2.14.

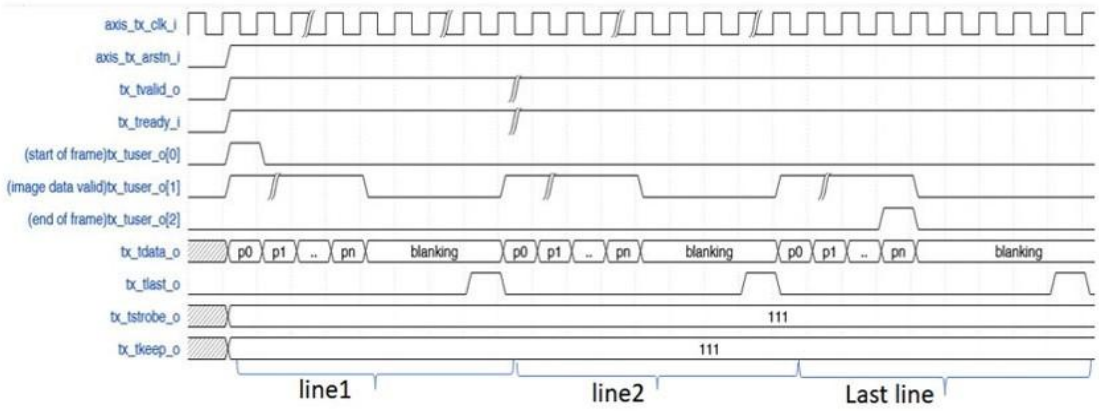

Figure 2.13. AXI-Stream Transmitter with PPC=1 and BPP =8

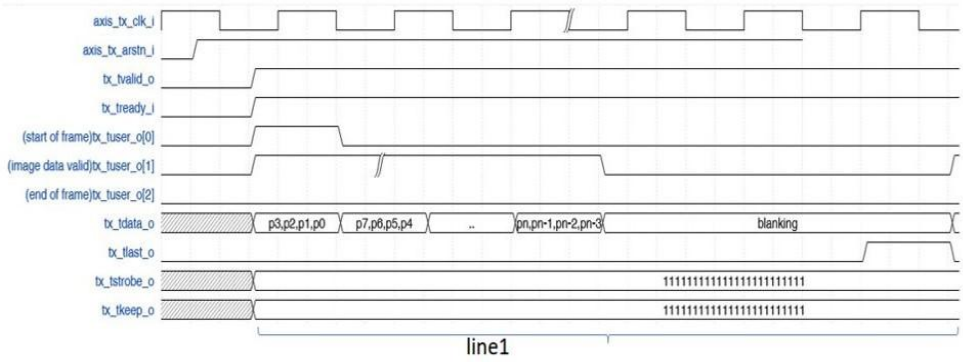

Figure 2.14. AXI-Stream Transmitter with PPC=4 and BPP = 16 for RGB Pattern

2.7. Configuration and Control

The Configuration and Control 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 CCM
- Configuration update
- Translation vector values
- Linear map values
- Query pixel location
- Query pixel value

For details on the configuration register, refer to Table 3.4.

2.8. AXI-Lite Subordinate

AXI-Lite consists of five channels. Table 2.1 shows the channels and 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

2.8.1. Write Operation

The write operation sequence is described below.

1. The manager asserts aw_valid_i along with the address.
2. The subordinate asserts the aw_ready_i and captures the address.
3. The manager asserts w_valid_i along with the write data.
4. The subordinate asserts w_ready_i and captures the data.
5. The subordinate sends an OKAY response and asserts b_valid_o.
6. The manager asserts b_ready_i, completing the transaction.

2.8.2. Read Operation

The read operation sequence is described below.

1. The manager asserts ar_valid_i signal along with the address.
2. The subordinate captures the address by asserting the ar_ready_i.
3. The subordinate sends the r_data_o along with r_valid_o and r_response_o.
4. The manager asserts r_ready_i, completing the transaction.

3. Signal Description

The widths used for the signal busses in the interface table are defined in Table 3.1. The input/output interface signals for CCM IP are indicated in Table 3.2.

Table 3.1. Description of Width Parameters

Width Parameter Name	Description
AXI_STREAM_DATA_WIDTH	The width of the AXI-Stream bus is equal to the next higher multiple of 8 of the quantity (PPC*BPP*3).
AXI_LITE_ADDR_WIDTH	5
AXI_LITE_DATA_WIDTH	32

Table 3.2. CCM 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_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 is 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 the 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_DATA_WIDTH-1:0]/8	TSTROBE is the byte qualifier that indicates whether the content of the associated byte is processed as a data byte or a position byte.
rx_tkeep_i	I	[AXI_STREAM_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 the Transmitter 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_DATA_WIDTH-1:0]	TDATA is the primary payload that is used to provide the data that is passing across the interface.

Port Name	I/O	Size	Description
tx_tuser_o	O	3	Sideband information is 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
tx_tlast_o	O	1	TLAST indicates the end of the line for each horizontal line.
tx_tstrobe_o	O	[AXI_STREAM_DATA_WIDTH-1:0]/8	TSTROBE is the byte qualifier that indicates whether the content of the associated byte is processed as a data byte or a position byte.
tx_tkeep_o	O	[AXI_STREAM_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 data access or 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_ready_o	O	1	Write ready. This signal indicates that the subordinate can accept the write data.
w_strb_i	I	[AXI_LITE_DATA_WIDTH-1:0]/8	Write strobes. This signal indicates which byte lanes hold valid data. There is one write strobe bit for every eight bits of the write data bus (Not used).
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 a valid read address.
ar_address_i	I	[AXI_LITE_ADDR_WIDTH-1:0]	Read address.
ar_ready_o	O	1	Read address ready. This signal indicates that the subordinate is ready to accept an 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 data access or instruction access. (Not used).
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.

3.1. Attributes Summary

The attributes set through the user interface are described in Table 3.3.

Table 3.3. Attributes Table

Attribute	Selectable Values	Default	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 PPC2 and PPC4, when the resolution is not divisible by PPC.
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 is skipped. If tx_tready_i is asserted before the Tx buffer depth cycles, then the data flow is not interrupted
CSR Values			
CCM Enable	Enabled, Disabled	Enabled	1 – enable (Color Correction matrix operation) 0 – disable (the output pixel data is the same as the input data)
CCM Values			
Translation	Enabled, Disabled	Disabled	Whether the translation is required
Translation_R	[–255,255]	0	First (red) row of the translation vector
Translation_G	[–255,255]	0	First (green) row of the translation vector
Translation_B	[–255,255]	0	First (blue) row of the translation vector
MRR	–0.99 to 0.99	0.99	First (red) row, first (red) column of the CCM
MRG	–0.99 to 0.99	0	First (red) row, first (green) column of the CCM
MRB	–0.99 to 0.99	0	First (red) row, first (blue) column of the CCM
MGR	–0.99 to 0.99	0	First (green) row, first (red) column of the CCM
MGG	–0.99 to 0.99	0.99	First (green) row, first (green) column of the CCM
MGB	–0.99 to 0.99	0	First (green) row, first (blue) column of the CCM
MBR	–0.99 to 0.99	0	First (blue) row, first (red) column of the CCM
MBG	–0.99 to 0.99	0	First (blue) row, first (green) column of the CCM
MBB	–0.99 to 0.99	0.99	First (blue) row, first (blue) column of the CCM

A sample user interface for the CCM IP is shown in Figure 3.1.

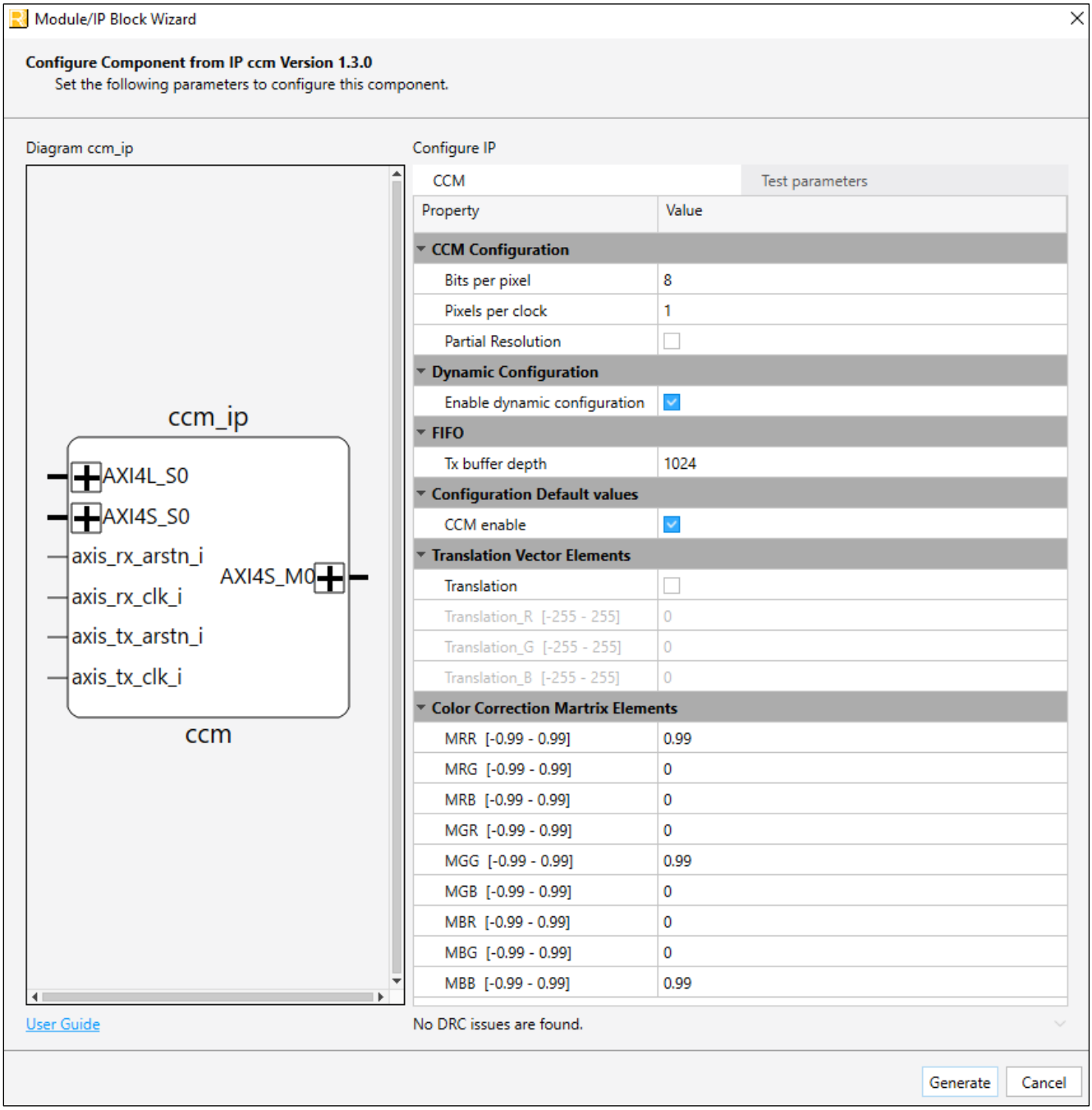

Figure 3.1. Lattice Radiant User Interface for CCM IP

3.2. Register Description

The registers used in the CSR module of the CCM IP are shown in Table 3.4. The CSR registers are accessible through the AXI-Lite interface.

Table 3.4. Summary of Configuration and Status Registers

Offset	Register Name	Access	Default Value	Range	Register Description		
					Bit	Field	Field Description
0x00	CCM_ENABLE	RW	0	0, 1	0	ccm_enable	0 – enable (Color Correction Matrix operation) 1 – disable (the output pixel data is the same as the input data)
					[31:1]	RSVD	Reserved bits
0x01	CONFIGURATION_UPDATE	RW	0	0, 1	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
0x02	TRANSLATION_R	RW	0	[-2^{BPP}-1,2^{BPP}-1]	[BITS_PER_PIXEL + 1 : 0]	Translation RED	First (red) row of the translation vector
0x03	TRANSLATION_G	RW	0	[-2^{BPP}-1,2^{BPP}-1]	[BITS_PER_PIXEL + 1 : 0]	Translation GREEN	First (green) row of the translation vector
0x04	TRANSLATION_B	RW	0	[-2^{BPP}-1,2^{BPP}-1]	[BITS_PER_PIXEL + 1 : 0]	Translation BLUE	First (blue) row of the translation vector
0x05	MRR	RW	0	[–32767,32767]	[15:0]	Matrix coefficient MRR	First (red) row, first (red) column of the CCM
0x06	MRG	RW	0	[–32767,32767]	[15:0]	Matrix coefficient MRG	First (red) row, second (green) column of the CCM
0x07	MRB	RW	0	[–32767,32767]	[15:0]	Matrix coefficient MRB	First (red) row, second (blue) column of the CCM
0x08	MGR	RW	0	[–32767,32767]	[15:0]	Matrix coefficient MGR	Second (green) row, first (red) column of the CCM
0x09	MGG	RW	0	[–32767,32767]	[15:0]	Matrix coefficient MGB	Second (green) row, first (green) column of the CCM
0xA	MGB	RW	0	[–32767,32767]	[15:0]	Matrix coefficient MGG	Second (green) row, first (blue) column of the CCM
0xB	MBR	RW	0	[–32767,32767]	[15:0]	Matrix coefficient MBR	Third (blue) row, first (red) column of the CCM
0xC	MBG	RW	0	[–32767,32767]	[15:0]	Matrix coefficient MBG	Third (blue) row, first (green) column of the CCM
0xD	MBB	RW	0	[–32767,32767]	[15:0]	Matrix coefficient MBB	Third (blue) row, first (blue) column of the CCM
0xE	LINE_QUERY_ADDR	RW	1	[1-4096]	[15:0]	Line number query	User provides Line number of input pixel they want to query
0xF	PIXEL_QUERY_ADDR	RW	1	[1-4096]	[15:0]	Pixel number query	User provides Pixel number of input pixel they want to query in the line specified via Line query
0x10	PIXEL_RED_ADDR	R	0	[0-65535]	[15:0]	Red Input	Input red value at query location

Offset	Register Name	Access	Default Value	Range	Register Description		
					Bit	Field	Field Description
0x11	PIXEL_GREEN_ADDR	R	0	[0-65535]	[15:0]	Green Input	Input green value at query location
0x12	PIXEL_BLUE_ADDR	R	0	[0-65535]	[15:0]	Blue Input	Input blue value at query location

Notes:

- For the nine matrix coefficients MRR to MBB, the user has to multiply the coefficient values by 32768 and use the integer part when configuring the CSR dynamically. For example, for an MRR=0.87, the user must give 28508 (0.87*32768=28508.16)
- The range for the matrix coefficients is from -32767 to 32767, corresponding to coefficient values of approximately -0.9999 to 0.9999.
- For TRANSLATION_R, TRANSLATION_G and TRANSLATION_B, the range is [-65535,65535] for BPP=16 and [-255,255] for BPP=8

4. IP Generation, Simulation, and Validation

This section provides information on how to generate the CCM 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.

4.1. Licensing the IP

The Color Correction Matrix IP is provided at no additional cost with the Lattice Radiant software.

4.2. 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 CCM IP in Lattice Radiant software is described below.

To generate the CCM IP:

1. Create a new Lattice Radiant software project or open an existing project.
2. In the IP Catalog tab, double-click on **Color Correction Matrix** under the **Audio_Video_and_Image_Processing** category. **The Module/IP Block Wizard** opens as shown in Figure 4.1.
3. Enter values in the **Component name** and the **Create in** fields.
4. Click **Next**.
5. Go to *RadiantIPLocal\ccm_test\testbench\tb_settings.v*. Update the file path parameter accordingly. Make sure the generated user interface matches the testbench parameters.

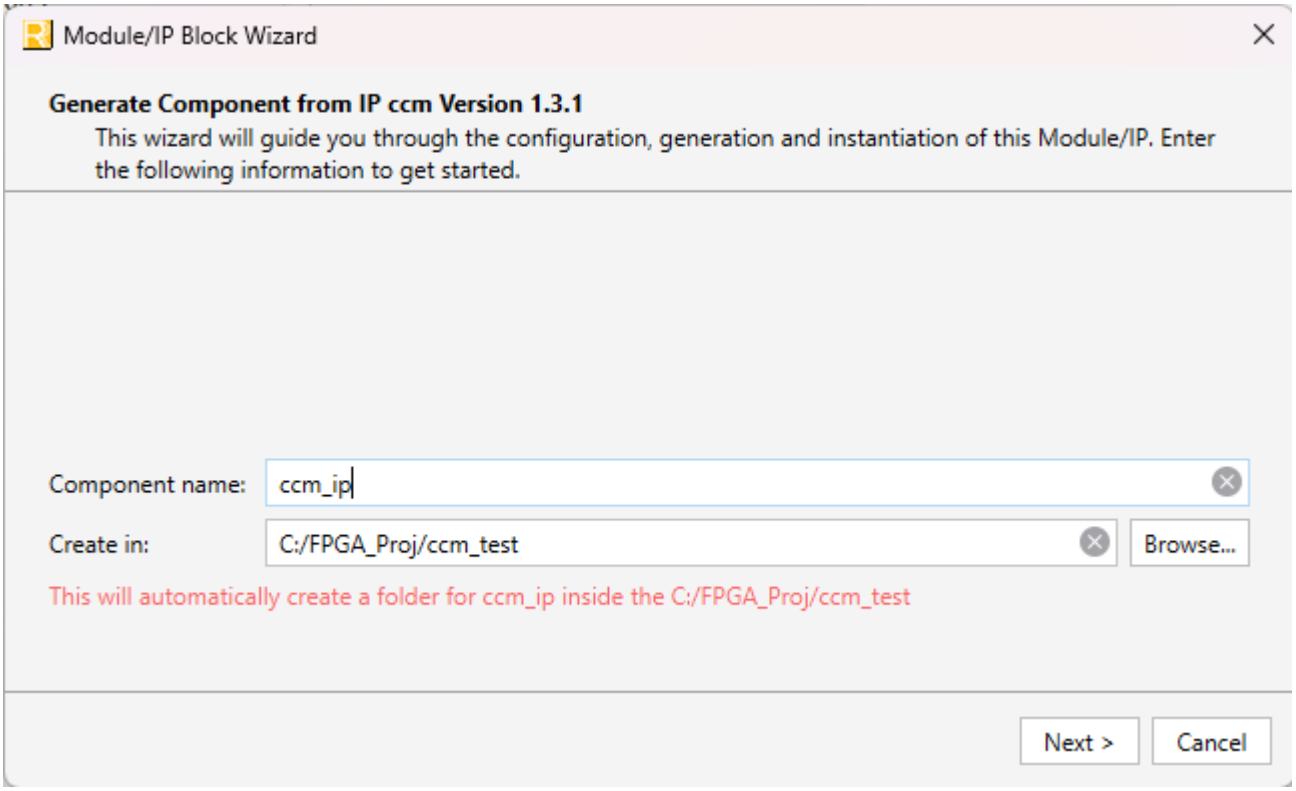

Figure 4.1. Module/IP Block Wizard

6. In the **Module/IP Block Wizard** page, customize the selected CCM IP. As a sample configuration, see Figure 4.2. For configuration options, see Table 3.3**.**

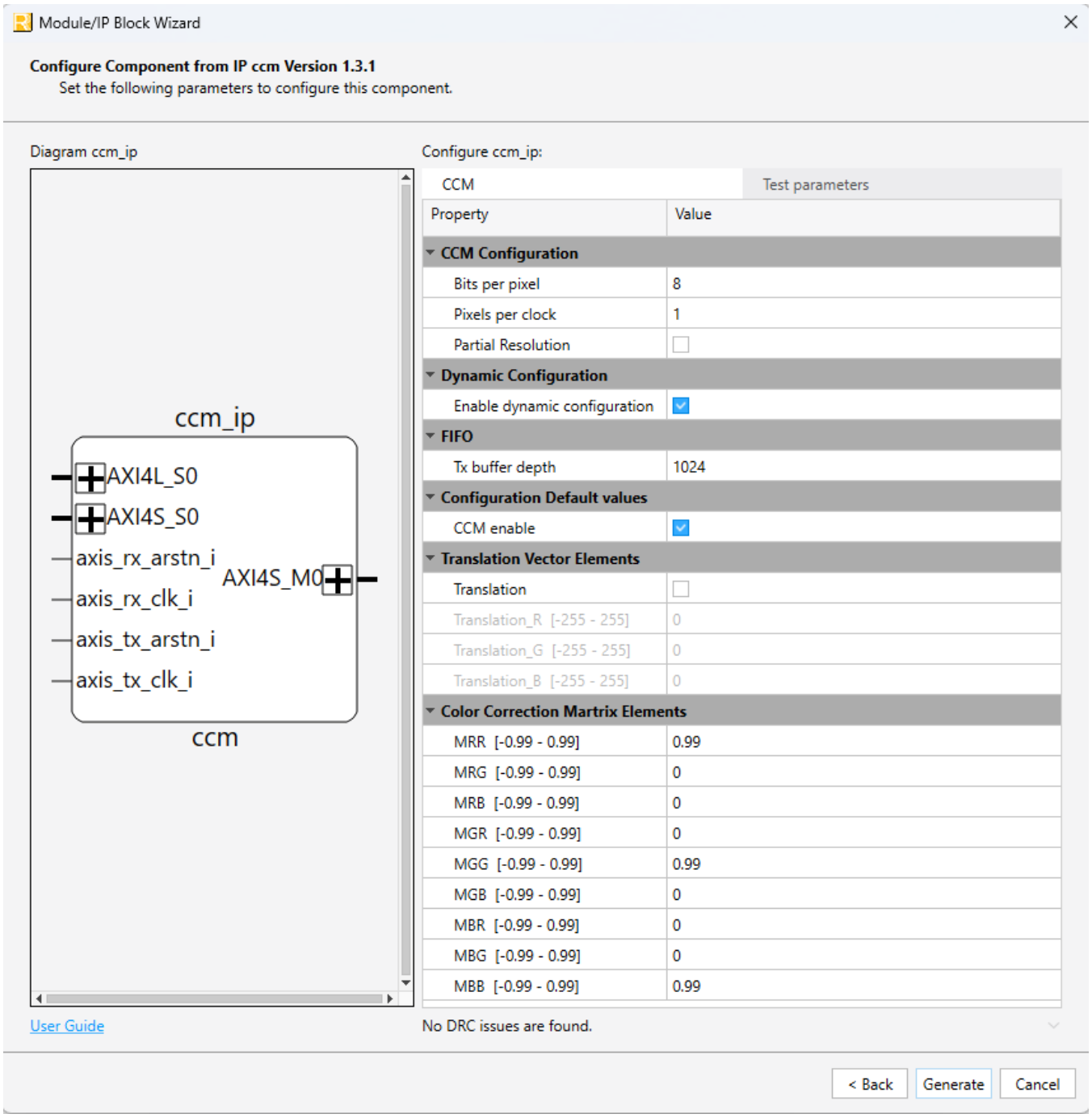

Figure 4.2. Configuration User Interface for CCM IP

7. Click **Generate**. The **Check Generated Result** page opens, showing design block messages and results as shown in Figure 4.3.

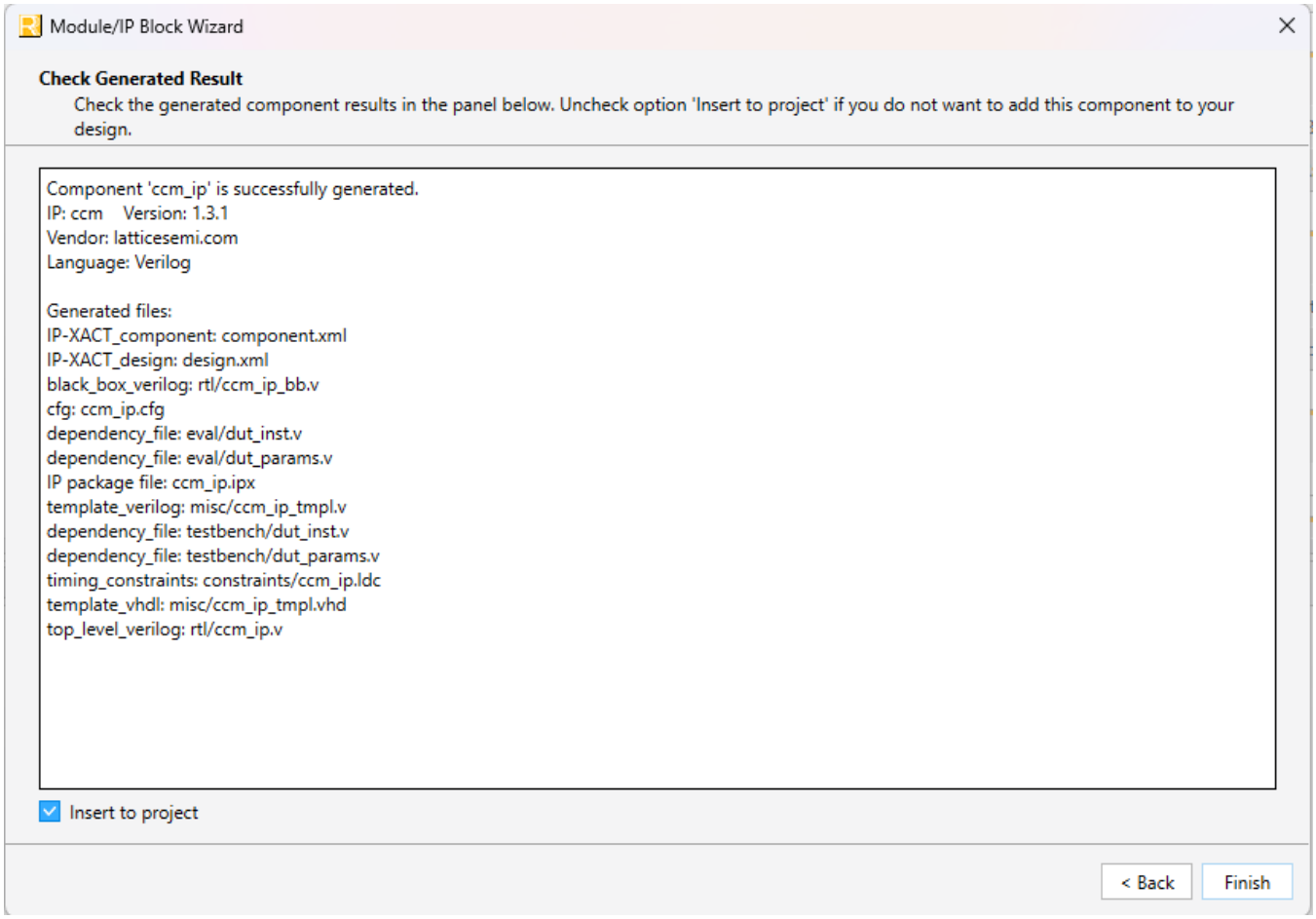

Figure 4.3. Check Generating Result

8. Click **Finish**. All the generated files are placed under the directory paths in the **Create in** and the **Component name** fields are shown in Figure 4.1.

The generated CCM 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. This top-level reference may also be used as the starting template for the top-level complete design. The generated files are listed in Table 4.1.

Table 4.1. Generated File List

Attribute	Description
<Component name>.ipx	Contains the information on the files associated with 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 closed-box.

4.3. Running Functional Simulation

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

1. Click the  button located on the toolbar to initiate the **Simulation Wizard** shown in Figure 4.4.

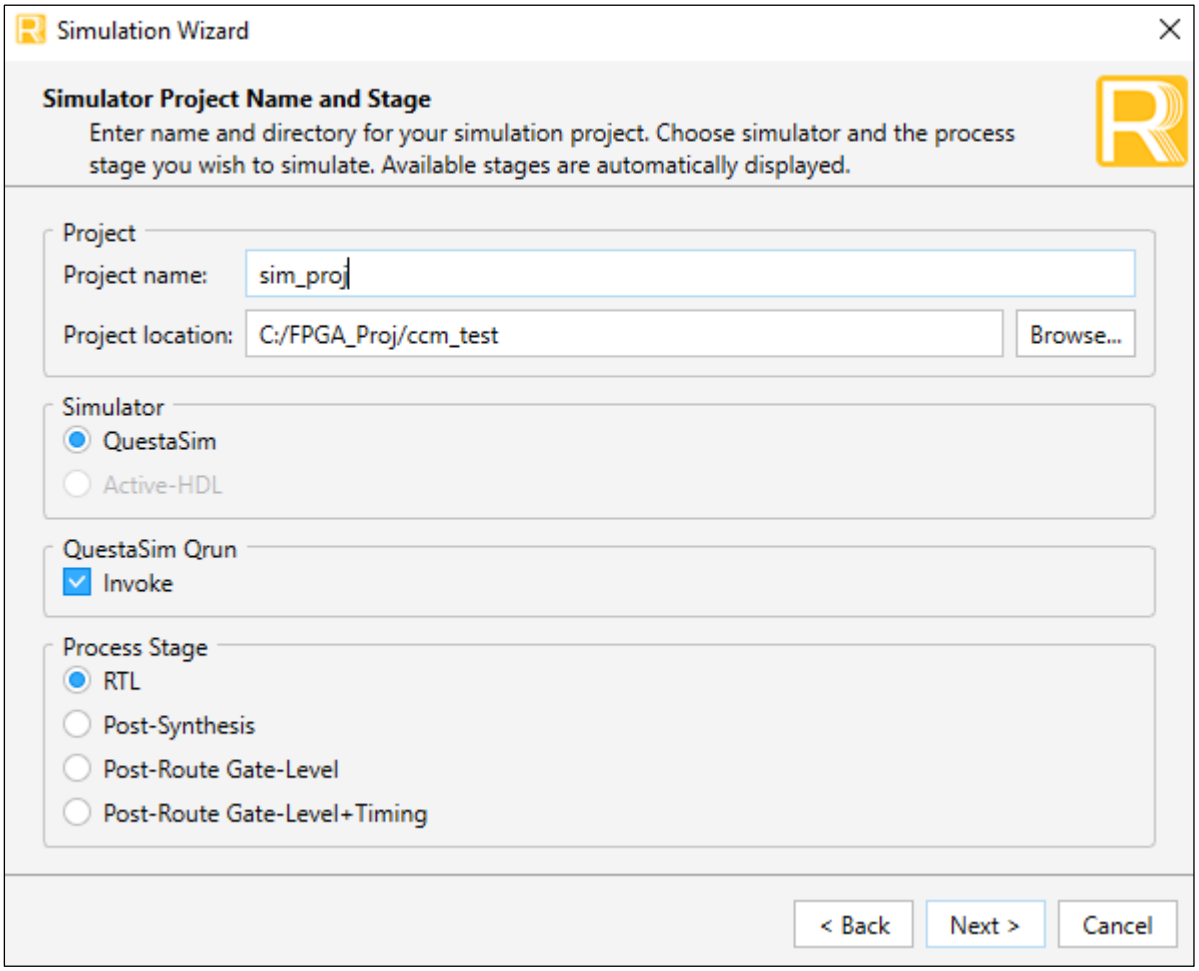

Figure 4.4. Simulation Wizard

2. Click **Next** to open the **Add and Reorder Source** page as shown in Figure 4.5.

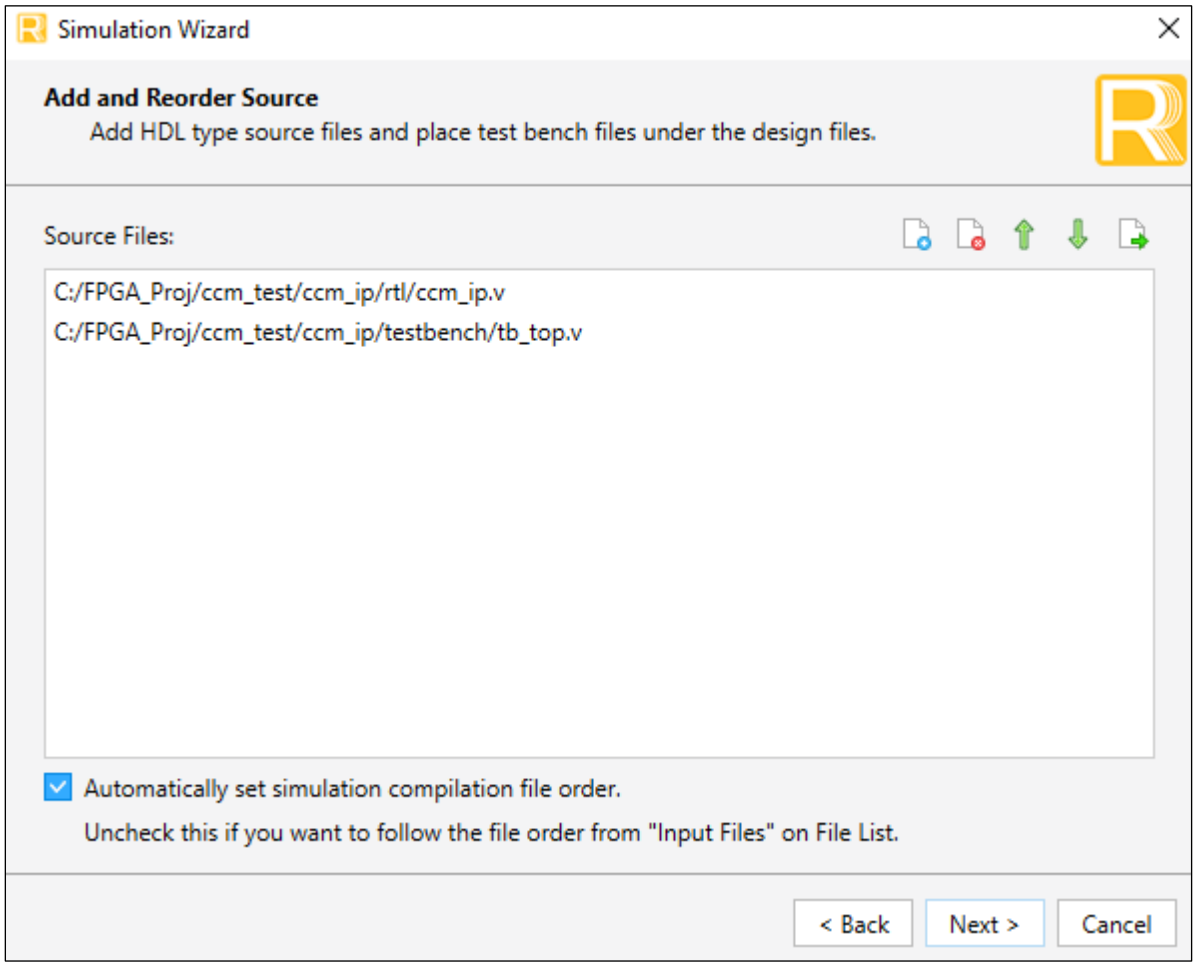

Figure 4.5. Adding and Reordering Source

3. Change the simulation time to **0** ns, which means run -all.

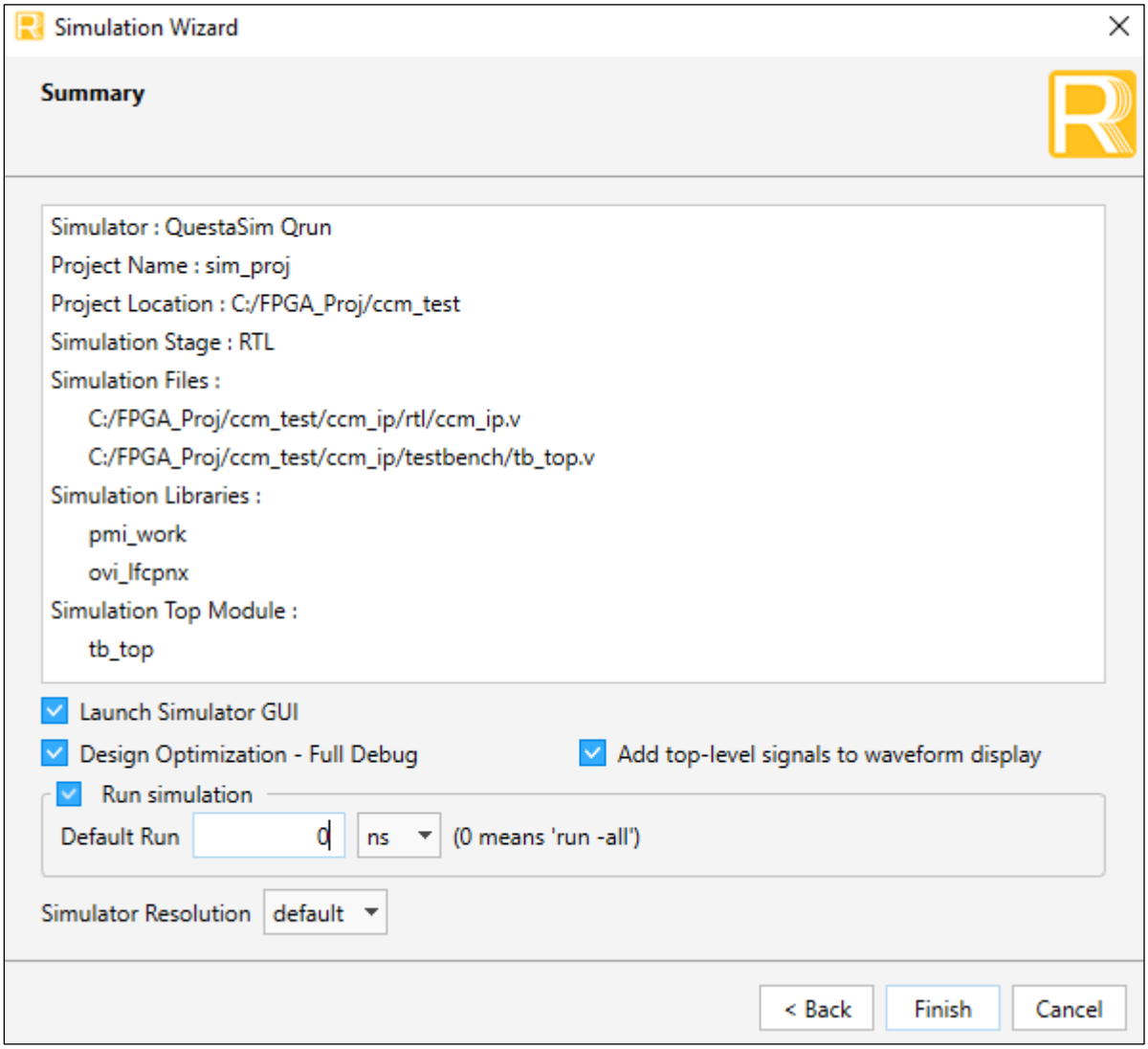

Figure 4.6. Run Simulation Value of 0 for Run All

4. Click **Next**. The **Summary** window opens.
5. Click **Finish** to run the simulation. The result of the simulation in the example is provided in Figure 4.7.

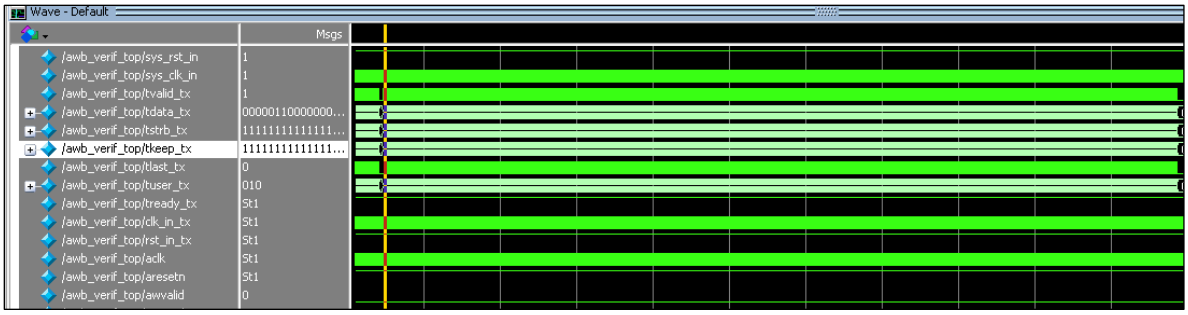

Figure 4.7. Sample Simulation Waveform

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

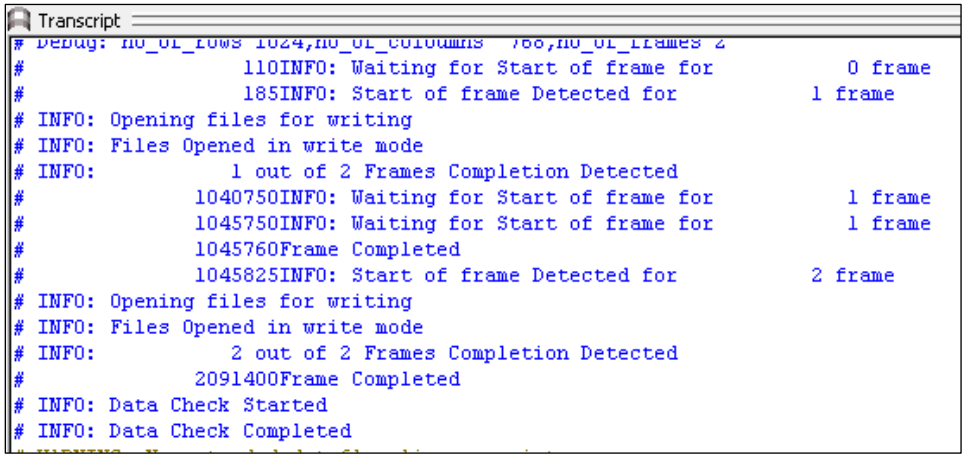

Figure 4.8. Data Check Passed

Appendix A. Resource Utilization

Table A.1, Table A.2, Table A.3, and Table A.4 show the resource utilization for a few select configurations for Nexus device with speed grades 7 and 9 and Avant device with speed grades 1 and 3. The results are based on the Synplify Pro synthesis tool and Lattice Radiant software version 2025.2.

Table A.1. CertusPro-NX Device (LFCPNX-100-9LFG672I) Resource Utilization

Configuration	Clock RX_max[1,2] (MHz)	Clock TX_max[1,2] (MHz)	Clock AXI_lite_max[1,2] (MHz)	Registers	LUTs[3]	DSP	sysMEM EBRs
8 Bits Per Pixel 1 Pixel Per Clock	168.55	173.82	200	1317	1411	9	4
8 Bits Per Pixel 2 Pixel Per Clock	173.88	170.24	200	1745	1869	18	6
8 Bits Per Pixel 4 Pixel Per Clock	171.76	157.75	200	2623	2792	36	12
16 Bits Per Pixel 1 Pixels Per Clock	170.91	168.61	200	1649	1660	9	6
16 Bits per pixel 2 Pixels per clock	161.08	161.42	200	2366	2383	18	12
16 Bits per pixel 4 Pixels per clock	174.80	159.08	200	3823	3805	36	22

Notes:

1. Fmax is generated using multiple iterations of Place and Route.
2. Fmax is generated when the FPGA design only contains CCM 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. CertusPro-NX Device (LFCPNX-100-7LFG672I) Resource Utilization

Configuration	Clock RX_max[1,2] (MHz)	Clock TX_max[1,2] (MHz)	Clock AXI_lite_max[1,2] (MHz)	Registers	LUTs[3]	DSP	sysMEM EBRs
8 Bits Per Pixel 1 Pixel Per Clock	137.57	138.72	200	1308	1391	9	4
8 Bits Per Pixel 2 Pixel Per Clock	132.63	128.04	200	1758	1867	18	6
8 Bits Per Pixel 4 Pixel Per Clock	138.72	131.93	200	2614	2786	36	12
16 Bits Per Pixel 1 Pixels Per Clock	140.51	133.40	200	1644	1652	9	6
16 Bits per pixel 2 Pixels per clock	128.97	126.53	200	2382	2383	18	12
16 Bits per pixel 4 Pixels per clock	135.72	124.72	200	3814	3809	36	22

Notes:

1. Fmax is generated using multiple iterations of Place and Route.
2. Fmax is generated when the FPGA design only contains CCM 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. Lattice Avant Device (LAV-AT-E70ES1-3LF1156I) Resource Utilization

Configuration	Clock RX_max[1,2] (MHz)	Clock TX_max[1,2] (MHz)	Clock AXI_lite_max[1,2] (MHz)	Registers	LUTs[3]	DSP	sysMEM EBRs
8 Bits Per Pixel 1 Pixel Per Clock	250	250	250	1096	1082	9	2
8 Bits Per Pixel 2 Pixel Per Clock	250	250	250	1341	1240	18	4
8 Bits Per Pixel 4 Pixel Per Clock	250	250	250	1762	1606	36	6
16 Bits Per Pixel 1 Pixels Per Clock	250	250	250	1358	1228	9	4
16 Bits per pixel 2 Pixels per clock	250	250	250	1821	1576	18	6
16 Bits per pixel 4 Pixels per clock	250	250	250	2674	2224	36	12

Notes:

1. Fmax is generated using multiple iterations of Place and Route.
2. Fmax is generated when the FPGA design only contains CCM 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. Lattice Avant Device (LAV-AT-E70ES1-1LF1156I) Resource Utilization

Configuration	Clock RX_max[1,2] (MHz)	Clock TX_max[1,2] (MHz)	Clock AXI_lite_max[1,2] (MHz)	Registers	LUTs[3]	DSP	sysMEM EBRs
8 Bits Per Pixel 1 Pixel Per Clock	249.56	203.87	250	1096	1082	9	2
8 Bits Per Pixel 2 Pixel Per Clock	250	210.04	250	1341	1240	18	4
8 Bits Per Pixel 4 Pixel Per Clock	233.26	210.84	250	1762	1606	36	6
16 Bits Per Pixel 1 Pixels Per Clock	250	205.38	250	1358	1228	9	4
16 Bits per pixel 2 Pixels per clock	243.84	208.986	250	1821	1576	18	6
16 Bits per pixel 4 Pixels per clock	250	215.66	250	2674	2224	36	12

Notes:

1. Fmax is generated using multiple iterations of Place and Route.
2. Fmax is generated when the FPGA design only contains CCM 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

- Color Correction Matrix (CCM) IP Release Notes (FPGA-RN-02030)
- Avant-E web page
- Avant-G web page
- Avant-X web page
- Certus-N2 web page
- Certus-NX web page
- CertusPro-NX web page
- CrossLink-NX web page
- Lattice Radiant Software web page
- Lattice Solutions IP Cores web page
- Lattice Insights web page 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 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 v1.3.1, December 2025

Section	Change Summary
Acronyms in This Document	Removed *OPN*.

Introduction	• In Table 1.1. Quick Facts:
	  • Added *Certus™-N2* as Supported Devices.
	  • Updated IP version from *IP Core v1.3.0 to IP Core v1.3.1.*
	  • Updated software version from *Lattice Radiant™ software 2025.1* to *Lattice Radiant™ software 2025.2*.
	• Added note, *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.*

IP Generation, Simulation, and Validation	• Updated Licensing the IP section, *The Color Correction Matrix IP is provided at no additional cost with the Lattice Radiant Software.* • Updated Figure 4.1. Module/IP Block Wizard – Figure 4.4. Simulation Wizard.
Ordering Part Number	Removed this section.
Appendix A. Resource Utilization	In Table A.1. CertusPro-NX Device (LFCPNX-100-9LFG672I) Resource Utilization – Table A.4. Lattice Avant Device (LAV-AT-E70ES1-1LF1156I) Resource Utilization: • Updated all table captions. • Updated values of *Clock RX_max*, *Clock TX_max*, *Registers*, and *LUTs*. • Added *Clock AXI_lite_max*. • Added notes.
References	Updated this section.
Revision History	Added note.

Revision 1.2, IP v1.3.0, July 2025

Section	Change Summary
All	• Added the IP version information on the cover page. • Made editorial fixes.
Disclaimers	Updated boilerplate.
Inclusive Language	Added boilerplate.
Acronyms in This Document	Added *OPN*.
Introduction	In Table 1.1. Quick Facts: • Updated *Supported FPGA Family* to *Supported Devices* and updated the devices. • Added *IP Changes*. • Removed *Targeted Devices*. • Updated *Lattice Implementation*.
Signal Description	Updated Figure 3.1. Lattice Radiant User Interface for CCM IP.
IP Generation, Simulation, and Validation	• Updated Figure 4.1. Module/IP Block Wizard – Figure 4.6. Run Simulation Value of 0 for Run All. • Updated *black box* to *closed-box* in the Generation and Synthesis section.
Ordering Part Number	Added this section.
Appendix A. Resource Utilization	• Updated resource utilizations for the Lattice Radiant software version 2025.1. • Updated all table captions in this section.
References	Updated this section.

Revision 1.1, December 2022

Section	Change Summary
Acronyms in This Document	Updated *BPP*. Added *RX* and *TX*.
Introduction	Updated Table 1.1. Quick Facts. • Added Lattice Avant in Supported FPGA Families. • Added LAV-AT-500E in Targeted Devices. • Update versions in Lattice Implementation.
Functional Description	Updated the AXI-Stream Receiver and the AXI-Stream Transmitter sections. • Modified the description of *input rx_tlast_i* and *output tx_tlast_o*. • Updated figures and/or figure captions. Updated parameters in Configuration and Control.
Signal Description	• In Table 3.1. Description of Width Parameters, updated AXI_LITE_DATA_WIDTH description. • In Table 3.2. CCM IP Signal Description, updated ports under Clock and Reset. Changed Rx to RX and Tx to TX. • In Table 3.3. Attributes Table, updated AXI-Lite, FIFO values. Added CSR Values. Changed default value of *Translation* under CCM Values. • General update to Table 3.4. Summary of Configuration and Status Registers.
IP Generation, Simulation, and Validation	• Updated figures in the Generation and Synthesis section. • Updated the procedure in Running Functional Simulation section.
Appendix A. Resource Utilization	General update to this section.
Technical Support Assistance	Added reference to the Lattice Answer Database on the Lattice website.

Revision 1.0, September 2022

Section	Change Summary
All	Initial release.

www.latticesemi.com