MPPHY Module

IP Version: 3.0.0

User Guide

FPGA-IPUG-02233-1.7

June 2026

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
AFE	Analog Front End
ANLT	Auto-Negotiation Link Training
BER	Bit Error Rate
CDR	Clock/Data Recovery
CMU	Clock Multiplier Unit
CPRI	Common Public Radio Interface
CTC	Clock Tolerance Compensation
CTLE	Continuous-Time Linear Equalizer
DFE	Decision Feedback Equalization
DFF	D Flip Flop
DMA	Direct Memory Access
DP	DisplayPort
DR	Dynamic Reconfiguration
eDP	Embedded DisplayPort
FC	Fire Code
FEC	Forward Error Correction
FIFO	First In, First Out
FPGA	Field Programmable Gate Array
FSM	Finite State Machine
GUI	Graphical User Interface
HDL	Hardware Description Language
HRC	High Rupturing Capacity
ID	Identification
IP	Intellectual Property
LMMI	Lattice Memory-Mapped Interface
MAC	Media Access Controller
MP	Multi-Protocol
MPLL	Multiplying Phase-locked Loop
MPP	Multi-Protocol Peripherals
MPPHY	Multi-Protocol PHY
PCIe	Peripheral Component Interconnect Express
PCS	Physical Coding Sublayer
PHY	Physical Layer
PIPE	PHY Interface for PCI Express
PLL	Phase-locked Loop
PMA	Physical Media Attachment
PRBS	Pseudo Random Bit Stream
RTL	Register Transfer Level
RSFEC	Reed-Solomon Forward Error Correction
RX	Receiver
SCFEC	Subset Code Forward Error Correction
SDI	Serial Digital Interface
SERDES	Serializer/Deserializer
SLVS-EC	Scalable Low Voltage Signaling – Embedded Clock
SRAM	Static Random-Access Memory

Acronym	Definition
SSC	Spread Spectrum Clock
SyncE	Synchronous Ethernet
TX	Transmitter
VCO	Voltage-Controlled Oscillator
VGA	Variable Gain Amplifier
XAUI	10 Gigabit Attachment Unit Interface
XGMII	10 Gigabit Media Independent Interface

1. Introduction

The MPPHY Module supports the most common high-speed SERDES protocols for inter-chip connectivity.

It is implemented as a Quad-based (4-lane) SERDES which is the fundamental block of this IP.

This design is implemented in Verilog. It can be targeted to Lattice Avant™ and Nexus™ 2 devices and implemented using the Lattice Radiant™ software Place and Route tool integrated with the Synplify Pro® synthesis tool.

1.1. Quick Facts

Table 1.1 shows a summary of the MPPHY Module.

Table 1.1. Quick Facts

IP Requirements	Supported Devices	Lattice Avant (Avant-G, Avant-X), Certus™-N2 (except LN2-CT-20ES).
	IP Changes[1]	For a list of changes to the IP, refer to the MPPHY Module Release Notes (FPGA-RN-02043).
Resource Utilization	Supported User Interface	Lattice Memory Mapped Interface (LMMI)
Design Tool Support	Lattice Implementation	IP Core v3.0.0 – Lattice Radiant software 2026.1 and Lattice Propel™ software 2026.1
	Synthesis	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. IP Support Summary

Table 1.2. MPPHY IP Support Readiness

Device Family	Protocol	Data Rate/Protocol Mode	Radiant Timing Model
Avant/Nexus 2	CoaXpress	12.5 Gbps	Preliminary
	CPRI	CPRI-2	Preliminary
		CPRI-3	Preliminary
		CPRI-4	Preliminary
		CPRI-5	Preliminary
		CPRI-6	Preliminary
		CPRI-7	Preliminary
		CPRI-7A	Preliminary
		CPRI-8	Preliminary
		CPRI-9	Preliminary
		CPRI-10	Preliminary
		CPRI-10-RSFEC	Preliminary
	DP/eDP	8.1 Gbps	Preliminary
		5.4 Gbps	Preliminary
		2.7 Gbps	Preliminary
		1.62 Gbps	Preliminary
	Ethernet	1000BASE-KX	Preliminary
		1000BASE-KX_10GHz[1]	Preliminary
		2.5GBASE-KX	Preliminary
		5GBASE R	Preliminary
		RXAUI 6.25 Gbps X2	Preliminary

Device Family	Protocol	Data Rate/Protocol Mode	Radiant Timing Model
		XAUI	Preliminary
		10GBASE-R	Preliminary
		10GBASE-R (ANLT = Enabled)	Preliminary
		10GBASE-R_SCFEC	Preliminary
		SyncE_10G	Preliminary
		25GBASE-R	Preliminary
		25GBASE-R (ANLT = Enabled)	Preliminary
		25GBASE-R_RSFEC	Preliminary
		25GBASE-R_SCFEC	Preliminary
		SyncE_25G	Preliminary
		25GAUI	Preliminary
	Generic	G64B66B	Preliminary
		G8B10B	Preliminary
		PMA_DIRECT	Preliminary
	JESD204B	1 Gbps to 12.5 Gbps	Preliminary
	JESD204C	1 Gbps to 26.5625 Gbps	Preliminary
	PCIe PIPE Direct	Gen4	Preliminary
		Gen3	Preliminary
		Gen2	Preliminary
		Gen1	Preliminary
	SLVS-EC	5 Gbps	Preliminary
		2.5 Gbps	Preliminary
		1.25 Gbps	Preliminary
	SDI	1.485 Gbps, Refclk = 148.5 MHz	Preliminary
		2.97 Gbps, Refclk = 148.5 MHz	Preliminary
		5.94 Gbps, Refclk = 148.5 MHz	Preliminary
		11.88 Gbps, Refclk = 148.5 MHz	Preliminary
		1.485 Gbps, Refclk = 74.25 MHz	Preliminary
		2.97 Gbps, Refclk = 74.25 MHz	Preliminary
		5.94 Gbps, Refclk = 74.25 MHz	Preliminary
		11.88 Gbps, Refclk = 74.25 MHz	Preliminary

Note:

1. The 1000BASEKX_10GHz Ethernet protocol mode uses a 10 GHz VCO clock frequency with a 10-bit PMA data width, whereas the 1000BASE-KX Ethernet protocol mode uses a 10.3125 GHz VCO clock frequency with a 20-bit PMA data width.

1.3. Features

Key features of the MPPHY Module include:

- Quad-based SERDES
- Supports up to gear ratio of 4
- Supports three data paths:
  - 8b10b data path
  - 64b66b data path
  - PMA only data path
- Integrated Loopback Modes for System Debugging

1.4. Conventions

1.4.1. Nomenclature

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

1.4.2. Signal Names

Signal names that end with:

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

1.4.3. Attribute

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

2. Functional Description

2.1. Overview

The MPPHY Module supports the most common high-speed SERDES protocols used for inter-chip connectivity. It is controlled by a number of registers that can be dynamically reconfigured by user logic or configuration module through LMMI ports.

The MPPHY Module supports the following two modes:

- PCS enabled mode—supports an array of popular data protocols including Ethernet, CoaXpress, CPRI, DP/eDP, G8B10B, G64B66B, JESD204B/C, PIPE Direct, and SLVS-EC.
- PCS bypass mode—allows a direct interface from PMA to fabric. The protocol-based logic can be fully or partially bypassed in a number of configurations to provide you the flexibility in implementing high-speed data interface.

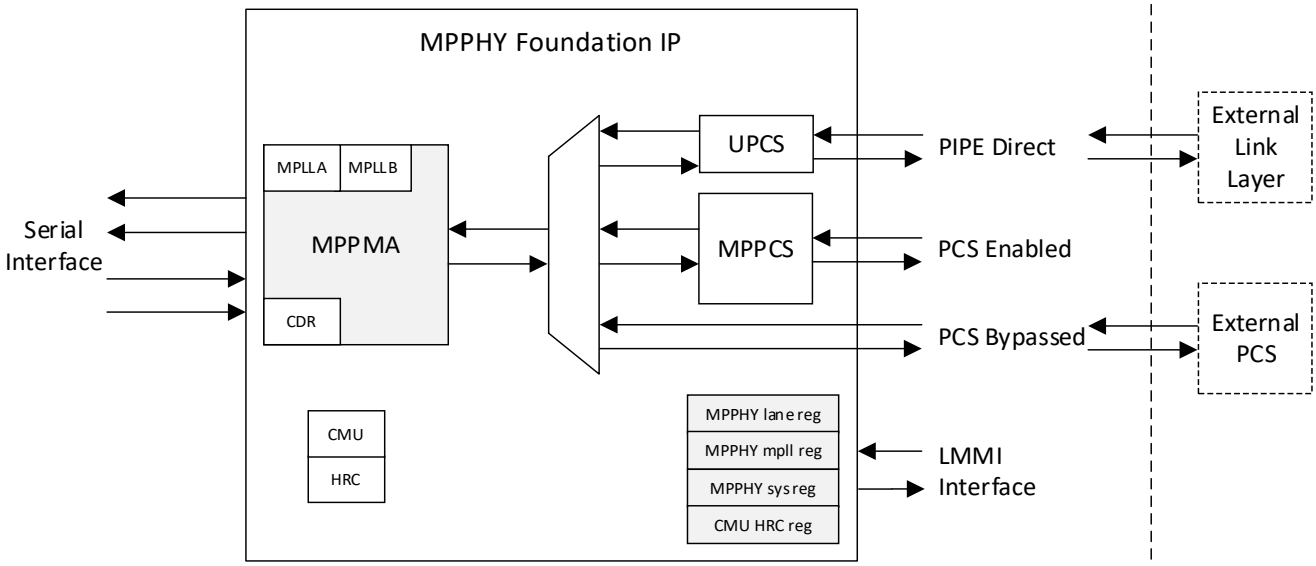

Figure 2.1. MPPHY Module Block Diagram

The MPPCS block consists of three data paths which are: 8b10b data path, 64b66b data path, and PMA only data path.

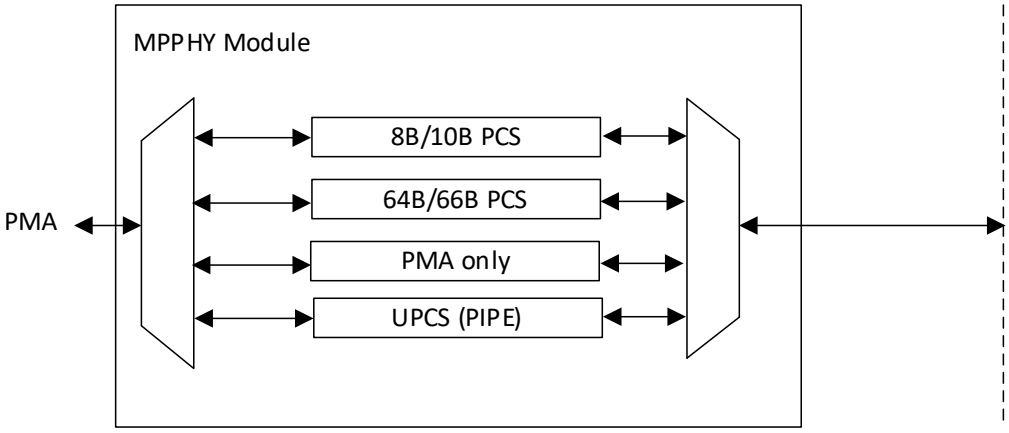

Figure 2.2. MPPCS Block Diagram

2.1.1. 8b10b PCS

The 8b/10b of MPPHY Module implements the most common functionalities required by 8b10b PCS. The functionalities include the following:

- 8b/10b encoding and decoding
- 10b code word boundary detection and alignment
- Clock frequency difference compensation
- Lane-to-lane deskew
- Bit swizzle and symbol swap

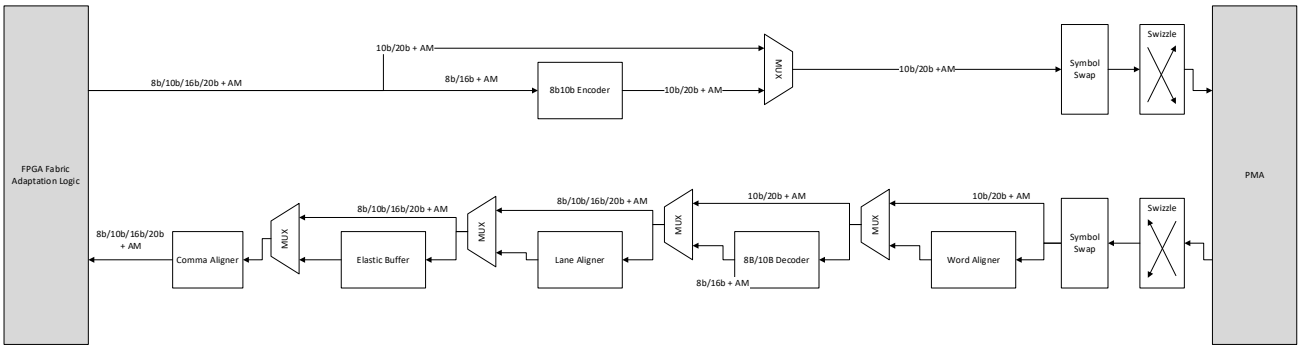

Figure 2.3. 8b10b PCS Chanel Functional Block Diagram

The 8b/10b PCS submodules and how to enable and disable them are explained in the following subsections.

2.1.1.1. TX FIFO

The TX FIFO submodule serves two purposes:

- Provide user logic input data with 2:1 and 4:1 gearing to internal MPPCS data path.
- Clock phase compensation FIFO to ease MPPCS-Fabric interface timing closure.

The features of TX FIFO are as follows:

- The *4:1 Gearing* can be optionally enabled to convert 8-byte data to 2-byte.
- The *2:1 Gearing* can be optionally enabled to convert 4-byte data to 2-byte.
- The *2:1 Gearing* can convert 2-byte data to 1-byte.
- The *TX FIFO* resolves clock phase difference between the read and write.
- The FIFO can signal the overflow and underflow status when occur. FIFO overflow or underflow may occur if the write and read clocks have frequency difference.
- In multiple-channel mode applications, the common clock and centralized control logic are optionally applied to all lanes to minimize lane-to-lane skew introduced by the uncertain latency of phase compensation FIFO.

2.1.1.2. 8b/10b Encoder

This encoder can be optionally bypassed by setting the *8b10b Encoder*. The encoder performs the 8-bit to 10-bit code conversion while maintaining the running disparity rules.

The features of 8b/10b encoder are as follows:

- Supports 1-byte mode; encodes one 8-bit input data in one clock cycle.
- Supports 2-byte mode; encodes two 8-bit input data in one clock cycle.
- Forces disparity function. For some protocols, the transmitter needs to start up with a certain disparity. The module allows you to replace the current running disparity calculated by this module with required disparity.
- Inverts disparity function.
- Forces disparity and inverts disparity are controlled by the tx_frcdisp and tx_dispval. The combinations of values are listed in the table below.

Table 2.1. Disparity

pma_tx*_data_i [90] [70] [30] [10] tx_frcdisp	pma_tx*_data_i [91] [71] [31] [11] tx_dispval	Disparity
1	0	Force negative disparity
1	1	Force positive disparity
0	0	Calculated by 8b10b encoder
0	1	Invert current disparity

- Forces this submodule to output the input 10b data (1-byte mode) or 20b data (2-byte mode).
- Supports interleaving mode performing 8b10b encoding. In this mode, the current input 8b data is used to calculate a new running disparity. The new value is used as running disparity for encoding the input data after the next data. The data streams are separated into two streams, encoded independently, and merged into one stream again. Lastly, the encoded data is transmitted to PMA with the same order as the data enters the 8b/10b encoder.

2.1.1.3. TX Lane Alignment

A multi-lane protocol defines a maximum skew for the data transmitted out from TX PMA across multiple lanes. To meet these specifications, follow these considerations:

- Bond TX FIFO across all lanes where the read clock of the FIFO needs to be from a single bonded clock source.
- Start the read operation of TX FIFO together across all the lanes.
- Clock the 8b10b TX path using a single bonded clock.
- The skew of bonded clock to TX PMA is controlled within 200 ps.

2.1.1.4. Word Aligner

This module can be optionally bypassed by setting *Word Alignment*. The data is serialized before transmission and then de-serialized at the receiver. The data loses the word boundary of the upstream transmitter upon deserialization. The word aligner receives parallel data from the de-serializer and restores the word boundary. To enable alignment, transmitters send a recognizable sequence (usually a comma) periodically. The receiver searches for the comma in the incoming data and when the receiver finds the comma, the receiver moves the comma to a byte boundary so the received parallel words match the transmitted parallel words.

The features of Word Aligner are as follows:

- Supports automatic alignment mode by setting *Automatic Word Alignment*.
- Provides link synchronization machine. The link synchronization state machine can be optionally enabled or disabled.
- The maximum allowed alignment pattern length is 20 bit. You can define the length as 10 bit or 20 bit.
- Provides programmable primary alignment pattern.
- Provides programmable secondary alignment pattern.
- Provides programmable alignment pattern mask code.
- Supports 1-byte mode and 2-byte mode of internal data bus width.
- Reports the number of bits slipped in the receiver word aligner.
- The length of synchronization code can be configured.
- All bits of synchronization code are configurable and maskable.
- Primary and secondary synchronization codes are provided separately.
- The *sync_det FSM* supports both 10b mode (bypassing 8b10b encoder) and 8b mode (after 8b10b decoding) of input data.

2.1.1.5. 10b/8b Decoder

This decoder can be optionally bypassed by setting *8b10b Decoder*. The features of the 10b/8b decoder include:

- Supports 1-byte mode; decodes one 10b data in one clock cycle.
- Supports 2-byte mode; decodes two 10b data in one clock cycle.
- Running disparity error detection.
- Invalid code detection.
- Running disparity output.

- Supports interleaving mode when performing 8b/10b decoding. In this mode, the current input 8B data is used to calculate a new value of running disparity. The new value is used as running disparity for decoding the input data after the next data. This function is symmetric to 8b/10b encoder interleaving mode.

2.1.1.6. RX Lane-to-Lane Deskew

This module can be optionally bypassed by setting *Lane Alignment*. This module transfers the input data in the respective recovered clock domains into a common clock domain. At the same time, this module aligns all lanes before forwarding the data to the next stage. This module looks for the predefined alignment pattern in the coming data and writes the data into a FIFO. When the first alignment pattern is received on a certain lane, the de-skew FIFO read and write pointers are both reset to zero. The write pointer is then incremented for the subsequent input data. The read pointer keeps at zero and the output data of the FIFO read is repeated for the last read-out data. When the alignment pattern is detected by all lanes, the read pointer of the FIFO starts incrementing for each read clock cycle, aligning all lanes.

2.1.1.7. Clock Frequency Compensation

This module can be optionally bypassed by setting *Clock Frequency Compensation*. This module performs clock frequency adjustment between the recovered receive clock domain and the local system clock domain. Inserting or deleting bytes at position where SKIP pattern is detected, without causing loss of packet data, performs clock compensation. A 32-byte elasticity FIFO is used to temporarily buffer coming data from recovered receive clock domain and transfer the data to local system clock domain.

The features of clock frequency compensation block are as follows:

- Match the programmable SKIP pattern as shown in the figure below. The SKIP pattern usually has this format: a COM byte optionally followed by 1 to 3 SKP bytes. All bytes (byte 0 to 3) are configured by registers.

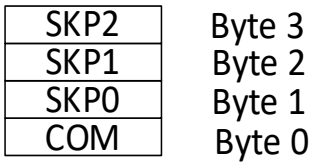

Figure 2.4. SKIP Pattern Format

- The length of SKIP pattern can be set to 1, 2, or 4 byte.
- Provides two SKIP patterns: Primary and Secondary SKIP.
- Provides a mask code to allow partially matching the SKIP pattern.

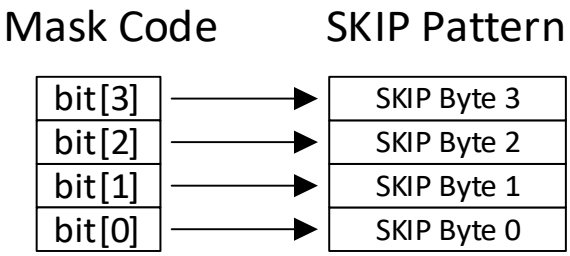

Figure 2.5. SKIP Pattern Mask Code

- Supports both 10b data mode (bypassing 8b10b decoder) and 8b data mode (after 8b10b decoding).
- Guarantees a minimum number of bytes between packets after SKIP deletion.
- Programmable high water line and low water line.
- Observes deletion and insertion of SKIP pattern using user logic.
- Signals the Elastic FIFO overrun and underrun status.

2.1.1.8. RX FIFO

The RX FIFO submodule serves two purposes:

- 1:2 and 1:4 gearing MPPCS data before forwarding to Fabric.
- Clock phase compensation FIFO to ease MPPCS-Fabric interface timing closure.

The features of the RX FIFO are as follows:

- The *1:4 Gearing* can be optionally enabled to convert 2-byte data to 8-byte.
- The *1:2 Gearing* can be optionally enabled to convert 2-byte data to 4-byte.
- The *1:2 Gearing* can convert 1-byte data to 2-byte.
- The *RX FIFO* resolves clock phase difference between read and write clocks.
- FIFO overflow or underflow may occur if the write and read clocks have frequency difference.
- Works as synchronous DFF to launch data to Fabric when the *RX FIFO* is bypassed.
- In multiple-channel mode, the common clock and centralized control logic are optionally applied to the RX FIFO on each lane to minimize the lane-to-lane skew introduced by the uncertain latency of the FIFO.

2.1.2. 64b66b PCS

The MPPHY Module implements 10GBASE-R PCS and 25GBASE-R PCS defined by IEEE802.3 protocol. This module has the following functional sub-blocks:

TX path:

- Packet body encoding: encode 72b XGMII data (64b TXD + 8b TXC) to 66b packet (2b header + 64b body)
- Transmit FSM: checks the validity of the 64-bit data from the MAC layer and ensures proper block sequencing
- Packet body scrambler
- PCS-PMA gearbox (66b to 32b conversion)
- Test pattern generation (square wave and scrambled idle test pattern)
- Loopback mode

RX path:

- PMA-PCS gearbox (32b to 66b conversion)
- 66b packet alignment (delimit) and lock state machine
- Packet body descrambler
- Packet body decoding: decode 66b packet to 72b XGMII data (64b RXD + 8b RXC)
- Receive FSM
- Test pattern compare and error counting
- BER monitor in normal mode (disabled in test mode)
- Loopback mode
- Clock compensation: compensate up to a ±100 ppm clock difference between the remote transmitter and the local receiver (insert idles, delete idles, or delete sequence ordered sets)

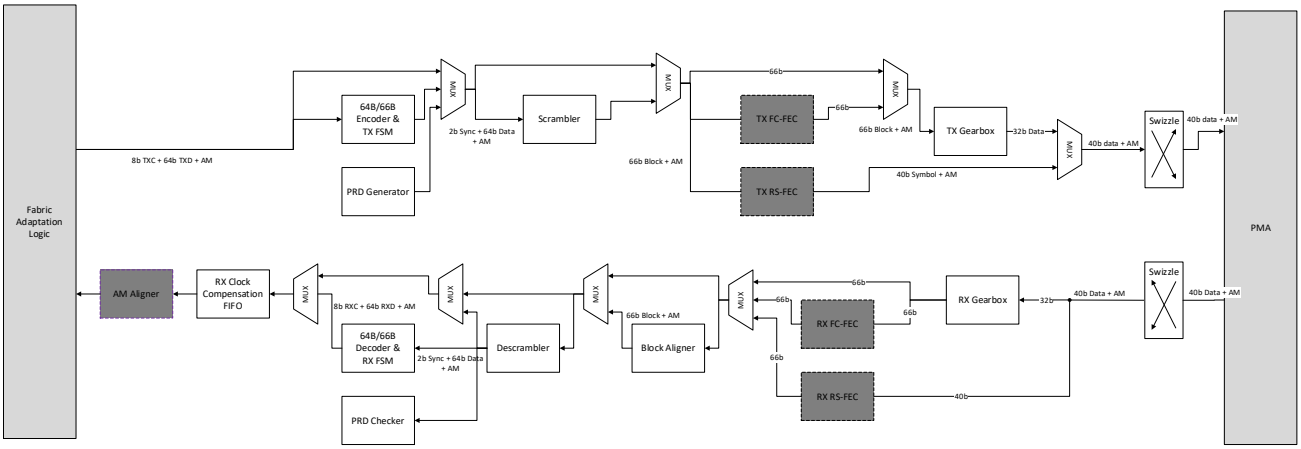

Figure 2.6. 64b66b PCS Chanel Functional Block Diagram

2.1.2.1. 64B66B Encoder

This module encodes 64-bit XGMII data and 8-bit XGMII control into 10GBASE-R 66-bit control or data blocks in accordance with Clause 49 of IEEE 802.3-2008 specification.

The features of encoder are as follows:

- This module can be optionally bypassed by setting *64b66b Encoder.*
- Directly sends the 66-bit input data to the output of this module in force data mode.
- Encodes 64-bit XGMII data and 8-bit XGMII control into 10GBASE-R 66-bit packet if not bypassed.
- Implements the TX FSM in accordance with IEEE 802.3-2008 specification.
- The coding of FSM is registered and accessed by register access interface for debug purpose.

2.1.2.2. Scrambler

This module scrambles the 64-bit block payload data using x58 + x39 +1 polynomial specified by Clause 49 of IEEE 802.3-2008 specification. This module also generates the pseudo-random pattern for test purposes.

The features of scrambler are as follows:

- This module can be optionally bypassed by setting *Scrambler*.
- Scrambles the entire 64-bit payload of a 66-bit block, except for the two sync header bits.
- Sends the 66-bit input data directly to the output of this module if bypassed.
- Generates the pseudo-random test-pattern as 64-bit payload of packet.
  - The seed of pseudo-random test-pattern is configurable as follows:
    - Seed A
    - Seed B
  - The data pattern is configurable as follows:
    - 64-bit zeros or inverse, depending on the seed setting.
    - 64-bit encoding for two local fault order_sets or inverse, depending on seed setting.
  - The sync header is fixed to the control sync header, which is 2’b10.
- Generates the scrambled idle test pattern (25GBASE-R).
  - The input to the scrambler is a control block (block type = 0x1E) with all idles.
  - The sync header is fixed to the control sync header, which is 2’b10.

2.1.2.3. Reed Solomon FEC (RS-FEC)

This module is implemented in accordance with Clause 108 of IEEE 802.3-2022 specification and Clause 6.9 of CPRI Specification V7.0. This module is compatible with 10GBASE-R, 25GBASE-R, and CPRI protocol modes. This module can be bypassed by setting *FEC Mode*.

When this block is enabled, use the following settings depending on your test application:

- For software simulation, enable *Fast Simulation Mode*.
- For hardware testing and bitstream generation, disable *Fast Simulation Mode*.

2.1.2.4. Firecode FEC (FC-FEC)

This module is implemented based on Clause 74 of IEEE 802.3-2022 specification. This module is meant for 10GBASE-R and other BASE-R PHY. The FEC code used is a shortened cyclic code (2112, 2080) for error checking and forward error correction.

The FEC block length is 2112 bits. The code encodes 2,080 bits of payload (or information symbols) and adds 32 bits of overhead (parity symbols).

The shortened cyclic code (2112, 2080) is guaranteed to correct an error bus of up to 11 bits per block.

2.1.2.5. TX Gearbox

This module adapts between the 66-bit width of the 64B/66B block and the 40-bit/32-bit/16-bit width of the PMA data bus. This module also generates bit sequence of square wave and sends the bit sequence to PMA for transmitter or receiver tests.

The features of TX gearbox are as follows:

- Converts the 66-bit 10GBASE-R or 25GBASE-R block into 40-bitor 32-bit or 16-bit PMA data.

- Generates square wave pattern.
  - Sends a repeating pattern of n ones followed by n zeros where n is configurable between 4 and 11.
  - Sends 32-bit data of the square wave pattern to PMA at a time.

2.1.2.6. Bit Swizzle

Some protocols require 66b data block to have bit swizzling before being transmitted out to serial line. The received 66b block data from serial also needs to be bit swizzled before MPPCS processes the data. The parallel data width to PMA is 32b in non-FEC mode and 40b for FEC mode, thus bit swizzling need to be performed on the bits out of the 40b data bus.

2.1.2.7. RX Gearbox

This module adapts between the 66-bit width of the 64B/66B block and the 32-bit width of PMA. This module converts 40-bitor 32-bit or 16-bit PMA data into 66-bit data block (without block aligning).

2.1.2.8. Block Aligner

This module determines the 66b block boundary based on the data received from RX gearbox. The incoming data stream is slipped one bit at a time until the required number of valid synchronization header (bit 0 and bit 1) is detected in the received data stream.

The features of the block aligner are as follows:

- Implements block lock FSM defined in Clause 49 of IEEE 802.3-2008.
- Shifts the incoming data stream and determines the block boundary.
- Reports bit number of shifting through status register.
- Drives block_lock high to signal fabric when block lock is achieved.
- This module can be optionally bypassed by setting *Block Aligner*.

2.1.2.9. Descrambler

This module descrambles the 64-bit block payload and checks the received pseudo-random test-pattern. The details are specified in Clause 49 of IEEE 802.3-2008 specification.

The features of the descrambler are as follows:

- This module can be optionally bypassed by setting *Descrambler*.
- Descrambles the entire 64-bit payload of a 66-bit block, except for the two sync header bits.
- Checks the pseudo-random test-pattern and counts the mismatch errors.

2.1.2.10. 64B66B Decoder

This module reverses the 64B/66B encoding process. The decoder block also contains a state machine (RX SM) designed in accordance with the IEEE 802.3-2008 specification, Clause 49.

The features of decoder are as follows:

- This module can be optionally bypassed by setting *64b66b Decoder*.
- Converts 10GBASE-R 66-bit packet to 64-bit XGMII data and 8-bit XGMII control if not bypassed.
- Implements the RX FSM in accordance with IEEE 802.3-2008 specification.
- Performs functions such as sending local faults to the Media Access Control (MAC) and Reconciliation Sublayer (RS) under reset and substituting error codes when the 10GBASE-R PCS rules are violated.

2.1.2.11. Clock Compensation FIFO

This module can be optionally bypassed by setting *Clock Compensation*. This asynchronous FIFO is used to adapt RX path clock frequency and phase difference between 64B/66B PCS and FPGA fabric adaptation logic.

The features of the RX FIFO are as follows:

- Reports the following FIFO status:
  - FIFO partial full
  - FIFO partial empty
  - FIFO half full

2.1.3. PCS Bypass

In PCS Bypass mode, both 8b/10b PCS and 64b/66b PCS are bypassed. with the MPPHY Module has a direct connection from Fabric Adaption block to PMA. PMA only mode is needed for the protocol that requires low latency. This mode is also useful for any protocol that is not supported by 8b/10b PCS and 64b/66b PCS and can be implemented in soft IP together with PMA only mode.

PMA only mode allows data bus to be configured from 8 bit up to 40 bit.

In this mode, MPPHY Module supports PMA direct 66 bit mode, which is mainly applied for the JESD204C protocol. This mode is enabled when you select data width of 66 bit.

2.1.4. Loopbacks

The figure below shows the data flow of MPPCS and MPPMA loopback modes. For 8b10b PCS and 64b66b PCS channel functional block diagrams, refer to Figure 2.3 and Figure 2.6 respectively.

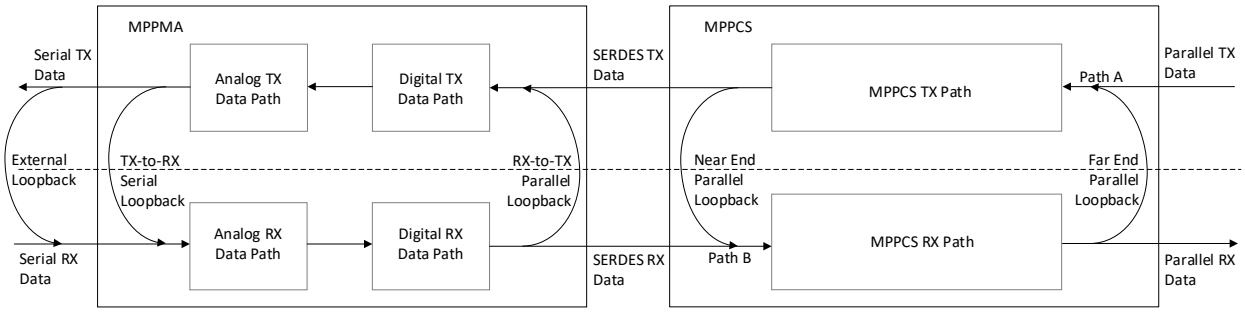

Figure 2.7. MPPCS and MPPMA Loopback Diagram

Table 2.2. MPPCS Loopback Description

Loopback	Description
Path A – Far End Parallel Loopback	When enabled, MPPCS RX data path loops back to the MPPCS TX data path. Can be used to test and debug the digital data path end-to-end.
Path B – Near End Parallel Loopback	When enabled, MPPCS TX data path loops back to the MPPCS RX data path.

Table 2.3. MPPMA Loopback Description

Loopback	Description
TX-to-RX Serial Loopback	Loops back the data from analog TX path back to analog RX path. In this mode, you send data pattern on the TX data path and the data is looped back in the PHY analog transceiver. The data is then returned through the RX data path in the PCS block.
RX-to-TX Parallel Loopback	When enabled, SERDES RX data path loops back to the SERDES TX data path.

2.1.5. Lane Merging

The Lattice Avant and Nexus 2 devices support seven multi-protocol quads where each quad contains 4 lanes with a single reference clock and 2 PLLs. For different modules to be merged into a quad, the modules must meet the following requirements:

- When PCIe operates at max_link_width of x4, x2, or x1, unused channels can be configured as PCIe PIPE Direct mode.
- For non PCIe protocols, the same protocol of the same rate with granularity of x1 or x2 can be merged into a single quad.
- For protocols that operate >8 lanes, use clock bonding to bond multiple quads to minimize transmit skew.
- For simplex protocols regardless of PCS encoding, the modules can be merged into a single lane if the modules are running on the same reference clock and same rate.

- Ethernet state mode merging allows merging the 1G, 2.5G, 5G, 10G, or 10G and 25G protocols into a single quad.
- Same or different protocols with different rates only can be merged into the same quad if the modules have the same reference clock and different PLLs or same PLL with similar PLL settings. For example, if protocol A and B have the same reference clock, while protocol A uses PLLA and protocol B uses PLLB, the modules can be merged into a quad. Another example, if protocol A and B have the same reference clock and both protocols use PLLA, the modules can be merged into a quad if all PLLA settings of these 2 protocols are similar.

The following ports must be shared for quad merging to work:

```
// only 1 ref clock per quad
refclk_p_q0_i
refclk_n_q0_i

// only 1 lmmi arbiter per quad
lmmiclk_q0_i
lmmireset_n_q0_i
lmmierror_interrupt_q0_o

// reset signal per quad
soft_phy_rst_n_q0_i

// only 1 sram per quad
phy_sram_ext_ld_done_dyn_q0_i
phy_sram_bypass_dyn_q0_o
phy_sram_init_done_q0_o

// only 2 plls per quad
phy_mplla_state_q0_o
phy_mpllb_state_q0_o

// choose the highest frequency
txoutgclk_pll1_q0_o
txoutgclk_pll0_q0_o
```

2.1.6. Dynamic Reconfiguration

Dynamic reconfiguration is a method for MPPHY to switch between different protocol modes to support rate changes or protocol mode transitions.

2.1.6.1. Scope

- Dynamic reconfiguration is allowed only between different variants of the same protocol.
- The reference clock must remain unchanged.
- *Link Width*, *Lane ID*, and *Link Direction* attributes must remain unchanged.
- Dynamic reconfiguration is allowed only from the base profile (Profile 1) to targeted profiles (Profile 2, 3, 4). To perform dynamic reconfiguration between two target profiles (for example, from Profile 2 to Profile 3), the IP must first be reconfigured from Profile 2 back to the base profile (Profile 1) and then reconfigured to Profile 3.
- The maximum number of supported profiles are 4, including base profile.
- For multi-lane design, because of lane merging limitation when a lane is dynamically reconfigured to another profile, ensure that there is only one unique setting for each MPLL. Refer to the Lane Merging section for more details on lane merging.

- Dynamic reconfiguration supports the following protocols:
  - Ethernet, except XAUI and RXAUI
  - DP_eDP
  - CPRI
  - SDI
  - SLVS-EC
  - JESD204B
  - JESD204C

2.1.6.2. PHY Reconfiguration File (PRF)

The PHY Reconfiguration File (PRF) is generated when dynamic reconfiguration is enabled and follows the Lattice MEM format. Each line is a 32-bit hexadecimal value where the first 16 bits are the register address and the subsequent 16 bits are the register values. The file contains only the writable registers. Depending on the MPLL, only the relevant register for that MPLL is generated. The 2 types of PRF files are as follows:

- Full file—Contains full registers of each profile.
- Diff file—Contains the differences between registers address and values of the base protocol mode and the profiles.

```
#Format=Hex
#Depth=205
#Width=32
#AddrIndex=16
#DataIndex=0
5000000A
50080000
500A0000
500B0000
500C007C
500D0001
500E0083
500F0002
50100000
50110000
50120403
50130504
501550BC
50160000
50170001
5018C5BC
50190000
501A0001
501B0000
501C0000
501D000A
501E0000
501F007C
```

Figure 2.8. Example of the PRF File

2.1.6.3. Dynamic Reconfiguration Profile

The dynamic reconfiguration profile is a predefined set of PHY/SERDES configuration parameters that can be switched at runtime, without synthesizing or re-programming the FPGA.

The PRF output file paths when number of profiles = 2 are as follows:

```
generated_IP_directory/
└──── prf/
   ├ profile1/
   │   ├──── profile1.prf
   │   └──── profile1_profile2_prf_diff.prf
   └──── profile2/
      ├──── profile2.prf
      └──── profile2_profile1_prf_diff.prf
```

During the dynamic reconfiguration process, only the *diff.prf files are utilized in step 13 in the Dynamic Reconfiguration Sequence section. The data within these files is written to the registers via the LMMI interface, based on the specified register addresses and the corresponding values defined in each *diff.prf file.

2.1.6.4. Dynamic Reconfiguration Sequence

To perform dynamic reconfiguration, follow these steps:

1. Assert the tx_pcs_rst_n_q[m]_i and rx_pcs_rst_n_q[m]_i signals.
2. Wait for the tx_pcs_reset_done_q[m]_o and rx_pcs_reset_done_q[m]_o signals to deassert.
3. Set the reg173.pma_ctrl_rx_dr_en register to 1’b1.
4. Wait for the reg17A.pma_ctrl_rx_dr_ready register to be asserted.
5. Set the reg15E.phy_tx_pstate register to 2’b10.
6. Perform a four-way request-ack handshaking using the phy_tx_req and phy_tx_ack registers.
   a. Assert the phy_tx_req register.
   b. Wait for the phy_tx_ack register to be asserted.
   c. Deassert the phy_tx_req register.
   d. Wait for the phy_tx_ack register to be deasserted.
7. Set the reg14C.phy_rx_pstate register to 2’b10.
8. Perform a four-way request-ack handshaking using the phy_rx_req and phy_rx_ack registers.
   a. Assert the phy_rx_req register.
   b. Wait for the phy_rx_ack register to be asserted.
   c. Deassert the phy_rx_req register.
   d. Wait for the phy_rx_ack register to be deasserted.
9. Configure the MPLL. If the target protocol uses PLLA, set the reg15F.phy_tx_mpll_sel register to 0. If the target protocol uses PLLB, set the reg15F.phy_mpll_sel register to 1.
10. Perform a four-way request-ack handshaking using the phy_tx_req and phy_tx_ack registers (Repeat step 6).
11. Disable the MPLL by setting the reg15F.phy_tx_mpll_en register to 0.
12. Perform a four-way request-ack handshaking using the phy_tx_req and phy_tx_ack registers (Repeat step 6).
13. Configure MPLL and lane registers according to the PRF diff file.
14. For initial dynamic reconfiguration, set the reg58.phy_mpll_recal_force_en register (for MPLLA) or the reg59.phy_mpll_recal_force_en register (for MPLLB) to 1. Before doing so, ensure the MPLL is disabled by setting the reg15F.phy_tx_mpll_en register to 0.
15. Enable the MPLL by setting the register15F.phy_tx_mpll_en register to 1.
16. Perform a four-way request-ack handshaking using the phy_tx_req and phy_tx_ack registers (Repeat step 6).
17. Perform a four-way request-ack handshaking using the phy_rx_req and phy_rx_ack registers (Repeat step 8).
18. If step 14 is performed, set the reg58.phy_mpll_recal_force_en register (for MPLLA) or the reg59.phy_mpll_recal_force_en register (for MPLLB) to 0. If step 14 is not performed, skip this step.

19. Set the reg15E.phy_tx_pstate register to 2’b00 and the reg14C.phy_rx_pstate register to 2’b00.
20. Perform a four-way request-ack handshaking using the phy_tx_req and phy_tx_ack registers (Repeat step 6).
21. Perform a four-way request-ack handshaking using the phy_rx_req and phy_rx_ack registers (Repeat step 8).
22. Set the reg173.pma_ctrl_rx_dr_en register to 1’b0.
23. Set the tx_pcie_pma_rst_n_q[m]_i and rx_pma_rst_n_q[m]_i signals to 1’b0.
24. Wait for the rx_pma_reset_done_q[m]_o and tx_pcie_pma_reset_done_q[m]_o signals to deassert.
25. Set the tx_pcie_pma_rst_n_q[m]_i and rx_pma_rst_n_q[m]_i signals to 1’b1.
26. Set the tx_pcs_rst_n_q[m]_i and rx_pcs_rst_n_q[m]_i signals to 1’b1.

Table 2.4 lists the available registers used in dynamic reconfiguration. Detailed information of each register is described in the subsequent tables. For the phy_tx_req, phy_tx_ack, phy_rx_req, and phy_rx_ack registers, refer to the Register Description section.

Table 2.4. MPP Registers (Offset Address in Byte)

LMMI Word Offset	LMMI Byte Offset	Register Name	Access Type	Description
8'hB9	8’h173	reg173	RW	PMA_CTRL3
8'hBD	8’h17A	reg17A	RO	PMA_CTRL_STATUS0
8'hAF	8’h15E	reg15E	RW	PHY_LANE_TX_REQUEST0
8’hAF	8’h15F	reg15F	RW	PHY_LANE_TX_REQUEST1
8'hA6	8’h14C	reg14C	RW	PHY_LANE_RX_REQUEST4

Table 2.5. Register (reg173) Description

Field	Name	Description	Access	Default
[2:0]	rsvd	Reserved.	—	3’b0
[3]	pma_ctrl_rx_dr_en	1'b0: Disables dynamic reconfiguration (Default). 1'b1: Enables dynamic reconfiguration.	RW	1’b0
[7:4]	rsvd	Reserved.	—	4’b0000

Table 2.6. Register (reg17A) Description

Field	Name	Description	Access	Default
[6:0]	rsvd	Reserved.	—	6’b000000
[7]	pma_ctrl_rx_dr_ready	Status indicates PMA control is ready for dynamic reconfiguration 1'b0: PMA control is still progressing, not ready for dynamic reconfiguration. 1'b1: PMA control is idle and ready for dynamic reconfiguration.	RO	1’b0

Table 2.7. Register (reg15E) Description

Field	Name	Description	Access	Default
[1:0]	phy_tx_pstate	Transmitter power state Sets the power state of the transmitter. 2'b00: P0 – Transmitter is fully powered up. 2'b01: P0s – Transmitter common mode is hold, TX analog clocks are active, but TX serializer is off. 2'b10: P1 – Transmitter common mode is hold, but TX analog clocks and TX serializer are off.	RW	2’b00

Field	Name	Description	Access	Default
		2'b11: P2 – Transmitter is powered down.		
[7:2]	rsvd	Reserved.	—	6'b000000

Table 2.8. Register (reg15F) Description

Field	Name	Description	Access	Default
[3:0]	rsvd	Reserved.	—	4’b0000
[4]	phy_tx_mpll_en	TX MPLL enable Powers up or powers down the MPLLA or MPLLB (determined using txX_mpllb_sel). For normal operation, this input must be set to 1 when txX_pstate[1:0] is in P0 or P0s state.	RW	1’b1
[5]	phy_tx_mpll_sel	TX MPLLB select Specifies the source of the TX analog clocks. If asserted, MPLLB is the source. If deasserted, MPLLA is the source.	RW	1’b0 if MPLLA 1’b1 if MPLLB
[7:6]	rsvd	Reserved.	—	2’b00

Table 2.9. Register (reg14C) Description

Field	Name	Description	Access	Default
[1:0]	phy_rx_pstate	Receiver power state Sets the power state of the receiver. The state mappings are as follows: 2'b00: P0 – Receiver is fully powered up and output receive clocks are active. 2'b01: P0s – Receiver voltage-controlled oscillator (VCO) is in continuous calibration mode, output receive clocks are not available. 2'b10: P1 – Receiver analog front-end (AFE) and voltage regulators are powered up, but RX VCO is in reset. 2'b11: P2 – Receiver signal detector is powered up and the rest of receiver is powered down.	RW	2'b00
[7:2]	rsvd	Reserved.	—	6'b000000

Table 2.10. MPLL Register

Register Address	Register Name	Access Type	Description
0x5802	reg58	RW	PHY_MPLL_CONTROL1
0x5902	reg59	RW	PHY_MPLL_CONTROL1

Table 2.11. Register (reg58 and reg59) Description

Field	Name	Description	Access	Reset
[1:0]	rsvd	Reserved.	—	2'b00
[2]	phy_mpll_recal_force_en	MPLL force re-calibration Forces a recalibration of the MPLL and TX DCC in the selected bank, regardless of the state of the corresponding done signal. **Note**: This input can be changed only when the MPLL is powered down (txX_mpll_en = 0 and mpll(a,b)_force_en =	RW	1’b0

Field	Name	Description	Access	Reset
		0). This input has priority over the mpll(a,b)_recal_skip_en signal.		
[7:3]	rsvd	Reserved.	—	5'b00000

2.2. Signal Description

Table 2.12. MPPHY Module Signal Description

Port Name	I/O	Width	Description
Serial I/O[1]			
tx_l[n]_p_o	Out	1	Serial transmitter lane, differential signal with positive polarity.
tx_l[n]_n_o	Out	1	Serial transmitter lane, differential signal with negative polarity.
rx_l[n]_p_i	In	1	Serial receiver lane, differential signal with positive polarity.
rx_l[n]_n_i	In	1	Serial receiver lane, differential signal with negative polarity.
Clock and Reset[2]			
dl_in_clk_q[m]_i	In	NL	Deterministic latency clock is used for MPPHY latency measurement. This clock must be set to any clock frequency that is different from the data path clock frequency, for example, DL clock is 100 MHz while data path clock is 125 MHz. A more complex ratio between these two clocks can result in a more accurate measurement, for example, a DL clock of 153.84615 MHz. This clock can be connected to any available clock source except the data path clock.
tx_in_clk_q[m]_i	In	NL	Input clock in the transmit path. Loopback clock from PMA for source-synchronous clocking. It is recommended to use a clock that is faster than tx_out_clk for asynchronous operation.
rx_in_clk_q[m]_i	In	NL	Input clock in the receive path. It is recommended to use a clock that is faster than rx_out_clk for asynchronous operation.
txoutclk_l[n]_o	Out	1	Output transmit clock from PMA with applicable down-gear.
txoutgclk_pll[r]_q[m]_o	Out	1	Output transmit clock forwarded to global clock distribution.
rxoutclk_l[n]_o	Out	1	Output receive clock on each lane from PMA with applicable down-gear. It is recommended to use this clock for source-synchronous clocking.
rxoutgclk_l[n]_o	Out	1	Output receive clock forwarded to global clock distribution.
mppcs_rx_rst_n_q[m]_i	In	NL	Active low asynchronous RX PMA reset.
mppcs_rx_rst_req_q[m]_i	In	NL	Request for MPPCS RX lane reset in common FIFO. Polarity definition: 0000 – No request 0001 – Request for MPPCS RX lane[0] reset 0010 – Request for MPPCS RX lane[1] reset 0100 – Request for MPPCS RX lane[2] reset 1000 – Request for MPPCS RX lane[3] reset 1111 – Request for MPPCS RX lane[3] reset (default) Others – Reserved bits

Port Name	I/O	Width	Description
pin_perst_n_q[m]_i	In	NL	Active low, per-lane asynchronous PCIe reset signal that is consumed by external device. The reset comes from GPIO pin to HRC.
rx_pcs_rst_n_q[m]_i	In	NL	Active low asynchronous RX PCS reset.
rx_pma_rst_n_q[m]_i	In	NL	Active low asynchronous PHY PMA receiver reset signal.
soft_core_rst_n_q[m]_i	In	NL	Active low asynchronous PCIe core reset signal.
soft_phy_rst_n_q[m]_i	In	1	Active low asynchronous MPPHY reset signal.
tx_pcie_pma_rst_n_q[m]_i	In	NL	Active low asynchronous PHY PMA and uPCS (PCIe) tranceiver reset signal.
tx_pcs_rst_n_q[m]_i	In	NL	Active low asynchronous MPPCS tranceiver reset signal.
rx_pcs_reset_done_q[m]_o	Out	NL	Per-lane reset complete indication for MPPCS receiver. This indication is distributed from HRC to fabric core. Polarity definition: 0 – MPPCS receiver is in reset. 1 – MPPCS receiver is out of reset.
rx_pma_ctrl_dr_ready_q[m]_o	Out	NL	Status signal to indicate PMA control is ready for dynamic reconfiguration.
rx_pma_reset_done_q[m]_o	Out	NL	Per-lane reset complete indication for PHY PMA receiver. This indication is distributed from HRC to fabric core. Polarity definition: 0 – PHY PMA receiver is in reset. 1 – PHY PMA receiver is out of reset.
tx_pcie_pma_reset_done_q[m]_o	Out	NL	Per-lane reset complete indication for PHY PMA and uPCS (PCIe) transceiver. This indication is distributed from HRC to fabric core. Polarity definition: 0 – PHY PMA and uPCS transceiver is in reset. 1 – PHY PMA and uPCS transceiver is out of reset.
tx_pcs_reset_done_q[m]_o	Out	NL	Per-lane reset complete indication for MPPCS transceiver. This indication is distributed from HRC to fabric core. Polarity definition: 0 – MPPCS transceiver is in reset. 1 – MPPCS transceiver is out of reset.
core_reset_done_q[m]_o	Out	NL	Indicates that the PCIe core reset is complete.
MPPCS Mode Signals			
phy_sram_ext_ld_done_dyn_q[m]_i	In	1	SRAM external load complete. Indicates that the signal you asserted after any updates to the SRAM have been loaded.
pma_tx[n]_data_i	In	160	Input transmit data on each lane. Refer to the Data Bus Mapping section for the data mapping of each mode.
pma_tx[n]_valid_i	In	1	Input flow-control feed-forward signal.
pm_rx[n]_deskew_en_i	In	1	The rising edge of this signal triggers the lane-to-lane de-skew operation.
pm_rx[n]_walign_en_i	In	1	User logic may re-enable word alignment when loss of synchronization is detected. This function is useful if the automatic synchronization is not enabled. The rising edge of this signal triggers the word alignment operation.

Port Name	I/O	Width	Description
rx_data_en_q[m]_i	In	NL	Enables the receiver data output. When asserted in P0 power state, the RX data path is enabled and the CDR begins tracking the received data.
tx[n]_am_i	In	4	DL bit for latency measurement.
tx[n]_en_i	In	4	Tx electrical idle to PMA.
phy_mplla_state_q[m]_o	Out	1	MPLLA state indicator. This signal is asserted when MPLLA is powered-up and phase-locked. Note that this signal is not a true PLL locked indicator and is based on a fixed timer.
phy_mpllb_state_q[m]_o	Out	1	MPLLB state indicator. This signal is asserted when MPLLB is powered-up and phase-locked. Note that this signal is not a true PLL locked indicator and is based on a fixed timer.
phy_rx_ack_q[m]_o	Out	NL	Receiver acknowledge. Assertion of this signal indicates the requested receiver setting is complete.
phy_sram_bypass_dyn_q[m]_o	Out	1	SRAM bypass. 1'b1 -- the adaptation and calibration algorithms are executed from the hardwired values within the raw PCS. 1'b0 -- the internal algorithms are first loaded by the Raw PCS into the SRAM at which into the SRAM at which point you can change the contents of the SRAM.
phy_sram_init_done_q[m]_o	Out	1	SRAM initialization complete. Signal indicating that the SRAM has been initialized by the boot loader.
phy_tx_ack_q[m]_o	Out	NL	Transmitter acknowledge. The assertion of this signal indicates the requested transmitter setting is complete.
pma_rx[n]_blk_lock_o	Out	1	The high level of this signal indicates the block lock is achieved.
pma_rx[n]_data_o	Out	160	Output received data on each lane. Refer to the Data Bus Mapping section for the data mapping of each mode.
pma_rx[n]_ebuf_empty_o	Out	1	1: The frequency compensation buffer (elastic buffer) is empty. 0: The buffer is not empty.
pma_rx[n]_ebuf_full_o	Out	1	1: The frequency compensation buffer (elastic buffer) is full. 0: The buffer is not full.
pma_rx[n]_fec_lock_o	Out	1	When the signal is high, this signal indicates the FEC (RSFEC or FCFEC) lock is achieved.
pma_rx[n]_get_lalign_o	Out	1	The output of automatic lane alignment FSM. 1: Alignment acquired 0: Loss of alignment
pma_rx[n]_get_lsync_o	Out	1	1: Link synchronization is acquired. This is the output of link synchronization FSM. 0: Loss of link synchronization **Note**: This port is not aligned to data path and the signal reaches FPGA fabric earlier than the aligned data. Thus, you need to wait for some times to sample the correct data after this signal is asserted.
pma_rx[n]_hi_ber_o	Out	1	When the signal is high, this signal indicates high BER is detected.

Port Name	I/O	Width	Description
pma_rx[n]_sigdet_hf_o	Out	1	Receive high-frequency signal detection output. When asserted, indicates that the receiver is receiving high-frequency signals.
pma_rx[n]_sigdet_lf_o	Out	1	Receive low-frequency signal detection output. When asserted, indicates that the receiver is receiving low-frequency signals.
pma_rx[n]_valid_o	Out	1	When asserted, indicates valid data from RX FIFO block.
rx[n]_am_o	Out	4	DL bit for latency measurement.
rx_cdr_valid_q[m]_o	Out	NL	Assertion of this signal indicates CDR-lock.
rx_valid_q[m]_o	Out	NL	When asserted, indicates CDR lock.
LMMI Interface			
lmmiclk_q[m]_i	In	1	LMMI clock input.
lmmireset_n_q[m]_i	In	1	Active low LMMI reset.
lmmirequest_q[m]_i	In	1	Start transaction.
lmmiwrrd_n_q[m]_i	In	1	Write = HIGH, Read = LOW.
lmmioffset_q[m]_i	In	16	Address/offset.
lmmiwdata_q[m]_i	In	16	Write data.
lmmirdatavalid_q[m]_o	Out	1	Valid data indicator.
lmmiready_q[m]_o	Out	16	LMMI ready signal.
lmmirdata_q[m]_o	Out	16	Read data.
lmmierror_interrupt_q[m]_o	Out	1	When asserted, indicates LMMI transaction is accessing an invalid range.
Quad-Quad Signals			
refclk_p_q[m]_i	In	1	Quad reference clock, positive polarity.
refclk_n_q[m]_i	In	1	Quad reference clock, negative polarity.
PIPE Interface			
pipe_lane[n]_asyncpowerchangeack_i	In	1	Power change acknowledgment for lane X; PIPE signal AsyncPowerChangeAck (Sec 6.1, PIPE 4.4.1/4.3). PCIe: • For PIPE 4.4.1, use this signal when transitioning between two power states without PCLK. After PHY, asserts PhyStatus to acknowledge the power state change, the MAC responds by asserting pcs_laneX_asyncpowerchangeack until the MAC samples PhyStatus deasserted. • For PIPE 4.3, reserved; tie to 1'b0.
pipe_lane[n]_if_width_[p]_i	In	2	PIPE interface width for lane X; PIPE signal Width (Sec 6.1, PIPE 4.4.1/4.3). Controls the data width of pcs_txX_data and pcs_rxX_data buses.
pipe_lane[n]_m2p_messagebus_[p]_i	In	8	Mac-to-PHY message bus for lane X PIPE signal M2P_MessageBus (Sec 6.1, PIPE 4.4.1/4.3). • MAC-driven Message Bus interface. • PIPE 4.4.1 supports all message bus commands. For PIPE 4.3, the supported command mapping is as follows: • NOP. • Write-committed, non-posted. • write_ack.
pipe_lane[n]_powerdown_[p]_i	In	4	Power state for lane X; PIPE signal PowerDown[3:0] (Sec 6.1, PIPE 4.4.1/4.3). Controls the PHY power states. The power-state mapping is as follows (PCIe):

Port Name	I/O	Width	Description
			• PIPE 4.4.1 - PIPE 4.3 • P0: 4'b0000 - P0: 4'b0000 • P0s: 4'b0001 - P0s: 4'b0001 • P1: 4'b0010 - P1: 4'b0010 • P2: 4'b0011 - P2: 4'b0011 • P1.CPM: 4'b0100a - P1.CPM: 4'b0100b • P1.1: 4'b0100a - P1.1: 4'b0101 • P1.2: 4'b0100a - P1.2: 4'b1100 • P2.CPM: 4'b1101 - P2.CPM: 4'b1101 RxEIDetectDisable and TxCommonModeDisable signals are added in PIPE 4.4. The P1.CPM power-down encoding is used along with RxEIDetectDisable and TxCommonModeDisable to encode P1.1 and P1.2. For timing diagram showing P1.1/P1.2 entry and exit, see figure *P1.1/P1.2 Entry and Locally Initiated Exit Using PIPE 4.4.1 Interface*. For timing diagrams showing P1.2 entry and exit using a PIPE 4.4.1/4.3 interface, see figures *P1.2 Entry and Remotely Initiated Exit Using PIPE 4.3 Interface* and *P1.1/P1.2 Entry and Locally Initiated Exit Using PIPE 4.4.1 Interface*. for information about power state.
pipe_lane[n]_rate_[p]_i	In	2	Protocol or mode for lane X; PIPE signal Rate[1:0] (Sec 6.1, PIPE 4.4.1/4.3). Controls the link signaling rate. PCIe: 2'b00: 2.5 Gbps 2'b01: 5 Gbps 2'b10: 8 Gbps 2'b11: 16 Gbps
pipe_rx[n]_blk_align_ctl_i	In	1	RX block align control for lane X; PIPE signal BlockAlignControl (Sec 6.1, PIPE 4.4.1/4.3). Controls whether the PHY performs block alignment.
pipe_rx[n]_disable_i	In	1	RX disable control for lane X. This a side-band signal that a PIPE 4.4.1/4.2 controller needs to enter and exit P1.1 and P1.2 power states. When asserted, all RX lane circuitry (including RX Electrical Idle Exit Detection) for lane X is disabled. When integrating with a PIPE 4.3 controller, pcs_rxX_disable must be set to 0. For P1.1 and P1.2 entry and exit sequences, see the *PCIe L1 Substate Transitions* section.
pipe_rx[n]_eq_eval_i	In	1	RX equalization enable for lane X; PIPE signal RxEqEval (Sec 6.1, PIPE 4.4.1/4.3). When the MAC asserts this signal, the PHY starts evaluation of the far-end transmitter TX EQ settings.
pipe_rx[n]_eq_in_prog_i	In	1	RX equalization in progress for lane X; PIPE signal RxEqInProgress (Sec 6.1, PIPE 4.4.1/4.3).
pipe_rx[n]_polarity_i	In	1	RX polarity inversion enable for lane X; PIPE signal RxPolarity (Sec 6.1, PIPE 4.4.1/4.3). Requests the PHY to perform a polarity inversion on the received data.
pipe_rx[n]_standby_i	In	1	RX standby enable for lane X; PIPE signal RxStandby (Sec 6.1, PIPE 4.4.1/4.3). When asserted, the RX CDR for lane X is disabled.

Port Name	I/O	Width	Description
pipe_lane[n]_p2m_messagebus_[p]_o	Out	8	PHY-to-Mac Message Bus for lane X PIPE signal P2M_MessageBus (Sec 6.1, PIPE 4.4.1/4.3). • PHY-driven Message Bus interface. • PIPE 4.4.1 supports all message bus commands. For PIPE 4.3, the supported command mapping is as follows: • NOP. • Write-committed, non-posted. • write_ack.
pipe_lane[n]_phystatus_o	Out	1	PHY status for lane X; PIPE signal PhyStatus (Sec 6.1, PIPE 4.4.1/4.3). Acknowledgment signal for the following control inputs: • pcs_laneX_if_width[1:0] • pcs_laneX_powerdown[3:0] • pcs_laneX_rate[1:0] • pcs_rxX_eq_eval • pcs_txX_detectrx (when pcs_laneX_powerdown[3:0] = P1)
pipe_lane[n]_ref_clk_req_n_o	Out	1	Reference clock request for lane X. This is a side-band signal that the following PIPE controllers need to disable the reference clock input to the PHY: PCIe: 4.4.1/4.3/4.2 The reference clock input to the PHY can be disabled when pcs_laneX_clkreq_n = 1 and pcs_laneX_ref_clk_req_n = 1 (acknowledge signal) for all lanes connected to the PHY. For information about using this signal, see *Reference*.
pipe_rx[n]_align_detect_o	Out	2	RX ALIGN symbol detected for lane X; PIPE 4.3 signal AlignDetect (Sec 6.1, PIPE 4.3). Indicates receiver detection of an Align.
pipe_rx[n]_data_o	Out	64	RX data for lane X; PIPE signal RxData[31:0] (Sec 6.1, PIPE 4.4.1/4.3). RX parallel data. The valid data bits for pcs_rx_data[39:0] are as follows: pcs_laneX_encdec_bypass - pcs_laneX_if_width[1:0] - Valid bits of pcs_rxX_data[39:0]: • 0 - 00 - [7:0] • 0 - 01 - [15:0] • 0 - 10 - [31:0] • 1 - 00 - [9:0] • 1 - 01 - • 1 - 10 - [39:0]
pipe_rx[n]_datak_[p]_o	Out	NL	RX data control for lane X; PIPE signal RxDataK[3:0] (Sec 6.1, PIPE 4.4.1/4.3). Data/control for the symbols of received data.
pipe_rx[n]_datavalid_o	Out	2	RX data valid; PIPE signal RxDataValid (Sec 6.1, PIPE 4.4.1/4.3). RX data valid to throttle RX data for non-8b10b encoding modes.
pipe_rx[n]_elecidle_o	Out	1	RX electrical idle detection for lane X; PIPE signal RxElecIdle (Sec 6.1, PIPE 4.4.1/4.3). Indicates receiver detection of an electrical idle.
pipe_rx[n]_eq_dir_change_o	Out	6	Direction change commands for link partner's TX equalization coefficients for lane X; PIPE signal LinkEvaluationFeedbackDirectionChange[5:0] (Sec 6.1, PIPE 4.4.1/4.3).

Port Name	I/O	Width	Description
			Provides link equalization evaluation feedback in the direction change format.
pipe_rx[n]_eq_fig_merit_o	Out	8	Figure of Merit for equalization settings of lane X; PIPE signal *LinkEvaluationFeedbackFigureMerit[7:0]* (Sec 6.1, PIPE 4.4.1/4.3). Provides the PHY link equalization evaluation Figure of Merit value.
pipe_rx[n]_eq_fs_o	Out	6	Equalization FS value for Lane n.
pipe_rx[n]_eq_lf_o	Out	6	Equalization LF value for Lane n.
pipe_rx[n]_eq_preset_coeff_o	Out	18	Equalization Preset Coefficient for Lane n.
pipe_rx[n]_eq_preset_coeff_vld_o	Out	1	When asserted, pipe_rx[n]_eq_preset_coeff_o[17:0] is valid.
pipe_rx[n]_standby_status_o	Out	1	RX standby status for lane X; PIPE signal RxStandbyStatus (Sec 6.1, PIPE 4.4.1/4.3). When asserted, indicates that the RX has entered Standby state in response to pcs_rxX_standby assertion.
pipe_rx[n]_startblock_o	Out	2	RX start block indicator for lane X; PIPE signal RxStartBlock (Sec 6.1, PIPE 4.4.1/4.3). Enables PHY to provide MAC with the starting byte for a 128b block.
pipe_rx[n]_status_[p]_o	Out	3	Receive status and error codes for lane X; PIPE signal RxStatus[2:0] (Sec 6.1, PIPE 4.4.1/4.3). Encodes receiver status and error codes for the received datastream when receiving data.
pipe_rx[n]_syncheader_[p]_o	Out	2	RX SYNC header for lane X; PIPE signal RxSyncHeader[3:0] (Sec 6.1, PIPE 4.4.1/4.3). Provides the SYNC header for the PHY to use in the next block. The pcs_rxX_syncheader[1:0] bits are valid for PCIe.
pipe_rx[n]_valid_o	Out	2	RX valid for lane X; PIPE signal RxValid (Sec 6.1, PIPE 4.4.1/4.3). Indicates symbol lock and valid data on pcs_rxX_data and pcs_rxX_datak.
pipe_tx[n]_compliance_i	In	2	TX compliance pattern signal; PIPE signal TxCompliance (Sec 6.1, PIPE 4.4.1/4.3). Sets the transmitter running disparity to negative when transmitting the PCIecompliance pattern. This signal is sampled by TxDataValid and is only required for PHYs that support PCIe operation.
pipe_tx[n]_data_i	In	64	TX data for lane X; PIPE signal TxData[31:0] (Sec 6.1, PIPE 4.4.1/4.3). TX parallel data. The valid data bits for pcs_tx_data[39:0] are as follows: pcs_laneX_encdec_bypass - pcs_laneX_if_width[1:0] - Valid bits of pcs_txX_data[39:0]: • 0 - 00 - [7:0] • 0 - 01 - [15:0] • 0 - 10 - [31:0] • 1 - 00 - [9:0] • 1 - 01 - • 1 - 10 - [39:0]
pipe_tx[n]_datak_[p]_i	In	NL	TX data control for lane X; PIPE signal TxDataK[3:0] (Sec 6.1, PIPE 4.4.1/4.3). Data and control for the symbols of transmitted data.
pipe_tx[n]_datavalid_i	In	2	TX data valid; PIPE signal TxDataValid (Sec 6.1, PIPE 4.4.1/4.3).

Port Name	I/O	Width	Description
			TX data valid to throttle TX data for non-8b10b encoding modes.
pipe_tx[n]_deemph_i	In	18	TX de-emphasis control for lane X; PIPE signal TxDeemph. Controls transmitter de-emphasis and equalization settings during link operation.
pipe_tx[n]_detectrx_i	In	2	Rx detection request for lane X; PIPE signal TxDetectRX (Sec 6.1, PIPE 4.4.1/4.3). Used to tell the PHY to begin a receiver detection operation.
pipe_tx[n]_disable_i	In	1	TX disable control for lane X. This a side-band signal that a PIPE 4.4.1/4.2 controller needs to enter and exit P1.1 and P1.2 power states. When asserted, all TX lane circuitry (including TX Electrical Idle Exit Detection) for lane X is disabled. When integrating with a PIPE 4.3 controller, pcs_txX_disable must be set to 0. For P1.1 and P1.2 entry and exit sequences, see the *PCIe L1 Substate Transitions* section.
pipe_tx[n]_elecidle_i	In	2	TX electrical idle detection for lane X; PIPE signal TxElecIdle (Sec 6.1, PIPE 4.4.1/4.3). Forces TX output to electrical idle when asserted except in loopback.
pipe_tx[n]_eq_preset_i	In	5	TX equalization preset selection for lane X; PIPE signal TxEqPreset. Selects the transmitter equalization preset during link equalization.
pipe_tx[n]_eq_preset_coeff_req_i	In	1	TX equalization preset coefficient request for lane X; PIPE signal TxEqPresetCoeffReq. Requests the PHY to provide equalization preset coefficients.
pipe_tx[n]_margin_i	In	3	This field selects transmitter voltage level.
pipe_tx[n]_startblock_i	In	2	TX start block indicator for lane X; PIPE signal TxStartBlock (Sec 6.1, PIPE 4.4.1/4.3). Enables PHY to provide MAC with the starting byte for a 128b block.
pipe_tx[n]_syncheader_[p]_i	In	2	TX SYNC header for lane X; PIPE signal TxSyncHeader[3:0] (Sec 6.1, PIPE 4.4.1/4.3). Provides the SYNC header for the PHY to use in the next block. The pcs_rxX_syncheader[1:0] bits are valid for PCIe.

Notes:

1. [n] indicates lane/channel number.
2. [m] indicates Quad number.
3. [r] indicates PLL number, where PLL0 – PLLA and PLL1 – PLLB.
4. [p] indicates the number of port 0 or 1.
5. NL means Number of Lanes enabled in the Quad.

2.2.1. MPP Reset Scheme

Table 2.13. MPP Reset Scheme

Reset Request Input Port	PMA Rx Channel	PMA Tx Channel	PCS Rx Channel	PCS Tx Channel	PCIE Core	PCIe Core User Interface	DMA
PCIe PCS (Duplex)							
pin_perst_n_q*_i[1, 3]	Yes	Yes	Yes	Yes	Yes	Yes	Yes
tx_pcie_pma_rst_n_q*_i[3:0][2, 4]	Yes	Yes	Yes	Yes	Yes	Yes	Yes

soft_core_rst_n_q*_i[5]	No	No	No	No	Yes	Yes	Yes
MP PCS (Simple/duplex)							
rx_pma_rst_n_q*_i[3:0][4]	Yes	No	No	No	—	—	—
tx_pcie_pma_rst_n_q*_i[3:0][4]	No	Yes	No	No	—	—	—
rx_pcs_rst_n_q*_i[3:0][4]	No	No	Yes	No	—	—	—
tx_pcs_rst_n_q*_i[3:0][4]	No	No	No	Yes	—	—	—

Notes:

1. Global reset which reset all channels for PCIe PCS only.
2. Reset for each channel. Support pipe_direct mode without reseting PCIe core, user interface, and DMA.
3. pin_perst_n only support when PCIe hard controller is enabled.
4. When tx/rx_pma_rst_n are asserted, the TX/RX PCS must also enter reset.
5. Internal use mode only.

2.2.2. Data Bus Mapping

2.2.2.1. 8b10b PCS

Table 2.14. Bit Mapping of TX Data Bus with MPPCS 2-Byte Internal Data Width

The data is transmitted per the sequence in this table in the transmission order as follows:

- First: pma_tx{n}_data[7:0]
- Last: pma_tx{n}_data[159:153]

IP Port Name n = 0,1,2….7	8-bit Mode	10-bit Mode	2-Byte	4-Byte	8-Byte
pma_tx{n}_data[7:0]	din[0][7:0]	din[0][7:0]	Byte 0	Byte 0	Byte 0
pma_tx{n}_data[8]	control[0]	din[0][8]			
pma_tx{n}_data[9]	cordisp[0]	din[0][9]			
pma_tx{n}_data[10]	frcdisp[0]	frcdisp[0]			
pma_tx{n}_data[11]	dispval[0]	dispval[0]			
pma_tx{n}_data[12]	frcdata[0]	frcdata[0]			
pma_tx{n}_data[19:13]	—	—			
pma_tx{n}_data[27:20]	din[1][7:0]	din[1][7:0]	Byte 1	Byte 1	Byte 1
pma_tx{n}_data[28]	control[1]	din[1][8]			
pma_tx{n}_data[29]	cordisp[1]	din[1][9]			
pma_tx{n}_data[30]	frcdisp[1]	frcdisp[1]			
pma_tx{n}_data[31]	dispval[1]	dispval[1]			
pma_tx{n}_data[32]	frcdata[1]	frcdata[1]			
pma_tx{n}_data[39:33]	—	—			
pma_tx{n}_data[47:40]	din[2][7:0]	din[2][7:0]	—	Byte 2	Byte 2
pma_tx{n}_data[48]	control[2]	din[2][8]			
pma_tx{n}_data[49]	cordisp[2]	din[2][9]			
pma_tx{n}_data[50]	frcdisp[2]	frcdisp[2]			
pma_tx{n}_data[51]	dispval[2]	dispval[2]			
pma_tx{n}_data[52]	frcdata[2]	frcdata[2]			
pma_tx{n}_data[59:53]	—	—			
pma_tx{n}_data[67:60]	din[3][7:0]	din[3][7:0]		Byte 3	Byte 3
pma_tx{n}_data[68]	control[3]	din[3][8]			
pma_tx{n}_data[69]	cordisp[3]	din[3][9]			
pma_tx{n}_data[70]	frcdisp[3]	frcdisp[3]			
pma_tx{n}_data[71]	dispval[3]	dispval[3]			
pma_tx{n}_data[72]	frcdata[3]	frcdata[3]			
pma_tx{n}_data[79:73]	—	—			

IP Port Name n = 0,1,2….7	8-bit Mode	10-bit Mode	2-Byte	4-Byte	8-Byte
pma_tx{n}_data[87:80]	din[4][7:0]	din[4][7:0]		—	Byte 4
pma_tx{n}_data[88]	control[4]	din[4][8]			
pma_tx{n}_data[89]	cordisp[4]	din[4][9]			
pma_tx{n}_data[90]	frcdisp[4]	frcdisp[4]			
pma_tx{n}_data[91]	dispval[4]	dispval[4]			
pma_tx{n}_data[92]	frcdata[4]	frcdata[4]			
pma_tx{n}_data[99:93]	—	—			
pma_tx{n}_data[107:100]	din[5][7:0]	din[5][7:0]			Byte 5
pma_tx{n}_data[108]	control[5]	din[5][8]			
pma_tx{n}_data[109]	cordisp[5]	din[5][9]			
pma_tx{n}_data[110]	frcdisp[5]	frcdisp[5]			
pma_tx{n}_data[111]	dispval[5]	dispval[5]			
pma_tx{n}_data[112]	frcdata[5]	frcdata[5]			
pma_tx{n}_data[119:113]	—	—			
pma_tx{n}_data[127:120]	din[6][7:0]	din[6][7:0]			Byte 6
pma_tx{n}_data[128]	control[6]	din[6][8]			
pma_tx{n}_data[129]	cordisp[6]	din[6][9]			
pma_tx{n}_data[130]	frcdisp[6]	frcdisp[6]			
pma_tx{n}_data[131]	dispval[6]	dispval[6]			
pma_tx{n}_data[132]	frcdata[6]	frcdata[6]			
pma_tx{n}_data[139:133]	—	—			
pma_tx{n}_data[147:140]	din[7][7:0]	din[7][7:0]			Byte 7
pma_tx{n}_data[148]	control[7]	din[7][8]			
pma_tx{n}_data[149]	cordisp[7]	din[7][9]			
pma_tx{n}_data[150]	frcdisp[7]	frcdisp[7]			
pma_tx{n}_data[151]	dispval[7]	dispval[7]			
pma_tx{n}_data[152]	frcdata[7]	frcdata[7]			
pma_tx{n}_data[159:153]	—	—			

Table 2.15. Bit Mapping of RX Data Bus with MPPCS 2-Byte Internal Data Width

The data is transmitted per the sequence in this table in the transmission order as follows:

- First: pma_rx{n}_data[7:0]
- Last: pma_rx{n}_data[159:154]

IP Port Name n = 0,1,2….7	8-bit Mode	10-bit Mode	2-Byte	4-Byte	8-Byte
pma_rx{n}_data[7:0]	din[0][7:0]	din[0][7:0]	Byte 0	Byte 0	Byte 0
pma_rx{n}_data[8]	control[0]	din[0][8]			
pma_rx{n}_data[9]	disp[0]	din[0][9]			
pma_rx{n}_data[10]	disperr[0]	disperr[0]			
pma_rx{n}_data[11]	errcode[0]	errcode[0]			
pma_rx{n}_data[12]	ins[0]	ins[0]			
pma_rx{n}_data[13]	del[0]	del[0]			
pma_rx{n}_data[19:14]	—	—			
pma_rx{n}_data[27:20]	din[1][7:0]	din[1][7:0]	Byte 1	Byte 1	Byte 1
pma_rx{n}_data[28]	control[1]	din[1][8]			
pma_rx{n}_data[29]	disp[1]	din[1][9]			
pma_rx{n}_data[30]	disperr[1]	disperr[1]			

IP Port Name n = 0,1,2….7	8-bit Mode	10-bit Mode	2-Byte	4-Byte	8-Byte
pma_rx{n}_data[31]	errcode[1]	errcode[1]			
pma_rx{n}_data[32]	ins[1]	ins[1]			
pma_rx{n}_data[33]	del[1]	del[1]			
pma_rx{n}_data[39:34]	—	—			
pma_rx{n}_data[47:40]	din[2][7:0]	din[2][7:0]	—	Byte 2	Byte 2
pma_rx{n}_data[48]	control[2]	din[2][8]			
pma_rx{n}_data[49]	disp[2]	din[2][9]			
pma_rx{n}_data[50]	disperr[2]	disperr[2]			
pma_rx{n}_data[51]	errcode[2]	errcode[2]			
pma_rx{n}_data[52]	ins[2]	ins[2]			
pma_rx{n}_data[53]	del[2]	del[2]			
pma_rx{n}_data[59:54]	—	—			
pma_rx{n}_data[67:60]	din[3][7:0]	din[3][7:0]		Byte 3	Byte 3
pma_rx{n}_data[68]	control[3]	din[3][8]			
pma_rx{n}_data[69]	disp[3]	din[3][9]			
pma_rx{n}_data[70]	disperr[3]	disperr[3]			
pma_rx{n}_data[71]	errcode[3]	errcode[3]			
pma_rx{n}_data[72]	ins[3]	ins[3]			
pma_rx{n}_data[73]	del[3]	del[3]			
pma_rx{n}_data[79:74]	—	—			
pma_rx{n}_data[87:80]	din[4][7:0]	din[4][7:0]		—	Byte 4
pma_rx{n}_data[88]	control[4]	din[4][8]			
pma_rx{n}_data[89]	disp[4]	din[4][9]			
pma_rx{n}_data[90]	disperr[4]	disperr[4]			
pma_rx{n}_data[91]	errcode[4]	errcode[4]			
pma_rx{n}_data[92]	ins[4]	ins[4]			
pma_rx{n}_data[93]	del[4]	del[4]			
pma_rx{n}_data[99:94]	—	—			
pma_rx{n}_data[107:100]	din[5][7:0]	din[5][7:0]			Byte 5
pma_rx{n}_data[108]	control[5]	din[5][8]			
pma_rx{n}_data[109]	disp[5]	din[5][9]			
pma_rx{n}_data[110]	disperr[5]	disperr[5]			
pma_rx{n}_data[111]	errcode[5]	errcode[5]			
pma_rx{n}_data[112]	ins[5]	ins[5]			
pma_rx{n}_data[113]	del[5]	del[5]			
pma_rx{n}_data[119:114]	—	—			
pma_rx{n}_data[127:120]	din[6][7:0]	din[6][7:0]			Byte 6
pma_rx{n}_data[128]	control[6]	din[6][8]			
pma_rx{n}_data[129]	disp[6]	din[6][9]			
pma_rx{n}_data[130]	disperr[6]	disperr[6]			
pma_rx{n}_data[131]	errcode[6]	errcode[6]			
pma_rx{n}_data[132]	ins[6]	ins[6]			
pma_rx{n}_data[133]	del[6]	del[6]			
pma_rx{n}_data[139:134]	—	—			
pma_rx{n}_data[147:140]	din[7][7:0]	din[7][7:0]			Byte 7
pma_rx{n}_data[148]	control[7]	din[7][8]			
pma_rx{n}_data[149]	disp[7]	din[7][9]			

IP Port Name n = 0,1,2….7	8-bit Mode	10-bit Mode	2-Byte	4-Byte	8-Byte
pma_rx{n}_data[150]	disperr[7]	disperr[7]			
pma_rx{n}_data[151]	errcode[7]	errcode[7]			
pma_rx{n}_data[152]	ins[7]	ins[7]			
pma_rx{n}_data[153]	del[7]	del[7]			
pma_rx{n}_data[159:154]	—	—			

Table 2.16. Bit Mapping of TX Data Bus with MPPCS 1-Byte Internal Data Width

The data in this table is transmitted in sequence in the transmission order as follows:

- First: pma_tx{n}_data[7:0]
- Last: pma_tx{n}_data[159:153]

IP Port Name n = 0,1,2….7	8-bit Mode	10-bit Mode	1-Byte	2-Byte	4-Byte
pma_tx{n}_data[7:0]	din[0][7:0]	din[0][7:0]	Byte 0	Byte 0	Byte 0
pma_tx{n}_data[8]	control[0]	din[0][8]			
pma_tx{n}_data[9]	cordisp[0]	din[0][9]			
pma_tx{n}_data[10]	frcdisp[0]	frcdisp[0]			
pma_tx{n}_data[11]	dispval[0]	dispval[0]			
pma_tx{n}_data[12]	frcdata[0]	frcdata[0]			
pma_tx{n}_data[19:13]	—	—			
pma_tx{n}_data[27:20]	—	—	—	—	—
pma_tx{n}_data[28]	—	—			
pma_tx{n}_data[29]	—	—			
pma_tx{n}_data[30]	—	—			
pma_tx{n}_data[31]	—	—			
pma_tx{n}_data[32]	—	—			
pma_tx{n}_data[39:33]	—	—			
pma_tx{n}_data[47:40]	din[1][7:0]	din[1][7:0]		Byte 1	Byte 1
pma_tx{n}_data[48]	control[1]	din[1][8]			
pma_tx{n}_data[49]	cordisp[1]	din[1][9]			
pma_tx{n}_data[50]	frcdisp[1]	frcdisp[1]			
pma_tx{n}_data[51]	dispval[1]	dispval[1]			
pma_tx{n}_data[52]	frcdata[1]	frcdata[1]			
pma_tx{n}_data[59:53]	—	—			
pma_tx{n}_data[67:60]	—	—		—	—
pma_tx{n}_data[68]	—	—			
pma_tx{n}_data[69]	—	—			
pma_tx{n}_data[70]	—	—			
pma_tx{n}_data[71]	—	—			
pma_tx{n}_data[72]	—	—			
pma_tx{n}_data[79:73]	—	—			
pma_tx{n}_data[87:80]	din[2][7:0]	din[2][7:0]			Byte 2
pma_tx{n}_data[88]	control[2]	din[2][8]			
pma_tx{n}_data[89]	cordisp[2]	din[2][9]			
pma_tx{n}_data[90]	frcdisp[2]	frcdisp[2]			
pma_tx{n}_data[91]	dispval[2]	dispval[2]			
pma_tx{n}_data[92]	frcdata[2]	frcdata[2]			
pma_tx{n}_data[99:93]	—	—			

IP Port Name n = 0,1,2….7	8-bit Mode	10-bit Mode	1-Byte	2-Byte	4-Byte
pma_tx{n}_data[107:100]	—	—			—
pma_tx{n}_data[108]	—	—			
pma_tx{n}_data[109]	—	—			
pma_tx{n}_data[110]	—	—			
pma_tx{n}_data[111]	—	—			
pma_tx{n}_data[112]	—	—			
pma_tx{n}_data[119:113]	—	—			
pma_tx{n}_data[127:120]	din[3][7:0]	din[3][7:0]			Byte 3
pma_tx{n}_data[128]	control[3]	din[3][8]			
pma_tx{n}_data[129]	cordisp[3]	din[3][9]			
pma_tx{n}_data[130]	frcdisp[3]	frcdisp[3]			
pma_tx{n}_data[131]	dispval[3]	dispval[3]			
pma_tx{n}_data[132]	frcdata[3]	frcdata[3]			
pma_tx{n}_data[139:133]	—	—			
pma_tx{n}_data[147:140]	—	—			—
pma_tx{n}_data[148]	—	—			
pma_tx{n}_data[149]	—	—			
pma_tx{n}_data[150]	—	—			
pma_tx{n}_data[151]	—	—			
pma_tx{n}_data[152]	—	—			
pma_tx{n}_data[159:153]	—	—			

Table 2.17. Bit Mapping of RX Data Bus with MPPCS 1-Byte Internal Data Width

The data in this table is transmitted in sequence in the transmission order as follows:

- First: pma_rx{n}_data[7:0]
- Last: pma_rx{n}_data[159:154]

IP Port Name n = 0,1,2….7	8-bit Mode	10-bit Mode	2-Byte	4-Byte	8-Byte
pma_rx{n}_data[7:0]	din[0][7:0]	din[0][7:0]	Byte 0	Byte 0	Byte 0
pma_rx{n}_data[8]	control[0]	din[0][8]			
pma_rx{n}_data[9]	disp[0]	din[0][9]			
pma_rx{n}_data[10]	disperr[0]	disperr[0]			
pma_rx{n}_data[11]	errcode[0]	errcode[0]			
pma_rx{n}_data[12]	ins[0]	ins[0]			
pma_rx{n}_data[13]	del[0]	del[0]			
pma_rx{n}_data[19:14]	—	—			
pma_rx{n}_data[27:20]	—	—	—	—	—
pma_rx{n}_data[28]	—	—			
pma_rx{n}_data[29]	—	—			
pma_rx{n}_data[30]	—	—			
pma_rx{n}_data[31]	—	—			
pma_rx{n}_data[32]	—	—			
pma_rx{n}_data[33]	—	—			
pma_rx{n}_data[39:34]	—	—			
pma_rx{n}_data[47:40]	din[1][7:0]	din[1][7:0]		Byte 1	Byte 1
pma_rx{n}_data[48]	control[1]	din[1][8]			
pma_rx{n}_data[49]	disp[1]	din[1][9]			

IP Port Name n = 0,1,2….7	8-bit Mode	10-bit Mode	2-Byte	4-Byte	8-Byte
pma_rx{n}_data[50]	disperr[1]	disperr[1]			
pma_rx{n}_data[51]	errcode[1]	errcode[1]			
pma_rx{n}_data[52]	ins[1]	ins[1]			
pma_rx{n}_data[53]	del[1]	del[1]			
pma_rx{n}_data[59:54]	—	—			
pma_rx{n}_data[67:60]	—	—		—	—
pma_rx{n}_data[68]	—	—			
pma_rx{n}_data[69]	—	—			
pma_rx{n}_data[70]	—	—			
pma_rx{n}_data[71]	—	—			
pma_rx{n}_data[72]	—	—			
pma_rx{n}_data[73]	—	—			
pma_rx{n}_data[79:74]	—	—			
pma_rx{n}_data[87:80]	din[2][7:0]	din[2][7:0]			Byte 2
pma_rx{n}_data[88]	control[2]	din[2][8]			
pma_rx{n}_data[89]	disp[2]	din[2][9]			
pma_rx{n}_data[90]	disperr[2]	disperr[2]			
pma_rx{n}_data[91]	errcode[2]	errcode[2]			
pma_rx{n}_data[92]	ins[2]	ins[2]			
pma_rx{n}_data[93]	del[2]	del[2]			
pma_rx{n}_data[99:94]	—	—			
pma_rx{n}_data[107:100]	—	—			—
pma_rx{n}_data[108]	—	—			
pma_rx{n}_data[109]	—	—			
pma_rx{n}_data[110]	—	—			
pma_rx{n}_data[111]	—	—			
pma_rx{n}_data[112]	—	—			
pma_rx{n}_data[113]	—	—			
pma_rx{n}_data[119:114]	—	—			
pma_rx{n}_data[127:120]	din[3][7:0]	din[3][7:0]			Byte 3
pma_rx{n}_data[128]	control[3]	din[3][8]			
pma_rx{n}_data[129]	disp[3]	din[3][9]			
pma_rx{n}_data[130]	disperr[3]	disperr[3]			
pma_rx{n}_data[131]	errcode[3]	errcode[3]			
pma_rx{n}_data[132]	ins[3]	ins[3]			
pma_rx{n}_data[133]	del[3]	del[3]			
pma_rx{n}_data[139:134]	—	—			
pma_rx{n}_data[147:140]	—	—			—
pma_rx{n}_data[148]	—	—			
pma_rx{n}_data[149]	—	—			
pma_rx{n}_data[150]	—	—			
pma_rx{n}_data[151]	—	—			
pma_rx{n}_data[152]	—	—			
pma_rx{n}_data[153]	—	—			
pma_rx{n}_data[159:154]	—	—			

2.2.2.2. 64b66b PCS

Table 2.18. Bit Mapping of TX Data Bus for 64b66b Mode

The data is transmitted per the sequence in this table in the transmission order as follows:

- First: pma_tx{n}_data[1:0]
- Last: pma_tx{n}_data[159:153]

IP Port Name n = 0,1,2….7	xgmii Mode	66b Mode	1 Word Mode	2 Word Mode
pma_tx{n}_data[1:0]	xgmii_txd[0][1:0]	din_66b[0][3:2]	Word 0	Word 0
pma_tx{n}_data[7:2]	xgmii_txd[0][7:2]	din_66b[0][9:4]		
pma_tx{n}_data[15:8]	xgmii_txd[1][7:0]	din_66b[0][17:12]		
pma_tx{n}_data[23:16]	xgmii_txd[2][7:0]	din_66b[0][25:20]		
pma_tx{n}_data[31:24]	xgmii_txd[3][7:0]	din_66b[0][33:28]		
pma_tx{n}_data[39:32]	xgmii_txd[4][7:0]	din_66b[0][41:36]		
pma_tx{n}_data[47:40]	xgmii_txd[5][7:0]	din_66b[0][49:44]		
pma_tx{n}_data[55:48]	xgmii_txd[6][7:0]	din_66b[0][57:52]		
pma_tx{n}_data[63:56]	xgmii_txd[7][7:0]	din_66b[0][65:60]		
pma_tx{n}_data[64]	xgmii_txc[0]	din_66b[0][0]		
pma_tx{n}_data[65]	xgmii_txc[1]	din_66b[0][1]		
pma_tx{n}_data[66]	xgmii_txc[2]	—		
pma_tx{n}_data[67]	xgmii_txc[3]	—		
pma_tx{n}_data[68]	xgmii_txc[4]	—		
pma_tx{n}_data[69]	xgmii_txc[5]	—		
pma_tx{n}_data[70]	xgmii_txc[6]	—		
pma_tx{n}_data[71]	xgmii_txc[7]	—		
pma_tx{n}_data[72]	xgmii_frcpkt[0]	—		
pma_tx[n]_data[79:73]	—	—		
pma_tx{n}_data[81:80]	xgmii_txd[8][1:0]	din_66b[1][3:2]	—	Word 1
pma_tx{n}_data[87:82]	xgmii_txd[8][7:2]	din_66b[1][9:4]		
pma_tx{n}_data[95:88]	xgmii_txd[9][7:0]	din_66b[1][17:12]		
pma_tx{n}_data[103:96]	xgmii_txd[10][7:0]	din_66b[1][25:20]		
pma_tx{n}_data[111:104]	xgmii_txd[11][7:0]	din_66b[1][33:28]		
pma_tx{n}_data[119:112]	xgmii_txd[12][7:0]	din_66b[1][41:36]		
pma_tx{n}_data[127:120]	xgmii_txd[13][7:0]	din_66b[1][49:44]		
pma_tx{n}_data[135:128]	xgmii_txd[14][7:0]	din_66b[1][57:52]		
pma_tx{n}_data[143:136]	xgmii_txd[15][7:0]	din_66b[1][65:60]		
pma_tx{n}_data[144]	xgmii_txc[8]	din_66b[1][0]		
pma_tx{n}_data[145]	xgmii_txc[9]	din_66b[1][1]		
pma_tx{n}_data[146]	xgmii_txc[10]	—		
pma_tx{n}_data[147]	xgmii_txc[11]	—		
pma_tx{n}_data[148]	xgmii_txc[12]	—		
pma_tx{n}_data[149]	xgmii_txc[13]	—		
pma_tx{n}_data[150]	xgmii_txc[14]	—		
pma_tx{n}_data[151]	xgmii_txc[15]	—		
pma_tx{n}_data[152]	xgmii_frcpkt[1]	—		
pma_tx{n}_data[159:153]	—	—		
tx[n]_am[0]	tx_am[0]	tx_am[0]	Word 0	Word 0
tx[n]_en[0]	tx_en[0]	tx_en[0]		
tx[n]_am[1]	tx_am[1]	tx_am[1]	—	Word 1
tx[n]_en[1]	tx_en[1]	tx_en[1]		

Table 2.19. Bit Mapping of RX Data Bus for 64b66b Mode

The data in this table is transmitted per the sequence in this table in the transmission order as follows:

- First: pma_rx{n}_data[1:0]
- Last: pma_rx[n]_data[159:158]

IP Port Name n = 0,1,2….7	xgmii Mode	66b Mode	1 Word Mode	2 Word Mode
pma_rx{n}_data[1:0]	xgmii_rxd[0][1:0]	dout_66b[0][3:2]	Word 0	Word 0
pma_rx{n}_data[7:2]	xgmii_rxd[0][7:2]	dout_66b[0][9:4]		
pma_rx{n}_data[15:8]	xgmii_rxd[1][7:0]	dout_66b[0][17:12]		
pma_rx{n}_data[23:16]	xgmii_rxd[2][7:0]	dout_66b[0][25:20]		
pma_rx{n}_data[31:24]	xgmii_rxd[3][7:0]	dout_66b[0][33:28]		
pma_rx{n}_data[39:32]	xgmii_rxd[4][7:0]	dout_66b[0][41:36]		
pma_rx{n}_data[47:40]	xgmii_rxd[5][7:0]	dout_66b[0][49:44]		
pma_rx{n}_data[55:48]	xgmii_rxd[6][7:0]	dout_66b[0][57:52]		
pma_rx{n}_data[63:56]	xgmii_rxd[7][7:0]	dout_66b[0][65:60]		
pma_rx{n}_data[64]	xgmii_rxc[0]	dout_66b[0][0]		
pma_rx{n}_data[65]	xgmii_rxc[1]	dout_66b[0][1]		
pma_rx{n}_data[66]	xgmii_rxc[2]	—		
pma_rx{n}_data[67]	xgmii_rxc[3]	—		
pma_rx{n}_data[68]	xgmii_rxc[4]	—		
pma_rx{n}_data[69]	xgmii_rxc[5]	—		
pma_rx{n}_data[70]	xgmii_rxc[6]	—		
pma_rx{n}_data[71]	xgmii_rxc[7]	—		
pma_rx{n}_data[72]	xgmii_ctc_ins[0]	—		
pma_rx{n}_data[73]	xgmii_ctc_ins[1]	—		
pma_rx{n}_data[74]	xgmii_ctc_del[0]	—		
pma_rx{n}_data[75]	xgmii_ctc_del[1]	—		
pma_rx[n]_data[76]	xgmii_rx_sob[0]	xgmii_rx_sob[0]		
pma_rx[n]_data[77]	xgmii_rx_cw_bad[0]	xgmii_rx_cw_bad[0]		
pma_rx[n]_data[79:78]	—	—		
pma_rx{n}_data[81:80]	xgmii_rxd[8][1:0]	dout_66b[1][3:2]	—	Word 1
pma_rx{n}_data[87:82]	xgmii_rxd[8][7:2]	dout_66b[1][9:4]		
pma_rx{n}_data[95:88]	xgmii_rxd[9][7:0]	dout_66b[1][17:12]		
pma_rx{n}_data[103:96]	xgmii_rxd[10][7:0]	dout_66b[1][25:20]		
pma_rx{n}_data[111:104]	xgmii_rxd[11][7:0]	dout_66b[1][33:28]		
pma_rx{n}_data[119:112]	xgmii_rxd[12][7:0]	dout_66b[1][41:36]		
pma_rx{n}_data[127:120]	xgmii_rxd[13][7:0]	dout_66b[1][49:44]		
pma_rx{n}_data[135:128]	xgmii_rxd[14][7:0]	dout_66b[1][57:52]		
pma_rx{n}_data[143:136]	xgmii_rxd[15][7:0]	dout_66b[1][65:60]		
pma_rx{n}_data[144]	xgmii_rxc[8]	dout_66b[1][0]		
pma_rx{n}_data[145]	xgmii_rxc[9]	dout_66b[1][1]		
pma_rx{n}_data[146]	xgmii_rxc[10]	—		
pma_rx{n}_data[147]	xgmii_rxc[11]	—		
pma_rx{n}_data[148]	xgmii_rxc[12]	—		
pma_rx{n}_data[149]	xgmii_rxc[13]	—		
pma_rx{n}_data[150]	xgmii_rxc[14]	—		
pma_rx{n}_data[151]	xgmii_rxc[15]	—		
pma_rx{n}_data[152]	xgmii_ctc_ins[2]	—		

IP Port Name n = 0,1,2….7	xgmii Mode	66b Mode	1 Word Mode	2 Word Mode
pma_rx{n}_data[153]	xgmii_ctc_ins[3]	—		
pma_rx{n}_data[154]	xgmii_ctc_del[2]	—		
pma_rx{n}_data[155]	xgmii_ctc_del[3]	—		
pma_rx[n]_data[156]	xgmii_rx_sob[1]	xgmii_rx_sob[1]		
pma_rx[n]_data[157]	xgmii_rx_cw_bad[1]	xgmii_rx_cw_bad[1]		
pma_rx[n]_data[159:158]	—	—		
rx[n]_am[0]	rx_am[0]	rx_am[0]	Word 0	Word 0
rx[n]_am[1]	rx_am[1]	rx_am[1]	—	Word 1

2.2.2.3. PCS Bypass

Table 2.20. TX Path Data Bus Mapping

The data in this table is transmitted in sequence in the transmission order as follows:

- First: txdata[7:0]
- Last: txdata[159:152]

IP Port Name	8-bit Mode	10-bit Mode	16-bit Mode	20-bit Mode	32-bit Mode	40-bit Mode	1 Word Mode	2 Word Mode	4 Word Mode
txdata[7:0]	din[0][7:0]	din[0][7:0]	din[0][7:0]	din[0][7:0]	din[0][7:0]	din[0][7:0]	Word 0	Word 0	Word 0
txdata[9:8]	—	din[0][9:8]	din[0][9:8]	din[0][9:8]	din[0][9:8]	din[0][9:8]			
txdata[15:10]	—	—	din[0][15:10]	din[0][15:10]	din[0][15:10]	din[0][15:10]			
txdata[19:16]	—	—	—	din[0][19:16]	din[0][19:16]	din[0][19:16]			
txdata[31:20]	—	—	—	—	din[0][31:20]	din[0][31:20]			
txdata[39:32]	—	—	—	—	—	din[0][39:32]			
txdata[47:40]	din[1][7:0]	din[1][7:0]	din[1][7:0]	din[1][7:0]	din[1][7:0]	din[1][7:0]	—	Word 1	Word 1
txdata[49:48]	—	din[1][9:8]	din[1][9:8]	din[1][9:8]	din[1][9:8]	din[1][9:8]			
txdata[55:50]	—	—	din[1][15:10]	din[1][15:10]	din[1][15:10]	din[1][15:10]			
txdata[59:56]	—	—	—	din[1][19:16]	din[1][19:16]	din[1][19:16]			
txdata[71:60]	—	—	—	—	din[1][31:20]	din[1][31:20]			
txdata[79:72]	—	—	—	—	—	din[1][39:32]			
txdata[87:80]	din[2][7:0]	din[2][7:0]	din[2][7:0]	din[2][7:0]	din[2][7:0]	din[2][7:0]	—	—	Word 2
txdata[89:88]	—	din[2][9:8]	din[2][9:8]	din[2][9:8]	din[2][9:8]	din[2][9:8]			
txdata[95:90]	—	—	din[2][15:10]	din[2][15:10]	din[2][15:10]	din[2][15:10]			
txdata[99:96]	—	—	—	din[2][19:16]	din[2][19:16]	din[2][19:16]			
txdata[111:100]	—	—	—	—	din[2][31:20]	din[2][31:20]			
txdata[119:112]	—	—	—	—	—	din[2][39:32]			
txdata[127:120]	din[3][7:0]	din[3][7:0]	din[3][7:0]	din[3][7:0]	din[3][7:0]	din[3][7:0]	—	—	Word 3
txdata[129:128]	—	din[3][9:8]	din[3][9:8]	din[3][9:8]	din[3][9:8]	din[3][9:8]			
txdata[135:130]	—	—	din[3][15:10]	din[3][15:10]	din[3][15:10]	din[3][15:10]			
txdata[139:136]	—	—	—	din[3][19:16]	din[3][19:16]	din[3][19:16]			
txdata[151:140]	—	—	—	—	din[3][31:20]	din[3][31:20]			
txdata[159:152]	—	—	—	—	—	din[3][39:32]			

Table 2.21. RX Path Data Bus Mapping

The data in this table is transmitted in sequence in the transmission order as follows:

- First: rxdata[7:0]
- Last: rxdata[159:152]

IP Port Name	8-bit Mode	10-bit Mode	16-bit Mode	20-bit Mode	32-bit Mode	40-bit Mode	1 Word Mode	2 Word Mode	4 Word Mode
rxdata[7:0]	dout[0][7:0]	dout[0][7:0]	dout[0][7:0]	dout[0][7:0]	dout[0][7:0]	dout[0][7:0]	Word 0	Word 0	Word 0
rxdata[9:8]	—	dout[0][9:8]	dout[0][9:8]	dout[0][9:8]	dout[0][9:8]	dout[0][9:8]			
rxdata[15:10]	—	—	dout[0][15:10]	dout[0][15:10]	dout[0][15:10]	dout[0][15:10]			
rxdata[19:16]	—	—	—	dout[0][19:16]	dout[0][19:16]	dout[0][19:16]			
rxdata[31:20]	—	—	—	—	dout[0][31:20]	dout[0][31:20]			
rxdata[39:32]	—	—	—	—	—	dout[0][39:32]			
rxdata[47:40]	dout[1][7:0]	dout[1][7:0]	dout[1][7:0]	dout[1][7:0]	dout[1][7:0]	dout[1][7:0]	—	Word 1	Word 1
rxdata[49:48]	—	dout[1][9:8]	dout[1][9:8]	dout[1][9:8]	dout[1][9:8]	dout[1][9:8]			
rxdata[55:50]	—	—	dout[1][15:10]	dout[1][15:10]	dout[1][15:10]	dout[1][15:10]			
rxdata[59:56]	—	—	—	dout[1][19:16]	dout[1][19:16]	dout[1][19:16]			
rxdata[71:60]	—	—	—	—	dout[1][31:20]	dout[1][31:20]			
rxdata[79:72]	—	—	—	—	—	dout[1][39:32]			
rxdata[87:80]	dout[2][7:0]	dout[2][7:0]	dout[2][7:0]	dout[2][7:0]	dout[2][7:0]	dout[2][7:0]	—	—	Word 2
rxdata[89:88]	—	dout[2][9:8]	dout[2][9:8]	dout[2][9:8]	dout[2][9:8]	dout[2][9:8]			
rxdata[95:90]	—	—	dout[2][15:10]	dout[2][15:10]	dout[2][15:10]	dout[2][15:10]			
rxdata[99:96]	—	—	—	dout[2][19:16]	dout[2][19:16]	dout[2][19:16]			
rxdata[111:100]	—	—	—	—	dout[2][31:20]	dout[2][31:20]			
rxdata[119:112]	—	—	—	—	—	dout[2][39:32]			
rxdata[127:120]	dout[3][7:0]	dout[3][7:0]	dout[3][7:0]	dout[3][7:0]	dout[3][7:0]	dout[3][7:0]	—	—	Word 3
rxdata[129:128]	—	dout[3][9:8]	dout[3][9:8]	dout[3][9:8]	dout[3][9:8]	dout[3][9:8]			
rxdata[135:130]	—	—	dout[3][15:10]	dout[3][15:10]	dout[3][15:10]	dout[3][15:10]			
rxdata[139:136]	—	—	—	dout[3][19:16]	dout[3][19:16]	dout[3][19:16]			
rxdata[151:140]	—	—	—	—	dout[3][31:20]	dout[3][31:20]			
rxdata[159:152]	—	—	—	—	—	dout[3][39:32]			

2.3. Attributes Summary

The configurable attributes of the MPPHY Module are shown in Table 2.22 and are described in Table 2.23. The attributes can be configured through the IP Catalog Module/IP wizard of the Lattice Radiant software.

Table 2.22. Attributes Table

Attribute	Selectable Values	Default	Dependency on Other Attributes
General			
Protocol	CoaXpress, CPRI, DP_eDP, Ethernet, Generic, JESD204B, JESD204C, PCIe PIPE Direct, SLVS_EC, SDI	Ethernet	—
Protocol Mode	Refer to Table 1.2	10GBASE-R	Dependent on *Protocol*

Attribute	Selectable Values	Default	Dependency on Other Attributes
Fast Simulation Mode	Checked, Unchecked	Checked	Dependent on *Protocol Mode*
Enable ANLT	Checked, Unchecked	Unchecked	—
Link Width (Instances x Number of Lanes)	1x1 – 1x16	1x1	Dependent on *Protocol*
Lane ID	0 – 27	AUTO	Dependent on *Protocol*
Link Direction	Tx and Rx, Tx only, Rx only	Tx and Rx	Dependent on *Protocol*
RefClk Frequency (MHz)	Dependent on *Data Rate* and *Protocol*	156.25	Dependent on *Data Rate* and *Protocol*
Clk_Cascade	Enabled, Disabled	Disabled	Dependent on *Link Width*
Data Rate (Gbps)	[1–26.5625]	10.3125	For more details, refer to the Supported Continuous Data Rate Range section
Calculated MPLL VCO Freq (GHz)	—	10.3125	Dependent on *Data Rate* and *Protocol*
Calculated RX VCO Freq (GHz)	—	10.3125	Dependent on *Protocol and Protocol Mode*.
Data Width	8, 10, 16, 20, 32, 40, 66	32	Dependent on *Protocol Mode*
PLC Gearbox Ratio	1, 2, 4	1	Dependent on *Protocol Mode*
TX Global Clock Port Enable	Enable, Disabled	Enabled	—
RX Global Clock Port Enable	Enable, Disabled	Enabled	—
MPPHY TX Output Clk Frequency	—	156.25	For more details, refer to the TX Clock Settings section
MPPHY RX Output Clk Freq (MHz)	—	156.25	For more details, refer to the RX Clock Settings section
PMA Setup			
TX Amplitude	765, 781, 796, 811, 826, 842, 857, 872, 887, 903, 918, 933, 948, 964, 979, 994	994	—
TX Main Cursor	[0–24]	24	—
TX Pre Cursor	[0–24]	0	—
TX Post Cursor	[0–32]	0	—
RX Adapt Mode	SHORT, LONG	SHORT	—
RX Coupling Mode	AC Coupling, DC Coupling	AC Coupling	—
RX Loss of Sig Port Enable	Checked, Unchecked	Checked	—
DFE Enable	Enabled, Disabled	Enabled	—
RX Adaptive Equalization Enable	Checked, Unchecked	Checked	—
RX Attenuation Block	–2dB, –13dB	–13dB	—
RX CTLE Boost	[0–31]	17	—
RX CTLE Pole	[0–3]	2	—
RX AFE Rate	[0–7]	4	—
RX VGA1 Gain	[0–7]	5	—
RX VGA2 Gain	[0–7]	5	—

Attribute	Selectable Values	Default	Dependency on Other Attributes
PMA Control RX Adaptation Mode	Do not turn on adaptation when rxX_valid, Turn on adaptation only once when rxX_valid is asserted, Turn on adaptation everytime rxX_valid is asserted	Do not turn on adaptation when rxX_valid	—
PCS Setup			
Invert TX Data Polarity	Enabled, Disabled	Disabled	—
64b66b Encoder	Enabled, Disabled	Enabled	—
Scrambler	Enabled, Disabled	Enabled	—
(TX)FEC Mode	None, Ethernet RS-FEC, FC-FEC, CPRI RS-FEC	None	—
Invert RX Data Polarity	Enabled, Disabled	Disabled	—
Block Aligner	Enabled, Disabled	Enabled	—
Descrambler	Enabled, Disabled	Enabled	—
64b66b Decoder	Enabled, Disabled	Enabled	—
(RX)FEC Mode	None, Ethernet RS-FEC, FC-FEC, CPRI RS-FEC	None	—
Clock Frequency Compensation	Enabled, Disabled	Enabled	—
Elastic FIFO High Water Line	[0–31]	22	—
Elastic FIFO Low Water Line	[0–31]	10	—
8b10b Encoder	Enabled, Disabled	Enabled	—
Word Alignment	Enabled, Disabled	Enabled	—
Word Alignment Bit Width	10 bit, 20 bit	10 bit	—
COMMA byte to LSByte	Enabled, Disabled	Disabled	—
Primary Word Alignment Pattern Symbol 1 10B (HEX)	—	000	—
Primary Word Alignment Pattern Symbol 0 10B (HEX)	—	17C	—
Word Alignment Pattern Mask Code Symbol 1 10B (HEX)	—	000	—
Word Alignment Pattern Mask Code Symbol 0 10B (HEX)	—	000	—
Secondary Word Alignment	Enabled, Disabled	Enabled	—
Secondary Word Alignment Pattern Symbol 1 10B (HEX)	—	000	—
Secondary Word Alignment Pattern Symbol 0 10B (HEX)	—	283	—
Use LSByte of the Word Alignment	Enabled, Disabled	Enabled	—
Use 'Sync_det' FSM	Enabled, Disabled	Disabled	—
Number of Valid Sync Code Groups	[3–255]	3	Dependent on *Use 'Sync_det' FSM*
Number of Bad Code Groups	[3–63]	4	Dependent on *Use 'Sync_det' FSM*
Number of Good Code Groups	[3–255]	4	Dependent on *Use 'Sync_det' FSM*

Attribute	Selectable Values	Default	Dependency on Other Attributes
Sync_det Pattern Length	1, 2, 4	2	Dependent on *Use 'Sync_det' FSM*
Sync Pattern Alignment	Enabled, Disabled	Enabled	Dependent on *Use 'Sync_det' FSM*
Sync Pattern Code	8b_code, 10b_code	8b_code	Dependent on *Use 'Sync_det' FSM*
Secondary Sync Pattern	Enabled, Disabled	Enabled	Dependent on *Use 'Sync_det' FSM*
Primary Sync_det Pattern Byte 0 (HEX)	—	1BC	Dependent on *Use 'Sync_det' FSM*
Primary Sync_det Pattern Byte 1 (HEX)	—	050	Dependent on *Use 'Sync_det' FSM*
Primary Sync_det Pattern Byte 2 (HEX)	—	000	Dependent on *Use 'Sync_det' FSM*
Primary Sync_det Pattern Byte 3 (HEX)	—	000	Dependent on *Use 'Sync_det' FSM*
Secondary Sync_det Pattern Byte 0 (HEX)	—	1BC	Dependent on *Use 'Sync_det' FSM*
Secondary Sync_det Pattern Byte 1 (HEX)	—	0C5	Dependent on *Use 'Sync_det' FSM*
Secondary Sync_det Pattern Byte 2 (HEX)	—	000	Dependent on *Use 'Sync_det' FSM*
Secondary Sync_det Pattern Byte 3 (HEX)	—	000	Dependent on *Use 'Sync_det' FSM*
Sync_det Pattern Mask Byte 0 (HEX)	—	200	Dependent on *Use 'Sync_det' FSM*
Sync_det Pattern Mask Byte 1 (HEX)	—	200	Dependent on *Use 'Sync_det' FSM*
Sync_det Pattern Mask Byte 2 (HEX)	—	000	Dependent on *Use 'Sync_det' FSM*
Sync_det Pattern Mask Byte 3 (HEX)	—	000	Dependent on *Use 'Sync_det' FSM*
8b10b Decoder	Enabled, Disabled	Enabled	—
Clock Frequency Compensation	Enabled, Disabled	Disabled	—
CTC FIFO	Enabled, Disabled	Disabled	—
Skip Pattern Mask Code	[0 – 15]	0	—
Skip Pattern Code	8b_code, 10b_code	8b_code	—
Skip Pattern Length	1_byte, 2_byte, 4_byte	1_byte	—
Primary Skip Pattern Byte 0 (HEX)	—	17C	—
Primary Skip Pattern Byte 1 (HEX)	—	000	—
Primary Skip Pattern Byte 2 (HEX)	—	000	—
Primary Skip Pattern Byte 3 (HEX)	—	000	—
Secondary Skip Pattern	Enabled, Disabled	Disabled	—
Secondary Skip Pattern Byte 0 (HEX)	—	17C	Dependent on *Secondary Skip Pattern*
Secondary Skip Pattern Byte 1 (HEX)	—	000	Dependent on *Secondary Skip Pattern*
Secondary Skip Pattern Byte 2 (HEX)	—	000	Dependent on *Secondary Skip Pattern*
Secondary Skip Pattern Byte 3 (HEX)	—	000	Dependent on *Secondary Skip Pattern*

Attribute	Selectable Values	Default	Dependency on Other Attributes
Lane Alignment	Enabled, Disabled	Disabled	—
Input Data Code Mode	8b_code, 10b_code	8b_code	—
Lane Alignment Pattern Length	1_byte, 2_byte, 4_byte, 4_byte	1_byte	—
Maximum Lane-to-Lane Skew	NO_SKEW, 1_BYTE_SKEW, 2_BYTE_SKEW, 3_BYTE_SKEW, 4_BYTE_SKEW, 5_BYTE_SKEW, 6_BYTE_SKEW, 7_BYTE_SKEW, 8_BYTE_SKEW, 9_BYTE_SKEW, 10_BYTE_SKEW,	NO_SKEW	—
Primary Lane Alignment Pattern Byte 0 (HEX)	—	17C	—
Primary Lane Alignment Pattern Byte 1 (HEX)	—	000	—
Primary Lane Alignment Pattern Byte 2 (HEX)	—	000	—
Primary Lane Alignment Pattern Byte 3 (HEX)	—	000	—
Secondary Lane Alignment	Enabled	Disabled	—
Secondary Lane Alignment Pattern Byte 0 (HEX)	—	17C	Dependent on *Secondary Lane Alignment*
Secondary Lane Alignment Pattern Byte 1 (HEX)	—	000	Dependent on *Secondary Lane Alignment*
Secondary Lane Alignment Pattern Byte 2 (HEX)	—	000	Dependent on *Secondary Lane Alignment*
Secondary Lane Alignment Pattern Byte 3 (HEX)	—	000	Dependent on *Secondary Lane Alignment*
Lane Alignment Pattern Mask Code	[0–15]	0	—
Control Setup			
Loopback	No Loopback, Near_End_Parallel_Loopback, Far_End_Parallel_Loopback, Tx_to_RX_Serial_Loopback, Rx_to_TX_Parallel_Loopback	No Loopback	—
DR			
Enable Dynamic Reconfiguration	Checked, Unchecked	Unchecked	Dependent on *Protocol*
Number of Profiles	2, 3, 4	2	—
Profile 1 Protocol	—	Same as *Protocol*	Dependent on *Protocol*
Profile 1 Protocol Mode	—	Same as *Protocol Mode*	Dependent on *Protocol Mode*
Profile 1 Data Rate (Gbps)	[1–26.5625]	10.3125	For more details, refer to the Supported Continuous Data Rate Range section
Profile 1 RefClk Frequency (MHz)	Dependent on *Data Rate* and *Protocol*	156.25	Dependent on *Data Rate* and *Protocol*
Profile N Protocol	—	Same as *Protocol*	—
Profile N Protocol Mode	—	Same as *Protocol Mode*	Dependent on *Protocol*
Profile N Fast Simulation Mode	Checked, Unchecked	Checked	Dependent on *Profile N*

Attribute	Selectable Values	Default	Dependency on Other Attributes
			Protocol Mode
Profile N Enable ANLT Mode	Checked, Unchecked	Unchecked	Dependent on *Profile N Protocol Mode*
Profile N Link Width	1x1 – 1x16	1x1	Dependent on *Profile N Protocol*
Profile N Lane ID	0 – 27	AUTO	Dependent on *Profile N Protocol*
Profile N Link Direction	TX and Rx, TX only, RX only	TX and RX	Dependent on *Profile N Protocol*
Profile N RefClk Frequency	Dependent on *Data Rate* and *Protocol*	156.25	Dependent on *Profile N Data Rate* and *Profile N Protocol*
Profile N Clk_Cascade	Enabled, Disabled	Disabled	Dependent on *Profile N Link Width*
Profile N Data Width	8, 10, 16, 20, 32, 40, 66	32	For more details, refer to the Supported Continuous Data Rate Range section
Profile N PLC Gearbox Ratio	1, 2, 4	1	Dependent on *Profile N Data Rate* and *Profile N Protocol*
Profile N Data Rate (Gbps)	[1–26.5625]	10.3125	Dependent on *Profile N Protocol and Profile N Protocol Mode*
Profile N Calculated MPLL VCO Freq (GHz) [8 – 13.28125]	Dependent on *Data Rate* and *Protocol*	10.3125	Dependent on *Profile N Protocol Mode*
Profile N Calculated RX VCO Freq (GHz) [6 – 13.3]	—	10.3125	Dependent on *Profile N Protocol Mode*
Profile N TX Global Clock Port Enable	Enable, Disabled	Enabled	—
Profile N RX Global Clock Port Enable	Enable, Disabled	Enabled	—
Profile N MPPHY TX Output Clk Freq (MHz)	—	156.25	For more details, refer to the TX Clock Settings section
Profile N MPPHY RX Output Clk Freq (MHz)	—	156.25	For more details, refer to the RX Clock Settings section
Profile N TX Amplitude	765, 781, 796, 811, 826, 842, 857, 872, 887, 903, 918, 933, 948, 964, 979, 994	994	—
Profile N TX Main Cursor	[0–24]	24	—
Profile N TX Pre Cursor	[0–24]	0	—
Profile N TX Post Cursor	[0–32]	0	—
Profile N RX Adapt Mode	SHORT, LONG	SHORT	—
Profile N RX Coupling Mode	AC Coupling, DC Coupling	AC Coupling	—
Profile N RX Loss of Sig Port Enable	Checked, Unchecked	Checked	—
Profile N DFE Enable	Enabled, Disabled	Enabled	—
Profile N RX Adaptive Equalization Enable	Checked, Unchecked	Checked	—
Profile N RX Attenuation Block	–2dB, –13dB	–13dB	—
Profile N RX CTLE Boost	[0–31]	17	—
Profile N RX CTLE Pole	[0–3]	2	—
Profile N RX AFE Rate	[0–7]	4	—

Attribute	Selectable Values	Default	Dependency on Other Attributes
Profile N RX VGA1 Gain	[0–7]	5	—
Profile N RX VGA2 Gain	[0–7]	5	—
Profile N PMA Control RX Adaptation Mode	Do not turn on adaptation when rxX_valid, Turn on adaptation only once when rxX_valid is asserted, Turn on adaptation everytime rxX_valid is asserted	Do not turn on adaptation when rxX_valid	—
Profile N Invert TX Data Polarity	Enabled, Disabled	Disabled	—
Profile N 64b66b Encoder	Enabled, Disabled	Enabled	—
Profile N Scrambler	Enabled, Disabled	Enabled	—
Profile N (TX)FEC Mode	None, Ethernet RS-FEC, FC-FEC, CPRI RS-FEC	None	—
Profile N Invert RX Data Polarity	Enabled, Disabled	Disabled	—
Profile N Block Aligner	Enabled, Disabled	Enabled	—
Profile N Descrambler	Enabled, Disabled	Enabled	—
Profile N 64b66b Decoder	Enabled, Disabled	Enabled	—
Profile N (RX)FEC Mode	None, Ethernet RS-FEC, FC-FEC, CPRI RS-FEC	None	—
Profile N Clock Frequency Compensation	Enabled, Disabled	Enabled	—
Profile N Secondary Lane Alignment Pattern Byte 0 (HEX)	—	17C	—
Profile N Secondary Lane Alignment Pattern Byte 1 (HEX)	—	000	—
Profile N Secondary Lane Alignment Pattern Byte 2 (HEX)	—	000	—
Profile N Secondary Lane Alignment Pattern Byte 3 (HEX)	—	000	—
Profile N Elastic FIFO High Water Line	[0–31]	22	—
Profile N Elastic FIFO Low Water Line	[0–31]	10	—
Profile N 8b10b Encoder	Enabled, Disabled	Enabled	—
Profile N Word Alignment	Enabled, Disabled	Enabled	—
Profile N Word Alignment Bit Width	10 bit, 20 bit	10 bit	—
Profile N COMMA byte to LSByte	Enabled, Disabled	Disabled	—
Profile N Primary Word Alignment Pattern Symbol 1 10B (HEX)	—	000	—
Profile N Primary Word Alignment Pattern Symbol 0 10B (HEX)	—	17C	—
Profile N Word Alignment Pattern Mask Code Symbol 1 10B (HEX)	—	000	—
Profile N Word Alignment Pattern Mask Code Symbol 0 10B (HEX)	—	000	—
Profile N Secondary Word Alignment	Enabled, Disabled	Disabled	—
Profile N Secondary Word Alignment Pattern Symbol 1 10B (HEX)	—	000	—

Attribute	Selectable Values	Default	Dependency on Other Attributes
Profile N Secondary Word Alignment Pattern Symbol 0 10B (HEX)	—	283	—
Profile N Use LSByte of the Word Alignment	Enabled, Disabled	Enabled	—
Profile N Use 'Sync_det' FSM	Enabled, Disabled	Enabled	—
Profile N Number of Valid Sync Code Groups	[3–255]	3	Dependent on *Profile N Use 'Sync_det' FSM*
Profile N Number of Bad Code Groups	[3–63]	4	Dependent on *Profile N Use 'Sync_det' FSM*
Profile N Number of Good Code Groups	[3–255]	4	Dependent on *Profile N Use 'Sync_det' FSM*
Profile N Sync_det Pattern Length	1, 2, 4	2	Dependent on *Profile N Use 'Sync_det' FSM*
Profile N Sync Pattern Alignment	Enabled, Disabled	Enabled	Dependent on *Profile N Use 'Sync_det' FSM*
Profile N Sync Pattern Code	8b_code, 10b_code	8b_code	Dependent on *Profile N Use 'Sync_det' FSM*
Profile N Secondary Sync Pattern	Enabled, Disabled	Enabled	Dependent on *Profile N Use 'Sync_det' FSM*
Profile N Primary Sync_det Pattern Byte 0 (HEX)	—	1BC	Dependent on *Profile N Use 'Sync_det' FSM*
Profile N Primary Sync_det Pattern Byte 1 (HEX)	—	050	Dependent on *Profile N Use 'Sync_det' FSM*
Profile N Primary Sync_det Pattern Byte 2 (HEX)	—	000	Dependent on *Profile N Use 'Sync_det' FSM*
Profile N Primary Sync_det Pattern Byte 3 (HEX)	—	000	Dependent on *Profile N Use 'Sync_det' FSM*
Profile N Secondary Sync_det Pattern Byte 0 (HEX)	—	1BC	Dependent on *Profile N Use 'Sync_det' FSM*
Profile N Secondary Sync_det Pattern Byte 1 (HEX)	—	0C5	Dependent on *Profile N Use 'Sync_det' FSM*
Profile N Secondary Sync_det Pattern Byte 2 (HEX)	—	000	Dependent on *Profile N Use 'Sync_det' FSM*
Profile N Secondary Sync_det Pattern Byte 3 (HEX)	—	000	Dependent on *Profile N Use 'Sync_det' FSM*
Profile N Sync_det Pattern Mask Byte 0 (HEX)	—	200	Dependent on *Profile N Use 'Sync_det' FSM*
Profile N Sync_det Pattern Mask Byte 1 (HEX)	—	200	Dependent on *Profile N Use 'Sync_det' FSM*
Profile N Sync_det Pattern Mask Byte 2 (HEX)	—	000	Dependent on *Profile N Use 'Sync_det' FSM*
Profile N Sync_det Pattern Mask Byte 3 (HEX)	—	000	Dependent on *Profile N Use 'Sync_det' FSM*
Profile N 8b10b Decoder	Enabled, Disabled	Enabled	—
Profile N Clock Frequency Compensation	Enabled, Disabled	Disabled	—
Profile N CTC FIFO	Enabled, Disabled	Disabled	—
Profile N Skip Pattern Mask Code	[0 – 15]	0	—
Profile N Skip Pattern Code	8b_code, 10b_code	8b_code	—
Profile N Skip Pattern Length	1_byte, 2_byte, 4_byte	1_byte	—

Attribute	Selectable Values	Default	Dependency on Other Attributes
Profile N Primary Skip Pattern Byte 0 (HEX)	—	17C	—
Profile N Primary Skip Pattern Byte 1 (HEX)	—	000	—
Profile N Primary Skip Pattern Byte 2 (HEX)	—	000	—
Profile N Primary Skip Pattern Byte 3 (HEX)	—	000	—
Profile N Secondary Skip Pattern	Enabled, Disabled	Disabled	—
Profile N Secondary Skip Pattern Byte 0 (HEX)	—	17C	Dependent on *Profile N Secondary Skip Pattern*
Profile N Secondary Skip Pattern Byte 1 (HEX)	—	000	Dependent on *Profile N Secondary Skip Pattern*
Profile N Secondary Skip Pattern Byte 2 (HEX)	—	000	Dependent on *Profile N Secondary Skip Pattern*
Profile N Secondary Skip Pattern Byte 3 (HEX)	—	000	Dependent on *Profile N Secondary Skip Pattern*
Profile N Lane Alignment	Enabled, Disabled	Disabled	—
Profile N Input Data Code Mode	8b_code, 10b_code	8b_code	—
Profile N Lane Alignment Pattern Length	1_byte, 2_byte, 4_byte, 4_byte	1_byte	—
Profile N Maximum Lane-to-Lane Skew	NO_SKEW, 1_BYTE_SKEW, 2_BYTE_SKEW, 3_BYTE_SKEW, 4_BYTE_SKEW, 5_BYTE_SKEW, 6_BYTE_SKEW, 7_BYTE_SKEW, 8_BYTE_SKEW, 9_BYTE_SKEW, 10_BYTE_SKEW,	NO_SKEW	—
Profile N Primary Lane Alignment Pattern Byte 0 (HEX)	—	17C	—
Profile N Primary Lane Alignment Pattern Byte 1 (HEX)	—	000	—
Profile N Primary Lane Alignment Pattern Byte 2 (HEX)	—	000	—
Profile N Primary Lane Alignment Pattern Byte 3 (HEX)	—	000	—
Profile N Secondary Lane Alignment	Enabled	Disabled	—
Profile N Secondary Lane Alignment Pattern Byte 0 (HEX)	—	17C	Dependent on *Profile N Secondary Lane Alignment*
Profile N Secondary Lane Alignment Pattern Byte 1 (HEX)	—	000	Dependent on *Profile N Secondary Lane Alignment*
Profile N Secondary Lane Alignment Profile N Pattern Byte 2 (HEX)	—	000	Dependent on *Profile N Secondary Lane Alignment*
Profile N Secondary Lane Alignment Pattern Byte 3 (HEX)	—	000	Dependent on *Profile N Secondary Lane Alignment*
Profile N Lane Alignment Pattern Mask Code	[0–15]	0	—

Attribute	Selectable Values	Default	Dependency on Other Attributes
Profile N Loopback	No Loopback, Near_End_Parallel_Loopback, Far_End_Parallel_Loopback, TX_to_RX_Serial_Loopback, RX_to_TX_Parallel_Loopback	No Loopback	—

Note:

1. N is the number of profiles you select. N ranges from 2 to 4 in the **DR** tab. For *Number of Profiles* = 2, only attributes for Profile 2 are shown. For *Number of Profiles* = 3, only attributes for Profile 2 and 3 are shown. For *Number of Profiles* = 4, all attributes for Profile 2–4 are shown.

Table 2.23. Attributes Descriptions

Attribute	Description
Protocol	Specifies the protocol of the MPPHY Module.
Protocol Mode	Specifies the mode and data path that MPPHY Module enables.
Fast Simulation Mode	Enables this mode for RSFEC protocol modes.
Enable ANLT	Available only for Ethernet 10GBASE-R and 25GBASE-R.
Link Width	Specifies width in the form of Instances × Number of Lanes.
Lane ID	Specifies the lane which the IP is generated.
Link Direction	Specifies the MPPHY direction usage. Generation of the module ports are dependent on this attribute.
RefClk Frequency	List of the supported MPPHY reference clocks.
Clk_Cascade	If enabled, the output of the reference clock from the first Quad is used for the subsequent Quads.
Data Width	PMA data width if the value is 8, 10, 16, 20, 32, or 40. If the value is 66, PMA66 mode is enabled. Refer to the PCS Bypass section.
Data Rate	Specifies the data rate within the supported range (1 Gps to 26.5625 Gps).
Calculated MPLL VCO Freq	VCO frequency in the transmitter path in the range of [8.0–13.2815]. For more information about the calculation of this attribute, see the Clock Settings section.
Calculated RX VCO Freq	VCO frequency in the receiver path in the range of [6.0–13.3]. For more information about the calculation of this attribute, see the Clock Settings section.
PLC Gearbox Ratio	PLC clock divider control.
TX Global Clock Port Enable	When enabled, generates all TX global clocks of the MPPHY Module.
RX Global Clock Port Enable	When enabled, generates all RX global clocks of the MPPHY Module.
MPPHY TX Output Clk Freq	Calculated clock frequency. For more details, refer to the Clock Settings section.
MPPHY RX Output Clk Freq	Calculated clock frequency. For more details, refer to the Clock Settings section.
TX Amplitude	Sets tx_vboost_lvl and tx_iboost_lvl through the selection of **Amplitude (mVppd)**.
TX Main Cursor	Control for setting the transmitter driver output amplitude.
TX Pre Cursor	Controls transmitter driver output pre-emphasis (pre-shoot coefficient).
TX Post Cursor	Control for setting the transmitter driver output post-emphasis (post-coefficient).
RX Adapt Mode	Selects an adaptation mode. Algorithm trade-off between speed and accuracy: capture window, iterations, gains.
RX Coupling Mode	Controls the terminations of RX. Selector for floating versus grounded RX termination.
RX Loss of Sig Port Enable	When enabled, MPPHY Module generates the pma_rx*_sigdet_hf/lf_o ports.
DFE Enable	Controls the bypassing of Decision Feedback Equalization (DFE) circuitry. The bypass path is meant to save power for low data rates where the DFE is not required. The DFE is powered off when in bypass mode.
RX Adaptive Equalization Enable	When disabled, all subsequent EQ attributes can be controlled.
RX Attenuation Block	Controls the AFE attenuation level.
RX CTLE Boost	Controls the Continuous-Time Linear Equalizer (CTLE) boost level.
RX CTLE Pole	Controls the CTLE boost pole location.

Attribute	Description
RX AFE Rate	Controls the amount of parasitic boost provided by the negative C circuit in the equalizer and Variable Gain Amplifier (VGA) stages.
RX VGA1 Gain	Controls the AFE first stage VGA gain.
RX VGA2 Gain	Controls the AFE second stage VGA gain.
PMA Control RX Adaptation Mode	Specifies an adaptation algorithm with varying degrees of length and accuracy, which defines the length of the capture window, number of iterations, and update gains.
Invert TX Data Polarity	Inverts logical sense of txX_p and txX_m outputs. This signal has the effect of inverting the output data stream.
64b66b Encoder	Controls the bypassing of the Encoder block in 64b66b path.
Scrambler	Controls the bypassing of the Scrambler block 64b66b path.
(TX) FEC Mode	Selects the type of FEC mode to use in 64b66b transmit path.
Invert RX Data Polarity	Inverts logical sense of rxX_p and rxX_m outputs. This signal has the effect of inverting the output data stream.
Block Aligner	Controls the bypassing of Block Aligner in 64b66b path.
Descrambler	Controls the bypassing of the Descrambler block in 64b66b path.
64b66b Decoder	Controls the bypassing of the Decoder block in 64b66b path.
(RX) FEC Mode	Selects the type of FEC mode to use in 64b66b receive path.
Clock Frequency Compensation (for 64b66b)	Controls the bypassing of the CTC block in 64b66b path.
Elastic FIFO High Water Line	Clock compensation FIFO high water line.
Elastic FIFO Low Water Line	Clock compensation FIFO low water line.
8b10b Encoder	Controls the bypassing of the Encoder block in 8b10b path.
Word Alignment	Controls the bypassing of the Word Alignment Module in 8b10b path.
Word Alignment Bit Width	When set to 10-bit, only bit 9 to 0 of the 20-bit pattern and mask code are used.
COMMA byte to LSByte	When enabled, the module always puts the COMMA byte to LSByte (byte 0) of the data bus.
Primary Word Alignment Patter Symbol 1 10B (HEX)	Upper 10 bit of the Primary Word Alignment in hex format.
Primary Word Alignment Patter Symbol 0 10B (HEX)	Lower 10 bit of the Primary Word Alignment in hex format.
Word Alignment Pattern Mask Code Symbol 1 10B (HEX)	Upper 10 bit of the Word Alignment Pattern Mask Code in hex format.
Word Alignment Pattern Mask Code Symbol 0 10B (HEX)	Lower 10 bit of the Word Alignment Pattern Mask Code in hex format.
Secondary Word Alignment	Controls the disabling of Secondary Word Alignment feature.
Secondary Word Alignment Pattern Symbol 1 10B (HEX)	Upper 10 bit of the Secondary Word Alignment Pattern in hex format.
Secondary Word Alignment Pattern Symbol 0 10B (HEX)	Lower 10 bit of the Secondary Word Alignment Pattern in hex format.
Use LSByte of the Word Alignment	Specifies whether to put the LSByte of the word alignment pattern to [9:0] of the data bus.
Use 'Sync_det' FSM	Specifies whether to use Sync_det FSM to control the automatic word alignment.
Number of Valid Sync Code Groups	The number of valid synchronization code groups or ordered sets in decimal form that Sync_det FSM must receive to achieve synchronization state.
Number of Bad Code Groups	The number of bad code groups in decimal form received by Sync_det FSM to conclude the loss of synchronization.
Number of Good Code Groups	The number of continuous good code in decimal form groups received by Sync_det FSM to reduce the error count by one.
Sync_det Pattern Length	Specifies the length of Sync_det pattern.
Sync Pattern Alignment	Controls the disabling of Sync Pattern Alignment feature.
Sync Pattern Code	Controls the Sync_det pattern width.
Secondary Sync Pattern	Controls the disabling of Secondary Sync Pattern feature.

Attribute	Description
Primary Sync_det Pattern Byte 0 (HEX)	Byte 0 of the primary Sync_det pattern in hex format.
Primary Sync_det Pattern Byte 1 (HEX)	Byte 1 of the primary Sync_det pattern in hex format.
Primary Sync_det Pattern Byte 2 (HEX)	Byte 2 of the primary Sync_det pattern in hex format.
Primary Sync_det Pattern Byte 3 (HEX)	Byte 3 of the primary Sync_det pattern in hex format.
Secondary Sync_det Pattern Byte 0 (HEX)	Byte 0 of the secondary Sync_det pattern in hex format.
Secondary Sync_det Pattern Byte 1 (HEX)	Byte 1 of the secondary Sync_det pattern in hex format.
Secondary Sync_det Pattern Byte 2 (HEX)	Byte 2 of the secondary Sync_det pattern in hex format.
Secondary Sync_det Pattern Byte 3 (HEX)	Byte 3 of the secondary Sync_det pattern in hex format.
Sync_det Pattern Mask Byte 0 (HEX)	Byte 0 of the Sync_det pattern mask in hex format.
Sync_det Pattern Mask Byte 1 (HEX)	Byte 1 of the Sync_det pattern mask in hex format.
Sync_det Pattern Mask Byte 2 (HEX)	Byte 2 of the Sync_det pattern mask in hex format.
Sync_det Pattern Mask Byte 3 (HEX)	Byte 3 of the Sync_det pattern mask in hex format.
8b10b Decoder	Controls the bypassing of the Decoder block in 8b10b path.
Clock Frequency Compensation (for 8b10b)	Controls the bypassing of the Clock Frequency Compensation block in 8b10b path.
CTC FIFO	Controls the bypassing of the CTC FIFO block in 8b10b path.
Skip Pattern Mask Code	Controls whether the SKIP pattern is ignored during SKIP pattern matching.
Skip Pattern Code	Specifies whether the input data is in 8b or 10b code mode.
Skip Pattern Length	Specifies the SKIP pattern length in byte format.
Primary Skip Pattern Byte 0 (HEX)	Byte 0 of the primary SKIP pattern in hex format.
Primary Skip Pattern Byte 1 (HEX)	Byte 1 of the primary SKIP pattern in hex format.
Primary Skip Pattern Byte 2 (HEX)	Byte 2 of the primary SKIP pattern in hex format.
Primary Skip Pattern Byte 3 (HEX)	Byte 3 of the primary SKIP pattern in hex format.
Secondary Skip Pattern Byte 0 (HEX)	Byte 0 of the secondary SKIP pattern in hex format.
Secondary Skip Pattern Byte 1 (HEX)	Byte 1 of the secondary SKIP pattern in hex format.
Secondary Skip Pattern Byte 2 (HEX)	Byte 2 of the secondary SKIP pattern in hex format.
Secondary Skip Pattern Byte 3 (HEX)	Byte 3 of the secondary SKIP pattern in hex format.
Lane Alignment	Controls the disabling of the Lane Alignment feature.
Input Data Code Mode	Controls the Lane Alignment Input Data Code width.
Lane Alignment Pattern Length	Controls the Lane Alignment Pattern Length in byte format.
Maximum Lane-to-Lane Skew	Controls the maximum lane-to-lane skew in byte format. Hardware can handle up to 10 byte.
Primary Lane Alignment Pattern Byte 0 (HEX)	Byte 0 of the primary lane alignment pattern in hex format.
Primary Lane Alignment Pattern Byte 1 (HEX)	Byte 1 of the primary lane alignment pattern in hex format.

Attribute	Description
Primary Lane Alignment Pattern Byte 2 (HEX)	Byte 2 of the primary lane alignment pattern in hex format.
Primary Lane Alignment Pattern Byte 3 (HEX)	Byte 3 of the primary lane alignment pattern in hex format.
Secondary Lane Alignment	Controls the disabling of secondary lane alignment pattern matching feature.
Secondary Lane Alignment Pattern Byte 0 (HEX)	Byte 0 of the secondary lane alignment pattern in hex format.
Secondary Lane Alignment Pattern Byte 1 (HEX)	Byte 1 of the secondary lane alignment pattern in hex format.
Secondary Lane Alignment Pattern Byte 2 (HEX)	Byte 2 of the secondary lane alignment pattern in hex format.
Secondary Lane Alignment Pattern Byte 3 (HEX)	Byte 3 of the secondary lane alignment pattern in hex format.
Lane Alignment Pattern Mask Code	Controls whether the lane alignment pattern mask code is ignored during alignment pattern matching.
Loopback	Specifies the type of loopback the MPPHY Module enables.
Enable Dynamic Reconfiguration	Enables dynamic reconfiguration with PRF generation and testbench.
Number of Profiles	Determines the number of profiles generated and run.
Profile 1 Protocol	Shows the protocol of Profile 1.
Profile 1 Protocol Mode	Shows the protocol mode of Profile 1.
Profile 1 Data Rate (Gbps)	Shows the data rate of Profile 1.
Profile 1 RefClk Frequency (MHz)	Shows the reference clock frequency of Profile 1.
Profile N Protocol	Specifies the protocol of Profile N.
Profile N Protocol Mode	Specifies the mode and data path that Profile N enables.
Profile N Enable ANLT Mode	Enables ANLT mode. When enabled, Ethernet 10GBASE-R and 25GBASE-R become 10GBASE-KR and 25GBASE-KR respectively
Profile N Link Width	Specifies width in the form of Instances × Number of Lanes.
Profile N Lane ID	Specifies the lane which the IP is generated.
Profile N Link Direction	Specifies the MPPHY direction usage of Profile N. Generation of the module ports are dependent on this attribute.
Profile N RefClk Frequency	Lists of the supported MPPHY reference clocks for Profile N.
Profile N Data Width	PMA data width if the value is 8, 10, 16, 20, 32, or 40. If the value is 66, PMA66 mode is enabled. Refer to the PCS Bypass section.
Profile N PLC Gearbox Ratio	PLC clock divider control.
Profile N Data Rate	Specifies the data rate within the supported range (1 Gps to 26.5625 Gps).
Profile N Calculated MPLL VCO Freq	VCO frequency in the transmitter path in the range of [8.0–13.2815]. For more information about the calculation of this attribute, see the Clock Settings section.
Profile N Calculated RX VCO Freq	VCO frequency in the receiver path in the range of [6.0–13.3]. For more information about the calculation of this attribute, see the Clock Settings section.
Profile N TX Global Clock Port Enable	When enabled, generates all TX global clocks of Profile N.
Profile N RX Global Clock Port Enable	When enabled, generates all RX global clocks of Profile N.
Profile N MPPHY TX Output Clk Freq	Calculated clock frequency. For more details, refer to the Clock Settings section.
Profile N MPPHY RX Output Clk Freq	Calculated clock frequency. For more details, refer to the Clock Settings section.
Profile N TX Amplitude	Sets tx_vboost_lvl and tx_iboost_lvl through the selection of **Amplitude (mVppd)**.
Profile N TX Main Cursor	Sets the transmitter driver output amplitude.
Profile N TX Pre Cursor	Controls transmitter driver output pre-emphasis (pre-shoot coefficient).
Profile N TX Post Cursor	Sets the transmitter driver output post-emphasis (post-coefficient).

Attribute	Description
Profile N RX Adapt Mode	Selects an adaptation mode. Algorithm trade-off between speed and accuracy: capture window, iterations, gains.
Profile N RX Coupling Mode	Controls the terminations of RX. Selector for floating versus grounded RX termination.
Profile N RX Loss of Sig Port Enable	When enabled, MPPHY Module of Profile N generates the pma_rx*_sigdet_hf/lf_o ports.
Profile N DFE Enable	Controls the bypassing of Decision Feedback Equalization (DFE) circuitry. The bypass path is meant to save power for low data rates where the DFE is not required. The DFE is powered off in bypass mode.
Profile N RX Adaptive Equalization Enable	When disabled, all subsequent EQ attributes can be controlled.
Profile N RX Attenuation Block	Controls the AFE attenuation level.
Profile N RX CTLE Boost	Controls the Continuous-Time Linear Equalizer (CTLE) boost level.
Profile N RX CTLE Pole	Controls the CTLE boost pole location.
Profile N RX AFE Rate	Controls the amount of parasitic boost provided by the negative C circuit in the equalizer and Variable Gain Amplifier (VGA) stages.
Profile N RX VGA1 Gain	Controls the AFE first stage VGA gain.
Profile N RX VGA2 Gain	Controls the AFE second stage VGA gain.
Profile N PMA Control RX Adaptation Mode	Specifies an adaptation algorithm with varying degrees of length and accuracy, which defines the length of the capture window, number of iterations, and update gains.
Profile N Invert TX Data Polarity	Inverts logical sense of txX_p and txX_m outputs. This signal has the effect of inverting the output data stream.
Profile N 64b66b Encoder	Controls the bypassing of the Encoder block in 64b66b path.
Profile N Scrambler	Controls the bypassing of the Scrambler block in 64b66b path.
Profile N (TX)FEC Mode	Selects the type of FEC mode to use in 64b66b transmit path.
Profile N Invert RX Data Polarity	Inverts logical sense of rxX_p and rxX_m outputs. This signal has the effect of inverting the output data stream.
Profile N Block Aligner	Controls the bypassing of Block Aligner in 64b66b path.
Profile N Descrambler	Controls the bypassing of the Descrambler block in 64b66b path.
Profile N 64b66b Decoder	Controls the bypassing of the Decoder block in 64b66b path.
Profile N (RX)FEC Mode	Selects the type of FEC mode to use in 64b66b receive path.
Profile N Clock Frequency Compensation	Controls the bypassing of the CTC block in 64b66b path.
Profile N Secondary Lane Alignment Pattern Byte 0 (HEX)	Byte 0 of the secondary lane alignment pattern in hex format.
Profile N Secondary Lane Alignment Pattern Byte 1 (HEX)	Byte 1 of the secondary lane alignment pattern in hex format.
Profile N Secondary Lane Alignment Pattern Byte 2 (HEX)	Byte 2 of the secondary lane alignment pattern in hex format.
Profile N Secondary Lane Alignment Pattern Byte 3 (HEX)	Byte 3 of the secondary lane alignment pattern in hex format.
Profile N Elastic FIFO High Water Line	Clock compensation FIFO high water line.
Profile N Elastic FIFO Low Water Line	Clock compensation FIFO low water line.
Profile N 8b10b Encoder	Controls the bypassing of the Encoder block in 8b10b path.
Profile N Word Alignment	Controls the bypassing of the Word Alignment Module in 8b10b path.
Profile N Word Alignment Bit Width	When set to 10-bit, only bit 9 to 0 of the 20-bit pattern and mask code are used.
Profile N COMMA byte to LSByte	When enabled, the module always put the COMMA byte to LSByte (byte 0) of the data bus.

Attribute	Description
Profile N Primary Word Alignment Pattern Symbol 1 10B (HEX)	Upper 10 bit of the Primary Word Alignment in hex format.
Profile N Primary Word Alignment Pattern Symbol 0 10B (HEX)	Lower 10 bit of the Primary Word Alignment in hex format.
Profile N Word Alignment Pattern Mask Code Symbol 1 10B (HEX)	Upper 10 bit of the Word Alignment Pattern Mask Code in hex format.
Profile N Word Alignment Pattern Mask Code Symbol 0 10B (HEX)	Lower 10 bit of the Word Alignment Pattern Mask Code in hex format.
Profile N Secondary Word Alignment	Controls the disabling of Secondary Word Alignment feature.
Profile N Secondary Word Alignment Pattern Symbol 1 10B (HEX)	Upper 10 bit of the Secondary Word Alignment Pattern in hex format.
Profile N Secondary Word Alignment Pattern Symbol 0 10B (HEX)	Lower 10 bit of the Secondary Word Alignment Pattern in hex format.
Profile N Use LSByte of the Word Alignment	Specifies whether to put the LSByte of the word alignment pattern to [9:0] of the data bus.
Profile N Use 'Sync_det' FSM	Specifies whether to use Sync_det FSM to control the automatic word alignment.
Profile N Number of Valid Sync Code Groups	The number of valid synchronization code groups or ordered sets in decimal form that Sync_det FSM must receive to achieve synchronization state.
Profile N Number of Bad Code Groups	The number of bad code groups in decimal form received by Sync_det FSM to conclude the loss of synchronization.
Profile N Number of Good Code Groups	The number of continuous good code in decimal form groups received by Sync_det FSM to reduce the error count by one.
Profile N Sync_det Pattern Length	Specifies the length of Sync_det pattern.
Profile N Sync Pattern Alignment	Controls the disabling of Sync Pattern Alignment feature.
Profile N Sync Pattern Code	Controls the Sync_det pattern width.
Profile N Secondary Sync Pattern	Controls the disabling of Secondary Sync Pattern feature.
Profile N Primary Sync_det Pattern Byte 0 (HEX)	Byte 0 of the primary Sync_det pattern in hex format.
Profile N Primary Sync_det Pattern Byte 1 (HEX)	Byte 1 of the primary Sync_det pattern in hex format.
Profile N Primary Sync_det Pattern Byte 2 (HEX)	Byte 2 of the primary Sync_det pattern in hex format.
Profile N Primary Sync_det Pattern Byte 3 (HEX)	Byte 3 of the primary Sync_det pattern in hex format.
Profile N Secondary Sync_det Pattern Byte 0 (HEX)	Byte 0 of the secondary Sync_det pattern in hex format.
Profile N Secondary Sync_det Pattern Byte 1 (HEX)	Byte 1 of the secondary Sync_det pattern in hex format.
Profile N Secondary Sync_det Pattern Byte 2 (HEX)	Byte 2 of the secondary Sync_det pattern in hex format.
Profile N Secondary Sync_det Pattern Byte 3 (HEX)	Byte 3 of the secondary Sync_det pattern in hex format.
Profile N Sync_det Pattern Mask Byte 0 (HEX)	Byte 0 of the Sync_det pattern mask in hex format.
Profile N Sync_det Pattern Mask	Byte 1 of the Sync_det pattern mask in hex format.

Attribute	Description
Byte 1 (HEX)	
Profile N Sync_det Pattern Mask Byte 2 (HEX)	Byte 2 of the Sync_det pattern mask in hex format.
Profile N Sync_det Pattern Mask Byte 3 (HEX)	Byte 3 of the Sync_det pattern mask in hex format.
Profile N 8b10b Decoder	Controls the bypassing of the Decoder block in 8b10b path.
Profile N Clock Frequency Compensation	Controls the bypassing of the Clock Frequency Compensation block in 8b10b path.
Profile N CTC FIFO	Controls the bypassing of the CTC FIFO block in 8b10b path.
Profile N Skip Pattern Mask Code	Controls whether the SKIP pattern is ignored during SKIP pattern matching.
Profile N Skip Pattern Code	Specifies whether the input data is in 8b or 10b code mode.
Profile N Skip Pattern Length	Specifies the SKIP pattern length in byte format.
Profile N Primary Skip Pattern Byte 0 (HEX)	Byte 0 of the primary SKIP pattern in hex format.
Profile N Primary Skip Pattern Byte 1 (HEX)	Byte 1 of the primary SKIP pattern in hex format.
Profile N Primary Skip Pattern Byte 2 (HEX)	Byte 2 of the primary SKIP pattern in hex format.
Profile N Primary Skip Pattern Byte 3 (HEX)	Byte 3 of the primary SKIP pattern in hex format.
Profile N Secondary Skip Pattern	Enables configuration for secondary SKIP pattern
Profile N Secondary Skip Pattern Byte 0 (HEX)	Byte 0 of the secondary SKIP pattern in hex format.
Profile N Secondary Skip Pattern Byte 1 (HEX)	Byte 1 of the secondary SKIP pattern in hex format.
Profile N Secondary Skip Pattern Byte 2 (HEX)	Byte 2 of the secondary SKIP pattern in hex format.
Profile N Secondary Skip Pattern Byte 3 (HEX)	Byte 3 of the secondary SKIP pattern in hex format.
Profile N Lane Alignment	Controls the disabling of the Lane Alignment feature.
Profile N Input Data Code Mode	Controls the Lane Alignment Input Data Code width.
Profile N Lane Alignment Pattern Length	Controls the Lane Alignment Pattern Length in byte format.
Profile N Maximum Lane-to-Lane Skew	Controls the maximum lane-to-lane skew in byte format. Hardware can handle up to 10 byte.
Profile N Primary Lane Alignment Pattern Byte 0 (HEX)	Byte 0 of the primary lane alignment pattern in hex format.
Profile N Primary Lane Alignment Pattern Byte 1 (HEX)	Byte 1 of the primary lane alignment pattern in hex format.
Profile N Primary Lane Alignment Pattern Byte 2 (HEX)	Byte 2 of the primary lane alignment pattern in hex format.
Profile N Primary Lane Alignment Pattern Byte 3 (HEX)	Byte 3 of the primary lane alignment pattern in hex format.
Profile N Secondary Lane Alignment	Controls the disabling of secondary lane alignment pattern matching feature.
Profile N Secondary Lane Alignment Pattern Byte 0 (HEX)	Byte 0 of the secondary lane alignment pattern in hex format.
Profile N Secondary Lane Alignment Pattern Byte 1 (HEX)	Byte 1 of the secondary lane alignment pattern in hex format.
Profile N Secondary Lane Alignment Profile N Pattern Byte 2 (HEX)	Byte 2 of the secondary lane alignment pattern in hex format.

Attribute	Description
Profile N Secondary Lane Alignment Pattern Byte 3 (HEX)	Byte 3 of the secondary lane alignment pattern in hex format.
Profile N Lane Alignment Pattern Mask Code	Controls whether the lane alignment pattern mask code is ignored during alignment pattern matching.
Profile N Loopback	Specifies the type of loopback the MPPHY Module enables.

Note:

1. N is the number of profiles you select. N ranges from 2 to 4 in the **DR** tab. For *Number of Profiles* = 2, only attributes for Profile 2 are shown. For *Number of Profiles* = 3, only attributes for Profile 2 and 3 are shown. For *Number of Profiles* = 4, all attributes for Profile 2–4 are shown.

2.4. Operation Details

2.4.1. Reset Sequence

2.4.1.1. Cold Boot Reset

When all resets are deasserted at power-up, the HRC manages the release sequence, and operation can begin when the *reset_done signals are asserted.

2.4.1.2. Warm Reset

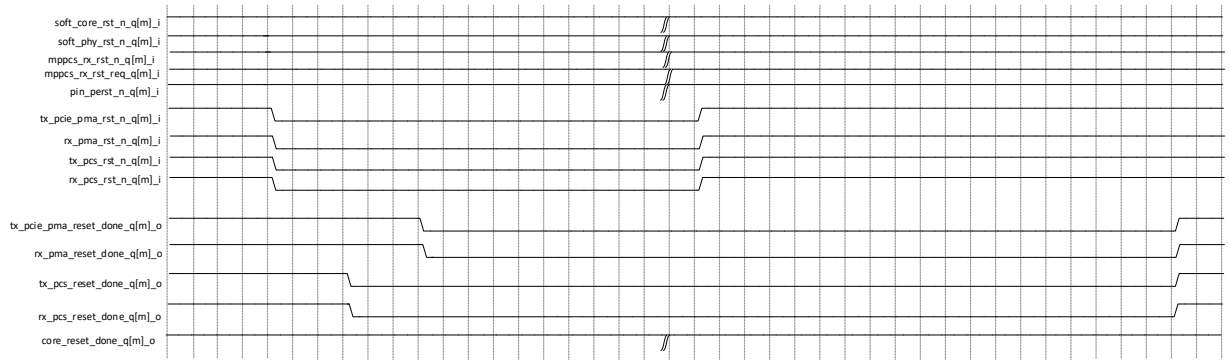

Figure 2.9. PHY Reset Sequence for Warm-Boot

During a warm boot, the PHY is not power cycled. You must perform the PHY reset sequence as follows:

1. Deassert the following reset signals to force the PHY into reset:
   - tx_pcie_pma_rst_n_q[m]_i
   - rx_pma_rst_n_q[m]_i
   - tx_pcs_rst_n_q[m]_i
   - rx_pcs_rst_n_q[m]_i
2. Keep the reset signals deasserted until the *reset_done signals are deasserted, indicating the PHY is in reset.
3. Assert the reset signals to release the PHY from reset.
4. When the *_reset_done signals are asserted, the PHY is out of reset and ready for user data transaction.

Notes:

- Throughout the PHY reset sequence, the soft_core_rst_n_q[m]_i, soft_phy_rst_n_q[m]_i, mppcs_rx_rst_n_q[m]_i, mppcs_rx_rst_req_q[m]_i, and pin_perst_n_q[m]_i signals must remain high.
- The *reset_done signals being high indicates that the reset process is complete and the block is ready to accept the next warm reset request. Warm reset requests are accepted only when the *reset_done signals are high. Any warm reset request issued when the *reset_done signals are low is not accepted because the reset process is ongoing.

- The mppcs_rx_rst_req_q0_i signal has dependency on the number of lanes. This figure assumes that the number of lanes = 1.
- If the protocol is PMA Direct mode, all PCS reset signals must be tied to 1.
- TX and RX reset signals can be toggled independently.
- m: Represents the quad number selected.

2.4.2. Clock Settings

MPPHY Module has predefined PLL and divider settings for each protocol. For Generic mode, you can set the following dividers and multipliers to obtain the target VCO and output clock frequencies. Note that the computation for TX and RX clock frequencies are different.

Note: GUI Attributes = Variable in the equation (user input or calculated)

RefClk Frequency (MHz) = Frefclk (user input)

Ref Clk MPLL Divider = refclk_mpll_div (calculated)

MPLL Multiplier = mpll_multiplier (calculated)

PHY RX Ref LD (calculated)

PHY RX VCO LD M (calculated)

PHY RX CDR PPM Max (calculated)

PHY TX Rate (calculated)

PHY RX Rate (calculated)

Data Width (user input)

PLC Gearbox Ratio = plc_clk_div_ctrl (user input)

Enable Div16p5 Clock for PLLA/B = div16p5_en (calculated)

2.4.2.1. TX Clock Settings

MPLLA/B TX Clock Divider = mpll_tx_clk_div (calculated)

MPLLA/B TX Word Clock Divider = mpll_word_clk_div (calculated)

TX PCS Clock Divider = tx_pcs_clk_div_ctrl (calculated)

TX MPPCS Clock Divider = tx_mppcs_clk_div_ctrl (calculated)

TX SERDES Clock Divider = tx_serdes_clk_div_ctrl (calculated)

$$\text{MPLL VCO Freq} = \frac{Frefclk}{2^{refclk\_mpll\_div}} \text{x} \left(2^{(1+mpll\_fb\_clk\_div4\_en)}\right) \text{x} \frac{mpll\_multiplier}{4}$$

$$Word\ Clock\ Frequency = \frac{(\text{MPLL VCO Freq}/\text{mpll\_tx\_clk\_div})}{\text{mpll\_word\_clk\_div}}$$

$$Div16p5\ Clock\ Frequency = \frac{\text{MPLL VCO Freq}}{16.5}$$

$$TX\ Output\ Clock\ Frequency = \frac{\text{PCS Clock Frequency}}{\text{Gearbox Ratio}}$$

Where

- For non-pipe mode, the recommended value for mpll_fb_clk_div4_en is 1.
- PCS clock frequency is dependent on the selectors as shown in the figure below.

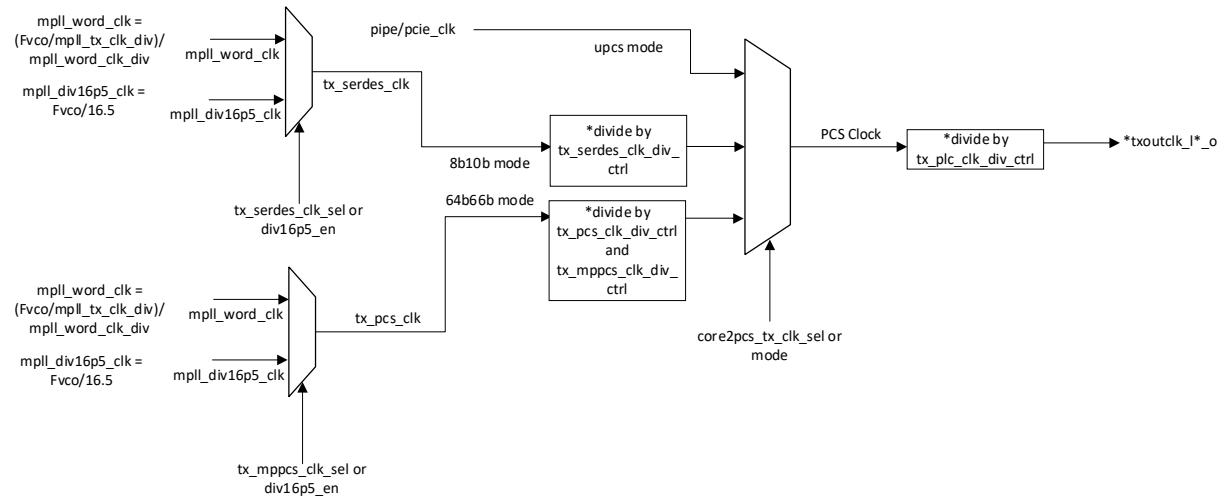

Figure 2.10. TX Clock Path

2.4.2.2. RX Clock Settings

RX PCS Clock Divider = rx_pcs_clk_div_ctrl (calculated)

RX MPPCS Clock Divider = rx_mppcs_clk_div_ctrl (calculated)

$$RX\ VCO\ Freq = datarate \times 2^{(rate-1)}$$

$$Div16p5\ Clock\ Frequency = \frac{\text{RX VCO Freq}}{16.5}$$

$$PHY\ RX\ Clock\ Freq = \frac{\text{datarate}}{\text{width}}$$

$$PHY\ RX\ PCS\ Clock\ Freq = \frac{(\text{Div16p5 Clock Freq}/\text{rx\_mppcs\_clk\_div\_ctrl})}{\text{rx\_pcs\_clk\_div\_ctrl}}$$

$$RX\ Output\ Clk\ Freq = \frac{\text{PCS Clock Frequency}}{\text{Gearbox Ratio}}$$

Where PCS clock frequency is dependent on the selectors as shown in the figure below.

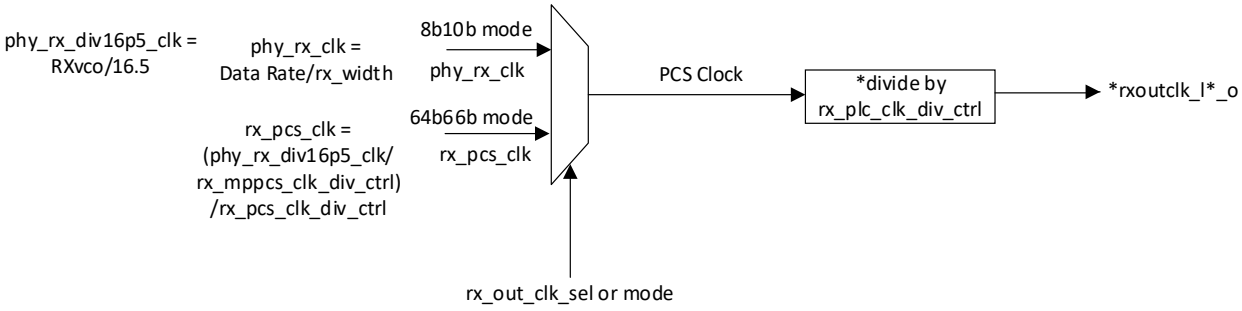

Figure 2.11. RX Clock Path

2.4.2.3. Supported Continuous Data Rate Range

For Generic mode, you can input the continuous data rate from 1 Gbps to 26.5625 Gbps based on the supported range listed in the table below. Data rate that is out of the range are not supported.

Table 2.24. Supported Continuous Data Rate

Supported Continuous Data Rate (Gbps)	
Min	**Max**
22	26.5625
16	20.625
11	13.28125
8	10.3125
7.333333	8.854167
5.333333	6.875
5.5	6.640625
4.4	5.3125
3.2	4.125
4	5.15625
3.666667	4.427083
2.666667	3.4375
2.75	3.320313
2	2.578125
2.2	2.65625
1.6	2.0625
1.375	1.660156
1	1.289063

2.4.3. Data Transmission

To begin PHY transaction, you must provide the required reference clocks with correct frequency to the MPPHY Module. MPLL settings must be properly set to get the desired TX/RX output clock frequency. After CDR lock, the rx_cdr_valid_q*_o signal is asserted indicating that the calibration is complete. The assertion of status ports depends on the protocol used. For details, refer to item 4 in the Test Sequence section. When the MPPHY Module testbench signal pcs_ready is asserted, MPPHY Module is ready to transmit and receive data.

Figure 2.12 shows the overall timing sequence from PHY initialization up to data transmission.

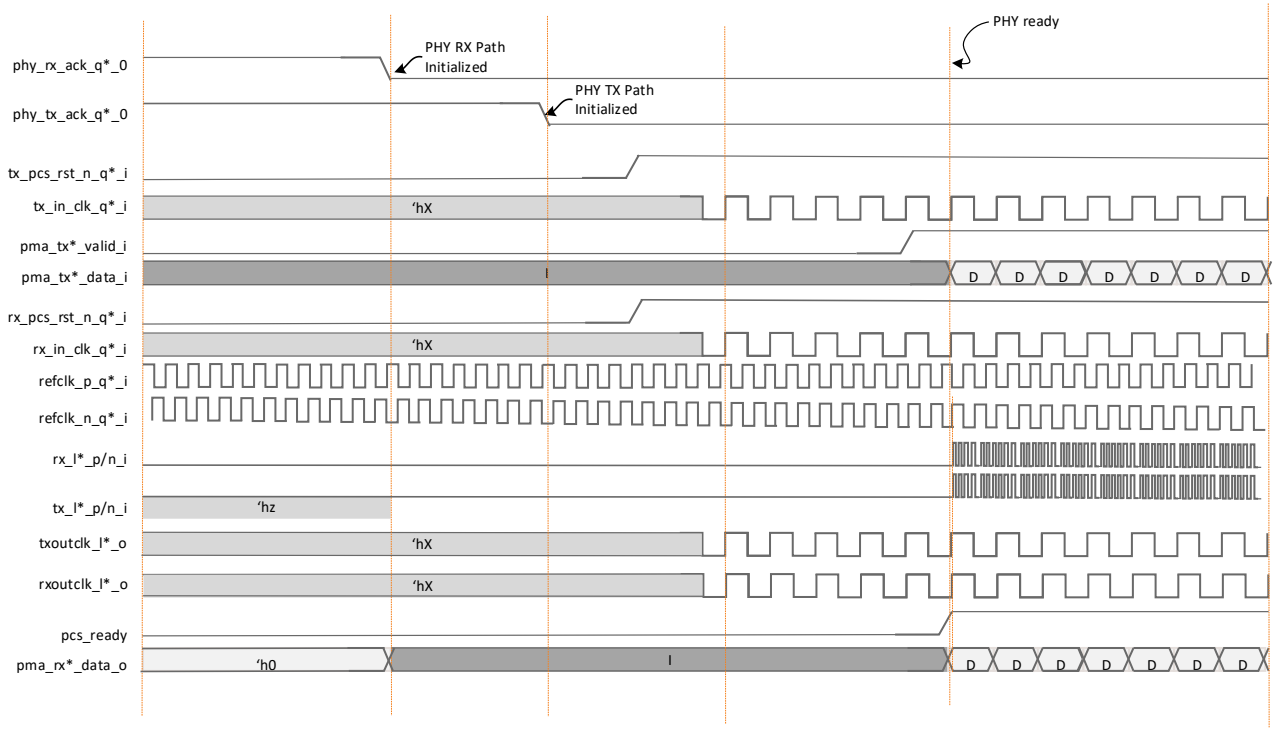

Figure 2.12. MPPHY Timing Diagram

2.4.4. MPPHY Placement

The Lattice Avant and Nexus 2 devices support seven multi-protocol quads, where MPPHY Module bonds up to 4 quads only. You can place the MPPHY instance as follows:

- Use the constraint file in ldc_set_location.

  Example: By using the following command, MPPHY is placed in Channel 1 of Quad 3.

  ```
  ldc_set_location -site {MPQ3_RX1N} [get_ports rx_l0_n_i]
  ```

- Set the *Lane ID* attribute in the MPPHY GUI.

Notes:

- If the constraint conflicts with the *Lane ID* attribute setting, the Lattice Radiant software follows the ldc_set_location.
- The Lattice Radiant software automatically sets the lane number if both ldc_set_location and the *Lane ID* attribute are not set.
- Any constraints for MPPHY merging and assignment must be in the *.ldc* or *.sdc* file, not in the *.pdc* file. Merging happens during post-syn (before map), however the pin location constraints defined in the *.pdc* file take effect only during map.

2.4.5. Quad-Quad Connection

Foundation IP automatically connects each quad as shown in the figure below.

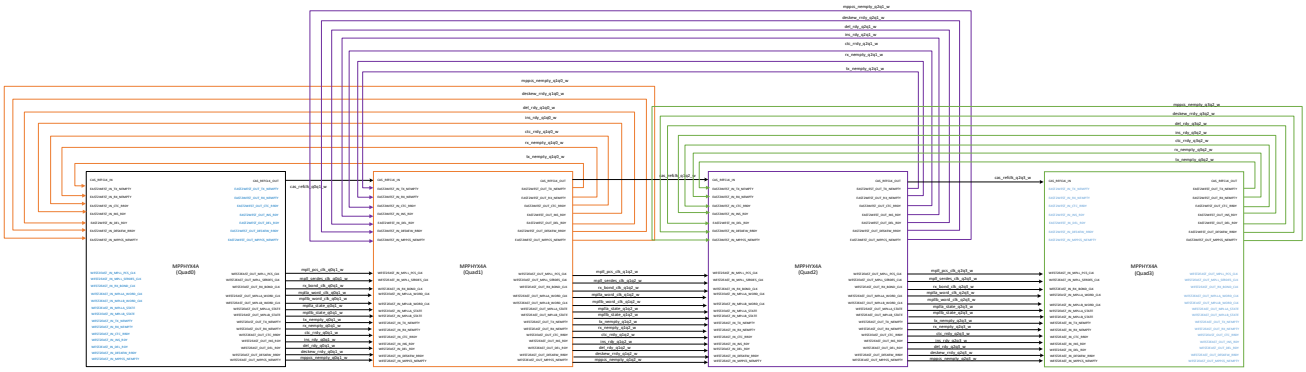

Figure 2.13. Quad-Quad Connection

2.4.6. LMMI Arbiter

Multi lanes in the same quad are sharing a single LMMI arbiter. LMMI arbiter prioritizes LMMI access. This component is inserted during synthesis.

3. Register Description

The MPPHY Module supports LMMI to access the SERDES registers. For more details about these registers, refer to the Lattice Avant SERDES/PCS User Guide (FPGA-TN-02313).

4. Testbench Simulation Support

This section provides information regarding the tests supported by testbench included in the MPPHY Module and provides recommended setup for the limitation.

4.1. Test Sequence

1. PHY initialization test
   - Testbench waits for the change of state of the phy_tx_ack_q*_o and phy_rx_ack_q*_o signals. Deassertion of these signals indicates that PHY initializes successfully.
2. TX/RX and global TX/RX clock check
   - Frequency check for txoutclk_l*_o, rxoutclk_l*_o, txoutgclk_pll*_q*_o, and rxoutgclk_l*_o.
   - Checking of global clocks is available only if *TX/RX Global Clock Port Enable* == Enabled.
3. LMMI read check
   - MPPHY Module sets the default values for the lane, mpll, system, and CMU HRC registers per protocol including the attributes set in the GUI. For more details about these registers, refer to the Lattice Avant SERDES/PCS User Guide (FPGA-TN-02313).
   - This test covers the LMMI reading to registers and compares the reading to the expected values based on the protocol used.
4. Waiting of pcs_ready
   - Assertion of pcs_ready indicates that MPPHY is now ready and can perform normal transaction test. This signal is available only in your testbench and is coded as follows:

     For 64b66b path in lane[0]:
     ```
     assign pcs_ready    = ~phy_rx_ack_q0_o[0] && rx_cdr_valid_q0_o[0] && pma_rx0_blk_lock_o;
     ```
     For 64b66b path with enabled FEC block in lane[0]:
     ```
     assign pcs_ready    = ~phy_rx_ack_q0_o[0] && rx_cdr_valid_q0_o[0] && pma_rx0_blk_lock_o
     && pma_rx0_fec_lock_o;
     ```
     For 8b10b path with enabled Lane Alignment block in lane[0]:
     ```
     assign pcs_ready    = ~phy_rx_ack_q0_o[0] && rx_cdr_valid_q0_o[0] &&
     pma_rx0_get_lalign_o;
     ```
     For other configurations:
     ```
     assign pcs_ready    = ~phy_rx_ack_q0_o[0] && rx_cdr_valid_q0_o[0];
     ```
5. Normal transaction test
   - Sends and receives data in the parallel side of the MPPHY Module. Checker compares the received data with the transmitted data.
6. LMMI write-read check
   - This test covers writing to all writable bits of the MPPHY Module, reading back from that specific register, and comparing to the expected data.
7. Dynamic reconfiguration test
   - This test runs only when the *Enable Dynamic Reconfiguration* attribute is checked and performs dynamic reconfiguration from one profile to another. The reconfiguration is successful when the output clocks are active and the pcs_ready signal is asserted.

4.2. Loopback Test Support

4.2.1. No Loopback

When set to this configuration, MPPHY does not set any loopback register in the MPPMA and MPPCS blocks. To test the TX and RX paths, testbench connects the output serial TX data to the input serial RX data and performs the normal transaction test in the parallel side of the PHY.

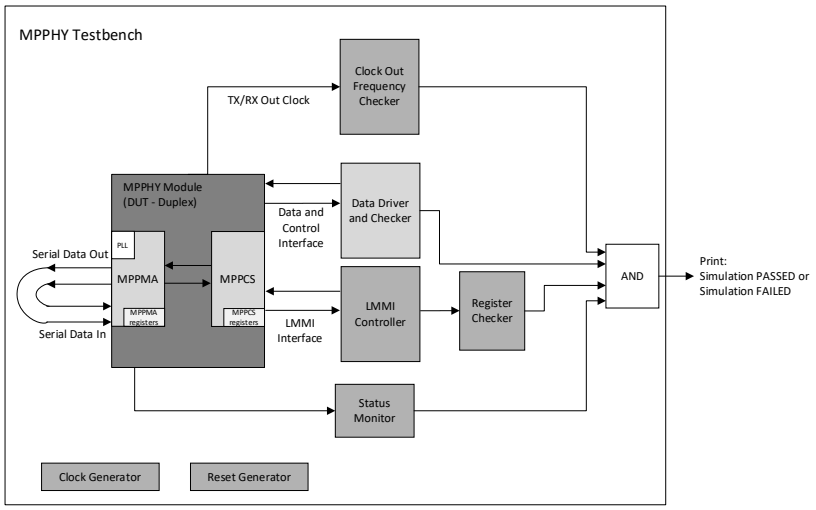

Figure 4.1. Test Setup for No Loopback

4.2.2. Near End Parallel Loopback and TX-to-RX Serial Loopback

Both loopback modes can use the recommended setup in the figure below. No connection is needed for the TX serial data out back to the RX serial data in. For more information regarding the loopback modes see Table 2.2.

This loopback test setup is supported by the testbench included in the MPPHY Module.

For register assertions, MPPHY asserts the near_lpbk_en register for Near End Parallel Loopback mode, and the phy_lane_tx2rx_ser_lb_en register for TX-to-RX Serial Loopback mode.

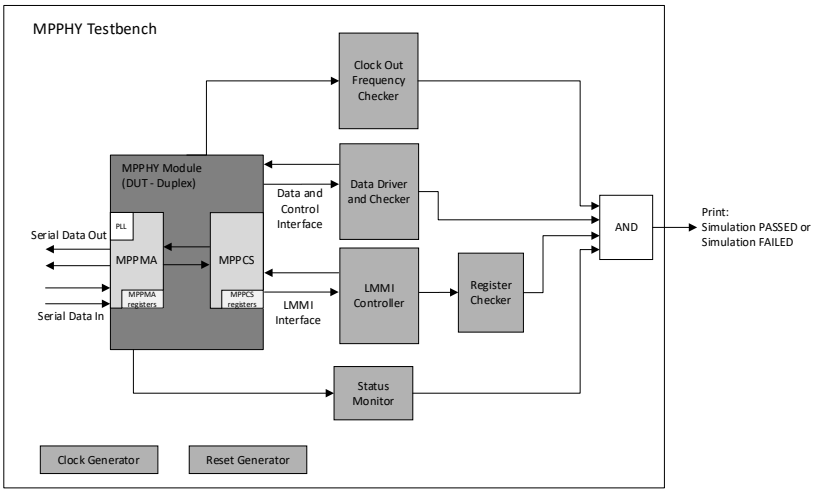

Figure 4.2. Test Setup for Near End Parallel Loopback

4.2.3. Far End Parallel Loopback and RX-to-TX Parallel Loopback

When set to this configuration, MPPHY asserts the Far End Parallel Loopback (far_lpbk_en) register. See Table 2.2 for details on the loopback modes. This loopback is supported by the testbench included in the MPPHY Module. You can use the recommended setup in the diagram below.

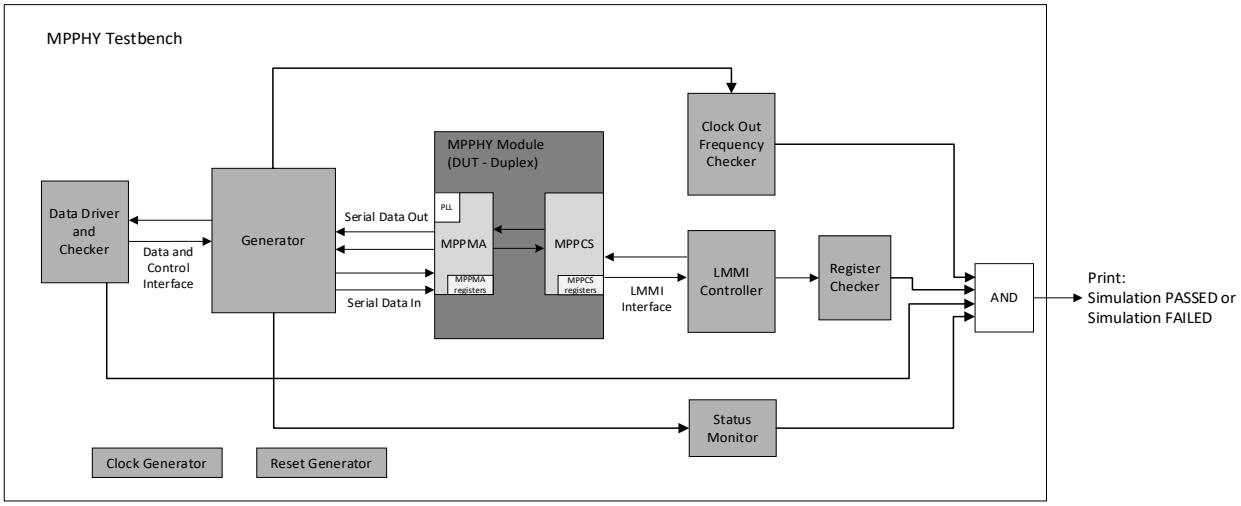

Figure 4.3. Test Setup for Far End Parallel Loopback

5. IP Generation and Evaluation

This section provides information on how to generate the MPPHY Module 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.

5.1. Licensing the IP

The MPPHY Module is provided at no additional cost with the Lattice Radiant software.

5.2. Generation and Synthesis

The Lattice Radiant software allows you to customize and generate modules and IPs and integrate them into the device architecture. The procedure for generating the MPPHY Module in Lattice Radiant software is described below.

To generate the MPPHY Module:

1. Create a new Lattice Radiant software project or open an existing project.
2. In the IP Catalog tab, double-click on **MPPHY** under **Module, Architecture_Modules** category.
3. The **Module/IP Block Wizard** opens as shown in Figure 5.1.
4. Enter values in the **Component name** and the **Create in** fields and click **Next**.

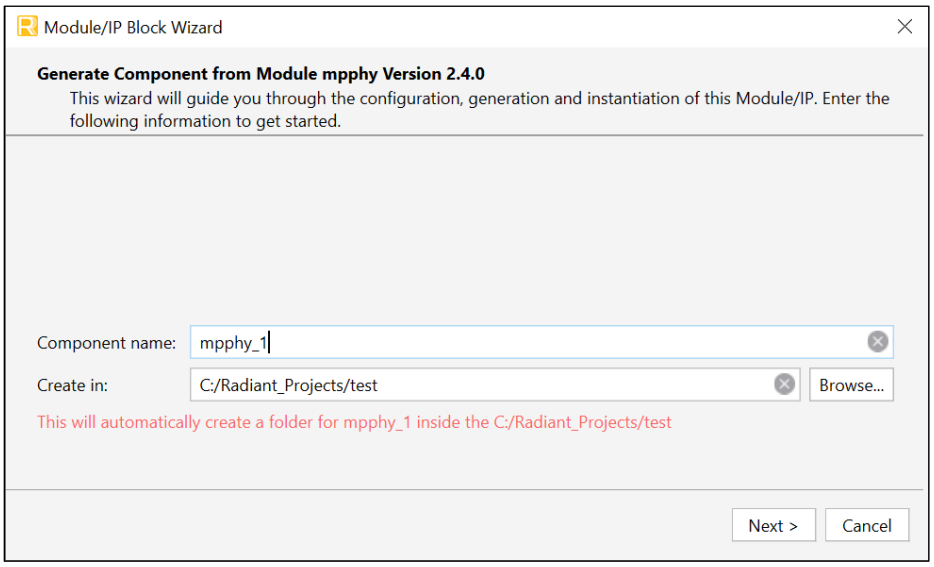

Figure 5.1. Module/IP Block Wizard

5. In the module dialog box of the **Module/IP Block Wizard** window, customize the selected MPPHY Module using the drop-down menus and check boxes. Refer to Figure 5.2 for a sample configuration. For configuration options, refer to the Attributes Summary section.

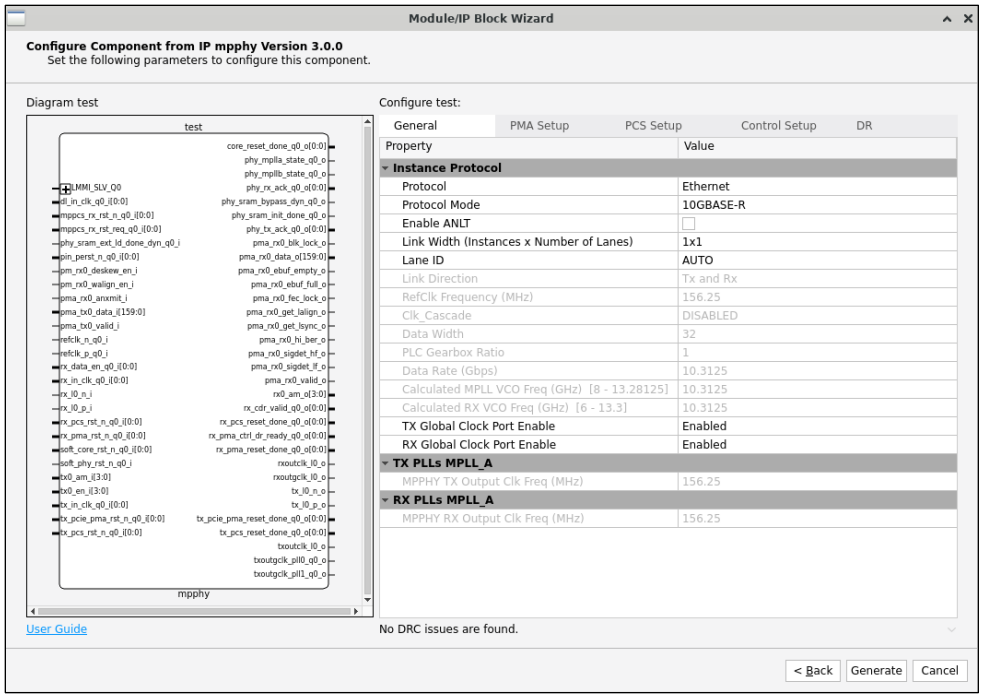

Figure 5.2. Configuring User Interface of the MPPHY Module

6. Click **Generate**. This shows the design block messages and results as shown in Figure 5.3.

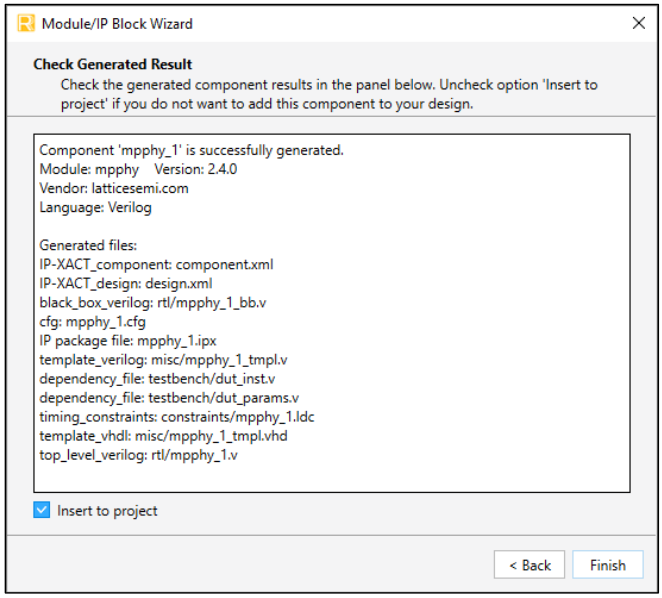

Figure 5.3. Check Generated Result

7. Click **Finish**. All the generated files are placed under the directory paths in the **Create in** and the **Component name** fields shown in Figure 5.1.

The generated MPPHY Module package includes the closed-box (<Component name>_bb.v) and instance templates (<Component name>_tmpl.v/vhd) that can be used to instantiate the core in a top-level design. An example RTL top-level reference source file (<Component name>.v) that can be used as an instantiation template for the module is also provided. You may also use this top-level reference as the starting template for the top-level for their complete design. The generated files are listed in Table 5.1.

Table 5.1. Generated File List

Attribute	Description
<Component name>.ipx	Contains the information on the files associated to the generated IP.
<Component name>.cfg	Contains the parameter values used in IP configuration.
component.xml	Contains the ipxact:component information of the IP.
design.xml	Documents the configuration parameters of the IP in IP-XACT 2014 format.
rtl/<Component name>.v	Provides an example RTL top file that instantiates the module.
rtl/<Component name>_bb.v	Provides the synthesis closed-box.
misc/<Component name>_tmpl.v misc /<Component name>_tmpl.vhd	Provides instance templates for the module.
prf/profile1/profile1.prf prf/profile1/profile1_profile[n]_diff_prf.prf	Provides the PRF file for the base profile (profile 1) and PRF diff from profile 1 to profile n (when n > 1).
prf/profile[n]/profile[n].prf prf/profile[n]/profile[n]_profile1_diff_prf.prf	Provides the standalone PRF file for profile n and PRF diff from profile 1 to profile n under profile n directory.

5.3. Running the Functional Simulation

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

To run functional simulation:

1. Click the  button located on the **Toolbar** to initiate the **Simulation Wizard** shown in Figure 5.4.

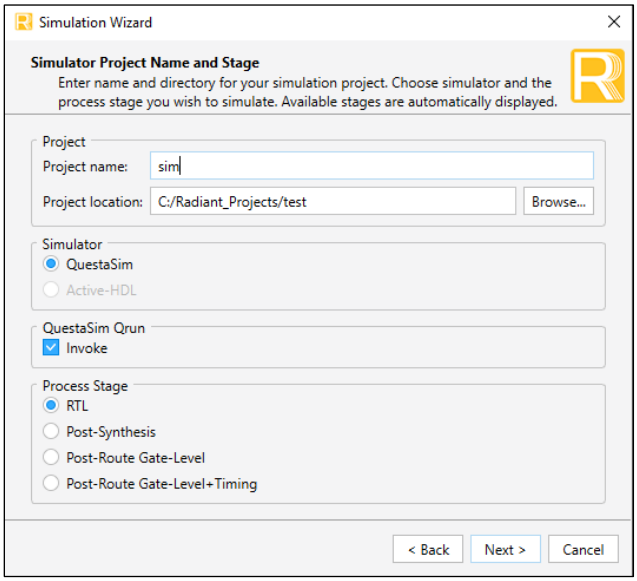

Figure 5.4. Simulation Wizard

2. Click **Next** to open the **Add and Reorder Source** window as shown in Figure 5.5.

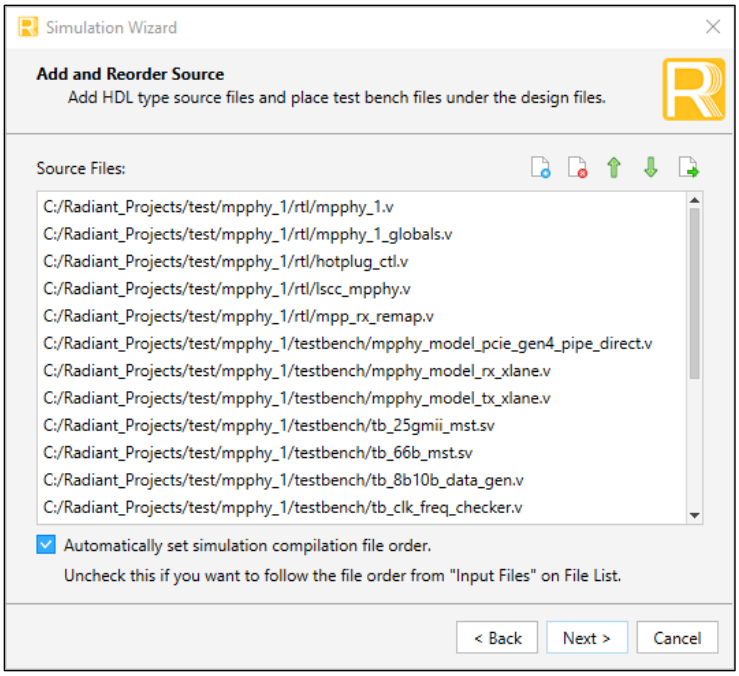

Figure 5.5. Adding and Reordering Source

3. Click **Next**. It shows the **Summary** window.
4. Click **Finish** to run the simulation.

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

The results of the simulation in our example are provided in Figure 5.6.

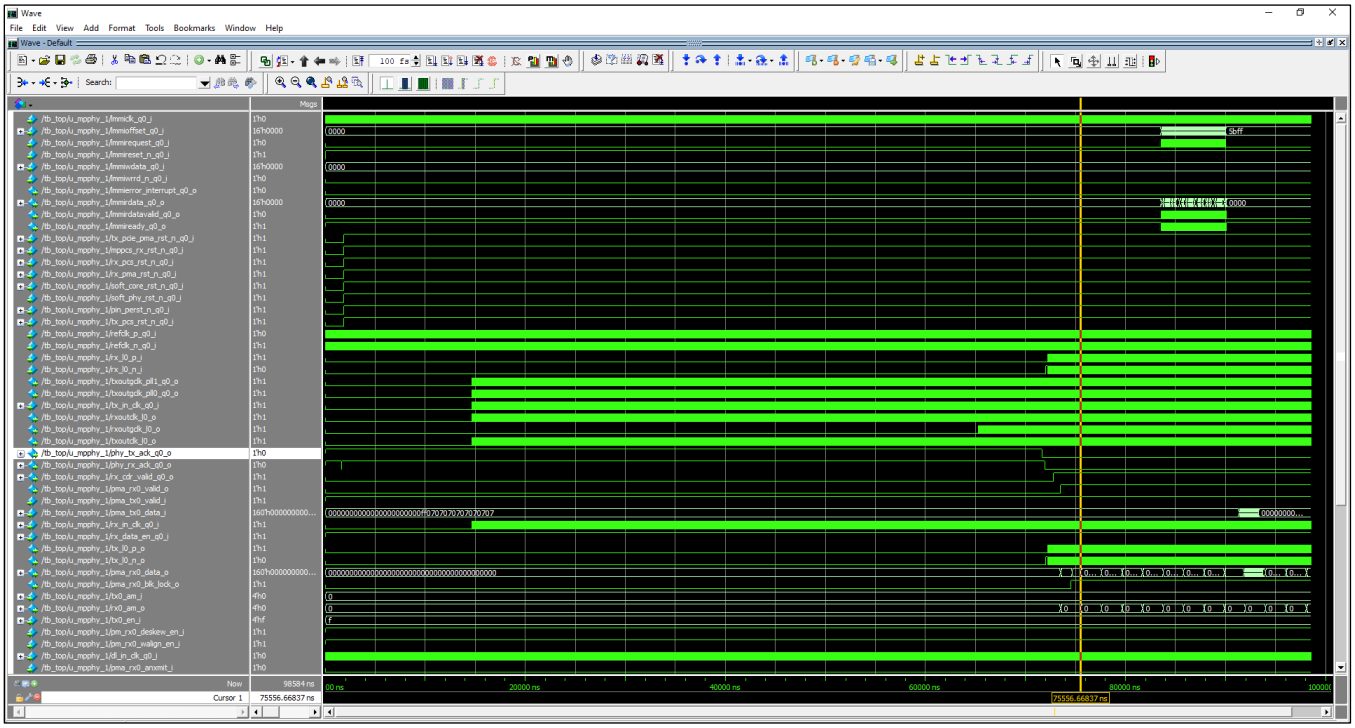

Figure 5.6. Simulation Waveform

5.3.1. Simulation Result

The simulation executes the test sequence described in the Test Sequence section. The figure below shows the expected completion message once the test sequence completes successfully.

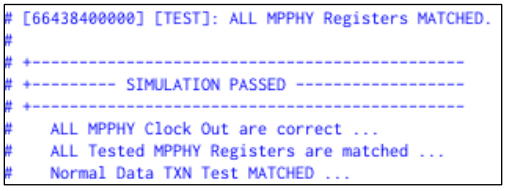

Figure 5.7. Simulation Completion Message

5.4. Constraining the IP

For the MPPHY Module, add clock constraints for the clock signals listed in the table below.

Table 5.2. Clock Signals to Be Constrained

Clock Signal	Frequency to Apply
txoutclk_l[n]_o, rxoutclk_l[n]_o	From IP GUI
txoutgclk_pll[r]_q[m]_o, rxoutgclk_l[n]_o	From IP GUI
refclk_p_q[m]_i, refclk_n_q[m]_i	From IP GUI
lmmiclk_q[m]_i	User-supplied
dl_in_clk_q[m]_i[NL-1:0]	User-supplied
Exclusively for asynchronous clocking mode: tx_in_clk_q[m]_i[NL-1:0], rx_in_clk_q[m]_i[NL-1:0]	Same frequency as txoutclk_l[n]_o and rxoutclk_l[n]_o

Notes:

1. [n] indicates lane or channel number.
2. [m] indicates Quad number.
3. [r] indicates PLL number, where PLL0 – PLLA and PLL1 – PLLB.
4. NL means Number of Lanes enabled in the Quad.

Note that the clock constraint must be applied to each clock signal per Quad per Lane per PLL. Refer to the example below.

```
#Example: 1 Quad, Lane=3, 1 PLL

#Reference Clock Frequency (148.5MHz)
create_clock -name {refclk_p_q0_i} -period 6.734 [get_ports {refclk_p_q0_i}]
create_clock -name {refclk_n_q0_i} -period 6.734 [get_ports {refclk_n_q0_i}]

#LMMI clock frequency (50MHz)
create_clock -name {lmmiclk_q0_i} -period 20 [get_ports {lmmiclk_q0_i}]

#Deterministic latency clock frequency (50MHz)
create_clock -name {dl_in_clk_q0_i_l0} -period 20 [get_ports {dl_in_clk_q0_i[0]}]

#The tx/rx_in_clk_q<m>_i clock constaints only needed for asynchronous mode.
#Asynchronous mode means the tx/rx_in_clk_q<>_i are not sourced from PMA output clocks
(tx/rxoutclk_l<n>_o)
```

```
create_clock -name {tx_in_clk_q0_i_l0} -period 13.468 [get_ports {tx_in_clk_q0_i[0]}]
create_clock -name {tx_in_clk_q0_i_l1} -period 13.468 [get_ports {tx_in_clk_q0_i[1]}]
create_clock -name {tx_in_clk_q0_i_l2} -period 13.468 [get_ports {tx_in_clk_q0_i[2]}]

create_clock -name {rx_in_clk_q0_i_l0} -period 13.468 [get_ports {rx_in_clk_q0_i[0]}]
create_clock -name {rx_in_clk_q0_i_l1} -period 13.468 [get_ports {rx_in_clk_q0_i[1]}]
create_clock -name {rx_in_clk_q0_i_l2} -period 13.468 [get_ports {rx_in_clk_q0_i[2]}]

#Due to SW limitations, users need to manually add clock constraint for the TX/RX PMA output
clocks.
create_clock -name {txoutclk_l0_o} -period 13.468 [get_pins {lscc_mpphy_inst/txoutclk_l0_o}]
create_clock -name {txoutclk_l1_o} -period 13.468 [get_pins {lscc_mpphy_inst/txoutclk_l1_o}]
create_clock -name {txoutclk_l2_o} -period 13.468 [get_pins {lscc_mpphy_inst/txoutclk_l2_o}]
create_clock -name {txgclk_pll0_q0_o} -period 13.468 [get_pins
{lscc_mpphy_inst/txoutgclk_pll0_q0_o}]

create_clock -name {rxoutclk_l0_o} -period 13.468 [get_pins {lscc_mpphy_inst/rxoutclk_l0_o}]
create_clock -name {rxoutclk_l1_o} -period 13.468 [get_pins {lscc_mpphy_inst/rxoutclk_l1_o}]
create_clock -name {rxoutclk_l2_o} -period 13.468 [get_pins {lscc_mpphy_inst/rxoutclk_l2_o}]
create_clock -name {rxoutgclk_l0_o} -period 13.468 [get_pins {lscc_mpphy_inst/rxoutgclk_l0_o}]
create_clock -name {rxoutgclk_l1_o} -period 13.468 [get_pins {lscc_mpphy_inst/rxoutgclk_l1_o}]
create_clock -name {rxoutgclk_l2_o} -period 13.468 [get_pins {lscc_mpphy_inst/rxoutgclk_l2_o}]
```

The example above assumes that the clock sources are external and direct from the FPGA port. Use get_pins and create_generate_clock if the clock source is within the FPGA. For more information on timing constraints, refer to the Lattice Radiant Timing Constraints Methodology (FPGA-AN-02059).

6. Known Limitations

6.1. Hold Time Violation in Lattice Avant Designs when Using the MPPHY Module

The Lattice Radiant software automatically inserts LMMI arbiter logic to enable access to the MPPHY LMMI interface. This implementation introduces a LUT4-based clock multiplexer in the LMMI clock path to enable Reveal Controller access to the MPPHY LMMI interface. The additional LUT4 logic can introduce significant clock skew, which may result in hold time violations in user designs.

This issue applies to the following Lattice IPs that incorporate the MPPHY module:

- DisplayPort IP
- JESD204B IP
- PCIe x8 IP
- Ethernet IPs (Tri-Speed, 2.5G, 5G, 10G, and 25G)

As a workaround, try running your design using different Lattice Radiant tool settings such as *Impose Hold Timing Correction* or seeds. For further assistance, contact Lattice support.

For the latest update, refer to the Lattice Knowledge Base article.

References

- MPPHY Module Release Notes (FPGA-RN-02043)
- Lattice Radiant Timing Constraints Methodology (FPGA-AN-02059)
- Lattice Avant SERDES/PCS User Guide (FPGA-TN-02313)
- Avant-G web page
- Avant-X web page
- Certus-N2 web page
- Lattice Radiant Software web page
- Lattice Propel Design Environment web page
- Lattice Insights for Lattice Semiconductor training courses and learning plan

Technical Support Assistance

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

Note: The latest changes are applicable to the latest silicon sample. To obtain a previous version of the document, submit a technical support case.

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 1.7, IP v3.0.0, June 2026

Section	Change Summary
All	Performed minor formatting and editorial edits.
Acronyms in This Document	Updated list of abbreviations.

Introduction	• Updated Table 1.1. Quick Facts as follows:
	  • Updated Lattice Avant devices.
	  • Added Certus-N2 devices.
	  • Updated IP version.
	• Updated Table 1.2. MPPHY IP Support Readiness as follows:
	  • Added the SDI protocol.
	  • Added the 1000GBASE-KX_10GHz, 10GBASE-R (ANLT = Enabled), and 25GBASE-R (ANLT = Enabled) Ethernet protocol modes.
	  • Updated the modes for Generic, JESD204B, and JESD204C protocols.
	  • Updated protocol PIPE Direct to PCIe PIPE Direct.
	  • Removed *Simulation Validated* and *Hardware Validated* columns.

Functional Description	• Updated the ports to be shared for quad merging in the Lane Merging section.
	• Updated the Dynamic Reconfiguration section and added the following subsections:
	  • Scope
	  • PHY Reconfiguration File (PRF)
	  • Dynamic Reconfiguration Profile
	  • Dynamic Reconfiguration Sequence
	• Updated Table 2.12. MPPHY Module Signal Description as follows:
	  • Removed the pma_rx[n]_anxmit_i signal to align with IEEE 802.3 requirements.
	  • Updated signal name from refclkp_q[m]_i to refclk_p_q[m]_i, refclkn_q[m]_i to refclk_n_q[m]_i, pipe_lane[n]_if_width_0_i to pipe_lane[n]_if_width_[p]_i, pipe_lane[n]_m2p_messagebus_0_i to pipe_lane[n]_m2p_messagebus_[p]_i, pipe_lane[n]_powerdown_0_i to pipe_lane[n]_powerdown_[p]_i, pipe_lane[n]_rate_0_i to pipe_lane[n]_rate_[p]_i, pipe_lane[n]_p2m_messagebus_0_o to pipe_lane[n]_p2m_messagebus_[p]_o, pipe_rx[n]_datak_0_o to pipe_rx[n]_datak_[p]_o, pipe_rx[n]_status_0_o to pipe_rx[n]_status_[p]_o, pipe_rx[n]_syncheader_0_o to pipe_rx[n]_syncheader_[p]_o, pipe_tx[n]_datak_0_i to pipe_tx[n]_datak_[p]_i, pipe_tx[n]_syncheader_0_i to pipe_tx[n]_syncheader_[p]_i.
	  • Added signals: pipe_tx[n]_*, rx_pma_ctrl_dr_ready_q[m]_o, and core_reset_done_q[m]_o.
	  • Updated the dl_in_clk_q[m]_i signal.
	  • Added a note on [p].
	• Updated Table 2.22. Attributes Table as follows:
	  • Updated attributes: *Protocol*, *Calculated MPLL VCO Freq (GHz)*, *TX Main Cursor*, *TX Pre Cursor*, *TX Post Cursor*.
	  • Added attributes: *Fast Simulation Mode*, *Enable ANLT*, *Clk_Cascade*, *TX Amplitude*, *RX Adapt Mode*, *Invert TX Data Polarity*, *Invert RX Data Polarity*, *8b10b Decoder*, *Clock Frequency Compensation*, *CTC FIFO*, *Skip Pattern Mask Code*, *Skip Pattern Code*, *Skip Pattern Length*, *Primary Skip Pattern*, *Secondary Skip Pattern*, *Lane Alignment*, *Input Data Code Mode*, *Lane Alignment Pattern Length*, *Maximum Lane-to-Lane Skew*, *Primary Lane Alignment Pattern*, *Secondary Lane Alignment*, *Secondary Lane Alignment Pattern*, *Lane Alignment Pattern Mask Code*.
	  • Changed attribute name from *TX Amplitude Control* to *TX Main Cursor*.
	  • Updated the default value for the *Use 'Sync_det' FSM* attribute.
	  • Added dependency on the *Use 'Sync_det' FSM* attribute.
	  • Added attributes for dynamic reconfiguration.

Section	Change Summary

	• Removed attributes that are not user editable.
	• Updated Table 2.23. Attributes Descriptions as follows:
	  • Added attributes: *Fast Simulation Mode*, *Enable ANLT*, *Lane ID*, *Clk_Cascade*, *Data Width*, *TX Amplitude*, *RX Adapt Mode*, *Invert TX Data Polarity*, *Invert RX Data Polarity*.
	  • Updated attribute name from *TX Amplitude Control* to *TX Main Cursor*.
	  • Updated attribute name from *RX Adaptive Control* to *Rx Adapt Mode*.
	  • Added attributes for dynamic reconfiguration.
	  • Removed attributes that are not user editable.
	• Removed the *Protocol Mode Table*.
	• Updated the following sections:
	  • Cold Boot Reset
	  • Warm Reset

Testbench Simulation Support	Added the dynamic reconfiguration test in the Test Sequence section.
IP Generation and Evaluation	• Updated Figure 5.2. Configuring User Interface of the MPPHY Module. • Added attributes for PRF files in Table 5.1. Generated File List. • Added the Simulation Result section. • Updated the Constraining the IP section.
Known Limitations	Added this section.
References	Updated references.

Revision 1.6, IP v2.6.0, December 2025

Section	Change Summary
All	• Renamed document from *Lattice Avant-G/X MPPHY Module* to *MPPHY Module*. • Added a note on IP version in Quick Facts and *Revision History* sections.
Acronyms in This Document	Updated list of acronyms.

Introduction	• Updated description to add support for Lattice Nexus 2 devices in the Introduction section.
	• Updated Table 1.1. Quick Facts as follows:
	  • Added LAV-AT-G30, LAV-AT-X30, and Lattice Nexus 2 devices.
	  • Added IP version.
	  • Removed earlier IP versions in the Lattice Implementation row.
	• Updated Table 1.2. MPPHY IP Support Readiness as follows:
	  • Added support for Nexus 2 devices.
	  • Added 2.7 Gbps (Oversampling PMA 8.1G) data rate for DP/eDP protocol.
	  • Added support for other data rates for JESD204B and JESD204C protocols.

Functional Description	• Added support for Nexus 2 devices in the following sections:
	  • Lane Merging
	  • MPPHY Placement
	• Added the Dynamic Reconfiguration subsection under the Overview section.
	• Updated Table 2.23. Attributes Table as follows:
	  • Updated the *Calculated MPLL VCO Freq (GHz)* attribute.
	  • Added the *PMA Control RX Adaptation Mode* attribute.
	• Updated Table 2.25. Protocol Mode Table as follows:
	  • Added the DP_eDP_2.7G (Oversampling PMA 8.1G) protocol mode for DP_eDP protocol.
	  • Removed the JESD204B and JESD204C protocols
	• Added the LMMI Arbiter subsection under the Operation Details section.

Testbench Simulation Support	• Updated that loopback is supported by the testbench in the Far End Parallel Loopback and RX-to-TX Parallel Loopback section. • Updated Figure 4.3. Test Setup for Far End Parallel Loopback.

Section	Change Summary
IP Generation and Evaluation	• Added a note on IP version in GUI in the IP Generation and Evaluation section. • Updated the Licensing the IP section.
References	Updated references.

Revision 1.5, IP v2.5.0, October 2025

Section	Change Summary
All	Performed minor formatting and editorial edits.
Acronyms in This Document	Updated list of acronyms.

Introduction	• Added IP version in Table 1.1. Quick Facts.
	• Updated Table 1.2. MPPHY IP Support Readiness as follows:
	  • Added Simulation Validation.
	  • Added the 25GAUI protocol mode.

Functional Description	• Added the following subsections under the Overview section:
	  • PCS Bypass
	  • Lane Merging
	• Updated Figure 2.7. MPPCS and MPPMA Loopback Diagram.
	• Added RX-to-TX Parallel Loopback in Table 2.3. MPPMA Loopback Description.
	• Added the PCS Bypass subsection under the Data Bus Mapping section.
	• Updated Table 2.14. Attributes Table as follows:
	  • Updated values for the *Protocol*, *Protocol Mode*, *RefClk Frequency (MHz)*, *Data Rate (Gbps)*, *Data Width*, and *Loopback* attributes.
	  • Added the *RX Adaptive Control* attribute.
	  • Updated Dependency on Other Attributes.
	• Updated Table 2.15. Attributes Descriptions as follows:
	  • Update the description for the *Data Rate*, *Calculated RX VCO Freq*, *Data Width*, *PLC Gearbox Ratio*, and *MPLLA/B TX Clock Divider* attributes.
	  • Added the *RX Adapt Mode* attribute.
	  • Changed attribute name from *PCS Loopback* to *Loopback*.
	• Added Table 2.16. Protocol Mode Table.
	• Updated the corresponding variable in the equation for the GUI attributes in the following sections:
	  • Clock Settings
	  • TX Clock Settings
	  • RX Clock Settings
	• Added the Supported Continuous Data Rate Range section.
	• Added note on the constraints for MPPHY merging and assignment in the MPPHY Placement section.
	• Added the Quad-Quad Connection section.

Testbench Simulation Support	• Mentioned that loopback test setup is supported by the testbench in the Near End Parallel Loopback and TX-to-RX Serial Loopback section. • Added support for RX-to-TX parallel loopback and renamed the section from *Far End Parallel Loopback* to Far End Parallel Loopback and RX-to-TX Parallel Loopback.

Revision 1.4, IP v2.4.0, June 2025

Section	Change Summary

Introduction	• Updated Table 1.1. Quick Facts as follows:
	  • Renamed *Supported FPGA Families* to *Supported Devices*.
	  • Removed the *Targeted Devices* row.
	  • Added IP version.
	• Added the IP Support Summary section.

Functional Description	• Updated the protocols supported for the PCS enabled mode in the Overview section.

Section	Change Summary
	• Renamed figure from *MPPCS Loopback Diagram* to Figure 2.7. MPPCS and MPPMA Loopback Diagram, and updated figure. • Removed figure: *MPPMA Loopback Diagram*. • Added values to the *Protocol Mode* attribute in Table 2.12. Attributes Table.
Register Description	Added this section.
IP Generation and Evaluation	Updated the following figures: • Figure 5.1. Module/IP Block Wizard • Figure 5.2. Configuring User Interface of MPPHY Module • Figure 5.3. Check Generated Result

Revision 1.3, IP v2.3.0, March 2025

Section	Change Summary
Introduction	Added IP version in Table 1.1. Quick Facts.

Functional Description	• Removed support for BASE-KR in the 64B66B Decoder section.
	• Updated the Loopbacks section.
	  • Added MPPMA loopback mode.
	  • Updated Table 2.2. MPPCS Loopback Description.
	  • Added Figure 2.8. MPPMA Loopback Diagram.
	  • Added Table 2.3. MPPMA Loopback Description.
	• Updated description for the *pma_rx[n]_anxmit_i* signal in Table 2.4. MPPHY Module Signal Description.
	• Updated the values for the *Protocol Mode*, *Lane ID*, and *PCS Loopback* attributes in Table 2.12. Attributes Table.

Testbench Simulation Support	• Changed *master* to *controller* in the following figures:
	  • Figure 3.1. Test Setup for No Loopback
	  • Figure 3.2. Test Setup for Near End Parallel Loopback
	  • Figure 3.3. Test Setup for Far End Parallel Loopback
	• Renamed section title from *Near End Parallel Loopback* to Near End Parallel Loopback and TX-to-RX Serial Loopback and updated the section.

IP Generation and Evaluation	Updated the following figures: • Figure 4.1. Module/IP Block Wizard • Figure 4.2. Configuring User Interface of MPPHY Module • Figure 4.3. Check Generated Result • Figure 4.4. Simulation Wizard • Figure 4.5. Adding and Reordering Source • Figure 4.6. Simulation Waveform

Revision 1.2, IP v2.2.0, December 2024

Section	Change Summary
Introduction	Updated Table 1.1. Quick Facts as follows: • Added IP changes. • Updated IP version.

Functional Description	• Added the supported data protocols for the PCS enabled mode in the Overview section.
	• Updated Table 2.3. MPPHY Module Signal Description as follows:
	  • Updated the width for the *tx_l[n]_p_o*, *tx_l[n]_n_o*, *rx_l[n]_p_i*, *rx_l[n]_n_i*, *tx[n]_am_i*, *tx[n]_en_i*, *rx[n]_am_o*, and *rx_valid_q[m]_o* signals.
	  • Updated signal names for the *rx_pcs_reset_done_q[m]_o*, *rx_pma_reset_done_q[m]_o*, *tx_pcie_pma_reset_done_q[m]_o*, *tx_pcs_reset_done_q[m]_o*, and *rx_valid_q[m]_o* signals.
	  • Updated the note about NL.
	• Updated Table 2.11. Attributes Table as follows:

Section	Change Summary
	• Updated the *Protocol*, *Protocol Mode*, *Link Width (Instances x Number of Lanes)* attributes. • Added the *Lane ID* attribute. • Added the MPPHY Placement section.
Testbench Simulation Support	• Updated the steps for PHY initialization test and for waiting of pcs_ready in the Test Sequence section. • Updated the Near End Parallel Loopback (near_lpbk_en) register and Far End Parallel Loopback (far_lpbk_en) register, and added reference for details on the loopback modes in the Near End Parallel Loopback and Far End Parallel Loopback sections.
References	Added link to the IP release notes.

Revision 1.1, September 2024

Section	Change Summary
All	• Renamed document from Lattice Avant-G/X MPPHY Module - Lattice Radiant Software -to Lattice Avant-G/X MPPHY Module. • Changed SerDes to SERDES. • Performed minor formatting and typo edits.
Acronyms in This Document	• Removed definition for BIST and CPRI. • Added definition for CDR, CTLE, DFE, IP, SSC, and VGA.
Introduction	• Updated software version in Table 1.1. Quick Facts. • Updated the key features of the MPPHY Module in the Features section.

Functional Description	• Updated the Overview section.
	• Removed the following subsections under the Overview section:
	  • *Transmit Path*
	  • *Receive Path*
	  • *PCS 64B66B Transmit-Receive Path*
	  • *Reset Control Scheme with RCM Enable*
	• Added the following subsections under the Overview section:
	  • 8b10b PCS
	  • 64b66b PCS
	• Updated the Loopbacks section.
	• Updated Table 2.3. MPPHY Module Signal Description.
	• Added the MPP Reset Scheme section.
	• Updated the Data Bus Mapping section and the subsections.
	  • Removed the *64b66b Protocols* section.
	  • Added the 8b10b PCS and 64b66b PCS sections.
	• Updated Table 2.11. Attributes Table.
	• Updated Table 2.12. Attributes Descriptions.
	• Renamed section from *PHY Initialization Sequence using test simulation model (RCM-disabled)* to Reset Sequence and added the following subsections:
	  • Cold Boot Reset
	  • Warm Reset
	• Added the Clock Settings section and the following subsections:
	  • TX Clock Settings
	  • RX Clock Settings
	• Removed the *PHY Reset Sequence with RCM Enabled* section and the following subsections:
	  • *PHY Initialization/Cold Reset Release*
	  • *Warm Reset (PMA reset)*
	• Updated the Data Transmission section.

Testbench Simulation Support	Added this section.
IP Generation and	• Changed *black box* to *closed-box* in the Generation and Synthesis section.

Section	Change Summary

Evaluation	• Updated the following diagrams:
	  • Figure 4.1. Module/IP Block Wizard
	  • Figure 4.2. Configuring User Interface of MPPHY Module
	  • Figure 4.3. Check Generated Result
	  • Figure 4.4. Simulation Wizard
	  • Figure 4.5. Adding and Reordering Source
	  • Figure 4.6. Simulation Waveform

References	Updated references.
Technical Support Assistance	Added a note to indicate that the latest changes are applicable to the latest silicon sample, and a previous version of the document can be requested if required.

Revision 1.0, December 2023

Section	Change Summary
All	• Replaced LAV-AT-500G with LAV-AT-G70 • Replaced LAV-AT-500X with LAV-AT-X70
Disclaimers	Updated with the latest disclaimers.
Inclusive Language	Newly added section.
Functional Description	• Updated Figure 2.1. MPPHY Module Block Diagram to Figure 2.4. PCS 64B66B TX/RX Path. • Added the Reset Control Scheme with RCM Enable section. • Updated Table 2.5. RCM IP Reset Scheme Support. • Removed table notes no.2 and no.3 from Table 2.5. RCM IP Reset Scheme Support. • Updated Table 2.6. MPPHY Module Signal Description. • Updated the title of Figure 2.7 HSSI PHY Initialization Sequence. • Updated Table 2.9. Attributes Table and Table 2.10. Attributes Descriptions to add RCM Enable and User Interface Type. • Updated Table 2.11. Expected rxvalid_o assertion for Disabled RCM block in RTL simulation to Table 2.13. Expected rxvalid_o assertion for Enabled RCM block using AXI Interface in RTL simulation. • Removed the Warm Reset (PCS Reset) (internal use only) section. • Updated Figure 2.7 HSSI PHY Initialization Sequence and Figure 2.8. PMA Tx and Rx Reset. Added the PHY Reset Sequence with RCM Enabled section.
IP Generation and Evaluation	Added the Constraining the IP section
References	Newly added section.

Revision 0.80, June 2023

Section	Change Summary
All	Preliminary Release

www.latticesemi.com