LogiCORE IP 10-Gigabit Ethernet PCS/PMA v4.0

Product Guide for Vivado Design Suite

PG068 October 2, 2013

XILINX®

XILINX

Table of Contents

Appendix E: Additional Resources

Introduction

The Xilinx LogiCORE™ IP 10-Gigabit Ethernet Physical Coding Sublayer/Physical Medium Attachment (PCS/PMA) core, also known as 10GBASE-R in this document, forms a seamless interface between the Xilinx 10-Gigabit Ethernet Media Access Controller (MAC) and a 10 Gb/s-capable PHY, enabling the design of high-speed Ethernet systems and subsystems.

10GBASE-KR is supported on 7 series GTH transceivers. 10GBASE-R is supported on Zynq®-7000 All Programmable SoCs, Virtex®-7, and Kintex®-7 devices containing GTX and GTH transceivers. 10GBASE-KR for backplane applications has additional features over 10GBASE-R which consist of Link Training and optional Forward Error Correction (FEC) and Auto-Negotiation (AN).

Features

- Designed to 10-Gigabit Ethernet specification IEEE 802.3-2012 clause 49, 72, 73, 74
- Optional Management Data Interface (MDIO) interface to manage PCS/PMA registers according to specification IEEE 802.3-2012 clause 45
- Delivered through the Xilinx Vivado® IP catalog
- Supports 10GBASE-SR, -LR and -ER optical links in Zynq-7000, Virtex-7 and Kintex-7 devices (LAN mode only)
- Supports 10GBASE-KR backplane links in Virtex-7 devices, including Auto-Negotiation (AN), Training and Forward Error Correction (FEC)
- SDR 10-Gigabit Ethernet Media Independent Interface (XGMII) connects seamlessly to the Xilinx 10G Ethernet MAC

LogiCORE IP Facts Table	
Core Specifics	
Supported Device Family(1) (2)	10GBASE-R: Zynq-7000, Virtex-7, Kintex-7(3)
	10GBASE-KR: Virtex-7(4)
Supported User Interfaces	MDIO, XGMII
Resources	See Table 2-1 and Table 2-2.
Provided with Core	
Design Files	Encrypted RTL
Example Design	Verilog and VHDL
Test Bench	Verilog and VHDL
Constraints File	Xilinx Design Constraint (XDC)
Simulation Model	Verilog or VHDL source HDL Model
Supported S/W Driver	N/A
Tested Design Flows(5)	
Design Entry	Vivado Design Suite
Simulation	For supported simulators, see the Xilinx Design Tools: Release Notes Guide.
Synthesis	Vivado Synthesis
Support	
Provided by Xilinx @ www.xilinx.com/support	

Notes:

1. For a complete list of supported devices, see the Vivado IP catalog.
2. For 10GBASE-KR channel analysis, contact your local Xilinx representative.
3. –2, –2L or –3.
4. GTHE2 transceivers only
5. For the supported versions of the tools, see the Xilinx Design Tools: Release Notes Guide.

Overview

10GBASE-R/KR is a 10 Gb/s serial interface. It is intended to provide the Physical Coding Sublayer (PCS) and Physical Medium Attachment (PMA) functionality between the 10-Gigabit Media Independent Interface (XGMII) interface on a Ten Gigabit Ethernet Media Access Controller (MAC) and a Ten Gigabit Ethernet network physical-side interface (PHY).

What distinguishes the 10GBASE-KR core from the 10GBASE-R core is that the 10GBASE-KR core includes a Link Training block as well as optional Auto-Negotiation (AN) and Forward Error Correction (FEC) features, all to help support a 10 Gb/s data stream across a backplane.

10GBASE-R

For Zynq®-7000, Virtex®-7 and Kintex®-7 devices, all of the PCS and Management blocks illustrated are implemented in logic, except for part of the Gearbox and SerDes. Figure 1-1 shows the architecture.

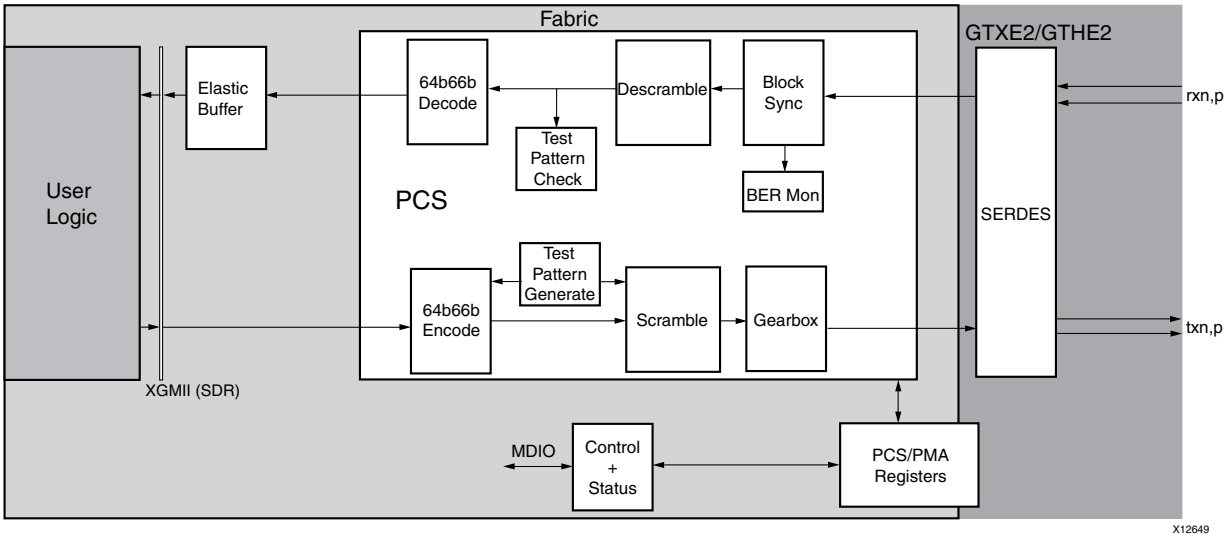

Figure 1-1: **Implementation of the 10-Gigabit Ethernet PCS/PMA (BASE-R) Core**

The major functional blocks of the core include the following:

- XGMII interface, designed for simple attachment of 10-Gigabit Ethernet MAC
- Transmit path, including Scrambler, 64B/66b Encoder and Gearbox
- Receive path, including Block Synchronization, Descrambler, Decoder and BER (Bit Error Rate) monitor
- Elastic Buffer in the receive datapath. The elastic buffer is 32 words deep (1 word = 64bits data + 8 control). If the buffer empties, local fault codes are inserted instead of data. This allows you to collect up to 64 clock correction (CC) sequences before the buffer overflows (and words are dropped). The buffer normally fills up to one half and then deletes CC sequences when over half full, and inserts CC sequences when under one half full. So from a half-full state, you can (conservatively) accept an extra 360 KB of data (that is, receiving at +200 ppm) without dropping any. From a half-full state you can cope with another 360 KB of data without inserting local faults (for –200 ppm).
- Test Pattern Generation and Checking
- Serial interface to optics
- Management registers (PCS/PMA) with optional MDIO interface

10GBASE-KR

Figure 1-2 illustrates a block diagram of the 10-Gigabit Ethernet PCS/PMA (BASE-KR) core implementation. The major functional blocks of the core include the following:

- XGMII interface, designed for simple attachment of 10-Gigabit Ethernet MAC
- Transmit path, including Scrambler, 64B/66B Encoder, FEC, AN and Training
- Receive path, including Block Synchronization, Descrambler, Decoder and BER (Bit Error Rate) monitor, FEC, AN and Training
- Elastic Buffer in the receive datapath. The elastic buffer is 32 words deep (1 word = 64bits data + 8 control). If the buffer empties, local fault codes are inserted instead of data. This allows you to collect up to 64 clock correction (CC) sequences before the buffer overflows (and words are dropped). The buffer normally fills up to one half and then deletes CC sequences when over half full, and inserts CC sequences when under one half full. So from a half-full state, you can (conservatively) accept an extra 360 KB of data (that is, receiving at +200 ppm) without dropping any. From a half-full state you can cope with another 360 KB of data without inserting local faults (for –200 ppm).
- Test Pattern Generation and Checking
- Serial interface to backplane connector
- Management registers (PCS/PMA) with optional MDIO interface

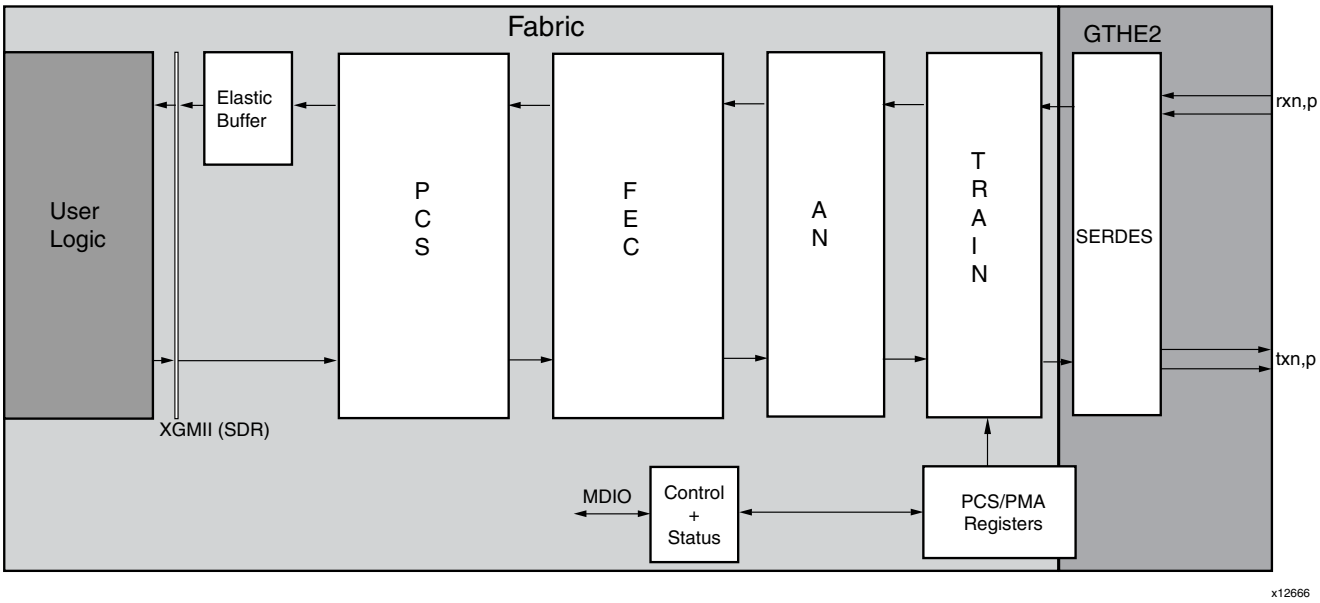

Figure 1-2: **Implementation of the BASE-KR Core**

Recommended Design Experience

Although the core is a fully-verified solution, the challenge associated with implementing a complete design varies depending on the configuration and functionality of the application. For best results, previous experience building high performance, pipelined Field Programmable Gate Array (FPGA) designs using Xilinx implementation software and Xilinx Design Constraints (XDC) is recommended.

Contact your local Xilinx representative for a closer review and estimation for your specific requirements.

Applications

Figure 1-3 shows a typical Ethernet system architecture and the 10-Gigabit Ethernet PCS/PMA core within it. The MAC and all the blocks to the right are defined in Ethernet IEEE specifications [Ref 1][Ref 2].

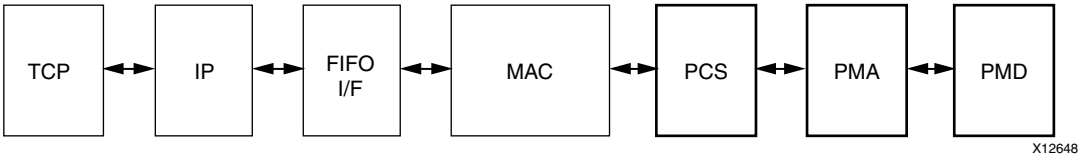

Figure 1-3: **Typical Ethernet System Architecture**

Figure 1-4 shows the 10-Gigabit Ethernet PCS/PMA core connected on one side to a 10-Gigabit MAC and on the other to an optical module (BASE-R) or backplane (BASE-KR) using a serial interface. The optional WAN Interface Sublayer (WIS) part of the 10GBASE-R standard is not implemented in this core.

The 10-Gigabit Ethernet PCS/PMA core is designed to be attached to the Xilinx IP 10-Gigabit Ethernet MAC core over XGMII. More details are provided in Chapter 3, Designing with the Core.

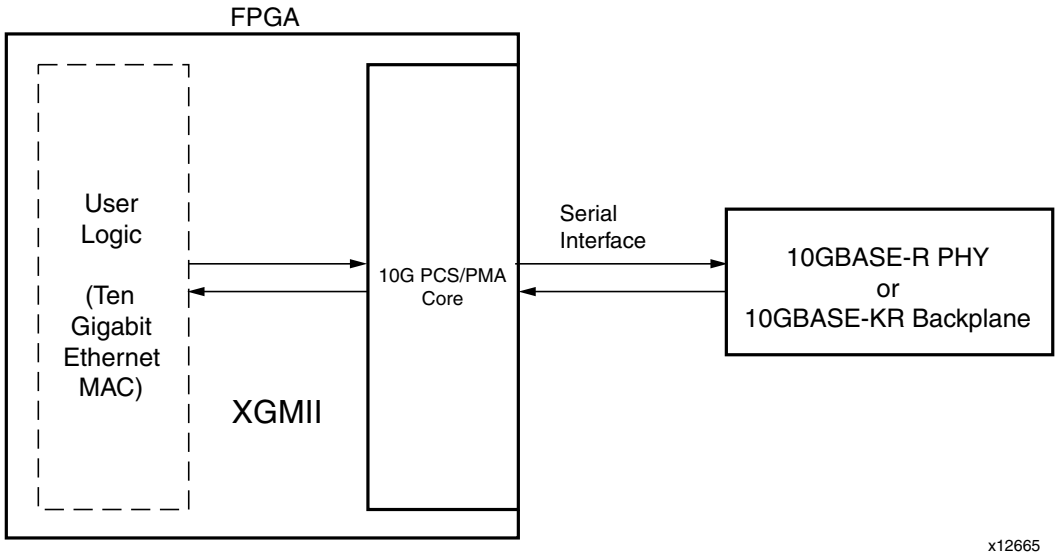

Figure 1-4: **Core Connected to MAC Core Using XGMII Interface**

Unsupported Features

While the Training Protocol is supported natively by the core, no logic is provided that controls the far-end transmitter adaptation based on analysis of the received signal quality. This is because extensive testing has shown that to be unnecessary.

However, an interface is provided on the core that allows access to all core registers and to the DRP port on the transceiver. You can employ this interface to implement your own Training Algorithm for 10GBASE-KR, if required.

Licensing and Ordering Information

License Checkers

If the IP requires a license key, the key must be verified. The Vivado® design tools have several license check points for gating licensed IP through the flow. If the license check succeeds, the IP can continue generation. Otherwise, generation halts with error. License checkpoints are enforced by the following tools:

- Vivado flow: Vivado Synthesis, Vivado Implementation, write_bitstream (Tcl Console command)

IMPORTANT: *IP license level is ignored at checkpoints. The test confirms a valid license exists. It does not check IP license level.*

License Types

10 Gigabit Ethernet PCS/PMA (10GBASE-R)

This Xilinx LogiCORE™ IP module is provided at no additional cost with the Xilinx Vivado® Design Suite under the terms of the Xilinx End User License. Information about this and other Xilinx LogiCORE IP modules is available at the Xilinx Intellectual Property page. For information about pricing and availability of other Xilinx LogiCORE IP modules and tools, contact your local Xilinx sales representative.

For more information, visit the 10 Gigabit Ethernet PCS/PMA (10GBASE-R) product web page.

10 Gigabit Ethernet PCS/PMA with FEC/Auto-Negotiation (10GBASE-KR)

This Xilinx LogiCORE IP module is provided under the terms of the Xilinx Core Project License Agreement.The module is shipped as part of the Vivado Design Suite. For full access to all core functionalities in simulation and in hardware, you must purchase a license for the core. Contact your local Xilinx sales representative for information about pricing and availability.

For more information, visit the 10 Gigabit Ethernet PCS/PMA with FEC/Auto-Negotiation (10GBASE-KR) product web page.

Information about other Xilinx LogiCORE IP modules is available at the Xilinx Intellectual Property page. For information on pricing and availability of other Xilinx LogiCORE IP modules and tools, contact your local Xilinx sales representative.

Product Specification

Standards

The 10GBASE-R/KR core is designed to the standard specified in clauses 45, 49, 72, 73 and 74 of the 10-Gigabit Ethernet specification IEEE Std. 802.3-2012.

Performance

Latency

These measurements are for the core only; they do not include the latency through the transceiver. The latency through the transceiver can be obtained from the relevant user guide. These apply to Zynq®-7000, Virtex®-7, and Kintex®-7 devices.

Transmit Path Latency

As measured from the input port `xgmii_txd[63:0]` of the transmitter side XGMII (until that data appears on `gt_txd[31:0]` on the transceiver interface), the latency through the core for the internal XGMII interface configuration in the transmit direction is 20 periods of `txclk322`. When the optional FEC functionality is included in the core, this increases to 26 periods of `txclk322`.

Receive Path Latency

Measured from the input into the core on `gt_rxd[31:0]` until the data appears on `xgmii_rxd[63:0]` of the receiver side XGMII interface, the latency through the core in the receive direction is nominally equal to 32 cycles of `rxclk322`, including ± 4 cycles in the elastic buffer. The latency depends on sync bit alignment position and data positioning within the transceiver 4-byte interface. When the optional FEC functionality is included in the core, this increases by 70 cycles of `rxclk322` and if error reporting to the PCS layer is enabled, there will be an extra 66 cycles of `rxclk322` latency.

Transceiver Latency

See *7 Series Transceivers User Guide* (UG476) [Ref 3] for information on the transceiver latency.

Resource Utilization

These resource numbers were created for the core in isolation. The numbers were generated using the Vivado® Design Suite.

Table 2-1 provides approximate slice counts for the BASE-R options on Zynq-7000, Virtex-7, and Kintex-7 devices.

Table 2-1: **Device Utilization - BASE-R on Zynq-7000, Virtex-7, and Kintex-7 Devices**

Parameter	LUTs	FFs
MDIO = NO	2,596	2,630
MDIO = YES	3,075	2,946

Table 2-2 provides the approximate slice counts for the BASE-KR options on Virtex-7 devices.

Table 2-2: **Device Utilization on Virtex-7 Devices**

FEC	AN	MDIO	LUTs	FFs
N	N	N	4,036	3,896
N	N	Y	4,782	4,253
N	Y	Y	5,532	5,540
Y	Y	Y	8,913	6,944

Port Descriptions

MAC-Side Interface: XGMII

The MAC (or client) side of the core has a 64-bit datapath plus 8 control bits implementing an XGMII interface. Table 2-3 defines the signals, which are all synchronous to the 156.25 MHz core clock. It is designed to be connected to either user logic within the FPGA or, by using SelectIO™ technology Double Data Rate (DDR) registers in your own design top-level, to provide an external 32-bit 312.5 MHz DDR XGMII, defined in clause 46 of *IEEE 802.3-2012*.

Table 2-3: **MAC-Side Interface Ports**

Signal Name	Direction	Description
xgmii_txd[63:0]	In	64-bit transmit data word
xgmii_txc[7:0]	In	8-bit transmit control word
xgmii_rxd[63:0]	Out	64-bit receive data word
xgmii_rxc[7:0]	Out	8-bit receive control word

Figure 2-1 illustrates transmitting a frame through the client-side interface.

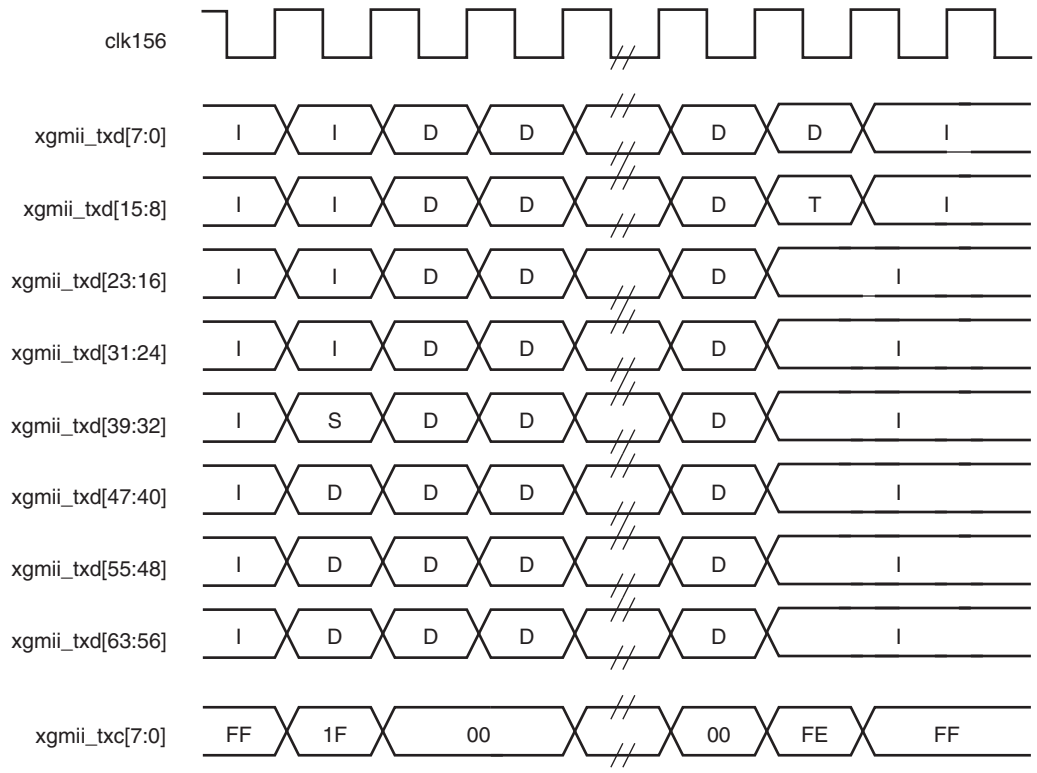

Figure 2-1: **Transmitting a Frame Through the Client-Side Interface**

Figure 2-2 illustrates receiving a frame through the client-side interface.

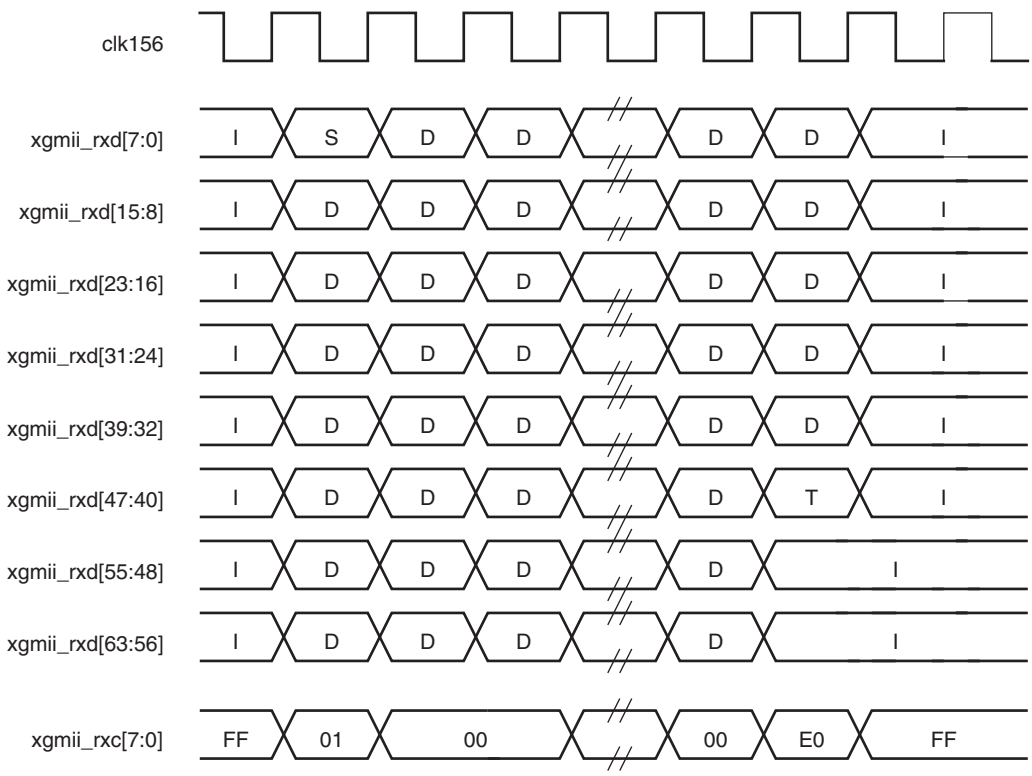

Figure 2-2: **Receiving a Frame Through the Client-Side Interface**

Serial Data Ports

The serial data ports should be connected to the PMD which is either an Optical module or a backplane.

Table 2-4: **Serial Data Ports**

Signal Name	Direction	Description
txn, txp	Output	Serial data to optics/backplane
rxn, rxp	Input	Serial data from optics/backplane

Optical Module Interface

The status and control interface to an attached optical module is a simple pin-to-pin interface on those pins that need to be connected. The signals are described in Table 2-5. See Chapter 3, Designing with the Core for details on connecting an optical module to the 10GBASE-R core. For 10GBASE-KR, it is recommended to tie the `signal_detect` input to 1, the tx_fault input to 0, and leave the `tx_disable` output dangling.

Table 2-5: **Optical Module Interface Ports**

Signal Name	Direction	Description
signal_detect	IN	Status signal from attached optical module.
tx_fault	IN	Status signal from attached optical module. [a] [b]
tx_disable	OUT	Control signal to attached optical module

a. This signal is not connected inside this version of the core. It is left to users to handle these inputs and reset their design as they see fit.
b. Connect to `SFP+ tx_fault` signal, or `XFP MOD_NR` signal, depending on which is present.

Management Interface (MDIO)

The optional MDIO interface is a simple low-speed two-wire interface for management of the 10-Gigabit Ethernet PCS/PMA core, consisting of a clock signal and a bidirectional data signal. The interface is defined in clause 45 of the *IEEE 802.3-2012* standard.

In this core, the MDIO interface is an optional block. If implemented, the bidirectional data signal MDIO is implemented as three unidirectional signals. These can be used to drive a 3-state buffer either in the FPGA IOB or in a separate device.

Table 2-6: **MDIO Management Interface Ports**

Signal Name	Direction	Description
mdc	In	Management clock
mdio_in	In	MDIO Input
mdio_out	Out	MDIO Output
mdio_tri	Out	MDIO 3-state control. 1 disconnects the output driver from the MDIO bus.
prtad[4:0]	In	MDIO port address. When multiple MDIO-managed ports appear on the same bus, this input can be used to set the address of each port.

Configuration and Status Signals

As an alternative to the MDIO interface, vector-based interfaces are provided to allow control and status to flow to and from the core. Table 2-7 describes these two vectors. Neither vector is completely populated so the actual number of pins required is much lower than the maximum widths of the vectors. For the status vector, correct default values are provided for all bits in the associated IEEE registers. Further details of these vectors can be found in Table 3-4 to Table 3-7.

Table 2-7: **Configuration and Status Vectors**

Signal Name	Direction	Description
configuration_vector[535:0]	In	Configures the PCS/PMA registers
status_vector[447:0]	Out	Reflects *recent* status of PCS/PMA registers

Clocking and Reset Signals - Zynq-7000, Virtex-7, and Kintex-7 Devices — Shared Logic Included in Example Design

If you selected to **Include Shared Logic in example design** during the core customization, then included in the example design top-level sources are circuits for clock and reset management. These can include clock generators, reset synchronizers, or other useful utility circuits that can be useful in your particular application.

Table 2-8 shows the ports on the netlist that are associated with system clocks and resets.

Table 2-8: **Clock and Reset Ports- Zynq-7000, Virtex-7, and Kintex-7 Devices (Logic included in example design)**

Signal Name	Direction	Description
clk156	IN	System clock for core
txusrclk, txusrclk2	IN	Transmit path clock, derived from TXCLKOUT on the GTXE2/GTHE2 transceiver
dclk	IN	Management/DRP clock, *must* be the same clock as clk156
areset	IN	Asynchronous (master) reset [a]
txclk322	OUT	TXOUTCLK from the GTXE2/GTHE2 output towards shared clock generation logic
areset_clk156	IN	Synchronous reset in clk156 domain
gttxreset	IN	Transceiver TX Reset signal in refclk domain
gtrxreset	IN	Transceiver RX Reset signal in refclk domain
qplllock	IN	Transceiver QPLL Lock signal
qplloutclk	IN	Transceiver QPLL clock
qplloutrefclk	IN	Transceiver QPLL refclk
reset_counter_done	IN	Indication that 500 ns have passed after configuration was complete
tx_resetdone	OUT	Transceiver TX reset-done
rx_resetdone	OUT	Transceiver RX reset-done

a. This reset will also reset all management registers.

Clocking and Reset Signals — Zynq-7000, Virtex-7, and Kintex-7 Devices (Shared Logic included in core)

If you selected to **Include Shared Logic in core** during core customization, most of the clocking and reset blocks are included within the core itself.

Table 2-9 shows the ports on the netlist that are associated with these clocks and resets, which can be reused by other user logic or IP cores.

Table 2-9: **Clock and Reset Ports — Zynq-7000, Virtex-7, and Kintex-7 Devices (Logic included in core)**

Signal Name	Direction	Description
refclk_p, refclk_n	IN	Differential clock input (for transceiver)
reset	IN	Asynchronous 'master' reset[a]
resetdone	OUT	Combined transceiver reset-done indication (in core_clk156_out domain)
core_clk156_out	OUT	System clock from the core
dclk_out	OUT	Transceiver DRP clock from the core (identical to core_clk156_out)
qplllock_out	OUT	Lock indication from QPLL block in core
qplloutclk_out	OUT	QPLL output clock from QPLL block in core
qplloutrefclk_out	OUT	QPLL output reference clock from QPLL block in core
txusrclk_out	OUT	txusrclk from shared logic block in core
txusrclk2_out	OUT	txusrclk2 from shared logic block in core
areset_clk156_out	OUT	reset signal synchronized to clk156
gttxreset_out	OUT	Signal that is used to hold transceiver in reset until 500 ns after configuration or 'master' reset, synchronized to the refclk
gtrxreset_out	OUT	Signal that is used to hold transceiver in reset until 500 ns after configuration or 'master' reset, synchronized to the refclk
txuserrdy_out	OUT	Transceiver control signal equivalent to the QPLLLOCK signal, synchronized to txusrclk2
reset_counter_done_out	OUT	Indication that 500 ns have passed after configuration or 'master' reset, synchronized to the refclk

a. This reset will also reset all management registers.

Training Interface — Zynq-7000, Virtex-7, and Kintex-7 Devices, BASE-KR Only

In the 7 series devices, an external Training Algorithm can optionally be connected to the Training Interface, which allows access to both the 802.3 registers in the core and the DRP registers in the GTXE2/GTHE2 transceiver. Table 2-10 shows the ports on the netlist that are associated with that interface.

Table 2-10: **Training Interface Ports - Zynq-7000, Virtex-7, and Kintex-7 Devices, BASE-KR Only**

Signal Name	Direction	Description
training_enable	in	Signal from external Training Algorithm to enable the training interface. This should not be confused with the IEEE register 1.150.1–Training Enable. A rising edge on training_enable initiates a register access.
training_addr[20:0]	in	Register address from Training Algorithm – bits [20:16] are the DEVAD for 802.3 registers
training_rnw	in	Read/Write_bar signal from Training Algorithm
training_ipif_cs	in	Select access to 802.3 registers in the core (1)
training_drp_cs	in	Select access to DRP registers in the GTXE2/GTHE2 transceiver
training_rddata[15:0]	out	Read data from DRP or 802.3 registers
training_rdack	out	Read Acknowledge signal to external Training Algorithm
training_wrack	out	Write Acknowledge signal to external Training Algorithm

1. This signal has no meaning or effect when the core is created without an MDIO interface because all registers are exposed through the configuration and status vectors. This should be tied to '0' in that case. Access to transceiver DRP registers through the Training interface is unaffected.

Figure 2-3 and Figure 2-4 show the timing diagrams for Using the Training Interface to Access Internal Core Registers and Transceiver Registers through the DRP Port. As shown in Figure 2-3 and Figure 2-4, `training_drp_cs`, `training_ipif_cs`, and `training_enable` should be Low between read or write accesses.

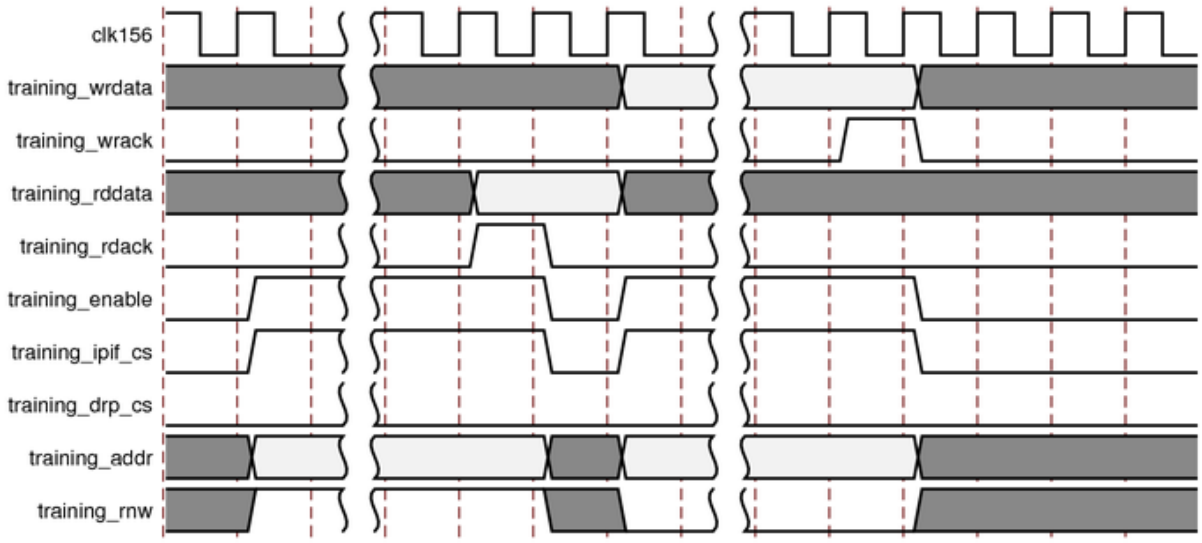

Figure 2-3: **Using the Training Interface to Access Internal Core Registers**

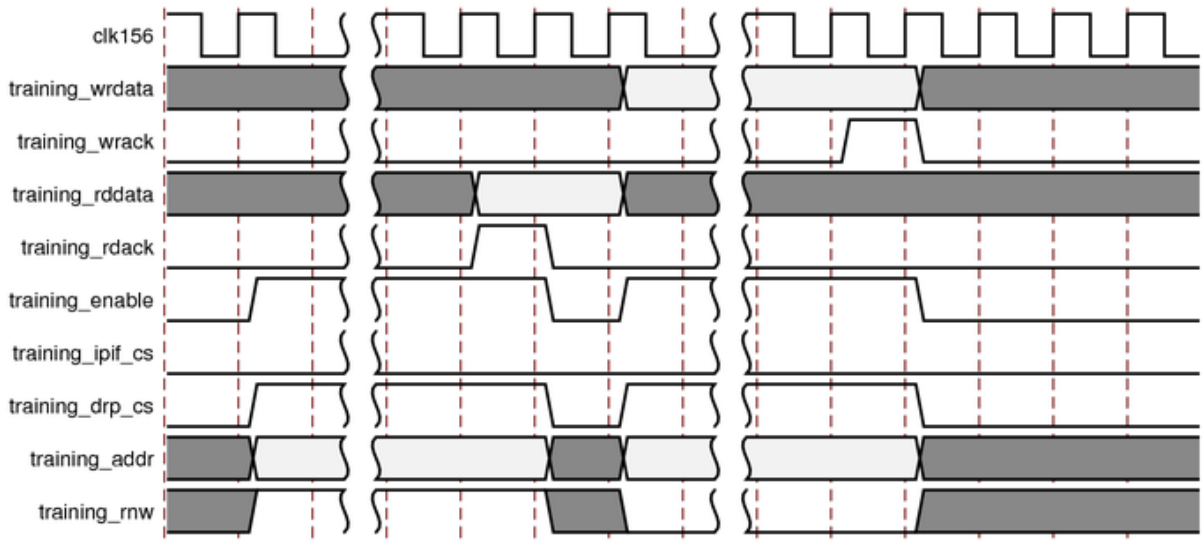

Figure 2-4: **Using the Training Interface to Access Transceiver Registers through the DRP Port**

DRP Interfaces — Zynq-7000, Virtex-7, and Kintex-7 Devices

Because you might wish to connect your own logic to the DRP interface of the transceiver, the interface between the core logic and the transceiver is brought out to an interface which can be connected to an external Arbiter block. The interface directly to the transceiver DRP is also provided.

If no user-logic or arbiter is required, the `core_gt_drp_interface` can be connected directly to the `user_gt_drp_interface` and `drp_req` can be connected directly to `drp_gnt`.

Table 2-11: **DRP Interface Signals**

Signal Name	Direction	Interface	Description
drp_req	OUT	N/A	This active-High signal can be used on an external arbiter, to request and hold onto access to the DRP.
drp_gnt	IN	N/A	This signal should be driven High when access is granted to the DRP by an external arbiter. If no external arbiter is present, connect this directly to the drp_req signal.
drp_daddr_o [15:0]	OUT	core_gt_drp_interface	This vector is driven by the core and is eventually used on the DADDR port on the transceiver.
drp_den_o	OUT	core_gt_drp_interface	This signal is driven by the core and is eventually used on the DEN port on the transceiver.
drp_di_o [15:0]	OUT	core_gt_drp_interface	This vector is driven by the core and is eventually used on the DI port on the transceiver.
drp_dwe_o	OUT	core_gt_drp_interface	This signal is driven by the core and is eventually used on the DWE port on the transceiver.
drp_drpdo_i [15:0]	IN	core_gt_drp_interface	This vector is driven by an external arbiter, or by drp_drpdo_o and is eventually driven by the DO port on the transceiver.
drp_drdy_i	IN	core_gt_drp_interface	This signal is driven by an external arbiter, or by drp_drdy_o and is eventually driven by the DRDY port on the transceiver.
drp_daddr_i [15:0]	IN	user_gt_drp_interface	This vector is driven by an external arbiter or by drp_daddr_o and is eventually used on the DADDR port on the transceiver.
drp_den_i	IN	user_gt_drp_interface	This signal is driven by an external arbiter or by drp_den_o and is eventually used on the DEN port on the transceiver.
drp_di_i [15:0]	IN	user_gt_drp_interface	This vector is driven by an external arbiter or by drp_di_o and is eventually used on the DI port on the transceiver.
drp_dwe_i	IN	user_gt_drp_interface	This signal is driven by an external arbiter or by drp_dwe_o and is eventually used on the DWE port on the transceiver.
drp_drpdo_o [15:0]	OUT	user_gt_drp_interface	This vector is driven by the DO port on the transceiver.
drp_drdy_o	OUT	user_gt_drp_interface	This signal is driven by the DRDY port on the transceiver.

Miscellaneous Signals — Zynq-7000, Virtex-7, and Kintex-7 Devices

Table 2-12: **Miscellaneous Signals**

Signal Name	Direction	Description
core_status[7:0]	OUT	Bit 0 = PCS Block Lock, Bits [7:6] are reserved BASE-KR cores: FEC Signal OK in bit 1[a], pmd_signal_detect (Training Done) in bit 2[b], AN Complete in bit 3, AN Enable is bit 4 and an_link_up is bit 5[c].
is_eval	OUT	Base-KR only: Constant output which is 1 if this is an Evaluation Licensed core
an_enable	IN	Base-KR only: Used to disable Autonegotiation during simulation – normally tie this to 1. Only for cores with Optional Autonegotiation block.
pma_pmd_type[2:0]	IN	BaseR only: Set this to a constant to define the PMA/PMD type as described in IEEE 802.3 section 45.2.1.6: 111 = 10GBASE-SR 110= 10GBASE-LR 101 = 10GBASE-ER

a. This bit is equivalent to the FEC block lock if FEC is included in the core *and* FEC is enabled AND Training Done AND `signal_detect` AND `an_link_up`. If FEC is not included or is not enabled, this bit is equivalent to Training Done AND `signal_detect` AND `an_link_up`.

b. In fact this is equivalent to Training Done AND `signal_detect`.

c. The latter two signals are required in the core to enable a switching of transceiver RX modes during AutoNegotiation. When the optional AutoNegotiation block is not included with the core, or is included but disabled by either the `an_enable` pin on the core (simulation-only) or by the management register 7.0.12, `an_link_up` (bit 5) will be fixed to a constant 1 and bits 3 and 4 will be constant 0.

Transceiver Debug Signals — Zynq-7000, Virtex-7, and Kintex-7 Devices

If you select to include transceiver debug ports on the core during core customization, these ports will be available. Consult the relevant transceiver user guide or product guide for more information.

Table 2-13: **Transceiver Debug Signals**

Signal Name	Direction	Description
gt0_eyescanreset	IN	Eye Scan Reset control
gt0_eyescantrigger	IN	Eye Scan Trigger control
gt0_rxcdrhold	IN	CDR Hold control
gt0_txprbsforceerr	IN	Force a single TXPRBS Error
gt0_txpolarity	IN	Switch the sense of txn and txp
gt0_rxpolarity	IN	Switch the sense of rxn and rxp
gt0_rxrate [2:0]	IN	RX Rate control
gt0_txprecursor [4:0]	IN	TX Precursor control (Base-R only)
gt0_txpostcursor [4:0]	IN	TX Postcursor control (Base-R only)
gt0_txdiffctrl [3:0]	IN	TX Differential Drive control (Base-R only)
gt0_eyescandataerror	OUT	Eye Scan Data Error indication
gt0_txbufstatus [1:0]	OUT	TX Buffer Status

Register Space

This core implements registers which are further described in 802.3 Clause 45. If the core is generated without an MDIO interface, these registers are still implemented but generally using only configuration or status pins on the core. For example, register 1.0, bit 15 (PMA Reset) is implemented as bit 15 of the configuration vector and register 1.1, bit 7 (PMA/PMD Fault) is implemented as status vector bit 23. These mappings are described in Configuration and Status Vectors in Chapter 3.

10GBASE-R PCS/PMA Register Map

If the core is configured as a 10GBASE-R PCS/PMA, it occupies MDIO Device Addresses 1 and 3 in the MDIO register address map, as shown in Table 2-14.

Table 2-14: **10GBASE-R/KR PCS/PMA MDIO Registers**

Register Address	Register Name
1.0	MDIO Register 1.0: PMA/PMD Control 1
1.1	MDIO Register 1.1: PMA/PMD Status 1
1.4	MDIO Register 1.4: PMA/PMD Speed Ability
1.5, 1.6	MDIO Registers 1.5 and 1.6: PMA/PMD Devices in Package
1.7	MDIO Register 1.7: 10G PMA/PMD Control 2
1.8	MDIO Register 1.8: 10G PMA/PMD Status 2
1.9	MDIO Register 1.9: 10G PMD Transmit Disable

Table 2-14: **10GBASE-R/KR PCS/PMA MDIO Registers *(Cont'd)***

Register Address	Register Name
1.10	MDIO Register 1.10: 10G PMD Signal Receive OK
1.11 to 1.32787	Reserved
1.32789 to 1.65534	Reserved
1.65535	MDIO Register 1.65535: Core Version Info
3.0	MDIO Register 3.0: PCS Control 1
3.1	MDIO Register 3.1: PCS Status 1
3.4	MDIO Register 3.4: PCS Speed Ability
3.5, 3.6	MDIO Registers 3.5 and 3.6: PCS Devices in Package
3.7	MDIO Register 3.7: 10G PCS Control 2
3.8	MDIO Register 3.8: 10G PCS Status 2
3.9 to 3.31	Reserved
3.32	MDIO Register 3.32: 10GBASE-R Status 1
3.33	MDIO Register 3.33: 10GBASE-R Status 2
3.34–37	MDIO Register 3.34–37: 10GBASE-R Test Pattern Seed A0–3
3.38–41	MDIO Register 3.38–41: 10GBASE-R Test Pattern Seed B0–3
3.42	MDIO Register 3.42: 10GBASE-R Test Pattern Control
3.43	MDIO Register 3.43: 10GBASE-R Test Pattern Error Counter
3.44 to 3.65534	Reserved
3.32769 to 3.65534	Reserved
3.65535	MDIO Register 3.65535: 125 Microsecond Timer Control

10GBASE-KR PCS/PMA Register Map

If the core is configured as a 10GBASE-KR PCS/PMA, it occupies MDIO Device Addresses 1, 3 and optionally 7 in the MDIO register address map, as shown in Table 2-15.

Table 2-15: **10GBASE-KR PCS/PMA Registers**

Register Address	Register Name
1.0	MDIO Register 1.0: PMA/PMD Control 1
1.1	MDIO Register 1.1: PMA/PMD Status 1
1.150	MDIO Register 1.150: 10GBASE-KR PMD Control
1.151	MDIO Register 1.151: 10GBASE-KR PMD Status
1.152	MDIO Register 1.152: 10GBASE-KR LP Coefficient Update
1.153	MDIO Register 1.153: 10GBASE-KR LP Status
1.154	MDIO Register 1.154: 10GBASE-KR LD Coefficient Update
1.155	MDIO Register 1.155: 10GBASE-KR LD Status

Table 2-15: **10GBASE-KR PCS/PMA Registers *(Cont'd)***

Register Address	Register Name
1.170	MDIO Register 1.170: 10GBASE-R FEC Ability (1)
1.171	MDIO Register 1.171: 10GBASE-R FEC Control(1)
1.172 to 1.173	MDIO Register 1.172: 10GBASE-R FEC Corrected Blocks (Lower) (1) MDIO Register 1.173: 10GBASE-R FEC Corrected Blocks (Upper) (1)
1.174 to 1.175	MDIO Register 1.174: 10GBASE-R FEC Uncorrected Blocks (Lower) (1) MDIO Register 1.175: 10GBASE-R FEC Uncorrected Blocks (Upper) (1)
1.4	MDIO Register 1.4: PMA/PMD Speed Ability
1.5, 1.6	MDIO Registers 1.5 and 1.6: PMA/PMD Devices in Package
1.7	MDIO Register 1.7: 10G PMA/PMD Control 2
1.8	MDIO Register 1.8: 10G PMA/PMD Status 2
1.9	MDIO Register 1.9: 10G PMD Transmit Disable
1.10	MDIO Register 1.10: 10G PMD Signal Receive OK
1.11 to 1.149	Reserved
1.176 to 1.65519	Reserved
1.65520	MDIO Register: 1.65520: Vendor-Specific LD Training (vendor-specific register where Local Device Coefficient Updates are to be written by Training Algorithm)
1.65521 to 1.65534	Reserved
1.65535	MDIO Register 1.65535: Core Version Info
3.0	MDIO Register 3.0: PCS Control 1
3.1	MDIO Register 3.1: PCS Status 1
3.4	MDIO Register 3.4: PCS Speed Ability
3.5, 3.6	MDIO Registers 3.5 and 3.6: PCS Devices in Package
3.7	MDIO Register 3.7: 10G PCS Control 2
3.8	MDIO Register 3.8: 10G PCS Status 2
3.9 to 3.31	Reserved
3.32	MDIO Register 3.32: 10GBASE-R Status 1
3.33	MDIO Register 3.33: 10GBASE-R Status 2
3.34–37	MDIO Register 3.34–37: 10GBASE-R Test Pattern Seed A0–3
3.38–41	MDIO Register 3.38–41: 10GBASE-R Test Pattern Seed B0–3
3.42	MDIO Register 3.42: 10GBASE-R Test Pattern Control
3.43	MDIO Register 3.43: 10GBASE-R Test Pattern Error Counter
3.44 to 3.32767	Reserved
3.65535	MDIO Register 3.65535: 125 Microsecond Timer Control
7.0	MDIO Register 7.0: AN Control (2)
7.1	MDIO Register 7.1: AN Status (2)

Table 2-15: **10GBASE-KR PCS/PMA Registers *(Cont'd)***

Register Address	Register Name
7.16, 17, 18	MDIO Register 7.16:17:18: AN Advertisement (2)
7.19, 20, 21	MDIO Register 7.19, 20, 21: AN LP Base Page Ability (2)
7.22, 23, 24	MDIO Register 7.22, 23, 24: AN XNP Transmit (2)
7.25, 26, 27	MDIO Register 7.25, 26, 27: AN LP XNP Ability (2)
7.48	MDIO Register 7.48: Backplane Ethernet Status(2)

1. For cores with optional FEC block
2. For cores with optional AN block

MDIO Register 1.0: PMA/PMD Control 1

Figure 2-5 shows the MDIO Register 1.0: Physical Medium Attachment/Physical Medium Dependent (PMA/PMD) Control 1.

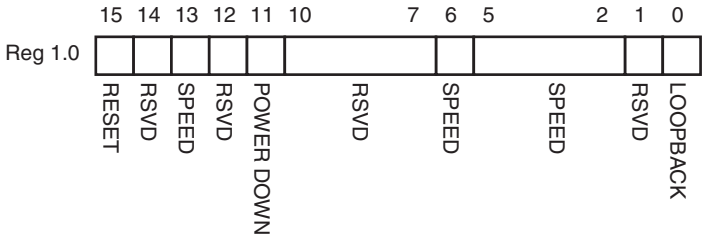

Figure 2-5: **PMA/PMD Control 1 Register**

Table 2-16 shows the PMA Control 1 register bit definitions.

Table 2-16: **PMA/PMD Control 1 Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
1.0.15	Reset	1 = Block reset 0 = Normal operation The 10GBASE-R/KR block is reset when this bit is set to 1. It returns to 0 when the reset is complete.	R/W Self-clearing	0
1.0.14	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	0
1.0.13	Speed Selection	The block always returns 1 for this bit and ignores writes.	R/O	1
1.0.12	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	0
1.0.11	Power down	This bit has no effect.	R/W	0
1.0.10:7	Reserved	The block always returns 0 for these bits and ignores writes.	R/O	All 0s
1.0.6	Speed Selection	The block always returns 1 for this bit and ignores writes.	R/O	1
1.0.5:2	Speed Selection	The block always returns 0s for these bits and ignores writes.	R/O	All 0s
1.0.1	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	All 0s
1.0.0	Loopback	1 = Enable PMA loopback mode 0 = Disable PMA loopback mode The vendor-specific register bits 1.32788.1:0 take precedence over this bit.	R/W	0

MDIO Register 1.1: PMA/PMD Status 1

Figure 2-6 shows the MDIO Register 1.1: PMA/PMD Status 1.

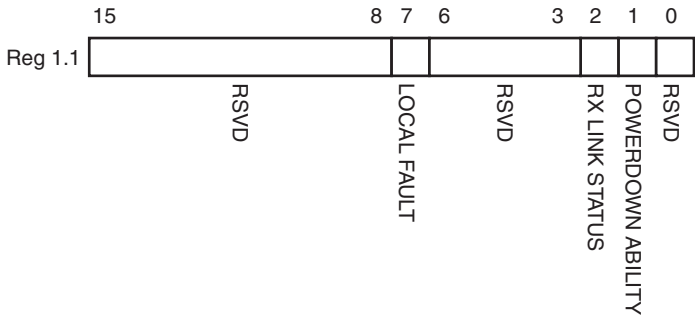

Figure 2-6: **PMA/PMD Status 1 Register**

Table 2-17 shows the PMA/PMD Status 1 register bit definitions.

Table 2-17: **PMA/PMD Status 1 Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
1.1.15:8	Reserved	The block always returns 0 for this bit.	R/O	0
1.1.7	Local Fault	Zynq-7000/Virtex-7/Kintex-7: 1 = Local Fault detected	R/O	0
1.1.6:3	Reserved	The block always returns 0 for this bit.	R/O	0
1.1.2	Receive Link Status	Zynq-7000/Virtex-7/Kintex-7: 1 = Receive Link UP	R/O Zynq/V7/K7: Latches Low	1
1.1.1	Power Down Ability	The block always returns 1 for this bit.	R/O	1
1.1.0	Reserved	The block always returns 0 for this bit.	R/O	0

MDIO Register 1.4: PMA/PMD Speed Ability

Figure 2-7 shows the MDIO Register 1.4: PMA/PMD Speed Ability.

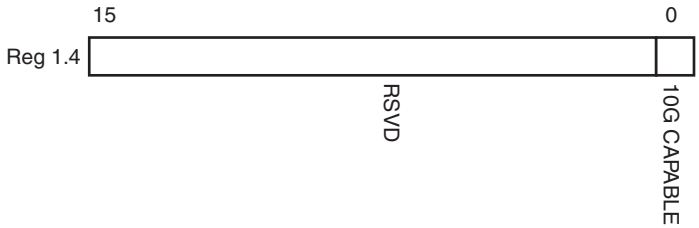

Figure 2-7: **PMA/PMD Speed Ability Register**

Table 2-18 shows the PMA/PMD Speed Ability register bit definitions.

Table 2-18: **PMA/PMD Speed Ability Register Bit Definitions**

Bits	Name	Description	Attribute	Default Value
1.4.15:1	Reserved	The block always returns 0 for these bits and ignores writes.	R/O	All 0s
1.4.0	10G Capable	The block always returns 1 for this bit and ignores writes.	R/O	1

MDIO Registers 1.5 and 1.6: PMA/PMD Devices in Package

Figure 2-8 shows the MDIO Registers 1.5 and 1.6: PMA/PMD Devices in Package.

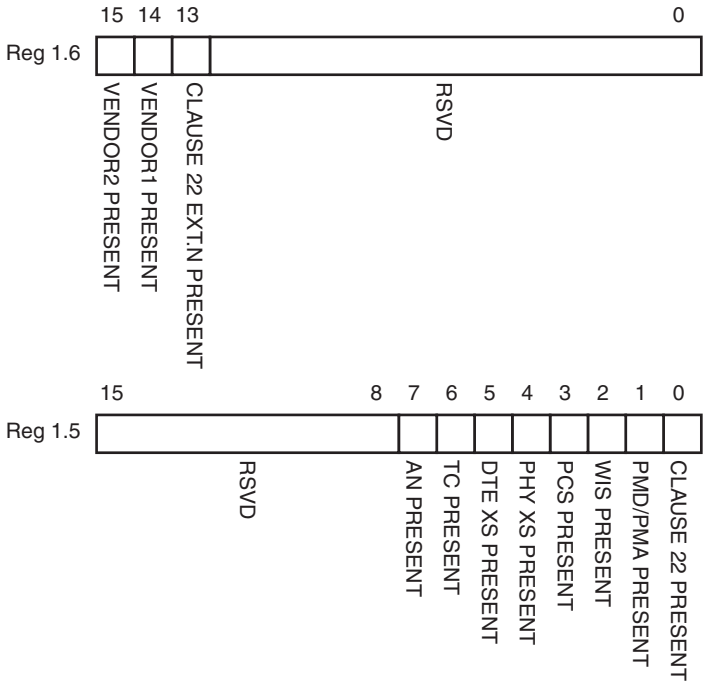

Figure 2-8: **PMA/PMD Devices in Package Registers**

Table 2-19 shows the PMA/PMD Device in Package registers bit definitions.

Table 2-19: **PMA/PMD Devices in Package Registers Bit Definitions**

Bits	Name	Description	Attributes	Default Value
1.6.15	Vendor- specific Device 2 Present	The block always returns 0 for this bit.	R/O	0
1.6.14	Vendor-specific Device 1 Present	The block always returns 0 for this bit.	R/O	0
1.6.13	Clause 22 Extension Present	The block always returns 1 for this bit.	R/O	1
1.6.12:0	Reserved	The block always returns 0 for these bits.	R/O	All 0s
1.5.15:8	Reserved	The block always returns 0 for these bits.	R/O	All 0s
1.5.7	Autonegotiation present	Zynq-7000/Virtex-7/Kintex-7: 1 = optional AN block is included	R/O	1
1.5.6	TC Present	The block always returns 0 for this bit	R/O	0
1.5.5	DTE XS Present	The block always returns 0 for this bit.	R/O	0
1.5.4	PHY XS Present	The block always returns 0 for this bit.	R/O	0

Table 2-19: **PMA/PMD Devices in Package Registers Bit Definitions *(Cont'd)***

Bits	Name	Description	Attributes	Default Value
1.5.3	PCS Present	The block always returns 1 for this bit.	R/O	1
1.5.2	WIS Present	The block always returns 0 for this bit.	R/O	0
1.5.1	PMA/PMD Present	The block always returns 1 for this bit.	R/O	1
1.5.0	Clause 22 Device Present	The block always returns 0 for this bit.	R/O	0

MDIO Register 1.7: 10G PMA/PMD Control 2

Figure 2-9 shows the MDIO Register 1.7: 10G PMA/PMD Control 2.

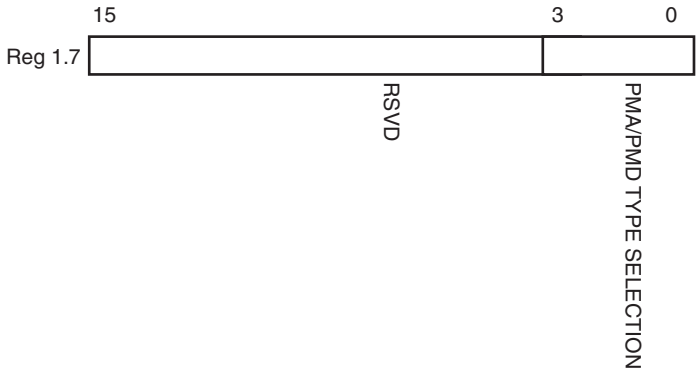

Figure 2-9: **10G PMA/PMD Control 2 Register**

Table 2-20 shows the PMA/PMD Control 2 register bit definitions.

Table 2-20: **10G PMA/PMD Control 2 Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
1.7.15:4	Reserved	The block always returns 0 for these bits and ignores writes.	R/O	All 0s
1.7.3:0	PMA/PMD Type Selection	Zynq-7000/Virtex-7/Kintex-7: This returns the value 0xyz, where xyz is set from the top level core port pma_pmd_type vector.	R/W	Zynq-7000/Virtex-7/Kintex-7: Base-R: Set from pma_pmd_type port. BASE-KR: returns 0xB

MDIO Register 1.8: 10G PMA/PMD Status 2

Figure 2-10 shows the MDIO Register 1.8: 10G PMA/PMD Status 2.

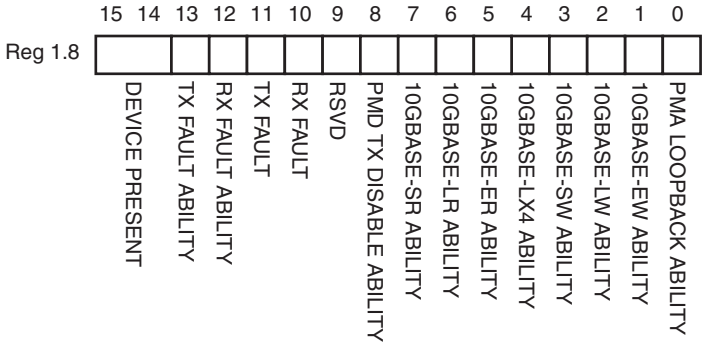

Figure 2-10: **10G PMA/PMD Status 2 Register**

Table 2-21 shows the PMA/PMD Status 2 register bit definitions.

Table 2-21: **10G PMA/PMD Status 2 Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
1.8.15:14	Device Present	The block always returns 10 for these bits.	R/O	‘10’
1.8.13	Transmit Local Fault Ability	Zynq-7000/Virtex-7/Kintex-7: The block always returns a 1 for this bit.	R/O	Zynq-7000/Virtex-7 /Kintex-7: 1
1.8.12	Receive Local Fault Ability	Zynq-7000/Virtex-7/Kintex-7: The block always returns a 1 for this bit.	R/O	Zynq-7000/Virtex-7 /Kintex-7: 1
1.8.11	Transmit Fault	Zynq-7000/Virtex-7/Kintex-7: 1 = Transmit Fault detected	Zynq/V7/K7: Latches High	0
1.8.10	Receive Fault	Zynq-7000/Virtex-7/Kintex-7: 1 = Receive Fault detected	Zynq/V7/K7: Latches High	0
1.8.9	Extended abilities	The block always returns 1 for this bit.	R/O	1
1.8.8	PMD Transmit Disable Ability	The block always returns 1 for this bit.	R/O	1
1.8.7	10GBASE-SR Ability	Zynq-7000/Virtex-7/Kintex-7: Base-R only: Returns a 1 if pma_pmd_type port is set to 111	R/O	Zynq-7000/Virtex-7 /Kintex-7: Depends on pma_pmd_type port
1.8.6	10GBASE-LR Ability	Zynq-7000/Virtex-7/Kintex-7: Base-R only: Returns a 1 if pma_pld_type port is set to 110	R/O	Zynq-7000/Virtex-7 /Kintex-7: Returns a '1' if pma_pld_type port is set to '110'
1.8.5	10GBASE-ER Ability	Zynq-7000/Virtex-7/Kintex-7: Base-R only: Returns a 1 if the pma_pmd_type port is set to 101	R/O	Zynq-7000/Virtex-7 /Kintex-7: Depends on pma_pmd_type port

Table 2-21: **10G PMA/PMD Status 2 Register Bit Definitions *(Cont'd)***

Bits	Name	Description	Attributes	Default Value
1.8.4	10GBASE-LX4 Ability	The block always returns 0 for this bit.	R/O	0
1.8.3	10GBASE-SW Ability	The block always returns 0 for this bit.	R/O	0
1.8.2	10GBASE-LW Ability	The block always returns 0 for this bit.	R/O	0
1.8.1	10GBASE-EW Ability	The block always returns 0 for this bit.	R/O	0
1.8.0	PMA Loopback Ability	The block always returns 1 for this bit.	R/O	1

MDIO Register 1.9: 10G PMD Transmit Disable

Figure 2-11 shows the MDIO 1.9 Register: 10G PMD Transmit Disable.

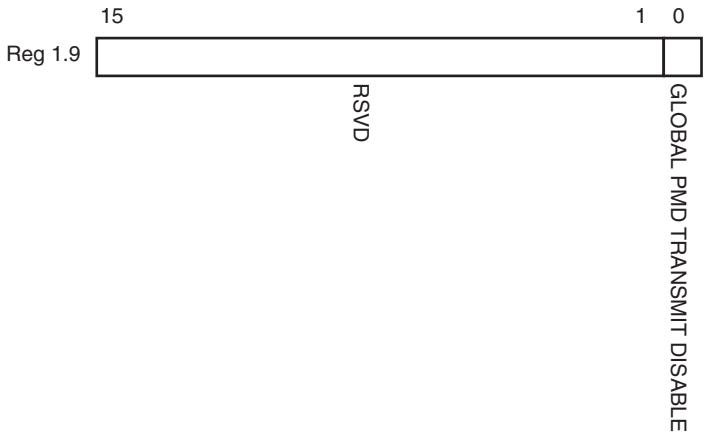

Figure 2-11: **10G PMD Transmit Disable Register**

Table 2-22: **10G PMD Transmit Disable Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
1.9.15:1	Reserved	The block always returns 0 for these bits and ignores writes.	R/O	All 0s
1.9.0	Global PMD Transmit Disable	1 = Disable Transmit path (also sets transmit_disable pin) 0 = Enable Transmit path	Zynq/V7/K7: R/W	Zynq/V7/K7: 0

MDIO Register 1.10: 10G PMD Signal Receive OK

Figure 2-12 shows the MDIO 1.10 Register: 10G PMD Signal Receive OK.

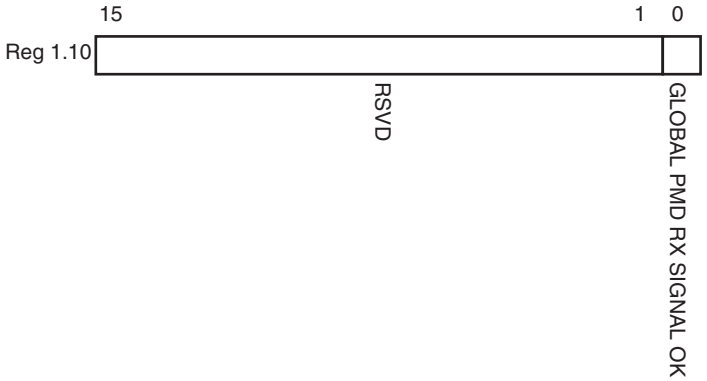

Figure 2-12: **10G PMD Signal Receive OK Register**

Table 2-21 shows the PMD Signal Receive OK register bit definitions.

Table 2-23: **10G PMD Signal Receive OK Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
1.10.15:1	Reserved	The block always returns 0 for these bits.	R/O	0s
1.10.0	Global PMD receive signal detect	1 = Signal detected on receive 0 = Signal not detected on receive	R/O	n/a

MDIO Register 1.150: 10GBASE-KR PMD Control

Figure 2-13 shows the MDIO Register 1.150: 10GBASE-KR PMD Control.

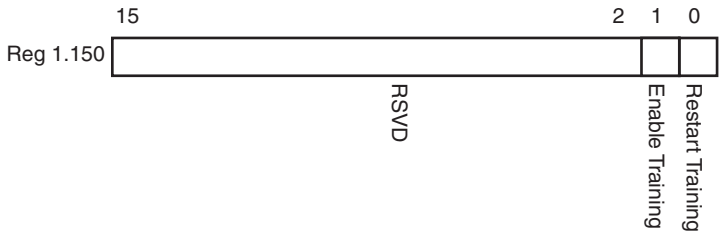

Figure 2-13: **10GBASE-KR PMD Control Register**

Table 2-24 shows the 10GBASE-KR PMD Control register bit definitions.

Table 2-24: **10GBASE-KR PMD Control Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
1.150.15:2	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	0
1.150.1	Training enable	1 = Enable the 10GBASE-KR start-up protocol 0 = Disable	R/W	0
1.150.0	Restart Training	1 = Reset the 10GBASE-KR start-up protocol 0 = Normal operation	R/W Self-clearing	0

MDIO Register 1.151: 10GBASE-KR PMD Status

Figure 2-14 shows the MDIO Register 1.151: 10GBASE-KR PMD Status.

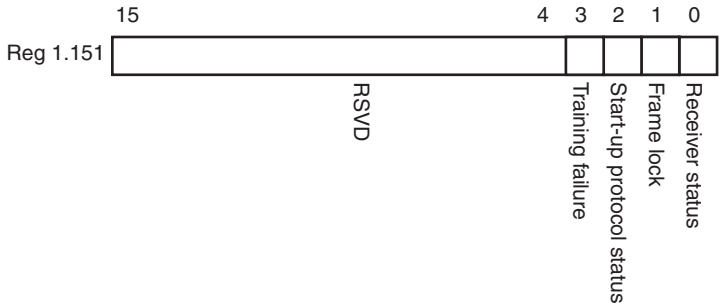

Figure 2-14: **10GBASE-KR PMD Status Register**

Table 2-25 shows the 10GBASE-KR PMD Status register bit definitions.

Table 2-25: **10GBASE-KR PMD Status Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
1.151.15:4	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	0
1.151.3	Training Failure	1 = Training Failure has been detected 0 = Not detected	R/O	0
1.151.2	Start-up Protocol status	1 = Start-up protocol in progress 0 = Protocol complete	R/O	0
1.151.1	Frame Lock	1 = Training frame delineation detected 0 = Not detected	R/O	0
1.151.0	Receiver status	1 = Receiver trained and ready to receive data 0 = Receiver training	R/O	0

MDIO Register 1.152: 10GBASE-KR LP Coefficient Update

Figure 2-15 shows the MDIO Register 1.152: 10GBASE-KR LP Coefficient Update.

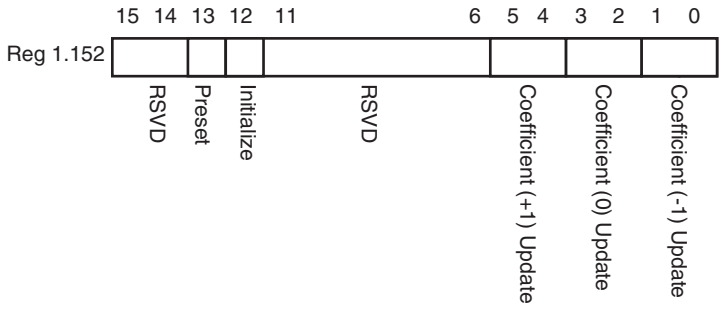

Figure 2-15: **10GBASE-KR LP Coefficient Update Register**

Table 2-26 shows the 10GBASE-KR LP coefficient update register bit definitions.

Table 2-26: **10GBASE-KR LP Coefficient Update Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
1.152.15:14	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	0
1.152.13	Preset	1 = Preset coefficients 0 = Normal operation	R/W[a]	0
1.152.12	Initialize	1 = Initialize coefficients 0 = Normal operation	R/W[a]	0
1.152.11:6	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	0s
1.152.5:4	Coefficient (+1) update	5:4 = 11 = reserved 10 = decrement 01 = increment 00 = hold	R/W[a]	00
1.152.3:2	Coefficient (0) update	3:2 = 11 = reserved 10 = decrement 01 = increment 00 = hold	R/W[a]	00
1.152.1:0	Coefficient (–1) update	1:0 = 11 = reserved 10 = decrement 01 = increment 00 = hold	R/W[a]	00

a. Writable only when register 1.150.1 = 0

MDIO Register 1.153: 10GBASE-KR LP Status

Figure 2-16 shows the MDIO Register 1.153: 10GBASE-KR LP status.

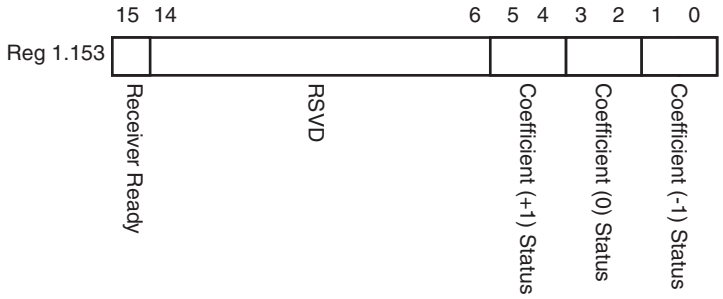

Figure 2-16: **10GBASE-KR LP Status Register**

Table 2-27 shows the 10GBASE-KR LP status register bit definitions.

Table 2-27: **10GBASE-KR LP Status Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
1.153.15:14	Receiver Ready	1 = The LP receiver has determined that training is complete and is prepared to receive data 0 = The LP receiver is requesting that training continue	R/O	0
1.153.14:6	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	0s
1.153.5:4	Coefficient (+1) status	5:4 = 11 = maximum 10 = minimum 01 = updated 00 = not updated	R/O	00
1.153.3:2	Coefficient (0) status	3:2 = 11 = maximum 10 = minimum 01 = updated 00 = not updated	R/O	00
1.153.1:0	Coefficient (–1) status	1:0 = 11 = maximum 10 = minimum 01 = updated 00 = not updated	R/O	00

MDIO Register 1.154: 10GBASE-KR LD Coefficient Update

Figure 2-17 shows the MDIO Register 1.154: 10GBASE-KR LD coefficient update.

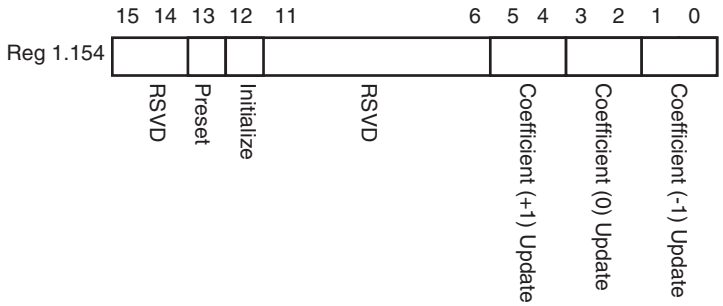

Figure 2-17: **10GBASE-KR LD Coefficient Update Register**

Table 2-28 shows the 10GBASE-KR LD coefficient update register bit definitions.

Table 2-28: **10GBASE-KR LD Coefficient Update Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
1.154.15:14	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	0
1.154.13	Preset	1 = Preset coefficients 0 = Normal operation	R/O[a]	0
1.154.12	Initialize	1 = Initialize coefficients 0 = Normal operation	R/O[a]	0
1.154.11:6	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	0s
1.154.5:4	Coefficient (+1) update	5:4 = 11 = reserved 10 = decrement 01 = increment 00 = hold	R/O[a]	00
1.154.3:2	Coefficient (0) update	3:2 = 11 = reserved 10 = decrement 01 = increment 00 = hold	R/O[a]	00
1.154.1:0	Coefficient (–1) update	1:0 = 11 = reserved 10 = decrement 01 = increment 00 = hold	R/O[a]	00

a. These registers are programmed by writing to register 1.65520.

MDIO Register 1.155: 10GBASE-KR LD Status

Figure 2-18 shows the MDIO Register 1.155: 10GBASE-KR LD status.

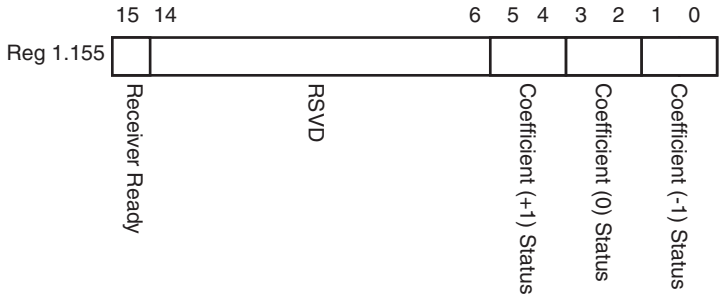

Figure 2-18: **10GBASE-KR LD Status Register**

Table 2-29 shows the 10GBASE-KR LD status register bit definitions.

Table 2-29: **10GBASE-KR LD Status Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
1.155.15	Receiver Ready	1 = The LD receiver has determined that training is complete and is prepared to receive data. 0 = The LD receiver is requesting that training continue.	R/O	0
1.155.14:6	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	0s
1.155.5:4	Coefficient (+1) status	5:4 = 11 = maximum 10 = minimum 01 = updated 00 = not updated	R/O	00
1.155.3:2	Coefficient (0) status	3:2 = 11 = maximum 10 = minimum 01 = updated 00 = not updated	R/O	00
1.155.1:0	Coefficient (–1) status	1:0 = 11 = maximum 10 = minimum 01 = updated 00 = not updated	R/O	00

MDIO Register 1.170: 10GBASE-R FEC Ability

Figure 2-19 shows the MDIO Register 1.170: 10GBASE-R FEC Ability.

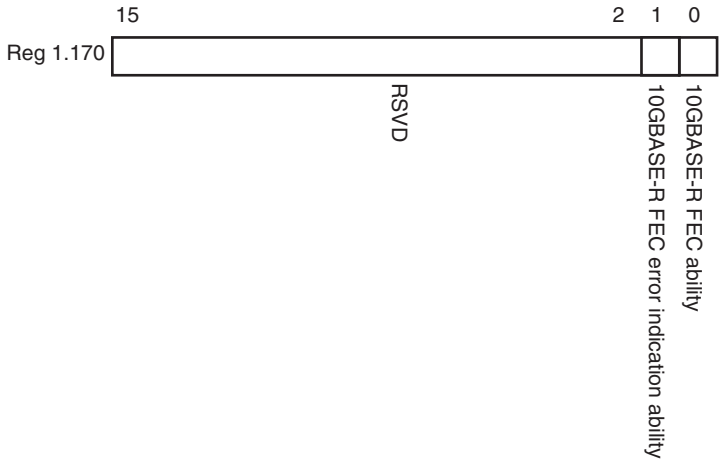

Figure 2-19: **10GBASE-R FEC Ability Register**

Table 2-30 shows the 10GBASE-R FEC Ability register bit definitions.

Table 2-30: **10GBASE-R FEC Ability Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
1.170.15:2	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	0s
1.170.1	10GBASE-R FEC error indication ability	1 = the PHY is able to report FEC decoding errors to the PCS layer	R/O	1
1.170.0	10GBASE-R FEC ability	1 = the PHY supports FEC	R/O	1

MDIO Register 1.171: 10GBASE-R FEC Control

Figure 2-20 shows the MDIO Register 1.170: 10GBASE-R FEC Control.

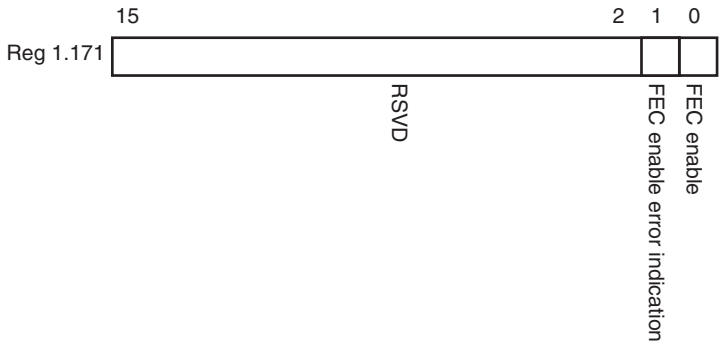

Figure 2-20: **10GBASE-R FEC Control Register**

Table 2-31 shows the 10GBASE-R FEC Control register bit definitions.

Table 2-31: **10GBASE-R FEC Control Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
1.171.15:2	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	0s
1.171.1	10GBASE-R FEC error indication ability[a]	1 = Configure the PHY to report FEC decoding errors to the PCS layer.	R/W	0
1.171.0	10GBASE-R FEC ability	1 = enable FEC 0 = disable FEC	R/W	0

a. If FEC Error Passing is enabled while FEC is enabled, errors will be seen temporarily. To avoid this, only enable Error Passing while FEC is disabled.

MDIO Register 1.172: 10GBASE-R FEC Corrected Blocks (Lower)

Figure 2-21 shows the MDIO Register 1.172: 10GBASE-R FEC Corrected Blocks (lower).

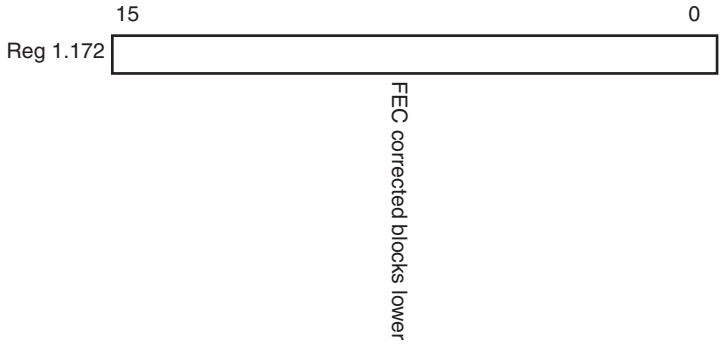

Figure 2-21: **10GBASE-R FEC Corrected Blocks (Lower) Register**

Table 2-32 shows the 10GBASE-R FEC Corrected Blocks (lower) register bit definitions.

Table 2-32: **10GBASE-R FEC Corrected Blocks (Lower) Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
1.172.15:0	FEC corrected blocks	Bits 15:0 of the Corrected Blocks count	R/O[a]	0s

a. Cleared when read.

MDIO Register 1.173: 10GBASE-R FEC Corrected Blocks (Upper)

Figure 2-22 shows the MDIO Register 1.173: 10GBASE-R FEC Corrected Blocks (upper).

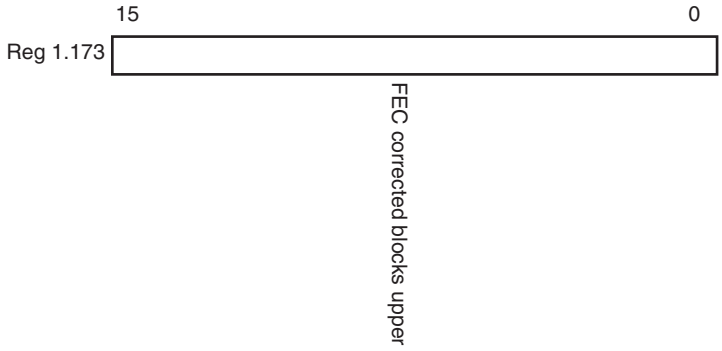

Figure 2-22: **10GBASE-R FEC Corrected Blocks (Upper) Register**

Table 2-33 shows the 10GBASE-R FEC Corrected Blocks (upper) register bit definitions.

Table 2-33: **10GBASE-R FEC Corrected Blocks (Upper) Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
1.173.15:0	FEC corrected blocks	Bits 31:16 of the Corrected Blocks count	R/O[a]	0s

a. Latched when 1.172 is read. Cleared when read.

MDIO Register 1.174: 10GBASE-R FEC Uncorrected Blocks (Lower)

Figure 2-23 shows the MDIO Register 1.174: 10GBASE-R FEC Uncorrected Blocks (lower).

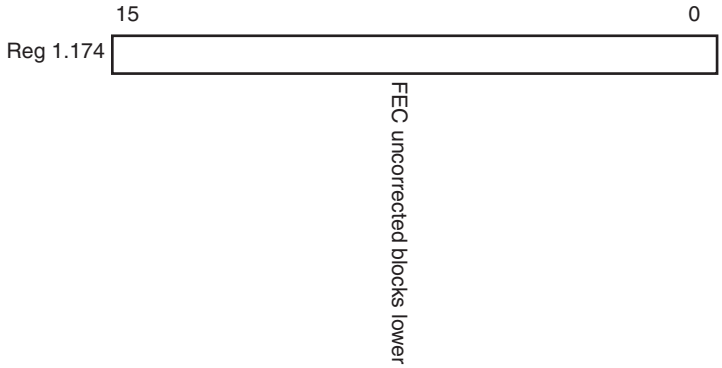

Figure 2-23: **10GBASE-R FEC Uncorrected Blocks (Lower) Register**

Table 2-34 shows the 10GBASE-R FEC Uncorrected Blocks (lower) register bit definitions.

Table 2-34: **10GBASE-R FEC Uncorrected Blocks (Lower) Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
1.174.15:0	FEC Uncorrected blocks	Bits 15:0 of the Uncorrected Blocks count	R/O[a]	0s

a. Cleared when read.

MDIO Register 1.175: 10GBASE-R FEC Uncorrected Blocks (Upper)

Figure 2-24 shows the MDIO Register 1.175: 10GBASE-R FEC Uncorrected Blocks (upper).

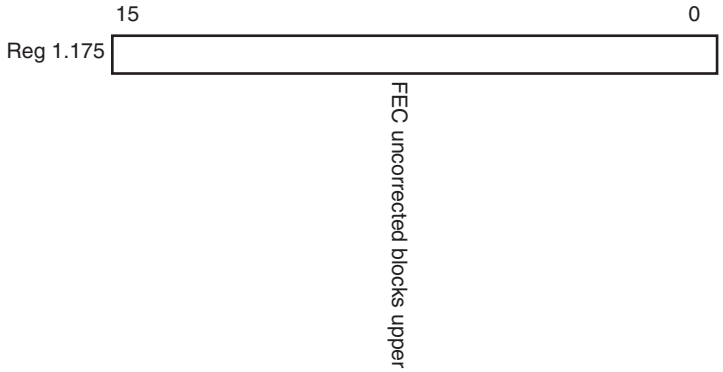

Figure 2-24: **10GBASE-R FEC Uncorrected Blocks (Upper) Register**

Table 2-35 shows the 10GBASE-R FEC Uncorrected Blocks (upper) register bit definitions.

Table 2-35: **10GBASE-R FEC Uncorrected Blocks (Upper) Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
1.175.15:0	FEC Uncorrected blocks	Bits 31:16 of the Uncorrected Blocks count	R/O[a]	0s

a. Latched when 1.174 is read. Cleared when read.

MDIO Register: 1.65520: Vendor-Specific LD Training

Figure 2-25 shows the MDIO Register 1.65520: Vendor-specific LD Training.

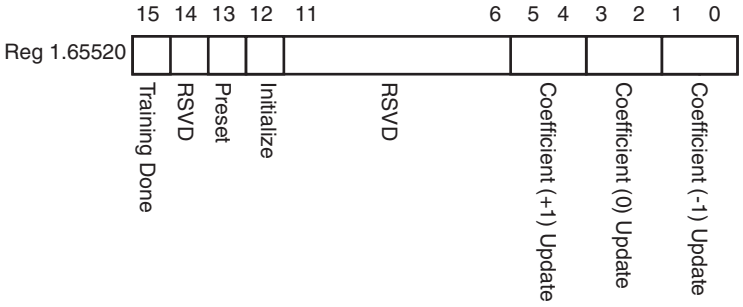

Figure 2-25: **Vendor-specific LD Training Register**

Table 2-36 shows the Vendor-specific LD Training register bit definitions.

Table 2-36: **Vendor-Specific LD Training Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
1.65520.15	Training Done	1 = Training Algorithm has determined that the LP transmitter has been successfully trained.	R/W[a]	0
1.65520.14	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	0
1.65520.13	Preset	1 = Preset coefficients 0 = Normal operation	R/O[b]	0
1.65520.12	Initialize	1 = Initialize coefficients 0 = Normal operation	R/O[b]	0
1.65520.5:4	Coefficient (+1) update	5:4 = 11 = reserved 10 = decrement 01 = increment 00 = hold	R/O[b]	00
1.65520.3:2	Coefficient (0) update	3:2 = 11 = reserved 10 = decrement 01 = increment 00 = hold	R/O[b]	00
1.65520.1:0	Coefficient (–1) update	1:0 = 11 = reserved 10 = decrement 01 = increment 00 = hold	R/O[b]	00

a. This register will be transferred automatically to register 1.155.15.
b. These registers will be transferred automatically to register 1.154.

MDIO Register 1.65535: Core Version Info

Figure 2-26 shows the MDIO 1.65535 Register: Core Version Info.

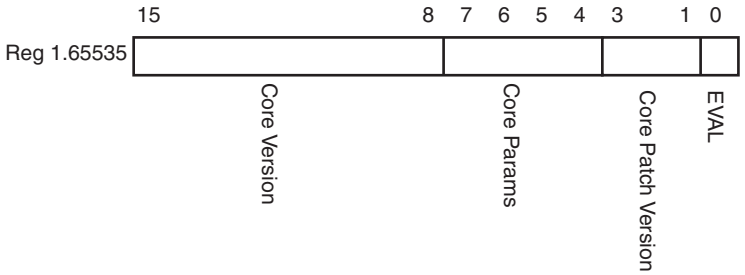

Figure 2-26: **Core Version Info Register**

Table 2-37: **Core Version Information**

Bits	Name	Description	Attributes	Default Value
1.65535.15:8	Core Version	Bits 15..12 give the major core version and bits 11..8 give the minor core version.	R/O	x'40' for version 4.0 of core
1.65535.7:4	Core parameters	Bit 7 = 1 = KR included Bit 6 – reserved Bit 5 = 1 = AN included Bit 4 = 1 = FEC included	R/O	Depends on core generation parameters
1.65535.3:1	Core Patch Version	Bits 3..1 give the patch number, if any, for the core.	R/O	'000'
1.65535.0	Zynq/V7/K7 only: EVAL	1 = This core was generated using a Hardware Evaluation license	R/O	'0'

MDIO Register 3.0: PCS Control 1

Figure 2-27 shows the MDIO Register 3.0: PCS Control 1.

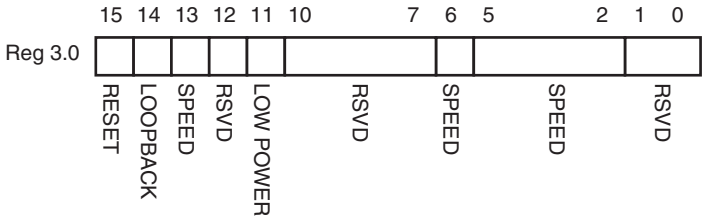

Figure 2-27: **PCS Control 1 Register**

Table 2-38 shows the PCS Control 1 register bit definitions.

Table 2-38: **PCS Control 1 Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
3.0.15	Reset	1 = Block reset 0 = Normal operation The 10GBASE-R/KR block is reset when this bit is set to 1. It returns to 0 when the reset is complete.	R/W Self-clearing	0
3.0.14	10GBASE-R/KR Loopback	1 = Use PCS Loopback 0 = Do not use PCS Loopback	R/W	0
3.0.13	Speed Selection	The block always returns 1 for this bit. 1 (and bit 6 = 1) = bits 5:2 select the speed	R/O	1
3.0.12	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	0
3.0.11	Power down	This bit has no effect.	R/W	0
3.0.10:7	Reserved	The block always returns 0 for these bits and ignores writes.	R/O	All 0s
3.0.6	Speed Selection	The block always returns 1 for this bit.	R/O	1
3.0.5:2	Speed Selection	The block always returns 0000 = 10Gb/s	R/O	All 0s
3.0.1:0	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	All 0s

MDIO Register 3.1: PCS Status 1

Figure 2-28 shows the MDIO Register 3.1: PCS Status 1.

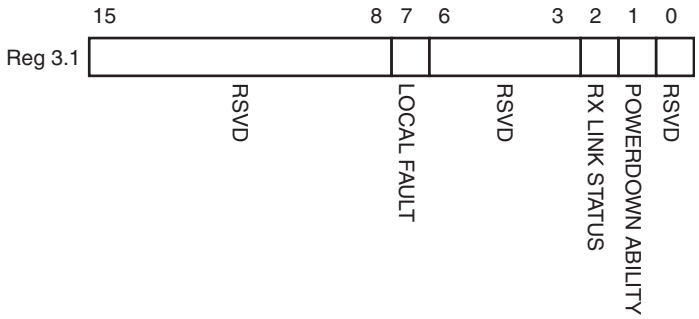

Figure 2-28: **PCS Status 1 Register**

Table 2-39 show the PCS 1 register bit definitions.

Table 2-39: **PCS Status 1 Register Bit Definition**

Bits	Name	Description	Attributes	Default Value
3.1.15:8	Reserved	The block always returns 0s for these bits and ignores writes.	R/O	All 0s
3.1.7	Local Fault	Zynq/V7/K7: 1 = Local Fault detected	R/O	0
3.1.6:3	Reserved	The block always returns 0s for these bits and ignores writes.	R/O	All 0s
3.1.2	PCS Receive Link Status	1 = The PCS receive link is up 0 = The PCS receive link is down This is a latching Low version of bit 3.32.12.	R/O Self-setting	-
3.1.1	Power Down Ability	The block always returns 1 for this bit.	R/O	1
3.1.0	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	0

MDIO Register 3.4: PCS Speed Ability

Figure 2-29 shows the MDIO Register 3.4: PCS Speed Ability.

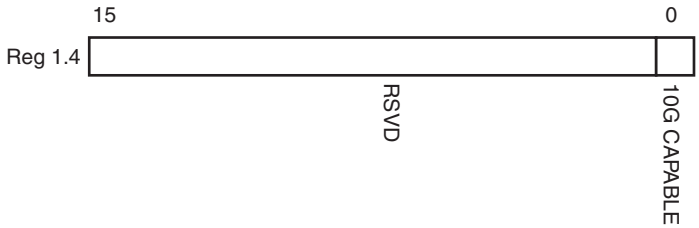

Figure 2-29: **PCS Speed Ability Register**

Table 2-40 shows the PCS Speed Ability register bit definitions.

Table 2-40: **PCS Speed Ability Register Bit Definitions**

Bits	Name	Description	Attribute	Default Value
3.4.15:1	Reserved	The block always returns 0 for these bits and ignores writes.	R/O	All 0s
3.4.0	10G Capable	The block always returns 1 for this bit and ignores writes.	R/O	1

MDIO Registers 3.5 and 3.6: PCS Devices in Package

Figure 2-30 shows the MDIO Registers 3.5 and 3.6: PCS Devices in Package.

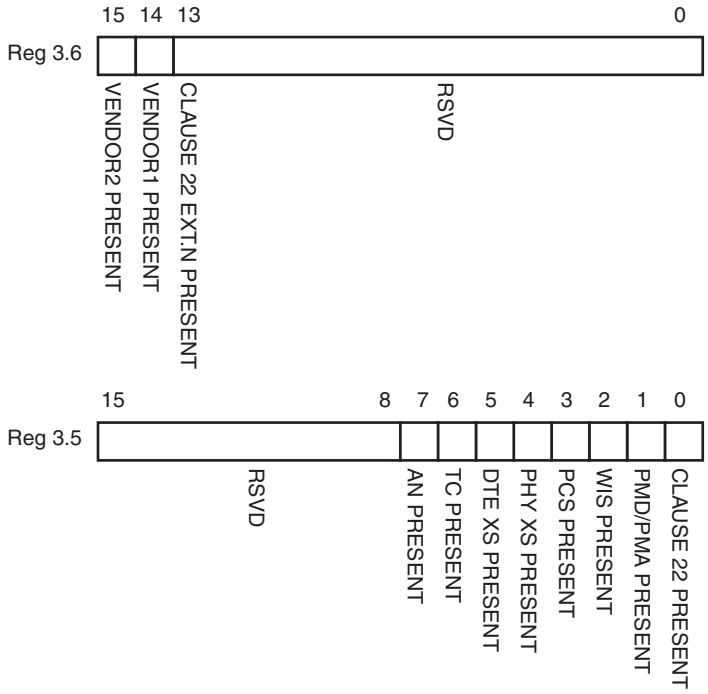

Figure 2-30: **PCS Devices in Package Registers**

Table 2-41 shows the PCS Devices in Package registers bit definitions.

Table 2-41: **PCS Devices in Package Registers Bit Definitions**

Bits	Name	Description	Attributes	Default Value
3.6.15	Vendor-specific Device 2 Present	The block always returns 0 for this bit.	R/O	0
3.6.14	Vendor- specific Device 1 Present	The block always returns 0 for this bit.	R/O	0
3.6.13	Clause 22 extension present	Zynq/V7/K7: The block always returns 0 for this bit.	1/0	1
3.6.12:0	Reserved	The block always returns 0 for these bits.	R/O	All 0s
3.5.15:8	Reserved	The block always returns 0 for these bits.	R/O	All 0s
3.5.7	Auto Negotiation Present	Zynq/V7/K7: 1 = AN Block included	1/0	1
3.5.6	TC present	The block always returns 0 for this bit.	R/O	0
3.5.5	PHY XS Present	The block always returns 0 for this bit.	R/O	0

Table 2-41: **PCS Devices in Package Registers Bit Definitions *(Cont'd)***

Bits	Name	Description	Attributes	Default Value
3.5.4	PHY XS Present	The block always returns 0 for this bit.	R/O	0
3.5.3	PCS Present	The block always returns 1 for this bit.	R/O	1
3.5.2	WIS Present	The block always returns 0 for this bit.	R/O	0
3.5.1	PMA/PMD Present	The block always returns 1 for this bit.	R/O	1
3.5.0	Clause 22 device present	The block always returns 0 for this bit.	R/O	0

MDIO Register 3.7: 10G PCS Control 2

Figure 2-31 shows the MDIO Register 3.7: 10G PCS Control 2.

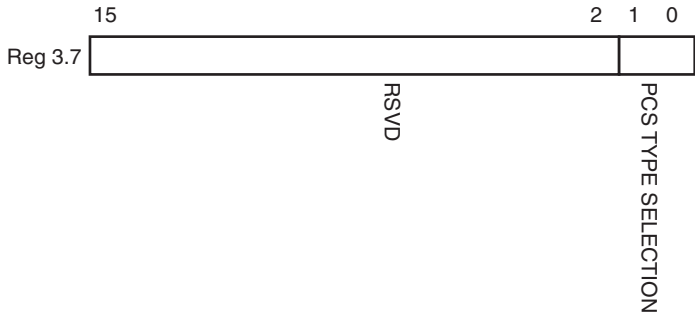

Figure 2-31: **10G PCS Control 2 Register**

Table 2-42 shows the 10 G PCS Control 2 register bit definitions.

Table 2-42: **10G PCS Control 2 Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
3.7.15:2	Reserved	The block always returns 0 for these bits and ignores writes.	R/O	All 0s
3.7.1:0	PCS Type Selection	"00" = Select 10GBASE-R PCS type. Any other value written to this register are ignored.	R/W	00

MDIO Register 3.8: 10G PCS Status 2

Figure 2-32 shows the MDIO Register 3.8: 10G PCS Status 2.

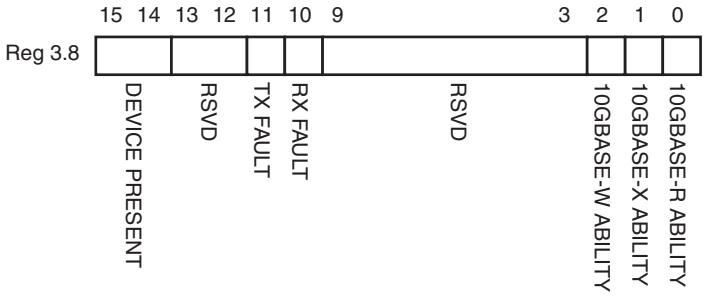

Figure 2-32: **10G PCS Status 2 Register**

Table 2-43 shows the 10G PCS Status 2 register bit definitions.

Table 2-43: **10G PCS Status 2 Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
3.8.15:14	Device present	The block always returns "10."	R/O	"10"
3.8.13:12	Reserved	The block always returns 0 for these bits.	R/O	All 0s
3.8.11	Transmit local fault	Zynq/V7/K7: 1 = Transmit Fault detected	R/O	0
3.8.10	Receive local fault	Zynq/V7/K7: 1 = Receive Fault detected	R/O	0
3.8.9:3	Reserved	The block always returns 0 for these bits.	R/O	All 0s
3.8.2	10GBASE-W Capable	The block always returns 0 for this bit.	R/O	0
3.8.1	10GBASE-X Capable	The block always returns 0 for this bit.	R/O	0
3.8.0	10GBASE-R Capable	The block always returns 1 for this bit.	R/O	1

MDIO Register 3.32: 10GBASE-R Status 1

Figure 2-33 shows the MDIO Register 3.32: 10GBASE-R Status 1.

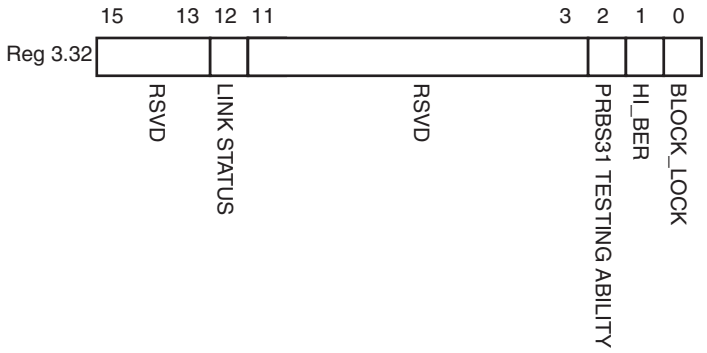

Figure 2-33: **10GBASE-R Status Register 1**

Table 2-44 shows the 10GBASE-R Status register bit definitions.

Table 2-44: **10GBASE-R Status Register 1 Bit Definitions**

Bits	Name	Description	Attributes	Default Value
3.32.15:13	Reserved	The block always returns 0 for these bits.	R/O	All 0s
3.32.12	10GBASE-R Link Status	1 = 10GBASE-R receive is aligned; 0 = 10GBASE-R receive is not aligned.	RO	0
3.32.11:3	Reserved	The block always returns 0 for these bits.	R/O	0s
3.32.2	PRBS31 Pattern Testing Ability	The block always returns 1 for this bit.	R/O	1
3.32.1	Hi BER	1 = RX showing hi-ber 0 = RX not showing hi ber	R/O	0
3.32.0	Block Lock	1 = RX is synchronized; 0 = RX is not synchronized.	R/O	0

MDIO Register 3.33: 10GBASE-R Status 2

Figure 2-34 shows the MDIO Register 3.33: 10GBASE-R Status 2.

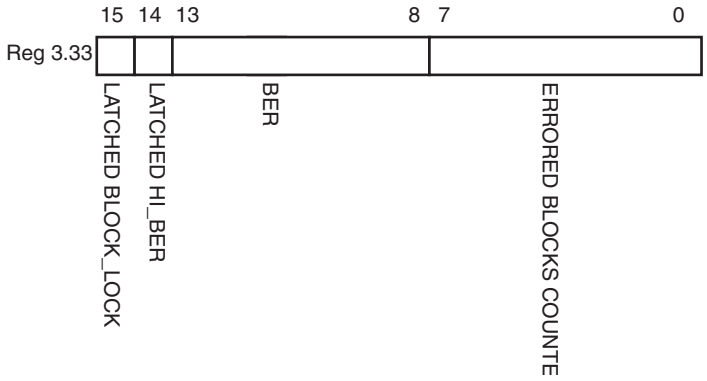

Figure 2-34: **10GBASE-R Status Register 2**

Table 2-45 shows the 10GBASE-R Status register bit definition. All bits are cleared when read.

Table 2-45: **10GBASE-R Status Register 2 Bit Definitions**

Bits	Name	Description	Attributes	Default Value
3.33.15	Latched Block Lock	Latch-Low version of block lock	R/O	0
3.33.14	Latched HiBER	Latch-High version of Hi BER	R/O	1
3.33.13:8	BER	BER Counter	R/O	0s
3.33.7:0	Errored Blocks Count	Counter for Errored Blocks	R/O	0s

MDIO Register 3.34–37: 10GBASE-R Test Pattern Seed A0–3

Figure 2-35 shows the MDIO Register 3.34–37 10GBASE-R Test Pattern Seed A.

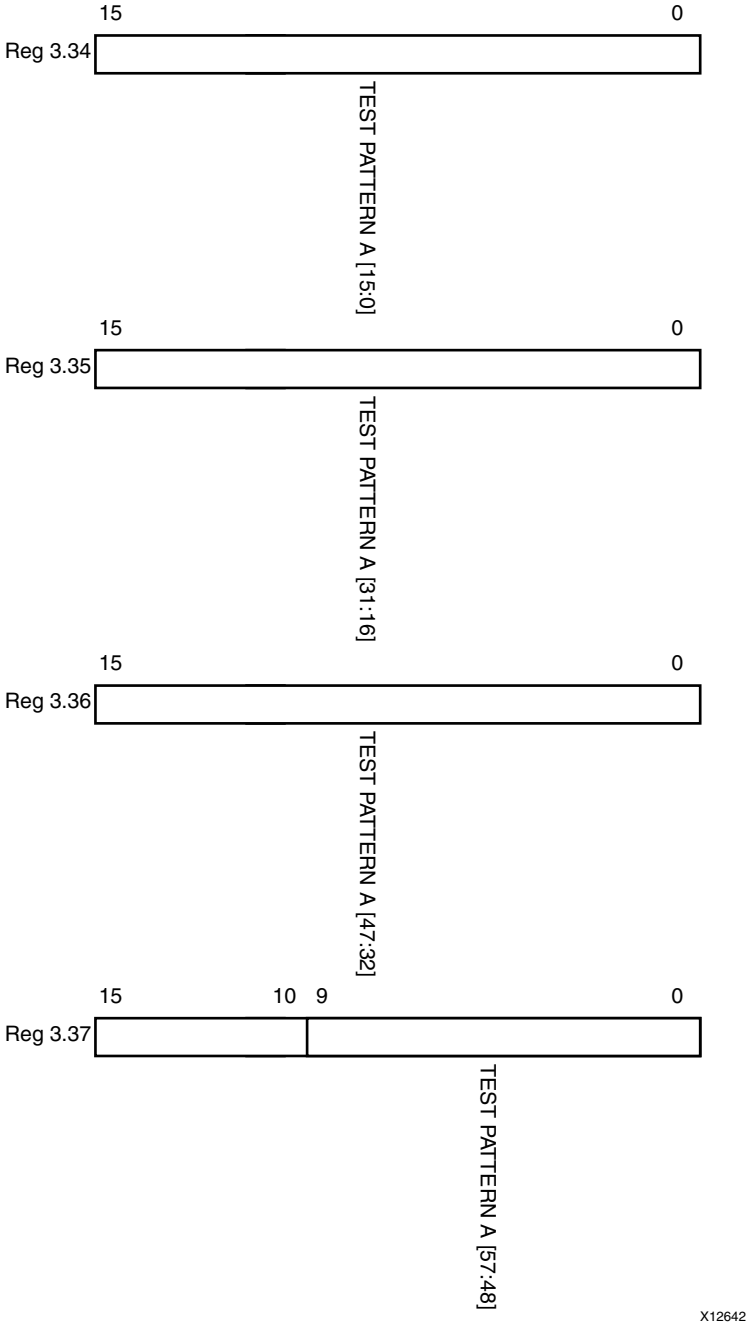

Figure 2-35: **10GBASE-R Test Pattern Seed A0-3 Registers**

Table 2-46 shows the 10GBASE-R Test Pattern Seed A0–2 register bit definitions.

Table 2-46: **10GBASE-R Test Pattern Seed A0-2 Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
3.34–36.15:0 3.37.9:0	Seed A bits 15:0, 31:16, 47:32, 57:48 resp	Seed for Pseudo-Random Test Pattern	R/W	Zynq/V7/K7: all 0s

MDIO Register 3.38–41: 10GBASE-R Test Pattern Seed B0–3

Figure 2-36 shows the MDIO Register 3.38–41: 10GBASE-R Test Pattern Seed B.

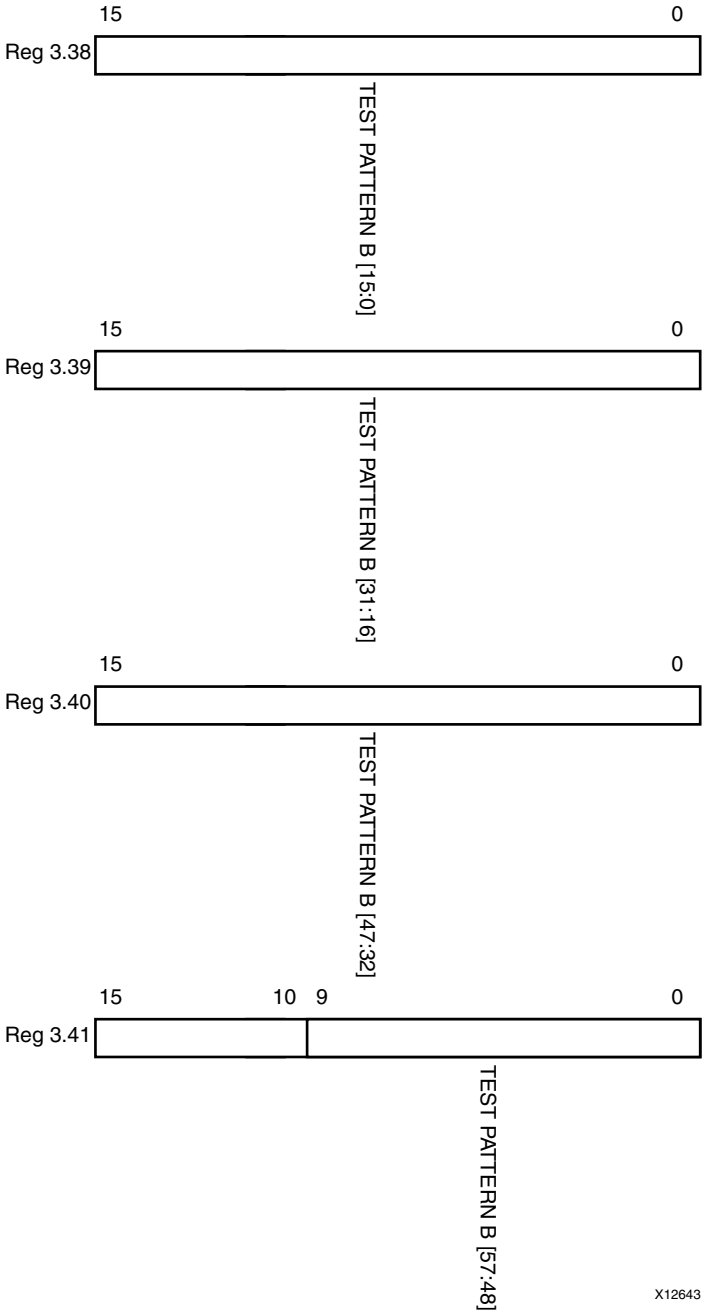

Figure 2-36: **10GBASE-R Test Pattern Seed B0-3 Registers**

Table 2-47 shows the 10GBASE-R Test Pattern Seed B0–3 register bit definitions.

Table 2-47: **10GBASE-R Test Pattern Seed B0-3 Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
3.38–40.15:0 3.41.9:0	Seed B bits 15:0, 31:16, 47:32, 57:48 resp	Seed for Pseudo-Random Test Pattern	R/W	Zynq/V7/K7: all 0s

MDIO Register 3.42: 10GBASE-R Test Pattern Control

Figure 2-37 shows the MDIO Register 3.42: 10GBASE-R Test Pattern Control.

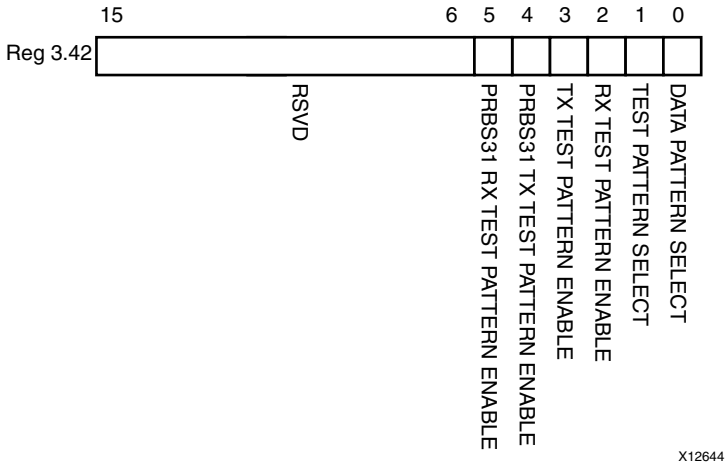

Figure 2-37: **10GBASE-R Test Pattern Control Register**

Table 2-48 shows the 10GBASE-R Test Pattern Control register bit definitions.

Table 2-48: **10GBASE-R Test Pattern Control Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
3.42.15:6	Reserved	The block always returns 0s for these bits.	R/O	All 0s
3.42.5	PRBS31 RX test pattern enable	1 = Enable PRBS RX tests 0 = Disable PRBS RX tests	R/W	0
3.42.4	PRBS31 TX test pattern enable	1 = Enable PRBS TX tests 0 = Disable PRBS TX tests	R/W	0
3.42.3	TX test pattern enable	Enables the TX Test Pattern which has been selected with bits [1:0].	R/W	0
3.42.2	RX test pattern enable	Enables the RX Test Pattern Checking which has been selected with bits [1:0]	R/W	0
3.42.1	Test pattern select	1 = Square wave 0 = Pseudo-Random	R/W	0

Table 2-48: **10GBASE-R Test Pattern Control Register Bit Definitions *(Cont'd)***

Bits	Name	Description	Attributes	Default Value
3.42.0	Data pattern select	1 = Zeros pattern 0 = LF Data pattern	R/W	0

1. PRBS31 test pattern generation and checking is implemented in the transceiver and the error count is read by the 10GBASE-R/KR core through the transceiver DRP interface. All other test pattern generation and checking where applicable is implemented in the PCS logic in the core

MDIO Register 3.43: 10GBASE-R Test Pattern Error Counter

Figure 2-38 shows the MDIO Register 3.43: 10GBASE-R Test Pattern Error Counter.

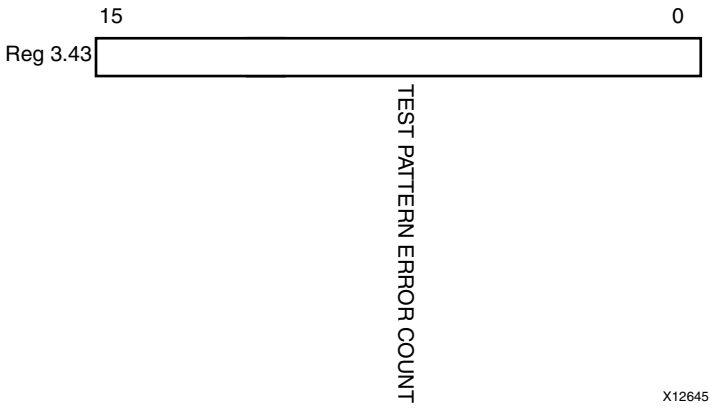

Figure 2-38: **10GBASE-R Test Pattern Error Counter Register**

Table 2-49 shows the 10GBASE-R Test Pattern Error Counter register bit definitions. This register is cleared when read.

Table 2-49: **10GBASE-R Test Pattern Error Counter Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
3.43.15:0	Test pattern error counter	Count of errors	R/O	All 0s

MDIO Register 3.65535: 125 Microsecond Timer Control

Figure 2-39 shows the MDIO 3.65535 Register: 125 microsecond timer control.

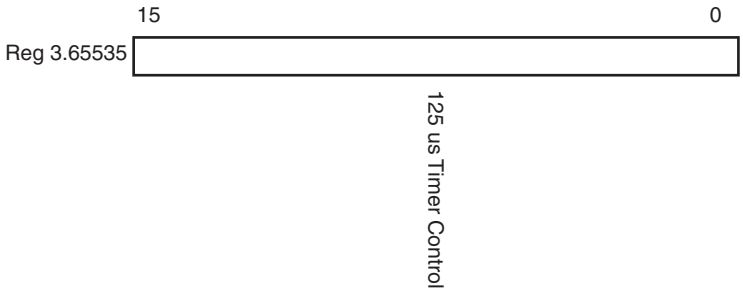

Figure 2-39: **125 Microsecond Timer Control Register**

Table 2-50: **125 µs Timer Control**

Bits	Name	Description	Attributes	Default Value
3.65535.15:0	125 µs timer control	Bits 15..0 set the number of clock cycles at 156.25 MHz to be used to measure the 125 µs timer in the BER monitor state machine. Useful for debug purposes (simulation speedup).	R/W	x'4C4B'

MDIO Register 7.0: AN Control

Figure 2-40 shows the MDIO Register 7.0: AN Control.

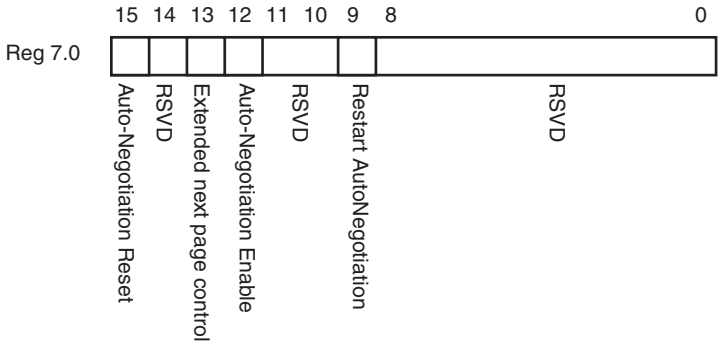

Figure 2-40: **AN Control Register**

Table 2-51 shows the AN Control register bit definitions.

Table 2-51: **AN Control Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
7.0.15	AN Reset	1 = AN Reset 0 = AN normal operation	R/W Self-clearing	0
7.0.14	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	0
7.0.13	Extended Next Page control	1 = Extended Next Pages are supported 0 = Not supported	R/W	0
7.0.12	AN Enable	1 = Enable AN Process 0 = Disable	R/W[a]	1
7.0.11:10	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	00
7.0.9	Restart AN	1 = Restart AN process 0 = Normal operation	R/W Self-clearing	0
7.0.8:0	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	0s

a. *For simulation purposes only*, to disable AN at start-up, the external core pin 'an_enable' should be tied Low.

MDIO Register 7.1: AN Status

Figure 2-41 shows the MDIO Register 7.1: AN Status.

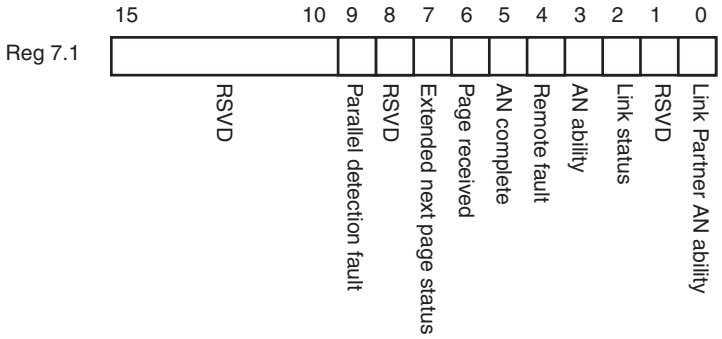

Figure 2-41: **AN Status Register**

Table 2-52 shows the AN Status register bit definitions.

Table 2-52: **AN Status Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
7.1.15:10	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	0s
7.1.9	Parallel Detection Fault	1 = A fault has been detected through the parallel detection function 0 = no fault detected	R/O Latches High	0
7.1.8	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	0
7.1.7	Extended Next Page status	1 = XNP format is used 0 = XNP format is not allowed	R/O	0
7.1.6	Page Received	1 = A page has been received 0 = No page received	R/O Latches High	0
7.1.5	AN Complete	1 = AN process has completed 0 = not completed	R/O	0
7.1.4	Remote fault	1 = remote fault condition detected 0 = not detected	R/O Latches High	0
7.1.3	AN ability	1 = PHY supports auto-negotiation 0 = PHY does not support auto-negotiation	R/O	1
7.1.2	Link status	1 = Link is up 0 = Link is down	R/O Latches Low	0

Table 2-52: **AN Status Register Bit Definitions *(Cont'd)***

Bits	Name	Description	Attributes	Default Value
7.1.1	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	0
7.1.0	LP AN ability	1 = LP is able to perform AN 0 = not able	R/O	0

MDIO Register 7.16:17:18: AN Advertisement

Figure 2-42 shows the MDIO Register 7.16: AN Advertisement.

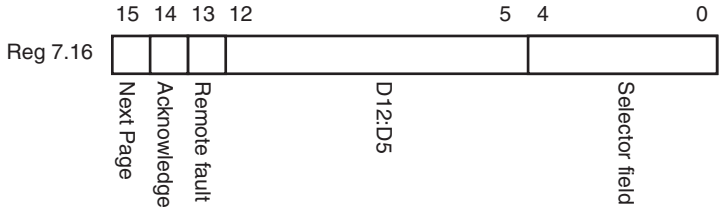

Figure 2-42: **AN Advertisement Register 0**

Table 2-53 shows the AN Advertisement register bit definitions.

Table 2-53: **AN Advertisement Register 0 Bit Definitions**

Bits	Name	Description	Attributes	Default Value
7.16.15	Next Page	Consult IEEE802.3	R/W	0
7.16.14	Acknowledge	The block always returns 0 for this bit and ignores writes.	R/O	0
7.16.13	Remote Fault	Consult IEEE802.3	R/W	0
7.16.12:5	D12:D5	Consult IEEE802.3	R/W	0s
7.16.4:0	Selector Field	Consult IEEE802.3	R/W	00001s

Figure 2-43 shows the MDIO Register 7.17: AN Advertisement.

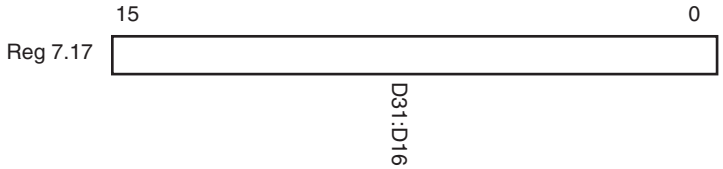

Figure 2-43: **AN Advertisement Register 1**

Table 2-54 shows the AN Advertisement register bit definitions.

Table 2-54: **AN Advertisement Register 1 Bit Definitions**

Bits	Name	Description	Attributes	Default Value
7.17.15:0	D31:D16	Consult IEEE802.3	R/W	0

Figure 2-44 shows the MDIO Register 7.18: AN Advertisement.

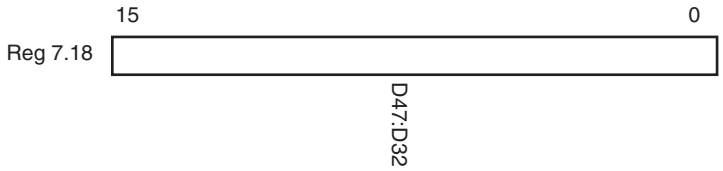

Figure 2-44: **AN Advertisement Register 2**

Table 2-55 shows the AN Advertisement register bit definitions.

Table 2-55: **AN Advertisement Register 2 Bit Definitions**

Bits	Name	Description	Attributes	Default Value
7.18.15:0	D47:D32	Consult IEEE802.3	R/W	0

MDIO Register 7.19, 20, 21: AN LP Base Page Ability

Figure 2-45 shows the MDIO Register 7.19: AN LP Base Page Ability.

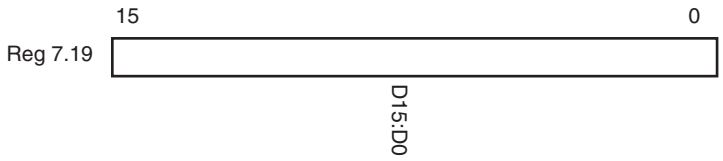

Figure 2-45: **AN LP Base Page Ability Register 0**

Table 2-56 shows the AN LP Base Page Ability register bit definitions.

Table 2-56: **AN LP Base Page Ability Register 0 Bit Definitions**

Bits	Name	Description	Attributes	Default Value
7.19.15:0	D15:D0	Consult IEEE802.3	R/O	0

Figure 2-46 shows the MDIO Register 7.20: AN LP Base Page Ability.

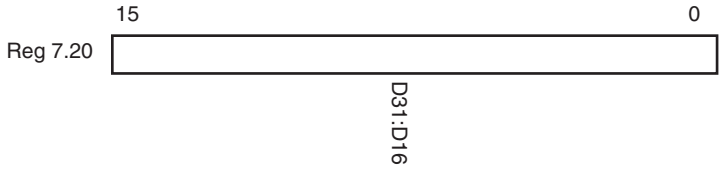

Figure 2-46: **AN LP Base Page Ability Register 1**

Table 2-57 shows the AN LP Base Page Ability register bit definitions.

Table 2-57: **AN LP Base Page Ability Register 1 Bit Definitions**

Bits	Name	Description	Attributes	Default Value
7.20.15:0	D31:D16	Consult IEEE802.3	R/W	0

Figure 2-47 shows the MDIO Register 7.21: AN LP Base Page Ability.

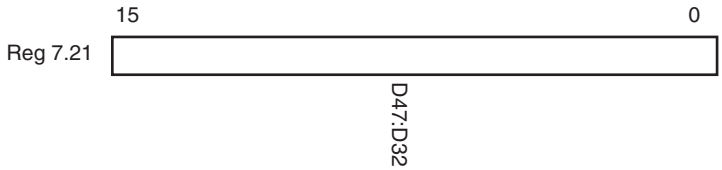

Figure 2-47: **AN LP Base Page Ability Register 2**

Table 2-58 shows the AN LP Base Page Ability register bit definitions.

Table 2-58: **AN LP Base Page Ability Register 2 Bit Definitions**

Bits	Name	Description	Attributes	Default Value
7.21.15:0	D47:D32	Consult IEEE802.3	R/W	0

MDIO Register 7.22, 23, 24: AN XNP Transmit

Figure 2-48 shows the MDIO Register 7.22: AN XNP Transmit.

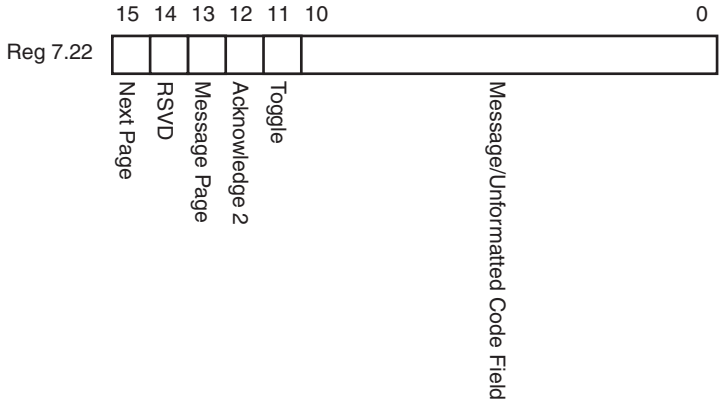

Figure 2-48: **AN XNP Transmit Register 0**

Table 2-59 shows the AN XNP Transmit register bit definitions.

Table 2-59: **AN XNP Transmit Register 0 Bit Definitions**

Bits	Name	Description	Attributes	Default Value
7.22.15	Next Page	Consult IEEE802.3	R/W	0
7.22.14	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	0
7.22.13	Message Page	Consult IEEE802.3	R/W	0
7.22.12	Acknowledge 2	Consult IEEE802.3	R/W	0
7.22.11	Toggle	Consult IEEE802.3	R/O	0
7.22.10:0	Message/Unformatted Code Field	Consult IEEE802.3	R/W	0s

Figure 2-49 shows the MDIO Register 7.23: AN XNP Transmit.

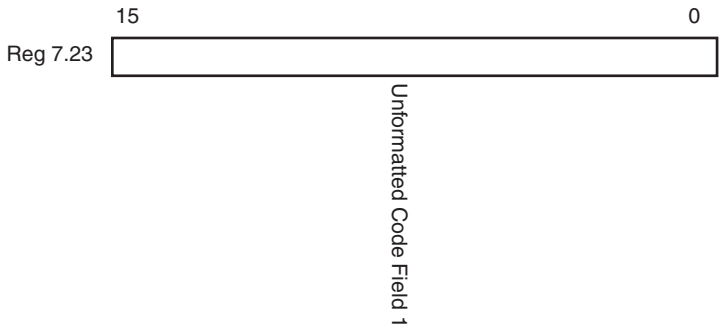

Figure 2-49: **AN XNP Transmit Register 1**

Table 2-60 shows the AN XNP Transmit register bit definitions.

Table 2-60: **AN XNP Transmit Register 1 Bit Definitions**

Bits	Name	Description	Attributes	Default Value
7.23.15:0	Unformatted Code Field 1	Consult IEEE802.3	R/W	0s

Figure 2-50 shows the MDIO Register 7.24: AN XNP Transmit.

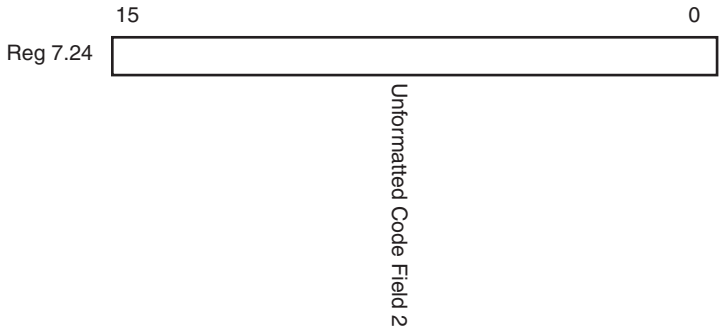

Figure 2-50: **AN XNP Transmit Register 2**

Table 2-61 shows the AN XNP Transmit register bit definitions.

Table 2-61: **AN XNP Transmit Register 2 Bit Definitions**

Bits	Name	Description	Attributes	Default Value
7.24.15:0	Unformatted Code Field 2	Consult IEEE802.3	R/W	0s

MDIO Register 7.25, 26, 27: AN LP XNP Ability

Figure 2-51 shows the MDIO Register 7.25: AN LP XNP Ability.

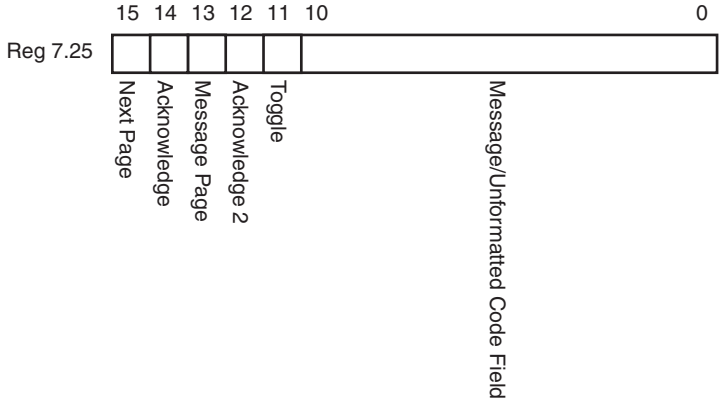

Figure 2-51: **AN LP XNP Ability Register 0**

Table 2-62 shows the AN LP XNP Ability register bit definitions.

Table 2-62: **AN LP XNP Ability Register 0 Bit Definitions**

Bits	Name	Description	Attributes	Default Value
7.25.15	Next Page	Consult IEEE802.3	R/O	0
7.25.14	Acknowledge	Consult IEEE802.3	R/O	0
7.25.13	Message Page	Consult IEEE802.3	R/O	0
7.25.12	Acknowledge 2	Consult IEEE802.3	R/O	0
7.25.11	Toggle	Consult IEEE802.3	R/O	0
7.25.10:0	Message/Unformatted Code Field	Consult IEEE802.3	R/O	0s

Figure 2-52 shows the MDIO Register 7.26: AN LP XNP Ability.

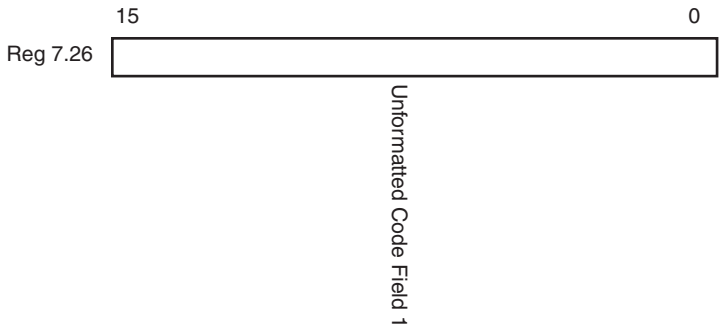

Figure 2-52: **AN LP XNP Ability Register 1**

Table 2-63 shows the AN LP XNP Ability register bit definitions.

Table 2-63: **AN LP XNP Ability Register 1 Bit Definitions**

Bits	Name	Description	Attributes	Default Value
7.26.15:0	Unformatted Code Field 1	Consult IEEE802.3	R/O	0s

Figure 2-53 shows the MDIO Register 7.27: AN LP XNP Ability.

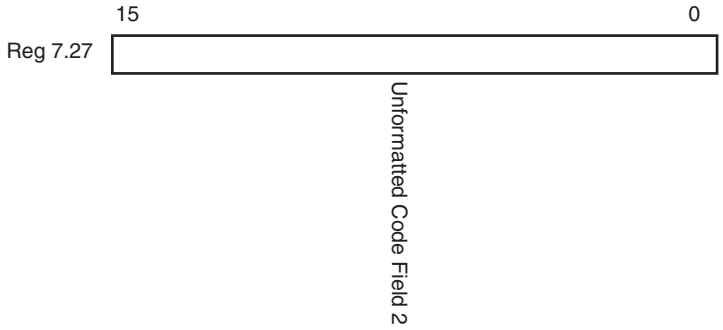

Figure 2-53: **AN LP XNP Ability Register 2**

Table 2-64 shows the AN LP XNP Ability register bit definitions.

Table 2-64: **AN LP XNP Ability Register 2 Bit Definitions**

Bits	Name	Description	Attributes	Default Value
7.27.15:0	Unformatted Code Field 2	Consult IEEE802.3	R/O	0s

MDIO Register 7.48: Backplane Ethernet Status

Figure 2-54 shows the MDIO Register 7.48: Backplane Ethernet Status.

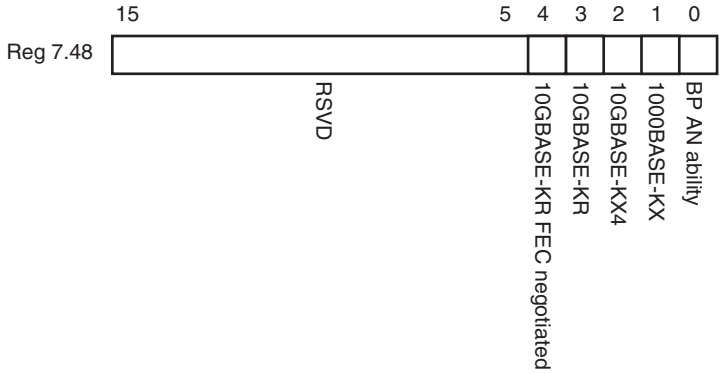

Figure 2-54: **Backplane Ethernet Status Register**

Table 2-65 shows the Backplane Ethernet Status register bit definitions.

Table 2-65: **Backplane Ethernet Status Register Bit Definitions**

Bits	Name	Description	Attributes	Default Value
7.48.15:5	Reserved	The block always returns 0 for this bit and ignores writes.	R/O	0
7.48.4	10GBASE-KR FEC negotiated	1 = PMA/PMD is negotiated to perform 10GBASE-KR FEC 0 = not negotiated	R/O	0
7.48.3	10GBASE-KR	1 = PMA/PMD is negotiated to perform 10GBASE-KR 0 = not negotiated	R/O	0
7.48.2	10GBASE-KX4	1 = PMA/PMD is negotiated to perform 10GBASE-KX4 0 = not negotiated	R/O	0
7.48.1	1000GBASE-KX	1 = PMA/PMD is negotiated to perform 1000GBASE-KX 0 = not negotiated	R/O	0
7.48.0	BP AN ability	1 = PMA/PMD is able to perform one of the preceding protocols 0 = not able	R/O	1

Designing with the Core

This chapter provides a general description of how to use the 10GBASE-R/KR core in your designs as well as describing specific core interfaces.

This chapter also describes the steps required to turn a 10GBASE-R/KR core into a fully-functioning design with user-application logic. It is important to realize that not all implementations require all of the design steps listed in this chapter. Follow the logic design guidelines in this manual carefully.

General Design Guidelines

Use the Example Design as a Starting Point

Each instance of the 10GBASE-R/KR core created by the Vivado® design tool is delivered with an example design that can be implemented in an FPGA and simulated.

This design can be used as a starting point for your own design or can be used to sanity-check your application in the event of difficulty.

See Chapter 8, Detailed Example Design, for information about using and customizing the example designs for the 10GBASE-R/KR core.

Know the Degree of Difficulty

10GBASE-R/KR designs are challenging to implement in any technology, and the degree of difficulty is further influenced by:

- Maximum system clock frequency
- Targeted device architecture
- Nature of your application

All 10GBASE-R/KR implementations need careful attention to system performance requirements. Pipelining, logic mapping, placement constraints, and logic duplication are all methods that help boost system performance.

Keep It Registered

To simplify timing and increase system performance in an FPGA design, keep all inputs and outputs registered between your application and the core. This means that all inputs and outputs from your application should come from, or connect to a flip-flop. While registering signals cannot be possible for all paths, it simplifies timing analysis and makes it easier for the Xilinx tools to place and route the design.

Recognize Timing Critical Signals

The timing constraint file that is provided with the example design for the core identifies the critical signals and the timing constraints that should be applied. See Chapter 5, Constraining the Core for further information.

Use Supported Design Flows

See Chapter 4, Customizing and Generating the Core for more information.

Make Only Allowed Modifications

The 10GBASE-R/KR core is not user-modifiable. Do not make modifications as they can have adverse effects on system timing and protocol functionality. Supported user configurations of the 10GBASE-R/KR core can only be made by selecting the options from within the IP catalog when the core is generated. See Chapter 4, Customizing and Generating the Core.

Shared Logic

Until recently the RTL hierarchy for the core was fixed. This resulted in some difficulty because shareable clocking and reset logic needed to be extracted from the core example design for use with a single instance or multiple instances of the core. Shared Logic is a new feature that provides a more flexible architecture that works both as a standalone core and as a part of a larger design with one or more core instances. This minimizes the amount of HDL modifications required, but at the same time retains the flexibility to address more uses of the core.

The new level of hierarchy is called `<component_name>_support`. Figure 3-1 and Figure 3-2 show two hierarchies where the shared logic block is contained either in the core or in the example design. In these figures, `<component_name>` is the name of the generated core. The difference between the two hierarchies is the boundary of the core. It is controlled using the Shared Logic option in the Vivado IDE (see Figure 4-1).

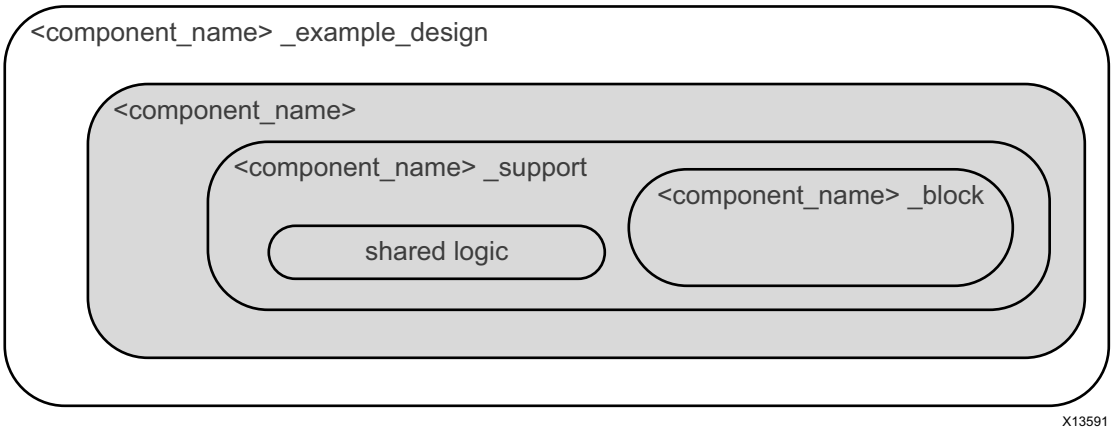

Figure 3-1: **Shared Logic Included in Core**

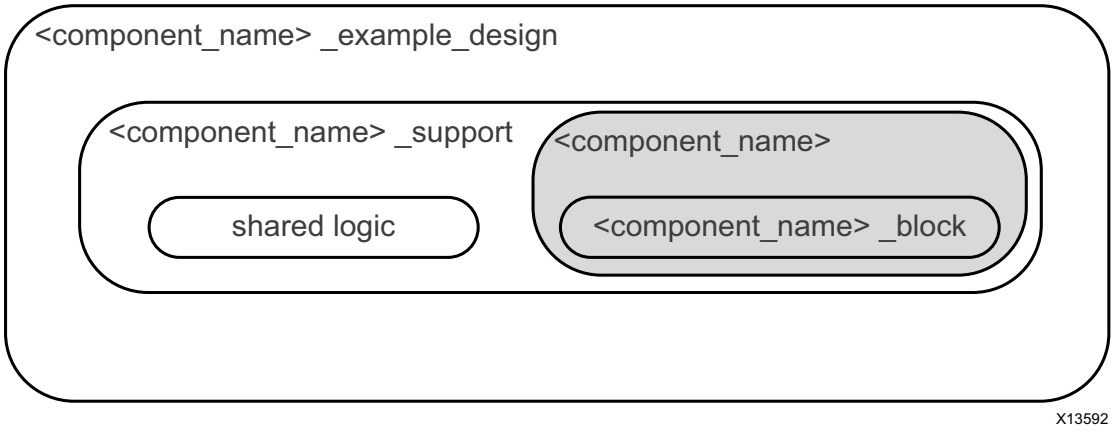

Figure 3-2: **Shared Logic Included in Example Design**

Interfacing to the Core

Data Interface: Internal Interfaces

Internal 64-bit SDR Client-side Interface

The 64-bit single-data rate (SDR) client-side interface is based upon the 32-bit XGMII interface. The bus is demultiplexed from 32-bits wide to 64-bits wide on a single rising clock edge. This demultiplexing is done by extending the bus upwards so that there are now eight lanes of data numbered 0–7; the lanes are organized such that data appearing on lanes 4–7 is transmitted or received *later* in time than that in lanes 0–3.

The mapping of lanes to data bits is shown in Table 3-1. The lane number is also the index of the control bit for that particular lane; for example, `xgmii_txc[2]` and `xgmii_txd[23:16]` are the control and data bits respectively for lane 2.

Table 3-1: **XGMII_TXD, XGMII_RXD Lanes for Internal 64-bit Client-Side Interface**

Lane	XGMII_TXD, XGMII_RXD Bits
0	7:0
1	15:8
2	23:16
3	31:24
4	39:32
5	47:40
6	55:48
7	63:56

Definitions of Control Characters

Reference is regularly made to certain XGMII control characters signifying Start, Terminate, Error, and so forth. These control characters all have in common that the control line for that lane is 1 for the character and a certain data byte value. The relevant characters are defined in the *IEEE Std. 802.3-2012* and are reproduced in Table 3-2 for reference.

Table 3-2: **Partial list of XGMII Characters**

Data (Hex)	Control	Name, Abbreviation
00 to FF	0	Data (D)
07	1	Idle (I)
FB	1	Start (S)
FD	1	Terminate (T)
FE	1	Error (E)

Interfacing to the Transmit Client Interface

Internal 64-bit Client-Side Interface

The timing of a data frame transmission through the internal 64-bit client-side interface is shown in Figure 3-3. The beginning of the data frame is shown by the presence of the Start character (the /S/ codegroup in lane 4 of Figure 3-3) followed by data characters in lanes 5, 6, and 7. Alternatively the start of the data frame can be marked by the occurrence of a Start character in lane 0, with the data characters in lanes 1 to 7.

When the frame is complete, it is completed by a Terminate character (the T in lane 1 of Figure 3-3). The Terminate character can occur in any lane; the remaining lanes are padded by XGMII idle characters.

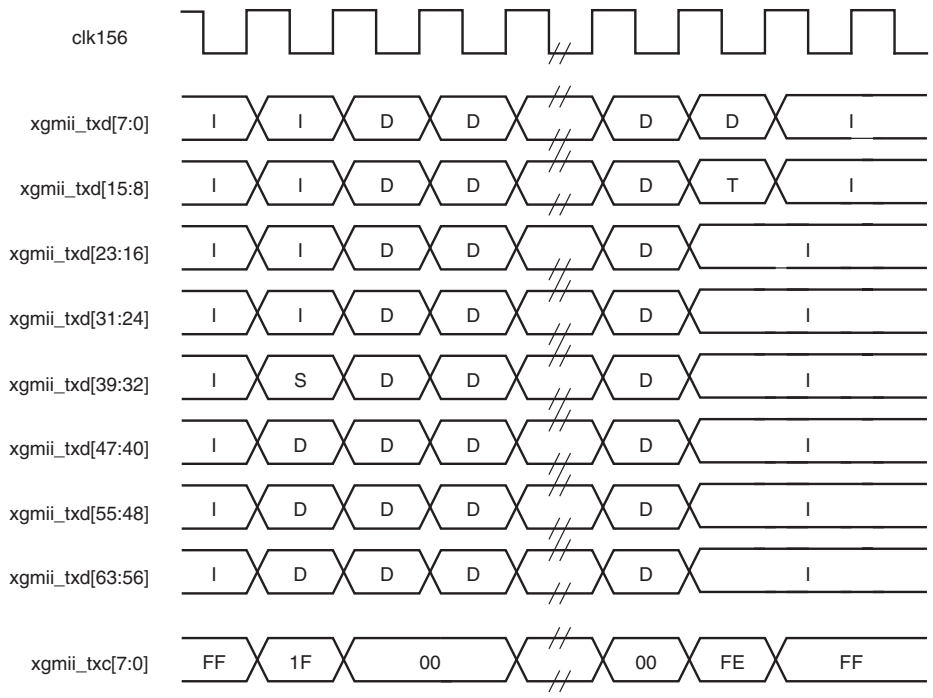

Figure 3-3: **Normal Frame Transmission Across the Internal 64-bit Client-Side I/F**

Figure 3-4 depicts a similar frame to that in Figure 3-3, with the exception that this frame is propagating an error. The error code is denoted by the letter E, with the relevant control bits set.

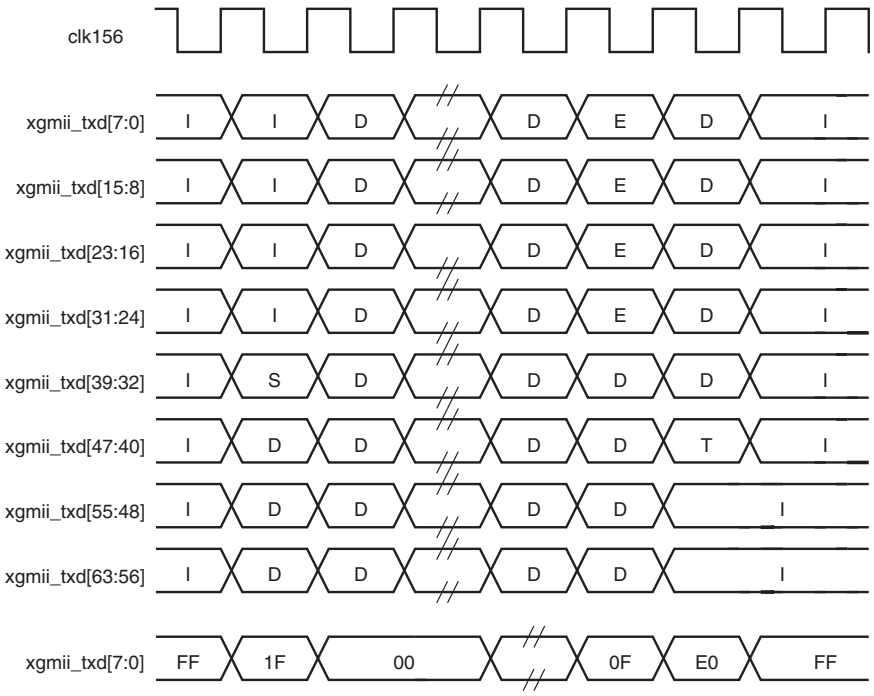

Figure 3-4: **Frame Transmission with Error Across Internal 64-bit Client-Side I/F**

Interfacing to the Receive Client Interface

Internal 64-bit Client-Side Interface

The timing of a normal inbound frame transfer is shown in Figure 3-5. As in the transmit case, the frame is delimited by a Start character (S) and by a Terminate character (T). The Start character in this implementation can occur in either lane 0 or in lane 4. The Terminate character, T, can occur in any lane.

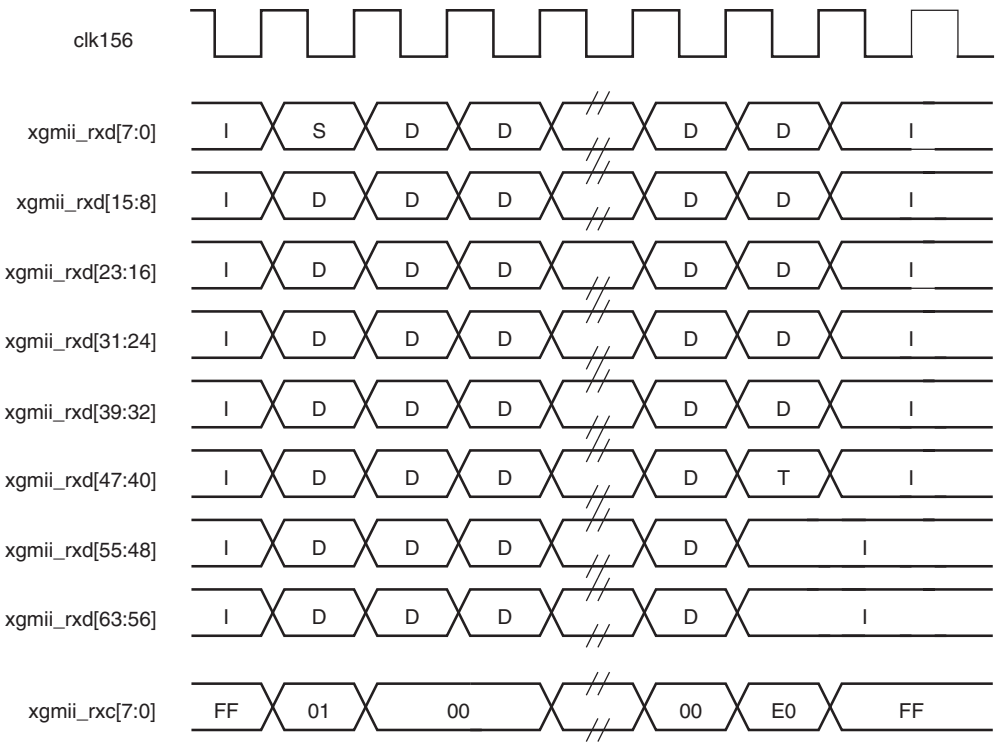

Figure 3-5: **Frame Reception Across the Internal 64-bit Client Interface**

Figure 3-6 shows an inbound frame of data propagating an error. In this instance, the error is propagated in lanes 4 to 7, shown by the letter E.

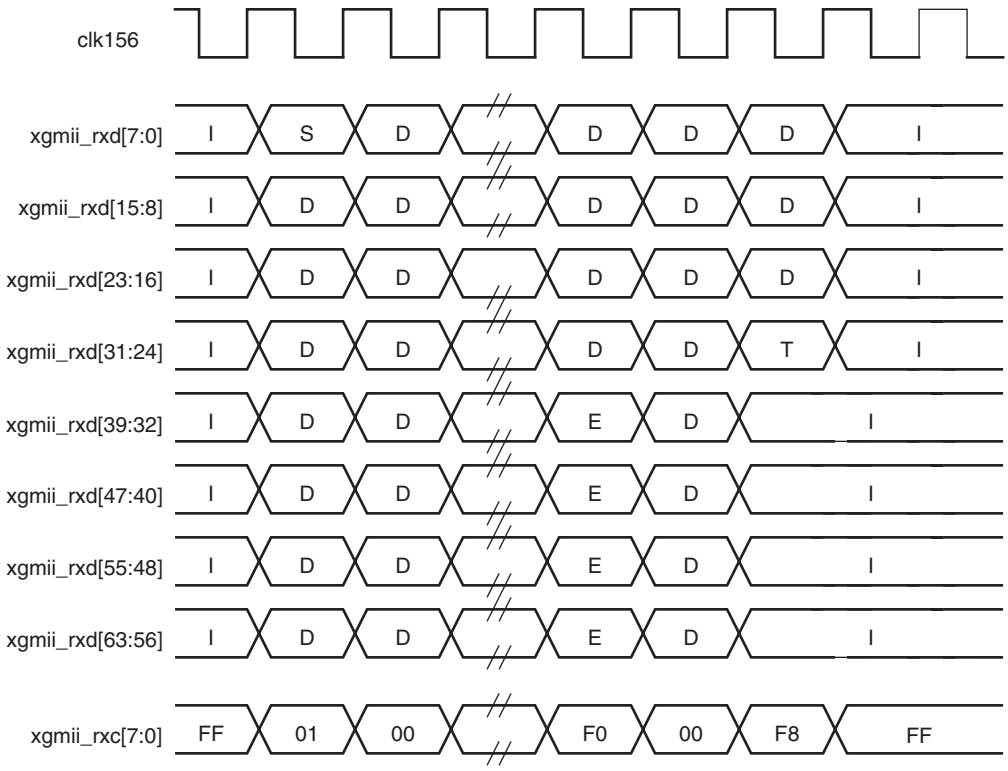

Figure 3-6: **Frame Reception with Error Across the Internal 64-bit Client Interface**

Configuration and Status Interfaces

This section describes the interfaces available for dynamically setting the configuration and obtaining the status of the 10GBASE-R/KR core. There are two interfaces for configuration; depending on the core customization, only one is available in a particular core instance. The interfaces are:

- MDIO Interface
- Configuration and Status Vectors

MDIO Interface

The Management Data Input/Output (MDIO) interface is a simple, low-speed 2-wire interface for management of the 10GBASE-R/KR core consisting of a clock signal and a bidirectional data signal. It is defined in clause 45 of *IEEE Standard 802.3-2012*.

An MDIO bus in a system consists of a single Station Management (STA) master management entity and several MDIO Managed Device (MMD) slave entities. Figure 3-7 illustrates a typical system. All transactions are initiated by the STA entity. The 10GBASE-R/KR core implements an MMD.

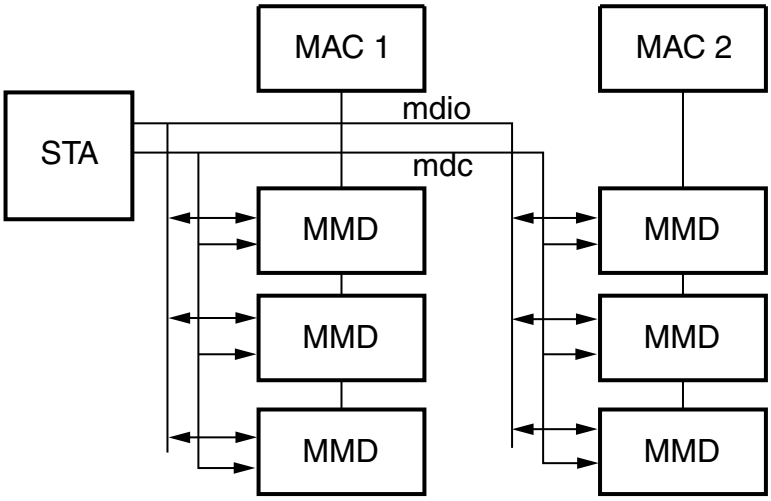

Figure 3-7: **A Typical MDIO-Managed System**

MDIO Ports

The core ports associated with MDIO are shown in Table 3-3.

Table 3-3: **MDIO Management Interface Port Description**

Signal Name	Direction	Description
mdc	IN	Management clock
mdio_in	IN	MDIO input
mdio_out	OUT	MDIO output
mdio_tri	OUT	MDIO 3-state. 1 disconnects the output driver from the MDIO bus.
prtad[4:0]	IN	MDIO port address

If implemented, the MDIO interface is implemented as four unidirectional signals. These can be used to drive a 3-state buffer either in the FPGA SelectIO™ interface buffer or in a separate device.

The `prtad[4:0]` port sets the port address of the core instance. Multiple instances of the same core can be supported on the same MDIO bus by setting the `prtad[4:0]` to a unique value for each instance; the 10GBASE-R/KR core ignores transactions with the PRTAD field set to a value other than that on its `prtad[4:0]` port.

MDIO Transactions

The MDIO interface should be driven from a STA master according to the protocol defined in *IEEE Std. 802.3-2012*. An outline of each transaction type is described in the following sections. In these sections, these abbreviations apply:

- PRE: preamble
- ST: start
- OP: operation code
- PRTAD: port address
- DEVAD: device address
- TA: turnaround

DEVAD

The device address in this case will be either “00001” for the PMA device or “00011” for the PCS device. For BASE-KR cores that include the optional Autonegotiation block, a DEVAD of "00111" should be used to access the associated registers.

Set Address Transaction

Figure 3-8 shows an Address transaction defined by OP='00.' Set Address is used to set the internal 16-bit address register which is particular to the given DEVAD, for subsequent data transactions (called the "current address" in the following sections). The core contains two such address registers, one for PCS and one for PMA.

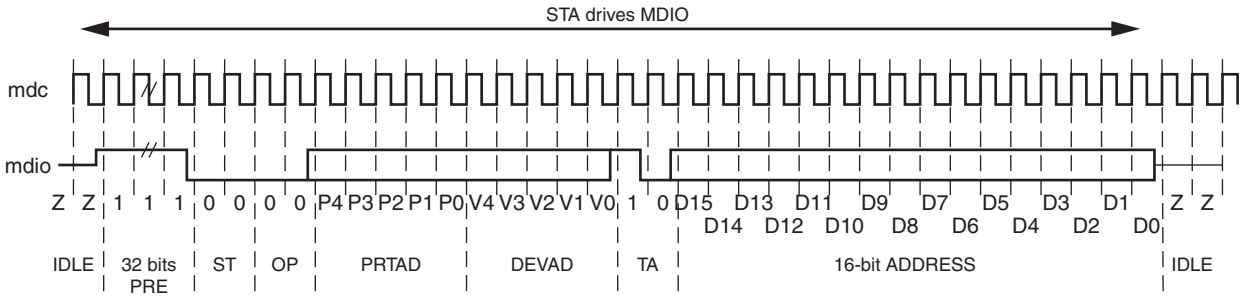

Figure 3-8: **MDIO Set Address Transaction**

Write Transaction

Figure 3-9 shows a Write transaction defined by OP='01.' The 10GBASE-R/KR core takes the 16-bit word in the data field and writes it to the register at the current address.

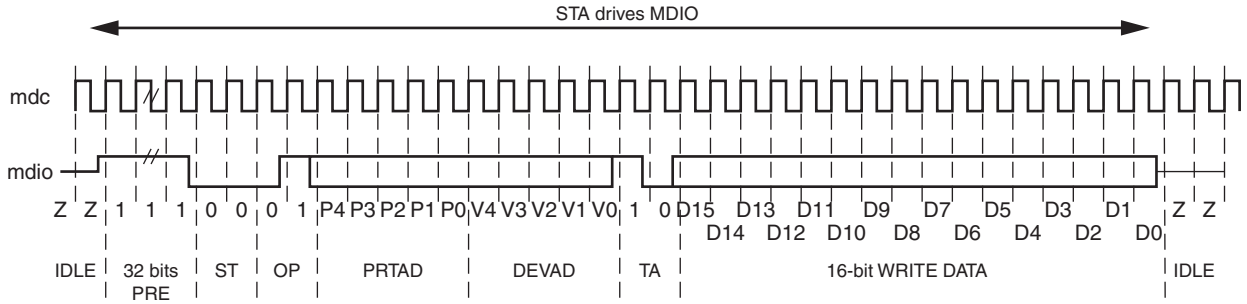

Figure 3-9: **MDIO Write Transaction**

Read Transaction

Figure 3-10 shows a Read transaction defined by OP='11.' The 10GBASE-R/KR core returns the 16-bit word from the register at the current address.

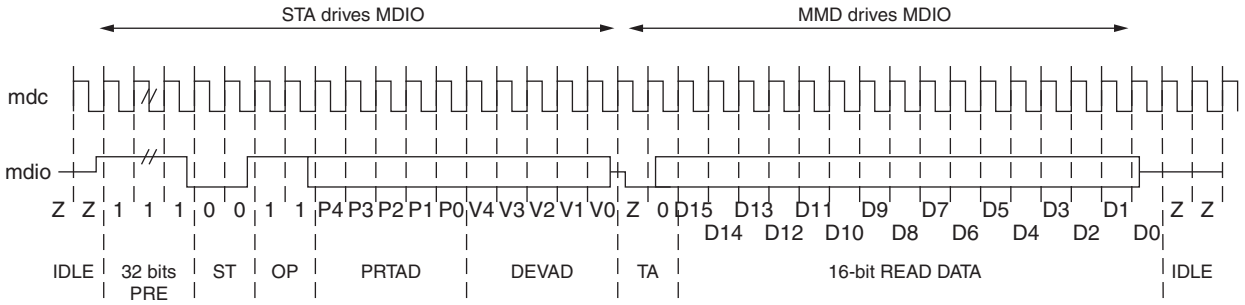

Figure 3-10: **MDIO Read Transaction**

Post-Read-increment-address Transaction

Figure 3-11 shows a Post-read-increment-address transaction, defined by OP='10.' The 10GBASE-R/KR core returns the 16-bit word from the register at the current address for the given DEVAD then increments that current address. This allows sequential reading or writing by a STA master of a block of contiguous register addresses.

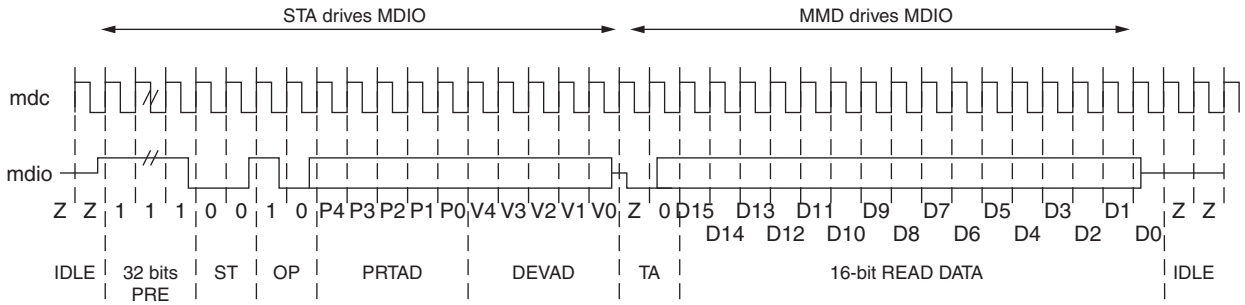

Figure 3-11: **MDIO Read-and-increment Transaction**

Configuration and Status Vectors

If the 10GBASE-R/KR core is generated without an MDIO interface, the key configuration and status information is carried on simple bit vectors, which are:

- configuration_vector[535:0]
- status_vector[447:0]

See Register Space in Chapter 2 which describes each of the registers that is emulated with these configuration and status vectors. Clicking the IEEE Register column entries will link to the relevant register description.

Read the notes at the end of this section relating to clearing latching information bits and counters. Some IEEE registers are defined as set/clear-on-read, and because there is no 'read' when using the configuration and status vectors, special controls have been provided to imitate that behavior.

BASE-R

Table 3-4 shows the breakdown of the 10GBASE-R-specific configuration vector and Table 3-5 shows the breakdown of the status vector. Any bits not mentioned are assumed to be 0s.

Table 3-4: **Configuration Vector - BASE-R**

Bit	IEEE Register	Description
0	1.0.0	PMA Loopback Enable
15	1.0.15	PMA Reset (1)
16	1.9.0	Global PMD TX Disable
110	3.0.14	PCS Loopback Enable
111	3.0.15	PCS Reset(1)
169:112	3.37–3.34	MDIO Register 3.34–37: 10GBASE-R Test Pattern Seed A0–3
233:176	3.41–3.38	MDIO Register 3.38–41: 10GBASE-R Test Pattern Seed B0–3
240	3.42.0	Data Pattern Select
241	3.42.1	Test Pattern Select
242	3.42.2	RX Test Pattern Checking Enable
243	3.42.3	TX Test Pattern Enable
244	3.42.4	PRBS31 TX Test Pattern Enable
245	3.42.5	PRBS31 RX Test Pattern Checking Enable
399:384	3.65535.15:0	125 µs timer control
512	(1.1.2) (2)	Set PMA Link Status
513	(1.8.11) (2) (1.8.10) (2)	Clear PMA/PMD Link Faults
516	(3.1.2) (2)	Set PCS Link Status
517	(3.8.11) (2) (3.8.10) (2)	Clear PCS Link Faults
518	(3.33) (2)	Clear 10GBASE-R Status 2
519	(3.43) (2)	Clear 10GBASE-R Test Patter Error Counter

1. These reset signals should be asserted for a single clock tick only.
2. Reset controls for the given registers

Table 3-5: **Status Vector - BASE-R**

Bit	IEEE Register	Description
15	1.0.15	PMA Reset
18	1.1.2	PMA/PMD RX Link Status (Latching Low)
23	1.1.7	PMA/PMD Fault (1)
42	1.8.10	PMA/PMD RX Fault (Latching High)
43	1.8.11	PMA/PMD TX Fault (Latching High)
48	1.10.0	Global PMD RX Signal Detect
207:192	1.65535	Core Info (Zynq-7000, Virtex-7 and Kintex-7 Devices only)
223	3.0.15	PCS Reset (Zynq-7000, Virtex-7, and Kintex-7 Devices only)
226	3.1.2	PCS RX Link Status (Latching Low)
231	3.1.7	PCS Fault(1)
250	3.8.10	PCS RX Fault (Latching High)
251	3.8.11	PCS TX Fault (Latching High)
256	3.32.0	10GBASE-R PCS RX Locked
257	3.32.1	10GBASE-R PCS high BER
268	3.32.12	10GBASE-R PCS RX Link Status
279:272	3.33.7:0	10GBASE-R PCS Errored Blocks Counter
285:280	3.33.13:8	10GBASE-R PCS BER Counter
286	3.33.14	Latched High RX high BER
287	3.33.15	Latched Low RX Block Lock
303:288	3.43.15:0	10GBASE-R Test Pattern Error Counter

1. This bit is a logical OR of two latching bits and so will exhibit latching behavior without actually being latching itself.

BASE-KR

Table 3-6 shows the additional signals in the configuration vector which are specific to BASE-KR functionality.

Table 3-6: **Configuration Vector - BASE-KR Specific**

Bit	IEEE Register	Description
32	1.150.0	Restart Training
33	1.150.1	Enable Training
53:48	1.152.5:0	MDIO Register 1.152: 10GBASE-KR LP Coefficient Update
60	1.152.12	LP Coefficient Initialize (valid when 1.150.1 = '0')
61	1.152.13	LP Coefficient Preset (valid when 1.150.1 = '0')
64	1.171.0	Enable FEC (1)
65	1.171.1	FEC signal errors to PCS (1) (5)
281	7.0.9	Restart Autonegotiation (2)
284	7.0.12	Enable Autonegotiation (2)
285	7.0.13	Extended Next Page Support (2)
287	7.0.15	Reset Autonegotiation (2)
300:293	7.16.12:5	AN Advertisement Data D12..D5
301	7.16.13	AN Advertisement Data – Remote fault
303	7.16.15	AN Advertisement Data – Next Page
319:304	7.17.15:0	AN Advertisement Data – D31..D16
335:320	7.18.15:0	AN Advertisement Data – D47..D32
346:336	7.22.10:0	AN XNP – Message Unformatted Code Field
348	7.22.12	AN XNP Acknowledge 2
349	7.22.13	AN XNP Message Page
351	7.22.15	AN XNP Next Page
367:352	7.23.15:0	AN XNP Unformatted Code Field 1
383:368	7.24.15:0	AN XNP Unformatted Code Field 2
405:400	1.65520.5:0	MDIO Register: 1.65520: Vendor-Specific LD Training
412	1.65520.12	LD Training Initialize
413	1.65520.13	LD Training Preset
415	1.65520.15	Training Done
514	(1.173:172)(3)	MDIO Register 1.173: 10GBASE-R FEC Corrected Blocks (Upper) MDIO Register 1.172: 10GBASE-R FEC Corrected Blocks (Lower)
515	(1.175:174)(3)	MDIO Register 1.175: 10GBASE-R FEC Uncorrected Blocks (Upper) MDIO Register 1.174: 10GBASE-R FEC Uncorrected Blocks (Lower)
520	(7.1.2)(3)	Set AN Link Up/Down

Table 3-6: **Configuration Vector - BASE-KR Specific *(Cont'd)***

Bit	IEEE Register	Description
521	(7.1.4)(3)	Clear AN Remote Fault
522	(7.1.6)(3)	Clear AN Page Received
523	(7.18:16)(4)	MDIO Register 7.16:17:18: AN Advertisement
524	(7.24:22)(4)	MDIO Register 7.22, 23, 24: AN XNP Transmit

1. Only valid when the optional FEC block is included
2. Only valid when the optional AN block is included
3. Reset controls for the given registers
4. Toggle to load the AN Page data from the associated configuration vector bits
5. If FEC Error Passing is enabled while FEC is enabled, errors will be seen temporarily. To avoid this, only enable Error Passing while FEC is disabled.

Table 3-7 shows the additional signals in the status vector which are specific to BASE-KR functionality.

Table 3-7: **Status Vector - BASE-KR Specific**

Bit	IEEE Register	Description
67:64	1.151.3:0	MDIO Register 1.151: 10GBASE-KR PMD Status
85:80	1.153.5:0	MDIO Register 1.153: 10GBASE-KR LP Status
95	1.153.15:14	LP Status Report Training Complete
101:96	1.152.5:0	MDIO Register 1.152: 10GBASE-KR LP Coefficient Update
108	1.152.12	LP Coefficient Initialize
109	1.152.13	LP Coefficient Preset
117:112	1.155.5:0	MDIO Register 1.155: 10GBASE-KR LD Status
127	1.155.15	LD Status Report Training Complete
159:128	1.173:172	MDIO Register 1.173: 10GBASE-R FEC Corrected Blocks (Upper)(1) MDIO Register 1.172: 10GBASE-R FEC Corrected Blocks (Lower)(1)
191:160	1.175:174	MDIO Register 1.175: 10GBASE-R FEC Uncorrected Blocks (Upper)(1) MDIO Register 1.174: 10GBASE-R FEC Uncorrected Blocks (Lower)(1)
319	7.0.15	AN Reset(2)
320	7.1.0	LP AN Capable(2)
322	7.1.2	AN Link Up/Down (latching Low)(2)
323	7.1.3	AN Ability(2)
324	7.1.4	AN Remote Fault (latching High)(2)
325	7.1.5	AN Complete(2)
326	7.1.6	AN Page Received (latching High)(2)
327	7.1.7	AN Extended Next Page Used(2)
383:336	7.21:19	MDIO Register 7.19, 20, 21: AN LP Base Page Ability
394:384	7.25.10:0	AN LP XNP – Message Unformatted Code Field(2)

Table 3-7: **Status Vector - BASE-KR Specific *(Cont'd)***

Bit	IEEE Register	Description
395	7.25.11	AN LP XNP – Toggle(2)
396	7.25.12	AN LP XNP – Acknowledge2(2)
397	7.25.13	AN LP XNP – Message Page(2)
399	7.25.15	AN LP XNP – Next Page(2)
415:400	7.26.15:0	AN LP XNP – Unformatted Code Field 1(2)
431:416	7.27.15:0	AN LP XNP – Unformatted Code Field 2(2)
432	7.48.0	Backplane AN Ability
435	7.48.3	Backplane Ethernet Status – KR negotiated
436	7.48.4	Backplane Ethernet Status – FEC negotiated

1. Only valid when the optional FEC block is included
2. Only valid when the optional AN block is included

Bit 286 of the Status Vector is latching-high and is cleared Low by bit 518 of the configuration_vector port. Figure 3-12 shows how the status bit is cleared.

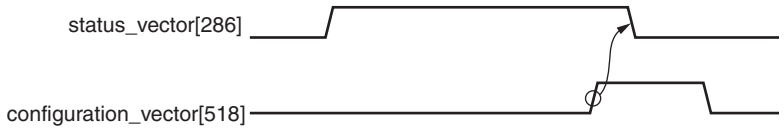

Figure 3-12: **Clearing the Latching-High Bits**

Bits 18, 226, and 287 of the status_vector port are latching-low and set High by bits 512, 516, and 518 of the configuration vector. Figure 3-13 shows how the status bits are set.

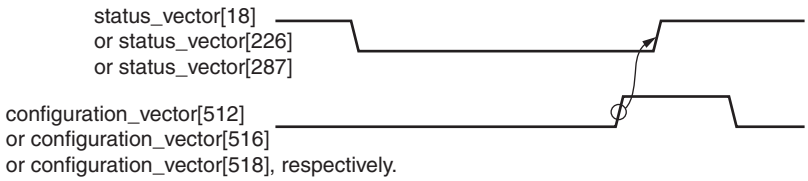

Figure 3-13: **Setting the Latching-Low Bits**

Similarly for Zynq®-7000, Virtex®-7, and Kintex®-7 device designs, Latching High Status Vector bits 42 and 43 can be reset with Configuration Vector bit 513, and bits 250 and 251 can be reset with Configuration Vector bit 517.

- Status bits 285:272 are also reset using configuration vector bit 518
- Status bits 303:288 are reset using configuration vector bit 519

For Base KR cores, similar reset behaviors exist for the following status vector bits:

- status vector bits 159:128 – cleared with configuration vector bit 514
- status vector bits 191:160 – cleared with configuration vector bit 515
- status vector bit 322 – set with configuration vector bit 520
- status vector bit 324 – cleared with configuration vector bit 521
- status vector bit 326 – cleared with configuration vector bit 522

Finally, configuration vector bits 335:293 and 383:336 each implement three 16-bit registers which are normally latched into the core when the lower register is written, keeping the data coherent. Because there is no need for this behavior when the entire vector is exposed, these bits are latched into the core whenever configuration register bits 523 and 524 respectively are toggled High.

Clocking

The clocking schemes in this section are illustrative only and can require customization for a specific application. These apply to Zynq-7000, Virtex-7, and Kintex-7 devices.

Reference Clock

For Zynq-7000, Virtex-7, and Kintex-7 device transceivers, the reference clock must be running at 156.25 MHz.

Transceiver Placement

A single IBUFDS_GTE2 block is used to feed the reference clocks for up to 12 GTXE2_CHANNEL and GTHE2_CHANNEL transceivers, through a GTXE2_COMMON or GTHE2_COMMON block. The COMMON blocks can be shared by up to 4 CHANNEL blocks, depending on the relative placement of the CHANNEL blocks.

For details about Zynq-7000, Virtex-7, and Kintex-7 device transceiver clock distribution, see the section on Clocking in the *7 Series Transceivers User Guide* (UG476) [Ref 3].

Internal Client-Side Interface

The clocking scheme for the internal client interface is shown in Figure 3-14.

The GTXE2_CHANNEL and GTHE2_CHANNEL primitives require a 156.25 MHz reference clock, as well as 322.26 MHz TX and RX user clocks. The latter must be created from the 322.26 MHz `TXOUTCLK` and `RXOUTCLK` outputs from that block.

The 156.25 MHz user-logic clock `clk156` must be created from the transceiver reference clock to keep the user logic and transceiver interface synchronous.

A dedicated management/configuration clock, `dclk`, is used by the user logic and the transceiver and *must* be the same clock as clk156.

Figure 3-14 shows a possible clocking architecture with a single GTXE2_CHANNEL or GTHE2_CHANNEL block. This is the hierarchy if shared logic is included in the example design (see Shared Logic).

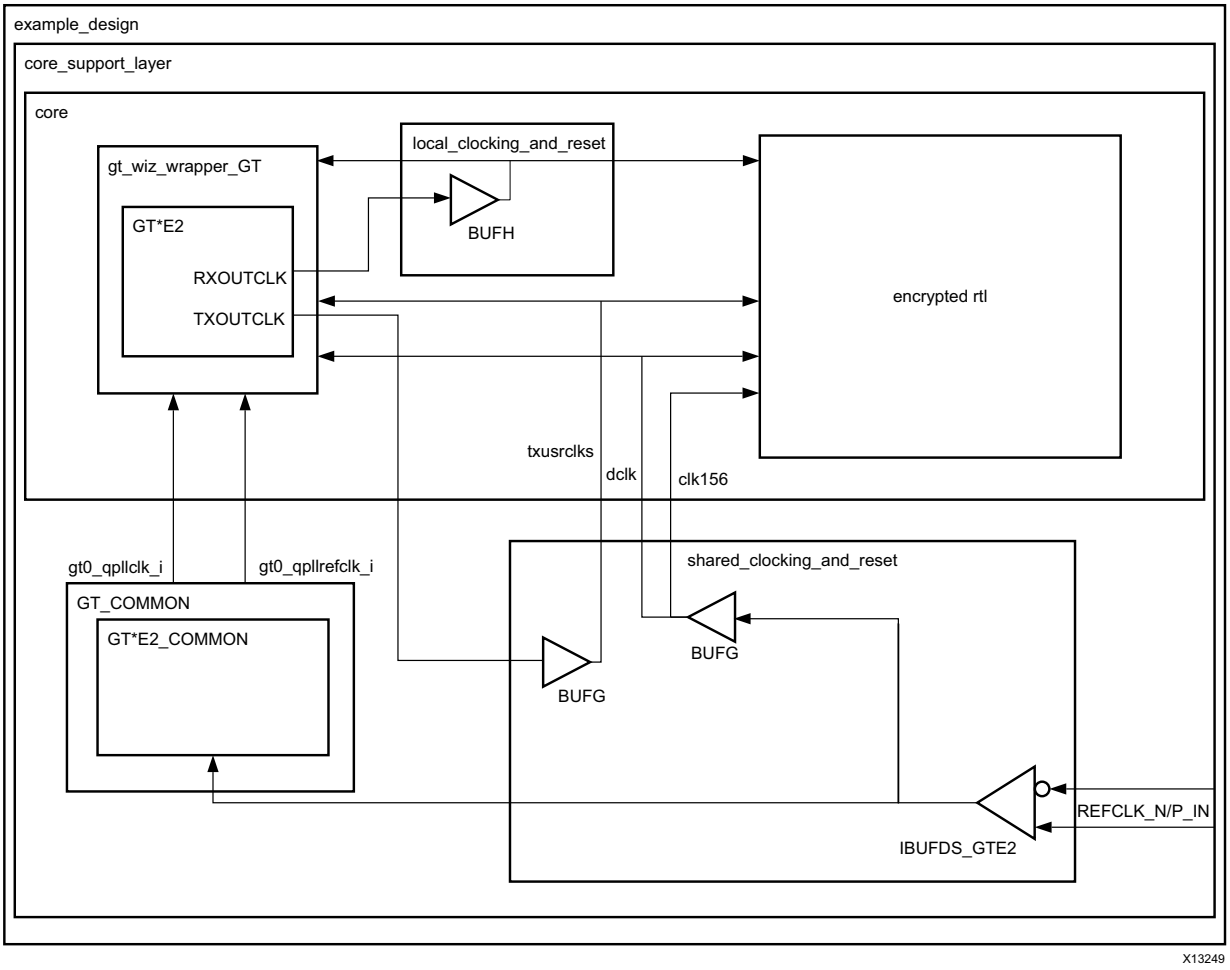

Figure 3-14: **Clocking Scheme for Internal Client-Side Interface: 7 Series FPGAs**

Resets

All register resets within the 10GBASE-R/KR core netlist are synchronized to the relevant clock port.

Receiver Termination

The receiver termination for Zynq-7000, Virtex-7, and Kintex-7 devices must be set correctly. See the *7 Series Transceivers User Guide* (UG476) [Ref 3].

Customizing and Generating the Core

This chapter includes information on using Xilinx tools to customize and generate the core using the Vivado® design tools.

Vivado Integrated Design Environment

You can customize the IP for use in your design by specifying values for the various parameters associated with the IP core using the following steps:

1. Select the IP from the IP catalog.
2. Double-click the selected IP or select the Customize IP command from the toolbar or popup menu.

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 5] and the *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 6].

Note: Figures in this chapter are illustrations of the Vivado Integrated Design Environment (IDE). This layout might vary from the current version.

If you are customizing and generating the core in the IP Integrator, see the *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [Ref 7] for detailed information. Vivado IDE might auto-compute certain configuration values when validating or generating the design, as noted in this section. You can view the parameter value after successful completion of the **validate_bd_design** command.

Figure 4-1 displays the main screen for customizing the 10GBASE-R/KR core. When targeting devices containing only GTXE2 transceivers, the 10GBASE-KR options do not appear.

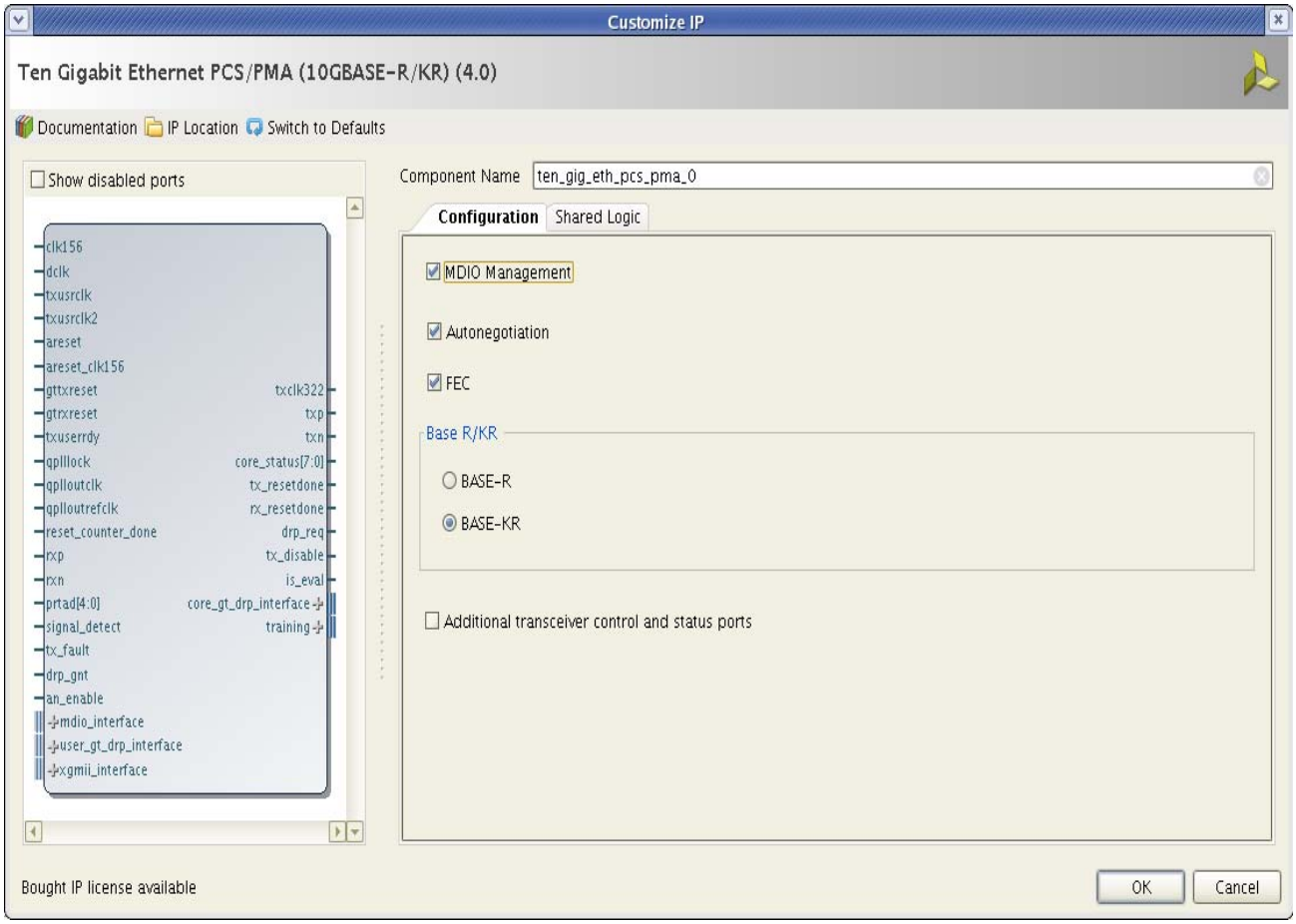

Figure 4-1: **Vivado IP Catalog 10GBASE-R/KR Main Screen**

Component Name

The component name is used as the base name of the output files generated for the core. Names must begin with a letter and must be composed from the following characters: a through z, 0 through 9 and "_" (underscore).

MDIO Management

Select this option to implement the MDIO interface for managing the core. Deselect the option to remove the MDIO interface and expose a simple bit vector to manage the core.

The default is to implement the MDIO interface.

BASE-R or BASE-KR

Select the Base-KR option to get a Base-KR core and have access to the following two options. This is only available when targeting devices containing GTHE2 transceivers.

AN Support

Select this option to include the Autonegotiation (AN) block in the Base-KR core.

FEC Support

Select this option to include the FEC block in the Base-KR core.

Shared Logic

Select **Include Shared Logic in Core** if you want the core itself to contain the shared logic (the signals generated by the shared logic will be available on the core interface).

Otherwise the Shared Logic will be exposed in the Example Design. See Shared Logic in Chapter 3 and Appendix D, Special Design Considerations for more information.

Transceiver Debug

Select **Additional Transceiver Control and Status Ports** to expose some extra, useful transceiver ports on the core interface. These are detailed in Transceiver Debug Signals — Zynq-7000, Virtex-7, and Kintex-7 Devices in Chapter 2, Table 2-13.

Output Generation

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 5].

The core has various selectable output products. These can be generated by right-clicking on the customized piece of IP in the **Sources** window.

- **Examples** – Source HDL and constraints for the example project
- **Simulation** – Simulation source files
- **Synthesis** – Synthesis source files
- **Instantiation Template** – Example instantiation template for the core level module.

Constraining the Core

This chapter contains information about constraining the core using the Vivado® Design Suite.

Required Constraints

This section defines the constraint requirements for the core. Constraints are provided in several XDC files which are delivered with the core and the example design to give a starting point for constraints for the user design. The required constraints are shown in the following sections.

There are four XDC constraint files associated with this core:

- <corename>_example_design.xdc
- <corename>_clocks.xdc
- <corename>_ooc.xdc
- <corename>.xdc

The first is used by the example design; The second defines clock(s) that can be used by other logic and is marked for automatic early processing within the Vivado design tools to facilitate that. The third file is used for Out Of Context support where this core can be synthesized without any wrappers. The fourth is the main XDC file for this core.

Device, Package, and Speed Grade Selections

You are only able to generate the core for supported device, package and speed grade combinations; –1 speed grades are not supported for this core.

Clock Frequencies

The core requires a 156.25 MHz reference clock which can be shared among several cores; so it is defined in the example design XDC file:

```
create_clock -name Q1_CLK0_GTREFCLK -period 6.400 [get_ports refclk_p]
```

The reference clock is used as the main clock for the core, `clk156`, and for the DRP interface to the transceiver, `dclk`.

These clocks are automatically defined and they can be shared between multiple cores so only the identifiers need be extracted in the XDC files (dclk is in fact the exact same clock as clk156):

```
set clk156name [get_clocks -of_objects [get_ports clk156]]
set dclkname [get_clocks -of_objects [get_ports dclk]]
```

The transceiver creates 322.26 MHz clocks that must be constrained in the XDC files. These constraints are required for devices with GTXE2 transceivers.

```
create_clock -period 3.103 [get_pins -of_objects [get_cells * -hierarchical -filter
{REF_NAME=~ GTXE2_CHANNEL}] -filter {NAME =~ *TXOUTCLK}]
set TXOUTCLK_OUT [get_clocks -of  [get_pins -of_objects [get_cells * -hierarchical
-filter {REF_NAME=~ GTXE2_CHANNEL}] -filter {NAME =~ *TXOUTCLK}]]
```

The Receive Recovered clock is defined in the <corename>.xdc file:

```
create_clock -period 3.103 [get_pins -of_objects [get_cells * -hierarchical -filter
{REF_NAME=~ GTXE2_CHANNEL}] -filter {NAME =~ *RXOUTCLK}]
set RXOUTCLK_OUT [get_clocks -of  [get_pins -of_objects [get_cells * -hierarchical
-filter {REF_NAME=~ GTXE2_CHANNEL}] -filter {NAME =~ *RXOUTCLK}]]
```

Similar constraints are required for devices with GTHE2 transceivers.

The example design contains a DDR register that can be used to forward the XGMII_RX clock off-chip.

```
create_generated_clock -name ddrclock -divide_by 1 -invert -source [get_pins
*rx_clk_ddr/C] [get_ports xgmii_rx_clk]
set_output_delay -max 1.500 -clock [get_clocks ddrclock] [get_ports * -filter {NAME
=~ *xgmii_rxd*}]
set_output_delay -min -1.500 -clock [get_clocks ddrclock] [get_ports * -filter {NAME
=~ *xgmii_rxd*}]
set_output_delay -max 1.500 -clock [get_clocks ddrclock] [get_ports * -filter {NAME
=~ *xgmii_rxc*}]
set_output_delay -min -1.500 -clock [get_clocks ddrclock] [get_ports * -filter {NAME
=~ *xgmii_rxc*}]
```

Clock Management

Any clock management tiles (MMCMs) are exposed in the source code in the example design files for the core, enabling sharing these resources with other cores and designs.

Clock Placement

Location constraints for MMCMs might be required depending on the rest of the system.

Banking

All ports should be given Location constraints appropriate to your design within Banking limits

Transceiver Placement

Transceivers should be given location constraints appropriate to your design. An example of these LOC constraints can be found in the example design XDC file.

I/O Standard and Placement

All ports should be given I/O Standard and Location constraints appropriate to your design.

These constraints are required if the optional MDIO interface is included and if the MDIO interface is on the chip boundary.

In the XDC file:

```
set_property IOB TRUE [get_cells * -hierarchical -filter {NAME =~ *mdc_reg1*}]
set_property IOB TRUE [get_cells * -hierarchical -filter {NAME =~ *mdio_in_reg1*}]
set_property IOB TRUE [get_cells * -filter {NAME =~ *mdio_out*reg*}]
set_property IOB TRUE [get_cells * -filter {NAME =~ *mdio_tri*reg*}]
```

Simulation

For comprehensive information about Vivado® simulation components, as well as information about using supported third-party tools, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 4].

Simulation of 10GBASE-R/KR at the core level is not supported without the addition of an appropriate test bench (not supplied). Simulation of the example design is supported.

Synthesis and Implementation

This chapter contains information about synthesis and implementation in the Vivado® Design Suite.

For details about synthesis and implementation, see "Synthesizing IP" and "Implementing IP" in the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 5].

All synthesis sources that are required by the core are included. For this core there is a mix of both encrypted and unencrypted sources. Only the unencrypted sources are visible and optionally editable by using the 'Managed IP' property option.

Detailed Example Design

This chapter contains information about the provided example design in the Vivado® Design Suite.

Example Design and Core Support Layer

This section shows an example HDL wrapper for Zynq®-7000, Virtex®-7, and Kintex®-7 devices.

In Figure 8-1, the example HDL wrapper generated contains the following:

- Core Support Layer
- Pipeline registers on the XGMII interfaces, to aid timing closure when implemented stand alone.
- Double Data Rate (DDR) register on `xgmii_rx_clk`
- 10GBASEKR with no MDIO interface only: Simple Logic to preserve `configuration_vector` and `status_vector` through synthesis to avoid optimizing away logic (because there are not enough I/Os on some devices, to route these signals to and preserve the logic in that way)

The Core Support Layer contains the following:

- The Block-level core instance containing the encrypted RTL for the core itself, a local clocking and reset block, and a transceiver wizard wrapper for the GT CHANNEL block. This is a per-core block and the logic cannot be shared between multiple cores.
- A wrapper for the GT COMMON block, which can be shared between up to four cores.
- A shared clocking and reset block, which as the name suggests, can be shared between multiple cores.

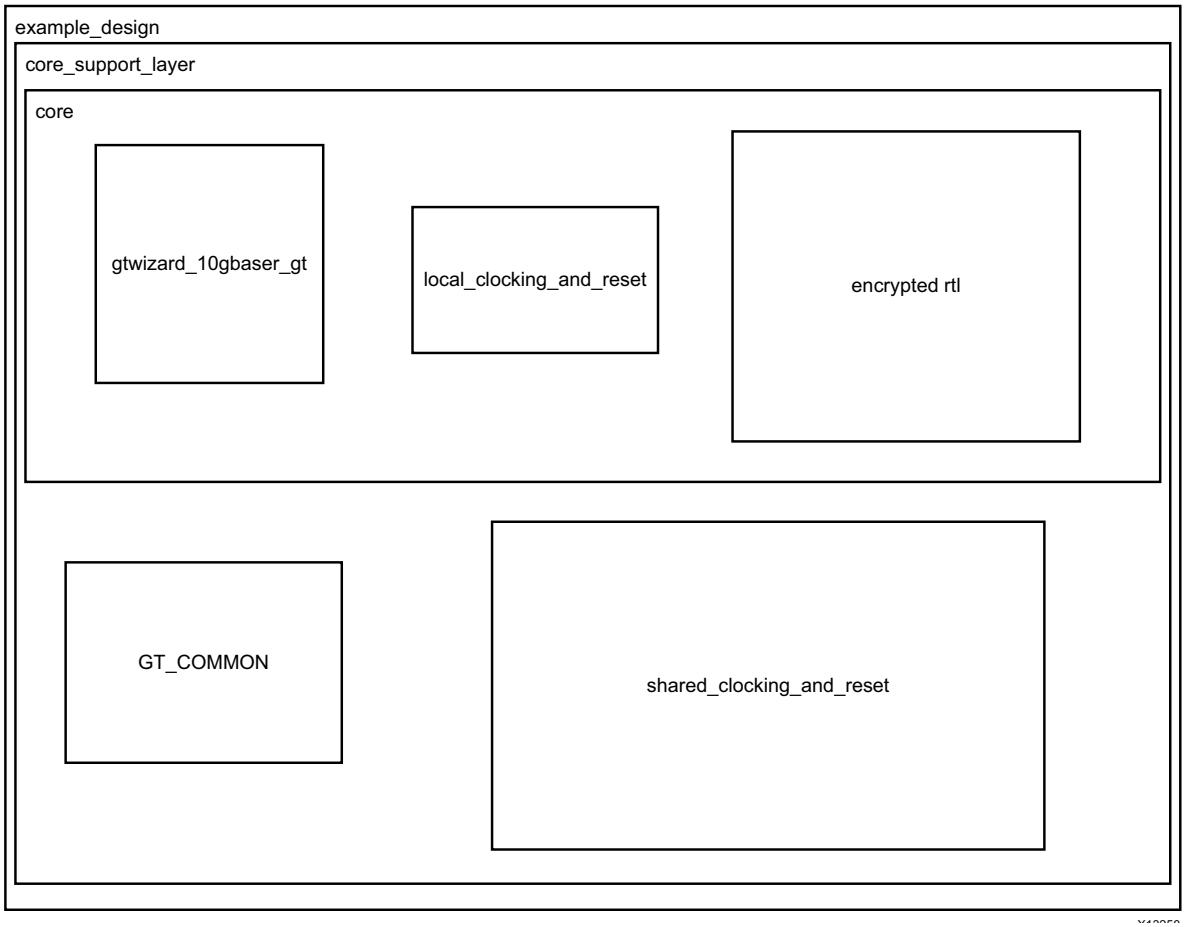

Figure 8-1: **Example HDL Wrapper for 10GBASE-R/KR**

Shared Logic and the Core Support Layer

Depending on the selection made for shared logic in the core customization Vivado IDE, the Core Support Layer can either itself be the 'core' top-level (Include Shared Logic in core) or can simply contain the 'core' top-level (Include Shared Logic in example design).

The difference is subtle but selecting **Include Shared Logic in the core** produces a core that includes all the shared logic and has outputs for clocks and control signals that can be shared between multiple 10GBaseR/KR IP cores.

Selecting **Include Shared Logic in the Example Design** allows you to access the shared logic.

Typically in a multi-core design, you can create one core, core 'A' with Shared Logic included in the core, and one core, core 'B' with the opposite setting. A single instance of core 'A' will then provide the clocks for several instances of core 'B'. See Appendix D, Special Design Considerations for more information.

Test Bench

This chapter contains information about the provided test bench in the Vivado® Design Suite.

In Figure 9-1, the demonstration test bench is a simple VHDL or Verilog program to exercise the example design and the core itself. This test bench consists of transactor procedures or tasks that connect to the major ports of the example design and a control program that pushes frames of varying length and content through the design and checks the values as they exit the core. The test bench is supplied as part of the Example Simulation output product group.

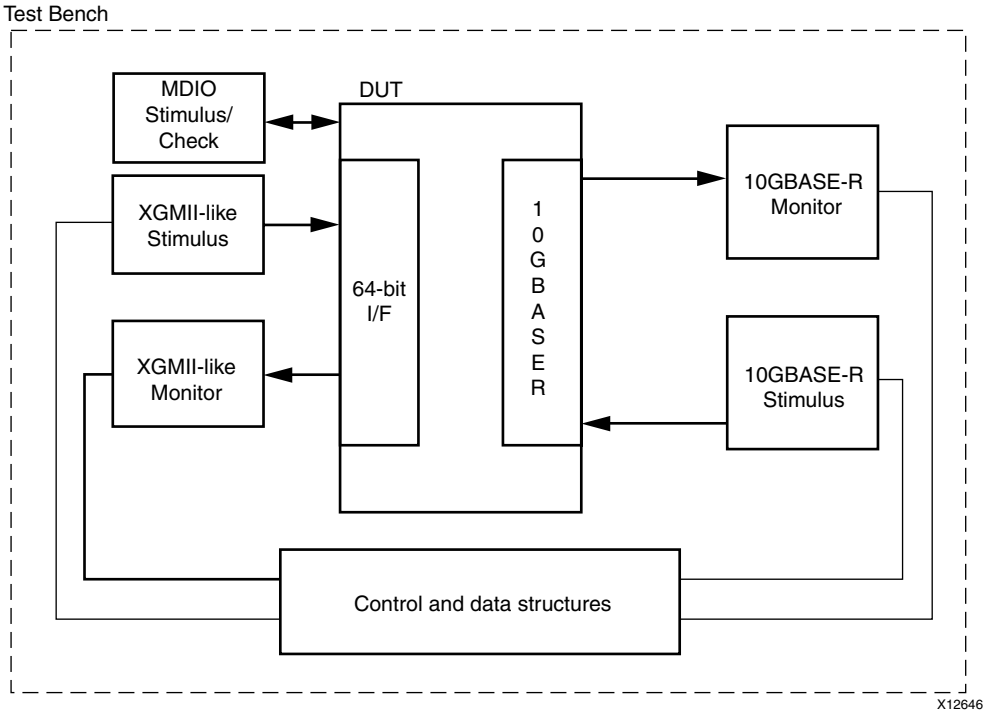

Figure 9-1: **Demonstration Test Bench**

Verification, UNH Testing, and Interoperability

The 10GBASE-R/KR LogiCORE™ IP core has been verified in Vivado® design tools with the production Zynq®-7000, Virtex®-7, and Kintex®-7 device speed files.

Simulation

A highly parameterizable transaction-based simulation test suite has been used to verify the core. Tests included:

- Register access over MDIO or Configuration/Status vectors
- Loss and re-gain of synchronization
- Loss and re-gain of alignment
- Frame transmission
- Frame reception
- Clock compensation
- Recovery from error conditions
- Autonegotiation phase
- Training phase
- FEC with correctable and uncorrectable errors

Hardware Testing

The 10GBASE-R and 10GBASE-KR cores have been validated on Kintex-7 and Virtex-7 devices. Hundreds of millions of Ethernet frames have been successfully transmitted and received on each board and other features such as hot-plugging, autonegotiation and FEC error correction have been tested with the setup.

Testing

The 10GBASE-R and 10GBASE-KR core has successfully undergone validation at the University of New Hampshire Interoperability Lab. Detailed test reports are available from Xilinx. All tests were successful.

Migrating and Upgrading

This appendix contains information about migrating a design from ISE® to the Vivado® Design Suite, and for upgrading to a more recent version of the IP core. For customers upgrading in the Vivado Design Suite, important details (where applicable) about any port changes and other impact to user logic are included.

Migrating

For information on migrating to the Vivado Design Suite, see the *ISE to Vivado Design Suite Migration Guide* (UG911) [Ref 9].

Upgrading in the Vivado Design Suite

This section provides information about any changes to the user logic or port designations that take place when you upgrade to a more current version of this IP core in the Vivado Design Suite.

In the 4.0 version of the core, there have been several changes that make the core pin-incompatible with the previous version(s). These changes were required as part of the general one-off hierarchical changes to enhance the customer experience and are not likely to occur again.

Shared Logic

As part of the hierarchical changes to the core, it is now possible to have the core itself include all of the logic that can be shared between multiple cores, which was previously exposed in the example design for the core.

If you are updating a previous version to the 4.0 version with shared logic, there is no simple upgrade path; it is recommended to consult the Shared Logic sections of this document for more guidance.

Port Changes from v3.0 to v4.0

Ports Added

The following ports were added to the core (non-shared logic).

Table B-1: **Ports Added (non-shared logic)**

In/Out	Port name and width	Description	What to do
Output	drp_req	Previously 'hidden' in the block level code, this signal can be used on an external arbiter to allow the core to request access to the transceiver DRP ports.	If you do not require user-access to the transceiver DRP ports, simply connect this to drp_gnt (see the next row).
Input	drp_gnt	Signal from an external arbiter as described in the previous row	See drp_req in the previous row.
Outputs	drp_den_o, drp_dwe_o, drp_daddr_o[15:0], drp_di_o[15:0]	These signals come from the core and must be connected to the transceiver, perhaps through an external arbiter.	If there is no external arbiter, connect these signals directly to the associated core inputs, for example: drp_den_i
Inputs	drp_den_i, drp_dwe_i, drp_daddr_i[15:0], drp_di_i[15:0]	These signals go to the transceiver, either from an external arbiter or from the signals described in the previous rows.	If there is no external arbiter, connect these signals directly to the associated core outputs, for example: drp_den_o.
Outputs	drp_drdy_o, drp_drpdo_o[15:0]	These signals come from the transceiver and should be connected either to an external arbiter or to the signals described in the following row	If there is no external arbiter, connect these signals directly to the associated core inputs, for example: drp_drdy_i.
Inputs	drp_drdy_i, drp_drpdo_i[15:0]	These signals go to the transceiver, either from an external arbiter or from the signals described in the previous rows.	If there is no external arbiter, connect these signals directly to the associated core outputs, for example: drp_drdy_o.

The following ports were added to the core, but only if the Transceiver Debug feature was requested during core customization. Consult the relevant Transceiver User Guide for more information on using these control/status ports.

Table B-2: **Ports Added for Transceiver Debug Feature**

In/Out	Port Name and Width	Description	What to do
Input	gt0_eyescanreset	Reset the EYE Scan logic	If you want to be more compatible with the previous version of the core, do not request the Transceiver Debug feature. Otherwise, drive these signals according to the relevant transceiver user guide.
Input	gt0_eyescantrigger	Trigger the EYE Scan logic	
Input	gt0_rxcdrhold	Freeze the CDR loop	
Input	gt0_txprbsforceerr	Force a single bit error during PRBS31 transmission	
Input	gt0_txpolarity	Switch the sense of the txn/p pins	
Input	gt0_rxpolarity	Switch the sense of the rxn/p pins	
Input	gt0_rxrate[2:0]	Change the QPLL Divider value	
Output	gt0_eyescandataerror	Signals an error during Eye Scan	
Output	gt0_txbufstatus[1:0]	Signals when the Transceiver TX Buffer over/underflows	
Input	gt0_txprecursor[4:0]	(BaseR only!) Can be used to tune the transceiver TX waveform	
Input	gt0_txpostcursor[4:0]	(BaseR only!) Can be used to tune the transceiver TX waveform	
Input	gt0_txdiffctrl[3:0]	(BaseR only!) Can be used to tune the transceiver TX waveform	

Ports Removed

The following ports were removed from the core (non-shared Logic).

Table B-3: **Ports Removed (non-Shared Logic)**

In/Out	Port name and width	Description	What to do
Input	gttxreset_txusrclk2	This signal was used at one time but is no longer required.	Remove this port and its driver from the port map on your instance(s) of this core.
Input	areset_refclk_bufh	This signal was previously used to reset some logic that is now contained inside the shared logic block.	Remove this port and its driver from the port map on your instance(s) of this core.
Input	mmcm_locked_clk156	This signal was used at one point but is no longer required. There is now no MMCM required in the core.	Remove this port and its driver from the port map on your instance(s) of this core.

Debugging

This appendix includes details about resources available on the Xilinx Support website and debugging tools.

TIP: *If the IP generation halts with an error, there might be a license issue. See License Checkers in Chapter 1 for more details.*

Finding Help on Xilinx.com

To help in the design and debug process when using the Ten Gigabit Ethernet PCS/PMA core, the Xilinx Support web page (www.xilinx.com/support) contains key resources such as product documentation, release notes, answer records, information about known issues, and links for obtaining further product support.

Documentation

This product guide is the main document associated with the Ten Gigabit Ethernet PCS/PMA core. This guide, along with documentation related to all products that aid in the design process, can be found on the Xilinx Support web page (www.xilinx.com/support) or by using the Xilinx Documentation Navigator.

Download the Xilinx Documentation Navigator from the Design Tools tab on the Downloads page (www.xilinx.com/download.) For more information about this tool and the features available, open the online help after installation.

Solution Centers

See the Xilinx Solution Centers for support on devices, software tools, and intellectual property at all stages of the design cycle. Topics include design assistance, advisories, and troubleshooting tips. The Solution Center specific to the Ten Gigabit Ethernet PCS/PMA core is the Xilinx Ethernet IP Solution Center.

Answer Records

Answer Records include information about commonly encountered problems, helpful information on how to resolve these problems, and any known issues with a Xilinx product. Answer Records are created and maintained daily ensuring that users have access to the most accurate information available.

Answer Records can also be located by using the Search Support box on the main Xilinx support web page. To maximize your search results, use proper keywords such as

- Product name
- Tool message(s)
- Summary of the issue encountered

A filter search is available after results are returned to further target the results.

Master Answer Record for the Ten Gigabit Ethernet PCS/PMA Core

AR: 54669

Contacting Technical Support

Xilinx provides technical support at www.xilinx.com/support for this LogiCORE™ IP product when used as described in the product documentation. Xilinx cannot guarantee timing, functionality, or support of product if implemented in devices that are not defined in the documentation, if customized beyond that allowed in the product documentation, or if changes are made to any section of the design labeled DO NOT MODIFY.

Xilinx provides premier technical support for customers encountering issues that require additional assistance.

To contact Xilinx Technical Support:

1. Navigate to www.xilinx.com/support.
2. Open a WebCase by selecting the WebCase link located under Additional Resources.

When opening a WebCase, include:

- Target FPGA including package and speed grade.
- All applicable Xilinx Design Tools and simulator software versions.
- Additional files based on the specific issue might also be required. See the relevant sections in this debug guide for guidelines about which file(s) to include with the WebCase.

Note: Access to WebCase is not available in all cases. Login to the WebCase tool to see your specific support options.

Debug Tools

There are many tools available to address Ten Gigabit Ethernet PCS/PMA core design issues. It is important to know which tools are useful for debugging various situations.

Vivado Lab Tools

Vivado® lab tools insert logic analyzer and virtual I/O cores directly into your design. Vivado lab tools allow you to set trigger conditions to capture application and integrated block port signals in hardware. Captured signals can then be analyzed. This feature represents the functionality in the Vivado IDE that is used for logic debugging and validation of a design running in Xilinx devices in hardware.

The Vivado logic analyzer is used to interact with the logic debug LogiCORE™ IP cores, including:

- ILA 2.0 (and later versions)
- VIO 2.0 (and later versions)

See *Vivado Design Suite User Guide: Programming and Debugging* (UG908) [Ref 8].

Reference Boards

Contact your Xilinx representative for information on development platforms for this IP.

Simulation Debug

The simulation debug flow for Mentor Graphics Questa® SIM is illustrated in Figure C-1. A similar approach can be used with other simulators.

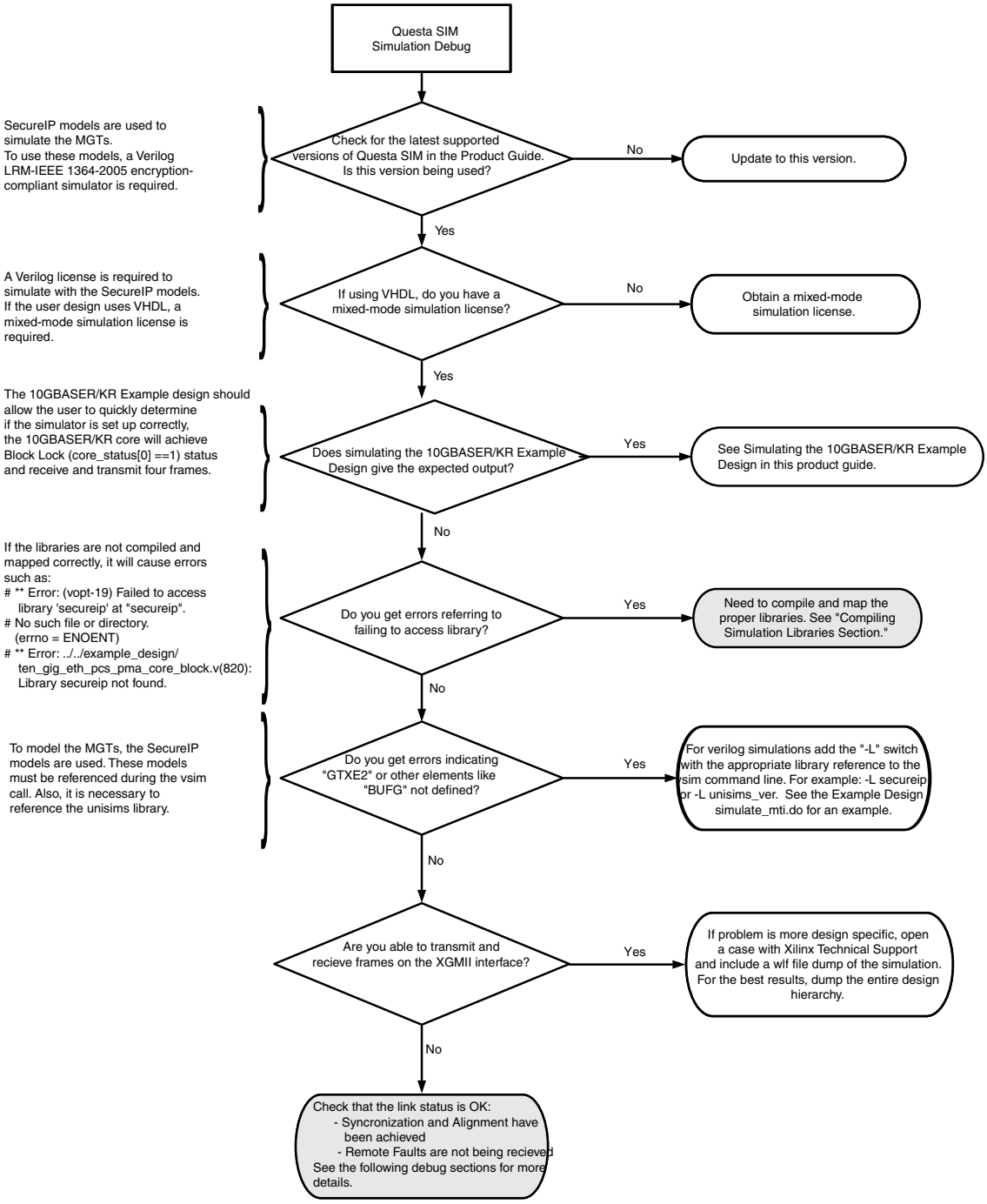

Figure C-1: **Simulation Debug Flow**

Hardware Debug

Hardware issues can range from link bring-up to problems seen after hours of testing. This section provides debug steps for common issues. The Vivado lab tools are a valuable resource to use in hardware debug. The signal names mentioned in the following individual sections can be probed using the Vivado lab tools for debugging the specific problems.

Many of these common issues can also be applied to debugging design simulations. Details are provided on:

- General Checks
- What Can Cause a Local or Remote Fault?
- Link Bring Up – Basic
- Link Bring Up – Base-KR
- What Can Cause Block Lock to Fail?
- What Can Cause the 10Gb PCS/PMA Core to Insert Errors?
- Transceiver Specific Checks
- Problems with the MDIO
- Link Training

General Checks

Ensure that all the timing constraints for the core were properly incorporated and that all constraints were met during implementation.

- Does it work in post-place and route timing simulation? If problems are seen in hardware but not in timing simulation, this could indicate a PCB issue. Ensure that all clock sources are active and clean.
- If using MMCMs in the design, ensure that all MMCMs have obtained lock by monitoring the `LOCKED` port.
- If your outputs go to 0 after operating normally for several hours, check your licensing.

What Can Cause a Local or Remote Fault?

Local Fault and Remote Fault codes both start with the sequence TXD/RXD=0x9C, TXC/RXC=1 in XGMII lane 0. Fault conditions can also be detected by looking at the status vector or MDIO registers. The Local Fault and Link Status are defined as both immediate and latching error indicators by the IEEE specification.

IMPORTANT: *This means that the latching Local Fault and Link Status bits in the status vector or MDIO registers must be cleared with the associated Reset bits in the Configuration vector or by reading the MDIO registers, or by issuing a PMA or PCS reset.*

Local Fault

The receiver outputs a local fault when the receiver is not up and operational. This RX local fault is also indicated in the status and MDIO registers. The most likely causes for an RX local fault are:

- The transceiver has not locked or the receiver is being reset.
- The block lock state machine has not completed.
- The BER monitor state machine indicates a high BER.
- The elastic buffer has over/underflowed.

Remote Fault

Remote faults are only generated in the MAC reconciliation layer in response to a Local Fault message. When the receiver receives a remote fault, this means that the link partner is in a local fault condition.

When the MAC reconciliation layer receives a remote fault, it silently drops any data being transmitted and instead transmits IDLEs to help the link partner resolve its local fault condition. When the MAC reconciliation layer receives a local fault, it silently drops any data being transmitted and instead transmits a remote fault to inform the link partner that it is in a fault condition. Be aware that the Xilinx 10GEMAC core has an option to disable remote fault transmission.

Link Bring Up – Basic

High Level Link Up (10GBASE-R or 10GBASE-KR with Auto-Negotiation + Training Disabled)

The following link initialization stages describe a possible scenario of the Link coming up between device A and device B.

Stage 1: Device A Powered Up, but Device B Powered Down

1. Device A is powered up and reset.
2. Device B powered down.
3. Device A detects a fault because there is no signal received. The Device A 10Gb PCS/PMA core indicates an RX local fault.
4. The Device A MAC reconciliation layer receives the local fault. This triggers the MAC reconciliation layer to silently drop any data being transmitted and instead transmit a remote fault.
5. RX Link Status = 0 (link down) in Device A.

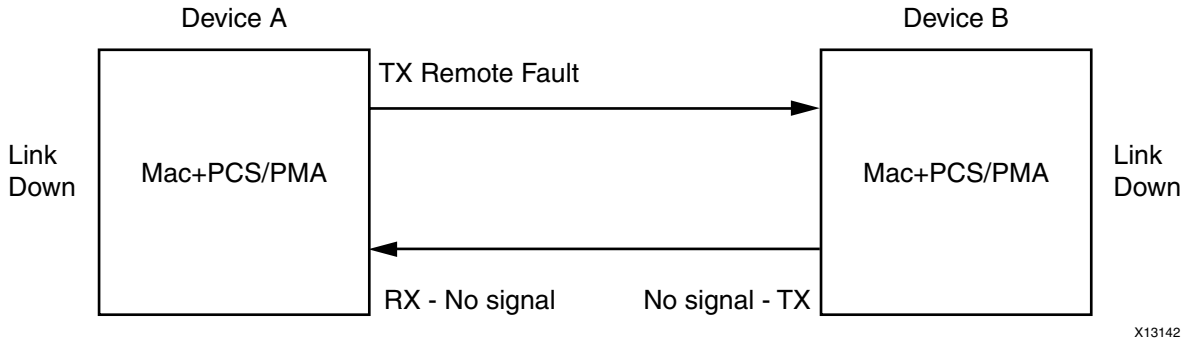

Figure C-2: **Device A Powered Up, but Device B Powered Down**

Stage 2: Device B Powers Up and Resets

1. Device B powers up and resets.
2. Device B 10Gb PCS/PMA completes block lock and high BER state machines.
3. Device A does not have block lock. It continues to send remote faults.
4. Device B 10Gb PCS/PMA passes received remote fault to MAC.
5. Device B MAC reconciliation layer receives the remote fault. It silently drops any data being transmitted and instead transmits IDLEs.
6. Link Status = 0 (link down) in both A and B.

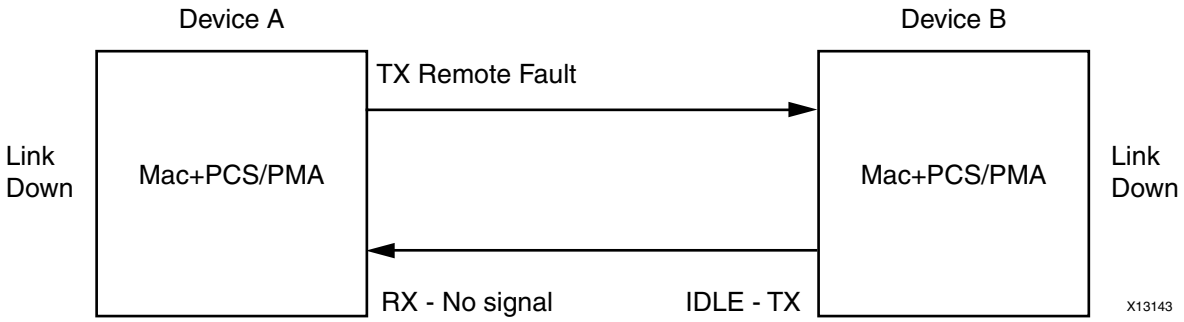

Figure C-3: **Device B Powers Up and Resets**

Stage 3: Device A Receives Idle Sequence

1. Device A PCS/PMA RX detects idles, synchronizes and aligns.
2. Device A reconciliation layer stops dropping frames at the output of the MAC transmitter and stops sending remote faults to Device B.
3. Device A Link Status=1 (Link Up)
4. When Device B stops receiving the remote faults, normal operation starts.

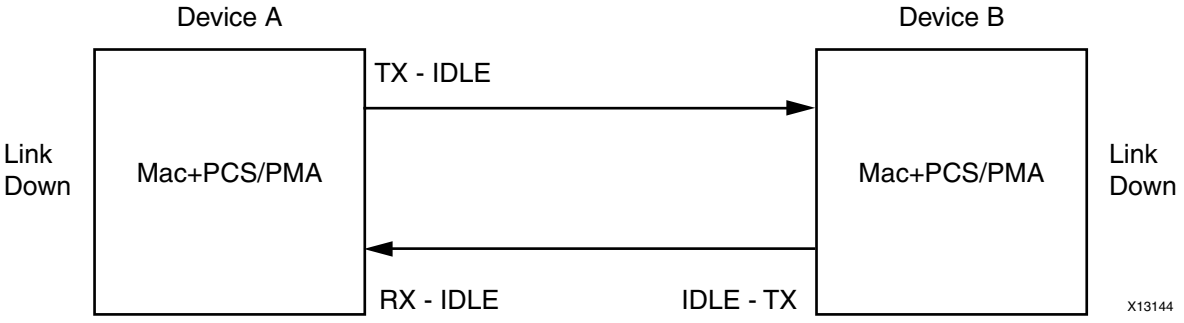

Figure C-4: **Device A Receives Idle Sequence**

Stage 4: Normal Operation

In Stage 4 shown in Figure C-5, Device A and Device B have both powered up and been reset. The link status is 1 (link up) in both A and B and in both the MAC can transmit frames successfully.

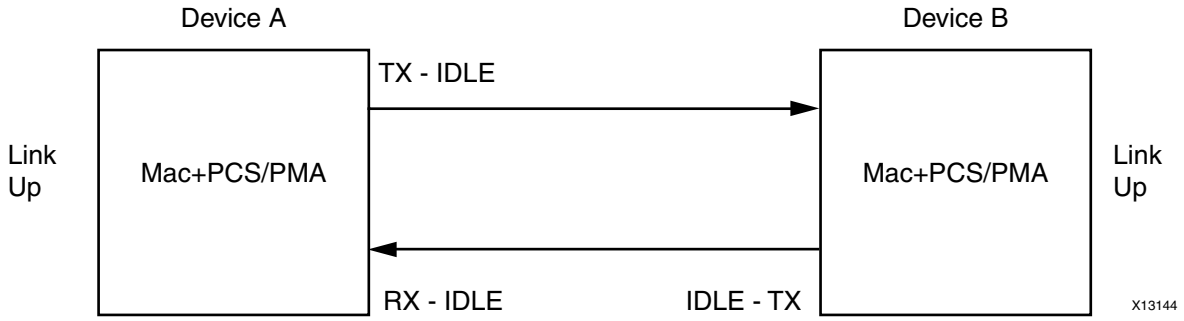

Figure C-5: **Normal Operation**

Link Bring Up – Base-KR

For a 10GBASE-KR core with optional Auto-negotiation, the bring-up of the link is more complex than for a 10GBASE-R core. First of all, both ends of the link disable transmission to bring down any existing link.

Then some low-data rate Auto-Negotiation (AN) frames are exchanged and checked at either end. The transceiver is placed into a different receive mode for this low-rate protocol. When this initial exchange of AN frames is complete, the AN_GOOD_CHECK state is entered and then transmission will switch to a higher data rate Training protocol frame exchange, which must complete within 500 ms of AN starting/restarting. The transceiver is placed into yet another different mode for this part of the bring-up.

Training involves detection and measurement of the received signal and transmission of commands that alter the far-end transmitter characteristics to improve that received signal. This happens in both directions until both ends of the link are receiving the best possible signal.

At that point, Training is flagged as COMPLETE and the AN protocol also completes and sets the AN Link Good flag, which then enables normal Ethernet transmission and reception, with the transceiver being placed into normal operating mode. Any time that AN is restarted or reset, this entire process is repeated.

Using the Configuration Vector for Link Bring-Up

When the optional MDIO interface is omitted from the core, the 10GBASE-KR core block level design which is provided as part of the core deliverables contains some simple logic which automatically restarts Training at the correct stage of the AN protocol. When the AN block is not included with the core, you need to drive the Training control bits of the Configuration Vector in an appropriate manner.

Using the MDIO interface for Link Bring-Up

When the optional MDIO interface is included with the core, there is no specific logic included with the 10GBASE-KR block level design to control Training. You are expected to have a microprocessor controlling the system through the MDIO interface, using the Management interface on the associated MAC. They need to monitor the AN registers which show the current state of the AN protocol and drive the Training protocol control registers according to the IEEE 802.3 standard.

What Can Cause Block Lock to Fail?

Following are suggestions for debugging loss of Block Lock:

- Monitor the state of the `signal_detect` input to the core. This should either be:
  - connected to an optical module to detect the presence of light. Logic 1 indicates that the optical module is correctly detecting light; logic '0' indicates a fault. Therefore, ensure that this is driven with the correct polarity.
  - tied to logic 1 (if not connected to an optical module).

  Note: When `signal_detect` is set to logic 0, this forces the receiver synchronization state machine of the core to remain in the loss of sync state.
- Too many Invalid Sync headers are received. This might or might not be reflected in the HIBER output status bit or MDIO register, depending on how many invalid sync headers are received.

Transceiver-Specific:

- Ensure that the polarities of the txn/txp and rxn/rxp lines are not reversed. If they are, these can be fixed by using the TXPOLARITY and RXPOLARITY ports of the transceiver.
- Check that the transceiver is not being held in reset or still being initialized. The RESETDONE outputs from the transceiver indicate when the transceiver is ready.

What Can Cause the 10Gb PCS/PMA Core to Insert Errors?

On the receive path the 10Gb PCS/PMA core receive state machine can insert block errors RXD=FE, RXC=1. The RX block error happens any time the RX_E state is entered into. It could happen if C, S, D, T are seen out of order. Or it could also happen if an /E/ block type is received, (which is defined in IEEE802.3, Section 49.2.13.2.3 as a 66-bit code with a bad sync header or a control word (C S or T) with no matching translation for the block type field.)

If the RX Elastic Buffer underflows, the core will insert an /E/ block, followed by /L/ blocks.

Transceiver Specific Checks

- Place the transceiver into parallel or serial near-end loopback.
- If correct operation is seen in the transceiver serial loopback, but not when loopback is performed through an optical cable, it might indicate a faulty optical module.
- If the core exhibits correct operation in the transceiver parallel loopback but not in serial loopback, this might indicate a transceiver issue.
- A mild form of bit error rate might be solved by adjusting the transmitter Pre-Emphasis and Differential Swing Control attributes of the transceiver.

Problems with the MDIO

See MDIO Interface for detailed information about performing MDIO transactions.

Things to check for:

- Ensure that the MDIO is driven properly. Check that the `mdc` clock is running and that the frequency is 2.5 MHz or less. Overclocking of the MDIO interface is possible with this core, up to 12.5 MHz.
- Ensure that the 10 Gb Ethernet PCS/PMA core is not held in reset.
- Read from a configuration register that does not have all 0s as a default. If all 0s are read back, the read was unsuccessful. Check that the PRTAD field placed into the MDIO frame matches the value placed on the `prtad[4:0]` port of the core.
- Verify in simulation and/or a Vivado lab tools core capture that the waveform is correct for accessing the host interface for a MDIO read/write.

Link Training

There is currently no Link training algorithm included with the core so it is left up the customers to implement what they need.

It has been noted in hardware testing that the RX DFE logic in the Xilinx transceivers is usually capable of adapting to almost any link so far-end training might not be required at all and the Training Done register/configuration bit can be set as default.

When you decide to implement your own Training Algorithm, that should not only include hardware to monitor the received data signal integrity and provide the inc/dec/preset/initialize commands to send to the far end device, but also must follow the protocol of sending a command until 'updated' is seen on return, and then sending 'hold' until 'not updated' is seen on return.

IMPORTANT: *Also, the priority of commands defined in IEEE 802.3 must be adhered to, such as never transmitting 'Preset' with 'Initialize'.*

The 10GBASE-KR core does include logic that allows it to be trained by a far-end device without user-interaction.

For special considerations when using Link Training with Auto Negotiations see:

- Using Training and Auto Negotiation with the MDIO Interface
- Using Training and AutoNegotiation with No MDIO Interface

Special Design Considerations

This appendix describes considerations that can apply in particular design cases.

Connecting Multiple Core Instances

The example design provided with the core shows the use of a single core but it is possible to use this and the core support layer code to create a design with multiple instances of the core.

This revision of the core has attempted to simplify the integration of multiple cores. Inside the core support layer, it is possible to replicate the block (core) level instances, which each include their core-specific transceiver, clock and reset logic. It is possible to share the rest of the logic in the core support layer between multiple cores.

The GT Common block can be used to supply the reference clock to up to four transceivers, depending on their relative placements.

The shared clock and reset block can be similarly shared between multiple cores. Where multiple cores are to share that block, only one `txclk322` signal needs to be connected from a single core instance to that shared clock and reset block. The `txclk322` outputs from the other cores can be left dangling.

When creating a design with multiple core instances, you need to take care to replicate the correct items and not to replicate items which should be shared.

There is logic in the core support layer which combines the synchronized TX and RX `resetdone` signals to create a single `resetdone` signal.

When multiple instances of the core are required, the synchronized TX and RX `resetdone` signals from each core should be included in this combined signal.

Zynq-7000, Virtex-7, and Kintex-7 Devices

Where multiple 10GBASER/KR cores are required, you should generate one core with 'Include Shared Logic in core' selected and a second core with 'Include Shared Logic in example design'.

The former core, core 'A', can be used to provide the clocks and control signals required by all instances of the latter, core 'B', with no further editing of core output products required.

This simplifies the previous process (detailed in the following section) where you would need to edit the core output products to produce the same result. The architecture of the multi core design resembles that in Figure D-1.

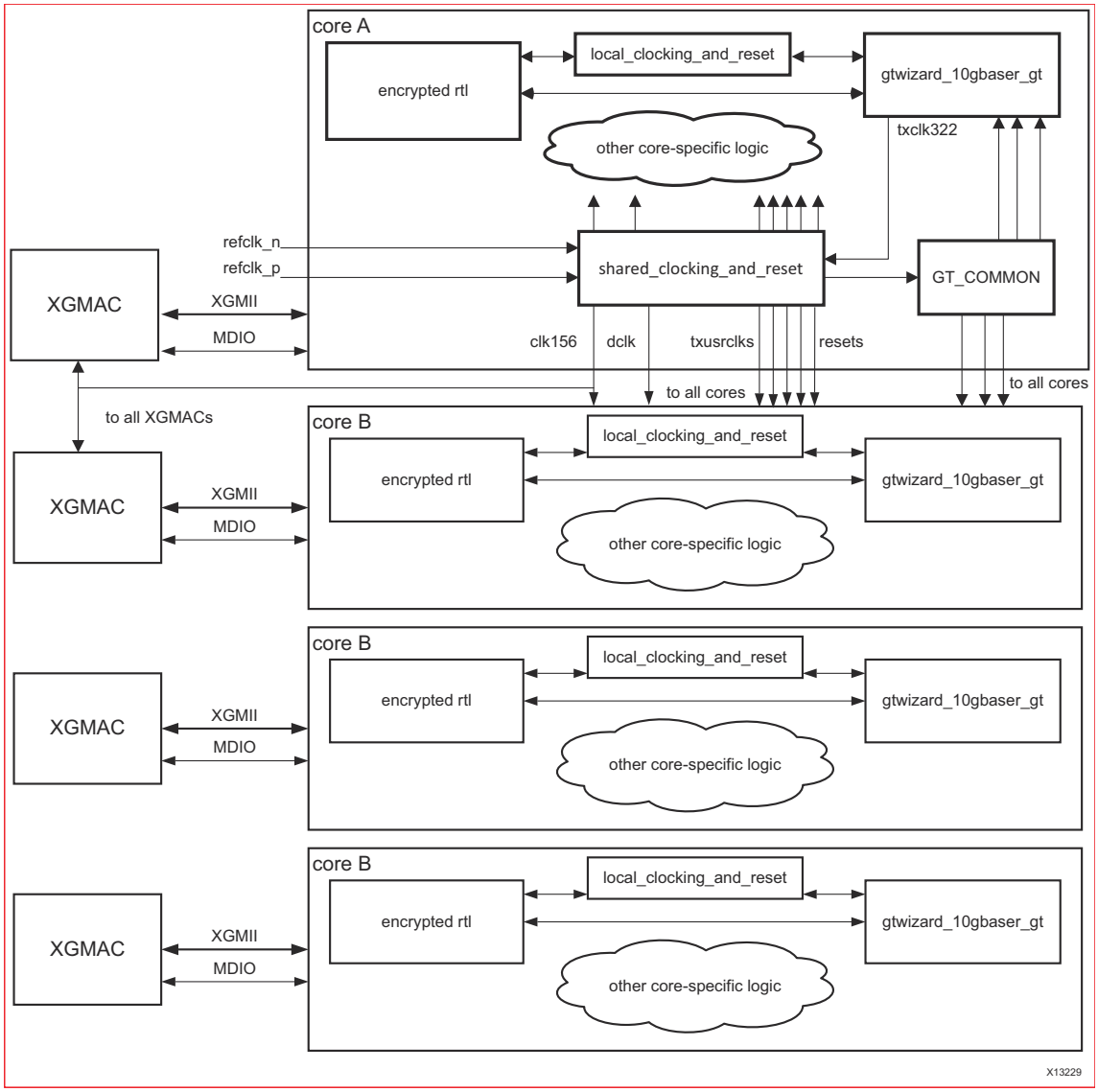

Figure D-1: **Attaching Multiple Cores to a GT_QUAD Tile Using the Shared Logic Feature**

Legacy Design Information

To instance multiple cores, instantiate those in the 10GBASER/KR core support layer component. The shared clocking resources in the core support layer and GT_COMMON blocks can be shared between multiple cores, while the core level must be replicated for each core.

Also shown in Figure D-2 are several Ten Gigabit Ethernet MAC (XGMAC) cores, to illustrate the simplicity of connecting those to the 10GBASE-R/KR cores, to the clock resources and to the MDIO bus. The MDIO 3-state controls are not required for on-chip MDIO connections – connect `mdio_in` on the 10GEMACs to `mdio_out` on the associated 10GBASE-R/KR cores, and vice versa.

The 10GEMAC cores supply the MDC clock to the associated 10GBASE-R/KR cores. Using the `prtad` input to the 10GBASE-R/KR cores, it is possible to differentiate between several cores and have a single XGMAC core controlling all of the 10GBASE-R/KR cores through MDIO.

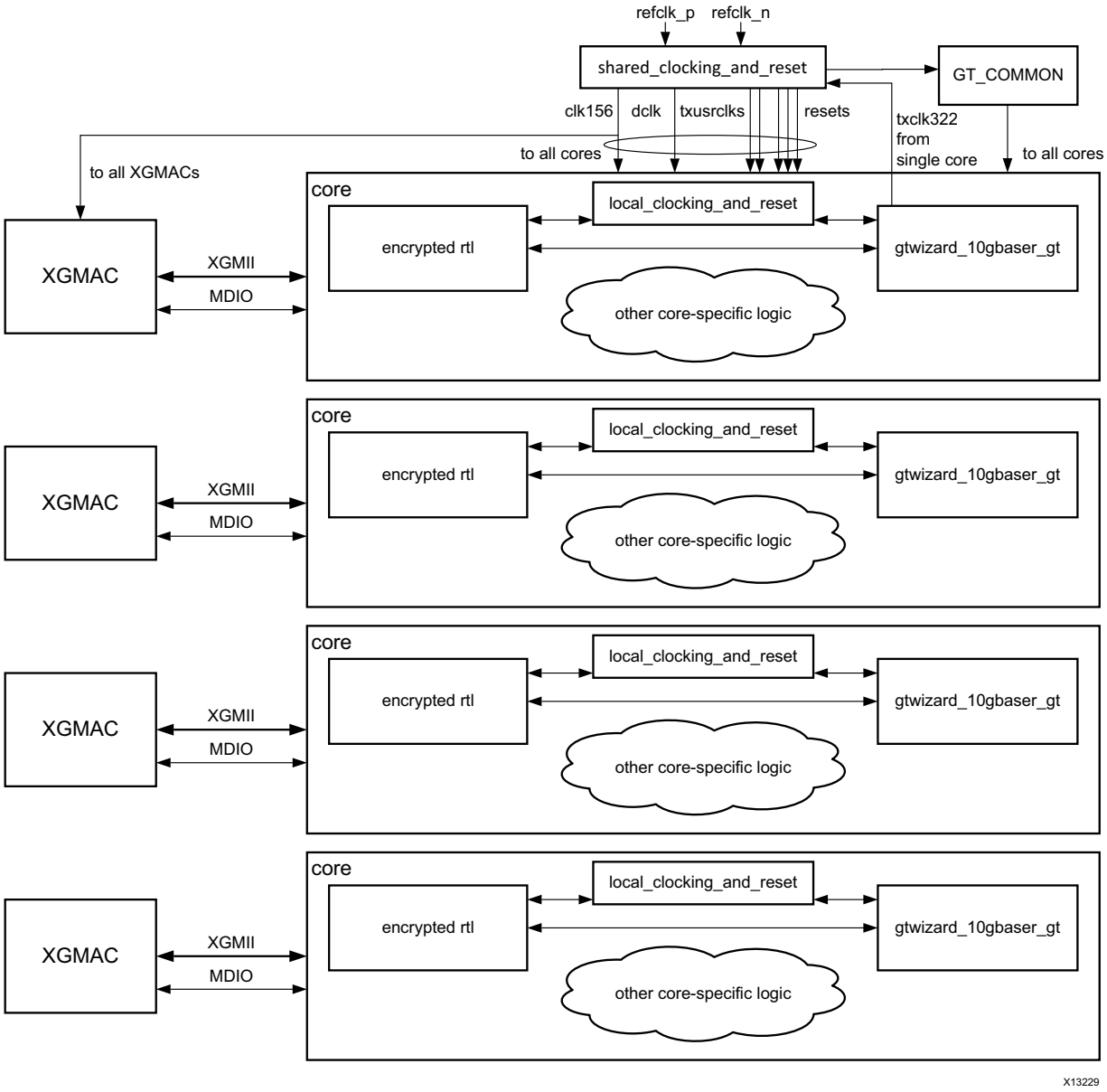

Figure D-2: **Attaching Multiple Cores to a GT_QUAD Tile**

Using Training and Auto Negotiation with the MDIO Interface

This section applies to Zynq®-7000, Virtex®-7, and Kintex®-7 devices.

When a Base KR core is created with the optional MDIO interface and with the optional AutoNegotiation block, there are extra steps that you must take when bringing the core up in a link.

Firstly you need to write to the MDIO registers to disable Training (register bit 1.150.1), and then set the Training Restart bit (register bit 1.150.0, which will self-clear).

Then you need to monitor either the core_status[5] output from the core, or register bit 7.1.2 (which latches Low and clears on read), to wait for the AN Link Up indication which is set in the AN_GOOD_CHECK state. Now you need to Enable Training (1.150.1) and then immediately Restart Training (1.150.0).

Training must complete within 500 ms in order for AutoNegotiation to also complete and set AN Complete. The Training block automatically disables itself if it does get to the Training Done state.

RECOMMENDED: *Currently Xilinx recommends setting the Training Done bit – register bit 1.65520.15. This means that the core will not attempt to train the far-end device but can still be trained by the far-end device.*

If Training does not complete within the time allowed by AutoNegotiation, then you must manually disable Training (register 1.150.1) and restart Training (1.150.0) to allow AutoNegotiation to restart the process.

Using Training and AutoNegotiation with No MDIO Interface

This section applies to Zynq-7000, Virtex-7, and Kintex-7 devices.

When a Base KR core is created with no MDIO interface, logic in the block level can be used to control the interaction between AutoNegotiation and Training.

When AutoNegotiation is not included with the core, or when it is present and it reaches AN Link Up, the Training block is automatically enabled (if the Configuration vector bit 33 to enable Training is also set) and restarted. If AutoNegotiation needs to restart, Training is automatically disabled until AutoNegotiation again reach An Link Up.

If Training is disabled using the Configuration vector bit 33, Training will never be run. If AutoNegotiation is either not included with the core, or is disabled by Configuration vector bit 284, Training can still be used by programming the Configuration bits to drive the process.

Using FEC in the Core with AutoNegotiation

IMPORTANT: *When the FEC feature is recognized in the AutoNegotiation Advertisement (Base Page Ability) data from another 10GBaseKR device, and FEC is requested by the far end, FEC in the core will not be automatically enabled. It is up to you to enable FEC as required.*

The FEC Request bit is in register bit 7.21.15, or on status vector bit 383 if there is no MDIO interface.

Additional Resources

Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see the Xilinx Support website at:

www.xilinx.com/support.

For a glossary of technical terms used in Xilinx documentation, see:

www.xilinx.com/company/terms.htm.

References

These documents provide supplemental material useful with this product guide.

1. *IEEE Std. 802.3-2012*, Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications (standards.ieee.org/findstds/standard/802.3-2012.html)
2. *IEEE Std. 802.3-2012*, Media Access Control (MAC) Parameters, Physical Layers, and Management Parameters for 10-Gb/s Operation (standards.ieee.org/findstds/standard/802.3-2012.html)
3. *7 Series Transceivers User Guide* (UG476)
4. *Vivado Design Suite User Guide - Logic Simulation* (UG900)
5. *Vivado Design Suite User Guide: Designing with IP* (UG896)
6. *Vivado Design Suite User Guide: Getting Started* (UG910)
7. *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994)
8. *Vivado Design Suite User Guide: Programming and Debugging* (UG908)
9. *ISE to Vivado Design Suite Migration Guide* (UG911)
10. *Vivado Design Suite User Guide - Implementation* (UG904)

To search for Xilinx documentation, go to www.xilinx.com/support.

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
07/25/2012	1.0	Initial Xilinx product guide release. This document is based on ds739 and ug692.
10/16/2012	2.0	• Updated for 14.3 and 2012.3 • Added description of MDIO Register 3.4: PCS Speed Ability • Updated for ease of use of document; Updated core interfaces; Added more descriptions of core usage.
12/18/2012	3.0	• Updated for 14.4, 2012.4, and core version 2.6 • Updated Debugging appendix. • Updated false path command and maximum delay paths. • Updated resource numbers. Updated screen captures.
03/20/2013	4.0	• Updated for Vivado Design Tools and core version 3.0. • Removed all ISE® design tools and Virtex®-6 FPGA material. • Updated core hierarchy, constraints, diagrams, resource numbers, and screen captures.
10/02/2013	4.0	• Revision number changed to 4.0 to align with core version number. • Updated core interfaces. • Updated validation status. • Added more material on elastic buffer. • Added more information on core_status vector. • Added Upgrading from Previous Version(s) section to Migration chapter • Updated Figure 3-14. • Updated screen capture in Chapter 4.

Notice of Disclaimer