LogiCORE IP XPS LL FIFO (v1.02a)

DS568 March 1, 2011 **Product Specification**

Introduction

The LogiCORE™ IP XPS LL FIFO is a soft IP core designed for Xilinx FPGAs.

This core allows memory mapped access to a LocalLink interface. The core can be used to interface to the XPS LL TEMAC without the need to use DMA. Other uses include interfacing to the LocalLink interfaces on PLBv46 PCIe and PLBv46 PCI.

The principal operation of this core allows the write or read of data packets to or from a device without any concern over the LocalLink interface. The LocalLink interface is transparent to the user.

Features

- 32-bit PLBv46 slave interface with point to point optimizations
- Independent internal 2 Kb TX and RX data FIFOs
- Full duplex operation
- Provides interrupts for many error and status conditions

LogiCORE IP Facts Table				
Core Specifics				
Supported Device Family	Spartan-3A/-3A DSP, Spartan-3, Spartan-3E, Automotive Spartan-3/-3E/-3A DSP, Spartan-6, Virtex-4/-4Q/-4QV, Virtex-5/-5FX, Virtex-6/-6CX			
Supported User Interfaces	LocalLink			
Resources				
Configuration	LUTs	FFs	DSP Slices	Block RAMs
Config1	996 Min 1002 Max	577 Min 581 Max	473 Min 496 Max	2 Min 2 Max
Provided with Core				
Documentation	Product Specification			
Design Files	VHDL			
Example Design	Not Provided			
Test Bench	Not Provided			
Constraints File	UCF			
Simulation Model	VHDL			
Tested Design Tools				
Design Entry Tools	EDK 11.4 or later			
Simulation	ModelSim PE/SE 6.4b or later			
Synthesis Tools	XST			
Support				
Provided by Xilinx, Inc.				

Functional Description

The Xilinx XPS LL FIFO may be configured for specific systems by selecting the appropriate features through parameter values, thereby providing a design with the minimum necessary resources while providing the desired operational functions. Parameterizable features of the design are discussed in Design Parameters.

Overview

Figure 1 shows the major components in the XPS LL FIFO. The core consists of two FIFOs: a TX FIFO and an RX FIFO. The TX uses one FIFO for the transmit data and for the length data. The RX uses one FIFO for the receive data and the length data. In addition to the FIFOs, there is a register block, a PLBv46 interface, and a LocalLink interface.

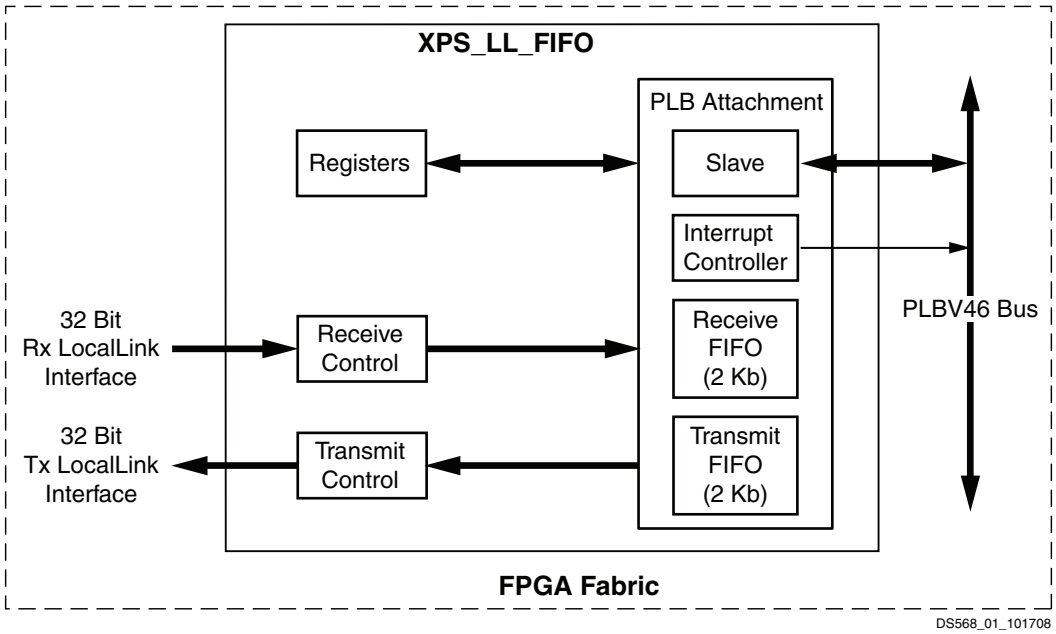


Figure 1: **XPS LL FIFO Block Diagram**

I/O Signals

The XPS LL FIFO core uses a *transparent bus* EDK format to simplify generation of embedded systems by simplifying the connection of signals between the XPS LL FIFO's LocalLink interface and other IP LocalLink interfaces such as the XPS LL TEMAC core. This is the same technique that allows the EDK tools to automatically connect the PLBv46 signals.

The ports on the XPS LL FIFO which connect to the LocalLink are grouped into a virtual bus called LLink. For example, the corresponding signals on the XPS LL TEMAC are grouped into virtual buses called LLink0 andLLink1 depending on what half is used.

To connect a XPS LL FIFO to an XPS LL TEMAC, the user designates to which half of the XPS LL TEMAC they wish to connect by assigning the name to the virtual bus of the XPS LL FIFO that matches the name assigned to the half of the XPS LL TEMAC to be used. In the signal list shown in Table 1, the signals that are assigned to the virtual bus are designated with an interface value of LLink.

Table 1: **I/O Signals**

Signal Name	Interface	Signal Type	Init Status	Description
Top Level System Signals				
SPLB_CLK	System	I		System clock
SPLB_RST	System	I		System Reset (active high)
IP2INTC_Irpt	System	O		System Interrupt
PLBv46 Signals				
PLB_ABus(0:31)	PLB bus	I		PLB address bus
PLB_UABus(0:31)	PLB bus	I		PLB upper address bus
PLB_PAValid	PLB bus	I		PLB primary address valid indicator
PLB_SAValid	PLB bus	I		PLB secondary address valid indicator
PLB_rdPrim	PLB bus	I		PLB secondary to primary read request indicator
PLB_wrPrim	PLB bus	I		PLB secondary to primary write request indicator
PLB_masterID(0:C_SPLB_MID_ WIDTH -1)	PLB bus	I		PLB current master indentifier
PLB_abort	PLB bus	I		PLB abort bus request indicator
PLB_buslock	PLB bus	I		PLB bus lock
PLB_RNW	PLB bus	I		PLB Read not Write
PLB_BE(0:(C_SPLB_DWIDTH/8)-1)	PLB bus	I		PLB byte enables
PLB_MSize(0:1)	PLB bus	I		PLB master data bus size
PLB_size(0:3)	PLB bus	I		PLB transfer size
PLB_type(0:2)	PLB bus	I		PLB transfer type
PLB_lockErr	PLB bus	I		PLB lock error indicator
PLB_wrDBus(0:C_sPLB_ DWIDTH-1)	PLB bus	I		PLB write data bus
PLB_wrBurst	PLB bus	I		PLB burst write transfer indicator
PLB_rdBurst	PLB bus	I		PLB burst read transfer indicator
PLB_wrpendReq	PLB bus	I		PLB pending bus request indicator
PLB_rdpendReq	PLB bus	I		PLB pending bus request indicator
PLB_wrpendPri(0:1)	PLB bus	I		PLB pending request priority
PLB_rdpendPri(0:1)	PLB bus	I		PLB pending request priority
PLB_reqPri(0:1)	PLB bus	I		PLB current request priority
PLB_TAttribute(0:15)	PLB bus	i		PLB Attribute
Sl_addrAck	PLB bus	O		Slave address acknowledge

Table 1: **I/O Signals *(Cont'd)***

Signal Name	Interface	Signal Type	Init Status	Description
Sl_SSize(0:1)	PLB bus	O		Slave data bus size
Sl_wait	PLB bus	O		Slave wait indicator
Sl_rearbitrate	PLB bus	O		Slave rearbitrate bus indicator
Sl_wrDAck	PLB bus	O		Slave write data acknowledge
Sl_wrComp	PLB bus	O		Slave write transfer complete indicator
Sl_wrBTerm	PLB bus	O		Slave terminate write burst transfer
Sl_rdDBus(0:C_SPLB_DWIDTH-1)	PLB bus	O		Slave read data bus
Sl_rdWdAddr(0:3)	PLB bus	O		Slave read word address
Sl_rdDAck	PLB bus	O		Slave read data acknowledge
Sl_rdComp	PLB bus	O		Slave read transfer complete indicator
Sl_rdBTerm	PLB bus	O		Slave terminate read burst transfer
Sl_MBusy(0:C_SPLB_NUM_ MASTERS-1)	PLB bus	O		Slave busy indicator
Sl_MWrErr(0:C_SPLB_NUM_ MASTERS-1)	PLB bus	O		Slave error indicator
Sl_MRdErr(0:C_SPLB_NUM_MASTERS-1)	PLB bus	O		Slave error indicator
Sl_MIRQ(0:C_SPLB_NUM_MASTERS-1)	PLB bus	O		
LocalLink Signals				
LLink_rst	LLink	O		LocalLink Reset
TX LocalLink Signals				
Tx_llink_din(31:0)	LLink	O		TX LocalLink Data Bus
Tx_llink_src_rdy_n	LLink	O		TX LocalLink Source Ready
Tx_llink_dest_rdy_n	LLink	I		TX LocalLink Destination Ready
Tx_llink_sof_n	LLink	O		TX LocalLink Start of Frame
Tx_llink_sop_n	LLink	O		TX LocalLink Start of Payload
Tx_llink_eof_n	LLink	O		TX LocalLink End of Frame
Tx_llink_eop_n	LLink	O		TX LocalLink End of Payload
Tx_llink_rem_n(3:0)	LLink	O		TX LocalLink Rem
RX LocalLink Signals				
Rx_llink_din(31:0)	LLink	I		RX LocalLink Data Bus
Rx_llink_src_rdy_n	LLink	I		RX LocalLink Source Ready
Rx_llink_dest_rdy_n	LLink	O		RX LocalLink Destination Ready
Rx_llink_sof_n	LLink	I		RX LocalLink Start of Frame
Rx_llink_sop_n	LLink	I		RX LocalLink Start of Payload

Table 1: **I/O Signals *(Cont'd)***

Signal Name	Interface	Signal Type	Init Status	Description
Rx_llink_eof_n	**LLink**	I		RX LocalLink End of Frame
Rx_llink_eop_n	LLink	I		RX LocalLink End of Payload
Rx_llink_rem_n(3:0)	LLink	I		RX LocalLink Rem

Design Parameters

To allow the user to generate a XPS LL FIFO that is tailored for their system, certain features are parameterizable. This allows the user to have a design that only utilizes the resources required by their system and runs at the best possible performance. The features that are parameterizable in the Xilinx XPS LL FIFO design are shown in Table 2.

Table 2: **Design Parameters**

Feature/Description	Parameter Name	Allowable Values	Default Values	VHDL Type
Top Level				
Device family	C_FAMILY	virtex6, virtex5, virtex4, qvirtex4, qrvirtex4, spartan3e, aspartan3e, spartan3a, aspartan3a, spartan3an, spartan3adsp, aspartan3adsp, spartan6	virtex5	string
Device base address	C_BASEADDR	See note 1.	FFFFFFFF	std logic vector
Device maximum address	C_HIGHADDR	See note 1.	0x0	std logic vector
PLBv46 Interface				
PLB number of masters	C_SPLB1_NUM_MASTERS	The number of Master Devices connected to the PLB bus	1	integer
PLB master ID width	C_SPLB_MID_WIDTH	The width of the Master ID bus. This is set to roundup(log2 (C_SPLB_NUM_MASTERS) See note 2.	3	integer
PLB Smallest Master Width	C_SPLB_SMALLES_MASTER	Width of the smallest master that will be interacting with this slave. See note 2.	128	integer
PLB master device width	C_SPLB_NATIVE_DWIDTH	32	32	Integer non-VHDL
PLB address bus width (in bits)	C_SPLB_AWIDTH	32	32	integer
PLB data bus width (in bits)	C_SPLB_DWIDTH	32,64,128	32	integer
Selects point-to-point or shared PLB topology	C_SPLB_P2P	0 = Shared Bus Topology 1 = Point-to-Point Bus Topology	0	integer

Notes:

1. The default value will insure that the actual value is set, i.e., if the value is not set, a compiler error will be generated. The address range must be at least 0x4000 (for example, `0x10000000` and `0x10003FFF`). C_BASEADDR must be a multiple of the range, where the range is C_HIGHADDR - C_BASEADDR +1.
2. The value of these parameters are automatically calculated by the XPS tools based on the system specified in the MHS file. The value calculated by XPS for these parameters will override any values supplied in the system MHS file.

Detailed Parameter Descriptions

C_FAMILY

The family parameter is required to implement the core using family specific architecture features. This parameter is automatically updated by the EDK tools based on the project target device information.

C_BASEADDR and C_HIGHADDR

These values are used to generate the read and write enables for the FIFOs and registers and are byte addresses. The address range defined by these parameters must be at least `0x4000`. For example, if the C_BASEADDR is set to `0x10000000`, then C_HIGHADDR must be set to at least `0x10003FFF`. These parameters must be initialized because the default values have been selected so that they will generate an error during build if left unchanged.

C_SPLB_NUM_MASTERS

This parameter is automatically updated by the EDK tools based on the project information.

C_SPLB_MID_WIDTH

This parameter is automatically updated by the EDK tools based on the project information.

C_SPLB_SMALLEST_MASTER

This parameter is defined as an integer and is equal to the native data width of the smallest Master connected to the PLB bus that will be accessing the plbv46_slave attachment. This generic is used to generate and optimize steering logic in the slave attachment. This parameter is automatically updated by the EDK tools based on the project information.

C_SPLB_AWIDTH and C_SPLB_DWIDTH

These parameters should always be set to 32 and 32, 64, and 128 respectively. They are updated by the EDK Tools automatically based on the project information.

C_SPLB_P2P

This parameter is defined as an integer. Setting this parameter to 0 will configure the PLB slave for a PLB shared bus application. Setting this parameter to 1 will configure the PLB slave for a PLB point-to-point bus application. In a point-to-point configuration the core acknowledges all address cycles on the PLB. This reduces some FPGA resources. Latency is also reduced in a point-to-point configuration.

Parameter - Port Dependencies

The width of some of the XPS LL FIFO signals depend on parameters selected in the design. The dependencies between the XPS LL FIFO design parameters and I/O signals are shown in Table 3.

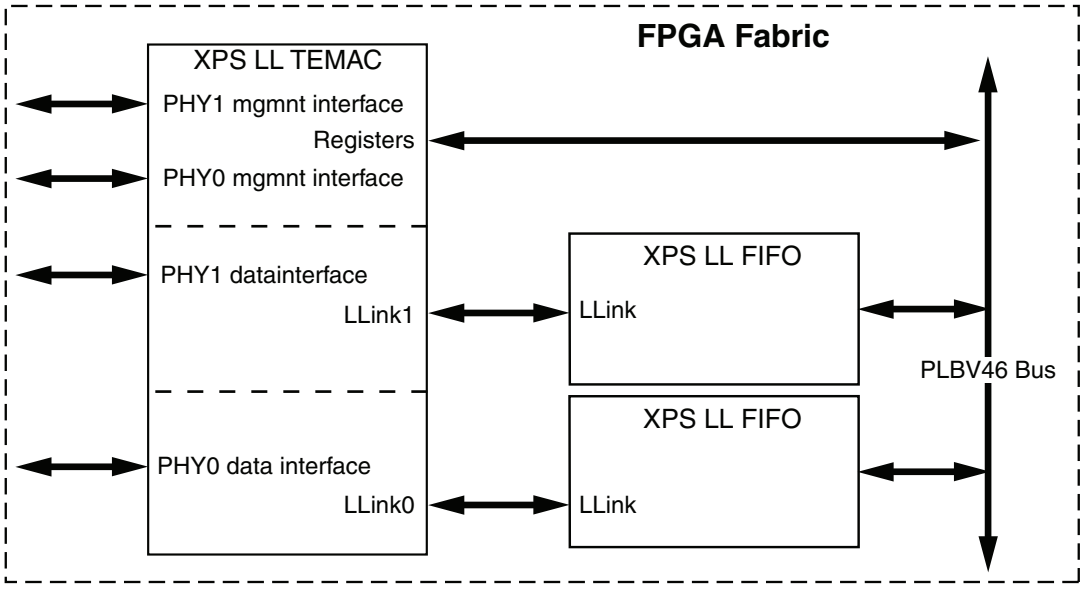
Table 3: Parameter Port Dependencies

Generic or Port	Name	Affects	Depends	Relationship Description
Design Parameters				
G1	C_SPLB_DWIDTH	P1, P3, P4, P6, P7, P8		Specifies the Data Bus width
G2	C_SPLB_AWIDTH	P2, P5		Specifies the Address Bus width
G3	C_SPLB_NUM_MASTERS	P9, P10		Specifies the number of masters on the PLB bus

Table 3: Parameter Port Dependencies

Generic or Port	Name	Affects	Depends	Relationship Description
G4	C_SPLB_MID_WIDTH	P11		Specifies the Master ID bus width
P1	PLB_BE(0:(C_SPLB_DWIDTH/8)-1)		G1	Width varies with the size of the Data bus.
P2	PLB_ABus(0:C_SPLB_WIDTH-1)		G2	Width varies with the size of the Address bus.
P3	PLB_wrDBus(0:C_SPLB_DWIDTH-1)		G1	Width varies with the size of the Data bus.
P4	M_BE(0:(C_SPLB_AWIDTH/8)-1)		G1	Width varies with the size of the Data bus.
P5	M_ABus(0:C_SPLB_AWIDTH-1)		G2	Width varies with the size of the Address bus.
P6	M_wrDBus(0:C_SPLB_DWIDTH-1)		G1	Width varies with the size of the Data bus.
P7	PLB_RdDBus(0:C_SPLB_DWIDTH-1)		G1	Width varies with the size of the Data bus.
P8	SI_RdDBus(0:C_SPLB_DWIDTH-1)		G1	Width varies with the size of the Data bus.
P9	SI_MBusy(0:C_SPLB_NUM_MASTERS-1)		G3	Width varies with the number of masters on the PLB bus
P10	SI_MErr(0:C_SPLB_NUM_MASTERS-1)		G3	Width varies with the number of masters on the PLB bus
P11	PLB_masterID(0:C_SPLB_MID_WIDTH-1)		G4	Width varies with the number of masters on the PLB bus

Figure 2 shows the XPS LL FIFO connected to the XPS LL TEMAC.



Notes:
1. The use of two XPS LL FIFO cores is optional. When using only one half of

Figure 2: **System with XPS LL FIFO connected to dual XPS LL TEMAC1)**

Figure 3 shows a partial code segment from an EDK MHS file which shows an example connection between two XPS LL FIFO cores and a XPS LL TEMAC core.

```
BEGIN xps_ll_temac
 PARAMETER INSTANCE = xps_ll_temac_0
 PARAMETER HW_VER = 1.00.a
 PARAMETER C_BASEADDR = 0x81200000
 PARAMETER C_HIGHADDR = 0x8120ffff
 PARAMETER C_BUS2CORE_CLK_RATIO = 2
 PARAMETER C_TEMAC_TYPE = 0
 PARAMETER C_PHY_TYPE = 1
 PARAMETER C_TEMAC1_ENABLED = 1
 BUS_INTERFACE SPLB = plb_v46_0
 BUS_INTERFACE LLINK0 = LLINK0
 BUS_INTERFACE LLINK1 = LLINK1
 PORT SPLB_Clk = CLK_100MHz
 PORT Core_Clk = CLK_50MHz
 PORT REFCLK = CLK_200MHz
 PORT TemacPhy_RST_n = temacPhy_RST_n
 PORT GTX_CLK_0 = clk_125mhz
 PORT LlinkTemac0_CLK = CLK_100MHz
 PORT LlinkTemac1_CLK = CLK_100MHz
 PORT GMII_TXD_0 = int_GMII_TXD_0
 PORT GMII_TX_EN_0 = int_GMII_TX_EN_0
 PORT GMII_TX_ER_0 = int_GMII_TX_ER_0
 PORT GMII_TX_CLK_0 = int_GMII_TX_CLK_0
 PORT GMII_RXD_0 = int_GMII_RXD_0
 PORT GMII_RX_DV_0 = int_GMII_RX_DV_0
 PORT GMII_RX_ER_0 = int_GMII_RX_ER_0
 PORT GMII_RX_CLK_0 = int_GMII_RX_CLK_0
 PORT MII_TX_CLK_0 = int_MII_TX_CLK_0
 PORT MDC_0  = int_MDC_0
 PORT MDIO_0 = int_MDIO_0
PORT GMII_TXD_1 = int_GMII_TXD_1
 PORT GMII_TX_EN_1 = int_GMII_TX_EN_1
 PORT GMII_TX_ER_1 = int_GMII_TX_ER_1
 PORT GMII_TX_CLK_1 = int_GMII_TX_CLK_1
 PORT GMII_RXD_1 = int_GMII_RXD_1
 PORT GMII_RX_DV_1 = int_GMII_RX_DV_1
 PORT GMII_RX_ER_1 = int_GMII_RX_ER_1
 PORT GMII_RX_CLK_1 = int_GMII_RX_CLK_1
 PORT MII_TX_CLK_1 = int_MII_TX_CLK_1
 PORT MDC_1 = int_MDC_1
 PORT MDIO_1 = int_MDIO_1
PORT TemacIntc0_Irpt = int_TemacIntc0_Irpt
 PORT TemacIntc1_Irpt = int_TemacIntc1_Irpt
END

BEGIN xps_ll_fifo
 PARAMETER INSTANCE = xps_ll_fifo_1
 PARAMETER HW_VER = 1.00.a
 PARAMETER C_BASEADDR = 0x81400000
 PARAMETER C_HIGHADDR = 0x8140ffff
 BUS_INTERFACE SPLB = plb_v46_0
 BUS_INTERFACE LLINK = LLINK1
PORT PLB_Clk = CLK_100MHz
 PORT IP2INTC_Irpt = int_FifoInc1_Irpt
END

BEGIN xps_ll_fifo
 PARAMETER INSTANCE = xps_ll_fifo_0
 PARAMETER HW_VER = 1.00.a
 PARAMETER C_BASEADDR = 0x81300000
 PARAMETER C_HIGHADDR = 0x8130ffff
 BUS_INTERFACE SPLB = plb_v46_0
 BUS_INTERFACE LLINK = LLINK0
 PORT PLB_Clk = CLK_100MHz
 PORT IP2INTC_Irpt = int_FifoInc0_Irpt
END
```

DS568_03_101708

Figure 3: **EDK MHS File Code Segment**

Registers Definition

The registers in Table 4 are contained in the XPS LL FIFO.

Table 4: **XPS LL FIFO Registers**

Register Name	PLB Address	Access
Interrupt Status Register (ISR)	C_BASEADDR + 0x0	Read/Clear on Write(1)
Interrupt Enable Register (IER))	C_BASEADDR + 0x4	Read/Write
Transmit data FIFO reset (TDFR)	C_BASEADDR + 0x8	Write(2)
Transmit data FIFO Vacancy (TDFV)	C_BASEADDR + 0xC	Read
Transmit data FIFO 32bit wide data write port (TDFD)	C_BASEADDR + 0x10	Write
Transmit Length FIFO (TLF)	C_BASEADDR + 0x14	Write
Receive data FIFO reset (RDFR)	C_BASEADDR + 0x18	Write(2)
Receive data FIFO Occupancy (RDFO)	C_BASEADDR + 0x1C	Read
Receive data FIFO 32bit wide data read port (RDFD)	C_BASEADDR + 0x20	Read
Receive Length FIFO (RLF)	C_BASEADDR + 0x24	Read
LocalLink reset (LLR)	C_BASEADDR + 0x28	Write(2)
Reserved	C_BASEADDR + 0x2C - C_BASEADDR + 0x3C	N/A(3)

Notes:

1. The latched interruptible condition is cleared by writing a 1 to that bit location. Writing a 1 to a bit location that is 0 has no effect. Likewise, writing a 0 to a bit location that is 1 has no effect. Multiple bits may be cleared in a single write.
2. Reset if written with 0xA5.
3. If read, these registers will return 0x0. Writing these registers will have no effect.

Interrupt Interface

The interrupt signals generated by the XPS LL FIFO are managed by the ISR and IER registers. An overview diagram of the interrupt control structure is show in Figure 4.

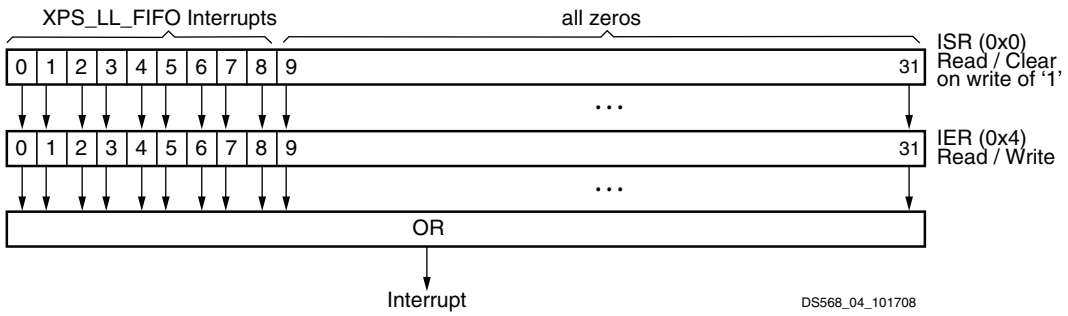


Figure 4: **Interrupt Control Structure**

Interrupt Status Register (ISR)

The Interrupt Status Register is shown in Figure 5. This register combined with the IER register define the interrupt interface of the XPS LL FIFO. The Interrupt Status register uses one bit to represent each XPS LL FIFO internal interruptible condition.

Once an interruptible condition occurs, it will be captured in this register (represented as the corresponding bit being set to 1) even if the condition goes away. The latched interruptible condition is cleared by writing a 1 to that bit location. Writing a 1 to a bit location that is 0 has no effect. Likewise, writing a 0 to a b0it location that is 1 has no effect. Multiple bits may be cleared in a single write.

For any bit set in the Interrupt Status Register, a corresponding bit must be set in the Interrupt Enable Register for the IP2INTC_Irpt signal to be driven active high out of the XPS LL FIFO.

The bits are detailed in Table 5.Interrupt Enable Register (IER)

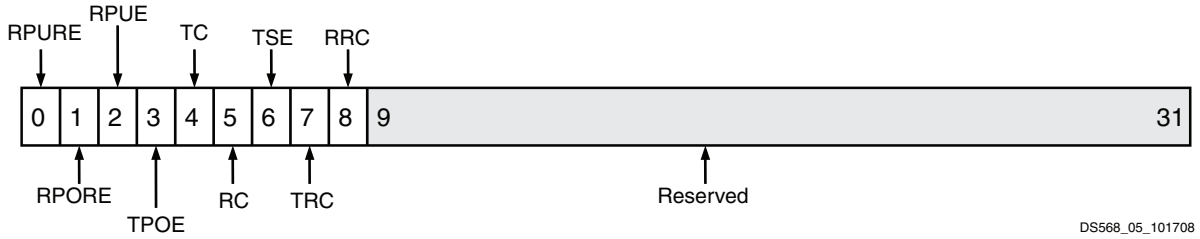


Figure 5: **Interrupt Status Register (offset 0x0)**

Table 5: **Interrupt Status Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
0	RPURE	Read/Clear on Write of "1"	0	**Receive Packet Underrun Read Error**: This interrupt occurs when an attempt is made to read the receive length register when it is empty. The data read is not valid. A reset of the receive logic is required to recover. "0" = No interrupt pending. "1" = Interrupt pending.
1	RPORE	Read/Clear on Write of "1"	0	**Receive Packet Overrun Read Error**. This interrupt occurs when more words are read from the receive data FIFO than are in the packet being processed. Even though the FIFO may not be empty, the read has gone beyond the current packet and removed data from the next packet. A reset of the receive logic is required to recover. "0" = No interrupt pending. "1" = Interrupt pending.
2	RPUE	Read/Clear on Write of "1"	0	**Receive Packet Underrun Error.** This interrupt occurs when an attempt is made to read the receive FIFO when it is empty. The data read is not valid. A reset of the receive logic is required to recover. "0" = No interrupt pending. "1" = Interrupt pending.
3	TPOE	Read/Clear on Write of "1"	0	**Transmit Packet Overrun Error.** This interrupt is generated if an attempt is made to write to the transmit data FIFO when it is full. A reset of the transmit logic is required to recover. "0" = No interrupt pending. "1" = Interrupt pending.

Table 5: **Interrupt Status Register Bit Definitions *(Cont'd)***

Bit(s)	Name	Core Access	Reset Value	Description
4	TC	Read/Clear on Write of "1"	0	**Transmit Complete.** Indicates that at least one transmit has completed. "0" = No interrupt pending. "1" = Interrupt pending.
5	RC	Read/Clear on Write of "1"	0	**Receive Complete.** Indicates that at least one successful receive has completed and that the receive packet data and packet data length is available. This signal is not set for unsuccessful receives. This interrupt may represent more than one packet received so it is important to check the receive data FIFO occupancy value to determine if additional receive packets are ready to be processed. "0" = No interrupt pending. "1" = Interrupt pending.
6	TSE	Read/Clear on Write of "1"	0	**Transmit Size Error.** Transmit Size Error. This interrupt is generated if the number of 32-bit words (including partial words in the count) written to the transmit data FIFO does not match the value written to the transmit length register (bytes) divided by 4 and rounded up to the higher integer value for trailing byte fractions. Interrupts occur only for mismatch of word count (including partial words). Interrupts do not occur due to mismatch of byte count. A reset of the transmit logic is required to recover. "0" = No interrupt pending. "1" = Interrupt pending.
7	TRC	Read/Clear on Write of "1"	0	**Transmit Reset Complete.** This interrupt indicates that a reset of the transmit logic has completed. "0" = No interrupt pending. "1" = Interrupt pending.
8	RRC	Read/Clear on Write of "1"	0	**Receive Reset Complete.** This interrupt indicates that a reset of the receive logic has completed. "0" = No interrupt pending. "1" = Interrupt pending.
9-31	Reserved	Read	0x0	**Reserved.** These bits are reserved for future definition and will always return all zeros.

The Interrupt Enable Register, Figure 6, determines which interrupt sources in the Interrupt Status Register are allowed to generate interrupts.

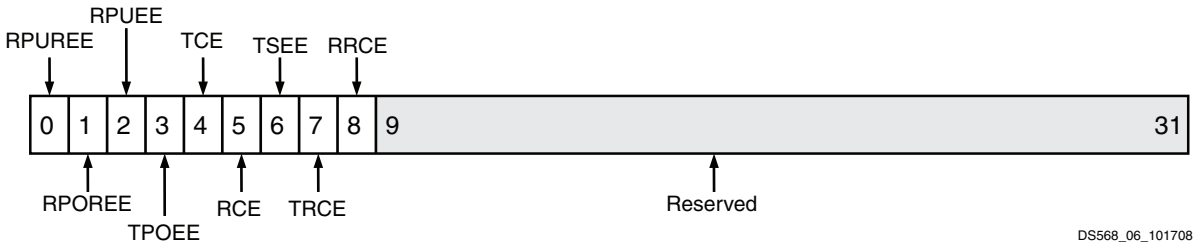


Figure 6: **Interrupt Enable Register (offset 0x4)**

Table 6: **Interrupt Enable Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
0	RPUREE	Read/Write	0	**Receive Packet Underrun Read Error Enable**. This bit is the interrupt enable for the RPURE bit in the ISR. "0" = Mask Interrupt. "1" = Enable Interrupt.
1	RPOREE	Read/Write	0	**Receive Packet Overrun Read Error Enable**. This bit is the interrupt enable for the RPORE bit in the ISR. “0” = Mask Interrupt. "1" = Enable Interrupt.
2	RUREE	Read/Write	0	**Receive Packet Underrun Error Enable.** This bit is the interrupt enable for the RURE bit in the ISR. "0" = Mask Interrupt. "1" = Enable Interrupt.
3	TOREE	Read/Write	0	**Transmit Packet Overrun Error Enable.** This bit is the interrupt enable for the TORE bit in the ISR. "0" = Mask Interrupt. "1" = Enable Interrupt.
4	TCE	Read/Write	0	**Transmit Complete Enable.** This bit is the interrupt enable for the TC bit in the ISR. "0" = Mask Interrupt. "1" = Enable Interrupt.
5	RCE	Read/Write	0	**Receive Complete Enable.** This bit is the interrupt enable for the RC bit in the ISR. "0" = Mask Interrupt. "1" = Enable Interrupt.
6	TMSEE	Read/Write	0	**Transmit Size Error Enable.** This bit is the interrupt enable for the TMSE bit in the ISR. "0" = Mask Interrupt. "1" = Enable Interrupt.
7	TRCE	Read/Write	1	**Transmit Reset Complete Enable.** This bit is the interrupt enable for the TRC bit in the ISR. "0" = Mask Interrupt. "1" = Enable Interrupt.
8	RRCE	Read/Write	1	**Receive Reset Complete Enable.** This bit is the interrupt enable for the RRC bit in the ISR. "0" = Mask Interrupt. "1" = Enable Interrupt.
9-31	Reserved	Read	0x0	**Reserved.** These bits are reserved for future definition and will always return all zeros.

Transmit Data FIFO Reset Register (TDFR)

The Transmit Data FIFO Reset Register, Figure 7, is not an actual register. It is a write-only address, which when written with a specific value, generates a reset for the Transmit Data FIFO. This reset will not occur until transmit activity on the TX LocalLink has completed. The reset can occur only during inactive times on the TX LocalLink and will affect only the transmit circuitry in this core, thereby preventing the core on the other end of the LocalLink from receiving a partial packet which may cause a failure condition in that core.

Because of this mode of operation, it is possible that if the LocalLink becomes stuck in the middle of a LocalLink transaction, for instance, waiting for the destination ready to go active in the middle of a transfer, that the reset will never occur. In such cases it will be necessary to use the LocalLink Reset.

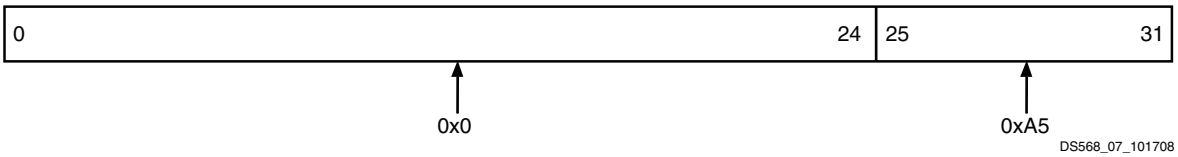


Figure 7: **Transmit Data FIFO Reset Register (offset 0x8)**

Table 7: Transmit Data FIFO Reset Register Bit Definitions

Bit(s)	Name	Core Access	Reset Value	Description
0-31	Reset Key	Write	N/A	Reset Write Value. "0x000000A5" - Generate a reset. others - No effect.

Transmit Data FIFO Vacancy Register (TDFV)

The Transmit Data FIFO Vacancy Register, Figure 8, is a read-only register that gives the vacancy status of the Transmit Data FIFO.

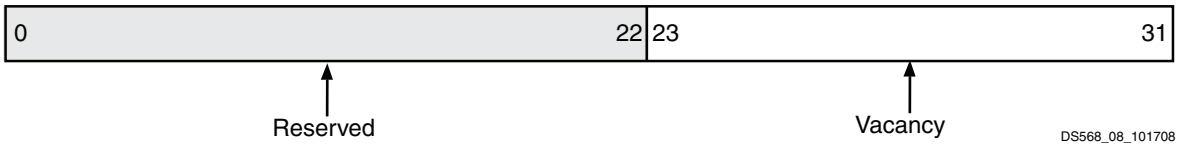


Figure 8: **Transmit Data FIFO Vacancy Register (offset 0xC)**

Table 8: **Transmit Data FIFO Vacancy Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
0-22	Reserved	Read	0x0	**Reserved.** These bits are reserved for future definition and will always return all zeros.
23-31	Vacancy	Read	0x01FE	**Transmit Data FIFO Vacancy**. This is the unsigned value reflecting a current snapshot of the number of 32-bit wide locations available for data storage in the transmit Data FIFO memory core.

Notes:

1. One location is reserved for the storage of the transmit length value for the first packet.

Transmit Data FIFO Data Write Port (TDFD)

The Transmit Data FIFO Data Write Port, Figure 9, is a 32-bit wide address location for writing data into the Transmit Data FIFO. The smallest packet that may be transmitted is four 32-bit words (including partial final word)

which corresponds to 13 to 16 bytes. The maximum packet that may be transmitted is limited by the size of the FIFO and is 510 words (including partial final word) or 2037 to 2040 bytes.

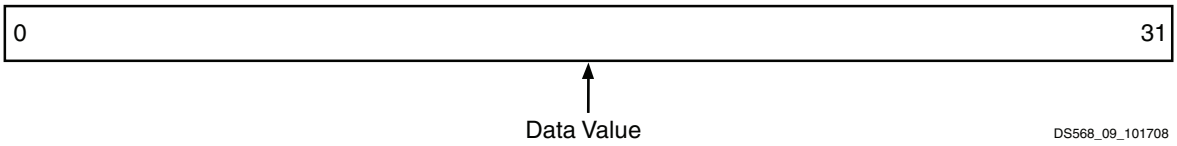


Figure 9: **Transmit Data FIFO Data Write Port (offset 0x10)**

Table 9: **Transmit Data FIFO Data Write Port Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
0-31	Write Data Value	Write	N/A	Transmit Data FIFO Write Value.

Receive Data FIFO Reset Register (RDFR)

The Receive Data FIFO Reset Register, Figure 10, is actually not a register but, rather, a write-only address, which, when written with a specific value, generates a reset for the Receive Data FIFO.

This reset will not occur until receive activity on the RX LocalLink has completed (for example, it only can occur during inactive times on the RX LocalLink) and will only affect the receive circuitry in this core. This prevents the core on the other end of the LocalLink from transmitting a partial packet which may cause failure condition in that core.

Because of this mode of operation, it is possible that if the LocalLink is stuck in the middle of a LocalLink transaction (i.e., if a packet is received over the LocalLink that exceeds the FIFO size of this core causing this core's destination ready to go inactive in the middle of a transfer), that the reset will never occur. In such cases it will be necessary to use the LocalLink Reset.

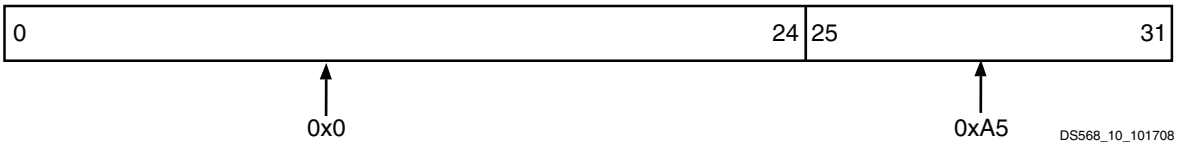


Figure 10: **Receive Data FIFO Reset Register (offset 0x18)**

Table 10: Receive Data FIFO Reset Register Bit Definitions

Bit(s)	Name	Core Access	Reset Value	Description
0-31	Reset Key	Write	N/A	Reset Write Value. "0x000000A5" - Generate a reset. Others - No effect.

Receive Data FIFO Occupancy Register (RDFO)

The Receive Data FIFO Occupancy Register, Figure 11, is a read-only register that gives the occupancy status of the Receive Data FIFO.

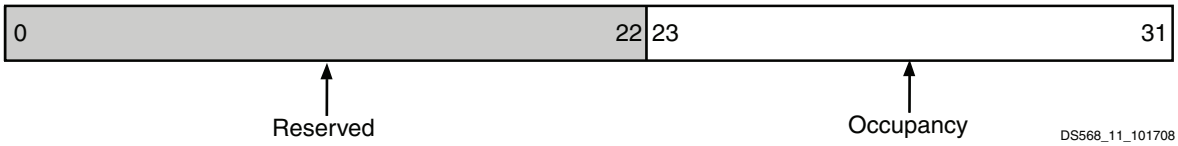


Figure 11: **Receive Data FIFO Occupancy Register (offset 0x1C)**

Table 11: **Receive Data FIFO Occupancy Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
0-22	Reserved	Read	0x0	**Reserved.** These bits are reserved for future definition and will always return all zeros.
23-31	Occupancy	Read	0x0	**Receive Data FIFO Occupancy**. This is the unsigned value reflecting a current snapshot of the number of 32-bit wide locations in use for data storage in the receive Data FIFO memory core. This value is only updated after a packet is successfully received and therefore can be used to determine (a non-zero value) if a receive packet is ready to be processed.

Receive Data FIFO Data Read Port RDFD)

The Receive Data FIFO Data Read Port, Figure 12, is a 32-bit wide address location for reading data from the Receive Data FIFO. The smallest packet that may be received is four 32-bit words (including partial final word) which corresponds to 13 to 16 bytes. The maximum packet that may be received is limited by the size of the FIFO and is 510 words (including partial final word) or 2037 to 2040 bytes.

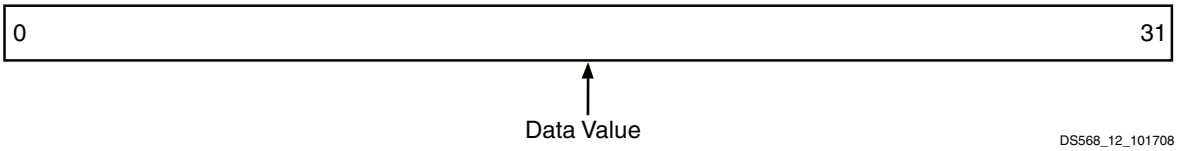


Figure 12: **Receive Data FIFO Data Read Port (offset 0x20)**

Table 12: **Receive Data FIFO Data Read Port Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
0-31	Read Data Value	Read	N/A	Receive Data FIFO Read Value.

Transmit Length Register (TLR)

The transmit length Register shown in Figure 13 is used to store packet length values (the number of bytes in the packet) corresponding to valid packets ready for transmit. The data for the packet is stored in the transmit data FIFO. The data is written to the XPS LL FIFO over the external processor bus interface either by Central DMA or by direct memory mapped access. When presenting a transmit packet to the XPS LL FIFO, the packet data should first be written to the transmit data FIFO then write the length of the packet into the TLR.

Once the packet length is written to the TLR it is automatically moved to the transmit data FIFO with the packet data freeing up the TLR for another value. The packet length must be written to the TLR after the packet data is written to the transmit data FIFO. It is **not** valid to write data for multiple packets to the transmit data FIFO before writing the packet length values.

The action of writing to the transmit length register is used by the XPS LL FIFO to initiate the processing of transmit packets across the LocalLink interface. This continues until all of the TLR values stored are processed. The transmit packet size has to be more than 8 bytes in length.

The width of the TLR is wide enough to support packets up to 2048 bytes in length. The smallest packet that may be transmitted is four 32-bit words (including partial final word) which corresponds to 13 to 16 bytes. The maximum packet that may be transmitted is limited by the size of the FIFO and is 510 words (including partial final word) or 2037 to 2040 bytes.

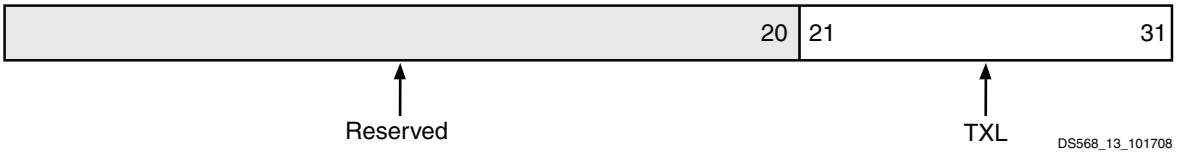


Figure 13: **Transmit Length Register (offset 0x14)**

Table 13: **Transmit Length Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
0-20	Reserved	N/A	0x0	**Reserved.** These bits are reserved for future definition and will always return all zeros.
21-31	TXL	Write	0x0	**Transmit Length**. The number of bytes of the corresponding transmit packet stored in the transmit data FIFO.

Receive Length Register (RLR)

The receive length register shown in Figure 14is used to retrieve packet length values (the number of bytes in the packet) corresponding to valid packets received. The data for the packet is stored in the receive data FIFO.

The length is written by the XPS LL FIFO when the packet is received across the RX LocalLink interface and the receive data FIFO had enough locations that all of the packet data has been saved.

The RLR should only be read when a receive packet is available for processing (the receive occupancy is not zero). Once the RLR is read, the receive packet data should be read from the receive data FIFO before the RLR is read again.

The RLR values are stored in the receive data FIFO by the XPS LL FIFO with each packet's data. The RLR value for the next packet to be processed is moved to the RLR when the previous RLR value has been read.

This register is wide enough to support packets up to 2048 bytes in length. The smallest packet that may be received is four 32-bit words (including partial final word) which corresponds to 13 to 16 bytes. The maximum packet that

may be received is limited by the size of the FIFO and is 510 words (including partial final word) or 2037 to 2040 bytes.

Figure 14: **Receive Length Register (offset 0x24)**

Table 14: **Receive Length Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
0-20	Reserved	Read	0x0	**Reserved.** These bits are reserved for future definition and will always return all zeros.
21-31	RXL	Read	0x0	**Receive Length**. The number of bytes of the corresponding receive data stored in the receive data FIFO.

LocalLink Reset Register (LLR)

The LocalLink Register, Figure 15, is not an register. It is a write-only address, which when written with a specific value, generates an immediate reset for the entire core as well as driving a reset on the external output Llink_rst which can be used to reset the core on the other end of the LocalLink.

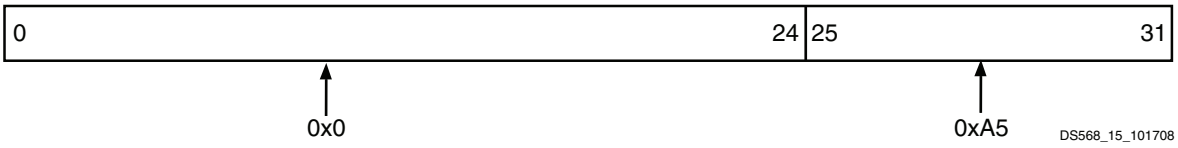


Figure 15: **LocalLink Reset Register (offset 0x28)**

Table 15: **LocalLink Reset Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
0-31	Reset Key	Write	N/A	Reset Write Value. "0x000000A5" - Generate a reset. Others - No effect.

Reserved Registers

Reading from reserved registers will return zeros, and writing to reserved registers will not have any effect. However, any accesses to address offset 0x40 and above will cause undefined results.

Basic Usage

The XPS LL FIFO was designed to provide processor bus access to a LocalLink interface to be used with other IP such as the XPS LL TEMAC. Systems must be built through the Embedded Development Kit to attach the XPS LL FIFO, XPS LL TEMAC, processor, memory, busses, clocking and addition embedded components.

The section is not intended to describe the building of systems, EDK documents describe this process. This section briefly describes operation of the XPS LL FIFO through register accesses using the XPS LL TEMAC as an example.

Packets will automatically be transmitted by writing packet data to the transmit data FIFO followed by writing a length in the TLR.

Receiving packets works in a fashion similar to transmission, although the steps are reversed. Polling of the ISR receive complete or the receive data FIFO occupancy register (not zero) indicates reception of a packet. Reading the RLR provides the packet length in bytes. Given the number of received packet bytes, the appropriate number of reads of the Read data FIFO will provide the packet data.

Table 16 illustrates a power-up read of the registers followed by a transmission and reception of a single packet. Refer to the register definitions for further information and options.

Table 16: **XPS LL FIFO Sample TX/RX Usage**

Register	Access	Value	Activity
Power-up read of register values			
ISR	Read Word	0x01800000	Read interrupt status register
ISR	Write Word	0xFFFFFFFF	Write to clear reset done interrupt bits
ISR	Read Word	0x00000000	Read interrupt status register
IER	Read Word	0x00000000	Read interrupt enable register
TDFV	Read Word	0x000001FE	Read the transmit FIFO vacancy
RDFO	Read Word	0x00000000	Read the receive FIFO occupancy
Transmit a Packet			
TXFIFO_DATA	Write Word	0xFFFFFFFF	4 bytes of destination address
TXFIFO_DATA	Write Word	0xFFFF9ABC	2 bytes of destination address (0xFFFF) 2 bytes source address (0x9ABC)
TXFIFO_DATA	Write Word	0x12345678	4 bytes of source address
TXFIFO_DATA	Write Word	0x002E0001	2 bytes length (0x002E) 2 bytes packet data (0x0001)
TXFIFO_DATA	Write Word	0x00010203	4 bytes of packet data
TXFIFO_DATA	Write Word	0x04050607	4 bytes of packet data
TXFIFO_DATA	Write Word	0x08090A0B	4 bytes of packet data
TXFIFO_DATA	Write Word	0x0C0D0E0F	4 bytes of packet data
TXFIFO_DATA	Write Word	0x10111213	4 bytes of packet data
TXFIFO_DATA	Write Word	0x14151617	4 bytes of packet data
TXFIFO_DATA	Write Word	0x18191A1B	4 bytes of packet data
TXFIFO_DATA	Write Word	0x0C0D0E0F	4 bytes of packet data
TXFIFO_DATA	Write Word	0x20212223	4 bytes of packet data
TXFIFO_DATA	Write Word	0x24252627	4 bytes of packet data
TXFIFO_DATA	Write Word	0x28292A2B	4 bytes of packet data
TDFV	Read Word	0x000001EF	Read the transmit FIFO vacancy
TLR	Write Word	0x0000003C	Transmit length (0x3C = 60 bytes), this starts transmission
ISR	Read Word	0x08000000	A typical value after Tx Complete is indicated by interrupt
ISR	Write Word	0xFFFFFFFF	Write to clear reset done interrupt bits

Table 16: **XPS LL FIFO Sample TX/RX Usage *(Cont'd)***

ISR	Read Word	0x00000000	Read interrupt status register
TDFV	Read Word	0x000001FE	Read the transmit FIFO vacancy
Receive a packet			
ISR	Read Word	0x04000000	A typical value after Rx Complete is indicated by interrupt
ISR	Write Word	0xFFFFFFFF	Write to clear reset done interrupt bits
ISR	Read Word	0x00000000	Read interrupt status register
RDFO	Read Word	0x00000011	Read the receive FIFO occupancy
RLR	Read Word	0x00000040	Receive length (0x40 =64 bytes) indicates number of bytes to read
RDFO	Read Word	0x00000010	Read the receive FIFO occupancy
RXFIFO_DATA	Read Word	0xFFFFFFFF	4 bytes of destination address
RXFIFO_DATA	Read Word	0xFFFF9ABC	2 bytes destination address, (0xFFFF), 2 bytes source address (0x9ABC)
RXFIFO_DATA	Read Word	0x12345678	4 bytes source address
RXFIFO_DATA	Read Word	0x002E0001	2 bytes length, (0x002E), 2 bytes packet data (0x0001)
RXFIFO_DATA	Read Word	0x00010203	4 bytes of packet data
RXFIFO_DATA	Read Word	0x04050607	4 bytes of packet data
RXFIFO_DATA	Read Word	0x08090A0B	4 bytes of packet data
RXFIFO_DATA	Read Word	0x0C0D0E0F	4 bytes of packet data
RXFIFO_DATA	Read Word	0x10111213	4 bytes of packet data
RXFIFO_DATA	Read Word	0x14151617	4 bytes of packet data
RXFIFO_DATA	Read Word	0x18191A1B	4 bytes of packet data
RXFIFO_DATA	Read Word	0x1C1D1E1F	4 bytes of packet data
RXFIFO_DATA	Read Word	0x20212223	4 bytes of packet data
RXFIFO_DATA	Read Word	0x24252627	4 bytes of packet data
RXFIFO_DATA	Read Word	0x28292A2B	4 bytes of packet data
RXFIFO_DATA	Read Word	0x3DC21B56	4 bytes of CRC value
RDFO	Read Word	0x00000000	Read the receive FIFO occupancy (no further receive packets to process)

Design Implementation

Design Tools

The XPS LL FIFO design is implemented using VHDL code. Xilinx XST is the synthesis tool used for synthesizing the XPS LL FIFO core.

Target Technology

The target technology is an FPGA listed in the Supported Device Family field of the LogiCORE IP Facts Table.

Device Utilization and Performance Benchmarks

Because the XPS LL FIFO is a module that will be used with other design pieces in the FPGA, the utilization and timing numbers reported in this section are estimates only. As the XPS LL FIFO core is combined with other pieces of the FPGA design, the utilization of FPGA resources and timing of the XPS LL FIFO core design will vary from the results reported here. The XPS LL FIFO core benchmarks are shown in Figure 17 for a Virtex5-LXT FPGA.

Table 17: **XPS LL FIFO FPGA Performance and Resource Utilization Benchmarks**

Parameter Values	Device Resources			F_{MAX} (MHz)
C_SPLB_P2P	Slices	Slice Flip- Flops	LUTs	F_{MAX}
0	496	577	995	170
1	473	581	1002	166

Reference Documents

1. DS562 PLBv46 Slave Burst
2. LocalLink Interface Specification (SP006)
3. EDK Processor IP Reference Guide
4. IBM CoreConnect 128-Bit Processor Local Bus, Architectural Specification Version 4.6

Support

Xilinx provides technical support for this LogiCORE 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*.

Ordering Information

This Xilinx LogiCORE IP module is provided at no additional cost with the Xilinx ISE Design Suite Embedded Edition software under the terms of the Xilinx End User License. The core is generated using the Xilinx ISE Embedded Edition software (EDK).

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

Reference Documents

1. DS562 PLBv46 Slave Burst
2. LocalLink Interface Specification (SP006)
3. EDK Processor IP Reference Guide
4. IBM CoreConnect 128-Bit Processor Local Bus, Architectural Specification Version 4.6

Revision History

Date	Version	Revision
10/17/08	1.0	Initial Xilinx release.
02/11/09	1.1	Incorporate CR#501446; Added Reserved Registers information
4/24/09	1.6	Replaced references to supported device families and tool name(s) with hyperlink to PDF file.
6/23/10	1.7	Incorporated CR480350 to update SOP and EOP terminology.
9/21/10	1.8	Updated for 12.3 release; made minor edits.
3/1/11	1.9	Updated for 13.1 release.

Notice of Disclaimer