LogiCORE IP Mailbox v1.01b

Product Guide

PG088 December 18, 2012

XILINX®

XILINX

Table of Contents

SECTION I: SUMMARY

Introduction

In a multiprocessor environment, the processors need to communicate data with each other. The easiest method is to set up inter-processor communication through a mailbox. Mailbox features a bidirectional communication channel between two processors. The Mailbox can be connected to the processor either through PLB, AXI4-Lite, AXI4-Stream or FSL interface. The PLB interface option is available for the MicroBlaze™ processor, PowerPC® processor, or any other PLBv46 master. The AXI4-Lite, AXI4-Stream and Fast Simplex Link (FSL) options are available for connection to any IP that supports them, for example MicroBlaze.

Features

- Supports AXI4-Lite, AXI4-Stream, PLB v4.6 and FSL independently on each of the ports
- Configurable depth of mailbox
- Configurable interrupt thresholds and maskable interrupts
- Configurable synchronous or asynchronous operation
- Bidirectional communication

LogiCORE IP Facts Table	
Core Specifics	
Supported Device Family(1)	Zynq™-7000(2), Virtex®-7, Kintex™-7, Artix™-7, Virtex-6, Virtex-5, Spartan®-6, Virtex-4, Spartan-3
Supported User Interfaces	AXI4-Lite, AXI4-Stream, PLB v4.6, FSL
Resources	See Table 2-3.
Provided with Core	
Design Files	ISE: VHDL Vivado: RTL
Example Design	Not Provided
Test Bench	Not Provided
Constraints File	Not Provided
Simulation Model	VHDL Behavioral
Supported S/W Driver(3)	mbox
Tested Design Flows(4)	
Design Entry	Xilinx Platform Studio v14.4 Vivado™ Design Suite v2012.4(5)
Simulation	Mentor Graphics ModelSim Vivado Simulator
Synthesis	ISE® Design Suite Vivado Synthesis(5)
Support	
Provided by Xilinx @ www.xilinx.com/support	

Notes:

1. For a complete list of supported derivative devices, see Embedded Edition Derivative Device Support.
2. Supported in ISE Design Suite implementations only.
3. Standalone driver details can be found in the EDK or SDK directory (<*install_directory*>/doc/usenglish/xilinx_drivers.htm). Linux OS and driver support information is available from //wiki.xilinx.com.
4. For the supported versions of the tools, see the Xilinx Design Tools: Release Notes Guide.
5. Supports only 7 series devices.

Overview

The Mailbox is used for bidirectional inter-processor communication. A mailbox is a link between two otherwise separate processor systems. Other multi-port IP blocks, such as a memory controller, can also be shared by the two sub systems.

In addition to sending the actual data between processors, the mailbox can be used to generate interrupts between the processors.

The Mailbox in a typical AXI4-Lite system is shown in the top-level block diagram in Figure 1-1. The same system partitioning is also used for PLBv46 interface option. FSL and AXI4-Stream options have the Mailbox interface connected directly to a master with no bus in between.

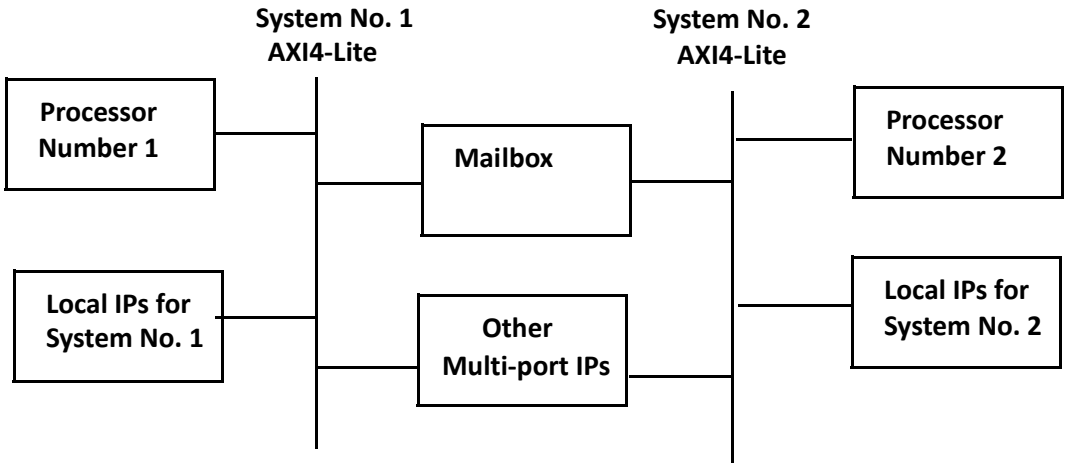

Figure 1-1: **Mailbox in an AXI4-Lite System**

Feature Summary

Bus Interfaces

The Mailbox has two bus interfaces to access the internal resources, usually connected to different processors in a multi-processor system. Both interfaces can be independently configured to use an AXI4-Lite, AXI4-Stream, PLBv46, or FSL interface.

Registers

The Mailbox provides several types of registers, available with AXI4-Lite and PLBv46 bus interfaces, to exchange information and handle interrupts:

- Read and Write Data registers, which provide the primary way to transfer data with the mailbox. These registers act as a FIFO, to allow data transfers from one processor (writing to the FIFO) to the other (reading from the FIFO). The FIFO size can be configured to hold from 16 up to 8192 values.
- Status and control registers, to determine FIFO and interrupt threshold status.
- Interrupt registers, which control the behavior of interrupts, in particular FIFO fill thresholds to determine when an interrupt is generated.

Streaming Access

When using AXI4-Stream or FSL streaming bus interfaces, data transfer FIFOs are available to read from or write to an interface. It is possible to check if the FIFO is full before writing or empty before reading, by using a non-blocking test instruction (for example, `tnput` or *`tnget`*).

Licensing and Ordering Information

This Xilinx® LogiCORE™ IP module is provided at no additional cost with the Xilinx Vivado™ Design Suite and ISE® Design Suite Embedded Edition tools 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 on pricing and availability of other Xilinx LogiCORE IP modules and tools, contact your local Xilinx sales representative.

Product Specification

Standards

The Mailbox adheres to the *ARM® AMBA AXI and ACE Protocol Specification* [Ref 3].

The Mailbox adheres to the *ARM AMBA AXI4-Stream Protocol Specification* [Ref 4].

The Mailbox implements a Processor Local Bus slave interface (see I*BM 128-Bit Processor Local Bus Architectural Specification (v4.6)* [Ref 2]).

Performance

The frequency and latency of the Mailbox are optimized for use with MicroBlaze™. This means that the frequency targets are aligned to MicroBlaze targets.

Maximum Frequencies

Table 2-1 lists clock frequencies for the target families. The maximum achievable clock frequency can vary. The maximum achievable clock frequency and all resource counts can be affected by the tool flow, other tool options, additional logic in the FPGA, different versions of the Xilinx tools, and other factors.

Table 2-1: **Maximum Frequencies**

Architecture	Speed grade	Max Frequency
Spartan®-6	-4	195
Virtex®-6	-3	300
Artix™-7	-3	225
Kintex™-7	-3	320
Virtex-7	-3	320

Latency and Throughput

The latency and throughput of accesses to the Mailbox FIFO depends on the bus interface. The latency for each interface when reading or writing, as well as the throughput, is shown in Table 2-2, according to the parameter settings affecting the measurements.

Table 2-2: **Latency and Throughput**

Bus Interface	Read Latency (clock cycles)	Write Latency (clock cycles)	Throughput (clock cycles/word)
Synchronous Distributed RAM (C_ASYNC_CLKS = 0, C_IMPL_STYPE = 0):			
AXI4-Lite	3	3	6
AXI4-Stream	10	10	20
PLBv46	5	5	10
FSL	10	10	20
Synchronous Block RAM (C_ASYNC_CLKS = 0, C_IMPL_STYPE = 1):			
AXI4-Lite	3	4	7
AXI4-Stream	10	11	21
PLBv46	5	6	11
FSL	10	11	21
Asynchronous Distributed RAM (C_ASYNC_CLKS = 1, C_IMPL_STYPE = 0):			
AXI4-Lite	3	3	10
AXI4-Stream	10	10	24
PLBv46	5	5	13
FSL	10	10	24
Asynchronous Block RAM (C_ASYNC_CLKS = 1, C_IMPL_STYPE = 1):			
AXI4-Lite	3	4	11
AXI4-Stream	10	11	25
PLBv46	5	6	14
FSL	10	11	25

Resource Utilization

Because the Mailbox core is used with other design modules in the FPGA, the utilization and timing numbers reported in this section are estimates only. When the Mailbox core is combined with other designs in the system, the utilization of FPGA resources and timing of the Mailbox design will vary from the results reported here. These values are generated from a minimal dual MicroBlaze system, each with a UART Lite and a shared Mailbox as the only peripherals.

The Mailbox resource utilization for various parameter combinations measured with Virtex-6 as the target device and using the ISE® Design Suite are detailed in Table 2-3.

Table 2-3: **Performance and Resource Utilization Benchmarks on Virtex-6 (xc6vlx240t-ff1156-3)**

Parameter Values (other parameters at default value)				Device Resources				Perfor-mance
C_ASYNC_CLKS	C_INTERCONNECT_PORT_0	C_INTERCONNECT_PORT_1	C_MAILBOX_DEPTH	Slices	Slice Flip-Flops	LUTs	BRAMs	F_{MAX} (MHz)
0	1	1	16	122	194	324	0	326
0	1	1	64	158	225	440	0	323
0	1	1	2048	143	304	358	4	291
0	1	2	16	149	218	328	0	322
0	1	3	16	92	132	233	0	324
0	1	4	16	77	132	236	0	314
1	1	1	16	161	307	394	0	309
1	1	1	64	180	387	501	0	320
1	1	1	2048	200	450	566	4	314
1	1	2	16	135	307	368	0	311
1	1	3	16	98	213	259	0	321
1	1	4	16	86	213	256	0	320

Port Descriptions

The Mailbox has two interfaces that are used to connect to the rest of the system. Both interfaces can be independently configured to use the PLBv46, AXI4-Lite, AXI4-Stream, or FSL interface. The signal descriptions are included in five tables:

1. The PLB signals are described in Table 2-4.
2. The AXI4-Lite signals are described in Table 2-5.
3. The AXI4-Stream signals are described in Table 2-6.
4. The FSL signals are described in Table 2-7.
5. The common signals are described in Table 2-8.

All signals in Table 2-4 through Table 2-7 apply to both interface sides; <x> denotes the interface number, which can be 0 or 1.

Table 2-4: **PLBv46 I/O Signal Description**

Port	Signal Name	Interface	I/O	Initial State	Description
System Signals					
P1	SPLB<x>_Clk	System	I	-	PLB clock
P2	SPLB<x>_Rst	System	I	-	PLB reset, active-High
PLB Interface Signals					
P3	PLB<x>_ABus[0:31]	PLB	I	-	PLB address bus
P4	PLB<x>_PAValid	PLB	I	-	PLB primary address valid
P5	PLB<x>_masterID[0:C_SPLB<x>_MID_WIDTH - 1]	PLB	I	-	PLB current master identifier
P6	PLB<x>_RNW	PLB	I	-	PLB read not write
P7	PLB<x>_BE[0:(C_SPLB<x>_DWIDTH/8) - 1]	PLB	I	-	PLB byte enables
P8	PLB<x>_size[0:3]	PLB	I	-	PLB size of requested transfer
P9	PLB<x>_type[0:2]	PLB	I	-	PLB transfer type
P10	PLB<x>_wrDBus[0:C_SPLB<x>_DWIDTH - 1]	PLB	I	-	PLB write data bus
Unused PLB Interface Signals					
P11	PLB<x>_UABus[0:31]	PLB	I	-	PLB upper address bits
P12	PLB<x>_SAValid	PLB	I	-	PLB secondary address valid
P13	PLB<x>_rdPrim	PLB	I	-	PLB secondary to primary read request indicator

Table 2-4: **PLBv46 I/O Signal Description *(Cont'd)***

Port	Signal Name	Interface	I/O	Initial State	Description
P14	PLB<x>_wrPrim	PLB	I	-	PLB secondary to primary write request indicator
P15	PLB<x>_abort	PLB	I	-	PLB abort bus request
P16	PLB<x>_busLock	PLB	I	-	PLB bus lock
P17	PLB<x>_MSize[0:1]	PLB	I	-	PLB data bus width indicator
P18	PLB<x>_lockErr	PLB	I	-	PLB lock error
P19	PLB<x>_wrBurst	PLB	I	-	PLB burst write transfer
P20	PLB<x>_rdBurst	PLB	I	-	PLB burst read transfer
P21	PLB<x>_wrPendReq	PLB	I	-	PLB pending bus write request
P22	PLB<x>_rdPendReq	PLB	I	-	PLB pending bus read request
P23	PLB<x>_wrPendPri[0:1]	PLB	I	-	PLB pending write request priority
P24	PLB<x>_rdPendPri[0:1]	PLB	I	-	PLB pending read request priority
P25	PLB<x>_reqPri[0:1]	PLB	I	-	PLB current request priority
P26	PLB<x>_TAttribute[0:15]	PLB	I	-	PLB transfer attribute
PLB Slave Interface Signals					
P27	SI<x>_addrAck	PLB	O	0	Slave address acknowledge
P28	SI<x>_SSize[0:1]	PLB	O	0	Slave data bus size
P29	SI<x>_wait	PLB	O	0	Slave wait
P30	SI<x>_rearbitrate	PLB	O	0	Slave bus rearbitrate
P31	SI<x>_wrDAck	PLB	O	0	Slave write data acknowledge
P32	SI<x>_wrComp	PLB	O	0	Slave write transfer complete
P33	SI<x>_rdDBus[0: C_SPLB<x>_DWIDTH - 1]	PLB	O	0	Slave read data bus
P34	SI<x>_rdDAck	PLB	O	0	Slave read data acknowledge
P35	SI<x>_rdComp	PLB	O	0	Slave read transfer complete
P36	SI<x>_MBusy[0: C_SPLB<x>_NUM_MASTERS - 1]	PLB	O	0	Slave busy
P37	SI<x>_MWrErr[0: C_SPLB<x>_NUM_MASTERS - 1]	PLB	O	0	Slave write error
P38	SI<x>_MRdErr[0: C_SPLB<x>_NUM_MASTERS - 1]	PLB	O	0	Slave read error
Unused PLB Slave Interface Signals					
P39	SI<x>_wrBTerm	PLB	O	0	Slave terminate write burst transfer
P40	SI<x>_rdWdAddr[0:3]	PLB	O	0	Slave read word address
P41	SI<x>_rdBTerm	PLB	O	0	Slave terminate read burst transfer
P42	SI<x>_MIRQ[0: C_SPLB<x>_NUM_MASTERS - 1]	PLB	O	0	Master interrupt request

Table 2-5: **AXI4-Lite I/O Signal Description**

Port	Signal Name	Interface	I/O	Initial State	Description
System Signals					
P43	S<x>_AXI_ACLK	System	I	-	AXI Clock
P44	S<x>_AXI_ARESETN	System	I	-	AXI Reset, active-Low
AXI Write Address Channel Signals					
P45	S<x>_AXI_AWADDR[C_S<x>_AXI_ADDR_WIDTH-1:0]	AXI	I	-	AXI Write address. The write address bus gives the address of the write transaction.
P46	S<x>_AXI_AWVALID	AXI	I	-	Write address valid. This signal indicates that valid write address is available.
P47	S<x>_AXI_AWREADY	AXI	O	0	Write address ready. This signal indicates that the slave is ready to accept an address.
AXI Write Channel Signals					
P48	S<x>_AXI_WDATA[C_S<x>_AXI_DATA_WIDTH - 1: 0]	AXI	I	-	Write data
P49	S<x>_AXI_WSTB[C_S<x>_AXI_DATA_WIDTH/8-1:0] (1)	AXI	I	-	Write strobes. This signal indicates which byte lanes to update in memory.(1)
P50	S<x>_AXI_WVALID	AXI	I	-	Write valid. This signal indicates that valid write data and strobes are available.
P51	S<x>_AXI_WREADY	AXI	O	0	Write ready. This signal indicates that the slave can accept the write data.
AXI Write Response Channel Signals					
P52	S<x>_AXI_BRESP[1:0]	AXI	O	0x0	Write response. This signal indicates the status of the write transaction. 00 - OKAY 10 - SLVERR 11 - DECERR
P53	S<x>_AXI_BVALID	AXI	O	0	Write response valid. This signal indicates that a valid write response is available.
P54	S<x>_AXI_BREADY	AXI	I	-	Response ready. This signal indicates that the master can accept the response information.

Table 2-5: **AXI4-Lite I/O Signal Description *(Cont'd)***

Port	Signal Name	Interface	I/O	Initial State	Description
AXI Read Address Channel Signals					
P55	S<x>_AXI_ARADDR[C_S<x>_ AXI_ADDR_WIDTH -1:0]	AXI	I	-	Read address. The read address bus gives the address of a read transaction.
P56	S<x>_AXI_ARVALID	AXI	I	-	Read address valid. This signal indicates, when High, that the read address is valid and remains stable until the address acknowledge signal, S<x>_AXI_ARREADY, is High.
P57	S<x>_AXI_ARREADY	AXI	O	1	Read address ready. This signal indicates that the slave is ready to accept an address.
AXI Read Data Channel Signals					
P58	S<x>_AXI_RDATA[C_S<x>_AXI_ DATA_WIDTH -1:0]	AXI	O	0x0	Read data
P59	S<x>_AXI_RRESP[1:0]	AXI	O	0x0	Read response. This signal indicates the status of the read transfer. 00 - OKAY 10 - SLVERR 11 - DECERR
P60	S<x>_AXI_RVALID	AXI	O	0	Read valid. This signal indicates that the required read data is available and the read transfer can complete
P61	S<x>_AXI_RREADY	AXI	I	-	Read ready. This signal indicates that the master can accept the read data and response information

Notes:

1. This signal is not used. The Mailbox assumes that all byte lanes are active.

Table 2-6: **AXI4-Stream I/O Signal Description**

Port	Signal Name	Interface	I/O	Initial State	Description
System Signals					
P62	S<x>_AXIS_ACLK	System	I	-	AXI Clock
P63	M<x>_AXIS_ACLK	System	I	-	AXI Clock
AXI Slave Channel Signals					
P64	S<x>_AXIS_TDATA[C_S<x>_AXIS_DATA_WIDTH - 1: 0]	AXIS	I	-	Data
P65	S<x>_AXIS_TLAST	AXIS	I	-	Last data flag, indicates that this is the last word.
P66	S<x>_AXIS_TVALID	AXIS	I	-	Data valid. This signal indicates that valid data and last flag are available.
P67	S<x>_AXIS_TREADY	AXIS	O	0	Data ready. This signal indicates that the slave can accept the data.
AXI Master Channel Signals					
P68	M<x>_AXIS_TDATA[C_M<x>_AXIS_DATA_WIDTH -1:0]	AXIS	O	0x0	Data
P69	M<x>_AXIS_TLAST	AXIS	O	0	Last data flag, indicates that this is the last word.
P70	M<x>_AXIS_TVALID	AXIS	O	0	Data valid. This signal indicates that valid data and last flag are available.
P71	M<x>_AXIS_TREADY	AXIS	I	-	Data ready. This signal indicates that the slave can accept the data.

Table 2-7: **FSL I/O Signal Description**

Port	Signal Name	Interface	I/O	Initial State	Description
FSL Master Interface Signals					
P72	FSL<x>_M_Clk	MFSL	I	N/A	This port provides the input clock to the FSL master interface of the mailbox when used in the asynchronous FIFO mode (C_ASYNC_CLKS = 1). All transactions on the master interface use this clock when implemented in the asynchronous mode
P73	FSL<x>_M_Data	MFSL	I	0	The data input to the FSL master interface of the mailbox
P74	FSL<x>_M_Control	MFSL	I	0	Unused for mailbox
P75	FSL<x>_M_Write	MFSL	I	0	Input signal that controls the write enable signal of the FSL master interface of the FIFO. When set to 1, the value of FSL<x>_M_Data is pushed into the mailbox FIFO on a rising clock edge.
P76	FSL<x>_M_Full	MFSL	O	N/A	Output signal on the FSL master interface of the FIFO indicating that the FIFO is full.
FSL Slave Interface Signals					
P77	FSL<x>_S_Clk	SFSL	I	N/A	This port provides the input clock to the FSL slave interface on the mailbox when used in the asynchronous FIFO mode (C_ASYNC_CLKS = 1). All transactions on the slave interface use this clock when implemented in the asynchronous mode
P78	FSL<x>_S_Data	SFSL	O	N/A	The data output bus onto the FSL slave interface of the mailbox
P79	FSL<x>_S_Control	SFSL	O	N/A	Unused for mailbox
P80	FSL<x>_S_Read	SFSL	I	0	Input signal on the FSL slave interface that controls the read acknowledge signal of the FIFO. When set to 1, the value of FSL<x>_S_Data is popped from the FIFO on a rising clock edge.
P81	FSL<x>_S_Exists	SFSL	O	N/A	Output signal on the FSL slave interface indicating that FIFO contains valid data.

Table 2-8: **Mailbox Common I/O Signal Description**

Port	Signal Name	Interface	I/O	Initial State	Description
FSL Common Interface Signals					
P82	FSL_Clk	System	I	N/A	This is the input clock to the mailbox when used in synchronous FIFO mode (C_ASYNC_CLKS = 0) and both interfaces are FSL or AXI4-Stream based (C_INTERCONNECT_PORT_<x> = 3 or 4). The FSL_Clk is in this case used to clock the core, in all other cases are the internal mailbox clock automatically derived from either SPLB<x>_Clk, S<x>_AXI_ACLK or FSL<x>_M_Clk depending on the settings.
P83	SYS_Rst	System	I	N/A	External system reset. This signal is only required when both interfaces are configured to be streaming interfaces (FSL or AXI4-Stream). If any PLB or AXI4-Lite interface is available this signal is optional.
P84	FSL_Rst	System	O	0	Output reset signal generated by the FSL reset logic. Any peripherals connected to the FSL bus can use this reset signal to operate the peripheral reset.
Common Signals					
P85	Interrupt_0	System	O	0	Interrupt signal that data is available at interface 0
P86	Interrupt_1	System	O	0	Interrupt signal that data is available at interface 1

Register Space

Each interface of the Mailbox core has the same set of information registers. The information at each interface is not identical but rather localized for that interface because the communication is bidirectional.

Table 2-9 shows all the Mailbox registers and their addresses for the PLB and AXI4-Lite cases. Much of the information can be acquired for the FSL and AXI4-Stream cases with the use of FSL<x>_M_Full/FSL<x>_S_Exists or S<x>_AXIS_TREADY/M<x>_AXIS_TVALID respectively.

Table 2-9: **Mailbox Registers**

Base Address + Offset (hex)	Register Name	Access Type	Default Value (hex)	Description
BASEADDR + 0x0	WRDATA	Write	N/A	Write Data address. Write only.
BASEADDR + 0x4	Reserved	N/A	N/A	Reserved for future use
BASEADDR + 0x8	RDDATA	Read	N/A	Read Data address. Read only
BASEADDR + 0xC	Reserved	N/A	N/A	Reserved for future use
BASEADDR + 0x10	STATUS	Read	0x1	Status flags for mailbox. Read only.
BASEADDR + 0x14	ERROR	Read	0x0	Error flags, clear on read. Read only.
BASEADDR + 0x18	SIT	-	-	Send Interrupt Threshold. Read/Write
BASEADDR + 0x1C	RIT	-	-	Receive Interrupt Threshold. Read/Write
BASEADDR + 0x20	IS	-	-	Interrupt Status register. Read/Write
BASEADDR + 0x24	IE	-	-	Interrupt Enable register. Read/Write
BASEADDR + 0x28	IP	-	-	Interrupt Pending register. Read only
BASEADDR + 0x2C	Reserved	-	-	Reserved for future use
BASEADDR + 0x30	Reserved	-	-	Reserved for future use
BASEADDR + 0x34	Reserved	-	-	Reserved for future use
BASEADDR + 0x38	Reserved	-	-	Reserved for future use
BASEADDR + 0x3C	Reserved	-	-	Reserved for future use

Write Data Register (WRDATA)

Writing to this register results in the data being transferred to the RDDATA register at the other interface. Trying to write while the full flag is set results in an error and the FULL_ERROR bit is set. The register is write only and a read request issued to WRDATA is ignored. Bit assignment in the WRDATA register is described in Table 2-11.

Table 2-10: **Write Data Register**

WRDATA	
0	C_FSL_DWIDTH-1

Table 2-11: **Mailbox Write Data Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
0 - C_FSL_DWIDTH - 1	WRDATA	Write	-	Write register to send data to the other interface

Mailbox Read Data Register (RDDATA)

Reading from this register pops one value from the mail FIFO. Trying to read while the empty flag is set results in an error and the EMPTY_ERROR bit is set. The register is read only and a write request issued to RDDATA is ignored. Bit assignment in the RDDATA register is described in Table 2-13.

Table 2-12: **Read Data Register**

RDDATA	
0	C_FSL_DWIDTH-1

Table 2-13: **Mailbox Read Data Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
0 - C_FSL_DWIDTH - 1	RDDATA	Read	-	Read register to get data word sent from the other interface

Mailbox Status Register (STATUS)

The Mailbox Status Register contains the current status of the mailbox. The register is read only and a write request issued to STATUS is ignored. Bit assignment in the STATUS register is described in Table 2-15.

Table 2-14: **Status Register**

Reserved		RTA	STA	Full	Empty
0	27	28	29	30	31

Table 2-15: **Mailbox Status Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
0 - 27	Reserved			Reserved for future use
28	RTA	Read	0	Receive Threshold Active indicates the current FIFO status of this interface in the receive direction 0 = The receive FIFO level is less than or equal to the RIT threshold 1 = The receive FIFO level is greater than the RIT threshold
29	STA	Read	0	Send Threshold Active indicates the current FIFO status of this interface in the send direction 0 = The send FIFO level is greater than the SIT threshold 1 = The send FIFO level is less than or equal to the SIT threshold
30	Full	Read	0	Indicates the current status of this interface in the send direction 0 = There is room for more data 1 = The FIFO is full; any attempts to write data are ignored and an error is generated
31	Empty	Read	1	Indicates the current status of this interface in the receive direction 0 = There is data available 1 = The FIFO is empty, any attempts to read data are ignored and an error is generated

Mailbox Error Register (ERROR)

The Mailbox Error Register contains the error flags for PLB and AXI4-Lite accesses from this interface. The error register is cleared at read, this means that all bits are sticky and that they indicate any errors that occurred since last time the error register was read. The register is read only and a write request issued to ERROR is ignored. Bit assignment in the ERROR register is described in Table 2-17.

Table 2-16: **Error Register**

Reserved		Full Error	Empty Error
0	29	30	31

Table 2-17: **Mailbox Error Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
0 - 29	Reserved			Reserved for future use
30	Full Error	Read	0	Indicates if there has been any attempts to write to the WRDATA register while the Full flag was asserted since the error register was last read 0 = No error has occurred 1 = One or more attempts to write while FSL link was full
31	Empty Error	Read	0	Indicates if there has been any attempts to read from the RDDATA register while the Empty flag was asserted since the error register was last read 0 = No error has occurred 1 = One or more attempts to read while FSL link was empty

Mailbox Send Interrupt Threshold Register (SIT)

The Mailbox Send Interrupt Threshold Register contains the interrupt threshold for this interface in the send direction. Depending on the send FIFO data level writing a new SIT can cause a rising edge on STA that can generate a STI interrupt if it is enabled in the IE register. Bit assignment in the SIT register is described in Table 2-19.

Table 2-18: **SIT Register**

	SIT	
0	32-Log2(C_MAILBOX_DEPTH)	31

Table 2-19: **Mailbox SIT Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
Log2(C_MAILBOX_DEPTH)	SIT	Read/Write	0	Lower Log2(C_MAILBOX_DEPTH) bits used, right justified to bit 31

Mailbox Receive Interrupt Threshold Register (RIT)

The Mailbox Receive Interrupt Threshold Register contains the interrupt threshold for this interface in the receive direction. Depending on the receive FIFO data level writing a new RIT can cause a rising edge on RTA that can generate a RTI interrupt if it is enabled in the IE register. Bit assignment in the RIT register is described in Table 2-21.

Table 2-20: **RIT Register**

	RIT	
0	32-Log2(C_MAILBOX_DEPTH)	31

Table 2-21: **Mailbox RIT Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
Log2(C_MAILBOX_DEPTH)	RIT	Read/ Write	0	Lower Log2(C_MAILBOX_DEPTH) bits used, right justified to bit 31

Mailbox Interrupt Status Register (IS)

The Mailbox Interrupt Status Register contains the current interrupt status for this interface. There are three types of interrupts that can be generated. Mailbox Error interrupt are generated when any of the bits in the ERROR register is set. The other two interrupts are FIFO related: RTI is generated for a rising edge on the RTA bit in the STATUS register and STI that is generated for a rising edge on the STA STATUS register bit. RTI and STI are used to indicate that it is time to read from or write to the FIFOs to avoid any stalls in the data flow. Bit assignment in the IS register is described in Table 2-23.

Table 2-22: **IS Register**

Reserved		ERR	RTI	STI
0	28	29	30	31

Table 2-23: **Mailbox IS Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
0 - 28	Reserved			Reserved for future use
29	ERR	Read/Write	0	Mailbox Error Interrupt Status for this interface. Values for read: 0 = No interrupt event has occurred. 1 = A Mailbox error has occurred. Values for write: 0 = No change 1 = Acknowledge and clear the interrupt if it is active

Table 2-23: **Mailbox IS Register Bit Definitions *(Cont'd)***

Bit(s)	Name	Core Access	Reset Value	Description
30	RTI	Read/Write	0	Mailbox Receive Threshold Interrupt pending status for this interface. Values for read: 0 = No interrupt event has occurred. 1 = Data level in the receive FIFO has caused a RTI. Values for write: 0 = No change 1 = Acknowledge and clear the interrupt if it is active
31	STI	Read/Write	0	Mailbox Send Threshold Interrupt pending status for this interface. Values for read: 0 = No interrupt event has occurred. 1 = Data level in the send FIFO has caused a STI. Values for write: 0 = No change 1 = Acknowledge and clear the interrupt if it is active

Mailbox Interrupt Enable Register (IE)

The Mailbox Interrupt Enable Register contains the mask for the allowed interrupts on this interface. Bit assignment in the IE register is described in Table 2-25.

Table 2-24: **IE Register**

Reserved		ERR	RTI	STI
0	28	29	30	31

Table 2-25: **Mailbox IE Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
0 - 28	Reserved			Reserved for future use
29	ERR	Read/Write	0	Mailbox Error Interrupt Enable for this interface 0 = ERR interrupt is disabled 1 = ERR interrupt is enabled
30	RTI	Read/Write	0	Mailbox Receive Threshold Interrupt Enable for this interface 0 = RTI interrupt is disabled 1 = RTI interrupt is enabled
31	STI	Read/Write	0	Mailbox Send Threshold Interrupt Enable for this interface 0 = STI interrupt is disabled 1 = STI interrupt is enabled

Mailbox Interrupt Pending Register (IP)

The Mailbox Interrupt Pending Register contains the currently pending interrupts from this interface. It is a read only register generated by performing a bitwise AND between the IS and IE registers. A write request issued to the IP is ignored. Bit assignment in the IP register is described in Table 2-27. All the bits in this register are OR'd together to generate the interrupt output signal for this interface. When an interrupt has been serviced it is acknowledged by writing the corresponding bit to the IS Register.

Table 2-26: **IP Register**

Reserved		ERR	RTI	STI
0	28	29	30	31

Table 2-27: **Mailbox IP Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
0 - 28	Reserved			Reserved for future use
29	ERR	Read	0	Mailbox Error Interrupt Pending status for this interface 0 = No pending interrupt 1 = Pending interrupt for Mailbox errors
30	RTI	Read	0	Mailbox Receive Threshold Interrupt Pending status for this interface 0 = No pending interrupt 1 = Pending interrupt for data level in receive FIFO
31	STI	Read	0	Mailbox Send Threshold Interrupt Pending status for this interface 0 = No pending interrupt 1 = Pending interrupt for data level in send FIFO

Designing with the Core

General Design Guidelines

This chapter includes guidelines and additional information to facilitate designing with the core.

Clocking

The SPLBn_Clk (n = 0, 1) input is only used when the PLBv46 bus is used. Then it should normally be connected to the same clock as the bus.

The Sn_AXI_ACLK (n = 0, 1) input is only used when the AXI4-Lite interconnect is used. Then it should normally be connected to the same clock as the interconnect.

The Mn_AXIS_ACLK or Sn_AXIS_ACLK (n = 0, 1) are only used when AXI4-Stream is used. Then they should be connected to the corresponding stream clock.

With synchronous operation (C_ASYNC_CLKS = 0), the two clock inputs used must both be connected to the same clock signal in all the cases above.

The FSLn_M_CLK and FSLn_S_CLK (n = 0, 1) are only used with asynchronous operation (C_ASYNC_CLKS = 1) and when FSL is used. Then they should be connected to the corresponding FSL clock.

The FSL_Clk input is only used with synchronous operation (C_ASYNC_CLKS = 0) and when both interfaces use FSL. Then it should be connected to the common FSL clock signal.

Resets

The SPLBn_Rst (n = 0, 1) input is only used when the PLBv46 bus is used. Then it should normally be connected to the same reset as the bus.

The `Sn_AXI_ARESETN` (n = 0, 1) input is only used when the AXI4-Lite interconnect is used. Then it should normally be connected to the same reset as the interconnect.

The `SYS_Rst` input is necessary when both interfaces use AXI4-Stream or FSL, because the streaming interfaces do not have dedicated resets.

The `FSL_Rst` output is generated from the used reset inputs above (depending on the selected interfaces), synchronized to the corresponding clock.

Protocol Description

See the ARM® AMBA® *AXI and ACE Protocol Specification* [Ref 3] for a description of the AXI4-Lite protocol.

See the *ARM AMBA AXI4-Stream Protocol Specification* [Ref 4] for a description of the AXI4-Stream protocol.

See the *IBM 128-Bit Processor Local Bus Architectural Specification (v4.6)* [Ref 2] for a description of the PLBv46 protocol.

See the *LogiCORE™ IP Fast Simplex Link (FSL) V20 Bus* [Ref 5] for a description of the FSL protocol.

SECTION II: VIVADO DESIGN SUITE

Customizing and Generating the Core

Constraining the Core

Customizing and Generating the Core

This chapter includes information about using Xilinx tools to customize and generate the core in the Vivado™ Design Suite environment.

GUI

The Mailbox parameters are divided in two categories: System and Mailbox. When using the Vivado™ IP integrator feature, the addresses are auto-generated.

The configuration screen is shown in Figure 4-1.

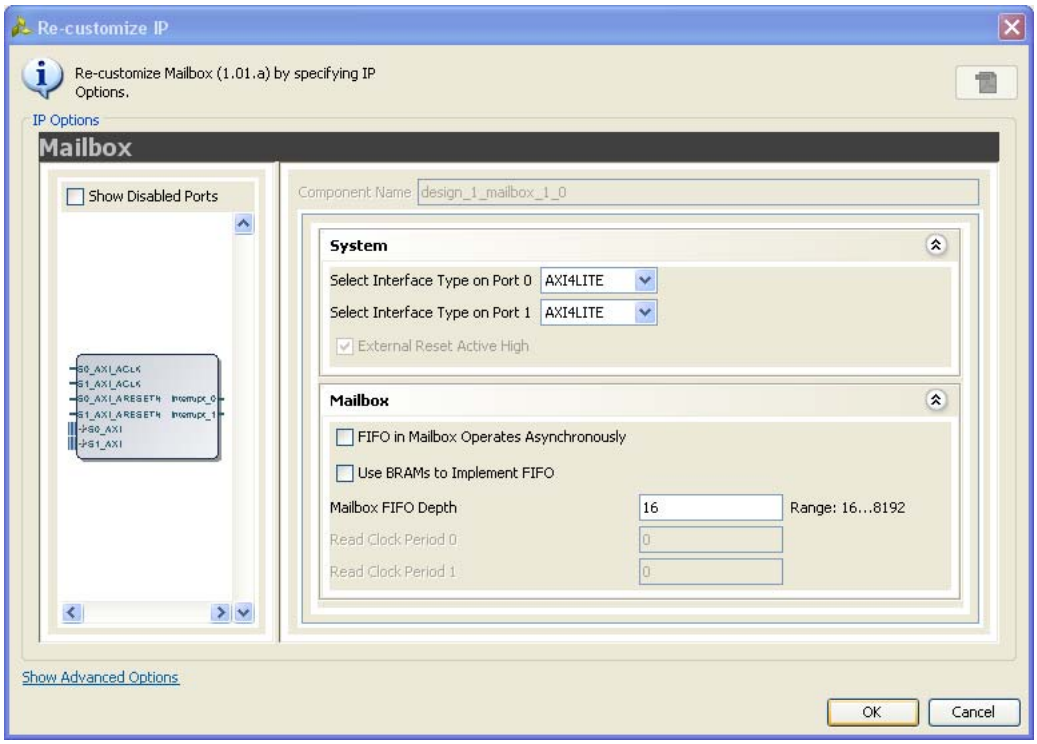

Figure 4-1: **Configuration Screen**

- **Select Interface Type** - Sets the bus interface on both ports to either AXI4-Lite or AXI4-Stream.
- **External Reset Active High** - Sets the reset polarity. Auto-generated by the tool.

- **FIFO in Mailbox Operates Asynchronously** - Enables asynchronous operation, when the clocks of the two interfaces are not identical.
- **Use BRAMs to Implement FIFO** - A mask indicating which address bits the LMB BRAM Interface Controller takes into account when decoding an access.
- **Mailbox FIFO Depth** - Sets the number of words available in the FIFO, from 16 to 8192.
- **Read Clock Period** - Sets the clock period in picoseconds for asynchronous operation.

Parameters

To allow the user to obtain a Mailbox that is uniquely tailored for the system, certain features can be parameterized in the Mailbox design. This allows the user to configure a design that utilizes the resources required by the system only and that operates with the best possible performance. The features that can be parameterized in the Mailbox design are as shown in Table 4-1.

Table 4-1: **Mailbox Design Parameters**

Generic	Feature/Description	Parameter Name	Allowable Values	Default Value	VHDL Type
System Parameter					
G1	Target FPGA family	C_FAMILY	Supported architectures	virtex7	string
G2	Level of external reset	C_EXT_RESET_HIGH	0 or 1	1	integer
Mailbox Parameters					
G20	Specify if interfaces are synchronous or asynchronous	C_ASYNC_CLKS	0 - 1	0	integer
G21	Use BRAMs to implement FIFO	C_IMPL_STYLE	0 - 1	1	integer
G23	Select interface type to be used on port 0: 2 - AXI4-Lite 4 - AXI4-Stream	C_INTERCONNECT_PORT_0	2, 4	0	integer
G24	Select interface type to be used on port 1: 2 - AXI4-Lite 4 - AXI4-Stream	C_INTERCONNECT_PORT_1	2, 4	0	integer
G25	FIFO depth of mailbox	C_MAILBOX_DEPTH	16 - 8192	16	integer
G26	Read Clock period for interface 0 when asynchronous LUTRAM is used (in ps)	C_READ_CLOCK_PERIOD_0	>0 when enabled	0	integer
G27	Read Clock period for interface 1when asynchronous LUTRAM is used (in ps)	C_READ_CLOCK_PERIOD_0	>0 when enabled	0	integer

Constraining the Core

This chapter contains information about constraining the core in the Vivado™ Design Suite environment.

Required Constraints

There are no required constraints for this core.

Device, Package, and Speed Grade Selections

There are no Device, Package or Speed Grade requirements for this core.

Clock Frequencies

There are no specific clock frequency requirements for this core.

Clock Management

The Mailbox can either be fully synchronous with all clocked elements clocked by the same physical clock, or asynchronous with different clocks on the two connected bus interfaces.

With an asynchronous configuration, the parameter C_ASYNC_CLKS (FIFO in Mailbox Operates Asynchronously) must be set manually, as well as the read clock period in picoseconds for each bus interface using the two parameters C_READ_CLOCK_PERIOD_0 (Read Clock Period 0) and C_READ_CLOCK_PERIOD_1 (Read Clock Period 0).

To operate properly when connected to MicroBlaze™, the corresponding bus interface clock must be the same as the MicroBlaze `Clk`.

Clock Placement

There are no specific Clock placement requirements for this core.

Banking

There are no specific Banking rules for this core.

Transceiver Placement

There are no Transceiver Placement requirements for this core.

I/O Standard and Placement

There are no specific I/O standards and placement requirements for this core.

SECTION III: ISE DESIGN SUITE

Customizing and Generating the Core

Constraining the Core

Customizing and Generating the Core

This chapter includes information about using Xilinx tools to customize and generate the core in the ISE® Design Suite environment.

The Mailbox parameters are divided in two categories: User and System. The User configuration tab is shown in Figure 6-1.

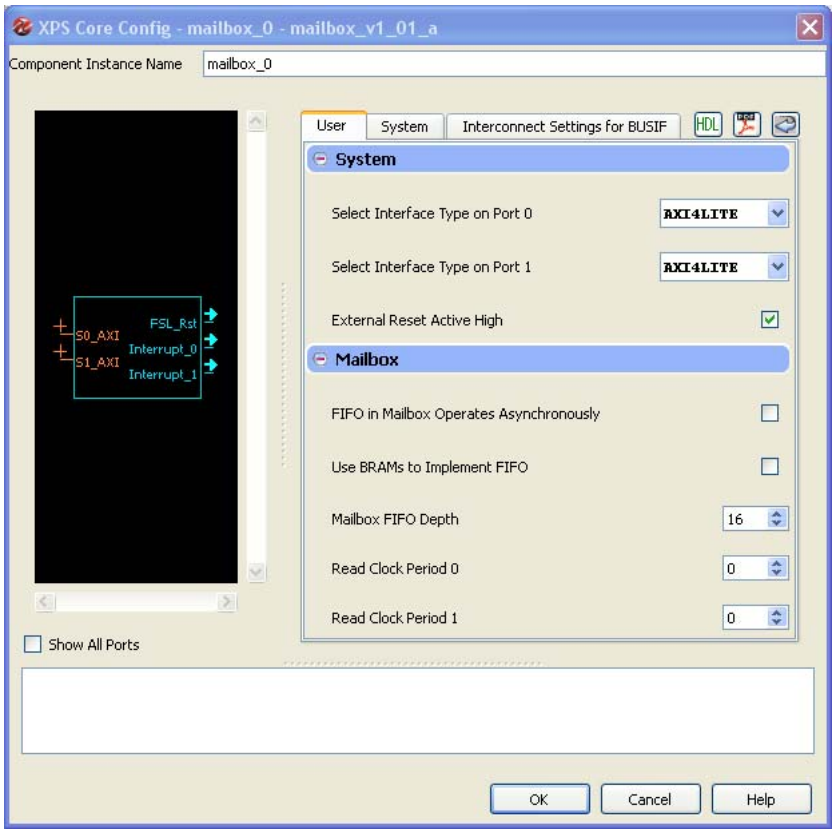

Figure 6-1: **User Configuration Tab**

- **Select Interface Type** - Sets the bus interface on both ports to either AXI4-Lite, PLBv46, AXI4-Stream, or FSL.
- **External Reset Active High** - Sets the reset polarity.
- **FIFO in Mailbox Operates Asynchronously** - Enables asynchronous operation, when the clocks of the two interfaces are not identical.

- **Use BRAMs to Implement FIFO** - A mask indicating which address bits the LMB BRAM Interface Controller takes into account when decoding an access.
- **Mailbox FIFO Depth** - Sets the number of words available in the FIFO, from 16 to 8192.
- **Read Clock Period** - Sets the clock period in picoseconds for asynchronous operation.

The System configuration tab is shown in Figure 6-1.

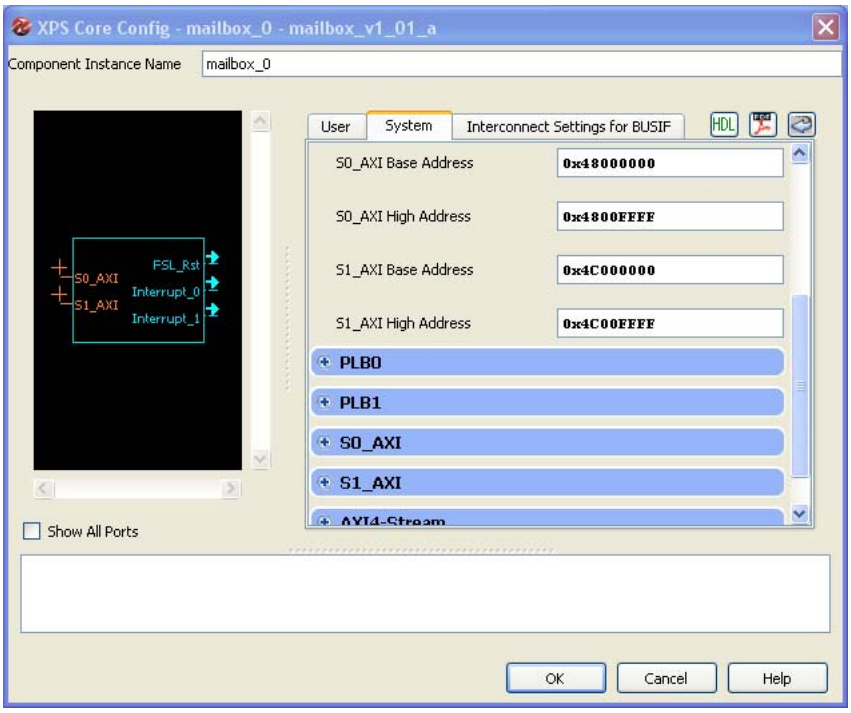

Figure 6-2: **System Configuration Tab**

- **Base Address** - The base address of each non-stream bus interface in use, if any.
- **High Address** - The high address of each non-stream bus interface in use, if any.

Parameters

To allow the user to obtain a Mailbox that is uniquely tailored for the system, certain features can be parameterized in the Mailbox design. This allows the user to configure a design that utilizes the resources required by the system only and that operates with the best possible performance. The features that can be parameterized in the Mailbox design are as shown in Table 6-1. The interface related generics, G3 through G19, are separately configured for each interface.

Table 6-1: **Mailbox Design Parameters**

Generic	Feature/Description	Parameter Name	Allowable Values	Default Value	VHDL Type
System Parameter					
G1	Target FPGA family	C_FAMILY	Supported architectures	virtex6	string
G2	Level of external reset	C_EXT_RESET_HIGH	0 or 1	1	integer
PLB Parameters					
G3	PLB Base Address	C_SPLB<x>_BASEADDR	Valid Address(1)	None(2)	std_logic_vector
G4	PLB High Address	C_SPLB<x>_HIGHADDR	Valid Address(3)	None(2)	std_logic_vector
G5	PLB least significant address bus width	C_SPLB<x>_AWIDTH	32	32	integer
G6	PLB data width	C_SPLB<x>_DWIDTH	32, 64, 128	32	integer
G7	Selects point-to-point or shared bus topology	C_SPLB<x>_P2P	0 = Shared Bus Topology 1 = Point-to-Point Bus Topology(4)	0	integer
G8	PLB Master ID Bus Width	C_SPLB<x>_MID_WIDTH	$\log_2$(C_SPLB_NUM_MASTERS) with a minimum value of 1	1	integer
G9	Number of PLB Masters	C_SPLB<x>_NUM_MASTERS	1 - 16	1	integer
G10	Support Bursts	C_SPLB<x>_SUPPORT_BURSTS	0	0	integer
G11	Width of the Slave Data Bus	C_SPLB_NATIVE_DWIDTH	32	32	integer
G12	Frequency of PLB interface	C_SPLB<x>_CLK_FREQ_HZ	integer	100_000_000	integer
AXI4-Lite Parameters					
G13	AXI Base Address	C_S<x>_AXI_BASEADDR	Valid Address(1)	None(2)	std_logic_vector
G14	AXI High Address	C_S<x>_AXI_HIGHADDR	Valid Address(3)	None(2)	std_logic_vector
G15	AXI address bus width	C_S<x>_AXI_ADDR_WIDTH	32	32	integer
G16	AXI data bus width	C_S<x>_AXI_DATA_WIDTH	32	32	integer
G17	AXI interface type	C_S<x>_AXI_PROTOCOL	AXI4LITE	AXI4-Lite	string
AXI4-Stream Parameters					
G18	AXI data bus width	C_S<x>_AXIS_DATA_WIDTH	32	32	integer
G19	AXI data bus width	C_M<x>_AXIS_DATA_WIDTH	32	32	integer

Table 6-1: **Mailbox Design Parameters *(Cont'd)***

Generic	Feature/Description	Parameter Name	Allowable Values	Default Value	VHDL Type
Mailbox Parameters					
G20	Specify if interfaces are synchronous or asynchronous	C_ASYNC_CLKS	0 - 1	0	Integer
G21	Use BRAMs to implement FIFO	C_IMPL_STYLE	0 - 1	1	Integer
G22	FSL bus width	C_FSL_DWIDTH	32	32	Integer
G23	Select interface type to be used on port 0: 1 - PLBv46 2 - AXI4-Lite 3 - FSL 4 - AXI4-Stream	C_INTERCONNECT_PORT_0	1 - 4	0	Integer
G24	Select interface type to be used on port 1: 1 - PLBv46 2 - AXI4-Lite 3 - FSL 4 - AXI4-Stream	C_INTERCONNECT_PORT_1	1 - 4	0	Integer
G25	FIFO depth of mailbox	C_MAILBOX_DEPTH	16 - 8192	16	Integer
G26	Read Clock period for interface 0 when asynchronous LUTRAM is used (in ps)	C_READ_CLOCK_PERIOD_0	> 0 when enabled	0	Integer
G27	Read Clock period for interface 1when asynchronous LUTRAM is used (in ps)	C_READ_CLOCK_PERIOD_0	> 0 when enabled	0	Integer

Notes:

1. The user must set the values. The C_<interface>_BASEADDR must be a multiple of the range, where the range is C_<interface>_HIGHADDR - C_<interface>_BASEADDR + 1.
2. No default value is specified to ensure that the actual value is set, that is, if the value is not set, a compiler error is generated.
3. C_<interface>_HIGHADDR - C_<interface>_BASEADDR must be a power of 2 greater than equal to C_<interface>_BASEADDR + 0xFF.
4. Value of 1 is not supported in this core.

Parameter - Port Dependencies

The dependencies between the Mailbox core design parameters and I/O signals are described in Table 6-2. In addition, when certain features are deselected, the related logic is no longer a part of the design. The unused input and output signals are set to a specified value.

Table 6-2: **Mailbox Parameter-Port Dependencies**

Generic or Port	Name	Affects	Depends	Relationship Description
Design Parameters				
G15	C_S<x>_AXI_ADDR_WIDTH	P45, P55	-	Defines the width of the ports
G16	C_S<x>_AXI_DATA_WIDTH	P48, P49, P58	-	Defines the width of the ports
G18	C_S<x>_AXIS_DATA_WIDTH	P64	-	Defines the width of the ports
G19	C_M<x>_AXIS_DATA_WIDTH	P68	-	Defines the width of the ports
I/O Signals				
P45	S<x>_AXI_ARADDR[C_S<x>_AXI_ADDR_WIDTH -1:0]	-	G15	Port width depends on the generic C_S<x>_AXI_ADDR_WIDTH
P48	S<x>_AXI_WDATA[C_S<x>_AXI_DATA_WIDTH-1:0]	-	G16	Port width depends on the generic C_S<x>_AXI_DATA_WIDTH
P49	S<x>_AXI_WSTB[C_S<x>_AXI_DATA_WIDTH/8-1:0]	-	G16	Port width depends on the generic C_S<x>_AXI_DATA_WIDTH
P55	S<x>_AXI_ARADDR[C_S<x>_AXI_ADDR_WIDTH -1:0]	-	G15	Port width depends on the generic C_S<x>_AXI_ADDR_WIDTH
P58	S<x>_AXI_RDATA[C_S<x>_AXI_DATA_WIDTH -1:0]	-	G16	Port width depends on the generic C_S<x>_AXI_DATA_WIDTH
P64	S<x>_AXIS_TDATA[C_S<x>_AXIS_DATA_WIDTH -1:0]	-	G18	Port width depends on the generic C_S<x>_AXIS_DATA_WIDTH
P68	M<x>_AXIS_TDATA[C_M<x>_AXIS_DATA_WIDTH -1:0]	-	G19	Port width depends on the generic C_M<x>_AXIS_DATA_WIDTH

Constraining the Core

This chapter contains information about constraining the core in the ISE® Design Suite environment.

Required Constraints

There are no required constraints for this core.

Device, Package, and Speed Grade Selections

There are no Device, Package or Speed Grade requirements for this core.

Clock Frequencies

There are no specific clock frequency requirements for this core.

Clock Management

The Mailbox can either be fully synchronous with all clocked elements clocked by the same physical clock, or asynchronous with different clocks on the two connected bus interfaces.

With an asynchronous configuration, the parameter C_ASYNC_CLKS (FIFO in Mailbox Operates Asynchronously) must be set manually, as well as the read clock period in picoseconds for each bus interface using the two parameters C_READ_CLOCK_PERIOD_0 (Read Clock Period 0) and C_READ_CLOCK_PERIOD_1 (Read Clock Period 0).

To operate properly when connected to MicroBlaze™, the corresponding bus interface clock must be the same as the MicroBlaze `Clk`.

Clock Placement

There are no specific Clock placement requirements for this core.

Banking

There are no specific Banking rules for this core.

Transceiver Placement

There are no Transceiver Placement requirements for this core.

I/O Standard and Placement

There are no specific I/O standards and placement requirements for this core.

SECTION IV: APPENDICES

Migrating

Debugging

Application Software Development

Additional Resources

Appendix A

Migrating

This appendix describes migrating from older versions of the IP to the current IP release.

For information on migrating to the Vivado™ Design Suite, see the *Vivado Design Suite Migration Methodology Guide* [Ref 6].

Appendix B

Debugging

This appendix includes details about resources available on the Xilinx Support website and debugging tools. In addition, this appendix provides a step-by-step debugging process to guide you through debugging the Mailbox core.

The following topics are included in this appendix:

Finding Help on Xilinx.com

To help in the design and debug process when using the Mailbox, 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 opening a Technical Support WebCase.

Documentation

This product guide is the main document associated with the Mailbox. 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.

Release Notes

Known issues for all cores, including the Mailbox are described in the IP Release Notes Guide (XTP025).

Contacting Technical Support

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 Support Quick Links.

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.

Debug Tools

The main tool available to address Mailbox design issues is the ChipScope Pro tool.

ChipScope Pro Tool

The ChipScope™ Pro debugging tool inserts logic analyzer, bus analyzer, and virtual I/O cores directly into your design. The ChipScope Pro debugging tool allows you to set trigger conditions to capture application and integrated block port signals in hardware. Captured signals can then be analyzed through the ChipScope Pro logic analyzer tool. For detailed information for using the ChipScope Pro debugging tool, see www.xilinx.com/tools/cspro.htm.

Reference Boards

All Xilinx development boards support Mailbox. These boards can be used to prototype designs and establish that the core can communicate with the system.

Simulation Debug

The simulation debug flow for ModelSim is described below. A similar approach can be used with other simulators.

- Check for the latest supported versions of ModelSim in the Xilinx Design Tools: Release Notes Guide. Is this version being used? If not, update to this version.
- If using Verilog, do you have a mixed mode simulation license? If not, obtain a mixed-mode license.
- Ensure that the proper libraries are compiled and mapped. In Xilinx Platform Studio this is done within the tool using **Edit > Preferences > Simulation**, and in Vivado Design Suite using **Flow > Simulation Settings.**
- Have you associated the intended software program for the MicroBlaze™ processor with the simulation? Use **Project > Select Elf File** in Xilinx Platform Studio to do this. Make sure to regenerate the simulation files with **Simulation > Generate Simulation HDL Files** afterwards. The equivalent command in Vivado™ Design Suite is **Tools > Associate ELF Files**.
- When observing the traffic on the interfaces connected to the Mailbox, see the timing in the relevant specification:
  - For PLBv46, see the *IBM CoreConnect 128-Bit Processor Local Bus, Architectural Specification (v4.6)* [Ref 2].
  - For AXI4-Lite, see the *AMBA® AXI and ACE Protocol Specification* [Ref 3].
  - For AXI4-Stream, see the *AMBA 4 AXI4-Stream Protocol Specification* [Ref 4].
  - For FSL, see the *LogiCORE™ IP Fast Simplex Link (FSL) V20 Bus (v2.11f)* [Ref 5].

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 ChipScope debugging tool is a valuable resource to use in hardware debug. The signal names mentioned in the following individual sections can be probed using the ChipScope debugging tool for debugging the specific problems.

Many of these common issues can also be applied to debugging design simulations.

General Checks

Ensure that all the timing constraints for the core were properly incorporated from the example design 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.

Interface Debug

AXI4-Lite Interfaces

Read from a register that does not have all 0s as a default to verify that the interface is functional. Output `Sn_AXI_ARREADY` asserts when the read address is valid, and output `Sn_AXI_RVALID` asserts when the read data/response is valid, where *n* is the interface number (0 or 1). If the interface is unresponsive, ensure that the following conditions are met:

- The `Sn_AXI_ACLK` input is connected and toggling.
- The interface is not being held in reset, and `Sn_AXI_ARESETN` is an active-Low reset.
- The common core clock `FSL_Clk` is toggling, if FSL is used.
- The common core reset is not active, and `SYS_Rst` is an active-High reset.
- If the simulation has been run, verify in simulation and/or a ChipScope debugging tool capture that the waveform is correct for accessing the AXI4-Lite interface.

AXI4-Stream Interfaces

If data is not being transmitted or received, check the following conditions:

- If transmit `Mn_AXIS_TREADY` is stuck Low following the `Mn_AXIS_TVALID` input being asserted, the core cannot send data.
- If the receive `Sn_AXIS_TVALID` is stuck Low, the core is not receiving data.
- Check that the `Mn_AXIS_CLK` and `Sn_AXIS_CLK` inputs are connected and toggling.
- Check that the common core reset is not active, and `SYS_Rst` is an active-High reset.
- Check that the AXI4-Stream waveforms are being followed
- Check core configuration.

PLBv46 Interfaces

Read from a register that does not have all 0s as a default to verify that the interface is functional. Output PLB*n*_PAValid asserts when the read address is valid, and output Sl*n*_rdDAck asserts when the read data/response is valid, where *n* is the interface number (0 or 1). If the interface is unresponsive, ensure that the following conditions are met:

- The SPLB*n*_CLK input is connected and toggling.
- The interface is not being held in reset, and SPLB*n*_Rst is an active-High reset.
- If the simulation has been run, verify in simulation and/or a ChipScope debugging tool capture that the waveform is correct for accessing the PLBv46 interface.

FSL Interfaces

If data is not being transmitted or received, check the following conditions:

- If transmit FSL*n*_S_Exists is stuck Low following the FSL*n*_S_Read input being asserted, the core cannot send data.
- If the receive FSL*n*_M_Full is stuck High, the core is not receiving data.
- Check that the FSL*n*_S_Clk and FSL*n*_M_Clk inputs are connected and toggling.
- Check that the common core resets are not active, and both SYS_Rst and FSL_Rst are active-High resets.
- Check that the FSL waveforms are being followed
- Check core configuration.

Appendix C

Application Software Development

Device Drivers

The Mailbox is supported by the mbox driver, included with Xilinx Software Development Kit.

Appendix D

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. Vivado™ Design Suite user documentation
2. IBM CoreConnect128-Bit Processor Local Bus, Architectural Specification (v4.6)
3. AMBA® AXI and ACE Protocol Specification (ARM IHI 0022D)
4. AMBA® 4 AXI4-Stream Protocol Specification (ARM IHI 0051A)
5. LogiCORE™ IP Fast Simplex Link (FSL) V20 Bus (v2.11f) (DS449)
6. Vivado Design Suite Migration Methodology Guide (UG911)

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
10/16/12	1.0	Initial Xilinx release. This Product Guide is derived from DS776. Vivado support for 2012.3 added.
12/18/12	1.1	Updated for 2012.4/14.4. Document updated with new core version and new Debug Appendix.

Notice of Disclaimer

Automotive Applications Disclaimer

XILINX PRODUCTS ARE NOT DESIGNED OR INTENDED TO BE FAIL-SAFE, OR FOR USE IN ANY APPLICATION REQUIRING FAIL-SAFE PERFORMANCE, SUCH AS APPLICATIONS RELATED TO: (I) THE DEPLOYMENT OF AIRBAGS, (II) CONTROL OF A VEHICLE, UNLESS THERE IS A FAIL-SAFE OR REDUNDANCY FEATURE (WHICH DOES NOT INCLUDE USE OF SOFTWARE IN THE XILINX DEVICE TO IMPLEMENT THE REDUNDANCY) AND A WARNING SIGNAL UPON FAILURE TO THE OPERATOR, OR (III) USES THAT COULD LEAD TO DEATH OR PERSONAL INJURY. CUSTOMER ASSUMES THE SOLE RISK AND LIABILITY OF ANY USE OF XILINX PRODUCTS IN SUCH APPLICATIONS.