LogiCORE IP Mailbox v2.0

Product Guide for Vivado Design Suite

PG114 March 20, 2013

XILINX®

XILINX®

Table of Contents

IP Facts

Chapter 1: Overview

Feature Summary 6
Licensing and Ordering Information 6

Chapter 2: Product Specification

Standards 7
Performance 7
Resource Utilization 9
Port Descriptions 10
Register Space 13

Chapter 3: Designing with the Core

General Design Guidelines 20
Clocking 20
Resets 20
Protocol Description 21

Chapter 4: Customizing and Generating the Core

GUI 22
Parameters 23

Chapter 5: Constraining the Core

Required Constraints 24
Device, Package, and Speed Grade Selections 24
Clock Frequencies 24
Clock Management 24
Clock Placement 25
Banking 25
Transceiver Placement 25
I/O Standard and Placement 25

Appendix A: Migrating

Appendix B: Debugging

Finding Help on Xilinx.com 27
Debug Tools 28
Simulation Debug. 29
Hardware Debug 29
Interface Debug 30

Appendix C: Application Software Development

Device Drivers 31

Appendix D: Additional Resources

Xilinx Resources 32
References 32
Revision History 32
Notice of Disclaimer. 33
Automotive Applications Disclaimer 33

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 AXI4-Lite or AXI4-Stream interfaces. The interfaces are available for connection to any IP that supports them, for example MicroBlaze™.

Features

- Supports AXI4-Lite and AXI4-Stream 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, Virtex®-7, Kintex™-7, Artix™-7
Supported User Interfaces	AXI4-Lite, AXI4-Stream
Resources	See Table 2-3.
Provided with Core	
Design Files	Vivado: RTL
Example Design	Not Provided
Test Bench	Not Provided
Constraints File	Not Provided
Simulation Model	VHDL Behavioral
Supported S/W Driver(2)	mbox
Tested Design Flows(3)	
Design Entry	Vivado™ Design Suite
Simulation	Mentor Graphics Questa® SIM Vivado Simulator
Synthesis	Vivado Synthesis
Support	
Provided by Xilinx @ www.xilinx.com/support	

Notes:

1. For a complete listing of supported devices, see Vivado IP Catalog.
2. Standalone driver details can be found in the SDK directory (<*install_directory*>/doc/usenglish/xilinx_drivers.htm). Linux OS and driver support information is available from //wiki.xilinx.com.
3. For the supported versions of the tools, see the Xilinx Design Tools: Release Notes Guide.

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 AXI4-Stream option has the Mailbox interface connected directly to a master with no bus in between.

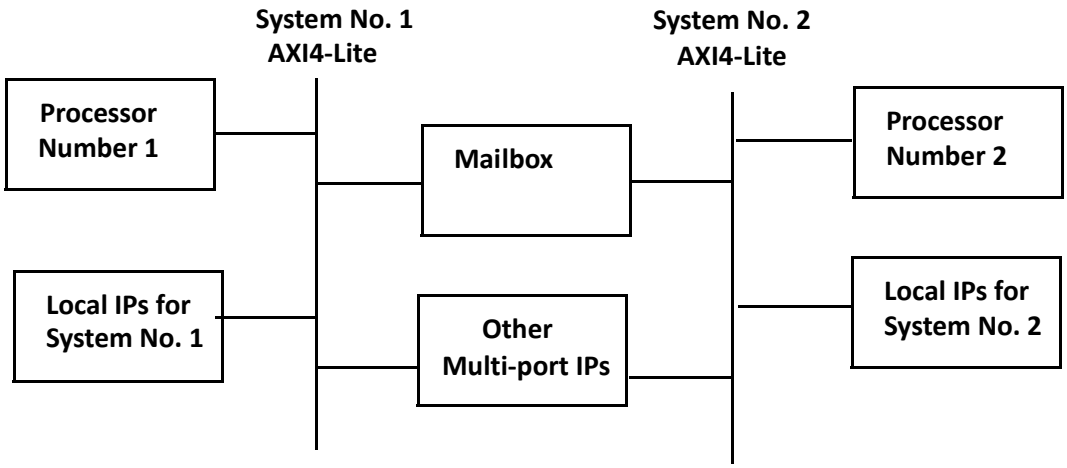

Figure 1-1: **Mailbox in an AXI4-Lite System**

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 or AXI4-Stream interface.

Registers

The Mailbox provides several types of registers, available with the AXI4-Lite bus interface, 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 the AXI4-Stream bus interface, 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 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 2].

The Mailbox adheres to the *ARM AMBA AXI4-Stream Protocol Specification* [Ref 3].

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
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)	
	Minimum	Typical	Minimum	Typical	Minimum	Typical
Synchronous Distributed RAM (C_ASYNC_CLKS = 0, C_IMPL_STYPE = 0):						
AXI4-Lite	3	12	3	12	6	24
AXI4-Stream	1	10	1	10	2	20
Synchronous Block RAM (C_ASYNC_CLKS = 0, C_IMPL_STYPE = 1):						
AXI4-Lite	3	12	4	13	7	25
AXI4-Stream	1	10	1	11	2	21
Asynchronous Distributed RAM (C_ASYNC_CLKS = 1, C_IMPL_STYPE = 0):						
AXI4-Lite	3	12	3	12	10	28
AXI4-Stream	1	10	1	10	2	24
Asynchronous Block RAM (C_ASYNC_CLKS = 1, C_IMPL_STYPE = 1):						
AXI4-Lite	3	12	4	13	11	29
AXI4-Stream	1	10	1	11	2	25

The minimum number only takes into account the effect of the hardware implementation, whereas the typical value also accounts for the typical default software driver overhead.

The throughput denotes the time for one write access followed by one read access. If several writes and reads are performed, they may partly overlap. As the number of accesses increases, this overlap causes the throughput to approach the access latency.

When using AXI4-Stream, software can be optimized by reading or writing several words in sequence, but in this case care must be taken to avoid stalls due to intermediate instructions. The stream instructions must be consecutive to achieve the minimum latency.

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 Kintex-7 as the target device are detailed in Table 2-3.

Table 2-3: **xc7k325t-2-ffg900 FPGA Resource Estimates**

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 AXI4-Lite or AXI4-Stream interface. The signal descriptions are included in three tables:

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

All signals in Table 2-4 through Table 2-6 apply to both interface sides; <x> denotes the interface number, which can be 0 or 1.

Table 2-4: **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.

Table 2-4: **AXI4-Lite I/O Signal Description *(Cont'd)***

Port	Signal Name	Interface	I/O	Initial State	Description
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.
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 2-5: **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 2-6: **Mailbox Common I/O Signal Description**

Port	Signal Name	Interface	I/O	Initial State	Description
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 AXI4-Stream based (C_INTERCONNECT_PORT_<x> = 4). The FSL_Clk is in this case used to clock the core, in all other cases the internal mailbox clock is automatically derived from S<x>_AXI_ACLK.
P83	SYS_Rst	System	I	N/A	External system reset. This signal is only required when both interfaces are configured to be streaming interfaces (AXI4-Stream). If any AXI4-Lite interface is available this signal is optional.
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 2-7 shows all the Mailbox registers and their addresses for AXI4-Lite case. Much of the information can be acquired for the AXI4-Stream case with the use of S<x>_AXIS_TREADY and M<x>_AXIS_TVALID.

Table 2-7: **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 2-9.

Table 2-8: **Write Data Register**

WRDATA	
0	C_FSL_DWIDTH-1

Table 2-9: **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 2-11.

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

RDDATA	
0	C_FSL_DWIDTH-1

Table 2-11: **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 2-13.

Table 2-12: **Status Register**

Reserved		RTA	STA	Full	Empty
0	27	28	29	30	31

Table 2-13: **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 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 2-15.

Table 2-14: **Error Register**

Reserved		Full Error	Empty Error
0	29	30	31

Table 2-15: **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 Mailbox FIFO is 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 Mailbox FIFO is 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 2-17.

Table 2-16: **SIT Register**

	SIT	
0	32-Log2(C_MAILBOX_DEPTH)	31

Table 2-17: **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 2-19.

Table 2-18: **RIT Register**

	RIT	
0	32-Log2(C_MAILBOX_DEPTH)	31

Table 2-19: **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 2-21.

Table 2-20: **IS Register**

Reserved		ERR	RTI	STI
0	28	29	30	31

Table 2-21: **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 2-21: **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 2-23.

Table 2-22: **IE Register**

Reserved		ERR	RTI	STI
0	28	29	30	31

Table 2-23: **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 2-25. 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 2-24: **IP Register**

Reserved		ERR	RTI	STI
0	28	29	30	31

Table 2-25: **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 `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 `FSL_Clk` input is only used with synchronous operation (`C_ASYNC_CLKS` = 0) and when both interfaces use AXI4-Stream. Then it should be connected to the common AXI4-Stream clock signal.

Resets

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, because the streaming interfaces do not have dedicated resets.

Protocol Description

See the ARM® AMBA® *AXI and ACE Protocol Specification* [Ref 2] for a description of the AXI4-Lite protocol.

See the *ARM AMBA AXI4-Stream Protocol Specification* [Ref 3] for a description of the AXI4-Stream protocol.

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.

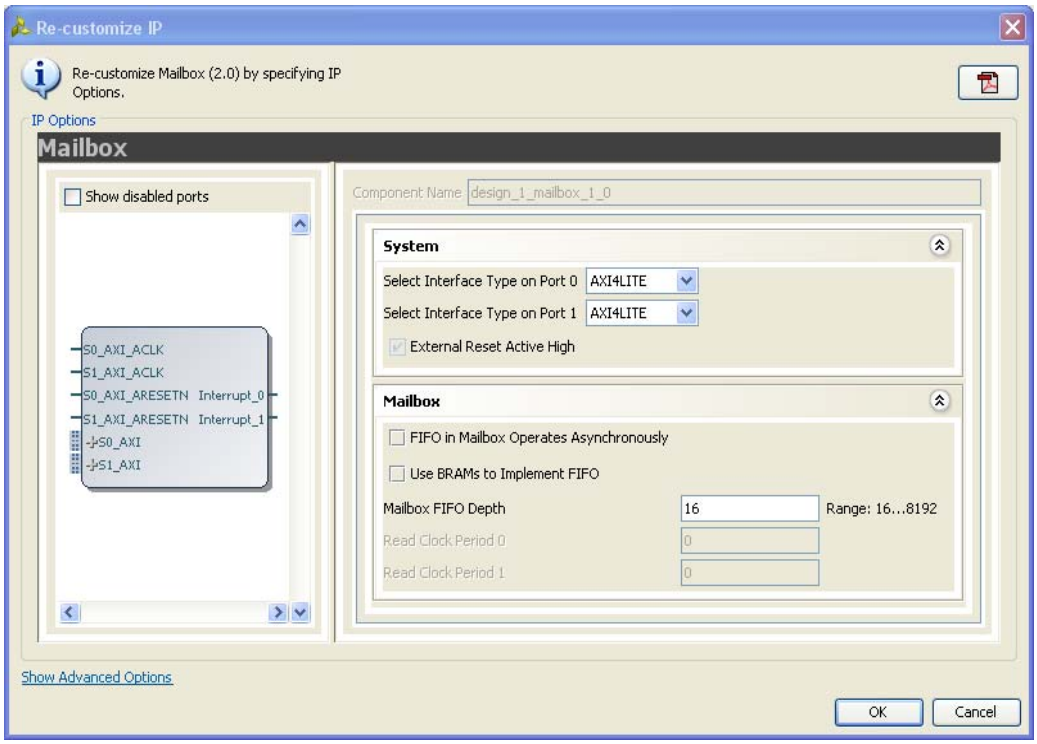

Figure 4-1: **Configuration Screen**

- **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.

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 4].

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
- Debug Tools
- Simulation Debug
- Hardware Debug
- Interface Debug

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.

Answer Records for the Mailbox Core

- AR54415

Contacting Technical Support

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

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 tools available to address Mailbox design issues are the Vivado Lab tools.

Vivado Lab Tools

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

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

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

Reference Boards

All 7-Series 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 Questa® SIM is described below. A similar approach can be used with other simulators.

- Check for the latest supported versions of Questa SIM 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 the Vivado Design Suite **Flow > Simulation Settings** can be used to define the libraries.
- Have you associated the intended software program for the MicroBlaze™ processor with the simulation? Use the command **Tools > Associate ELF Files** in Vivado Design Suite.
- When observing the traffic on the interfaces connected to the Mailbox, see the timing in the relevant specification:
  - For AXI4-Lite, see the *AMBA® AXI and ACE Protocol Specification* [Ref 2].
  - For AXI4-Stream, see the *AMBA 4 AXI4-Stream Protocol Specification* [Ref 3].

Hardware Debug

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

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

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 S*n*_AXI_ARREADY asserts when the read address is valid, and output S*n*_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 S*n*_AXI_ACLK input is connected and toggling.
- The interface is not being held in reset, and S*n*_AXI_ARESETN is an active-Low reset.
- 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 M*n*_AXIS_TREADY is stuck Low following the M*n*_AXIS_TVALID input being asserted, the core cannot send data.
- If the receive S*n*_AXIS_TVALID is stuck Low, the core is not receiving data.
- Check that the M*n*_AXIS_CLK and S*n*_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.

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. AMBA® AXI and ACE Protocol Specification (ARM IHI 0022D)
3. AMBA® AXI4-Stream Protocol Specification (ARM IHI 0051A)
4. Vivado Design Suite Migration Methodology Guide (UG911)
5. Vivado Design Suite User Guide, Designing with IP (UG896)

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
03/20/2013	1.0	This Product Guide replaces PG088. There are no documentation changes for this release.

Notice of Disclaimer

Automotive Applications Disclaimer