Xilinx Standalone Library Documentation

XilMailbox Library v1.4

UG1367 (v2021.2) October 13, 2021

Table of Contents

Chapter 1: XilMailbox API Reference **3**
Functions 4
Enumerations 12

Chapter 2: Data Structure Index **13**
XMailbox 13
XMailbox_Agent 14

Appendix A: Additional Resources and Legal Notices **15**
Xilinx Resources 15
Documentation Navigator and Design Hubs 15
Please Read: Important Legal Notices 16

Chapter 1

XilMailbox API Reference

This file contains the definitions for Zynq® UltraScale+™ MPSoC and Versal™ inter-processor interrupt (IPI) implementation. This file contains the definitions for Xilinx® mailbox library top level functions. The XilMailbox library provides the top-level hooks for sending or receiving an IPI message using the Zynq UltraScale+ MPSoC IPI hardware.

For a full description of IPI features, please see the hardware specifications. This library supports the following features:

Table 1: **Quick Function Reference**

Type	Name	Arguments
u32	XIpiPs_Init	`XMailbox` * InstancePtr u8 DeviceId
u32	XIpiPs_Send	`XMailbox` * InstancePtr u8 Is_Blocking
u32	XIpiPs_SendData	`XMailbox` * InstancePtr void * MsgBufferPtr u32 MsgLen u8 BufferType u8 Is_Blocking
u32	XIpiPs_PollforDone	`XMailbox` * InstancePtr
u32	XIpiPs_RecvData	`XMailbox` * InstancePtr void * MsgBufferPtr u32 MsgLen u8 BufferType
XStatus	XIpiPs_RegisterIrq	XScuGic * IntcInstancePtr `XMailbox` * InstancePtr u32 IpiIntrId
void	XIpiPs_ErrorIntrHandler	void * XMailboxPtr

Table 1: **Quick Function Reference** *(cont'd)*

Type	Name	Arguments
void	XIpiPs_IntrHandler	void * XMailboxPtr
u32	XMailbox_Initialize	XMailbox * InstancePtr u8 DeviceId
u32	XMailbox_Send	XMailbox * InstancePtr u32 RemoteId u8 Is_Blocking
u32	XMailbox_SendData	XMailbox * InstancePtr u32 RemoteId void * BufferPtr u32 MsgLen u8 BufferType u8 Is_Blocking
u32	XMailbox_Recv	XMailbox * InstancePtr u32 SourceId void * BufferPtr u32 MsgLen u8 BufferType
s32	XMailbox_SetCallBack	XMailbox * InstancePtr XMailbox_Handler HandlerType void * CallBackFuncPtr void * CallBackRefPtr

Functions

XIpiPs_Init

Initialize the ZynqMP Mailbox Instance.

Prototype

```
u32 XIpiPs_Init(XMailbox *InstancePtr, u8 DeviceId);
```

Parameters

The following table lists the `XIpiPs_Init` function arguments.

Table 2: **XIpiPs_Init Arguments**

Type	Name	Description
`XMailbox` *	InstancePtr	is a pointer to the instance to be worked on
u8	DeviceId	is the IPI Instance to be worked on

Returns

XST_SUCCESS if initialization was successful XST_FAILURE in case of failure

XIpiPs_Send

This function triggers an IPI to a destnation CPU.

Prototype

```
u32 XIpiPs_Send(XMailbox *InstancePtr, u8 Is_Blocking);
```

Parameters

The following table lists the `XIpiPs_Send` function arguments.

Table 3: **XIpiPs_Send Arguments**

Type	Name	Description
`XMailbox` *	InstancePtr	Pointer to the `XMailbox` instance.
u8	Is_Blocking	if set trigger the notification in blocking mode

Returns

XST_SUCCESS in case of success XST_FAILURE in case of failure

XIpiPs_SendData

This function sends an IPI message to a destnation CPU.

Prototype

```
u32 XIpiPs_SendData(XMailbox *InstancePtr, void *MsgBufferPtr, u32 MsgLen,
u8 BufferType, u8 Is_Blocking);
```

Parameters

The following table lists the `XIpiPs_SendData` function arguments.

Table 4: **XIpiPs_SendData Arguments**

Type	Name	Description
`XMailbox` *	InstancePtr	Pointer to the `XMailbox` instance
void *	MsgBufferPtr	is the pointer to Buffer which contains the message to be sent
u32	MsgLen	is the length of the buffer/message
u8	BufferType	is the type of buffer
u8	Is_Blocking	if set trigger the notification in blocking mode

Returns

XST_SUCCESS in case of success XST_FAILURE in case of failure

XIpiPs_PollforDone

Poll for an acknowledgement using Observation Register.

Prototype

```
u32 XIpiPs_PollforDone(XMailbox *InstancePtr);
```

Parameters

The following table lists the `XIpiPs_PollforDone` function arguments.

Table 5: **XIpiPs_PollforDone Arguments**

Type	Name	Description
`XMailbox` *	InstancePtr	Pointer to the `XMailbox` instance

Returns

XST_SUCCESS in case of success XST_FAILURE in case of failure

XIpiPs_RecvData

This function reads an IPI message.

Prototype

```
u32 XIpiPs_RecvData(XMailbox *InstancePtr, void *MsgBufferPtr, u32 MsgLen,
u8 BufferType);
```

Parameters

The following table lists the `XIpiPs_RecvData` function arguments.

Table 6: **XIpiPs_RecvData Arguments**

Type	Name	Description
`XMailbox` *	InstancePtr	Pointer to the `XMailbox` instance
void *	MsgBufferPtr	is the pointer to Buffer to which the read message needs to be stored
u32	MsgLen	is the length of the buffer/message
u8	BufferType	is the type of buffer

Returns

- XST_SUCCESS if successful
- XST_FAILURE if unsuccessful

XIpiPs_RegisterIrq

This function registers an irq.

Prototype

```
XStatus XIpiPs_RegisterIrq(XScuGic *IntcInstancePtr, XMailbox *InstancePtr,
u32 IpiIntrId);
```

Parameters

The following table lists the `XIpiPs_RegisterIrq` function arguments.

Table 7: **XIpiPs_RegisterIrq Arguments**

Type	Name	Description
XScuGic *	IntcInstancePtr	Pointer to the scugic instance
`XMailbox` *	InstancePtr	Pointer to the `XMailbox` instance
u32	IpiIntrId	is the interrupt id of the IPI

Returns

- XST_SUCCESS if successful
- XST_FAILURE if unsuccessful

XIpiPs_ErrorIntrHandler

This function implements the interrupt handler for errors.

Prototype

```
void XIpiPs_ErrorIntrHandler(void *XMailboxPtr);
```

Parameters

The following table lists the `XIpiPs_ErrorIntrHandler` function arguments.

Table 8: **XIpiPs_ErrorIntrHandler Arguments**

Type	Name	Description
void *	XMailboxPtr	Pointer to the `XMailbox` instance

Returns

None

XIpiPs_IntrHandler

This function implements the interrupt handler.

Prototype

```
void XIpiPs_IntrHandler(void *XMailboxPtr);
```

Parameters

The following table lists the `XIpiPs_IntrHandler` function arguments.

Table 9: **XIpiPs_IntrHandler Arguments**

Type	Name	Description
void *	XMailboxPtr	Pointer to the `XMailbox` instance

Returns

None

XMailbox_Initialize

Initialize the `XMailbox` Instance.

Prototype

```
u32 XMailbox_Initialize(XMailbox *InstancePtr, u8 DeviceId);
```

Parameters

The following table lists the `XMailbox_Initialize` function arguments.

Table 10: **XMailbox_Initialize Arguments**

Type	Name	Description
`XMailbox` *	InstancePtr	is a pointer to the instance to be worked on
u8	DeviceId	is the IPI Instance to be worked on

Returns

XST_SUCCESS if initialization was successful XST_FAILURE in case of failure

XMailbox_Send

This function triggers an IPI to a destination CPU.

Prototype

```
u32 XMailbox_Send(XMailbox *InstancePtr, u32 RemoteId, u8 Is_Blocking);
```

Parameters

The following table lists the `XMailbox_Send` function arguments.

Table 11: **XMailbox_Send Arguments**

Type	Name	Description
`XMailbox` *	InstancePtr	Pointer to the `XMailbox` instance
u32	RemoteId	is the Mask of the CPU to which IPI is to be triggered
u8	Is_Blocking	if set trigger the notification in blocking mode

Returns

- XST_SUCCESS if successful
- XST_FAILURE if unsuccessful

XMailbox_SendData

This function sends an IPI message to a destination CPU.

Prototype

```
u32 XMailbox_SendData(XMailbox *InstancePtr, u32 RemoteId, void *BufferPtr,
u32 MsgLen, u8 BufferType, u8 Is_Blocking);
```

Parameters

The following table lists the `XMailbox_SendData` function arguments.

Table 12: **XMailbox_SendData Arguments**

Type	Name	Description
`XMailbox` *	InstancePtr	Pointer to the `XMailbox` instance
u32	RemoteId	is the Mask of the CPU to which IPI is to be triggered
void *	BufferPtr	is the pointer to Buffer which contains the message to be sent
u32	MsgLen	is the length of the buffer/message
u8	BufferType	is the type of buffer (XILMBOX_MSG_TYPE_REQ (OR) XILMBOX_MSG_TYPE_RESP)
u8	Is_Blocking	if set trigger the notification in blocking mode

Returns

- XST_SUCCESS if successful
- XST_FAILURE if unsuccessful

XMailbox_Recv

This function reads an IPI message.

Prototype

```
u32 XMailbox_Recv(XMailbox *InstancePtr, u32 SourceId, void *BufferPtr, u32
MsgLen, u8 BufferType);
```

Parameters

The following table lists the `XMailbox_Recv` function arguments.

Table 13: **XMailbox_Recv Arguments**

Type	Name	Description
`XMailbox` *	InstancePtr	Pointer to the `XMailbox` instance
u32	SourceId	is the Mask for the CPU which has sent the message
void *	BufferPtr	is the pointer to Buffer to which the read message needs to be stored
u32	MsgLen	is the length of the buffer/message

Table 13: **XMailbox_Recv Arguments** *(cont'd)*

Type	Name	Description
u8	BufferType	is the type of buffer (XILMBOX_MSG_TYPE_REQ or XILMBOX_MSG_TYPE_RESP)

Returns

- XST_SUCCESS if successful
- XST_FAILURE if unsuccessful

XMailbox_SetCallBack

This routine installs an asynchronous callback function for the given HandlerType.

Prototype

```
s32 XMailbox_SetCallBack(XMailbox *InstancePtr, XMailbox_Handler
HandlerType, void *CallBackFuncPtr, void *CallBackRefPtr);
```

Parameters

The following table lists the `XMailbox_SetCallBack` function arguments.

Table 14: **XMailbox_SetCallBack Arguments**

Type	Name	Description
`XMailbox` *	InstancePtr	is a pointer to the `XMailbox` instance.
`XMailbox_Handler`	HandlerType	specifies which callback is to be attached.
void *	CallBackFuncPtr	is the address of the callback function.
void *	CallBackRefPtr	is a user data item that will be passed to the callback function when it is invoked.

Returns

- XST_SUCCESS when handler is installed.
- XST_FAILURE when HandlerType is invalid.

Enumerations

Enumeration XMailbox_Handler

This typedef contains XMAILBOX Handler Types.

Table 15: **Enumeration XMailbox_Handler Values**

Value	Description
XMAILBOX_RECV_HANDLER	For Recv Handler.
XMAILBOX_ERROR_HANDLER	For Error Handler.

Data Structure Index

The following is a list of data structures:

- XMailbox
- XMailbox_Agent

XMailbox

Data structure used to refer XilMailbox.

Declaration

```
typedef struct
{
  u32(* XMbox_IPI_Send)(struct XMboxTag *InstancePtr, u8 Is_Blocking),
  u32(* XMbox_IPI_SendData)(struct XMboxTag *InstancePtr, void *BufferPtr,
u32 MsgLen, u8 BufferType, u8 Is_Blocking),
  u32(* XMbox_IPI_Recv)(struct XMboxTag *InstancePtr, void *BufferPtr, u32
MsgLen, u8 BufferType),
  XMailbox_RecvHandler RecvHandler,
  XMailbox_ErrorHandler ErrorHandler,
  void * ErrorRefPtr,
  void * RecvRefPtr,
  XMailbox_Agent Agent
} XMailbox;
```

Table 16: **Structure XMailbox member description**

Member	Description
XMbox_IPI_Send	Triggers an IPI to a destination CPU.
XMbox_IPI_SendData	Sends an IPI message to a destination CPU.
XMbox_IPI_Recv	Reads an IPI message.
RecvHandler	Recieve handler.
ErrorHandler	Callback for RX IPI event.
ErrorRefPtr	To be passed to the error interrupt callback.
RecvRefPtr	To be passed to the receive interrupt callback.
Agent	Agent to store IPI channel information.

XMailbox_Agent

Data structure used to refer Xilmailbox agents.

Declaration

```
typedef struct
{
  XIpiPsu IpiInst,
  XScuGic GicInst,
  u32 SourceId,
  u32 RemoteId
} XMailbox_Agent;
```

Table 17: **Structure XMailbox_Agent member description**

Member	Description
IpiInst	IPI instance.
GicInst	Interrupt instance.
SourceId	Source ID.
RemoteId	Remote ID.

Additional Resources and Legal Notices

Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see Xilinx Support.

Documentation Navigator and Design Hubs

Xilinx® Documentation Navigator (DocNav) provides access to Xilinx documents, videos, and support resources, which you can filter and search to find information. To open DocNav:

- From the Vivado® IDE, select **Help → Documentation and Tutorials**.
- On Windows, select **Start → All Programs → Xilinx Design Tools → DocNav**.
- At the Linux command prompt, enter `docnav`.

Xilinx Design Hubs provide links to documentation organized by design tasks and other topics, which you can use to learn key concepts and address frequently asked questions. To access the Design Hubs:

- In DocNav, click the **Design Hubs View** tab.
- On the Xilinx website, see the Design Hubs page.

***Note*:** For more information on DocNav, see the Documentation Navigator page on the Xilinx website.

Please Read: Important Legal Notices

AUTOMOTIVE APPLICATIONS DISCLAIMER

Copyright