Xilinx Standalone Library Documentation

XilMailbox Library v1.1

UG1367 (2019.2) October 30, 2019

XILINX

Table of Contents

Chapter 1: XilMailbox

Overview 3

Data Structure Documentation 4

struct XMailbox 4

Enumeration Type Documentation 5

XMailbox_Handler 5

Function Documentation 5

XMailbox_Send 5

XMailbox_SendData 6

XMailbox_Recv 6

XMailbox_SetCallBack 7

XMailbox_Initialize 7

Appendix A: Additional Resources and Legal Notices

Chapter 1

XilMailbox

Overview

The XilMailbox library provides the top-level hooks for sending or receiving an inter-processor interrupt (IPI) message using the Zynq® UltraScale+™ MPSoC IPI hardware.

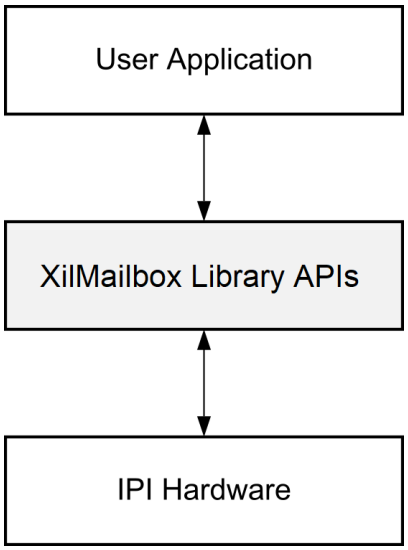

Figure 1.1: Overview

For more details on the IPI interrupts, see the Zynq UltraScale+ MPSoC Technical Reference Manual (UG1085). This library supports the following features:

- Triggering an IPI to a remote agent.
- Sending an IPI message to a remote agent.
- Callbacks for error and recv IPI events.
- Reading an IPI message.

Software Initialization

1. XMailbox_Initialize() function initializes a library instance for the given IPI channel.
2. XMailbox_Send() function triggers an IPI to a remote agent.
3. XMailbox_SendData() function sends an IPI message to a remote agent, message type should be either XILMBOX_MSG_TYPE_REQ (OR) XILMBOX_MSG_TYPE_RESP.
4. XMailbox_Recv() function reads an IPI message from a specified source agent, message type should be either XILMBOX_MSG_TYPE_REQ (OR) XILMBOX_MSG_TYPE_RESP.
5. XMailbox_SetCallBack() using this function user can register call backs for receive and error events.

Data Structures

- struct XMailbox

Enumerations

- enum XMailbox_Handler {
  XMAILBOX_RECV_HANDLER,
  XMAILBOX_ERROR_HANDLER }

Functions

- 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)
- u32 XMailbox_Initialize (XMailbox *InstancePtr, u8 DeviceId)

Data Structure Documentation

struct XMailbox

XMailbox structure.

Parameters

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	Callback for receive IPI event
ErrorHandler	Callback for error event
ErroRef	To be passed to the error interrupt callback
RecvRef	To be passed to the receive interrupt callback.
Agent	Used to store IPI Channel information.

Enumeration Type Documentation

enum XMailbox_Handler

This typedef contains XMAILBOX Handler Types.

Enumerator

XMAILBOX_RECV_HANDLER For Recv Handler.

XMAILBOX_ERROR_HANDLER For Error Handler.

Function Documentation

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

This function triggers an IPI to a destination CPU.

Parameters

InstancePtr	Pointer to the XMailbox instance
RemoteId	Mask of the CPU to which IPI is to be triggered
Is_Blocking	If set, triggers notification in the blocking mode

Returns

- XST_SUCCESS if successful
- XST_FAILURE if unsuccessful

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

This function sends an IPI message to a destination CPU.

Parameters

InstancePtr	Pointer to the XMailbox instance
RemoteId	Mask of the CPU to which IPI is to be triggered
BufferPtr	Pointer to Buffer which contains the message to be sent
MsgLen	Length of the buffer/message
BufferType	Type of buffer (XILMBOX_MSG_TYPE_REQ (OR) XILMBOX_MSG_TYPE_RESP)
Is_Blocking	If set, triggers the notification in blocking mode

Returns

- XST_SUCCESS if successful
- XST_FAILURE if unsuccessful

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

This function reads an IPI message.

Parameters

InstancePtr	Pointer to the XMailbox instance
SourceId	Mask for the CPU which has sent the message
BufferPtr	Pointer to Buffer to which the read message needs to be stored
MsgLen	Length of the buffer/message
BufferType	Type of buffer (XILMBOX_MSG_TYPE_REQ or XILMBOX_MSG_TYPE_RESP)

Returns

- XST_SUCCESS if successful
- XST_FAILURE if unsuccessful

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

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

HandlerType	Callback Function Type
XMAILBOX_RECV_HANDLER	Recv handler
XMAILBOX_ERROR_HANDLER	Error handler

Parameters

InstancePtr	Pointer to the XMailbox instance
HandlerType	Specifies which callback is to be attached
CallBackFunc	Address of the callback function
CallBackRef	User data item that will be passed to the callback function when it is invoked

Returns

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

Note

Invoking this function for a handler that already has been installed replaces it with the new handler.

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

Initialize the XMailbox Instance.

Parameters

InstancePtr	is a pointer to the instance to be worked on
DeviceId	is the IPI Instance to be worked on

Returns

XST_SUCCESS if initialization was successful XST_FAILURE in case of failure

Appendix A

Additional Resources and Legal Notices

Xilinx Resources

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

Solution Centers

See the Xilinx Solution Centers for support on devices, software tools, and intellectual property at all stages of the design cycle. Topics include design assistance, advisories, and troubleshooting tips.

Please Read: Important Legal Notices

The information disclosed to you hereunder (the "Materials") is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available "AS IS" and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of Xilinx's limited warranty, please refer to Xilinx's Terms of Sale which can be viewed at http://www.xilinx.com/legal.htm#tos; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in such critical applications, please refer to Xilinx's Terms of Sale which can be viewed at http://www.xilinx.com/legal.htm#tos.

Automotive Applications Disclaimer

AUTOMOTIVE PRODUCTS (IDENTIFIED AS ”XA” IN THE PART NUMBER) ARE NOT WARRANTED FOR USE IN THE DEPLOYMENT OF AIRBAGS OR FOR USE IN APPLICATIONS THAT AFFECT CONTROL OF A VEHICLE (”SAFETY APPLICATION”) UNLESS THERE IS A SAFETY CONCEPT OR REDUNDANCY FEATURE CONSISTENT WITH THE ISO 26262 AUTOMOTIVE SAFETY STANDARD (”SAFETY DESIGN”). CUSTOMER SHALL, PRIOR TO USING OR DISTRIBUTING ANY SYSTEMS THAT INCORPORATE PRODUCTS, THOROUGHLY TEST SUCH SYSTEMS FOR SAFETY PURPOSES. USE OF PRODUCTS IN A SAFETY APPLICATION WITHOUT A SAFETY DESIGN IS FULLY AT THE RISK OF CUSTOMER, SUBJECT ONLY TO APPLICABLE LAWS AND REGULATIONS GOVERNING LIMITATIONS ON PRODUCT LIABILITY.

© Copyright 2019 Xilinx, Inc. Xilinx, Inc. Xilinx, the Xilinx logo, Alveo, Artix, ISE, Kintex, Spartan, Versal, Virtex, Vivado, Zynq, and other designated brands included herein are trademarks of Xilinx in the United States and other countries. OpenCL and the OpenCL logo are trademarks of Apple Inc. used by permission by Khronos. HDMI, HDMI logo, and High-Definition Multimedia Interface are trademarks of HDMI Licensing LLC. AMBA, AMBA Designer, Arm, ARM1176JZ-S, CoreSight, Cortex, PrimeCell, Mali, and MPCore are trademarks of Arm Limited in the EU and other countries. All other trademarks are the property of their respective owners.