Xilinx Standalone Library Documentation

XilFPGA Library v4.1

UG1229 (2018.2) June 6, 2018

XILINX

XILINX

Table of Contents

XILINX

Chapter 1

Overview

The XilFPGA library provides an interface to the Linux or bare-metal users for configuring the programmable logic (PL) over PCAP from PS.
The library is designed for Zynq® UltraScale+™ MPSoC to run on top of Xilinx standalone BSPs. It is tested for A53, R5 and MicroBlaze. In the most common use case, we expect users to run this library on PMU MicroBlaze with PMUFW to serve requests from Linux for Bitstream programming. In this release, the XilFPGA library Supports full, Authenticated and encrypted Bitstream download.

XilFPGA library Interface modules

XilFPGA library uses the below major components to configure the PL through PS.

Processor Configuration Access Port (PCAP)

The processor configuration access port (PCAP) is used to configure the programmable logic (PL) through the PS.

CSU DMA driver

The CSU DMA driver is used to transfer the actual Bit stream file for the PS to PL after PCAP initialization.

Xilsecure_library

The LibXilSecure library provides APIs to access secure hardware on the Zynq® UltraScale+™ MPSoC devices.

Note

- The current version of library supports only Zynq® UltraScale+™ MPSoC devices.
- The XilFPGA library is capable of loading only .bin format files into PL. The library will not support the other file formats.

Design Summary

XilFPGA library acts as a bridge between the user application and the PL device. It provides the required functionality to the user application for configuring the PL Device with the required bit-stream. The following figure illustrates an implementation where the XilFPGA library needs the CSU DMA driver APIs to transfer the bit-stream from the DDR to the PL region. The XilFPGA library also needs the XilSecure library APIs to support while programming the authenticated and the encrypted bitstream files.

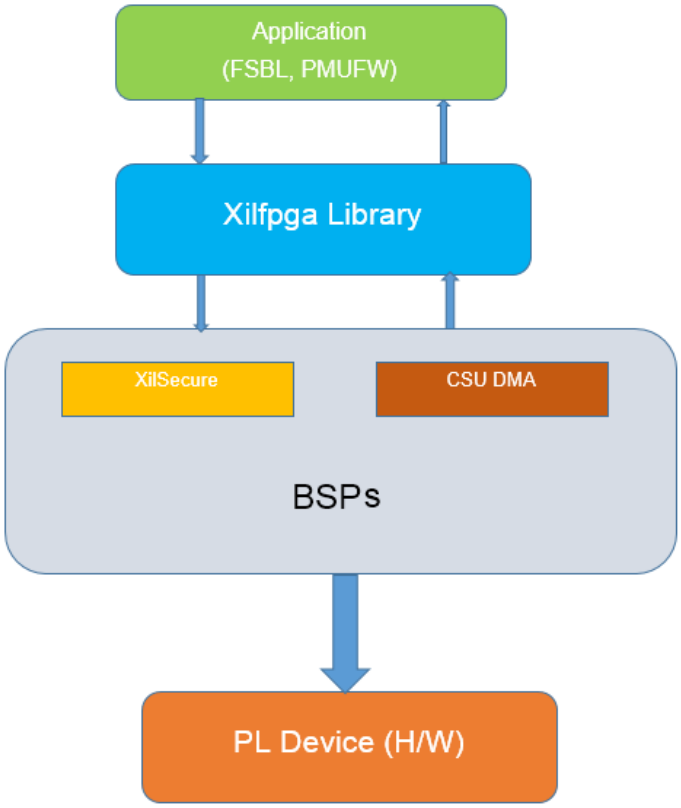

Figure 1.1: XilFPGA Design Summary

Flow Diagram

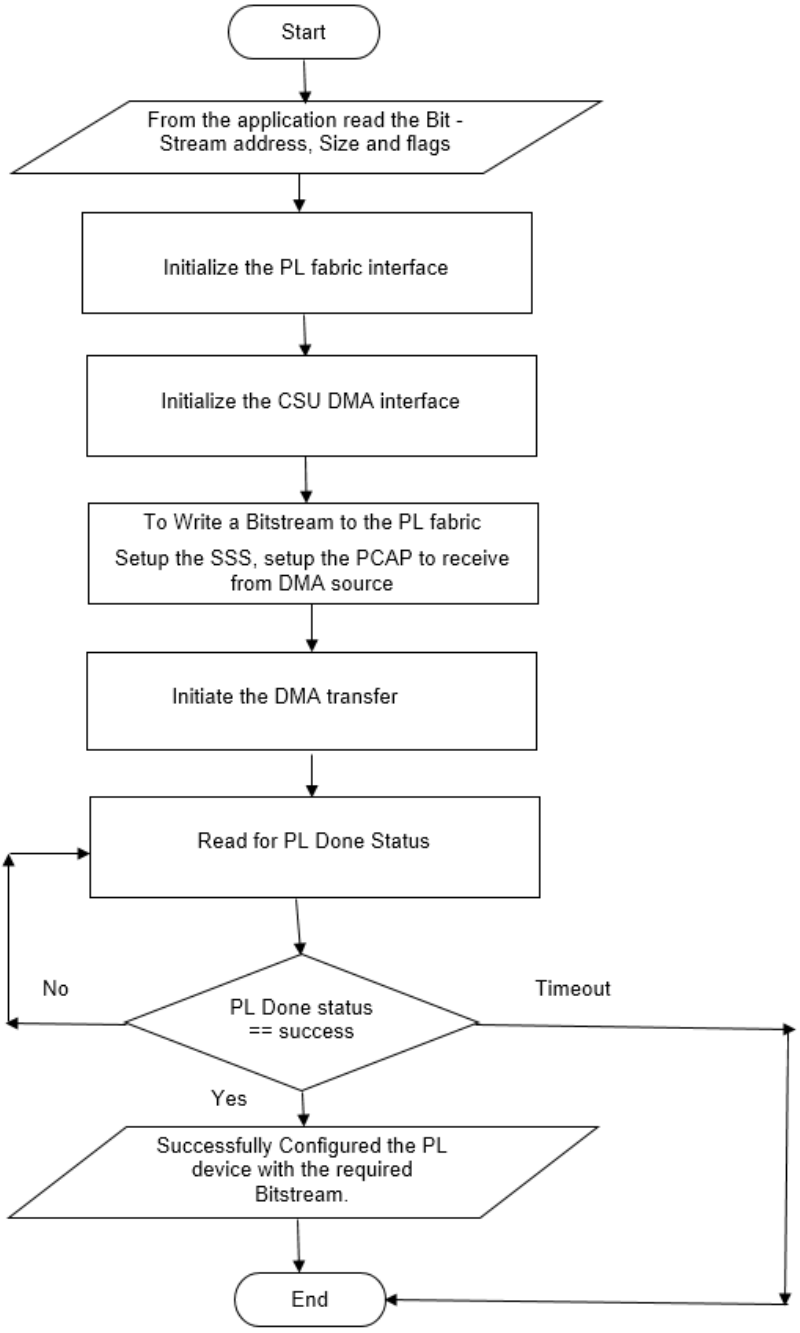

Figure 1.2: XilFPGA Library Workflow

Setting up the Software System

To use XilFPGA in a software application, you must first compile the XilFPGA library as part of software application.

1. Launch Xilinx SDK. Xilinx SDK prompts you to create a workspace.
2. Select **File** > **New** > **Xilinx Board Support Package**. The **New Board Support Package** wizard appears.
3. Specify a project name.
4. Select **Standalone** from the **Board Support Package OS** drop-down list. The **Board Support Package Settings** wizard appears.
5. Select the **xilfpga** library from the list of **Supported Libraries**.
6. Expand the **Overview** tree and select **xilfpga**. The configuration options for xilfpga are listed.
7. Configure the xilfpga by providing the base address of the Bit-stream file (DDR address) and the size (in bytes).
8. Click **OK**. The board support package automatically builds with XilFPGA library included in it.
9. Double-click the **system.mss** file to open it in the **Editor** view.
10. Scroll-down and locate the **Libraries** chapter.
11. Click **Import Examples** adjacent to the XilFPGA 4.1 entry.

Enabling Secure Mode in PMUFirmware

To support encrypted and authenticated bit-stream loading, you must enable secure mode in PMUFW.

1. Launch Xilinx SDK. Xilinx SDK prompts you to create a workspace.
2. Select **File** > **New** > **Application Project**. The **New Application Project** wizard appears.
3. Specify a project name.
4. Select **Standalone** from the **OS Platform** drop-down list.
5. Select a supported hardware platform.
6. Select **psu_pmu_0** from the **Processor** drop-down list.
7. Click Next. The **Templates** page appears.
8. Select **ZynqMP PMU Firmware** from the **Available Templates** list.
9. Click **Finish**. A PMUFW application project is created with the required BSPs.
10. Double-click the **system.mss** file to open it in the **Editor** view.

11. Click the **Modify this BSP's Settings** button. The **Board Support Package Settings** dialog box appears.
12. Select **xilfpga**. Various settings related to the library appears.
13. Select **secure_mode** and modify its value to **true** .
14. Click **OK** to save the configuration.

Bitstream Authentication Using External Memory

Authentication of Bitstream is different from that of all other partitions. The FSBL can be altogether contained within the OCM, and therefore authenticated and decrypted inside the device. For Bitstream, the size of the file is too large to be contained inside the device and external memory must be used. The use of external memory could pose access security issues. The following section describes how Bitstream is authenticated securely using external memory.

Bootgen

When a Bitstream is requested for authentication, Bootgen divides the Bitstream into blocks of 8MB each and assigns an authentication certificate for each block. If the size of a Bitstream is not in multiples of 8 MB, the last block contains the remaining Bitstream data.

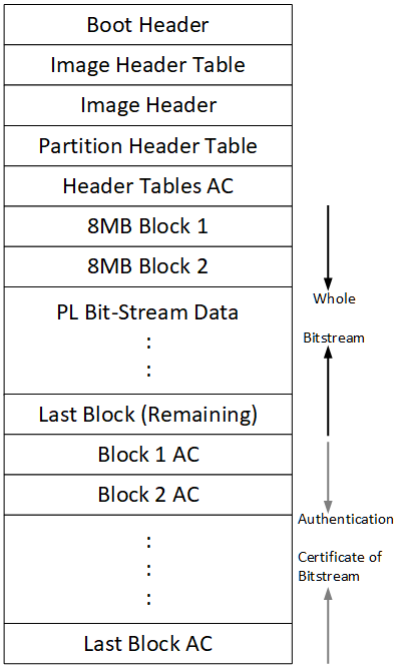

Figure 1.3: Bitstream Blocks

When both authentication and encryption are enabled, encryption is first done on the Bitstream. Bootgen then divides the encrypted data into blocks and assigns an Authentication certificate for each block.

Authenticated Bitstream Loading Using OCM

To authenticate the Bitstream partition securely, XilFPGA uses the FSBL chapter's OCM memory to copy the bitstream in chunks from DDR.
The software workflow for authenticating Bitstream is as follows:

1. XilFPGA identifies DDR secure Bitstream image base address. XilFPGA has two buffers in OCM, Read Buffer of size 56KB and hash's of Chunks to store intermediate hashes calculated for each 56 KB of every 8MB block.
2. XilFPGA copies a 56KB chunk from the first 8MB block to Read Buffer.
3. XilFPGA calculates hash on 56 KB and stores in HashsOfChunks.
4. XilFPGA repeats steps 1 to 3 until the entire 8MB of block is completed.

   Note

   The chunk that XilFPGA copies can be of any size. A 56KB chunk is taken for better performance.

5. XilFPGA authenticates the Bitstream.
6. Once the authentication is successful, XilFPGA starts copying information in batches of 56KB starting from the first block which is located in DDR to Read Buffer, calculates the hash, and then compares it with the hash stored at HashsOfChunks.
7. If the hash comparison is successful, FSBL transmits data to PCAP using DMA (for un-encrypted Bitstream) or AES (if encryption is enabled).
8. XilFPGA repeats steps 6 and 7 until the entire 8MB block is completed.
9. Repeats steps 1 through 8 for all the blocks of Bitstream.

Note

You cannot use the warm restart when the FSBL OCM memory is used to authenticate the Bitstream.

Authenticated Bitstream Loading Using DDR

XilFPGA uses DDR to authenticate as In-sufficient OCM memory (OCM memory occupies with ATF and FSBL).
The software workflow for authenticating Bitstream is as follows:

1. XilFPGA identifies DDR secure Bitstream image base address.
2. XilFPGA calculates hash for the first 8MB block.
3. XilFPGA authenticates the 8MB block
4. If Authentication is successful, XilFPGA transmits data to PCAP via DMA (for unencrypted Bitstream) or AES (if encryption is enabled).
5. Repeats steps 1 through 4 for all the blocks of Bitstream.

XILINX

Chapter 2

XilFPGA APIs

Overview

This chapter provides detailed descriptions of the XilFPGA library APIs.
The XILFPGA library provides the interface to the application to configure the programmable logic (PL) though the PS.

Supported Features

- Full Bit-stream loading
- Partial Bit-stream loading
- Encrypted Bit-stream loading
- Authenticated Bit-stream loading
- Authenticated and Encrypted Bit-stream loading

Xilfpga_PL library Interface modules

Xilfpga_PL library uses the below major components to configure the PL through PS.

- CSU DMA driver is used to transfer the actual Bit stream file for the PS to PL after PCAP initialization
- Xilsecure_library provides APIs to access secure hardware on the Zynq® UltraScale+™ MPSoC devices. This library includes:
    - SHA-3 engine hash functions
    - AES for symmetric key encryption
    - RSA for authentication

These algorithms are needed to support to load the Encrypted and Authenticated bit-streams into PL.

Note

XilFPGA library is capable of loading only .bin format files into PL. The library does not support other file formats. The current implementation supports only Full Bit-stream.

Initialization & Writing Bit-Stream

Use the u32 XFpga_PL_BitSream_Load(); function to initialize the driver and load the bit-stream.

Functions

- u32 Xfpga_GetConfigReg (u32 ConfigReg, u32 *RegData)
- u32 XFpga_PL_BitSream_Load (UINTPTR WrAddr, UINTPTR KeyAddr, u32 flags)
- u32 XFpga_PcapStatus (void)

Function Documentation

u32 Xfpga_GetConfigReg (u32 *ConfigReg,* u32 * *RegData*)

Returns the value of the specified configuration register.

Parameters

InstancePtr	is a pointer to the XHwIcap instance.
ConfigReg	is a constant which represents the configuration register value to be returned.
RegData	is the value of the specified configuration register.

Returns

- XST_SUCCESS if successful
- XST_FAILURE if unsuccessful

u32 XFpga_PL_BitSream_Load (UINTPTR *WrAddr,* UINTPTR *AddrPtr,* u32 *flags*)

The API is used to load the user provided bitstream file into Zync MPSoC PL region.
This function does the following jobs:

- Power-up the PL fabric.
- Performs PL-PS Isolation.
- Initialize PCAP Interface
- Write a bitstream into the PL
- Wait for the PL Done Status.
- Restore PS-PL Isolation (Power-up PL fabric).

Note

This function contains the polling implementation to provide the PL reset wait time due to this polling implementation the function call is blocked till the time out value expires or gets the appropriate status value from the PL Done Status register.

Parameters

WrAddr	Linear memory image base address
AddrPtr	Aes key address which is used for Decryption.

flags	Flags are used to specify the type of bitstream file.
	• BIT(0) - Bit-stream type
	  ◦ 0 - Full Bit-stream
	  ◦ 1 - Partial Bit-stream
	• BIT(1) - Authentication using DDR
	  ◦ 1 - Enable
	  ◦ 0 - Disable
	• BIT(2) - Authentication using OCM
	  ◦ 1 - Enable
	  ◦ 0 - Disable
	• BIT(3) - User-key Encryption
	  ◦ 1 - Enable
	  ◦ 0 - Disable
	• BIT(4) - Device-key Encryption
	  ◦ 1 - Enable
	  ◦ 0 - Disable

Note

The current implementation will not support partial Bit-stream loading.

Returns

- Error status based on implemented functionality (SUCCESS by default).

u32 XFpga_PcapStatus (void)

Provides the STATUS of PCAP interface.

Parameters

None	

Returns

Status of the PCAP interface.

XILINX

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