Summary

This lab application note describes the creation and implementation of a single-chip general purpose 2 channel compute system using the Low Power Domain as one channel and a triple modular redundant MicroBlaze™ processor in the programmable logic (PL) both interconnected by a Mailbox also in the PL.

Complete step-by-step instructions are given for the entire process, explaining the use of the Isolation Design Flow (IDF). This document explains how to implement isolated functions in a single Xilinx® Zynq® UltraScale+™ MPSoC device for the example solution.

With this application note, designers can develop a fail-safe single chip solution using the Xilinx IDF that meets fail-safe and physical security requirements for an example high-assurance application.

The rules for IDF defined in this application note follow the rules defined in the Isolation Design Flow for Zynq UltraScale+ Application Note (XAPP1335) and in principle do not differ from those defined in 7 Series Isolation Design Flow Lab Using ISE Design Suite 14.4 (XAPP1085), but the methodology for implementation using Vivado® tools does.

Download the isolation design example reference files from the Xilinx website. For detailed information about the design files, see Reference Design.

For detailed information about the Isolation Design Flow (IDF), refer to the IDF website at (www.xilinx.com/idf).

Introduction

The Xilinx® Isolation Design Flow (IDF) is the design methodology that allows for Information Assurance, Functional Safety implementations, or any other application requiring module both physical and logical isolation. This methodology is backed by significant schematic analysis and software verification—Vivado® Isolation Verifier (VIV)—to ensure elimination of single points of failure. SCC is one specific application of IDF allowing the implementation of a multichip cryptography system in a single FPGA or SoC.

Note: This lab targets Vivado Design Suite 2018.3. Versions beyond this might have slightly different screen images.
Lab Design Overview

The Zynq Ultrascale+ Isolation Design Flow (IDF) rules are outlined in the Isolation Design Flow for Zynq Ultrascale+ MPSoC Application Note (XAPP1335).

This lab gives details on how functions are to be isolated, specific differences between a normal partition flow and an IDF partition flow, information on IDF-specific hardware description language (HDL) code mnemonics, and trusted routing rules.

To illustrate the IDF and its capabilities, this design implements an isolated, triple modular redundant MicroBlaze processing subsystem, which communicates to the Arm® Cortex™-R5 core in the Low Power Domain. Refer to the Triple Modular Redundancy (TMR) LogiCORE IP Product Guide (PG268) for TMR guidance. The following figure shows a hierarchical diagram of the sub-blocks used in the implementation of this design.

Figure 1: Design Hierarchy Block Diagram

![Design Hierarchy Block Diagram]

IMPORTANT! For functional safety, to mitigate the probability of Common Cause Failures, isolate each MicroBlaze CPU with each other and the processing system (PS).
Isolation Design Flow

The Xilinx® Isolation design flow (IDF) adds the standard design flow by mapping the design hierarchy into isolated containers called Pblocks. The reason is to stop logical optimization between Pblocks and to separate concerns (isolation from a single fault).

Referring to the following figure, the standard flow is shown on the left with the IDF enhancements on the right side. This lab is broken down into eight phases; for reference, each phase is highlighted in this application note.

Figure 2: Isolation Design Flow Enhancements

The lab is driven by a set of Tcl scripts broken down into eight phases. Each phase has to be run to completion, in order. You can save the design at any point and continue to run the lab at a later date.

Along with the eight phase files, There is a Tcl file called lab_server_option.tcl. This file is used to set the jobs count higher (16) if you are running this lab on a server and it speeds up the lab's execution. This variable is temporary so it needs to be sourced again if your saved your lab, at the beginning of your next session.
Phase 1

This phase runs the initial script that performs all the housekeeping for the lab. After the project is created, Vivado® Isolation Verifier (VIV) 2.0 is enabled, and a single-instance of a MicroBlaze™ processor and instantiation of the processing system (PS). The hierarchy is then created for the MicroBlaze processor, followed by the instantiation of the TMR Manager, which is then used to create the triple mode redundancy (TMR), as shown in the following figure.

Figure 3: Initial Script Result

1. Create a target design folder. For this example, use ZU+_LAB.
2. Copy the eight lab scripts (lab_phase1.tcl through lab_phase8.tcl) into the target design folder.
4. In the Tcl Console, point the console input to the design folder.
5. In the Tcl Console, enter source ./lab_phase1.tcl.
Phase 2

This phase adds wrappers to each of the three MicroBlaze™ containers completing three hierarchies and also creating two more hierarchies—one for the Mailbox and the other for the PS, as shown in the following figure. This allows the tools to split nets between isolated blocks which is required for a multi-region net. Such a net needs to be split in a fashion so a fault from a block effecting that net cannot fault any other blocks if that net were not split.

In the Tcl Console, enter `source ./lab_phase2.tcl`. 
This phase completes with the design split into five hierarchies as shown in the following figure.

Figure 5: Adding Wrappers Step

Figure 6: Result Showing the Five Wrappers
Phase 3

This phase performs the register transfer level (RTL) elaboration and then sets the HD.ISOLATED property for each hierarchical wrapper, as shown in the following figures. Setting HD.ISOLATED is the key property that changes the rule set for the Placer and Router driving an isolated design.

In the Tcl Console, enter `source ./lab_phase3.tcl`.

Figure 7: Elaboration and HD.ISOLATED Properties Step
Figure 8: Phase 3 Elaborated Design Result
Phase 4

This phase synthesizes the design which supports and creates the actual logical resources needed to complete the design as shown in the following figures.

Figure 9: Synthesized Step

In the Tcl Console, enter `source ./lab_phase4.tcl.`
After this phase completes, on the Synthesis Completed popup window, click **Cancel** to proceed to the next phase.

*Figure 10: Phase 4 Synthesized Design Result*
Phase 5

This phase is the floorplanning step where Pblocks are created and assigned to each logical hierarchy. This defines which resource the placer can use to implement the hierarchy of the design.

Figure 11: Floor Planning Step

When creating a Pblock, verify the adequate resources that are available based on the synthesis of each hierarchy. This is verified by looking at the Physical Resources Estimates table in the Pblock Properties window. To view this table, select the Statistics tab, as shown in the following figure.

In the Tcl console, enter `source ./lab_phase5.tcl` to keep the flow in-line with previous Phases. If there are any entries in red (which indicates that there are no resources for a desired site type), the design will not route. If the %Utilization is high, the design might not route. Xilinx® recommends that the % Util be below 50%. The other activity is pin placement where pin resources are assigned to resourced I/O blocks.
Figure 12: Pblock and Resources, Phase 5 Result
Phase 6

This phase (see the following figure) is used to verify the creation of the fence by checking the Pblock placement, as well as I/O Bank violations caused by incorrect pin assignments.

Figure 13: **VIV 2.0 Floor Planning DRC Step**

Vivado® Isolation Verifier (VIV) 2.0 is used to run four design rule checks (DRCs) (DRC #1, #2, #3 and #4), as shown in the following figures. Refer to the Vivado Isolation Verifier User Guide (UG1291) for details on the DRCs.
Figure 14: Phase 6 VIV 2.0 DRC 1,2,3, and 4 Run Result
The report shows information on the device provenance (IDF_VIV2-1) and the negative results for constraints testing IDF_VIV2-2 I/O bank violation, IDF_VIV2-3 package pin violation, IDF_VIV2-4 floorplan violation. No IDF violations will be seen in the DRC report. Refer to the Vivado Isolation Verifier User Guide for Zynq UltraScale+ MPSoC (UG1291) for DRC details.
Phase 7

This phase (see the following figure), implements the design by:

1. Optimizing the design within the constraints of the Xilinx® Isolation Design Flow (IDF).
2. Placing the design based on Pblocks.
3. Routing the design.
4. In the Tcl console, enter source ./lab_phase7.tcl. to keep the flow in-line with previous phases.

Figure 16: Implementation Step
After successfully implementing the design, in the Implementation Completed popup window, click **Cancel** to proceed to the next phase as shown in the following figure.

**Figure 17: Phase 7 Implementation Result**
Phase 8

After implementation, using the Vivado® Isolation Verifier (VIV) 2.0 tool, a final set of design rule checks (DRCs) are required to verify both placement and routing comply with the Xilinx® Isolation Design Flow (IDF) rules, as shown in the following figure.

*DRC #1 is re-run to maintain the providence in the report, followed by DRC #5 and DRC #6, which are ran on the finished design. Refer to the Vivado Isolation Verifier User Guide for Zynq Ultrascale+ MPSoC (UG1291) for DRC details. In the Tcl console, enter `source ./lab_phase8.tcl` to keep the flow in-line with previous phases.*
Figure 19: Phase 8 VIV 2.0 DRC Result
The result after VIV 2.0 is run with the report is shown in the following figure. Note that no IDF violations will be seen in the DRC report. Refer to the *Vivado Isolation Verfier User Guide for Zynq Ultrascale+ MPSoC* (UG1291) for DRC details.

*Figure 20: VIV 2.0 #5,6 DRC Report*
Reference Design

The reference design file contains example Tcl scripts and can be downloaded from the Xilinx website. The following table shows the reference design matrix.

Table 1: Reference Design Matrix

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developer name</td>
<td>Xilinx</td>
</tr>
<tr>
<td>Target devices</td>
<td>UltraScale+ devices</td>
</tr>
<tr>
<td>Source code provided?</td>
<td>Yes</td>
</tr>
<tr>
<td>Source code format (if provided)</td>
<td>Tcl</td>
</tr>
<tr>
<td>Design uses code or IP from existing reference design, application note, 3rd party or Vivado software? If yes, list.</td>
<td>N/A</td>
</tr>
<tr>
<td>Functional simulation performed</td>
<td>N/A</td>
</tr>
<tr>
<td>Timing simulation performed?</td>
<td>N/A</td>
</tr>
<tr>
<td>Test bench provided for functional and timing simulation?</td>
<td>N/A</td>
</tr>
<tr>
<td>Test bench format</td>
<td>N/A</td>
</tr>
<tr>
<td>Simulator software and version</td>
<td>N/A</td>
</tr>
<tr>
<td>SPICE/IBIS simulations</td>
<td>N/A</td>
</tr>
<tr>
<td>Synthesis software tools/versions used</td>
<td>N/A</td>
</tr>
<tr>
<td>Implementation software tool(s) and version</td>
<td>Isolation Design Flow (IDF) and Vivado Design Suite v2018.3</td>
</tr>
<tr>
<td>Static timing analysis performed?</td>
<td>N/A</td>
</tr>
<tr>
<td>Hardware verified?</td>
<td>N/A</td>
</tr>
<tr>
<td>Platform used for verification</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Conclusion

This application note provides a step-by-step example to implement a complete Zynq® UltraScale+™ MPSoC isolated design. All of the necessary IDF steps are shown, and the rules and guidelines detailed in the Isolation Design for Zynq UltraScale+ Application Note (XAPP1335) are highlighted. This lab is built on self-contained Tcl scripts that allow the user to single-step through the design process at will or source each phase as outlined in this application note.
Revision History

The following table shows the revision history for this document.

<table>
<thead>
<tr>
<th>Section</th>
<th>Revision Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>02/15/2019 Version 1.0</td>
<td>Initial Xilinx release.</td>
</tr>
</tbody>
</table>

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.

References

1. *Isolation Design Flow for Zynq UltraScale+ Application Note* (XAPP1335)
9. *Aerospace and Defense Security Monitor IP Core Product Marketing Brief*
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 https://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 https://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

© Copyright 2019 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. 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.