Enhancements in UltraScale CLB Architecture

Shant Chandrakar
Xilinx Inc.
Hyderabad, India
shantc@xilinx.com

Dinesh Gaitonde
Xilinx Inc.
San Jose, CA
dineshg@xilinx.com

Trevor Bauer
Xilinx Inc.
Longmont, CO
trevor@xilinx.com

ABSTRACT

Each generation of FPGA architecture benefits from optimizations around its technology node and target usage. In this paper, we discuss some of the changes made to the CLB for Xilinx's 20nm UltraScale product family. We motivate those changes and demonstrate better results than previous CLB architectures on a variety of metrics. We show that, in demanding scenarios, logic placed in an UltraScale device requires 16% less wirelength than 7-series. Designs mapped to UltraScale devices also require fewer logic tiles. In this paper, we demonstrate the utilization benefits of the UltraScale CLB attributed to certain CLB enhancements. The enhancements described herein result in an average packing improvement of 3% for the example design suite. We also show that the UltraScale architecture handles aggressive, tighter packing more gracefully than previous generations of FPGA. These significant reductions in wirelength and CLB counts translate directly into power, performance and ease-of-use benefits.

Categories and Subject Descriptors

B.5.2 [REGISTER-TRANSFER-LEVEL IMPLEMENTATION]: Design Aids—*Optimization*; B.6.1 [LOGIC DESIGN]: Design Styles—*Logic arrays*; B.7.2 [INTEGRATED CIRCUITS]: Design Aids—*Placement and routing*

Keywords

FPGA, CLB, packing, placement, routing, logic array, control set, wirelength

1. INTRODUCTION

At each technology node, the architecture of an FPGA fabric needs to be reconsidered. Customer use models and physical implementation algorithms also play a significant role in the design and evaluation of the FPGA fabric. Xilinx recently introduced the UltraScale architecture for TSMC's

20nm process. In this paper, we compare several key aspects of the UltraScale CLB with the previous generation, 7-series CLB. Through experimental results, we motivate some of the changes made between these two CLB definitions. We describe how these decisions were informed by the technology constraints and opportunities. We also describe what modifications were performed to support the increasingly complex demands customers place on their FPGAs. This paper is focused on a subset of the changes made to the CLB structure. A small change to the CLB is shown to make a large impact in the resources required to route difficult designs. Using fewer routing resources enables lower congestion, improved power and higher performance. More information about other aspects of the FPGA fabric and other blocks in UltraScale has been published elsewhere[10].

We demonstrate the benefits of the changes using large designs representing real customer use cases.

2. 7-SERIES CLB & SLICE DEFINITION

The Configurable Logic Block (CLB) is the FPGA's main logic resource for implementing combinatorial and sequential logic functions. Each CLB is connected to the general routing fabric through a switch matrix. The Xilinx 7-series CLB, shown in Figure 1, consists of two slices. Figure 2 represents a typical SLICEL found in 7-Series CLBs. The SLICEL is built from Basic Logic Elements (BELs) such as Look up Tables (LUTs), flops, carry logic and wide-function multiplexers. Each slice consists of four 6-input LUTs, each of which can implement any combinatorial function of up to six inputs. The 6-input LUT can also be used as two 5-input LUTs, provided the combined number of unique inputs is less than six. Each slice also contains eight flops, four of which are primary, having direct connectivity to and from the switch matrix; the other four are secondary with indirect fabric connectivity only through LUTs. Another type of slice, called SLICEM, enables distributed memories and shift registers in addition to SLICEL capability.

There are two types of flops in the SLICEL, primary and secondary, differentiated by their connectivity. Primary flops have direct connectivity to the switch matrix through their respective Q pins (i.e. AQ, BQ, CQ, DQ). The secondary flop outputs do not have direct connectivity to the switch matrix; rather, they are required to share an output with other SLICEL signals. Further, each primary/secondary pair of flops shares a single bypass input (i.e. AX, BX, CX, DX). Thus, only one flop in each pair can be driven directly from the switch matrix; the other requires a LUT configured as a logic function or buffer. This

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the Owner/Author. Copyright is held by the owner/author(s).
FPGA'15, February 22–24, 2015, Monterey, California, USA.
ACM 978-1-4503-3315-3/15/02.
<http://dx.doi.org/10.1145/2684746.2689077>.

Figure 1: Arrangement of Slices within the CLB taken from Virtex 7 User Guide [6]

connectivity biases the packer to use only one of the two flops for independent registering.

Another factor determining the usage of flops in CLBs is control set resolution. The signals that control a flop are the clock (CLK), set/reset (SR) and clock enable (CE). The collection of these three signals is called the control set of a flop. Two flops are said to have compatible control sets if they share the same CLK, SR and CE nets. Figure 3 shows the connectivity of the control sets to all the flops in a 7-series CLB. Only one control set can exist in each slice. So, for example, two distinct clock signals cannot drive flops within the same slice. Since a 7-series CLB has two slices, flops that get mapped to one CLB have to be partitionable to at most two control sets, each of which cannot exceed eight flops.

3. FPGA FABRIC TRENDS

CLBs and interconnect are key resources in each FPGA fabric. Overall system performance, cost and power are impacted by the characteristics of these ubiquitous resources. The CLB and interconnect design needs to support the latest technology requirements and new user demands. Moreover, the tools and architecture must work effectively together.

3.1 Routability

As has been documented [1], interconnect is a key bottleneck to technology scaling. Wires are not scaling as well as gates. Increased bandwidth demands in customer designs translate into wider buses, meaning more wires are required. This trend will continue. Thus, anything that can be done in an FPGA fabric or physical implementation tool to reduce the number of routing resources required for a given design will help respond to this trend.

3.2 Performance

Similar to routability, design performance is also, to a large extent, determined by interconnect use. Interconnect delay has come to be a significant fraction of total delay

Figure 2: 7-Series SLICEL taken from Virtex 7 User Guide [6]

Figure 3: Connectivity of control set signals to the flops of 7-Series CLB

Figure 4: A simplified representation of routing in 7-series

in critical paths. The UltraScale CLB architecture is optimized to aid placement and routing algorithms achieve critical paths with fewer interconnect resources. Another technique often used to improve performance is pipelining. Achieving the maximum benefit from pipelining requires the pipeline flops to be placed in ideal locations. Any perturbation in flop placement due to architectural restrictions will impact the benefit of pipelining.

3.3 Power

Power consumption is also important. In recent FPGA generations, more designs have been limited by their power budgets. One technique that has achieved good results is aggressive clock gating. Algorithms exist [3], supported by implementation tools [5], which can synthesize aggressive dynamic clock gating conditions to save power. Synthesis of these structures results in fragmented control sets for flops, further impacting packing density.

3.4 Software Tools

A CLB architecture definition has implications on software tools. Consider, for example, the disparate demands on the slice structure. Due to shared connectivity within a slice, there are restrictions on what logic can be packed into a single slice. At the level above a slice, few restrictions exist on placement. Questions about whether a given placement of slices is legal, in terms of whether it violates any FPGA restrictions, are easy to answer. Whether a collection of BELs that are packed into a slice is legal is a much tougher question to answer [4]. Fewer restrictions on what logic can be packed into a slice improves tool run-time, tool predictability and design performance.

4. ULTRASCALE CLB

The UltraScale CLB is designed to improve routability, performance and power. It also helps improve tool runtime and predictability. The changes primarily involved making the flops in a slice more accessible. This increased usability is provided via two key changes in the architecture. First, independent access to all flop inputs and from all flop outputs is possible in UltraScale. Second, more unique control sets are legally placeable in the same slice.

The UltraScale CLB has the same number of LUTs and flops as a 7-series CLB. However, the UltraScale CLB is organized as a single, coarser slice having the same capacity as two 7-series slices. The functionality of the LUTs and flops has been retained. Figure 5 shows one of the eight LUT-flop pairs available in the UltraScale CLB.

In contrast to the 7-series CLB, which has one bypass input for every two flops, the UltraScale CLB has just as many bypass inputs as flops. This allows direct access to all CLB flops without consuming the corresponding LUTs. Additionally, while secondary flops in 7-series share an output with other slice functionality, all flops in UltraScale slices have their own, direct and independent output pins connecting to the switch matrix. Normally, adding extra pins is expensive since the switch matrix also has to be scaled to accommodate the new pins. In UltraScale, however, this cost has been amortized over the other pins. There have been significant changes to the architecture in UltraScale compared to 7-series. What we describe here are UltraScale modifications relevant to the changes described above. As shown in Figure 6, general interconnect wires drive CLB in-

puts and are in turn driven by CLB outputs. In 7-series, the general interconnect drives 56 inputs in a CLB made of just SLICELs and 60 inputs in a CLB made of SLICEL and SLICEM. The general interconnect is driven by 24 outputs from the CLB. In UltraScale, the general interconnect connects to each CLB through 64 inputs and 32 outputs.

Although the number of input and output pins on the CLB increased from 7-series to UltraScale, the number of general interconnect signals (singles, doubles) driving the CLB did not. A simple implementation of the input change would require 8 more input muxes for SLICELs to accommodate the extra inputs and the associated cost of a more complicated switch matrix. Instead of incurring this cost, we repurposed existing muxes in the 7-series. As a result, some amount of the flexibility at the input switch matrix is lost. Experiments showed that the marginal loss in flexibility was a price worth paying for all the other benefits of this change (details of the benefits are described later on in the paper). Characteristics of designs mapped to UltraScale are expected to be different than in 7-series. The CLB architecture was changed to accommodate the new design styles.

The number of CLB outputs increased from 24 in 7-series to 32 in UltraScale. The extra outputs support the independent flip-flop outputs. This increase did not drive any downstream modifications to the general interconnect; rather, flexibility in the reach of each output was reduced proportionally. In short, both for non-control inputs and outputs, extra CLB flexibility was introduced at the expense of flexibility with respect to general interconnect.

These changes have made the job of placing flops on a device easier. Consider the objective of a placer. The placer tries to create a legal, routable placement which meets a design's timing constraints and minimizes wirelength. In order to do this, the placer needs to have an idea of how much capacity each slice has. In 7-series, the number of LUTs available in the slice depends on how many flops in the slice require a LUT route-thru and the number of flops available in the slice depends on how many LUTs are prevented from being used as route-thrus. This makes accounting for capacity cumbersome for the placer. Specifically, it is not easy for the placer to decide whether it should replace a logical LUT with a route-thru LUT to accommodate an additional flop or remove an existing flop to replace a route-thru LUT with a logical LUT. The UltraScale CLB solves this problem with bypass inputs and independent outputs for all flops. This enhancement benefits highly pipelined designs, allowing the device to support high performance designs. It also reduces the number of resources wasted - LUTs used as route-thrus or flop locations left empty.

One other enhancement from the 7-series CLB to the UltraScale CLB is the ability to pack more diverse control sets in one CLB. Specifically, flops that have more unique clock enables and reset nets can now be packed together in one CLB, as shown in Figure 7. Figure 8 shows the connectivity of the control sets to the 7-Series and UltraScale CLB. The control set signals of adjacent CLBs are driven primarily by common global control wires. There are 12 control set signals for a pair of adjacent 7-Series CLBs which are driven by global control wires. In order to accommodate extra clock enables in the UltraScale CLB, the number of control muxes was increased by 6.

This change of having flops with more diverse control sets packable in the same CLB also benefits power consumption.

Figure 5: 1 of the 8 LUT-flop pair from UltraScale CLB [8]

Figure 6: Interface between CLB input and output with general interconnect wires

Figure 7: Connectivity of control set signals to the flops of UltraScale CLB

Figure 8: Interface between CLB's control set signals with general interconnect wires and global control signals.

One very effective way of reducing dynamic power in the fabric is by synthesizing clock gating logic. This saves unnecessary clock and data switching. Both 7-series and UltraScale CLBs and tools support power reduction this way in Vivado [9]. More clock enables imply more control sets for the flops, thereby reducing the likelihood that two randomly chosen flops can coexist in a slice. If the wirelength of a placed design increases due to the synthesis of more control sets, then the power savings from reduced switching on the clock tree would be squandered due to longer routes.

In summary, changes were made to the UltraScale CLB to enable tighter packing. This reduces power consumption, both due to shorter wirelengths and enabling more aggressive clock gating. This also improves performance and reduces demand on interconnect. In the next section we show experimental results that validate these claims.

5. RESULTS

5.1 Experimental Setup

We now describe experiments we performed to validate the advantages of UltraScale CLBs. Since these CLB enhancements are not the only thing that changed from 7-series, we extract the effect of these changes by designing targeted experiments. For example, one would expect UltraScale de-

signs to be faster than 7-series due to technology scaling or a variety of other architectural enhancements. As a result, we don't compare 7-series metrics directly with UltraScale. Instead we compute how metrics are impacted by different kinds of stress in 7-series and UltraScale.

The designs we chose for this work are all customer designs or IPs. Their identifying characteristics have been removed from our results. These designs include a variety of FPGA resources in addition to flops and LUTs, such as Block RAMs, DSPs, etc... These designs are implemented on comparable 7-Series and UltraScale devices. We have used the Vivado Design Suite for implementing the designs [9].

Designers want to use as many resources as possible from a given device. Therefore, while implementing a design, they would target close to 100% device utilization if possible. In order to force the placer to pack aggressively, we create several scenarios for each design. Each scenario represents an increasingly aggressive packing requirement. This is achieved by imposing a series of increasingly tightened area constraints. Figure 9 shows the flow used to achieve the increasingly aggressive scenarios.

As the placeable area reduces, the tools struggle to pack the design. While doing this, the tools have to trade off some design metric, wirelength or performance, to achieve legal packing. At the extreme, if we force the design to fit in an area which does not have enough resources, then the placer will simply fail. These area constraints are imposed on the Vivado placer using pblocks.

UltraScale architecture is different from 7-series in more than just the CLB changes. We isolated the impact of just the CLB changes by modifying Vivado's flow for placing a design. Placers in general and Vivado's placer in particular, use an approximate view of the targeted architecture when placing a design. Specifically, placers use bounding box as a metric (or some variant thereof) to estimate how much routing resources a specific placement will take. They also employ simple metrics to judge whether a given placement will meet timing and route successfully. All these metrics are very architecture dependent, except possibly the bounding box metric. In our experiments, we turn off all response from Vivado to any architecture specific tuning, except for the new CLB structure. We do this to ensure that the tool is not optimizing for any feature of UltraScale besides the CLB changes. Secondly, during placement, in order to meet timing constraints, the Vivado placer estimates the delay for each connection. This estimate depends on other aspects of the architecture. We force Vivado to use the same delay estimates irrespective of which architecture is targeted. Finally, unless otherwise stated, the results reported represent the placer's predictions. In order to avoid favoring any one architecture, we don't run the router. Results reported post placement are purely a function of CLB logic changes and reflect no benefit of any other architectural changes.

In all the scenarios, we try to keep the aspect ratio of the pblocks as square as possible. By doing this we ensure that the shape does not affect the wirelength in either horizontal or vertical direction. Since we don't bias the pblock in any direction, we don't favor the particular characteristics of any one architecture. We show in subsequent sections how the UltraScale CLB is able to handle this stress more gracefully than 7-series, achieving higher utilization and lower wirelength.

Figure 9: Methodology to create aggressive packing scenario

In order to measure the UltraScale and 7-Series CLB response to aggressive packing, there is a need to identify the smallest possible region below which the design would fail. Since we are interested in understanding CLB changes only, we did not constrain the placement of non CLB resources to within the pblock. If we did try to do this, placement would fail due to particular characteristics of the device chosen rather than anything inherent in the CLB architecture. Therefore, the area constraint is only applied on the CLB resources.

5.2 Routability

One predictor for the number of routing resources required for a given design is the wirelength of the placement that the software generates. Wirelength is not a metric with complete fidelity to routability, but it is a close proxy. Most placement tools, all else being equal, attempt to minimize wirelength for a given design [2]. Improvements in placement and routing algorithms have a huge impact on total wirelength. Similarly, improvements in architecture, all else being equal, impact routability and wirelength. In an attempt to isolate the impact of the CLB changes, the same design suite and same software were used.

The extent to which the tool has to sacrifice wirelength in the interests of legality is an indication of how flexible the CLB architecture is. Figure 10 shows the improvement in wirelength generated by tools on UltraScale devices compared to 7-series under similar packing constraints, i.e. similar sized area constraints. The size of each area constraint chosen is the one that corresponds to the smallest successful 7-series placement. Wirelength of a placed design is computed as the sum of the Manhattan bounding box of each net in the design. For each net, Manhattan bounding box is computed in tile based units. This ensures that the wirelengths for the two experiments are comparable. We see an

Figure 10: Percentage improvement in wirelength for UltraScale compared to 7-series

average wirelength reduction of 16% in UltraScale compared to 7-series.

Since we only report bounding box wirelength, the improvement in wirelength is primarily explained by changes to the CLB logic architecture. Since wirelength is a primary metric optimized by placers to help routability, these results show that designs on UltraScale devices would be easier to route than 7-series even if the area constraints are tight. Smaller wirelength is also correlated with better performance and lower power consumption.

5.3 Effective Packing

We demonstrate the packing benefits of the UltraScale CLB by considering how densely we can pack our designs and still have successful placement. Given a design and an unconstrained FPGA device, the tools are free to use the entire device to place and route. Getting the densest packing is not one of the primary cost functions that the Vivado software optimizes. Instead, by not forcing the densest placement, the software is able to optimize for the best performance and power of a given design in the selected device. As a result, dense area constraints are required to demonstrate the minimal number of CLBs required for a design. Figures 11 and 12 show examples of placements achieved through unconstrained and constrained implementation of a design on 7-series using the above methodology.

Figure 13 shows the percentage of fewer CLBs UltraScale requires compared to 7-series under the tightest area constraints. The tightest area constraints represent the smallest number of CLBs used to achieve legal placement in both the architectures. Designs required as much as 14% fewer CLBs in UltraScale compared to 7-Series. On average, these CLB changes resulted in a 3% CLB reduction in UltraScale compared to 7-series. The UltraScale architecture provides many other improvements in density thanks to other changes from 7-series to UltraScale, including clocking, routing and IP [7].

As the graph in Figure 13 shows, the reduction in the number of CLBs achieved by UltraScale is not uniform. It depends on the design characteristics. The connectivity to and from secondary flops and control set restrictions impose constraints on the placer. Designs for which these restrictions impose a burden show more improvement in UltraScale compared to 7-series.

Figure 11: Default placement for a design on 7-series, taken from floorplanning view of Vivado [9]

Figure 12: Constrained placement for a design in the smallest area on 7-series, taken from floorplanning view of Vivado [9]

Figure 13: Percentage improvement in number of CLBs used by UltraScale compared to 7-series

In both 7-series and UltraScale, two “smaller” LUTs can be placed in one LUT6 location as long as the number of unique inputs from the two LUTs is less than six. This provides an opportunity to place flops in a CLB even if a secondary input is not available. As a result, the amount of usable flexibility offered by UltraScale CLB changes depends on the distribution of LUT sizes in the design. As the packer tries to merge small LUTs, it is successful more often in UltraScale than in 7-series. This is because the merging operation is not possible if the flop also requires the LUT to be configured as a route-through buffer.

Designs which are LUT-dominated and have a distribution of LUT sizes permitting extensive LUT merging show improved packing in UltraScale. For flop dominated designs, two different aspects of the design determine how effective our changes are. If the design has a lot of unique control sets, or if a design is flop dominated with a large fraction requiring independent access, the UltraScale CLB achieves improved packing efficiency. We expect that high performance design techniques such as aggressive pipelining and low power design techniques such as clock gating will result in more designs benefiting from UltraScale changes.

5.4 Performance

Similar to the previous experiments, we constrained each design to be placed in as small a region as possible. We expect a trend in predicted performance similar to what we see for predicted wirelength. As we tighten the region given to the placer performance suffers as the placer struggles to accommodate the design in the region. As the region is constrained further, the design fails to place. One measure of how forgiving the architecture is, is the ratio of the frequency achieved for a given area constraint as a percentage of the frequency achieved with 80% LUT utilization (baseline frequency). Figure 14 shows how the frequency of one particular design changes during this experiment. This particular design had 97K flops, 70K LUTs and 1100 control sets. While many other designs do not have trouble achieving much higher utilizations than 80% in 7-series, already a noticeable improvement over previous architectures, UltraScale is able to achieve close to maximum LUT utilization. Further, at the tightest achieved packing, UltraScale maintains performance within 12% of the baseline frequency. The same design when packed tightly into 7-series, shows about 20% degradation before struggling to pack.

Figure 15 shows the performance degradation across multiple designs in UltraScale and 7-Series for similar sized area

Figure 14: How frequency of one design changes in UltraScale and 7-series during the packing experiment

Figure 15: How performance changes For UltraScale and 7-series as design is constrained to tighter regions

constraints. The average performance degradation of these designs in UltraScale is 9% and in 7-Series is 24%. This implies that the performance of the UltraScale CLB architecture is maintained over more aggressive packing i.e. higher CLB utilizations.

It is important to restate here that these experiments were done in order to isolate the effect of CLB changes. In Figures 14 and 15, we do not report results after routing. Results reported after routing would have been contaminated with non-CLB architecture changes as well. We only report results post-placement. All results quoted are assuming the same delay model for interconnect delay across the two architectures. Needless to say, the improvements reported here are sustained through routing.

5.5 Power Optimization

In order to achieve more power reduction benefits from clock gating, we expect more diverse control sets in the future. Reducing power consumption has become an important optimization goal for FPGA users. We mimic a customer implementing clock gating for dynamic power reduction. We reduce power by using the *power_opt_design* [9] command. This command synthesizes extra clock enables for flops whose clocks can be safely gated.

In this experiment we observe that in some designs, power optimization does not create new clock enables (Figure 16). In other designs, it does (Figure 17). Figure 18 shows that, with power optimization, the UltraScale architecture can handle the increase in control sets. In the context of

Figure 16: No effect of power optimization on the control set count

Figure 17: Marked increase in the number of control sets with power optimization

power optimization, Vivado is able to gate clocks more aggressively in UltraScale without having adverse impacts on CLB utilization, wirelength and design performance.

Figure 18: Percentage improvement in number of CLBs used by UltraScale compared to 7-series after power optimization

6. CONCLUSIONS

We've compared the CLBs from UltraScale and 7-series, specifically with respect to the connectivity of their flops. Analysis has shown that the independent access provided to the flop inputs, outputs and control sets can improve the quality of a design's placement, including reductions in wirelength and increases in utilization. Increases in design complexity require more routing resources. We've shown here that there are ways of dealing with this besides just increasing the number of routing resources in the interconnect. Changes to the CLB logic can address the interconnect demand side of the problem as well.

7. ACKNOWLEDGEMENTS

We would like to thank several colleagues at Xilinx, too many to be individually named, for helping run some of the experiments and in discussing their implications. We also appreciate the constructive feedback from the anonymous reviewers.

8. REFERENCES

- [1] X. Chen, J. Liang, and H.-S. P. Wong. Interconnect scaling into the sub-10nm regime. In *Proceedings of the International Workshop on System Level Interconnect Prediction, SLIP '12*, pages 2–2, New York, NY, USA, 2012. ACM.
- [2] J. Cong, J. R. Shinnerl, M. Xie, T. Kong, and X. Yuan. Large-scale circuit placement. *ACM Trans. Des. Autom. Electron. Syst.*, 10(2):389–430, Apr. 2005.
- [3] A. P. Hurst. Automatic synthesis of clock gating logic with controlled netlist perturbation. In L. Fix, editor, *DAC*, pages 654–657. ACM, 2008.
- [4] J. Luu, J. H. Anderson, and J. Rose. Architecture description and packing for logic blocks with hierarchy, modes and complex interconnect. In *Proceedings of the ACM/SIGDA 19th International Symposium on Field Programmable Gate Arrays, FPGA 2011, Monterey, California, USA, February 27, March 1, 2011*, pages 227–236, 2011.
- [5] F. Rivoallon and J. Balasubramanian. *Reducing Switching Power with Intelligent Clock Gating*. Xilinx.
- [6] Xilinx. *7 Series FPGAs Configurable Logic Block - User Guide*.
- [7] Xilinx. *UltraScale Architecture and Product Overview*.
- [8] Xilinx. *UltraScale Architecture Configurable Logic Block Advance Specification User Guide*.
- [9] Xilinx. *Vivado Design Suite User Guide*.
- [10] S. Young, D. Gaitonde, and T. Bauer. Xilinx FPGAs case study: High capacity and performance 20nm FPGAs. HotChips, 2014.

DISCLAIMER

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>.