Technology Advancements for Dynamic Function eXchange in Vivado ML Edition

WP534 (v1.0) July 28, 2021

Abstract

As systems become more complex and designers are asked to do more with less, adaptability is a critical asset. While Xilinx® FPGAs and SoCs always provided the flexibility to perform on-site device reprogramming, current constraints including increased cost, tighter board space, and power consumption demand even more efficient design strategies. Xilinx Dynamic Function eXchange (DFX) extends the inherent flexibility of the silicon by allowing specific regions to be reprogrammed with new functionality while applications continue to run in the remainder of the device. DFX addresses three fundamental challenges while enabling the designer to:

- Make more efficient usage of hardware resources
- Keep a deployed system operational while application changes are made
- Reduce dynamic power consumption

Silicon and software technology for DFX in Vivado® ML edition continues to evolve over successive generations.

Introduction

Generations of Xilinx devices are uniquely able to dynamically reconfigure, which expands the capability and flexibility of the devices. From the groundbreaking experimental days in the 1990s with the XC6200 family designed for reconfigurable computing, to the latest Versal™ ACAPs with embedded processors and the dedicated network on chip (NoC) feature, each successive process node improves the dynamic reprogrammability of the devices. The most recent architectures are designed with DFX capabilities at their core. Xilinx understands that in-field flexibility is critical to meet fast changing conditions. DFX enables new solution possibilities.

Enabling Dynamic Solutions in Any Market

DFX is a core silicon and software technology feature across nearly all devices offered by Xilinx. These capabilities can also be considered for all markets. DFX is not for *every* application, but can be for *any* application. Examine your design structure for mutually-exclusive functions, requirements to update features on-the-fly, or the need to reduce power consumption or device count. The following examples describe where DFX enabled solutions are used for unique advantages.

Data Center

DFX is at the heart of many Alveo™ data center accelerator cards, producing broad adoption across many markets. Alveo cards are adaptable to changing acceleration requirements and algorithm standards. They are also capable of accelerating any workload without changing hardware, and can reduce overall cost of ownership. Alveo platforms remain active and connected to a host while new applications are loaded, changing context on demand. The Xilinx runtime library (XRT) enables application developers to leverage DFX to swap out different accelerator binaries on Xilinx platforms without worrying about low-level details.

Microsoft Azure uses Alveo U250 data center accelerator cards to enable FPGA-as-a-service (FaaS) (also known as the Azure FPGA Runtime Platform), which delivers seamless migration of applications between on-premises and the cloud. This acceleration platform uses DFX with two reconfigurable partitions and delivers standardized DMA plus advanced features including the AXI4 slave connection to your data mover, and data retention using DDR4 self-refresh while swapping kernels. These platforms can accelerate workloads including machine learning inference, video transcoding, high performance computing, and database search and analytics, greatly improving productivity across all these applications and more.

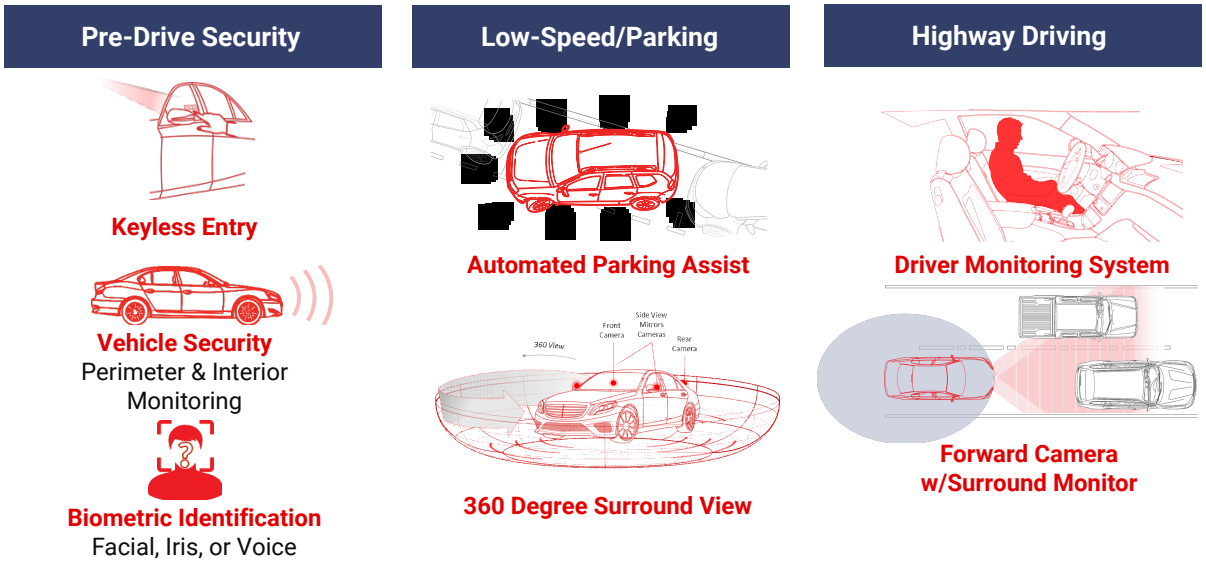
Figure 1: **Alveo Data Center Accelerator Cards are Powered by DFX**

Automotive

Driver assistance features operate at different times, depending on the current mode of the vehicle. Xilinx SoC solutions can take advantage of this by time multiplexing the programmable logic, loading in specific functions only when they are needed. This silicon resource efficiency leads to smaller footprints for both area and power consumption.

For example, before the vehicle is started, a Zynq® UltraScale+™ MPSoC can power pre-drive security functions, such as keyless entry, perimeter monitoring, and biometric identification. After the vehicle is started, the MPSoC changes context to enable surround-view cameras for assisting in or performing automated parking routines. Once the vehicle reaches normal driving speeds, the device can switch to modes involving forward-cameras for lane departure warnings, LiDAR sensors for proximity detection, and driver-monitoring capabilities to ensure the driver remains awake and alert.

Figure 2: **Automotive Applications: Prime Candidates for DFX Applications that Swap Functions Based on Driving Context**



Because many of these modes are mutually exclusive, DFX offers the advantage of loading functions as the context switches, more efficiently using programmable logic resources. Cameras enabled can be sensitive to forward or reverse gears, or to the speed of the vehicle. Software upgrades can be delivered to the vehicle, offering new or updated capabilities long after the vehicle is rolled off the assembly line. The flexibility of a single programmable device is on full display in this application.

Wired and Wireless Communication

DFX offers many benefits to the wired and wireless communication markets. Network switch applications can expand their capabilities by loading different traffic protocols on demand, providing service as needed. Multi-channel switches can change service on a single channel on-the-fly, while service continues uninterrupted on all other channels. When conditions change, partial bitstreams for new protocols can be delivered to deployed systems, extending the lifespan of the product.

Wired or wireless testers can leverage DFX to expand their range of coverage. Similar to switch applications, different test applications can be implemented and swapped within a single tester. Even if the application does not need to be loaded on-the-fly, the DFX design flow provides advantages. The infrastructure of the silicon design can remain fixed while different applications are implemented in a dynamic workspace. This allows design reuse techniques including abstract shell to improve compile times through the Vivado tools. Coupled with the incremental design flow, changes can be turned around very quickly.

A Legacy of Technological Silicon Advancements

While early Virtex® FPGAs were partially reconfigurable, the solution remained primarily reserved for research purposes until the partial reconfiguration (PR) feature was released as a product in the ISE® software suite. This solution supported Virtex-6 and 7 series architectures with a few key advancements that brought PR to a broad customer base. These fundamental silicon features laid the foundation for future generations and included:

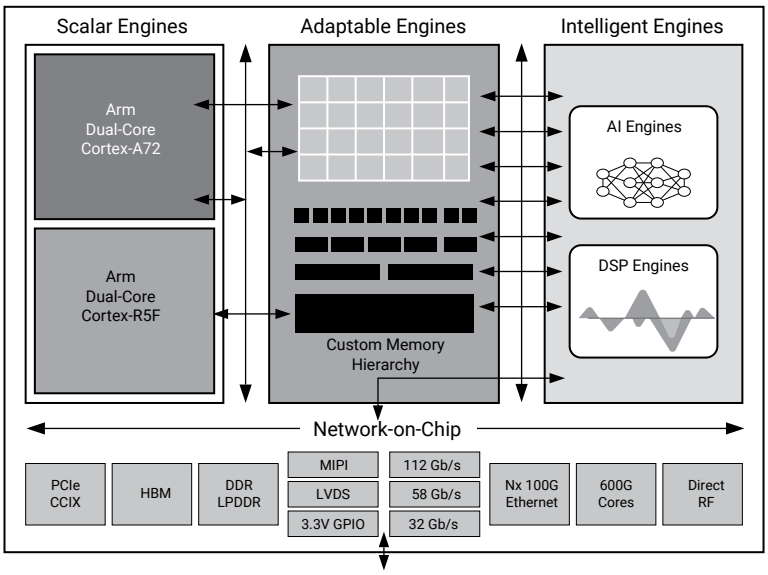
- Dedicated initialization. All partial images can be delivered with a dedicated initialization event that brings all sequential logic to a predictable initial state, while the remaining static logic is unaffected. No local reset or start-up routines are necessary—just load and go.
- Glitchless static routing. Any routes for the static design, from clocks and other global signals to local interconnect, can freely pass through a dynamic region with no risk of glitching. This provides great opportunities for efficiency for place and route algorithms.
- Validation of incoming partial images. The devices in the Xilinx 7 series introduced per-frame cyclic redundancy checks (CRC) ensuring that each incoming partial bitstream arrived intact. Any corruption within a delivered partial image is rejected by the configuration engine, ensuring the operating design is not disturbed and recovery events can be done without requiring a full reboot.
- Hardware-based reconfiguration management. The internal configuration access port (ICAP) allows you to interact with the configuration engine from the programmable logic design. FPGAs can be self-reconfiguring by determining conditions within the hardware design itself.

A great leap forward in silicon evolution occurred with the 20 nm UltraScale™ family, which introduced fine-grained dedicated reconfiguration initialization. Designers could reconfigure areas as small as a pair of CLBs, and automatically have the new logic initialized after reconfiguration, all while the rest of the device continued to operate seamlessly. UltraScale devices also added dynamic reconfiguration of clocking, I/O, and transceiver resources, and provided higher performance dynamic reconfiguration modes—delivering partial bitstreams at rates of up to 800 MB/s. UltraScale+™ devices (16 nm) fine-tuned these features, expanding to include reconfiguration via Quad SPI or BPI flash. This permitted the reload of user application images for the *Tandem PCIe® with the Field Updates* configuration mode, a feature called *reconfigurable stage twos*. Zynq UltraScale+ MPSoCs and RFSoCs use all 16 nm silicon capability with function exchange management handled by software running within the processor subsystem.

The evolution to the present day brings dynamic reconfiguration of Versal devices. Versal introduces the (re)programmable NoC that allows greater flexibility for design layouts; these high-speed connections throughout the device can be declared as static or dynamic or a mixture thereof. Versal devices add faster and more varied primary and secondary boot modes. You can swap large functions in just milliseconds from shared local DDR memory or over a PCI Express® link, delivering programming images at rates up to 6.4 GB/s. Firmware running in the central platform management controller (PMC) dictates all DFX events from driver reloads and event sequencing to delivery of the partial images that reprogram the programmable logic and NoC with new functionality.

Integrated blocks in Versal devices not only bring high-performance Ethernet, Interlaken, and crypto IP, but they allow the core programmable logic to be reserved as a large contiguous user application space that can be reprogrammed with new accelerated functions. Shared DDR memory and high-speed transceivers bring efficiency and performance to all designs, and these resources can remain operational during a transition or be reconfigured to supply new functionality on-the-fly. Finally, the Intelligent Engines that run high-performance DSP applications compiled through the Vitis™ tools can be reloaded with new kernels to accelerate or reconfigured with new compute engines.

Figure 3: **Versal Device Layout Example**



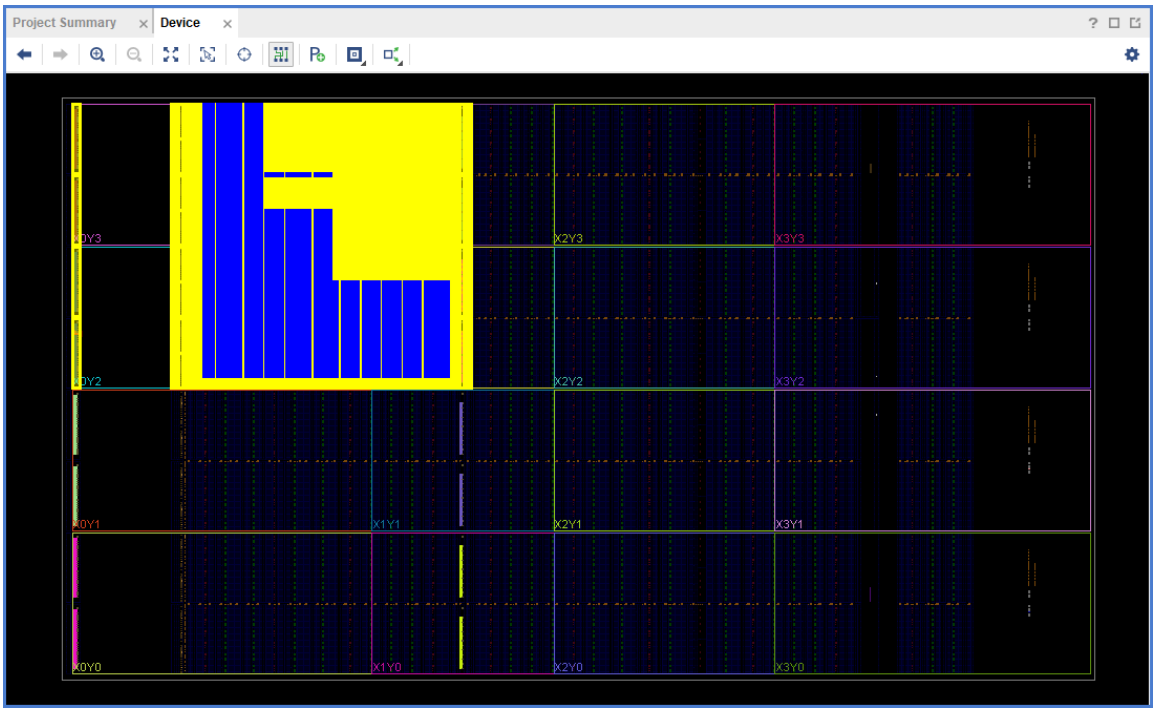
Evolving Vivado Software Design Capabilities

Expanded Xilinx silicon features required growth in software flows to support them. The transition from ISE tools to the Vivado Design Suite enhanced access to DFX features and improved design performance. The development of IP for DFX gives designers the needed access to hardware-based controller management, intelligent isolation decoupling, and safety and security monitoring during partial reconfiguration. Other Vivado tool features advance the performance and efficiency for most fundamental dynamic designs. The following sections describe these advancements.

Core Implementation Tool Technology

The base DFX silicon technology creates expanded routing regions to improve design efficiencies. While dedicated initialization requires that dynamic region floorplans align to basic programmable unit boundaries to ensure only the newly reconfigured logical elements are affected, the same is not true for routing resources. By expanding the routing footprint around the dynamic region, overall routability is improved. Fewer corners reduce localized congestion, and the extra breathing room at the edges give the Vivado tools router freedom to find solutions. While this technique slightly increases partial bitstream sizes, it also results in faster compile times and improved design performance because Vivado tools are bound by fewer restrictions.

Figure 4: **Reconfigurable Partition Floorplan Showing Region for Placement (Blue) and Routing (Blue and Yellow)**

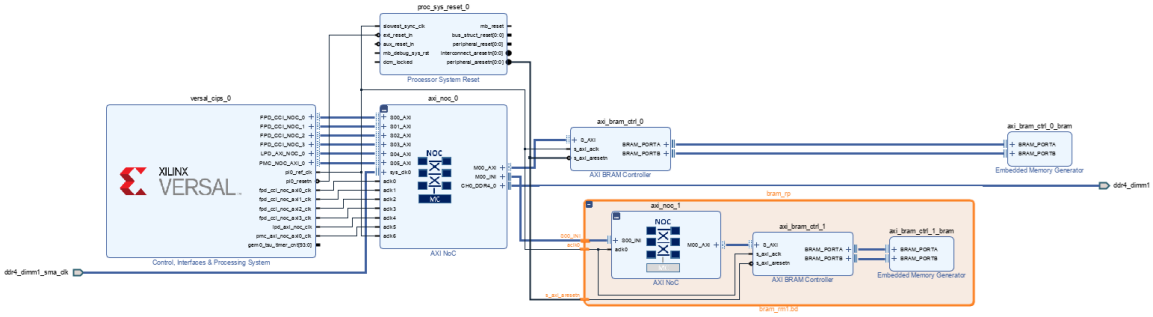


Block Design Containers for IP Integrator

The higher levels of abstraction and automation needed by designers can be satisfied by using the Vivado IP integrator, where complex system designs are created by instantiating and interconnecting IP cores from the Vivado tools IP catalog onto a design canvas. Block design containers were introduced in Vivado tools 2021.1 enables DFX designers to build dynamic systems for any architecture by working at this higher abstracted level.

Block design containers (BDC) permit a block design to be placed within a block design, creating a hierarchical representation of a system that aligns with the fundamental structure for DFX. Containers can hold multiple sources, each representing a different reconfigurable module that can be loaded on-the-fly. The IP integrator BDC solution grants flexibility for addressing apertures and port boundaries, allowing designers to insert a broad range of accelerated functions with different performance and connectivity requirements. Block designs can also be shared amongst different projects, enhancing design reuse methodologies. After block designs are created and validated, the IP integrator solution continues a robust DFX implementation flow to place and route all the design configurations, complete with cross-configuration analysis to ensure hardware compatibility.

Figure 5: **Block Design Containers allow Definition of Dynamic Regions in IP Integrator**

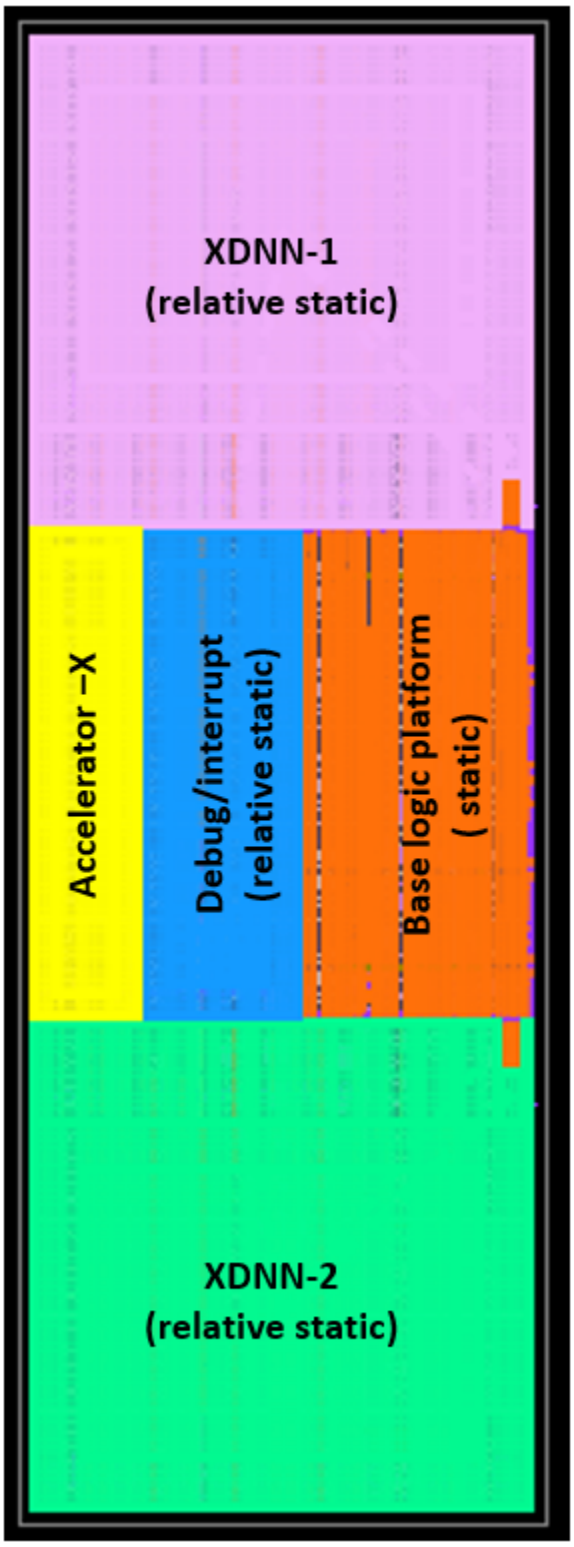


Nested DFX

Expansion of the flexibility of a dynamic workspace is a fundamental yet powerful capability in the DFX design flow for UltraScale and UltraScale+ device designs. Designers can decide if a single large dynamic region is the best option for a complex accelerated function, or if multiple smaller dynamic regions make the best use of the silicon resources available. Nested DFX enables designers to subdivide a dynamic workspace into smaller second-order reconfigurable partitions, each independently reconfigurable. Floorplans for sub-regions can even change, giving tremendous flexibility to the smaller workspaces. They can also switch back and forth on demand to meet current application requirements.

In the following platform example, the base platform on the middle of the right side is configured upon power-up and remains active while other parts of the device are reconfigured. Each of the remaining modules for the two XDNN modules (top and bottom), the debug module (center) and user accelerator module (left) can be independently reloaded while everything else remains running. Alternately, other than the static region, the entire design can be completely swapped for a new design structure with a different lower-level floorplan, with new submodules that are independently reconfigured.

Figure 6: **Dynamic Accelerator Platform using Nested DFX**

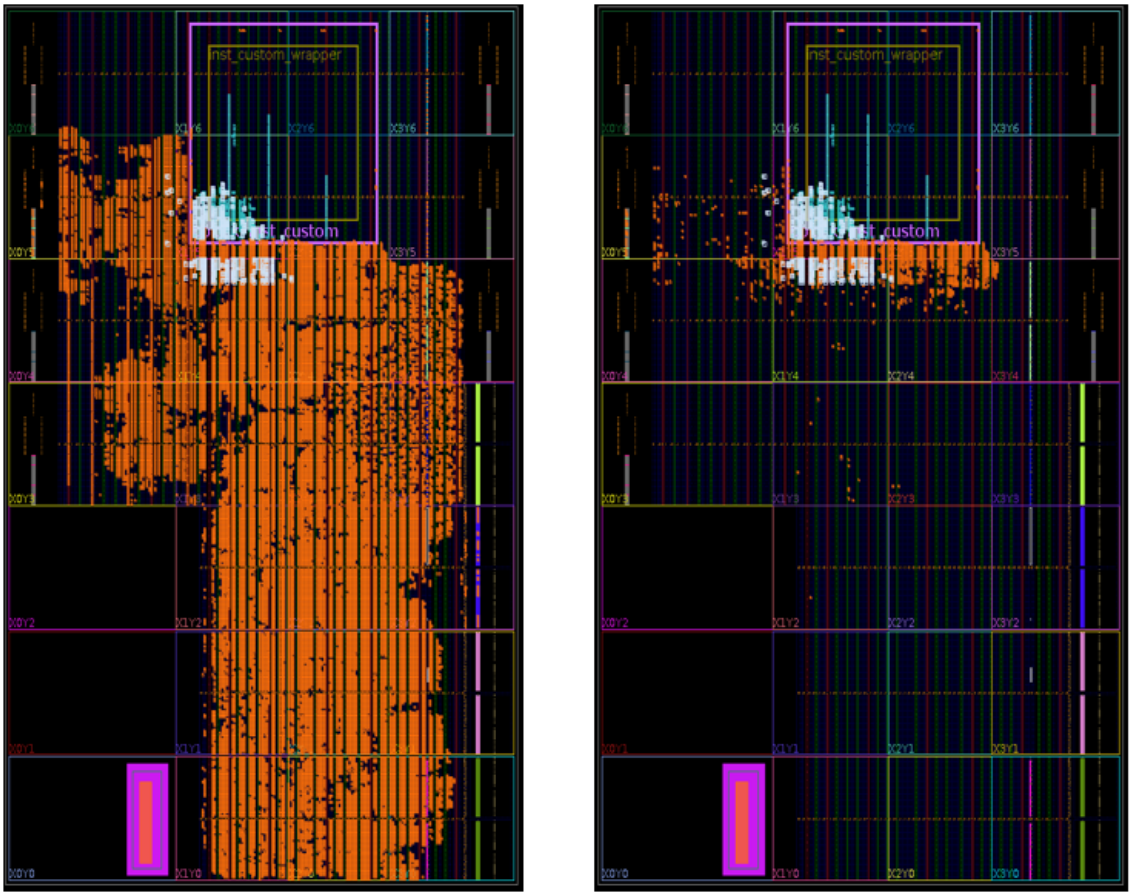


When using nested DFX, hardware acceleration platforms can offer a variety of workspaces with differing resource counts. They can also update platform capabilities without forcing a reboot of the card or a recompile of accelerated function images. When platform details are dynamic, they can be isolated from other parts of the solution to enable field upgrades for bug fixes or new features, or to change end-user requirements like the depth and type of memory access required for a particular application. Platform developers can deliver new features to a deployed system and their customers do not have to recompile their applications to match this modified base design.

Abstract Shell for DFX

Another Vivado tool solution that enhances the standard DFX design flow for all UltraScale+ devices is called abstract shell. In the standard flow, full static design checkpoints must be loaded in memory for the Vivado tools to have the proper context for implementing new dynamic functions. Abstract shells reduce the static design information to the bare minimum—only design elements and constraints that are necessary to create this static context in the shell. By eliminating all superfluous design data, static databases are much smaller and load more quickly into memory, and place and route algorithms are not distracted by the parts of the logical design without permission to modify. Typically, design runs using abstract shells compile three times faster, but can exceed 10X for designs with larger static regions; these designs can also see a 10X improvement in memory usage compared to the standard DFX flow.

Figure 7: **Full Static Shell (Left) vs. Abstract Shell (Right)**



When large portions of the static design are removed from the design database, proprietary design details are not contained in the abstract shell. For multi-user design environments, this means that platform design checkpoints can be safely shared with other groups without the risk of exposing design secrets. The secondary user is simply presented with an empty workspace and a virtual interface to connect to their application. Dynamic functions are compiled through to bitstream generation and loaded into a target device that is first programmed using a full device bitstream supplied by the primary user.

***Note*:** For further information on the abstract shell flow, see *Solution Efficiencies for Dynamic Function eXchange Using Abstract Shells* (WP533).

Conclusion

Dynamic Function eXchange is a powerful and capable feature within the broad product offerings from Xilinx, empowering more efficient and flexible solutions across all markets. Versal ACAPs are the ultimate dynamic platform, combining a powerful processing environment with a robust programmable logic architecture. DFX expands this capable platform by giving it the real-time flexibility required by the most demanding solutions. For more information on this feature, see the DFX page.

References

These documents provide supplemental material useful with this guide:

1. *Vivado Design Suite User Guide: Dynamic Function eXchange* (UG909)
2. *Vivado Design Suite Tutorial: Dynamic Function eXchange* (UG947)
3. *Solution Efficiencies for Dynamic Function eXchange Using Abstract Shells* (WP533)

Revision History

The following table shows the revision history for this document.

Section	Revision Summary
7/28/2021 Version 1.0	
Initial release.	N/A

Please Read: Important Legal Notices

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