Achieving the Highest Levels of Integration in Programmable Logic

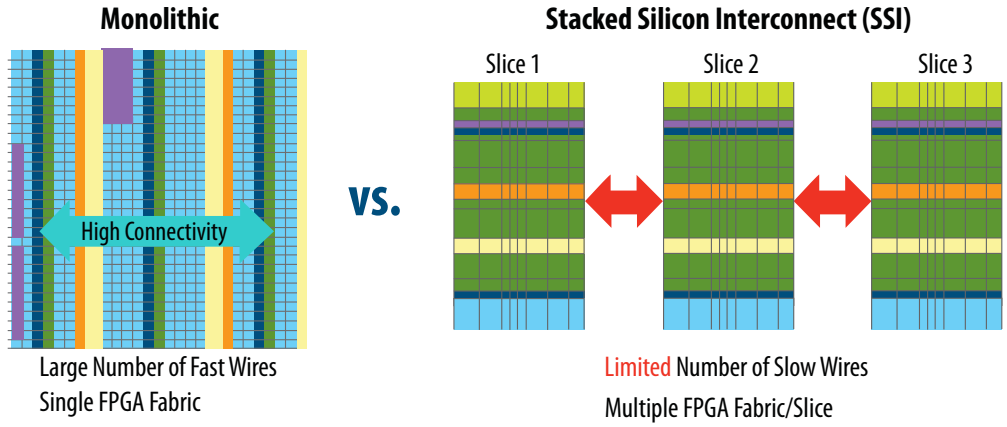
Rama Venkata, Product Marketing Manager, Altera, Now Part of Intel
Martin Won, Senior Member of Technical Staff, Altera, Now Part of Intel
Manish Deo, Senior Product Marketing Manager, Altera, Now Part of Intel

WP-01258-1.0 White Paper

There are two key trends in the programmable device industry. The largest FPGA devices available today easily exceed a million logic elements. Additionally, the need to increase system-level performance, and reduce PCB footprint has given rise to the need to combine diverse semiconductor functions, such as processors and memory, into a single device package. Some of these devices are monolithic, and some are multi-die solutions. In the future, multi-die solutions will become more common, so it is important to understand their impact on designs. This paper describes well-known performance advantages of single-die (monolithic) FPGAs, but also elaborates on a more subtle issue—design partitioning—that has a big impact on time to market.

Heterogeneous integration keeps the FPGA fabric monolithic, which is then integrated with other dice, each with dedicated functions, such as transceivers and memory. Homogeneous integration breaks the FPGA fabric into multiple dice (also referred to as die tiles). The latter approach uses the Stacked Silicon Interconnect (SSI) approach to connect the die tiles—stacking here refers to horizontal stacking of the die on an interposer. See Reference 1 for more details. These two methods are shown in Figure 1. Homogeneous is a reference to identical components—Slices 1, 2, and 3 in Figure 1are identical.

Figure 1. FPGA Fabrics—Monolithic Versus SSI

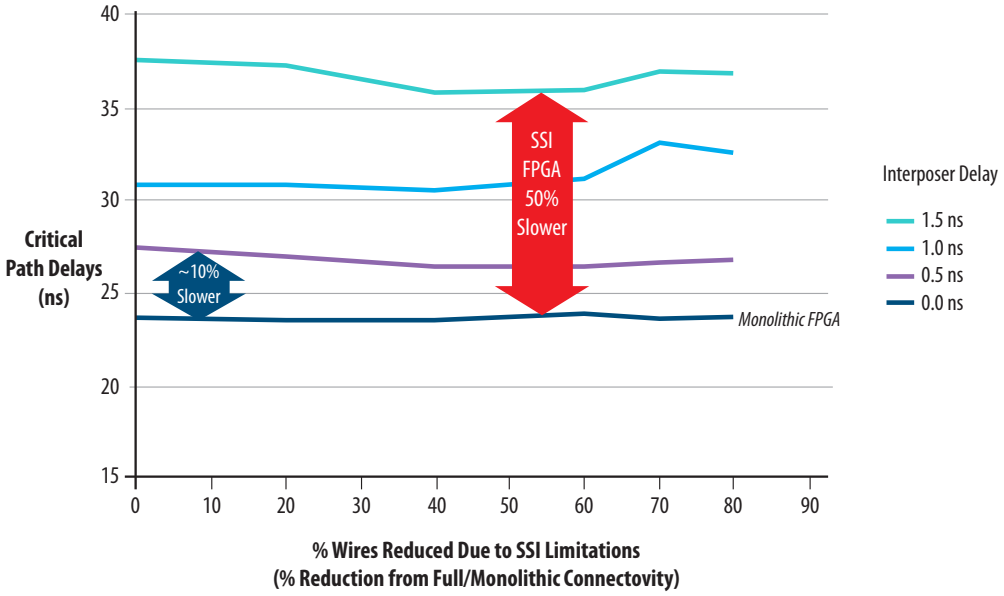


Impact on Performance and Critical Paths

A monolithic FPGA core fabric is critical to provide maximum performance and utilization, and ensure that data can be processed at the highest rates possible without running into routing congestion, utilization bottlenecks, or degraded performance. To understand this, look closely at the physical structure of multi-die FPGA modules. The inter-die paths have significantly longer delays than intra-die paths. The results of research by researchers at the University of Toronto and University of Sao Paulo show up to 50% performance degradation for a 1.5 ns interposer delay relative to a purely monolithic solution. The results, depicted in Figure 2, shows the critical path delay with different - 0.5, 1.0, and 1.5 ns—interposer delays experienced by heterogeneous fabric FPGAs that attempt to connect logic using conventional interposers.

If the fabric is partitioned between die tiles, when those delays are on critical paths there will be serious impact on f_{MAX}. Note that a path may become critical when routed between die tiles, even if it wasn't critical before. Figure 2 also shows that place-and-route tools cannot compensate for these longer inter-die delays, even when the tools modify the cost functions to account for the delays, and even for when the connections between dice are 80% of that of a monolithic FPGA.

Figure 2. 50% Performance Degradation for SSI Fabric



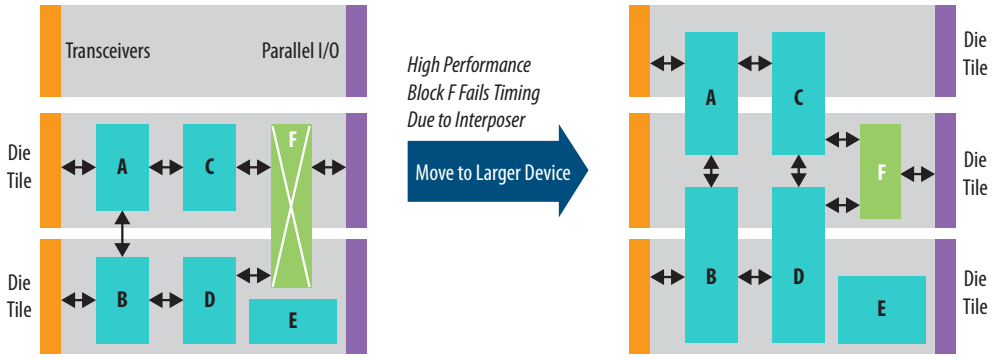
The top line represents critical paths in SSI fabric FPGAs that are available on the market today. 1.5 ns is the approximate routing delay through the interposer. Traversing a full SSI device with four die tiles from top to the bottom requires pipelining for performance greater than 125 MHz. This is because in addition to three hops of 1.5 ns via the interposer, and the usual routing delay within a tile, the clock skew across the die tiles also degrades performance down to 125 MHz. The bottom line in Figure 2 represents a monolithic fabric, which has no interposer delay.

Partitioning Challenges of SSI FPGA Fabric

While the previous section reviewed the performance degradation of using SSI FPGA fabric, it is also important to point out the partitioning challenges. Conventional wisdom holds that manual partitioning—so no potentially critical paths cross a die boundary—can solve this problem. However, conventional wisdom also holds that such partitioning also results in degradation in utilization. An ideal auto-partitioning tool would partition a user design into the multiple dice without any performance or utilization degradation. But, while EDA tools have made significant strides in this direction, real-world experience with tools dictates that there is a trade-off between utilization and performance with partitioning. As a result, significant manual intervention is required to partition away from the SSI die crossings.

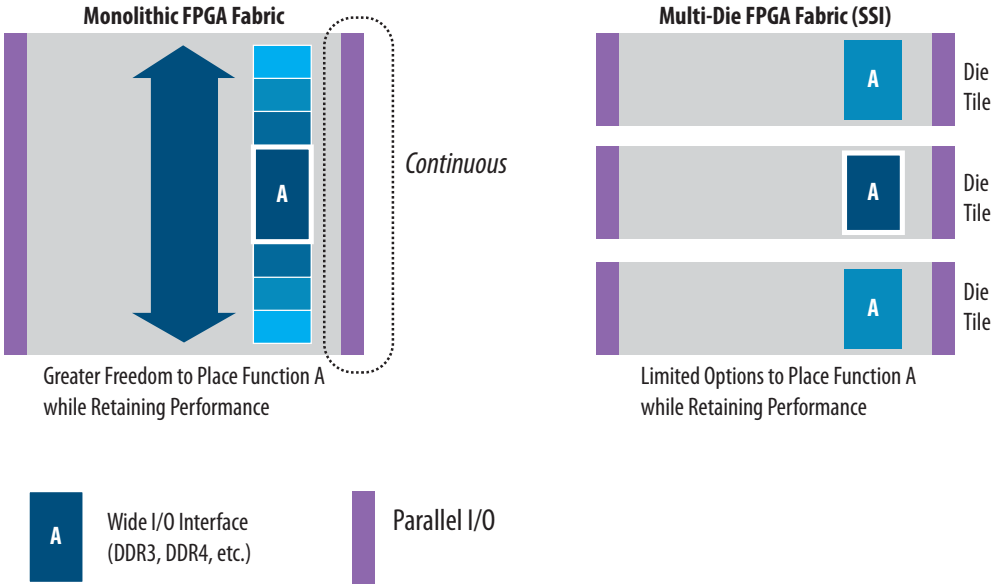
Figure 3 shows that such manual partitioning to avoid the 1.5 ns plus delays across the interposer can not only result in lower utilization, but also migration to a larger device. A typical real world situation depicted here shows blocks A and B can only have fixed placement due to connectivity requirements with transceivers on the left. Block F could be a high-performance memory controller interfacing to a memory interface on the right. Restricted signal sharing with SSI FPGAs dictates that in most cases memory buses need to all be connected to a particular die tile. This restriction constrains placement of interface blocks (F) into a single die tile. The twin goals of meeting performance and connecting to the I/O on the right can force a move to a larger device

Figure 3. Partitioning Decreases Utilization in SSI Devices



Note that the general purpose I/O, which is also configurable for double data rate (DDR) memory interfaces such as DDR4, is continuous in a monolithic fabric as shown in Figure 4. These interfaces that range from 100 MHz to 2.6 GHz are much more compliant to technology scaling making them easier for embedding in the monolithic fabric. This offers a much more flexible connection to fabric logic as shown in Figure 4. SSI FPGAs limit placement of critical interfaces blocks (such as F in Figure 3).

Figure 4. Monolithic Architecture Provides Flexible and Optimal Placement Options for Wide I/O Interfaces



The time-to-market delay due to manual partitioning ranges from one month for a simple design to three months for a complex design.

With performance preservation, utilization degrades by 15% to 25% due to the significant discontinuity in the SSI FPGA fabric. Figure 3 shows significant amounts of unutilized area in the fabric if hard boundaries are used to constrain the blocks within a tile.

Alternatively, performance degrades by up to 50% even for moderate utilization (greater than 70%) with SSI FPGAs. The challenges are especially acute for wide datapath and high-performance designs.

Figure 5. Inefficient Clustering into a Die Tile

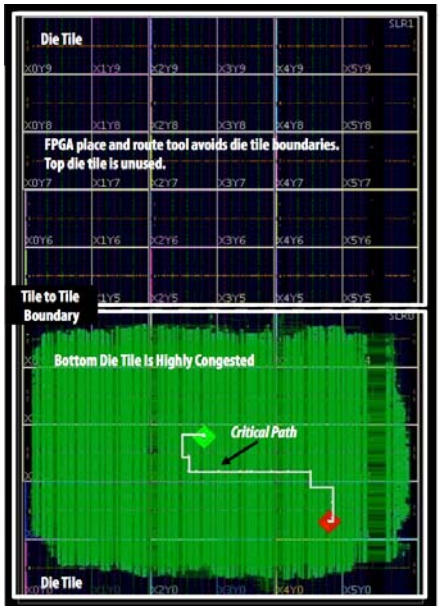


Figure 5 shows an example of the inefficiencies of SSI partitioning. In this example of a military application, the tool has constrained the design into one tile. The utilized tile has high utilization of 90% plus, while one tile is completely unutilized. The overall utilization is 45%. This creates hot spots on the die, and thus degrades performance, and decreases reliability. Additionally, the board pinout of this device is further complicated with strong clustering on the lower half of the package.

Figure 6. Critical Path Crisscrosses Die Tiles

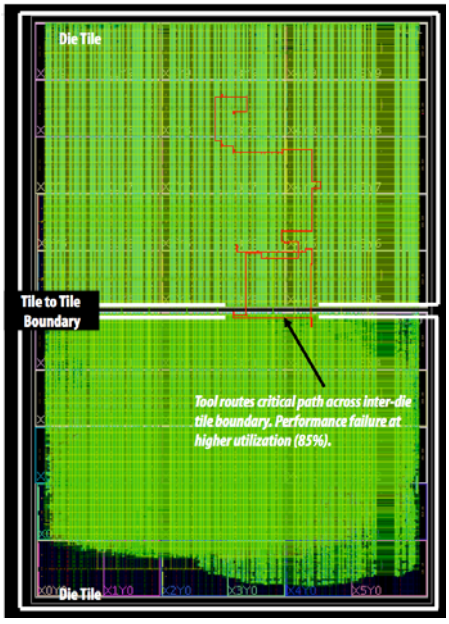


Figure 6 shows an example of a telecom router design where the critical path crisscrosses the die tile boundaries in an SSI device resulting in a performance drop of 50%. In this particular example the utilization was 85%. The same design on a monolithic fabric (Arria® 10 1.15 million LE device) had similar utilization (88%), but with twice the performance.

Optimal Integration—Transceivers and FPGA Fabric

A critical distinction should be made for stream-oriented data transfers from transceivers and bulk memory to fabric versus "Fabric to Fabric". Figure 7 depicts transceiver tile blocks connected to FPGA fabric via Intel's patented, state-of the-art Embedded Multi-die Interconnect Bridge (EMIB).

Transceiver-Tiles-to-FPGA-Fabric Transfers

These transfers (shown in Figure 7) are restricted to a limited set of modes. Not only are the data bus widths fixed and known prior to FPGA silicon tapeout, but also all the data transfer modes are thoroughly simulated, and tested in silicon. Furthermore, Altera's Quartus® place-and-route flow can prioritize (with Quartus assignments) these transfers in the place-and-route process over intra-fabric transfers. Finally, EMIB (see reference 1 for more details) connectivity avoids the issues of through-silicon vias used in traditional multi-die integration, resulting in signal integrity and performance comparable to monolithic technology.

Figure 7. Transceivers to Fabric Transfers

Fabric-to-Fabric Transfers

Intra-fabric transfers are random in nature—for example: they can either be buses, or control signals of complex state machines. The performance of these signals is known only after place and route, where the efficiencies of a uniform, monolithic fabric add a clear advantage.

Conclusion

SSI or homogeneous integration splits the FPGA fabric across multiple dice resulting in lower utilization, and lower performance while also increasing both time to market and market risk.

Altera's Arria 10 device family built on a 20 nm process offers a speed grade advantage over competing offerings. Similarly, the Stratix® 10 device family—built on a 14 nm Tri-Gate process technology and using the new HyperFlex™ core architecture–derives performance advantages from heterogeneous integration and delivers 2X performance gains on average compared to previous generations. One of the drivers of this performance advantage is the monolithic FPGA fabric technology in these device families. Monolithic fabrics maximize utilization and performance, while avoiding time consuming and manual floorplanning.

Heterogeneous integration applies multi-die technology for integration of transceiver die, or memory die with a monolithic FPGA fabric in a single package. The optimal process technology can be chosen for the functions implemented on each die resulting in higher performance, and faster time to market.

References

- Enabling Next-Generation Platforms Using Altera's 3D System-in-Package Technology:
  www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/wp/wp-01251-enabling-nextgen-with-3d-system-in-package.pdf
- Intel Custom Foundry EMIB:
  www.intel.com/content/www/us/en/foundry/emib.html

Document Revision History

Table 1 shows the revision history for this document.

Table 1. Document Revision History

Date	Version	Changes
February 2016	1.0	Initial release.