Prototyping ASICs Using Xilinx FPGAs and Certify™

by Jeff Garrison, Senior Product Marketing Manager, Synplicity, jeff@synplicity.com

The high capacity and high performance of Virtex FPGAs combined with the unique partition-driven synthesis algorithms in Certify means that even the largest multi-million gate ASIC can be prototyped with a manageable number of Virtex devices.

There are an increasing number of applications that can no longer be verified using software simulators alone. The time required to simulate a large data processing application such as video with a traditional HDL simulator is no longer practical. Furthermore, many applications need to achieve specific speed targets for the prototype to be useful. For this type of application there is a strong movement in the market toward prototyping ASICs using FPGAs. As Xilinx celebrates 15 years of success in the programmable logic market, the latest, high-density Virtex family offers an ideal solution.

ASIC designs are often prototyped on a single FPGA device, and Synplify is the ideal implementation tool for these designs. However, for very large ASIC designs, there is a need to partition the design across multiple FPGA devices. One big problem with this emerging prototype methodology is that the time and expertise required to partition a design across multiple FPGAs is very high. Designers need to make multiple time-consuming iterations between synthesis and partitioning to find a "legal" partition for the design. In addition, synthesis is typically done without understanding the partitioning, resulting in a prototype which does not run at the target speed.

Synplicity has enabled a powerful prototyping methodology through a new tool called Certify. Certify takes a fresh approach to the prototyping problem by integrating partitioning and logic synthesis. By performing synthesis during the partitioning phase, you are presented with accurate area and pin utilization information so that partitioning decisions can be made in a fast, interactive environment. A second benefit is that Certify performs partition-driven synthesis. In other words, the physical partitioning data is used to affect synthesis along with your timing constraints.

Certify is the only tool that combines multi-chip partitioning with RTL synthesis. The features and benefits of Certify are summarized in Table 1 below.

Certify's Approach is Better

The level of abstraction and automation in FPGA design tools is moving continually upward to keep pace with the new high-complexity, high-performance FPGAs being introduced. One common element that is critical in linking these new tools together, in the deep subμ world of FPGA design, is logic optimization and technology mapping-the two main steps involved in logic synthesis.

Table 1. Certify Features & Benefits

FEATURE	BENEFIT
Best-in-class FPGA synthesis	Optimal speed and efficiency of prototype
Partition-driven synthesis	Manages time budgets across FPGAs
RTL Partitioning	Partition user's HDL code, not ASIC netlist; Easier to use, higher productivity
Partitioning aids and impact analysis to guide user	Shortens time to prototype
Fast, accurate feedback on I/O and area utilization	Reduces iterations between partitioning & layout
Million+ gate capacity	No need to break up design into many small blocks
Manages Logic Replication	Optimal partitioning without changing source code
Manages Probe Point Creation	High observability for debugging without changing source
Optimized For Iterative Design	Reduces time to fix errors
Understands characteristics of prototype board	Optimal speed and efficiency of prototype
Supports popular FPGA vendors	Flexibility in implementation choice

In today's HDL-based design methodologies:

- The HDL source describes the function of the device.
- The first synthesis step (compilation) interprets the HDL in terms of registers, logic and arithmetic operations, and control circuits.
- Logic optimization attempts to reduce the complexity of the implementation using sophisticated algorithms. To this point, the design has remained independent of the implementation device architecture (FPGA device family).
- Technology mapping represents the design in terms of a specific device architecture.
- Timing optimization attempts to improve the design to meet the designer's timing constraints.
- Placement locates each technology cell within the chip.
- Routing allocates the wiring resources to provide the necessary interconnections.

The naïve assumption that each of these steps is independent leads to non-convergent iteration loops in reaching a timing goal. At each stage, the tools (or the designer, in the case of HDL source) make implementation decisions based on estimates of what will happen in the remainder of the flow. Each following stage is stuck with the decisions made at earlier stages. In the independent model, the placement stage cannot change logic to make its job easier.

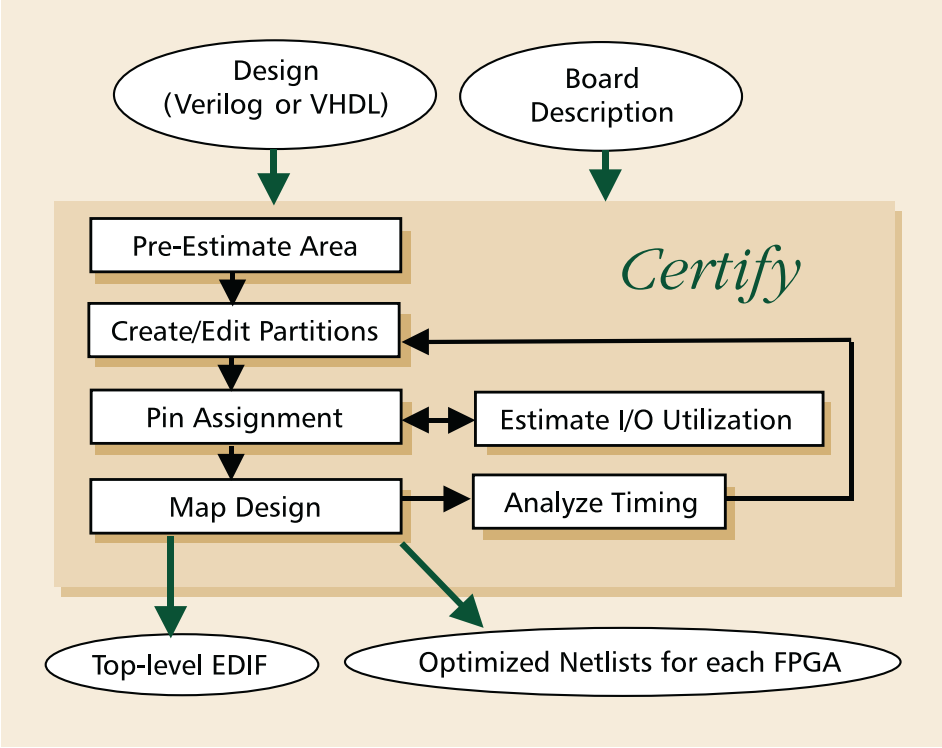
The fundamental limitation to this flow is that changes made in early stages have the most impact, but the ability to estimate is weakest there, while the later stages have the best information, but the ability to make significant changes is gone. To overcome the limitations of the flow, we must improve estimation for the early stages, and preserve high-level information for the later stages. This approach is called partition-driven synthesis. It is at the heart of the powerful new algorithms in Certify, and allows you to achieve the fastest possible prototype speed.

The Timing Estimation Problem

If improved timing estimation is necessary to break the endless iteration cycle, then we must first understand the components of timing, and what optimization tools can do to change the timing. With FPGAs of 100K gates or more, good estimation requires an understanding of the underlying architecture of the device. The speed of an FPGA design depends on many factors:

- The delay in the simplest programmable logic cell.
- The delay introduced by the programmable routing.
- The interconnection of logic to form a critical path through the circuit.
- The timing requirements of the external circuitry.

The delay in the simplest programmable cell is easy to estimate once the design has been mapped to a particular device architecture, but very difficult to estimate before mapping. The cell is intentionally very rich in its ability to represent logic functions, but the cells' capabilities vary greatly among FPGA device families. Hence, good estimation requires mapping to the chosen device architecture.



The speed of an FPGA design depends on many factors ...

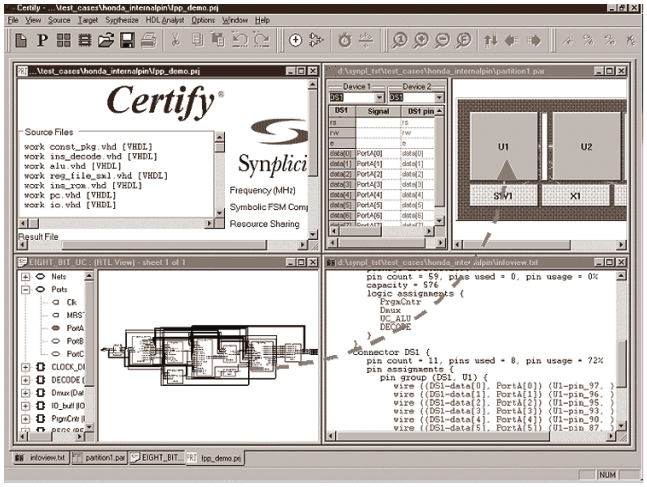
Continued on the following page

Of course, the most effective changes are major changes to the circuit topology that derive from a deep understanding of the design. This is the technique used by Certify's unique partition-driven synthesis algorithms to produce optimal results.

In a typical FPGA, the sum of the routing delays along the critical path comprises well over half of the total delay. In a mapped design, routing delay is by far the most important estimation. The delay due to a particular interconnection depends on the capacitance and resistance of the interconnection, which are in turn dependent on the physical location of the connections. It also depends on the routing architecture for the programmable device, and on the size of the particular device. Good estimation of routing delays depends on good estimates of physical placement, and on a detailed knowledge of the device architecture.

Estimating timing accurately becomes even more important when dealing with designs partitioned across multiple FPGAs. The board delays must be accurately taken into account in estimating any critical paths, and the synthesis tool must account for these delays in its optimization algorithms. The static timing analysis algorithms for finding the most critical paths, and the techniques for considering the external timing requirements are complex but well understood. Given a mapped design with good interconnect estimates, static timing analysis will reliably identify the most critical path.

Timing optimization tools have only a few options to improve the timing:



Partition by dragging and dropping HDL modules to different FPGAs

- Change the circuit topology of the critical path so that the new critical path has fewer delays.
- Change the placement so that it is easier to route with less resistance and capacitance.
- Change the routing to reduce resistance and capacitance.

Of course, the most effective changes are major changes to the circuit topology that derive from a deep understanding of the design. This is the technique used by Certify's unique partition-driven synthesis algorithms to produce optimal results.

Certify follows in its predecessor's (Synplify) footsteps for providing an intuitive and easy to use user interface.You input Verilog or VHDL code for the design along with a PCB description (also in HDL). Certify then performs a fast estimation of area and pin utilization for the HDL modules in your design. This utilization information is instantly provided to you as you drag and drop modules of HDL in the various FPGAs in the system.

After a partion has been created, Certify uses the physical partition to drive the synthesis process. Certify has the ability to optimize logic across FPGA boundaries resulting in higher performance and better overall device utilization. The output of Certify is a top-level EDIF description of the system, and highly optimized netlists for each of the FPGAs, ready for Xilinx place and route tools.

Conclusion

By utilizing Synplify's lightning-fast synthesis algorithms with Behavior Extracting Synthesis Technology™, Certify has the ability to handle huge designs. Even a prototype that requires six XCV1000s (6 Million FPGA gates) may be partitioned and fully synthesized in a few days as opposed to several weeks, not to mention the better results achieved with Certify by synthesizing across FPGA boundaries.

For more information on Certify, please contact your local Synplicity sales representative or call (408) 548-6000. Σ