WP230 (v1.1) May 16, 2007

Physical Synthesis and Optimization with ISE 9.1i

By: David Dye

Advances in process technology have led to dramatic increases in FPGA device densities. Several Virtex™ devices that exceed 100,000 logic cells. The increase in device density and the use of 300 mm wafers have made FPGAs affordable for volume production.

Designs that were exclusively targeted at ASICs in the past are now being implemented in programmable devices. The largest 65 nm Virtex-5 devices contain over 300,000 logic cells, 11 MB of block RAM, and over 600 DSP blocks. Creating a design to efficiently use the available resources in these devices and meeting performance requirements can be challenging. Fortunately, today's EDA software tools have evolved to effectively meet these challenges.

Introduction

Logic optimization, logic placement, and minimizing interconnection delays are important in achieving maximum performance. Timing-driven synthesis technology has provided a significant improvement in design performance. However, its effectiveness is limited by the accuracy of estimating the routing delays. Physical synthesis, the use of physical placement and routing information during synthesis, has been at the forefront of effectively addressing the accuracy issues.

Physical synthesis and optimization, within the place-and-route software, further expands on timing-driven synthesis technology by involving synthesis in the implementation decisions after the netlist is generated. This allows for dynamic reexamination of synthesis mapping and packing decisions, based on actual placement and routing information during implementation.

Benefits of Physical Synthesis and Optimization

Interconnection delays among levels of logic are affected by proximity of placement for the logic elements, routing congestion, and local competition between nets for the fastest routing resources. To avoid these delays, revisit synthesis decisions during the mapping, placement, and routing phases of design implementation. During the mapping phase, the netlist can be reoptimized, packed, and placed based on the criticality of individual timing paths. This approach reduces the number of implementation cycles required for timing closure.

How Do I Use the Physical Synthesis and Optimization Flows?

ISE™ software by Xilinx provides several options to enable the physical synthesis and optimization process. These options can be used individually or together, depending on the specific needs of your design. The flows and options described here are applicable to the following families (unless otherwise noted): Virtex-5, Virtex-4, Virtex-II and Virtex-II Pro as well as Spartan-3A, Spartan-3AN, Spartan-3E, and Spartan-3 devices.

Define Timing Requirements

The most important step for effective physical synthesis is to set up accurate, comprehensive timing constraints. With these constraints in place, the implementation tools can make more informed decisions that improve the overall results. The clocks and I/O pins that have firm requirements can be constrained to allow the rest of the design to have relaxed constraints.

The easiest way to define these timing constraints is to use the Constraints Editor. This graphical tool allows you to enter clock frequencies, multicycle and false path constraints, I/O timing requirements, and a host of other clarifying requirements. Constraints are written to a User Constraint File (UCF), which can be edited in any text editor.

If user-defined timing constraints are not provided, a new feature introduced in the 8.1i software automatically generates timing constraints for each internal clock. Named the Performance Evaluation Mode, this feature enables users to get the high performance results of physical synthesis and optimization without having to provide timing targets.

Run Global Optimization

For designs containing IP cores or other netlists, the NGD file that is available after the Translate (NGDBuild) phase of implementation represents the first time the entire design has been completely assembled. Global Optimization, a new feature available since the 7.1.01i version of Map, takes this fully assembled design and attempts to improve the design performance by re-optimizing the combinatorial and register logic. Global Optimization (`map -global_opt on` for command line users) can increase design clock frequencies by an average of 7%.

Two other options allow you to further control the optimization that is done during this phase: Retiming (`map -retiming on`) moves registers forward and back to balance combinatorial logic delays, and Equivalent Register Removal (`map -equivalent_register_removal on`) removes registers with redundant functionality. The Global Optimization switches are available for Virtex-4 and Virtex-5 devices only.

Enable Timing-Driven Packing and Placement

Timing-Driven Packing and Placement (`map -timing`) is at the heart of the physical synthesis capabilities available within the implementation flow. When this option is enabled, the placement phase of place and route is done within Map, allowing packing decisions to be revisited when initial results are less than optimal. This iterative flow eliminates the notion of unrelated logic packing.

Different levels of optimization are available in the Xilinx physical synthesis and optimization process. The first level was introduced in ISE 6.1i and began with logic transformations, including fanout control, logic replication, congestion control, and improved delay estimation. These routines led to much more efficient packing and placement of designs, resulting in faster clock frequencies and denser logic use.

The next level added basic logic and register optimization; Map was given the ability to rearrange elements to improve critical path delays. These transformations provide much greater flexibility in meeting the timing requirements of the design. A number of different techniques, including pin swapping, basic element switching, and logic recombination, are used to massage the physical elements into a different, yet logically identical, structure that meets the design requirements.

Another option for Virtex-4, Virtex-5, Spartan-3A, and Spartan-3E devices is Combinatorial Logic Optimization (`map -logic_opt on`). This switch performs advanced logic optimization routines beyond simple logic recombination. The combinatorial logic is reexamined with placement and timing considerations, and more extensive manipulation can be done.

When Timing-Driven Packing and Placement is enabled, users can control whether or not Register Duplication (`map -register_duplication`) is performed. Users can also set placement options using the Effort Level (`map -ol std|med|high` and `map -xe c|n`) and Cost Table (`map -t <1 to 100>`) options. See Table 1.

Table 1: **Summary of Physical Synthesis and Optimization Switches**

	Global Optimization Flow	Timing-Driven Packing and Placement Flow
Supported Families	Virtex-5 and Virtex-4 FPGAs	Virtex-5(1), Virtex-4, Virtex-II, Virtex-II Pro Spartan-3, Spartan-3A, Spartan-3AN, and Spartan-3E FPGAs
Switch	-global_opt [on | **off**](2)	-timing
Related Switches	-retiming [on | **off**]	-ol [std | med | **high**]
	-equivalent_register_removal [**on** | off]	-xe [c | **n**]
		-t [1-100]
		-register_duplication [**off** | on]
		-logic_opt [**off** | on]

Notes:

1. -timing is always enabled for Virtex-5 devices.
2. Default values are in BOLD.

Which Options Should You Use?

The effectiveness of the flows and options listed in this document depend on a number of factors. If the synthesis tools were not able to operate at maximum efficiency, these implementation options would have more opportunities to make improvements. This situation can occur if you use modular or incremental flows that prevent the synthesis tools from optimizing the entire design at once. Also, if you practice design reuse, and the netlist cores were not included during synthesis, these physical synthesis options would have a great impact.

However, even standard flows can see benefits from physical synthesis and optimization. Users see the greatest benefit by enabling the Timing Driven Packing and Placement option, followed by the Global Optimization, and then the High Effort and the Extra Effort switches. An easy way to examine the impact these switches can have on your design is to use the Xplorer script. Xplorer runs through many combinations of these and other options by scripting multiple passes through the implementation tools. For more information about the Xplorer script, please consult http://www.xilinx.com/ise/implementation/Xplorer.htm.

Some Examples of Physical Synthesis and Optimization

Logic Duplication: If a LUT or flip-flop drives multiple loads, and the placement of one or more of those loads is too far away from the source to meet timing, the LUT or flip-flop can be replicated and placed close to that group of loads, thus reducing routing delays (Figure 1 – Figure 2).

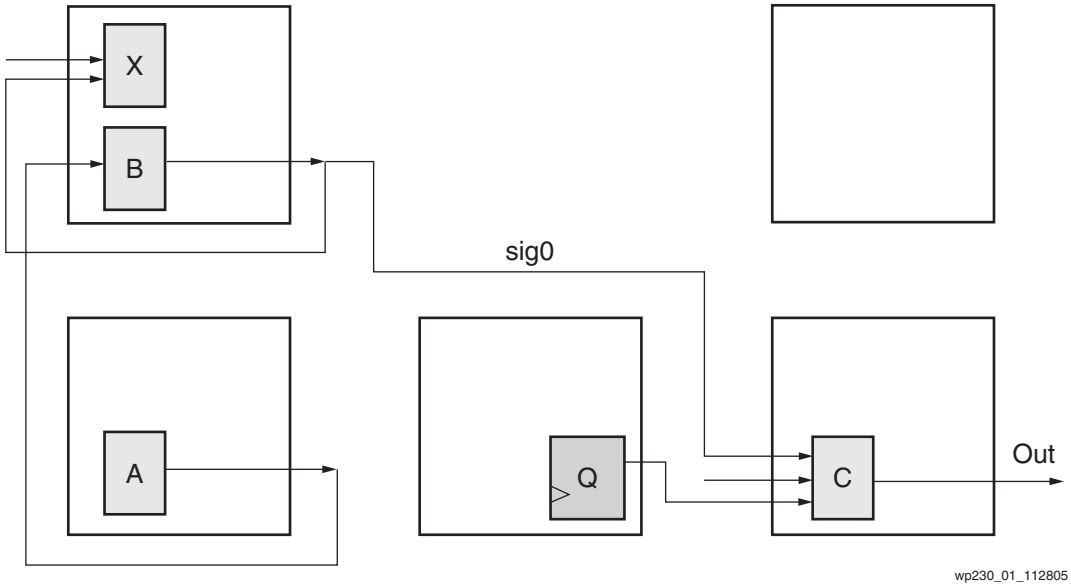

Figure 1: **Logic Duplication: Path LUT A→LUT B→LUT C→Out is Critical**

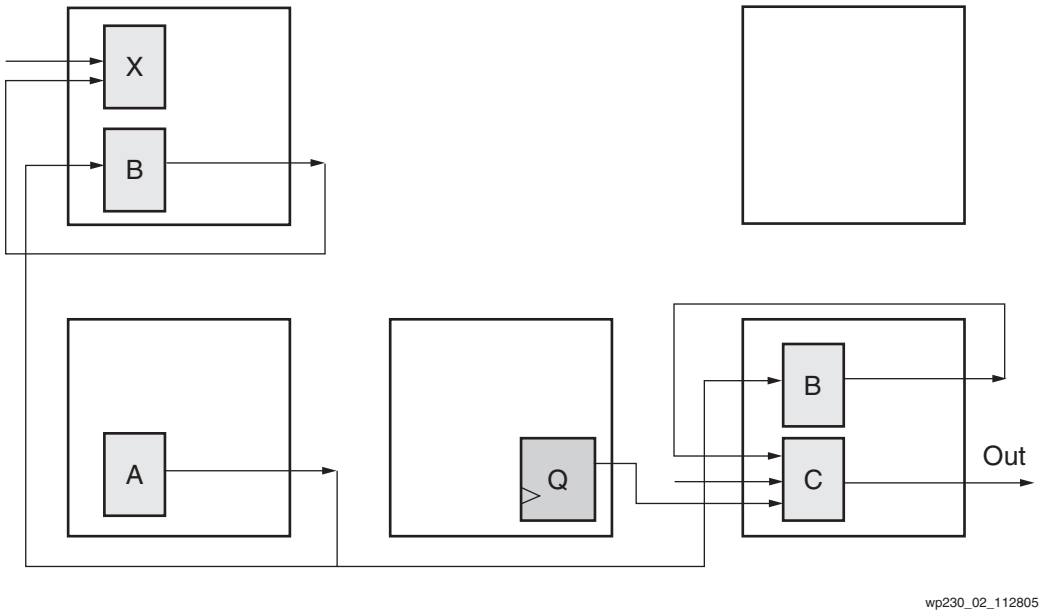

Figure 2: **LUT B is Driving Two Critical Loads - Can be Duplicated to Reduce Path Delay**

Logic Recombination: If the critical path traverses through multiple LUTs, through multiple slices, the logic can be reassembled using fewer slices and using a more timing-efficient combination of LUTs and muxes to reduce the routing resources needed for that path (Figure 3 – Figure 4).

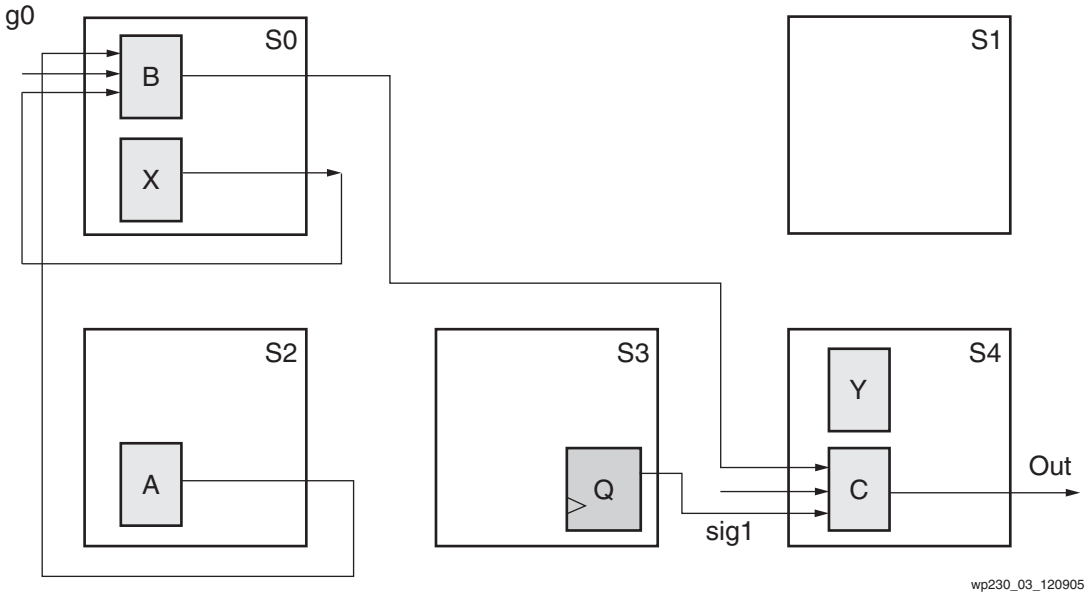

Figure 3: **Logic Recombination: Path LUT A→LUT B→LUT C→Out is Critical**

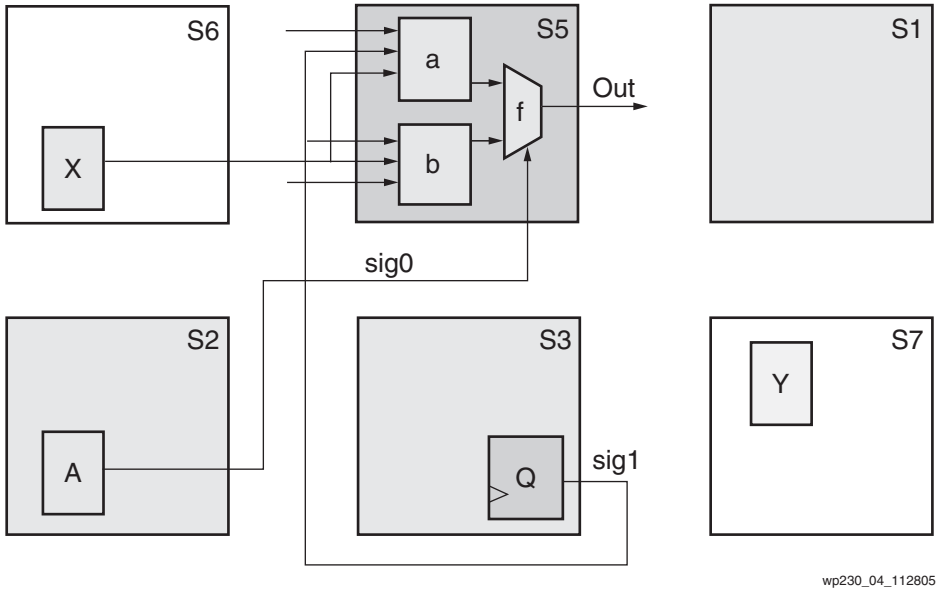

Figure 4: **LUTs B and C can be Combined and Replaced by (LUT a, LUT b, F5Mux f)**

Basic Element Switching: If a function is built with LUTs and MUXs within a slice, physical synthesis and optimization can rearrange the function to give the fastest path (usually through the mux select pin) to the most critical signal (Figure 5 – Figure 6).

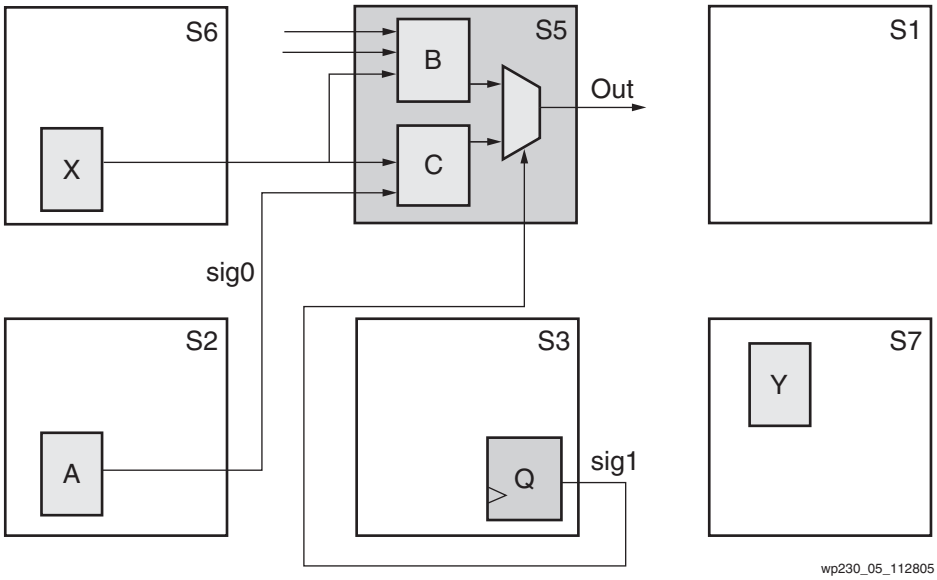

Figure 5: **Basic Element Switching: Path LUT A→LUT C→Mux F5→Out is Critical**

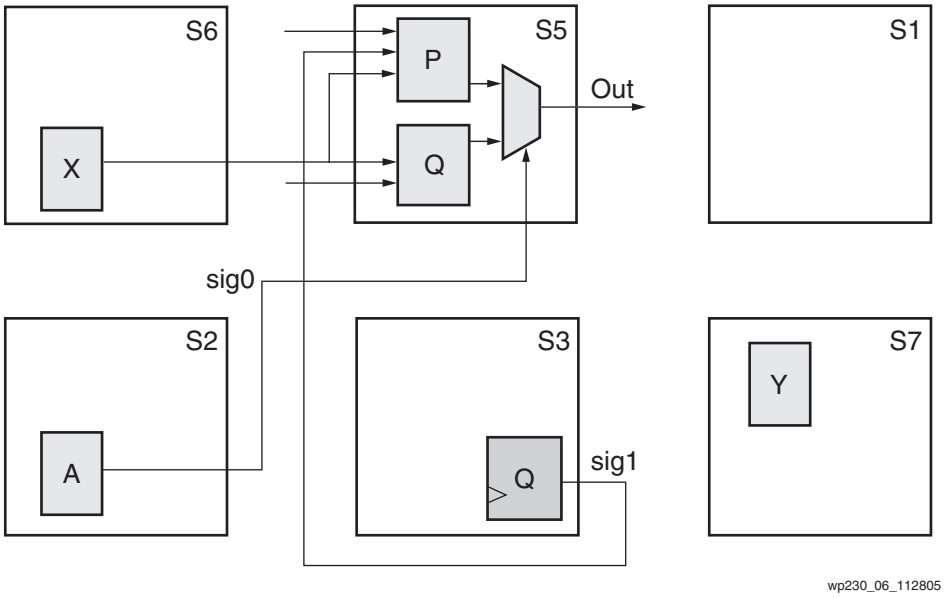

Figure 6: **Path Through Mux Select Pin is Faster than Through LUT**

Pin Swapping: Each input pin of a LUT can have a different delay, so Map has the ability to swap pins (and the associated LUT equation) so that the most critical signal is placed on the fastest pin (Figure 7 – Figure 8.)

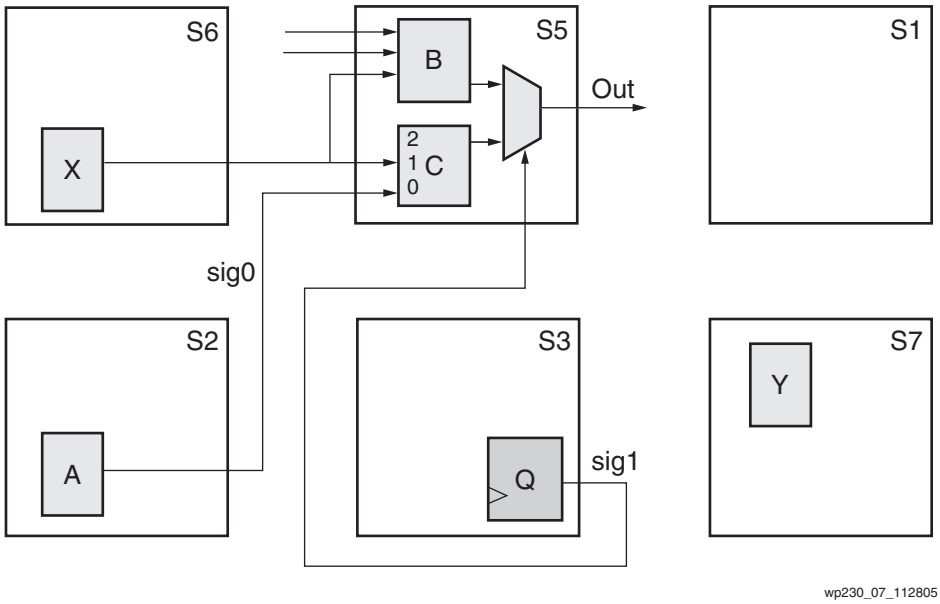

Figure 7: **Pin Swapping: Path LUT A→LUT C→Mux F5→Out is Critical**

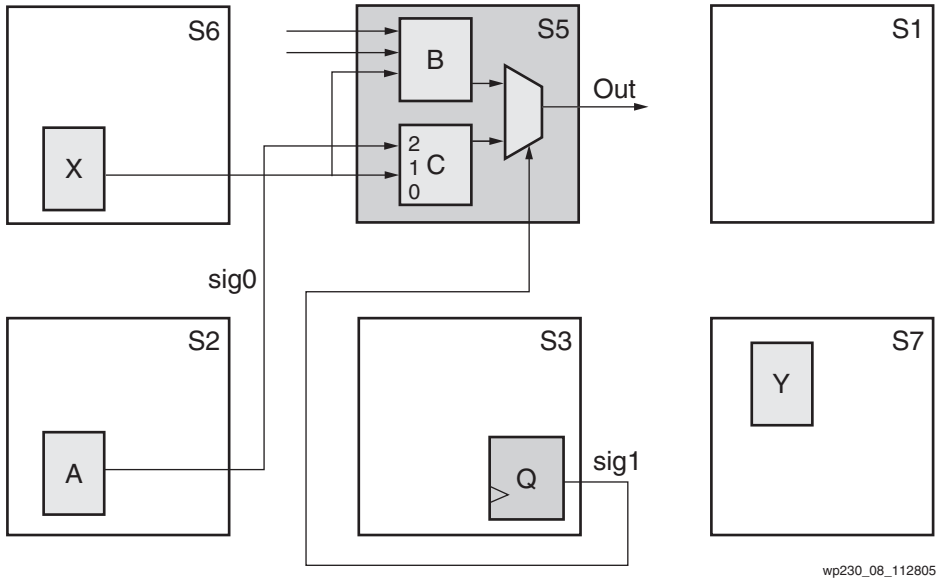

Figure 8: **Pin 2 is Faster than Pin 0 for LUT C; Swap Pins 0 and 2 for LUT C**

Roadmap to the Future

The physical synthesis and optimization capabilities within the Xilinx toolset continue to mature and expand with each software release. Along with an improved quality of results, expect to see greater user control over the types of optimizations that can be done. Planned enhancements include consideration of more design elements in the reoptimization phase (like timing-driven optimization of registers into and out of the I/O blocks or dedicated functions like block RAM and DSP blocks) and the inclusion of the routing phase into the reiterative physical synthesis and optimization system.

Summary

The Physical Synthesis and Optimization tools in the Xilinx ISE software have been created to reexamine the structure of your FPGA design during the packing and placement phases of implementation. With the knowledge of timing constraints and physical layout, optimizing synthesis decisions during Map and Place & Route can significantly improve your results.

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
12/09/05	1.0	Initial Xilinx release.
05/16/07	1.1	Updated for ISE 9.1i (previous version addressed 8.1i) and the latest Virtex and Spartan devices. Added Table 1.