Extension MEDIA
Chip Design
Tools, Technologies & Methodologies
www.chipdesignmag.com June/July 2004

FPGAs Go, Go, Go

Solving the FPGA timing closure challenge for high-speed designs

It is now possible to implement complete radar processing algorithms on a single multi-million gate FPGA. Most of these systems involve FFT processing along with other DSP algorithms. Clearly, FPGAs are coming into their own as devices for complex designs.

Dillon Engineering Inc. (Edina, MN) provides FPGA design services focused on creating DSP and high-bandwidth designs for real-time digital signal and image processing. Our multi-million gate designs include proprietary cores and complex DSP algorithms that comprise EDIF, Verilog and VHDL source, as well as various IP cores. Dillon has implemented several radar processing systems on single FPGAs in the last year, and our success in large part was based on an ability to reuse IP and the reliability of the tools available for its design flow.

Often, our success with these highly complex designs was hard won as we wrestled with severe timing closure problems in the projects caused by interconnect delays. We now know that high speed, multi-million gate designs that can be implemented in FPGAs, but those designs can have significant timing closure problems which can put project schedules at risk. New classes of FPGA design tools and methodologies are clearly required to address these challenges. We believe we've found those tools.

GUIDELINES FOR SUCCESS

At Dillon, our design flows emphasize rapid design implementation—most multi-million gate designs are completed in one or two months—which we achieve by following a few basic guidelines. First, create and reuse parametric IP for as much of the design as is practical. Second, solve timing problems at the architectural level; manual optimizations and floor planning are too time consuming for clients schedules and result in IP that is device specific and less reusable. Finally, use RTL synthesis, place and route, and physical synthesis/optimization to resolve the remainder of the design.

The engineers at Dillon have created all-reusable logic in a parametric form using our in-house ParaCore Architect tool.

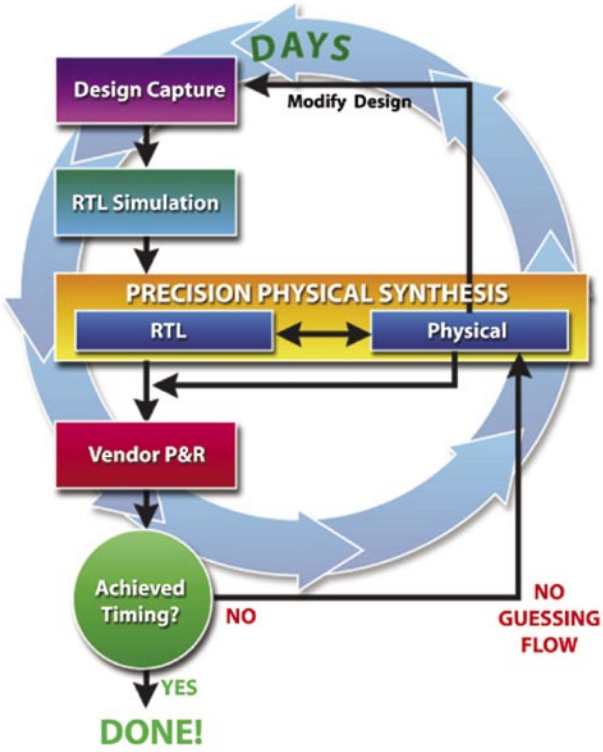

Figure 1: The Precision Physical Synthesis flow

The HDL is auto-generated from the parametric description, providing significant opportunities to reuse functional logic many times over. We also include IP from other sources in designs anytime it will speed the completion of one of our projects. IP from other sources comes in many forms, including VHDL, Verilog, and EDIF. Importantly, such IP must be easy to integrate into our flow or we don't use it.

Solving timing problems at the architectural level means creating logic with sufficient pipeline stages, so the levels of logic between any two registers are kept to a minimum. Most FPGA architectures are register rich. The goal is to decrease the levels of logic between registers as modules are created, and thereby meet clock performance requirements.

Dillon's FPGA flow uses Precision RTL synthesis from Mentor Graphics Corp. (Wilsonville, OR), then ISE place and route from Xilinx, Inc. (San Jose,CA), followed by Mentor's Precision Physical Synthesis to clean up the design and squeeze the highest possible performance from the final design. (see Figure 1)

We have implemented several radar processing systems in single FPGAs. One recent design involved creating a complex FPGA-based digital signal processor for a two-dimensional Doppler radar system developed for military applications. This design posed significant performance challenges, due to its size and because it required Dillon's Fast Fourier Transform (FFT) IP core to operate at a higher frequency than was ever previously attained. The design flow and techniques described here enabled Dillon to meet the client's schedule expectations, and to increase the clock rate high enough to reduce the logic requirements such that the design fit into a device only half as big as had been initially anticipated.

IMPLEMENTING A RADAR-PROCESSING DESIGN

The Doppler radar-processing FPGA we designed is quite complex. The device receives data at 80 MSPS (mega-samples per second) from the A/D, applies a Hanning window, an 8K-point FFT, interleaves data via external memory, applies a 32-point FFT, and finally passes the data to the client's target detection IP block. The 32-point FFT includes a CORDIC (cordinate rotation digital computer) to produce phase and magnitude results. A PCI interface is used for run-time interface and configuration. The samples from the A/D (analog to digital converter) are 80 MSPS of continuous data, so all modules must keep up with a continuous data stream. However, the latency of the results is not critical in this system.

By far the most logic intensive parts of this device are the two FFTs running at 80 MHz and operating on continuous data. Dillon's FFT IP core was used in both instances; the 8K-point FFT requires eight radix-2 butterflies (computed with two nested DO loops), while the 32-point FFT requires four radix-2 butterflies to keep up with continuous data. Our initial estimates to the client indicated that the design would fit into a Xilinx Virtex II XC2V6000-4 FPGA.

The first challenge in any FPGA design is coding the logic and producing correct functional simulation results. In any project, Dillon engineers start by creating a bit-accurate model of the system using our in-house ParaCore Architect tools. Once this model is correct, which usually takes just a few hours of calculation, Dillon gets client approval on the results. Then when logic coding begins, the math-modeled results are used to verify the operation of the logic in module and system-level HDL simulations. By the time the process of coding the logic is complete, a testbench exists to checks the functionality of the full system. Once the logic is functionally accurate, it's time to synthesize the design.

As mentioned, the tools we use for this process are from the Mentor Graphics Precision Physical family of products. Dillon uses Precision RTL Synthesis to provide the synthesis, analysis, and debug interface for performing RTL Synthesis. In contrast to our earlier efforts, we have found that starting and maintaining the project is markedly easier now that the RTL Synthesis methodology has been implemented. Dillon engineers simply choose the target technology, add all source files—this design was comprised of many source files including EDIF, VHDL and Verilog files, as well as constraint files (UCF)—and select the top-level module to complete the work.

Constraints from the UCF are used by RTL Synthesis, while files required for place and route are passed along with the design automatically. In this particular FPGA design, RTL Synthesis allowed the engineers to easily organize and manage design files into a single project of associated files.

All told, a variety of cores were used, including a Verilog PCI bus and an EDIF CORDIC IP core. Before using the RTL synthesis design methodology, we found it was a constant challenge to manage files and ensure that all were incorporated; design management was frequently error prone and tedious. Also before using RTL Synthesis, incorporating external IP into a design was tricky—simulation would go smoothly, but a bottleneck would inevitably occur at the synthesis stage. Now our engineers have no need to synthesize portions of the design separately for merging later on. The full design is always synthesized together and produces great performance results.

Following synthesis, engineers at Dillon use Mentor's Precision Physical Synthesis for physically aware synthesis, a process which takes into account the placement and routing of the design. Physical Synthesis imports a wide range of source file types, reading the netlist from initial synthesis, along with placement and delay information from the vendor's place and route. The tool then performs automated register replication, re-timing, re-synthesis, and placement optimization to incrementally achieve timing closure. We have found that using Physical Synthesis to perform these tasks requires only a faction of the time previously needed for conventional synthesis and place-and-route tools.

RAPID TIMING CLOSURE

Dillon engineers used RTL Synthesis to meet the timing goals of 80 MSPS—all modules in the design had to keep up with

an 80 Mhz continuous data stream. This was easy for the FFT modules since they had all run over 125 Mhz in the past.

However, the Interleaver module was new and needed some source code modifications to meet its timing goals. Our engineers found they were able to identify the gross timing problems in just four RTL Synthesis iterations, and were able to fix the problems before running place and route. The entire process took about 30 minutes, as opposed to the two-plus hours needed for a full run through a traditional place-and-route tool. The RTL process saved the team a total of eight hours in compute time.

Meanwhile, the Dillon team noticed that timing results for the FFT portions of the design were being produced at about 140 MHz, which was quite a bit faster than anticipated. Since the FFTs are the majority of the logic in the design, we considered trying to run the FFTs at 160 MHz, which would in effect cut the logic requirements in half and allow them to fit into a much smaller FPGA.

RTL Synthesis allowed our designers to set up constraints for each clock domain and simultaneously analyze the discrete timing requirements relative to their clock domains. Timing violations and slow paths could then be incrementally corrected without re-running place and route, using automated re-timing, and by placing registers.

Fortunately, our engineers get very good results when allowing RTL Synthesis to do register re-timing, with minimal impact on synthesis runtimes. Register re-timing is a capability, which balances delays between registers to improve performance by moving logic across registers. After applying the RTL synthesis register re-timing feature, the achievable clock speed was increased to 150 MHz.

But we were still 10 MHz short of our goal for the performance of the FFT portion of the design. Dillon often uses Physical Synthesis to get extra performance from a design and did so in this case. The Physical Synthesis tool takes a design following the Xilinx ISE place-and-route phase of the design, and does a final optimization that usually results in a 5-to-15 percent timing improvement—the larger the design the better the improvement. In this case, Physical Synthesis improved the design's performance, and achieved the 160 MHz speed on the first pass, without intervention.

SURPRISES ALONG THE WAY

Dillon engineers were surprised to find that by running the fully automated physical synthesis flow, the post place-and-route performance was further increased to 160 MHz. Precision Physical Synthesis has the ability to replicate registers and perform re-timing automatically. In the project described here, the tool produced results that met the performance requirement six out of ten times. In cases where the results were not quite fast enough, it took about 30 minutes with the Physical Synthesis interactive design environment to make legal physical design modifications in order to achieve the desired timing result.

In interconnect-dominated designs such as the ones that Dillon produces, some types of timing problems materialize only after

Virtex II 3000 Implementation - $1500 cheaper

160 MHZ FFT DEVICE USAGE				
FUNCTION	**SOURCE**	**SLICES**	**MULT 18X18S**	**BLOCK RAM**
8K-point FFT IP – 160 MHz	DE IP – VHDL	4,120	16	40
32-point FFT IP – 160 MHz	DE IP – VHDL	2,100	8	12
2D Interleaver	DE IP – VHDL	1,120	0	0
Hanning window	DE IP – VHDL	810	1	8
Target detector	Client IP – EDIF	2,362	16	24
PCI IP core	Other IP – Verilog	2,300	0	2
	Total	12,812	41	86

Virtex II 6000 implementation

80 MHZ FFT DEVICE USAGE				
FUNCTION	**SOURCE**	**SLICES**	**MULT 18X18S**	**BLOCK RAM**
8K-point FFT IP – 80 MHz	DE IP – VHDL	6,150	32	80
32-point FFT IP – 80 MHz	DE IP – VHDL	3,100	16	16
2D Interleaver	DE IP – VHDL	1,120	0	0
Hanning window	DE IP – VHDL	810	1	8
Target detector	Client IP – EDIF	2,362	16	24
PCI IP core	Other IP – Verilog	2,300	0	2
	Total	15,842	65	130

Table 1: Comparison of device usage for 80 MHz and 160 MHz Virtex II alternative implementations of Dillon's radar-processing design. By enabling the design to be optimally retargeted to an alternative FPGA, Precision Synthesis helped save $1500 per device.

place and route. Previously, Dillon engineers have had to resort to reducing logic manually between registers in order to meet timing requirements. Each change would require place and route to be re-run, an often slow and time-consuming process. We found that the automated flow in Precision Physical Synthesis, performed these operations automatically without the need for multiple place-and-route iterations. Our Dillon team saw improvements of 10-to-20 percent in slack on critical paths.

LESSONS LEARNED

The Dillon design team synthesized the radar-processing design using Precision RTL Synthesis and, after an initial run with the Xilinx place-and-route tools, used Precision Physical Synthesis to generate a placement-optimized design that fully met our performance objectives. With each incremental iteration, the team was able to analyze and incrementally improve the design both in the RTL domain and in the more accurate physical domain. Dillon engineers could quickly spot the timing impact of any changes made in the process.

In the final analysis, our initial implementation of this particular radar processor used a Xilinx Virtex II 6000 device, 50 percent of the available slices, 75 percent of the block RAM, and a substantial number of multipliers on the device. To explore alternatives, the designers then re-targeted the design on a less expensive device, moving it from a slower speed grade, higher-capacity Virtex II 6000 part down, through two part sizes, to a faster speed grade, lower-capacity Virtex II 3000 device. That effort resulted in cost savings of $1,500 per device. Although, achieving this transition required doubling the clock rate for the FFT blocks in order to keep up with the incoming data. This meant incorporating two clock domains in the design, one running at half the speed of the other. (see Table 1)

All told, our work executing the design on both Virtex devices was a success. FPGAs provided all of the required capacity and performance necessary for the radar processor. Mentor Graphics® Precision Synthesis tools added speed and efficiency to the design process, and those tools have become part of our standard flow. Advanced tools from the EDA vendors are making it possible to push the envelope with the increasingly sophisticated FPGA devices available on the market today.

Tom Dillon *is president of Dillon Engineering Inc., which specializes in DSP applications on FPGAs. He has more than twenty years' experience in designing electronic products and is recently credited with having developed a single-FPGA radar processing design and the world's fastest FFT processor.*

Mentor Graphics Corporation
Design Creation and Synthesis Division
1001 Ridder Park Drive, San Jose, CA 95131, USA
Tel: 1.800.547.3000 or 1.503.685.8000
Fax: 1.408.436.1501
Email: synthesis_info@mentor.com
www.mentor.com/synthesis/

Dillon Engineering, Inc.
3925 West 50th Street, Suite 202, Edina, MN 55424, USA
Tel: 1.952.836.2413 • Fax: 1.952.927.6514
Email: info@dilloneng.com • www.dilloneng.com