FPGA Technology Drives Design Software REVOLUTION

By looking at the changing use of FPGAs over time, we can understand the demands for a new generation of FPGA design software.

by Steve Bailey, HDL Solutions Manager, VeriBest Inc., sbailey@veribest.com

Over the last 15 years, Xilinx and the FPGA industry have evolved from providing FPGAs with densities of 100s of gates to devices with over one million system gates. Not only have system designs been adapted to exploit the capabilities of today's FPGAs in contexts not possible 15 years ago, designers have found the need to adopt new design methods and design software solutions that did not exist 15 years ago.

Soaring FPGA Device Complexity

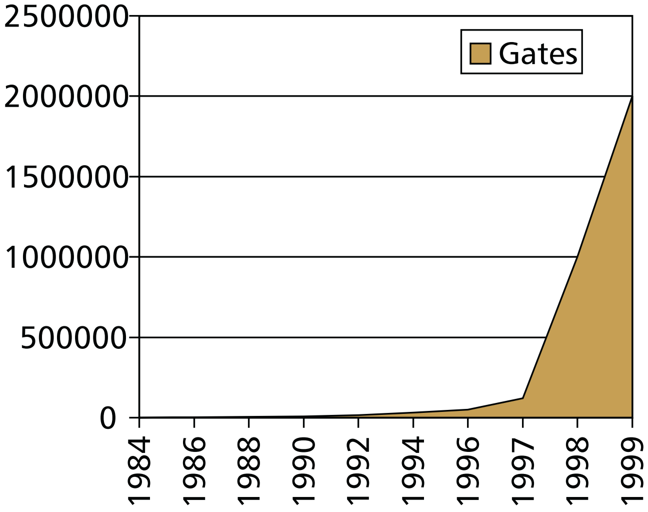


Figure 1

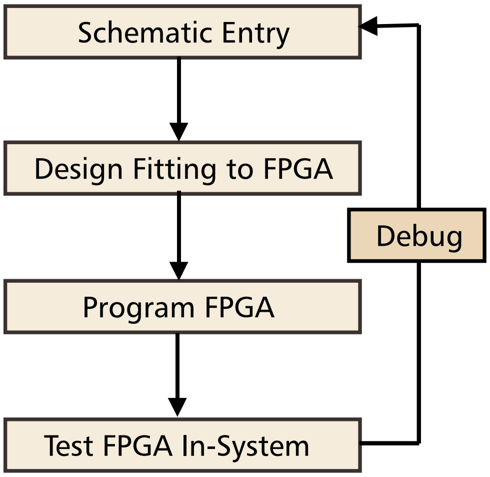
FPGA Usage Then and Now

Fifteen years ago, FPGAs were designed into systems primarily to reduce system component costs by consolidating board-level logic into fewer devices. A few hundred logic gates were replaced by a single FPGA that implemented the same functionality. Because the role of an FPGA was the consolidation of board-level logic into fewer components, FPGAs were designed by the board engineer.

The complexity of today's FPGAs, as shown in Figure 1, allows system architects to replace a broad range of ASICs with FPGAs and further consolidate and integrate the system logic into fewer and fewer components. FPGAs provide you with unprecedented flexibility at attractive costs. The advantage of no non-recurring engineering (NRE) costs, easy design modification, and in-system re-programmability make FPGAs a very attractive alternatives to ASICs. In fact, Ron Collett[1] reports that the number of boards with at least one ASIC has declined to 22% from 45% since 1994. Clearly, FPGAs are replacing more ASICs with each new technology update. As FPGA design complexity increases and more ASICs are replaced by FPGAs, more and more FPGA design is performed by the traditional chip design teams and not by the board engineer.

FPGA Design Methodology Then and Now

When the board engineer was using FPGAs primarily to integrate and consolidate 100s of gates of board-level logic, he used his time-tested board design methodology anchored by schematic capture as shown in Figure 2. It was easy and straight forward to insert a new level of hierarchy into a schematic for the FPGA and push the portion of the board schematic, that was being consolidated into the FPGA, down into the new hierarchical block. The engineer had already done



Simple Design Method for Simple Devices

Figure 2

Complexity Drives Adoption of HDL-Based Design

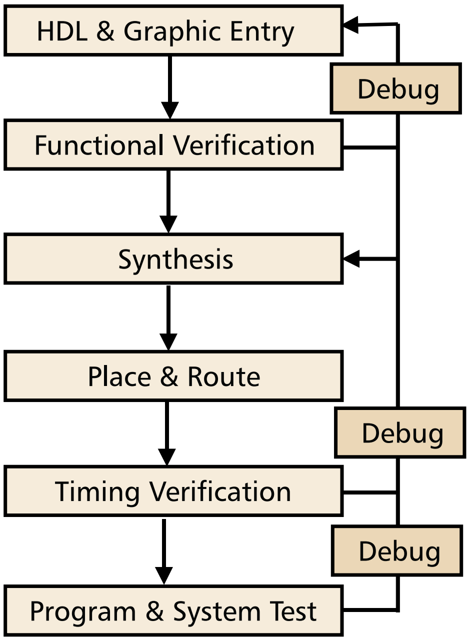


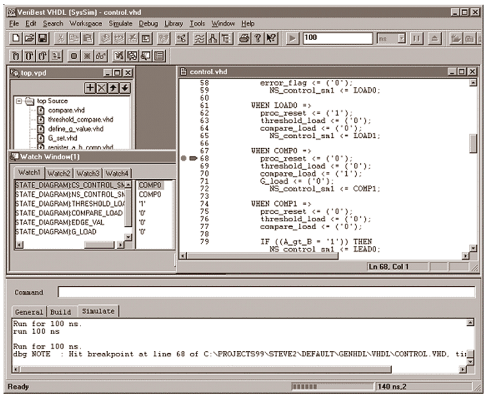
Figure 3

manual logic minimization and captured the design at the boolean logic and macrocell level. The only additional tool needed was a fitter (or FPGA place and route tool) to map the logic design into the FPGA with correct routing. The fitter then generated a chip programming file. Verification often consisted of prototyping.

Fifteen years later, the thought of designing a million-gate FPGA using a schematic design methodology defies rational thought. Today's FPGA designers are adopting HDL-based design methods at astonishing rates. HDL-based design, as shown in Figure 3, increases your productivity by allowing you to work at higher levels of abstraction — the register-transfer level instead of the boolean logic (gate) level.

Central to HDL-based design and the increased size of FPGAs are two strategically important tools: simulation for design verification and synthesis for automatic implementation of the RTL design to the gate-level (FPGA place and route level). Bread-board prototyping falls apart as a practical design verification method due to the cost of debugging functionality after layout. Simulation allows design problems to be discovered earlier when it is more cost-effective to fix them.

Powerful, Fast HDL Simulation Speeds Design Verification



STATE_DIAGRAM}CS_CONTROL_SN	COMP0
STATE_DIAGRAM}NS_CONTROL_SN	COMP0
STATE_DIAGRAM}THRESHOLD_LOA	'1'
STATE_DIAGRAM}COMPARE_LOAD	'0'
STATE_DIAGRAM}EDGE_VAL	'0'
STATE_DIAGRAM}G_LOAD	'0'

```
58            error_flag <= ('0');
59              NS_control_sm1 <= LOAD0;
60
61          WHEN LOAD0 =>
62            proc_reset <= ('1');
63            threshold_load <= ('0');
64            compare_load <= ('0');
65              NS_control_sm1 <= LOAD1;
66
67          WHEN COMP0 =>
68            proc_reset <= ('0');
69            threshold_load <= ('0');
70            compare_load <= ('1');
71            G_load <= ('0');
72              NS_control_sm1 <= COMP1;
73
74          WHEN COMP1 =>
75            proc_reset <= ('0');
76            threshold_load <= ('0');
77            compare_load <= ('0');
78
79            IF ((A_gt_B = '1')) THEN
                NS_control_sm1 <= LEAD0;
```

```
Run for 100 ns.
run 100 ns

Run for 100 ns.
dbg NOTE  : Hit breakpoint at line 68 of C:\PROJECTS99\STEVE2\DEFAULT\GENHDL\VHDL\CONTROL.VHD, ti
```

Figure 4

Schematics and block diagrams still have a role in FPGA design, but that role is limited to manual implementation of tightly constrained functional blocks or to help manage complexity by graphically partitioning the design into smaller blocks.

FPGA Design Software Then and Now

Now that FPGA design is transitioning from the board engineer to the chip engineering design team, and design methodology is transitioning from schematic to HDL, FPGA design software must evolve to meet new needs.

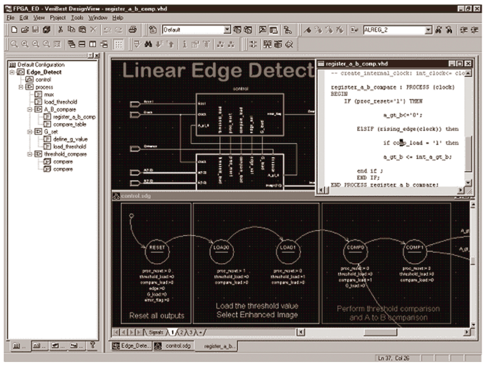
Today's FPGA designers require the power of ASIC design tools within the traditionally, tightly integrated FPGA design environment. Fast, high-capacity HDL simulation gives you the ability to functionally verify FPGA designs at the Register Transfer Level and verify the dynamic timing and functionality after implementation.

The size and functional complexity of today's FPGA designs require powerful simulators capable of handling the traditional needs of ASIC designers. VeriBest's™ HDL simulation products, shown in Figure 4, deliver technology-leading simulation speed and capacity, as well as powerful debugging capabilities, easily handling the largest of today's FPGA designs with fast verification and design debugging turnaround times.

Similarly, synthesis tools deliver productivity power by translating the design from the abstracted RTL to an architecturally optimized implementation. Synthesis tools must be able to handle the same language subset as their ASIC counterparts to facilitate re-targeting designs from ASICs to FPGAs. They must also provide ASIC-type quality of results for a variety of FPGA architectures. VeriBest's FPGA Desktop™ is available with FPGA Express™ from the industry's leading ASIC synthesis provider, Synopsys®. FPGA Desktop also fully supports integration with Synplicity's® Synplify™ for Xilinx designers who find that Synplify better meets their needs.

While FPGA design moves towards an HDL-dominated methodology, mixed schematic and HDL designs will be common for some time. Also, HDL editors are not always the best way to capture certain functional blocks. State diagrams are commonly employed to capture state machines. Similarly, many designers find flowcharts, truth tables, and state tables a more natural means to capture certain types of functionality. Finally, graphical design specification often results in design blocks that are more easily reused. Therefore, neither schematic-only nor HDL-only design capture environments are sufficient to support today's FPGA designs. That''s why VeriBest's FPGA Desktop, shown in Figure 5, provides a rich set of design capture editors including

Continued on the following page



```
-- create_internal_clock: int_clock<= clo

register_a_b_compare : PROCESS (clock)
BEGIN
    IF (proc_reset='1') THEN

                a_gt_b<='0';

        ELSIF (rising_edge(clock)) then

                if comp_load = '1' then

                a_gt_b <= int_a_gt_b;

        end if ;
        END IF;
END PROCESS register_a_b_compare;
```

Graphical and HDL Entry in FPGA Desktop

Figure 5

schematic/block, HDL editor, state diagram, truth tables and state tables, flowcharts, and boolean equations.

Although today's FPGAs need powerful design software, you still require the productivity of the traditional FPGA design software solutions, but you need them extended to teams of designers; FPGAs still give you time-to-market advantages.

Powerful point tools address one dimension of your productivity and your ability to meet project and market schedules. Tool integration, design data, and flow management address the remaining dimensions. All of the tools within the FPGA Desktop environment are tightly integrated including third-party tools such as FPGA Express, Synplify, and the Xilinx Foundation Series place and route software. Design kits support each Xilinx device family providing additional productivity benefits. Tight integration provides you with a single environment to learn and master.

Design data and flow management are the catalysts for the productivity benefits of the integrated FPGA Desktop environment. FPGA Desktop not only understands the tools in the integrated flow, but it also manages the design data for you, whether you are working alone or as part of a team.

Data management is both user explicit, as with the automated management of the design block hierarchy and source files, and user implicit. FPGA Desktop implicitly manages the design data and flow by preparing each step in the design flow, such as creating the synthesis tool's project, configuring synthesis, and passing the synthesized netlist to the Xilinx Foundation Series software for place and route. By automating routine data management tasks, you are free to focus on the engineering specifics of executing each step in the design flow such as constraining synthesis or specifying block functionality.

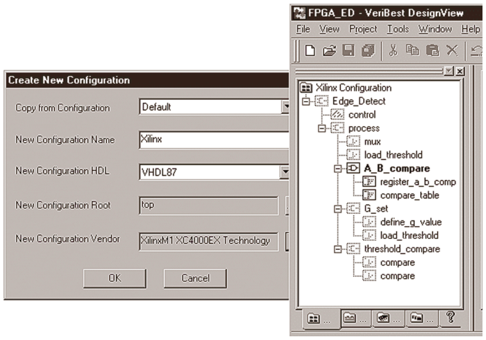
Consistent with the need for a powerful design solution, FPGA Desktop passively manages the design flow by facilitating the transitions between steps in the flow. It does not constrain you by actively managing your activity within a proscribed flow.

To explicitly enable teams of designers, FPGA Desktop allows the definition of multiple configurations for partitioning designs across multiple designers. You can also identify any block in the hierarchy as your design root for use as the focal point for each design step such as simulation and synthesis. Figure 6 shows the creation of a new design configuration with a focus on the portion of the design starting with the block A_B_Compare.

Conclusion

As Xilinx continues to lead the industry it helped create 15 years ago by driving FPGA device size and capability forward, FPGA (and ASIC) designers are evolving to fully exploit the new generation of FPGA devices by replacing more ASICs with FPGAs. As FPGAs replace ASICs, FPGA design is moving from board engineers into chip design teams.

Both new and experienced FPGA designers find that adopting HDL design methods helps them meet their tight time-to-market requirements while designing ever larger and more complex FPGAs. The combination of these FPGA technology and design trends increase the need for FPGA design solutions that provide tools powerful enough to handle ASIC designs while also delivering the productivity of an integrated FPGA design flow. VeriBest's FPGA Desktop delivers the power and productivity that today's and tomorrow's teams require, for designing with Xilinx devices.



Create New Configuration

Copy from Configuration: Default

New Configuration Name: Xilinx

New Configuration HDL: VHDL87

New Configuration Root: top

New Configuration Vendor: XilinxM1 XC4000EX Technology

Configuring Designs and Managing Hierarchy

Figure 6

[1] *"ASICs Not What It Was," Ronald Collett, <u>Design Technology ROI</u>,* ***EE Times****, 1 February 1999, page 45.*