MINC's Upgraded PLSynthesizer Supports

by Greg Brown,
Sr. Product
Marketing Manager,
gregb@synario.com

MINC recently released a substantial upgrade of its leading edge, VHDL and Verilog synthesis product for programmable IC devices, PLSynthesizer. The new release, version 6.1, adds support for the XC4000XL and XC4000XV high-density devices from Xilinx. Runtimes for the new version are three to five times faster with vastly improved quality of results, which mean faster circuitry that occupies less space.

Key features of PLSynthesizer v6.1 include:

- Advanced VHDL/Verilog synthesis including support for IEEE and Synopsys synthesis subsets and coding styles
- Specific technology optimization for AREA / SPEED criteria with several optimization efforts
- Hierarchy handling
- Automatic resource sharing
- Technology-specific macro inference
- Automatic state machine recognition/extraction
- Several automatic state encoding algorithms
- User constraint-based synthesis and optimization
- Technology re-targeting flow
- Support of the most popular FPGA and CPLD/PLD devices
- Comprehensive and easy-to-use GUI
- Windows 95/NT and Unix support

Advanced Operator Inference Engine and Macro Generation

Operator inference refers to the procedure of identifying elements of the design that may be implemented as macro blocks rather than as more complex and costly glue-logic (Boolean equations). To achieve the highest quality of results from an HDL design, PLSynthesizer includes an extremely powerful operator inference engine that can use a built-in macro generator or Xilinx LogiBLOX. Operator inference is the procedure by which a macro is recognized within a behavioral HDL description and implemented in a technology-specific and therefore efficient implementation.

```
...
signal A,B,C,O : std_logic_vector (7 downto 0);
signal S : std_logic_vector(1 downto 0);
...
with S select
        O <= A+B when “01”,
             A when “-0”,
             B-C when others;
```

Figure 1. Operator Inference and Resource Sharing

Further, using LogiBLOX and its low-level NGD output netlist ensures you of the best operator implementation possible. This is very important on data-path and timing critical designs where inefficient operator synthesis may require you to resort to handcrafted, schematic-based designs to achieve your design goals. The operators that can be inferred by PLSynthesizer are: Adder, Adder/Subtractor, Comparator, Counter, Decrementor, Incrementor, Multiplexer, Multiplier, Registers, Subtractor, and Combinational logic shifters.

Arithmetic Resource Sharing

PLSynthesizer features automatic resource sharing (also known as folding) of adders/subtractors and multipliers. The goal is to minimize the number of such operators and the subsequent logic in the synthesized design. This optimization is based on the principle that two similar arithmetic resources may be implemented as one single arithmetic operator if they are never used at the same time. The optimizer performs both resource sharing and, if required, reduction of the number of multiplexers that are created in the process. For example, consider the code fragment shown in **Figure 1.**

PLSynthesizer can implement the architecture for this as shown in **Figure 2.**

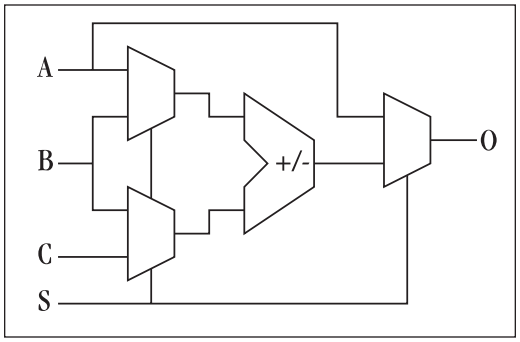

Figure 2. Intelligent Resource Sharing Results in Efficient Arithmetic Implementations

This example illustrates how PLSynthesizer can fold both adders and subtractors together and generate adder/subtractors for better optimization of the result. In addition to this technique, PLSynthesizer utilizes unique multiplexer and XOR optimizations. PLSytnthesizer can even infer combinational shifters from IF- and CASE-type code structures.

High-Density Xilinx FPGAs

Finite State Machine Extraction and Encoding

PLSynthesizer performs advanced finite state machine extraction and encoding. You can control these features on a global, entity/module, or even signal level. With this level of control, you can synthesize and optimize multiple state machine structures at the same time, each with its own optimal encoding scheme. The encoding options for Xilinx technologies are: One-hot, Compact, Sequential, Gray, Johnson, and User defined.

A unique feature of the advanced extraction process is that it is *independent* of the VHDL or Verilog coding style employed.

Support for Bottom-up and Hierarchical Design Methods

PLSynthesizer supports bottom-up and hierarchical design methods by giving you control over hierarchical flattening on a global and/or entity/module basis and whether or not I/O pads are to be inserted on the current top-level design. The synthesized output files can even preserve the original bus structures at the port level if desired.

Synthesis/Optimization Constraints

The use of design constraints (properties) is a powerful method to control several aspects of synthesis and post-synthesis results. Some are taken into account by PLSynthesizer to control its own operation, while others are simply passed on to back-end tools. For example, a pin assignment property applied on signals allows you to specify a given pinout in the target device.

There are three mechanisms by which constraints can be applied: within the GUI, with a separate constraint file, and with VHDL attributes. Constraints can be applied globally or locally on an entity/module basis and in some cases on individual signals or nodes. The constraints give you a high degree of control over such features as speed/area optimization tradeoffs, CPU effort level, hierarchy flattening, state machine extraction and encoding, enumerated type encoding, state register type, multiplexer and XOR extraction, combinational shifter extraction, pin assignment, and signal preservation.

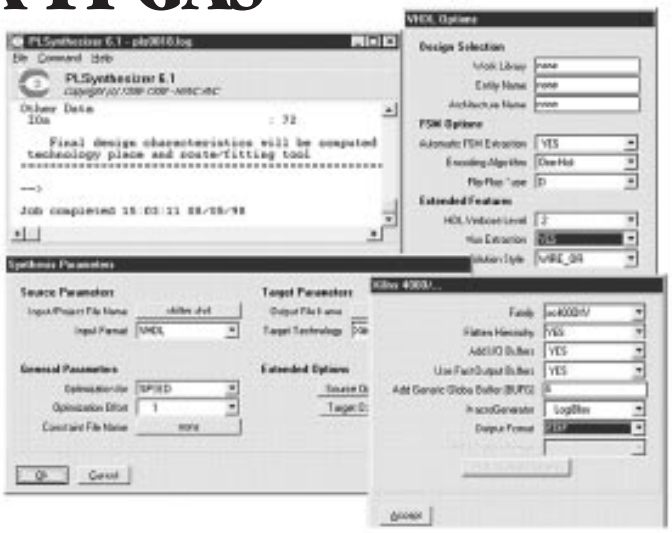

Figure 3: PLSynthesizer

Simple, Yet Powerful and Easy to Use Interface

PLSynthesizer can be run from the command line on Unix platforms or from its GUI on Windows® and Unix platforms. The GUI supports a simple, three-step process (load design, select target technology, GO!) or a constraint-driven process utilizing all the power and flexibility PLSynthesizer offers. The constraint-driven mode gives you control over all PLSynthesizer features and utilizes simple and easy to understand dialogs based on source options and technology options.

Conclusion

PLSynthesizer v6.1 has the performance, capacity and quality of results to help you make the most out of Xilinx high-density FPGAs. PLSynthesizer supports the XC3000A, XC3000L, XC3100A, XC3100L, XC4000E, XC4000EX, XC4000L, XC4000XL, XC40000XV, XC5200, XC7000, and XC95000 devices. The Spartan devices will be added in v6.2, scheduled for Q4 of 1998. PLSynthesizer is available for Window® 95/NT, SunOS, Solaris, and HP-UX. PLSynthesizer is also being integrated into the MINC Synario design environment to provide a full range design entry, analysis, and implementation environment. Σ

Greg Brown has more than ten years experience in the EDA industry and has held positions in development, consulting, management, and marketing of IC, ASIC, and programmable logic design tools at Mentor Graphics, VeriBest, and MINC Incorporated. Currently, he is the Sr. Product Marketing Manager at MINC.

For more information, contact: **MINC Incorporated,** 6755 Earl Drive Colorado Springs, Colorado 80918, www.minc.com or www.synario.com