Selected Features

Atmel's System Designer™: EDA Tool Suite for Co-verification

System Designer with System Level Co-verification

System Designer is a fully integrated co-verification tool suite that allows hardware/software co-design of FPSLIC™, programmable system-level devices in a unified environment. Its co-verification framework fully synchronizes hardware andfdsf software execution, provides source- and assembly-level software debugging with full visibility into the FPSLIC's memory and registers and hardware logic. In addition, place and route, macro generators, HDLPlanner™ and extensive bitstream utilities combine to provide a rapid logic design and development environment. The System Designer tool seamlessly integrates hardware and software design in a single, easy-to-use development tool.

Co-verification

During co-verification, the designer can set up to 256 microcode breakpoints where program execution will be halted; can step through the code sequentially and watch its execution, select whether calls should be traced; place the cursor on a particular instruction to effect code execution up to that point; or write execution scripts to drive the simulation.

The System Designer instruction set simulator for the microcode runs concurrently with the HDL FPGA design simulator, so the entire system can be designed and debugged together. An unlimited number of breakpoints can be set on C statements, assembler instructions, and on any address with any combination of access type of read, write and op code fetch.

A C-like macro language can be used to tailor the debugging environment with system macros for host file I/O simulation, reset, startup, shutdown, for and while loops, and if and return statements. Interrupt simulation can launch specific interrupts at a specific cycle count or periodically.

Benefits

Eliminates Prototype Boards – There is no way to effectively verify discrete systems using a microcontroller and an FPGA without creating a prototype board. Consequently, the design can be in the final stages before bugs are discovered in the interface between the microcontroller and the logic in a design. There are no tools at all for software development and debugging of MCU cores that are put into large FPGAs, which are being touted as system-on-a-chip solutions. Atmel's System Designer co-verification environment allows the microcode and the hardware design to be developed simultaneously without a prototype board.

Allows Hardware/Software Tradeoffs – System Designer allows designers to quickly explore and test various hardware/software implementations to arrive at a highly optimized design. Extensive "what-if" evaluation can rapidly be done – allowing systems architects to make power and performance trade-offs in their designs by trying different algorithm partitions between software and hardware.

Faster Design Cycles – Because the hardware and software can be designed concurrently, design cycles are faster. Up to half of the typical design project is spent in the integration and test phase, which in reality becomes an exercise in correcting the accumulation of errors from the front end of the design cycle. These errors often reach all the way back to the specification and partitioning phase, where ambiguities in the hardware/software interface were introduced, and then amplified during the hardware/software implementation phase. Often, the remedy of these errors is forced into software due to the long lead times and high cost of ASIC turns, even though a software fix may mean compromised performance or functionality in the final product. System Designer allows designers to do the complete hardware and software design with complete confidence. Software/hardware trade-off can be made and tested until an optimized implementation is arrived at. Design cycles can be cut by as much as 90%. In addition to combining the software development tools with logic simulation, the co-verification environment delivers high-performance co-verification months ahead of a discrete solution. The co-verification environment enables software and hardware development to be parallel activities, removing the software from the critical path and reducing the risk of hardware prototype iterations resulting from integration errors.

Macro Generators for Reusable IP Design

System Designer includes pushbutton macro generators that allow the quick design of fully parameterizable, fully optimized hard or soft intellectual property cores for the FPGA array on the FPSLIC. Macros are generated according to the user's specification. In other words, a multiplier can be 8 x 8, 12 x 2, 3 x 3, or any other size. Arbitrary word widths (e.g., 13 bits) can be specified and pipelining is available for many macros. The more than 50 macro generators included with System Designer include: cyclic redundancy check (CRC), integer divider, linear feedback shift register (LSFR), fast pre-scale counters to 64 bits, multipliers, accumulators, adders, FIFO, counters, comparators, decoders, deductors, flip-flops, latches, muxes, negation, RAM, ROM, shifters, subtractors and tri-state bus control.

Newly generated macros may be optimized for speed or area and can be used to build larger macros of virtually any complexity by employing hierarchical techniques.

Benefits

Faster Design Cycles – Macro generators relieve designers of the need to build complex functions from scratch. Reusability means that design iterations are fast and next generation designs can be built easily from the current design.

Predictable Timing – Macro-generator macros are pre-routed and have predictable timing. Timing parameters will not change even if the macro is moved or rotated.

Faster Placement and Routing – Macros designed with System Designer's macro generators are internally pre-routed so placement and routing of the overall design is faster.

Silicon-efficient Designs – Because macros developed with the macro generators are fully optimized for the FPSLIC's FPGA architecture, they result in the most silicon-efficient and high-performance implementations possible for the FPSLIC's FPGA architecture – like having an application engineer with you through the design process.

HDLPlanner Generates Syntactically Correct HDLs

Syntactically correct Verilog or VHDL templates and HDL code from System Designer's macro generators simplify the development of HDL code for FPSLIC products.

Benefits

No Hand Coding of HDL Code – Designers can arrive at a synthesizable HDL design without having to hand code it – accelerating time to market. Because the HDL is syntactically correct, there will be no errors during simulation or synthesis.

No Knowledge of HDLs Required – HDLs can be rather difficult to master. Because HDLPlanner automatically generates the HDL, designers do not have to be HDL experts to get the design done. Designers can focus on the design and do not have to focus on the implementation details.

Component Instantiation

System Designer allows the specification of intellectual property cores in the behavioral description by employing component instantiation. Firm IP cores that have been custom designed using System Designer's macro generators can be automatically inserted in the HDL description. For example, a designer can create a macro of an array multiplier, which can be invoked multiple times to build a firm IP core of a complex function, such as a 2-D convolver. The complex function is saved to a library with a support file and schematic symbol. During the behavioral phase of the design process, the designer simply specifies the name of the HDL convolver file automatically generated by System Designer in the VHDL or Verilog description. The synthesis tool will generate a gate-level netlist of the function for simulation, but the behavioral model will contain an "instance" of the convolver that is optimized for the FPSLIC architecture. During layout, the pre-built convolver core will be fetched from the library and placed on the FPSLIC FPGA array.

Benefits

VHDL or Verilog Designs That Are Optimized for Atmel FPSLIC Architecture – A common obstacle to using hierarchical design methodologies for FPGAs is that the implementation is usually extremely silicon inefficient because the synthesis tools do not understand the architecture of the FPGA. Portions of the design must be handcrafted after synthesis to get acceptable performance or area results unless operator inferencing can be exploited.

Component instantiation gives designers the freedom to create and use complex cores that are optimized for the FPSLIC architecture while still reaping the benefits of behavioral level design. In addition, designs remain technology independent and can be used to target an Atmel ASIC if required.

Predictable Timing – Because the instantiated core is internally pre-routed, timing is predictable.

Better Performance – Instantiated components will provide the best possible performance.

Better Silicon Efficiency – The firm IP cores and macros are already optimized for timing and area. These characteristics will not change during placement and routing. Silicon efficiency can be as much as 50% better using instantiated components.

Operator Inferencing

System Designer provides a seamless interface between its macro generators and most commercially available synthesis tools. During the placement and routing of the FPSLIC's FPGA array, System Designer automatically picks up on and extracts arithmetic or other structured logic blocks from the synthesized design, executes the appropriate component generator(s), and creates physical layouts of these functions that are optimized for FPSLIC. No manual intervention is required. The generated components may be optimized for area or timing. Synthesis-Gateway queries the designer regarding any macro specifications during the layout process. Operator inferencing of +, -, *, <, =, and > can be implemented using synthesis tools from Exemplar, Synopsis, Synplicity, Everest and View Logic.

Benefit

Fast and Efficient Layout of Behavioral Designs – System Designer ensures that VHDL and Verilog designs are optimized for the FPSLIC's FPGA architecture. The generated components are internally pre-routed and can be replaced in any location on the FPGA array, in any orientation, without affecting its performance, area, or power consumption characteristics.

Support for Dynamically Reconfigurable Designs

FPSLIC devices can be partially or completely reconfigured without affecting the rest of the FPGA array. Different data-path, counter or other logic functions and constants can be loaded and reloaded in the same location on the device in response to real-time events. Usually reconfigurable designs must be redone from scratch. System Designer lets users change a design parameter and save the design with a new name. The tool compares the two designs and extracts any parts of the design that are different and generates bitstreams of these delta designs. Using Atmel's patented Cache Logic®, various design parameters can be loaded and reloaded into the same space on the FPGA array in response to real-time events.

Logic or constants that are stored in FPSLIC's FPGA lookup tables can be updated during system operation by loading new values in them as needed.

Benefits

Enables Dynamically Reconfigurable Designs – All other FPGA tools require a new design to be compiled for any logic change, resulting in increased design time and risk. FPGAs must be stopped and the entire design reloaded. This operation is time-consuming. In addition, it can require multiple FPGAs if the system must operate continuously.

System Designer lets the designer change selected parameters and automatically generates the bitstream for the reconfigurable portion of the design.

Minimal Program Store Required – By using bitstream windowing, only the changing portion of the design needs to be stored in memory. A processor can also generate the windowed bitstream dynamically, in response to real-time events.

Timing-driven FPGA Array Placement and Routing for High-performance Designs

System Designer allows the specification of clock frequencies and other related timing constraints for the FPSLIC FPGA array. The tool will place and route the device to meet the timing constraints. The use of firm cores and macros that have been designed using System Designer's macro generators greatly facilitates timing-driven placement and routing because the individual macros are already internally pre-routed and have optimized internal timing.

Benefits

Better Performance – By allowing designers to specify timing constraints to the placement and routing software, System Designer allows them to "squeeze" the maximum performance out of a design.

Critical Nets Are Accommodated First – Since timing for critical paths can be specified prior to placement and routing, timing problems can be avoided and timing specifications can be met.

Multiple Views for Microcode Development and Debugging

System Designer provides a source window and pointer that mark the code currently being executed, so designers can compare the source C-code to the compiled code during debugging. Additional windows display the values of defined symbols, the contents of all 32 of FPSLIC's microcontroller registers, the contents of all of FPSLIC's microcontroller memory resources, the address of the next instruction to be executed, the value of the stack pointer and the number of clock cycles that have elapsed since the last reset and show the status of the peripheral devices, I/O registers and timer/counters.

Benefits

Complete Instruction Information for Quick Debugging – System Designer's comprehensive information lets designers quickly locate the cause of software bugs.

Quicker Time to Market – The ease of debugging FPSLIC microcode speeds up system development time and gets product to market faster.

Runs on MS Windows® 95/98 and Windows NT® Platforms

System Designer runs under virtually all Windows platforms.

Benefit

Low-cost, Widely Available Platform – Since most designers are already running Windows on a high-performance PC, System Designer will easily install on most desktop systems without modification.