Blue Pearl Software

Indigo RTL Analysis

Blue Pearl Software provides innovative solutions, addressing some of the most time consuming aspects of the design cycle – verification of RTL design code. Deploying **Indigo RTL Analysis™** during RTL creation detects complex bugs earlier in the design process and speeds your chips through the synthesis flow, reducing design iterations in the digital chip design flow.

Indigo's innovative predictive RTL analysis capability simplifies and speeds up design checking using a combination of functional and structural analysis. Unlike other RTL analysis tools that synthesize to gates and use only structural techniques, **Indigo** employs a functional approach that works on the full chip, at a high level of abstraction, rapidly identifying problems caused by interactions between design modules. Using **Indigo** during design creation saves time spent generating test benches to verify designs as Indigo automatically checks RTL for problems that are difficult to check using simulation.

Clock Domain Boundary Checks

- *Synchronization of data crossing clock domain boundaries*
- *Correct use of synchronized control signals*
- *Use of specified synchronization cells*
- *Convergence of independently synchronized data*

Race Detection

- *Write-Write*
- *Read-Write*
- *Latch-Latch*
- *Combinational loop*
- *Clock gating races*

Design reuse methodology

- *Coding style*
- *Reuse Methodology Manual (RMM)*
- *Semiconductor Reuse Standard (SRS)*

Synthesis checks

- *Simulation-synthesis mismatches*
- *Synthesis compatibility*
- *Implied latches*

Testability checks

- *RTL ATPG checks*
- *Clock gating*
- *Asynchronous sets/resets*
- *Asynchronous clocks*

Net-list checks

- *X-source problems*
- *Redundant logic*
- *Conflicting assignments*
- *Range Violation*
- *Non-resettable flip-flops*

Detect clock domain synchronization errors

Indigo's powerful clock domain boundary checking ensures that data crossing clock domain boundaries in multiple-clock designs is synchronized. **Indigo** recognizes common synchronization schemes such as double register buffering, memory/register files or synchronized control signals. Custom schemes can be specified. **Indigo** flags potential timing problems caused by the convergence of independently synchronized data – hard to find with simulation. **Indigo** identifies data that is generated and consumed by different edges of the same clock and checks for the appropriate use of lock-up latches.

Detect races before simulation

Race conditions are a major cause of critical design errors that can take significant amounts of time and effort to debug. Indigo's powerful analysis techniques reduce design debug time and effort by detecting potential race conditions before simulation and synthesis.

Indigo identifies race conditions, such as write-write, read-write, and combinational loop races, and automatically pin-points the lines of source code that are the cause of the race conditions.

Automatically check design re-usability

Indigo includes the following types of built-in methodology checks:

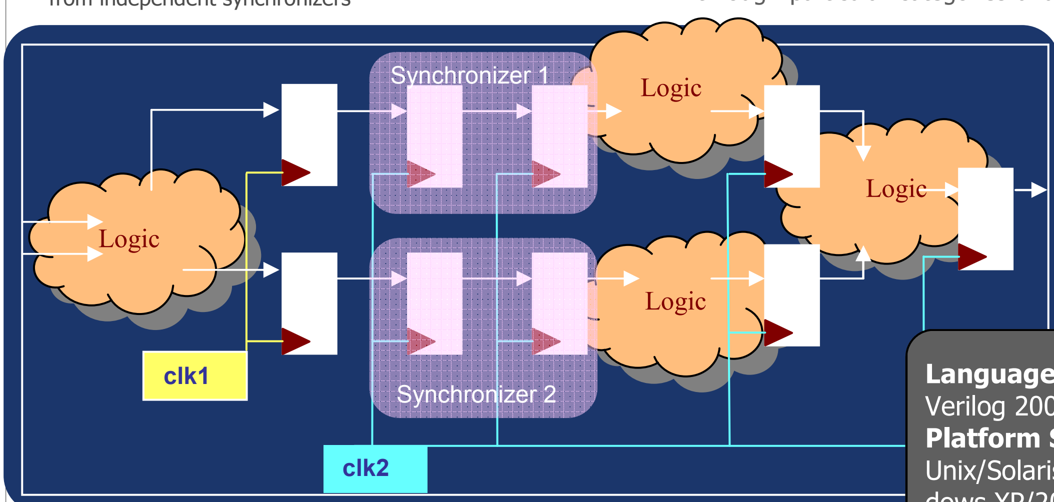
- Reuse Methodology Manual (RMM)
- Semiconductor Reuse Standard (SRS)
- Design for testability (DFT)
- Design for verifiability (DFV)

Indigo allows checking of traditional netlist and lint rules. These checks help you catch design errors before simulation and synthesis, and provide a way of ensuring that design teams follow a consistent design style and methodology that enables design re-use. In addition, **Indigo** allows user-defined checks making it easy to add rules and messages to the extensive library of built-in checks.

Enforce your design methodology

Indigo's support for user re-configurable rules and rule packages allows designers to customize rule packages that address specific design checking requirements. Predefined packages simplify the enforcement of design methodologies, while an integrated Tcl interpreter allows access to **Indigo's** design database providing the ability to write your own rules using Tcl commands.

Example check: Reconvergence from independent synchronizers



Find design for testability errors at RTL stage

Indigo checks that your RTL complies with design for testability (DFT) rules so reducing the amount of time and effort spent at the gate level finding and fixing scan DFT violations.

Eliminate design errors earlier

Indigo eliminates complex design errors at all stages of your design implementation cycle from RTL to gate level, from module level to full chip. Deployment of Indigo early in the design cycle drastically reduces the amount of effort spent finding bugs using time-consuming test-bench methods.

Be productive immediately

Indigo provides an easy to use interactive display of results, allowing you to click on a message to view the corresponding line of RTL code. Indigo can also be run in batch mode from the command line.

Indigo's filters allow you to focus on only the checks that are important for your design. You can exclude groups of checks or individual checks prior to running **Indigo** or you can filter messages based on category, severity, module, message id or wild card text matches before generating reports, so you see only what is important to you at a particular time. You can step through particular categories and see the lines of code that caused errors, or step through all the errors in a particular design module.

Language Support
Verilog 2001
Platform Support
Unix/Solaris, Windows XP/2000, & Linux

Blue Pearl Software, Inc. ■ 4677 Old Ironsides Dr. #430 ■ Santa Clara, CA 95054
Phone: 408.961.0121 ■ Fax: 408.961.0125 ■ info@bluepearlsoftware.com
www.bluepearlsoftware.com