HyperFault Mixed-Level Fault Simulator
What is HyperFault?

- HyperFault Mixed-Level Fault Simulator is a Verilog IEEE-1364-2001 compliant fault simulator that analyses test vectors’ ability to detect faults.

- Supports mixed levels of gate, behavioral, and switch with SDF timing. HyperFault’s proven algorithms enable efficient multi-pass fault simulation over distributed CPUs to achieve accurate results with excellent runtime performance.
Mixed-Level Fault Simulation: From Block to System

**Mixed-Level Fault Simulation**

Mixed level fault simulation analyses a gate level block within an RTL design.

**Fault Simulation of Each Block After Synthesis**

Multiple designers can analyze their blocks independently.

**Design Flow**

Merged results yield system fault coverage.
Key Features and Benefits of HyperFault

- Verilog HDL IEEE 1364-2001 compliant fault simulator
- Uses standard Verilog source files and libraries for mixed-level fault simulation with gate, behavioral and switch devices
- Complements BIST and ATPG in finding interconnect faults for critical missions
- Efficient multi-pass concurrent fault simulation algorithm with iterative fault collapsing gives optimal memory allocation, excellent runtime performance and accurate fault detection
- Automatic design partitioning supports distributed CPUs with load balancing for fast grading of large designs
- Accurate fault grading models include stuck-at high/low output and input faults
- Full timing fault simulation encompasses SDF back annotation for post-route delay analysis
Accurate Fault Detection for Mission Critical Products

- Full timing fault simulation encompasses SDF back annotation for post-route delay analysis
- Accurate fault grading models include stuck-at high/low output and input faults
- Manageable fault simulation runtimes are achieved using fault sampling
- Random sampling algorithm provides for accurate fault grading
- Because there is no need to modify any design files or libraries, no new errors are introduced
• IEEE 1364-2001 Verilog language compliant for designs, libraries and test benches
• Enhancements in the Verilog Programming Language Interface (PLI) provide greater simulation control and improved interoperability
• Compliant Standard Delay Format (SDF) for back annotation
• Mixed-level fault simulation with gate, behavioral and switch devices
• Accepts standard Verilog cell, I/O, memory, and other IP libraries
Applications for Fault Simulation When BIST and ATPG is Not Enough

- Critical paths hand coded for speed that cannot have scan chains inserted
- Designs for low power consumption that cannot tolerate additional gates
- Legacy designs that were not built with BIST or scan chains
- Asynchronous paths that are not static such as an input port that is also an asynchronous reset or an input used as both a data signal and clock
- When full post-route timing is required to correctly simulate fault behavior
- Feedback paths such as an internal ring oscillator, where inserting control to break the feedback loops would disturb the element balancing
- Libraries that do not support ATPG
- Cost of ATPG software is greater than cost of fault simulation software
- Finding interconnect faults in mission critical designs between individual IP blocks constructed with BIST and ATPG
HyperFault is used by both design engineers and test engineers
No need to modify any design files and possibly introduce errors
No need to create or modify any library files
Choice of intuitive graphic user interface, Unix shell prompt for batch operation
Regular expressions can be used to select faults from specific design blocks, or all of the faults in the device under test (DUT) can be selected
Supports PLI driven test benches with appropriate code to support the multiple pass strategy of fault simulation
Runs on Windows, Solaris and Linux platforms
Complete runtime control shows simulation progress in real time.

Five HyperFault algorithms work together to drastically reduce simulation time.
• HyperTrophic fault simulation algorithm minimizes processing of fault effects—significantly reducing runtime by 10X
• Efficient multi-pass concurrent fault simulation algorithm with fault collapsing gives optimal memory allocation, excellent runtime performance and accurate fault detection
• Value Change Dump (VCD) input of test vectors is supported
• Fault reporting is listed by instance in hierarchy for easy recognition
• Fault Injection can be scheduled after DUT is power up, eliminating false detections
• Automatic design partitioning supports multi-core computers with load balancing for fast grading of large designs
• Iterative Fault Simulation accumulates fault coverage as successive test patterns are applied
• Master process divides up the fault simulation job among multiple cores for a linear decrease in fault simulation time (eight cores reduce fault simulation time by a factor of 8)
• Multi-Pass assures all faults will be processed in the memory available. Faults that do not “fit into memory” are deferred from fault simulation into a subsequent pass
• Large design simulations finish overnight instead of a weeks
Hyperfault Performance on Multi-Core Computers

![Bar chart showing fault simulation time in hours for different numbers of CPU cores.](chart.png)

- **1 Core**: 130 hours
- **4 Cores**: 32.5 hours
- **8 Cores**: 16.25 hours
- **16 Cores**: 8.125 hours
Benefits of Multi-core Fault Simulation

• Runs on inexpensive, fast multi-core 32 and 64 bit computers

• Optimal use of computer memory

• No network overhead between master/slave processes

• Multi-core license provides cost effective solution