onespin

OneSpin 技术概览

OneSpin 帮助工程师创建可靠的、无错的FPGA/SOC数字设计

making electronics reliable

OneSpin 全球领先的自动形式化验证技术提供商

强健的公司组织结构

- 分公司: San-Jose, Munich, Tokyo
- 全球代表、技术支持网络

庞大的用户群

- 保持多年的高客户数年增长率
- 被最大多数大型半导体公司采用

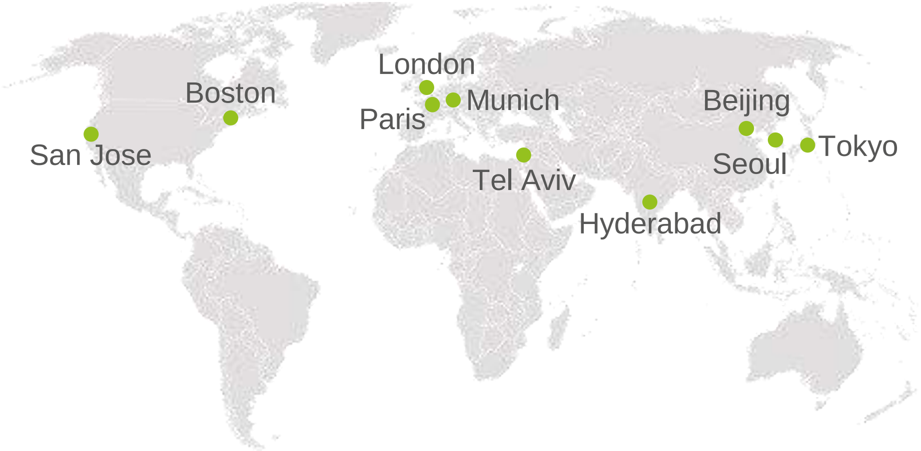
最佳解决方案

- 先进的形式化验证技术
- 超百个项目的实际应用
- 高效的技术服务网络

» 全球领先的形式化验证引擎技术

» 丰富的形式化验证解决方案

» 面向于工程客户使用的解决方案 **易用性好**



OneSpin 方案关注点

先进的形式化验证技术
可解决当今面临的最复杂的验证问题

Metric-Driven
Verification

MDV-基于度量的验证

Block Integ.
Validation

模块级/集成级验证

FPGA Impl.
Verification

FPGA实现的验证

超前的形式化验证技术
解决未来的验证难点

Agile Design
Evaluation

敏捷设计及验证

Safety Critical
Verification

高安全性\苛刻性验证

C++/SysC Des.
Verification

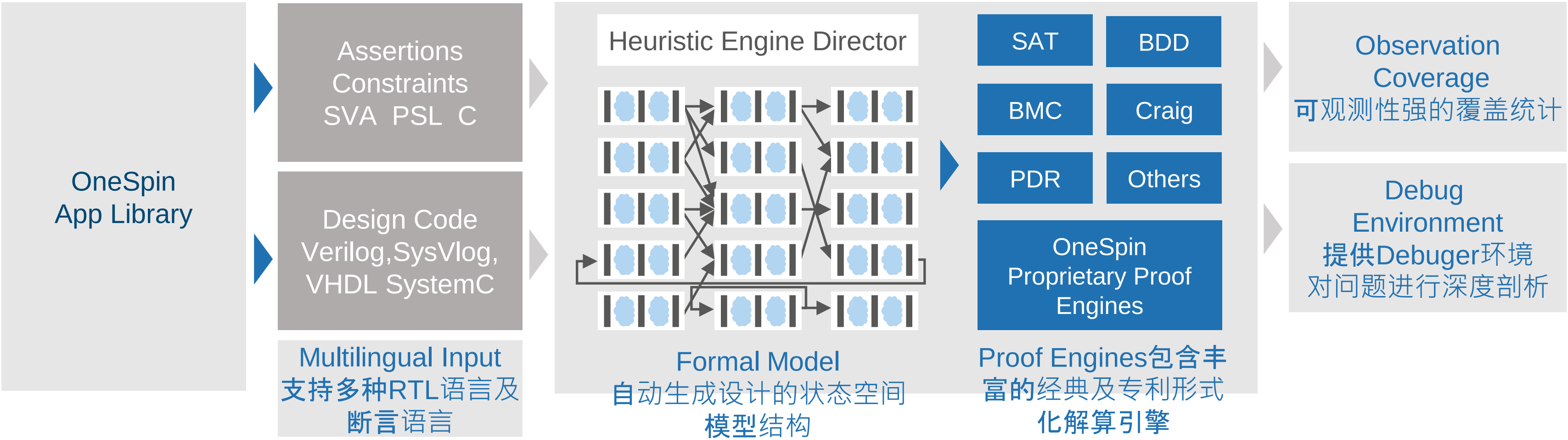
C/C++/SYSC设计验证

OneSpin 获得先进验证平台奖

2013: Top 3 EDA companies to watch

- Most advanced formal platform available 多种先进形式化解算引擎
- Usability & automation throughout 全面自动化 让客户尽可能袖手旁观
- 300+ development years maturity 技术成熟度 累积300人年的开发时间

OneSpin’s Technology-Leading Formal Platform 技术平台架构


OneSpin App Library

Assertions
Constraints
SVA PSL C

Design Code
Verilog,SysVlog,
VHDL SystemC

Multilingual Input
支持多种RTL语言及断言语言

Heuristic Engine Director

Formal Model
自动生成设计的状态空间模型结构

SAT
BDD
BMC
Craig
PDR
Others

OneSpin
Proprietary Proof
Engines

Proof Engines包含丰富的经典及专利形式化解算引擎

Observation
Coverage
可观测性强的覆盖统计

Debug
Environment
提供Debuger环境对问题进行深度剖析

高效、成熟的验证技术功能模块

先进的形式化验证技术
可解决当今面临的最复杂的验证问题

Metric-Driven Verification
MDV-基于度量的验证

Block Integ. Validation
模块级/集成级验证

FPGA Impl. Verification
FPGA实现的验证

超前的形式化验证技术
解决未来的验证难点

Agile Design Evaluation
敏捷设计及验证

Safety Critical Verification
高安全性\苛刻性验证

C++/SysC Des. Verification
C/C++/SYSC设计验证

技术平台子功能模块

- Automated Inspection
- Design Exploration
- Sequential EC RTL-RTL
- Assertion Verification
- Operational Assertions
- Observation Coverage
- DV Apps
  - X-Prop
  - Connect
  - Register
  - Scoreboard
  - Protocol Comp.
  - Activation
- SystemC/C++ Arithmetic
- Sequential EC FPGA
- Fault Qualification
- Security Verification
- Specification Verification

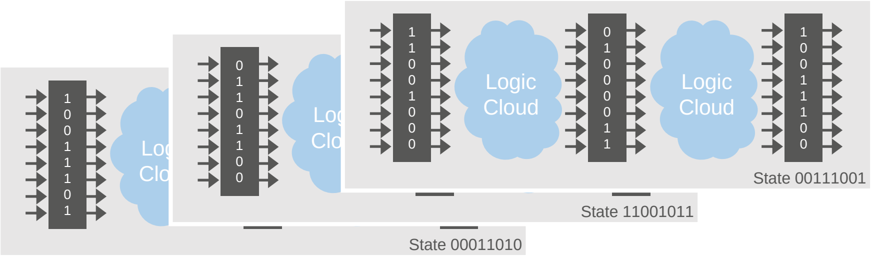
Proof Engines → Formal Model → Adv. Debug → LaunchPad

SystemVerilog . VHDL . SystemC . SVA . PSL

High Performance, Comprehensive Technology Platform

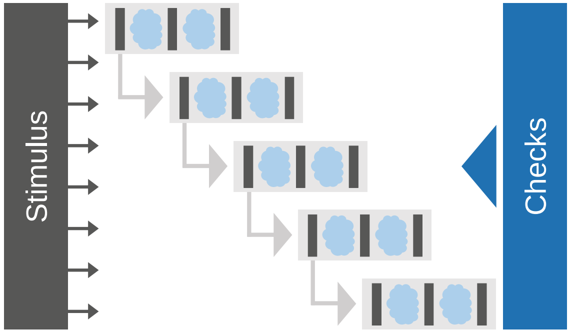
形式化验证方法

一种高效、易实施的方法



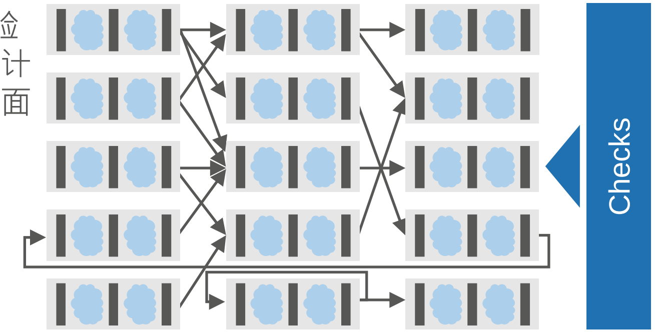
设计的动态特征就是在不同操作激励下的内部状态空间变换过程

Simulation 模拟仿真技术需要人员自己去设计激发不同功能场景的测试序列



易错, 耗时、碎片化、无法穷尽状态空间及组合

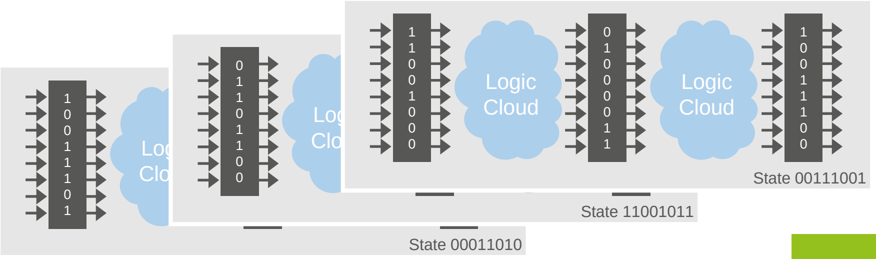
Formal 形式化验证技术提供对设计的状态空间的全面遍历

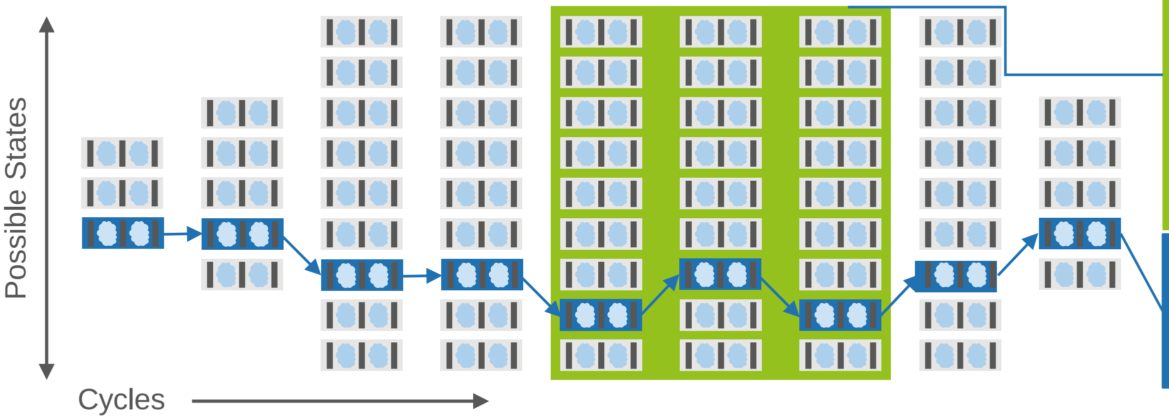


遍历所有状态空间及组合，无需人工设计激励场景

形式化验证过程中包含了正交验证方法

更快和更高的状态空间遍历





Formal 形式化验证则会自动遍历相关的所**有状态**及路径，自动确定这些转移过程中每**一拍的功能正确性；在形式化**验证技术中设计的功能需求是采用**SVA/PSL/ITL/TIDAL等**业界专用的验证语言编写，一个需求称为一**个**PROPERTY；**无需人工**设计激励信号序**列以及**测试台

Simulation **模**拟仿真只是单纯的按照人员给**定的激励信号序列沿着**设计中特定的状态路**径**变换，严格的说需要人为去确定此转移过**程中每一拍的功能正确性**

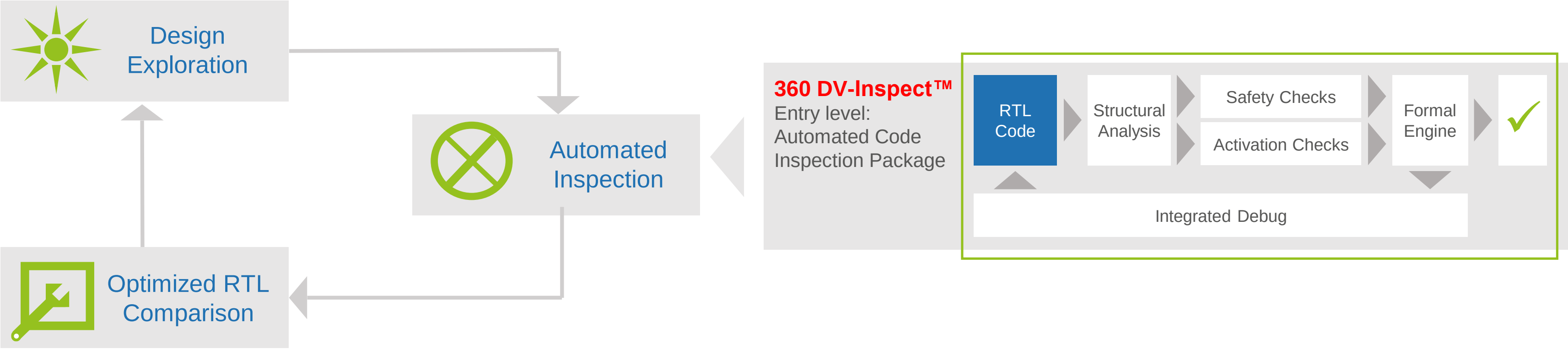
Agile Design Evaluation 敏捷式设计评估迭代

Push-Button Designer’s Toolbox 一键式工具箱

Allow design engineers to verify code on-the-fly, without hours of stimulus creation 无需设计激励

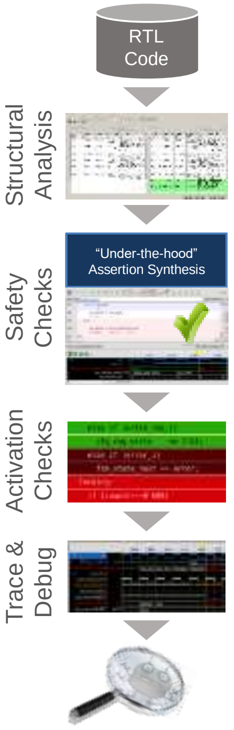
- Automated, “noise free” code inspection 自动分析代码缺陷 **无虚警**
- Interactive, push-button “what-if” analysis 一键式分析 **也可加入**环境约束
- Fast functional check of code optimizations 代码优化分析 **如不可达**代码 **不可达**状态 **不可达**转移

Iterative Agile Verification 敏捷迭代验证



Design Inspection 设计评审模块-INSPECT

Eliminate Common Issues Without Noisy Logs 无虚警



- Catch broad range of issues early in process **早期即可精确定位关**键缺陷及缺陷的详细路径追踪
- Formal checks execution, not just syntax linting **精确源自**语义计算，并非止步于语法检查类LINT
- Fully automatic, no manual assertions or stimulus **全自**动，无需人工设计激励或者断言
- High performance, e.g. 200K lines, 15K toggle checks, 3K block checks in under one hour

高效， 一小时可分析20万行， 包含15000个信号的翻转检查，3000个代码子块

- 缺陷类型丰富

Structure (Easy Lint)	Safety Checks (Assertion Synthesis)			Activation (Coverage)
Mismatch/port /wire	Runtime Errors	Sim-Synth Issue	Safe Function	Dead code checks
Signal trunc / no sink	Array / Range checks	SNPS full case	Neg / Zero div, exp, rem	Stuck signal (toggle test)
Sensitivity list issues	Function without return	SNPS parallel case	X / Z resolution	FSM trans and states
Unused signal / param	Signal domain checks	Write-write race detect	Arithmetic shifts	**MORE…**

Inspect Example: Array Out-of-Bounds 数组越界

Eliminate Noisy Linting Logs （与LINT的区别）

- Static linting points out potential out of bounds access based on code types

静态LINT工具采用语法分析，不会去做语义分析以及表达式计算和路径遍历，强调问题的“**可能性**”， 需要额外的人工确定

  - Unclear whether array is really accessed out of bounds during code operation
- OneSpin Inspect `array_index` check:
  - Either proves that access is never out of bounds or 通过形式化证明越界不可能发生
  - Shows simulation trace from reset with boundary violation **如果有越界**，则会生成从**RESET开始的波形序列直到越界情况的**发生，便于设计\验证人员快速理解和修善问题

```
reg [2:0] i;
reg [5:0] array;

always @(posedge clk)
for(i=0 ; i<=5; i=i+1)
  array[i] <= 1;
```

Lint: This code “might” result in out of bounds access

Inspect: Design operation does not result in an out-of-bounds access

Sequential RTL to RTL Equivalency Checking

Push-button verification of code optimizations

不同RTL设计版本间的功能一致性自动比对

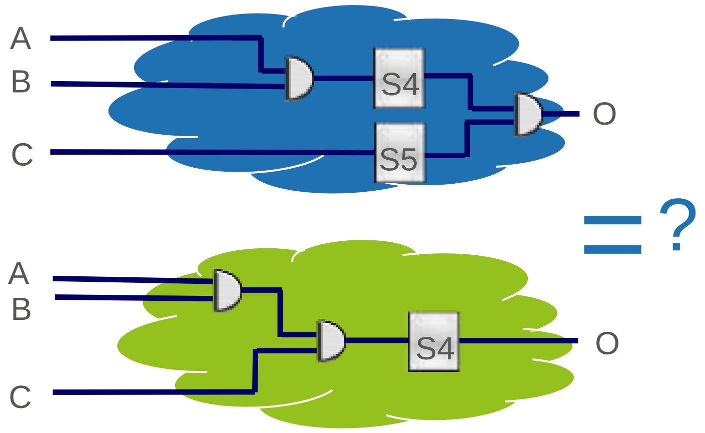
有时候设计人员会对RTL进行局部代码优化，但要保证功能不变，可以不重新回归仿真，直接快速形式化比对

Avoid re-simulation for common RTL modifications, e.g.

- Re-encoding of FSMs
  状态机重编码
- Register optimizations
  寄存器优化
- Sequential optimizations
  触发器优化
- Power optimizations
  基于降功耗的RTL优化

```
@posedge clk
begin
    S4 <= A && B;
    S5 <= C;
end
assign O = S4 && S5;
```

```
@posedge clk
    S4 <= A && B && C;
assign O = S4;
```

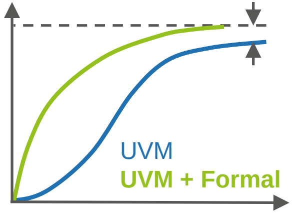


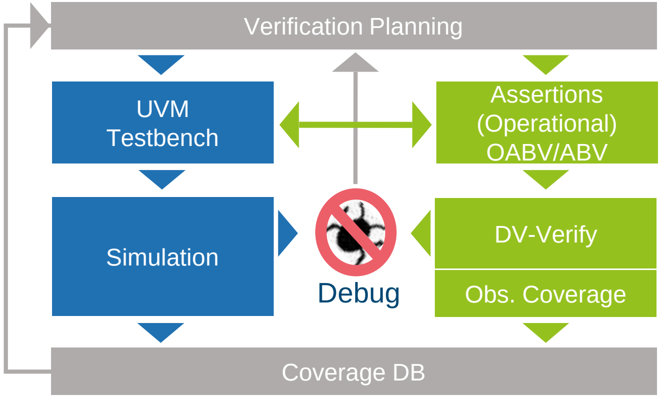
Metric-Driven Verification基于度量驱动的模块验证

Accelerated Block Verification Closure 加速模块验证进程

Accelerate coverage closure through effective formal application 加速覆盖率目标达成

- High-performance/capacity via advanced platform
- Observation coverage: 提供更精确和与功能需求覆盖相关的统计数据
- Integrated flow with simulation friendly use model
  - New partnerships with Synopsys and Mentor 将和主流模拟器无缝集成




Verification Planning

UVM Testbench

Assertions (Operational) OABV/ABV

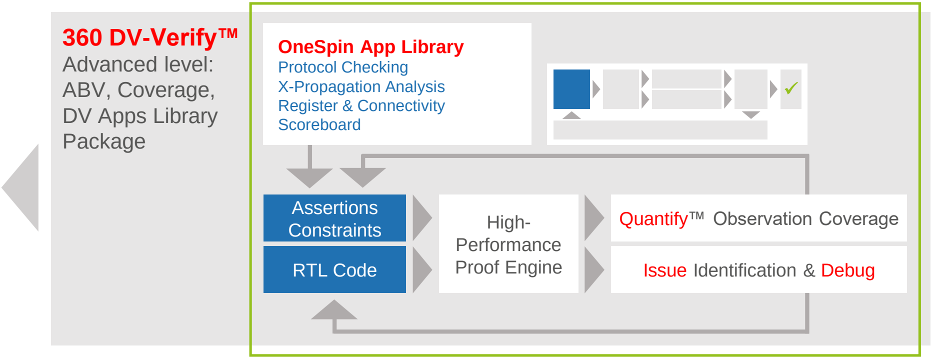
Simulation

Debug

DV-Verify

Obs. Coverage

Coverage DB


360 DV-Verify™
Advanced level:
ABV, Coverage,
DV Apps Library
Package

OneSpin App Library
Protocol Checking
X-Propagation Analysis
Register & Connectivity
Scoreboard

Assertions Constraints

RTL Code

High-Performance Proof Engine

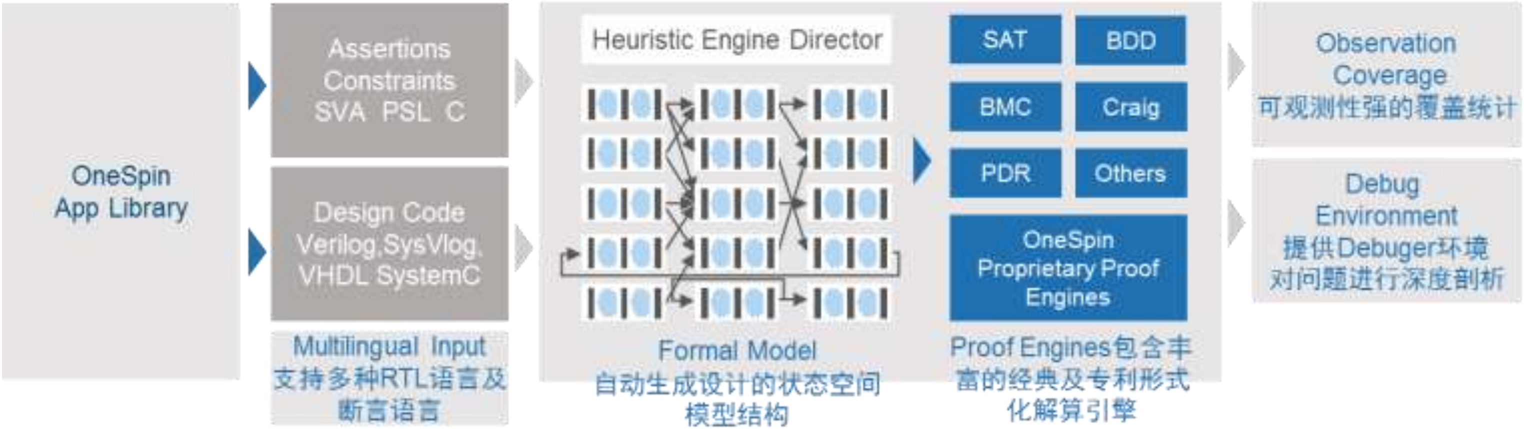
Quantify™ Observation Coverage

Issue Identification & Debug

Effective, Mature Formal Platform
高效成熟的形式化平台

- The most technically advanced formal platform available **最先**进的形式化验证技术
  - Optimized formal model: high performance, capacity **高度**优化的形式化模型
  - Broad range algorithms: high proof depth, convergence **丰富的**验证引擎技术
  - Heuristic director: Automate engine application for best results 验证引擎智慧选择
- Advanced usability & automation built-in **高可用性 并非形式化**专家才能用的技术
- 300+ development years, 1000s of designs ensures maturity 300**人年的开**发 1000**个**实际项目考验

OneSpin's Technology-Leading Formal Platform 技术平台架构



OneSpin App Library

Assertions Constraints SVA PSL C

Design Code Verilog,SysVlog, VHDL SystemC

Multilingual Input 支持多种RTL语言及断言语言

Heuristic Engine Director

Formal Model 自动生成设计的状态空间模型结构

SAT, BDD, BMC, Craig, PDR, Others

OneSpin Proprietary Proof Engines

Proof Engines包含丰富的经典及专利形式化解算引擎

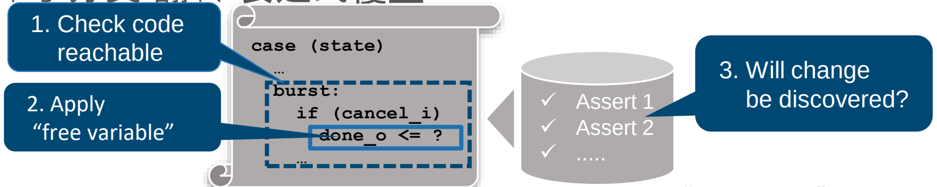
Observation Coverage 可观测性强的覆盖统计

Debug Environment 提供Debugger环境 对问题进行深度剖析

OneSpin Quantify™ Observation Coverage

Precise Coverage Closure 可观测性覆盖统计技术- Quantify™

- Most precise formal coverage algorithm available 最先进的覆盖统计算法
- Assertion development guidance towards closure对功能断言需求集开发的指导作用 代码大于/等于/小于需求？
  - Reduces redundant testing, shows uncovered areas
- Verification process management 验证计划的管理
- Detects unreachable code 不可达代码识别
  - Over constrained, unreachable code
- Code coverage model similar to simulation coverage包含传统覆盖信息 语句 分支 翻转 表达式覆盖



1. Check code reachable

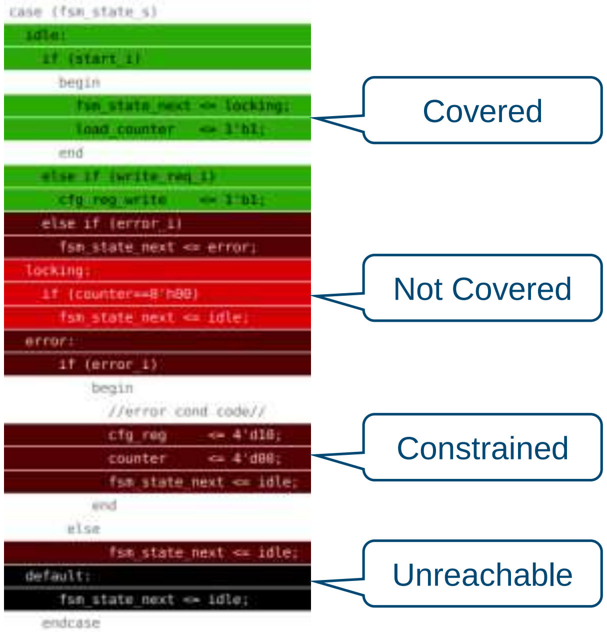
2. Apply “free variable”

```
case (state)
…
burst:
  if (cancel_i)
    done_o <= ?
…
```

✓ Assert 1
✓ Assert 2
✓ .....

3. Will change be discovered?

Observation Coverage more precise and efficient than other “Mutation” or “Cone-of-Influence” -based methods比其它的“变异”和“影响锥”技术好

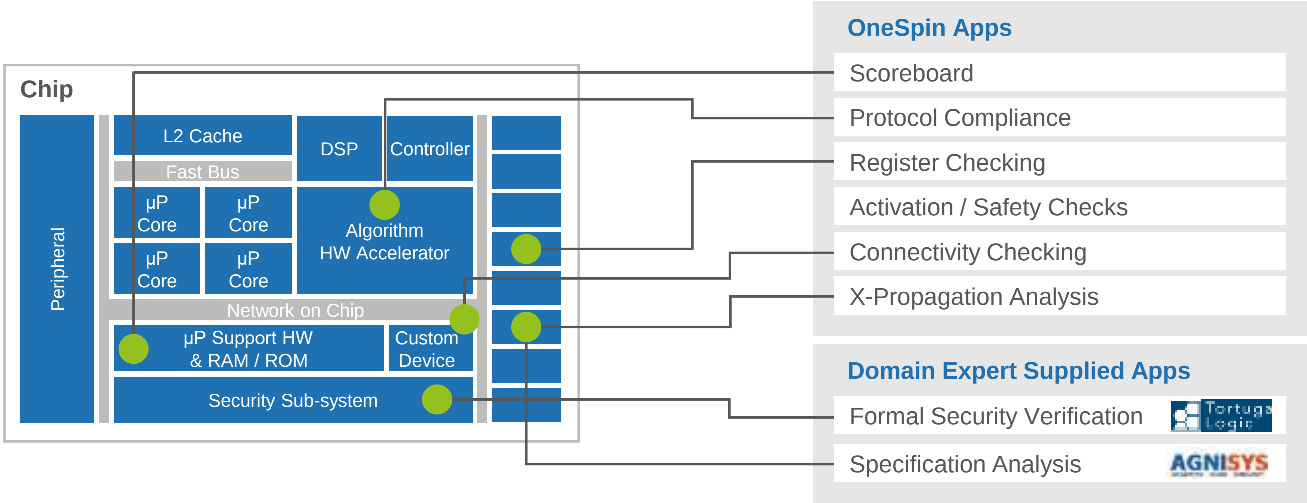


```
case (fsm_state_s)
  idle:
    if (start_i)
      begin
        fsm_state_next <= locking;
        load_counter   <= 1'b1;
      end
    else if (write_req_i)
      cfg_reg_write    <= 1'b1;
    else if (error_i)
      fsm_state_next <= error;
  locking:
    if (counter==8'h00)
      fsm_state_next <= idle;
  error:
      if (error_i)
        begin
          //error cond code//
          cfg_reg      <= 4'd10;
          counter      <= 4'd00;
          fsm_state_next <= idle;
        end
      else
          fsm_state_next <= idle;
  default:
      fsm_state_next <= idle;
endcase
```

Automated Formal Apps提供多种专业领域验证包

Automatically solve tough verification problems

- Solve complex, error-prone verification issues 验证复杂问题 **减小易错**的人工行为
- Exhaustive testing without significant simulation effort **无需人工模拟**仿真 **自**动全状态空间验证
- LaunchPad: enabling third-party, domain expert apps **可与第三方**专家工具协同验证



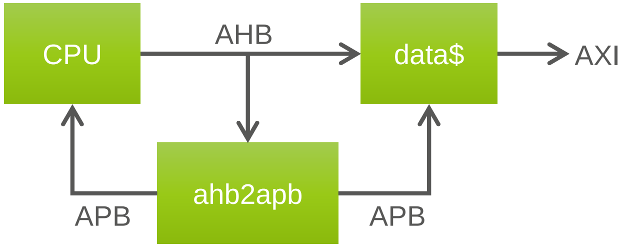
OneSpin Apps
- Scoreboard
- Protocol Compliance
- Register Checking
- Activation / Safety Checks
- Connectivity Checking
- X-Propagation Analysis

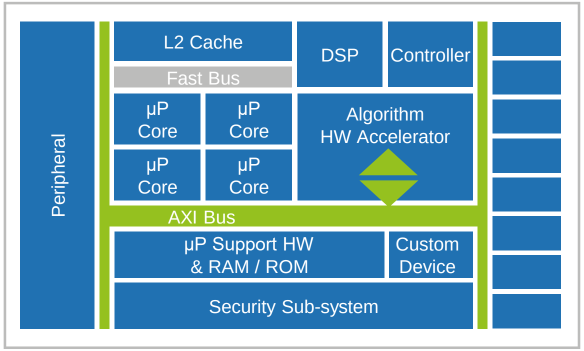
Domain Expert Supplied Apps
- Formal Security Verification
- Specification Analysis

Protocol Compliance Verification

Verify complex protocols rapidly without stimulus 复杂协议验证包 无需模拟仿真

- Exhaustively verifies protocol compliance 全面验证协议兼容性
- Fully automated, no sim stimulus 全自动 无需设计激励
- Can be used to constrain external interfaces 外部接口模式可约束
  - Active and passive modes
- Efficient debugging of failing checks 调试不兼容的协议实现
- Broad Protocol Library 丰富的协议库
  - Currently available: AHB, AHBLite, APB, AXI3, AXI4, AXI4Lite, AXI4Stream
  - Other protocols available through VIP partners





AXI Bus Protocol Operating Correctly?

X-Propagation Checking X信号状态扩散检查

Determine if uninitialized states cause device failure

- Ensures that X’s do not propagate to FSM’s, critical states, functional outputs, etc.

检擦X信号状态是否会扩散到状态机、关键触发器以及输出口

- Fully automated, exhaustive analysis

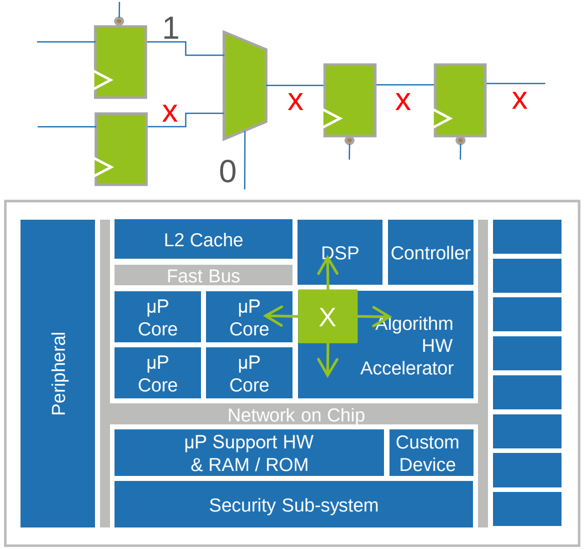
全自动、全状态分析

- Easy to set up

极简配置过程

- 4-state analysis without RTL X-optimism

采用“悲观”式X扩散检查



Undefined state propagating to damaging locations

Register Map Verification 寄存器内存映射验证

Compare register implementation vs. memory map

- Check register implementation matches IP-XACT, RDL or custom register spec

检查寄存器实现与IP-XACT\RDL\自定义格式的寄存器规格文档的一致性

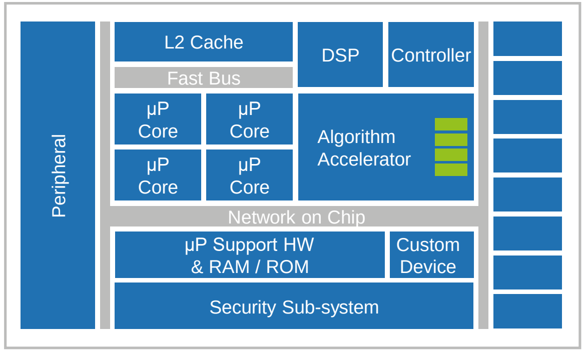
- Operates with protocol check for full hardware verification
- Highly automated flow, easy to set up

高度自动化 极简配置

- Runs much faster than simulation, no stimulus required 验证速度数十倍于模拟仿真 无需人工设计激励

Address

IP-XACT



Register implementation to memory map match

Connectivity Checking 连结正确性检查

Check connectivity through complex circuit structures **可用于**复杂电路结构互联

- Highly automated, easy to setup

高度自动化 **极**简配置

- Assertion-based and structural connectivity for maximum confidence

基于自动断言的验证方式保证了结论的可靠性

- Supports delays and conditional connectivity

支持延迟连接以及条件连接

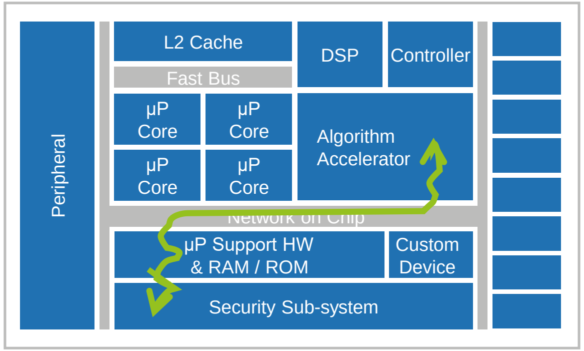
- Proven on very large designs and an essential part of SoC integration

在超大型集成验证中得到过应用

- Efficient debug of connectivity issues

易于调试发现的连接问题

Connectivity



Check key connections through complex structures

Scoreboard Analysis 记分板分析

Ensuring data transport integrity 确保数据传输的完整性

- Verify data is not dropped, duplicated, created or otherwise modified

验证数据在传输过程中不会非预期的丢失\复制\创建\篡改

- Eliminate simulation intensive operation

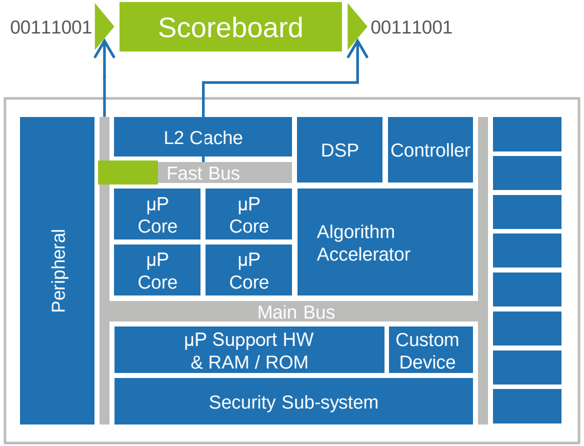
减少模拟仿真的激励数据序列的组合爆炸

- Exhaustively test any data combination

数据序列全组合空间验证

- No requirement for stimulus or assertions

无需人工设计激励序列和功能断言



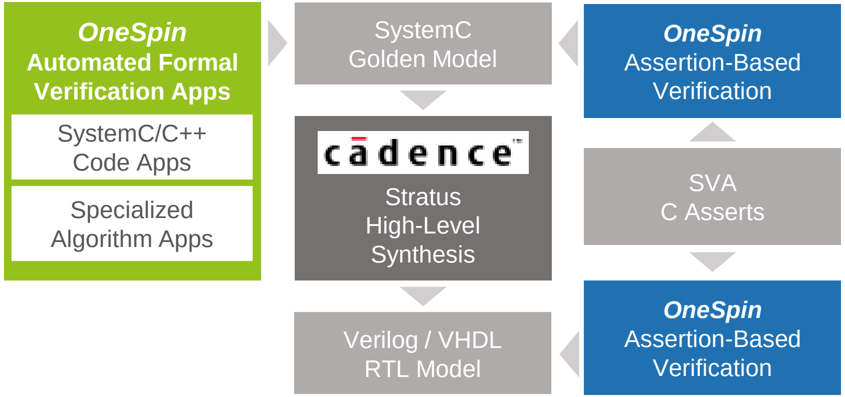
Check bus bridge data transport integrity

SystemC/C++ Design Verification

Verify C++ / SystemC source code, not disconnected RTL

直接验证C++ / SystemC 源码

- Detect and analyze common SystemC language artifact problems 直接检查SystemC 源代码中的缺陷
- Automated apps for algorithm design, e.g. fixed point precision 检查设计中的算法精确性 如定点计算精确性
- SVA on SystemC: Temporal, reusable assertions to verify code vs spec 基于SVA的功能\时序验证
- Cooperation with Cadence HLS team to build complete flow 与Cadence HLS高级综合团队合作搭建完整工作流



OneSpin
Automated Formal Verification Apps

SystemC/C++
Code Apps

Specialized
Algorithm Apps

SystemC
Golden Model

cādence
Stratus
High-Level
Synthesis

Verilog / VHDL
RTL Model

OneSpin
Assertion-Based
Verification

SVA
C Asserts

OneSpin
Assertion-Based
Verification

Handling SystemC Initialization Issues

Potential Simulation Synthesis Mismatches-SystemC 语言的变量初始化语义分为仿真语义和综合语义，二者不完全相同，存在差异

Automatic variable initialization in SystemC (due to C++ mother language)

- All sc_ datatypes automatically initialized to default value

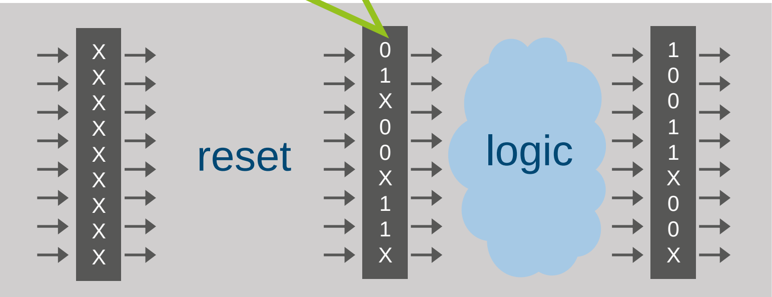
However, synthesizable subset standard states:

- Initializations from Module constructor ignored
- Reason: Reset behavior under user’s control

Inevitable Sim/Synth mismatches hard to debug

OneSpin 360 DV SystemC

- ✓ Checks which registers are initialized
- ✓ Check (intentionally) undefined reg effect
- ✓ Switch between sim & synth semantics



OneSpin 360 DV-SystemC

SystemC Code Apps & Checks 类似于INSPECT

- Many SystemC issues automatically identified 自动检查SystemC 代码缺陷
- No need for stimulus or class library simulation **无需人工激励**设计以及类仿真
- Problems easily debugged up-front **缺陷可定位**调试观察

SystemC Coding Apps

Activation / Safety Checks

Scoreboarding

X-Propagation Analysis

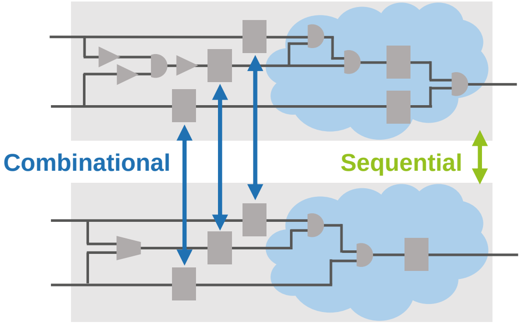
Write-Write Race Checks

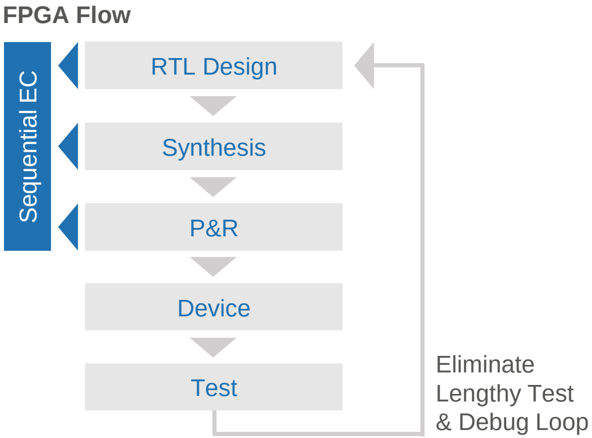
Structure (Easy Lint)	**Safety Checks (Assertion Synthesis)**			**Activation (Coverage)**
Mismatch/port /wire	Runtime Errors	Sim-Synth Issue	Safe Function	Dead code checks
Signal trunc / no sink	Array / Range checks	Initialization	Arithmetic overflow	Stuck signal (toggle test)
Sensitivity list issues	Function without return	X-Propagation	Redundant bits	FSM trans and states
Unused signal / param	Signal domain checks	Write-write race	Arithmetic shifts	**MORE…**

FPGA Implementation Verification逻辑等效性验证

Sequential Equivalency Checking: Increasing FPGA QoR

- Accelerates design flow, reduces testing **加速**设计周期 **减少等效性**验证消耗
- Enables aggressive optimization usage **支持所有**逻辑优化技术
- Significant post-production risk reduction **减少后端修正/流片失**败风险

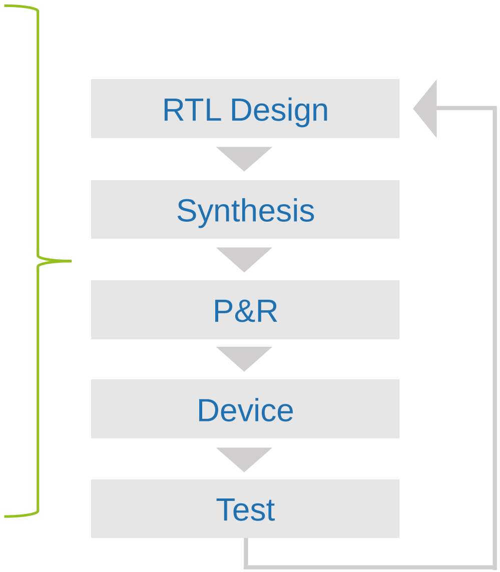




Real FPGA Design Flow Bug Examples
一些在实际验证发现的等效性问题

Example Design Flow Issues Encountered

- Bus connection ordering 总线连接问题
- Coincident read discrepancies 并行读操作的一致性问题
- Wrong FSM re-encoding 错误的状态重编码
- Undriven or unconnected wires 未驱动\未连接的wire
- Incorrectly coded pipeline 错误编码的流水操作
- Incorrect BRAM parameter settings 错误的BRAM参数设定
- Clock gating for low power issues 不恰当的低功耗门控时钟
- P&R connection issues 布局布线连接问题
- Additional, unspecified logic added 多余的逻辑


RTL Design

Synthesis

P&R

Device

Test

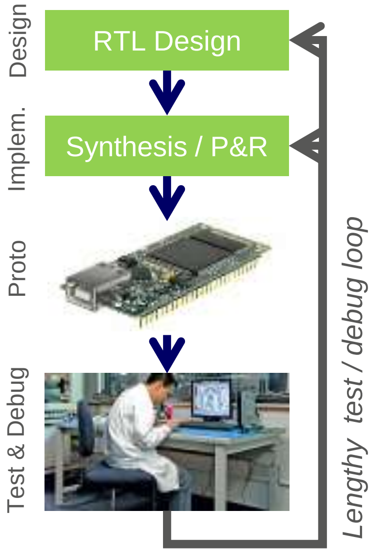
Modern FPGA Design Flow Issues
现代FPGA设计流程遇到的问题

- FPGA Requires Advanced Optimizations FPGA需要更高级的优化

- Systematic Error Probability Increases 增加系统问题可能性
  - Created by design flow refinement automation 自动改善设计流程
  - Hard to find, require days of debug 难以发现，需多日纠错
  - Require many complex tests to discover 需要很复杂的测试
  - May cause damaging field issues 会损坏区域
  - Limits use of powerful optimizations 功耗优化受限

- Prototype-Based Testing No longer Viable 基于样本机测试不可行
  - Excluding systematic errors require lots of additional tests 排除错误需要更多的测试
  - Errors often triggered by unexpected corner cases 错误会触发未知的边界情况
  - Safety Critical regulations mandate formal techniques 安全性规则需要形式化验证

Generalized FPGA Flow

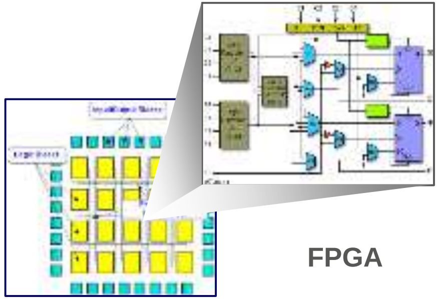


Critical issues remain undetected!

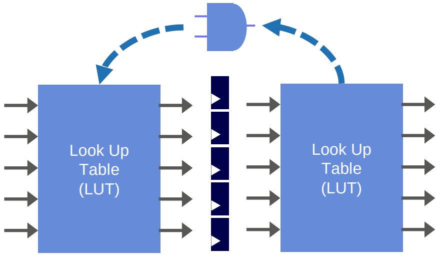
FPGA Synthesis Optimization FPGA综合优化

Key Factor in Design Performance 设计性能的关键因素

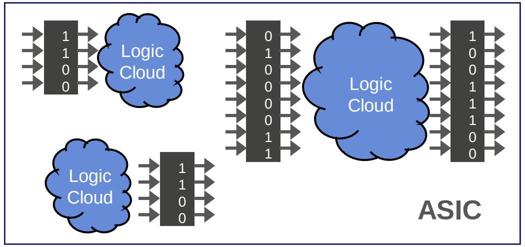
- **FPGA Specifics: FPGA特性**
- Fixed interconnect grid, LUTs, Shift-Registers, Block RAMs and configurable DSP blocks etc. 模块固定
- Many timing, fan-out, capacity restrictions 很多时序，扇出，功能限制
- Synthesis maximizes utilization by register duplication, retiming and other sequential optimizations
- 综合器通过复用寄存器，时序调整，其他顺序优化



Fixed Pre-Manufactured Structure



Example: FPGA synthesis tools balance logic between LUTs to improve QoR



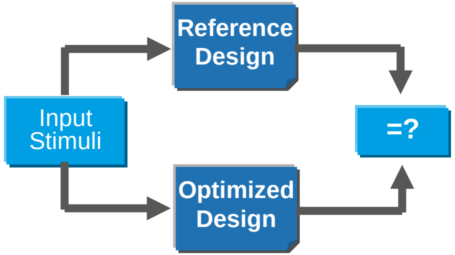
Fully Flexible Interconnect and Logic

Maximize FPGA QoR Through Aggressive Optimizations!

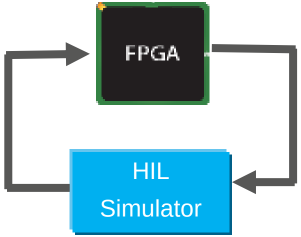
为了FPGA质量结果会极端优化

Verification Choices 验证选择

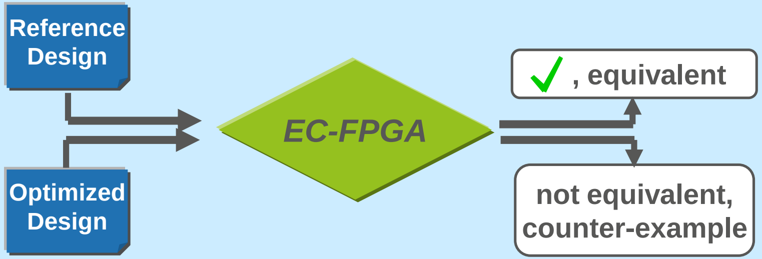
Gate-Level Simulation


Reference Design
Input Stimuli
=?
Optimized Design

HW-in-the-Loop / On-Board Verification


FPGA
HIL Simulator

Not exhaustive (100 input bits ~ 10^{30} cases)
Errors can – and do – escape !
Simulation and debugging are very time consuming

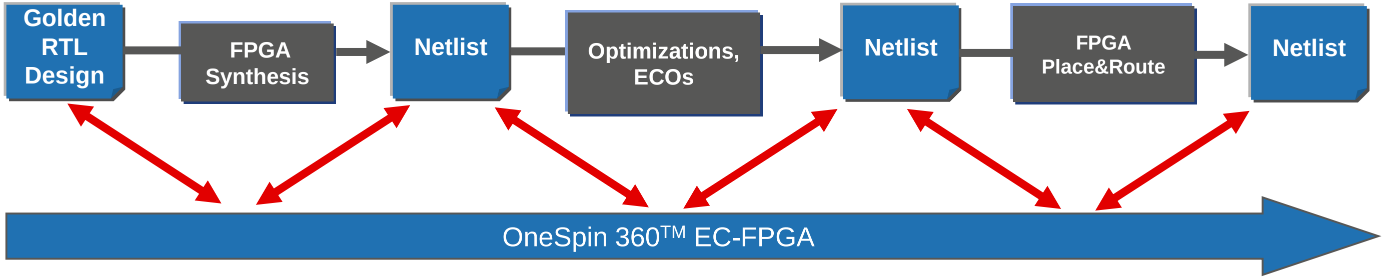

Reference Design
Optimized Design
EC-FPGA
✓ , equivalent
not equivalent, counter-example

Equivalence Checking

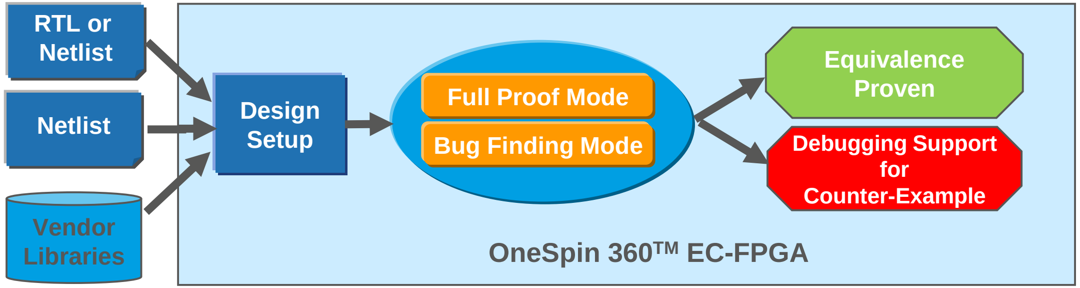
“same inputs ⇨ same outputs – *always* !”

- Exhaustive for all designs
- ALL errors are found
- Short runtimes
- Fast debugging

EC-FPGA Overview



- Synthesis tools: Vivado, SynplifyPremier, ISE, Quartus II
- Place&Route and FPGA families: Xilinx, Microsemi, Altera
- Languages: VHDL, Verilog, SystemVerilog, EDIF, mixed
- Platforms: Linux, Windows



What’s the Difference to ASIC EC?

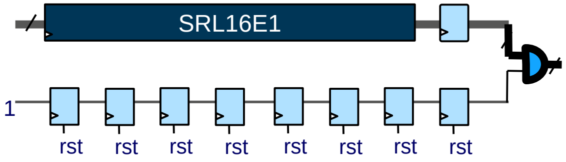
- **Conventional Equivalence Checkers 传统一致性验证器**
- Do not support advanced FPGA optimizations 不支持高级FPGA优化
- Require extensive manual intervention and complex scripting 需要额外手动介入和复杂的脚步
- Require and rely on information from synthesis “side files” 需要并依靠综合器生成各种文件支持

- **OneSpin 360$^{\text{TM}}$ EC-FPGA**
- Handles **ALL** FPGA-specific optimizations 解决所有FPGA具体的优化
- Does not rely on synthesis side files 不需要依靠综合器的
- Verifies whole-chip flat netlists “as is” 扁平后的网表
- Provides high degree of automation and simple scripting 提供高度自动简洁脚本

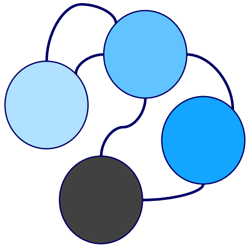
FPGA Synthesis Optimization Examples

综合优化举例

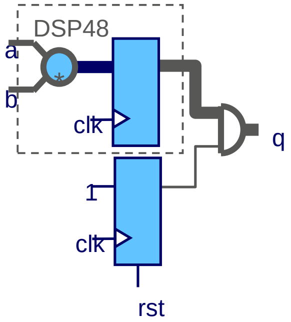
- 360 EC recognizes structures to reduce dependency on hint files: 自动识别结构，降低提示文件需求
  - Reduced scripting and black boxing
  - Increased QoR and Ease of Use
  - Support for DSP, Distributed RAM



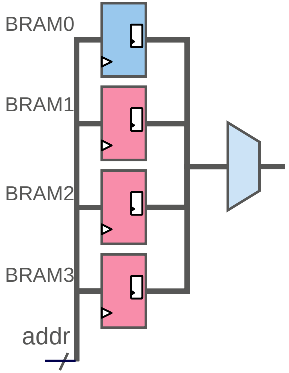
Configurable SRL blocks, including reset flops



FSM comparison regardless of encoding

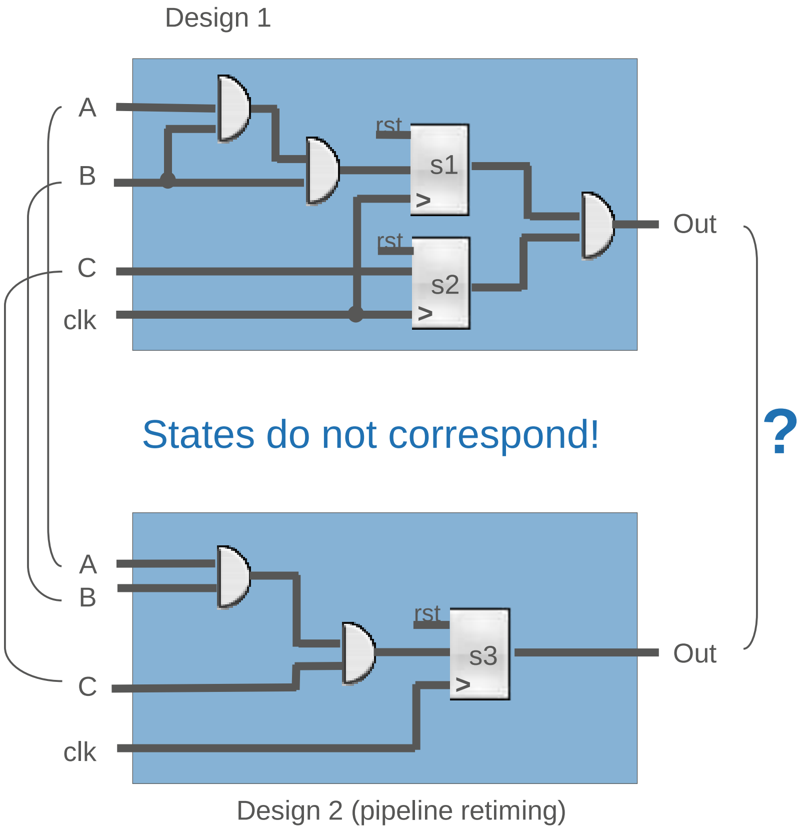


DSP registers added by synthesis handled



Block RAM split to optimize no need to black box (for Xilinx)

Advanced Synthesis Optimizations 高级综合优化



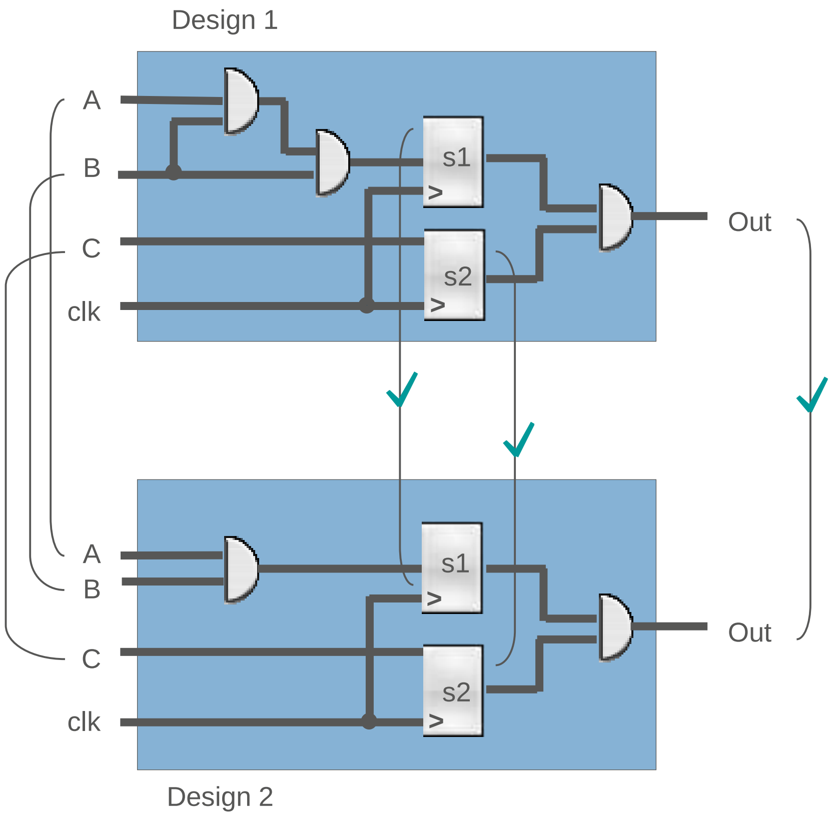
States do not correspond!

FPGA optimizations significantly change the state structure

- Designs are equivalent in their reset states:
  - s1=0, s2=0
  - s3=0
- However
  - **"map":** cannot identify corresponding states
  - "**compare**": cannot prove outputs to be equivalent

⇨ Classic EC fails !

Technology Background: Combinational EC

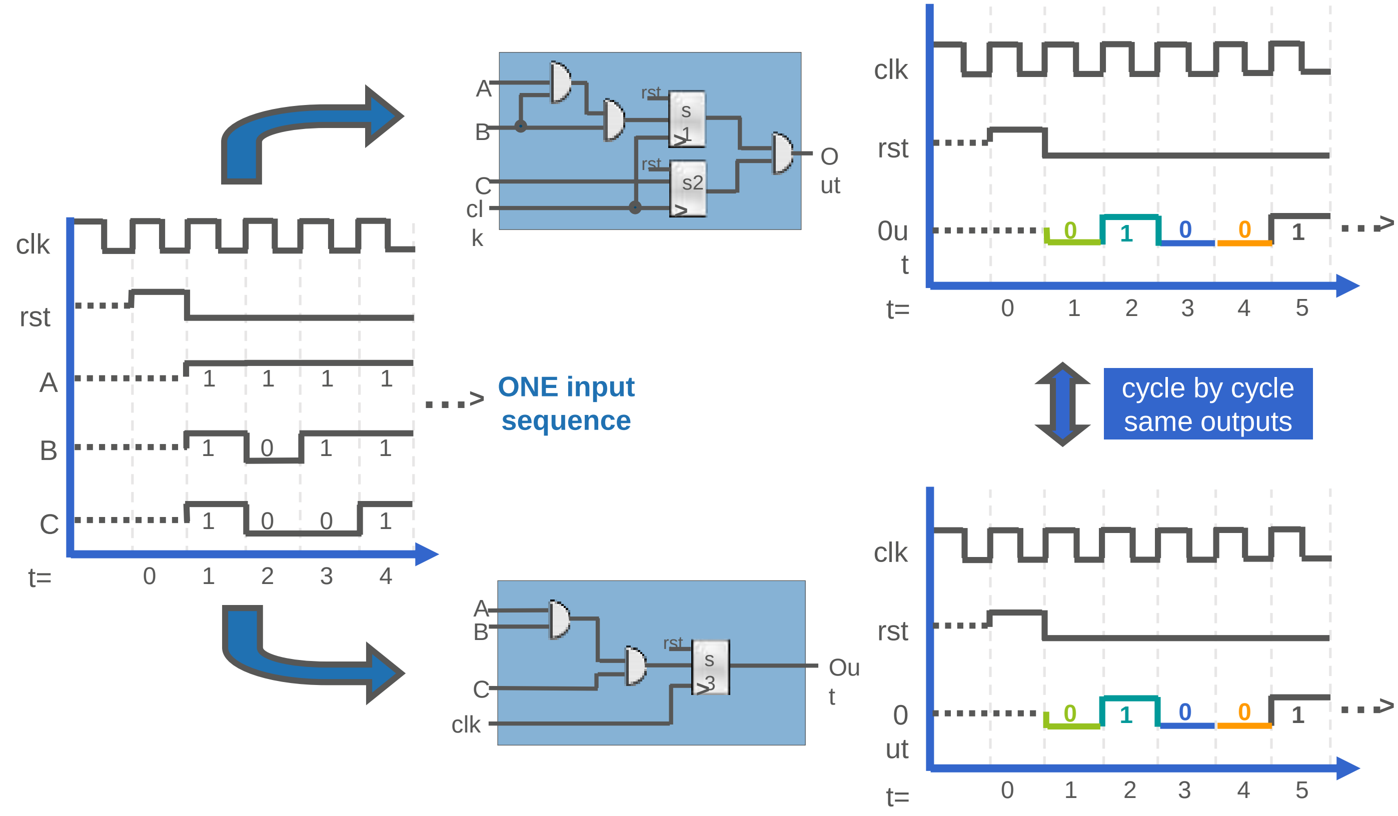


Equivalence:

- Same inputs result in identical outputs – ***always* !**
- Exploits **equivalent states** of the designs
- Identifies corresponding inputs, states, outputs (“map”)
- Proves that corresponding states and outputs have same value for all possible input combinations (“compare”)

⇨ Classic combinational equivalence checking – widely used in ASIC flows

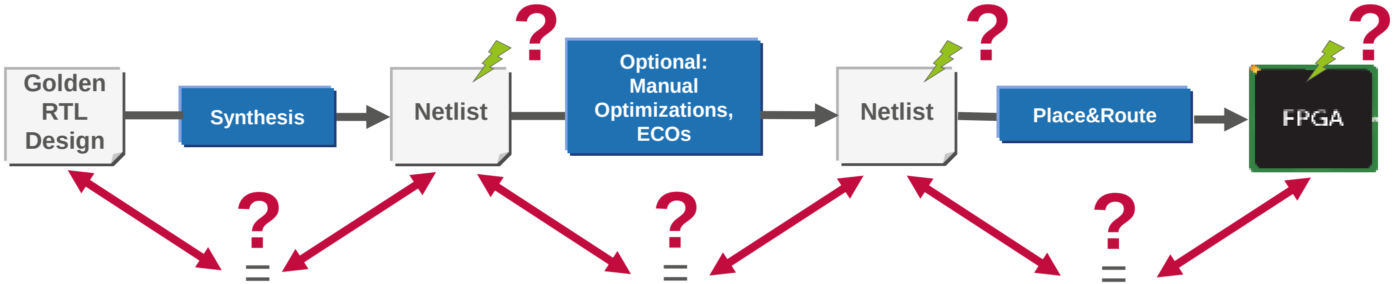
Sequential “Equivalence” using Simulation



ONE input sequence

cycle by cycle same outputs

Synthesis & Verification Challenges



- Synthesis and manual optimizations are error prone
- Critical issues:

  incorrect wiring, user-directed logic optimizations (pragmas et. al.), logic retiming, pipelining, arithmetic optimizations, state initialization, …

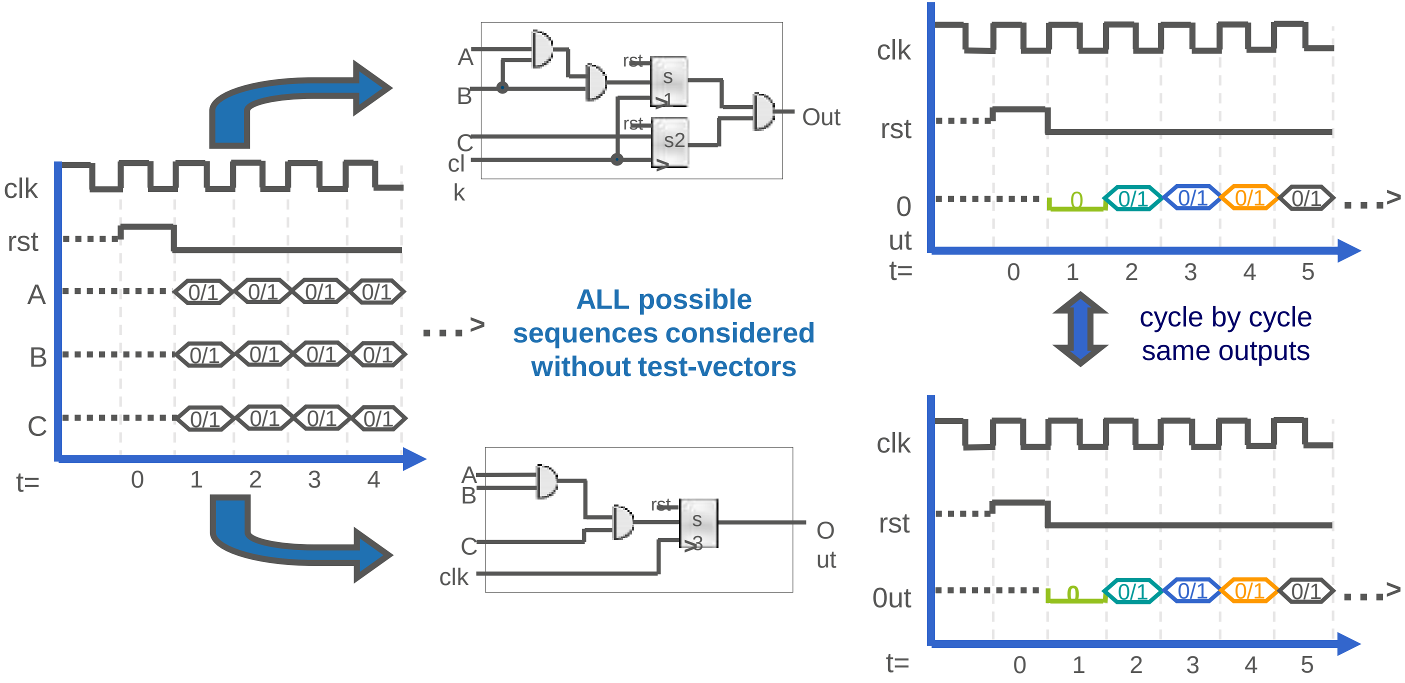
⇨ FPGA will not work correctly !

⇨ Debugging can be very time consuming !

Sequential Equivalence Checking

Using 360 EC-FPGA

Corresponding outputs values proven to be identical for **ALL** possible input sequences
⇨ Designs guaranteed to be **sequentially equivalent**

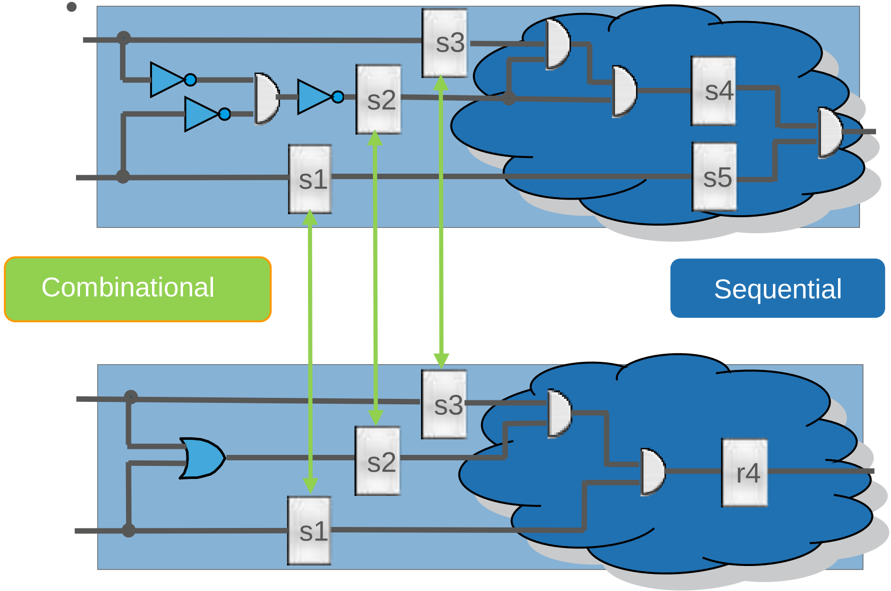


ALL possible sequences considered without test-vectors

cycle by cycle same outputs

Mixed Scenarios Handled On-The-Fly
混合方案

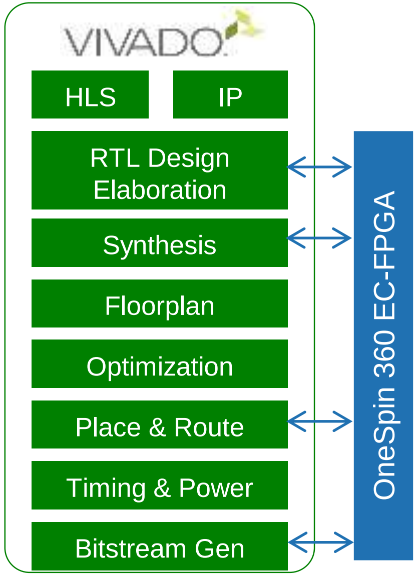
-



- Most of the time sequential optimization only affect part of the design
- Separates the “easy” combinational parts from the sequential parts

Xilinx® Vivado® Flow

Large Design QoR



VIVADO

HLS

IP

RTL Design Elaboration

Synthesis

Floorplan

Optimization

Place & Route

Timing & Power

Bitstream Gen

OneSpin 360 EC-FPGA

XILINX®

- Xilinx significant OneSpin customer Xilinx是Onespin的重要客户之一
- Validate the largest FPGA designs 应用在Xilinx的大型FPGA项目上
- Close cooperation allows full optimization support 支持所有优化技术
- Full support of Vivado flow and device range 支持ISE/VIVADO和器件

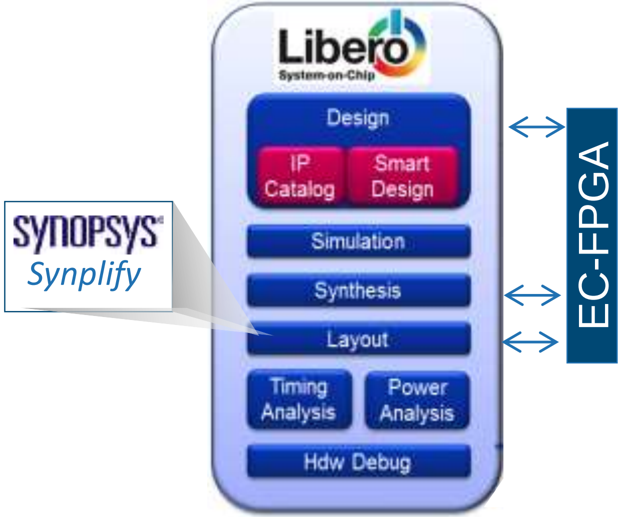
Xilinx Device Support 部分列表

Spartan2, Spartan2E, Spartan3, Spartan-XL, Spartan6, Spartan7, Virtex E, Virtex 2-7, Kintex7, UltraScale+

"OneSpin has a powerful Sequential EC tool, **OneSpin 360 EC**, that we at **Xilinx** use extensively.
It is a technology that should not be ignored!"
Xilinx Engineer, DeepChip, Oct 2015

MicroSemi® Libero® (with Synplify®) Flow

Safety Critical FPGA Assurance



- Full support of Libero flow and MicroSemi devices全面支持Libero和器件
- MicroSemi is EC-FPGA customer MicroSemi 是OneSpin重要客户之一
- OneSpin Synplify partnership:
  up-to-date optimization support 与Synplify优化技术保持同步
- Multiple Safety Critical customers

Microsemi Device Support

Igloo2, Igloo, ProASIC3, Fusion, Smartfusion2, Smartfusion, RTAX, SCA, Axcelerator, eX, MX, RTG4

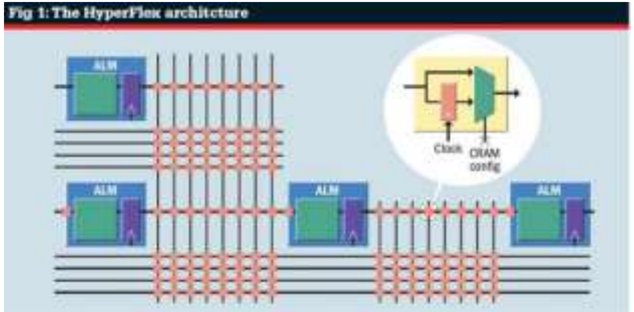
"OneSpin Solutions has created innovative formal-based design verification and equivalence checking solutions that are being used to fully vet some of the most safety critical designs in production today."

Bruce Weyer, Vice President and Business Unit Manager, Microsemi, Inc.

Altera® Quartus® Flow

Latest retimed HyperFlex™ architecture参与验证最新的HyperFlex架构

ALTERA®
now part of Intel



HyperFlex: registers + retiming everywhere

HyperFlex™ ARCHITECTURE

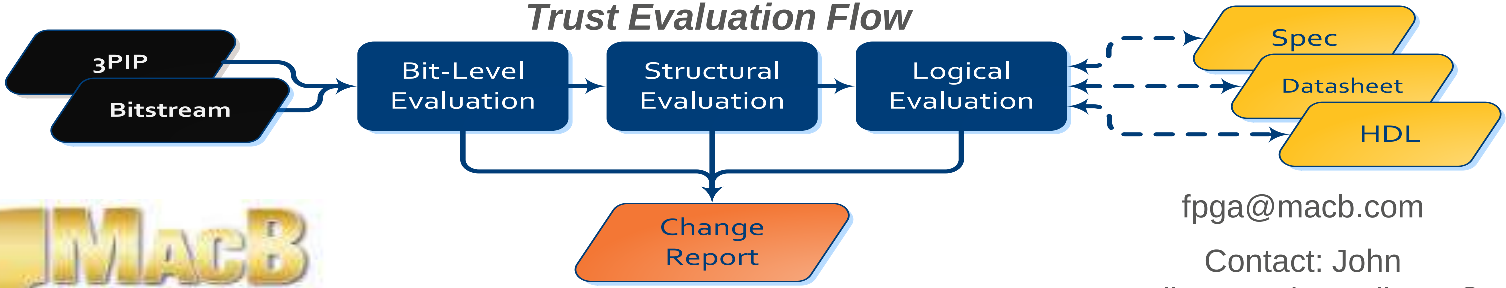
- Close partnership with Altera
  - OneSpin customer
- Currently revamping Altera device support with next-generation Quartus®
  - Delivery: 2017
- Cooperating on leading edge HyperFlex™ retiming technology and RTL-Gates
- Focus on of Stratix10 and Arria10 families

MacAuley-Brown FPGA Trust

EC-FPGA Based Security Solution

- FPGAs are susceptible to bitstream, EDA tool, and 3rd Party IP attacks
  - Attacker could add, remove, or modify function, or exploit a vulnerability
- FPGAs must be trusted at both the silicon and the firmware layers
- MacB Assured (Trusted) Design Assessment Service
  - Automated, guided trust evaluation of FPGA bitstreams and Third-party IP
  - Determine that bitstreams and IP do what they are designed to do …***and nothing more***
  - Certified as a Trusted Microelectronics Supplier for Design Services
  - 11+ years of FPGA Trust Research, Development and Deployed Technologies

Trust Evaluation Flow



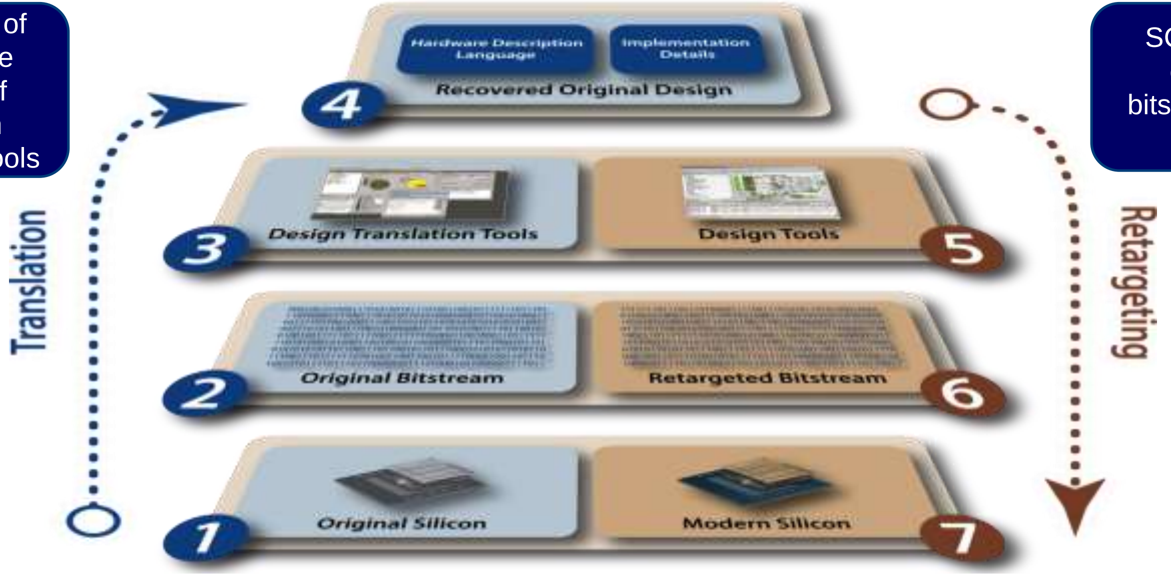
fpga@macb.com

Contact: John Hallman John.Hallman@macb.com

MacAuley Brown

FPGA Design Retargeting

SCC uniquely capable of extracting source code from the bitstreams of obsolete FPGAs with automated translation tools



SCC then retargets the design to an FPGA bitstream deployable on a modern part

Finally, SCC closes the loop by formally verifying the logical equivalence between the design in the new bitstream and the design in the old bitstream

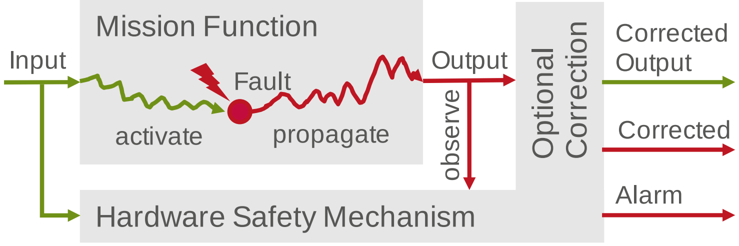
Using OneSpin EC-FPGA

Safety Critical Verification 高可靠性验证

Assuring High Reliability

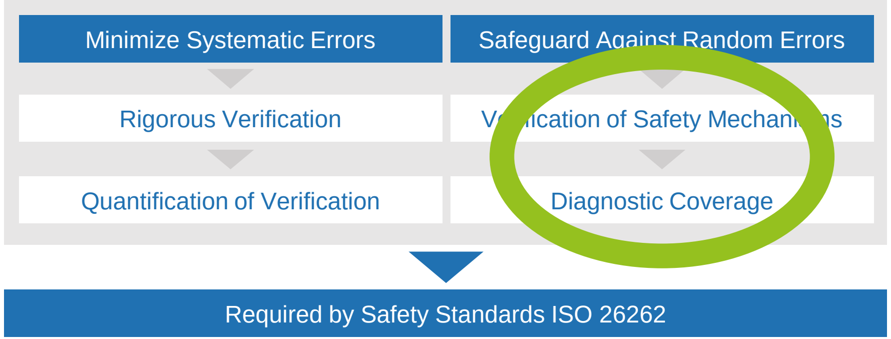
Meeting tough functional safety standards, e.g. ISO 26262 满足26262可编程器件可靠性准则

Minimize systematic errors & protect against random faults 最小化系统级错误以及预防硬件随机错误



- Rigorous, exhaustive formal 严格的全面形式化验证
- Industry-leading quantification and qualification of formal properties 领先的定性、定量分析
- Efficient verification of functional safety mechanism
- 功能安全验证 FAIL-SAFE验证
- Precise diagnostic coverage through formal fault analysis

  可诊断性功能安全覆盖统计

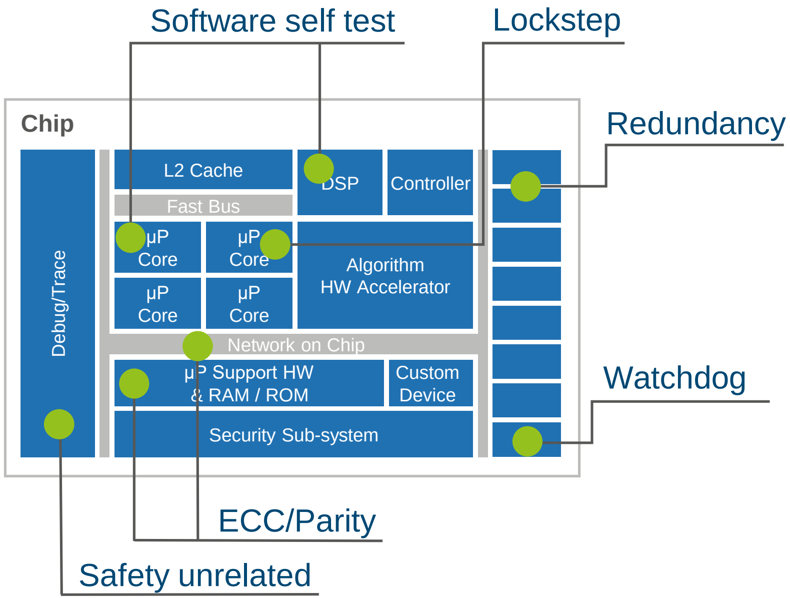

Minimize Systematic Errors
Safeguard Against Random Errors
Rigorous Verification
Verification of Safety Mechanisms
Quantification of Verification
Diagnostic Coverage
Required by Safety Standards ISO 26262

Diagnostic Coverage of the Safety Mechanisms

Quantitative Random Error Analysis

Challenges

- ISO 26262 requires a quantitative analysis of random hardware errors and their outcome
- The quantitative analysis provides key metrics to determine device safety integrity level (SIL)
- New application domains, such as autonomous driving, drive higher safety integrity levels
- Due to the high fault number and diversity / complexity of safety mechanisms, quantitative analysis very challenging and time consuming
- Expert judgment limit reached, automation required



Single Point Fault Metric

ISO 26262 provides and requires the framework

Safe faults

- Not in safety relevant parts of the logic
- In safety relevant logic but unable to impact the design function (cannot violate a safety goal)

Single point faults

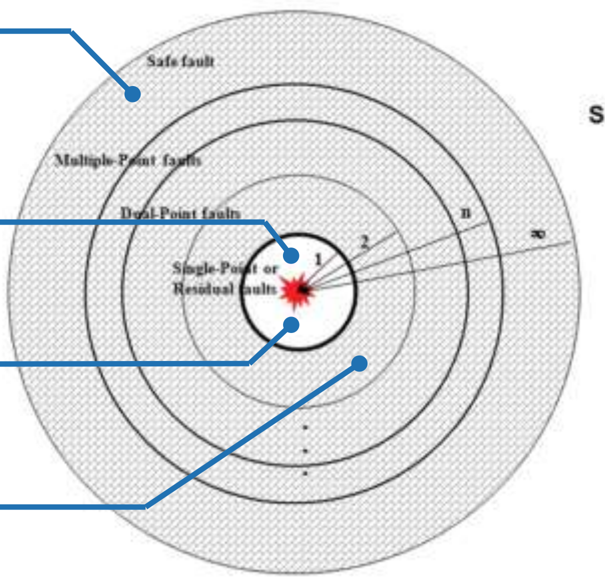
- Dangerous, can violate the safety goal and no safety mechanism

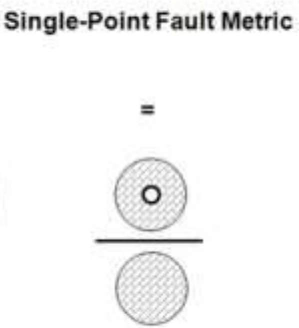
Residual faults

- Dangerous, can violate the safety goal and escape the safety mechanism

Multipoint faults

- Can violate the safety goal but are observed by a safety mechanism
- Sub-classified as “detected”, “perceived” or “latent”





up to 99% required

Diagram: Courtesy International Standards Organization (ISO)

Single Point Fault Metric

Requirement to ensure all “safe” faults are indeed safe

Safe faults

- Unidentified safe fault must be considered residual
- Lower fault metric and diagnostic coverage

Single point faults

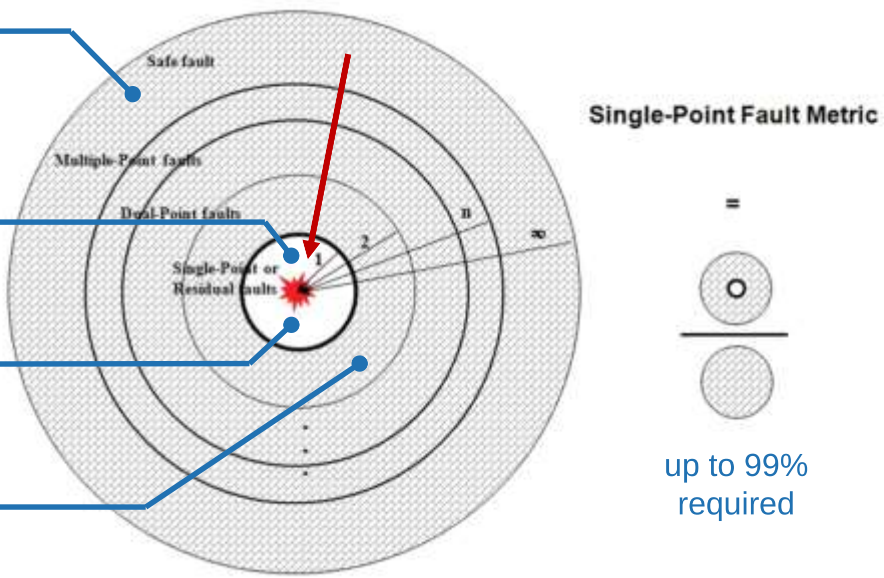
- Dangerous, can violate the safety goal and no safety mechanism

Residual faults

- Dangerous, can violate the safety goal and escape the safety mechanism

Multipoint faults

- Can violate the safety goal but are observed by a safety mechanism
- Sub-classified as “detected”, “perceived” or “latent”



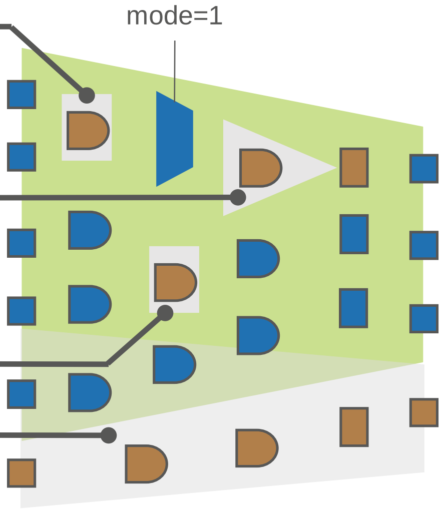
up to 99% required

Diagram: Courtesy International Standards Organization (ISO)

How Can Faults Be Safe?

Considering StuckAt faults only

- Safe faults due to static IC operation modes
  - Debug mode disabled
  - Test logic
- Explicit redundancy in hardware masks the effect of fault
  - Performance impact only
  - States never used in safe operation mode
- Truly redundant logic such as synthesis deficiencies
- Safety unrelated logic
  - Design parts which do not impact the safety goal



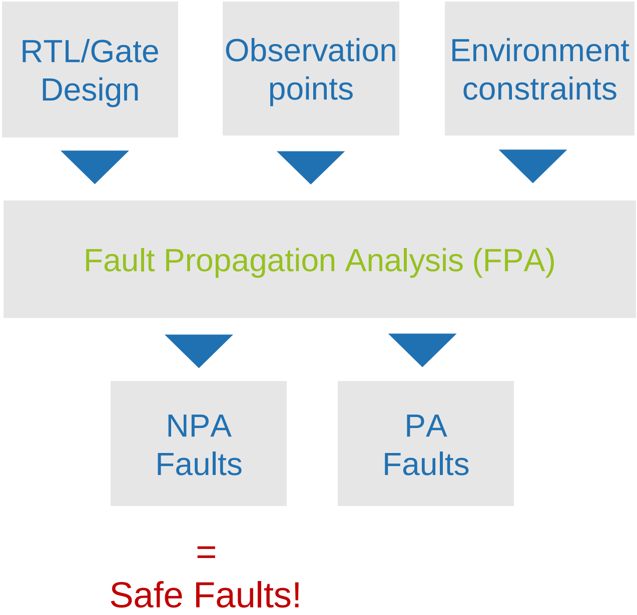
All safe faults cannot propagate to observation points

- Mission outputs
- Internal registers

Formal Fault Propagation Analysis (FPA)

Automated App, no formal knowledge required

- Automatically classifies faults into:
  - Non-propagatable faults (safe)
  - Propagatable faults
    - Dangerous (single point & residual)
    - Potentially detected (multipoint detected)
- Operates at RTL and gate level
- Requires limited additional input
  - Observation points define where faults can propagate (default is primary outputs)
  - Environment constraints, assigned values to test pins, debug modes, control registers
- Fault lists can be provided by the user or generated by the tool
  - StuckAt fault model supported


RTL/Gate Design

Observation points

Environment constraints

Fault Propagation Analysis (FPA)

NPA Faults

PA Faults

=
Safe Faults!

Two Important Modes of the FPA

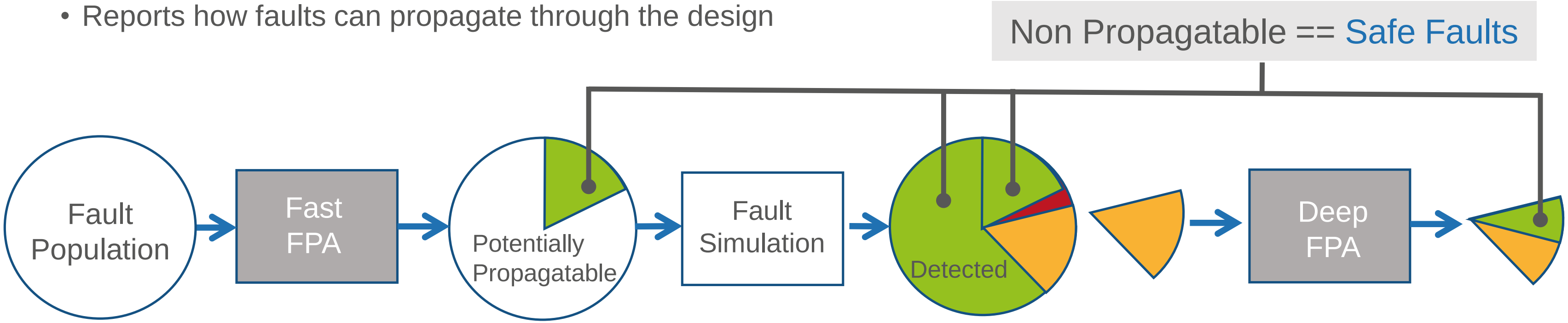
Fast and deep fault analysis

Initial, fast safe fault extraction

- Analysis thousands of faults per second
- Coffee break or overnight run (max) for a complete design
- Find non-propagatable faults only, accelerate fault-sim process

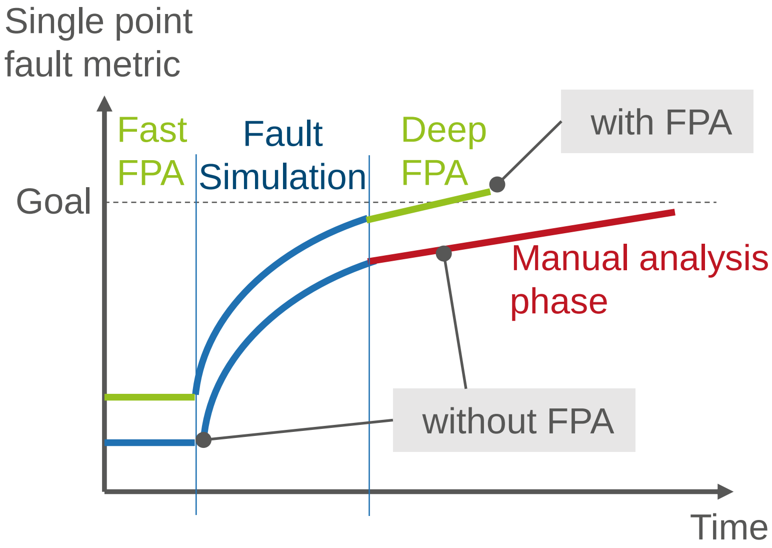
Follow on, deep fault propagation analysis

- Does a deep formal sequential analysis
- Finds non-propagatable (safe) and propagatable faults
- Reports how faults can propagate through the design



Usecase: Formally Identify Safe Faults

- Today engineers use fault simulation to determine fault metrics
- If fault simulation does not achieve metric?
  - Manually identify safe faults
  - Improve safety mechanism, re-simulate
  - Check number of safe faults
- Formal fault propagation identifies safe faults automatically
  - Fast up front analysis to increase safe fault population, improve fault-sim run
  - Deep propagation analysis to close the remaining gap
  - Reduce overall time and increase confidence



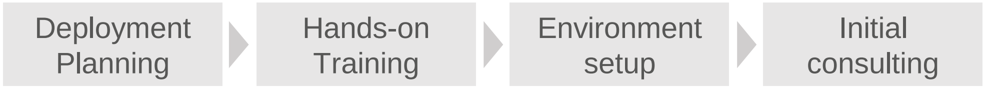
Only formal technology can efficiently combine the design and environmental constraints to bring fault analysis to the next level.

OneSpin 提供的咨询及服务

全面的形式化验证方案

咨询及部署服务

- 专业的AE支持团队
- 可定制的解决方案
- 快速服务模式:


Deployment Planning

Hands-on Training

Environment setup

Initial consulting

合作服务伙伴

- 全方位的专家团队
- 获得OneSpin技术认证的服务机构
- No service / tool provider conflict

> “I want to thank you for the exceptional support you have provided to our team. Your efforts have been instrumental in ensuring success with our project. You and your company have really exceeded expectations”
>
> **US Defense Contractor**
> name withheld by request

解决最难的验证问题

onespin

Design Evaluation
设计缺陷精确定位

4 day certification delay lost money

Design Inspect found severely issues in minutes

快速定位严重缺陷

Inspect now standard part of company flow
成为流程标准环节

Metric Driven ABV
基于断言度量驱动的功能验证

Verification hole identification detected critical bugs 发现深层功能错误

Quantify now drives project management

Presented by Infineon
Infineon实际项目应用情况

IP Integration
IP集成验证

Simulation-based environment slow,
加速验证

Formal-based solution accelerated: development 8X, tool execution 10X

Highlighted by Renesas
瑞萨电子实际项目应用情况

FPGA Quality
FPGA实现验证

Root cause of errors unresolved after several days

Problematic synthesis optimization found in single run发现综合器缺陷

Schedule saved.
逻辑等效一致性验证

部分长期客户：

ALTERA XILINX LOCKHEED MARTIN Westinghouse infineon NOKIA RENESAS

WD Western Digital Microsemi BOSCH Melexis ublox HITACHI Alcatel·Lucent

OneSpin Solutions

A New Spin On Verification

加速验证 面向未来

- DV continuum accelerates schedule, decreases risk
- Targeting next generation verification trends

技术先进性

- Award-winning technology foundation
- 100s years usage & development experience

技术可部署性

- Usability, capacity, performance
- Leading support & service network

更多信息可访问
www.onespin.com