JasperGold Formal Verification Platform: Best-in-Class Formal by Far

Tim Sun
CDNLive Taiwan 2015
Agenda

- Cadence formal and automated verification
- Why formal?
- Next-generation JasperGold® platform
- JasperGold Apps overview
- Focus on debug: Visualize™ interactive environment
- Summary
Cadence formal and automated verification

One year since Jasper merger created world’s biggest formal team

- Industry’s largest development investment in formal
- Industry’s largest support team with global reach
- Cadence® investment continues to grow post-merger
Why formal?
Highest ROI for finding bugs

• Verification crisis continues
  – 70% of fast-rising SoC project costs
  – Formal is the fastest-growing verification segment

  – **Finds more bugs in less time** than simulation
    – With fewer engineers & less compute power
  – **Find bugs earlier** in the design process
  – **Exhaustive**: verifies DUT with all possible stimuli
  – **Higher quality** from formal proofs

• Broad usage by regular D/V engineers
  – Major capacity and performance improvements
  – Intuitive UI and debug
  – Apps automate property creation
JasperGold formal verification apps enable productivity and quality gains through SoC flow

- Automates tasks throughout the design cycle
- Removes risks, improves productivity and quality

Visualize™ Interactive UI & Debug

Savings from Formal
Verification Taxonomy: The Aha Moment

- **Architectural**
  - Micro-Architectural Analysis
  - Proper module partitioning and modeling for formal verification
  - Ensure clean interfaces
  - Bug Avoidance
    - Formal during RTL bring-up
    - Catch bugs early
    - Eliminate throw-away testbench creation effort
  - Bug Hunting
    - Find bugs at block and system level
    - Automation and regression on server-farm friendly

- **RTL**
  - Bug Analysis
    - Investigate late-cycle bugs
    - Isolate corner-case bugs (observed in field, lab)
    - Confirm the correctness of bug fix

- **Post-silicon**
  - Bug Absence
    - Prove critical properties to get 100% assurance
    - May require considerable user expertise and effort

Source: ARM presentation by Laurent Arditi at JUG, Oct. 2013
ARM: High ROI for formal on processor designs

Design bring-up benefits

- Another GPU example:
  - Not more, not less bugs, but the bugs are found much earlier
  - So less RTL changes (code churn), especially late

<table>
<thead>
<tr>
<th>Block</th>
<th>Formal bring-up usage</th>
<th>Bug density total</th>
<th>Code churn total</th>
<th>Code churn bug</th>
<th>Code churn dir</th>
</tr>
</thead>
<tbody>
<tr>
<td>L2C</td>
<td>High</td>
<td>6</td>
<td>242</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>L1C</td>
<td>Medium</td>
<td>8</td>
<td>577</td>
<td>2</td>
<td>34</td>
</tr>
<tr>
<td>LS</td>
<td>No</td>
<td>27</td>
<td>460</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>TEX</td>
<td>No</td>
<td>8</td>
<td>369</td>
<td>4</td>
<td>54</td>
</tr>
<tr>
<td>JM</td>
<td>No</td>
<td>6</td>
<td>265</td>
<td>3</td>
<td>53</td>
</tr>
<tr>
<td>HT</td>
<td>No</td>
<td>10</td>
<td>254</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>

Very few late RTL changes when JasperGold technology is used

Cortex® A12 formal verification results

- Formal with Jasper found 18% of the real bugs in Jira, probably closer to 25%
- Reasonable “garbage” ratio: 20%, exactly similar to simulation. Means low number of false-negatives to debug
- Some bugs would never have been found without formal (clock issues, LSU FSM, X-propagation)
- Formal buildbot usually first to warn about RTL issues

Overall ROI advantage is 2.5x for formal vs simulation

Source: ARM presentation by Laurent Arditi at JUG, Oct. 2013
Next-generation JasperGold formal verification
Combining Jasper and Incisive formal strengths

JasperGold® Apps Platform

- Incisive® common front end
- Incisive run-time integration
- Selected Incisive formal engines

Incisive Simulation and Formal Platforms

- Incisive Enterprise Simulator
- Incisive Enterprise Verifier

Next-generation JasperGold Formal Verification Platform delivers up to 15X performance gain versus previous solutions

Visualize™ Formal Debug and What-If Analysis
Next-generation JasperGold technology: Incisive integration

- No change for existing JasperGold® users
- Easy migration for Incisive® formal users
- Common compilation for JasperGold formal and Incisive simulation users
- Common run-time environment for formal-assisted simulation use models
- Incisive constraint-solving engine integrated with JasperGold technology
- New “Tri” engine based on Incisive multi-cooperating engine technology
Integration with System Development Suite

- SimVision™ Debug
- UVM View
- ESWD
- IDA
- Visualize™ Solution Formal Verification View
- Formal Assisted Debug
- Formal Assisted Verification Closure
- Indago™ Debug Infrastructure
- Incisive® Metrics Center
- Cross-Platform Infrastructure
- Incisive Simulation Engines
- Formal Assisted Simulation
- JasperGold® Apps with Jasper & Incisive Formal Engines
- Formal Assisted Emulation
- Palladium® Emulation & Acceleration
- IEEE Standard Languages & Industry Standard APIs

JasperGold® Apps with Jasper & Incisive Formal Engines

vManager™ Solution

Plan

Construct

Execute

Measure/Analyze
Accelerated VIP = regular VIP minus most of the checkers
  – JG-IPK puts the checkers back – in SVA that can be compiled with Palladium® technology

Also investigating behavioral property synthesis to create SVA from traces and to help manage large quantities of trace data for debug
Formal-assisted verification closure
JasperGold integration with vManager solution

- Take credit in the verification plan for tasks completed by formal
- Completely understandable for non-formal verification managers!
  - Did we cover it?
  - Did it pass or fail?
Agenda

• Cadence formal and automated verification
• Why formal?
• Next-generation JasperGold® platform
• JasperGold Apps overview
• Focus on debug: Visualize™ interactive environment
• Summary
JasperGold formal verification platform

JasperGold® Apps

- **Formal Property Verification App**
- **Behavioral Property Synthesis App**
- **Structural Property Synthesis App**
- **X-Propagation Verification App**
- **Coverage Unreachability App**
- **Design Coverage Verification**
- **Control/Status Register Verif. App**
- **Connectivity Verification App**
- **Sequential Equivalence Checking App**
- **Low Power Verification App**
- **Security Path Verification App**
- **Post-Silicon Debugging**

Visualize™ Interactive UI & Debug

JasperGold Platform plus Incisive® Core Technologies

- **Assertion Based VIPs for AMBA and other common protocols**
- **Programmable Interface via TCL**
- **Parallel and Multiple Engines and Solvers**
- **Links to Incisive Simulation Engines**
- **Links to Metric-Driven Verification Methodologies and Tools**
Formal Property Verification App
Setting the bar for performance, capacity, and usability

- Specify the target and let the formal engines automatically generate the stimulus ("output driven" method)
- Interactively add constraints and extract properties from waveforms
- End-to-end properties provide highest return on investment
Assertion-based VIPs for protocol compliance
Production-proven to shorten time-to-market and reduce risk

- Combined with JasperGold® Apps and Visualize™ interactive UI, assertion-based VIP enable rapid RTL integration and/or protocol customization/extension
- Assertion-based VIP are optimized for formal verification and compatible with all simulators
- Interface protocols:
  - OCP, AHB, APB, ATB, AXI-3, AXI-4, AXI4-Stream, AMBA 4 ACE, AMBA 5 CHI
- Memory controller protocols:
  - SDRAM, DDR/DDR2/DDR3, LPDDR/LPDDR2/LPDDR3, DFI, DFI3
Superlint: Exhaustive set of checks

HDL Analysis Linting (HAL)

- Naming
- Coding style
- Sim Synth mismatch
- Synthesis
- Structural
- DFT
- More

JasperGold® SPS Checks

- Coverage reachability
  - Block, FSM, toggle
- Checks
  - X assignment
  - FSM livelock/deadlock
  - Bus contention
  -Pragma
  - Range overflow
  - Arithmetic overflow
Structural Property Synthesis App
Automatically creates properties for early RTL validation

1. Input design RTL into the app
2. App automatically extracts & sorts properties into assertions and covers
3. Run properties with RTL simulation or formal property verification
   - Lint checks (unintentional latches, out of range indexing)
   - Coverage checks (dead code, FSM checks)
   - Checks for common design errors (arithmetic overflow, bus conflicts, etc.)
4. User assigns “waivers” for illegal / impossible cases for follow-on runs
   - Waivers are persistent as long as corresponding RTL exists
SPS-HAL integration
Violation messages view pane

Filter by severity
Add/remove labels
SPS-HAL integration
Link to source code
Agenda

- Cadence formal and automated verification
- Why formal?
- Next-generation JasperGold® platform
- JasperGold Apps overview
- Focus on debug: Visualize™ interactive environment
- Summary
JasperGold Visualize components

- **Debugging Environment**
  - Why?
  - Highlight relevant logic
  - In-line value annotation
  - Reset analysis
  - Non-linear cycle operations
  - Signal/time anchor

- **Waveform Generation**
  - QuietTrace™ technology
  - What-if/WaveEdit
  - Freeze
  - Replot
  - Visualize -line
  - Visualize -explore

- **Formal Test / Property Capture**
  - Indexed behavior
  - Capture behavior
  - Capture recipe
  - Executable spec

No testbench required.
Visualize environment: key feature summary

- QuietTrace™ technology
- Highlight Relevant Logic
- Preview with values for property or why results
- Why?
- RTL line with value annotation
- What-If
- Minimum trace length
- Property that fails earliest
- Why?
Summary: best-in-class formal by far

- Formal is now a mainstream verification technology
  - Formal apps automate property creation and debug
  - Easy to adopt and deploy

- Formal component of verification mix is growing rapidly
  - Highest ROI of any verification technology
  - Complementary to simulation and emulation

- Industry-leading formal technology now available from Cadence
  - Largest formal R&D team on the planet continues to extend the lead
  - Largest formal applications team to support your adoption
Upgrade your verification with JasperGold® Apps!

JasperGold® Apps

- Common Database
- Common Interface
- Simplified Interaction Between Apps
- Flexible Deployment

**Formal Property Verification App**
- Block-level or end-to-end properties
- Interactive debug, what-if & constraint setting
- High performance and capacity

**Behavioral Property Synthesis App**
- Creates properties from simulation traces
- Automated & manual property ranking
- SHM, VCD, FSDB and PLI support

**Structural Property Synthesis App**
- Automatic RTL checks
- Overflow, dead code, livelock/deadlock...
- Automated & manual property ranking

**X-Propagation Verification App**
- Automatic property generation
- Unexpected X detection & debug

**Control/Status Register Verif. App**
- IP-XACT input
- Comprehensive access policy checks
- Standard & proprietary protocols

**Connectivity Verification App**
- Excel or IP-XACT input
- Sub-system and chip-level connectivity
- Conditional, combinational or sequential connections

**Coverage Unreachability App**
- Proves reachability of coverage holes
- Simulation coverage DB and RTL inputs
- No formal expertise required

**Design Coverage Verification App**
- Provides formal coverage metrics
- Analyzes property set completeness
- Shows verification from bounded proofs

**Sequential Equivalence Checking App**
- Sequential, temporal & functional equivalence
- Reference versus modified RTL
- Side-by-side debug
- Full chip capacity

**Low Power Verification App**
- Verifies power-aware formal model
- RTL and power intent file inputs
- Structural, functional, & sequence checks

**Security Path Verification App**
- Identifies secure data leakage paths
- Verifies data sanctity & fault-tolerant security

**Post-Silicon Debugging**
- Failure signature matching
- Root cause isolation
- Candidate cause elimination

**Higher Capacity**
- Verify complex 100M+ gate designs

**Visualize™ Interactive Debug**
- Modify/create properties on the fly to explore design behavior

**Increased Throughput**
- Utilize multiple proof engines on parallel compute resources

**Wider Deployment**
- Proliferate across engineering teams with unique adoption model

**Assertion-based Verification IPs**
- Certification of AMBA 4/ACE checkers
- Popular standard protocols
- Configurable, illustrative, optimized for formal

**NOTE:** UNR is targeted for Incisive users hence is run with Incisive invocation method (irun)