Incisive® Formal Verification
R&D Update 2014

15 May, 2014
Jörg Müller
Agenda

1. Incisive Verification Platform
2. Incisive Formal Verifier
3. X-Propagation
4. Low Power
5. Summary
Incisive® Platform Combines Simulation and Formal

- Common Parser/Elaborator
- Common Simvision Debug
- Integrated flows (e.g. X, Reg, UNR, LP,...)
- Metric-Driven Verification

Simulation

Formal
Agenda

1. Incisive Verification Platform
2. Incisive Formal Verifier
3. X-Propagation
4. Low Power
5. Summary
Incisive Verification Apps Improve Productivity
And require little to no formal expertise

- Applies the most efficient combination of formal and simulation engines
- Automated property generation and custom debug views
Incisive Formal Core R&D Examples

• **Performance**
  – Engine parallelization
  – Engine collaboration
  – Word-level analysis
  – Semiautomatic and Manual abstractions
  – Assume-guarantee analysis

• **Functionality**
  – New Automatic Checks
  – Coverage and Completeness
  – **X-Propagation** consistency
  – **Low Power**: native UPF/CPF support
  – Integration into Metric Driven Verification
Agenda

1. Incisive Verification Platform
2. Incisive Formal Verifier
3. X-Propagation
4. Low Power
5. Summary
X Propagation

cond = X

if (cond)
data = 4'b0011
else
data = 4'b0101

Gate: data = 0011
RTL: data = 0101

But X is don't care for synthesis

What if synthesis tool optimizes cond=X to cond=1?

Solution – **X semantics** in RTL to resolve X pessimistically

CAT: data = 0XX1
FOX: data = XXXX
X-Prop Application

• **Generation**: IEV is used to generate X checking assertions
  – Clocks, resets, and outputs: never X
  – Flops: once non-X should never be X again

• **Analysis**:  
  – IEV used by designers at the block level
  – IES used by verification engineers at subsystem and above
Hybrid X-Propagation Use Models
Executed in RTL leveraging dynamic and formal

• Reset sequence verification
  – Design must consistently cold or warm boot
  – X-Propagation task is identify real unresolved X values
  – Generate assertions at block level to check for X (automated)
  – Use Incisive X-Propagation simulation to detect and debug at SoC

• Power domain startup
  – Shutoff domains must consistently restart
  – Identify missing isolation (Conformal LP uses static analysis for this)
  – Trace X back to shutdown domain with missing isolation

• Uninitialized memory
  – Memory mapped I/O must be initialized to control logic properly
  – Identify uninitialized memory
  – Trace X back through control logic to uninitialized memory
X-Prop: Differentiated with Incisive platform
Example: Reset Verification

- vManager tracks reset metrics to plan
- IEV app finds potential X issues
  - Generates assertions for simulation
- IES speeds X-prop reset simulation
  - Complex, critical SoC verification problem
  - Previous methodology relied on gate sim
- SimVision speeds X debug
  - Different wave traces for different X source
- Reset verification requires all four
Agenda Review

1. Incisive Verification Platform
2. Incisive Formal Verifier
3. X-Propagation
4. Low Power
5. Summary
Cadence offers complete Low Power Verification Solution

- Functional problems are targeted by low power simulation (IES)
  - Modeling virtual low power intent
- Structural Problems are targeted by Conformal Low Power (CLP)
  - Verifying and Comparing intent with design
- Special problems require functional Formal Property Checking IEV
  - Modeling virtual low power intent
  - Leveraging power of formal engines

→ Formal Low Power Solution (IEV)
Power Aware Formal Property Checking

- **Features**
  - UPF/CPF reading and reporting
    - Supports CPF and UPF 1801-2009
  - Power supply network modeling
  - Isolation and state retention modeling
  - Power-off corruption
  - Assertion control (suspend, abort)

- **User Input**
  - RTL
  - Properties
  - Formal Environment
  - Low Power intent
  - Power controller module (PCM)

- **Flow**
  - IEV models power intent
  - Assertions now start failing
  - Unintended assertion failures are dismissed by assertion control
  - Remaining failures indicate bugs due to low power implementation

Corruption causes design malfunction
SoC Connectivity Checking App
- Verifies connections from high level spec

Added Low Power Intent Modeling
- Introduces isolation, corruption

Enhanced Spreadsheet
- Specify required power domains

Connection failures
- Unintended isolators along connection
- Unintended power domain pass through
- Wrong specification of power domain

Value
- Find LP related issues on SoC level
- Start before full SoC (IPs) available

<table>
<thead>
<tr>
<th>Power Behavior</th>
<th>Src</th>
<th>Dest 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDE</td>
<td>:in_data_i[31:24]</td>
<td>IDE3:data_i</td>
</tr>
<tr>
<td>IDE</td>
<td>:in_data_i[23:16]</td>
<td>IDE2:data_i</td>
</tr>
<tr>
<td>IDE</td>
<td>:in_data_i[15:8]</td>
<td>IDE1:data_i</td>
</tr>
<tr>
<td>IDE</td>
<td>:in_data_i[7:0]</td>
<td>IDE0:data_i</td>
</tr>
<tr>
<td>Reset(PD_IDE0)</td>
<td>IDE0:error_o</td>
<td>CSR:error_0_i</td>
</tr>
<tr>
<td>Reset(PD_IDE1)</td>
<td>IDE1:error_o</td>
<td>CSR:error_1_i</td>
</tr>
<tr>
<td>Reset(PD_IDE2)</td>
<td>IDE2:error_o</td>
<td>CSR:error_2_i</td>
</tr>
<tr>
<td>Reset(PD_IDE3)</td>
<td>IDE3:error_o</td>
<td>CSR:error_3_i</td>
</tr>
<tr>
<td>Reset(PD_IDE3)</td>
<td>IDE0:req_o</td>
<td>AOD:req_i[0]</td>
</tr>
<tr>
<td>Reset(PD_IDE1)</td>
<td>IDE1:req_o</td>
<td>AOD:req_i[1]</td>
</tr>
<tr>
<td>Reset(PD_IDE2)</td>
<td>IDE2:req_o</td>
<td>AOD:req_i[2]</td>
</tr>
<tr>
<td>Reset(PD_IDE3)</td>
<td>IDE3:req_o</td>
<td>AOD:req_i[3]</td>
</tr>
<tr>
<td>Reset</td>
<td>IDE3:data_o</td>
<td>AOD:data_i[31:24]</td>
</tr>
<tr>
<td>Reset</td>
<td>IDE2:data_o</td>
<td>AOD:data_i[23:16]</td>
</tr>
<tr>
<td>Reset</td>
<td>IDE1:data_o</td>
<td>AOD:data_i[15:8]</td>
</tr>
<tr>
<td>Reset</td>
<td>IDE0:data_o</td>
<td>AOD:data_i[7:0]</td>
</tr>
</tbody>
</table>

Isolator breaks connection
New Low Power Apps – Example
Example Example: Reset versus Clamp Value

• Problem:
  – Designer implements IP level reset
  – Low Power architect specifies SoC level isolation
  – Unintended mismatches of reset and isolation value can cause system hang!

• Solution:
  – Analysis of reset value
  – Analysis of isolation value
  – Comparison, reporting and debugging in IEV
  – “Correct Isolation Rule” creation capability

• Benefit:
  – Early detection of mismatches
  – Finding corner case scenarios
  – Identifying not-unique reset values

Design
RTL
Reset
Sequence

UPF/CPF
Domain
Network
Isolation

IEV LP_RVC

Mismatch
caused design
malfunction

TB.inst.a: ISO 0 RST 0 : Match!
TB.inst.b: ISO 0 RST X : No Unique Reset Value!
TB.inst.c: ISO 0 RST 0 : Match!
TB.inst.z: ISO 0 RST 1 : Mismatch!
Agenda

1. Incisive Verification Platform
2. Incisive Formal Verifier
3. X-Propagation
4. Low Power
5. Summary
Summary and Trends

• Apps are successful
  – Provides automated solutions to common specific problems
  – Widely adopted in industry since introduction at DVCon 2012
  – Possibly growing fast in 2014/15

• Apps expanding into new areas
  – X-Propagation
  – Reset verification
  – Low Power

• Apps driving Integration
  – Common Metrics
  – Consistent Semantics
  – Common Debug

• Expert Formal Focus
  – Performance, Capacity, Productivity

Formal Analysis (IFV)
  • Mathematical
  • Breadth-first
  • Static Analysis
  • No Testbench
  • Command: prove

Assertion-Driven Simulation
  • Dynamic Simulation
  • Linear
  • Dynamic
  • No Testbench
  • Command: search

Mixed Approaches