Abstract

Identifying the right balance between simulation and formal verification has always been a challenge. We have learned that in some cases where formal verification is the only viable approach to verify critical features in SoCs because they require exhaustive coverage. These critical features included reset, clock, fuse, power management controller, and so on. Using a couple of power management IPs as an example, this paper describes our experience working with formal technology from verification planning, scoping, to signing off on multiple power management blocks completely. The tools we used are FV Formal and Certitude from Synopsys. We feel the outcome is very valuable and it leads us to establish the flow in the verification process across projects deploying formal verification to fully validate all possible reset states and their potential transition scenarios. Coverage measurements are used as the guidance throughout the flow from beginning to signoff. We hope you can benefit from what we have learned in our experience.

Power Management IP with Formal Verification

- Design characteristics
  - Control logic centric
  - Complex FSM
  - Smaller size blocks
  - Suitable for formal
- Challenges
  - Require formal expertise
  - Ad hoc methodology
  - Manual process
  - Lack of signoff progress measurement

Coverage Rises Up to the Challenge

- “Have I written enough assertions?”
- “Have I over-constrained my formal environment?”
- “How effective are my checkers at catching design bugs?”

Coverage Driven Formal

- SVA Checker Example
  ```sverilog
define high when enter fuse
define low when reset
reset property {
  create property (posedge CLK) disable iff (disable) && !WarmResetFlag
  (stable) iff (IP_RstState == `IP_RST && !BypassFuse && !BistDone)
}
define assertion enable {
  assertion enable (posedge FuseSel) || (!disable) && !WarmResetFlag
  (stable) iff (IP_RstState == `IP_RST && !BypassFuse && !BistDone)
}
```

- SVA Constraint Example
  ```sverilog
define assertion enable {
  assertion enable (posedge FuseSel) || (!disable) && !WarmResetFlag
  (stable) iff (IP_RstState == `IP_RST && !BypassFuse && !BistDone)
}
```

Verification Test Plan Example for a Reset IP

- Verify correct functionality of the FSM
- Verify no deadlock, unexpected uncoverable states, invalid transitions for the FSM
- Verify no X’s are on the inputs/outputs
- Verify reset IP stays in the same state once the “GO” flag for that state is asserted until the “DONE” flag is asserted.
- Verify reset IP pauses in the desired state when “pause” signal is asserted during silicon debug
- Verify “Ready” signal is asserted once the reset completes
- Verify outputs from the reset IP is generated properly as expected.

FV Environment Development

- VC Formal Property Verification Flow

Fault Injection + Formal

- Fault Analysis Results
  - “How effective are my assertions?”
  - “Have I written enough assertions?”
  - “Have I over-constrained my formal testbench?”
  - Coverage to check for property completeness
  - Identify register elements outside of cone of influence of any properties

Measure Property Completeness

- Lightweight, fast run time
- Good for initial assertion coverage measurement

Measure Environment Coverage

- “Have I over-constrained my formal testbench?”
  - Code coverage
  - Deadcode
  - Uncoverable targets due to constraints

Conclusion

- Coverage flow provided guidance and measurable progress
- High confidence in IP quality
- Reduced verification cycle
- Efficiency in resource planning

Summary

- 50 RTL bugs found in 12 power management IPs
- Junior engineers completed formal tasks from start to finish
- Coverage closure deployed in formal verification environment
- Created automated regression flow for all blocks

What We Learned