February 28 – March 1, 2012

The Case for Low-Power Simulation-to-Implementation Equivalence Checking

Himanshu Bhatt               John Decker                     Hiral Desai
MCS                               Architect                          SMCS
Cadence Design Systems       Cadence Design Systems       Cadence Design Systems

Cadence Design Systems
Power Format Unifies Intent
... but each tool uses that information differently

- Does verification have the same isolation model as implementation?
- Are the isolation cells placed in right location on the functional net?
Potential Problem Areas

• Fundamental differences in RTL interpretation between simulation and implementation
  - power rails
  - isolation cells handling

• Methodology is still evolving
  - CPF and UPF specs are not detailed enough to cover all corner cases
  - Power-format is a critical starting point, but tools must make decisions to fill in gaps
  - No formal means exist to compare simulation and implementation
Why is Low Power Unique for EC

• First – If we could do a LEC using simulation database, we would.
  • Fundamental modeling differences make this impossible
• 20 years ago, many issues were found between simulation and synthesis
  - Coding styles were developed
  - Lint checkers and error messages during synthesis added to detect
  - 1000’s of testcases with lots of gate level simulation proved consistency
  - It was a slow, painful process

• Low Power
  - Speed of deployment is much greater than original synthesis
  - Use of gate-level simulation to validate is greatly reduced
  - Nature of LP allows this type of formal proof
Closed Loop Verification Today

- Check Power intent Early
- Simulate and synthesize the same power intent
- Implementation flow
  - Each design transformation uses Equivalency Checking to verify
- Simulation Flow
  - Simulate same source
  - Gate level simulation used to validate the implementation
- Issues:
  - No formal proof that what was simulated matches what was implemented
  - Gate-level simulation check is good but limited
    - Small number of tests run at gate level

Himanshu Bhatt, Cadence
Enhanced Closed Loop flow

- Use LEC to formally prove that simulation matches original power intent and RTL

- Through Sim2Lec, a closed-loop check between the simulation and implementation flows is established
Sim2Lec flow for isolation

Compile the design

Elaborate the design and cpf using an option e.g. ncelab <option name> add_iso.do

Simulate the elaborated snapshot

IES

CLP

set lowpower option  –power_domain_check \ –golden CPF –revised CPF
read library <list>
read design <path_to_design_files> -golden
read design <path_to_design_files> -revised
read power intent <power intent files> -both –cpf

commit cpf –revised
dofile  add_iso.do

commit cpf –golden  –insert_isolation

commit library instance

compare

Revised database using isolation cell info from IES

Golden database

Golden database
Issues Detected by Flow (1)

- **Methodology**
  - Edits to power format for physical implementation are assumed to have no simulation implication
  - User didn’t rerun simulation because it takes too long and they “knew” the change was safe

- **Feed through**
  - Simulated \{A, B\} as concatenation
  - Implemented as feed through with isolation between I0 and I2
  - Result is functionally different between simulation and implementation
  - Both tools “correctly” interpreted the code with the simulator treating operator more literally in accordance with the Verilog LRM
Issues Detected by Flow (2)

- Back-to-back isolation
  - Order of isolation depends on isolation location specified in the power intent (see ex.)
  - Simulator rarely worries about location other than for assigning the correct power domain
  - Logic function can be affected because isolation value seen at the input of PD2 can differ based on the isolation location specified

CPF
Create_isolation_rule –name Iso1 –from PD1 -isolation_output low
  –isolation_target from –isolation_condition X
Create_isolation_rule –name Iso2 –to PD2 -isolation_output high
  –isolation_target to –isolation_condition Y

UPF
Set_isolation iso1 –domain PD1 –applies_to outputs -source_clamp 0
  –isolation_signal X –
Set_isolation iso2 –domain PD2 –applies_to inputs -sink_clamp 1
  –isolation_signal Y

With -location to for Iso1, and –location from for Iso2 (CPF)
Single rail isolation cells can be used.

With -location parent in both CPF and UPF
Future Work

• Current paper discusses isolation
• Extend to check all aspects of the power intent
  – Ensure that the state retention registers between simulation and implementation are consistent
  – Hierarchical Power Intent
    • Domain Mapping/composite domains handled consistently
Summary

• Power formats such as CPF and UPF unify intent across the flow

• Implementation and verification both read the same isolation data, but have different abstractions in which to apply the data

• Simulation to implementation methodology adds formal rules to find bugs introduced when the power-format data is applied in each separate flow
THANK YOU.

QUESTIONS?