Stepping into UPF 2.1 world: Easy solution to complex Power Aware Verification

Amit Srivastava

Madhur Bhargava
Agenda

• Introduction
• Power Aware Verification
• Unified Power Format
• Evolution of UPF
• Why UPF 2.1
• Verification Challenges
• General Migration Tips
• Conclusion
Introduction

Electronic Devices have become
• Complex
• Energy Aware

Require sophisticated Power Management
• Power Gating
• Multi Voltage
• Biasing
• DVFS

Advanced Power Aware Verification
• To ensure functional correctness
Verifying Power Management is even more Complex!!

- Many strategies used in conjunction
  - Power gating, multi voltage, etc.
- Power management structures interact
  - Affect design functionality
- More scenarios to verify
  - Power Modes, Complex protocols
- Various stages of the design flow
  - RTL, Gate Level, Place & Route
- Complete knowledge about power management
  - Power Architecture, Power Cell behavior
- HDLs are not equipped to provide such information
  - Power Intent Specification formats
Unified Power Format (UPF)

• IEEE Standard for expressing Power Intent
  – To define power management
  – To minimize power consumption
  – Especially static leakage
  – Enables early verification of power intent

• An Evolving Standard
  – Accellera UPF in 2007 (1.0)
  – IEEE 1801-2009 UPF (2.0)
  – IEEE 1801-2013 UPF (2.1)

• Based upon Tcl
  – Tcl syntax and semantics
  – Can be mixed with non-UPF Tcl

• And HDLs
  – SystemVerilog, Verilog, VHDL

• For Verification
  – Simulation or Emulation
  – Static/Formal Verification

• And for Implementation
  – Synthesis, DFT, P&R, etc.
Evolution of UPF

- **UPF 1.0 was defined by Accellera**
  - Focused on adding power intent to HDL
  - Relatively simple concepts and commands

- **UPF 2.0 approved in March 2009**
  - Backward compatible with UPF 1.0
  - Supports IP development, refinement

- **UPF 2.0, 2.1 were defined by IEEE**
  - Building on UPF 1.0 concepts
  - Adding new abstractions, flow support

- **UPF 2.1 approved in March 2013**
  - Clarifies and enhances UPF 2.0 features
  - Adds a few new capabilities
Why UPF 2.1

UPF 2.0 has some limitations

• Inability to capture complex scenarios
  – Missing information
  – Gap in Power Aware Verification

• Unclear and inconsistent concepts
  – Different interpretations
  – Non-portable

UPF 2.1 to the rescue!

– New features to address limitations of UPF 2.0
– Provides more clear and consistent semantics
  • Promotes interoperability
Verification Challenge: Repeater Insertion

- Repeaters are inserted at long boundary crossings
- May use incorrect supplies
  - Result in a functional bug
- Need to guide tools to use proper supplies
  - Same information can be used by verification tools
- UPF 2.0 provides some capability
  - Incomplete and lacked proper semantics
- Use proprietary commands to achieve the desired behavior
UPF 2.1 Solution: Repeater Insertion

- UPF 2.1 provides a new strategy command
  - set_repeater
- Similar to other strategy commands
  - Well defined semantics
- More flexible
- Enables verification at RTL stage

```bash
create_power_domain PD_Src -supply {primary aon_ss}
create_power_domain PD_Mid -supply {primary sw_ss}
create_power_domain PD_Snk -supply {primary aon_ss}
set_repeater rep_sw -domain PD_Mid -repeater_supply_set PD_Src.primary -source PD_Src.primary -sink PD_Snk.primary
```
Verification Challenge: Retention Cells

- Master/Slave-alive retention flops
  - Value is retained in always on master/slave latch
  - Occupies lesser area than balloon style retention
  - No additional controls

- UPF 2.0 didn’t model them
  - Verification was dependent on proprietary implementation
UPF 2.1 Solution: Retention Cells

- UPF 2.1 extended retention to master/slave alive cells
  - Well defined semantics to enable early verification
- Use set_retention without –save/restore_signal
Verification Challenge: Soft IP Constraints

- IP providers define constraints related to power management
  - Powering of regions within IP
  - Clamping constraints
  - Retention constraints
  - Power States
- IP integrator has to ensure that constraints are not violated
- UPF 2.0 can model the constraints, but has a limitation

```
create_power_domain pd_softIP \
-include_scope
load_upf soft_ip.upf \ 
-scope softIPinst
#.... Integrate the softIP ...
create_power_domain pd_other \ 
-elements { softIPinst/child }
```
UPF 2.1 Solution: Soft IP Constraints

- Defines Atomic Power Domains
  - `create_power_domain` – `atomic`
- Cannot remove elements from Atomic power domain
- Verification tools can flag error if atomic property is lost during integration

```plaintext
create_power_domain pd_softIP \ 
-include_scope -atomic \ 
load_upf soft_ip.upf \ 
-scope softIPinst
#.... Integrate the softIP ...
create_power_domain pd_other \ 
-elements { softIPinst/child }
```

In-valid for atomic PD

Causes Error
Verification Challenge: Hard Macro Boundary

- Isolation/level shifters need to be placed at the boundary of Hard Macros
- UPF 2.0 requires explicit domain boundary
- Need to be careful of redundant isolation
- Large number of such instances increases the verification complexity
UPF 2.1 Solution: Hard Macro Boundary

- Automatically considers hard macro boundary pin powered by different supply as a boundary
- Avoids creation of explicit power domains
Verification Challenge: Supply Equivalence

- Iso/LS cells depend on source/sink supplies
- Strategy commands use supply sets as source/sink filters
  - requires supply matching
- UPF 2.0 does not define semantics for supply matching
  - Inconsistent interpretation between tools

```plaintext
# ss1 and ss2 are equivalent
create_supply_set ss1 -function { power vdd1 } -function { ground vss }
create_supply_set ss2 -function { power vdd1 } -function { ground vss }
create_supply_set ss3 -function { power vdd2 } -function { ground vss }

create_power_domain pd1 -supply {primary ss1}
create_power_domain pd2 -supply {primary ss2}
create_power_domain pd3 -supply {primary ss3}

set_isolation iso
  -sink ss1
# ... Other options ...
```
UPF 2.1 Solution:
Supply Equivalence

- UPF 2.1 defines rules for matching of supplies
  - Concept of “Supply Equivalence” defined
- New command “set_equivalent”
  - Explicitly state the supply equivalence incase it is not evident in design
- Default to match equivalent supplies
  - When matching to be done only for identical supplies, use option “-use_equivalence” to be FALSE

```python
create_supply_set ss1
create_supply_set ss2
create_supply_set ss3
set_equivalent -sets { ss1 ss2 }
create_power_domain pd1 -supply {primary ss1}
create_power_domain pd2 -supply {primary ss2}
create_power_domain pd3 -supply {primary ss2}
set_isolation iso \
-sink ss1 \
# ... Other options ...
```
Verification Challenge: Strategy interactions

- A complex system may have hundreds of strategies
  - Many strategies may interact
- Relative placement of power management cells
  - How insertion of one strategy cells affects placement subsequent cells
- UPF 2.0 was unclear about strategy interactions
- Problem of interoperability
  - Different tools have interpreted this in different ways

```
create_power_domain pd_tx  
  -elements {tx} \  
  -supply {primary sw_ss} 
create_power_domain pd_sw  
  -elements {mid} \  
  -supply {primary sw_ss} 
create_power_domain pd_rx  
  -elements {rx} \  
  -supply {primary always_on_ss} 
set_isolation iso_tx  
  -domain pd_tx \  
  -sink pd_rx.primary \  
  -applies_to outputs 
set_isolation iso_rx  
  -domain pd_rx \  
  -source pd_tx.primary \  
  -applies_to inputs 
set_port_attributes  
  -domain pd_sw \  
  -applies_to inputs \  
  -repeater_supply always_on_ss
```
UPF 2.1 Solution:
Strategy interactions

• UPF 2.1 defines clear semantics for strategy interaction
• UPF 2.1 defines order of strategy implementation
  – Retention > Repeater > Isolation > Level Shifter
  – A strategy may affect the -source/sink filters of a subsequent applied strategy
• Consistent behavior across tools

```
create_power_domain pd_tx  ~elements {TX} 
  ~supply {primary sw_ss}
create_power_domain pd_sw  ~elements {MID} 
  ~supply {primary sw_ss}
create_power_domain pd_rx  ~elements {RX} 
  ~supply {primary always_on_ss}
set_isolation iso_tx  ~domain pd_tx 
  ~sink pd_rx.primary 
  ~applies_to_outputs
set_isolation iso_rx  ~domain pd_rx 
  ~source pd_tx.primary 
  ~applies_to_inputs
set_repeater ~domain pd_sw 
  ~source pd_tx.primary ~sink pd_rx.primary 
  ~repeater_supply always_on_ss
```
And many more
UPF 2.1 features...

• **Some more additions/extensions**
  – Power cell modeling
  – Hard Macro modeling
  – Supply constraints

• **Some deprecations/restrictions**
  – Supply set functions
  – Supply nets for strategies

• **More clarifications**
  – Supply sets

Refer to full paper for Exhaustive List
General Migration Tips

• Need to **translate** proprietary commands to new UPF 2.1 commands
• Be aware of **deprecations** and avoid using them in new UPF code
  — Refer to UPF 2.1 lrm for details
• Careful about the **syntax changes** and migrate towards new syntax
  — Refer to Table 2 in the Appendix section of the paper
• Understand the **semantics differences** and update UPF code to honor the updated semantics
  — Refer to the Verification challenges and Table 1 in the Appendix
• Be aware of **restrictions** added in the standard and avoid using styles that violate the restrictions
• Use **proper methodology** for achieving verification success and interoperability
Conclusion

• Power Aware Verification has become complex
  – Lots of challenges to verify power management
• Limitations in UPF 2.0 and 1.0 has started to limit power aware verification
• UPF 2.1 has taken leaps to ease the verification burden
  – New additions to fill the gap
  – Clarification of many concepts
  – Some deprecations/restrictions to simply concepts
• A more powerful and finely tuned standard
THANK YOU

Questions ??