Low Power Coverage: The Missing Piece in Dynamic Simulation

Progyna Khondkar, Verification Engineer
Mentor Graphics, A Siemens Business
Agenda: Part One- Problem Formulation

- Identify the Contributors of Low-Power (LP) Coverage data
  - UPF and relevant HDL objects.

- Discuss LP Coverage Computation Uniqueness
  - Power States and Power State Transitions
  - How they are different from non-LP state machines.

- The Missing Piece to Complete LP Coverage Computation Models
  - Semantics for formation of Power State Machines/State-Transition
  - Adaptable database with API
    - to collect, access and represent the Computed LP Coverage.
Agenda: Part Two- Proposed Solution

- To fulfill these **missing pieces**,
  - First identified all the resources of the LP coverage contributors
  - Categorized them in *UPF cover-bins*
  - Further identified *UPF cross-cover-bins* in a complex hierarchical UPF flow
  - Proposed Adaptable and Universal Coverage Database

- **Bonus** – Explanation with *Examples* and *Case Studies*,

PART ONE: PROBLEM FORMULATION
What is Coverage?

Coverage - Meaningful insight into design verification completeness

Coverage Metric - Standardize Verification Measurement

- Describe the degree to which the design is exercised
  - With certain design objects or parameters for a particular test suite / testplan execution
  - Even the test Recapitulated to contribute to the total resultant coverage metric for the design.
- Resultant metrics are stored in a common, unified coverage database (UCDB).

UCDB – Allows Accessibility to further enhance the Coverage Metrics

- With new coverage results from different new sources through coverage merging,
- Mechanism to analyze and generate the coverage reports through API,
  - e.g. Industry standard Accellera UCIS API.
What is Low-Power (LP) Coverage?

- LP Coverage
  - Originates from the abstraction of UPF & Relevant HDL Objects.

- In a LP Dynamic Simulation State Space,
  - Power States and Power State Transitions are asynchronous in nature
  - Power States may refer or depend on other power states
  - Even more than one power state can remain true at a time
  - While it is possible to mark any power state as illegal anytime.

- The Unique & Contradictory nature with non-LP State Machines
  - Make it difficult to formulate LP coverage computation models and
  - Coordinate with standard database like UCDB
Foundations of LP-DV Concepts

- UPF Objects
  - Power Domains
  - Power Supplies
  - Power States
  - Power Strategies, etc.
Characteristics of Power States

- Power states nature
  - Abstract at higher levels and Physical (supply port and nets) at lower levels of designs

- Power States are for
  - Different UPF objects –
    - e.g. Power Supplies, Power Domains, Design Groups, design models, and design instances

- Power states may
  - Denote different operation modes based on
    - Different combinations of Power Domains and their Power Supplies,
    - Reference descendant power domains or power supply states
    - Subject to interdependency between different UPF objects
Key Contributors of LP Coverage

What are the ‘Sources of Power States and Their Transitions’?

**UPF Constructs like:**

- Supply Port States from `add_port_state`,
- Supply Net States from Power State Table (PST),
- PST States from `add_pst_state`,
- Power Domain States from `add_power_state`,
- Supply Port, Supply Net, and Supply Set Function States from `add_supply_state`,
- Power States of the Power Supply Sets from `add_power_state`, etc.
Key Contributors of LP Coverage

Other Sources of ‘Power States and Power State Transitions’?

UPF Strategies

- Isolation “Enable” Signal
- Retention “Save and Restore” Signals
- Power Switch ‘States’ and Power Switch ‘State Transitions’
- Power Switch “Control Port”
- Power Switch “Ack Port”
Key Contributors of LP Coverage

Transitions of Control Signals for UPF Strategies

- High-to-Low &
- Low-to-High Transitions

States of Control Signals for UPF Strategies

- Active
  - Through presenting a value (level sensitive) or
  - Transition (edge sensitive),
- Inactive (opposite to the active)
- Active x (driving unknown)
- Active z (remain floating or un-driven)
Key Contributors of LP Coverage

State Values of Power Switch, Control, and Acknowledge Ports

- ON state,
- OFF state,
- Partial ON state and
- UNDETERMINED (ERROR) state.
PART TWO: PROPOSED SOLUTION
Proposed Foundation on LP Coverage

Whenever the design encounter the ‘**Key Contributors**’

– LP-SIM or coverage analytical engine will generate **UPF cover-bins**.
– We define **UPF cover-bins**, as shown below

**UPF cover-bins**: This is a *counter construct* with *specific decorated items*. These *items are generalized and based on UPF coverage constructs*, i.e.

– name of state,
– status (legal/illegal),
– scope (design scope),
– attribute (ports or nets) etc.

The **UPF cover-bins** represents LP coverage data collected from corresponding UPF coverage constructs.
LP Coverage from LP Dynamic Checks

May Based on:
- LP testbench and LP augmented RTL (Code Coverage),
- Automated LP Sequence Checkers (ISO, Save/Restore Toggle etc.)
- Custom LP Checkers (bind_checker)
LP Coverage from Power States and Power State Transitions

May Based on

- Design controls,
- Supply ports and nets created in the UPF and design,
- Power domains and their power states,
- Supply sets and their states,
- Power Switch States and their Transitions,
- State transitions for ISO, RFF, PSW Control and Ack signals,
Coverage from Cross-Power Domain Power States Dependency

May Based on

- All possible combinations of interdependent power states,
- As well as their possible spontaneous transitions.

```add_power_state PD_top -state SYS_ON {-logic_expr {PD_sub1 == SUBSYS1_ON && PD_sub2 == SUBSYS2_ON}}
add_power_state PD_top -state SYS_OFF {-logic_expr {PD_sub1 == SUBSYS1_OFF && PD_sub1== SUBSYS1_RET && PD_sub2 == SUBSYS2_OFF}}
```
Additional Proposal for the Foundation of LP Coverage

Cross-Power Domain Power States Dependency

- In hierarchical UPF flow
  - Power states and transitions are highly interdependent,
- A new *UPF cross-cover-bins* are defined

*UPF cross-cover-bins*

- Extensions of *UPF cover-bins*,
- But possess additional decoration items to determine the depth of hierarchical crossings
**Semantically Extended**

```
describe_state_cross_coverage
[-domains domains_list]
[-depth cross_coverage_depth]
```

**Dependency Unfolded from Graph**

<table>
<thead>
<tr>
<th>Power Domains</th>
<th>PD_top</th>
<th>PD_sub1</th>
<th>PD_sub2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power States</td>
<td>SYS_ON</td>
<td>SUBSYS1_ON</td>
<td>SUBSYS2_ON</td>
</tr>
<tr>
<td>Power States</td>
<td>SYS_ON</td>
<td>SUBSYS1_RET</td>
<td>SUBSYS2_ON</td>
</tr>
<tr>
<td>Power States</td>
<td>SYS_OFF</td>
<td>SUBSYS1_OFF</td>
<td>SUBSYS2_OFF</td>
</tr>
</tbody>
</table>

Cross-Coverage Data for `-depth=1` (default) for
PD_top -to> PD_sub1 -to> PD_sub2

SYS_ON –to > SUBSYS1_ON –to > SUBSYS2_ON
SYS_ON –to > SUBSYS1_RET –to > SUBSYS2_ON
SYS_OFF –to > SUBSYS1_OFF –to > SUBSYS2_OFF
Case Study: Coverage Computation for UPF Cross-Cover-Bins

add_power_state PD_OUT -state PD_OUT_on {-logic_expr {PD_OUT.primary == PD_OUT_primary_on}}

add_power_state PD_OUT -state PD_OUT_off {-logic_expr {PD_OUT.primary == PD_OUT_primary_off}}

add_power_state PD_OUT -state PD_OUT_ret {-logic_expr {PD_OUT.primary == PD_OUT_primary_off && PD_OUT.default_retention == PD_OUT_ret_on}}

add_power_state PD_OUT2 -state PD_OUT_on {-logic_expr {PD_OUT == PD_OUT_on}}

add_power_state PD_SUBSYS2 -state PD_SUBSYS2_on 
{-logic_expr {PD_SUBSYS2.primary == PD_SUBSYS2_primary_on}}

### configure cross coverage ###

describe_state_cross_coverage -domains {PD_SYS} -depth 3

describe_state_cross_coverage -domains {PD_SUBSYS1} -depth 2

describe_state_cross_coverage -domains {PD_OUT2}
### Case Study: Coverage Computation for UPF Cross-Cover-Bins

<table>
<thead>
<tr>
<th>UPF OBJECT</th>
<th>Metric</th>
<th>Goal</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TYPE : POWER STATE CROSS</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/alu_tester/dut/PD_SYS(ID:PD1),</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/alu_tester/dut/PD_SUBSYS2(ID:PD2),</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/alu_tester/dut/PD_SUBSYS1(ID:PD3),</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/alu_tester/dut/PD_OUT2(ID:PD4),</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/alu_tester/dut/PD_OUT(ID:PD5)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100.00% 100 Covered</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>POWER STATE CROSS coverage instance</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/alu_tester/dut/pa_coverageinfo/PD_SYS/PD_SYS_PS_CROSS/PS_CROSS_PD_SYS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100.00% 100 Covered</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Power State Cross</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bin \PD1:SLEEP-PD2:PD_SUBSYS2_off</td>
<td>2</td>
<td>1</td>
<td>Covered</td>
</tr>
<tr>
<td>bin \PD1:RUN-PD2:PD_SUBSYS2_on-PD3:PD_SUBSYS1_on-PD4:PD_OUT_on-PD5:PD_OUT_on</td>
<td>2</td>
<td>1</td>
<td>Covered</td>
</tr>
</tbody>
</table>
Case Study: Coverage Computation for Power State Transitions

# PSW example for Collecting State Transition Coverage
create_power_switch IN_sw \  
  -domain PD_SUBSYS2 \  
  -output_supply_port {vout_p VDD_IN_net} \  
  -input_supply_port {vin_p MAIN_PWR_moderate} \  
  -control_port {ctrl_p IN_PWR} \  
  -on_state {normal_working vin_p {ctrl_p}} \  
  -off_state {off_state {!ctrl_p}}

# controlling State Transition Coverage by UPF for PSW (IN_sw) shown above
add_state_transition -model IN_sw \  
  -transition {t0 -from {ON} -to {}}} \  
  -transition {t1 -from {ON} -to {OFF}}\  
  -transition {t2 -from {ON} -to {}}} \  
  -transition {t3 -from {ON} -to {ERROR} -illegal}
## Case Study: Coverage Computation for Power State Transitions

<table>
<thead>
<tr>
<th>UPF OBJECT</th>
<th>Metric</th>
<th>Goal</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>TYPE: Power Switch /cpu_tester/dut/IN_sw</td>
<td>0.00%</td>
<td>100</td>
<td>ZERO</td>
</tr>
<tr>
<td>Power Switch coverage instance /cpu_tester/dut/pa_coverageinfo/IN_sw/IN_sw_PS/PS_TRANS_IN_sw</td>
<td>0.00%</td>
<td>100</td>
<td>ZERO</td>
</tr>
</tbody>
</table>

**Power State Transitions**

- illegal_bin ON -> ERROR: 0% 100 ZERO
- illegal_bin ON -> PARTIAL_ON: 0% 0 ZERO
- illegal_bin ON -> OFF: 2% Occurred
- bin dummy: 0% 1 ZERO

**TYPE: Power Switch Control Port /cpu_tester/dut/IN_sw/ctrl_p**

- 50.00% 100 Uncovered

**Power Switch Control Port coverage instance /cpu_tester/dut/pa_coverageinfo/IN_sw/ctrl_p/PS_ctrl_p**

- 50.00% 100 Uncovered

**Power State**

- **ACTIVE_LEVEL**
  - bin ACTIVE: 100.00% 1 Covered

- **INACTIVE**
  - bin ACTIVE: 100.00% 1 Covered

- **ACTIVE_Z**
  - bin ACTIVE: 0.00% 0 ZERO

- **ACTIVE_X**
  - bin ACTIVE: 0.00% 0 ZERO

**Power Switch Control Port coverage instance /cpu_tester/dut/pa_coverageinfo/IN_sw/ctrl_p/PS_TRANS_ctrl_p**

- 100.00% 100 Covered

**Power State Transitions**

- **HIGH_TO_LOW**
  - bin: 2 1 Covered

- **LOW_TO_HIGH**
  - bin: 2 1 Covered
Case Study: Adhoc Approach for LP Coverage Database
Proposed Adaptive Coverage Database

Objects: Are primary holders of information
- They are accessed by handle ID / UPF Handle
- Objects represent UPF, HDL or a relationship between them
- So, there are three major classes of objects
- HDL Objects: Models objects that are representing HDL design
- UPF Objects: Models objects that are created by UPF
- Relationship Objects: Objects that model the relationship between UPF and HDL objects.

Properties: Are collection of information about an object
- They are accessed by property IDs
- Properties are classified into
  - Basic Types: String, Integer, Boolean etc.
  - Complex Types: Handle to properties, list of handles to other objects etc.
- Dynamic properties: Accessible only from the HDL package functions
Algorithm for Adaptive Coverage Database

Initiatives and Proposals at a Glance

- Identify the missing pieces of LP coverage modeling
- Identify the complete source of LP coverage contributors
- Define LP cover bins — the *UPF cover-bins* and *UPF cross-cover-bins*
- Identify LP coverage and testplan association mechanism through UCDB
- Implement standardization mechanism for LP coverage bins through IMDB defined by UPF 3.0
- Extend LP cover bins in IMDB as subset of UCDB
- Identify database accessibility through mapping HDL API defined by both UPF 3.0 and UCDB standards
- Propose adaptive coverage database through UPF 3.0 in IMDB and extend it with UCDB standard for integrating the non-LP coverage, And
- Identify the requirements of heterogeneous merge algorithms for merging LP and non-LP data in UCDB
Heterogeneous Merging of LP and non-LP Coverage in IMDB

- SIM Coverage
- Non-PA Coverage IMDB (UCDB)
- PA Coverage IMDB (UCDB)
- Combined Testplan IMDB (UCDB)
- Combined IMDB (UCDB)

Autogenerated LP-Testplan
Linked
Master Testplan
Conclusions

- Completed the initial framework for ‘A Complete LP coverage’ Computation Model

- Also Standardization and integration with existing UCIS coverage database

- Further research is required to completely map
  - The functionalities of HDL API defined by both UPF 3.0 and UCDB standards