May the powers be with you! –
Unleashing powerful new features in
UPF IEEE 1801

Srinivasan Venkataramanan, DV Consultant, VerifWorks LLC,
srini@verifworks.com,
Ajeetha Kumari, VerifWorks
Thejas @CVC
Agenda

• Introduction
• Sample SoC, UPF basics
• Model Simstates
• Sim Reply control
• Sim Assertion Control
• Information Model In UPF
• Automating PST coverage closure
• Summary
Introduction

• Low power requirements - omnipresent now
• IEEE 1801 UPF - a TCL based language to capture Low Power requirements
  – Various levels of abstraction
  – Various tools consume & update UPF code
• UPF evolution:
  – 1.0, 2.0, 2.1 and 3.0
  – New features, enhancements, deprecations
• We highlight few key new features/powers of UPF 3.0
Concepts in Low Power DV

- Power Domains
- Isolation mechanism
- Level Shifters
- State Retention
- Power State Tables
Sample SoC – first-cut UPF

- Ref: Google’s Opentitan RoT chip
- Listing power domains is easy
  - Architecture
  - Why that PD? – different topic
- Coding them in UPF can be tedious
  - First timers
  - Too many options, verbose TCL
- We used a UPF generator
  - DVCreate-LP
  - List all hierarchies (upto a depth)
  - Let users pick

Credits: Google
Power State Tables - PSTs

- Next step - identify the necessary power strategies
- Power State Table (PST) - necessary to arrive at appropriate strategies
- PST - a detailed list of various modes in which the design is intended to be used.

<table>
<thead>
<tr>
<th>Mode</th>
<th>RISC_V_PD</th>
<th>MEM_SUBSYS_PD</th>
<th>PERI_PD</th>
<th>USB_11_PD</th>
<th>TOP_PD</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>LP0</td>
<td>ON</td>
<td>ON_LOW</td>
<td>ON_LOW</td>
<td>ON_LOW</td>
<td>ON</td>
</tr>
<tr>
<td>LP1</td>
<td>ON_LOW</td>
<td>OFF</td>
<td>OFF</td>
<td>ON_LOW</td>
<td>ON</td>
</tr>
<tr>
<td>SLEEP</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
</tbody>
</table>
Concept of SIMSTATE in UPF

- Simstate defines precise simulation semantics in a given power state.
- Used to describe the expected behavior of the cells connected to this supply set.
- UPF 1.0 – had basic support:
  - NORMAL (When ON)
  - CORRUPT (When OFF)
- Designs have more than the above two states:
  - Normal as long as there is no new activity/change etc.
Power Mode changes in simulation

```
g2u_lp_supply_on("VDD_IP1", 1.08)
```

Can be coded inside UVM SEQ

```
function bit supply_on
( string pad_name,
  real value = 1.0);
endfunction : supply_on

function bit supply_off
( string pad_name);
endfunction : supply_off

begin
  int status;
  status = supply_on("VDD_IP1", 1.08);
  if (!status) begin
    `uvm_error("VDD_IP1",
      "Unable to turn supply_on on given supply net, check net name")
  end
end
```
SIMSTATEs in waveform

- Mentor’s Questa SIM
Sim Reply Control

• Verilog semantics of *initial* block ensures that these blocks get executed at time 0 of a simulation

• What about ON $\rightarrow$ OFF $\rightarrow$ ON transition?
  – Re-load memories ($readmemh$)
  – Re-initialize PLL/ADC/DAC etc. parameters/states
  – Verilog-only semantics do NOT do these 😞

• sim_replay_control - enables this modelling
Sim Reply Control

```plaintext
add_power_state -supply PD_MEM.primary \  
  -state { NORM_pst_ss -supply_expr \  
    { (power == {FULL_ON 1.8} ) && \  
      (ground == {FULL_ON 0} ) } } }

sim_replay_control \  
  -elements {*} \  
  -domain PD_MEM \  
  -model top_earlygrey \  
```

Sim Assertion Control

• Consider the assertions inside power down logic, they need special care to ensure they do not provide false negatives.
• UPF 3.0 provides control over these assertions written in SVA.

```
sim_assertion_control \
  -elements {*} \ 
  -domain PD_IP1 \ 
  -model chip_top \ 
  -type suspend
```
Introduction – UPF information model

- Reflection APIs – great for automation
- UPF has Information Model API
- Query:
  - PDs
  - Strategies/ISO/LS etc.
  - PSTs
- Applications:
  - Custom LP rule checker
  - Automated PST closure etc.
Automating Power-coverage closure

- UVM is widely used in TB domain
- PSTs define a state-space for UVM to hit/cover
- Manual coding of UVM sequences to hit each state/transition
  - Time consuming
  - Error prone, debug cycles
- We used a tool to generate UVM sequences
  - Inputs: UPF PST, config knobs, UVM base SEQ
  - Use Information Model/custom API to query the PST
  - Generate UVM sequences
Automating PST coverage closure

<table>
<thead>
<tr>
<th>Mode</th>
<th>RISC_V_PD</th>
<th>MEM_SUBS_VPD</th>
<th>PERI_PD</th>
<th>USB_11_PD</th>
<th>TOP_PD</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>LP0</td>
<td>ON</td>
<td>ON_LOW</td>
<td>ON_LOW</td>
<td>ON_LOW</td>
<td>ON</td>
</tr>
<tr>
<td>LP1</td>
<td>ON_LOW</td>
<td>OFF</td>
<td>OFF</td>
<td>ON_LOW</td>
<td>ON</td>
</tr>
<tr>
<td>SLEEP</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
</tbody>
</table>

UPF Information Model
Summary, Q&A

• Generate first-cut UPF via tool
• SIMSTATE enhancements explained
• New simulation control features introduced with use cases
• Information model in UPF
  – Programming API to Low Power structures
  – Next generation innovation in LP Design Verification
• THANKS!
• Questions?