The UPF 2.1 Library Commands: Truly Unifying the Power Specification Formats
Amit Srivastava, Awashesh Kumar, Vinay Singh
Mentor Graphics Corp. 8005 SW Boeckman Rd. Wilsonville, OR 97070

Abstract

Today’s low power SoC’s involve a variety of IPs including hard-soft macros and complex power management architecture described by IEEE 1801 UPF. This architecture is implemented by special power management cells like isolation/level shifter/retention, etc. The earlier versions of UPF described commands for capturing the power intent but not the library cells or hard macros. The designers have to rely on other standards like Liberty formats to completely capture the details of such cells. The UPF 2.1 (latest version) has introduced new commands to capture the intent of special power management cells and the hard macros. In this paper, we take a deeper look at library commands in UPF 2.1 and analyze their merits and demerits.

Why Library Commands?

- IPs have power management architecture which is pre-implimented.
- It is important to know power management details at the interface of IP for proper integration in SoC environment.
- A typical set of information comprises of the details about pg_pins, related_supply, operating voltages of the supplies, information about power management cells etc.
- UPF 2.0 relied on external formats (Liberty) to describe power management information of pre-implimented IPs.
- The lack of well defined semantics of low power attributes in Liberty complicates the situation, often causing ambiguity in tool behavior.

UPF 2.1 Library Commands

UPF 2.1 addressed the limitations of earlier version of UPF by introducing two sets of commands to capture power intent for hard IPs and power management cells.

Power Model Commands

begin_power_model, end_power_model:

These commands define a power model using other UPF commands. UPF commands can be used to capture a number of liberty attributes –
- power/ground_pg_pins of IP
- related_power_port, related_ground_port of the IP pins
- power_down_function of IP pins

Example Syntax:

begin_power_model upf_model
... UPF Commands ...
end_power_model

apply_power_model:

Instantiates the power model and allows connection of supplies via supply_set handles.

Example Syntax:

apply_power_model upf_model =elements 1/2 -> supply_map (PD.ssh1 ss1) (PD.ssh2 ss2)

Power Management Cell Commands

- Identifies the power management cells present in the library and define the characteristics of those cells. Captures following key details about the power management cell:
  - Identifies a power management cell.
  - Identifies the supply pins of the power management cell. For example, for always on, on the cell options – power/ground specify the power and ground pins of the cell respectively. Wherein the options power_switchable/ground_switchable specify the power and ground pin connected to a switchable power supply.
  - Identifies the control pin of the power management cell.
  - Specify expressions in those pins which can be used to specify additional conditions. E.g., --save_check/-restore_check options of the define_retention_cell command

UPF 2.1 provides following define_* commands:

define_always_on_cell:
  Identifies always on cells

define_diode_clamp:
  Identifies the diode clamp cells

define_isolation_cell:
  Identifies isolation cells

define_level_shifter_cell:
  Identifies level shifter cells

define_power_switch_cell:
  Identifies power switch cells

define_retention_cell:
  Identifies state retention cells

Migration from Liberty to UPF 2.1 Library Commands

The following table shows how a hard macro modeled in liberty can be expressed using UPF 2.1 commands:

<table>
<thead>
<tr>
<th>Power model defined in liberty</th>
<th>Explored UPF 2.1 commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>library_cell_1 (VDD) {</td>
<td></td>
</tr>
<tr>
<td>related_power_pin       }</td>
<td></td>
</tr>
<tr>
<td>library_cell_2 (VSS) {</td>
<td></td>
</tr>
<tr>
<td>related_ground_pin      }</td>
<td></td>
</tr>
<tr>
<td>library_cell_3 (PD.ssh1) {</td>
<td>library_cell_3 (VDD)</td>
</tr>
<tr>
<td>related_power_pin       }</td>
<td>related_power_pin</td>
</tr>
<tr>
<td>library_cell_4 (PD.ssh2) {</td>
<td>library_cell_4 (VSS)</td>
</tr>
<tr>
<td>related_ground_pin      }</td>
<td>related_ground_pin</td>
</tr>
</tbody>
</table>

Verification Impact of UPF 2.1 Library Commands

Power model commands:

- Check for consistency during IP integration process
- Detect when an incorrect supply is connected to IP which operates at voltage levels beyond the normal operation of IP
- Clamp value of isolation cells with constraints specified in IP using set_port attributes command
- May provide power aware simulation behavior for non-PA simulation models by applying simulate semantics on the pins of the IP as per power state description

Power management cell commands:

- Check to ensure the verification matches the power management cell requirements.

Example:

define_level_shifter_cell -cells (LS Cell) {
  -level_shifter_lsi =rule low_to_high 
  -use_interface_cell my_interface 
  -set_strategy lsi 
  -lib_cells LS_CELL 
  -for function
}

Conclusion

The Library and power model commands in UPF have bridged the gap in UPF for capturing the power management of library cells. This truly unifies the UPF and provides a comprehensive standard for expressing the power management of low power designs. The tighter integration of library commands with power intent commands and more complete semantics ensure that users get more flexibility and reuse capability in capturing the information related to power management. This will result in much more comprehensive verification at earlier phases thereby reducing the costly re-spins.

Pros & Cons of Using UPF 2.1 Library Commands

Pros:

- Better compatibility with other UPF commands.
- Flexibility in capturing key features, such as power states.
- Concise representation of related power information of power management cells using define_* commands.
- Lack of well-defined semantics in Liberty causes different implementations.
- Liberty files may not be available in the early phases in the verification cycle.

Cons:

- Relatively new, not many tools support them.
- Legacy libraries present in user flows.

References