2023 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

SAN JOSE, CA, USA
FEBRUARY 27-MARCH 2, 2023

Hierarchical UPF Design – The 'Easy' Way

Brandon Skaggs, Chris Turman, & Joe Whitehouse

Cypress Semiconductor, an Infineon Technologies Company

Agenda

- Motivation and Contributions of this Paper
- Simple Hierarchical UPF Design
- Common Hierarchical UPF Design Issues with Parameterizable IP
- Experimental Results
- Analysis & Concluding Remarks

Motivation

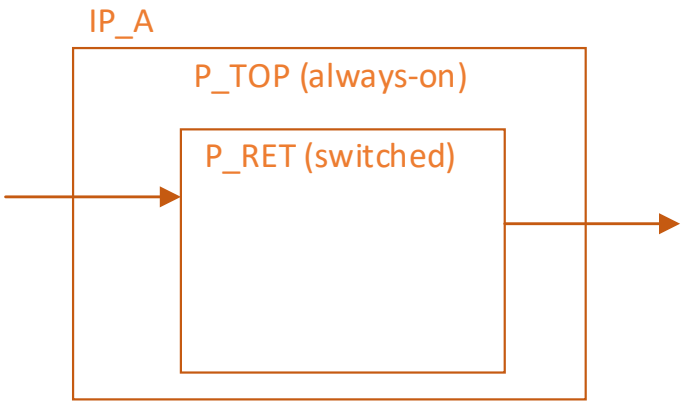
- Time-to-market concerns drive reuse and hierarchical approaches to design and verification…
- Reuse of IP in RTL design is accomplished with clear definitions of scope, port maps, SV interfaces, parameters, etc.…
- But UPF language gaps – or gaps in implementation by EDA vendors – makes reuse of IP power intent less straightforward.
  - Not all language options are supported across tool vendors; some language options do not do what you would expect…
- An understanding of these issues can inform best practices – and possibly suggest improvements in future IEEE 1801 revisions.

Contributions

- This paper presents a case study for a hierarchical UPF design approach using highly-parameterized, power-aware IP– including the specific language constructs attempted and the issues encountered.
- This presentation will:
  - Present a simple hierarchical UPF design for discussion
  - Discuss common issues related to hierarchical UPF design with IP re-use
  - Present experimental findings
  - Conclude with a review of final recommendations – including possible enhancements to future UPF LRM revisions

Simple Hierarchical UPF Design

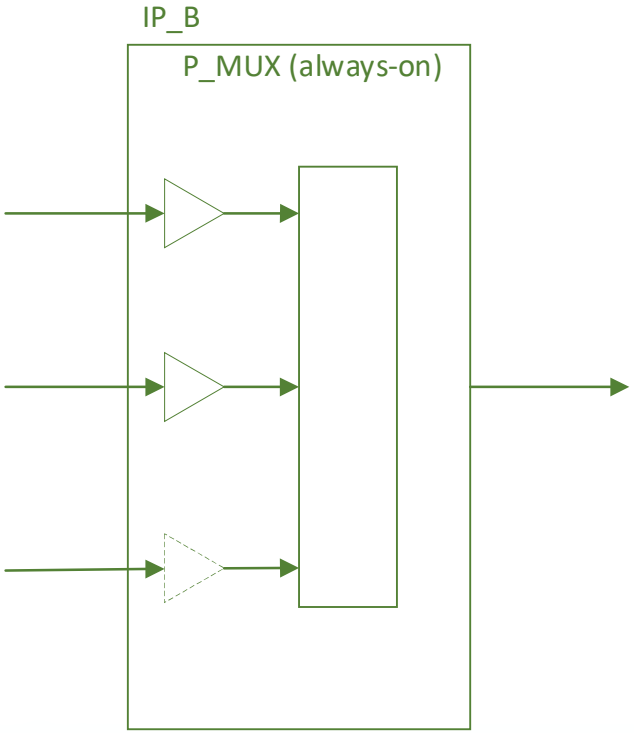
IP A



State	*P_RET* Voltage
OFF	Off
ON	1.0
RET	0.7

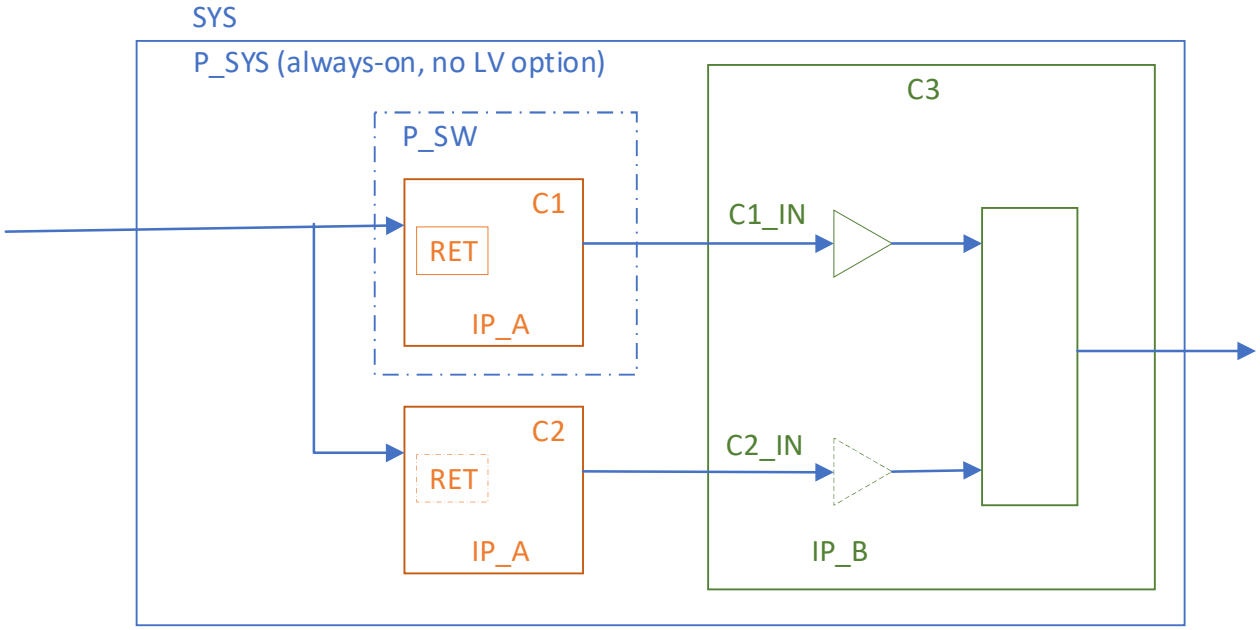
- Top domain *P_TOP* is relatively always-on compared to subdomain *P_RET*.
- *P_RET* domain can be switched off – with key elements retained.
- Retention cells can optionally be run at lower voltage.

IP B



- Top domain *P_MUX* is relatively on compared to the system.
- Inputs to IP_B contain ISO strategies to handle situations where driving supplies are switched off.

System



- C1 instance of IP_A has *P_RET* switched off
- C2 instance of IP_A has *P_RET* only in ‘ON’ state.
- C2 retention and C2_IN isolation not needed

Common Hierarchical UPF Design Issues with Parameterizable IP

Scoping Issues

- UPF *load_upf* command allows loading of IP UPF at a specific System hierarchy; this allows instance paths, ports, etc. written at IP scope to resolve.
- However, objects loaded at a given scope have their own namespace:
  - C1/*PD_TOP* is unique and independent with respect to C2/*PD_TOP*
- There is no ‘set equivalent domain’ language within the UPF LRM…
- The *create_composite_domain* command was added in UPF 2.1 for ‘clubbing’ domains into groups…

Supply Set Abstraction

- UPF provides the concept of *supply sets* for grouping and handling supply nets collectively.
- Commands exist for associating supply sets with domains, setting supply sets as equivalent for analysis, and connecting supply sets to macro *pg_pin* type pins…
- There is also a *connect_supply_set* command for implicitly connecting a supply set to a given set of elements. Looks easy enough…

Specifying ‘Optional’ Strategies

- Re-usable IP could be instantiated in contexts where defined strategies are not required.
- Isolation strategies *can* be written to only be active when a true boundary exists (i.e., *-diff_supply_only* option); however, there is no equivalent option for retention strategies.
- Parameterization of the IP UPF with TCL variables is an option; *load_upf_protected* (UPF 2.0) and *load_upf* (UPF 3.0+) both provide options for passing parameters to IP UPF via TCL variables…

IP Power State Reuse

- Support for hierarchical power states has been available in UPF since the UPF 2.0 LRM.
- However, IP could define power states that may not be used—depending on the instantiating system power states.
- Important to identify if defined system power states conflict with IP power state definitions…
- …but no UPF language facility for specifying IP or system power state precedence.

Experimental Results

Hierarchical Design Case Study

- Existing (mature) design of a modest-sized ‘wearable’ SoC was re-written using hierarchical approach.
- The most abstract methods available in the UPF LRM for integrating IP power intent at system level were attempted; more explicit language was used only when issues were encountered.
- The results from simulation, static multi-voltage rule checking, synthesis, and layout tools were compared to existing (flat) method.

Hierarchical Design Case Study

- Preferred production versions of EDA tools were used; notes are added where newer versions had improved support.
- ‘N/A’ table entries represent situations where support for the given command was not evaluated due to limitations in other portions of the flow.

Results: IP/System Scoping

- The *create_composite_domain* command was not well supported:

UPF Command	Simulator A	Synth Tool B	Formal Tool C	Phy Imp Tool D
create_composite_domain	Supported	Not supported	Not supported	Not Supported

- Furthermore, Section 6.13 of the UPF 2.1 LRM indicates it was never intended for the creation of implementable domains:

> *"A composite power domain is a simple container for a set of power domains. Unlike a power domain, a composite domain has no corresponding physical region on the silicon…"*

Results: IP/System Scoping

- Impact on implementation: manual handling of each newly created IP-scoped domain…

```
...
set_attribute library_domain lib_list1 [find / -power_domain PD_SYS_AON]
set_attribute library_domain lib_list1 [find / -power_domain C1/PD_TOP]
set_attribute library_domain lib_list1 [find / -power_domain C2/PD_TOP]
...
set_attribute library_domain lib_list2 [find / -power_domain PD_SYS_SW]
set_attribute library_domain lib_list2 [find / –power_domain C1/PD_RET]
set_attribute library_domain lib_list2 [find / -power_domain C2/PD_RET]
...
```

- Maintenance is tedious and error-prone…
- **The UPF LRM should provide a method for setting domain equivalence.**

Results: Supply Set Abstraction

- The commands related to assigning supply sets were not well supported by the EDA tools:

UPF Command	Simulator A	Synth Tool B	Formal Tool C	Phy Imp Tool D
associate_supply_set	Supported*	Partial: domain	N/A	Not supported
connect_supply_set	N/A	Not supported	Not supported	N/A
set_equivalent	N/A	Not supported	Supported	Not Supported

* Bug fixed in newer release

Results: Supply Set Abstraction

- Design reverted to making explicit supply port/supply net connections…
- **UPF constructs that operate on supply nets should be supported by EDA vendors in a manner that allows connectivity – not just equivalence for power state table analysis.**

Results: ‘Optional’ Strategies

- Synthesis results showed that the presence of a retention strategy was sufficient to cause retention cells to be used – regardless of whether the power state table indicated retention was needed.
- Ultimately handled by including separate UPF files with retention strategy—loaded only for instances known to require retention, i.e.:

```
...
load_upf ip_a.upf C1
load_upf ip_a_ret-strategy.upf C1
load_upf ip_a.upf C2
...
```

- **The UPF LRM should include an ‘if necessary’ option to retention strategies to allow optional retention strategies.**

Results: ‘Optional’ Strategies

- UPF LRM commands that would manage parameterized IP intent via TCL variables in a name-scoped way were found to be not well supported by EDA vendors:

UPF Command	Simulator A	Synth Tool B	Formal Tool C	Phy Imp Tool D
load_upf_protected (2.1)	Supported	Version-dependent (1)	Not supported (2)	Not Supported
load_upf (3.x)	Supported	Not Supported	Not Supported	Not Supported

(1) Newer version added support
(2) Command supported, but ‘param’ option of command not supported

Results: ‘Optional’ Strategies

- Handling IP parameterization via TCL variables manually is possible but also risky.
- All required TCL variables must have safely-defined default values; however, this creates the possibility of namespace collisions
  - Any TCL variables used by an IP power intent would share a namespace with the sourcing (top-level) UPF.
  - Newly-defined variables at top-or IP-level could have unintended impact on unrelated portions of the design if names collide…
- **EDA vendors must support UPF language constructs that provide a mechanism for parameterized intent definitions that avoid namespace conflicts**.

Results: IP Power State Reuse

- Experimentally, it was difficult to reuse IP power state tables where unused (IP) states existed (e.g. instance C2's *OFF* and *RET* states)
- Formal tools complained of 'unused' IP states:

```
// Error: (1801_PST_STATE_DROPPED_ROOT) Power state specified at root level is not
//   consistent with all the power state tables and is being ignored (occurrence:1)
```

- In other cases, conflicts between IP and system power state tables resulted in the 'dropping' of *both* – which resulted in incorrect crossover analysis.
  - Recommendation from the vendor was to disable IP states via tool option…
- **The UPF LRM should provide a mechanism for specifying how to resolve IP to system conflicts**.

Summary of Results

- Care must be taken to maintain lists of IP-scoped domains for use in synthesis and PNR scripting…
- Direct net-to-port supply connections are the only reliable way to make connections across UPF scope.
- ‘Optional’ IP retention strategies should be maintained in independent UPF files to make them easier to apply only where needed.
- IP and system power state definition conflicts can be difficult to resolve; defining the system power states from top-down can avoid this.

Analysis & Conclusions

Analysis

- The UPF LRM should provide a mechanism for explicitly setting domain equivalence between parent and submodule domain definitions.
- UPF constructs that operate on supply nets should be supported by EDA vendors in a manner that allows connectivity – not just equivalence for power state table analysis.
- The UPF LRM should include an ‘if necessary’ option to retention strategies to allow optional retention strategies within a switched supply.

Analysis

- EDA vendors must support UPF language constructs that provide a mechanism for parameterized intent definitions that avoid namespace conflicts.
- The UPF LRM should provide more flexibility when defining power state tables – to enable IP to specify which states can be dropped/overridden at integration level and which should be considered essential.
- Additionally, the LRM should define clear rules for defining hierarchical states and how conflicts should be handled.

Conclusions

- Design complexity and time-to-market requirements continually push for more efficient leveraging of design and verification of subcomponents, and practical hierarchical low-power design methodologies are critical for accomplishing this.
- However, while the components of this methodology are in place, limited support by EDA vendors for the more ‘contemporary’ LRM concepts—as well as some (albeit minor) inconsistencies in the UPF LRM itself—stand in the way.
- As we approach the tenth anniversary of the LRM revision that provided the constructs and methodology to make it possible, this paper identifies that the practical hierarchical reuse of power intent—while possible—is a long way from being ‘easy’…

2023 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

SAN JOSE, CA, USA
FEBRUARY 27-MARCH 2, 2023

Questions?

Thank you for your attention!

accellera SYSTEMS INITIATIVE