2020 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

UPF 1.0, UPF 2.0, UPF 2.1, UPF 3.0, and now UPF 3.1: The big Q "Which is the Right Standard for My Design"?

Madhur Bhargava, Mentor, A Siemens Business

(madhur_bhargava@mentor.com)

Mentor® A Siemens Business

accellera SYSTEMS INITIATIVE

2020 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

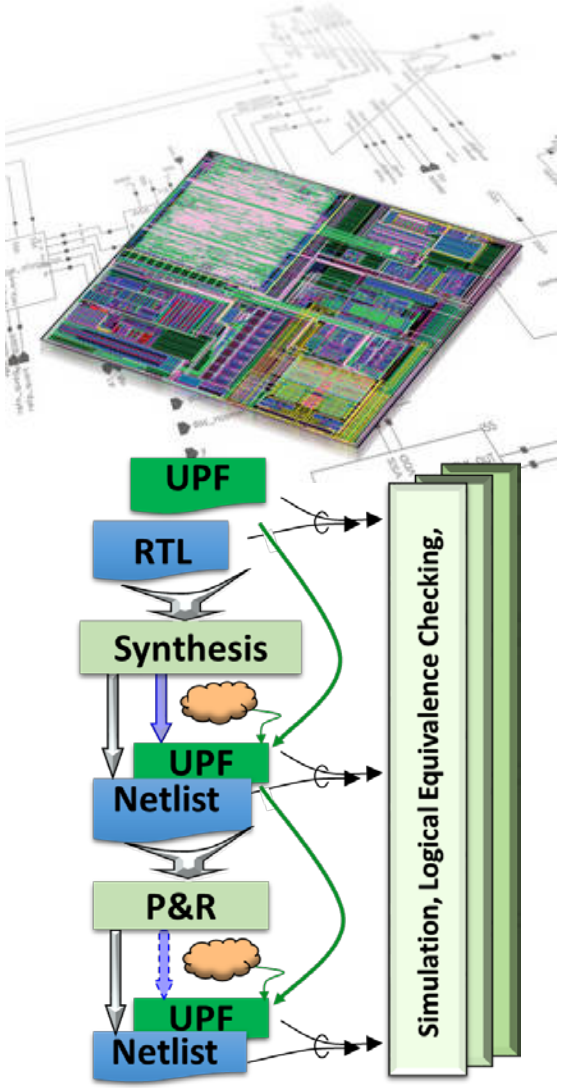
Agenda

- Introduction
- Evolution of UPF
- Challenges in Migration
- Backward Compatibility
- What's new in UPF 3.1
- Semantic difference b/w standards
- UPF design Guidelines
- Conclusion

accellera SYSTEMS INITIATIVE

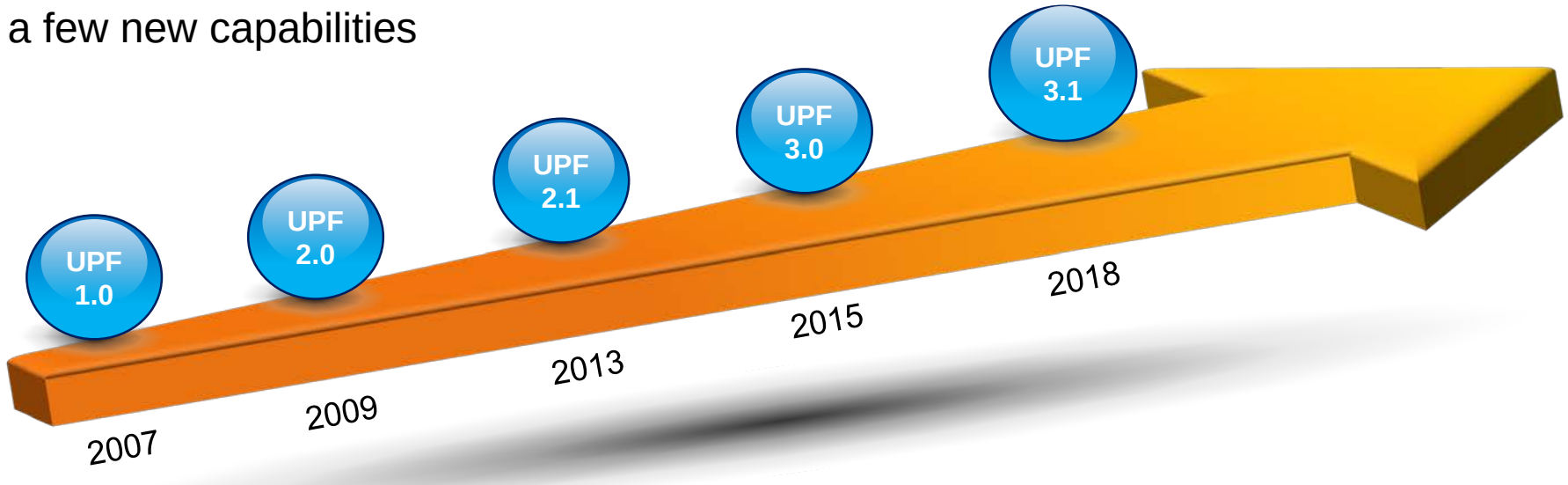
Introduction

- Power Management & Verification Complexity
  - Complex & energy aware chips
  - Maximize battery life
  - Requiring sophisticated power management
    - Power Gating, Multi Voltage, DVFS, Biasing
    - Affect design functionality
    - IPs using own power management posing integration challenges
  - Need for power verification
    - HDL not equipped, Power formats share burden
- Unified Power Format
  - Define power management
  - Based on Tcl
  - Provide HDL Interface
  - Information Model to capture processed data



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 defined by IEEE
  - Backward compatible with UPF 1.0
  - Supports IP development, refinement
- UPF 2.1
  - Clarifies and enhances UPF 2.0 features
  - Adds a few new capabilities
- UPF 3.0
  - Several new capabilities added
  - Updated existing concepts, viz. power states
- UPF 3.1 – latest standard
  - New commands for simulation control
  - Clarification of semantics



New Challenges

- Five UPF standards
  - Compatibility, Differences & Migration challenges
- Starting a new design: Which UPF version to choose for your design ?
- Building on top of your existing design
  - And some functionality requires a standard upgrade
- Building an SoC : Integration challenges
  - One of the IPs was built using a different UPF standard than another IP in the same design
- Verification tools support different standard as a tool-default
  - Designs are not easily interoperable
- UPF standards are not entirely backward compatible
- Which is the right standard for my design ?

Migration to UPF 3.1: Backward Compatibility

- UPF standards are not backward compatible
  - Old commands get deprecated
  - Syntax update for existing commands
  - Semantics also get changed
- Challenges in reuse of existing IPs
- Semantic compatibility
  - Need to understand the semantic difference b/w releases
  - May be required to edit the UPF files
- Syntax compatibility
  - May need to edit to make UPF 3.1 compatible

Migration to UPF 3.1: Common FAQs

- Does UPF command upf_version controls both the syntax and semantics of a UPF file?
  - The standard does not define how a verification tool uses the specified UPF version argument;
  - UPF command upf_version can be used multiple times for the UPF specification
  - Semantics are generally defined for the whole design
  - The UPF command upf_version controls the syntax of subsequent UPF commands
  - In general verification tool's follow a default particular standard for semantics with tool options to modify the same.
- Designs with multiple IPs: Is it possible to have multiple semantics (UPF standards) for a same design?
  - Different UPF files in the same design may use different upf_version command to specify the different standard.
  - It is **not** possible to follow multiple semantics for a same design. UPF does not have any concept of namespace to limit the scope of standard version to be followed
- If starting with a new design, which UPF standard to follow?
  - Unless absolutely necessary always follow the latest standard and UPF semantics.

2020 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

What's new in UPF 3.1

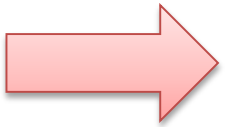
- Address the challenges in low-power verification
- New features introduced
- Semantics updates
  - Some semantics have been changed over UPF standard releases

accellera SYSTEMS INITIATIVE

Challenge: *Controlling SV Assertions in Low-Power Designs*

```
always @(clk)
begin
  assert (q == 1'b1)
     else error(“ERROR”)”;
end
```

Domain OFF



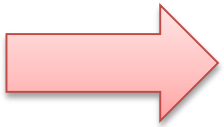
```
always @(clk)
begin      ‘x’
  assert (q == 1'b1)
     else error(“ERROR”)”;
end
```

- Assertion based verification is one of most common verification methodology
- During power-off, its possible that the signals used in assertion control check got corrupted
  - FALSE assertion failure messages
- No standard definition of how SV assertions controlled in low-power simulations
  - Relying on assertion control RTL system task $assertcontrol
  - Tool’s capabilities & mechanism

Solution: *Controlling SV Assertions in Low-Power Designs*

```
always @(clk)
begin
  assert (q == 1'b1)
     else error(“ERROR”)”;
end
```

Domain OFF



```
always @(clk)
begin
  assert (q == 1'b1)
     else error(“ERROR”)”;
end
```

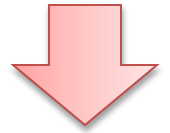
- Introduced new command “sim_assertion_control” to control the behavior of assertions during low-power verification
- Allows users to specify when the assertions will remain active or get inactive/killed/reset based on the control criteria
- Eg. Disable assertions during power down period

```
sim_assertion_control
[-elements element_list]
[-exclude_elements
exclude_list]
[–domain domain_name]
[-model model_name]
[-controlling_domain
domain | -control_expr
boolean_expression]
[-type <reset | suspend |
kill>]
[-transitive [<TRUE |
FALSE>]]
```

Challenge: *Controlling replaying of initial blocks*

- Initial blocks in Verilog are used to provide initial values to logic in the module
  - Reinitialize logic in behavioral modules of ROMs or PLLs after all power up events
  - During the power off – on sequence, certain logic may remain corrupted
- Standard lacked any semantic definitions for controlling the initial blocks
  - Rely on simulation tool capabilities to specify the initial blocks to be replayed at power up
  - Non interoperable

```
initial begin
  //initialize mem
  mem[size:0] = 1;
end
```



Power Down

```
mem[size:0] = 'x'
```

Power Up

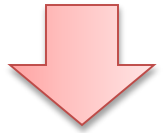
```
//Remains 'x'
mem[size:0] still 'x'
```

Solution: *Controlling replaying of initial blocks*

- New command “sim_reply_control” to specify initial blocks to be replayed when a domain powers up
- Each initial block targeted by this command will be replayed when the primary supply set of the domain/controlling domain transitions into the NORMAL state

```
sim_replay_control
[-elements element_list]
[-exclude_elements exclude_list]
[-model model_name]
[–domain domain_name]
[-controlling_domain domain]
[-transitive [<TRUE | FALSE>]]
```

```
initial begin
  //initialize mem
  mem[size:0] = 1;
end
```



Power Down

```
mem[size:0] = ‘x’
```

Power Up

```
//Retrigger initial
block
mem[size:0] is now 1
```

Challenge: Precedence Rules

- Common UPF build up methodology
  - Define a power domain/strategy/attributes for a more generic set of elements
  - Further write subsequent commands defining power intent for more refined set of elements
  - Reduces number of lines of UPF code
- No clear precedence rules
- Conflicting scenario's

```
set_isolation iso1 –domain pd –source pd1 –sink pd2
set_isolation iso2 –domain pd –source pd1
```

  - Both the above UPF command applies to ports which have source as PD1

Solution: Precedence Rules

- Precedence rules defined
  - Precedence rules for power domains
  - Precedence resolution for retention strategies
  - Precedence resolution for isolation, level_shifter, and repeater strategies,
  - Precedence resolution for name affixes
  - Precedence resolution for supply connections
  - Precedence resolution for attributes
  - Precedence resolution for simstates

```
set_isolation iso1 –domain pd –source pd1 –sink pd2
set_isolation iso2 –domain pd –source pd1
```

- Command that has both the options –source and –sink has higher precedence as compared to the command which has one of the options –source or –sink specified

What's new in UPF 3.1: Many more

- Challenge: *Controlling corruption semantics of specific design elements*
  - *Solution:* New command "sim_corruption_control"
- Challenge: Macro & Terminal boundaries
  - What's a hard boundary, no analysis beyond this
  - Solution: Clearly defined the terminal boundary beyond limiting the access beyond this boundary
- Challenge: *Verifying supply constraints*
  - *UPF 2.1 introduced available_supplies, but doesn't satisfy all criteria's*
  - *Solution: UPF 3.1 introduced boundary_supplies, and clarified available_supplies*

And Many More..

SEMANTIC DIFFERENCES BETWEEN UPF STANDARD'S

- With every UPF release, a number of semantics or concepts gets clarified.
  - Some may not be backward compatible
  - As general practice it is important to follow the latest semantics
  - Migration of existing deigns is challenge
- Verification tools follow a common semantics for low-power verification of the whole design
- Important to know all the semantic differences
  - Specifically ones which are backward incompatible

Power State's

- Used for both verification and implementation of power management
  - Define the simulation aspect of various state the system is in to
  - Checking the power management structures
- UPF 2.0
  - *Power State Tables (PST) :* Tabular representation of the possible state combinations for the given supplies
    - **Limitations :** Dependency on supply port/nets ; Tabular representation causes an explosion of states ; Lack of hierarchical composition capability in PSTs
  - *Power States :* allows users to define power states on supply sets and power domains using *add_power_state* command
    - Express power states in terms of boolean expressions via –logic_expr and –supply_expr switches
    - Very powerful in capturing more complex relationships including hierarchical dependencies
    - **Limitations** : No proper semantics about how the states are handled when supply sets are associated with other supply sets or handles ; does not restrict the transfer of power states in the supply set associations

Power State's

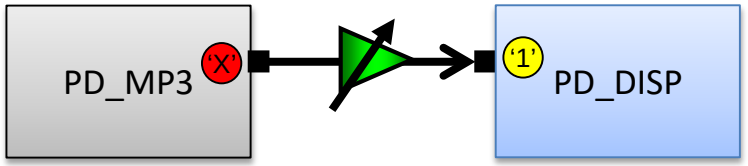
- UPF 2.1
  - Marked PST as legacy
  - New Clarifications
    - Supply set handles are local supply sets. Power states do not get transferred
    - Added –supply, -domain and –complete to add_power_state
    - Addition of another simstate "CORRUPT_STATE_ON_ACTIVITY"
  - Restrictions
    - Number of restrictions to avoid state explosion & unlimited possibilities of defining power state expressions ; facilitate better methodology for power intent modeling
  - Limitations
    - Failed to provide clear definitions about the relationship between power states of various IPs and how the system behaves a whole.
    - Lacked to provide any information about system level power modeling
    - Further the simulation behavior based on power states definitions relied on loose concept of multiple power states being active at same time and most corrupt simstate being the active simstate driving the simulation.

Power State's : UPF 3.0/3.1

- New command "create_power_state_group" to create a group of related power states
  - Define the power state of a whole system which relies on the power state definitions of its IPs
- Overhauled the definitions & restrictions of power states
  - Predefined states DEFAULT_NORMAL and DEFAULT_CORRUPT for supply sets removed
    - Two new predefined states ON and OFF for supply sets : Backward incompatible
    - Users can update these power state by defining logic expression for these deferred power states.
  - Two new predefined power states for all objects, UNDEFINED and ERROR state.
    - ERROR state: Represents error condition in which two mutually exclusive states are both active at the same time.
    - Undefined power state: This power state initially represents the undifferentiated set of all possible functional states of that object.
  - Definitive power state: If its defining / logic expression consists of a single term or conjunction of terms
  - The new standard allowed building the power states upon existing ones
    - One being the refined power state of an abstract power states. A power state S of an object is a *fundamental power state* if it is a power state that is not a refinement of any other power state of that object.
  - Multiple power states can be active at the same time, however LRM defined clearly that there will be a single current power state of an object.
    - The simstate of current power state is the one which drives simulation behavior of that object.

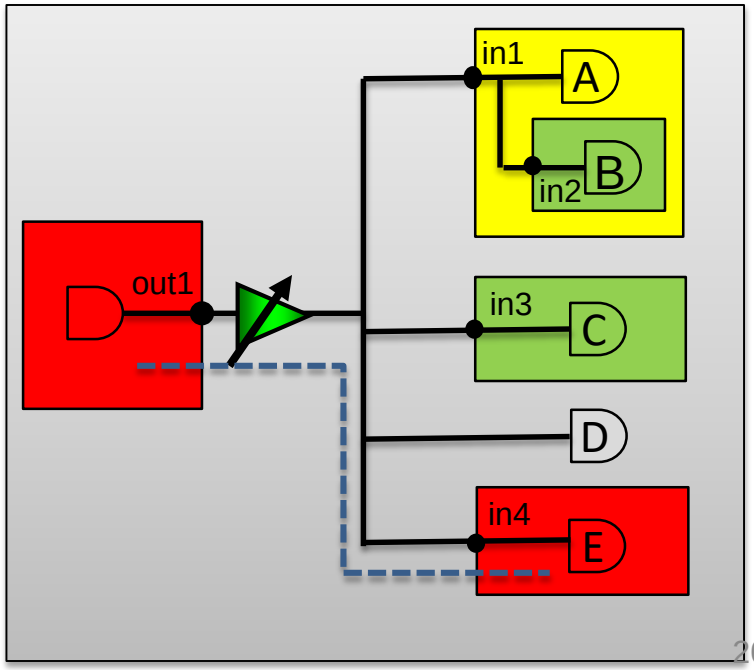
Port Vs Path Based Semantics

- Source logic goes OFF and sink logic in an ON state, there is a need for Isolation




```
set_isolation strategy_name
[-source <source_domain_name | source_supply_ref >] [-sink <sink_domain_name | sink_supply_ref >]
[-location <self | other | parent | fanout>] [-clamp_value <0 | 1 | Z | latch | value | {<0 | 1 | Z | latch | value>*}>]
…
```

- If the isolation is not inserted properly at the right path, then it can lead to functional failure
- If the isolation is placed at a location where it is not required, it is a redundant cell and leads to waste of area and power.



UPF 3.0

```
set_isolation iso3 -domain red –location parent
```

- UPF commands do specify the path where the isolation needs to be applied and the domain at which the cell needs to be inserted
- Port Based Semantics : can result in collateral damage

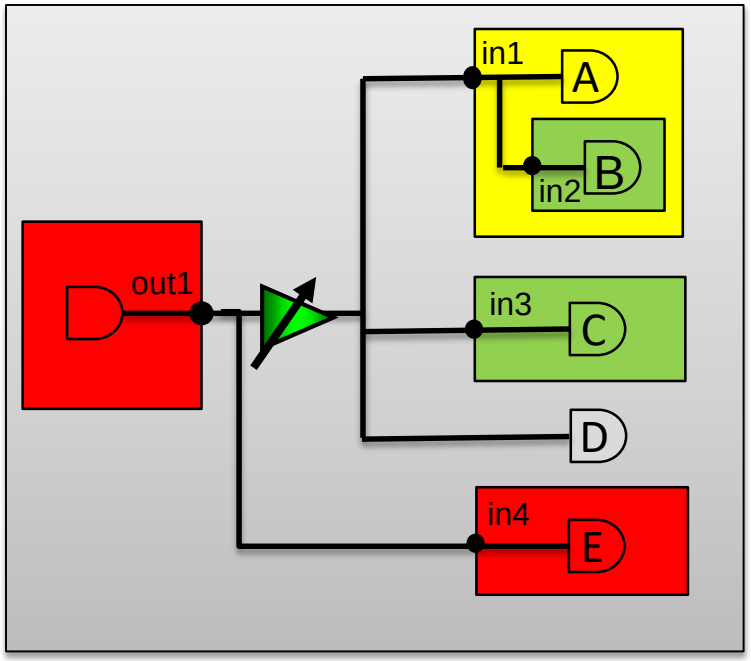
UPF 3.1 Solution: Port Vs Path Based Semantics

- Isolation strategies are applied on a **per path basis** in the design
- **Net splitting** to avoid to minimize collateral damage
- -**location fanout :** target insertion port is port on location domain boundary closest to the receiving logic.
- -**location fanout** is **not** specified: target insertion port is the port of the location domain that is (for the self domain), or corresponds to (for the parent or child domain), the port to which the strategy applies
- Error if the isolation power intent cannot be implemented without duplicating ports

UPF 3.1

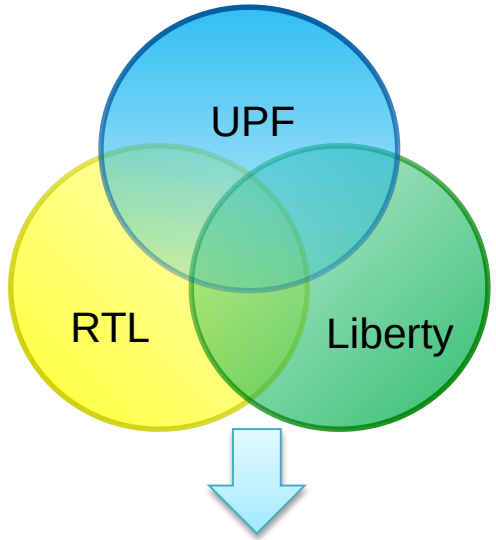
```
set_isolation iso3 -domain red
```

- Iso3 –location self: Error
- **Iso3 –location parent: At highconn of out1 but will not isolate path that goes to in4.**
- Iso3 –location fanout: At in1 before A (will not isolate path goes to in2), at in2 in green, at in3 in green, at highconn of out1 but will only isolate path that goes to D.



UPF Attributes

- Attributes provide information that supports or affects the meaning of related UPF commands
- Multiple sources
- UPF 2.1
  - Information about what are the set of attributes that can be applied on model/instances/ports
- UPF 3.0
  - Clarified the precedence of attributes
  - Concept of characteristics attributes : represent characteristics of a module or cell
    - Verification tools error out when constraints are not met
- UPF 3.1
  - Removed any ambiguities in the definitions and restrictions of attributes
  - Characteristics attributes were replaced with more clear definitions
    - Model specific, some only instance specific and which can be applied on both model and instance level
    - More clear constraint definitions

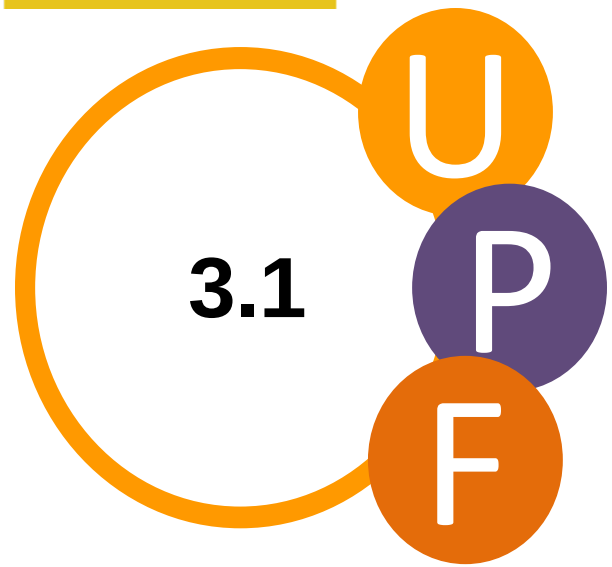


PRIORITY ??

UPF Design Guidelines

- ✓ Use UPF command “upf_version” to specify the syntax of commands
  - Specify this command per UPF file
    - If the file has a different way of syntax from its parent UPF file from which it got loaded
- ✓ Check the verification’s tool default support of UPF standard with respect to semantics
  - Use tool’s options or attributes to control a specific functionality
- ✓ Unless absolutely necessary always follow the latest semantics
- ✓ Starting-off with a new design
  - Follow the same UPF version for all the design & UPF file
  - Semantics cannot be controlled for a particular part of the design however it is applicable for the whole design.

Conclusion



- Power Aware Verification has become **complex**
  - Lots of **challenges** to verify power management
- UPF is fastest evolving IEEE standard
- UPF 3.1 is a major milestone in evolution of UPF
  - **New additions** to fill the gap
  - **Clarification** of many concepts

- **"Which is the right standard for my design".**
  - There is **no single answer** for the question
- With clear understanding of the syntax and semantic changes over the UPF releases
  - Better judgment of the UPF version for their respective design
  - Easy migration to latest UPF standard

2020 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

THANK YOU

Questions ??

accellera SYSTEMS INITIATIVE