2025 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION INDIA

accellera SYSTEMS INITIATIVE™

Introduction of IEEE 1801-2024 (UPF4.0) Improvements for the Specification and Verification of Low-Power

Lakshmanan Balasubramanian[1], Raguvaran E[2], Shubham Saha[3], Santhosh Potlapalli[4], Manash Ranjan Raiguru[5], Jeevan Medaramitta[6], John Decker[1,4], Amit Srivastava[1,5], Daniel Cross[4]

IEEE SA STANDARDS ASSOCIATION [1] IEEE [2] intel [3] Texas Instruments [4] cadence [5] SYNOPSYS® [6] SIEMENS

Lakshmanan Balasubramanian

Senior Member IEEE, MIET, CEng (EC UK), Prof. MACM, Ind. MACCS, MVSI, Ex-SMTS & Technologist, TI India

Recently (-2025): Technologist (AMS & Low Power) & Senior Member Technical Staff (SMTS), responsible for AMS SoC design integration & verification

BSc Phy., CAHC, Univ. of Madras; BTech Elect. Eng., MIT, Anna Univ.; MTech EDT, IISc; Predoc., IMEC/KUL

25+ year industry experience

- Expertise in analog & embedded PM design, custom low power (LP) digital design, LP & AMS design & verification methodologies, and analog DFT/Test technologies
- Published more than 130 conference and journal papers with 10+ best/honourable mention paper/presentation awards, 3 nominations
- 11 patents, 3 TI manufacturing incentive awards (MIA)

Senior Member, IEEE; Member ACM & ACCS; CEng EC (UK); Member VSI

Member, IPRA, & PRI, The IET

- Honorary Treasurer, IEE/IET Chennai Local Network (2008-2010) Oversaw the membership and activity explosion with YP NW introduction
- Member & IET Knowledge Network representative in South Asia Regional Board (SARB), IET (2010-2012)

Actively contributing to IEEE, Accellera and Si2 standards working groups on AMS extensions of HDLs, AMS test aspects, and low power aspects

- Secretary of IEEE 1801 Standards Working Group
- 1800 (SV-AMS), 1801 (UPF), 1687.2 (Analog Test Access), 2427 (Analog Coverage), 2416 (UPM),
- Si2 CDC, Si2 SiG on ML
- Accellera SV-AMS, SC-AMS, UVM-AMS

Expert and go-to person for AMS and LP design, verification, and silicon debug aspects; consulting across businesses within TI and industry

As an Industrial Liaison for TI in SRC, actively collaborating and driving industry sponsored academic research worldwide

- Recipient of Mahboob Khan Outstanding Liaison Award 2016 and 2023
- Active collaboration with CEG/AU, Arizona SU, Iowa SU, BITSP Hyderabad, IIT Kharagpur, IIT Roorkee, Oxford Univ., UCB,

For complete profile & list of external publications follow https://www.linkedin.com/in/lakshmanan-balasubramanian-1b60422/

Raguvaran Easwaran

Silicon Domain Architect, Intel Technology India Pvt. Ltd.

Raguvaran E is a Senior Member of IEEE and currently serves as a Silicon Domain Architect at Intel Technology India Pvt Ltd.
With over 13 years of experience in the pre-silicon verification domain, he has made substantial contributions to the semiconductor industry.
Raguvaran has authored 10 publications in international journals and conferences, showcasing his technical depth and thought leadership.
He has been an active member of the IEEE 1801 Standard Working Group for more than six years, focusing on power intent standards.
His professional journey includes impactful roles at Qualcomm and Intel, where he has consistently driven innovation in silicon architecture and verification methodologies.
He holds a BE in ECE (2009) from Sri Krishna College of Engineering, Anna University and an M.Tech in VLSI Design from PSG College of Technology, Coimbatore.

https://www.linkedin.com/in/raguvaran-e-2b452247

Shubham Saha

Digital Design Engineer, Texas Instruments (India) Pvt. Ltd.

Shubham Saha is a Digital Design Engineer working in the Radar group in Texas Instruments (India).
He started this role as a NCG in 2020, and since then has been comprehensively working on low power techniques for SoCs, digital signal processing chains, IP development, safety use-cases (TUV certified Functional Safety Professional) for Automotive SoCs and is currently leading the RTL design for next generation Radar SoCs.
He is also pursuing his MTech. in AI from IISc alongside work.
He holds a Btech in Electronics and Communication Engineering from NIT Trichy (2020).
He has been a part of the IEEE-1801 standards work group since October 2023 representing TI.

https://www.linkedin.com/in/shubhamsaha

Santhosh Potlapalli

Senior Principal Product Validation Engineer, Cadence Design Systems

Potlapalli Santhosh, a **Senior Principal Product Validation Engineer** at **Cadence Design Systems**.
With over **14 years of deep technical expertise in Mixed-Signal Verification**, Santhosh has been instrumental in validating complex semiconductor solutions that power today's most advanced technologies.
His work reflects a strong commitment to quality, innovation, and engineering excellence in the EDA industry.
Today, he brings his insights and experience to share valuable perspectives on cutting-edge validation methodologies and industry trends.
He holds a BE in Electrical, Electronics and Communication Engineering from JNTU (2010).

https://www.linkedin.com/in/santhosh-potlapalli-82276918

Manash Ranjan Raiguru

Product Engineer, Synopsys Bangalore

Low power design verification expert with 13+ years of experience in EDA and design companies.
Currently working as Product Engineer in Synopsys Bangalore and a member of WG IEEE 1801 standard.
Earlier he was a Senior Staff Engineering Design with Infineon for a year, and prior to that a Lead Application Engineer with Cadence Design Systems for 9 years.
He holds a B.Tech in Electronics and Communication Engineering specializing in VLSI/Semiconductors from NIST, Berhampur (2012), and an M.Tech in Microelectronics from BITS Pilani (2022).

https://www.linkedin.com/in/manash-ranjan-raiguru

Jeevan Medaramitta

Product Engineer, Siemens EDA

Jeevan is a Product Engineer, DVT, Siemens EDA responsible for Functional Verification, Simulation, Low Power Design & Verification.
He holds a B.Tech., in Electronics and Communication Engineering from Karunya Institute of Technology and Sciences (2018).

https://www.linkedin.com/in/jeevan-medaramitta-25678a207

Introduction of IEEE 1801-2024 (UPF4.0) Improvements for the Specification and Verification of Low-Power

Lakshmanan Balasubramanian[1], Raguvaran E[2], Shubham Saha[3], Santhosh Potlapalli[4], Manash Ranjan Raiguru[5], Jeevan Medaramitta[6], John Decker[1,4], Amit Srivastava[1,5], Daniel Cross[4]

accellera SYSTEMS INITIATIVE™
IEEE SA STANDARDS ASSOCIATION 1 IEEE 2 intel 3 Texas Instruments 4 cadence 5 SYNOPSYS® 6 SIEMENS
2025 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION INDIA

Acknowledgements

- IEEE 1801 WG Office-Bearers
  - John Decker – IEEE 1801 WG Chair & Cadence Design Systems
  - Amit Srivastava – IEEE 1801 WG Vice-Chair & Synopsys
  - Lakshmanan Balasubramanian – IEEE 1801 WG Secretary & Sr. Member IEEE
- Other Contributors
  - Daniel Cross – Cadence Design Systems
  - Marcelo Glusman – Cadence Design Systems
  - Paul Bailey – Nordic Semiconductor
  - Rick Koster – Siemens EDA
  - Progyna Khondkar – Cadence Design Systems
- Special thanks to the 1801 WG
  - Currently over 40 members representing 16 companies
  - Former members John Biggs (previous Chair), Phil Giangarra, David Cheng
- Thanks to IEEE Standards Association and Accellera
  - This presentation solely represents the views of the author(s), and does not necessarily represent a position of either the IEEE P1801 Working Group, the IEEE Design Automation Standards Committee, IEEE or the IEEE Standards Association
  - Design Automation Standards Committee of the IEEE Computer Society, “IEEE Standard for Design and Verification of Low-Power, Energy-Aware Electronic Systems”, IEEE Std. 1801™-2024

Agenda

- Introduction
- Interconnect between UPF supplies and arbitrary HDL types
- Improvements in successive refinement and refinable macros
- Overview of Retention Changes
- Virtual Supply and Virtual Equivalence
- General Updates
- Beyond 4.0
- Q/A

IEEE 1801, Unified Power Format (UPF): An Introduction

Raguvaran E

Intel Technology India Pvt. Ltd.

Vijayakumar Sankaran, et al, Modern Recipes for Brewing the Inevitable Methodology for Today's ICs: Low-Power Mixed-Signal Design Verification, Tutorial, DAC 2019.

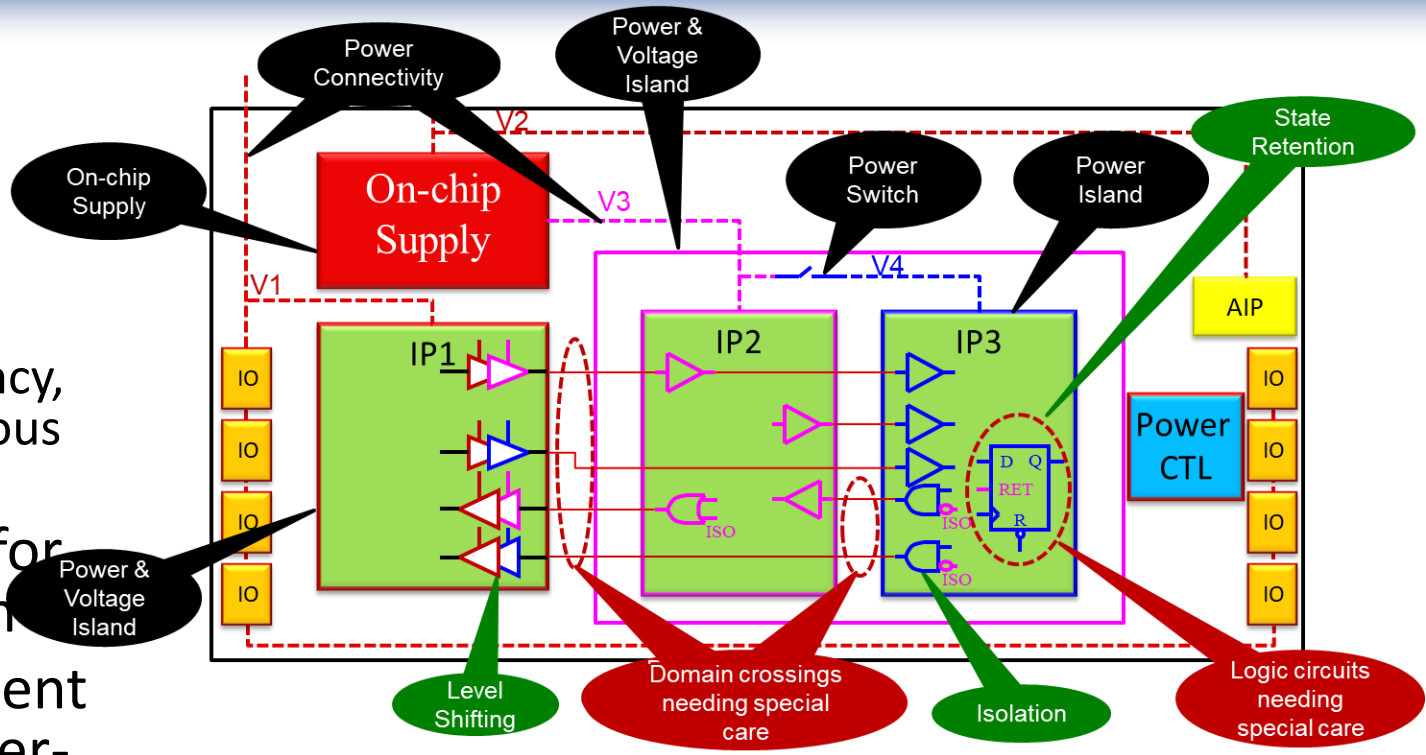
accellera SYSTEMS INITIATIVE™

intel

2025 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION INDIA

Background | Typical Low-Power System

- Electronic systems evolve
  - Increasing demands for energy efficiency, esp. mobile, embedded and autonomous applications
- Low-power (LP) design techniques for dynamic and static power reduction
- Functionally and electrically consistent LP design → User-controllable power-versus-performance trade-off
  - Choice of devices/components used
  - Adaptive and static body biasing
  - Clock frequency reduction for periods of low performance
  - Adaptive voltage and frequency scaling
    - On-chip supplies and oscillators
  - Power gating

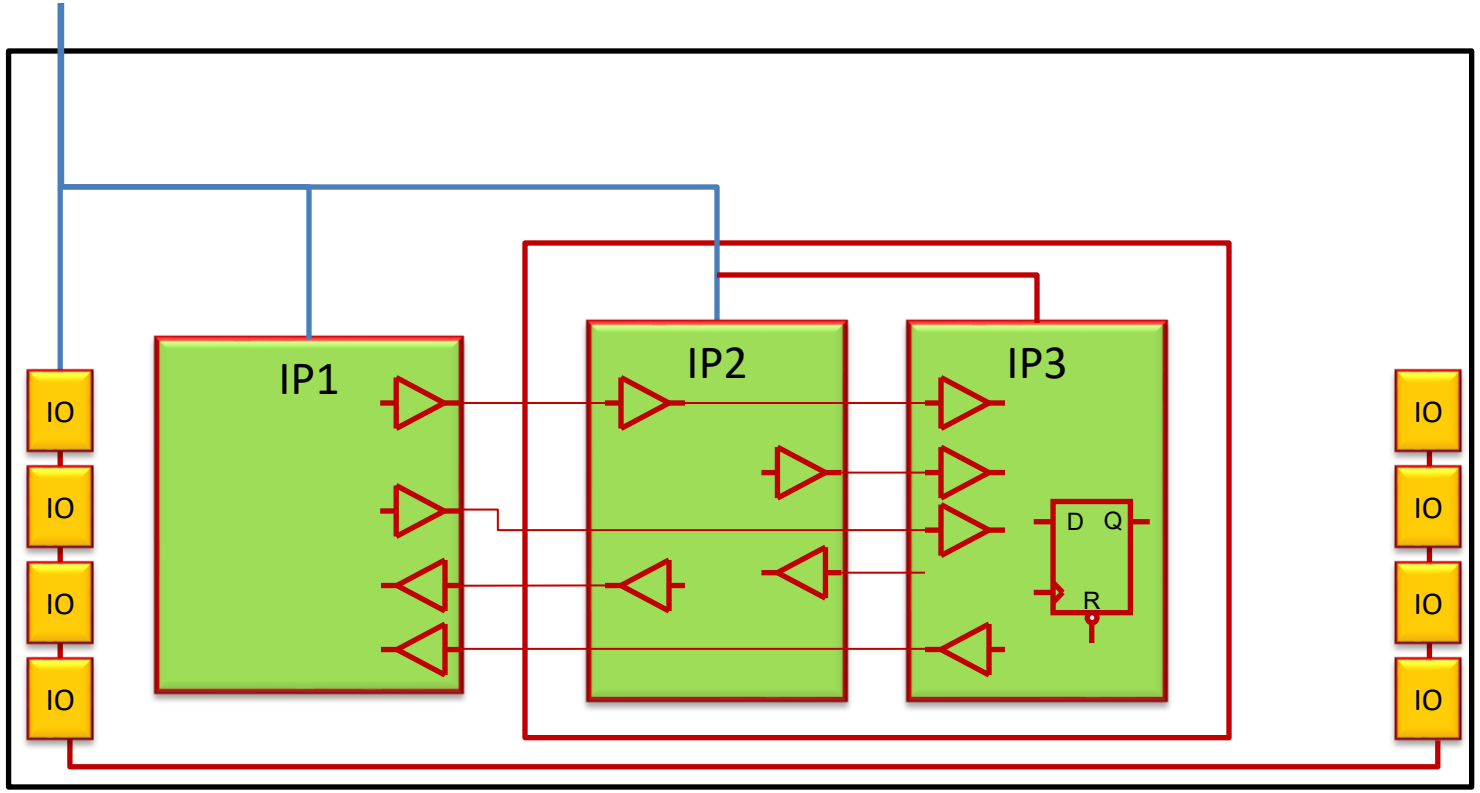


Techniques for avoiding functional and power consumption issues

- Consistent power and voltage domain partitioning
- Isolation of signals from switchable i.e., power gated domains to clearly define safe states
- Level shifting for signals crossing between domains of incompatible supply voltage levels
- State retention (SR) of memory elements

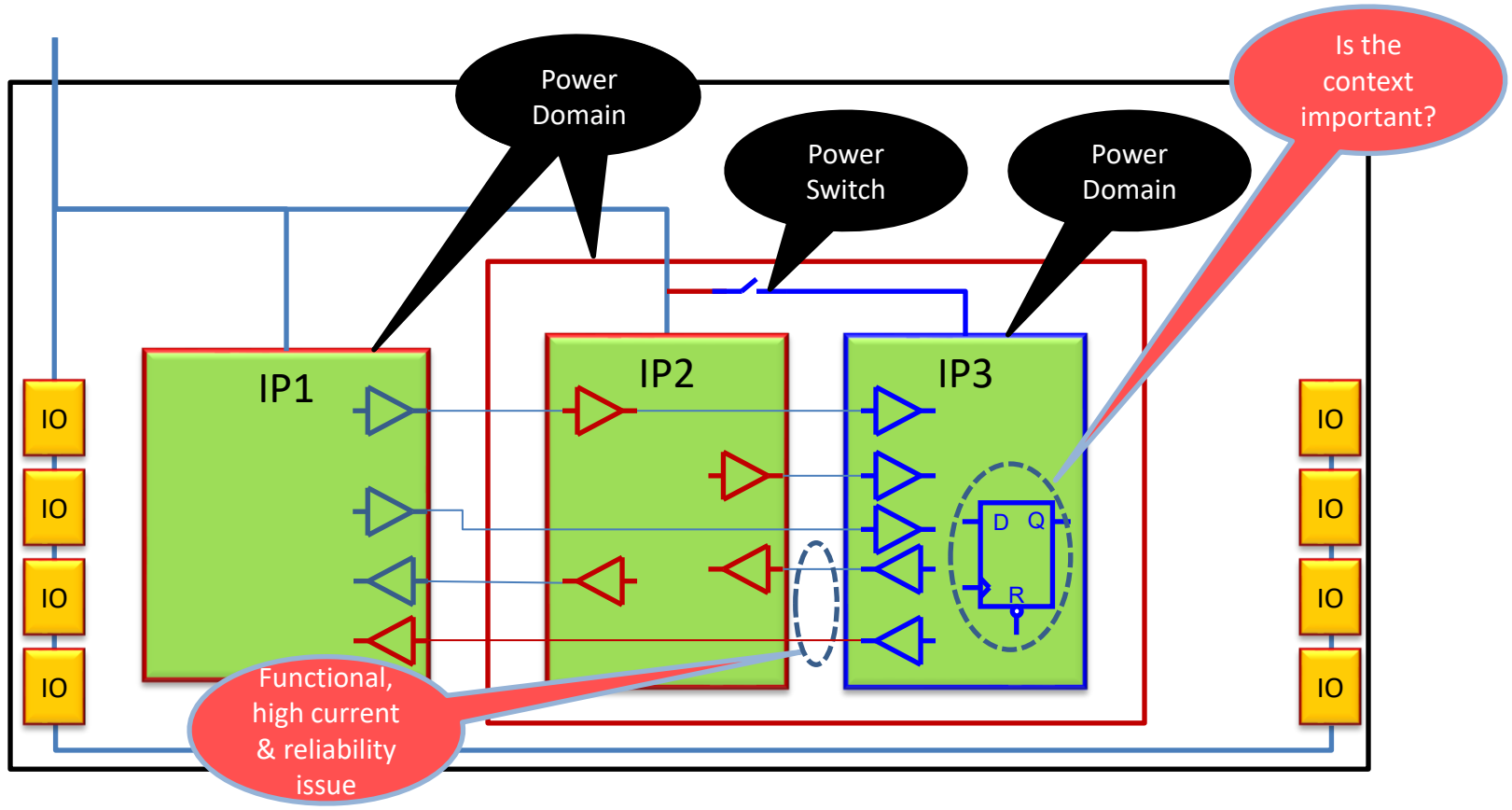
- Clock and reset domain crossing (CDC & RDC) techniques (Not a focus of 1801)
  - Clock and reset synchronization → Handled functionally

System Illustration: No Power Management

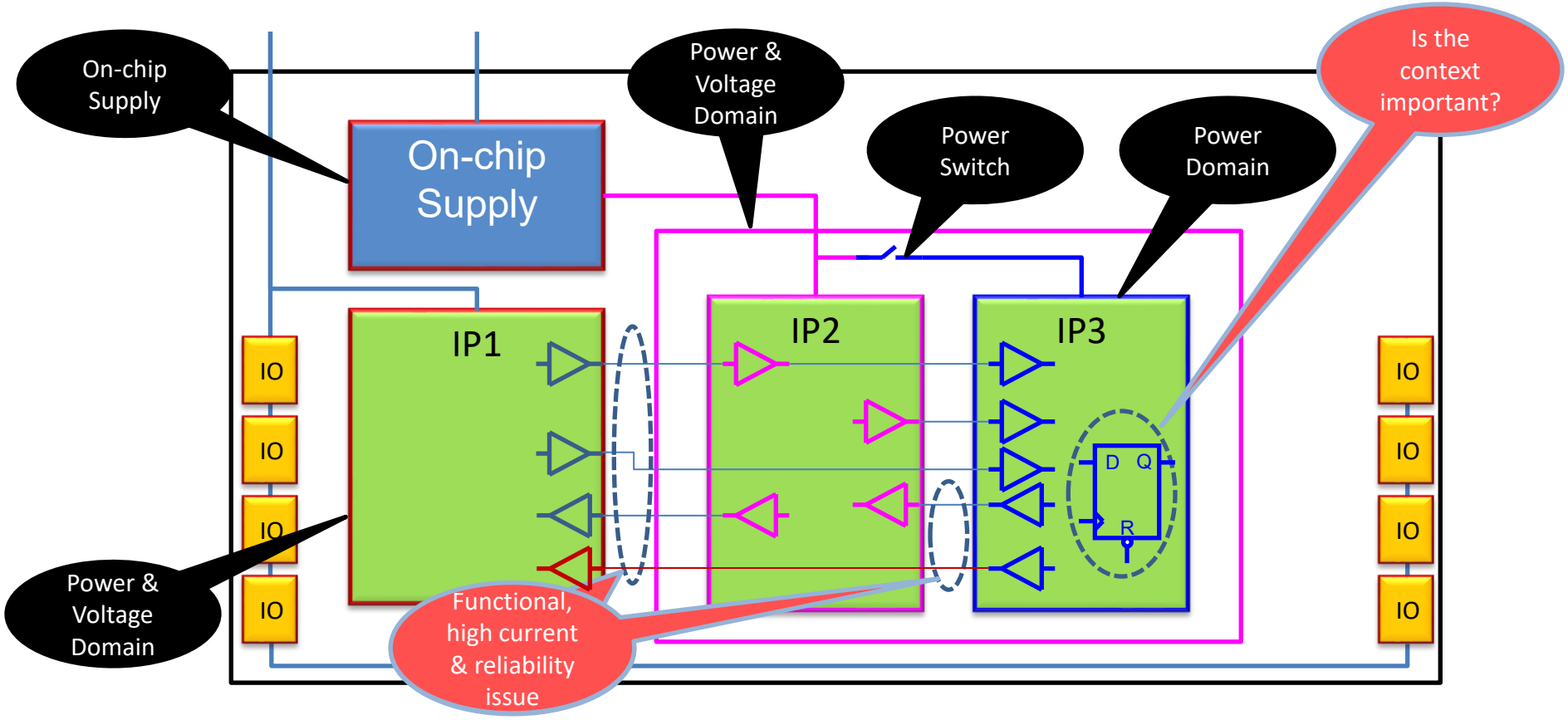


Vijayakumar Sankaran, et al, Modern Recipes for Brewing the Inevitable Methodology for Today's ICs: Low-Power Mixed-Signal Design Verification, Tutorial, DAC 2019.

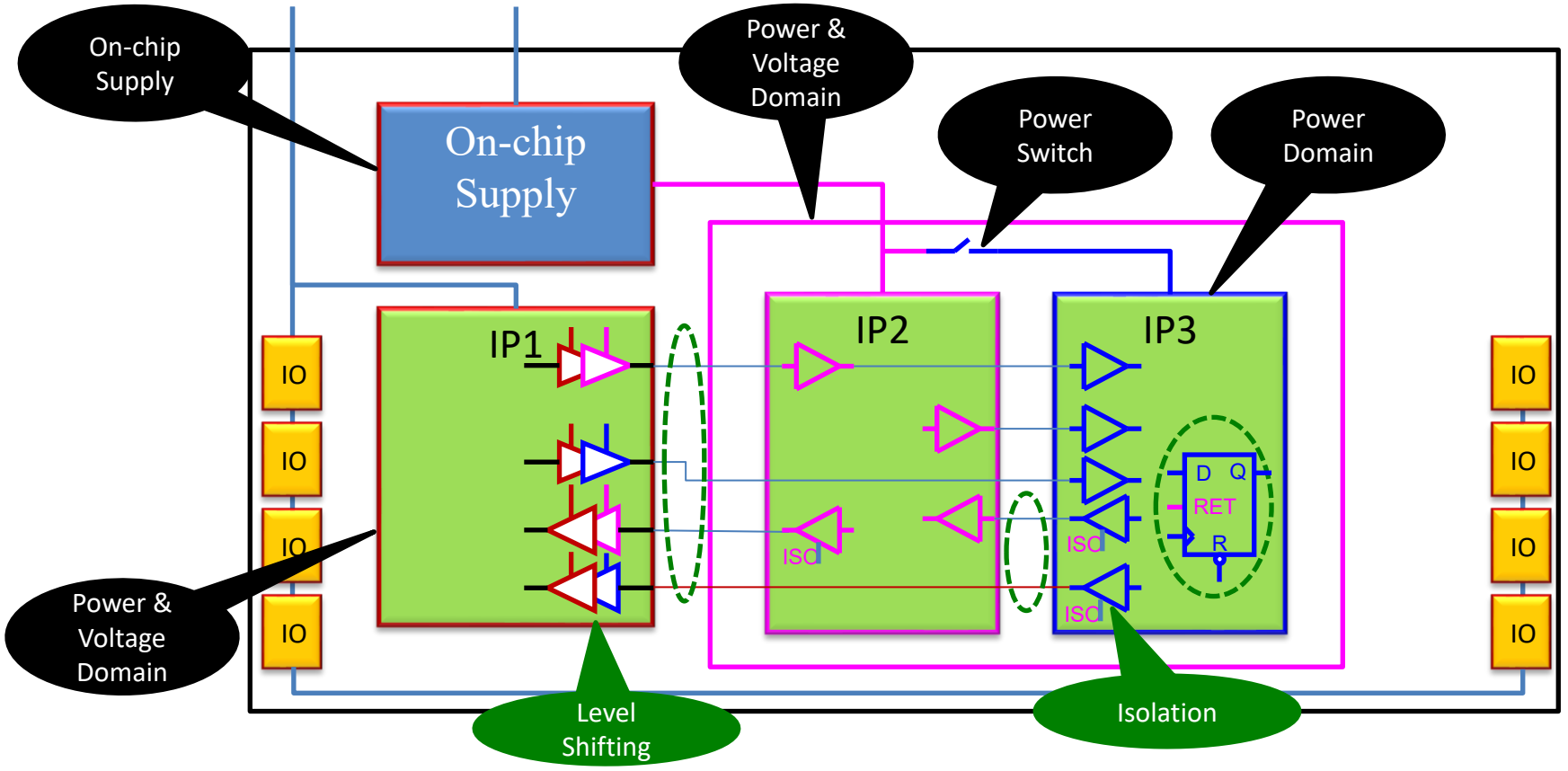
System Illustration: Power Domain



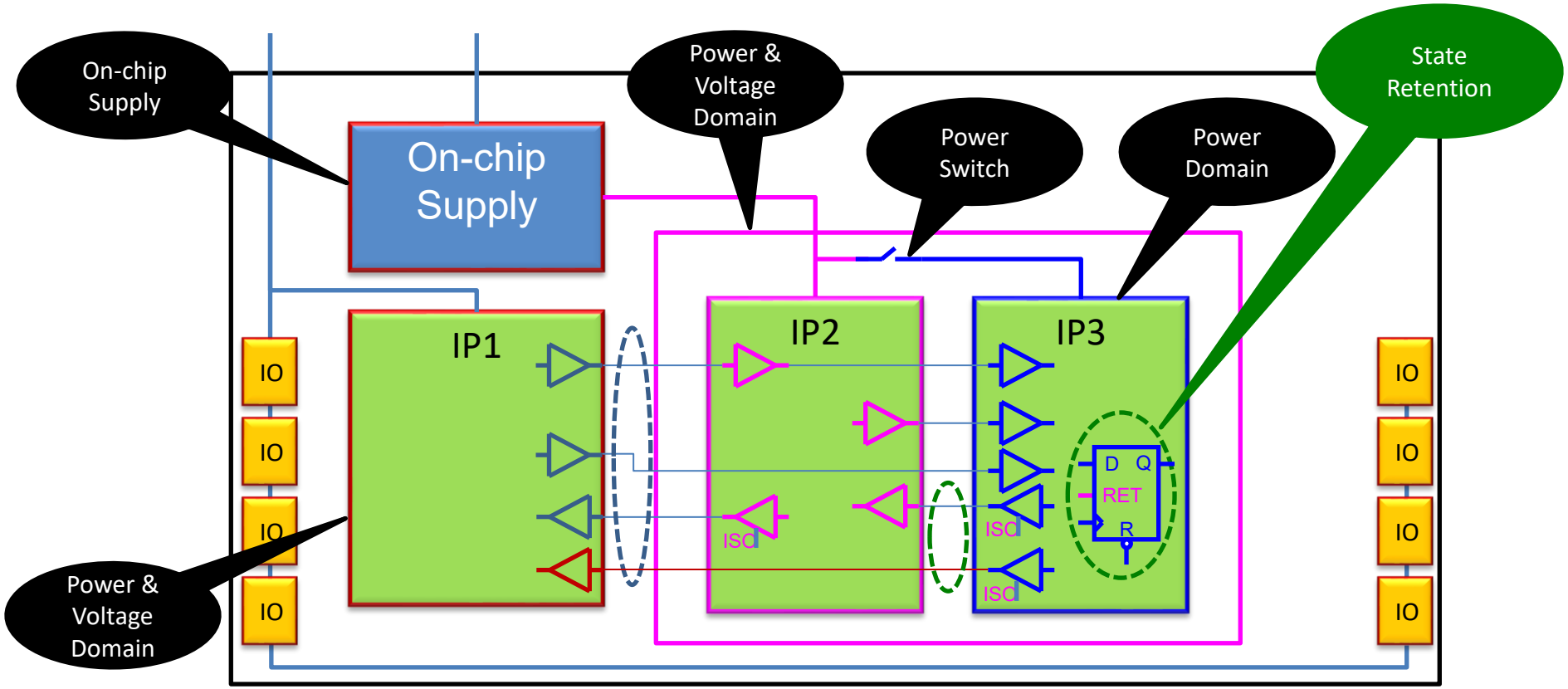
System Illustration: Voltage Domain



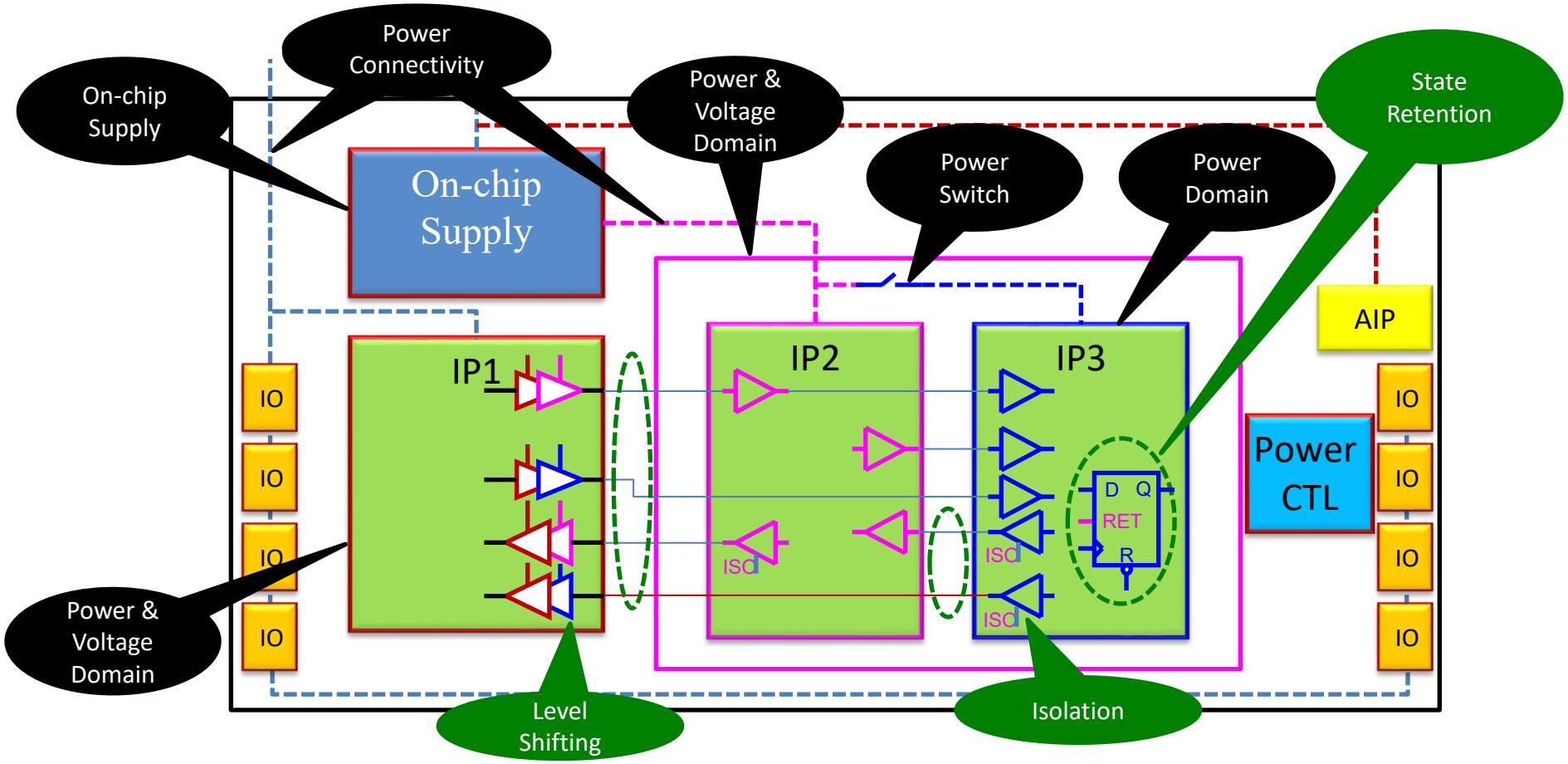
System Illustration: Isolation & Level Shifting



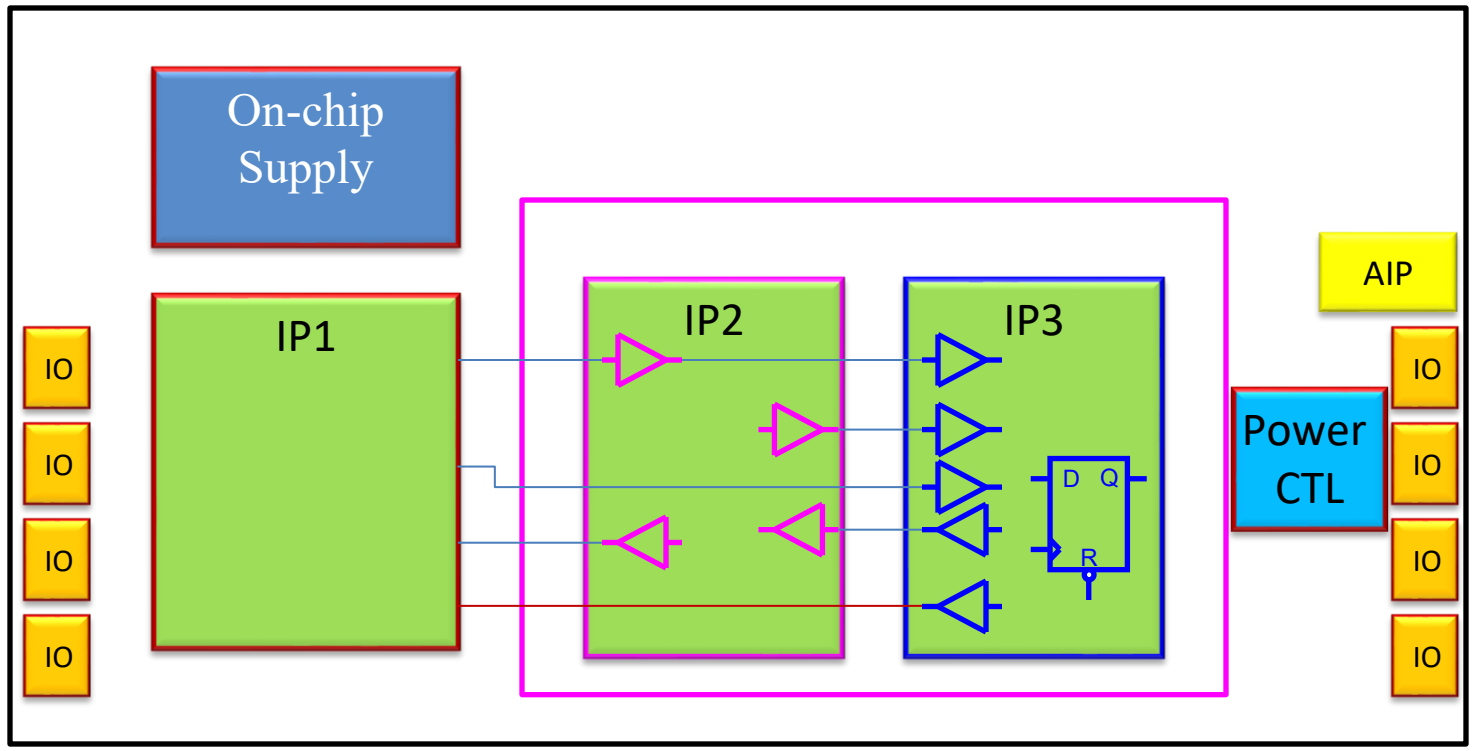
System Illustration: Retention



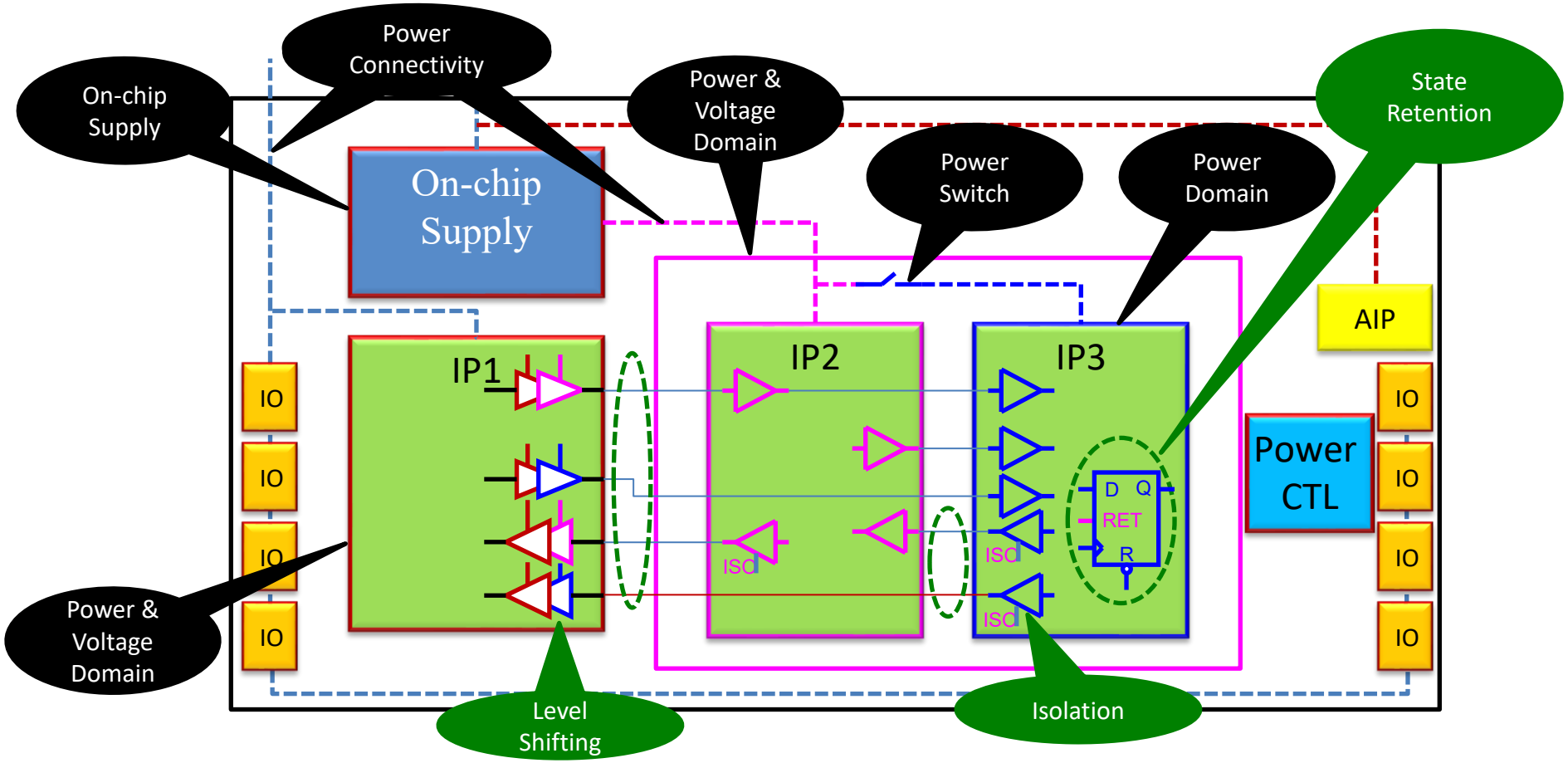
System Illustration: Power Controller



System Illustration: RTL View

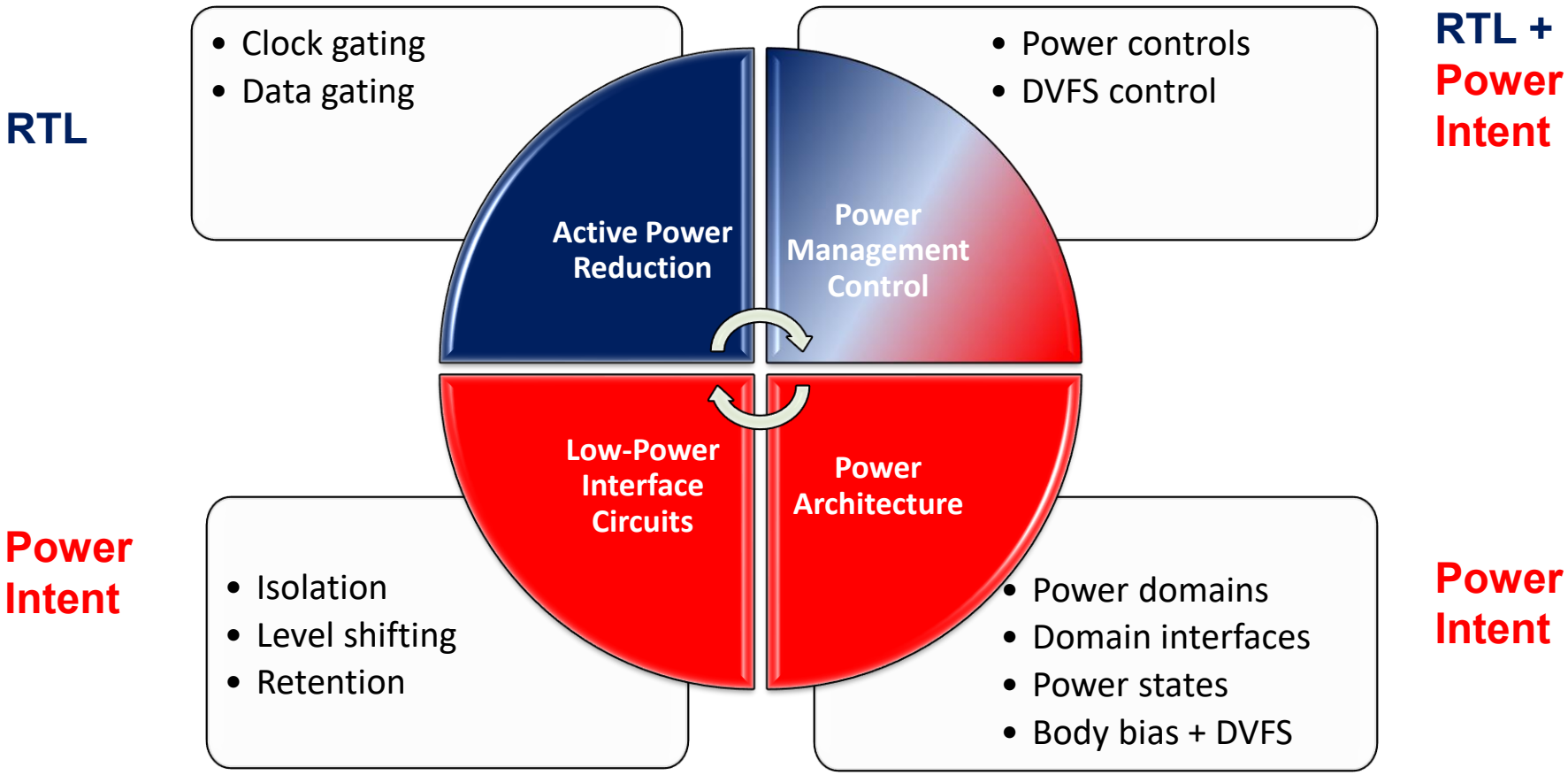


System Illustration: Power Intent Annotated



Vijayakumar Sankaran, et al, Modern Recipes for Brewing the Inevitable Methodology for Today's ICs: Low-Power Mixed-Signal Design Verification, Tutorial, DAC 2019.

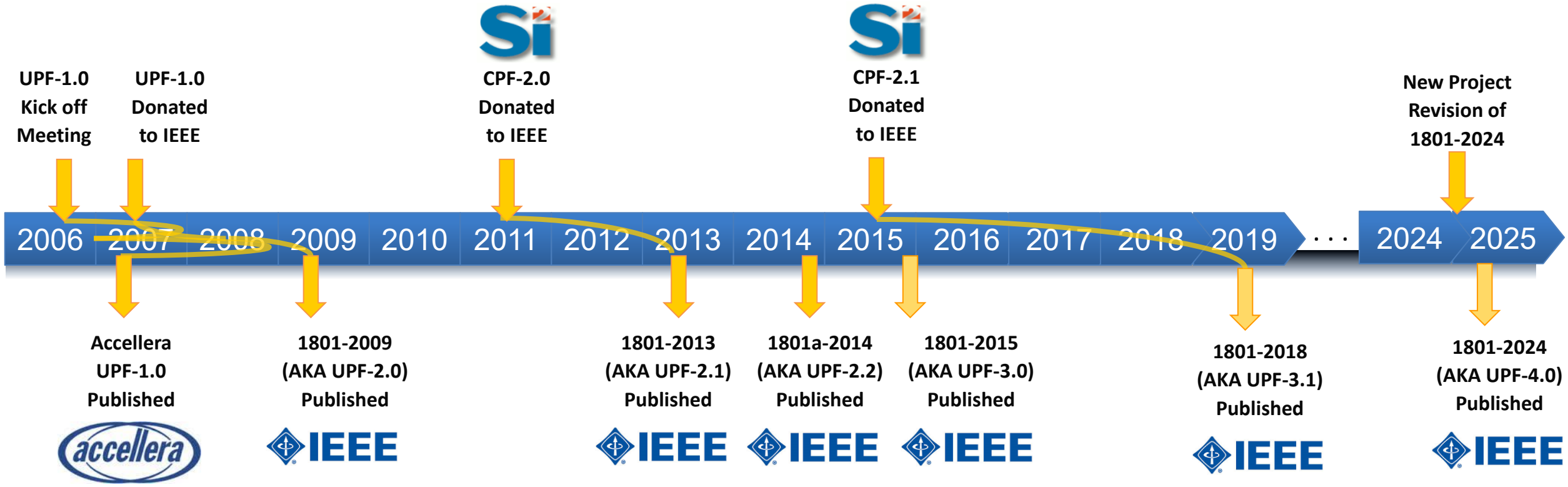
Low-Power Design Components



Unified Power Format (UPF)

- IEEE Standard for expressing Power Intent
  - To define power architecture and power management control
  - To minimize power consumption
  - Enables a consistent representation of power intent across all aspects of the design and verification flow
  - Enables early verification of power intent
- An Evolving Standard
  - 6 versions of IEEE -1801 over ~18 years
  - Donations from Accellera UPF1.0 and Si2 CPF 2.0 and 2.1
  - 1801 has had contributions from more than 20 chip design and EDA companies
- Based upon Tcl
  - Tcl syntax and semantics
- And HDLs
  - SystemVerilog, VHDL, SystemC
- For Verification
  - Simulation, Emulation, Static/Formal
- For Implementation
  - Synthesis, DFT, P&R, etc.
- And for System Level Power Modeling
  - Abstract power models with *power_expr*

Evolution of the Standard



UPF 4.0 – Major Goals

- Enable low-power simulation with mixed-signal features like real number modeling
- Enable accurate modeling of state retention
- Enhance IP design reuse with refinable macros
- Improve successive refinement flow
- Address over 200 Mantis items tracking improvement requests.
  - Enable Virtual Supplies
  - Updates/Clarifications to semantics
  - Ease-of-use features

Where to get the IEEE 1801-2024 Spec

- 1801-2024 specification is available from the IEEE GET program

- IEEE SA open repository (NEW)
  - Select examples and packages from the 1801-2024 specification
  - Planned community space to provide comments, advice, additional examples
  - Available at: https://opensource.ieee.org/upf

Summary of Change Topics

- New Concepts
  - Refinable Macro
  - Implementation UPF
  - Virtual nets/ports/sets/equivalence
  - Tunneling
  - Connections to real (VCM)
- Major Updates
  - Power distribution section 4.5.1
  - Simulation of state retention (9.7)
  - Annex I – VCM usage examples
  - Supply equivalence
- Clarifications (partial list)
  - Major improvements to Definitions
  - Resolved elements list
  - Literal supply
- Open SA repository
- Complete update of Annex E (example)
- Precedence
  - SPA, retention, composite types, Macros
- Naming Related
  - Rooted vs Simple name clarifications
  - Escaped naming styles
  - Generate block delimiter
  - Library name (5.3.3.2)
- Command Updates
  - **`map_retention_clamp_cell`**
  - **`set_isolation –async_set_reset –async_clamp_value`**
  - **`set_port_attributes –is_analog`** allowed on instance pins
  - **`set_port_attributes –feedthrough`** improvements
  - **`set_design_attributes`** with no object creates a “UPF“ wide attribute
  - **`find_objects –expand_to_bits`**
  - **`set_repeater –repeater_supply`** mandatory

Command/Option Change Summary

New Commands
create_vcm
create_upf_library
load_upf_library
use_upf_library
map_retention_clamp_cell
create_abstract_power_source

New Options
set_isolation *-async_set_reset* *-async_clamp_value*
connect_supply_net *-vcm* *-tunneling*
set_port_attribute *-is_refinable_macro* *-async_clamp_value*
load_upf *-implementation*
define_power_model *-update* *-implementation* *-complete*
create_supply_net *-virtual*
create_supply_port *-virtual*
create_supply_set *-virtual*
set_retention *-applies_to {latch ff both}* *-restore_period_condition* *-powerdown_period_condition* *-restore_event_condition* *-save_event_condition*
find_objects *-expand_to_bits*

Legacy/Deprecated
set_isolation *-applies_to_sink_clamp* *-applies_to_source_clamp*
create_supply_net *-reuse* *-domain*
create_power_switch *-domain*
create_supply_port *-domain*
set_port_attribute *-sink_off_clamp* *-source_off_clamp*
create_upf2hdl_vct
create_hdl2upf_vct
set_retention_elements *-transitive*
set_retention -save_condition -restore_condition -retention_condition

New Feature for UPF 4.0: Interconnect between UPF Supplies and Arbitrary HDL Types

Santhosh Potlapalli

Cadence Design Systems

Daniel Cross, "Applications for UPF HDL Supply Tunneling in Mixed Signal Design," DVCON 2025.

accellera SYSTEMS INITIATIVE™

cadence

2025 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION INDIA

Motivation

- An increasing number of designs are mixed-signal in nature and have significant analog and mixed-signal content
- Co-verification of analog and mixed-signal design elements with purely digital components has increased in importance
  - Analog and digital portions of designs increasingly share power supplies & with on-chip supplies
- Use of Real Number Modeling (RNM) to represent analog functions, synchronization of analog and UPF representations of the power supply network has become critical
- Fully formalized and seamless flows for complex power managed mixed-signal designs is needed with proper power intent specification

New Concepts

- **Value Conversion Method (VCM)**

  extends and enhances Value Conversion Table (VCT)

- **Tunneling**

  allows analog connections to be made via UPF

- **UPF Library**

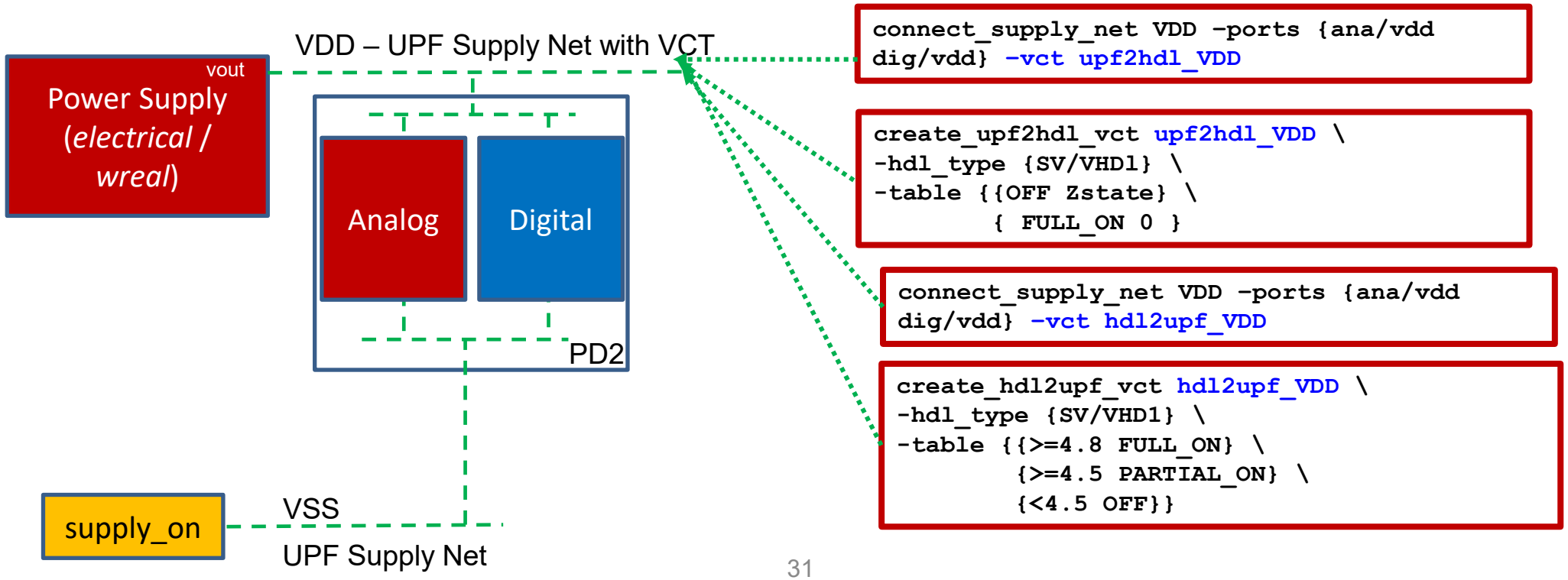
  helps avoid name collisions between otherwise global objects

- **Automatic VCM selection by nettype and data type**

  allows select right VCM based on connection

VCT – Earlier to VCM & Till UPF 3.1

- Value Conversion Table (VCT) defines voltage value mapping between UPF and an HDL model
- VCT commands are legacy: **create_hdl2upf_vct** and **create_upf2hdl_vct**
  - Their functionality is a subset of VCMs



VCM – Value Conversion Method

- A richer and more flexible mechanism to translate between UPF supply nets and HDL nets
  - Improved simple table conversions vs VCT
  - Enable advanced conversions of more complex types by using user defined Modules and Functions
  - Ability to contain a list of other VCMs to enable automated type conversions

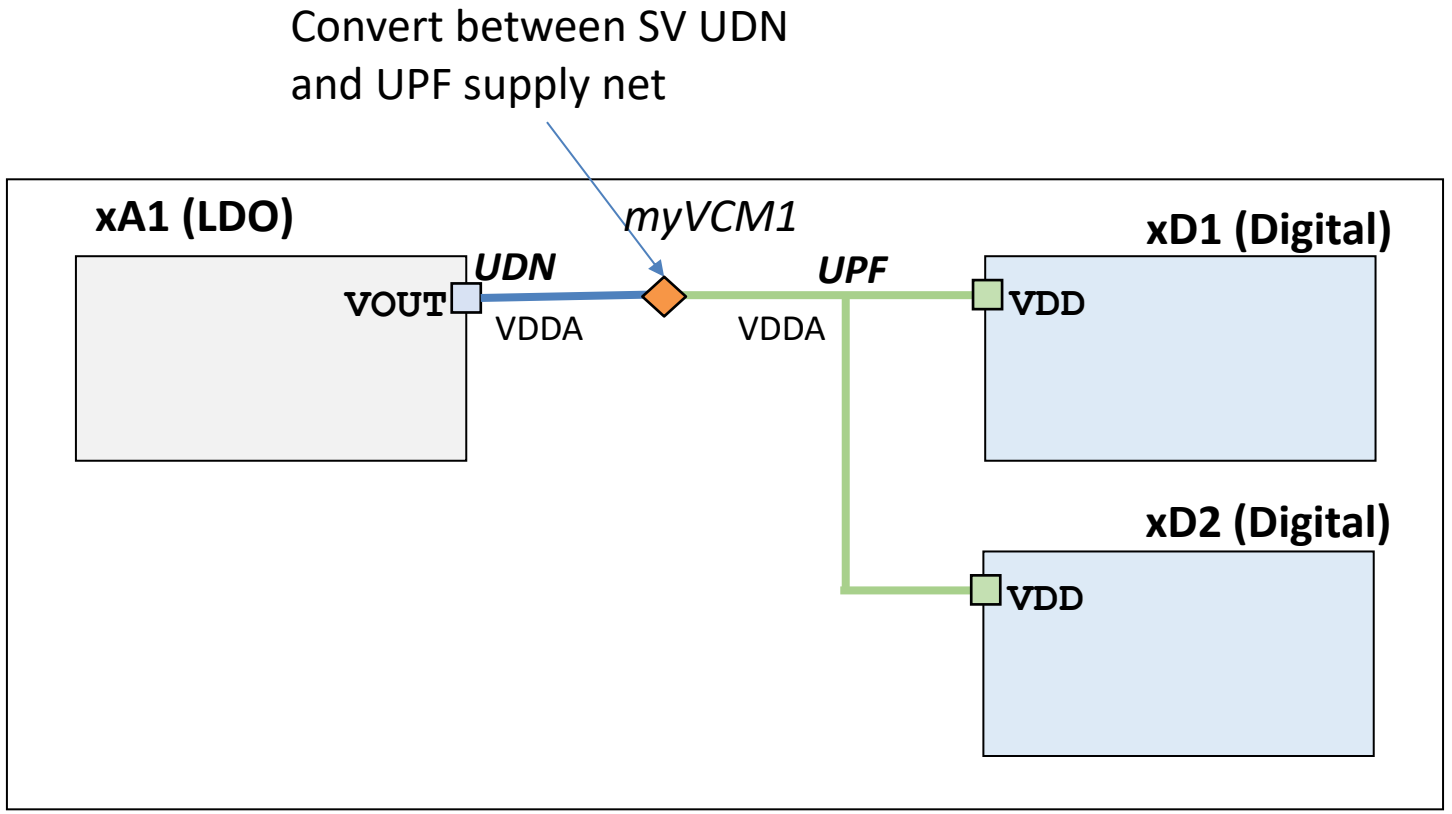
Syntax:

`create_vcm` *vcm_name*

with one of the following option sets:

- **`-table`** {{*from_value to_value*}*}
  -hdl_type {<vhdl | sv> [*HDL_typename*]}
  -conversion_direction <hdl2upf | upf2hdl>
  -field *field_name*
- **`-function`** *hdl_package::function_name*
- **`-model`** *module_name*
  -parameters {{*param_name param_value* }*}
- **`-vcms`** *ext_vcm_list*

Example | Where VCMs are Inserted




```
connect_supply_net VDDA -ports {xA1/VOUT xD1/VDD xD2/VDD} \
        -vcm myVCM1
```

- In addition to single-bit logic, the VCM allows connecting UPF supply nets to:
  - integer
  - enum
  - real
  - User-Defined Nettype (UDN)
  - UDN with struct
  - record
- Only UPF supply nets (not logic nets or supply sets) can be connected with these methods

Example | Table VCM

- Consider a SystemVerilog package with a UDN defined as follows:

```
package ldo_net_pkg;
typedef struct {
            real     volts;
            real     current;
} ldo_struct_t ;
nettype ldo_struct_t ldo_supply_net;
endpackage
```

- A VCM that maps values on an HDL port of type *ldo_supply_net* can be declared as follows:

```
create_vcm LDONET2UPF \
-hdl_type {sv ldo_net_pkg::ldo_supply_net} \
-conversion_direction hdl2upf \
-field volts \
-table { \
            {{5.0   * } {OFF             }} \
            {{1.0 5.0} {FULL_ON    1.1   }} \
            {{0.6 1.0} {PARTIAL_ON 0.9   }} \
            {{ *  0.6} {OFF             }} \
      }
```

Mapping of HDL voltage to UPF supply net state and voltage mapping is done values defined in table argument.

Example | Function VCM

```
package myVCM_pkg;
   import UPF::*;
   import ldo_net_pkg::*;

   function automatic upfSupplyTypeT func_h2u_snap (ldo_struct_t hdl_in);
      upfSupplyTypeT             upf_out;

      upf_out.voltage = 0;
      upf_out.state              = UNDETERMINED;

      if (hdl_in.volts <= 0.2) begin
                                 upf_out.voltage       = 0;
                                 upf_out.state                    = OFF;
      end
      else if ((hdl_in.volts > 0.2)
            && (hdl_in.volts <= 0.9)) begin
                                 upf_out.voltage       = 0.9;
                                 upf_out.state                    = FULL_ON;
      end
           . . .
      return          upf_out;
   endfunction
endpackage
```

```
# Using the package in UPF
create_vcm LDONET2UPF_function     -function myVCM_pkg::func_h2u_snap
```

Example | Model VCM

```
import UPF::*; import ldo_net_pkg::*;
module snap_volt_vcm #(
          //default parameter defs
          parameter ov_threshold                  = 5.0,
          parameter hi_snap_volts                 = 1.1,
          parameter hi_threshold                  = 1.0,
          parameter on_snap_volts                 = 0.9,
          parameter on_threshold                  = 0.2
 )   (
          input                ldo_supply_net                hdl_in,
          output               upfSupplyTypeT                upf_out
);

always @(hdl_in.volts)
   begin
      . . .
   end
endmodule
```

```
create_vcm LDONET2UPF_module \
  -model  my_module_lib.snap_volt_vcm \
  -parameters { \
          {ov_threshold       5.0} \
          {hi_snap_volts      1.5} \
          {hi_threshold       1.4} \
          {on_snap_volts      1.0} \
          {on_threshold       0.5} }
```

Advantages of Functions over Modules

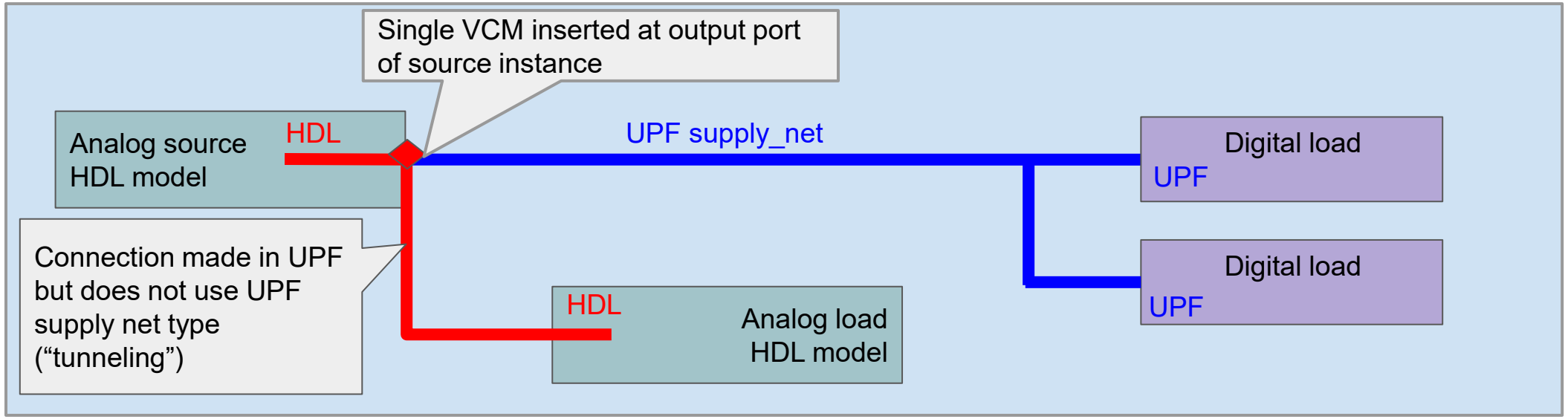
- Functions are less resource intensive
  - They exist for one event and then disappear
  - Modules are instantiated in the netlist, and exist for the entire simulation, occupying memory even when nothing happens to them
- Functions can be imported from other languages (e.g. C++)
  - Modules must be HDL

Advantages of Modules over Functions

- Modules can be easily parameterized
  - One module description can be used as the basis for many VCMs (using **create_vcm –*parameters***)
- Modules can model time delay effects
  - Since they are static objects, they can respond to stimuli over several event times
  - More complex behaviors can be modeled

Tunneling

- Tunneling allows a connection between same type of HDL net to preserve the full extent of that type
  - Normal UPF conversions result in loss of information
    - UPF supply type only has an integer voltage and *supply_state*
  - HDL type may contain richer information set that load HDL model may take advantage of



UPF Library

- Introduced to provide a way to avoid name collisions between VCM definitions, which are otherwise global in scope
  - Helps define independent VCMs for different IPs and SoC, avoiding name collisions
  - Hierarchical and modular use-model
- New commands which support UPF library use:

```
create_upf_library upf_library_name \
   -contents { \
      upf commands \
   }

use_upf_library upf_library_name

load_upf_library upf_library_file
```

Only ***create_vcm***, **load_upf_library** **use_upf_library** currently allowed

Automatic Selection of VCM by Net/Data Type

- Motivation:
  - UPF supply nets may connect to multiple HDL types
  - The ***–vcms*** option allows the specification of a list of VCMs
  - Tools can choose the matching VCM based on the HDL type and do the proper conversion

```
create_vcm vcm_bundle \
    -vcms {sv_logic2upf sv_real2upf sv_udn2upf}
```

Example | List VCM

```
create_vcm vcm_bundle \
    -vcms {sv_logic2upf sv_real2upf sv_udn2upf}
```

- Can include VCMs for both directions in a list, as long as the ports connect using it are **input** or **output** (but not **inout**)
- Can only include one VCM with a given HDL type and direction in the list
- All VCMs in the list have to be previously defined with a **create_vcm** command
- The list can be a mix of table VCMs, function VCMs, module VCMs, or even other list VCMs
- Use the name of the list VCM to make UPF supply net connections to HDL ports

Commands and Option Changes

- New Commands and options
  - create_vcm
  - create_supply_net -tunneling
  - connect_supply_net -vcm
  - create_upf_library
  - use_upf_library
  - load_upf_library

- Legacy Commands and options
  - The commands **`create_hdl2upf_vct`** and **`create_upf2hdl_vct`** are legacy
    - Can be supported alongside VCMs by redirecting calls to these commands to **`create_vcm`**, supplying necessary **`-conversion_direction`**
  - **`-vct`** option in **`connect_supply_net`** is also legacy
    - Tools may continue to support **`-vct`**

Refinable Macros and Terminal Boundaries in UPF 4.0: Empowering Soft IPs of the Future

Manash Ranjan Raiguru

Synopsys (India) Pvt. Ltd

Amit Srivastava, Lakshmanan Balasubramanian, John Decker, "Refinable Macros and Terminal Boundaries in UPF 4.0: Empowering Soft IPs of the Future," DVCON 2025.

accellera SYSTEMS INITIATIVE™

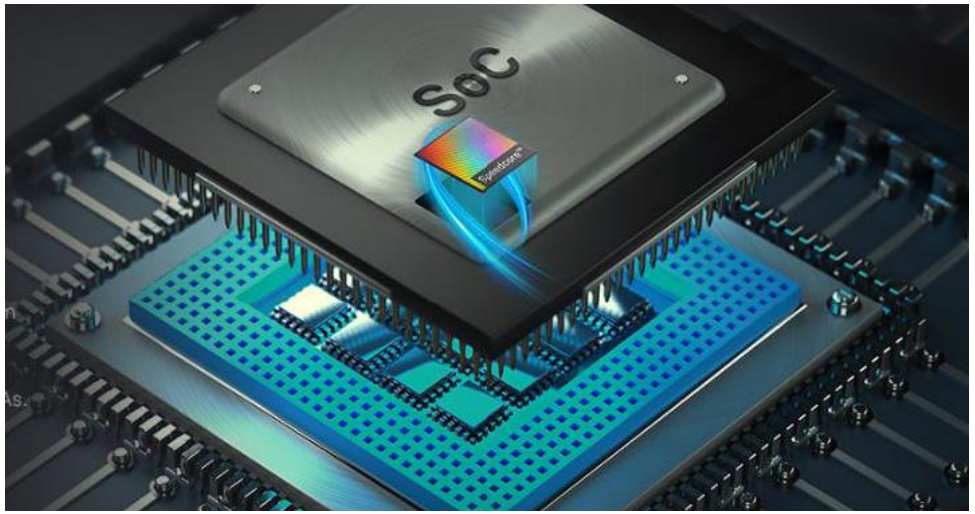
SYNOPSYS®

2025 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION INDIA

Refinable Macros in UPF 4.0

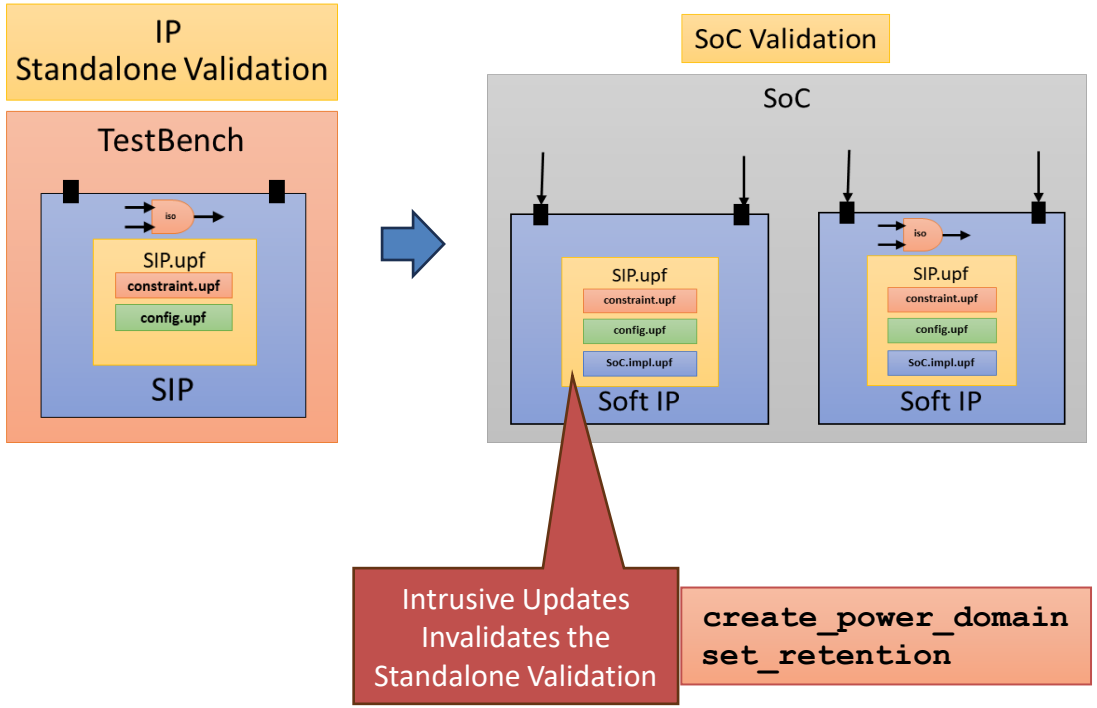
Empowering Soft IPs of the Future

- ***Enabling non-intrusive refinements in bottom-up verification flows***
- **Agenda**
  - Why we need Refinable Macros
  - How they differ from Soft Macros
  - Marking IPs as Refinable Macro
  - Refinable Macro in action



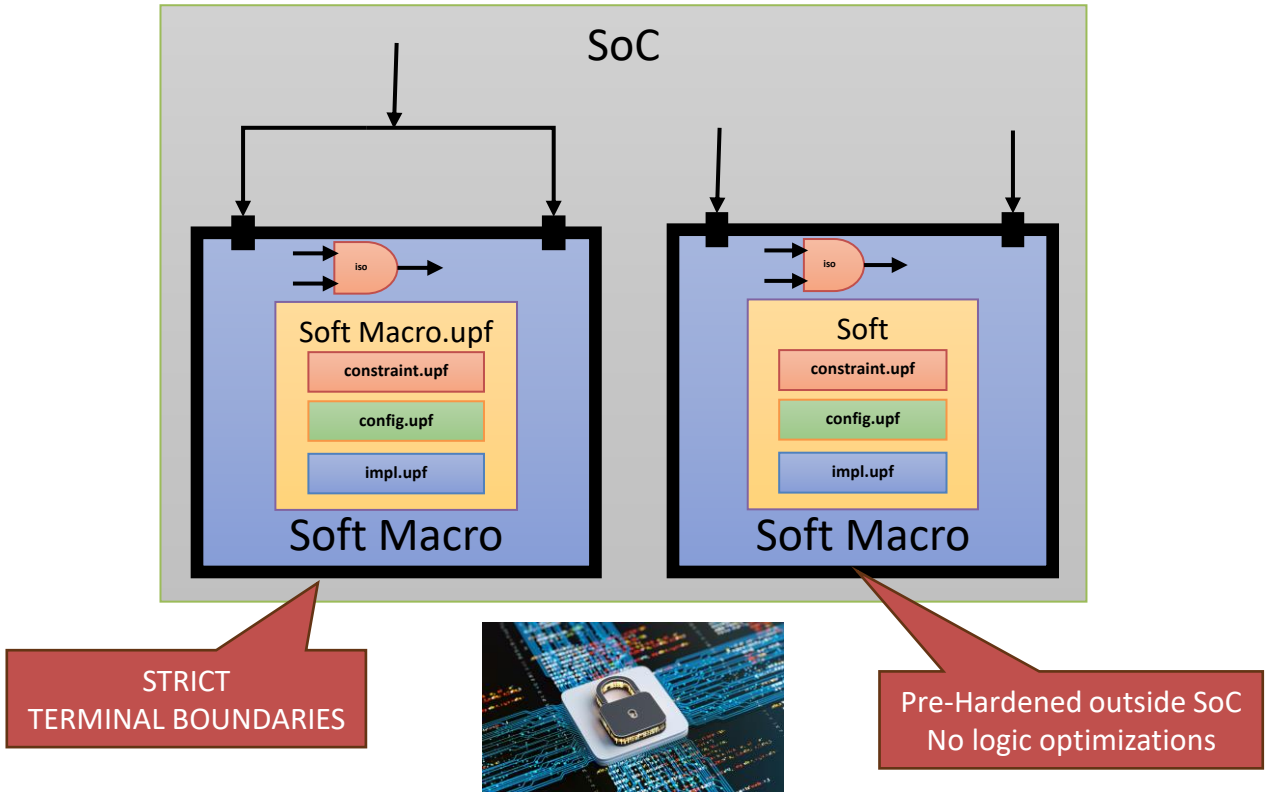
Motivation: Challenges with SIPs in Bottom-Up Verification

- **SoC Complexity**
  - 50+ SIPs, each with unique power rules
  - Higher-Level Implementation Requires Power Intent **Updates**
- **The Dilemma** after Modifying SIPs
  - **Not revalidate** SIPs → risk introducing bugs
  - **Revalidate** SIPs → delay Time to Market
- **UPF 3.1's Fix: Soft Macro**
  - Good for implementation
  - Too rigid for pre-verified SIPs



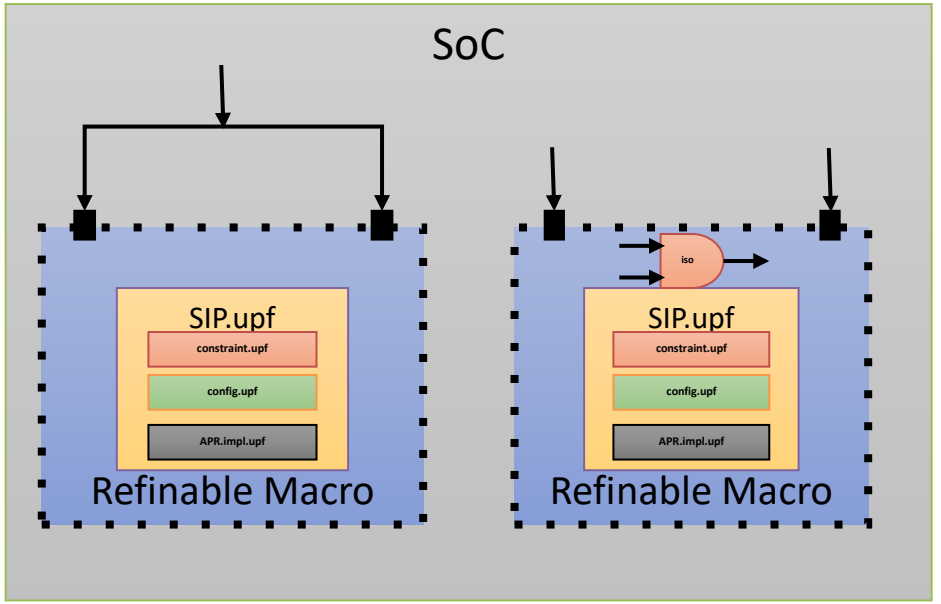
Why Soft Macros Fall Short for Bottom-Up Verification?

- **Rigid Implementation Boundaries**
  - Protects IP from external overrides
  - Suitable for Bottom-Up Implementation Flows
- No Room for **Non-Intrusive Refinements**
- Forces **Intrusive Edits** and Re-Validation
- Lacks **System-Level Optimization**



Refinable Macros in UPF 4.0

- Refinable Terminal Boundaries
- **Tool-Enforced Safety**
  - Non-Intrusive Power Intent Updates
- **Preserved Verification**
  - Original IP UPF remains untouched
- **Enables** System-Level **Optimization** during Implementation
- Ideal for **Bottom-Up Verification**



Identical Views for SoC RTL Simulation and SoC Implementation

Marking IPs as Refinable Macros

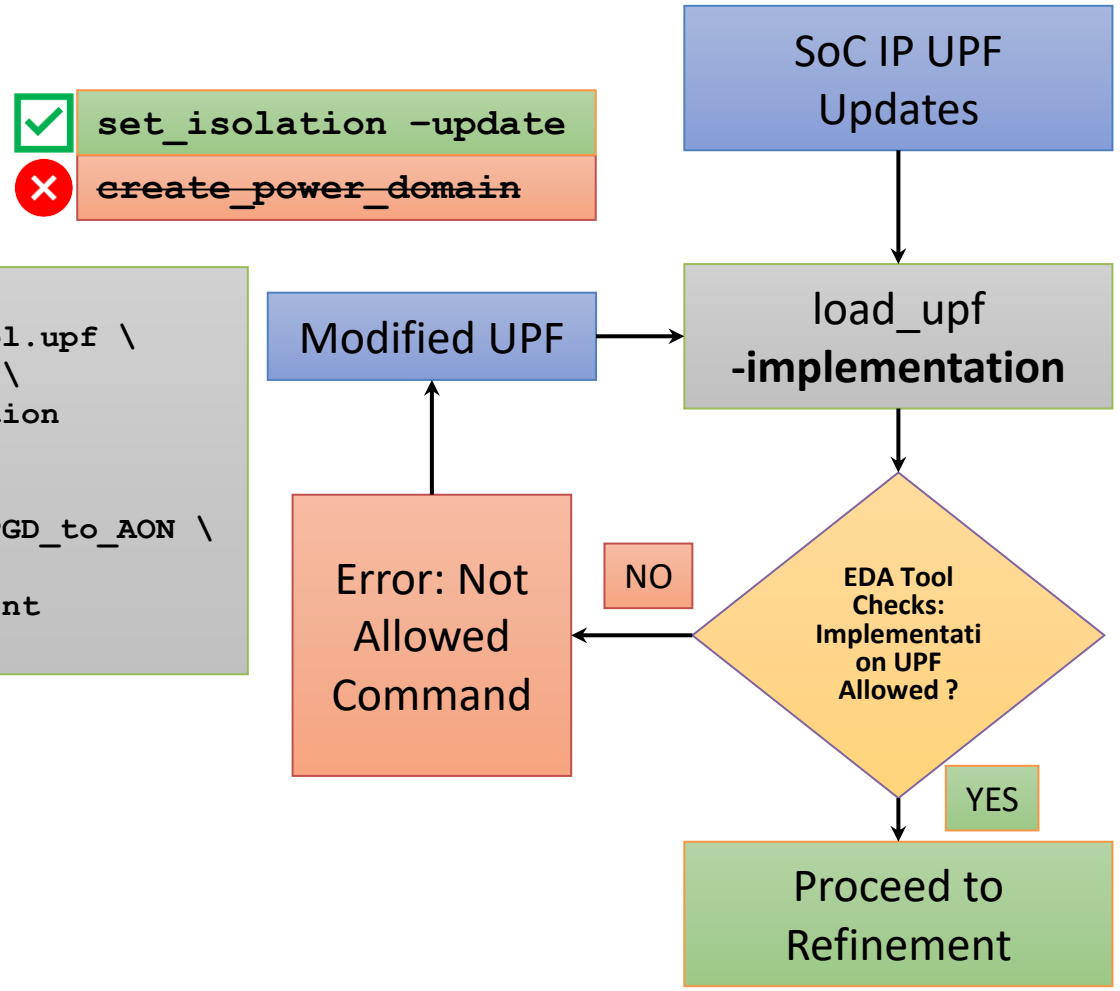
- Simple UPF Attribute
- Mark IP internally or externally
- Maintains IP Verification
- Override to Soft Macro if Needed


```
# Mark directly in IP UPF:
set_design_attributes -models . -is_refinable_macro true

# Or mark externally:
set_design_attributes -models IP_Design -attribute {UPF_is_refinable_macro TRUE}
```

Implementation UPF: Enforcing Safe Refinements

- **Safe Refinements** via Allowed Commands and options
  - UPF 4.0 defines allowed commands and options
- **-implementation** Enforces **Correct-by-Construct** UPF
  - EDA Tools Enforce Compliance
- **No Alteration** of Original IP UPF
- **Preserves Verification** Integrity

```
# SoC UPF
load_upf ip_impl.upf \
   -scope myIP \
   -implementation

# ip_impl.upf
set_isolation PGD_to_AON \
 -domain PGD \
 -location parent
 -update
```

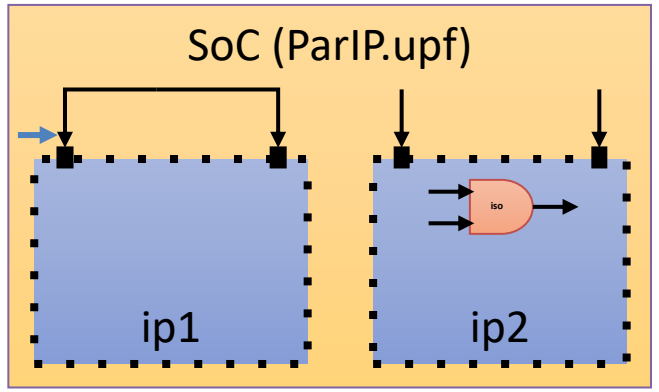


Example: Refinable Macros in Action

ip.upf

```
set_design_attributes -models . \
-is_refinable_macro TRUE

create_supply_set ss_IP_AON
create_supply_set ss_IP_PGD

create_power_domain AON -elements {.}
create_power_domain PGD -elements
{ip1_pgd_wrapper}

## Isolates all outputs where different
## supplies power source and sink
set_isolation PGD_to_AON -domain PGD \
 -isolation_supply_set ss_IP_AON \
 -applies_to outputs -source ss_IP_PGD \
 -diff_supply_only TRUE \
 -isolation_signal pwr_manager/iso_en_b \
 -isolation_sense low
```

AON and PGD supplies are shorted for one instance of the IP



ParIP.upf

```
create_power_domain par_AON -elements {.}
create_supply_set ss_SOC_AON
create_supply_set ss_SOC_PGD

load_upf ip.upf -scope ip1
load_upf ip_impl.upf -scope ip1 -implementation
associate_supply_set {ss_SOC_AON ip1/ss_IP_AON}
associate_supply_set {ss_SOC_AON ip1/ss_IP_PGD}

load_upf ip.upf -scope ip2
load_upf ip_impl.upf -scope ip2 -implementation
associate_supply_set {ss_SOC_AON ip2/ss_IP_AON}
associate_supply_set {ss_SOC_PGD ip2/ss_IP_PGD}
```

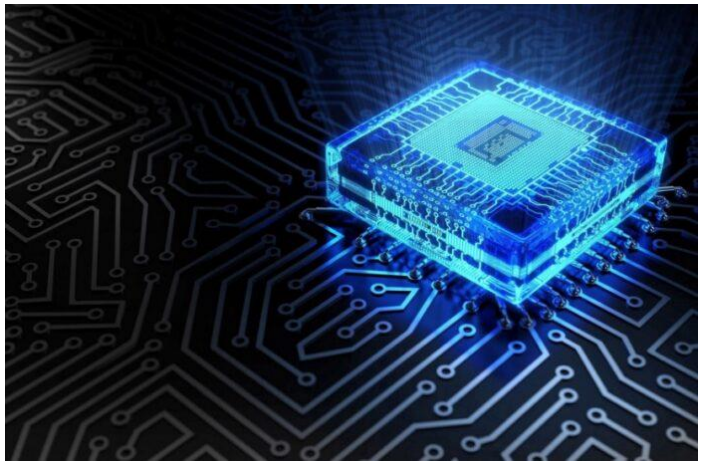
ip_impl.upf

```
set_isolation PGD_to_AON \
 -domain PGD \
 -location parent
 -update
```

Implementation updates only

Conclusion: Empowering Soft IPs of the Future

- UPF 4.0 → Bridges SIP Verification Gaps
- Refinable Macros = Flexibility + Safety + Performance
- Implementation UPF → Correct-by-Construct
- Preserves Verification & Saves Time



Retention Modeling in UPF 4.0

Shubham Saha

Texas Instruments (India) Pvt. Ltd.

Lakshmanan Balasubramanian, et al, "Future Proofing Power Intent Specification through Unified Power Format 4.0 for Evolving Advanced State Retention Strategies," DVCON 2025

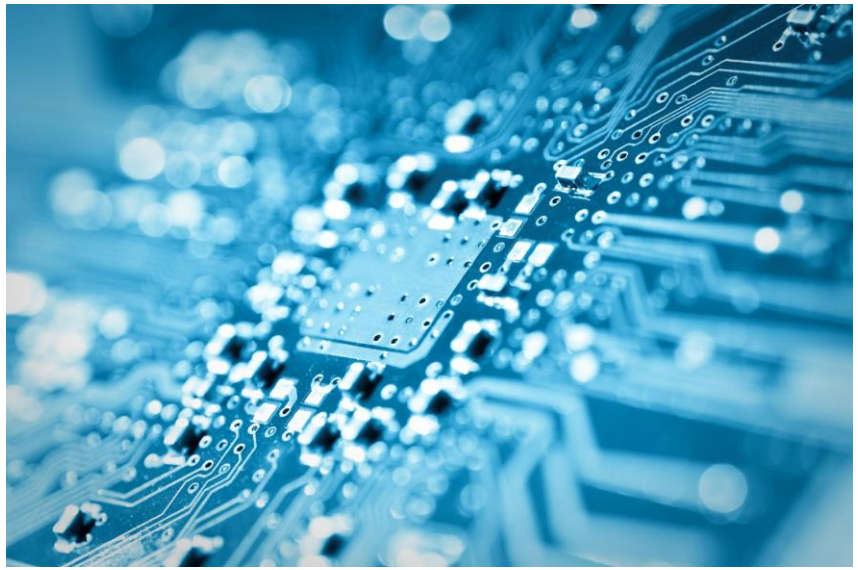
accellera SYSTEMS INITIATIVE™

Texas Instruments

2025 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION INDIA

Future Proofing Retention in UPF 4.0

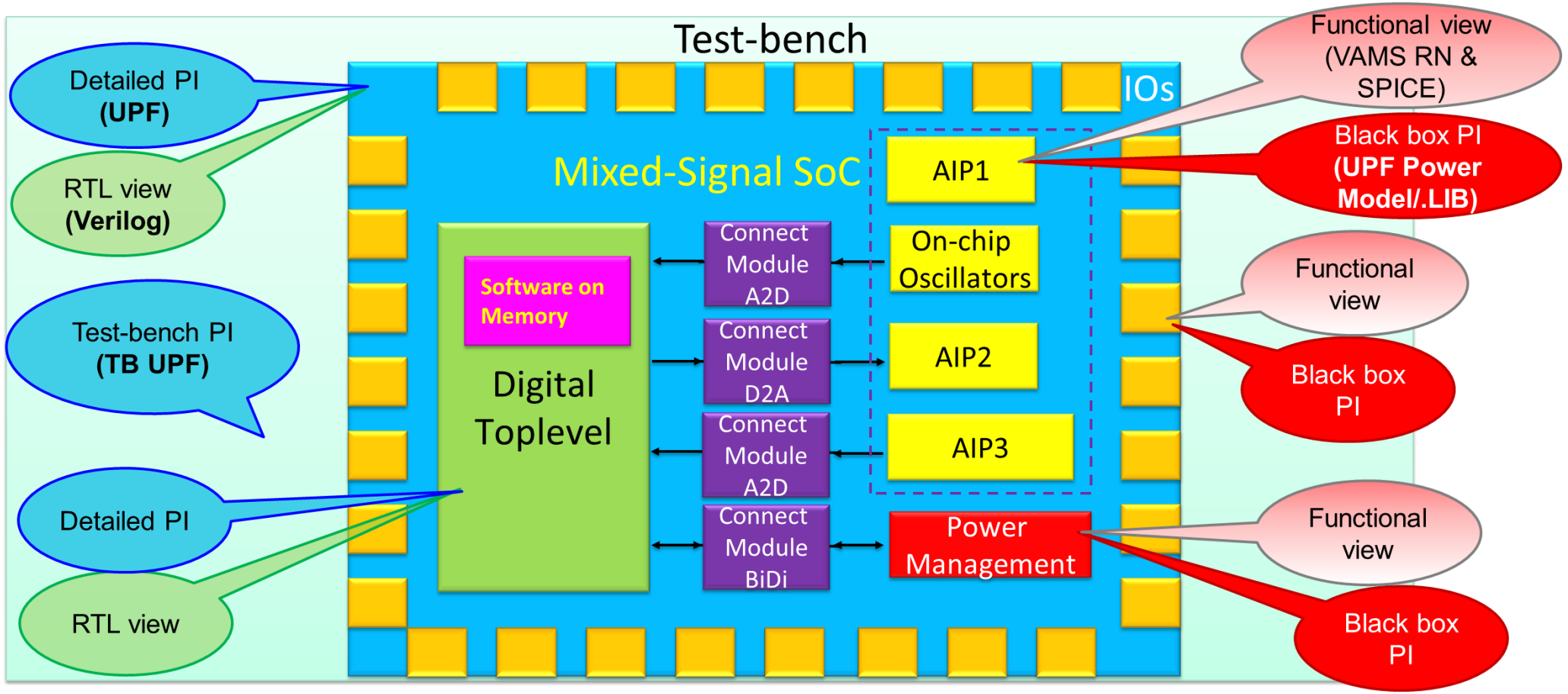
- **Agenda**
  - What is Advanced SR?
  - Why retention semantics were updated?
  - What is new in UPF 4.0?
  - Examples enabled by new changes



Motivation

- Advances in state retention cell design have exposed limitations in earlier versions of the UPF LRM
- Enhancements needed to model more complex clock, setup, retention relationships provided by these new technologies
- Improved modeling will catch issues early in the design cycle
- UPF 4.0 improvements designed to be forward looking and provide a flexible platform that can adjust to future requirements

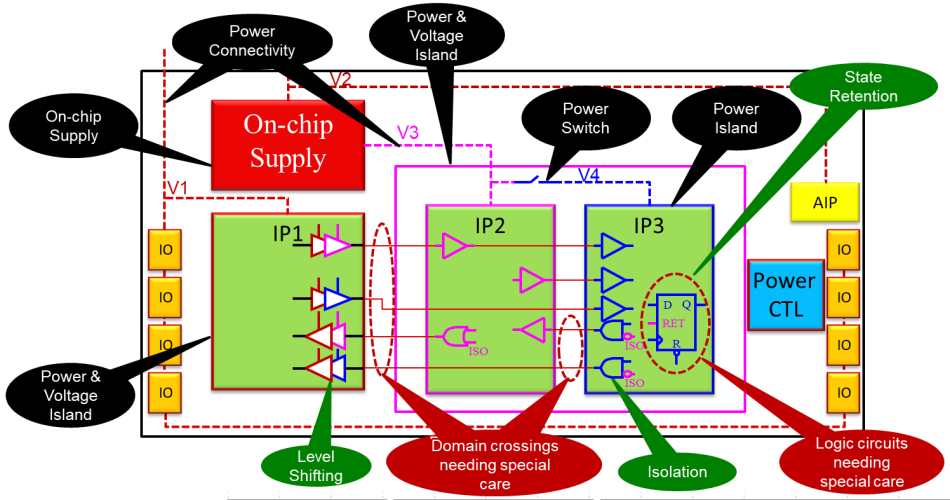
A Typical Power-Aware Verification Setup



Vijayakumar Sankaran, et al, Modern Recipes for Brewing the Inevitable Methodology for Today's ICs: Low-Power Mixed-Signal Design Verification, Tutorial, DAC 2019

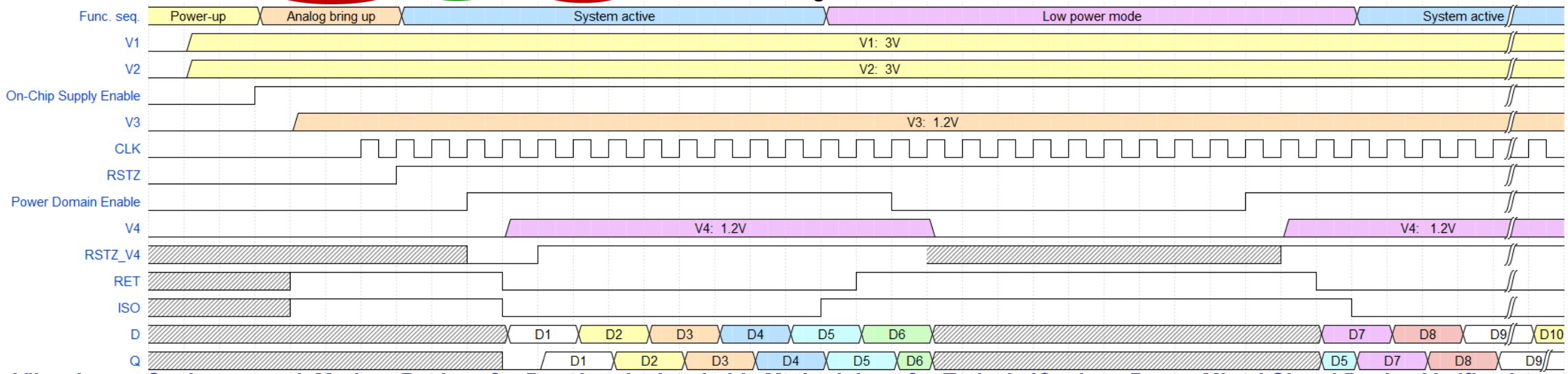
Lakshmanan B, et al, A holistic approach to low power mixed-signal design verification using power intent: CPF-Based Interface Elements, Methods, and Guidelines, DVCON 2016

A Typical Low-Power System & Behaviour



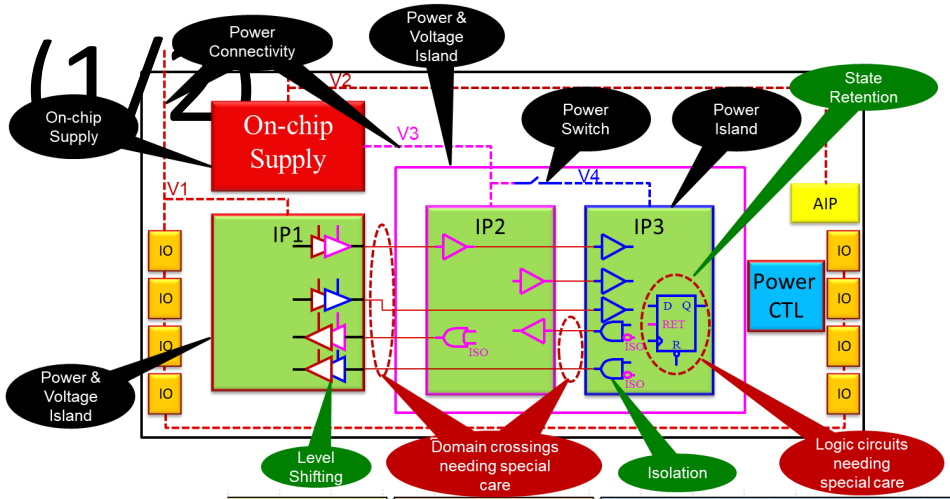
Legend:

V1, V2: Primary power supplies
V3, V4: On-chip switched power supplies
CLK: Clock signal
RSTZ: Active low reset signal
RET: Active high state retention control signal
ISO: Active high isolation control signal
D: Input data to a state retention cell
Q: Output of a state retention cell
RSTZ_V4: Reset for the power domain V4

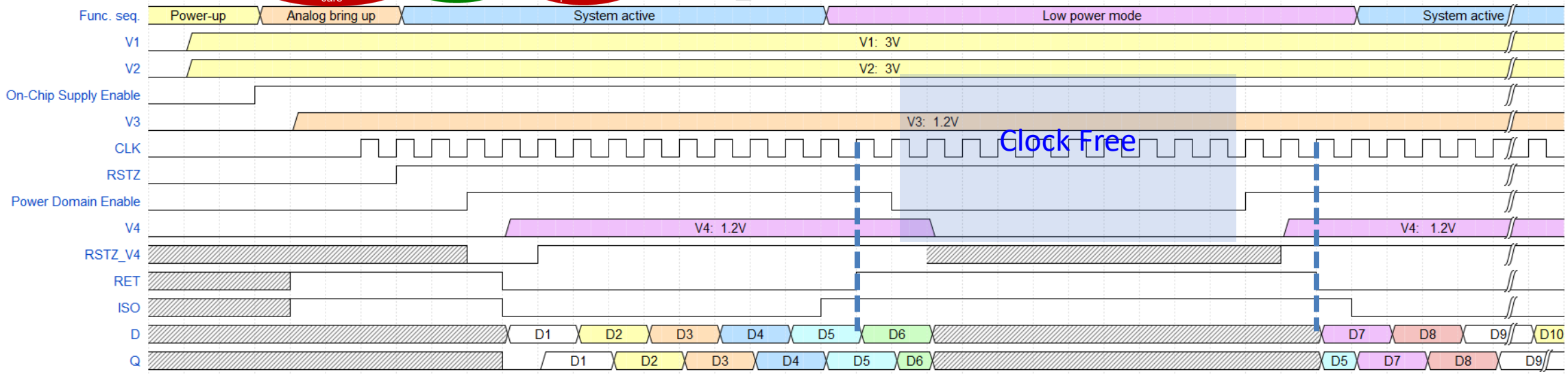


Vijayakumar Sankaran, et al, Modern Recipes for Brewing the Inevitable Methodology for Today's ICs: Low-Power Mixed-Signal Design Verification, Tutorial, DAC 2019

Typical LP System| UPF 3.1 Abstraction & Signal-Level Behaviour

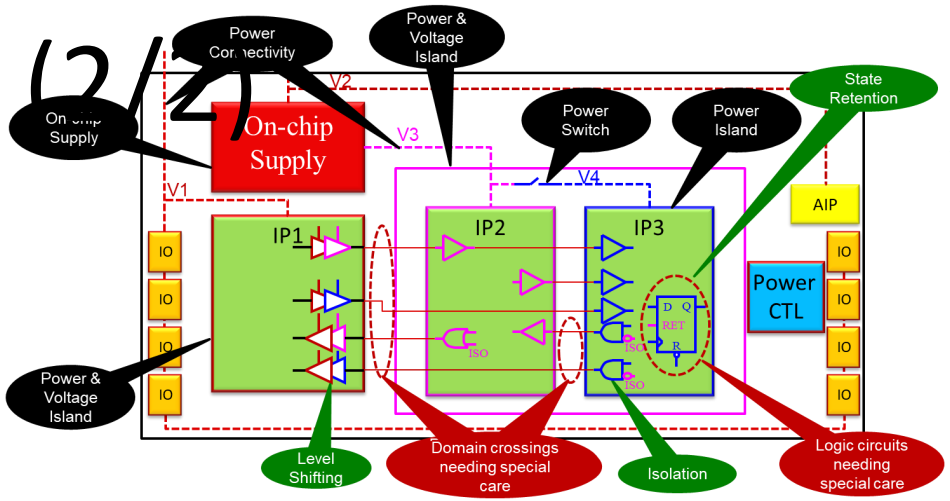



```
upf_version 3.1

set_retention ret -domain PD_V4 \
-save_signal {RET posedge}
-restore_signal {RET negedge} ...

define_retention_cell -cells SR1 -clock_pin CLK \
-save_function {RET posedge} -restore_function {RET negedge} \
-power_switchable VDD -power VDD_RET -ground VSS
```

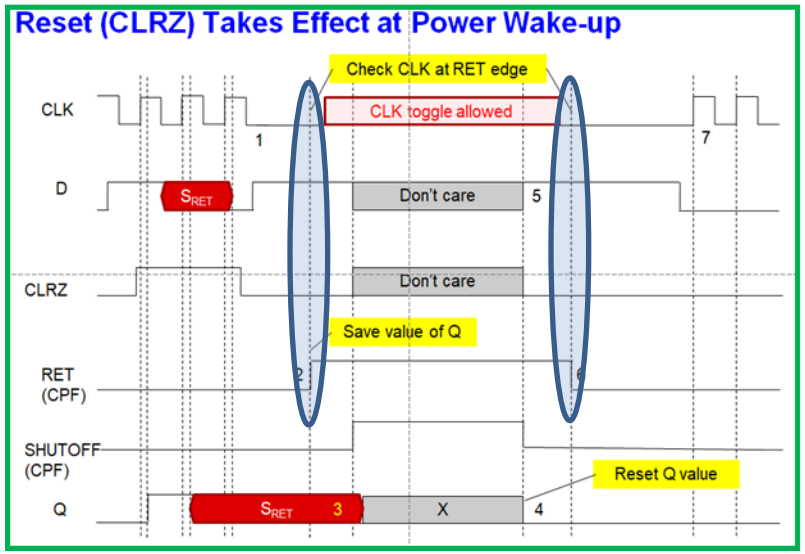


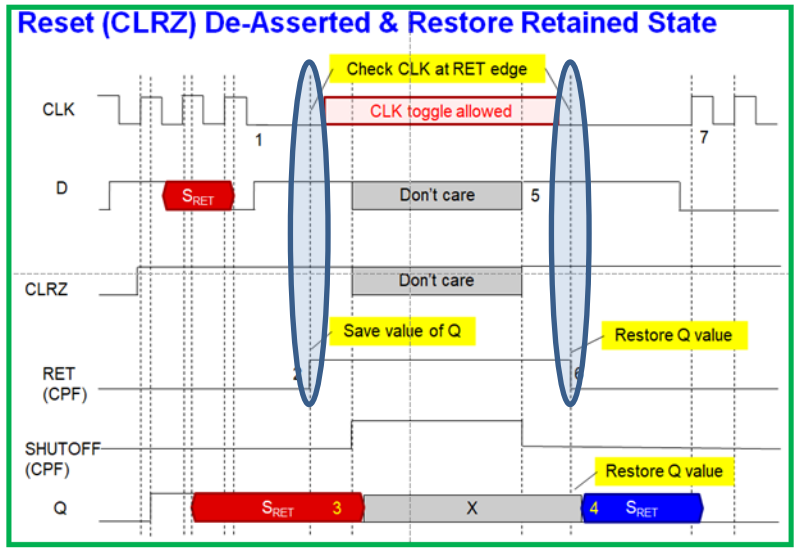
Vijayakumar Sankaran, et al, Modern Recipes for Brewing the Inevitable Methodology for Today's ICs: Low-Power Mixed-Signal Design Verification, Tutorial, DAC 2019

Typical LP System| UPF 3.1 Abstraction & Signal-Level Behaviour




```
upf_version 3.1

set_retention ret -domain PD_V4 -save_signal {RET posedge} \
-restore_signal {RET negedge} \
-save_condition {!CLK} -restore_condition {!CLK} …
```

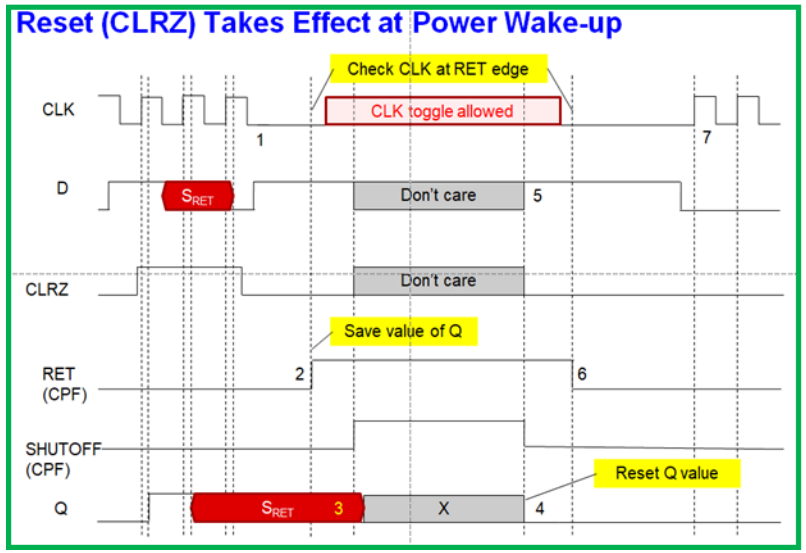


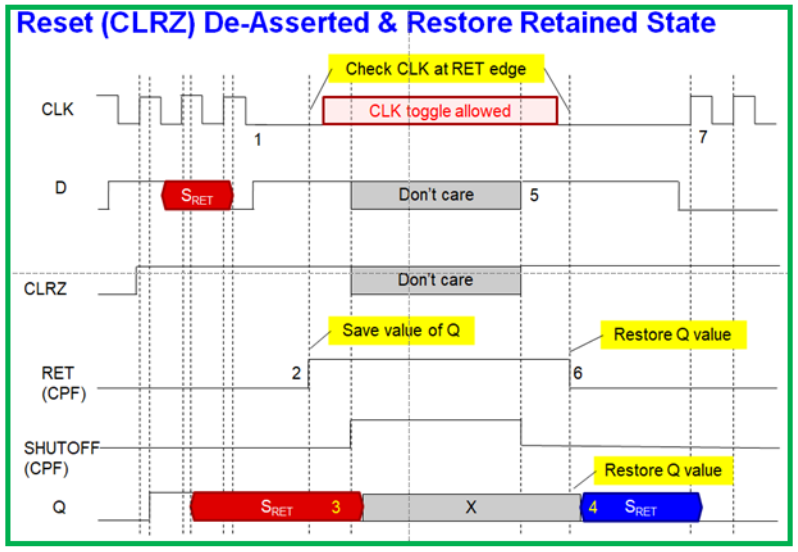


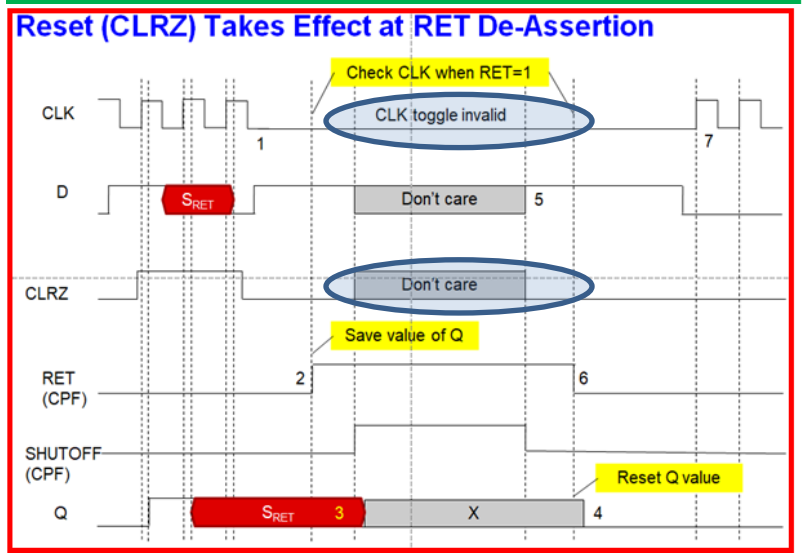
Lakshmanan Balasubramanian, et al, Advanced Low Power Retention Simulation Framework, Designer Track, DAC 2019

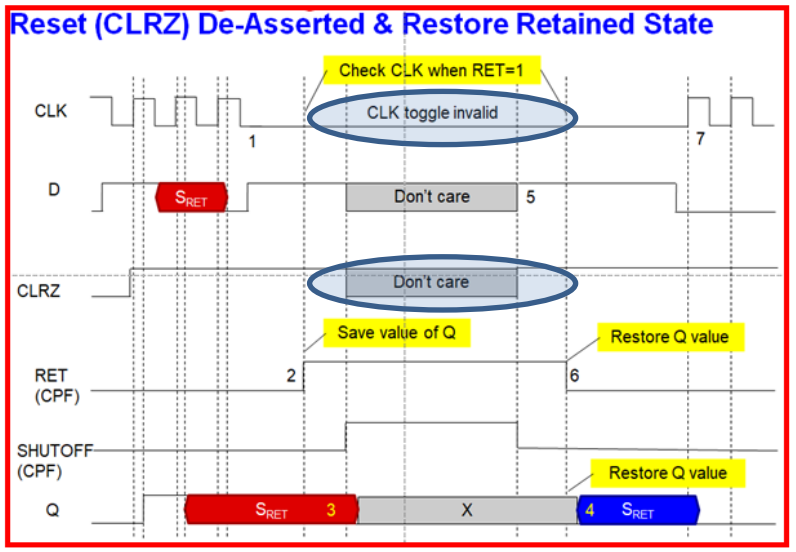
Evolving SR Strategies

- Conventional SR techniques
  - Sub-optimal implementation for evolving technologies and end applications
- Evolving SR strategies
  - Area and power efficient SR elements utilising lower i.e., device-level optimisations
  - May impose additional functional and electrical boundary conditions



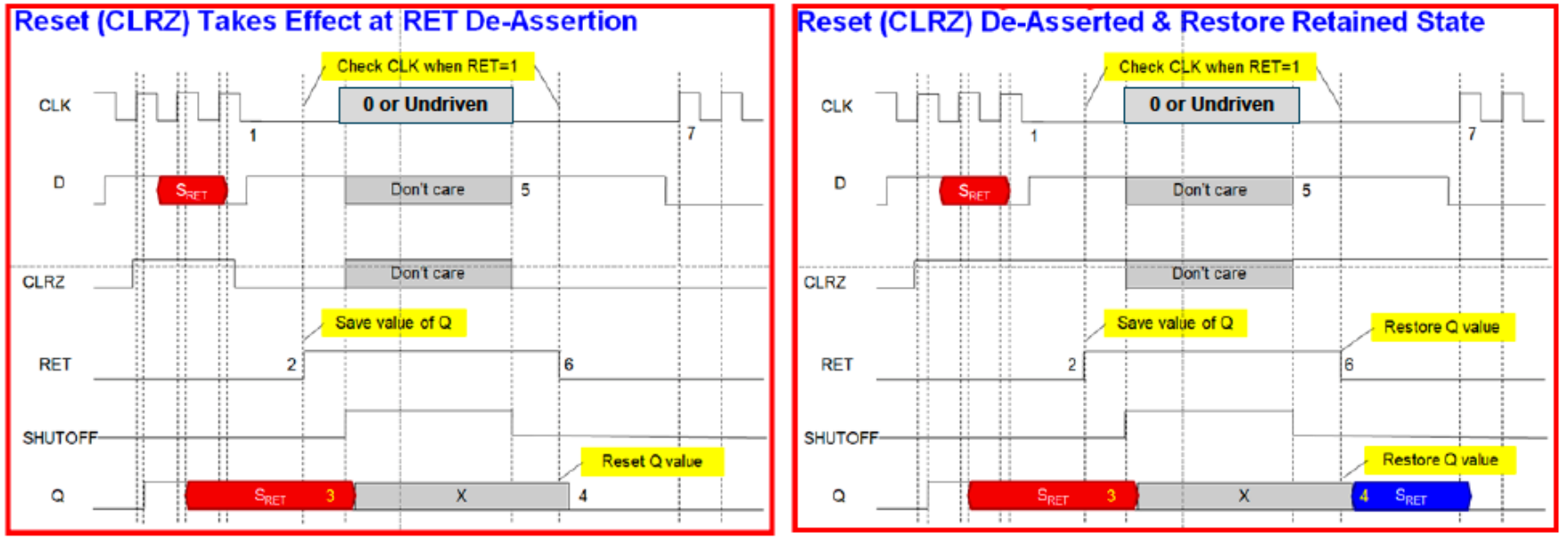






Lakshmanan Balasubramanian, et al, Advanced Low Power Retention Simulation Framework, Designer Track, DAC 2019

Advanced SR



(i) Reset Functionality *(ii) Retention Restore Functionality*

- Improves area & optimizes power by allowing the CLK to
  - be ignored during retention, enabling the clock driver to be powered off during shutdown (state retention handles this with more resilient circuitry)
  - not gate clock internally but put system level constraints to ensure it's in a known state (ex – logic 0) when RET is asserted
- Such strategies require further nuances in the syntactical representation and interpretation of set_retention

Lakshmanan Balasubramanian, et al, Advanced Low Power Retention Simulation Framework, Designer Track, DAC 2019

Retention Overview of changes

- Existing set_retention conditions were expanded and redefined to be more accurate
- Behavior will closely match the retention cells and maintain high performance of RTL
- Ability to specify how set/reset will affect the behaviors

Earlier UPF set_retention
-restore_condition
-save_condition
-retention_condition

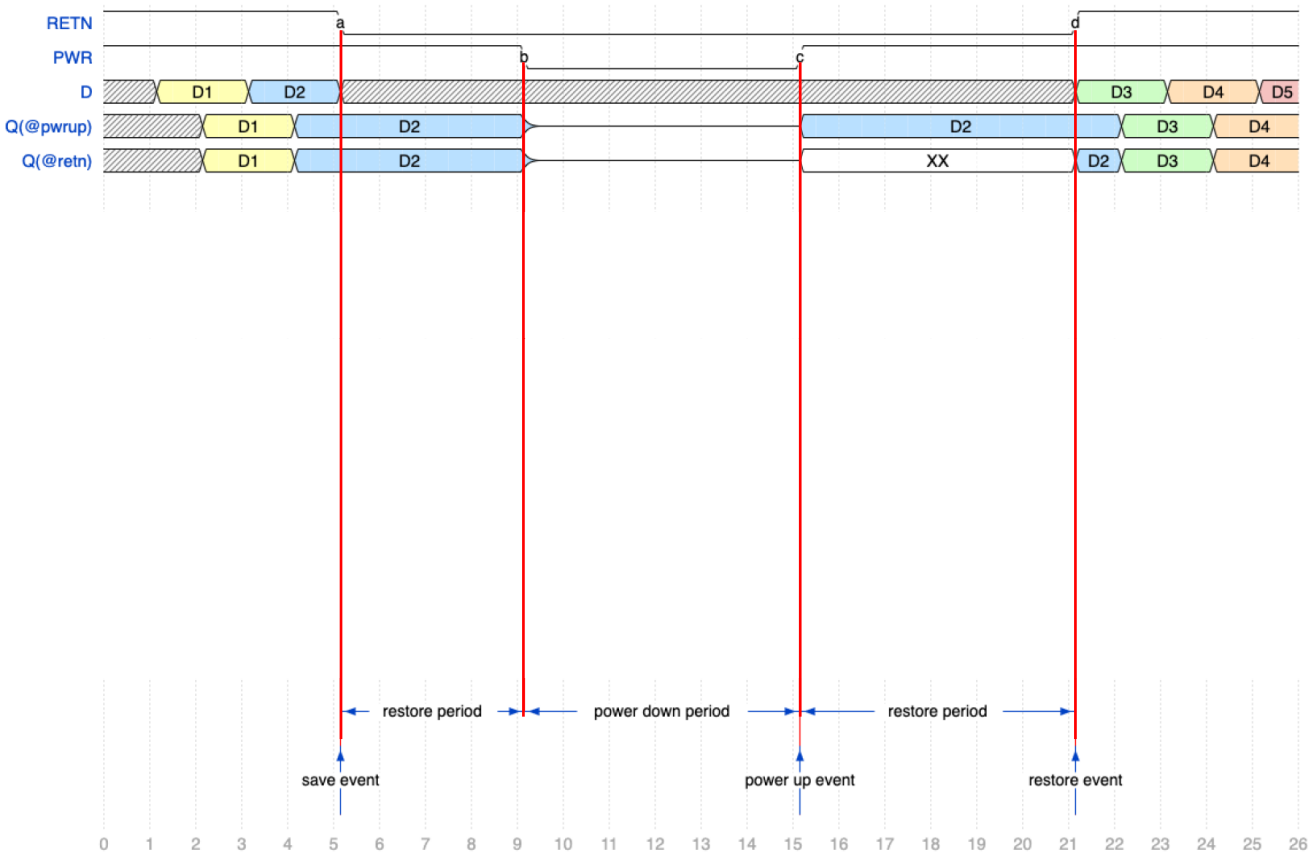
4.0 set_retention
-restore_event_condition
-save_event_condition
-restore_period_condition
-powerdown_period_condition
-async_set_reset_effect

Extensions in UPF 4.0

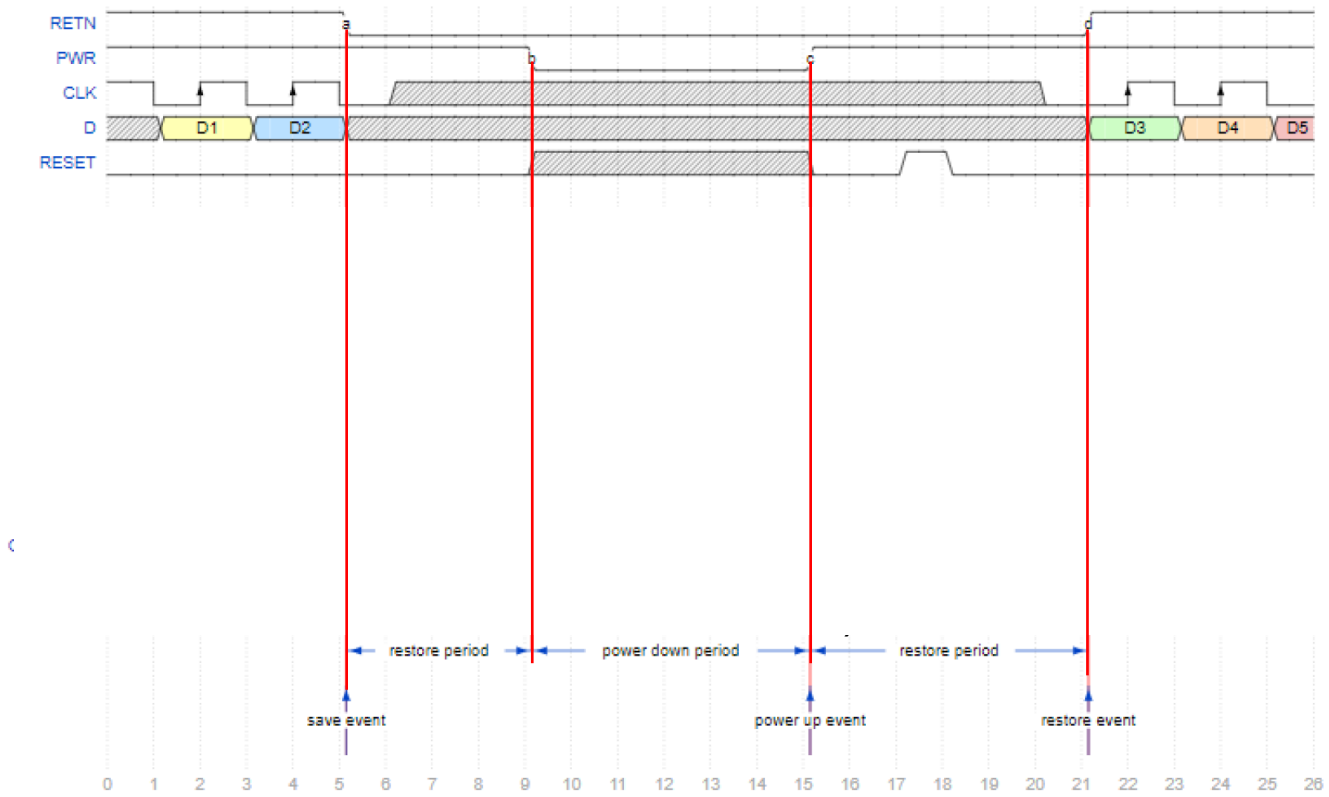
upf_version 4.0
set_retention *retention_name* **-domain** *domain_...* \
[**-save_signal {***logic_net* **<high | low | posedge | negedge>}** \
[**-restore_signal {***logic_net* **<high | low | posedge | negedge >}**] \
[**-save_event_condition {***boolean_expression***}**] \
[**-restore_event_condition {***boolean_expression***}**] \
[**-powerdown_period_condition {***boolean expression***}**] \ — **New Semantic Features**
[**-restore_period_condition {***boolean expression***}**] \
[**-async_set_reset_effect <ignored | retained_value | output_value>**] \
[**-*applies_to*** *<flop | latch | any>]* \ — **New Filter**
… [**-save_condition {***boolean_expression***}**] \ — **Declared Legacy**
[**-restore_condition {***boolean_expression***}**] \ — **for Backward Compatibility**
[**-retention_condition {***boolean_expression***}**] — **→ Cannot Co-Exist with New Features**

Example | Single-Control SR

- 4.0 clearly defines the periods of the full retention cycle
- Each period can have an independent set of requirements relative to clock, reset, and other design signals
- Overcomes limits and ambiguities in previous versions



Example | Single-Control SR
async_set_reset_effect



a) Reset is ignored during the restore period

***-async_set_reset_effect** ignored*

b) Reset affects the retained value and therefore the output value during the restore period

***-async_set_reset_effect** retained_value*

c) Reset affects the output value during the restore period

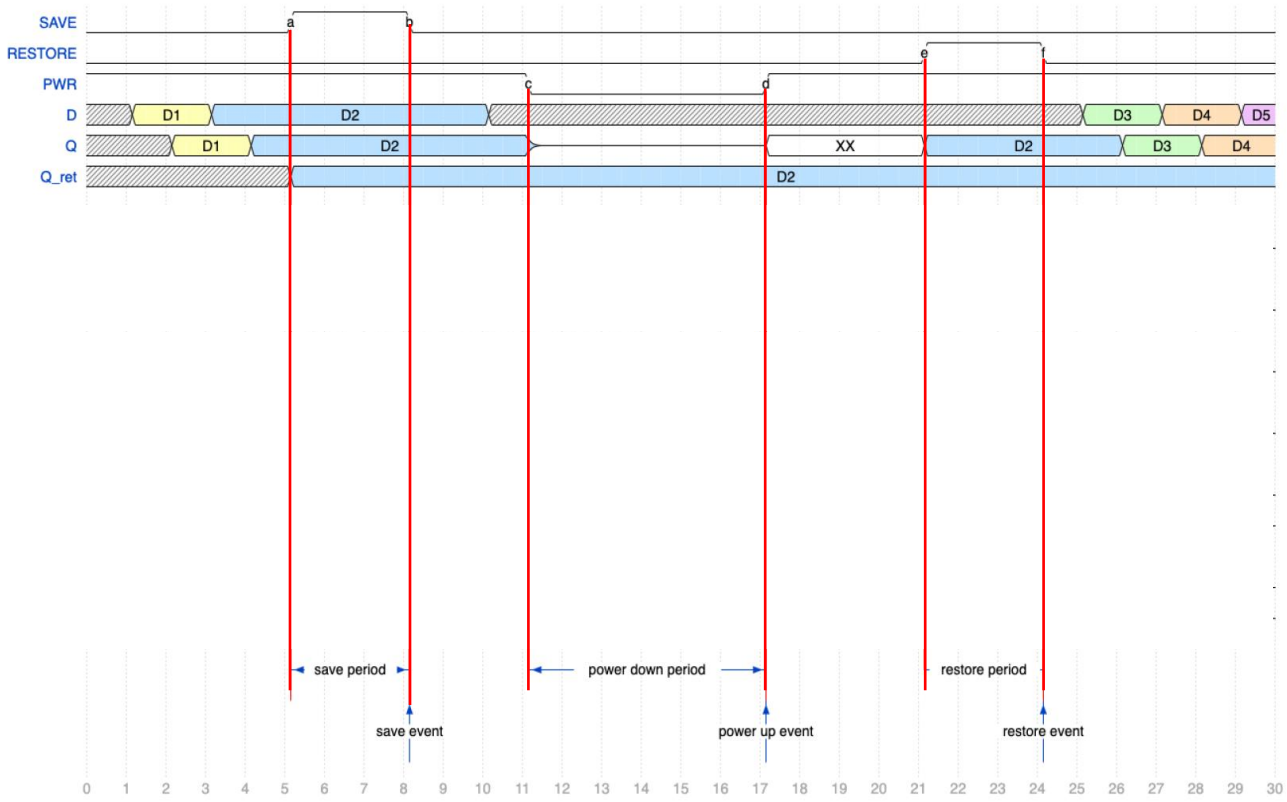
***-async_set_reset_effect** output_value*

Additional Changes for Retention

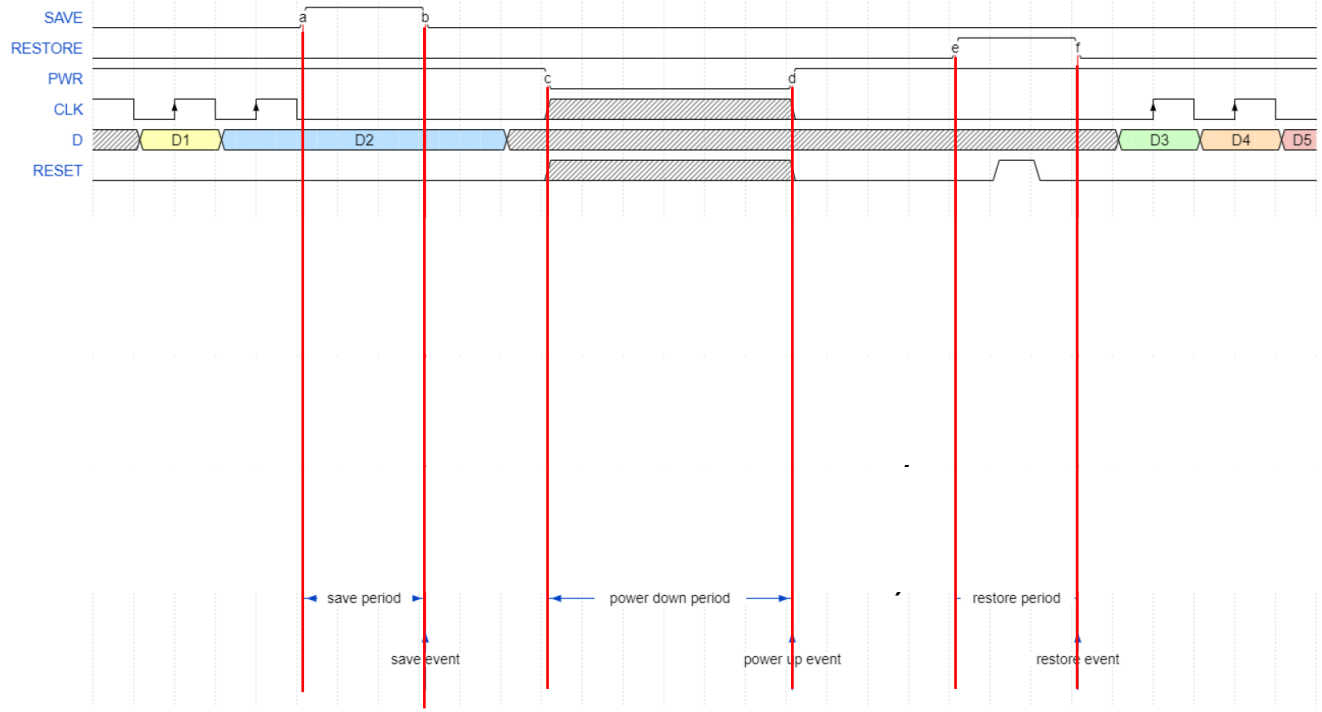
- Improved definition of **UPF_GENERIC CLOCK**
- Allow **UPF_GENERIC_CLOCK** and **UPF_GENERIC_ASYNC_SET_RESET** to be used in all conditions
- 9.7 simulation of retention section overhauled
  - Greatly enhanced examples with detailed waveforms for most common retention types

Example | Dual-Control SR

- *'Save'* & *'Restore'* conditions are on independent pins, giving more flexibility and micro-architecture management for retention strategies
- Advanced SR implementations as shown can now be realized through PI (UPF4.0)



Example | Dual-Control SR
async_set_reset_effect



a) Reset is ignored during the restore period

-async_set_reset_effect *ignored*

b) Reset affects the retained value and therefore the output value during the restore period

-async_set_reset_effect *retained_value*

c) Reset affects the output value during the restore period

-async_set_reset_effect *output_value*

Virtual Supply and Virtual Equivalence

Jeevan Medaramitta

Siemens EDA

accellera SYSTEMS INITIATIVE™

SIEMENS

2025 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION INDIA

Virtual Supplies & Equivalence

- Motivation
  - Pre-4.0, no way to model a supply net that did not physically exist in the design
    - No way to create driver/receiver supply to model external supplies
    - Required use of power models to create internal supplies for macros and use them in power states and strategy filters
  - Many tools already have ad-hoc methods to address these
- Solution
  - Define virtual *supply nets, ports* and *supply_sets*
    - Supply nets/ports/sets that are virtual have no physical implementation
    - Can be used in **add_power_state, connect_supply_net, *source/sink*** filtering, etc.
    - Have the same simulation semantics as non-virtual supplies
    - New concept of virtual equivalence – general concept is the same as electrical equivalence but without interchangeability

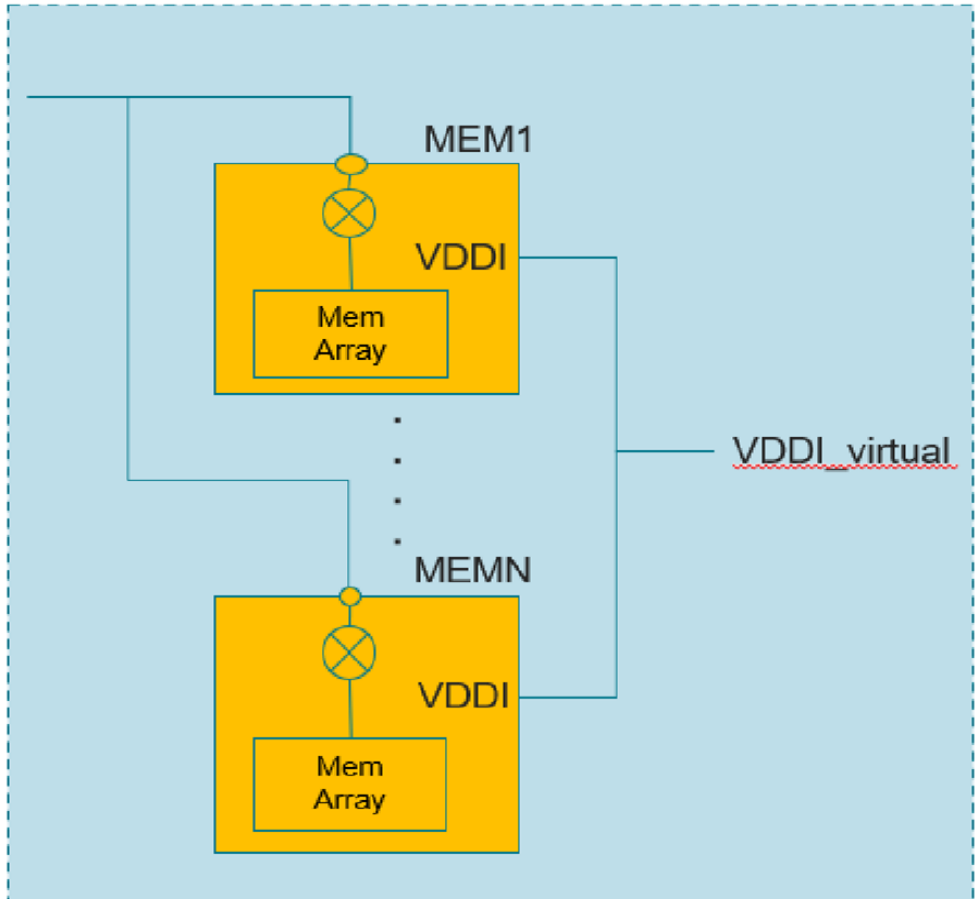
Virtual Supplies, Supply Sets and Ports

- Virtual supplies allow
  - to describe supplies that are not physically connected to the block
  - virtual connections between internal supplies of macros to setup equivalence
  - specification of driver and receiver supply of ports and macro pins to enable ***-source/-sink*** filtering
  - power states to be easily defined for cases where there is no physical supply net
- Virtual supply restrictions
  - Cannot be used to power any active logic in the design
  - Cannot be a primary supply of a domain, or used as the supply for any strategy
  - Cannot be written out in the physical design outputs
  - Connection of supply *subnets* does not create interchangeability
- Syntax changes

```
create_supply_net -virtual
create_supply_port -virtual
create_supply_set -virtual
```

Case1: Virtual Supply used to Model Functionally Equivalent Supplies

- Motivation
  - Simplify the power state and level shifting (LS)/isolation (ISO) strategies for macros
- Methodology
  - The internal supply (VDDI) pins are connected with a virtual supply net making them virtually equivalent
  - Allows creation of virtual supply sets that can be used for power states and ***-source/-sink*** ISO/LS strategies
  - Implementation tools are forbidden from using a virtual net to power logic
  - Implementation tools will not write these virtual connections into the output HDL (Verilog)



Virtual Supply UPF code

Define the supply net as virtual
Create a supply set using the virtual net

```
# Create the virtual supply net and virtual supply set
create_supply_net VDDI_virtual -virtual -resolve parallel
create_supply_set SS1_virtual -function {power VDDI_virtual} -function {ground VSS} -virtual
```

An internal pin, virtual connection - no physical connection

```
# Connect the virtual supply net to each memory's internal supply pin VVDDI
connect_supply_net VDDI_virtual -ports {MEM1/VDDI …. MEMN/VDDI}
```

Virtual supplies can have states specified in same way as physical

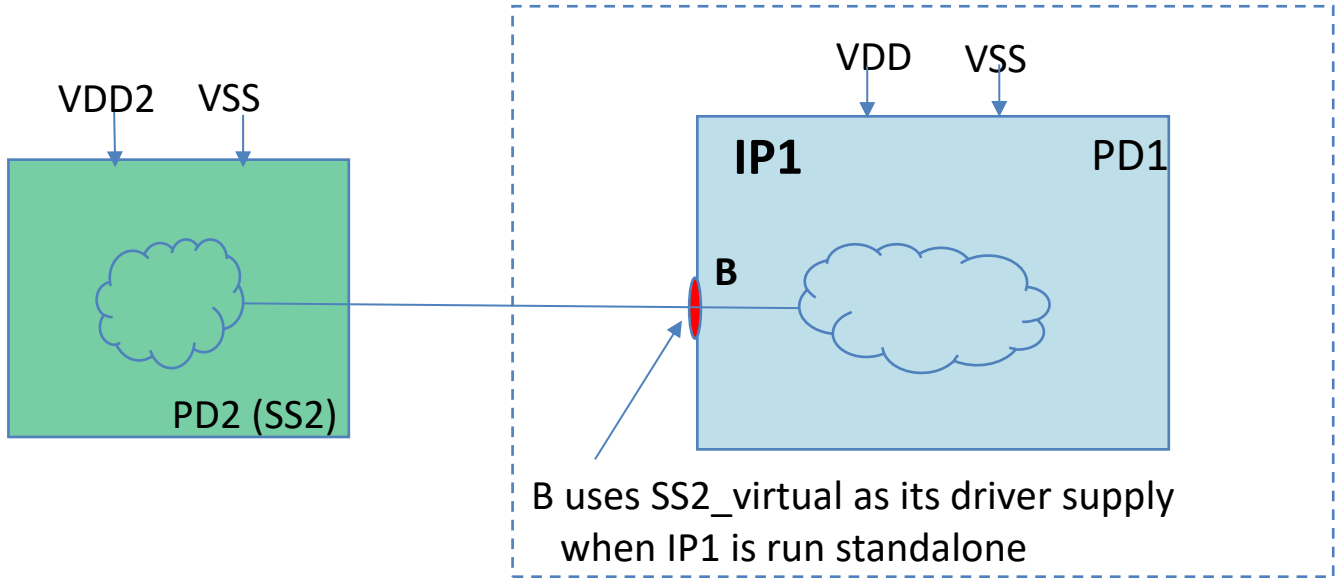
```
# A single supply level power state instead of one per MEM block
add_power_state SS1_virtual -supply  \
-state {OFF -supply_expr {power == OFF}} \
-state {ON -supply_expr {power == {FULL_ON} && ground == FULL_ON}}
```

```
# The logic expression has a single term, instead of one term per MEM block
add_power_state PD1 -domain -state {ON -logic_expr {SS1_virtual == ON}}
```

Virtual supplies can be used as source/sink for ISO/LS strategies

```
# A single set_isolation covers any output driven by any of the connected MEM blocks
set_isolation ISO1 -domain PD1 -source SS1_virtual -applies_to outputs
```

Case2: Virtual Supply to Model External Supplies

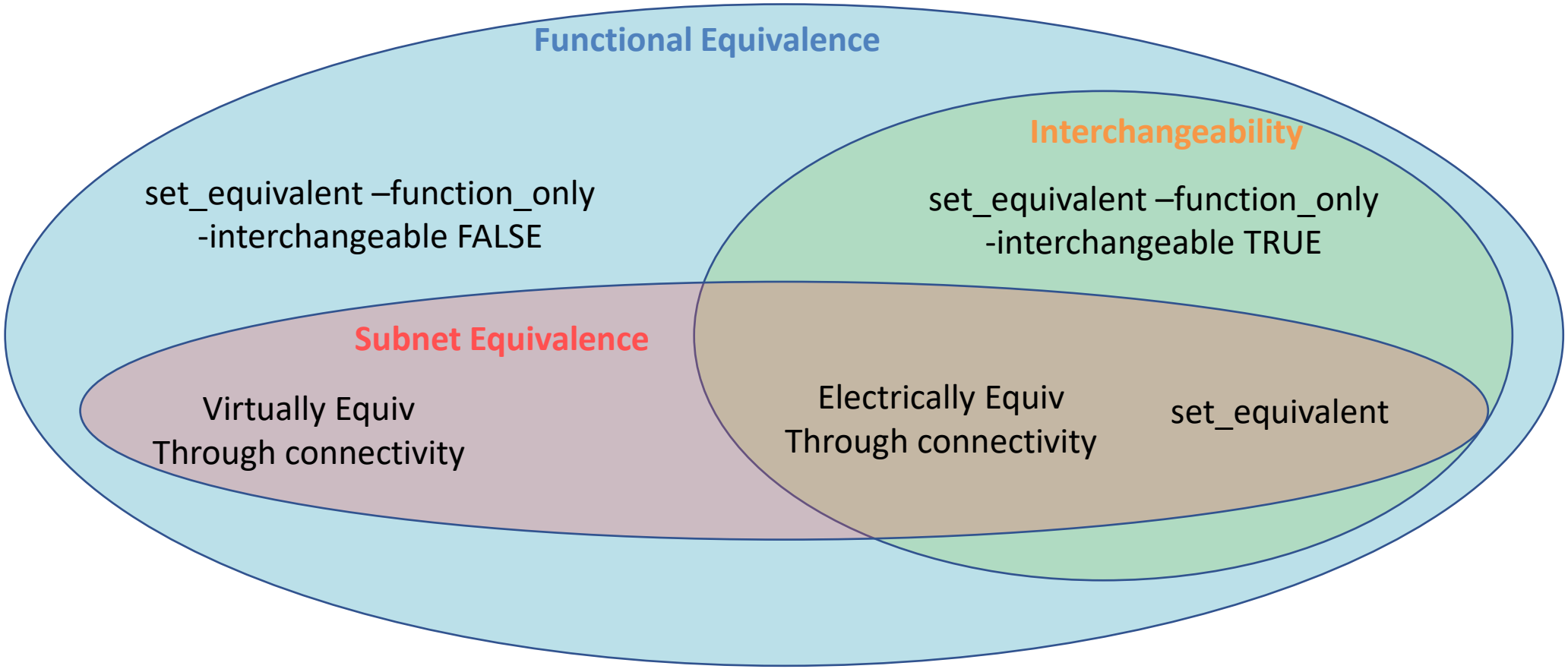



```
create_supply_net VDD2_virtual -virtual
create_supply_set SS2_virtual -virtual
      -function {power VDD2_virtual}
     -function {ground VSS}

set_port_attribute -ports B
       -driver_supply SS2_virtual

set_isolation iso1 -domain PD1
        -source SS2_virtual
        -applies_to inputs
```

Virtual Equivalence

- Prior to 4.0, any supply nets connected to each other were electrically equivalent
  - Subnet equivalent: Any drivers on any of the connected supply nets had to be treated as one net and had to be resolved in simulation
  - The nets were interchangeable: Nets could be used interchangeably including in physical design
- Virtual Nets are not real connections, they are not interchangeable
- Virtual Equivalence
  - Keeps the subnet equivalence for simulation, but does not include interchangeability
  - Affects transitive properties
    - *A(real)* connects to *B(real)*, and *B(real)* connects to *C(real)*; Then A and C are **electrically equivalent**
    - *A(real)* connects to *B(virtual)* and *B(virtual)* connects to *C(real)*; then A and C are only **virtually equivalent** and **are non-interchangeable**

Understanding Equivalence



General Updates & Beyond 1801-2024 (UPF 4.0)

Jeevan Medaramitta

Siemens EDA

accellera SYSTEMS INITIATIVE™

SIEMENS

2025 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION INDIA

Details on Select Topics

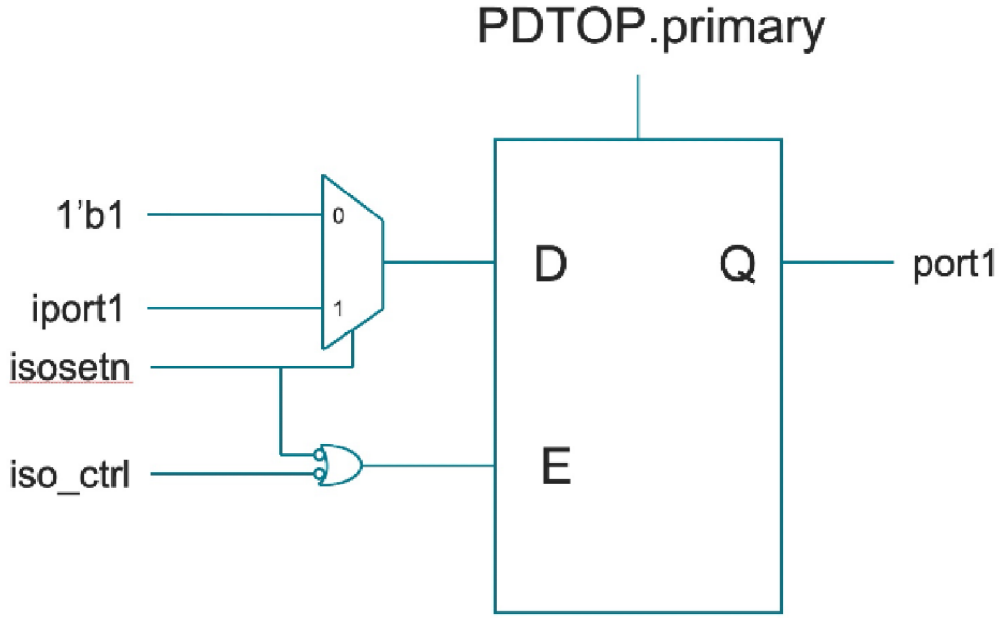
- Support for set/reset on Latch Isolation
- **`map_retention_clamp_cell`**
- **`find_objects -expand_to_bits`**
- Precedence Updates
- **`set_port_attributes -feedthrough`**
- Naming Updates

Support for **set/reset** on Latch Isolation

- Support for set/reset on Latch based isolation

```
set_isolation
    -async_set_reset {net_name <high|low>} #specify
    cntrl signal
    -async_clamp_value  <0|1>              #specify
    set or reset
```

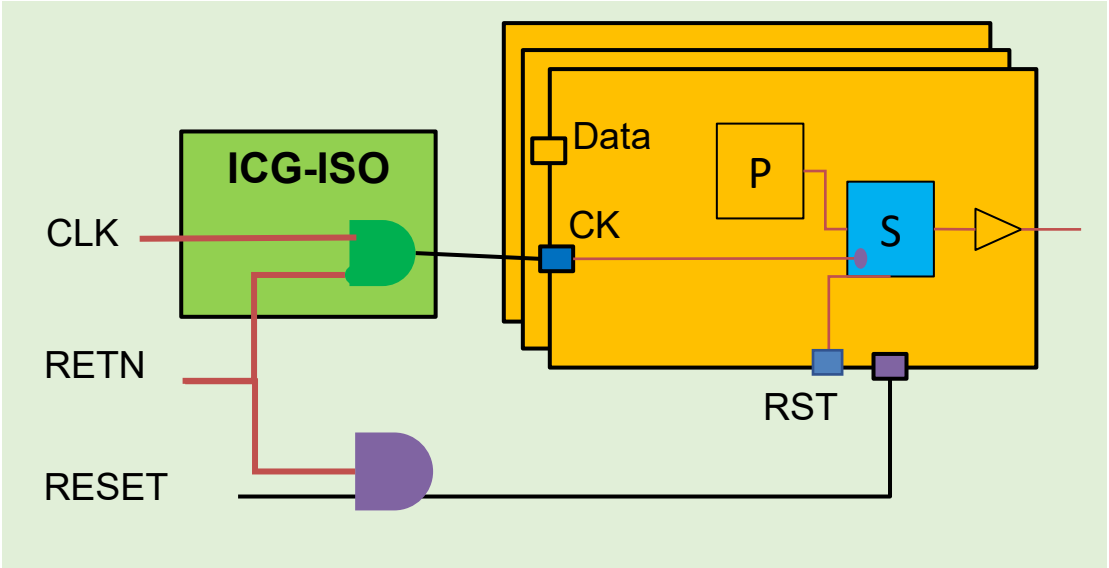
- Create a new port attribute to specify async_clamp_value
  - **UPF_async_clamp_value**
  - **set_port_attributes -async_clamp_value** <0|1>




```
set_isolation ISO -domain
-isolation_signal iso_ctrl
-isolation_sense high
-clamp_value latch
-async_set_reset { isosetn
low}
-async_clamp_value 1
. . .
```

map_retention_clamp_cell

- For “zero-pin” state retention a clamp cell is automatically inserted on the *clock/reset* pins
- **map_retention_clamp_cell** allows the specification of what cell to use to implement that clamp



- Example:

```
set_retention RET1 -domain PD1 ...
     -save_signal {RETN high}  -restore_signal {RETN low}
map_retention_cell RET1 -domain PD1 -lib_cells {SCL9T_ZPR_X2}
map_retention_clamp_cell {RET1} -domain PD1
    -clock_clamp_lib_cells {SCL9T_ISOT1_X1}    # green isolation in diagram
    -async_clamp_lib_cells {SCL9T_ISOT2_X1}    # orange isolation in diagram
```

find_objects –*expand_to_bits*

- **find_objects** can return a single object for a bus or a list of individual bits
  - Possible before by using patterns like “xyz\[*\]” to return list of individual bits
- In UPF 4.0, this process made easier by adding an “**`-expand_to_bits`**” option
  - When set *true*, the individual bits will always be returned
  - When *false* (or not set), the pre-4.0 behavior will apply
- The outcome can affect the precedence of this list in other commands
  - Example: **set_isolation** *ISO1* **–elements [find_objects ….]**
  - In the elements list, bits will have higher precedence than the full bus

Find_objects . –object_type port	**Return Value**
`-pattern {pmda\[*\]}`	`{pmda[1] pmda[0]}`
`-pattern {pmda\[*\]} -expand_to_bits`	`{pmda[1][0] pmda[1][1] pmda[0][1] pmda[0][0]}`

Examples of Precedence Updates

- **`set_retention`**
  - **`set_retention`** with **`–no_retention`** now has precedence over **`set_retention`** without **`–no_retention`**
- **`set_port_attributes –driver_supply/-receiver_supply`**
  - If after applying the precedence rules above, the predefined port attributes **UPF_receiver_supply** or **UPF_driver_supply** are defined on a given port using both hierarchical and non-hierarchical names then the hierarchical name shall take precedence
- Composite types
  - When determining precedence, composite data types are treated as a multibit signal
  - A record field or array index of a composite data type referred to explicitly by name is also treated as a part of the multibit signal
- **`set_design_attributes -is_hard_macro|-is_soft_macro|-is_refinable_macro`**
  - If the macro has multiple ***-is_*_macro*** attributes set, then **`-is_hard_macro`** has highest precedence, followed by **`-is_soft_macro`**

set_port_attributes *-feedthrough*

- In 3.1, the semantics around multiple feedthrough groups was unclear.
- In 4.0:
  - **set_port_attributes -ports** {*port_list*} **-feedthrough** *feedthrough_name*
  - All ports connected with the UPF_feedthrough attribute set to the same feedthrough name, are defined as connected
- Example:
  - The following code defines two separate feedthrough groups: X that includes a, b and c, and Y that only contains e and f.

```
set_port_attributes -ports {a b} -feedthrough X
set_port_attributes -ports {c} -feedthrough X
set_port_attributes -ports {e f} -feedthrough Y
```

Naming Updates

- Clarify what character should be used in UPF to specify the generate block delimiter
  - In the design flow generate blocks are unique
  - Creates a hierarchy for simulation, but not for implementation
  - The naming style also can differ based on tools settings
  - UPF 4.0 allows to define a single style for entire UPF
  - **generate block delimiter character:** A special character used in composing names containing generate block labels
  - The generate block delimiter character is a dot (.)
- New Library naming
  - When referencing a model in a command argument, its name may be prefixed by its library name followed by a dot (".")
  - Limits the effect of a command to the particular version of that model compiled into the specified library
  - A model name specified with the "<library>.<model>" syntax is considered a simple name (5.3.3.2)

Beyond 1801-2024

- 1801-2024 major features are in response to new technologies and design methodologies
- Beyond 1801-2024
  - Continued innovation on mixed-signal design and interfacing
    - Enable supply networks to carry information about power generation and consumption
    - Bi-directional supply ports
    - Features to improve static checking of designs with mixed analog/digital components
    - Power modeling in the context of mixed-signal integrations
  - Information model improvements beyond SV and VHDL
  - Improvements to support new technology cells
  - Ease of use/Ease of specification improvements

Refererences

- Vijayakumar Sankaran, et al, Modern Recipes for Brewing the Inevitable Methodology for Today's ICs: Low-Power Mixed-Signal Design Verification, Tutorial, DAC 2019.
- Daniel Cross, "Applications for UPF HDL Supply Tunneling in Mixed Signal Design," DVCON 2025.
- Amit Srivastava, Lakshmanan Balasubramanian, John Decker, "Refinable Macros and Terminal Boundaries in UPF 4.0: Empowering Soft IPs of the Future," DVCON 2025.
- Lakshmanan Balasubramanian, et al, "Future Proofing Power Intent Specification through Unified Power Format 4.0 for Evolving Advanced State Retention Strategies," DVCON 2025.
- Lakshmanan B, et al, A holistic approach to low power mixed-signal design verification using power intent: CPF-Based Interface Elements, Methods, and Guidelines, DVCON 2016.
- Lakshmanan Balasubramanian, et al, Advanced Low Power Retention Simulation Framework, Designer Track, DAC 2019.
- Progyna Khonkdar, "What's New in IEEE 1801 and Why you Need to Know Now?," DVCON 2025.

Questions

Thanks for kind attention, time, interest and interaction