2018 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

Comprehensive Metrics-Based Methodology to Achieve Low-Power System-on-Chips

Ellie Burns – Director of Marketing for Calypto Systems Division
Gabriel Chidolue – Verification Technologist
Guillaume Boillet – Power Product Specialist

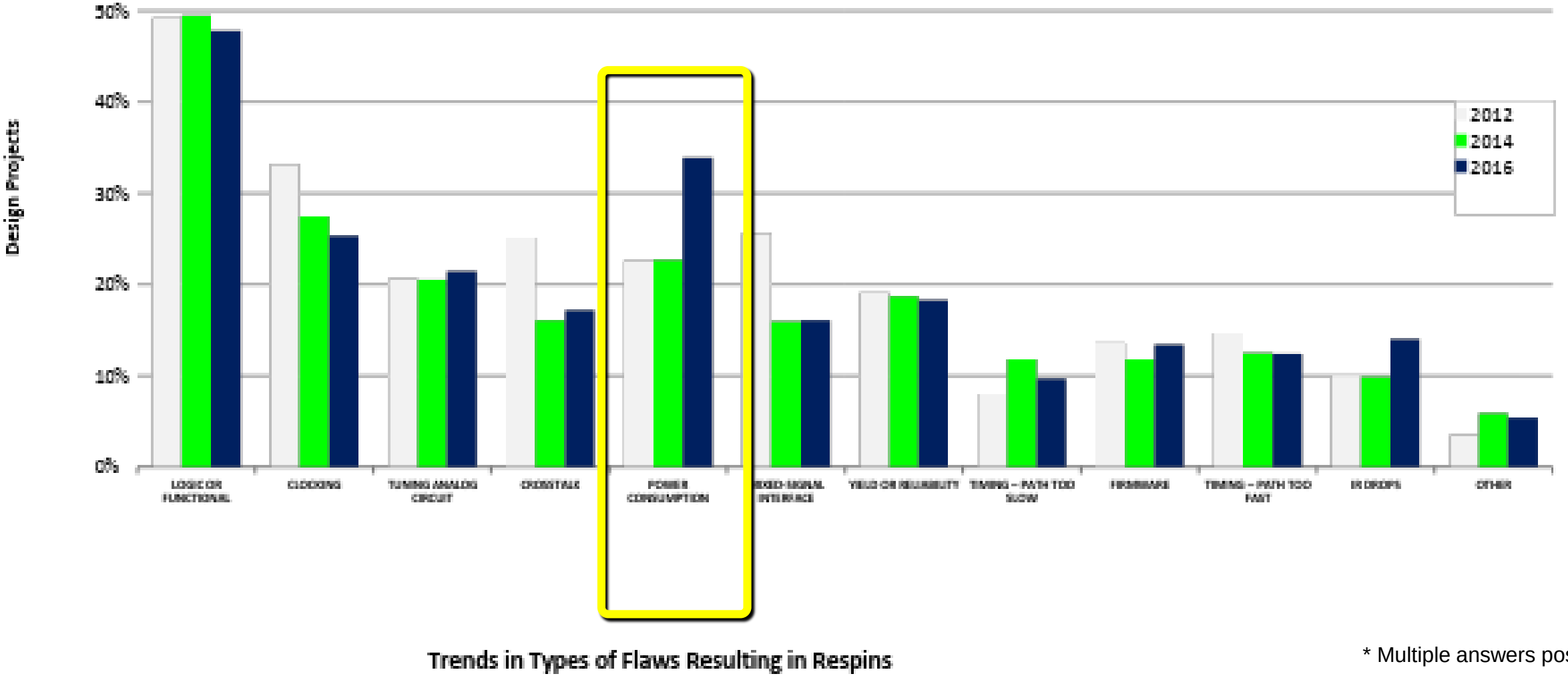
Mentor®
A Siemens Business

2018 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

Welcome and Introduction

2018 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

Power is Now the #2 Driver of Respins!



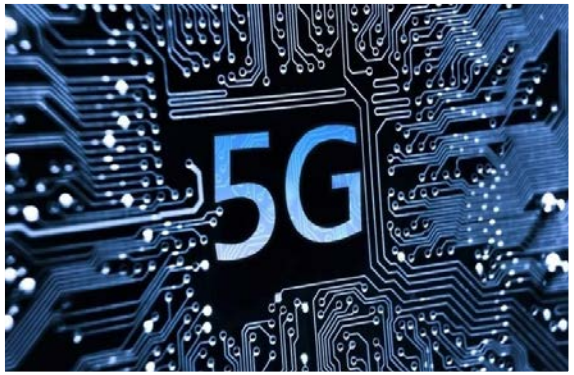
Trends in Types of Flaws Resulting in Respins

* Multiple answers possible

Source: Wilson Research Group and Mentor Graphics, 2016 Functional Verification Study

Markets Like 5G, Smart IoT, Automotive Low-Power Critical

- 5G both ultra low-power and high-performance mobile

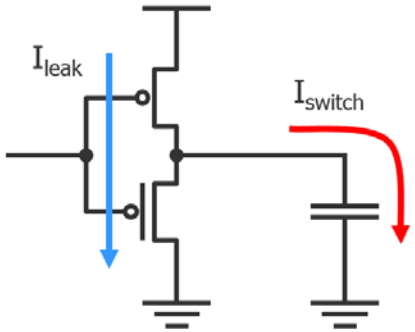


- IoT expanding digital content and changing landscape

- Path to Autonomous vehicles requires huge computations and power

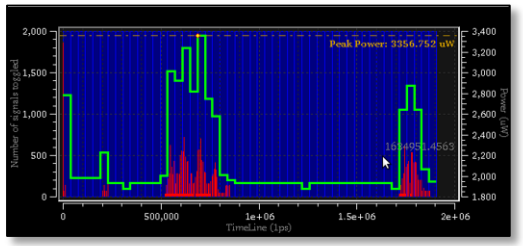
Dimensions of Low-Power Design

Leakage Power
Power consumed by current flow through transistors even when turned off



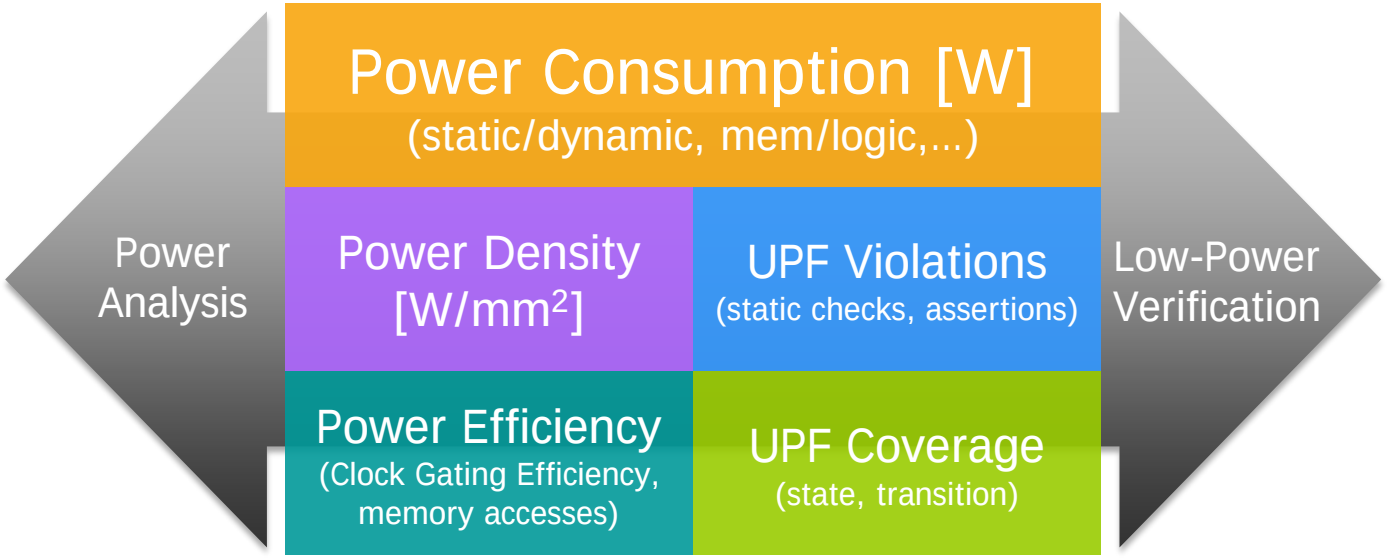
Dynamic Power
Power dissipated when circuit is active, charging and discharging the loads

Power Estimation (RTL/Gate)
Measuring power consumption of a design. Typically used to know if the design is within specified power budget



Power Reduction
Process of optimizing a design to reduce its power consumption. Necessary when the design is over budget or when competing on power

Important Metrics for Successful Low-Power Design and Verification



- There are many kinds of metrics for power that should be measured at various times in the design cycle

Lifecycle of Low-Power Design and Verification

Early RTL Power Metrics with no vectors

RTL/GL Metrics with vectors and power vectors

Power Management coverage metrics

SoC Power Management coverage metrics

Power Metrics for full SoC with real power scenarios

Measure SoC power and benchmark/SoC scenarios

Design/Test/Measure Power management

Measure power block/sub-system tests scenarios

RTL Designers
Use low-power design techniques

Top Level

Sub-System

Block Level

RTL Design and Verification Timeline

2018 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

Tutorial Agenda

- Introduction
- Introduction of Low-Power Coverage & Debug metrics in simulation
- Update on UPF 3.0 and 3.1
- Break
- Analysis and Reduction: Metrics for Designing low-power IP
- Emulation-Based Power Analysis and Verification
- Wrap up

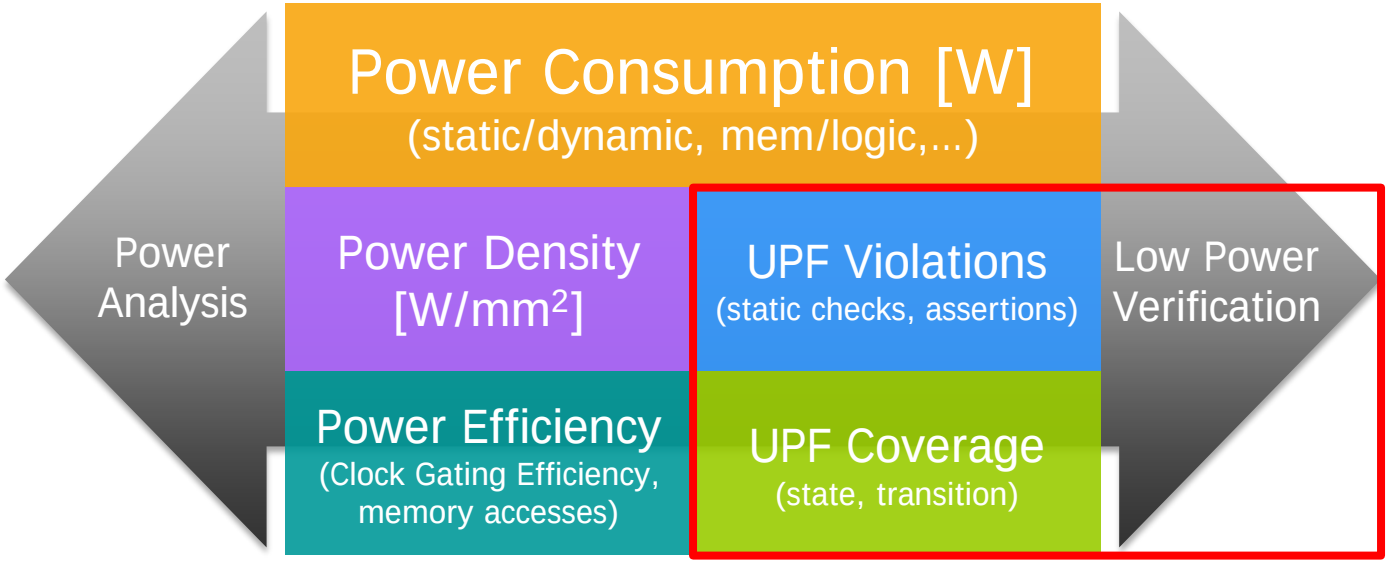
2018 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

Low-Power Verification Metrics (UPF Power Architecture)

Gabriel Chidolue, Verification Technologist

Mentor®
A Siemens Business

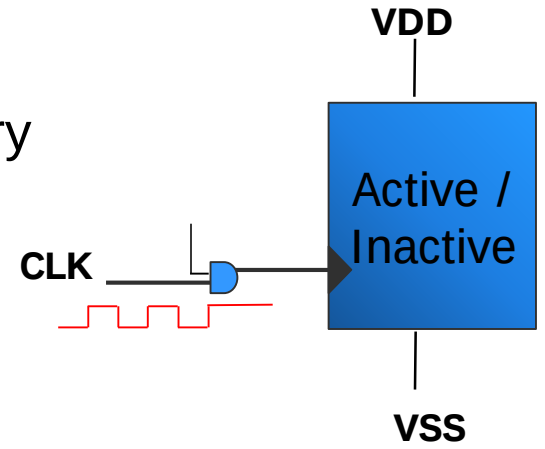
Metrics for UPF Based Designs



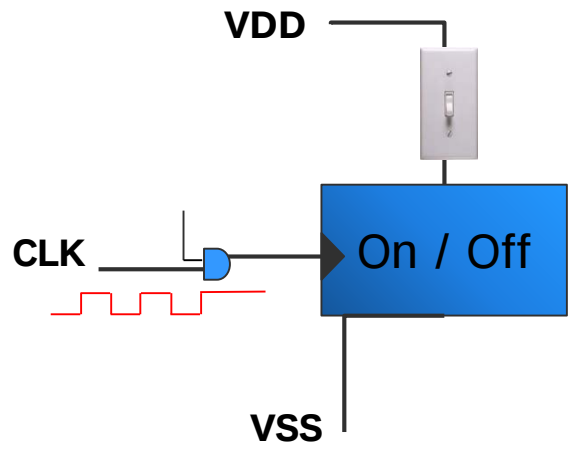
- Leakage power management requires active power reduction techniques

Power Reduction Techniques

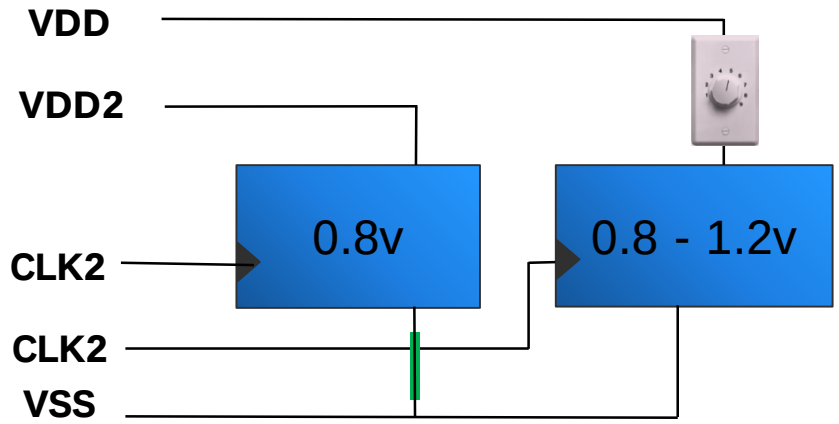
- Clock Gating
  - Avoid unnecessary switching activity



- Power Gating
  - Shut off when logic is not in use

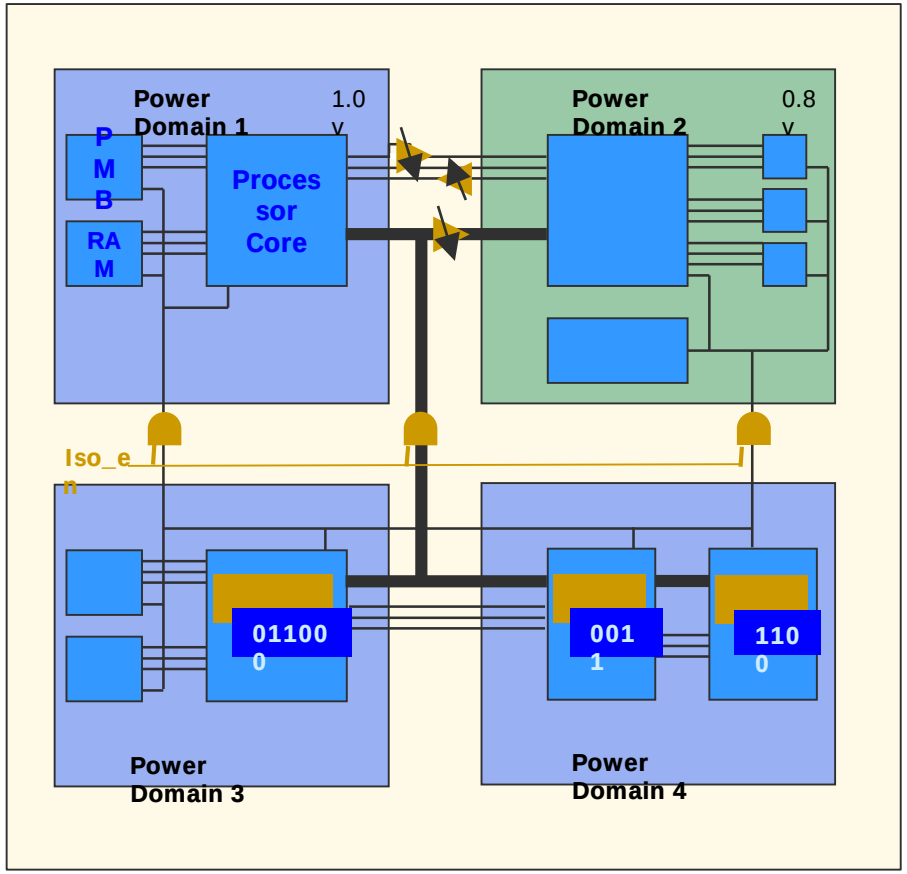


- Various Multi-Voltage Techniques
  - Optimize power used for required performance
  - Dynamically adjust voltage/frequency
  - Non-operational back biasing to reduce leakage



Power Management Concepts

- Power Domains
  - Independently powered regions
  - Apply different power reduction technique in each region
- State Retention
  - Save essential data when power is off
  - To enable quick resumption after power up
- Isolation
  - To ensure correct electrical and logical interactions between domains in different power states
- Level shifting
  - To ensure correct communication between domains operating at different voltage levels



Introducing IEEE1801 (Unified Power Format)

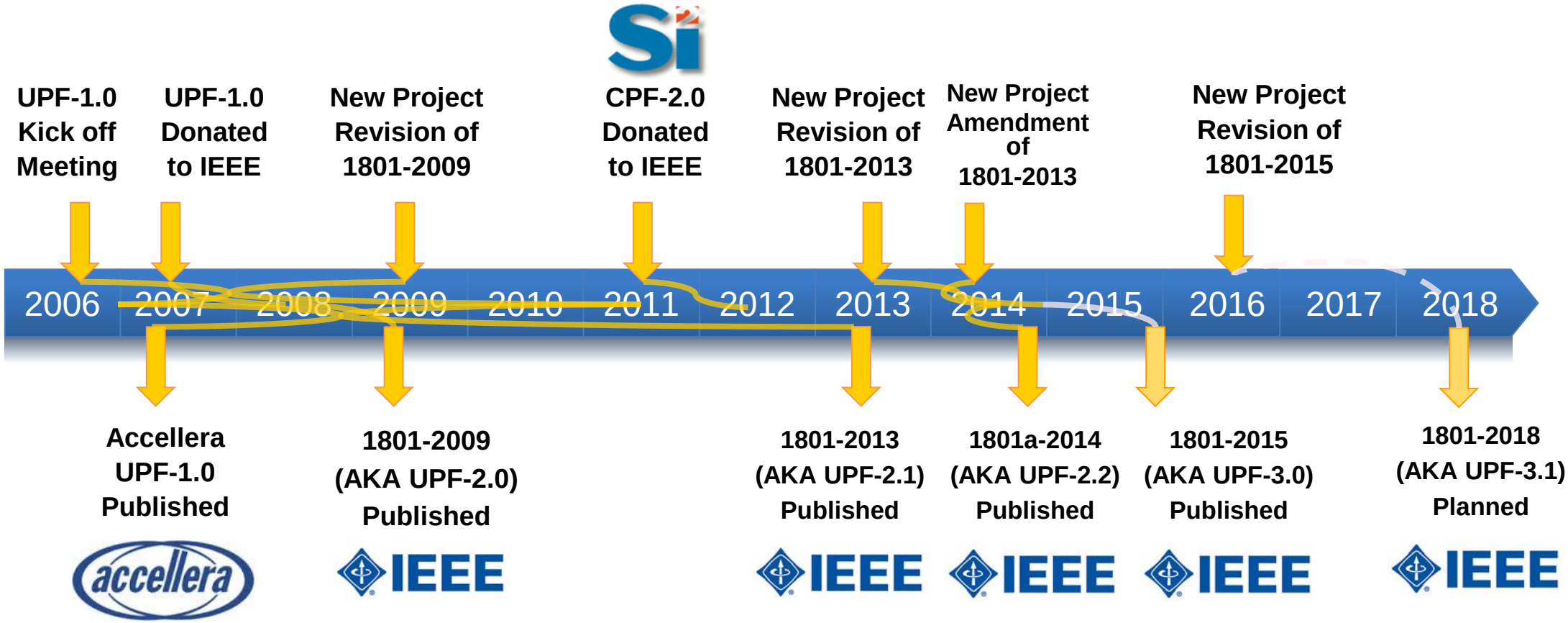
- **IEEE 1801 UPF**
  - Standard format for defining power management
  - Extension of “Tcl” language
  - Defined separately from the HDL
  - Enables early verification of power intent
  - Drives verification and implementation from RTL to Layout
  - Contains commands for power intent and driving testbenches

- **An Evolving Standard**
  - Accellera UPF in 2007 (1.0)
  - IEEE 1801-2009 UPF (2.0)
  - IEEE 1801-2013 UPF (2.1)
  - IEEE 1801a-2014 UPF (2.2)
  - IEEE 1801-2015 UPF (3.0)

- **For Power Intent**
  - To define power management
  - To optimize power consumption
- **For Power Analysis (in 3.0)**
  - Component Power Modeling

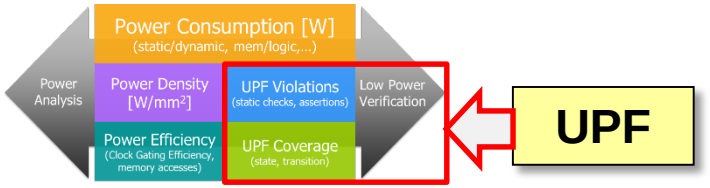
IEEE Std 1801™-2015
UPF

IEEE1801 is an Evolving Standard

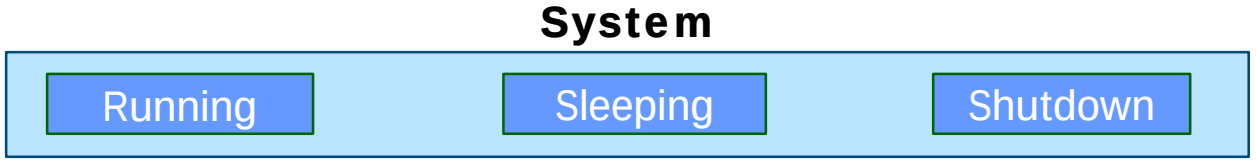


Power State Definition

UPF 3.0



- A key part of the power architecture specification of a design
- Representing
  - Operational Modes of IP within system as well as System
  - Ability for Supply states of supply_sets
- Specified on Power domains, supply sets, groups and other objects
- Specified using **`add_power_state`** command
  - **`-logic_expr`**: functional mode (control signals, power states)
    - **`Interval()`**: clock frequency
  - **`-supply_expr`**: supply set function state and voltage and simstate
- With state space controlled using
  - **`-complete`** to indicate that specified states are the only legal power states
    - All unspecified power states are illegal
  - **`-illegal`**|**`-legal`** to mark specific states as illegal states
    - Default is Legal

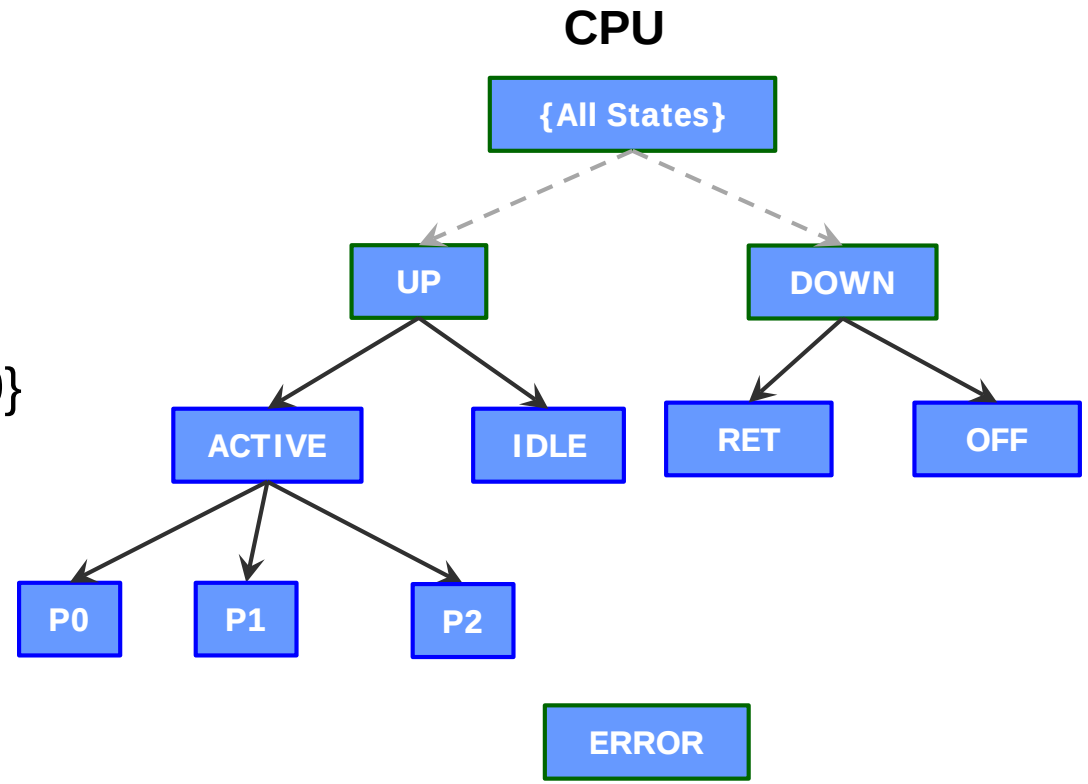



```
add_power_state –domain System \
  -state {Running  –logic_expr {...}}
  -state {Sleeping -logic_expr {...}}
  -state {Shutdown –logic_expr {...}}
```

-logic_expr defines condition in which object enters the state

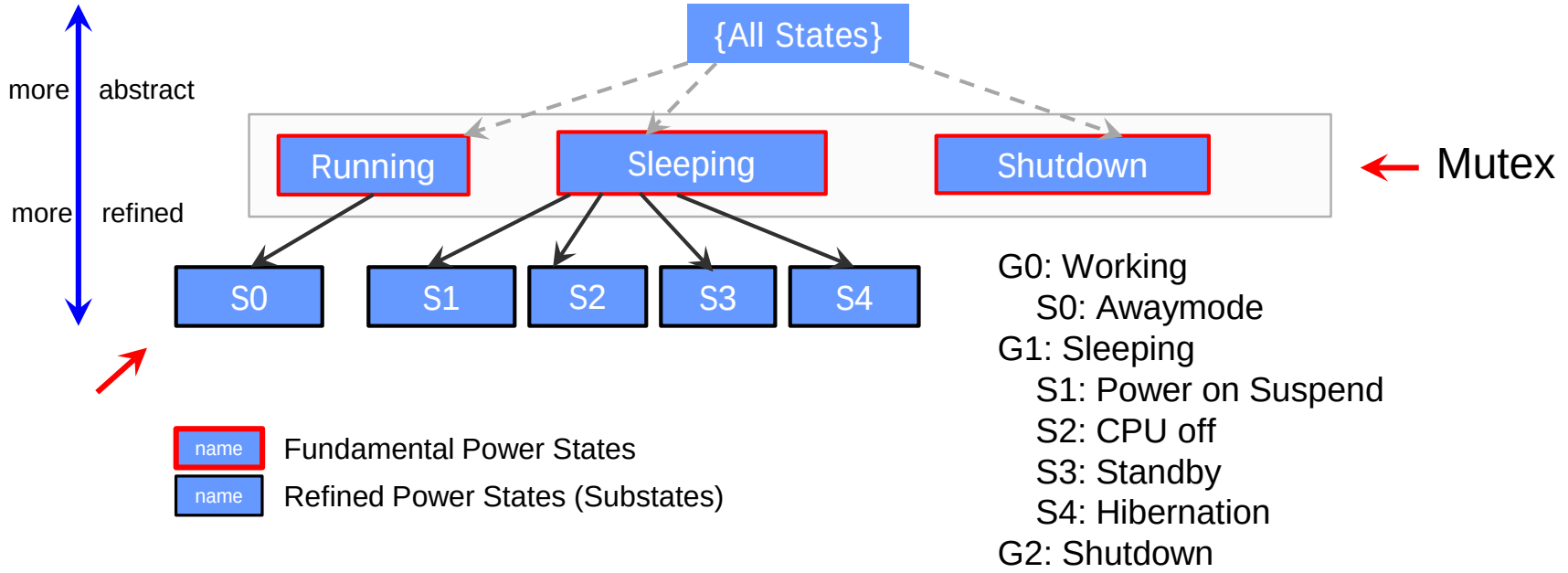
Abstract/Refined Power States

- Create a more refined state of an object by referring to another state of the **same object**
  - UP : -logic_expr {pwron}
    is a fundamental state
  - UP.ACTIVE : {CPU == **UP** && running}
    is a refinement of UP
  - UP.ACTIVE.P0 : {CPU ==**ACTIVE** && perflvl==2'b0}
    is a refinement of UP.ACTIVE
- The "{All States}" state shown represents all possible states of the CPU



Mutual Exclusivity of Power States

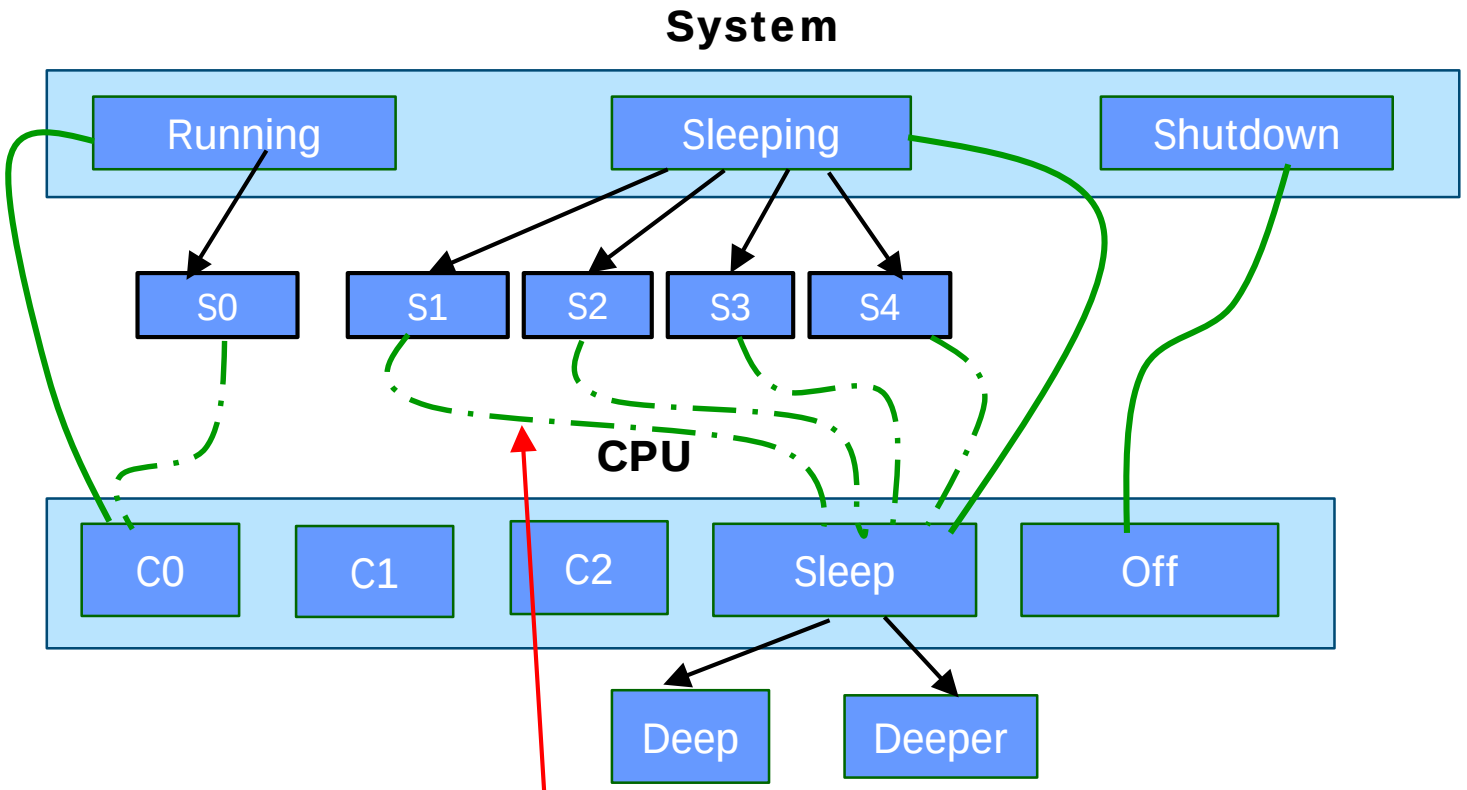
- States at same level of refinement are mutually exclusive
- Abstract states contain (overlap) sub-states – non Mutex
- Error if more than one mutually exclusive state becomes active



Hierarchical Composition of Power states

System States

```
add_power_state system
  -state {Running  -logic_expr { CPU == C0}}

add_power_state system
 -state {Sleeping –logic_expr {CPU == Sleep}}

add_power_state system
  -state {Shutdown –logic_expr {CPU == Off}}
```

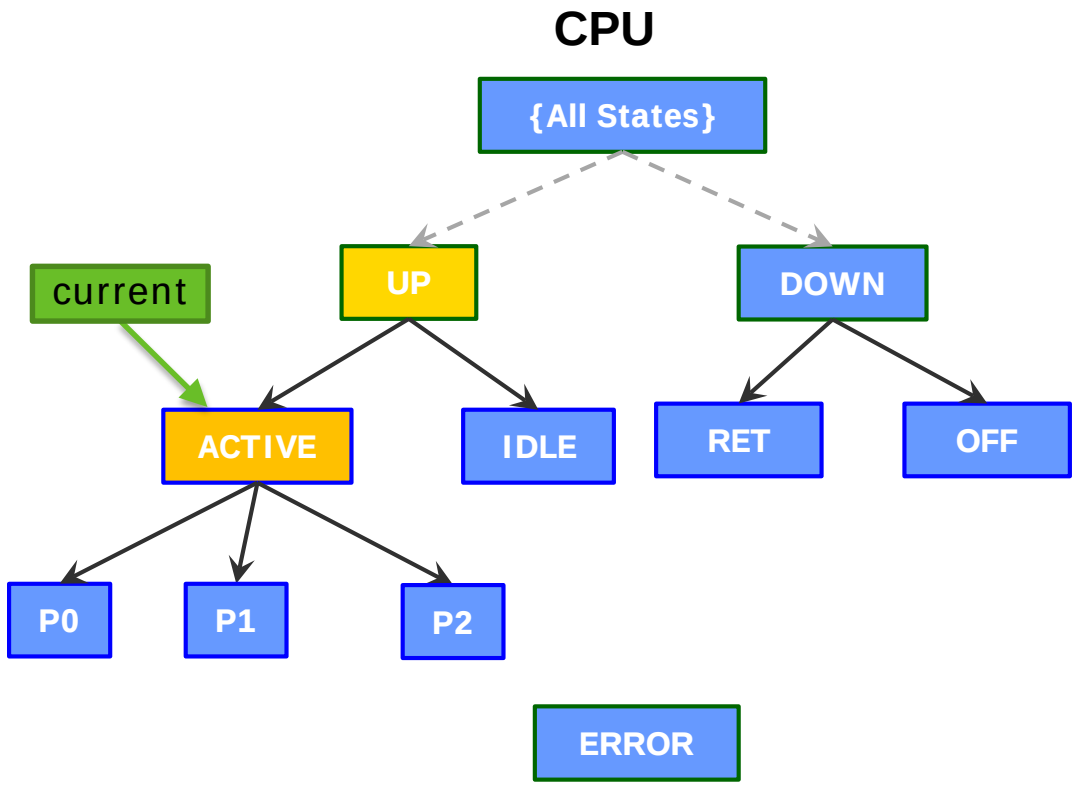


System Power States are defined in terms of the states of the dependent Power domain states.

A Substate inherits dependencies of the State it refines

Current vs Active State Semantics

- Active vs Current State semantics
  - A state is *active* if its logic expression is satisfied
  - A state is the *current* state if it is the most refined *active* state
    - ACTIVE and UP are *active* states
    - ACTIVE is the *current* state (since it is the most refined state that is activated)
- Predefined state ERROR is activated if 2 or more mutually exclusive states are activated at the same time



Power State Transitions

- Defined using
  - **`add_state_transition`** command
- States can be group and/or paired to aid in specification
- Identify legal and illegal transitions between power states using
  - **`-illegal, –legal(default)`**
- Control the possible transition space using
  - **`-complete`**
- Essential for transition coverage

```
add_state_transition –domain System \
  -from Sleeping -to {Running Shutdown}

add_state_transition –domain System \
 -from Running –to Sleeping

add_state_transition –domain System \
  -from Running –to Shutdown -illegal
```

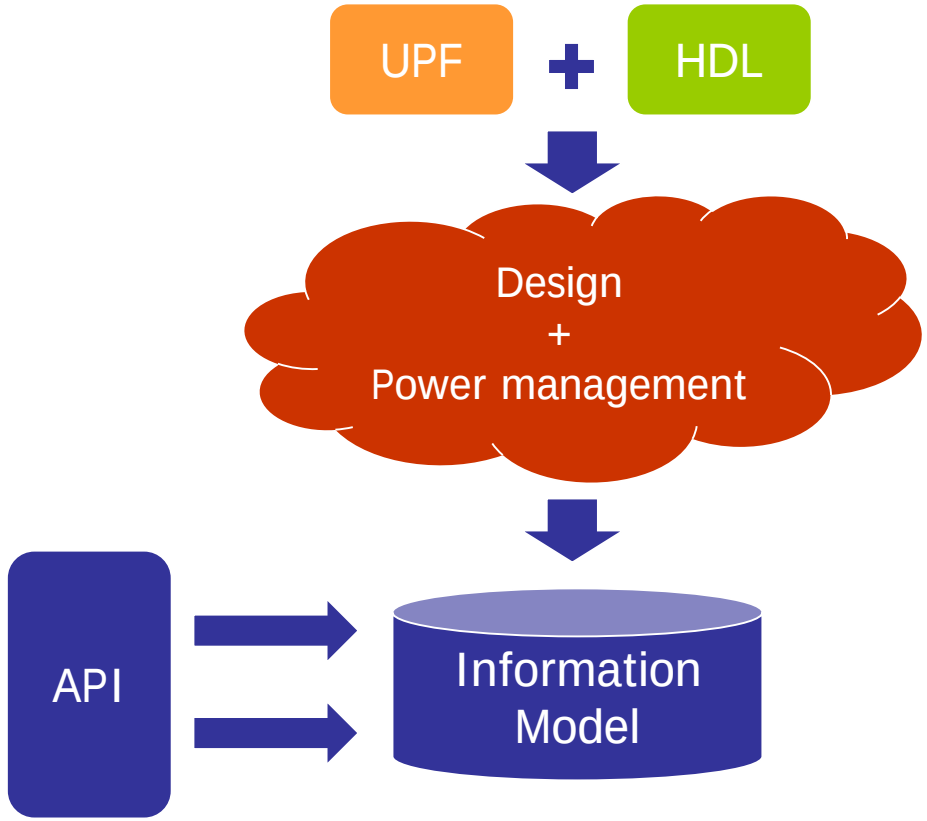
UPF Power State Definition in Verification

- Enables Static analysis tools
  - To determine isolation and level shifting requirements at Power Domain and Macro boundaries
    - Aided by hierarchically composed power states in UPF
- Enables Dynamic verification tools
  - To create test plans and
  - To create comprehensive coverage models of all legal system and subsystem power states and transitions
    - Aided by all power state specifications in UPF
  - Helps answer the question:
    - Have we verified correct operation of all legal power states of the system ?
    - Have we verified correct operation while system transitions between all legal power state transition combinations?

Other Key UPF 3.0 Feature

UPF PA Information Model

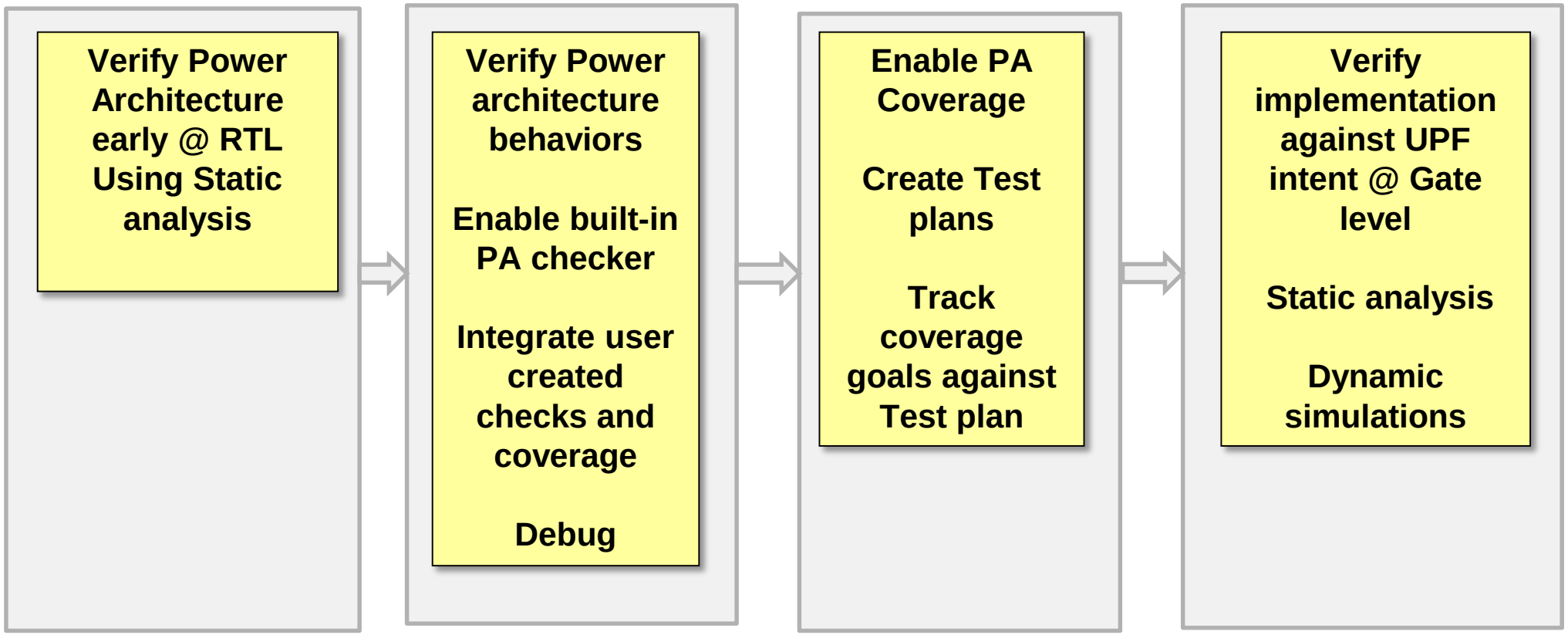
- UPF Power Aware Information Model
  - Database containing processed UPF Objects, properties and relationships with HDL
  - Native types and Access methods in HDL and TCL
    - UPF package
  - HDL Access methods can access dynamic information
    - E.g. current Power state of a Power domain, current voltage value on supply net
- Users can build
  - power aware coverage model
  - Power aware assertions
  - Power aware constraint random test benches



What's Coming in UPF 3.1

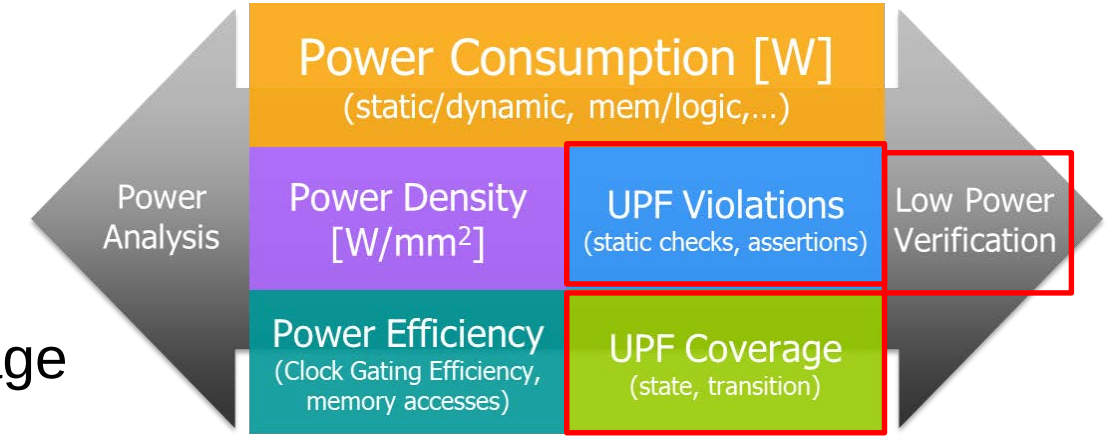
- New simulation control commands
  - Suspend non-PA user assertions dynamically at power down
- Ease of use enhancements to Power states
  - Create power state groups of supply sets
  - Use named states on supply objects created using add_supply_set in -supply_expr of add_power_state

Power Architecture Verification Flow

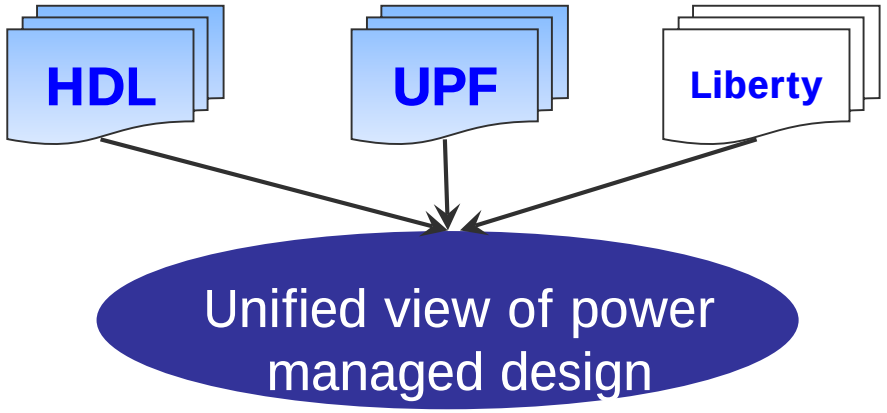


Verification Strategy for Power Managed Designs

- Static Analysis\Checking
  - Power Architecture Consistency at RTL
  - Structural Checks and implementation consistency checks post synthesis
- Automated Dynamic Power Aware checkers
  - To find common power control sequence protocol bugs
- User Created Power aware assertions and coverage
  - Build power aware coverage and checkers using UPF Information Model and TCL and HDL API functions
  - Bind to DUT (UPF + RTL) using UPF bind_checker
- Leverage Unified Power Aware Debug Environment
  - Visibility
  - Root cause Analysis
- Verification Completeness Metrics using Power Aware (PA) Coverage



Verify Power Management Architecture Using Static Analysis



IEEE Std 1801™-2015

- Static Check Analysis with Questa PA Sim
  - No Testbench required
  - RTL and Gate Level HDL support
  - Global analysis of all Power States
- Extended Static Checking Questa CDC (Power Aware)

Static Checking

Essential Checks at Different Design Abstraction Levels

RTL Checks

- Power architecture checks
  - ELS/LS /ISO are inserted where required
  - NO missing, redundant or back to back ELS/LS/ISO cells
- Information from add_power_state is checked against strategies in UPF during this checking
  - UPF (1.0) PST also supported
- Does not require test vectors

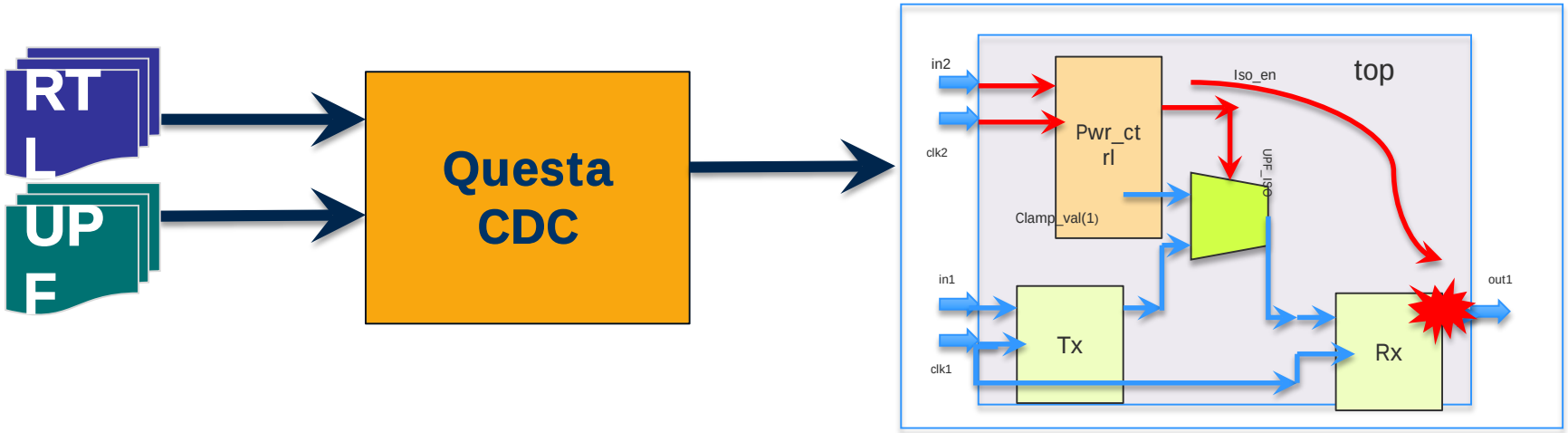
Gate Level Checks

- Micro architectural checks
  - Ensure control signals are driven from relatively always_on domain
    - Control signals are determined from UPF strategies and switch specifications
- Implementation checks
  - Check implemented ISO/LS/ELS/RFF/ PSW and AON buffer cells against UPF specifications (Strategies and power states)
    - Requires Liberty
    - E.g Correct polarity of clamp value, wrong cell type used , no back to back cells, etc

Extended Static Checks

with Power Aware CDC

- Power control signals unconnected in RTL
- Unified Power Format (UPF) defines power intent
- Power management structures from UPF could introduce CDC paths
- Verify these at the RTL level

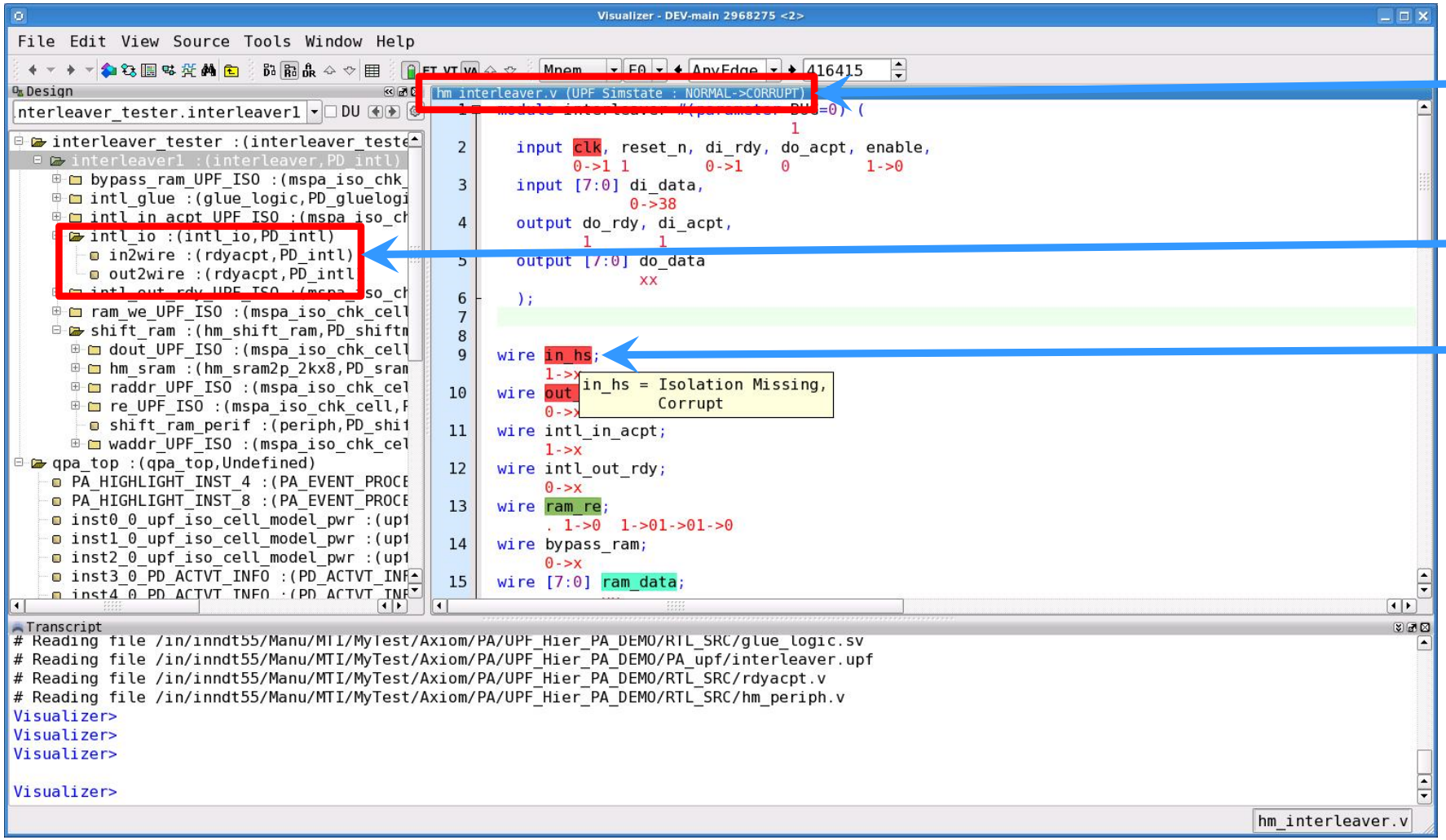


Verify Dynamic Power Aware Behaviors

- Verify power up/down Sequencing for each block/power Domain
  - nonoperation biasing, save operation on power down
  - reset/restore on power up
- Verify that interfaces are correctly isolated and level shifted
- Verify that appropriate retention protocols are followed
  - 0/1 pin retention FF (master slave alive style retention)
  - 1/2 pin edge sensitive Ballon latch style Retention FF
  - Non retention FFs may require resetting on power up
- Leverage Questa PA Dynamic checkers (assertions)
  - To check for common power management protocol violations
- Verify that all power states and transitions are covered
  - Enable automatic PA coverage collection

Visualize UPF objects in Design Hierarchy

Synchronized with Source



Current status of power

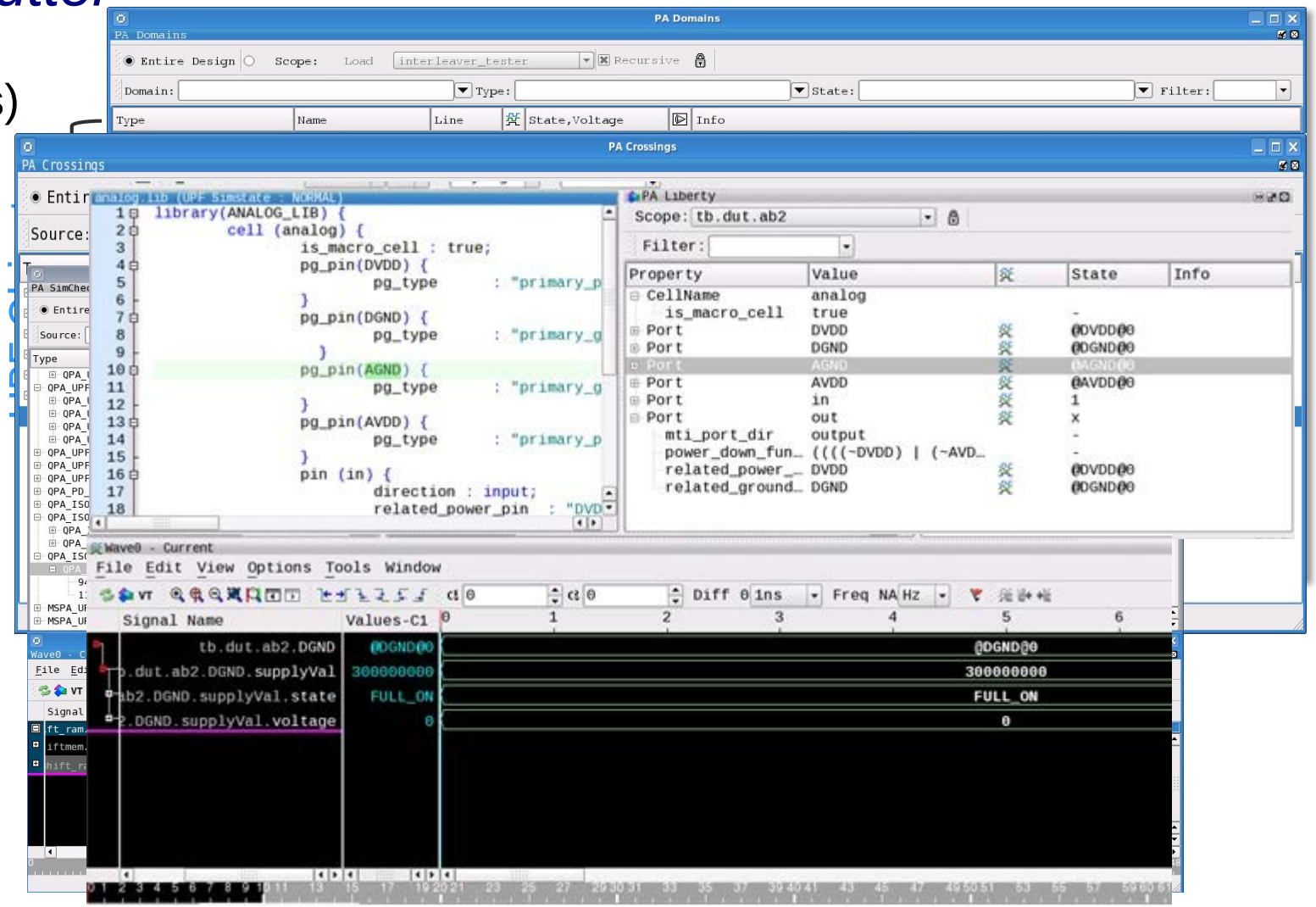
Power Domain information

Color coded to display PA violation

Visualize UPF in Context

Without HDL Clutter

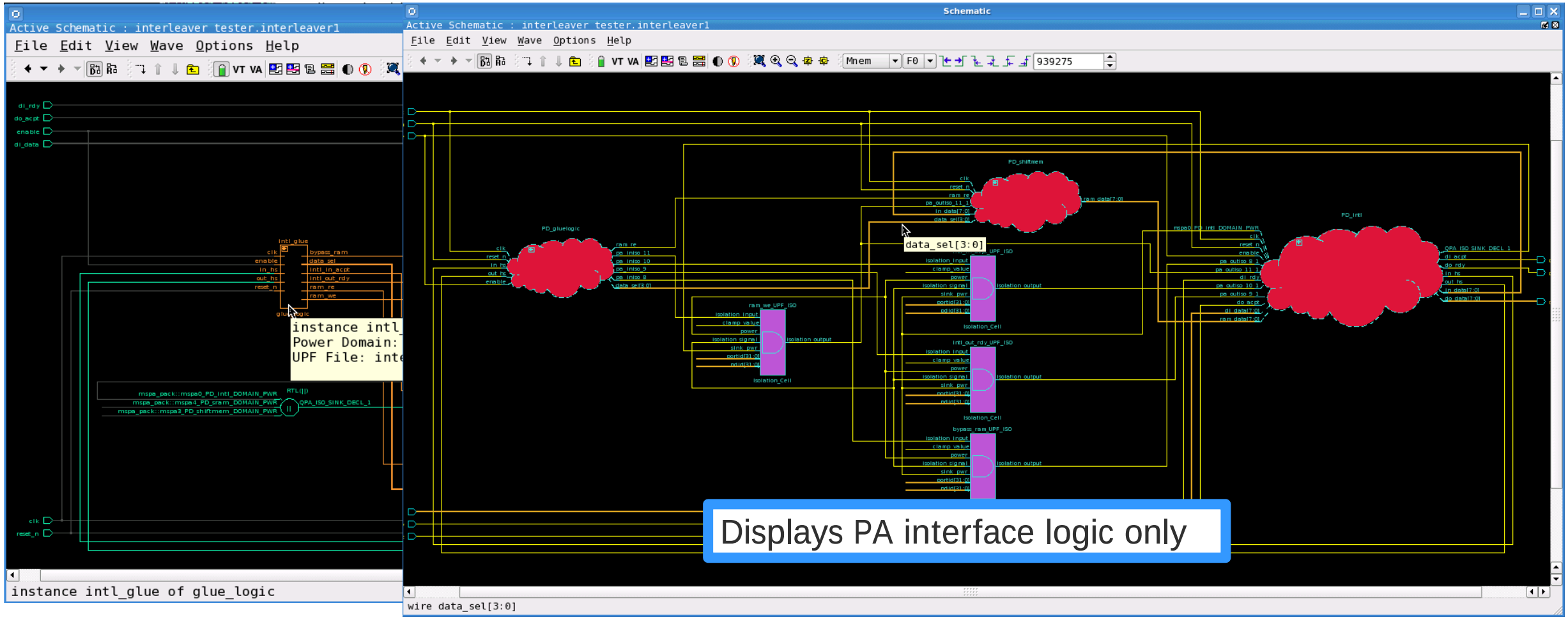
- PA View of the Design (PA Domains)
  - Power Domains
  - Supply Sets
  - Power States
  - Supply Nets/Ports
  - Logic Nets/Ports
  - Power Switch
- PA Crossing Window
  - Debug Static check results
  - View source and sink domain crossings
- PA SimChecks Window
  - Sort and Debug Dynamic Checker Violations
- PA Liberty Window
  - View Liberty PA attributes of Macros



2018 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

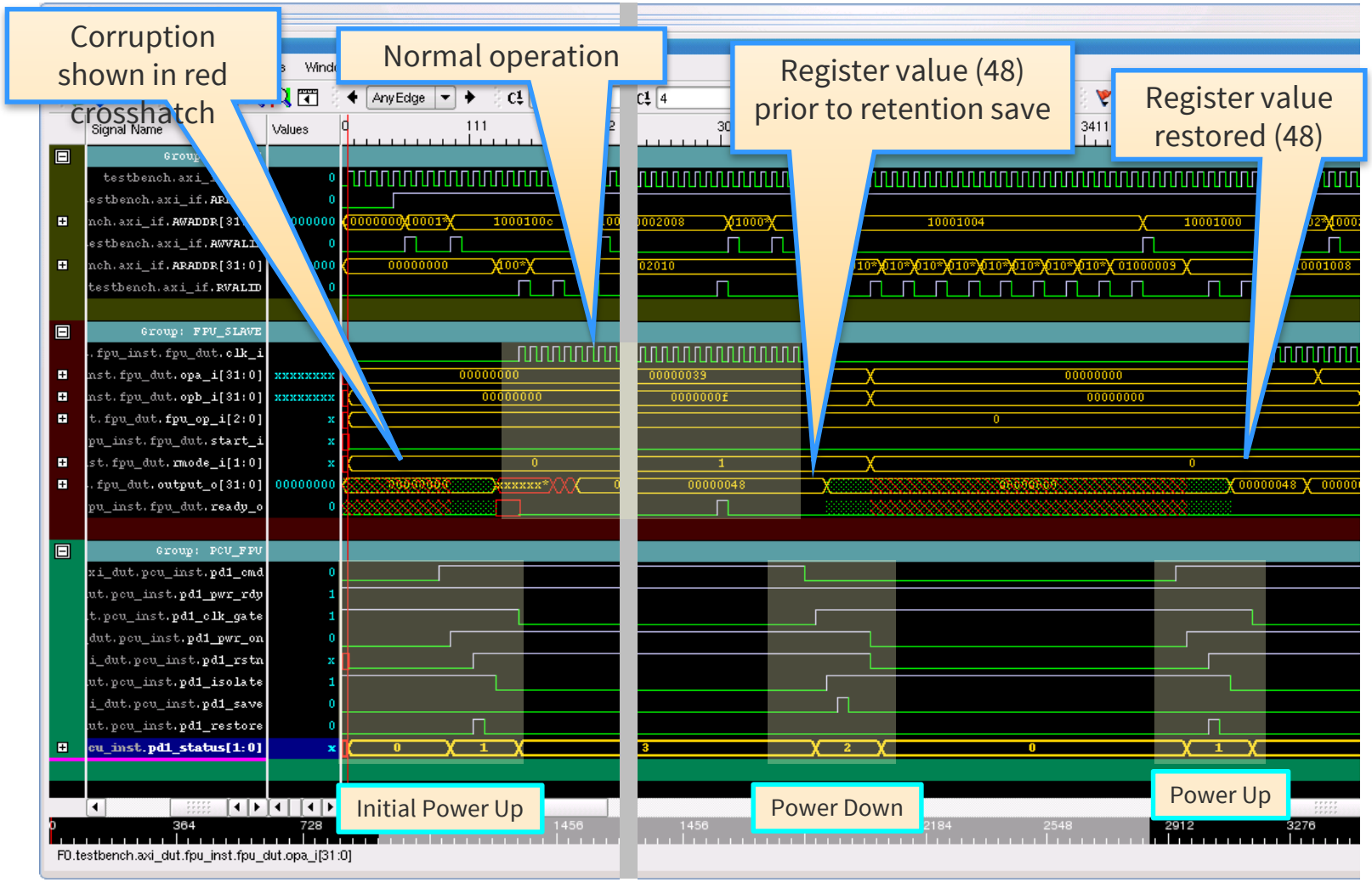
Power Aware Schematic

View Power Annotation and Interface PA Logic



Visualize Power Domain Behavior

Wave Window Highlighting



Coverage Closure in Power Aware Verification

- Start with a plan ..
- Collect relevant coverage metrics – List all of Questa's automated PA coverage metrics
- User can augment leveraging
  - PA Information Model to access the appropriate UPF object and HDL objects
  - Bind_checker UPF command to bind in the coverage collector at the appropriate design scope
- Bring it all together using automation that allows plan to be tracked against coverage metric

Essential Power Aware Coverage

- Power states and transitions from add_power_state, add_port_state, add_pst_state
  - To capture states of groups, power domains, supply sets, PSTs, supply ports
    - Including supply ports/nets of supply set functions used in –supply_expr of add_power_state
    - and supply set simstate and transition coverage for primary, isolation_supply and retention_supply of each domain
- Power Domain state cross coverage based on hierarchically composed power states of add_power_state
- Retention save event and restore event coverage
- Built-in PA Checks : when checker has activated and passed with no firings in simulation
- State and transition of isolation, retention and Power switch logic control and ack signals
- Power switch coverage
  - State and transition of power switch output supply port
  - State and transition of user states –on_states, -off_states, -error_states

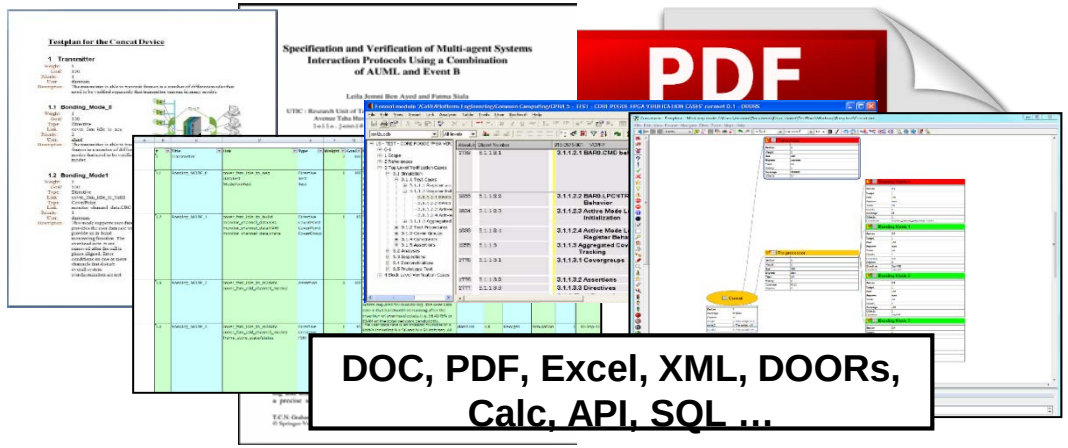
Unified Coverage

- All power aware coverage saved in UCIS compatible database UCDB
- Supports Merging coverage across PA regressions
- Unified Coverage waiving mechanism
  - coverage exclude –pstate default_off my_supply_ss –scope dut
- Extensive Reporting
  - coverage report –pa -html
  - Supported formats include Text, HTML, XML
- Flows for Merging PA and Non PA coverage
- Track coverage metrics against Test Plan

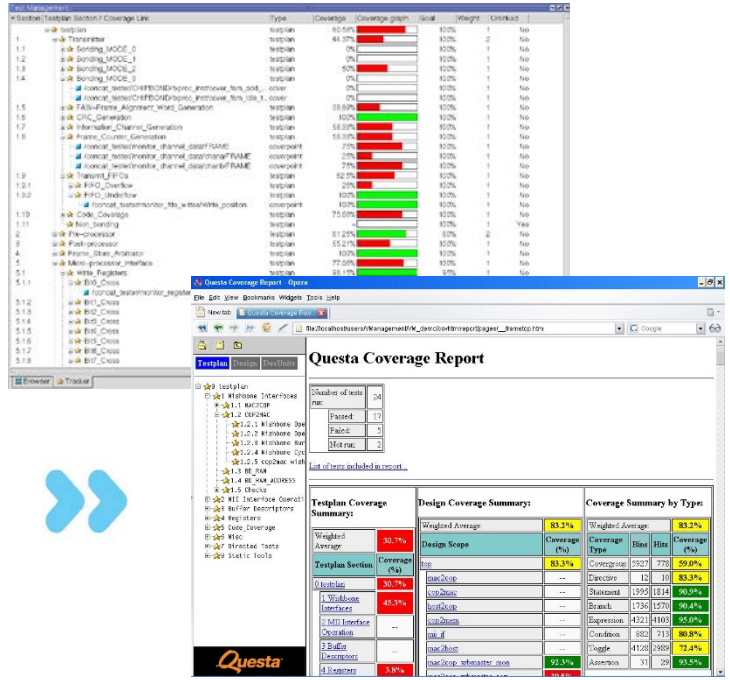
Plan Driven Verification

Testplan Capture, Native to Database

- UCDB natively supports Testplans
  - Import from any data/document source
  - Testplan Tracker provides visibility and progress



Testplan UCDB

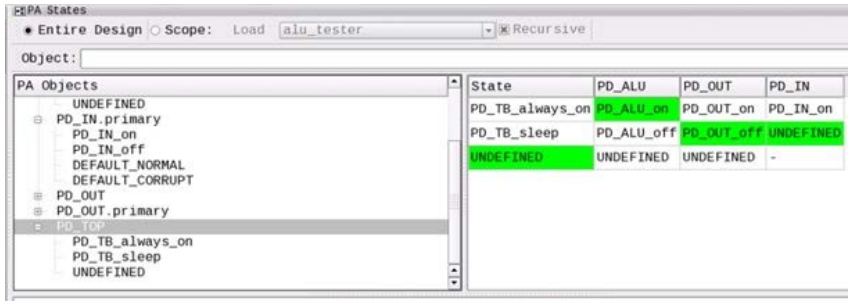


- Supports process of tracking to any metric
  - Project specific attributes/metrics
- Questa PA Sim generates Test plan based on UPF specification

Combining PowerAware Simulation

Coverage, Assertions and VM

Analysis



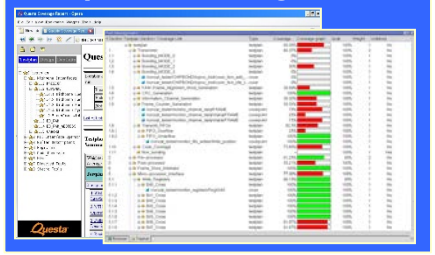
PowerAware Simulation

Power State Coverage,
PowerAware Assertions,
Other PA Coverage

PowerAware Testplan

IEEE Std 1801™-2015

Testplan Tracking



Read/Write API

Trend

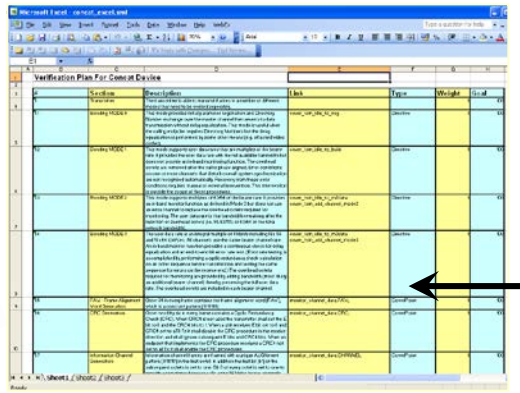
Merge

Tests

Testplan

UCDB/UCISDB

Top Level

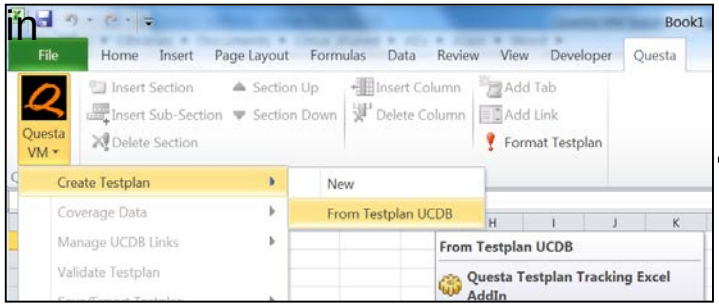


Sub-plan

PowerAware Testplan

Verification Plan For Power Aware State Coverage						
Section	Title	Description	Link	Type	Weight	Goal
1	Power Aware State Coverage	Coverage plan for PA State machines				
1.1	Power State Table	State coverage data for PST PASM	/tb/pa_coverageinfo/MyPowerState Table	FSM	1	100
1.2	Supply Port PD_ALU_PORT	State coverage data for supply port PASM	/tb/pa_coverageinfo/PD_ALU_sw/out_sw_PD_ALU	FSM	1	100
1.3	Supply Port PD_RAM_PORT	State coverage data for supply port PASM	/tb/pa_coverageinfo/PD_RAM_sw/out_sw_PD_RAM	FSM	1	100

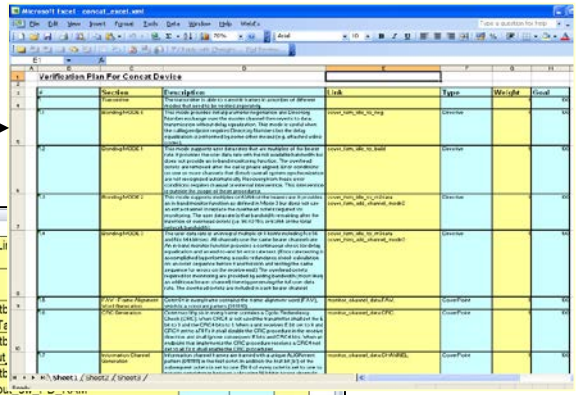
Excel Add-in

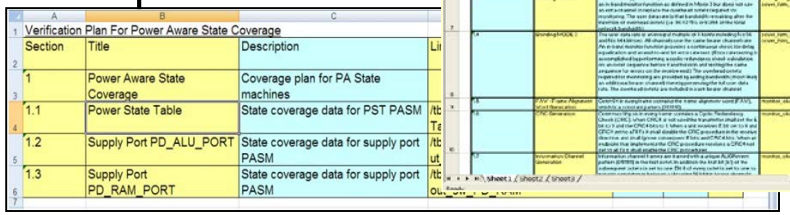


Combining Simulation and Emulation

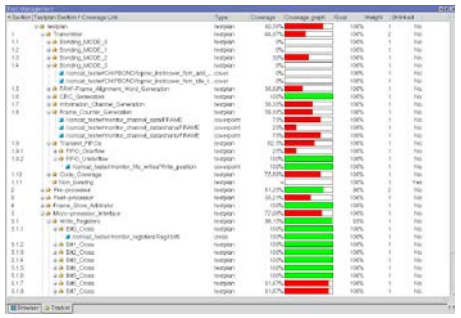
Top Level Testplan

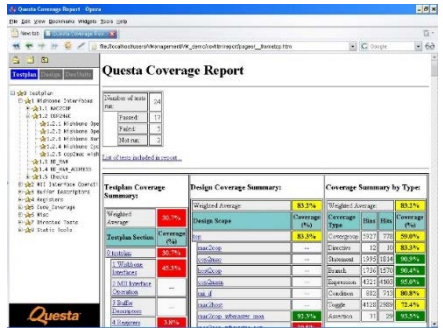
Sub-plan





PowerAware Testplan





Power Aware Test Plan

Simulation

Power Aware Coverage, Assertions

VELOCE® Emulation

Power Aware Coverage, Assertions

Read/Write API

Trend

Merge

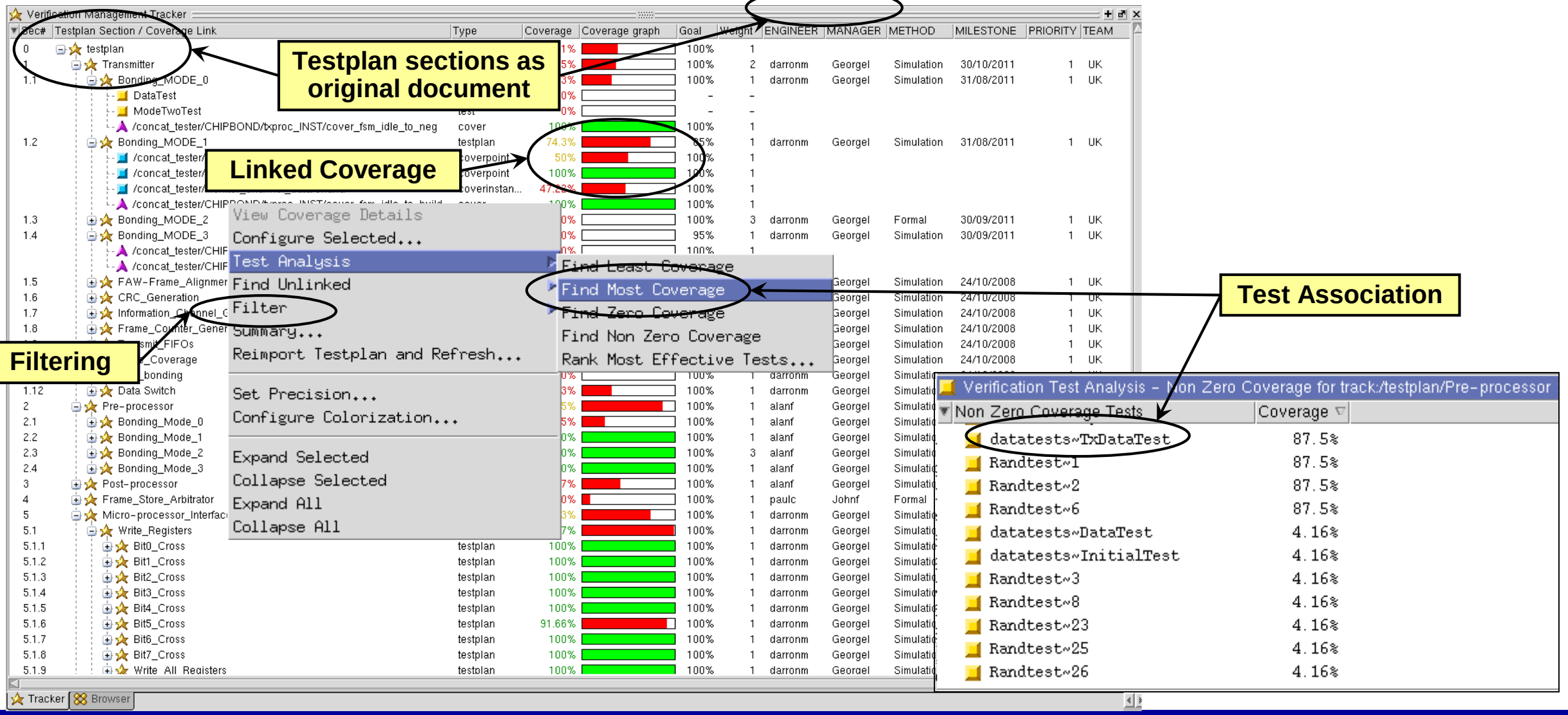
Tests

Testplan

UCDB/UCISDB

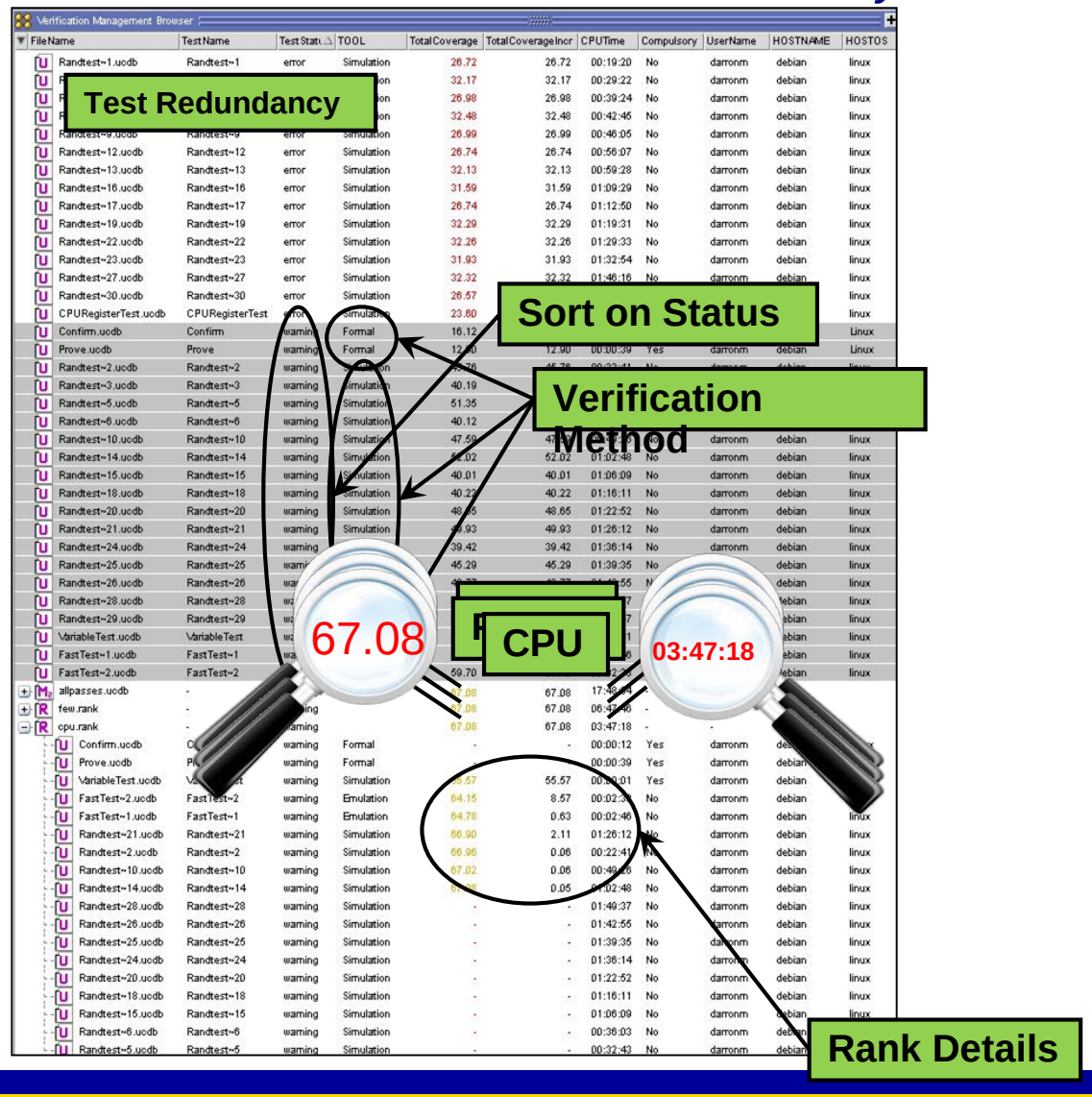
Tracking Verification Against Plan

User Interface, Filter, Query, Sort



Unified Powerful Analysis

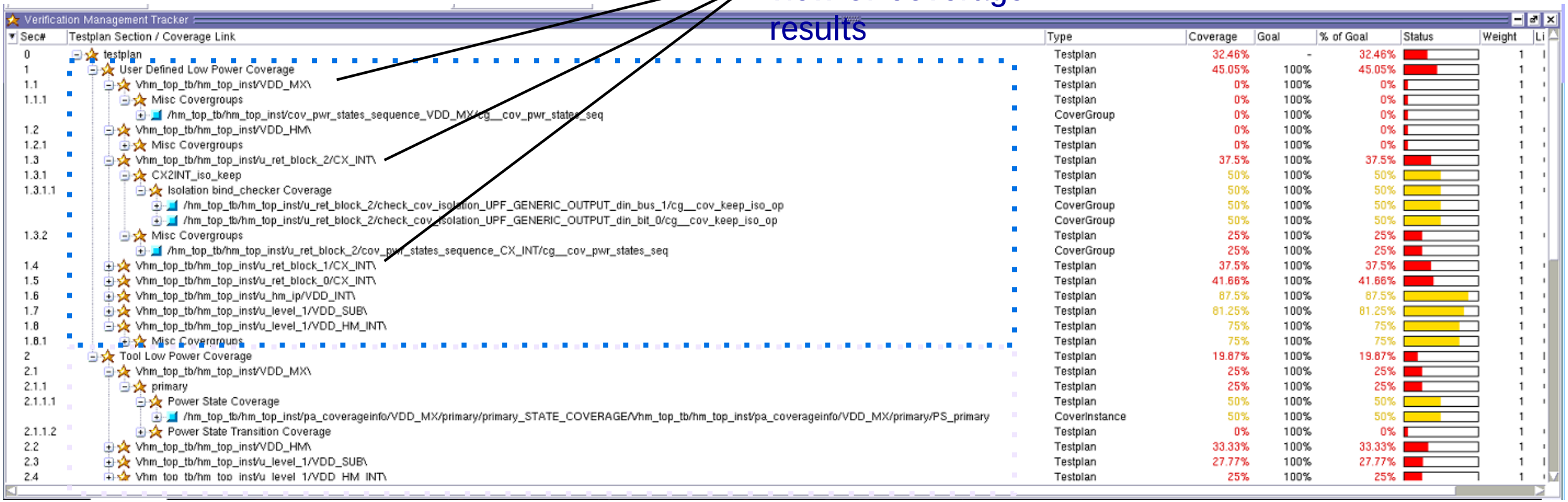
Who, What, When, Where, Why?



Action	Tests	Methods	Total Coverage	Tool Runtime
Sort	35	31 Simulations 2 Formal 2 Emulation	NA	NA
Merge	20	16 Simulations 2 Formal 2 Emulation	67.08	17:38:24
Rank Fewest	8	6 Simulations 2 Formal 0 Emulation	67.08	6:47:46
Rank Quickest	9	5 Simulations 2 Formal 2 Emulation	67.08	3:47:18

Coverage Results Aligned by Power Domain

Power Domain view of Coverage results



Summary

- UPF 3.0 eases specification of design power intent
  - Abstract/refinement of power states
  - Hierarchical composition of power states from IP to System
  - Information Model and API

- Comprehensive PA Coverage and Test plan
  - Extracted from UPF power states and UPF strategies

- Questa PA enables users deploy metrics based verification of low power designs

Let’s take a break

2018 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

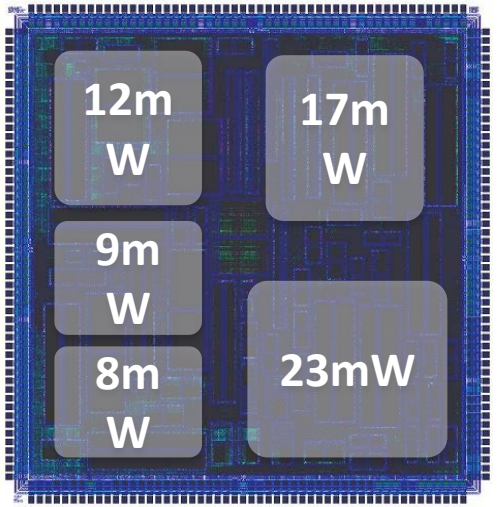
Analysis and Reduction: Metrics for Designing Low-Power IP

Ellie Burns, Director of Marketing for Calypto Systems Division

Mentor®
A Siemens Business

To Prevent "Power Surprises" Need IP Power Audit Methodology

- RTL IP Qualification
  - Customers require IP RTL Qualification Metrics for Power
  - Metrics required to define Power Signoff Criteria
  - Based on the criteria, IPs may be "ranked"



- RTL IPs may require "re-spin" if they do not meet power qualification criteria
  - Coding styles must not be power hungry
  - Wasted power must be reduced
  - RTL designers need to work from a "Low-Power" mindset

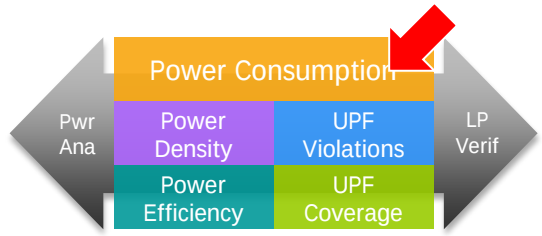
But Why Not Just Use Power Consumption Numbers?

- Looking at power consumption numbers through Power Regression
  - Micro-architectural Exploration – Evaluating how different micro-architectures impact power.
  - What is the power trend as RTL progresses?
  - Is the power budget being met?
- Power Optimization may need to be looked at “Objectively”
  - How many flops in the design are gated?
  - How many flops are highly efficient?
  - How effective are the enables?
  - Are there any redundant toggles in the design?

Power Group	Count	Leakage Power(uW)	Internal Power(uW)	Total Power(uW)	Percentage(%)
memory	2	0.388697	719.494	719.882	20.41%
black_box	1	0.314219	87.93	88.2442	2.5%
register	1171	7.50225	2124.29	2131.8	60.45%
combinational	3048	4.50639	540.253	544.759	15.45%
sequential	11	0.0879686	42.002	42.09	1.19%
total	4233	12.7995	3513.97	3526.77	100%
clock_network	NA	2.63006	18.8737	21.5038	NA

```
Number of flops                          :    26(100.00%)
Number of conditionally enabled flops    :     8(30.77%)
Number of always enabled flops           :    18(69.23%)
Number of CGIC enabled flops             :     0(0.00%)
Clock Gating Efficiency                  :       7.692%
```

Power Consumption Metrics



- Leakage power
  - Power consumed by current flow through transistors even when turned off. Concern in lower geometries and ultra-low-power devices
- Dynamic power = internal+switching power
- Internal power
  - Power consumed by the combination/sequential logic during the switching of the logic states. Depends upon the technology node, switching activity (SA), Load capacitance and input slew
- Switching power
  - Consumed by the nets and is a function of the load capacitances of the cells connected and the switching activity (SA)

Power Consumption Metrics

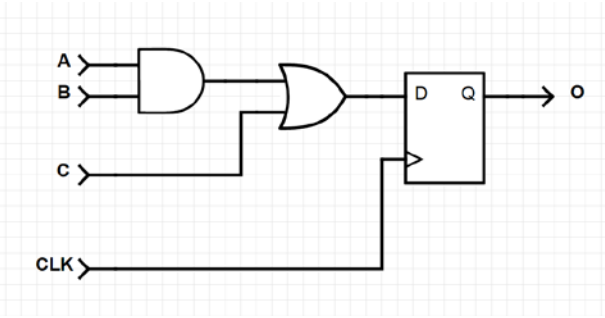
- Clock tree power
  - Summation of all the power groups (leakage, internal and switching) of the clock networks in the design mostly comprised of buffers/inverters and CGIC cells
- Memory power
  - Summation of all the power groups (leakage, internal and switching) of the memories in the design
- Peak power
  - Peak power is defined as the total power of time window in the entire simulation duration where the maximum switching occurs

Measuring Power Consumption: RTL vs Gate-Level

- RTL Power Estimation
  - Fast turnaround time
  - Target within 15% accuracy of gate-level
  - Best suited to track power during RTL development

```
always @(posedge CLK)
  O <= (A & B) | C;
```

- Gate-Level Power Estimation
  - Most accurate flow, close to silicon power consumption
  - Uses either synthesis or post P&R netlist
  - Can use SDF and find glitch power problems
  - Long turnaround time
    - Need to run synthesis and gate-level simulation
    - Good compromise is to be able to use RTL simulation vectors



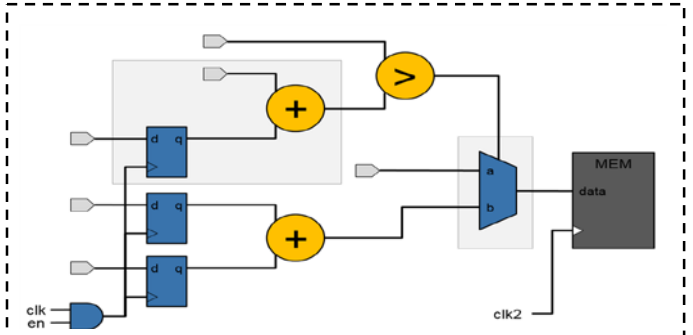
Accurate Power Consumption at RTL Need to Account for Physical

```
always @ (posedge clk or negedge rst)
begin
  if (~rst)
    cnt <= 0;
  else
    if (~en)
      if (~load)
        cnt <= val;
      else
        cnt[3:0] <= cnt[3:0] + 1;
end
```

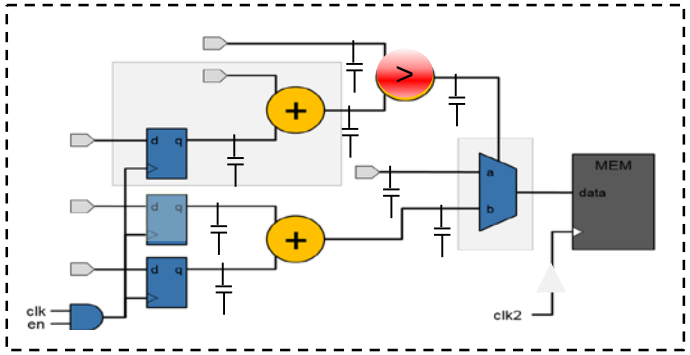
RTL

- Word level infrastructure
- Technology aware mapping
- Buffering
- Clock tree synthesis

Generate quick prototype



Gate level prototype



Physical prototype

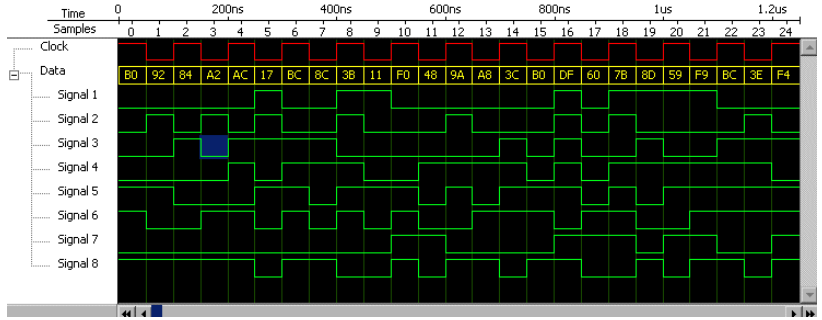
- Bind power critical cells – Flop, latch, CGIC
- Vth distribution
- Clock tree topology
- Fine-grain wire cap model

Physical char. ← SPEF

```
*SPEF "IEEE 1481-1998"
*DESIGN "top"
*DATE "Wed Oct 1 2014"
*VENDOR "Mentor Graphics Corp."
*PROGRAM "Olympus"
*VERSION "1.0"
*DESIGN_FLOW "PIN_CAP NONE" "NAME_SCOPE LOCAL"
*DIVIDER /
*DELIMITER :
*BUS_DELIMITER []
*T_UNIT  1.0000 PS
*C_UNIT  1.0000 FF
*R_UNIT  1.0000 KOHM
*L_UNIT  1.0000 HENRY
*143620 n20
*169 clk_buf
*73901 mips/dp/rf/auto_clk__c10525
…
```

SPEF

Accurate Power Consumption Requires Good Switching Activity Input



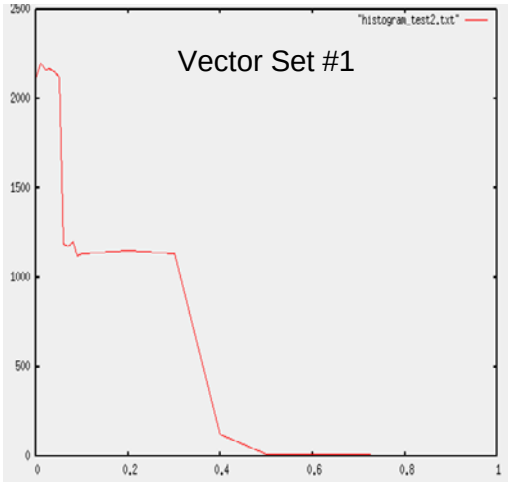
- Understanding switching activity scenarios are important for estimation
- User assumptions about vector sets are often incorrect
  - Could cause a power problem to be missed
- Format choices for switching activity
  - SAIF – Sums of toggles, smaller file size, only average power can be calculated
  - SDPD SAIF – More accurate
  - Waveform data – FSDB, VCD, other native formats – large file size, can be used for peak power and optimization
  - Emulation – Native waveform, Direct API

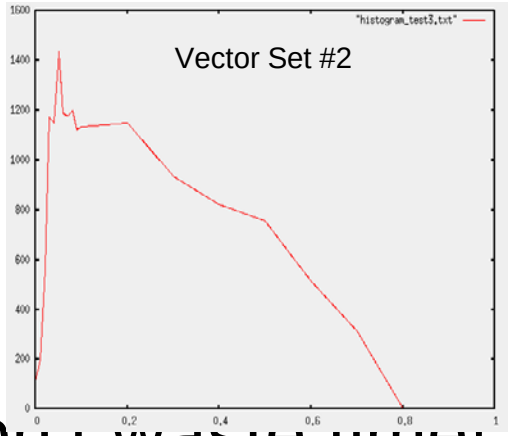
Idle Mode?
High Traffic?
Save Power?

Need to Have a Way to Measure the Quality of Switching Activity Data

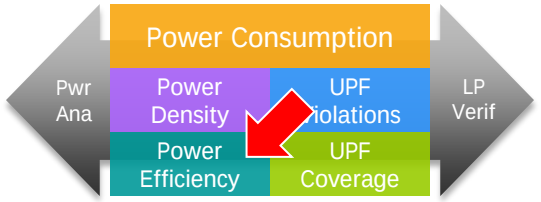
- Need to audit switching activity data
  - Helps Categorize: Idle, Sustained or Traffic
  - Generates toggle count histograms
  - Detailed reports for all critical signals
  - Flags problematic vectors
  - Flags dead/blocked clocks
- Saves valuable debug time
- Better productivity through better quality vector sets – Don't waste time!

Toggle rate histograms





Examples of Power Efficiency Metrics



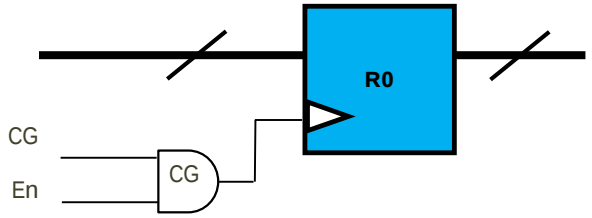
- Clock-gating Ratio
  - Metric used to measure the percentage of un-gated flops vs. gated flops
- Constant Enable/Clock for ICGs
  - Metric to measure if any ICG's are driven by a constant and can be removed
- Redundant Resets
  - Metric to measure how many flops have resets which are not required and therefore a lower power cell could be used

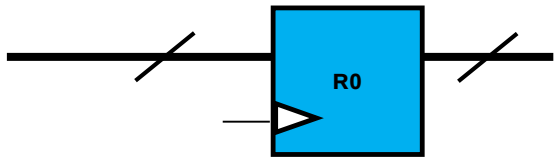
Examples of Power Efficiency Metrics

- Clock-gating efficiency
  - Metric used to measure the effectiveness of clock gating (ie., reduction of switching activity due to CGIC insertion) of registers in the design
- Memory-gating efficiency
  - efficiency of the memory enables of the memory and the effectiveness of the signals controlling the sleep modes available in the memory.
- Redundant Toggles/Wasted Toggles – “Power Leak”
  - Any toggle-activity in the design which is not required for correct functionality

Ungated vs Gated Flops

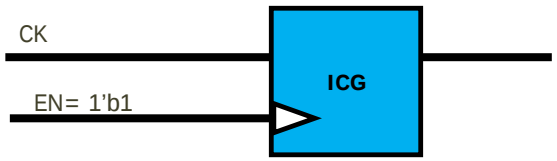
- Can be determined statically
- Just a number (ratio)
- Tells how many flops in the design are gated vs how many flops in the design are ungated
  - Excludes Synchronizer flops
- Ideally, all flops must have an enable
  - Indicates that RTL designer can be more power sensitive in coding

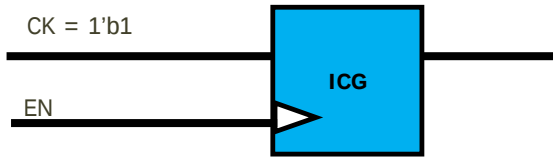




Constant Enable/Clock for ICGs

- Can be determined statically
- Check to see if ICG enable or clock is driven by a constant
- Based on this, user can decide to remove the ICG
- Can question validity of the enable (will consume leakage power)





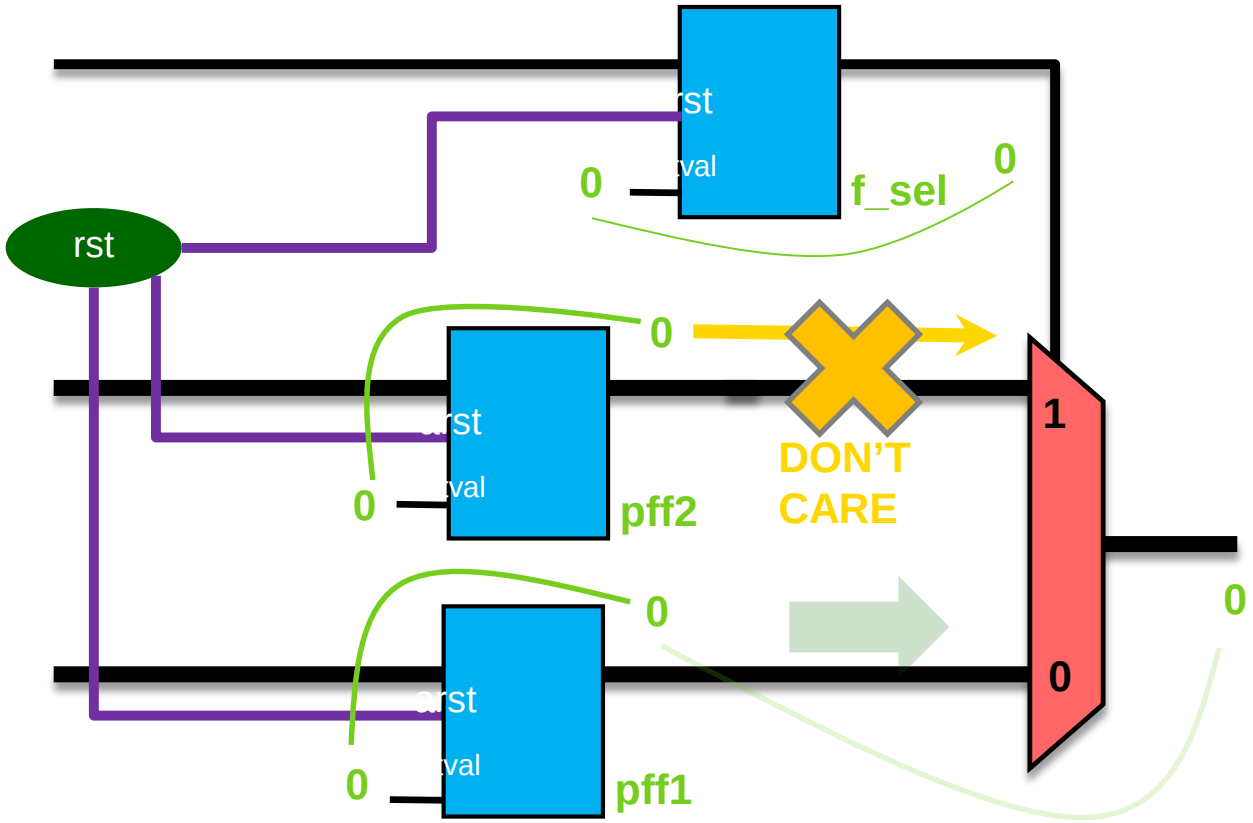
Redundant Resets

Whenever reset goes HIGH…

Output of pff2 would be DON'T CARE

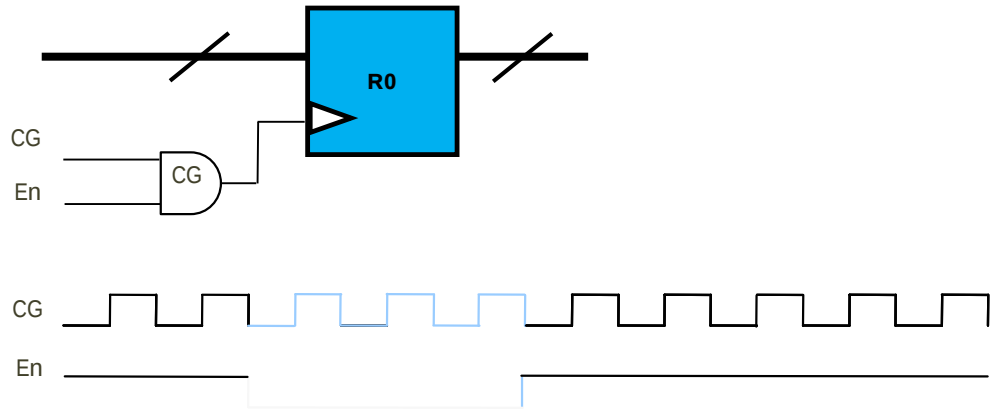
ResetVal of pff2 is never observable

Reset of pff2 is redundant



Clock Gating Efficiency

- Percentage of simulation cycles that the clock is gated off
- Higher CG Eff means lower dynamic switching power consumption
- 100% efficiency => Flop doesn't switch at all
  - Can be indicative of an inactive use mode
- More gating does not always save power
  - Gating logic for the enable can cost more than register power saved

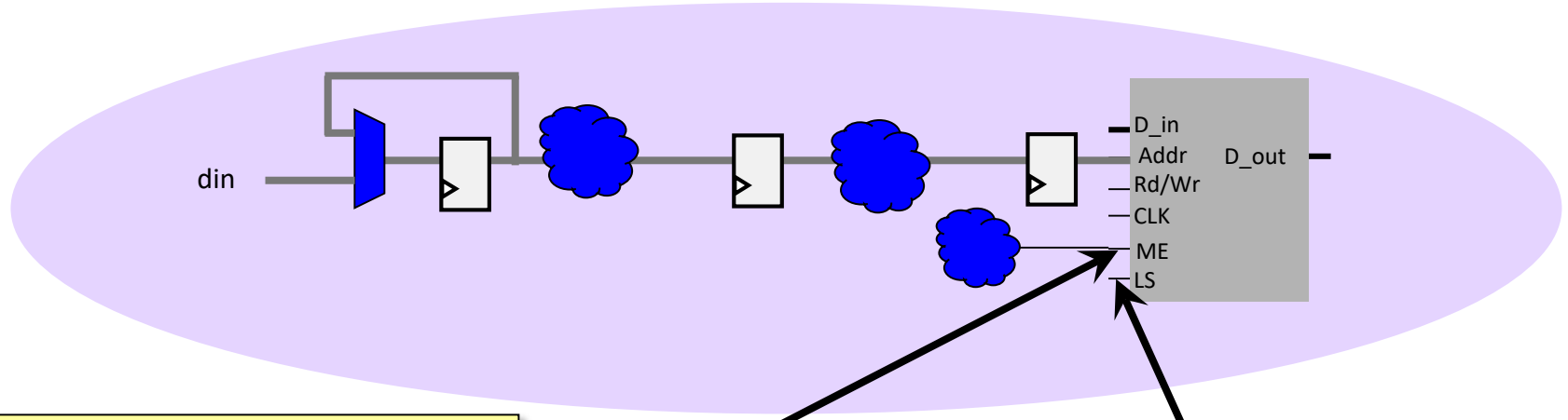


Total Clock Cycles = 10

Clock gated for = 3 Cycles

Clock-gating Efficiency = 30%

Memory Gating Efficiency

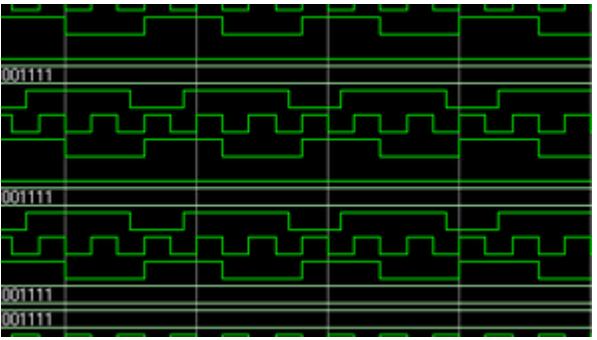


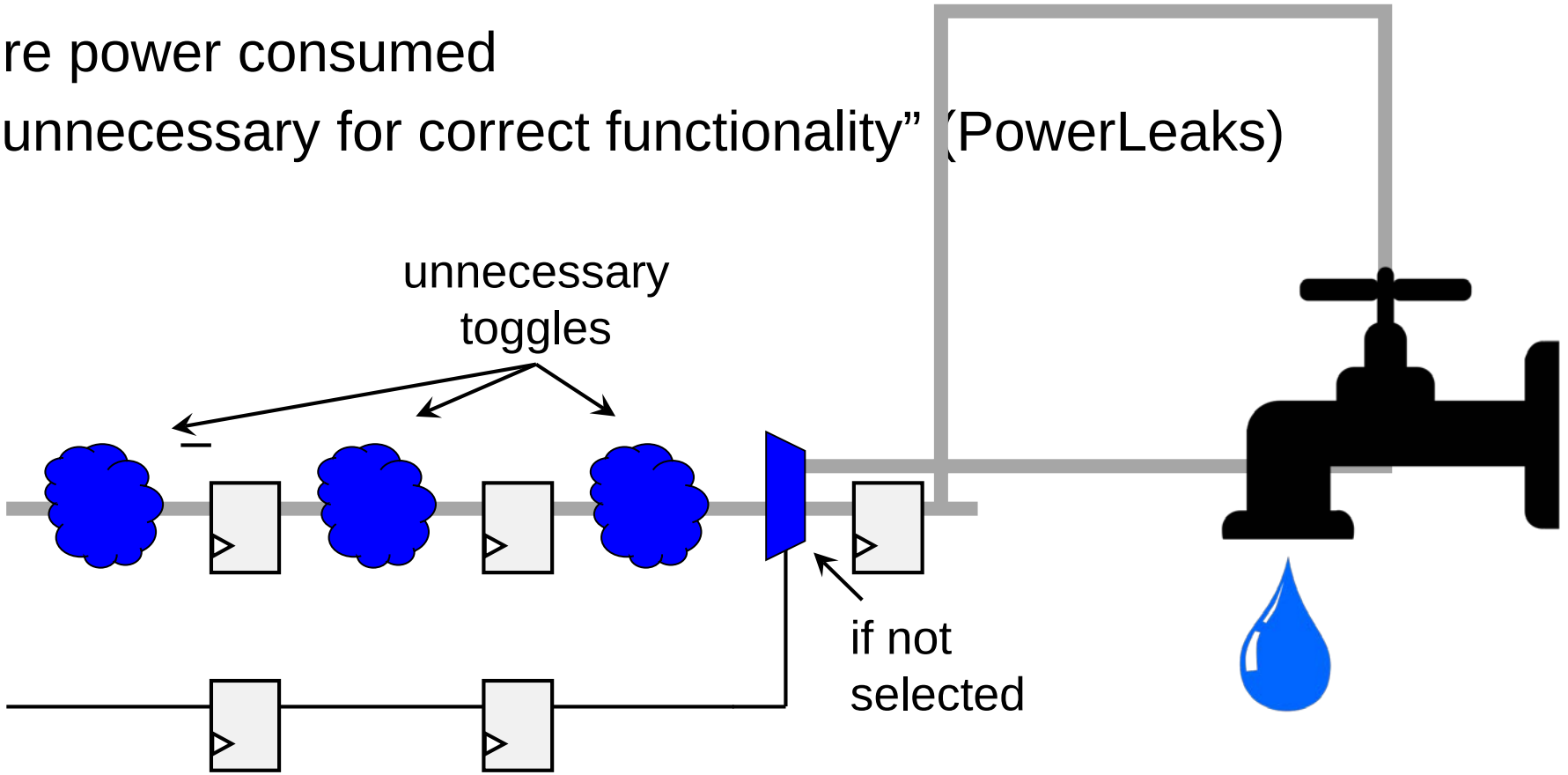
Probability-of-being-0 of “ME/CS” signal is a good indicator of potential dynamic power-saving due to memory-gating and can serve as proxy for measuring power savings

Probability-of-being-1 of “LS” signal is a good indicator of potential leakage power-saving due to sleep-gating and can serve as proxy for measuring power savings

Redundant Toggle Metric – Power Leak

- More toggles = More power consumed
- Many toggles are “unnecessary for correct functionality” (PowerLeaks)





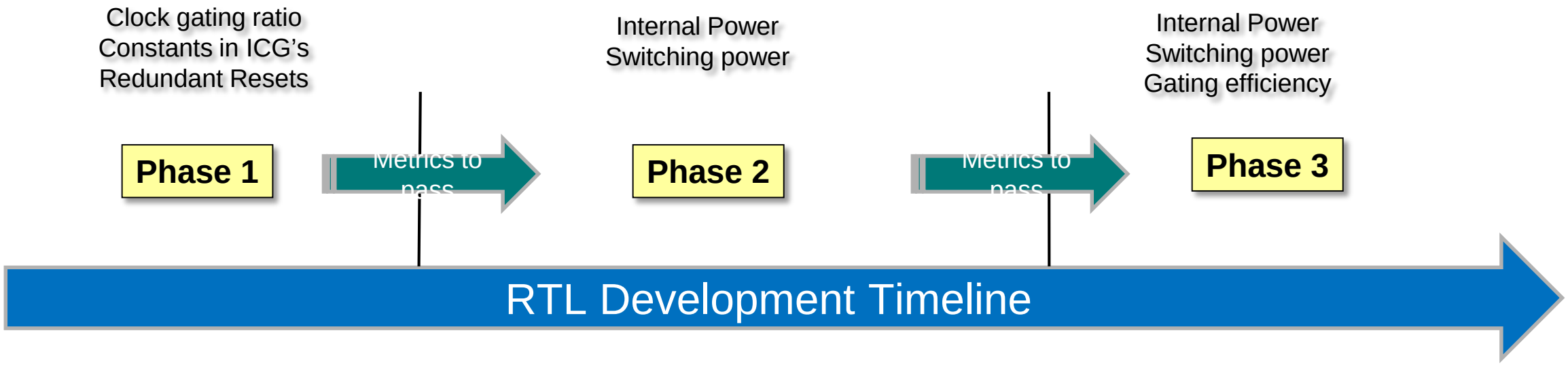
Any toggle which is not needed for correct functionality is a power leak

Types of Redundant Toggles and Events that Waste Power

- DATA GATING:
  - Datapath operators (adders, multipliers etc.) whose inputs can be gated to reduce redundant toggles.
- REDUNDANT MUX TOGGLES:
  - Multiplexers whose unselected inputs are consuming a lot of power.
- CLOCK-TOGGLE DATA-STABLE GATING POTENTIAL:
  - Reports potential power savings and efficiency improvement that can be achieved by applying Clock-Toggle Data-Stable based gating on stable flops.
- REDUNDANT MEMORY ADDRESS TOGGLES:
  - Memory address ports which are very active despite memory not being enabled for that duration.
- REDUNDANT MEMORY DATA TOGGLES:
  - Memory data ports which are very active despite memory not being enabled for that duration

Qualifying your IP for Power Throughout the RTL Development Flow

- Use metrics that don't require "power simulations" very early in RTL to gauge whether RTL designers have done adequate work for power
- Track metrics in regression as RTL blocks mature and working towards power budget
- Use metrics for efficiency to "scrub" for power and judge RTL readiness



PowerPro in the RTL Design Cycle

RTL Development Cycle/Timeline

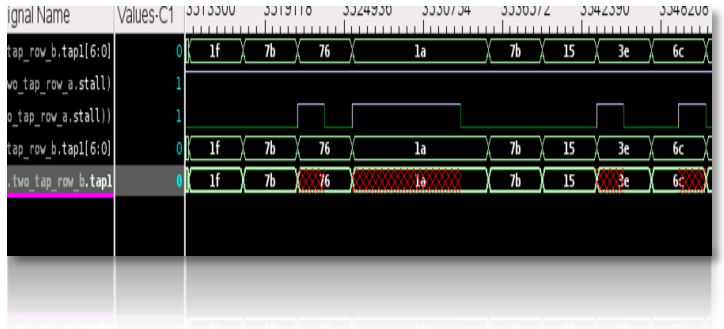
- Just need design

```
module adder
  #(length = 4)
   (input wire   [length-1:0] a, b,
    output logic [length-1:0] sum,
    output logic overflow);

   logic [length-1:-1] carry;

   always @(a or b)
     begin
        carry[-1] = 0;
        for (int i=0; i<length; i = i+1)
          begin :addloop
             sum[i] = a[i] ^ b[i] ^ carry[i-1];
             carry[i] = ((a[i]^b[i]) & carry[i-1]) | (a[i] & b[i]);
          end : addloop
        overflow = carry[length-1] ^ carry[length-2];
     end
endmodule :adder
```

Early RTL

PowerPro Static Checks

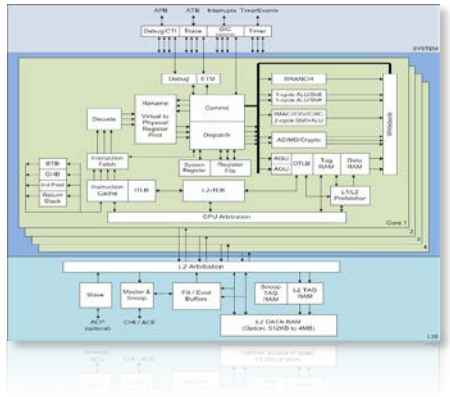
- Need design and "ok" toggle activity



Functional RTL

PowerPro Estimation and Guided Optimization

- Design both block and SoC
- Good activity and power scenarios

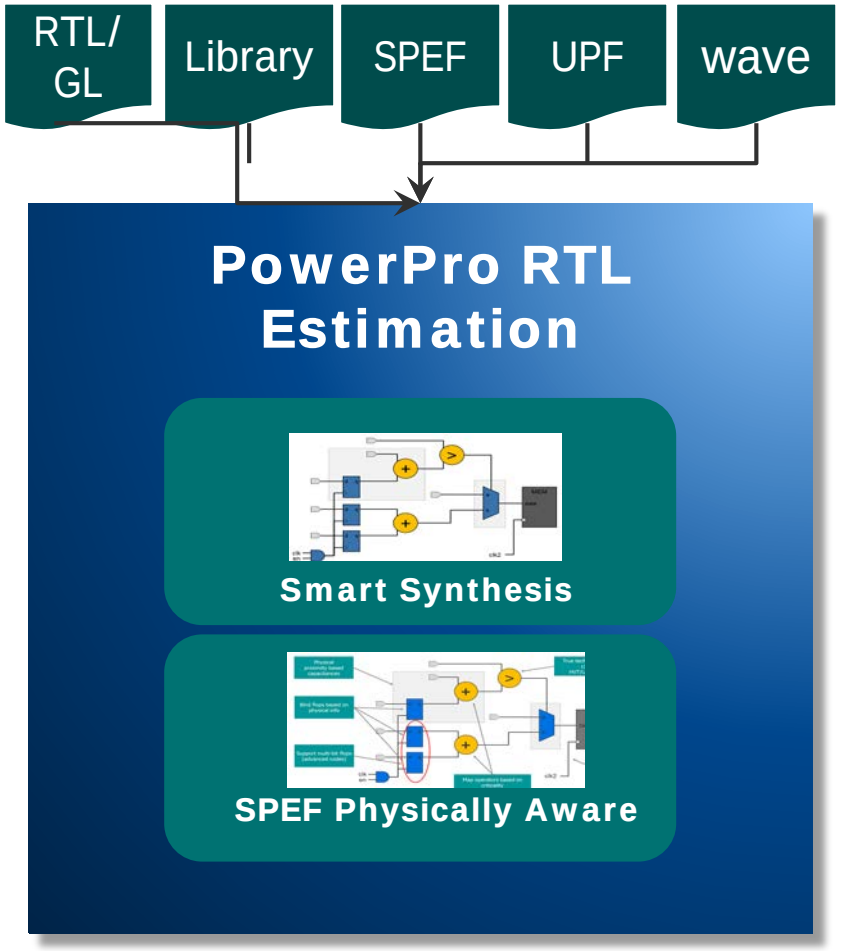


Stable RTL
Block and SoC Verification

PowerPro RTL and Gate Estimation and Automatic Optimization

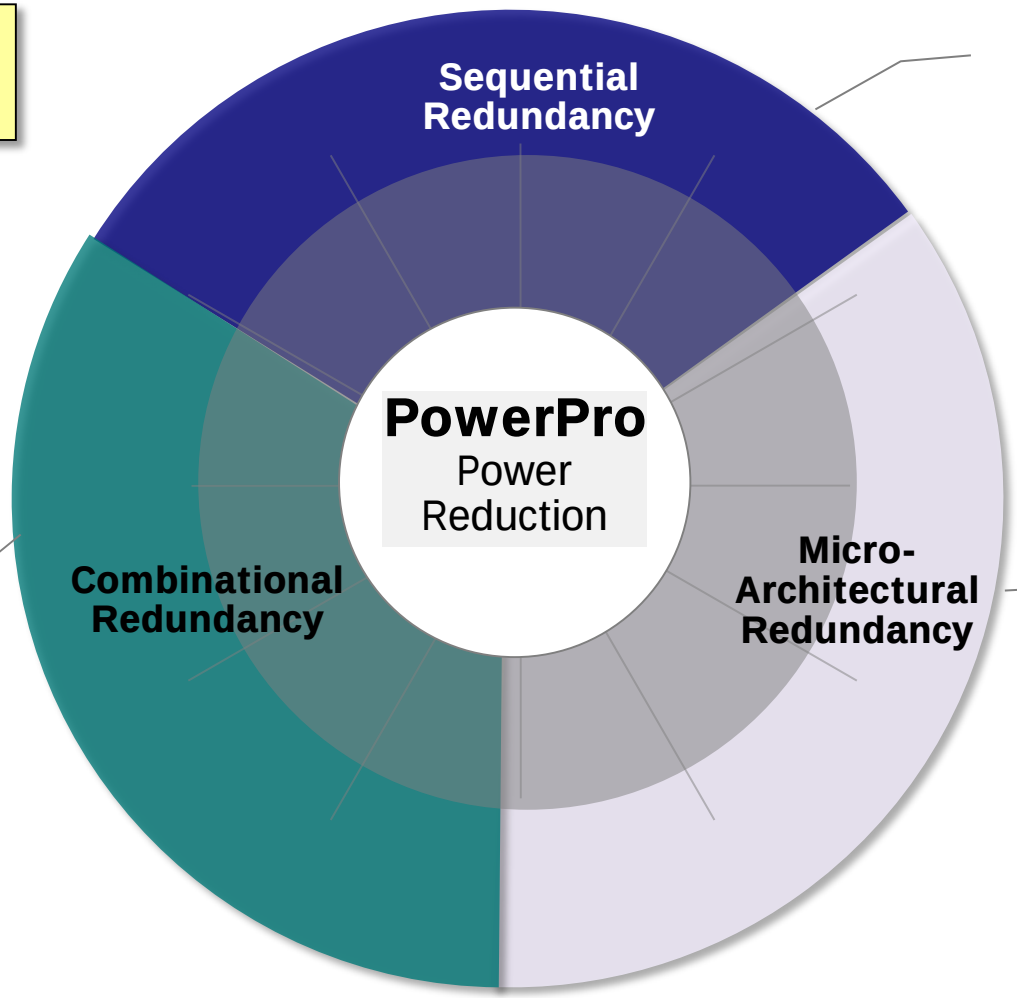
PowerPro Fast and Accurate Power Estimation

- "Physical Aware" flow
  - RTL accuracy: within 15% of layout
  - GL accuracy: within 2-5% of layout
- High Performance and Capacity
- Supports both RTL and Gate-level estimation
- Comprehensive power reports
  - Average and Peak
  - Dynamic and Leakage
  - Logic, memory, register, clock-tree
  - Hierarchical reporting



PowerPro Power Optimization Opportunities and Metrics

PowerPro Finds More Kinds of Opportunities than other Tools



- Identification of wasted power...

... in sequential logic...

... on one or more timed sequential boundaries

- Identification of wasted power...

... in combinational logic...

... on a single timing boundary

- Identification of wasted power...

... that requires more complex micro-architectural transformations

Visualizer Debugging for PowerPro

Design Tree

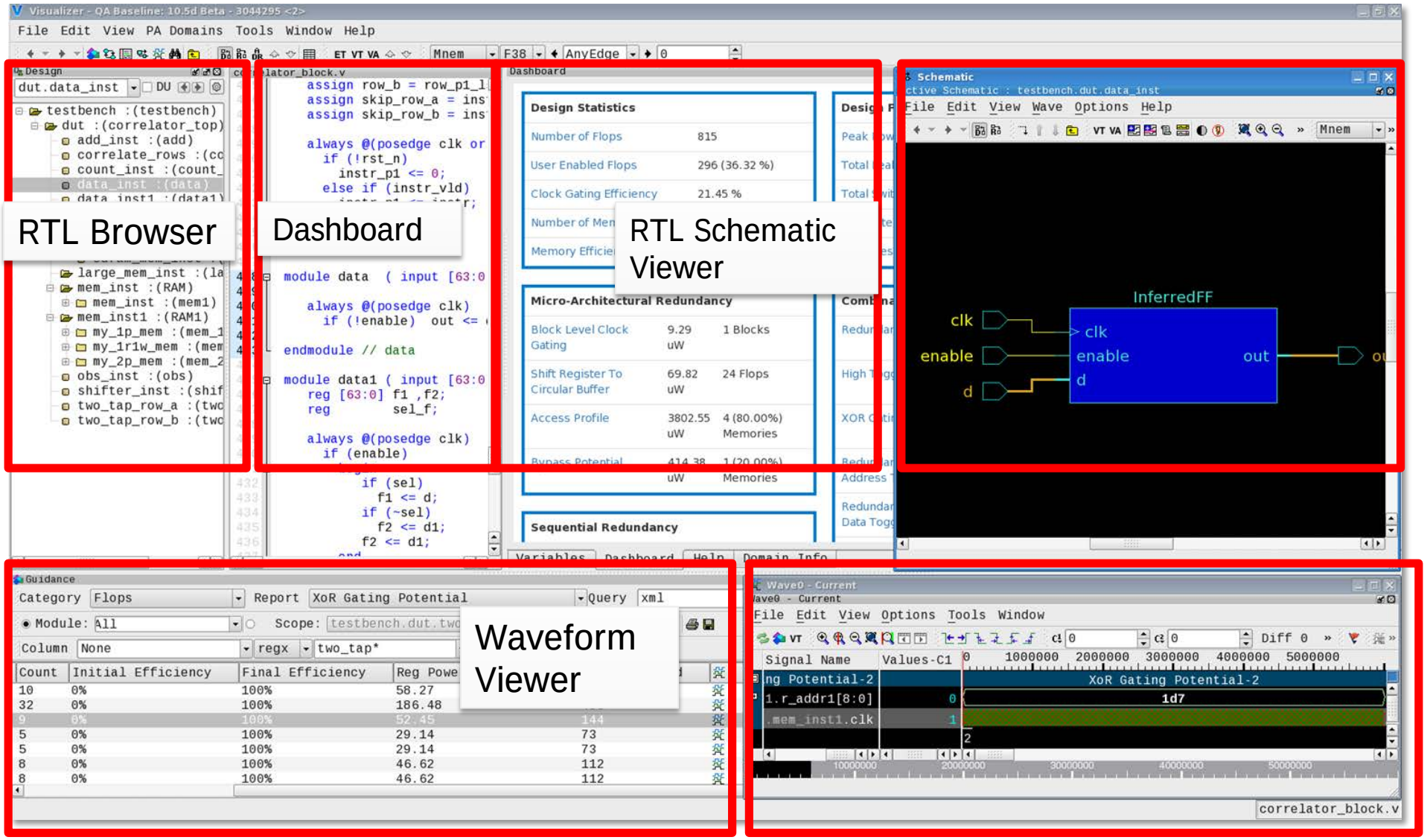
RTL Browser

Dashboard

RTL Schematic Viewer

Reports

Waveform Viewer



Arm® – Low-Power Mindset with PowerPro

- Used across Arm IP: CPUs, GPUs, interconnect sub-systems, and display cores
- Arm deploys PowerPro from the very start of a pro cycle
- All block IP's run PowerPro frequently before RTL check-in
  - Power benchmarks run in weekly regressions
- RTL estimation within 4-14% accuracy across projects
- Power savings of 10-15% on benchmarks across project schedule

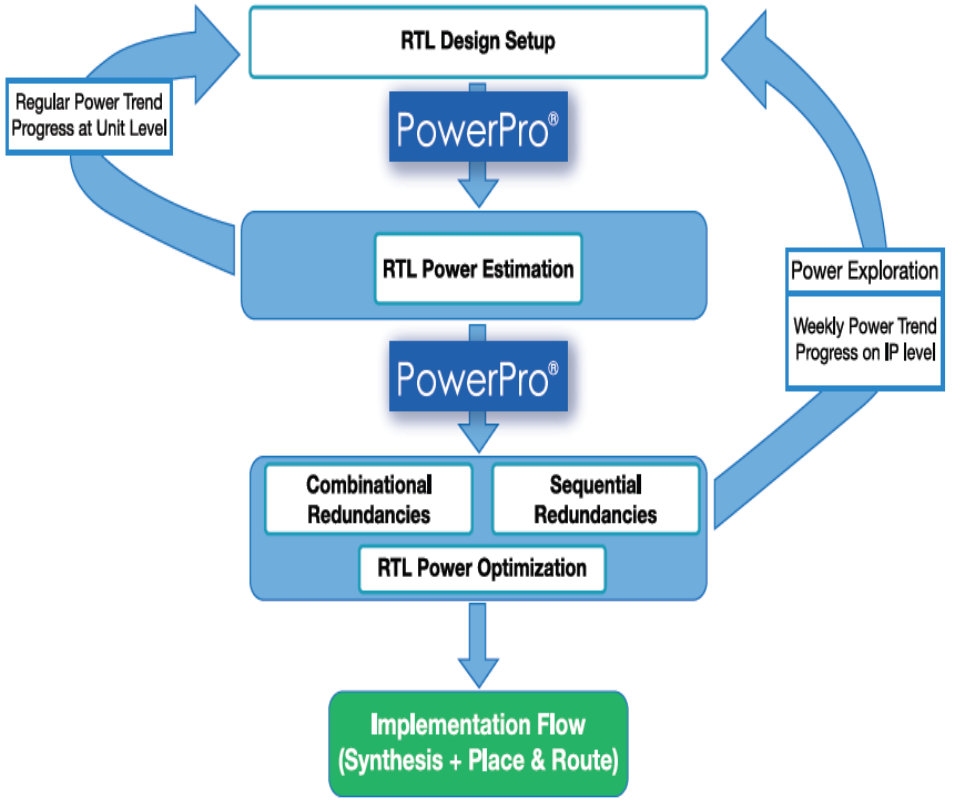


Figure 3: Low-power IP design flow

http://go.mentor.com/4PmgB

2018 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

Summary

- Today's ultra low-power designs will demand a predictable RTL design methodology to achieve power
- Meaningful metrics will be required to qualify IP for power
- Mentor's PowerPro is a complete Low-Power Solution providing metrics, debugging and optimization

2018 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

Using Emulation for Meaningful Metrics-Based Power Analysis and Verification

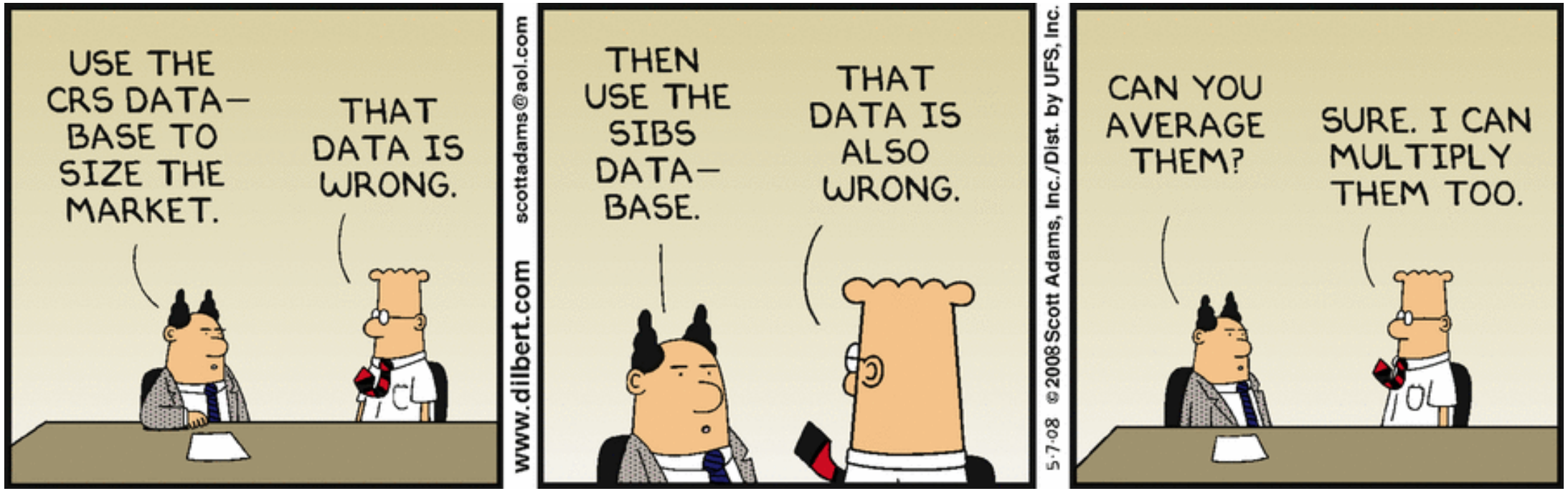
Guillaume Boillet, Power Product Specialist

Mentor®
A Siemens Business

Agenda

- Why is Emulation needed for Power Analysis?
- What emulation can help with
  - SAIF Generation
  - Dynamic Read Waveform API
  - Activity Plot
    - Real life example: System activity and Arm AXI protocol debug
  - Low-Power Verification with UPF

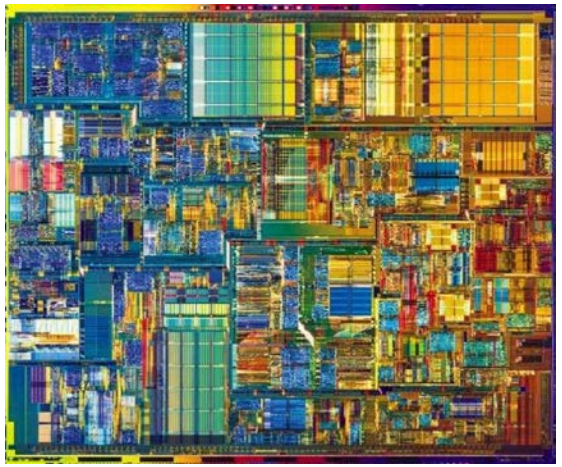
Garbage In Gospel Out



From http://dilbert.com/strip/2008-05-07

- How good is your simulation dataset for the power analysis task at hands?

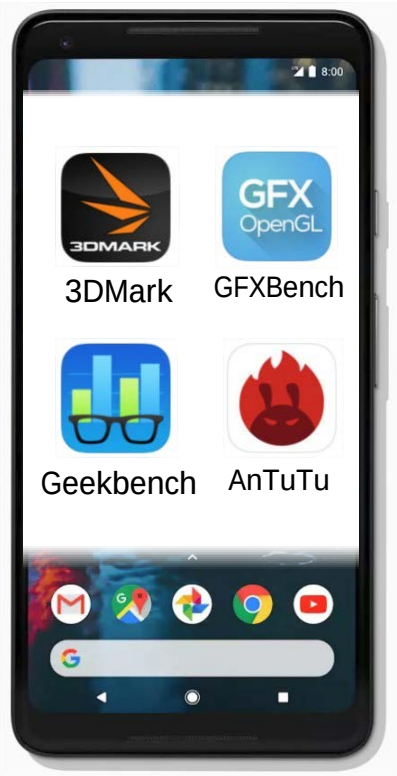
Why Emulation for Power?



Handle Large SoC
(RTL/Gate)

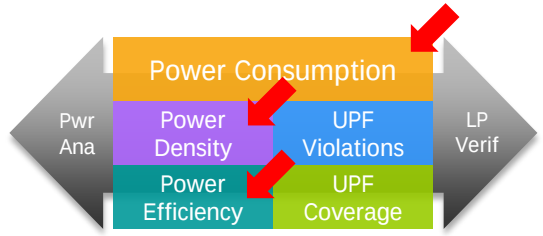
Performance for Complete Verification (e.g. OS Boot)
[100s Millions of Cycles !]

SPEED



Accurate Power Numbers Based on **Real Switching Activity**

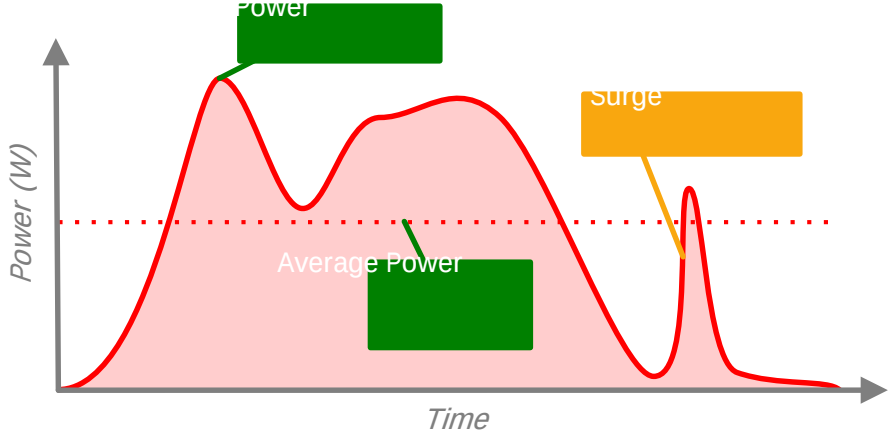
Motivations for Good Vectors & Emulation



- Representative Power Estimation
  - **Average power** (battery life, cooling requirements, cost of energy,…)
  - **Power peaks**
    - **Large peaks** (~1us) -> Supply integrity,…
    - **Narrow peaks** (~1ns) -> IR-Drop,…
  - **Hot spots** (Local IR-Drop…)
  - **Power domains partition** verification via UPF [Emerging trend – Users increasingly Ready]
  - **Power surges** (dI/dt voltage drop)
  - High power on very long periods (Electro-migration)

Primary Concerns

Secondary Concerns

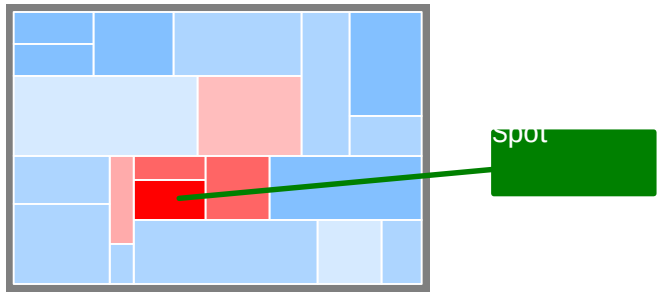


$$VDD_{PMOS} = VDD_{supply} - IR_{grid} - L_{interface} dI/dt$$

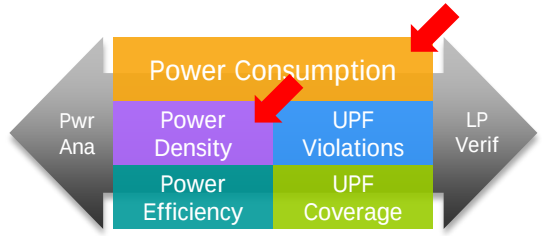
Voltage Drop

BUT ALSO…

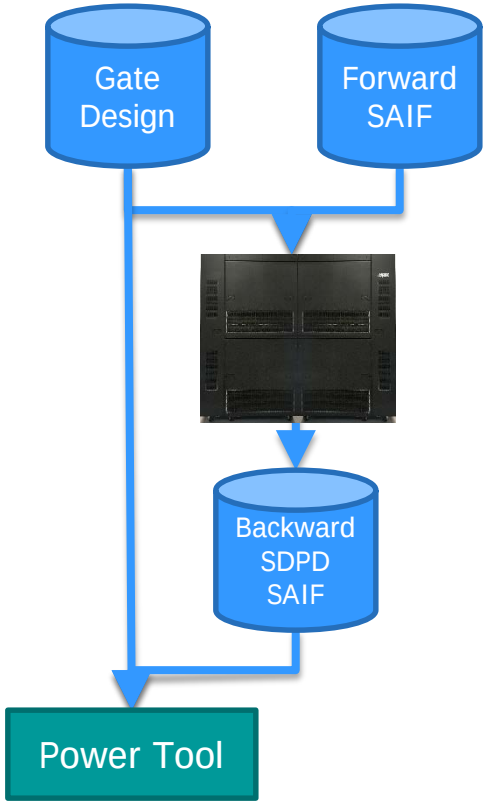
- Representative Power Efficiency analysis
  - Clock-Gating Efficiency (instantiated or inferred)
  - Memory accesses,…
- Relevant Power Reduction suggestions RTL
  - Some techniques are highly dependent on quality of local activity information (i.e. Sequential Clock Gating)



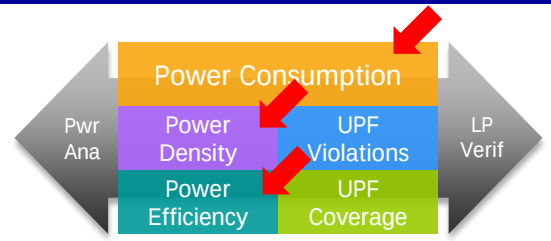
SAIF Generation



- For RTL or netlist
- Online (as emulator is running) or Offline generation
- State-Dependent Path Dependent at gate-level
  - E.g. Toggles of Net A caused when Net B has a rising edge
  - These condition/directives are stored in a Forward SAIF file



Regular vs. Integrated Power Analysis Flow (API)



Regular Flow

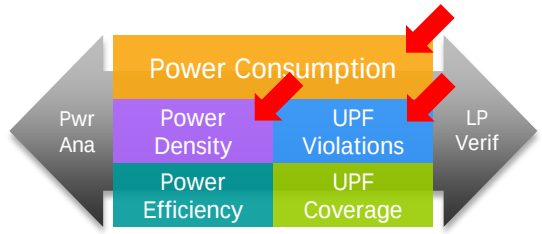
Emulator / Simulator → Activity File Generation → Power Tool Activity File Consumption → Power Data Generation

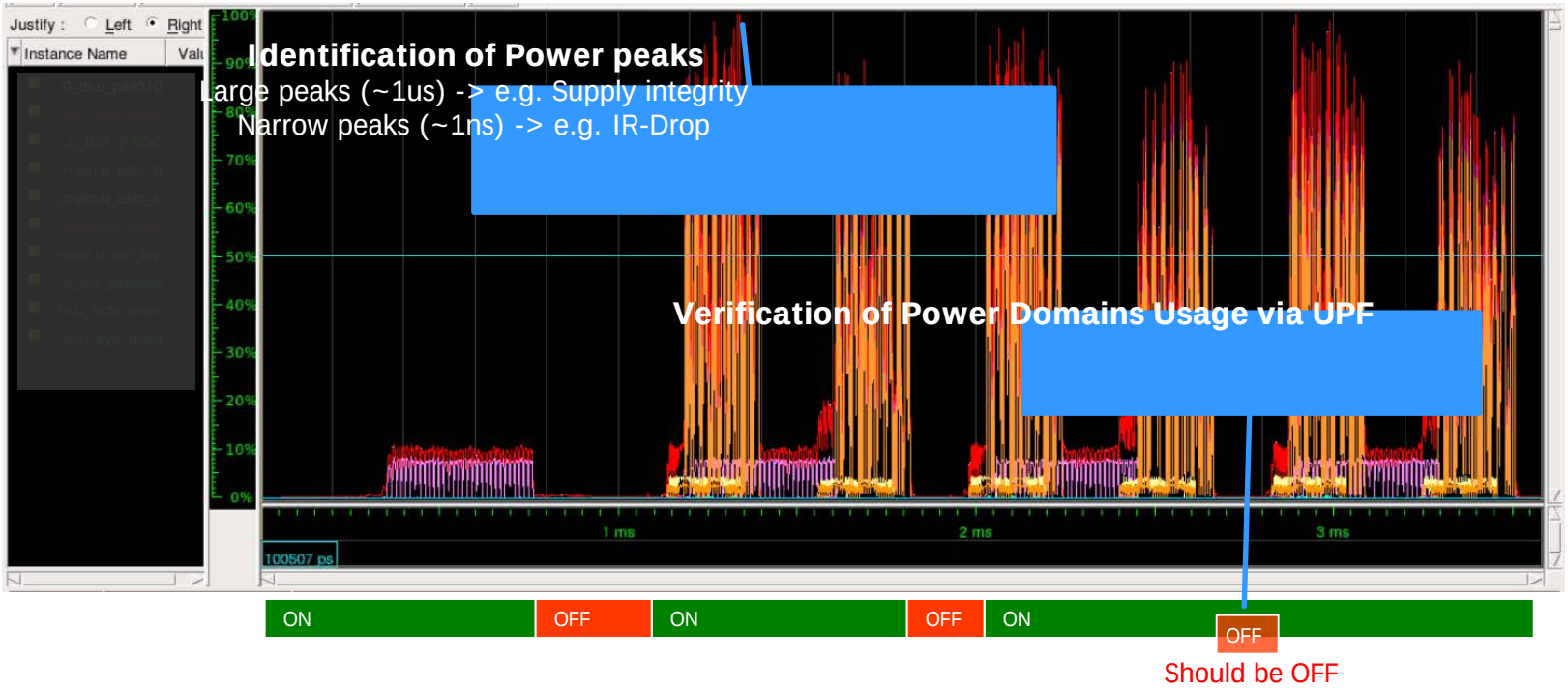
Integrated Flow

Waveform Transfer API → Power Data Generation

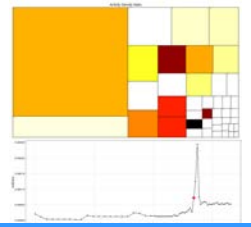
Up to 10X faster
Reduced disk space requirements

Activity Plot Applications





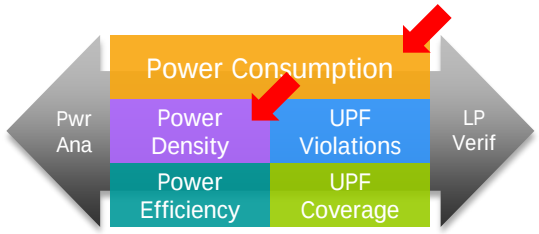
100X+ faster than with traditional Power Analysis tool



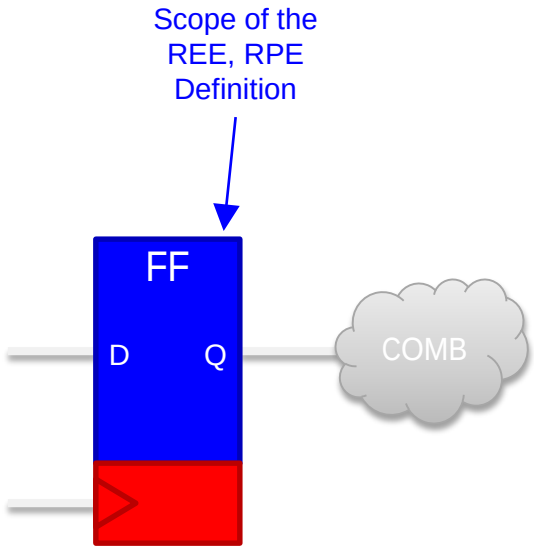
Hot Spots Identification
Optimization targets, Local IR-Drop,...

Power Trends
Compare activity plots across RTL drops

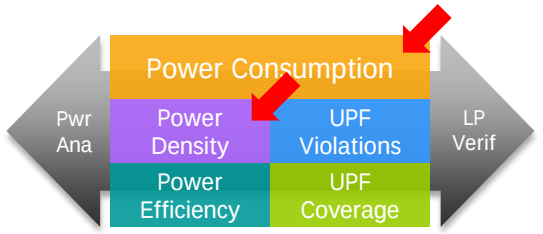
Activity Plot Concepts & Definitions



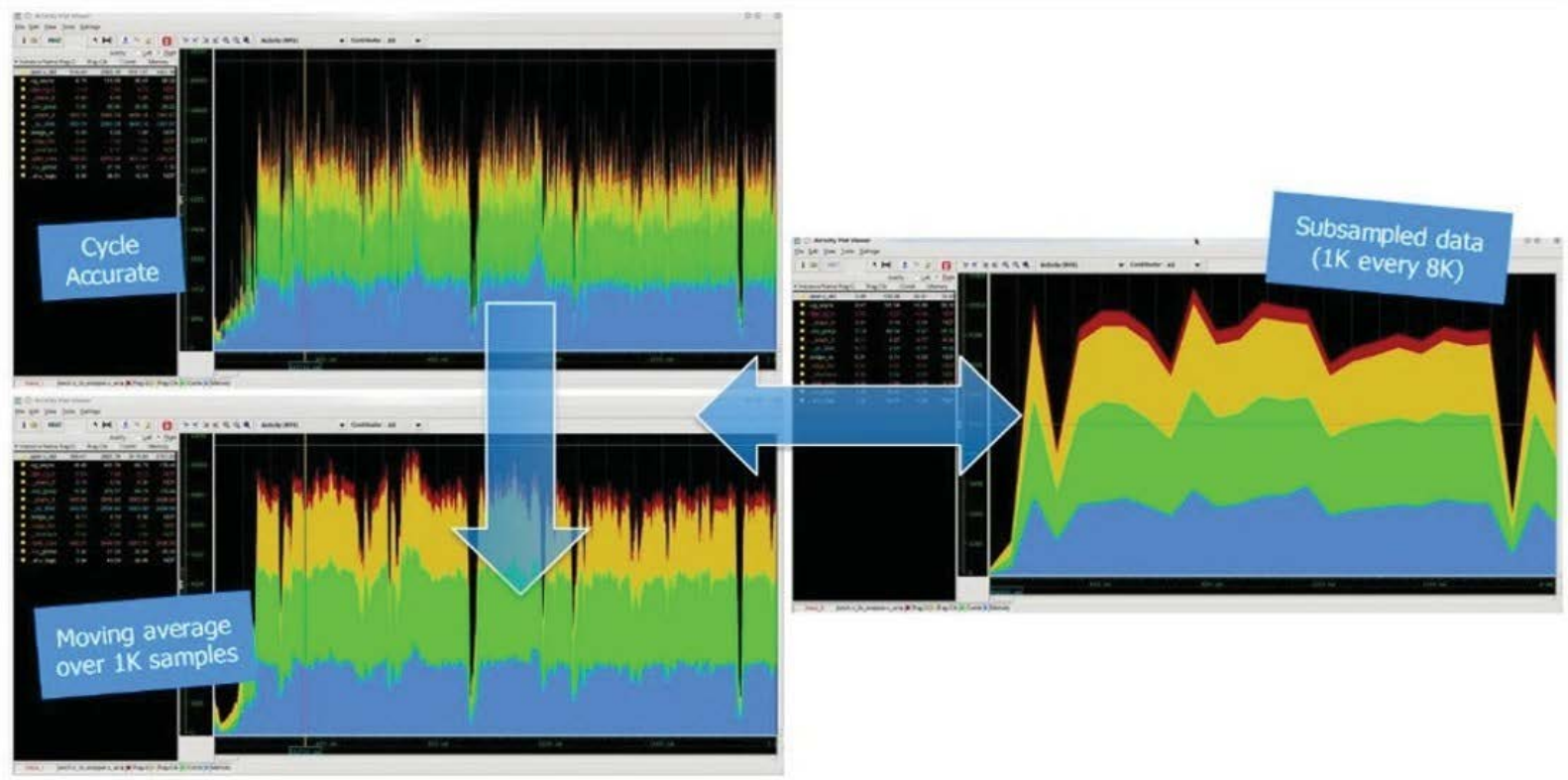
- Activity Plot is not a power estimation but a power analysis tool (using modeling approach)
  - No attempt at generating Watt numbers
- All metrics expressed with regard to a reference that is chosen to be a FF
  - Register Energy Equivalent (REE)
    - Typical **energy** consumed by a register data toggle
  - Register Power Equivalent (RPE)
    - Typical **power** consumed by a register whose data toggles *every ns*
  - Register Area Equivalent (RAE)
    - Typical register **area**
- Solution to provide viable approximation of
  - Power estimation metrics
    - Energy <-> Activity Events Plot (REE)
    - Power <-> Activity Plot (RPE)
    - Power Density <-> Activity Density Plot (RPE/RAE)



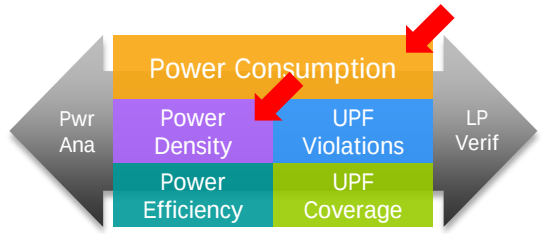
Activity Plot Use Model

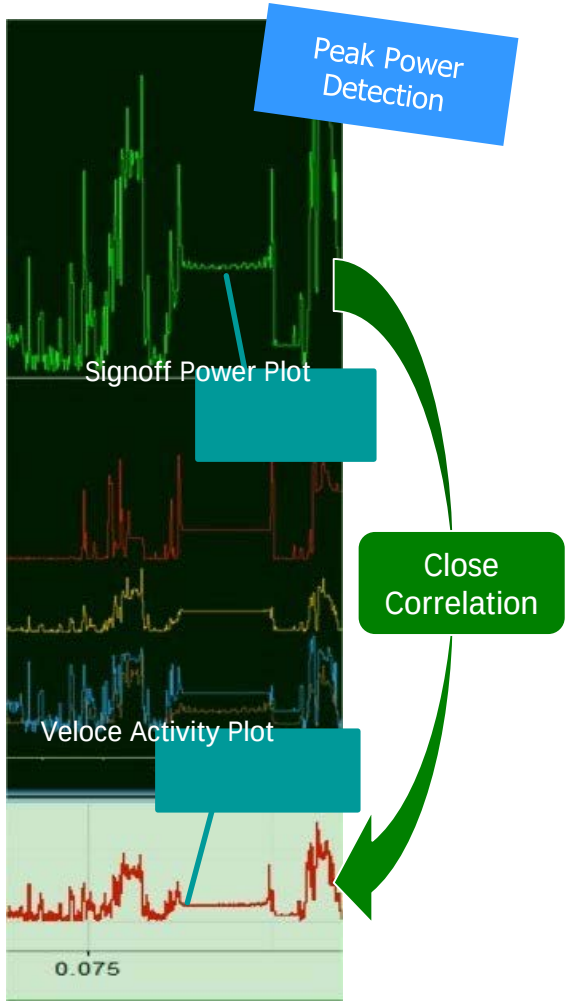


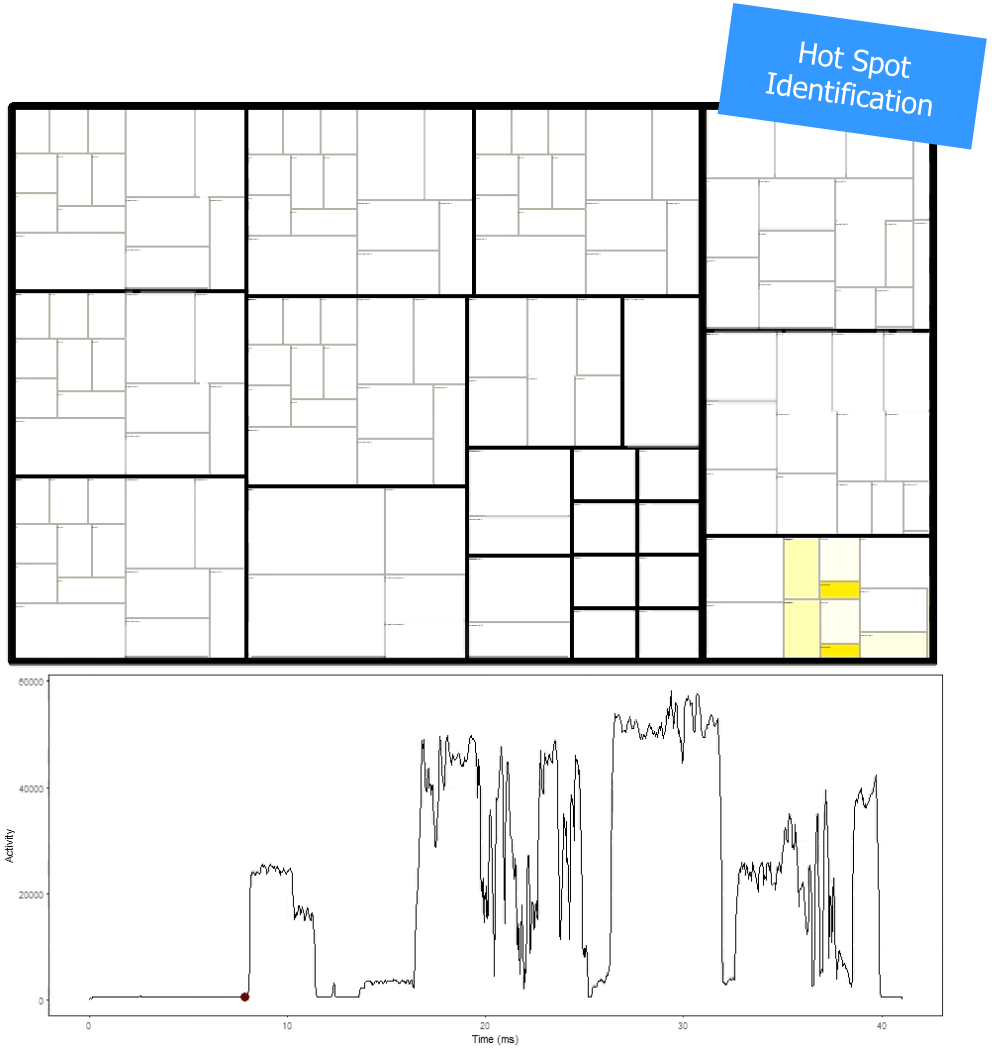
- All contributors broken down (Reg-Q, Reg-Clk, Comb, Mem)
- Multiple views (Power, Energy, Power Density,…)
- Responsive environment to analyze data

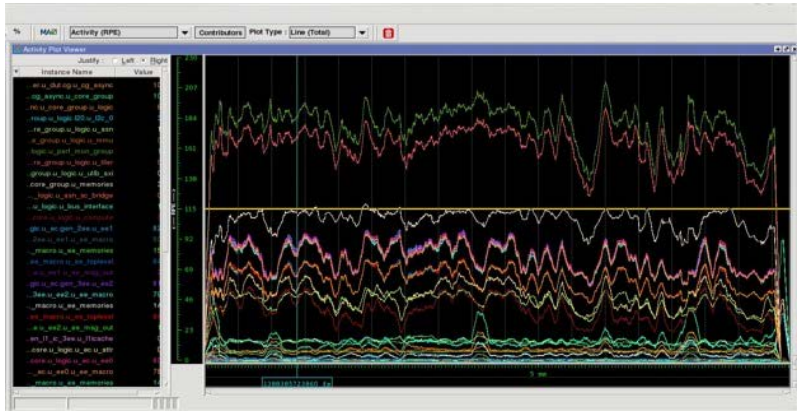


Some Applications…

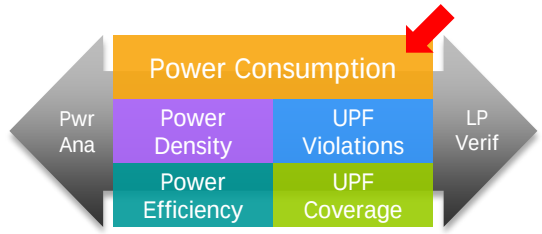






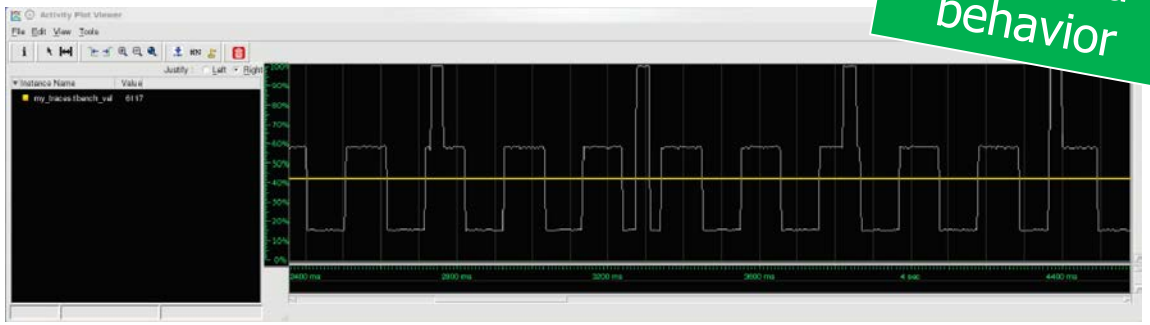


System Activity Validation



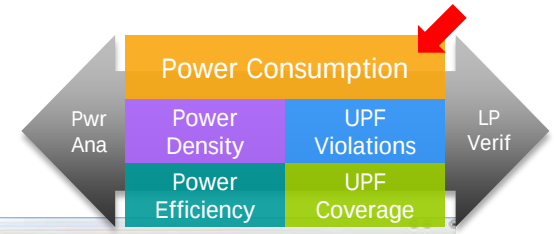
- Arm-based system physical prototype showing excessive power
- Problem reproduced with Activity Plot
  - Periodicity of tasks accelerated to increase probability of failure
  - 2 processes using AXI peripheral A and A&B alternatively
  - Process using peripheral A remains active after a while
  - But why?

Expected behavior

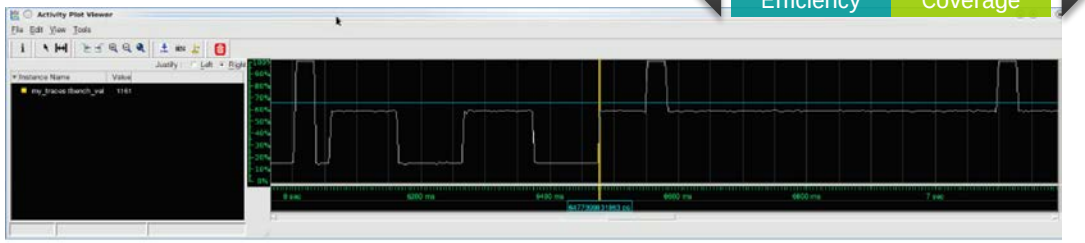


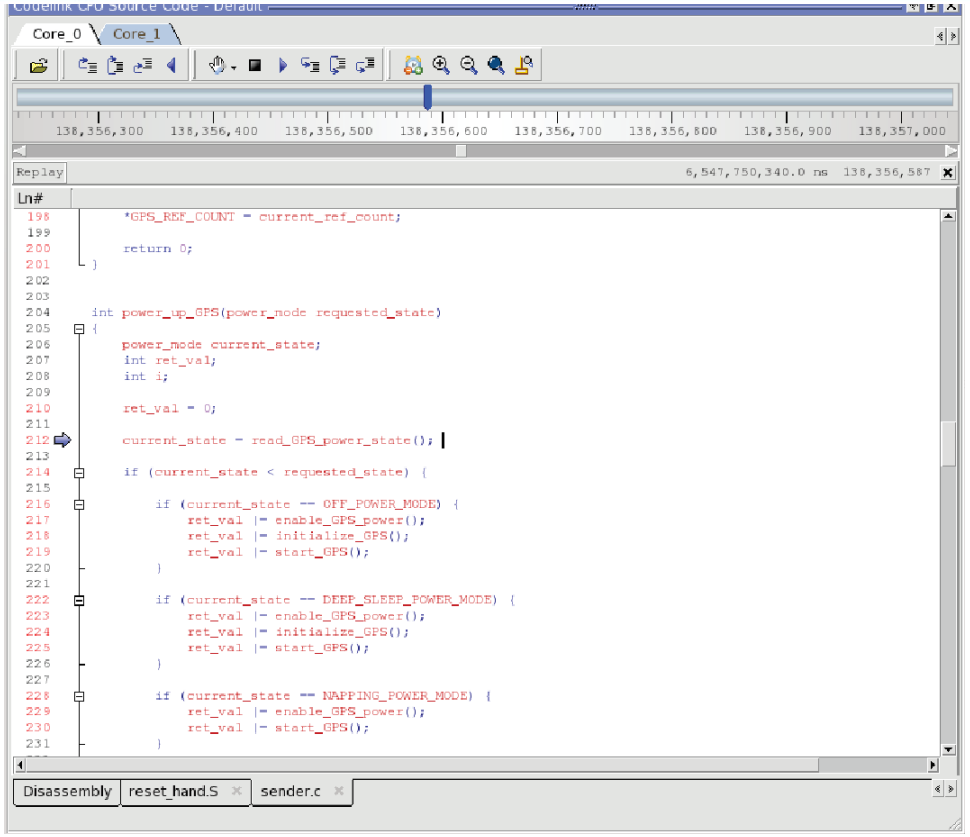
Faulty behavior

Correlation of HW events with SW

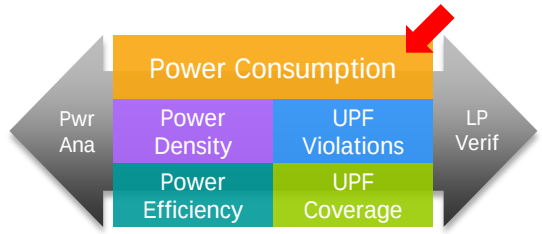


- Non-intrusive debug methodology using code source analyzer
- Traces the activity of the processors as they execute code
- Code source analyzer cursor was set to where the system should have switched off peripheral A
- Root cause of issue traced back to power controller

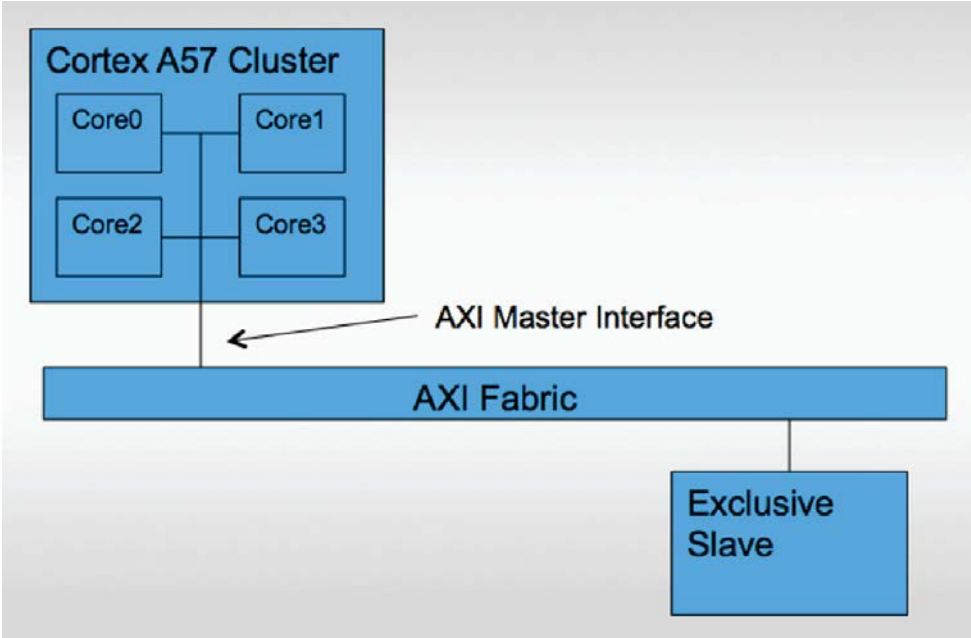




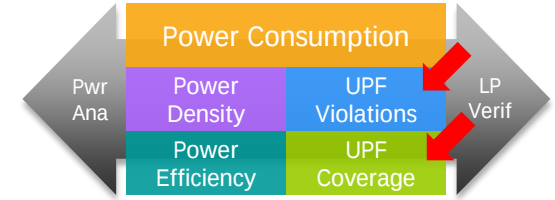
Explanation of Issue

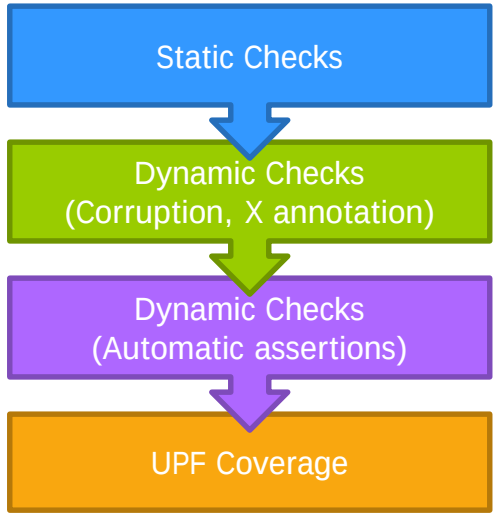


- 2 processes on two different cores trying to simultaneously turn off peripheral A
- Implementation was reliant on simple counter of active processes
  - Both processes read counter (answer = 2)
  - Both decrement to 1 and leave peripheral running
- Not just a simple race condition
  - SW actually uses Arm AXI exclusive access instructions
  - Pb is that all cores were identified with same AXI master bus port
- Fix consisted in added part of the transaction ID to the master identifier to truly differentiate originators



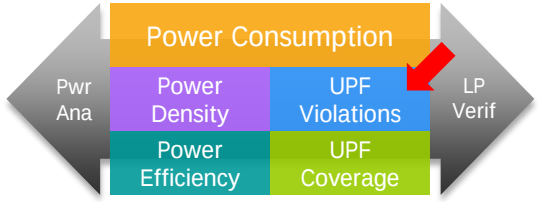
Emulation-based Low-Power Verification



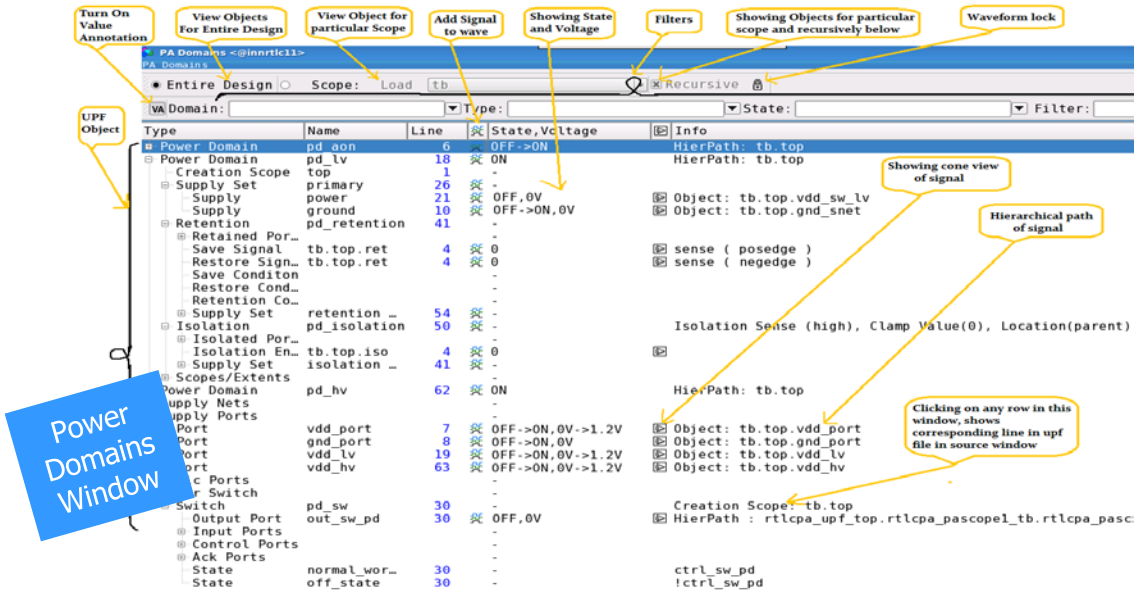


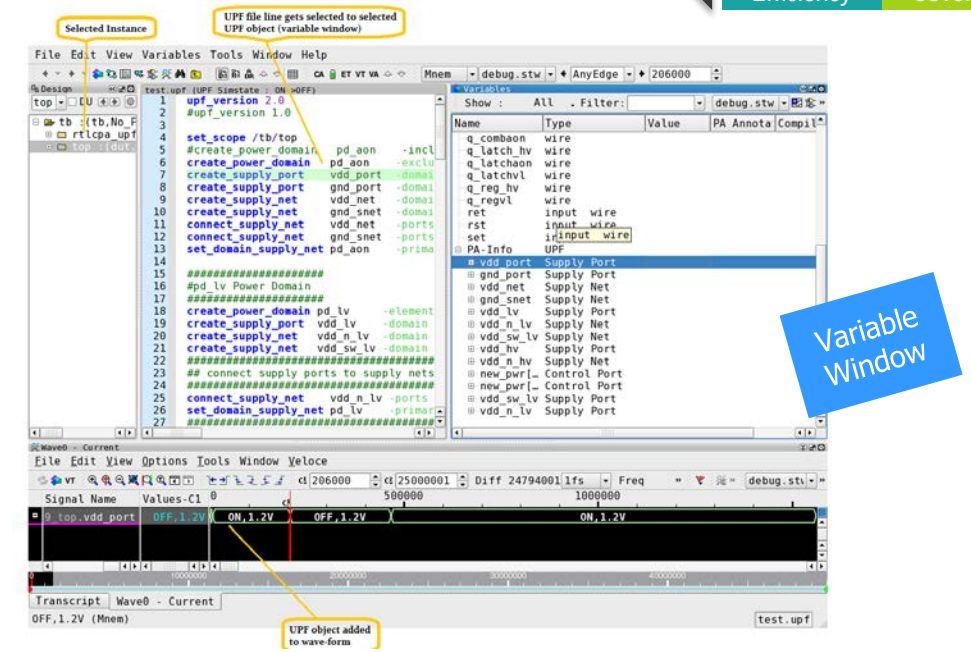
- Types of checks
  - Static, dynamic and coverage checks for realistic scenarios
  - Can address common case where power controller is implemented in SW
- UPF Support
  - UPF 2.x / UPF 3.0 support

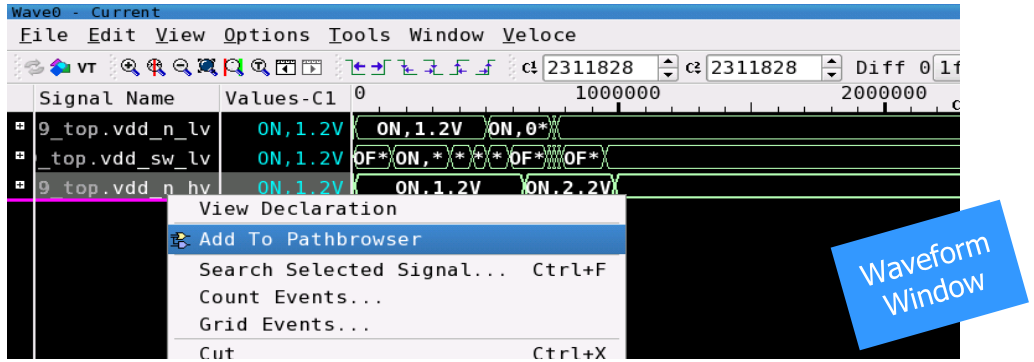
Need For Advanced Debug Capabilities



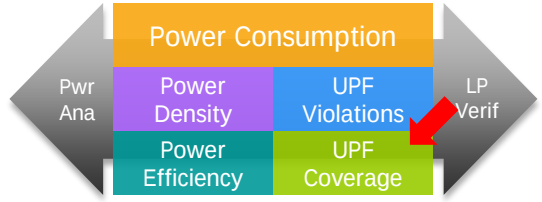
- Debug environment
  - UPF objects are available in the variable window
  - PA Domains to summarize UPF data
  - Waveform window displays supplies and power states, link to PathBrowser







UPF Coverage



- Emulators Allow For Exhaustive Exploration of Power Transition Scenarios
  - Power State Coverage
    - Which states are actually reached and how often
  - Power State Transition Coverage
    - Which of the allowed transitions are actually covered

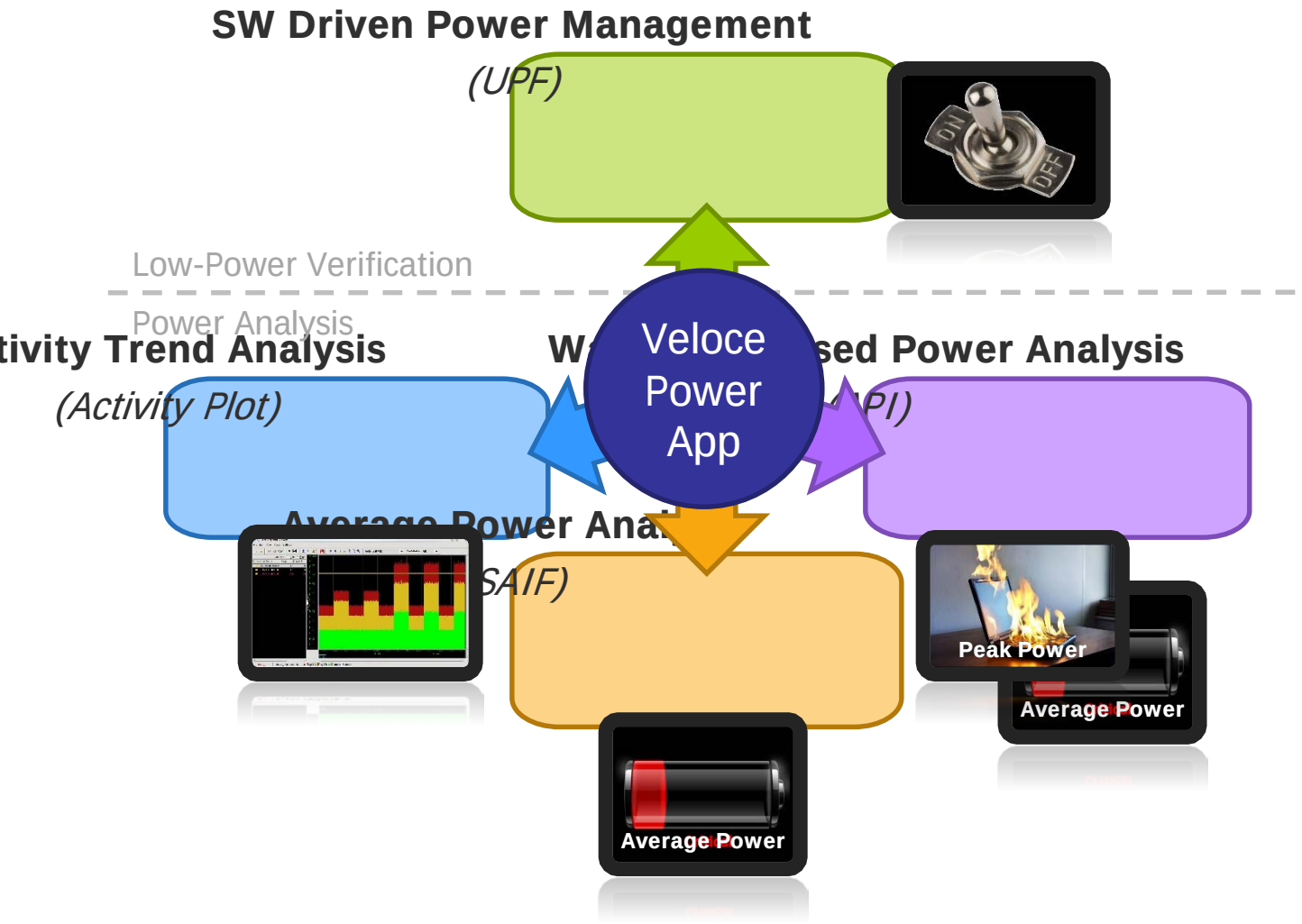
Coverage Stats

UPF OBJECT	Metric	Goal	Status
TYPE : SUPPLY SET /TESTBENCH/tb/TOP/top_PD.primary			
	50.0%	100	Uncovered
SUPPLY SET coverage instance cov6	50.0%	100	Uncovered
Power State ON	100.0%	100	Covered
bin ACTIVE	2	1	Covered
Power State OFF	0.0%	100	ZERO
bin ACTIVE	0	1	ZERO
Power State DEFAULT_NORMAL	100.0%	100	Covered
bin ACTIVE	2	1	Covered
Power State DEFAULT_CORRUPT	0.0%	100	ZERO
bin ACTIVE	0	1	ZERO
TYPE : SUPPLY SET /TESTBENCH/tb/TOP/top_PD.primary			
	16.6%	100	Uncovered
SUPPLY SET coverage instance cov7	16.6%	100	Uncovered
Power State ON_to_OFF	0.0%	100	ZERO
bin ACTIVE	0	1	ZERO
Power State ON_to_DEFAULT_NORMAL	100.0%	100	Covered
bin ACTIVE	31	1	Covered
Power State ON_to_DEFAULT_CORRUPT	0.0%	100	ZERO
bin ACTIVE	0	1	ZERO
Power State OFF_to_ON	0.0%	100	ZERO
bin ACTIVE	0	1	ZERO
Power State OFF_to_DEFAU	0.0%	100	ZERO
bin ACTIVE	0	1	ZERO
Power State OFF_to_DEFAU	0.0%	100	ZERO
bin ACTIVE	0	1	ZERO
Power State DEFAULT_NORMAL_to_ON	100.0%	100	Covered
bin ACTIVE	31	1	Covered
Power State DEFAULT_NORMAL_to_OFF	0.0%	100	ZERO
bin ACTIVE	0	1	ZERO
Power State DEFAULT_NORMAL_to_DEFAULT_CORRUPT	0.0%	100	ZERO
bin ACTIVE	0	1	ZERO
Power State DEFAULT_CORRUPT_to_ON	0.0%	100	ZERO
bin ACTIVE	0	1	ZERO
Power State DEFAULT_CORRUPT_to_OFF	0.0%	100	ZERO
bin ACTIVE	0	1	ZERO
Power State DEFAULT_CORRUPT_to_DEFAULT_NORMAL	0.0%	100	ZERO
bin ACTIVE	0	1	ZERO

Power State Coverage

Power State Transition Coverage

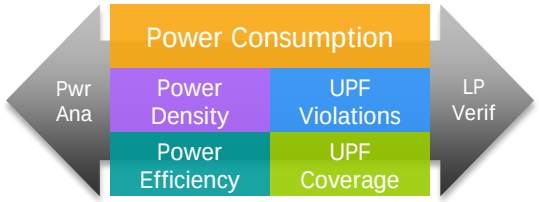
Veloce Power App



Value Proposition

- Capacity & Performance
  - Large SoC handling (RTL/Gate), real applications
  - Orders of magnitude faster than simulation & power tools
- Low-Power Verification at SoC level
  - HW or even SW-based power controller
- Activity Trend Analysis
  - Over time and across scenarios
  - Identification of realistic peaks and hotspots
- Realistic Power Vectors Generation
  - For estimation and reduction

Conclusion



- Emulation to generate **representative system payloads**
- Quick estimation of **power profile** with activity plots
- Ability to **dump power vectors** for power analysis tools
- **Vector streaming** without need for intermediary files
- Correlation of **hardware events with software**

- For more info download whitepapers on Mentor website (www.mentor.com)
  - "Using Emulation for Meaningful Power Analysis"
  - "System Activity Validation"

Thank you