Incomplete Low-Power Verification Flow & the oncoming Internet of Things (IoT) Tsunami!

Neyaz Khan – Maxim Integrated Neyaz.Khan@maximintegrated.com
Kamran Haqqani – Maxim Integrated Kamran.Haqqani@maximintegrated.com

1. Introduction

The *Internet of Things* (IoT) Tsunami is coming. However, despite the progress made in the EDA industry with CPF & UPF over the past few years, when it comes to Low-Power verification, the flow is incomplete.

The oncoming IoT wave has evolved from and being driven by the convergence of contemporary wireless technologies, micro-mechanical systems (MEMS) and the Internet. At the periphery are numerous autonomous, self-connected smart sensors, which are primarily analog in nature. As billions of devices come together – the "things" in IoT will need to be very power efficient. Therein lies the challenge. Current low power (LP) verification methodologies are just starting to address the analog elements in the SoC – which are soon becoming the dominant source of power dissipation. Current UPF/CPF based methodologies focus primarily on digital. Verification methodologies for cells and their behavior have limitations that prevent identification of LP cells beyond the analog boundary. There are huge limitations in both static & dynamic verification of low power. In most cases the current methodologies require analog blocks to be black-boxed out or limit the analysis and verification of low power at the boundary of analog/digital. Quite often, low power cells (e.g. level-shifters, isolation, retention, etc.) are hidden deep behind the analog boundaries, which make them difficult to verify.

Call to Action!

In this paper, the authors will examine the current low-power flow created by the EDA Industry, which is primarily focused on digital, and look closely at how to apply it to include analog components which are present in most modern SoCs. The paper closely examines existing flows, methodologies, tools and use models as applied to the verification of an analog Mixed-Signal SoC (MS-SoC). The example chosen here is an IoT application.

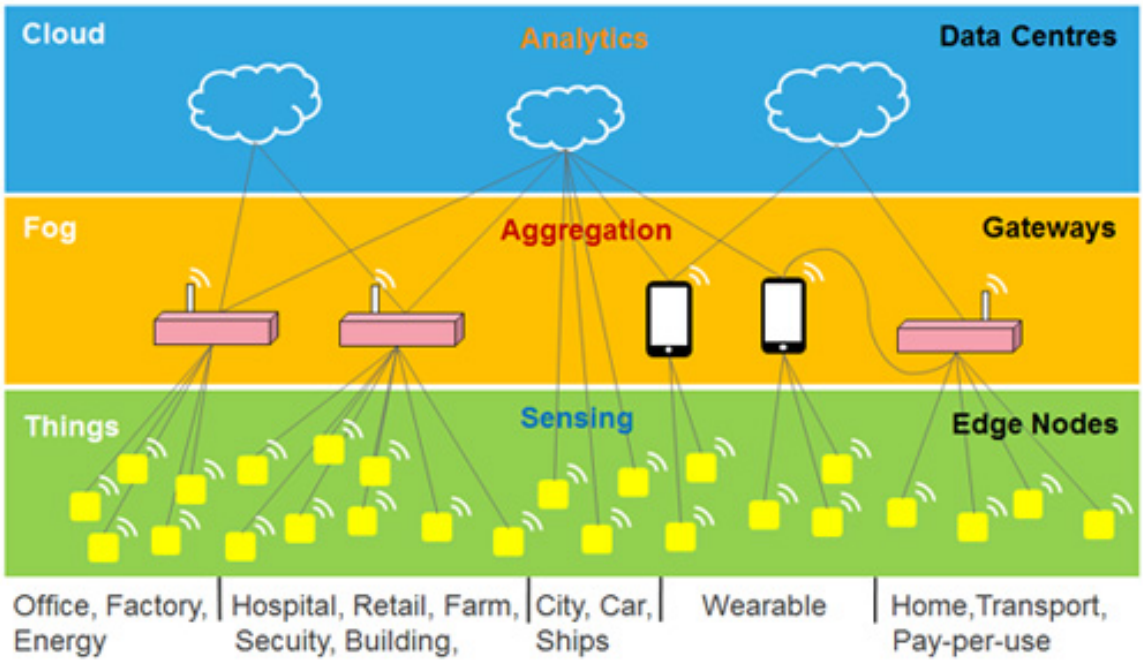
The experience of the authors in going through a complete low-power flow for an MS-SoC is captured here, and shortcomings to existing flows, tools and standards are highlighted as a call to action for the EDA industry.

Let us first take a closer look at the building blocks of IoT before looking at the many low power challenges. The primary focus will be on Dynamic (simulation based) low power Verification Challenges.

2. IoT Background and Architecture

The Internet of Things (IoT), also known as the Internet of Everything (IoE) is the interconnection of a multitude of uniquely identified autonomous computing devices through existing Internet infrastructure. These devices, typically located at the edge of the network, provide a sea of data from sensors, which are consolidated via intelligent networks and processed in the cloud. They also typically contain actuators that control some function – allowing us to seamlessly and effortlessly sense and control the physical word in an automated fashion.

Here is a representation of typical IoT devices working in a target echo system.



Ref: http://community.cadence.com/cadence_blogs_8/b/ii/archive/2014/11/05/mixed-signal-summit-panel-why-iot-design-is-harder-than-it-looks

Figure 1 —Architecture of typical IoT echo-system

2.1 Typical IoT Edge Device Architecture

A majority of connected devices in IoT are nodes located at the edge of the network, often referred to as "last inch" of the network. These last-inch nodes typically perform simple functions and have simple architectures focused on basic data collection, calculation, connectivity and encryption. They consist of low-power MCUs, sensors, actuators and radio. While the sensors may be always-on, the MCU ideally supports multiple power-domains. Power management circuits keep the MCU powered down with long sleep-to-wake ratios. The architecture of the MCU also supports fast wake-up and sleep cycles to conserve power. Other digital logic will be powered down as much as possible, while keeping critical digital functions awake to support the sensor data accumulated over the operating timeframe – for example, DMA and RF circuits will be woken up when buffers get full and supplied by energy harvested from onboard circuitry. These circuits then get powered down as soon as DMA operation is complete.

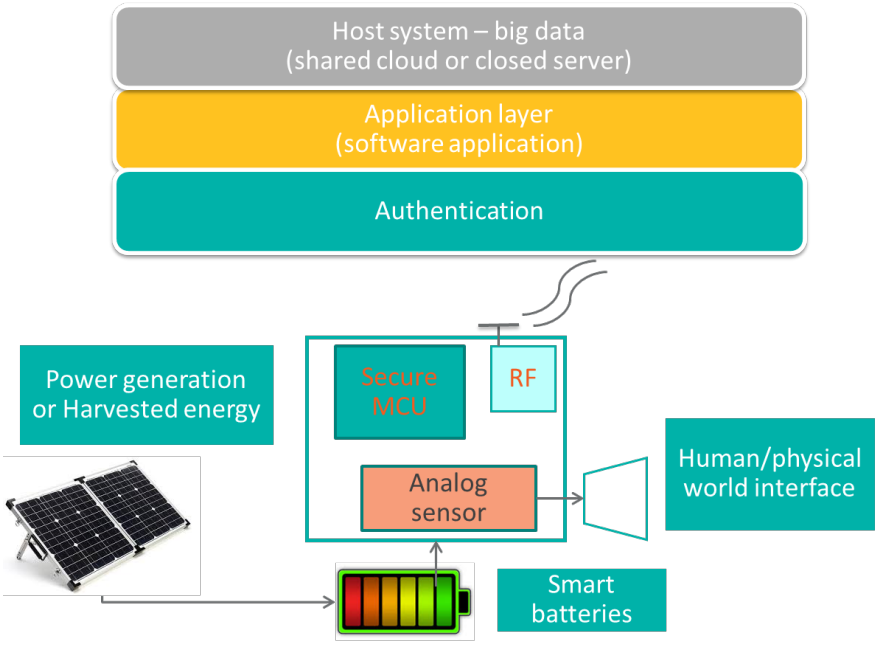


Figure 2—Example of a typical MS-SoC used in an IoT application

2.2 System Level Design Considerations related to Power Management

- Efficient Energy Management
    - Dynamically control power dissipation for target function - manage secure user transactions at the lowest energy per transaction
    - Measure power consumption using analog sensors, and closed-loop control to scale power by controlling analog components (LDOs) – Dynamic Voltage & Frequency Scaling (DVFS)
- System Level architecture and partitioning tradeoffs
    - Software vs. Latency vs. power performance
    - Missing common tools, languages and holistic solution to address early architectural tradeoffs
- Early and accurate system level power estimation
    - Missing tradeoff analysis between hardware and software
    - Quite often the power is managed at the software stage to compensate for hardware issues
- Analog
    - Very difficult to accurately budget and meet power performance for a majority of analog sensors
    - System level power modeling and estimation for always on sea of analog sensors varies over a large range and depends on the operating condition, which is unpredictable at system level

3. Power Management Tradeoffs and Challenges in IoT Devices

The success of IoT requires ubiquitous communication with a sea of remote autonomous devices. These are, primarily MS-SoCs with always-on sensors having digital content that is required to be in deep power conservation mode, with very aggressive energy consumption. It is not uncommon to have edge devices that are required to function on a single battery that is expected to last for a lifetime of the product, or have energy harvesting modes to replenish power.

3.1 Analog Power Consumption

Power dissipation in mixed-signal SoC devices often occurs predominantly in the analog blocks – in some cases approaching 75%. Always-ON IoT devices especially those requiring multiple radios – Bluetooth, GPS, Wi-Fi, 3G & 4G-LTE are some examples of this. This necessitates the need for aggressive power management in the analog including things like power gating of PLLs. On-chip regulators – cascaded LDOs are increasingly being used. While power gating inside analog may be possible, it affects stability and time to reach quiescent state.

3.2 Energy Harvesting

Energy harvesting is used to scavenge minute amounts of ambient energy and convert to electrical energy to typically power low-energy consuming devices such as autonomous sensors. Ambient energy is everywhere – vibration, light and heat are some of the common energy sources that can be harnessed efficiently to drive these low power devices. In practical applications, energy harvesting is used as a secondary source, especially where ambient energy sources are intermittent. A battery back-up is often used as a reliable, secondary source for uninterrupted use. Battery life can be extended dramatically since they are used only when the harvested energy source is absent. Most energy harvesting transducers produce only a few hundred mV, which typically requires the use of on-chip step-up DC/DC voltage convertors.

The introduction of energy harvesting circuitry throws in many challenges to low-power verification at the IC level. Secondary voltage supplies need to be comprehended and captured in golden-power intent files like CPF & UPF. Their effects on traditional low-power features like isolation, level-shifting and retention need to be accounted for in an environment where either of the power sources (battery, or energy harvesting transducers) may not be present, or present simultaneously.

Currently available power format standards and tools do not comprehend and support such features. Homegrown solutions are often used today, which are tedious, and can be error-prone.

Very aggressive power management is a must have for IoT devices. While there are numerous challenges that affect power management at the system level for IoT devices, the focus of the subsequent sections is primarily on the verification of power management in MS-SoC devices.

3.3 Low Power Design & Verification Challenges for MS-SoC Devices

The Figure 3 below shows power-management features and components used in a typical MS-SoC

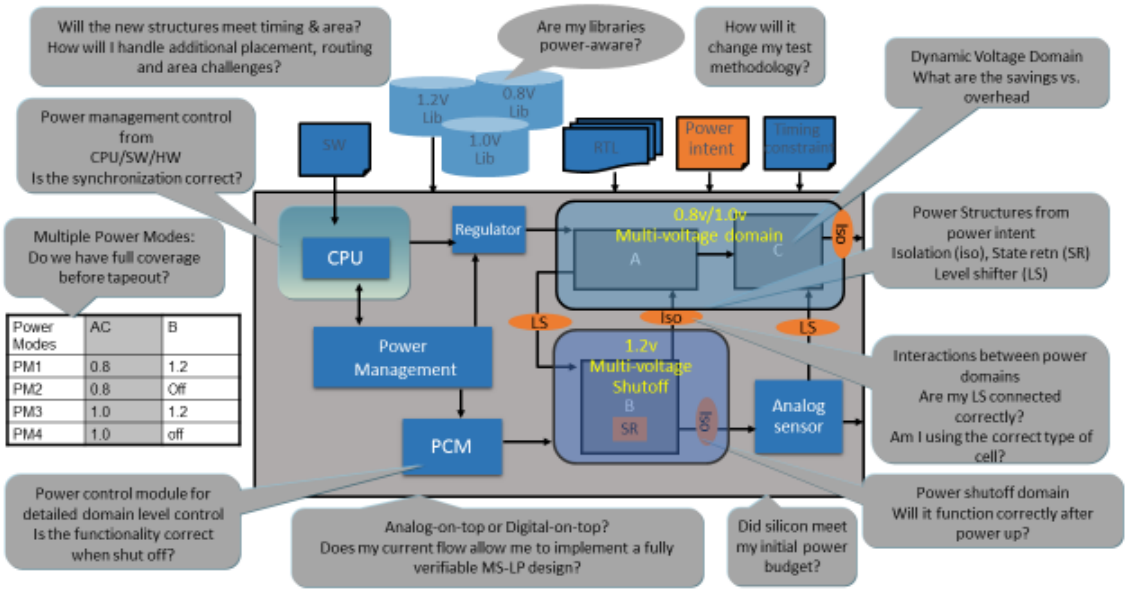


Figure 3— Some Design & Verification challenges in a low power MS-SoC used for an IoT Application

Here are some examples of challenges faced by architects, designers and verification engineers related to power management:

- Are the libraries power-aware?
- How will it change the test methodology?
- Dynamic Voltage Domain
- What are the savings vs. overhead
- Capture Power Structures from power intent to identify
    - Isolation (ISO), State retention (SR)
    - Level shifter (LS), other LP cells
- Interactions between different power/voltage domains
- Are all Level Shifters connected, and powered correctly?
- Are the lib-cells being used of the correct type?
- Power shutoff domains
- Is the functionality correct when domains are shut off?
- Will it function correctly after power up?
- Will final silicon meet the initial power budget?
- Analog-on-top or Digital-on-top?
- Does the current flow allow one to implement a fully verifiable MS-LP design?

- Power control module for detailed voltage-domain level control
- Multiple Power Modes:
    - Do we have full coverage before tapeout?
- Power management control from CPU/Software/Hardware
    - Is the synchronization & power-sequencing correctly executed?
- Will the new power structures meet timing & area?
    - How will I handle additional placement, routing and area challenges?

4. Power Management Verification in Mixed Signal Chips

The following sections examine the use of contemporary low-power verification flow used in the industry using UPF & CPF to capture golden power intent. These power formats and associated tools and methodologies were developed primarily for digital – here we apply it to the power management verification of a typical MS-SoC and examine & report our findings.

UPF/CPF define power intent for digital efficiently. Nominal operating voltages are virtually superimposed on top of RTL /Gates to emulate power down/up behavior on a net that toggles between 0 and a 1. However, Analog simulations model voltage & current accurately over a continuous range.

Some fundamental questions:

- How to connect virtual low-power behavior to actual voltages modeled by analog?
- How to create virtual power/ground blocks to an analog block from UPF/CPF constructs?

4.1 Evolution of Power Formats standards to address Analog

Contemporary power-format standards have started to address the presence of analog in MS-SoCs to capture and analyze power management across A/D domain boundaries. For example, in UPF (1801-2013) one can designate design attributes as type analog by assigning an attribute *UPF_is_macro_cell* to true. Similarly in CPF, *macro_model* can be used to describe some power related information for analog or other multi-rail macro blocks.

UPF based approach to include Analog

```
set_design_attributes -elements {macro_name}
     -attribute {UPF_is_macro_cell TRUE}
```

CPF based approach to include Analog

```
set_macro_model  macro_name
…
power information content
…
end_macro_model  [macro_name]
```

Both approaches treat analog as black-boxes with basic power attributes defined at boundary-pins. This enables both simulation tools (e.g. VCS-NativeLP & IUS) and structural checking tools (Verdi-LP & Conformal-LP) to analyze power connectivity at the boundary level of these cells.

Although, this is a step in the right direction, from a usability point of view, this is very basic, and only addresses a subset of low-power attributes and behavior of the analog. The A/D boundary is where a majority of the low-power bugs in a mixed signal design occur. An example of such a bug would be the presence of level-shifters at the A/D boundary outside the analog, while level-shifting is also being simultaneously performed by the analog itself. Another example would be a power-control signal controlling things like reset, or clock crossing A/D boundaries and being manipulated, or controlled by the same domain that it is intended to power-down or control power for. These "chicken-or-egg-first" scenarios are very difficult to find because support for simulation and structural analysis for such mixed-signal examples is not available in today's low-power flow and standards, which focus either on the analog, or the digital, but rarely take both into consideration at the same time.

4.2 Power Aware Modeling (PAM) for Analog with multi-rail macro

When modeling the power intent for a block with more than one (primary) power rail, it gets quite challenging for tools to determine which part of the module code is covered by which rail. In such a case, a good strategy would be to further modularize or partition the inner working of a functional block or a schematic. Not being able to clearly identify and map parts of the code to the corresponding voltage rails prevents the tools from corrupting the nodes, registers and wires for that specific block during simulation. However the current solutions using UPF & CPF are somewhat limited to boundary level analysis and interpretation.

One of the biggest differences between UPF & CPF is seen when it comes to defining multi-rail macro. In UPF based flow the relationship between power supply and associated port is defined in *liberty-format*, while in CPF based flows, the behavior is defined using *CPF-macros* within the CPF file. CPF provides additional capabilities to allow somewhat of a limited gray box view of power attributes, which help define the power related behavior of analog macros. These allow improved structural and simulation based verification. Here are some examples of using *cpf-macros* to capture low-power behavior of the analog gray-box:

```
create_pad_rule
 set_pad_ports
set_diode_ports
set_analog_ports
set_wire_feedthrough_ports
set_power_source_reference_pin
create_power_domain -name PD11 -boundary_ports { C D }
     -shutoff_condition  pso2 -base_domains PD1 -
     power_source
```

UPF-based liberty flow describes multi-rail macro information and other power related structural information using *pg_pin, pg_type, related_power/ground_pin* and *power_down_function*. Additional power related information can also be described in UPF (e.g. *internal_power*). One can also define structural power models in UPF 1801-2013, which is very similar to CPF-macros. This is a very positive step towards the convergence of the two power-formats. It also gives designers using UPF the ability to define power models in a power intent file, rather than in a separate liberty file, which is usually not modifiable by verification engineers.

UPF-liberty:

```
…
cell (INVD0HVT) {
  pg_pin (VDD) {
      pg_type : primary_power;
      voltage_name : COREVDD1;
    }
  pg_pin (VSS) {
      pg_type : primary_ground;
      voltage_name : COREGND1;
    }
pin(A) {
      related_ground_pin : VSS;
      related_power_pin : VDD;
    }
  pin(Y) {
      power_down_function : "!VDD + VSS";
      related_ground_pin : VSS;
      related_power_pin : VDD;
      }
      internal_power () {
…..
```

Or

UPF-power model:

```
…
begin_power_model upf_model -for cellA
create_power_domain PD1 -elements {top/U1}
-supply {primary}
-supply {mem_array ss.mem}
end_power_model
…
associate_supply_set top_level_SS -handle PD1.primary
…
```

One of the key features missing here is the ability to easily specify the target power budget for these macros for early power estimation at system level. Adding a simple construct here to do so would be a welcome addition to the standards.

Most analog functional models (written using *SystemVerilog real* or *wreal*) are now power aware. What's missing is that the power connectivity information in UPF-liberty or CPF-macro cannot be used directly with the analog functional models

written in *real* or *wreal* (.vams format). This key disconnect prevents verification tools from being able to accurately simulate (mixed signal simulation) in the presence of UPF/CPF. The same is true for formal tools, which are currently unable to read *real, wreal* models of analog functionality together with UPF/CPF. This is a significant gap today.

Another major hindrance is the automatically inserted connect modules between A/D boundaries in simulators today. These are still very primitive, error prone and hard to debug. There is a strong demand and need for a mechanism that can potentially enable simulation tools to automatically detect operating points from analog designs and automatically transition to the corresponding nominal operating voltage on the digital side and connect across domains without relying on these primitive connect modules.

4.3 Missing/Redundant Level Shifters at A/D boundary

The verification of the correct placement and connectivity of low power cells like Level Shifters (and other similar cells) in a mixed-signal SoC is a big challenge. Level shifters are traditionally considered to be an analog function - so quite often this results in the voltage level-shifting behavior to be hidden deep within the analog in the form of a custom cell rather than using a standard level-shifter library cell. During structural ERC stage an engineer has to re-define the power network intent in a tool (e.g. Caliber-PERC) to look for cells with a power differential to catch a missing or incorrect level shifter behavior. This breaks the flow and the reliance on using a single golden power intent file by introducing multiple power-intent files (one using UPF/CPF, and another power network description within Caliber-PERC). In addition, the LP standards also lack the ability to describe ESD as part of the power-intent.

Here are examples of deficiencies in some typical use-modes and scenarios with respect to the presence of level shifters in analog and mixed signal designs:

- If a level-shifter is already present in analog but is a non-standard/custom (non-liberty) cell, the tools will still infer an insert an additional level-shifter based on UPF/CPF description
- If redundant level-shifters are present between the 2 boundaries. Then the inferred cell will fight with the custom analog LS behavior
- At the boundary of real vs. SPICE, level-shifters hiding on both sides can get reduced to a non-electrical behavior by automatically inserted connect modules (in a non-standard/custom cell)
- Digital doesn't cleanly handle multiple drivers and bi-directional paths in the presence of UPF/CPF

Additional challenge occur when proper partitioning guidelines are not followed, which results in the tools not being able to identify the presence, absence or the correct connectivity of a level shifters in a mixed signal SoC.

All the above contribute to the incomplete and sometimes incorrect verification of low-power features across A/D boundaries, which implies missed bugs that effect critical design features like reset, PoR , clock-control, voltage scaling etc.

4.4 Power Domain Compatibility Verification

Currently both UPF & CPF do a good job in defining power domains (very similar), however they differ slightly in how the tools allow the merging of voltage-domains. A typical MS-SoC is put together using several IPs that often come from different sources. If each IP source/designer has the UPF/CPF defined for their block, it becomes quite difficult for a chip integrator to manage the power intent of the full SoC. Several domains have to be merged. A few individual tools from both Synopsys and Cadence do automatically merge the power domains and spit out a new power intent file, however this is not desirable as it changes the initial golden power intent. In case of CPF, one could use –*base_domains* to enforce domain merging

CPF:

```
create_power_domain -name Pd_PSO \
-shutoff_condition !pwr \
-boundary_ports {D2} \
-instances {mem*} \
-base_domains Pd_AON
```

However in the case of UPF, one needs to rely either on the power state table for tools to automatically merge the domains at higher levels or use *supply_sets* to potentially allow domain association at higher levels

UPF:

```
create_power_domain PD1 -elements {top/U1}
-supply {primary}
-supply {mem_array ss.mem}
```

4.5 Power modes and switch control used in a Mixed-Signal device

Below is an example of a typical mixed signal SoC device with multiple power modes. However, quite often in a mixed signal environment the top level (PDcore) can be an analog block, and the power switch can also sit in the analog domain. This can prevent the checking of power control connectivity and simulation of LP behavior accurately using UPF/CPF flow.

	PDcore	**PD_ana1**	**PD2**	**PD3**	**PD4**
PM_Grumpy	v3.6	v3.0	v3.0	v3.0	v3.0
PM_Dopy	v3.6	v3.0	v1.8	v1.8	v3.0
PM_drowsy	v3.0	off	v1.8	v1.8	v1.8
PM_sleepy	v3.0	v3.0	off	off	v1.8
PM_iceT	v3.0	off	off	v1.8	off

Table 1—Power modes of a mixed signal low power design for an IoT system

Currently if one wants to accurately verify both the dynamic and the structural checks including the switching behavior, one would need to convert the table above into UPF power state table or CPF power modes:

UPF (legacy UPF 2.0 and older):

```
create_pst Iot -supplies {PDcore_supply PD_ana1_supply PD2_supply
PD3_supply PD4_supply1}
add_pst_state PM_Grumpy -pst Iot {v3.6 v3.0 v3.0 v3.0 v3.0}
add_pst_state PM_Dopy   -pst Iot {v3.6 v3.0 v1.8 v1.8 v3.0}
add_pst_state PM_drowsy -pst Iot {v3.0 off  v1.8 v1.8 v1.8}
add_pst_state PM_sleepy -pst Iot {v3.0 v3.0 off  off  v1.8}
add_pst_state PM_iceT   -pst Iot {v3.0 off  off  v1.8 off }
```

CPF:

```
create_power_mode -name PM_Grumpy -default \
-domain_conditions {PDcore@v3.6  PD_ana1@v3.0  PD2@v3.0  PD3@v3.0  PD4@v3.0}

create_power_mode -name PM_Dopy \
-domain_conditions {PDcore@v3.6  PD_ana1@v3.0  PD2@v1.8  PD3@v1.8  PD4@v3.0}
create_power_mode -name PM_drowsy \
-domain_conditions {PDcore@v3.0  PD_ana1@off  PD2@v1.8  PD3@v1.8  PD4@v1.8}

create_power_mode -name PM_sleepy \
-domain_conditions {PDcore@v3.0  PD_ana1@3.0  PD2@off  PD3@off PD4@v1.8}

create_power_mode -name PM_iceT \
-domain_conditions {PDcore@v3.0  PD_ana1@off  PD2@off  PD3@v1.8  PD4@off}
```

However, the above power modes have to be tied to a control signal in UPF: *create_power_switch* (UPF) or *–shut_off* condition (CPF). Although robust, this approach is very cumbersome, error prone and indirect way of controlling the power behavior of a device, and hence, not very efficient.

In a complex system, the verification goals require that the simulation of low power operational mode be controlled by software. There is a need to directly control power-behavior of the device from firmware, either directly from the software protocol, or the applications layer of an operating system. It would be desirable and very efficient to allow the high-level software transactions for power-management to directly control and manipulate the power control signals that influence and control the power-formats.

Such a feature is completely missing in both UPF & CPF formats. It would be very useful to add to power-format standards like UPF & CPF.

4.6 Structural & Functional Verification of Low-Power structures using macromodels

As described in the previous section, *CPF-macro* could be used to define an analog block's power behavior. This method could be used to verify connectivity at the boundary of an analog block against the rest of the system. In this particular example below we can also use *CPF-macro* to describe a power source, although it is currently limited to an LDO type behavior only in the standards.

```
set_macro_model   regulator
create_power_domain -name  PDVOUT -default  -base_domains {PDVIN} -power_source
...
create_nominal_condition -name LDO_range  -voltage 1.1  -pmos_bias_voltage {1.1 1.3}
...
create_power_mode -name PM  -default  -domain_conditions \
{ PDREF@REF  PDVIN@HVDD  PDVOUT@LDO_range }
..
set_power_source_reference_pin  AVDD -domain PDVOUT  voltage_range 1.0:1.1
...
end_macro_model  regulator
```

Unfortunately such information cannot be combined with low-power analysis performed by other tools due to lack of a common database that can be used by all IC-verification tools. For example if one wants to use this information with Caliber-PERC, there is no way to read in the same power intent file (CPF or UPF) in that tool. One possible path is to use OA (openAccess) database, however the simulation tools (like IUS or VCS) don't read that information and have a separate and different database of their own.

There is a strong need for common standard constructs like *CPF-macro* to be used by all front-end and back-end tools from different vendors. This interoperability using a common database format would really help enhance productivity and is highly desirable.

Additional constructs in UPF/CPF should be created to support the needs of power analysis for features like energy harvesting and wireless charging.

4.7 Verification of Low-Power for Analog-on-Top type devices

In a mixed-signal SoC, quite often the top level of the design could be put together as a schematic, *real, wreal* models (*.vams*) or *SPICE*. This sub-block or sub-system could have multiple standalone digital components. Current flows with power intent file don't adequately support *.vams* or *SPICE* at the top level. It is highly recommended that the top level be digital. This allows us to take advantage of a mature set of automation with UPF/CPF. But in certain relatively less complex, analog heavy applications the traditional flow is to have the top-level as analog. Analog on top require strict partitioning guidelines and automation (synthesis, timing analysis, dft, etc.) is not very mature and cumbersome. This continues to be an ongoing challenge.

UPF & CPF language and tools need to be extended to support analog-on-top methodologies for mixed-signal applications like IoT.

4.8 Design, Verification Databases and libraries for LP-Verification

The use of design databases is very fragmented for a MS-SoC flow. Digital front-end design tools use a separate design database than simulation tools. The back-end and layout tools have their own formats for physical designs (Encounter *db, DEF, etc*.). On the analog side the database and libraries used are different from the digital side. This is very error-prone as multiple verification tools throughout the flow are required to read different databases, each containing a different set of information. Checking against libraries for front end and backend also creates a big hurdle throughout the design flow.

Currently OA (openAccess) and milkyway databases do allow capabilities to include such information which could be used by front-end and back-end verification tools (simulation, structural and formal) to verify low power connectivity. However they need to work in conjunction with UPF/CPF and analog design and physical information (OLA). This can allow tools the capabilities of checking for connectivity across domain and analog boundaries.

4.9 Assertion Based Temporal Checks Across Analog/Digital Boundaries

Current CPF and UPF versions allow assertions to be added in a digital RTL centric way:

CPF:
```
create_assertion_control -name AC_1 \
-domains {PDau PDlu PDalu} -type {reset}
```

UPF:
```
bind_checker chk_p_clks
-module assert_partial_clk
-bind_to design_block
-ports {{prt1 clknet2} {port3 net4}}
```

However it is quite a challenge to make it work for some data-types used for analog verification like real or electrical. The standards need to be enhanced to support electrical (voltage/current/temperature) to make verification exhaustive. This can substantially reduce the burden associated with analog modeling of analog devices as is currently being done to verify an MS-SoC.

4.10 Connectivity Checks across Analog/Digital Boundaries

One challenge every chip integrator and verification engineer faces is the limitation of connectivity checks when going across mixed signal boundaries and power domains. However, as described in the previous section (section 4.9) this is dependent on a common design database used against a golden power intent file. Having these connectivity checks to verify the power/ground connectivity for all the cells in the design across the digital/analog boundaries against a *golden power intent* is a must.

5. What is Missing in Existing UPF/CPF based Approaches Today?

Power formats (UPF&CPF) have enabled several things in an MS-SoC. For example

- Creation of golden power architecture reference for structural verification
    - This is the best thing that happened to bring front-end, back-end, software and analog designers together and have them talking to verification engineers
- Physical and logical partitioning of power islands
    - Partitioning for power islands based on power rails allowed a better partitioning between analog and digital domains
- Automatic insertion of power cells on the digital side
    - This enables synthesis, simulation and formal tools to automatically check for power behavior against a golden power intent

However there are several challenges faced by both UPF & CPF today for the effective verification of MS-SoCs

- UPF/CPF don't allow top level design files like *.vams*, supporting analog, *SPICE* or schematic (non-verilog or non-SystemVerilog)
- IC verification, circuit verification and sign-off (e.g. ERC) tools don't all read UPF/CPF to check against a golden power intent
- Power estimation of analog blocks from common design database and library cells are fragmented

6. A call to action for EDA Companies

There are already too many protocols and standards out there for design and verification to successfully take place. We are lucky to be at a point where the power standards (UPF & CPF) have started to converge from digital only to include analog. However they are too limited when it comes to software commands passed through multiple protocols down to domain control (refer to section: 4.5).

Here are some of the recommendations for UPF/CPF committees and EDA vendors:

- Do not create any new standards, instead enhance & expand current standards
- Continue convergence of advanced CPF constructs into UPF to enable Mixed Signal behavior
    - Enable analog models to be verified against golden UPF for structural, functional and LP verification
    - Enable checking of shared LP cells across analog/digital boundaries (e.g. enable level shifters, LDOs)
    - Accurate modeling and checking of power sources within analog (e.g. LDO)
    - Power closure to include analog against a golden-power intent
- Think software based verification
    - Automatically map system level operating modes to power modes/power state tables at the atomic level for easier verification and connectivity
- Encourage use of common design database for analog and digital blocks
    - This includes structural, logical and physical information
- Enable all low power verification tools to read physical design database
    - For accurate power budget numbers from system level
    - Converge DRC checking and other IC-verification/circuit-checking/sign-off tools with structural LP cell checking
    - IR drop and power estimates early in the verification cycle to take into account accurate power switch and delay behavior. This could enable accurate checking of resets, clocks and other LP control. It also enables early power architectural tradeoffs based on operating voltages and noise margins
- Interoperability of database
    - Allow tools to accurately meet power budget across digital and analog boundaries (DEF, OpenAccess, Milkyway, etc.)
- Converge power information between libraries: liberty, LEF, and PDK
    - This is a must to accurately verify logically, structurally and physically while keeping the power budget
- Allow tools to suggest automatic partitioning of cells
    - Based on power consumption, cell size (physical for analog), and timing

- Have tools and standards for battery models to accurately estimate battery lifetime based on the SoC power dissipation
- Allow harvested energy models to be read in conjunction with power dissipation
    - Dynamic trade-off of energy dissipation vs. harvesting/re-generation to enable autonomous systems
- Allow support for top level analog schematic, *SPICE* or *.vams* file
    - Analog on top flow

7. Conclusion

To keep up with the oncoming tsunami of IoT based mixed signal SoCs, the low power standards have to converge, and be enhanced and expanded. Interoperability of databases and libraries is a must. UPF and CPF have similarities and differences and the convergence of the two into IEEE 1801-2013 was a huge step forward, however there are still several things that need to be overcome in order to enable a holistic low power flow from top to bottom for mixed signal devices. This is necessary to verify and meet the aggressive low power goals like IoT devices.

8. References:

- IEEE 1801-2013 UPF standard
- CPF Si2 Standard 2.1
- Cadence low power simulation guide
- Synopsys low power reference guide