Unique Verification Case Studies of Low Power Mixed Signal Chips

Venkatesh Ranga[1] Madhusudhan Subramanya[2] Anand Shirwal[3] Pramod Rajan K S[4]
Jeff Goswick[5]

[1]Architect, NXP Semiconductors - Bangalore, India, venkatesh.ranga@nxp.com
[2]Architect, NXP Semiconductors – Bangalore, India, madhusudhan.subramanya@nxp.com
[3]Department Manager, NXP Semiconductors - Bangalore, India, anand.shirwal_2@nxp.com
[4]Senior Design Engineer, NXP Semiconductors – Bangalore, India, pramod.rajan@nxp.com
[5] Principle Design Engineer, NXP Semiconductors – Chandler AZ, USA, jeff.goswick@nxp.com

***Abstract*- The increasing demand of internet of things (IoT) products has shifted the focus of the semiconductor industry from the big system on chips to low power mixed signal chips. Power management is a critical requirement for these products. Verifying the complex power structure presents a plethora of challenges. This paper outlines 4 different case studies which expound the in-house solutions developed for enhancing low power mixed signal chip verification. These solutions proved important for ensuring bug free silicon and thereby meeting the time to market targets.**

I. INTRODUCTION

Internet of Things (IoT) is a fast growing application domain and is a key driver in taking the semiconductor industry revenue to over 400 billion dollars by 2020 as shown in Figure 1. The creation of a smarter world is aligned with the key mega trends of today's society such as healthcare, connected devices, security, and energy efficiency.

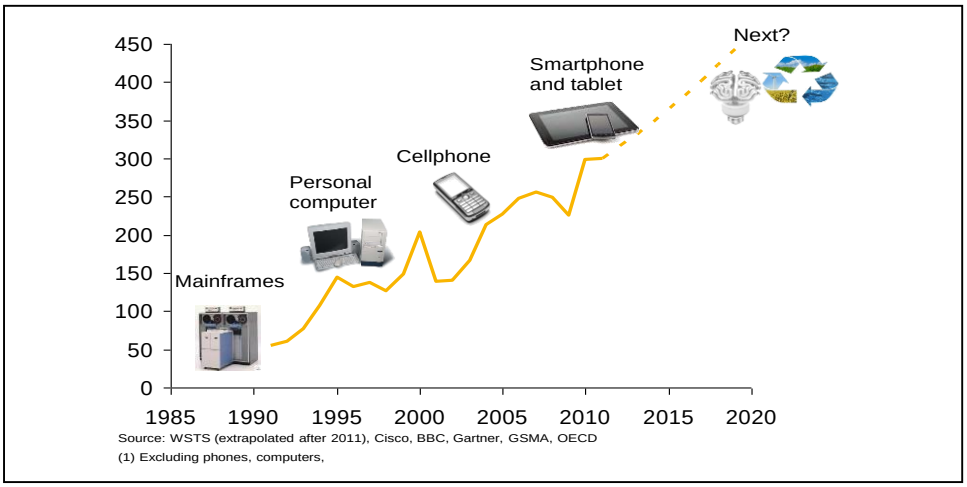

Figure 1. Semiconductor Industry Revenue (in billion US$)

From the chip design perspective, the IoT application area moves the focus away from the big system on chip (SOC) to the mixed signal world (Figure 2). These chips have highly integrated analog and digital blocks and are optimized for low power. Low power design techniques such as power gating, multiple threshold voltages, voltage scaling, and back biasing are extensively used. Engineers have to perform exhaustive verification that covers all possible analog and digital interactions for all the supported power modes and domains.

In addition to the typical digital chip verification flow, low-power mixed signal chip verification includes assertions for low power checks, Common Power Format (CPF) verification flow, and functional checks to verify all the power modes and conditions for the different domains. The complete verification database is passed through different levels of simulation with increasingly mature analog behavioral models, using Verilog, then Verilog wreal, and then Verilog AMS (analog-mixed signal).

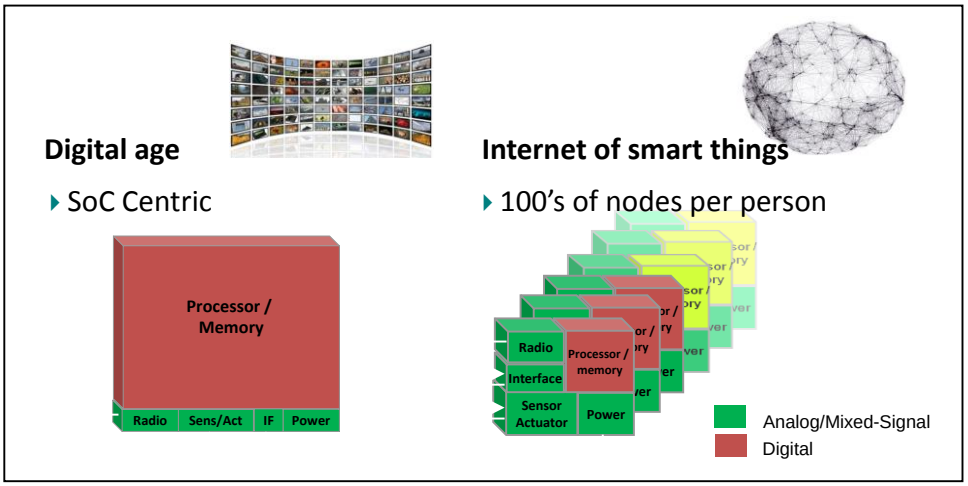

Figure 2. Moving Away from digital centric designs

II. Real Life Challenges Encountered in Low Power Mixed Signal SoC Verification

These are some of the unique challenges encountered during the verification process of induced power (e.g. NFC, RFID, etc.) mixed signal chips:

- Modeling of induced power devices in CPF and verification environment implications
- Analog behavioral modeling for verification of complex power management systems
- Self-Checking test bench environment and formal checks for connectivity
- Low drop out (LDO) sign-off criteria

A. Modeling of Induced Power Devices in CPF and Verification Environment Implications.

Near field communication (NFC) is a short-range wireless technology and traces its roots back to radio-frequency identification (RFID). NFC always involves a reader and a tag. The reader actively generates an RF field that can power a tag for identification, tracking and authentication, etc. (i.e. tagged). Initially the tag is in off state (i.e. untagged). It becomes powered when it moves into the vicinity of a reader and is thus enabled for communication. The IoT product under discussion has a NFC tag as a key component, which also acts as an alternate power source for the chip when in the "tagged" state. The key challenge was to model the NFC power domain and verify its structure using CPF.

At a high level of abstraction the chip consists of a power management unit (PMU), a digital core, an analog block, and an RFID. The block diagram of the RFID/NFC enabled IoT product is shown in Figure 3 depicting the power domains. The incoming RF energy is rectified by antennas LA and LB and powers the analog interface and control logic of the RFID block. There were four major power domains: RFID, always-on, digital, and analog. Power flow from the RFID domain to the internal domains is controlled by the PMU.

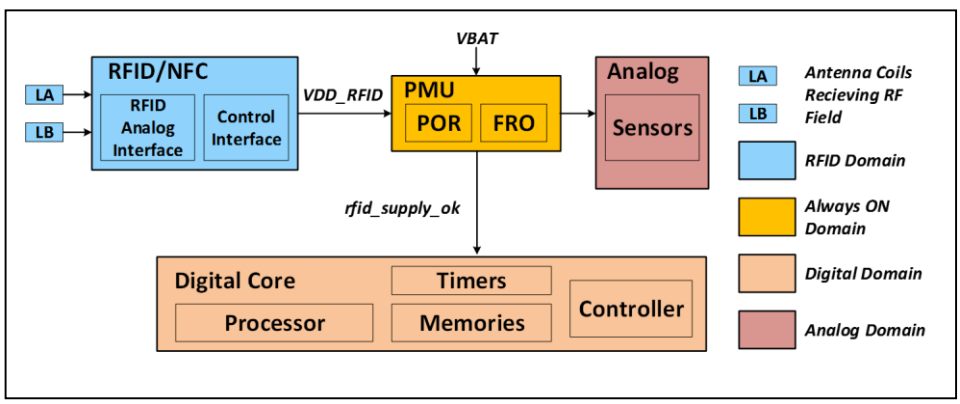

Figure 3. Power Flow Diagram

The NFC device resides in the RFID power domain and the PMU in always on domain. The first challenge was to model the RFID Power Domain as switchable. Typically a switchable domain has a shut off condition generated by the always on domain. In this case the shut off condition for the RFID was defined in the CPF as the rfid_supply_ok = 0. But this signal is a function of VDD_RFID which goes to "X" when the device is untagged, so rfid_supply_ok goes to "X" as well. Once this happens the simulator keeps the RFID domain in an unknown state because the shutoff condition is unknown.

To fix this, the RFID domain was instead described in the CPF as an externally controlled switchable domain using the "-external_controlled_shutoff" option. This modeled the self-powered, switchable aspect of RFID domain as intended for synthesis and formal verification, but in simulation effectively modeled the RFID domain as an always on domain. As a result we missed the situation wherein the RFID is shut off, creating a hole in verifying the isolation logic on signals going from RFID domain to other domains as illustrated in Figure 4.

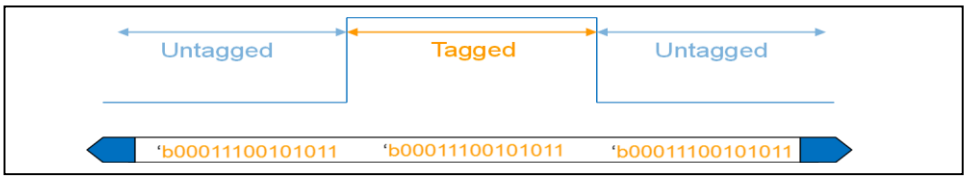

Figure 4. RFID Tagged Event

The issue was overcome by modifying the test bench to corrupt the signals in the switched off RFID domain when an untagged state was detected as shown in Figure 5, ensuring verification completeness.

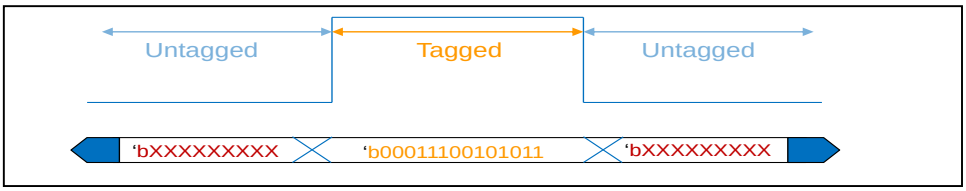

Figure 5. Testbench controlled RFID Tagged Event

B. Analog Behavioral Modeling for Verification of Complex Power Management Systems

Analog models are a critical requirement to start the mixed signal chip verification. Analog models of different abstraction levels are used through the project life cycle. The verification activity started with a high level System Verilog (SV) model then moved on to Verilog wreal models. Once the analog design was completed, the wreal models were replaced with Verilog AMS models for critical use cases. This technique was employed for the power management unit (PMU).

As expected an increase in runtime for the different levels of abstraction with the increasing levels of accuracy was observed similar to the depiction in Figure 6 [2].

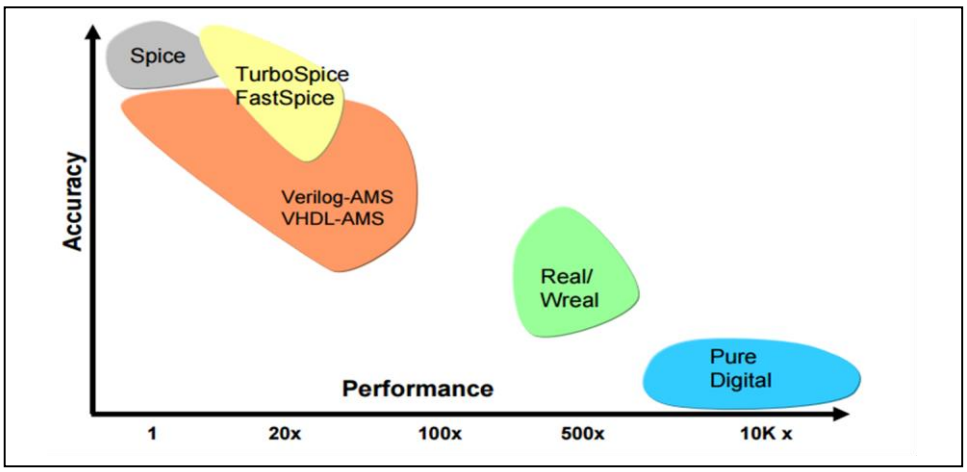

Figure 6. Runtime and Accuracy

The power up sequence of the IC depends on the healthy condition of the BOD. The device is turned "on" only when the supply voltage is above the BOD threshold value. When the supply falls below the threshold value, the device takes various actions based on many criteria: the amount of time it was below the threshold, the time taken for it to cross the threshold value, and the spiky nature of the signal. Figure 7 illustrates the SV modeled real time behavior of the BOD signal. The proposed approach can be employed for most of the signals crossing between the analog and digital blocks.

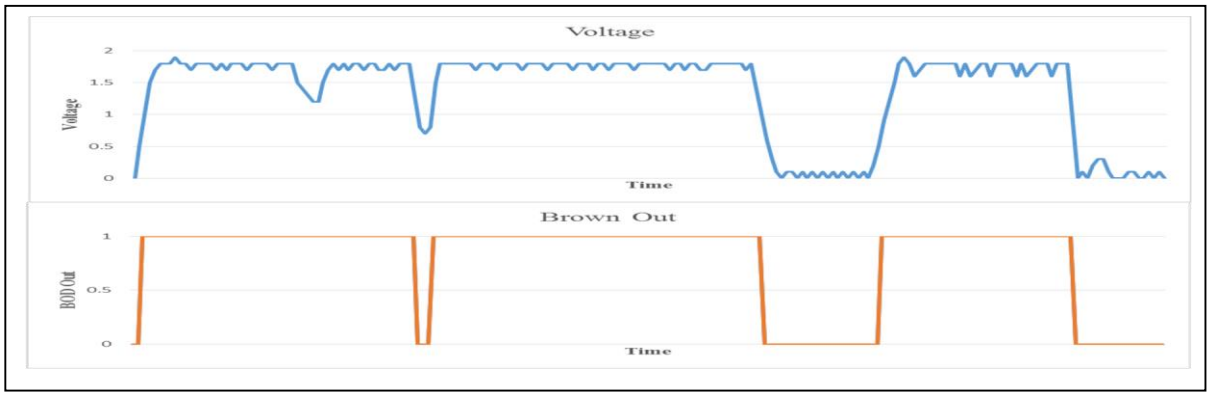

Figure 7. Brown Out Signal for Varying Voltage Levels

a) BOD Modeling in SV

Usually analog behavior is modeled with the fixed delay values. Modeling in SV gives us an advantage, allowing randomization of the analog parameters which better matches reality.

```
initial
begin
   a2p_bod_ldo_dig=1;
   bod_ldo_dig=1; // by default no BOD
   forever
   begin
      randomize(randDelayBodLdoDigHigh) with {randDelayBodLdoDigHigh>=10;
                                              randDelayBodLdoDigHigh<=50;};
      repeat (randDelayBodLdoDigHigh) @(posedge a2p_ulpo_clk_400khz);
      randomize(bod_ldo_dig);
      randomize(randDelayBodLdoDigLow) with {randDelayBodLdoDigLow>=10;
                                             randDelayBodLdoDigLow<=20;};
      repeat (randDelayBodLdoDigLow) @(posedge a2p_ulpo_clk_400khz);
      bod_ldo_dig = 1;
   end
end

// by default indicate as if voltages are at OK level
always @(*)
begin
   if(p2a_bod_ldo_dig_enable)
      a2p_bod_ldo_dig = p2a_bod_ldo_dig_latch_reset ? 1 : bod_ldo_dig;
   else
      a2p_bod_ldo_dig = 1'b1;
end
```

b) BOD Modeling in Wreal

Wreal modeling has a number of speed advantages over spice and AMS models. It allows real number values to propagate on Verilog wires and ports so can model complex analog behavior, but is fast because it only needs the digital simulation engine. It also only requires a digital simulator license.

An ideal model for the BOD include the trip point and hysteresis behavior:

```
reg vdd_dig_p, vdd_dig_hyst;
reg vdig_hold;
always @( posedge p2a_bod_ldo_dig_latch_reset) vdig_hold = 1;
always @( posedge (a2a_bod_vdig & !p2a_bod_ldo_dig_latch_reset)) vdig_hold = 0;
// vdd_dig_hyst  = rising supply
always @( vdd_dig ) begin
   vdd_dig_hyst = vdd_dig > vdd_dig_p ? 1:0;
   vdd_dig_p    = vdd_dig;
end
assign a2a_bod_vdig      =   !p2a_bod_ldo_dig_enable ? 1'b0 : (pmu_bod_ldo_dig ? (
                                                (vdd_dig > 0.495 + p2a_bod_ldo_dig_thr_set * 0.025 +
                                                vdd_dig_hyst * p2a_bod_hyst_set * 0.0165) ? 1'b0 : 1'b1) : 1'bZ);
// Still need extra assigment for a2p_bod_ldo_dig (for startup) as vdig_hold is edge triggered
assign a2p_bod_ldo_dig   =  p2a_bod_ldo_dig_latch_reset ? 1'b1 : (a2a_bod_vdig ? 1'b0 : vdig_hold);
assign pmu_bod_ldo_dig  = p2a_bod_ldo_dig_enable & pmu_bias & pmu_bgp;
```

A more complex model can be created from the bottom up by using a schematic capture and netlisting tool (e.g. Cadence Virtuoso). The analog design can be hierarchically netlisted, then functional models are written for the low level sub modules. This has a number of benefits. The top level model is simply netlisted, so easy to create and the ort names, directions, bus widths, etc. are automatically correct. The leaf cell behavioral models are simpler, easier to write, not as error prone. Top level model is largely correct by construction. Unexpected behavior can be found due to the interaction of the lower level models that often points to real analog IP problems.

For example, the BOD's resistor divider, trim tap mux, and trip point tap mux blocks can be accurately modeled using Verilog wreal. The subcircuit schematic is shown in figure 8 below. This includes the trim select block on the left, the resistor ladder / divider on the top right, and the divider tap select mux on the bottom.

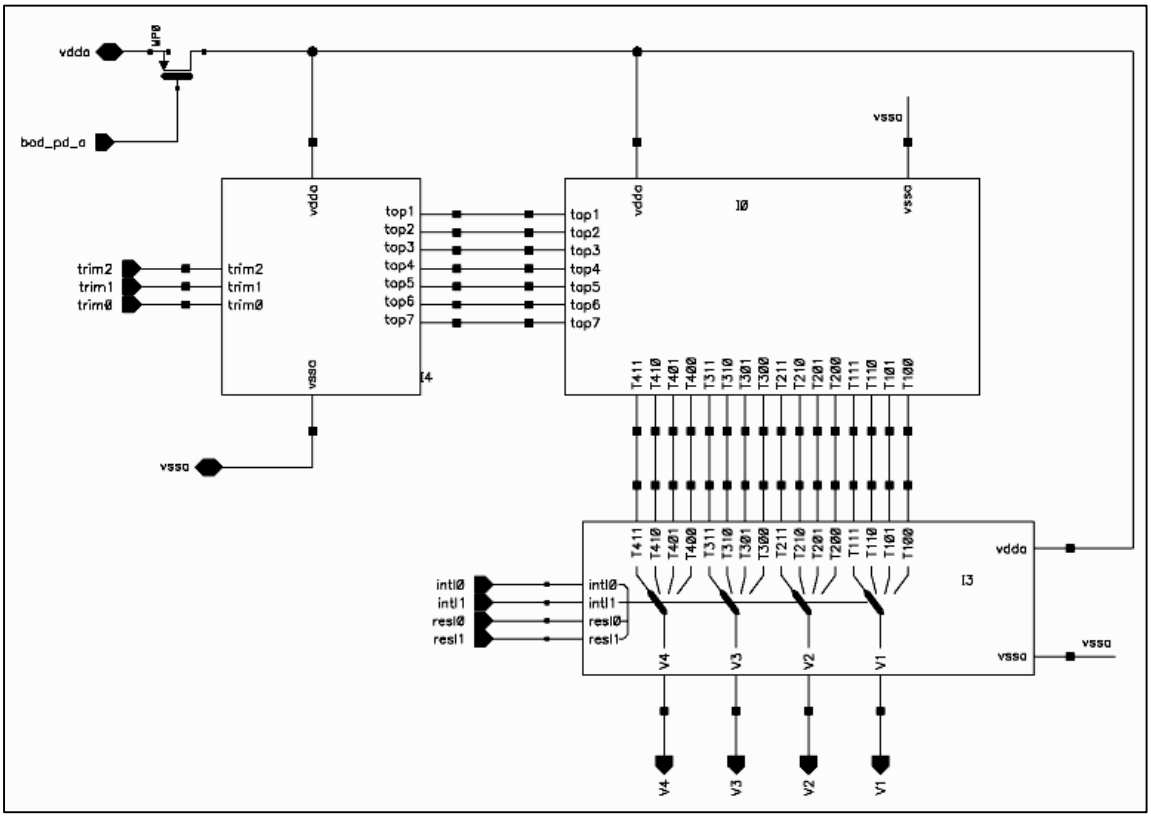

Figure 8. BOD trim mux, resistor divider, and trip point tap mux

The trim select consists of a 3 to 7 decoder that drives 7 pass gates that tie different tap points of the top of the resistor ladder to vdda. The decoder and pass gats are shown in figure 9.

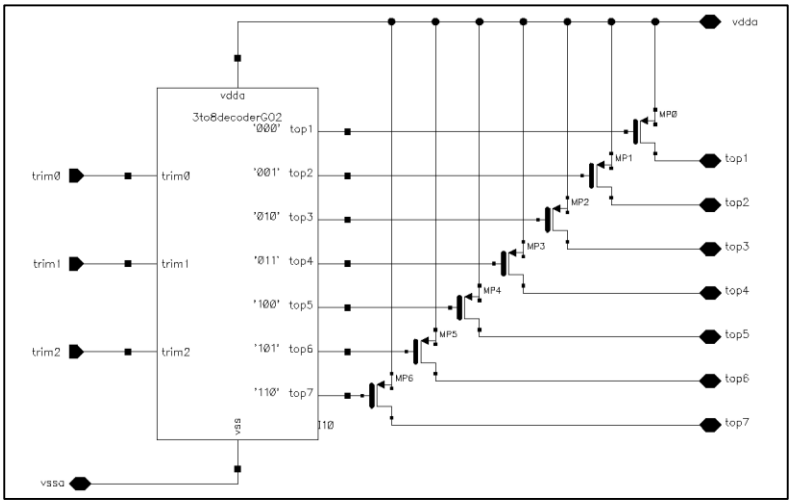

Figure 9. BOD trim select mux

The code to model this is straight forward to write as shown below. In this example, the vdda is modeled as a digital signal to avoid updating power/ground versions of logic library. The vdda value is fed to the bod model using a hierarchical assignment in the testbench. Wreal values are propagated from the resistor ladder tap outputs, through the trip point tap mux and into the comparator.

```
module trim_divider_switchunit_bod_lp ( top1, top2, top3, top4, top5,
     top6, top7, vdda, vssa, trim0, trim1, trim2 );

inout  top1, top2, top3, top4, top5, top6, top7, vdda, vssa;

input  trim0, trim1, trim2;

mco_c90_bod_lp_lib_3to8decoderGO2_schematic I10 ( .vss(vssa),
     .vdda(vdda), .top7(net054), .top6(net050), .top5(net049),
     .top4(net048), .top3(net031), .top2(net030), .top1(net28),
     .trim0(trim0), .trim1(trim1), .trim2(trim2));

pmos  MP2 ( top3, vdda, net031);
pmos  MP1 ( top2, vdda, net030);
pmos  MP0 ( top1, vdda, net28 );
pmos  MP6 ( top7, vdda, net054);
pmos  MP5 ( top6, vdda, net050);
pmos  MP3 ( top4, vdda, net048);
pmos  MP4 ( top5, vdda, net049);

endmodule
```

The resistor ladder is shown in figure 10 below. It is simply a string of resistors with one end tied to ground and the top end tied to vdda through one of the top* taps as selected by the trim select mux. The 16 outputs are connected to intermediate points in the ladder.

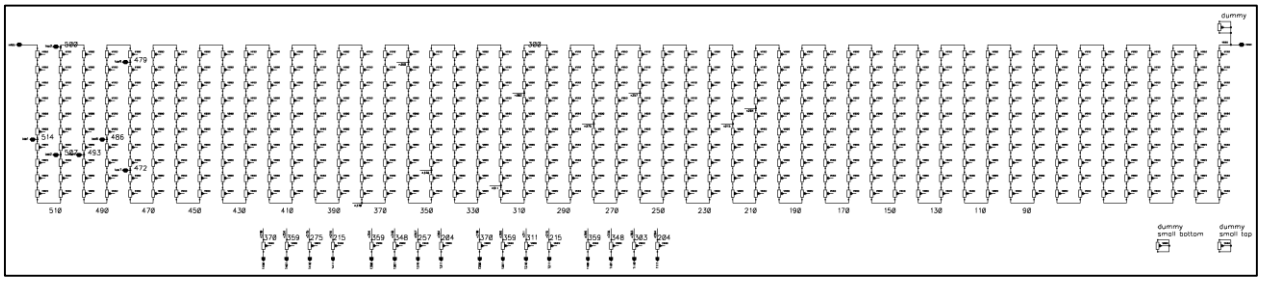

Figure 10. BOD resistor ladder

The code to model this simply detects which trim tap point is powered and calculates the correct real value for each tap based on a simple divider calculation:

```
module divider_string_bod_lp ( T100, T101, T110, T111, T200, T201,
     T210, T211, T300, T301, T310, T311, T400, T401, T410, T411, top1,
     top2, top3, top4, top5, top6, top7, vdda, vssa );

output  T100, T101, T110, T111, T200, T201, T210, T211, T300, T301,
     T310, T311, T400, T401, T410, T411;

inout  top1, top2, top3, top4, top5, top6, top7, vdda, vssa;

wreal T100, T101, T110, T111, T200, T201, T210, T211, T300, T301,
     T310, T311, T400, T401, T410, T411;

wire top1, top2, top3, top4, top5, top6, top7;

wreal    vdda_value, vdda_value_i;
integer ladder_length;
reg     bad_rladder;

always @(top1 or top2 or top3 or top4 or top5 or top6 or top7)
begin
  bad_rladder = 0;
  if (top7 === 1)
    ladder_length = 472;
  else if (top7 === 1'bz && top6 === 1)
    ladder_length = 479;
  else if (top7 === 1'bz && top6 === 1'bz && top5 === 1)
    ladder_length = 486;
  else if (top7 === 1'bz && top6 === 1'bz && top5 === 1'bz && top4 === 1)
    ladder_length = 493;
...
  else
    bad_rladder = 1;
end

assign vdda_value_i = bad_rladder === 0 ? vdda_value : `wrealXState;

assign T100 = vdda_value_i * 273 / ladder_length;
assign T101 = vdda_value_i * 231 / ladder_length;
assign T110 = vdda_value_i * 207 / ladder_length;
assign T111 = vdda_value_i * 188 / ladder_length;

assign T200 = vdda_value_i * 293 / ladder_length;
assign T201 = vdda_value_i * 245 / ladder_length;
assign T210 = vdda_value_i * 218 / ladder_length;
assign T211 = vdda_value_i * 197 / ladder_length;

...
```

The other pieces of the BOD are similarly modeled. The submodule code is stored in the schematic database so that the top level behavioral can be generated by netlisting the schematic using the Verilog netlister. Since the code travels with the submodule schematics it is easier to make sure changes to the schematic get migrated to the Verilog model.

c) BOD Modeling in VAMS/VA

```
`include "disciplines.vams"
`timescale  1ns / 10ps

`define step_size_trig 25e-3 // From spec tuning step of trigger level 25mV
`define step_size_hyst 20e-3 // From spec tuning step of hysteresis 20mV

module bod(p,n,enable,latch,reset,threshold,hyst);
  parameter real rvin = 0.450;
  parameter real vdd_thr_lo = 0.8;
  parameter real vdd_thr_hi = 1.5;

  inout p;
  inout n;
  input  enable;
  output latch;
  input  reset;
  input  threshold;
  input  hyst;

  electrical p;
  electrical n;
  wire       enable;
  reg        latch;
  logic      comparator_lo;
  logic      comparator_hi;
  electrical ref_lo_e;
  electrical ref_hi_e;
  wire       reset;
  wire[4:0]  threshold;
  wire[2:0]  hyst;

  real rvref_lo;
  real rvref_hi;

  initial begin
    rvref_lo = 0.0;
    rvref_hi = 0.0;
    latch    = 0;
  end

  always @ (threshold,hyst) begin
    rvref_lo = rvin + `step_size_trig * (threshold);
    rvref_hi = rvin + `step_size_trig * (threshold) + `step_size_hyst * (hyst);
  end

  a2d #(.vh(0.001),.vl(-0.001)) i_convert_lo(comparator_lo, ref_lo_e);
  a2d #(.vh(0.001),.vl(-0.001)) i_convert_hi(comparator_hi, ref_hi_e);

  analog begin
    V(ref_lo_e,n) <+ V(p,n)-transition(rvref_lo,0,100n,100n);
    V(ref_hi_e,n) <+ V(p,n)-transition(rvref_hi,0,100n,100n);
  end

  always @ (enable,comparator_lo, comparator_hi,reset) begin
    if (comparator_lo == 0 && enable == 1) begin
      latch = 0;
    end
    if (comparator_hi == 1 && enable == 1) begin
      latch = 1;
    end
    if (reset == 1 || enable == 0) begin
      latch = 1;
    end
  end

endmodule
```

C. Self-Checking Testbench and Verification Environment for Mixed Signal SoCs

Leveraging the digital verification techniques for mixed signal designs enables thorough verification in a timely and efficient manner. Employing techniques such as SVA, PSL proved to be of great advantage. With the SV bind capability assertions developed at module level can be re-used at top level. Assertions were coded in SV to check on the internal supply voltage of an analog block to be within a certain level.

Several methods of connectivity checks at the digital and analog boundary interface were deployed including a formal approach and a TcL based connectivity check as shown in Figure 11. Incisive Formal Verifier (IFV/ IEV) from Cadence was used for Formal Verification. An internally developed procedure for connectivity checks using TcL is shown below. For tracking, the verification progress eplanner tool provides a methodology to maintain the traceability matrix. As a next step, the plan is to evaluate the suitability of UVM for the mixed signal chips.

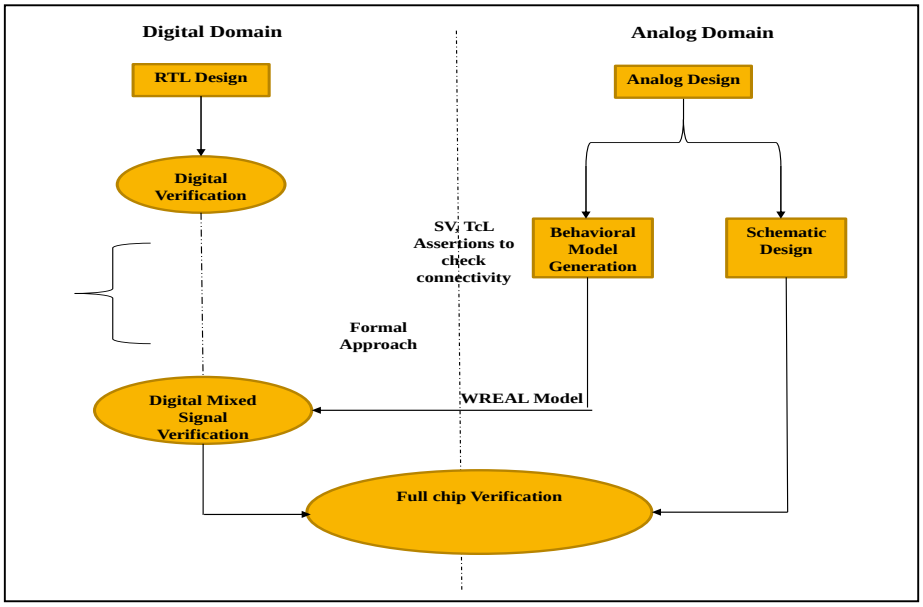

Figure 11. Verification Flow

1) SV Assertion Logic to check on the internal supply voltage:

```
initial begin
forever @(clk)
   begin
      if (!hvab_lag_ic_tb.dut.hvab_lag_dig_top.rst_n) #4000000;// 3ms delay
      assign vddd = hvab_lag_ic_tb.dut.hvab_lag_ana_top.vddd;
      assign vddd_reg_val =   rw_reg_B[11][15:8]; // Hierarchical path
      $cast(vddd_int_val, vddd_reg_val);
      vddd_val = 1800 - (vddd_int_val * 7);
      l_vddd_val = vddd_val - (0.05 * vddd_val);
      h_vddd_val = vddd_val + (0.05 * vddd_val);
      vddd_regulation_monitor : assert (((vddd * 1000) > l_vddd_val) && ((vddd * 1000) < h_vddd_val))
   else
      $display("\n @%0t *** vddd out of regulation with value: %f expected to be: %f", $time,vddd_val,vddd);
   end
end

// Procedure to set vgg_short_cnt, for scenarios where VGG short is released before timeout

int vgg_short_cnt;
always @(posedge clk)
begin
   if(def_assigns.vgg_reg_ok == 1)
   begin
      wait(.vgg_short);
      wait(!vgg_short);
     vgg_short_cnt=1;
   end
 end

reg [7:0] vgg_reg_val;
int vgg_int_val;
int vgg_val, l_vgg_val, h_vgg_val;
real G1_m, G2_m, G3_m;

always @(posedge clk)
begin
   assign vgg_reg_val = def_assigns.vgg_control_cfg;
   $cast(vgg_int_val, vgg_reg_val);
   vgg_val = 16000 - (vgg_int_val * 63);
   l_vgg_val = vgg_val - (0.04 * vgg_val);
   h_vgg_val = vgg_val + (0.04 * vgg_val);
   G1_m = G1;
   G2_m =G2;
   G3_m = G3;
    if((def_assigns.vgg_reg_ok == 1))
   begin
      if(.vgg_short == 0)
      // If VGG short released before vgg_reg_protection bit getting set, we wait for VGG to
      // come back to programmed value
      if(system_asserts.vgg_short_cnt == 1) #1370500
      begin
        VGG_regulation : assert (((VGG * 1000) > l_vgg_val) && ((VGG * 1000) < h_vgg_val))
                              else $display("VGG out of regulation with value: %f expected to be: %f", (VGG * 1000), vgg_val);
      end
   end
      else
      begin
        Gate_switch : assert ((G1_m <= 0.01) && (G2_m <= 0.01) && (G3_m <= 0.01))
                           else $display("GATE switches even when vgg is not regulated");
      end
end
```

2) TcL procedure for connectivity checks:

```
proc check_connectivity {src dst} {

   global serror gsrc gdst

   set gsrc $src
   set gdst $dst

   set serror 1

   stop -create -name watch_signal -silent -continue -object $gdst \
       -execute { if { [ value %b $gsrc ] != [ value %b $gdst ] } {
         if { [ value %b $gsrc ] == 1  &&  [ value %b $gdst ] == 'H'  } { set serror 0
                   } elseif {  [ value %b $gsrc ] == 'H'  &&  [ value %b $gdst ] == 1 } { set serror 0
         } else { set serror 1 }
                 } else { set serror 0 } }  -noexecout

   mforce $src 1; run 20 us; mforce $src 0; run 20 us;
   mforce $src 1; run 20 us; mforce $src 0; run 20 us;
   mforce $src 1; run 20 us; mforce $src 0; run 20 us;
   mforce $src 1; run 20 us; mforce $src 0; run 20 us;

   stop -delete watch_signal
   release $src
   run 100 us

   if {$serror == 1} {set flag FAILED} else {set flag PASSED}

   puts "Connectivity between $src $dst at [time ns] $flag"
}

proc analyse_connectivity_check {} {

   global path
   global const
   global serror
   set serror 0

   check_connectivity $path(SOURCE) $path(DESTINATION)
}
```

D. LDO Sign Off Criteria

LDO regulator, the fundamental building block of a power management unit are extensively used in battery operated chips to provide a stable supply voltage with high power efficiency. Individual LDOs are used for different domains so as to provide isolation among the subsystems. To design a low power compact LDO an accurate model of the chip power behavior across the various operating modes is needed. Around 5 use-cases were identified per LDO which simulated the different modes like startup mode, idle mode, processor running at different speeds etc. on the synthesized netlist initially and finally on the back annotated clock tree inserted netlist. Simulation data was given to red hawk tool to generate the CPM models for the above scenarios. These SPICE level chip power models were used in the LDO sign off simulations to finalize the design of the LDO blocks and determine the amount of de-caps to be used in the chip for every domain to ensure power integrity.

Figure 12 shows the SPICE simulation snapshot of peak current excursion for the digital, processor memory and DSP memory domains. Table 1 outlines the peak to average current consumption ratio, which turned out to be high for the memory domain.

Figure 13 shows the ripple on the LDO output. The processor memory domain was identified as an outlier (Peak/Average > 95). This was fixed by increasing the number of de-caps in this domain.

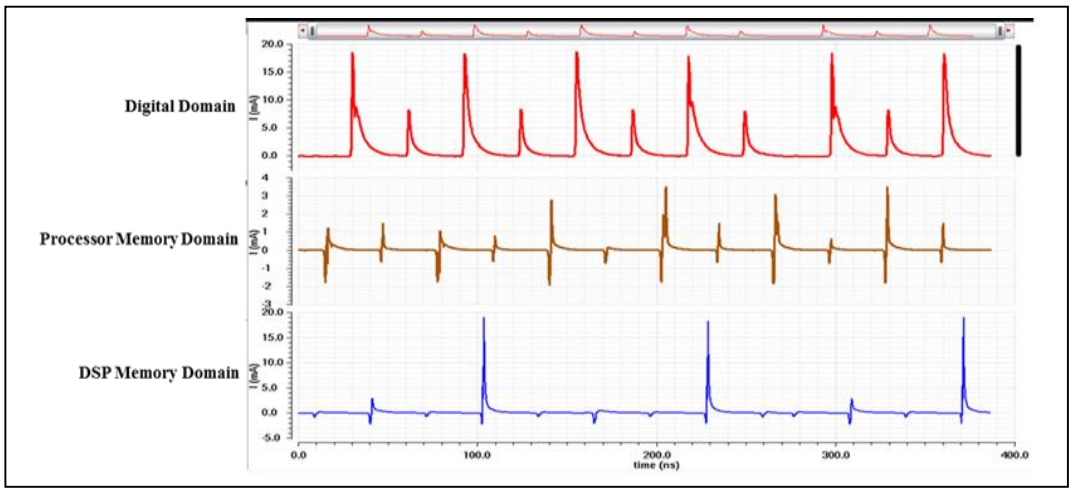

Figure 12. Domains Peak and Average Current

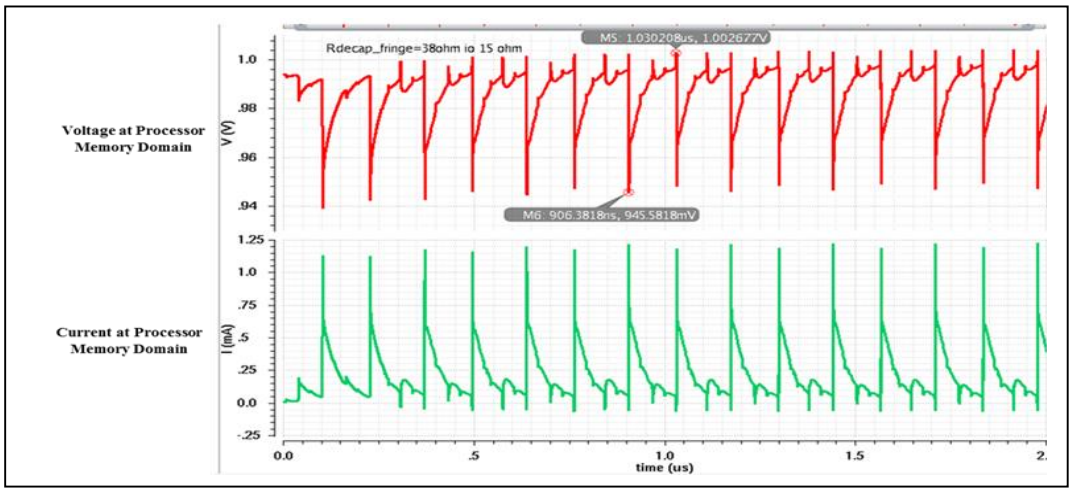

Figure 13. V(Peak-Peak) Ripple on LDO

Table I. CPM Average Current

Domains	**Peak Current (mA)**	**Average Current (mA)**
Digital Domain	18.8	1.53
Processor Memory domain	19	0.211
DSP Memory Domain	3.51	0.0668

III. Conclusion

- While CPF supports classical power scenarios like always on, shut off it required changes in the environment to model the Self Induced Power scenario.
- Power management systems have very intricate dependencies between the analog and the digital controller. To jump start the verification we started using randomized Verilog models then replaced it with Wreal models and eventually the Verilog AMS models, thus helping us to shorten the cycle time.
- Digital techniques like assertions and formal checks proved to be useful in the mixed signal world. Continuing on the same lines we intend to evaluate the suitability of UVM.
- A novel method to sign off LDO using chip level digital power models resulted in the accurate Design and Verification of power supply which is key in ensuring silicon power integrity.

Acknowledgment

Multiple team members belonging to Secure Monitoring and Control Business Line contributed directly or indirectly to this paper. We would like to acknowledge the Design Methodology team for their guidance during the course of work. We would like to make a special mention of Mukesh Nair, Kishore UPP, Nagendra Prasad, Rachid el Waffaoui, Ravishankar TS, Rudolph Alexandre, Sainath K, Sumeet Gulati, Vishal Choudhary, Vishwanath Kunchigi and Jean Charles Herbiet.

References

[1] Michael Bolt – Director NXP R&D Asia, "Internet of Things – Challenges and Opportunities for IC designs", ISIC conference, Singapore, December 2014.

[2] Walter Hartong and Scott Cranston, "Real Valued Modeling for Mixed Signal Simulation". Product Version IES 8.2, January 2009, Cadence.
DOI=https://www.cadence.com/rl/Resources/application_notes/real_number_appNote.pdf

[3] Noah Bamford, Saji George, Milind Padhye – Freescale Semiconductor, "Verification of Low Power Designs using CPF". DOI=https://www.cadence.com/rl/Resources/conference_papers/2.7Paper.pdf

[4] Prabal Bhattacharya and Don 'Riordan, Cadence Design Systems, "Verifying low-power intent in mixed-signal design", October 30, 2012.
DOI= http://www.techdesignforums.com/practice/technique/cadence-mixed-signal-methodology-guide

[5] RedHawk User Manual (Apache Design, Inc. A subsidiary of ANSYS Inc) Software Release 13.2, April 28, 2014.