Efficient SoC Level Mixed Signal Frontend Verification using Wreal Models

Anu Marisha
Texas Instruments
Bangalore, India

Nayana Prakash
Texas Instruments
Bangalore, India

Udit Kumar
Texas Instruments
Dallas, US

Rajesh Tiwari
Texas Instruments
Bangalore, India

Vijay Kumar Birange
Cadence
Bangalore, India

***Abstract*— With the growing demand of complex mixed-signal systems, it is almost impossible today to separate the digital and analog domain without compromising on the system functionality. To meet the market requirements, we need new efficient techniques, methodologies and faster models to simulate the interaction between the analog and digital portions. In this paper, we present how the WREAL model can be used to address functional and DFT verification requirements of mixed-signal designs.**

Keywords*— *SoC ; wreal; digital mixed-signal verification; analog modules;

I. INTRODUCTION

Today the mixed-signal applications are one among the fastest growing market segments in the semiconductor industry. The growing design challenges and complexities are becoming a bottleneck for the design process of a SoC.

The Fig 1 compares how the mixed signal designs have evolved over the period of time. The designs in the past were

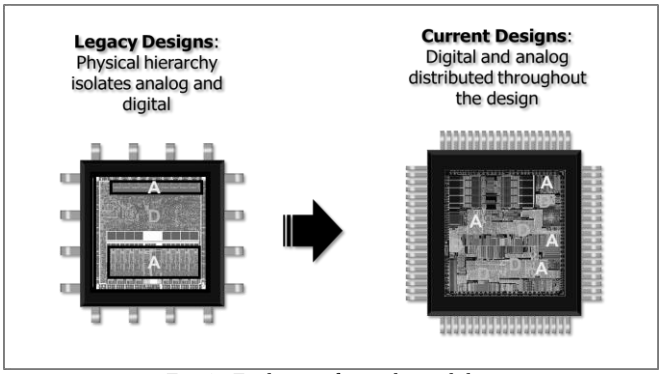

Fig. 1: Evolution of mixed signal designs

simpler where the analog and digital parts of the logic where developed separately and were integrated with a definite physical hierarchy. The world of electronics brings some interesting conflicts, perhaps none more interesting than the stark contrast between the analog and digital domains. The technologies are different, the design engineers are different and the ways in which designs are verified are different. But despite their differences, the analog and digital domains have to work alongside each other. Today the designs have become so complex that both analog and digital portions are integrated in a complex interleaved manner. When things were simpler in the past, the mixed-signal SoCs contained IP blocks that were designed separately and then bolted together during system integration. Designers simply brought a handful of "black boxes" - blocks of analog circuitry that were presumed to be pre-verified - into a mostly digital SoC design. Now, however, analog IP blocks are not only growing more numerous and complex, but also increasingly contain digital control logic.

The second approach used in past was mixed signal co-simulation, because of slow simulation speeds and time-consuming simulation setup we cannot use this approach for full chip simulations.

As more complex, mixed-signal SoC designs continue to stress verification methodologies and schedules, designers need new approaches in solving today's test challenges. Mixed-signal verification in digital environment presents a unique challenge as the analog portion of the design requires highly accurate and time-consuming verification setup. Furthermore, without a digital representation of the analog design, full digital regression simulations are not possible for the SoC. This is insufficient for verifying connectivity and basic functionality of the integrated SoC at the system level. This in turn results incomplete SoC-level and system level verification or uncertainties in verification coverage [1].

This paper talks about our experience on one of the 45nm SoCs recently designed for secure applications. Out of the 8 sections, sections II and III will cover 'the conventional approach' and 'the need of wreal models' respectively. Section IV will cover the wreal features and section V will share details on some of the design critical issues identified during SoC verification which helped to avoid design re-spin.

II. CONVENTIONAL WAY OF MIXED SIGNAL VERIFICATION

Mixed signal SoC verification is always a challenge for any design team. The verification environments used for analog is completely schematic driven and manual, on the other hand the digital portion verification is highly automated and HDL based. Analog verification is transistor level while it is event level for digital verification. The emphasis is more on performance in case of analog verification while for digital, it is on the functionality. The results obtained are non-deterministic as well in case of analog verification. Hence there is a clear gap in the verification methodologies.

In general, we rely on the IP level verification for analog portion which uses "Analog Mixed Signal Simulation (AMS) and these modules are used as a black-box models at SoC level.

The following approaches have been used at SoC level.

A. Pure Digital modeling

Inaccurate models which are sufficient for very basic checks. There are no automated checks or verification possible on these and can lead to bugs in SoCs because of the inaccuracies.

B. Black Box Approach

Analog modules are replaced with black boxes. No simulation based checking is done. Only connectivity assertions are done. These checks are as good as the DV or DFT engineer's knowledge about the analog to digital interface. Any assumptions made by analog engineers about digital circuit and vice versa (integration done at SoC level) are not validated and can lead to bugs.

C. AMS (Analog Mixed Signal Simulation) Approach

Very accurate mixed signal simulation but comes with a penalty of very high simulation times along with extra license costs for the Verilog-AMS simulator.

III. MIXED SIGNAL VERIFICATION USING WREAL MODELS

The WREAL also called as Real Value Models (RVM) helps us in bridging the gap between the analog and digital co-simulation. RVM is a mixed approach borrowing concepts from both analog and digital domains. The values are continuous as in the analog world. However, time is discrete, meaning the real signals change values based on discrete events. In this approach, the digital engine solves the RVM system without support of the analog solver. This guarantees a high simulation performance that is in the range of a normal digital simulation and orders of magnitudes higher than the analog simulation speed [3].

There are different HDL language standards that support RVM including Verilog, SystemVerilog, VHDL, *e*, and Verilog-AMS. Verilog-AMS supports WREAL while other supports a real data type. In this paper we cover the benefits gained in mixed signal SoC verification using wreal models for Analog modules.

In traditional Verilog, real values were modeled using 64-bit vectors, which encoded the real value in the IEEE floating point format. Two system tasks, $realtobits and $bitstoreal, were provided to encode and decode the real values in the 64-bit vectors. In this environment, you model a real value with a single, scalar entity, which does not map into the traditional Verilog representation of a 64-bit vector real. This also proved difficult in the mixed language world with VHDL reals not mapping cleanly to the 64-bit vectors used in traditional Verilog.

Wreal models are coded using verilog-AMS wreal language. In Verilog-AMS, the concept of a truly real-valued net or wire was introduced, called "wreal" – a real valued wire. These nets represent a real-valued physical connection between structural entities. Also advanced digital methodologies can be used with wreal models for verification of mixed signal designs. Hence the flow is called as Digital Mixed Signal (DMS) verification. It includes all the benefits of a digital signal in Verilog-AMS.

A snippet of wreal code is shown in figure 2 below.

```
module A (i);
  input i;
  wreal i;
  real no;
    initial begin
          while (no < 10.0) begin
            #1 no = no + 0.1;
      end
    $stop
    end
   assign i = no;
endmodule
```

Fig. 2: Snippet of wreal code

The pure digital models are very inaccurate and are only sufficient for limited verification tasks, like connectivity checks. The wreal models are much more accurate and also provide high simulation performance compared to spice simulations.

Figure 3 shows a general trend in the accuracy-performance tradeoff among spice, fastspice, real, wreal models simulation.

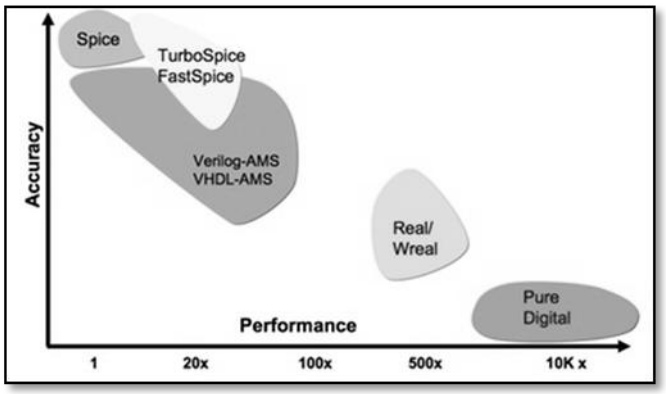

Fig. 3: General trend in accuracy and performance tradeoff

A. Wreal extensions

Verilog-AMS Language Reference Manual (LRM) lists the following restrictions on wreal nets:

- Can only connect to wreals, wires, or real expressions
- Scalar only and no support for arrays
- Can have at most one driver

Cadence provides some significant extensions to the above restrictions which enable seemless usage of wreal on SoC.

These extensions are:

a. Easy interaction between analog & digital portions of the design. Easy instantiation of WREAL model on a verilog module.

b. Ability to connect real to logic nets using automatically inserted Real2Logic-Logic2Real (R2L-L2R) connect modules. (Refer figure 4)
   - User can write their own Connect Rules. E.g. to covert a logic level of 1 to a real value of 1.8.
   - Multiple connect rules can be configured using configuration file.

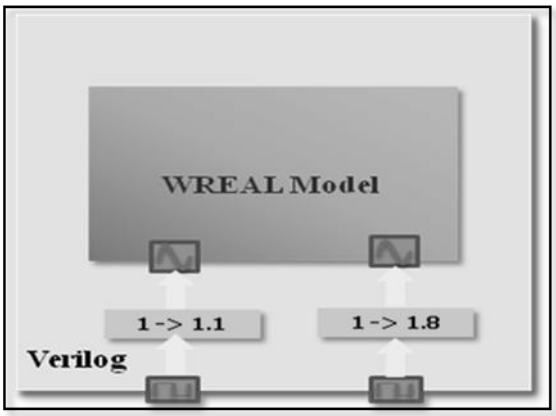

Fig. 4: Multiple connect modules insertion

c. Automatic Wreal Coercion between Verilog and Wreal module to enable connecting wreal with logic.

During the elaboration phase, the connectivity of a mixed-signal design is computed. This also involves determining and attaching types, such as logic type and electrical type to interconnects (wires). The Verilog-AMS LRM only allows wreal ports and nets to connect to wires. In this case, the wires get resolved to the type wreal. This process is called coercion to wreal. This offers tremendous value in terms of model portability across various design configurations. In a different configuration, interconnect might be used to connect electrical ports – this works seamlessly without any change in the source code. (Refer to fig. 5)

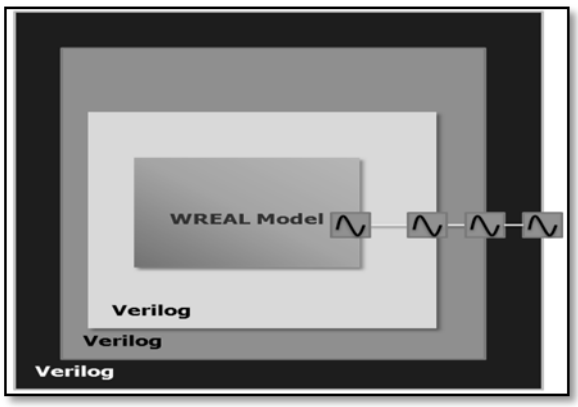

Fig. 5: Automatic coercion

d. Remove coercing using "rnm_coerce"
Wreal still does not support wreal to VHDL std_logic coercion. Whenever a wreal net crosses VHDL boundary the tool starts giving elaboration mismatch errors. To overcome that we can use remove coercing switch to stop coercing the wreal net from VHDL hierarchy. (Refer figure 6)

Fig. 6: Stopping coercion at VHDL using 'rnm_coerce'

e. OOMR – Out Of Module Reference

- Can drive real value to ports of Analog IP directly from Testbench using OOMR. This is required because some of the analog ports like voltage supply would not be present in rest of the RTL digital design. (Refer fig. 7)

f. Support multiple WREAL drivers using wreal_resolution switch

- Multiple drivers can be resolved with min, max or average value using this switch. This helped us in maintaining a single testbench across testcases. (Refer fig. 7)

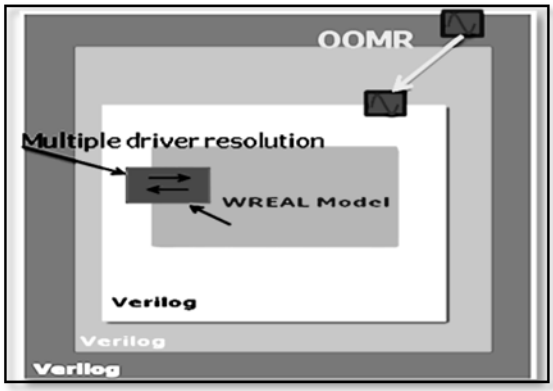

Fig. 7: OOMR and multiple driver resolution

g. Drivers can be traced using simple tcl commands

h. Schematic Tracing

i. Support for wreal arrays:

A wreal array groups multiple real values into a single, indexible entity. (Refer fig. 8)

```
module ams_tb;
    wreal y[3:2];
    sub_design d1(y);
    initial begin
    #10 $display("%f,%f",y[2],y[3]);
    end
```

Fig. 8: Usage of wreal array

j. Support for wreal X and Z states

The concept of an unknown – X and high impedance – Z state used in the 4-state logic is useful for wreal signals as well. The meaning of X and Z is equivalent for the wreal case. (Refer to fig 9)

```
module top();
    wreal s;
    real r;
    foo f1 (s);
    initial begin
        #1 r = 1.234;
        #1 r = `wrealZState;
        #1 r = 3.2;
        #1 r = `wrealXState;
        #1 r = -4.2;
        #1 $stop;
    end
    assign s = r;
endmodule
```

Fig. 9: usage of wreal X and Z states

B. Wreal Modelling

Models are generated by the IP designer and verified as part of their verification. Hence these models are pre-validated to be the same as original circuit.

These models can be optionally be generated by tools or hand coded. Virtuoso Schematic Model Generator (SMG) can be used to dump out models of individual blocks. It is tightly integrated into the design environment and enables the generation of analog, mixed-signal behavioral models using a schematic-like representation of the behavioral model. The schematic view is then processed to generate the behavioral model. With this approach, behavioral modeling is easier to comprehend [2].

SMG is easy to use and leverages the Schematic Editor to assemble the blocks that are placed, wired and calibrated. Model-schematics can be reused, shared, configured, and easily maintained. The graphical representation of design functionality makes the modeling process very transparent and understandable for analog circuit designers [4]. (Refer fig. 10)

Fig. 10: Virtuoso schematic model generator

The IP designer of the analog modules in our SoC, used AMS netlister (Cadence ADE OSS Netlister) to dump out the hierarchical wreal netlist for the wreal models from the schematics. Then he either generated individual block's logic through SMG or hand coded them. Hence in our case the wreal models were partially generated and partially hand coded.

C. Verification Setup

- In our design verification environment, wreal models for two Analog modules were used for verification.
- Wreal enabled us to drive and monitor voltages to and from analog domain.
- The real to logic auto-coercion and connect modules made the handshake between analog and digital domains feasible.
- Using Out of module referencing feature the 'real' voltages were driven to these Analog IPs directly from the testbench.

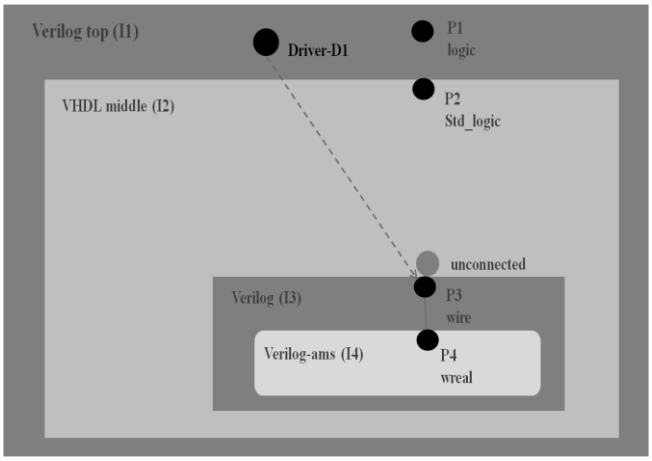

Fig. 11: Wreal verification setup

- Figure 11 shows the wreal model integration of one of the analog module on our SoC.
  - I1 is verilog top which has the testbench from where real values are being driven using OOMR.
  - I2 is intermediate vhdl module.
  - I3 is a verilog wrapper over the wreal model I4.
  - An example wreal port, P4, of I4 is driven with real values from testbench and is left unconnected from rest of the digital design.
  - All the other logic signals are directly connected from wreal model to vhdl intermediate using the verilog wrapper.
- We were able to drive in real voltages to Analog modules, also it facilitated monitoring of the outputs with respect to analog inputs.
- As shown in figure 12, verification setup with wreal is much closer to real system.

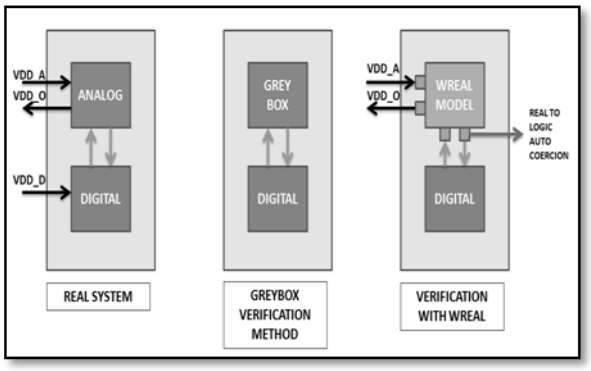

Fig. 12: Comparison of verification setups

- Debugging wreal signals in simulators is as easy as any other logic signal, they can be traced using schematic or tcl commands
- Test vectors were developed seeing the real time behavior of the model.
- Wreal verification setup is more efficient in terms of speed and accuracy when compared to AMS and greybox method respectively.
- Complete module functionality including power up sequencing was verified which helped in uncovering a lot of functional bugs early in the design cycle.
- The main categories of DFT tests were trimming, threshold measurements with & without hysteresis, delay measurements, transient response measurements etc.

IV. BENEFITS OF USING WREAL MODELS

Real value modelling tries to take the best from both the analog and digital simulation worlds. It uses the floating point numbers familiar in the analog domain, but blends this with discrete time values. This approach offers a level of performance similar to that of a digital simulation, and much faster than the speed of a purely analog simulation. Listed below are some of the benefits of using wreal real models in our SoC level verification.

- Models are generally generated by the IP team and verified as part of their verification. Hence they were pre-validated to be the same as original circuit.
- Model's hierarchies were easily dumped from the tool and the logic were hand coded.
- Ease of integration facilitated seamless usage on SoC

- Usage of wreal models validates assumptions made by IP team and SOC teams on interoperability between Analog and digital portions on the SoC.
- Enables modeling of accurate power supplies and other analog parameters.
- Trimming of various analog blocks can be modelled.
- Analog behavior due to glitches can be modelled.
- Test vectors were developed seeing the real time behavior of the module, which helped in get rid of iterations with respect to the vectors delivered to the ATE (Automatic Test Equipment)
- The wreal models are easily portable between design and verification environments.
- Used seamlessly in GLS (Gate level Simulation) and PAGLS.(Power-aware GLS)
- Boundary toggle coverage analysis possible with wreal models.
- Replacing the analog and mixed signal blocks in the SoC, under verification, with real and wreal models, the verification engineer can also run nightly regression runs since the verification would only use digital simulators thus avoiding the analog convergence issues.
- Connectivity of various analog test pads could be verified in simulations which is not possible in normal simulation setup.

V. RESULTS

In our design, we had two critical analog IP's designed to have voltage, frequency and temperature monitoring features. These modules were designed to detect and protect the chip from any external tamper event. One of the protection reactions was to assert the design reset if any tamper event is detected.

Since these Analog IPs were affecting our design reset path, it was critically important to have these IPs modeled and verified accurately during SoC verification.

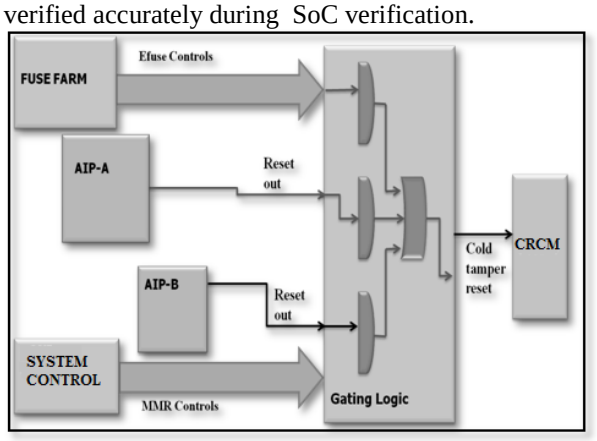

Fig. 13: SoC block diagram

Figure 13 shows these two IPs named as AIP-A and AIP-B. Both AIP-A and AIP-B modules give reset_out signals in case of tamper events, which go to Clock Reset Controller Module (CRCM) on the SoC through a gating logic. The other controlling input of gating logic comes from fuse farm and system control logic.

In this section we will cover six "design verification" critical issues identified during early design verification cycle and two important analog test categories which helped us in avoiding design re-spin and improving test pattern quality.

A. Design Verification Case Studies

1) Analog IP reset_out was sticky to PORz

- In event of tamper event, AIP-A asserts reset_out which goes as cold reset to the system and disables Power Management IC (PMIC).
- The AIP-A requires complete power recycling to bring the system out of reset (done by PMIC)
- Initially AIP-A was designed in such a way that its reset_out was dependent on efuse ready output of the fuse farm logic which is asserted after the fuse load operation.
- Let's say, in a scenario where AIP-A is enabled and chip PORz (Power On Reset) is asserted which will in turn de asserts efuse ready and hence the reset_out of the Analog IP would get asserted.
- Now if we deassert PORz, efuse ready will get asserted after fuse load operation as usual, but AIP-A reset-out will not get deasserted as it will wait for PMIC to do a complete power recycling. (PMIC will do a complete power recycling only in case of tamper events detection).
- Hence the system will remain in reset forever (even after deasserting PORz)
- This was a critical bug as the system would not have come out of reset in the event of PORz assertion-deassertion.

This bug was impossible to be caught with a blackbox model of the AIP-A as it required very accurate modeling of the analog module's functionality. IP team's detailed schematic verification would also not have caught this as this is outside the scope of IP verification. Wreal model of AIP-A module played the critical role of uncovering this bug early enough in the design cycle.

Resolution: AIP-A module was redesigned in such a way that if all system operating conditions are proper reset_out would be independent of PORz assertion. Reset_out logic was decoupled from efuse ready.

2) Glitch in digital domain in the event of AIP-A tamper reset

- In case of tamper event AIP-A gave reset_out which goes as cold reset to the system (as already specified) and gets logged in a digital module, named, DIP-A
- DIP-A, was supposed to be always ON and log tamper event details from AIP-A.
- The DIP-A which works on system clk would switch to an internally generated clk if the system clk is not running.
- The DIP-A interface clk was asynchronously gated in case of tamper event (cold reset assertion) and hence it might introduce glitches in DIP-A leading to corruption of registers
- This bug was caught after 5 RTL releases as it was not hit earlier.

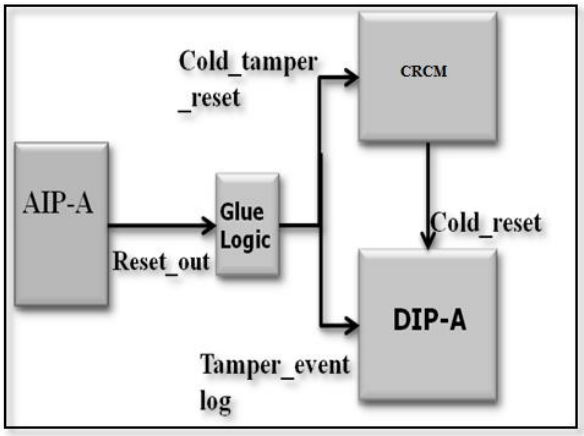

Fig. 14: AIP-A interaction with DIP-A

Resolution: Reset_out was given as a warm reset source, instead of cold reset, to CRCM. As it was warm reset, system clock to DIP-A got synchronously gated. Also, CRCM would give early warning to DIP-A so that it takes few internal clk cycle to switch its clk from system to internal and then give cold reset to CRCM.

3) STD_EFUSE reset dependent on Tamper_cold_reset

- Reset_out coming out of AIP-A was gated by two efuse bits before it reached CRCM as tamper_cold_reset.
- But efuse reset was in turn dependent on tamper_cold_reset output of CRCM
- Hence in a scenario where reset_out of AIP-A is asserted and the gatings are disabled by loading efuse, tamper_cold_rst will reset the fuse farm and hence the gating of the tamper reset will get enabled, which was not expected.
- Also the trim values coming to AIP from fuse farm would get lost, which was not intentional.

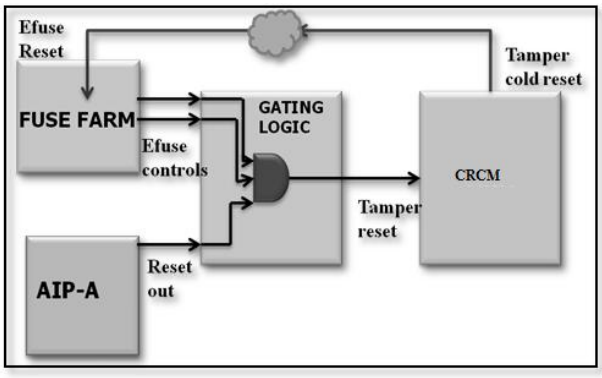

Fig 15: AIP-A and Farm fuse intergration

Resolution: Efuse reset was made independent of tamper cold reset source.

4) Four more SoC bugs were uncovered early because of using wreal models.

- Two of them were found in digital control logic implementated for Analog IP on SoC.
- One issue was found in the gating logic implemented for the Analog IP's reset_out to CRCM.
- One spec bug was found for incorrect threshold value of frequency tamper detect circuit. The circuit was updated for new threshold but updating the spec got missed.

B. DFT verification

1) Analog Trimming:

Trimming is defined as an iterative process of setting the appropriate trim code based on the feedback mechanism in which the measured current, voltage or frequency is mapped to the desired references. This is a common practice used in industry to improve the performance of Analog module based on strong or weak silicon. Generally, trimming will not be replicated in the simulation environment as the models would not have the feature modelled. But in the case of wreal models, trimming can be modelled efficiently and below example shows how trimming was replicated for one of the sub-components in the module in the DFT verification environment.

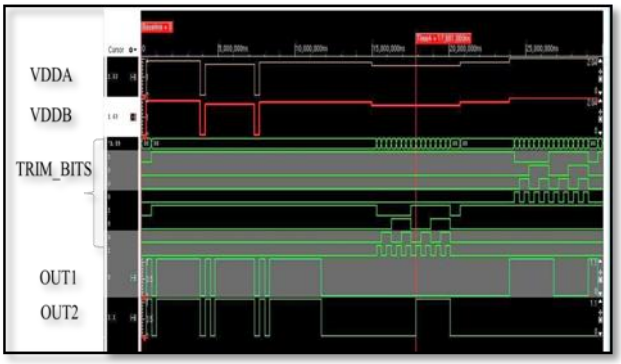

Fig. 16: Sample simulation waveform for trimming

- Figure 16 captures the trimming waveform for one of the sub-components inside AIP-A module. Here the trimming is done for a block whose inputs are VDDA,VDDB & TRIM_BITS.
- The trim value is selected based on the response seen on the two relevant outputs named as OUT1 & OUT2.

Since trimming could be verified in simulation, we were able to understand and confirm the exact behavior and requirement of the block during trimming and deliver vectors for silicon checkout, with much ease.

2) Threshold measurements

Threshold measurement is a kind of parametric test carried out for the analog blocks to record the switching threshold of certain outputs with respect to an input voltage. The test is done by ramping up the voltage while observing the response

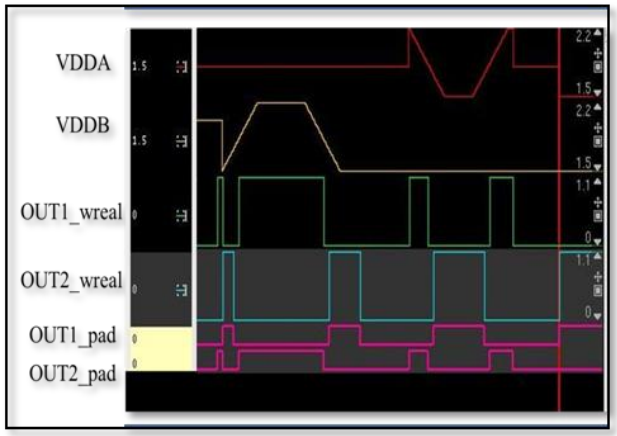

Fig. 17: Sample simulation waveform for threshold measurement

of certain outputs. The threshold values for the modules are those values that cause the relevant outputs to switch. This test is described next with the help of waveforms as in Fig 17.

- This test is done for a sub-component that has VDDA & VDDB as the input voltages with respect to which thresholds are measured.
- As shown in the Fig 17, when the voltages are ramped up and down, the outputs toggle from their default value to the opposite value as seen for OUT1_wreal & OUT2_wreal. The switching threshold can be measured based on when the switching happens.
- OUT1_pad & OUT2_pad are the digital versions corresponding to the wreal versions of the outputs OUT1_wreal & OUT2_wreal.

Threshold measurement is a typical example of the kind of test that could not have been verified at SoC level without a wreal verification environment. The capability to ramp the voltage is the key wreal feature which helped in verifying this test in the simulation environment.

If black box models are used for these modules, at the SoC level there would be no way to replicate such tests. This would have resulted in multiple iterations to get the test working on silicon. But, in our case since we had the test replicated in the wreal simulations and the vectors were developed knowing the real time behavior of the module in response to various inputs, we could very well get away with the iterations and enable quick silicon bring up especially in the case of modules which are relatively new.

C. Verification Statistics

- Seven DV bugs were caught on SoC early using these models which wouldn't have been easy to catch otherwise - **reduced time to market** (approx. 6 months).
- 30% of DV test suite for a mixed signal subsystem used WREAL IP for functional verification.
- Around 150 high quality test vectors were developed for analog DFT verification with the help of wreal model based simulation environment which helped in avoiding iterations with respect to silicon validation.

VI. CONCLUSION

Today, as designs emerge with more and more analog portions, thorough verification of the analog content along with the digital content and the connectivity between the two is emerging as the biggest challenge. Using wreal models for Analog IPs is an efficient and easy solution for this. Advanced digital methodologies can be used with wreal models for verification of mixed signal designs. Wreal model integrated simulation environment helps in substantially increasing the quality of DV and DFT verification without significantly impacting the simulation speed.

VII. ACKNOWLEDGMENTS

Authors would like to thanks Srivaths Ravi (Texas Instruments) & Rama Kowsalya (Texas Instruments) for their valuable guidance during the course of this work.

VIII. REFERENCES

[1] Kishore Karnane, Sathishkumar Balasubramanian, "Solutions for Mixed-Signal SoC Verification", Cadence Design Systems.

[2] "Virtuoso Mixed-Signal Behavioral Modeling Technology", Cadence Design Systems.

[3] "Real Valued Modeling for Mixed Signal Simulation",

http://www.cadence.com/rl/Resources/application_notes/real_

number_appNote.pdf

[4]http://www.techdesignforums.com/practice/guides/real-

value-wreal-modelling/