DV UVM based AMS co-simulation and verification methodology for mixed signal designs

Sandeep Sharma
Meta reality labs
sandeepsharmap@meta.com

***Abstract*-The comprehensive verification of Analog Mixed Signal (AMS) design often encounters challenges due to misalignment in development schedules between the analog and digital designs. These discrepancies stem from divergent priorities in simulation cycles, particularly the emphasis on simulation accuracy for SPICE compared to functional verification needs of digital RTL. Furthermore, the time-consuming nature of SPICE solver simulations for analog, as opposed to event-based verilog simulations for Digital, results in significant disparities in completion times, impacting development turnaround times.**

Over the years, various Mixed Signal Verification (MSV) approaches have emerged to address these challenges, seeking to strike a balance between accuracy and performance trade-offs. These trade-offs, often driven by design dependencies such as the high-frequency nature of SPICE or low-power modeling of RTL (UPF) [1], which introduce significant productivity bottlenecks in test plan completion and functional verification sign-off, especially considering the growing presence of Mixed-Signal IPs.

Building upon industry proven MSV methodologies, this paper describes a verification methodology tailored to resolve these challenges. This paper outlines the drivers behind this vision and provides detailed implementation insights. It begins with an overview of the design and the legacy verification flow, encompassing analog behavior using system verilog real number modeling (SV-RNM) for digital simulations and DV test cases. Subsequently, it delves into the intricacies of separate GUI-based SPICE simulations, including stimulus VCD vectors and their generation, followed by an exploration of verification gaps exposed by the "digital and analog verification in parallel" approach.

The paper proceeds to offer an architectural overview of the new MSV testbench, supporting Digital Mixed Signal (DMS) and Analog Mixed Signal (AMS) Co-simulation with spice netlist, alongside guiding principles, and implementation details. It explains the approach for dynamically switching RNMs and SPICE netlists in regression runs, outlines the randomized nature of DV UVM testbench, and explores the analog response.

I. INTRODUCTION

A. AMS Verification Challenges

Modern-day SoCs (System on Chips) are becoming increasingly complex. Mixed-signal systems have reached a point of size and complexity where transistor-level SPICE simulation methods cannot provide timely verification solutions. Particularly, simulating the entire system together using SPICE is nearly impossible due to time constraints. Figure 1 illustrates the tradeoff in accuracy and performance between SPICE and digital simulations.

This bottleneck can only be addressed by altering the design abstraction at a higher level, employing modern analog behavioral modeling methodologies such as Verilog-A/AMS and SV RNM. Support for new features in the System Verilog language standard [2] for behavioral modeling makes this task feasible. While continuous advancements are being made in analog tools to enhance performance, they still fall short of achieving the same level of automation seen in the domain of digital tools, primarily due to the inherent differences between analog and digital signals.

The binary and event-based nature of digital signals allows for easier development and automation of digital design and verification methodologies, leading to significant advancements in digital synthesis, PNR (place and route), and design verification tools. The field of digital verification within integrated circuits has witnessed significant advancements, contributing to the development of more sophisticated and reliable electronic systems. As semiconductor technology continues to shrink, and the complexity of integrated circuits grows, the need for robust verification methodologies becomes paramount.

Modern digital verification techniques leverage advanced simulation and formal verification tools, enabling engineers to thoroughly test complex designs and ensure their functionality before fabrication. Moreover, the integration of AI and machine learning algorithms in digital verification [3] has emerged as a powerful tool for identifying patterns,

optimizing test scenarios, and predicting potential design flaws. The evolution of hardware description languages, such as SystemVeriog, and the adoption of standardized verification methodologies like Universal Verification Methodology (UVM), have further streamlined the verification process, enhancing collaboration and efficiency in the development of intricate integrated circuits. These advancements collectively contribute to the creation of more reliable and high-performance electronic devices that power the ever-expanding digital landscape.

The verification gap between analog and digital creates a need to adopt some of these advancements in digital verification to analog. Digital functional verification is attainable using RTL (Register Transfer Level), and Digital Mixed-Signal (DMS) verification objectives can be met by utilizing SV RNM models for analog abstraction.

Nevertheless, there is always a necessity to perform co-simulation involving both digital and SPICE-level abstractions for certain analog portions of the design to ensure higher confidence in the success of the design. Therefore, there is a need to develop a process that involves the combination of digital RTL, SPICE-level models for select analog content, and behavioral descriptions of the remaining analog portions using SV-RNM, and to simulate these collectively

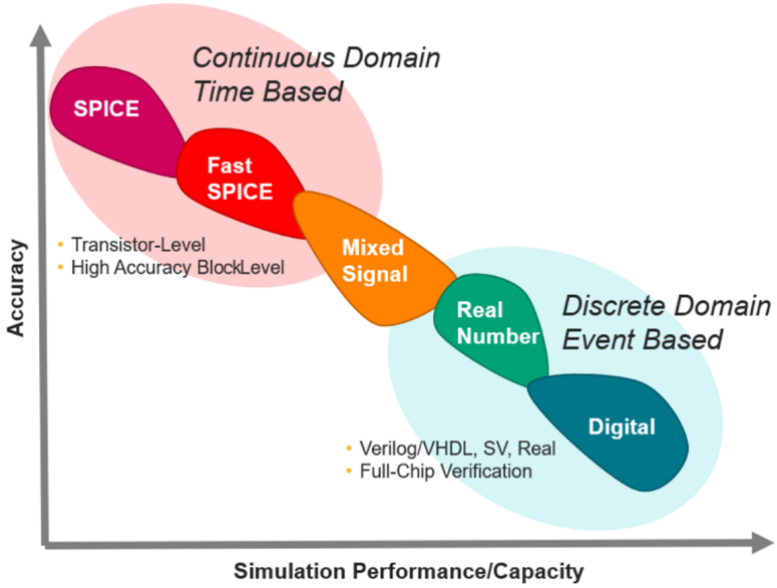

Figure 1: Accuracy Vs performance for different mixed signal verification strategies [4]

B. DUT and Design environment

This paper discusses a mixed signal verification methodology for a typical mixed signal subsystem. The primary application of such subsystems is to sample real-world signals, abstracted electrically as voltages or currents, which are then analyzed, processed, and propagated to an SoC for further applications. The AMS subsystem serves as the primary interface for the involved signals. Such a system is depicted in Figure 2. The input analog signal is first amplified by the analog front end (AFE) and then sampled for digital conversion. The resulting digital data is processed by the digital ADC controller and loaded into a FIFO. The final digitized data output from the AMS subsystem is then accessed by the SoC using the SPI interface. The data transfer between the AMS subsystem and SoC is bi-directional, with a 4-wire SPI interface used, where the SoC acts as the master device and also provides the clock to the AMS subsystem.

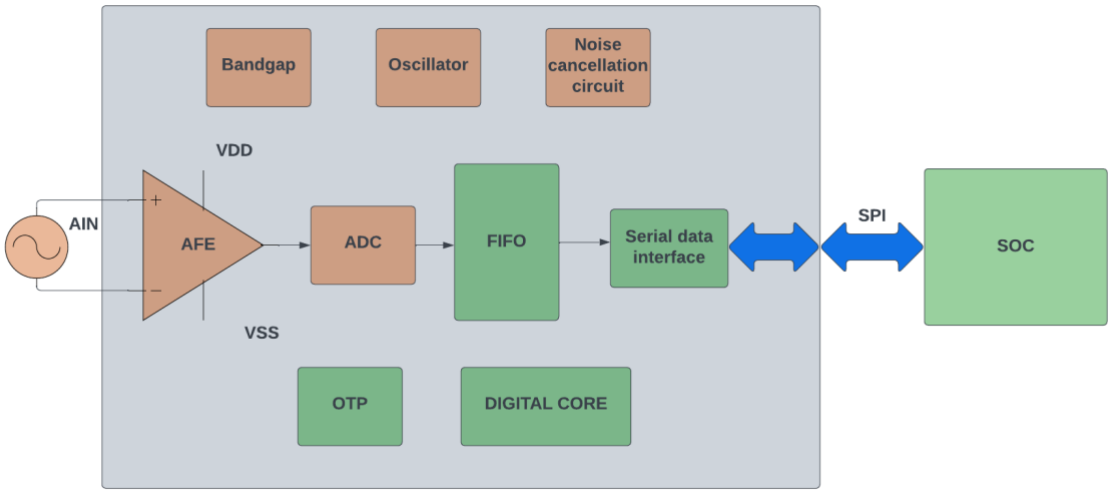

Figure 2: AMS Sub-System for digital SoC

II. LEGACY VERIFICATION ENVIRONMENT

A. VCD Based analog verification:

Environment for mixed signal SoC consisted of two separate flows for Analog and Digital. On the Digital side a traditional UVM environment was implemented while on the Analog side, digital simulation output captured as a VCD waveform dump was used as stimulus. The Analog simulation itself took place in a GUI based environment used for schematic capture and settings. Setup for a typical VCD flow for analog top simulations is captured in Figure 3 below.

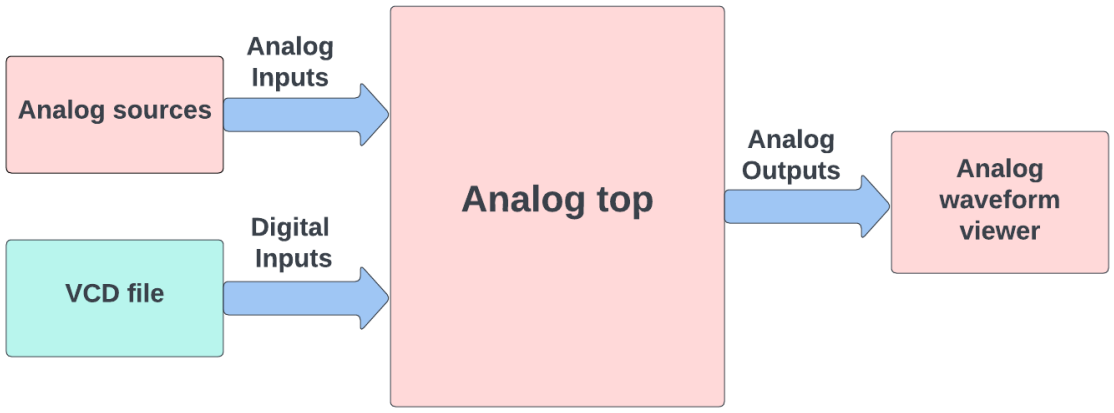

Figure 3: Analog design environment with VCD file to drive digital ports

Initially, the legacy environment was built upon the "analog on top" chip verification approach, using testbench library with analog drivers and VerilogA modules for creating a testbench to generate analog stimulus. However, mimicking the digital SPI driver command sequence to control the analog portion proved challenging. To overcome this, DV simulations were run to generate VCD files, which were then employed for analog simulation.

B. Verification gaps exposed in the legacy environment.

Yet, this process was cumbersome and introduced delays in the verification cycle. Furthermore, it prevented the use of well-architected Metric-driven DV UVM testbenches for analog simulations and suffered from limitations in

synchronizing analog and digital drivers. Consequently, a revamped approach was needed, exploring the reuse of DV UVM testbenches for mixed-signal simulation, encompassing spice netlists and SV RNM models.

The main challenge was the use of discrete stimuli in ASCII format VCD files, which only allowed for "1" and "0" to be driven onto the Analog SPICE netlist. The response was then sampled on waveforms and manually checked for accuracy. This approach also had several limitations:

- Inability to modify the stimulus mid-way through the simulation.
- All stimuli were non-random.
- All re-simulations had to start from time 0 and repeat the DC-initialization process.

III. NEW MSV ARCHITECTURE

In this section, author overviews the new revamped Mixed Signal Verification architecture:

A. New MSV Architecture:

To overcome the limitations highlighted, a new MSV architecture is needed. Figure 4 shows a new MSV testbench environment that includes both analog and digital design modules instantiated. This enables use of a well refined DV UVM testbench to drive the digital as well as analog portion of the design. As a result of this revamped approach, we have significantly enhanced functional coverage at the chip-level verification stage. Furthermore, analog verification has reaped the benefits of running nightly regressions at full digital speeds through DMS simulation. This approach has also empowered us to verify analog performance by seamlessly blending SPICE simulations with mixed-signal assertions in AMS simulations. Additionally, we have leveraged advanced digital verification concepts, including UVM, SV assertions, coverage analysis, constraint randomization, and more.

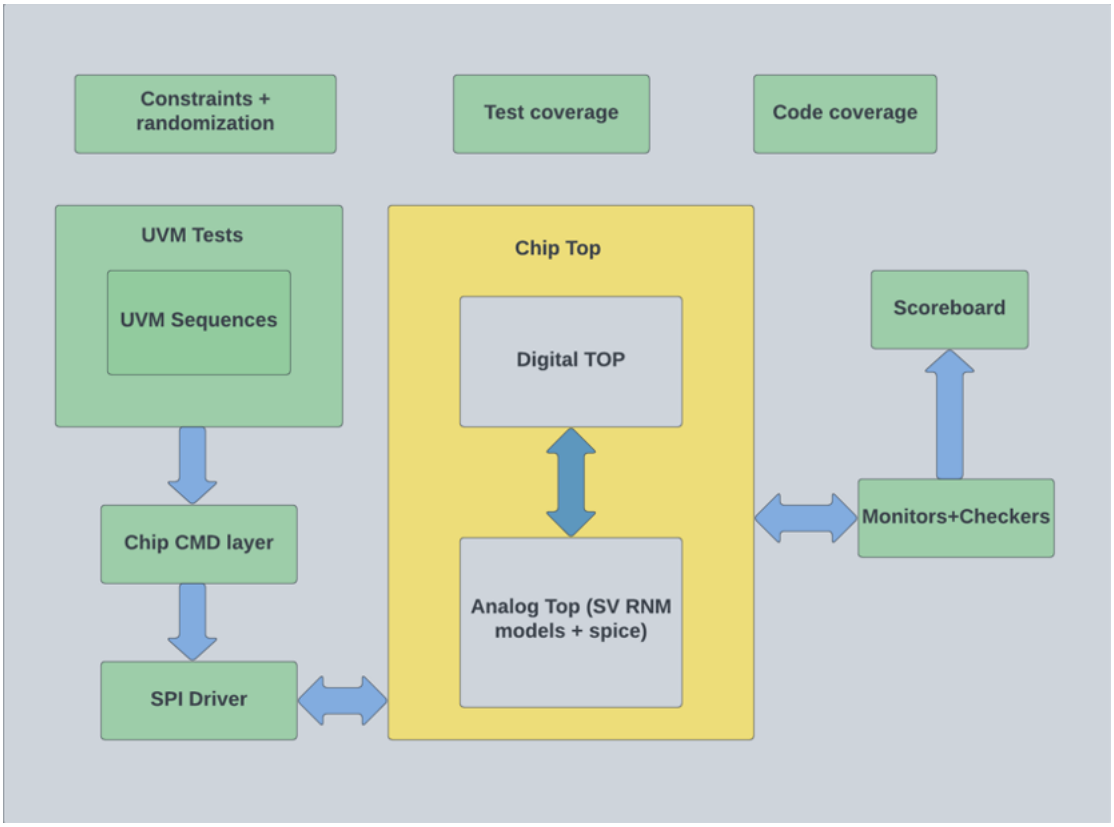

Figure 4: New UVM based DV Testbench to drive AMS DUT

IV. MIXED SIGNAL CO-SIMULATION SETUP

This section describes the setup needed to establish mixed signal co-simulation with spice in new MSV environment

A. Enablement of analog spice simulator in DV UVM testbench environment:

Initially the DV UVM environment was only used to run simulation with DMS (digital mixed signal) configuration. Hence only a digital simulation engine was needed. For co-simulation with spice, an analog simulation engine was also required. For this, an analog control file is needed to describe analog simulator options like min time step, start/end time, design corner info, design variables etc. An analog control file contains commands such as tran card, probes, and options cards, which control the behavior of the analog solvers. You can specify all the Spice simulation options in the analog control file. Sample analog control file is shown in Figure 5 below.

```
1
2 // Analog simulation control file.
3 // It specifies the options and analyses for the Spectre analog solver.
4
5 simulator lang=spectre
6
7 global 0
8
9 simulatorOptions options temp=27 tnom=27 scale=1.0 scalem=1.0 reltol=1e-3 \
10 vabstol=1e-6 iabstol=1e-12 gmin=1e-12 rforce=1 maxnotes=5 maxwarns=5 \
11 digits=5 pivrel=1e-3 checklimitdest=psf
12
13 tran tran start=1.79m stop=100m write="spectre.ic" writefinal="spectre.fc" \
14 annotate=status maxiters=5
15
16 finalTimeOP info what=oppoint where=rawfile
17
18 //modelParameter info what=models where=rawfile
19 //element info what=inst where=rawfile
20 //outputParameter info what=output where=rawfile
21
22 wave_out options rawfmt=sst2
```

Figure 5: Analog control file

An AMS control file is also needed to set up AMS configurations. The AMS control file centralizes all AMS controls needed for mixed signal simulations. Here are typical contents of the AMS control file:

- Inclusion of analog objects: SPICE/Spectre netlists, Verilog-A models, device models, and the analog control file (ACF)
- Design configuration (spice vs SV vs verilog-A etc.)
- Analog and digital connectivity
- Supply specification
- Handling of connect modules with connect rules.

The portmap statement tells the simulator which design blocks are going to be substituted as analog (using SPICE/Spectre netlists) or digital (using Verilog netlists). Figure 6 shows an example of such AMS control file.

```
1 // scs
2 simulator lang=spectre
3 include "$AH_ROOT/src/ah_models/misc_includes/amsControlSpectre.scs"
4 include "$AH_ROOT/src/ah_models/afe/afe_top.scs"
5
6 amsd{
7      ie vsup=2.5
8
9      portmap subckt=afe_top_die_a file="$AH_ROOT/src/ah_models/misc_includes/afe_top.pb"
10     config inst=ah_tb.ah_top_1.IANA.I_AFE_CH1.I_afe_top use=spice
11
12     ie connrules=CR_full_fast vsup=2.5  inst=ah_tb.ah_top_1.IANA.I_AFE_CH1 vdelta=0.000001 rout=0
13     ie vsup=2.5 instport=ah_tb.ah_top_1.IANA.I_AFE_CH1.Ibn_50nA_bg  currentmode=1 idelta=1e-9 rout=0 rz=5000M
14     ie vsup=2.5 instport=ah_tb.ah_top_1.IANA.I_AFE_CH1.Ibn_50nA_ptat_bg currentmode=1 idelta=1e-9 rout=0 rz=5000M
15     ie vsup=2.5 net=ah_tb.ah_top_1.IANA.I_AFE_CH1.Ibn_250nA_bg_temp currentmode=1 idelta=1e-9 rout=0 rz=5000M
16
17 }
```

Figure 6: AMS control file needed for command line AMS co-simulation flow

B. Connect module setup for mixed signal co-simulation:

The setup of the connect module can be incorporated within the AMS control file, contingent upon the specific requirements of the design. This configuration governs the insertion of connect modules and their associated parameters for electrical, logic, and real nets. Figure 6 illustrates an example of an AMS block with connect module settings tailored for a typical mixed-signal design.

Several considerations should be taken into account when configuring Connect modules, including the analog sampling speed, the parameter values for input/output impedance (rin, rout) and the nominal supply voltage value (vsup) for converting one domain signal to another domain. Incorrect settings in these aspects can lead to significant slowdowns in simulation and, in some cases, yield incorrect results. Therefore, careful attention to these parameters is crucial to ensure the optimal performance and accuracy of the mixed-signal verification process.

C. SV and Spice netlist extraction from analog design environment:

In the context of an analog design environment, it is imperative to extract both SystemVeriog (SV) and SPICE netlists. The command-line simulation flow can subsequently be configured to utilize either the SPICE netlist, the SV netlist, or a combination of both, depending on the specific requirements of the verification process. Notably, tool vendors offer support for extracting these netlists and converting them into a SPICE text file format, facilitating seamless integration into the simulation flow. This flexibility in netlist selection enhances the adaptability of the verification process, allowing for a tailored approach based on the unique needs of the analog design, and underscores the interoperability of tools in accommodating diverse simulation requirements

D. Command line simulation script with appropriate simulation switches

Finally, a master run script serves as the linchpin by consolidating all the aforementioned files for co-simulation. This comprehensive script encompasses the SV and SPICE netlists, AMS control file, Connect module setup, and any other relevant components essential for a cohesive verification process. Additionally, this script accommodates tool-specific simulation and debug switches, tailored for mixed-signal configuration. By orchestrating these diverse elements, the master run script streamlines the co-simulation workflow, ensuring a seamless integration of SystemVeriog and SPICE simulations while allowing for fine-tuned adjustments through tool-specific configurations. This consolidated approach enhances efficiency and convenience in managing the intricacies of mixed-signal verification.

V. ADVANTAGES OF NEW MSV ARCHITECTURE

This section describes several benefits of new Mixed Signal Verification architecture

A. Capability to run nightly regressions at full digital speeds (DMS simulations)

SV-RNM models prove to be invaluable in chip-level verification alongside digital RTL. Simulating the RNM model in conjunction with the digital System-on-Chip (SOC) at digital speeds allows for the inclusion of nightly regressions, enhancing the overall efficiency of the verification process. The capability to run regressions at digital speeds not only expedites the verification cycle but also fosters increased automation in the verification workflow. This heightened level of automation is pivotal, as it significantly contributes to the effectiveness of the regression testing, showcasing SV-RNM models as a key enabler in advancing automated verification methodologies at the chip level.

B. Increase in functional coverage for analog at chip level:

The integration of DMS models into a Design Verification (DV) based Universal Verification Methodology (UVM) environment has eliminated the simulation speed bottleneck. Consequently, a larger number of functional scenarios can now be executed, leading to a substantial increase in functional coverage. In this enhanced environment, digital control signals can be systematically varied through all possible states using constraint randomization. This expanded capability not only overcomes previous limitations but also ensures a more comprehensive exploration of functional scenarios, contributing to a more robust and thorough verification process in the DV-UVM framework

C. Capability to run nightly regressions for co-simulation tests with spice

The incorporation of the SPICE netlist into the command-line DV based Universal Verification Methodology (DV-UVM) environment has empowered the execution of nightly regressions with the SPICE netlist. Consequently, numerous new cosimulation test cases have been seamlessly integrated into the chip-level verification test suite, specifically focusing on critical design blocks like ADC and AFE using the SPICE netlist. This has enabled more robust verification of these important design blocks and the ability to evaluate their analog performance at the chip level.

D. Use of SV randomization for DMS and AMS co-simulation

By enabling this command line methodology, we can leverage a DV UVM testbench and System Verilog (SV) randomization in tests to verify numerous design configurations in both DMS and AMS-spice simulations.

Figure 7 illustrates the utilization of SV constraint randomization in a typical ADC test. This method allows for the randomization of various ADC design parameters, such as the number of device counts, ADC channels, resolution, gain, offset, sampling frequency, and more. These chip-level tests are conducted nightly. With just one AMS spice test, we can verify seven different configurations within a week's time.

```
`include "ah_seq_list.sv"
`include "ah_reset_if.sv"

import ah_reset_component_pkg::*;

class ah_adc_dual_8b_test extends ah_base_test;
  `uvm_component_utils(ah_adc_dual_8b_test)

  constraint test_cfg_c {
                        cfg.ah_cnt inside {1,2,3,4}; //supported range: 1-4
                        cfg.ch0_enb[0] inside {0,1};
                        cfg.ch1_enb[0] inside {0,1};
                        cfg.ch0_enb[1] inside {0,1};
                        cfg.ch1_enb[1] inside {0,1};
                        cfg.ch0_enb[2] inside {0,1};
                        cfg.ch1_enb[2] inside {0,1};
                        cfg.ch0_enb[3] inside {0,1};
                        cfg.ch1_enb[3] inside {0,1};
                        cfg.burst_length inside {16,32,64,128}; //supported range: 16-128
                        cfg.burst_count inside {[1:4]}; //supported range: 1-10
                        cfg.sample_width == 8; //inside {8,10,12}; //supported range: 8-13
                        cfg.clk_sel == ah_enum_pkg::OSC_CLK; //cfg.OSC_CLK; //supported values: OSC_CLK, EXT_CLK
                        cfg.ext_clk_freq_sel inside {EXT_CLK_1, EXT_CLK_2, EXT_CLK_3, EXT_CLK_4};
                        cfg.cali_enb inside {0,1}'h0; //bit
                        cfg.cali_gain == 16'h0082; // 0x80: 1 in 2.7 Fixed Point Format  0x82: 1 2/128 or 1.015625
                        cfg.cali_offset == 16'h2;
                        cfg.th_enb == 0; //bit
                        cfg.comp_sample_timer == 8'h1; //bit[7:0]
                        cfg.adc_samples == 8'h2; //bit[7:0]
                        }


  //Global members
  virtual ah_reset_if           ah_rst_vif;
  virtual ah_tb_verif_if        ah_tb_vif;
function void ah_base_test::randomize_cfg();
  assert(randomize()) else
    `uvm_fatal(get_type_name(), "Unable to randomize test configuration")

  `uvm_info(get_type_name(), $sformatf("Test CFG:\n%s", cfg.sprint()), UVM_NONE)
endfunction : randomize_cfg
```

Figure 7: SV constraint randomization for a typical adc test

E. Use of SV assertions to check digital and spice signals in design hierarchy.

Use of DV based UVM test bench also enables writing SV assertions to verify key digital and analog nets in AMS and DMS simulations. SV assertion use for analog nets enables automatic checking which in turn verifies correct device behavior for all possible stimulus variations. Figure 8 shows a working example to demonstrate SVA on electrical net to check node voltage does not fall below 0.6V.

```
  //SV assertions for spice nets:

    check_bg_voltage: assert property (
    @(posedge clk)
    assert_en |-> ##1 $cgav("ah_tb.ah_top_1_IANA.IAFE_CH1.Vbg","potential") < 0.6
  );
```

Figure 8: SVA (SystemVeriog Assertion) on spice electrical net in design

Mixed-signal waveform viewers will also display assertions that are fired by time on the waveform window, with green and red regions shown in figure 9. Green represents that an assertion has finished and passed, while red indicates that an assertion has failed.

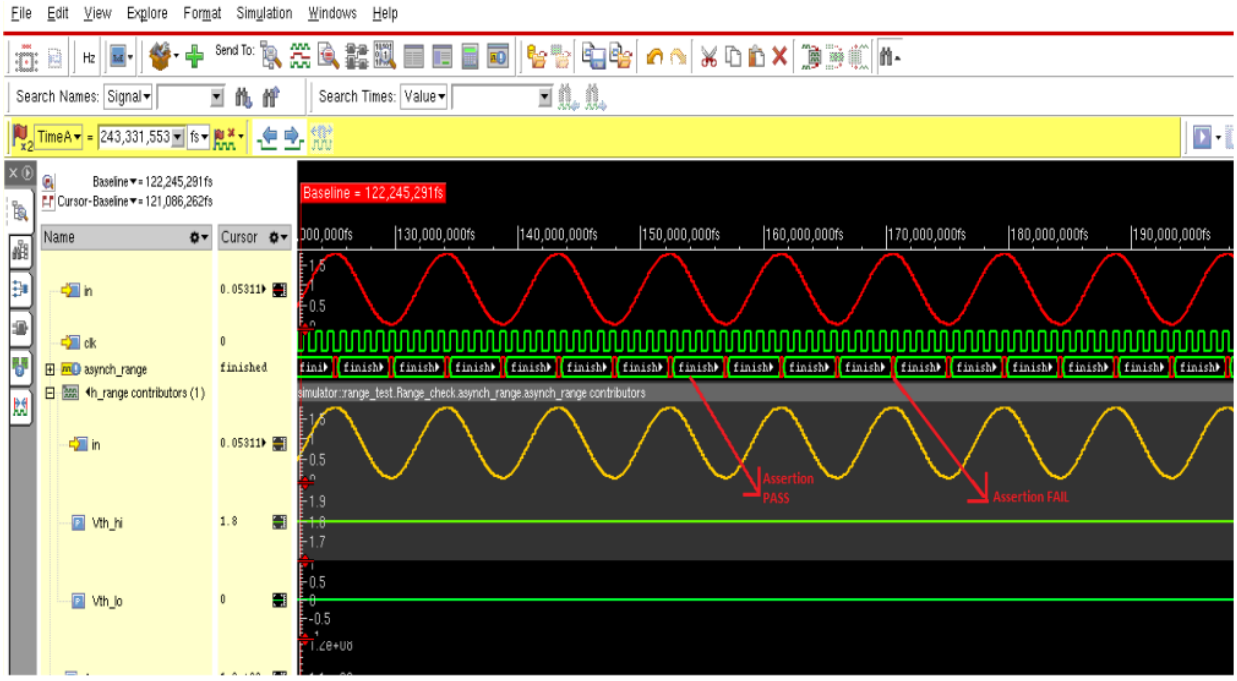

Figure 9: Mixed signal SV assertions pass and fail regions

F. Use of advanced digital tools to analyze and debug simulation

The command line DV flow facilitates the utilization of advanced digital tools and debugging switches for simulation and verification. These sophisticated digital tools offer a platform for implementing plan-based verification in mixed-signal designs. The capability to execute DV regressions using digital simulation and verification tools supports more automated verification processes. This, in turn, enables the generation of requirements-based reports, automatic creation of pass/fail information, and the development of a summary table to monitor verification progress

VI. FUTURE WORK

A. Explore SV EEnet modeling for analog

Even though enabling the AMS co-simulation flow allows for verifying analog performance at the chip level, this approach may not be feasible if the design is very complex and will create a bottleneck in simulation time. In such scenarios, more advanced methodologies like SV EEnet modeling [5] can be leveraged. With SV EEnet modeling, an SV model is capable of modeling electrical quantities like voltage, current, frequency, and phase on a single net using custom user-defined nettypes (UDN). This way, the SV model will be closer to the schematic design in terms of electrical accuracy.

SV RNM is a great solution any time a single value needs to be transferred from the output of one block to the input of another block. When one module is defining an output voltage and the next module is using that voltage as its input, RNM is all that is required to cleanly model the transfer characteristic. For typical high-level modeling operations, that is all that is needed for most of the signals. However, occasionally, there are cases where the wire is not just unidirectionally passing a simple voltage value but needs to define an interaction between the modules. Some common examples are listed below:

- One or more voltage drivers with finite resistance connected to a resistive load will result in a net voltage controlled by resistor-divider relationships, not just voltages.
- Current sources driving into a resistive load will result in a voltage dependent on both current value and load resistance value.

- Power supplies connected to multiple blocks which each sink current from the power supply may need to simultaneously provide voltage value in one direction and report current loading back to the supply through the same interconnect.
- Capacitive loading effects might be important to model for certain analog designs

These effects are not needed all the time, but when they are required for proper system operation, there is no way to directly implement them in a simple RNM context; an EEnet is required in such situations

B. Adopting UVM-AMS standard for mixed signal verification

DV UVM testbenches lack an easy way to generate analog source signals to drive analog nodes. A more sophisticated methodology (UVM-AMS) [6] is currently under development for this purpose. The use of such a methodology will enable more thorough verification of analog components at the system level. The UVM-AMS standard will provide a unified approach that allows UVM to be more mixed-signal aware, resulting in improved verification of analog and mixed-signal components and subsystems. The objective of the UVM-AMS standard is to standardize a method for driving and monitoring mixed-signal nets within UVM, including stimulus, scoreboarding, and analysis.

VII. SUMMARY

In conclusion, the paper provides details on the legacy verification flow and identifies gaps in the verification process. It then introduces a command-line-based MSV methodology along with implementation details aimed at enhancing the verification process. This involves integrating the AMS sub-system's testbench into the SoC environment and seamlessly incorporating advanced features at the SoC level. Additionally, the paper underscores the adaptability of this environment, allowing multiple projects to employ the same MSV flow and potentially facilitating the widespread adoption of a unified Mixed-Signal methodology.

As a result of implementing this improved MSV architecture, an increasing number of analog designers have begun to run mixed-signal verification from the command line to test their spice blocks at the chip level and leverage advanced verification techniques. Analog designers are also adopting the practice of reusing advanced testbenches created by the DV team. The paper further highlights several improvements in verification following the adoption of the new MSV architecture.

ACKNOWLEDGMENT

I would like to express sincere gratitude to people from Meta reality labs and Cadence Design Systems for providing useful insight and guidance in implementing this novel mixed signal verification methodology.

REFERENCES

[1] IEEE 1801-2018: IEEE Standard for Design and Verification of Low-Power, Energy-Aware Electronic Systems: https://standards.ieee.org/ieee/1801/6767/
[2] IEEE Std 1800-2012, IEEE Standard for SystemVeriog Unified Hardware Design, Specification, and Verification Language: https://ieeexplore.ieee.org/document/6469140
[3] V. Hamolia and V. Melnyk, "A Survey of Machine Learning Methods and Applications in Electronic Design Automation," 2021 11th International Conference on Advanced Computer Information Technologies (ACIT), Deggendorf, Germany, 2021, pp. 757-760, Doi: 10.1109/ACIT52158.2021.9548117
[4] Cadence Mixed signal verification community: https://www.cadence.com/en_US/home/solutions/mixed-signal-solutions/mixed-signal-verification.html
[5] Cadence Design Systems, “Using EEnet to perform Electrical Equivalent Modeling in SystemVeriog”, Tutorial paper, 2017
[6] UVM-AMS working group (accellera) https://www.accellera.org/activities/working-groups/uvm-ams