UVM Based Mixed-Signal Verification of a Display PMIC Designed for OLED Display Applications

Vijay Kumar & Adnan Malik
Samsung Semiconductor India Research

***Abstract*- The objective of this paper is to introduce readers to a Display PMIC designed to power OLED mobile display panels, compile some of the major challenges faced in verifying the PMIC in a UVM based mixed-signal verification environment (MSV), and the how we addressed those challenges. The PMIC is comprised of highly efficient multiple switching converters and low dropout linear voltage regulators to support different voltage rail requirements. Major challenges encountered while creating the MSV setup for the PMIC and running simulations were: (a) Handling interaction of signals between different blocks that are at different abstraction levels such as logic and real number under different supply domains (b) Mechanism to cover fault scenarios in a real number model (RNM) based Digital and Mixed-Signal (DMS) verification environment with UVM, that are typically addressed in the transistor-level abstraction and (c) Handling mixed-signal specific driver conflicts arising out of logic and real drivers, modelling style, etc. This paper explains about the aforementioned challenges in detail and how we addressed those challenges and ensured coverage of all the required test metrics for the PMIC.**

I. INTRODUCTION

The Display PMIC discussed in this paper is comprised of highly efficient switching regulators or converters and a low-dropout linear regulator (LDO). In addition, the PMIC consists of typical housekeeping circuits such as reference voltage and bias current generators, an on-chip LDO to provide bias voltage, a clock generation unit and circuits for protection logic. The core digital block drives the enables for the converters and other analog blocks. All converters are input to output isolated, have integrated rectifiers, come with TR Segment Control (TSC) and are designed to operate from a single-cell Li-Ion battery and charger. The register banks inside the digital and the converter outputs are programmable by I2C protocol. Unlike other chips that traditionally come with single slave address for all registers in the register bank, this Display PMIC comes with two slave addresses, each with its own set of register banks. While one set of registers are programmable by I2C protocol using the serial clock and data lines, the other set of registers are programmable by a separate set of lines. There is a GPIO input that decides which set of lines are used for I2C read and write.

The PMIC can drive the display with heavy panel current and high driving voltage to cover high brightness mode (HBM) with very good efficiency in light load condition. Each converter has some unique features of its own. One converter has Continuous-Conduction-Mode (CCM) topology that is configurable under light load condition (normal mode). It can change the operating mode automatically, and can also be low-power driven. Another converter is sequentially driven, and delivers an output current that is configurable based on the supply and its output voltage. Another one has an Always-ON Over Current Detect (OCD) feature. The PMIC finds its applications in OLED power supply for mobile devices, camcorders, digital cameras and multimedia players.

The supply voltage of all the converters remains the same. The fixed switching frequency of the converters allows the use of low-profile inductors and ceramic capacitors to minimize the thickness of the OLED driving circuit. The first converter in the power sequence supplies power to an OLED panel source driver IC and automatically enables the second converter which contains a pre-converter and an LDO. The last two converters in the sequence are used for the positive voltage and negative voltage of the OLED panel respectively. Each of these two converters are designed to have a default voltage, but can be programmable. The output voltage and voltage step of all the converters can be decided by counting the pulse numbers of any one of the two I2C interfaces.

Some of the main features of the Display PMIC are:

- Adaptive power-on sequence & shutdown mode
- Fast discharge of outputs after shutdown
- Output high-Z by Always-On Display (AOD) mode
- Single-byte and multi-byte write & read through one of the I2C interfaces
- Under-voltage lock-out (UVLO)
- Soft-start with inrush current limitation
- Thorough protection by triggering interrupts against various fault scenarios

A simplified block diagram of the architecture of the Display PMIC is shown below.

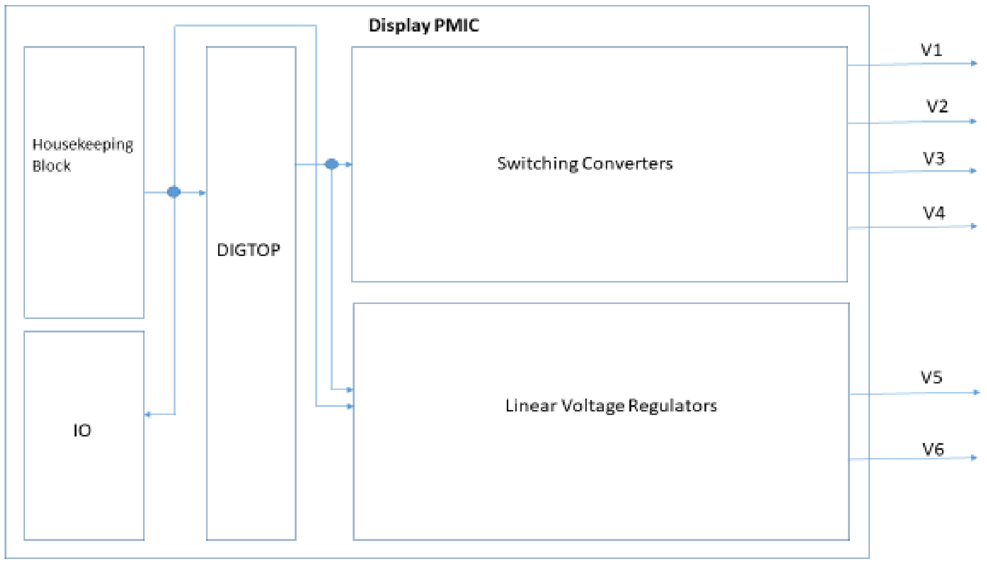


Figure 1. Display PMIC Architectural Block Diagram

II. POWER SEQUENCE

Power rails follow certain power up/down sequence with soft-start time and delay time as indicated in the below timing diagram. Converter 1 powers up first, followed by auto-enabling of Converter 2 (with a Pre-converter, viz., Converter 3 and an LDO) which is then followed by enabling Converters 4 and 5. A symmetric sequence during power down is followed. However, Converters 4 and 5 can be enabled to start ramping up without enabling Converter 1 at all. Usually under normal start-up sequence controlled by the Display Driver Interface (DDI), Converter 1 always powers up before all the other converters.

During power down, if Converters 1, 4 and 5 shut down simultaneously, there will be possible FOS (Front Of Screen) artifact. So as a general rule, Converters 4 and 5 are discharged completely before powering down Converter 1, unless any other consequence that is much more severe than FOS artifact happens. Otherwise, the PMIC would not be able to ensure proper operation. In other words, the Converter 1 will not be discharged unless the external supply is lesser than the internal POR threshold and the IO block supply is off.

The PMIC monitors the external supply voltage by checking the specified UVLO levels at rising edge and falling edge to prevent the system from malfunctioning at insufficient power level. The PMIC monitors the IO supply voltage to check if it crosses the POR threshold at rising edge so as to enable the digital logic. The POR is disabled when the IO supply voltage goes below the threshold at falling edge, which can be different from the rising threshold.

The PMIC contains protection circuits to guard it against various fault scenarios such as the ones shown below, by either disabling the converters or by enabling current limiters. Whenever a fault scenario is encountered, the corresponding interrupt requests (IRQB) are triggered from the registers by default, but these interrupt requests can also be masked by the IRQM register if needed.

- Thermal shut down (Pre TSD and TSD)
- UVLO
- Short Circuit Protection (SCP)
- Input Over Voltage Protection (IVP)
- Over Voltage & Under Voltage Protection (OVP & UVP)
- Over Current Detection (OCD)
- Startup Short Detect (SSD)

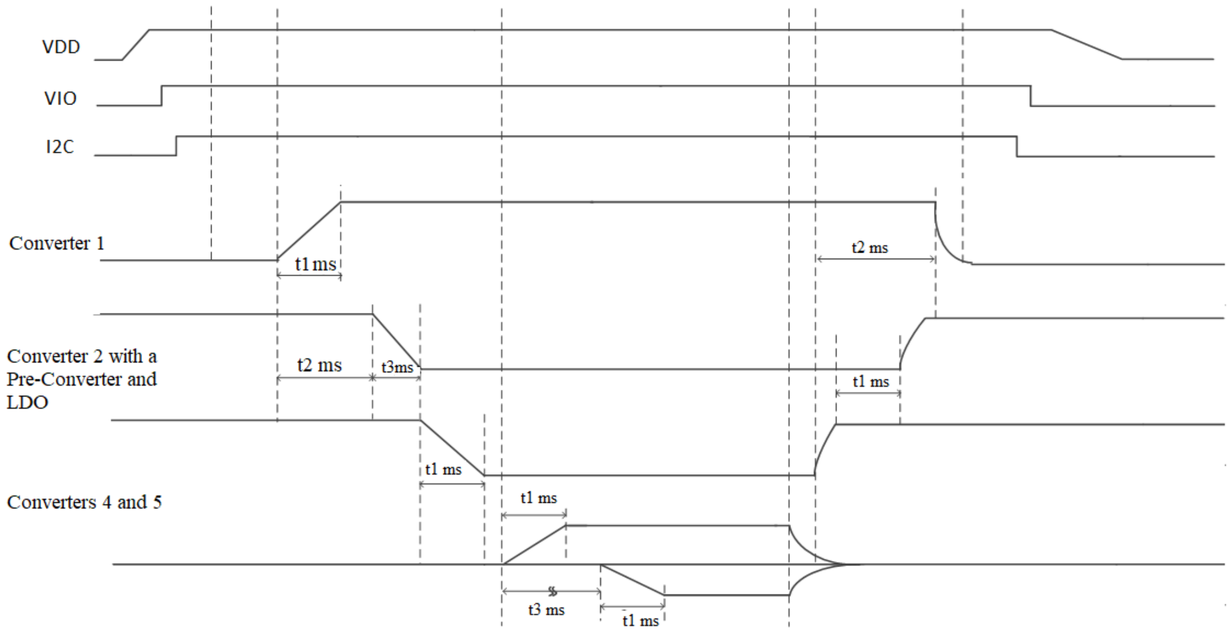


Figure 2. Display PMIC Power Sequence

III. MIXED-SIGNAL VERIFICATION FLOW AND CHALLENGES

The verification flow followed for this PMIC was similar to the one discussed in [6]. We started with the schematic of the PMIC chip-top level, from which we created a mixed-signal configuration. In this configuration, we bound the blocks of interest to "symbol" view, so that we can use behavioral real-number models (RNM) for the time-consuming analog blocks. We then extracted a Verilog-AMS netlist of this configuration which was post-processed to convert it to SV netlist, so that we can avoid electrical disciplines. This is because our intention is to keep the simulation purely event-driven and perform a Digital Mixed-Signal (DMS) simulation for regression. This SV netlist along with the RTL codes for the digital block and with the developed RNM models for analog blocks (Verilog-AMS *wreal*), were compiled and simulated using logic simulator. As we were primarily interested in top-level integration checks, we were not in need of SPICE level accuracy. Hence we integrated the developed RNM models together and took care of the Real-to-Logic (R2L) and Logic-to-Real (L2R) conversions using appropriate connect modules available in the simulator installation directory.

Creating analog behavioral models was by itself a multi-step task, which consisted of model creation, testbench creation, simulation and validation of models against the corresponding schematic. The developed models were simulated and the results were compared with their respective schematic simulation results using common testbenches for both representations.

A diagrammatic representation of the adopted flow is shown below.

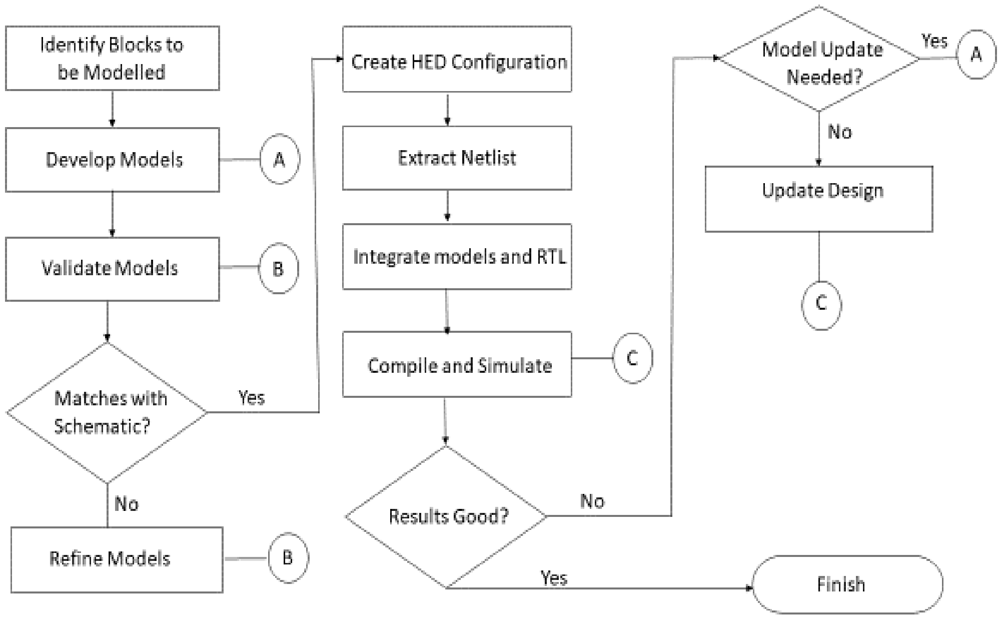


Figure 3. Display PMIC Verification Flow

Verification of such a PMIC has always been prone to challenges, either related to design updates in the course of verification owing to updates in the spec, or related to handling of different types of signals (datatypes or *disciplines* in the context of verification tools) owing to different levels of abstraction used in the design. One of the most common challenges as a result of design updates would be the addition or removal of pins either in the RTL or schematic or both. Every time such changes happen, it requires the verification team to sync to the latest version of the design to keep the analog and digital blocks connectivity compatible with each other and re-extract the netlist. While such changes are inevitable in the design, it would lead to re-execution of the tests in the test plan so as to make sure no functionality is broken.

Traditionally, for any change(s) in the design, verification engineers would be mandated with the task of re-running the whole regression in order to make sure no functionality is broken (unless otherwise intended). This is because regression setup has evolved over time and we have a high degree of automation available from EDA vendors to ease the regression process. However, this comes at the cost of regression time which has always remained a bottleneck despite various improvements in the simulator performance. This in turn is owing to the huge number of test cases which can be attributed to the complexity of today's designs. In our Display PMIC verification activity, we are addressing this challenge with a two pronged approach.

Firstly, we identify all the test cases to be executed, categorize the tests under different heads such as I2C read/write, Power up/down sequence, Protection/Interrupts, Dynamic Voltage Scaling (DVS) and Connectivity checks. We then maintain a table for each design block that goes through pin changes so that the designer would update the table with the information of the pin changes for the block. We would then decide on updating the corresponding RNM of the block with the changes as applicable. If the models are updated, as a first step only those test cases which will be effected by the changes would be re-executed so that we verify the change in behavior with respect to the pin changes. This saves lot of time in validating the changes and informing the designer of any discrepancies faced during verification, instead of waiting for the whole regression to complete and then looking for the results.

Later, we enable regression by adding SV-UVM assertions in each test case that checks for the status registers and converter outputs to display the pass/fail status. While regression of the complete test suite is carried on as a nightly activity, any local changes in the design/test case/models/testbench etc., are quickly verified by running the specific test case, thereby optimizing the execution time. One example of UVM based assertions added for verifying the converter outputs (in real number) and the POK status registers is shown below.

```
   p_sequencer.i2c_intf.I2C_WRITE (SLAVE_ADDR, REG1, VALUE1); //Enabling Converter 1 (Converters 2 & 3
are auto-enabled)
   #t_converter_1_enable; // Converters 1-3 enable wait time
   p_sequencer.i2c_intf.I2C_WRITE (SLAVE_ADDR, REG1, VALUE2); //Enabling converters 4 & 5

   #t_converter_4_5_enable; // Converters 4 & 5 enable wait time and display all converter output values
   $display("%f %f %f %f %f", V_Converter_1, V_Converter_2, V_Converter_3, V_Converter_4, V_Converter_5);

   //POK Registers Check
   p_sequencer.i2c_intf.I2C_READ (SLAVE_ADDR, REG2, i2c_rdata);
   $display("POK_STATUS at %f = %h", $time, i2c_rdata);
   if(i2c_rdata == expected_data)
   `uvm_info(get_full_name(), "POK Good\n", UVM_LOW)
   else
   `uvm_error("MAIN_CHIP", "POK Bad\n");
```

The other major challenge is that of handling interaction of signals between different blocks that are at different abstraction levels, which is even more pronounced. As a PMIC, the chip operates with different supplies for different parts of the design. While the whole digital block is at a particular supply domain, within the analog there exists three supply domains. One supply for powering up the common housekeeping blocks and the converters, another supply for powering up the IO cells, and then we have the on-chip LDO which is always ON and supplies power to various other internal blocks. Though care is taken in the design such that appropriate level shifters are added wherever there are signals crossing from one supply domain to the other, there could also be scenarios such as TIE_HIGH and TIE_LOW cell outputs connected to analog and digital blocks without level shifters. When it comes to mixed-signal verification involving R2L and L2R conversions, it becomes the duty of the verification engineer to associate the cells and terminals at analog (real) / digital boundaries with the correct supply voltages and high/low thresholds for proper conversion between real and logic values.

However, there are occasions where this would lead to verification artifacts like what we faced in the course of execution. At first, the digital block was power by the on-chip LDO output and the cells inside IO block were powered by the main VDD supply, which is the global supply for the design and is also much higher than the on-chip LDO output. So the verification setup contained the R2L and L2R connect modules supply specification accordingly. Later, the design was updated such that the IO cells too received their supply from the on-chip LDO. Since this change was implemented by changing the supply connections only at higher levels of the design, the verification team was not explicitly made aware of this change, but rather asked to just sync to the latest version of the design and re-extract the netlist. While simulating the new design, as the supply specification of the connect modules inserted on the nets between real and logic abstractions inside the IO cells used the higher global VDD supply (and so higher threshold for conversion), the conversion didn't happen as expected leading to functional failure. This is illustrated in the below figure. Once we came to know about the change in the supply, we first thought of adding a new supply specification for the connect module by specifying the instance names of all the logic primitives inside IO. But since that was huge in number and hence difficult to specify, we reduced the threshold used for conversion with the global VDD supply and got expected results.

However, a more scalable solution would be to move from static connect modules to SupplySensitive connect modules so that the connect module inserted on a net would refer to the connected port's supply. This would avoid changing the threshold parameters of the connect modules and also ensures correct conversion with dynamic and robust supply specification for the connect modules.

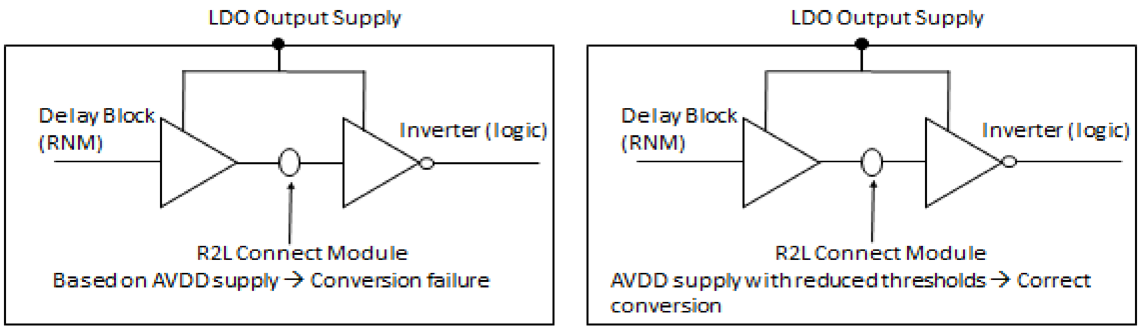


Figure 4. Connect modules with adjusted threshold parameters

IV. UVM TEST CASES FOR TRIGGERING FAULT SCENARIOS IN THE RNM MODELS

As mentioned in Section 1, the PMIC has to guard itself against various fault scenarios. Since we use RNM blocks and not the transistor level schematic, these fault scenarios may not occur in the simulation as we do not model these scenarios in the RNM. So in order to run test cases that check for the protection logic against such scenarios, we had to set the corresponding flag for each fault condition and see if the fault status registers and interrupt requests (IRQB) are triggered accordingly. An example of such a UVM test case that sets the flag for Over Voltage Protection (OVP) is shown in the below pseudo code.

```
p_sequencer.i2c_intf.I2C_WRITE (SLAVE_ADDR, REG1, VALUE1); //Enabling Converter 1 (Converters 2 & 3
are auto-enabled)
#t_converter_1_enable; // Converters 1-3 enable wait time
p_sequencer.i2c_intf.I2C_WRITE (SLAVE_ADDR, REG1, VALUE2); //Enabling converters 4 & 5

#t_wait_time_for_flag_set; //Wait time before setting the flag
uvm_hdl_force("top.Main.I_CONV_1.OVP", 1'b1); //Flag for OVP condition set for Converter 1


//OVP Register Check
p_sequencer.i2c_intf.I2C_READ (SLAVE_ADDR1, REG_OVP, i2c_rdata);
$display("OVP_STATUS = %h", i2c_rdata);
if(i2c_rdata ==  expected_data) //Checking the OVP register status
`uvm_info(get_full_name(), "OVP interrupt generated", UVM_LOW)
else
`uvm_error("MAIN_CHIP", "OVP interrupt generation failed");

#t_wait_time_for_flag_release; //Wait time before releasing the flag
uvm_hdl_force("top.Main.I_CONV_1.OVP", 1'b0); //Flag for OVP condition released for Converter 1
```

Once the OVP condition is triggered for Converter 1, all the converters shut down during which we check the status registers and display pass/fail status as shown above. After this, we re-enable the converters to see if they would power-up, followed by disabling of all the converters.

```
//Power Up
p_sequencer.i2c_intf.I2C_WRITE (SLAVE_ADDR, REG1, VALUE1); //Enabling Converter 1 (Converters 2 & 3
are auto-enabled)
#t_converter_1_enable; // Converters 1-3 enable wait time
p_sequencer.i2c_intf.I2C_WRITE (SLAVE_ADDR, REG1, VALUE2); //Enabling Converters 4 & 5

//Power Down
#t_wait_time;
p_sequencer.i2c_intf.I2C_WRITE (SLAVE_ADDR, REG1, VALUE1); //Disabling Converters 4 & 5
```

```
#t_converter_1_enable;
p_sequencer.i2c_intf.I2C_WRITE (SLAVE_ADDR, REG1, VALUE0); //Disabling Converters 1-3
```

V. TWO SLAVE ADDRESSES AND DRIVER CONFLICTS

As mentioned in Section 1, the Display PMIC comes with two slave addresses. The converter outputs are programmable from I2C by either serial clock/data signals or a separate set of signals, and a GPIO input decides which one is used. Since the abstraction level of each analog block was behavioral RNM, the datatype of all analog boundary ports including serial clock and serial data pins was *wreal*. In the schematic, the clock and data lines were pulled-up to the IO supply level through pull-up resistors. Hence, these pins that are toggled from the I2C interface were also driven with *wreal* values from the RNM models, thus leading to conflicts (X states). To avoid this problem, these pins were set to high impedance in the models so that the control is only from I2C and not internal to the design. This was imperative because any read/write operation on the registers via the clock and data pins have to be controlled only through the digital interface during verification, so that these operations are well controlled by the user.

Yet another issue with regard to driving values onto the input pins of the analog blocks happen due to the component modelling, something that is specific to mixed-signal verification only. Since passive elements such as resistors, inductors and capacitors are symmetric in nature with respect to their terminals, they can be added in the schematic without worrying about the input and output port consideration with respect to the connected block's terminal. However, as we are modelling all analog blocks including these components as RNM, the model considers the port direction for functionality. This leads to conflict when the port connection in the schematic does not correspond to the way the component is modelled. An illustration of the issue that we encountered with respect to resistor modelling and the way it was resolved is shown below.

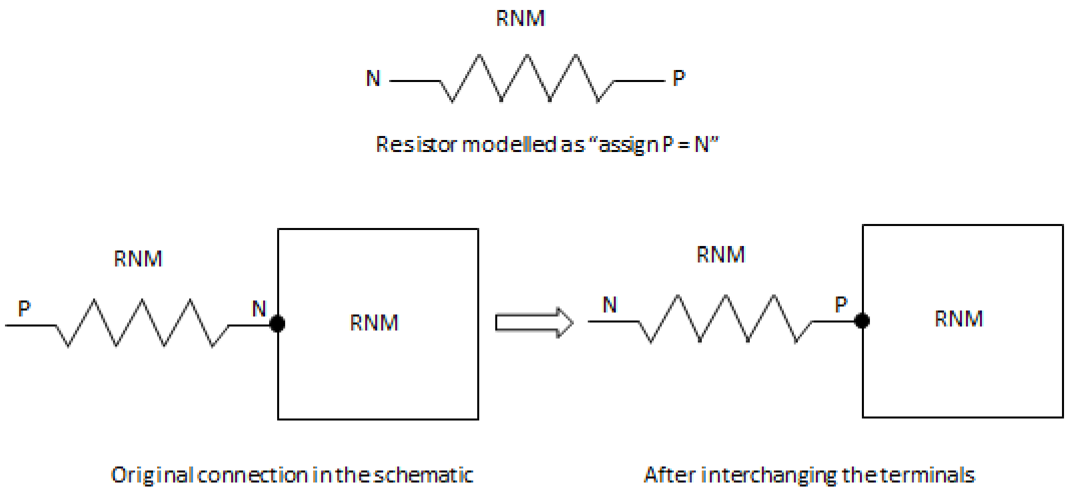

RNM
N
P
Resistor modelled as "assign P = N"
RNM
P
N
RNM
RNM
N
P
RNM
Original connection in the schematic
After interchanging the terminals

Figure 5. Resistor modelling – before and after interchanging the terminals

As shown above, with the model as P=N, we did not get the intended functionality since the N terminal was connected as input to the connected block as shown on the left-hand side. We found multiple such scenarios, including one on the main supply line. After interchanging the connections in the schematic as shown on the right-hand side, the issue was resolved.

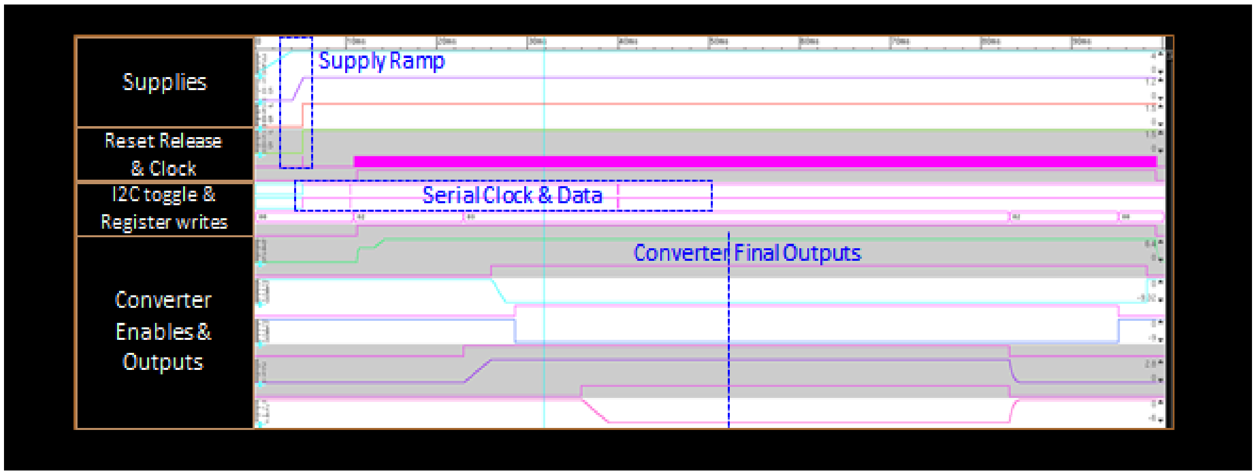
VI. RESULTS AND CONCLUSION

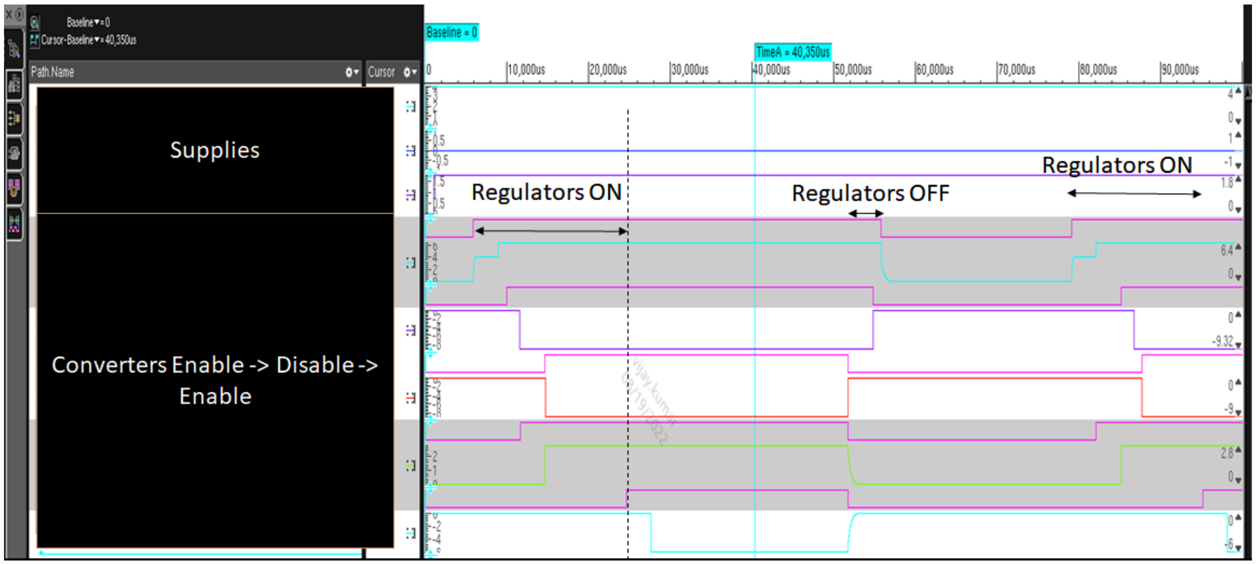
After addressing all the challenges as shown above, we were able to complete execution of various test cases in a UVM based mixed-signal verification environment. We could achieve adequate coverage by executing test cases across various scenarios spanning I2C read and write, power sequences with all required combinations to be tested, protection logic tests, and dynamic variation of converter outputs by writing appropriate register values. We added appropriate UVM based assertions to check the status registers such as POK, Interrupts, etc., and display pass/fail status in the results, thus enabling vManager based regression of all tests. We also implemented level shifter checks and pin connectivity checks by passing values between analog and digital blocks, wherein the value supplied on one side is checked for propagation on the other side to ensure there are no pin mismatches (induced either due to improper spec or human-induced error during design).

In summary,

- 142 test cases were completed on time with this approach spanning across various scenarios such as
    - Power-on and power-off
    - Write and read from dedicated clock and data lines or separate set of lines using I2C protocol
    - Dynamic Voltage Scaling
    - Protection logic
    - Pin connectivity checks between analog and digital
    - Level shifter checks across supply domains
- 45 VAMS *wreal* models were developed and validated against the schematic
- 15 weeks TAT for the whole activity

Plot of some of the key test results are shown below for illustration.





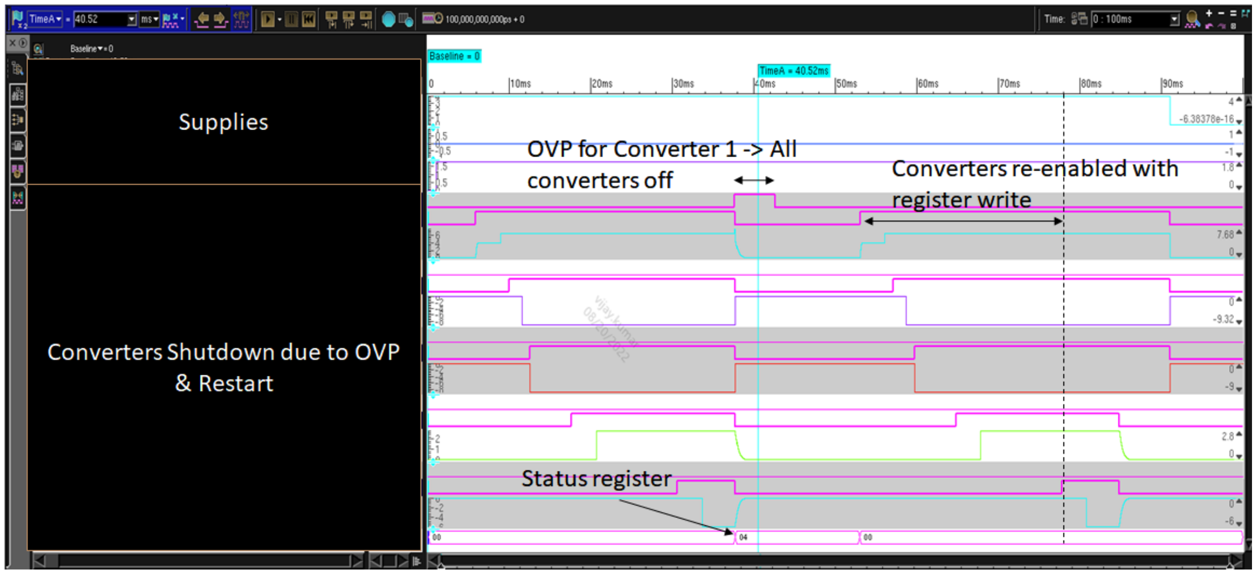

Supplies
Converters Shutdown due to OVP & Restart
OVP for Converter 1 -> All converters off
Converters re-enabled with register write
Status register
TimeA = 40.52ms
Baseline = 0

Figure 6. Basic Power Sequence Waveforms

As a future scope for improvement in the mixed-signal verification flow, we are planning to start adopting SupplySensitive connect modules for R2L & L2R conversions across different supply domains to ensure that conversions happen as per the supply of the block where the connect modules are inserted. This would ensure that we do not end up with false isolation and power-off of the connected blocks leading to unwanted debugging times, and also avoids tampering with the global connect module parameters such as threshold and output resistances. We are also planning to migrate from VAMS *wreal* RNM models (and SV-RNM models for other PMIC designs) to SV-EEnet models to account for voltage, current and impedance in the models so that loading effects are considered in the models similar to schematic level abstraction.

Acknowledgment

Authors would like to acknowledge the following people for providing all the required technical information towards execution of the activity, and for the constant support and guidance towards completion of the work.

1. Manish Parmar, Manish Goel, Balajee Sowrirajan, Deepa Devendran, Shyam Prasad Venkata Voleti, Divij Kumar and Vinayak Hegde of **Samsung Semiconductor India Research**
2. YuBeom Jeon of **Samsung Electronics**

References

[1]. Kiyeong Kim, Hwan-woo Shim, Chulsoon Hwang, "Analysis and Solution for RF Interference Caused by PMIC Noise in Mobile Platforms", IEEE Transactions on Electromagnetic Compatibility, June 2019
[2]. Abdullah Abdulslam, Patrick P. Mercier, "A Symmetric Modified Multilevel Ladder PMIC for Battery-Connected Applications", IEEE Journal of Solid-State Circuits, December 2019
[3]. Chunlei Shi, Brett Walker, Eric Zeisel, Brian Hu, Gene McAllister, "A Highly Integrated Power Management IC for Advanced Mobile Applications", IEEE Custom Integrated Circuits Conference, September 2006
[4]. Nicola Dall'Ora, Sara Vinco, Franco Fummi, "Functionality and Fault Modeling of a DC Motor with Verilog-AMS", IEEE 18th International Conference on Industrial Informatics, July 2020
[5]. Constantina Tsechelidou, Nikolaos Georgoulopoulos, Alkiviadis Hatzopoulos, "Design of a SystemVerilog-Based Sigma-Delta ADC Real Number Model", 22nd Euromicro Conference on Digital System Design, August 2019
[6]. Vijay Kumar, Shrikant Pattar, Yaswanth Chebrolu and Vinayak Hegde, "Harnessing SV-RNM Based Modelling and Simulation Methodology for Verifying a Complex PMIC designed for SSD Applications", DVCON INDIA 2022
[7]. Nikolaos Georgoulopoulos, Alkiviadis Hatzopoulos, "Real number modeling of a flash ADC using SystemVerilog", Panhellenic Conference on Electronics and Telecommunications, November 2017
[8]. Nikolaos Georgoulopoulos, Alkiviadis Hatzopoulos, "Efficiency evaluation of a SystemVerilog-based real number model", 7th International Conference on Modern Circuits and Systems Technologies, May 2018
[9]. Nikolaos Georgoulopoulos, Athanasios Mekras, Alkiviadis Hatzopoulos, "Design of a SystemVerilog-Based VCO Real Number Model", 8th International Conference on Modern Circuits and Systems Technologies, May 2019