Real Number Voltage aware behavioral modeling and verification of SRAM subsystem with Unified Power Format (UPF)

Amit Singh, STMicroelectronics, Greater Noida, India (amit.singh@st.com)

Isha Sharma, STMicroelectronics, Greater Noida, India (isha.sharma@st.com)

Abstract— **Power management in modern SoCs is becoming increasingly complex due to several evolving factors and design intricacies as compared to earlier times. To support this increased functionality while respecting strict power budgets of these SoCs , chips need to be designed to operate efficiently in various low-power modes. Such complex designs also need stringent low power verification techniques which can be performed early in the design cycle to avoid late cycle debugging issues. UPF IEEE 1801 standard [1]-based Soc verification for complex SoCs and subsystems plays a key role to verify the power management functions at RTL stage. This paper presents a solution of mimicking the power functional behavior using the real voltage values in digital simulations leading to voltage awareness in modeling of the power function of Ips and subsystems critical to achieve the power intent of the SoC. Using voltage-aware modeling with real-number voltage variables allows designers to transact actual voltage values during RTL simulation thereby improving accuracy and verification coverage [2]of power critical blocks and systems. This modeling methodology is scalable and can be applied across a wide range of hard macro IPs and subsystems. This methodology is demonstrated through a memory subsystem comprising a Retention SRAM (RETmem) and a Bias Generator (BIASG) block which is responsible for generating specific bias voltage for different RETmem operating modes as per the system's low-power design requirements.**

Keywords- Low Power, behavioral model, voltage aware, real number modeling, SystemVerilog, UPF IEEE 1801

I. INTRODUCTION

Today's most Low power SoC designs involve a complex interplay of various techniques and strategies wrapped inside different blocks to manage power consumption requirement while maintaining performance and functionality. Addressing these complexities requires a comprehensive approach that includes advanced low power design methodologies, robust verification processes, and careful consideration of integration while partitioning design into multiple power domains. To address this, new power reduction strategies such as power gating, biasing, and multi-voltage supply architectures have been introduced. These techniques demand sophisticated power management frameworks that not only support their implementation but also incorporate both hardware and software control logic. This control infrastructure is vital for initiating and managing transitions between various power states, ensuring efficient and reliable operation across the system. SOC Partitioning into various hard macro subsystems leads to partitioning complexity and integration, w.r.t low power design and verification.

In SoC RTL, hard macros are represented by equivalent behavioral models written in HDL languages like Verilog, SystemVerilog, etc. They are developed with accuracy and precision to mimic the actual designs which are either semi-custom or full custom designs. Power functionality modeling in these behavioral models is therefore crucial for accurately capturing, simulating and verifying the power management features of that hard macro model integrated in SOC at the RTL stage. Real Number Voltage Aware Modeling offers a compelling solution to this issue by enabling more efficient and accurate simulation of power functionality behavior which are analog in type within digital environments, thereby reducing verification time and improving overall design productivity. This paper presents a method for modeling the power behavior of multi-voltage hard macro-Ips/Subsystems using real-number voltage variables**.** It showcases the techniques to accurately capture the behavior of these hard macros, which are critical to achieving low-power functionality in SoC designs. A case study using Retention SRAM (RETmem) and Bias Generator (BIASG) blocks, where correct voltage values and sequencing is critical for low-power retention mode functionality.

II. CHALLENGES WITH PRESENT FLOW OF VERIFICATION

Low-power SoCs with mixed digital-analog designs face major verification hurdles due to limitations in simulation methods. Both digital and analog simulation are not adequate for functional signoff of the SoC. Below mentioned are the challenges in current verification flow.

A. High simulation time : SPICE Simulations [5]

Around 10-30 % of SoC is comprised of analog and mixed signal (AMS) blocks and it is getting enhanced due to evolution of AI/ML acceleration requirements (NPUs, IMC), Automotive, IoT and networking etc. These blocks are though limited but perform tasks which are not just digital computations but real complex functions. SPICE simulations are preferred choices such circuits. However, simulation time is one of the key compared to digital simulations despite being highly accurate. Typically, SPICE simulations [5] on any SoC will take from 3 hours to 1 week depending on the complexity and composition of analog block and testcases to be verified in accordance with it. This slows down verification and increases cost instead replacing analog behavior with HDL models can enable faster digital simulations.

B. Low Accuracy : Digial Simulation for Analog ciruits

To speed up the SoC verification, event based digital simulations are being widely used to ensure the time to market products in this fiercely competitive industry. In digital simulations, approximate models [4] for analog behavior, lacking real electrical value representation where Voltage-sensitive behaviors are often oversimplified, risking accuracy and confidence in verification.

C. Verification Coverage of Power functions

For verification of Low power SoC design components and behaviors, one must choose the simulation types smartly to ensure the coverage requirements. But the measurement of the verification coverage [2] is difficult to come by when put together for different types of simulations i.e digital and analog. Digital techniques (e.g., SV-UVM) offer strong coverage tools but don't apply to analog simulations. So, it is imperative to bring more and more analog circuit behavior in the form of digital models which then can leverage out the benefits of digital verification techniques.

III. Voltage Aware Models

To overcome the limitations of time-intensive SPICE simulations [5], behavioral models are used to accelerate digital design flows. Traditional Power-Aware (PA) [4] models, which treat supply signals as digital inputs/outputs, often lack the accuracy needed for complex analog and low-power designs. Voltage-aware models address this gap described in section II by incorporating voltage-dependent behaviors and their corresponding checkers, enabling accurate simulation of features like retention, isolation, voltage scaling, and power state transitions in modern low-power SoC architectures. This is especially critical for multi-voltage rail macros, where precise tracking of voltage changes ensures early detection of design issues in modern low-power SoCs. This enables outputs that closely mirrors real design behavior, helping designers detect and resolve issues early in the development cycle.

The Voltage-Aware Modeling (VA) methodology is built on three key components:

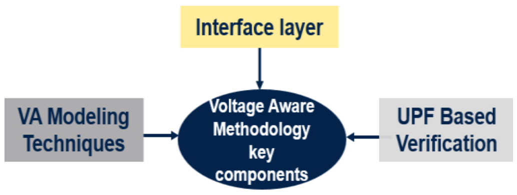


Figure 1. VA methodology key components

A. ***VA model Interface*** layer facilitates the declaration of real-number supply ports—such as input, output, inout, or bias signals—using the "supply_net_type" datatype, in accordance with the IEEE 1801 standard [1]. Carrying both voltage value as well as the power state of the supply, this UDN enables the real number voltage transactions over the supply network defined in the power architecture of the low power SoC or sub-system design in the form of UPF file.

B. ***VA behavioral modeling*** captures power-related functionality by identifying and integrating voltage-dependent behaviors into the model. Accurate modeling requires a thorough analysis of design specifications to extract all supply-related behaviors and constraints. These include operating voltage ranges, power-up/down sequences, ramp behaviors, mode transitions in multi-voltage systems, and dynamic voltage scaling. Once

captured these specifications guide the development of SystemVerilog-based components/blocks. The VA behavioral models can represent standalone Hard Macro IPs or mixed-signal subsystems. To structure the functionality effectively, the model is organized into classified sub-blocks as outlined below:
SupplyChecker : Ensures compliance with supply specifications by monitoring operating voltage ranges and power-up/down sequences based on the detected operating mode. It senses control inputs and supply levels, validating them against defined constraints. If compliance fails, the model outputs undefined logic (X); otherwise, it proceeds with the intended functional behavior of the block will be shown by the model.
Logic Functional Behavior: Represents the core logic of the design, similar to traditional behavioral model. It generates outputs based on design specifications, such as read/write operations in an SRAM or clock generation in an oscillator block.
Supply Functional Behavior: Models analog supply-related functions such as bias signal generation, voltage regulation, and voltage scaling. It captures complex analog behaviors essential to power management in the design.

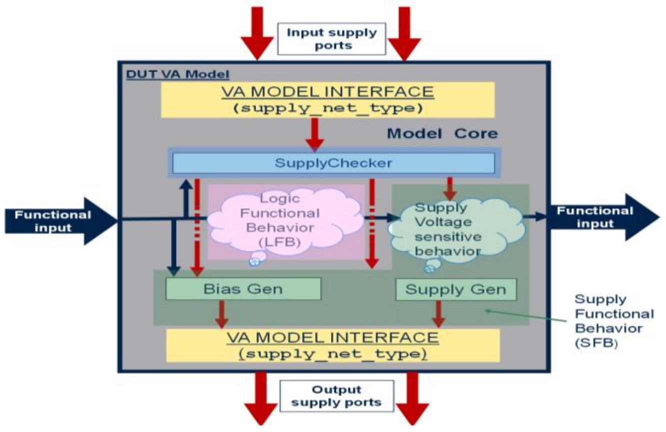

Input supply ports
DUT VA Model
VA MODEL INTERFACE (supply_net_type)
Model Core
SupplyChecker
Functional input
Logic Functional Behavior (LFB)
Supply Voltage sensitive behavior
Functional input
Bias Gen
Supply Gen
Supply Functional Behavior (SFB)
VA MODEL INTERFACE (supply_net_type)
Output supply ports

Figure 2. Typical VA Model Architecture

Figure 2 illustrates a sample voltage-aware model architecture for a design block, composed of the previously described components. These include the SupplyChecker, Logic Functional Behavior (LFB), and Supply Functional Behavior (SFB), interconnected through functional and supply signals. The VA model interface, as discussed earlier, supports real-number voltage transactions at power supply ports. The blocks are interdependent—valid outputs are produced only when input stimuli meet all modeled specification checkpoints.
This approach significantly improves verification accuracy and efficiency by enabling VA models to precisely capture analog supply behaviors. Integrated checkers reject unsupported supply values and sequences, allowing faster and more reliable power verification within digital simulations.

C. ***VA verification [3]*** is performed using digital simulators that support the IEEE 1801 UPF [1] standard, along with a Unified Power Format (.upf) file and Liberty files containing pg_pin definitions and low-power attributes. Combined with the VA behavioral model and a testbench that drives real voltage values, this setup enables accurate low-power verification by monitoring supply behavior across voltage levels as defined in the design specifications.

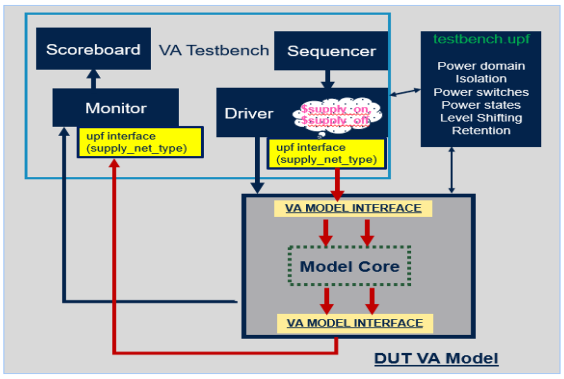
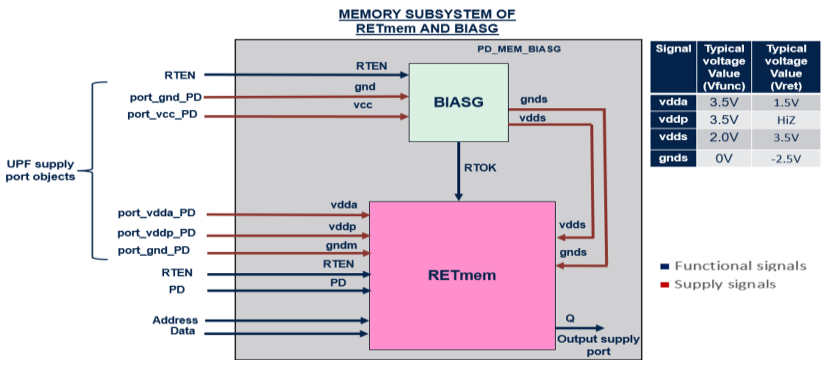


Figure 3. Voltage Aware low-power UPF verification architecture

Figure 3 illustrates the VA verification environment wrapping the VA DUT model. Real voltage stimuli are applied using SystemVerilog UPF functions $supply_on and $supply_off [3], enabling observation of supply-sensitive behavior within a purely digital simulation framework.

IV. MULTIVOLTAGE MEMORY SUBSYSTEM



Signal	Typical voltage Value (Vfunc)	Typical voltage Value (Vret)
vdda	3.5V	1.5V
vddp	3.5V	HiZ
vdds	2.0V	3.5V
gnds	0V	-2.5V

Figure 4. Memory Subsystem of RETmem and BIASG block with typical supply voltage values in-figure table.

Figure 4 shows the supply and signal connectivity for the RETmem and BIASG blocks. The corresponding operating voltage ranges are detailed in Table I. In Functional mode, RETmem operates within the Vfunc voltage range, providing data output through port Q. In Retention mode, it maintains stored data using a lower Vret voltage range. The required mode transition sequence, shown in Figure 5, is a critical design requirement and must be verified for functional sign-off.

TABLE I. OPERATING VOLTAGE OF RETMEM IN DIFFERENT MODES

Mode	Voltage Supply Set	vdda(V)	vddp(V)	vdds(V)	gnds(V)
SRAM functional	V_{func}	3.0x to 4.0x	3.0x to 4.0x	3.0x to 4.0x	Ground
SRAM Retention	V_{ret}	1.0x to 2.0x	Unconnected (HiZ)	1.50x to 2.50x	-2.0x to -3.0x

Here “x” in the Table I is a normalization factor. We have considered this to be “1” further in this paper.

Functional mode inputs: {PD, RTEN, RTOK} = (0,0,0) & (vdda, vddp, vdds, gnds) = (3.5V, 3.5V, 2.0V, 0V)
Retention mode inputs: {PD, RTEN, RTOK} = (1,1,1) & (vdda, vddp, vdds, gnds) = (1.5V, HiZ, 3.5V, -2.5V)

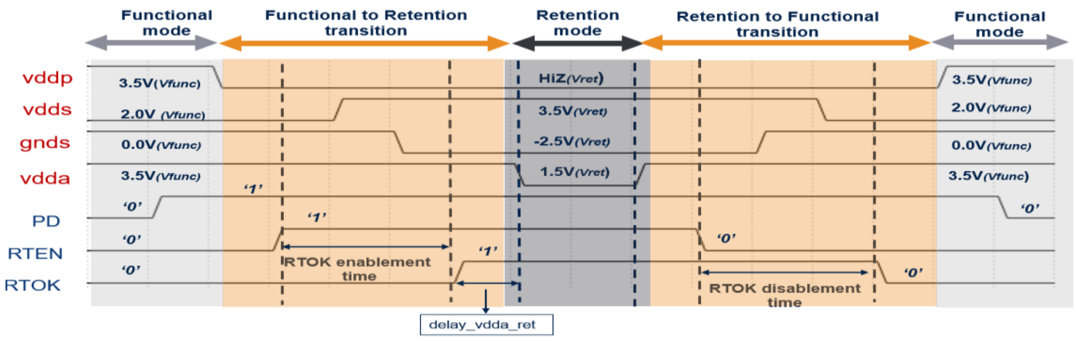


Figure 5. Sequence of operation while entering/exiting from retention mode

Functional to Retention mode transition Sequence:

- Assert PD → Disconnect vddp → Assert the RTEN signal to logic high → Wait for RTOK assertion from BIASG block for RTOK enablement time → vdda voltage level changed to 1.5V(Vret).

In BIASG block, RTEN signal assertion activates retention bias voltages *Vret* (vdds, gnds) = (3.5V, -2.5V) generation followed by the assertion of RTOK signal after a defined delay of *RTOK enablement time* (ref Figure 5). Finally, with vdda change to 1.5V start the retention mode where memory array contents will be retained.

Retention to Functional mode transition Sequence:

- Voltage of vdda raise to 3.5V → De-assert RTEN signal → Wait for RTOK de-assertion →De-assert PD to logic low → Volatge of vddp raised to 3.5V.

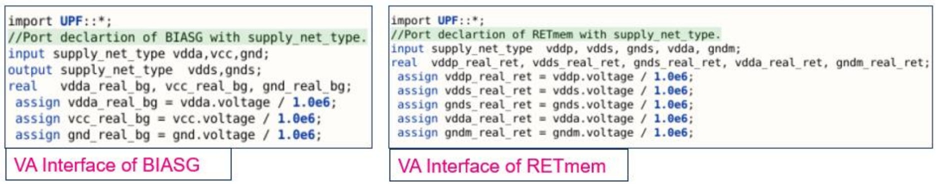
In BIASG block, RTEN signal de-assertion activates functional bias voltages Vfunc (vdds, gnds) = (2.5V, 0V) generation followed by the de-assertion of RTOK signal after a defined delay of *RTOK disablement time* (ref Figure 5). Finally, with vddp change to 3.5V start the functional mode where memory read/write can be done.

V. VOLTAGE AWARE MOELING IMPLEMENTATION OF RETMEM

The three major components explained in Section III are described with the implementation example of Subsystem of RETmem and BIASG block.

A. VA Interface

The code snippet in Figure 6 illustrates declaration of real voltage supplies at port level in a typical Voltage aware model. This is done using IEEE Std 1801 aligned SV UPF [1] version 3.0 package having supply_net_type UDN. To access voltage value in the model “. voltage" can be accessed for declared supply_net_type. It provides voltage value in micro volts which is required to further conversion into Volts by diving it with 1.0e6.



```
import UPF::*;
//Port declartion of BIASG with supply_net_type.
input supply_net_type vdda,vcc,gnd;
output supply_net_type  vdds,gnds;
real   vdda_real_bg, vcc_real_bg, gnd_real_bg;
 assign vdda_real_bg = vdda.voltage / 1.0e6;
 assign vcc_real_bg = vcc.voltage / 1.0e6;
 assign gnd_real_bg = gnd.voltage / 1.0e6;
```

VA Interface of BIASG

```
import UPF::*;
//Port declartion of RETmem with supply_net_type.
input supply_net_type  vddp, vdds, gnds, vdda, gndm;
real  vddp_real_ret, vdds_real_ret, gnds_real_ret, vdda_real_ret, gndm_real_ret;
 assign vddp_real_ret = vddp.voltage / 1.0e6;
 assign vdds_real_ret = vdds.voltage / 1.0e6;
 assign gnds_real_ret = gnds.voltage / 1.0e6;
 assign vdda_real_ret = vdda.voltage / 1.0e6;
 assign gndm_real_ret = gndm.voltage / 1.0e6;
```

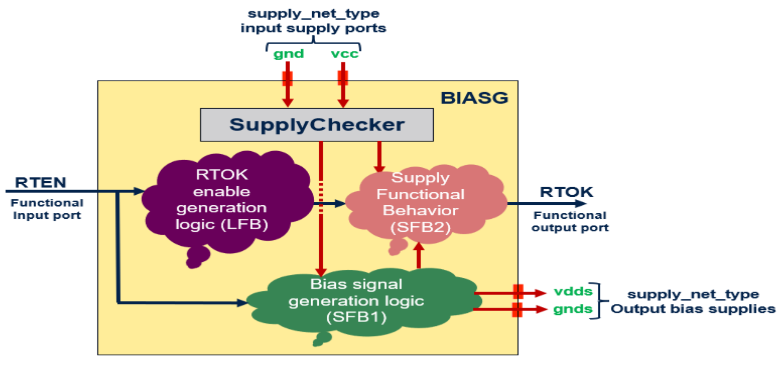
VA Interface of RETmem

Figure 6. VA model with supplies declared as supply_net_type for BIASG and RETmem

B. VA Modeling method (checkers and behavior)

The key components of any VA model as described Section III (B) are explained below for both RETmem and BIASG block taken as the implementation case of this paper.

Starting with BIASG block, Figure 7 shows the model architecture depicting both functional as well as supply behavior.



🟩 Real number ports 🟦 Digital Logic ports

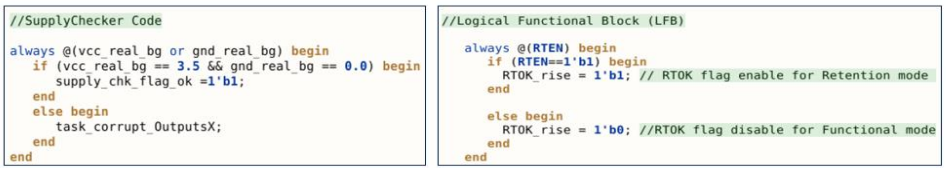
Figure 7. VA model representation of BIASG block

a. SupplyChecker : Validates the input voltage levels of vcc and gnd received via the VA interface. For correct operation, it checks that vcc = 3.5 V and gnd = 0.0 V. The SupplyChecker ensures these conditions are met to enable correct bias generation and mode operation.

b. Logic Functional Behavior (LFB): It depicts the RTOK signal flag initiation based on RTEN input signal to eventually generate RTOK signal through SFB2 as mentioned in point #d.

c. Supply Functional Behavior (SFB1): This block generates the bias signals after specified delay from RTEN signal assertion when SupplyChecker compliance is validated. Furthermore, these signals are real number voltage values utilizing supply_net_type UPF UDN.

d. Supply Functional Behavior (SFB2): This block generates the Functional output signal RTOK from the initiated flag signal of LFB block with a settling delay as per Figure 5 . With dependence of SupplyChecker compliance and Bias Generation logic of SFB1, RTOK signal generation is executed.

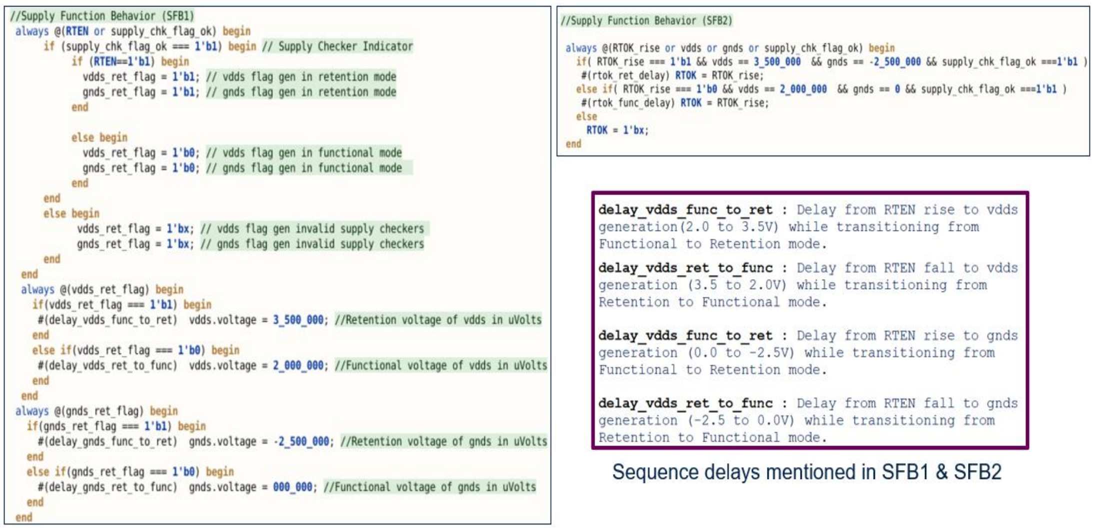


```
//SupplyChecker Code

always @(vcc_real_bg or gnd_real_bg) begin
   if (vcc_real_bg == 3.5 && gnd_real_bg == 0.0) begin
      supply_chk_flag_ok =1'b1;
   end
   else begin
      task_corrupt_OutputsX;
   end
end
```

```
//Logical Functional Block (LFB)

   always @(RTEN) begin
      if (RTEN==1'b1) begin
        RTOK_rise = 1'b1; // RTOK flag enable for Retention mode
      end

      else begin
        RTOK_rise = 1'b0; //RTOK flag disable for Functional mode
      end
   end
```

Figure 8. BIASG VA Model with SupplyChecker code and LFB code

SFB1 code snippet in Figure 9 is the bias generation logic of vdds and gnds and various sequencing delays as per Figure 5.



```
//Supply Function Behavior (SFB1)
 always @(RTEN or supply_chk_flag_ok) begin
     if (supply_chk_flag_ok === 1'b1) begin // Supply Checker Indicator
         if (RTEN==1'b1) begin
           vdds_ret_flag = 1'b1; // vdds flag gen in retention mode
           gnds_ret_flag = 1'b1; // gnds flag gen in retention mode
         end

         else begin
           vdds_ret_flag = 1'b0; // vdds flag gen in functional mode
           gnds_ret_flag = 1'b0; // gnds flag gen in functional mode
         end
     end
     else begin
          vdds_ret_flag = 1'bx; // vdds flag gen invalid supply checkers
          gnds_ret_flag = 1'bx; // gnds flag gen invalid supply checkers
     end
  end
  always @(vdds_ret_flag) begin
    if(vdds_ret_flag === 1'b1) begin
     #(delay_vdds_func_to_ret)  vdds.voltage = 3_500_000; //Retention voltage of vdds in uVolts
    end
    else if(vdds_ret_flag === 1'b0) begin
     #(delay_vdds_ret_to_func)  vdds.voltage = 2_000_000; //Functional voltage of vdds in uVolts
    end
  end
 always @(gnds_ret_flag) begin
   if(gnds_ret_flag === 1'b1) begin
     #(delay_gnds_func_to_ret)  gnds.voltage = -2_500_000; //Retention voltage of gnds in uVolts
   end
   else if(gnds_ret_flag === 1'b0) begin
     #(delay_gnds_ret_to_func)  gnds.voltage = 000_000; //Functional voltage of gnds in uVolts
   end
 end
```

```
//Supply Function Behavior (SFB2)

 always @(RTOK_rise or vdds or gnds or supply_chk_flag_ok) begin
   if( RTOK_rise === 1'b1 && vdds == 3_500_000  && gnds == -2_500_000 && supply_chk_flag_ok ===1'b1 )
    #(rtok_ret_delay) RTOK = RTOK_rise;
   else if( RTOK_rise === 1'b0 && vdds == 2_000_000  && gnds == 0 && supply_chk_flag_ok ===1'b1 )
    #(rtok_func_delay) RTOK = RTOK_rise;
   else
     RTOK = 1'bx;
 end
```

delay_vdds_func_to_ret : Delay from RTEN rise to vdds generation(2.0 to 3.5V) while transitioning from Functional to Retention mode.

delay_vdds_ret_to_func : Delay from RTEN fall to vdds generation (3.5 to 2.0V) while transitioning from Retention to Functional mode.

delay_vdds_func_to_ret : Delay from RTEN rise to gnds generation (0.0 to -2.5V) while transitioning from Functional to Retention mode.

delay_vdds_ret_to_func : Delay from RTEN fall to gnds generation (-2.5 to 0.0V) while transitioning from Retention to Functional mode.

Sequence delays mentioned in SFB1 & SFB2

Figure 9. Code snippet of BIASG Block VA components: SFB1 and SFB2

VA Model architecture of RETmem and its components (Figure 10) are explained below.

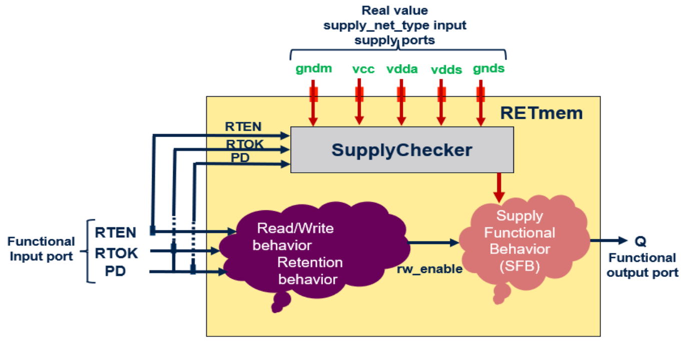
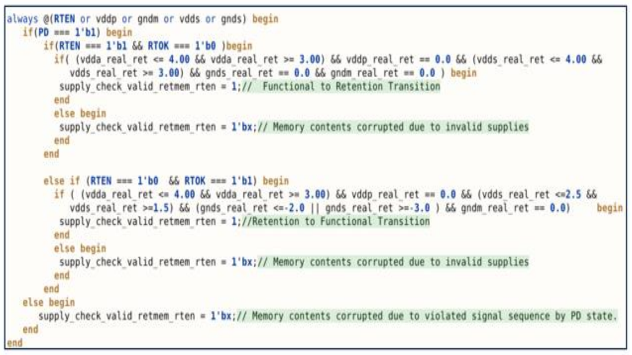


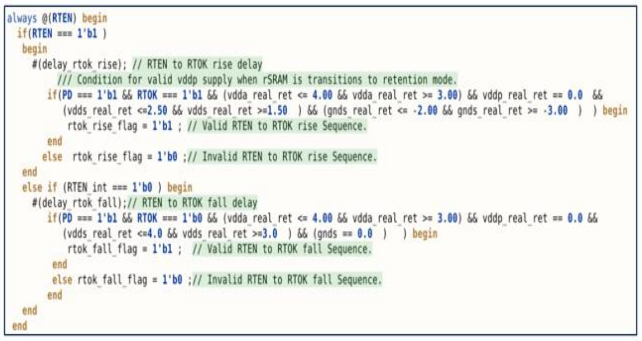
Figure 10 . VA model representation of RETmem block

a. SupplyChecker : Validates input voltage levels and supply transition sequences based on RETmem's operating modes. Key checks include mode transitions and corresponding voltage values. Figures 11.a–d show code snippets verifying RTEN, RTOK, and compliance with the transition sequence and voltage ranges from Figure 5. Checker-generated flags—such as *supply_check_valid_retmem_rten*, *Memory_retention_flag*, and *rtok_rise_flag*—influence the outputs of LFB and SFB blocks described below.



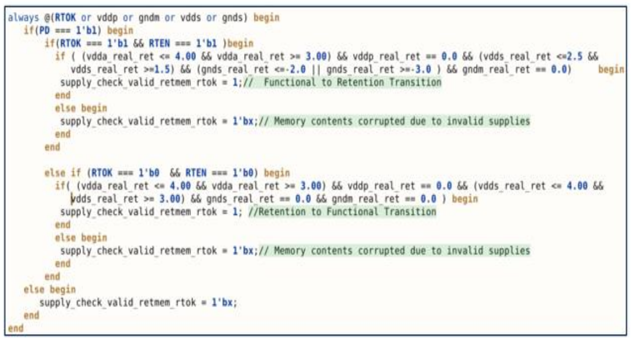
```
always @(RTEN or vddp or gndm or vdds or gnds) begin
   if(PD === 1'b1) begin
       if(RTEN === 1'b1 && RTOK === 1'b0 )begin
         if( (vdda_real_ret <= 4.00 && vdda_real_ret >= 3.00) && vddp_real_ret == 0.0 && (vdds_real_ret <= 4.00 &&
            vdds_real_ret >= 3.00) && gnds_real_ret == 0.0 && gndm_real_ret == 0.0 ) begin
          supply_check_valid_retmem_rten = 1;//  Functional to Retention Transition
         end
         else begin
          supply_check_valid_retmem_rten = 1'bx;// Memory contents corrupted due to invalid supplies
         end
       end

       else if (RTEN === 1'b0  && RTOK === 1'b1) begin
         if ( (vdda_real_ret <= 4.00 && vdda_real_ret >= 3.00) && vddp_real_ret == 0.0 && (vdds_real_ret <=2.5 &&
            vdds_real_ret >=1.5) && (gnds_real_ret <=-2.0 || gnds_real_ret >=-3.0 ) && gndm_real_ret == 0.0)     begin
          supply_check_valid_retmem_rten = 1;//Retention to Functional Transition
         end
         else begin
          supply_check_valid_retmem_rten = 1'bx;// Memory contents corrupted due to invalid supplies
         end
       end
   else begin
      supply_check_valid_retmem_rten = 1'bx;// Memory contents corrupted due to violated signal sequence by PD state.
   end
end
```

Figure 11.a . RETmem VA model SupplyCheckers for RTOK & RTEN Transition Checkpoints monitoring all RETmem supplies



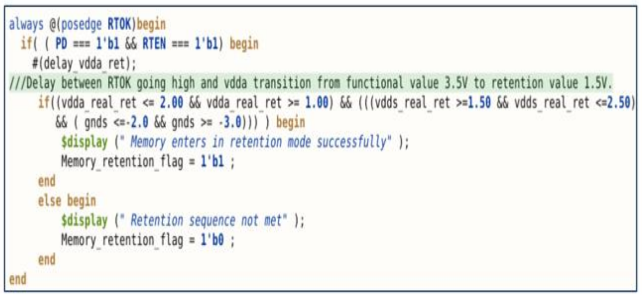
```
always @(RTEN) begin
  if(RTEN === 1'b1 )
   begin
     #(delay_rtok_rise); // RTEN to RTOK rise delay
         /// Condition for valid vddp supply when rSRAM is transitions to retention mode.
       if(PD === 1'b1 && RTOK === 1'b1 && (vdda_real_ret <= 4.00 && vdda_real_ret >= 3.00) && vddp_real_ret == 0.0  &&
          (vdds_real_ret <=2.50 && vdds_real_ret >=1.50  ) && (gnds_real_ret <= -2.00 && gnds_real_ret >= -3.00  )  ) begin
           rtok_rise_flag = 1'b1 ; // Valid RTEN to RTOK rise Sequence.
       end
      else  rtok_rise_flag = 1'b0 ;// Invalid RTEN to RTOK rise Sequence.
   end
   else if (RTEN_int === 1'b0 ) begin
     #(delay_rtok_fall);// RTEN to RTOK fall delay
       if(PD === 1'b1 && RTOK === 1'b0 && (vdda_real_ret <= 4.00 && vdda_real_ret >= 3.00) && vddp_real_ret == 0.0 &&
          (vdds_real_ret <=4.0 && vdds_real_ret >=3.0  ) && (gnds == 0.0  )   ) begin
           rtok_fall_flag = 1'b1 ;  // Valid RTEN to RTOK fall Sequence.
        end
        else rtok_fall_flag = 1'b0 ;// Invalid RTEN to RTOK fall Sequence.
       end
   end
 end
```

Figure 11.c . RETmem VA models Sequence checkers for RTEN to RTOK Transition monitoring all RETmem supplies



```
always @(RTOK or vddp or gndm or vdds or gnds) begin
   if(PD === 1'b1) begin
       if(RTOK === 1'b1 && RTEN === 1'b1 )begin
         if ( (vdda_real_ret <= 4.00 && vdda_real_ret >= 3.00) && vddp_real_ret == 0.0 && (vdds_real_ret <=2.5 &&
            vdds_real_ret >=1.5) && (gnds_real_ret <=-2.0 || gnds_real_ret >=-3.0 ) && gndm_real_ret == 0.0)     begin
          supply_check_valid_retmem_rtok = 1;//  Functional to Retention Transition
         end
         else begin
          supply_check_valid_retmem_rtok = 1'bx;// Memory contents corrupted due to invalid supplies
         end
       end

       else if (RTOK === 1'b0  && RTEN === 1'b0) begin
         if( (vdda_real_ret <= 4.00 && vdda_real_ret >= 3.00) && vddp_real_ret == 0.0 && (vdds_real_ret <= 4.00 &&
            vdds_real_ret >= 3.00) && gnds_real_ret == 0.0 && gndm_real_ret == 0.0 ) begin
          supply_check_valid_retmem_rtok = 1; //Retention to Functional Transition
         end
         else begin
          supply_check_valid_retmem_rtok = 1'bx;// Memory contents corrupted due to invalid supplies
         end
       end
   else begin
      supply_check_valid_retmem_rtok = 1'bx;
   end
end
```

Figure 11.b . RETmem VA model SupplyCheckers for RTOK & RTEN Transition Checkpoints monitoring all RETmem supplies



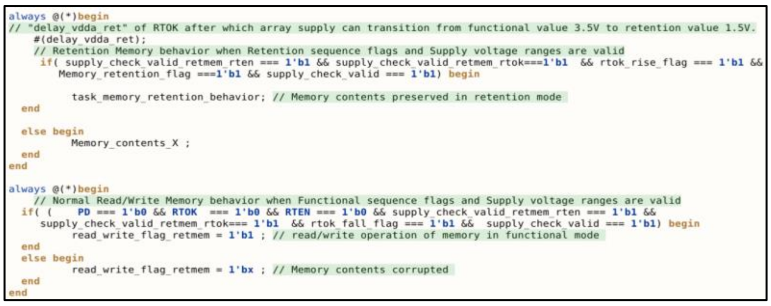
```
always @(posedge RTOK)begin
  if( ( PD === 1'b1 && RTEN === 1'b1) begin
    #(delay_vdda_ret);
///Delay between RTOK going high and vdda transition from functional value 3.5V to retention value 1.5V.
     if((vdda_real_ret <= 2.00 && vdda_real_ret >= 1.00) && (((vdds_real_ret >=1.50 && vdds_real_ret <=2.50)
        && ( gnds <=-2.0 && gnds >= -3.0))) ) begin
         $display (" Memory enters in retention mode successfully" );
         Memory_retention_flag = 1'b1 ;
     end
     else begin
         $display (" Retention sequence not met" );
         Memory_retention_flag = 1'b0 ;
     end
end
```

Figure 11.d . RETmem VA models Sequence checkers for RTOK to vdda Transition monitoring all RETmem supplies

Figure 11 . RETmem VA model SupplyCheckers for RETmem supplies

b. Logic Functional Behavior (LFB): Models the RETmem’s core logic—read/write operations in Functional mode and memory array data retention in Retention mode. Behavior is driven by RETmem inputs and the supply voltage transition sequence, as validated by the SupplyChecker (see Figure 5).

c. Supply Functional Behavior (SFB): Generates final output behavior by combining results from the SupplyChecker and LFB. If supply compliance fails, the RETmem output Q is set to X, and the memory array (modeled as a 2D array in SystemVerilog) is corrupted using the *read_write_flag_retmem* and *Memory_contents_X* task. If compliance passes, SFB produces outputs based on the current RETmem mode—Q in Functional mode or retained data in Retention mode, which can be read in subsequent Functional mode.
Both LFB and SFB behavior are shown in Figure 12, where they are intermixed for simplified implementation in sv coding.



```
always @(*)begin
// "delay_vdda_ret" of RTOK after which array supply can transition from functional value 3.5V to retention value 1.5V.
    #(delay_vdda_ret);
    // Retention Memory behavior when Retention sequence flags and Supply voltage ranges are valid
     if( supply_check_valid_retmem_rten === 1'b1 && supply_check_valid_retmem_rtok===1'b1  && rtok_rise_flag === 1'b1 &&
        Memory_retention_flag ===1'b1 && supply_check_valid === 1'b1) begin

          task_memory_retention_behavior; // Memory contents preserved in retention mode
  end

  else begin
          Memory_contents_X ;
  end
end

always @(*)begin
    // Normal Read/Write Memory behavior when Functional sequence flags and Supply voltage ranges are valid
  if( (    PD === 1'b0 && RTOK  === 1'b0 && RTEN === 1'b0 && supply_check_valid_retmem_rten === 1'b1 &&
     supply_check_valid_retmem_rtok=== 1'b1  && rtok_fall_flag === 1'b1 &&  supply_check_valid === 1'b1) begin
          read_write_flag_retmem = 1'b1 ; // read/write operation of memory in functional mode
  end
  else begin
          read_write_flag_retmem = 1'bx ; // Memory contents corrupted
  end
end
```

Figure 12 . RETmem VA model Combined LFB and SFB

The described checkers and behavior blocks enable verification of real voltage values and handshaking sequences within a digital simulation environment—capabilities traditionally limited to SPICE simulations.

C. *Voltage Aware Verification and Simulation Results*

Voltage-aware verification of the RETmem-BIASG subsystem is performed using the SystemVerilog-based verification environment shown in Figure 3. The setup includes the DUT VA model, testbench components, the DUT's Liberty file, and a UPF file defining power domains, components, and networks using IEEE 1801 [1] UPF objects. Test sequences cover Functional mode, Retention mode, and mode transitions, driven by the testbench. Real voltage values are applied using the $supply_on and $supply_off functions from the UPF package. Simulations are executed on a UPF-compliant low-power simulator, which utilizes pg_pin definitions from the Liberty file, power intent from the UPF file, and stimuli from the testbench.

a. First testcase is shown below in Figure 13 is depicting the primary use case of the RETmem design involving valid mode transition as per Figure 5.

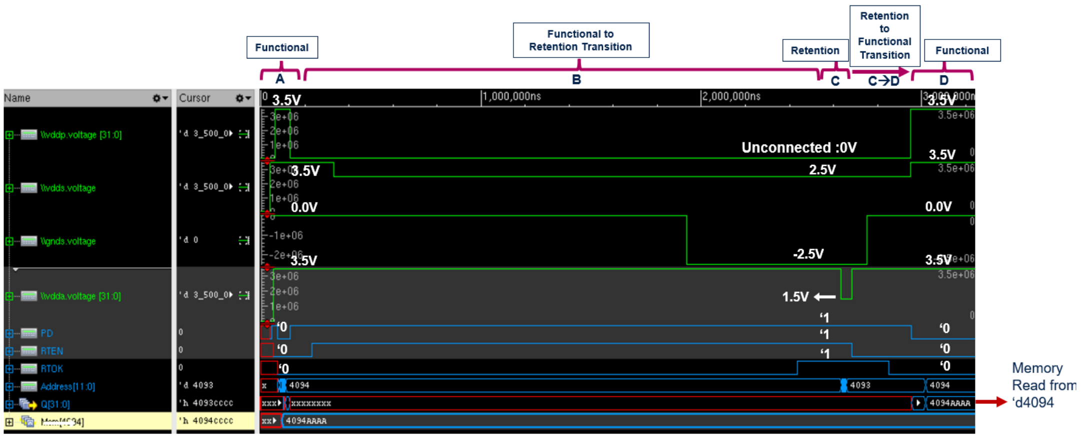


Figure 13. VA simulation with Functional $\rightarrow$ Retention $\rightarrow$ Functional mode transition at valid real supply values

Figure 13 shows the simulation waveform illustrating the RETmem mode transition from Functional to Retention and back. In Region A, vdda = 3.5 V, indicating Functional mode. Region B captures the transition to Retention mode, where bias generation occurs and the RTOK signal is asserted. In Region C, vdda drops to 1.5 V, marking Retention mode—during which no read/write activity occurs.
Following Region C, the waveform demonstrates the transition from Retention back to Functional mode. During this phase, bias voltages are regenerated and vdda is restored to 3.5 V, enabling a read operation. This validates data retention by reading the same address (d4094) before and after Retention mode and confirming the preserved value (h4094AAAA).

b. Second test is a negative test case (Figure 14) where stimuli is same as the first testcase except the vdda voltage value in case of retention mode. It is driven as 0.6V which is beyond spec of Retention mode and design behavior is not guaranteed in such case.

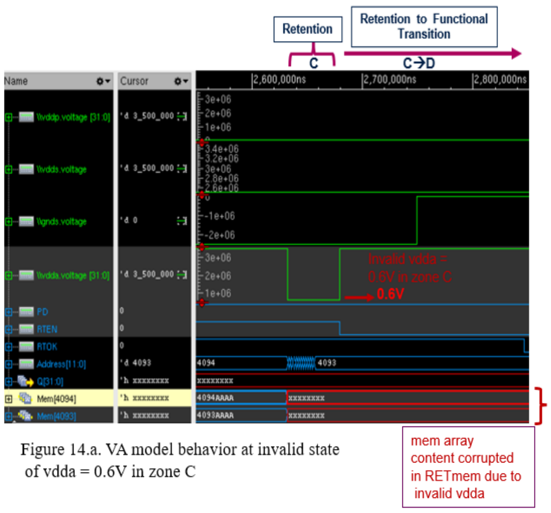


Figure 14.a. VA model behavior at invalid state of vdda = 0.6V in zone C

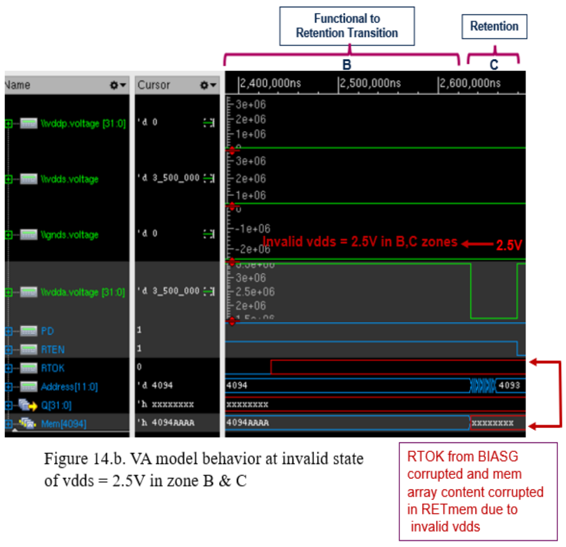


Figure 14.b. VA model behavior at invalid state of vdds = 2.5V in zone B & C

Figure 14. VA model behavior at invalid state of vdda = 0.6V in zone C & C->D and vdds = 0.5V in zone B &C

In this case, the VA model SupplyChecker is violated due an invalid vdda = 0.60 V applied during transition to Retention mode shown in Figure 14.a This violates the specification leading to memory corruption behavior in Mem[4094] location which was supposed to be retained, eventually a failed retention behavior is depicted.

c. Third test (Figure 14.b) is again a negative test case where stimuli is same as first test case except the vdds bias voltage value in case of retention mode. It is driven as 2.5V which is beyond spec of Retention mode and design behavior is not guaranteed in such cases. Here VA model output are corrupted (X) and valid behavior is not achieved due to SupplyChecker violations.

To summarize, the VA model accurately reflects the intended behavior of the RETmem design when inputs and voltage conditions meet specifications (as shown in Fig. 5). However, if the stimulus violates these conditions especially real voltage levels critical for ultra-low power operation ,the model corrupts outputs (Q and memory array), causing SoC-level test failures and enabling early RTL-stage issue detection.

VI. Advantages And Limitations of VA models

A. Advantages

VA modeling enables fast, accurate verification of complex power strategies by closely tracking supply voltage variations. It supports precise IP modeling for voltage-sensitive features like Retention, Isolation, and Voltage Scaling, improving low-power functional verification. Traditional power-aware models [4] lacked this accuracy, limiting verification scope.

- *Reduced SPICE dependency*: VA models use real-value supply variables, minimizing reliance on SPICE for power-related checks.
- *Faster verification*: Event-based digital simulations replace slower analog solvers. And faster adaptation of specification changes
- *Enhanced coverage*: Digital simulation with UPF allows use of UVM and extends coverage to power functions at block/SoC level.
- *Scalability*: The Voltage-Aware (VA) modeling and verification methodology is highly scalable across design hierarchies, from IP level to subsystem and ultimately to SoC. At the IP level. It applies to complex Analog/Mixed signal Hard Macro behavioral models such as Clock generator blocks (PLLs), Voltage regulators (LDOs) that is a supply source, bias generator blocks apart from SRAMs shown in this work . Moreover, this approach can be extended to sub-systems developed by clustering or grouping multiple IP blocks and have a need voltage dependent power behavior depiction at SoC to power functional verification of SoC. Furthermore, at the SoC level, where designs are partitioned into multiple power

domains and voltage islands, VA models enable domain-level power intent verification using UPF and facilitate cross-domain voltage sequencing checks, ensuring robust low-power functionality across the entire chip.

- *Performance:* VA models greatly enhance the performance of the IP, sub-system and SoC level verifications by reducing the dependency of the spice/mix-simulations to verify complex power functionality. The sequence of operation shown in Fig.5 when simulated in all spice environment took around 2-3 hours to simulate which typically simulated in couple of seconds (10-12 sec) in digital simulation environment using VA models. From this comparison, it can be extrapolated that the overall simulation time savings achieved through VA modeling are enormous.

B. *Limitations*

With all the advantages offered by Voltage aware behavioral models for low power designs , it does come with its own set of challenges or limitations.

- Adds complexity due to mixing real-number and digital data types (e.g., std_logic, supply_net_type)
- Cannot fully capture PVT (Process, Voltage, Temperature) variations like SPICE models
- Requires UPF-compliant simulators for voltage-aware simulations.

VII. Conclusion and Future Work

This paper demonstrates a scalable voltage-aware modeling approach using real-number voltage variables in SystemVerilog, aligned with IEEE 1801 UPF [1]. It enables accurate simulation of analog supply behavior in digital environments, reducing SPICE dependency and improving early-stage power verification. The RETmem-BIASG case study validates its effectiveness in capturing low-power functionality.

Future work will focus on extending this methodology to mixed-signal simulations, enabling co-verification of analog and digital domains for enhanced accuracy in complex SoC designs.

References

[1] "IEEE Standard for Design and Verification of Low-Power, Energy-Aware Electronic Systems," in IEEE Std 1801-2018 , vol., no., pp.1-548, 29 March 2019, doi: 10.1109/IEEESTD.2019.8686430

[2] Progyna Khondkar, et al., “Low Power Coverage: The Missing Piece in Dynamic Simulation”, February March, DVCon 2018

[3] Tutorial: Low Power Design, Verification, and Implementation with IEEE 1801 UPF, Presented at DVCon 2013 on February 25, 2013

[4] Mohit Jain, Amit Singh, J.S.S.S. Bharath, Amit Srivastava, and Bharti Jain, ”Power Aware Models: Overcoming barriers in Power Aware Simulation” DVCon 2014

[5] Nikolaos Georgoulopoulos and Alkiviadis Hatzopoulos, “Real number modeling of a flash ADC using SystemVerilog”, Panhellenic Conference on Electronics and Telecommunications (PACET), 2017