2025 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION EUROPE

MUNICH, GERMANY
OCTOBER 14-15, 2025

Real Number Voltage aware behavioral modeling and verification of SRAM subsystem with UPF

Amit Singh , STMicroelectronics Pvt. Ltd. ,India
Isha Sharma , STMicroelectronics Pvt. Ltd. ,India

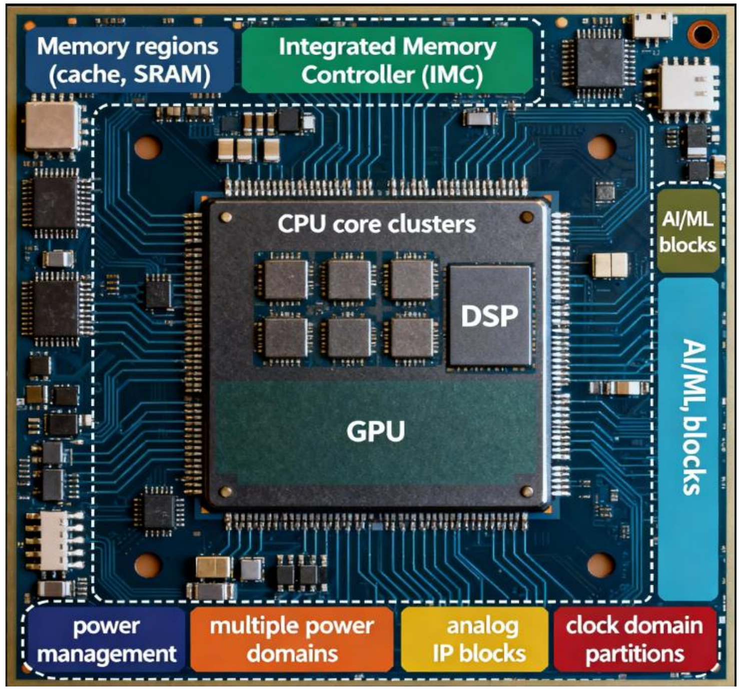
accellera SYSTEMS INITIATIVE

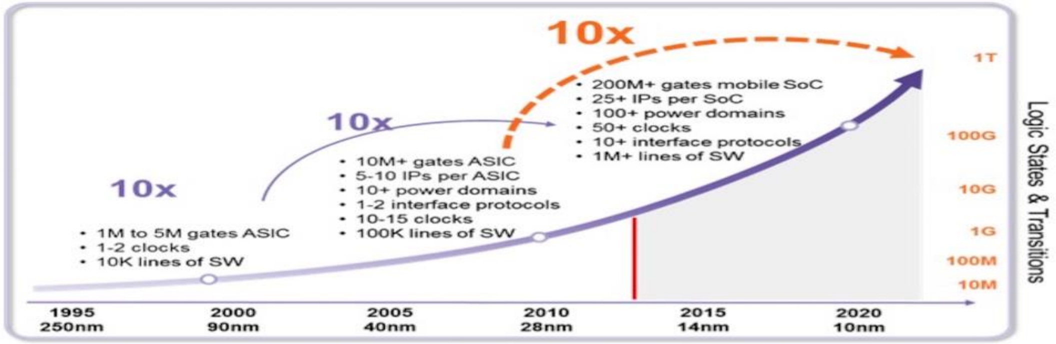
Agenda

- Introduction & Motivation
- Challenges with Current Verification Flow
- Voltage-Aware (VA) Modeling Methodology
- Case Study: RETmem + BIASG Subsystem
- Implementation
- Simulation Results.
- Tradeoffs.
- Future Work.

Introduction and Motivation

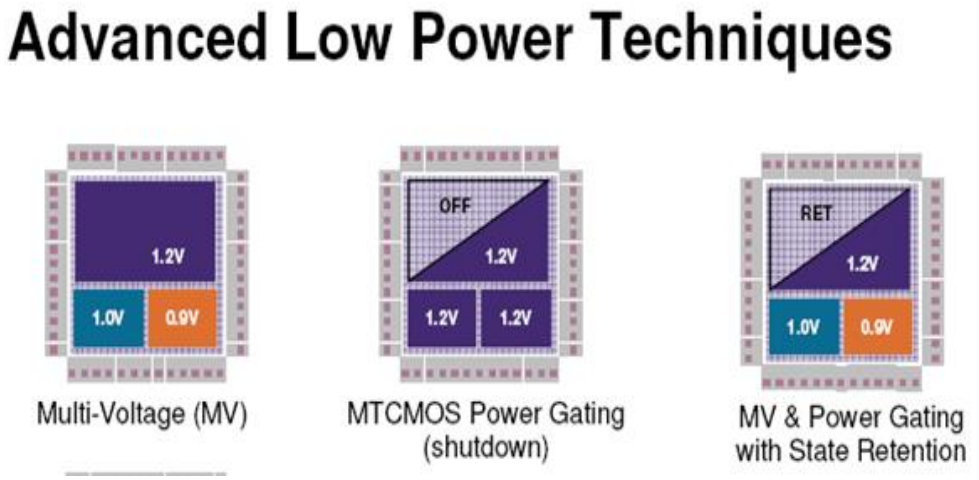
- Modern SoCs :
  - Ocean of transistors integrating CPU ,GPU, IMC,IPs,AMS blocks and diverse other Hard macros.
  - Contains on chip Power and voltage generation sources
- Design complexity demands sophisticated architecture.
- Advanced power management techniques are essential for energy-efficient operation.
- Exhaustive, Early and accurate power functional verification required to ensure reliable functional SoC design.
  - Prevents late cycle errors and debugs.





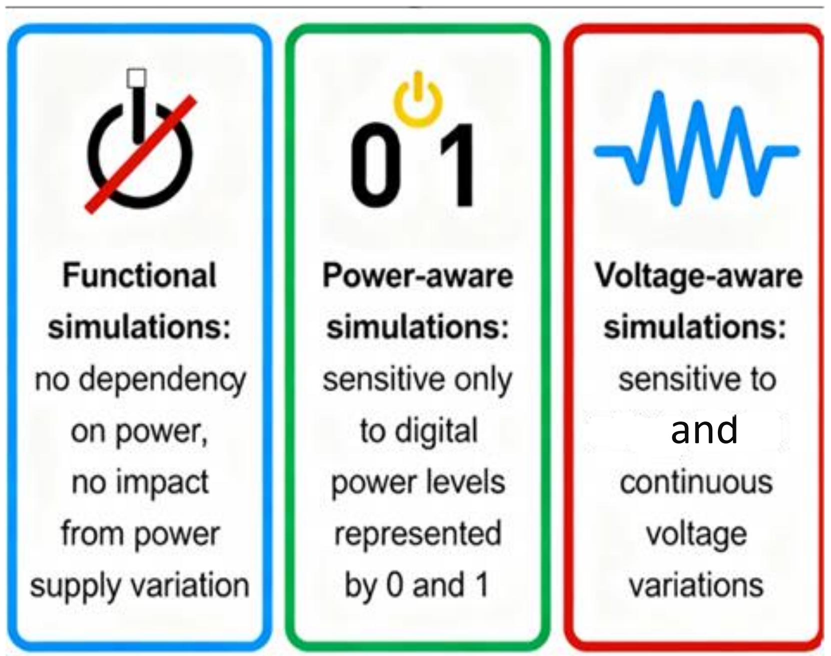
Challenges with Current Verification flow

- Advanced Power management design techniques complexify verification
  - Multi-voltage, Power gating/switching, DVFS, retention, etc
- Hard Marco LP Design Features:
  - Features designed for SoC LP requirements :
    - embedded power switch, retention, multi-volt designs, clock gating, protection circuits for ramp up&down
  - Represented by behavioral models with limited functionality


Advanced Low Power Techniques
1.2V
1.0V
0.9V
Multi-Voltage (MV)
OFF
1.2V
1.2V
1.2V
MTCMOS Power Gating (shutdown)
RET
1.2V
1.0V
0.9V
MV & Power Gating with State Retention

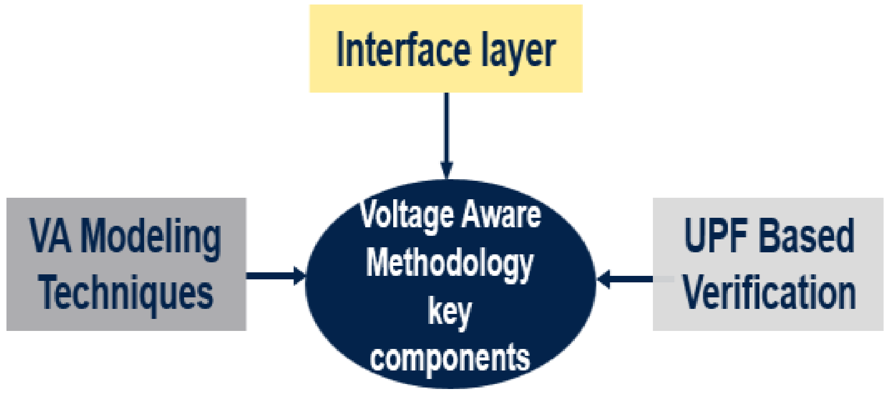
Challenges with Current Verification flow

- Current UPF based Power aware Behavioral Model :
  - Power signals represented by std logic values 0/1
  - No real sense of supply real voltage values .
  - Limitation in modeling electrical behavior.
- Coverage gaps :
  - UVM-style metrics are hard to apply to analog sims;
  - Hard to extract coverage.


Functional simulations: no dependency on power, no impact from power supply variation
0 1
Power-aware simulations: sensitive only to digital power levels represented by 0 and 1
Voltage-aware simulations: sensitive to and continuous voltage variations

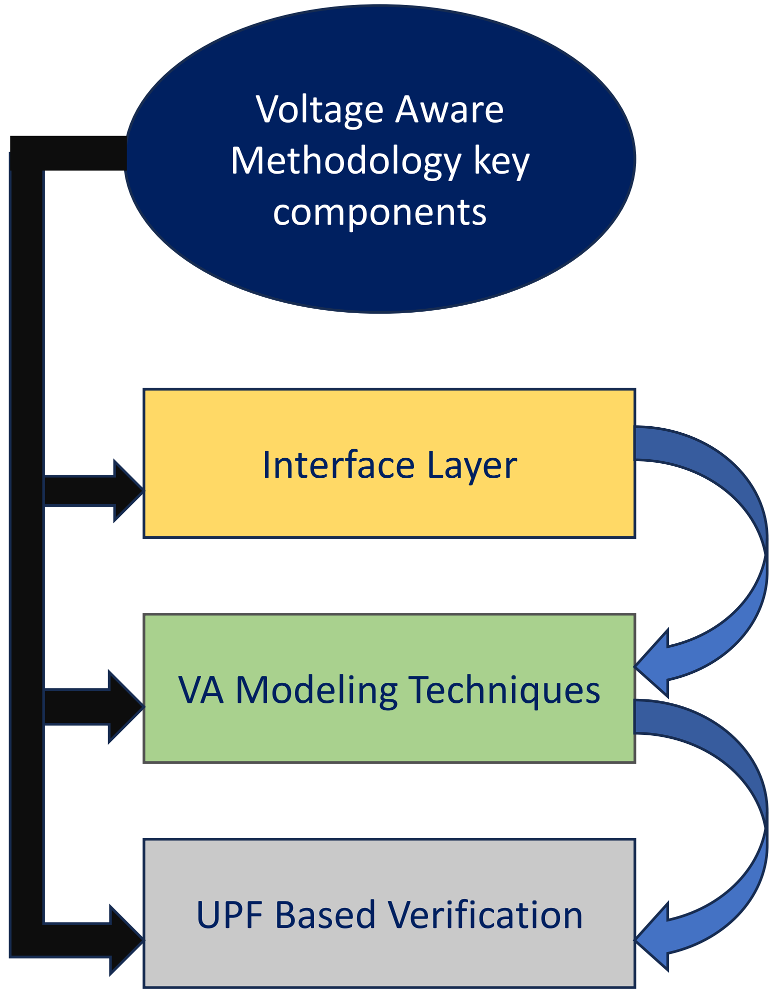
Voltage-Aware (VA) Modeling Methodology

- Accurate real voltage-value based behavioral models in System Verilog.
- UDN from IEEE1801 standard used for interfacing.
  - Supply representation with real-number values via IEEE 1801 "supply_net_type".
- Supply Electrical behavior modeling can be achieved with higher precision
- Built upon 3 components of
  - Interface Layer
  - Modeling behavior
  - UPF based verification



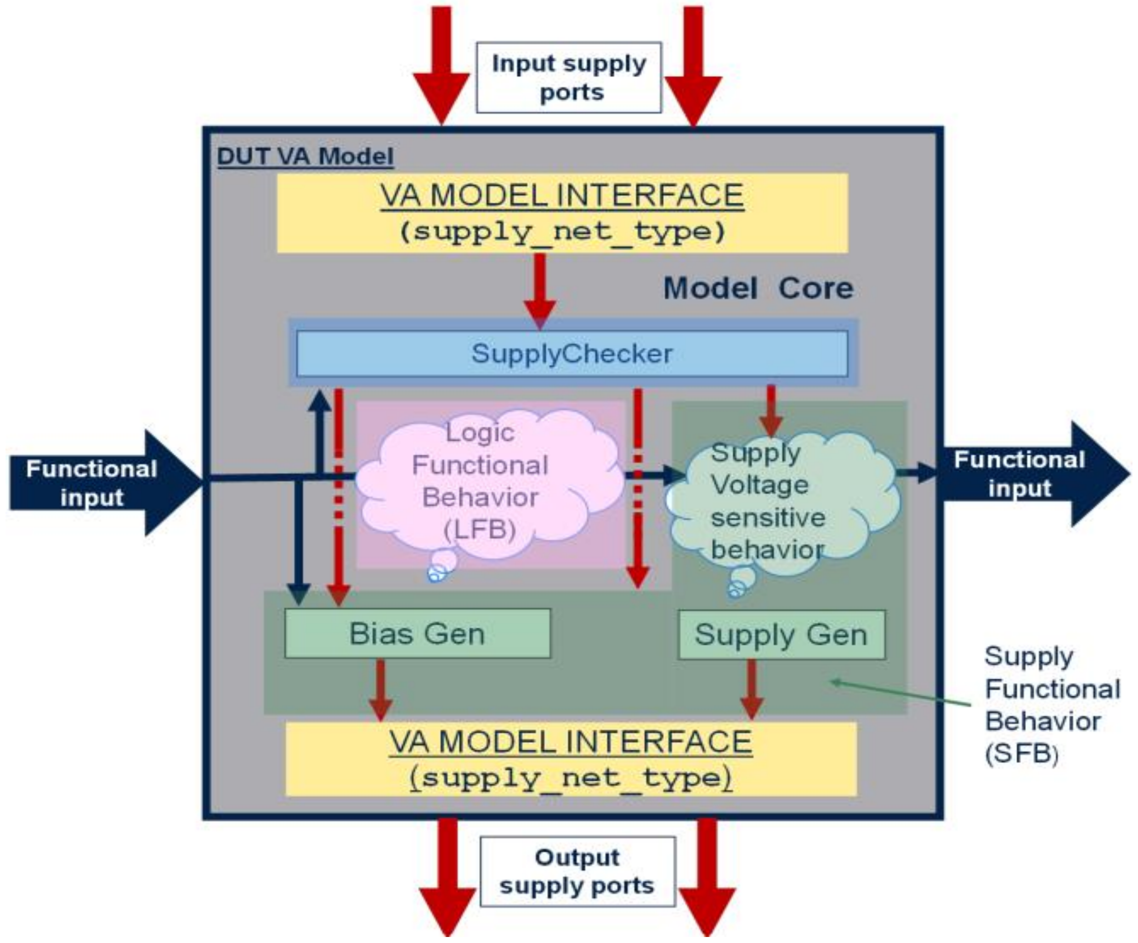
VA Key Components

- Interface Layer-
  - Facilitates the declaration of real-number supply ports using “supply_net_type” datatype.
  - Enables real number voltage transactions.
- VA Modeling Techniques-
  - Voltage dependent behavior exhibited through checkers.
  - Includes operating voltage ranges & supply sequencing.
- UPF Based verification-
  - Run on digital simulators supporting IEEE 1801 Std UPF & liberty files.
  - Testbench drives real voltage values.



VA Model Architecture

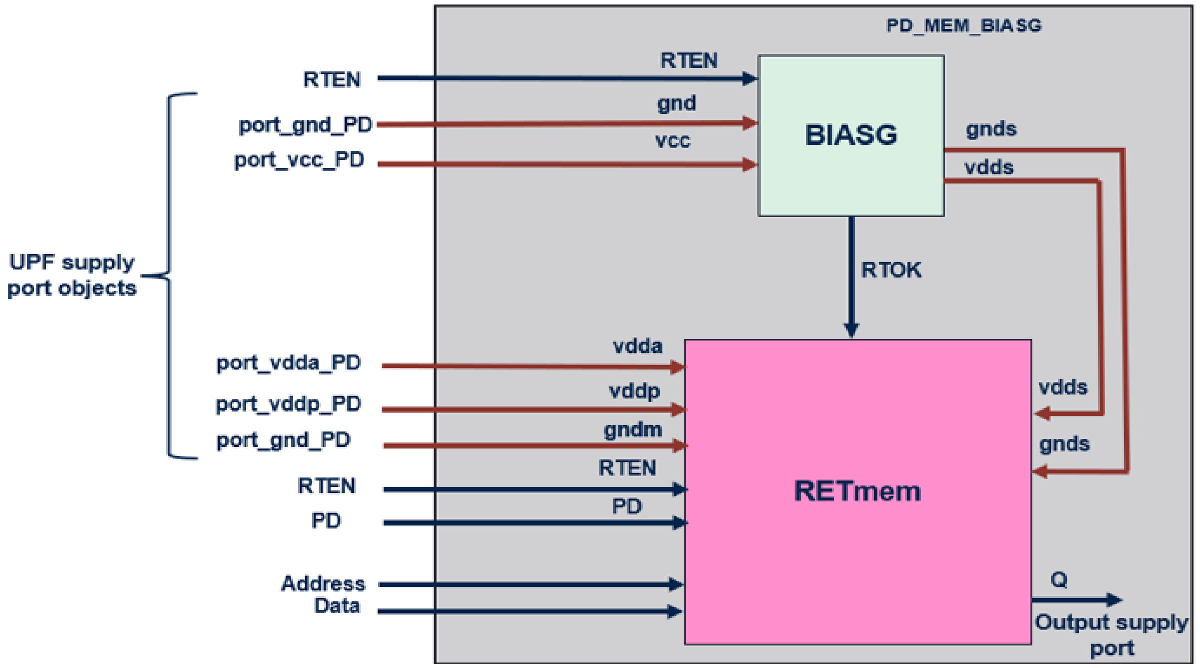
- Supply Checker :
  - Monitors voltage ranges and power sequences to ensure supply specification compliance based on operating mode.
- Logic Functional Behavior (LFB):
  - Represents the core logic of the design like traditional non power aware behavioral model.
- Supply Functional Behavior (SFB):
  - Models complex analog supply behaviors such as bias generation, voltage regulation, and scaling.



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

Multi-voltage Memory Subsystem : Case Study DUT

- Case study with Retention SRAM(RETmem) and Bias Generator (BIASG) blocks highlights the need for accurate voltage values and sequencing for low-power retention mode.



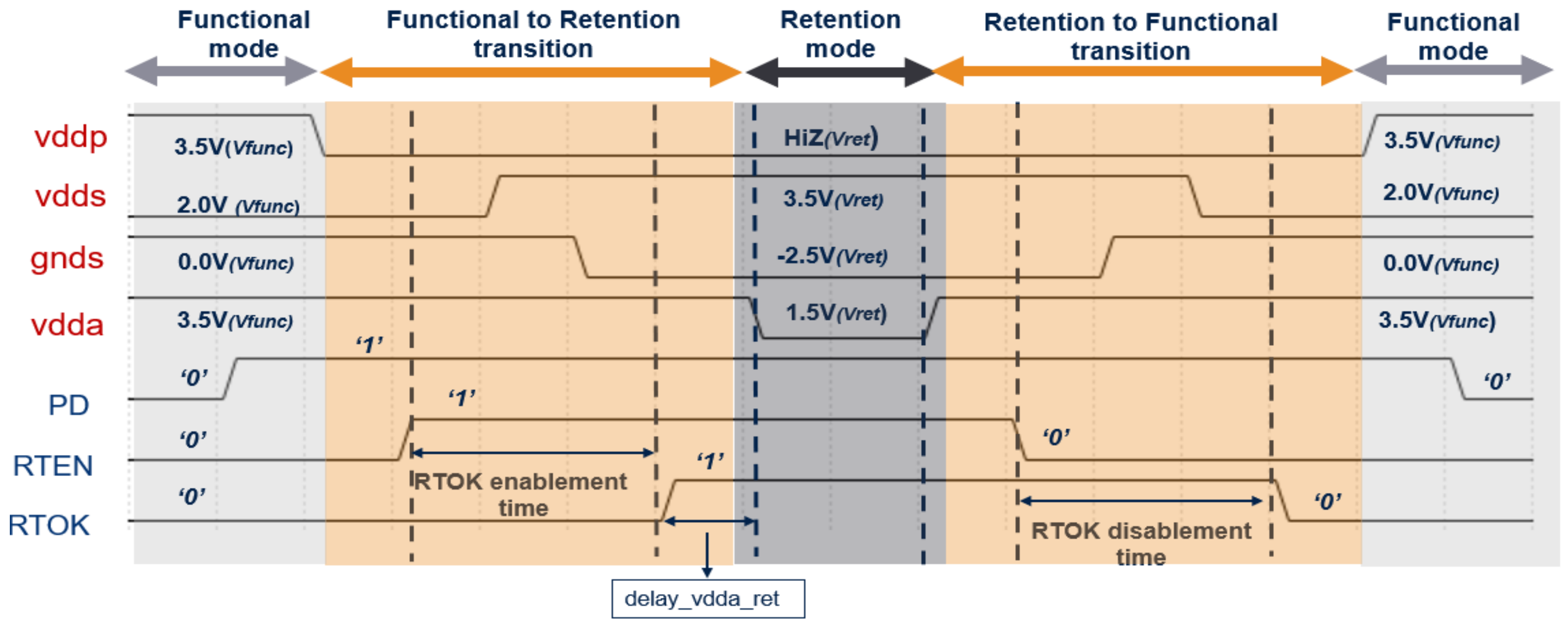
Signal	Typical voltage Value (Vfunc)	Typical voltage Value (Vret)
vdda	3.5V	1.5V
vddp	3.5V	HiZ
vdds	2.0V	3.5V
gnds	0V	-2.5V

🟦 Functional signals
🟥 Supply signals

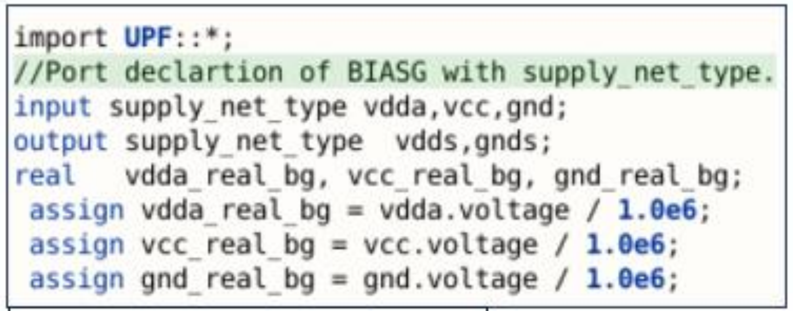
RETmem + BIASG Subsystem with typical supply voltage values in-figure table.

Multivoltage Memory Subsystem : Transition Sequence

- The mode transition sequence, is a critical design requirement for RETmem-BIASG subsystem and must be verified for functional sign-off.

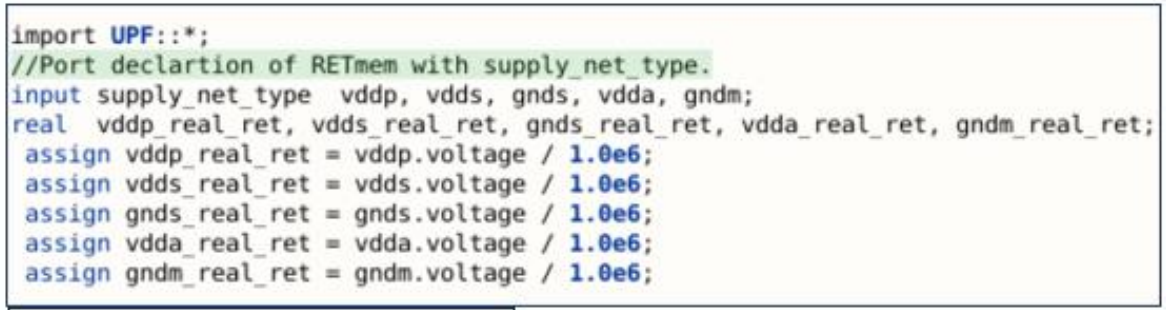


Implementation



```
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

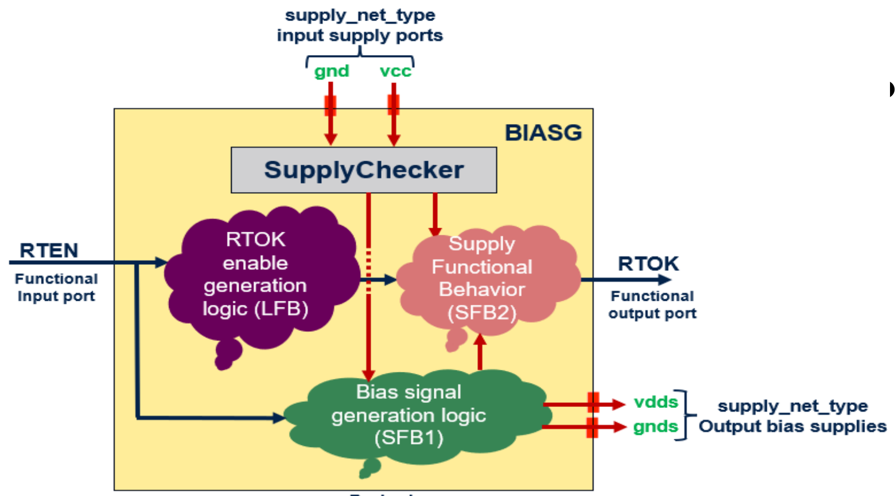


```
import UPF::*;
//Port declartion of RETmem with supply_net_type.
input supply_net_type  vddp, vdds, gnds, vdda, gndm;
real   vddp_real_ret, vdds_real_ret, gnds_real_ret, vdda_real_ret, gndm_real_ret;
 assign vddp_real_ret = vddp.voltage / 1.0e6;
 assign vdds_real_ret = vdds.voltage / 1.0e6;
 assign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

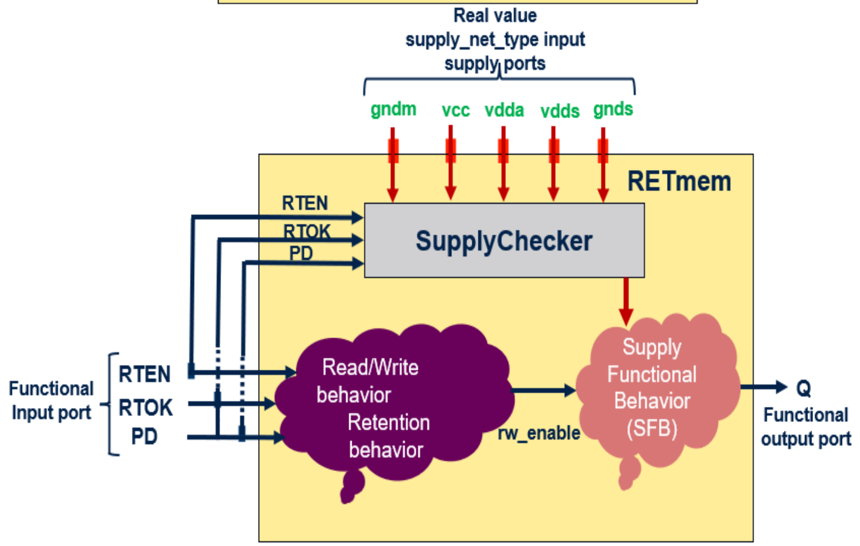
VA Interface of RETmem

- VA Interface –
  - Real voltage supply value declaration for BIASG block and RETmem block.
  - IEEE 1801 Std aligned SV-UPF with UDN *supply_net_type .*
  - Voltage value access of UDN supplies in model is done through *.voltage*

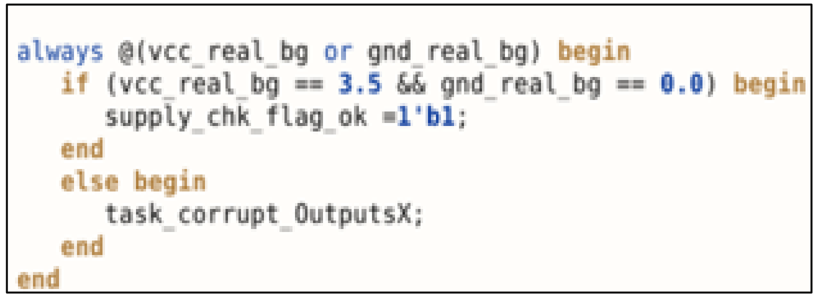
Implementation



supply_net_type input supply ports
gnd
vcc
BIASG
SupplyChecker
RTEN
Functional Input port
RTOK enable generation logic (LFB)
Supply Functional Behavior (SFB2)
RTOK
Functional output port
Bias signal generation logic (SFB1)
vdds
gnds
supply_net_type Output bias supplies

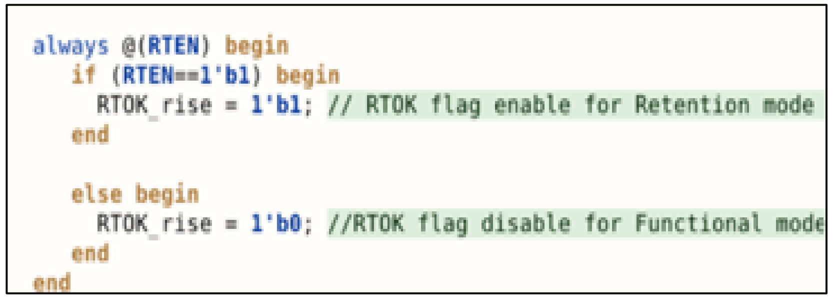


- VA Model –
  - Supply Checker ,SFB & LFB form core foundation for RETmem and BIASG.
  - Monitors supply states and transition sequence of subsystem.
  - If compliance fails, model outputs undefined logic (X) or else it proceeds with intended functional behavior .



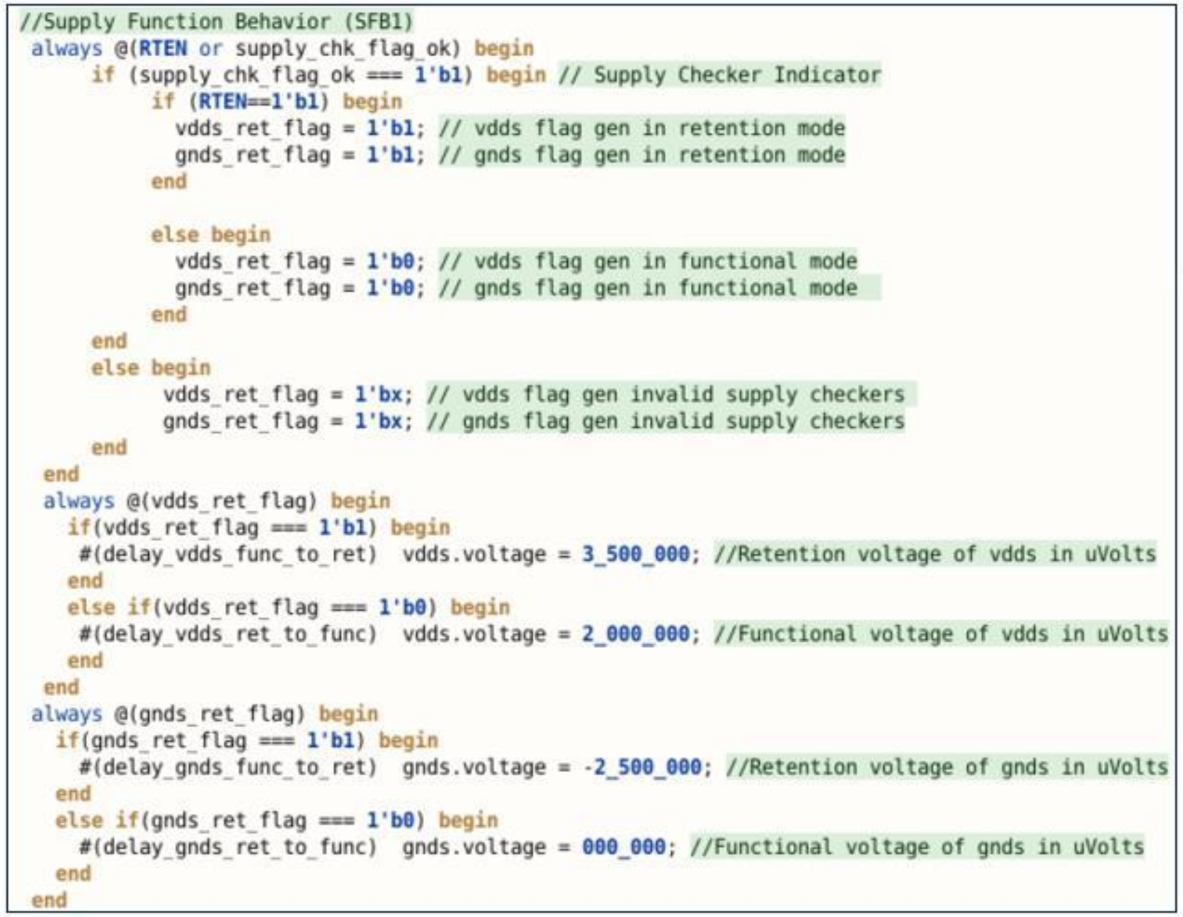
```
always @(vcc_real_bg or gnd_real_bg) begin
   if (vcc_real_bg == 3.5 && gnd_real_bg == 0.0) begin
      supply_chk_flag_ok =1'b1;
   end
   else begin
      task_corrupt_OutputsX;
   end
end
```

VA Model

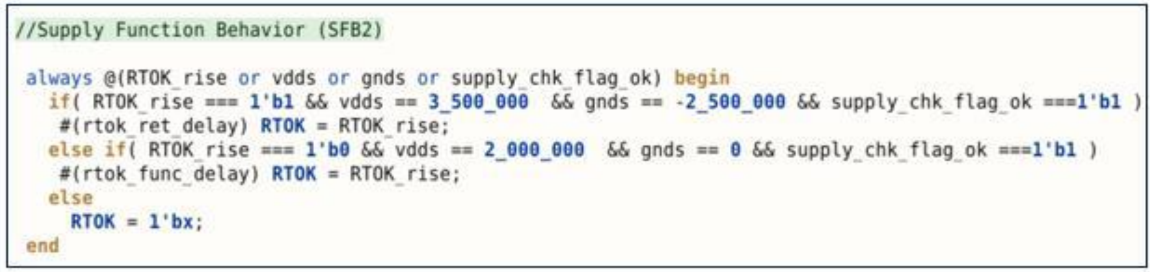


```
always @(RTEN) begin
   if (RTEN==1'b1) begin
     RTOK_rise = 1'b1; // RTOK flag enable for Retention mode
   end

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

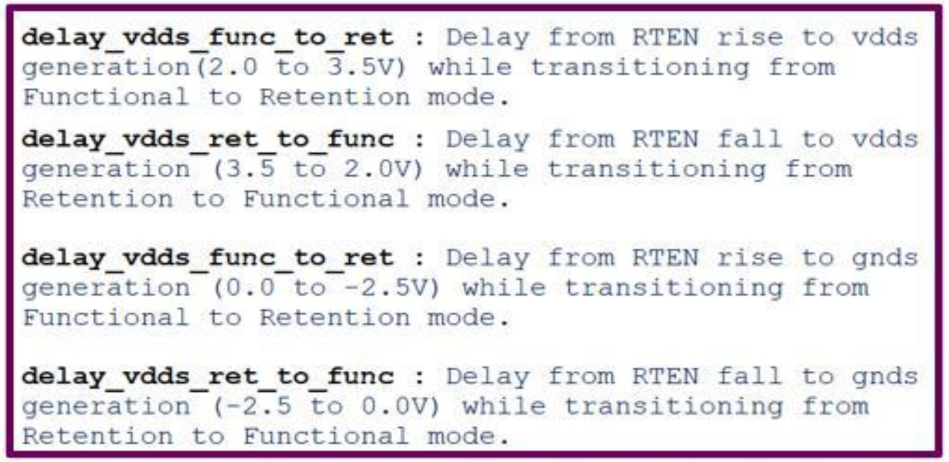
Implementation (BIASG)



```
//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.

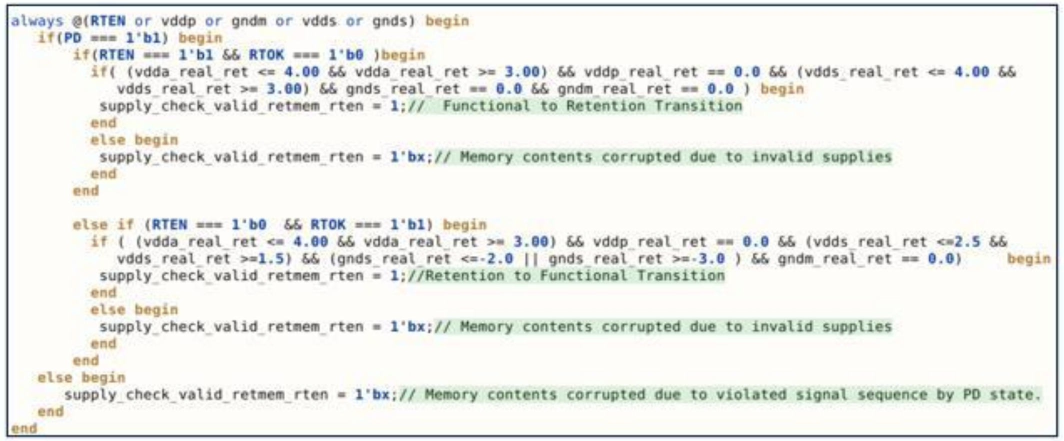
delay_vdds_ret_to_func : Delay from RTEN fall to vdds generation (3.5 to 2.0V) while transitioning from Retention to Functional mode.

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.

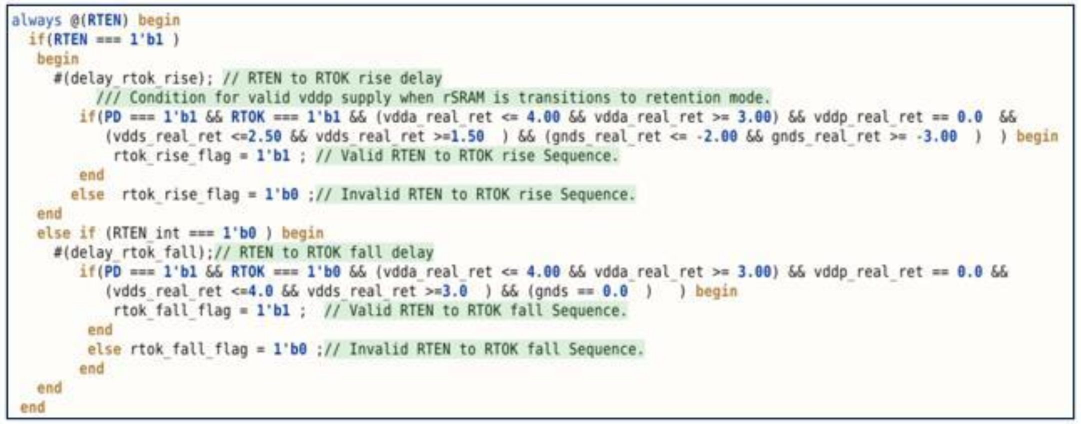
Sequence delays mentioned in SFB1 & SFB2

Implementation (Retmem)



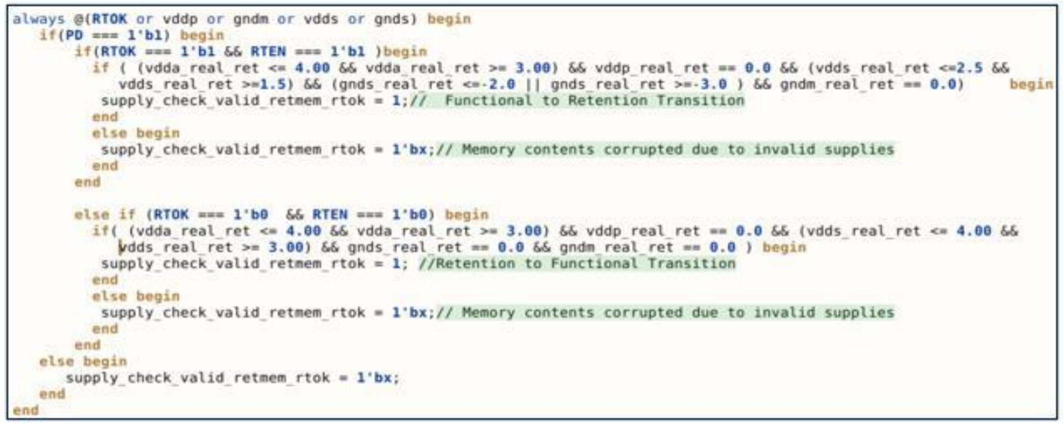
```
always @(RTEN or vddp or gndm or vdds or gnds) begin
   if(PD === 1'b1) begin
       if(RTEN === 1'b1 && RTOK === 1'b0 )begin
         if( (vdda_real_ret <= 4.00 && vdda_real_ret >= 3.00) && vddp_real_ret == 0.0 && (vdds_real_ret <= 4.00 &&
            vdds_real_ret >= 3.00) && gnds_real_ret == 0.0 && 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



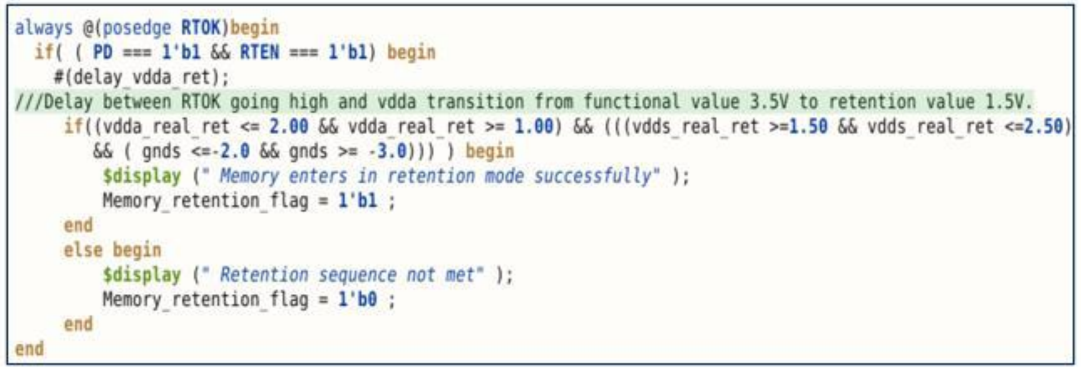
```
always @(RTEN) begin
  if(RTEN === 1'b1 )
   begin
     #(delay_rtok_rise); // RTEN to RTOK rise delay
          /// Condition for valid vddp supply when rSRAM is transitions to retention mode.
        if(PD === 1'b1 && RTOK === 1'b1 && (vdda_real_ret <= 4.00 && vdda_real_ret >= 3.00) && vddp_real_ret == 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

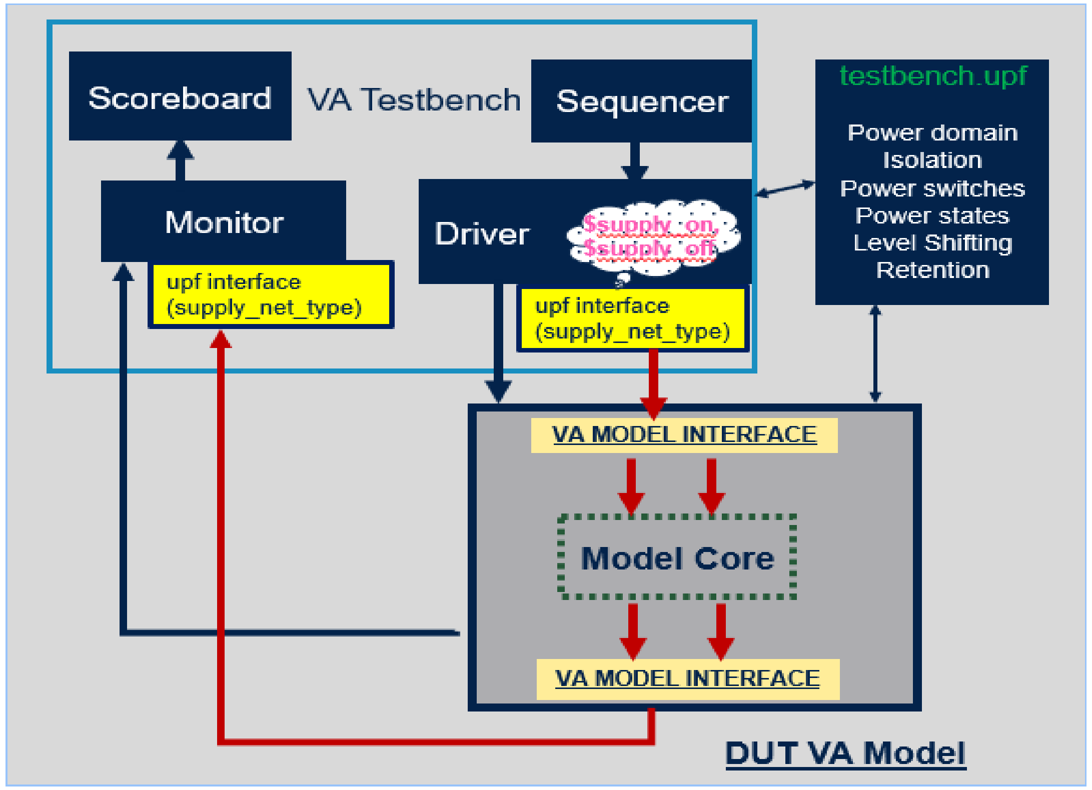


```
always @(posedge RTOK)begin
  if( ( PD === 1'b1 && RTEN === 1'b1) begin
     #(delay_vdda_ret);
///Delay between RTOK going high and vdda transition from functional value 3.5V to retention value 1.5V.
     if((vdda_real_ret <= 2.00 && vdda_real_ret >= 1.00) && (((vdds_real_ret >=1.50 && vdds_real_ret <=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

TB Implementation

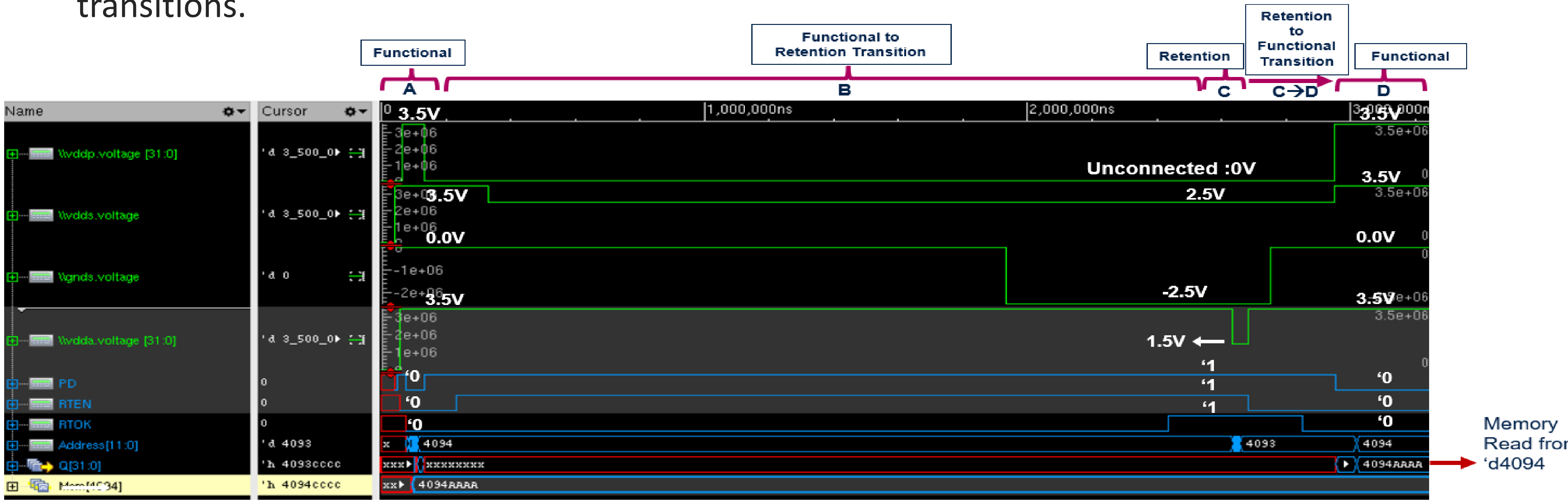
- VA Verification –
  - Test sequences cover Functional mode, Retention mode and mode transitions.
  - Real voltage values applied using the *$supply_on* and *$supply_off* functions.
  - Simulations executed on a UPF-compliant low-power simulator utilizing *pg_pin* definitions from Liberty file, power intent from UPF file.
  - Primary supply , functional test cases and invalid cases are executed and simulated.



Simulation Results

- Simulation results for Valid testcases
  - Test sequences cover Functional mode, Retention mode and mode transitions.

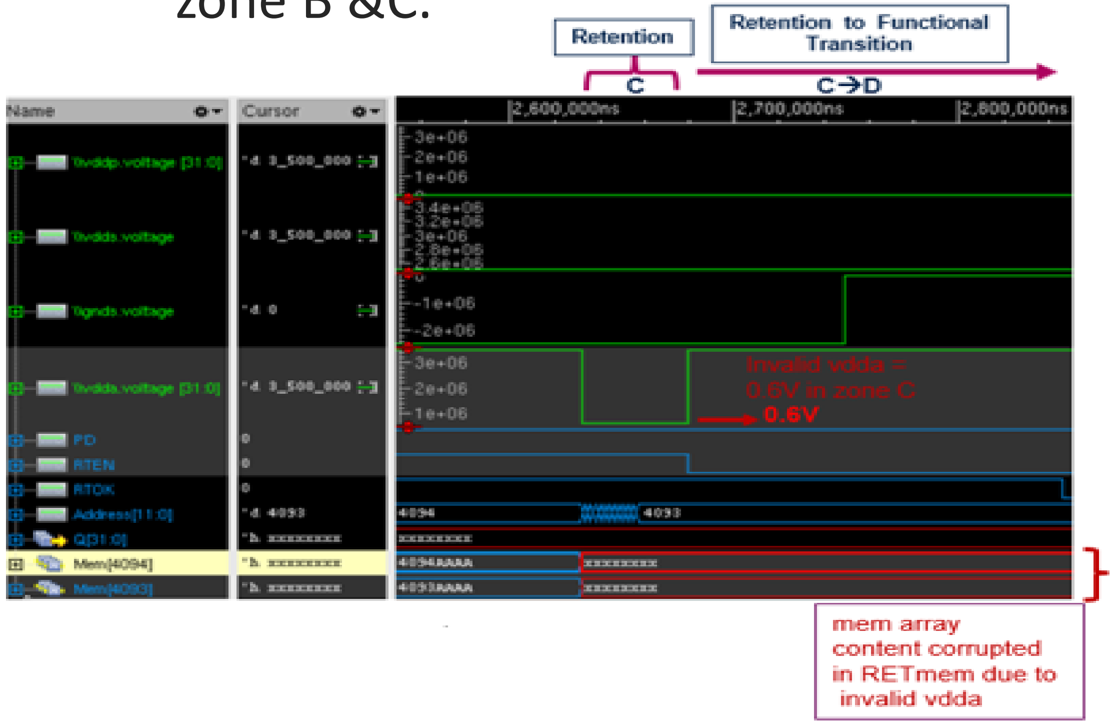
Mode	Voltage Set	vdda (V)	vddp (V)	vdds (V)	gnds (V)
SRAM Functional	Vfunc	3.0–4.0	3.0–4.0	3.0–4.0	0 (GND)
SRAM Retention	Vret	1.0–2.0	Hi-Z	1.5–2.5	-2.0 to -3.0

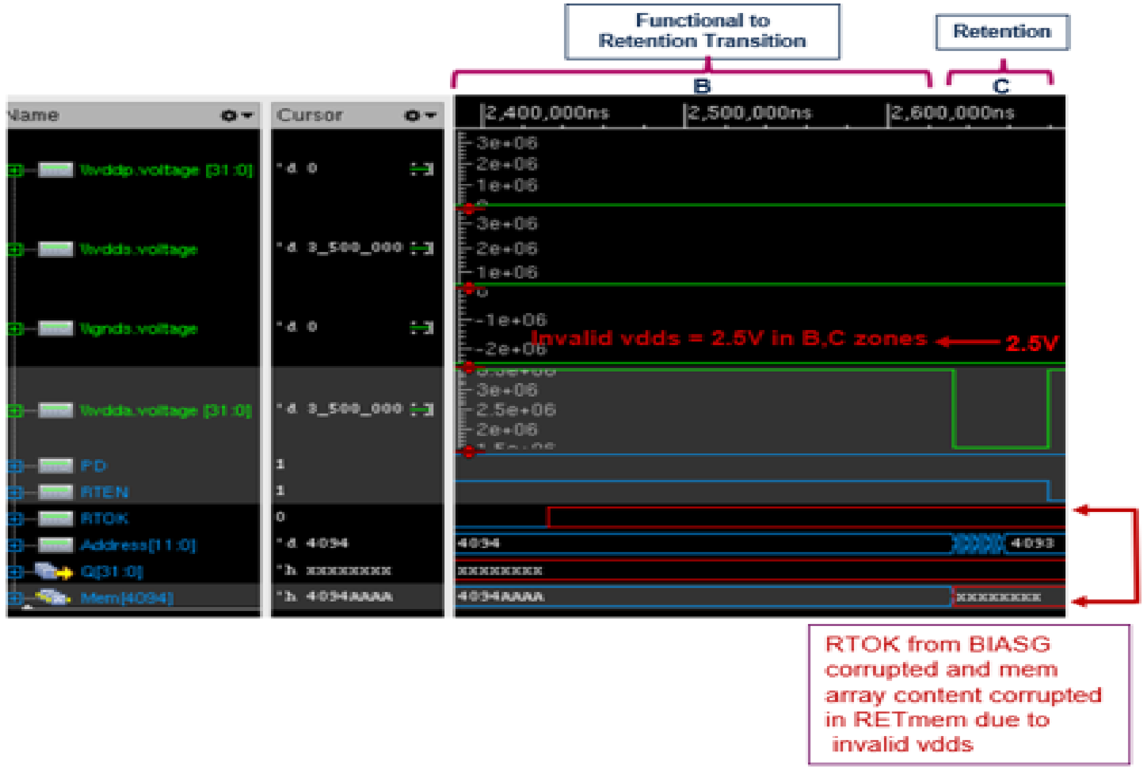


```
Name                         Cursor
\\vddp.voltage [31:0]        'd 3_500_0▸
\\vdds.voltage               'd 3_500_0▸
\\gnds.voltage               'd 0
\\vdda.voltage [31:0]        'd 3_500_0▸
PD                           0
RTEN                         0
RTOK                         0
Address[11:0]                'd 4093
Q[31:0]                      'h 4093cccc
Mem[4094]                    'h 4094cccc
```

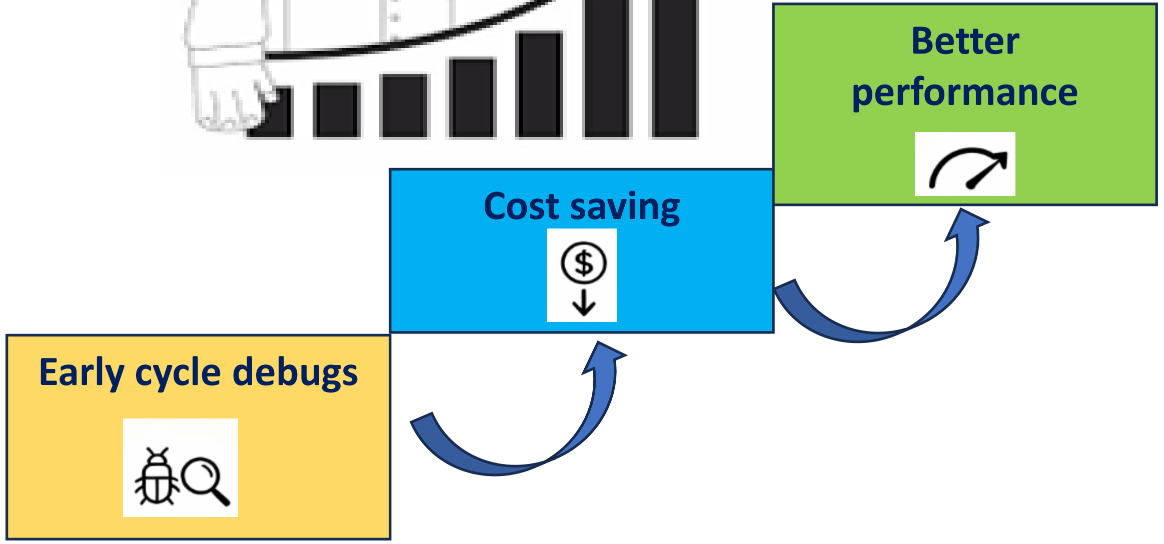
Simulation Results

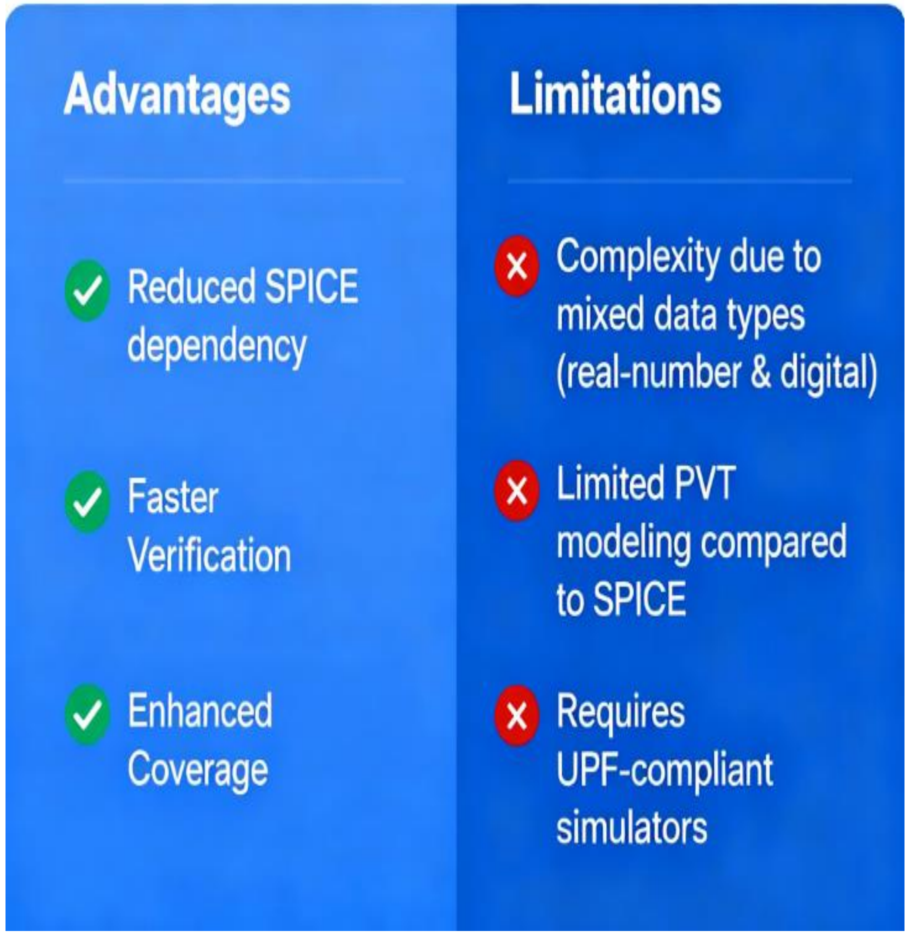
- Simulation results for Invalid testcases
  - Test sequences cover Functional mode, Retention mode and mode transitions.
  - VA model behavior at invalid state of vdda = 0.6V in zone C & C->D and vdds = 0.5V in zone B &C.





Tradeoffs

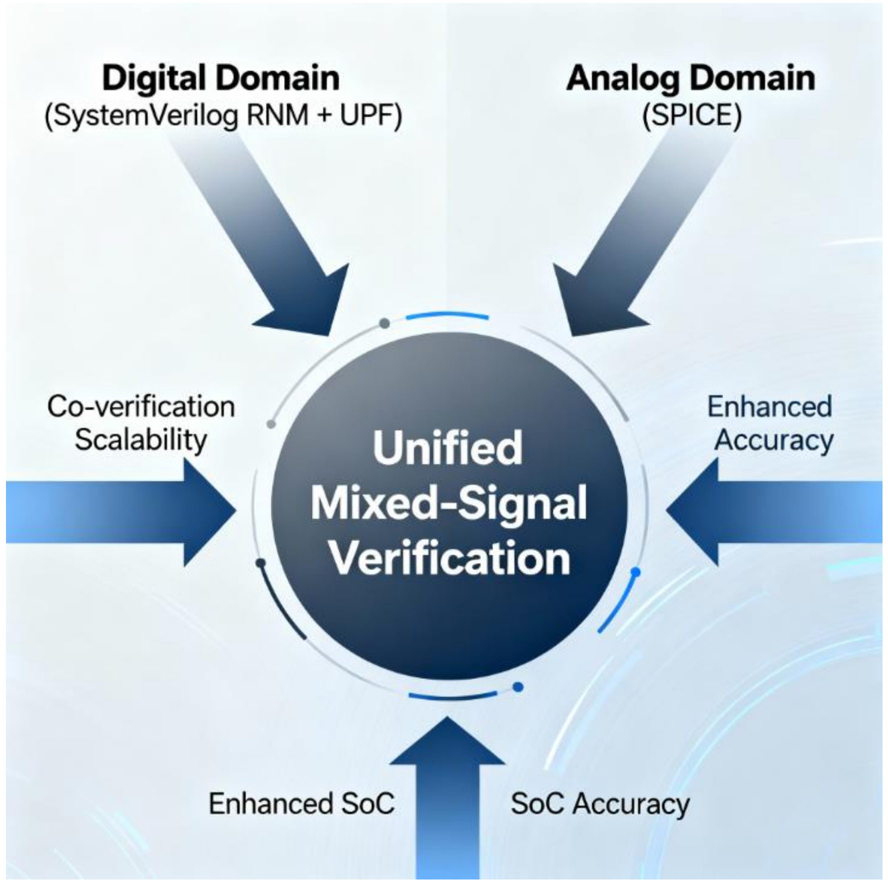


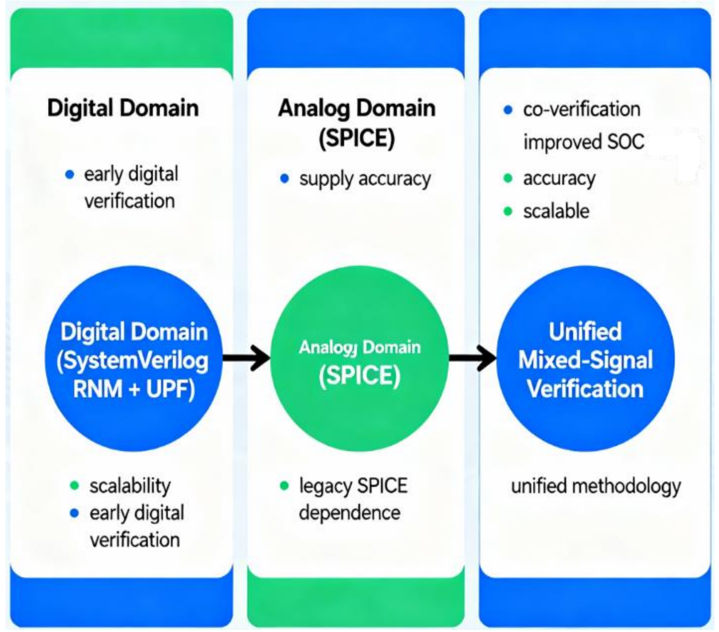


Advantages	Limitations
✓ Reduced SPICE dependency	✗ Complexity due to mixed data types (real-number & digital)
✓ Faster Verification	✗ Limited PVT modeling compared to SPICE
✓ Enhanced Coverage	✗ Requires UPF-compliant simulators

Future Work

- Extension to Mixed-Signal Simulations
  - Integrate real-number voltage modeling into mixed-signal flows.
  - Seamless co-verification of analog & digital domains.
- Impact
  - Strengthen low-power verification in advanced designs.
  - Support scalable, unified verification methodology across domains.


Digital Domain
(SystemVerilog RNM + UPF)
Analog Domain
(SPICE)
Co-verification Scalability
Unified Mixed-Signal Verification
Enhanced Accuracy
Enhanced SoC
SoC Accuracy



Thanks

Q&A