2014 DESIGN AND VERIFICATION DVCON CONFERENCE AND EXHIBITION INDIA
Sept 25-26, 2014
Hotel Park Plaza
Bangalore, India

Efficient methods for analog mixed signal verification

Interface handling methods, trade-offs and guidelines

Lakshmanan Balasubramanian, Bharath Kumar Poluri, Texas Instruments (India) Pvt. Ltd., Bangalore, India (*lakshmanan,bharath.poluri@ti.com*)

Shoeb Siddiqui, IIT Madras, Chennai, India (siddiquishoeb.iitm@gmail.com)

Vijay Kumar Sankaran, Cadence Design Systems (India) Pvt. Ltd., Bangalore, India (vijay@cadence.com)

***Abstract*—Increasing complexity of practical analog and mixed-signal systems on chip designs, mandate the use of several co-simulation methods including analog mixed signal and digital mixed signal co-simulation concepts for tangible, comprehensive and timely pre-silicon verification closure. All co-simulation methods require connect modules or interface elements in the path of the signals that cross simulator boundaries, for example simulation disciplines. In this paper we provide comparison of different style of CMs, identify the scenarios where they are applicable and the effort needed to support them. We will elaborate on the concept behind and the details of the automation to support each of the methods, including the relative cost of implementation. All these methods have been applied on several commercial mixed-signal SoC designs and the latest being the implementation of the automated solution for true supply sensitivity.**

***Keywords*—Analog mixed signal co-simulation; Digital mixed signal co-simulation; connect modules; interface elements**

I. Introduction

There has been a trend of ever increasing amount of analog and mixed-signal (AMS) content and complexity of systems on chip (SoC) especially for those applications targeting embedded processing, connectivity, sensor networks including internet of things (IOT) or ambient intelligence [1][2][3]. The amount of analog integration increases to reduce the overall cost to the end user, increase the system level flexibility to support varying end application scenarios, ease of handling field level trade-off between power and performance requirements. This also causes increase in the complexity of integrated SoC resulting in very high demand on pre-silicon verification and validation to minimise time to market and cost of development. A very significant portion of this verification complexity comes from AMS contents. While the digital and system level focus has dominated the SoC verification for long, these recent trends mandate varying levels of device / transistor level (TL) comprehension during verification to get the required focus on AMS contents. A high level view of a typical SoC level AMS co-simulation environment is illustrated in Figure 1 [4].

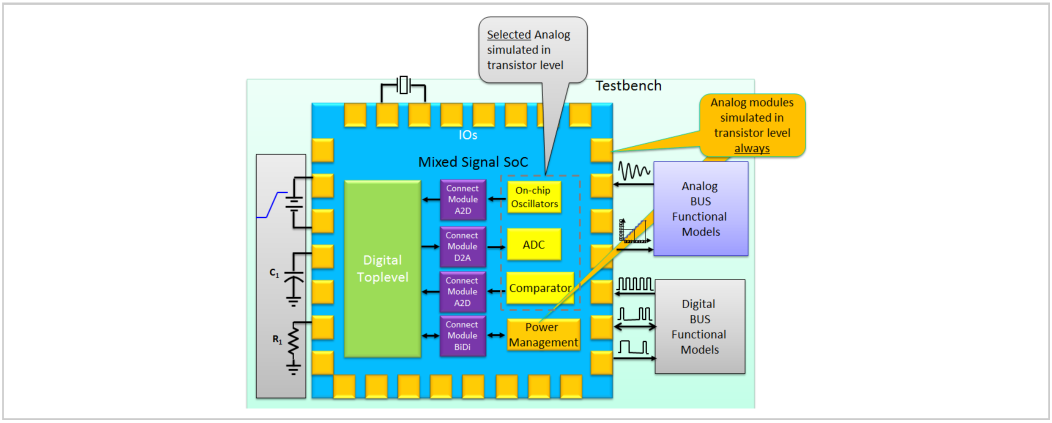

Figure 1. Typical mixed signal SoC AMS co-simulation environment

Performing AMS co-simulation keeping all of analog in TL is impractical due to heavy cost of large simulation run times even for moderate accuracy requirements. Hence intermediate abstraction levels for analog functions are necessary to speed up the simulations while having sufficient accuracy. These are provided by what are called analog functional behavioural models (ABMOD). Even in the presence of ABMOD, due to the gap in comprehensiveness of models and their validation against specification and implementation, there is always a requirement for AMS co-simulation with certain identified analog modules, though not all, in TL with an optimal mix of digital, analog in ABMOD and analog in TL.

In all such AMS co-simulation there always are signals that traverse disciplines (analog and digital, electrical and logical) and domains (Boolean and real number domains; voltage and power domains). These require special constructs called connect modules (CM) or interface elements (IE) to handle them appropriately.

This paper focuses on the various CM flavours, their evolution, relative strengths, and weaknesses or gaps that result in their limited applicability. This paper provides recommendations on their applicability and a flow that automates handling of these CM comprehensively and virtually frees a verification engineer completely from comprehending these artificial but necessary constructs. The recommendations have been applied on an industrial case study of verification of a commercial mixed-signal embedded system.

The rest of the paper is organised as follows: Section II explains the context and need for CMs in co-simulation, and different classifications of them; Practical automation needs in handling in CM especially in the context of power managed designs and a summary of their comparison are presented in section III followed by concluding remarks in section IV.

II. Connect Modules and Interface Elements

The basic functionality of a CM is to translate a signal that is crossing between disciplines and maintain consistency of the signal status or level in all connected disciplines [5][6][7][8]. The logical (L) and electrical (E) disciplines are common in all analog mixed signal co-simulations and hence we will limit our discussion to those disciplines in general. With the advent of digital mixed signal co-simulation (DMS), and real numbered (R) ABMOD to represent analog functions, there is also need to translate between the real and Boolean domains though they both are part of logical discipline. The basic need is same whether it is an electrical discipline or a real domain to interpret a continuous signal or variable quantity into a specific logic state based on pre-defined thresholds as illustrated in Figure 2. These thresholds may be either statically defined or can be defined as a proportion of a known supply level.

Power managed design with multiple voltage and/or power domains are getting so common and necessary for meaningful and complex power-performance trade-off, and system level flexibility needed for many mixed-signal embedded processing end use scenarios. In such a scenario the use of connect modules have to perform more than just signal discipline translation. They have to comprehend the availability and voltage level of driving/driven power supplies (especially the power supply of the design partition in logical discipline) and do the signal translation consistent with the electrical status of such power supply.

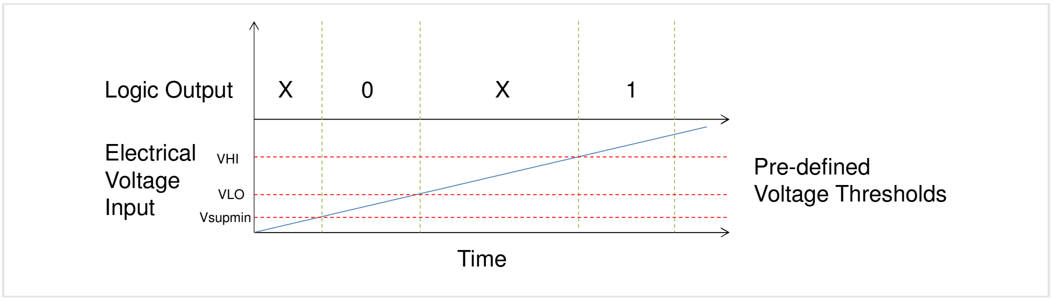

Figure 2. Basic functionality of a Connect Module in the Electrical-Logical discipline crossing

Depending on the details that are being focused during AMS co-simulation CMs may have multiple requirements, including the following

1. CMs need to be aware of the related supply in either a static or dynamic way,

2. CMs need to handle either potential (voltages) or flow (currents)
3. CMs further need to reflect or inherit the port impedance.

Based on the supply sensitivity they may be further classified as following, and as illustrated in Figure 3.

a) Static or non-supply aware, and

b) Supply sensitive

  a. Through global node, also called supply inheritance

  b. Through actual physical electrical connectivity, also called true supply sensitivity.

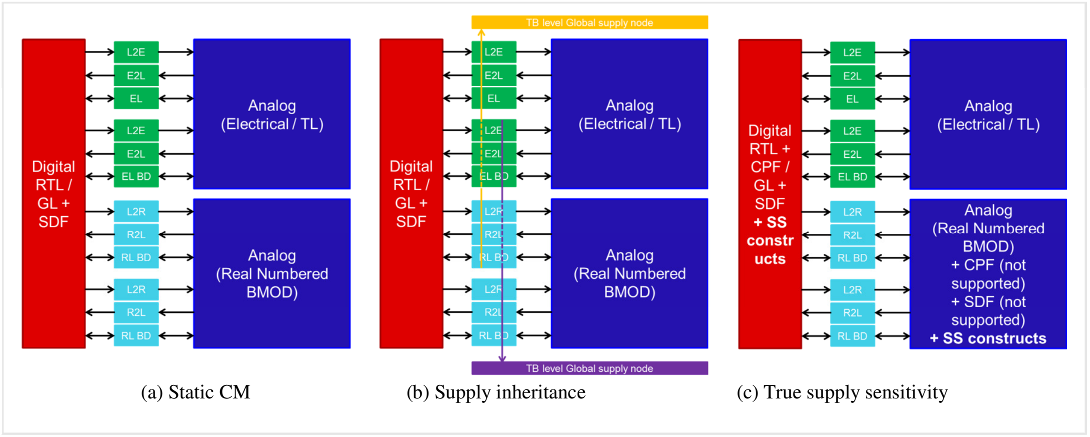
TB level Global supply node
Digital RTL / GL + SDF
L2E
E2L
EL
EL BD
L2R
R2L
RL BD
Analog (Electrical / TL)
Analog (Real Numbered BMOD)
Digital RTL + CPF / GL + SDF + SS constructs
Analog (Real Numbered BMOD) + CPF (not supported) + SDF (not supported) + SS constructs
(a) Static CM
(b) Supply inheritance
(c) True supply sensitivity

Figure 3. Classification of connect modules based on supply sensitivity

Depending on the depth and breadth of details of a signal path that are getting modelled in a CM, they may affect the accuracy and run-time of the simulation.

A. Static Connect Modules

The function of static CM is illustrated in Figure 4 are the easiest and most suitable to handle in simple yet non-power managed systems that are just powered by one master supply. They are also the naturally the earliest one in the evolution of CMs. But static CMs cannot inherently handle meaningfully a power managed system with more than one voltage and/or power domains. There exist practical work-around solutions possible requiring manually intensive, iterative, and error prone process to enable some basic handling of power managed designs which will be discussed in detail in the next section.

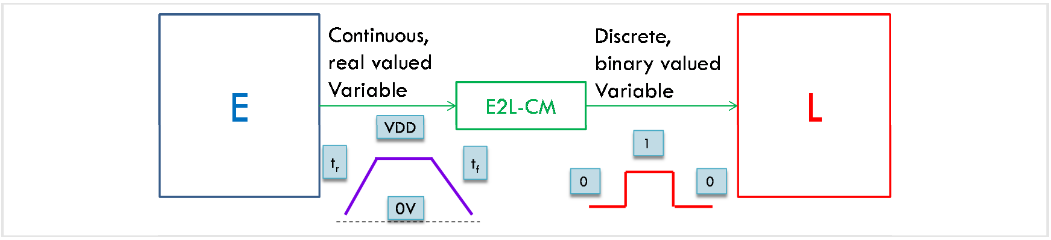
E
Continuous, real valued Variable
VDD
t_r
t_f
0V
E2L-CM
Discrete, binary valued Variable
1
0
0
L

Figure 4. Static CM

B. Supply Inherited Connect Modules

The supply inheritance (shown in Figure 5) or sensitivity is a natural next step in the evolution. Handling supply inheritance through global electrical connectivity is done by assigning a specific node, usually at the top most level of design hierarchy, which is electrically connected to the supply source stimuli and is representative of a particular voltage or power domain. One such node is identified for each domain, and each domain is associated with a dedicated set of CM source definition which has direct hard-coded supply inheritance or sensitivity information embedded in itself. Yet identifying and associating each interface with a right voltage or power domain is not a trivial and mean task, and require a lot manually intensive, iterative, and error prone process. The authors of this paper have come up with a custom, yet scalable process to derive this information from existing power management implementation and back-end flows in a automated way. But still this flow requires this information to be generated at the SoC level integration stage and handled at verification stage, which is a sub-optimal method.

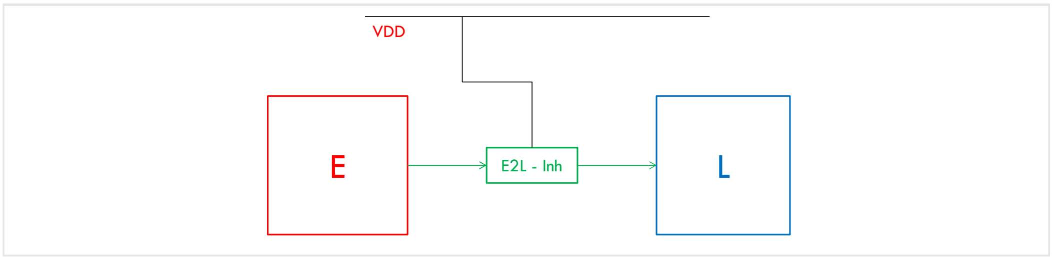

Figure 5. Supply inherited CM

C. *Supply Sensitive Connect Modules*

The supply sensitivity (shown in Figure 6) information is a specification to each IP, module or functional unit and all necessary information is already available at the IP design stage. Hence handling them and providing necessary supply sensitivity information as a construct at each IP deliverable (a functional model) is the most optimal way of handling this problem. But this requires a lot of manual effort, at initial stages of design once per IP design including all the standard cells, IO cells, and analog functional modules. Though this is not a recurring effort, it is still not a trivial and easy task. Hence the authors of this paper have come up with an automated flow to handle this based on existing power management specification or deliverables like CPF/UPF and others like IPXact views, and schematic design front end tools.

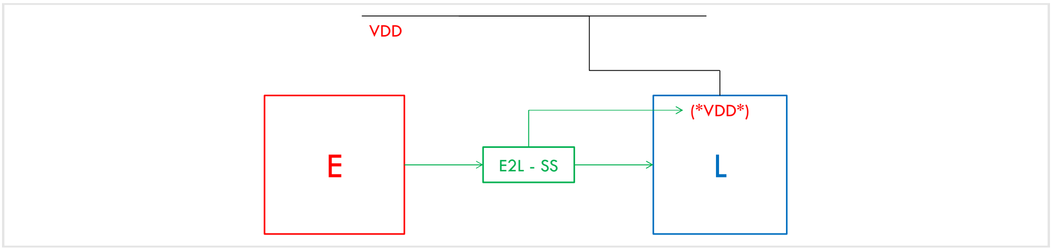

Figure 6. Supply sensitive CM

D. *CPF / UPF based Connect Modules*

There have been recent developments in the AMS co-simulation to handle them automatically using CPF/UPF but they are limited to only AMS co-simulations using RTL view. The supply domain definitions provided in the CPF/UPF are used to decide the status of the supply validity at any point in simulation which is then used to determine whether to corrupt (make 'x') the signals sensitive to that domain. There is a need for AMS co-simulation using gate level abstraction with timing annotation for covering timing critical analog-digital interface paths. To support such requirements, one need the true supply sensitivity without having to resort to direct use of CPF/UPF, since the use of CPF/UPF for supply sensitivity in AMS co-simulation over rides the actual physical / electrical supply connectivity available in the power connected gate level netlist.

III. Comparison and Practical Handling of Connect Modules in Power Managed Designs

As was briefly described in Section II, there is a dire need to comprehend the need for supply connectivity, its level at a given time point in simulation and its effect on the signal crossing between disciplines. In this context the supply inherited, CPF/UPF based and true supply sensitive connect modules are important. Each of them have different tool flow requirement which will be discussed in detail in this section.

A. *Supply sensitivity with static CM*

It is possible to get the function of supply sensitivity with static CMs through an iterative, manual effort. To understand this aspect better let us consider an example of the static CM coded in Verilog-AMS (VAMS) as shown in Example 1. As can be seen there is a local variable vsup declared and used to handle the supply level, and all the logic decision thresholds named vthi, vtlo, vtol are defined as a dependent function of vsup. Thus the logic decision thresholds follow the supply level and thus this CM can support a static supply sensitivity.

```
`include "disciplines.vams"
`timescale 1ns / 1ps
//==========================================================================
==============
connectmodule E2L (Ain, Dout);
  input Ain; electrical Ain;                    // electrical input
  output Dout; \logic Dout;                     // logic output
// INSTANCE PARAMETERS:
  parameter real vsup=3.6 from (0:inf);          // nominal supply voltage
  parameter real vthi=vsup/1.5 from (-inf:vsup); // upper input threshold
  parameter real vtlo=vthi/2 from (-inf:vthi);   // lower threshold
  parameter real vtol=vsup/100 from (0:(vthi-vtlo)/4]; // voltage tolerance
…
// LOCAL VARIABLES:
  reg Dreg;              // output register
…
//==========================================================================
==============
  initial begin
…
    Dreg=1'b0;           // initial level
…
  end
…
endmodule
```

Example 1. VAMS code snippet of static CM definition

To support a design with more than one supply voltage and/or power domains, one need to follow the following procedure.

1. Identify all the CM instances inserted by the tool after compilation and elaboration of the design including testbench.
2. Based on the design knowledge classify these CM instances among the different voltage domains of the design.
3. Assign the variable *vsup* for each CM instance to the respective real numerical value corresponding to the voltage level of the domain. An example is illustrated for a supply level of 3.0V below:

   defparam testbench.duv.a_3v_E2L.vsup=3.0;   *defparam testbench.duv.b_1p8v_E2L.vsup=1.8;*

4. Pursue with simulation; now the setup can comprehend the difference between different voltage domains even though in a static manner.

The limitation of this approach though, is any dynamic supply voltage level changes during the simulation will not have any effect on the signal levels, as against the reality to the contrary. The dynamic supply sensitivity will be useful especially in cases where the systems are with integrated power supplies; when the system level power-performance optimization requires exercising multiple modes of operation; Such modes include varying the supply voltage levels on demand at run time and shutting of on-chip supplies when not needed.

B. *Supply sensitivity with supply inherited CM*

Supply inheritance is implemented as discussed in section II.B by identifying a unique node representing a voltage domain for all voltage domains in the system and using them thus for dynamically inheriting the supply level. To understand this aspect better let us consider an example of the inherited CM coded in Verilog-AMS (VAMS) as shown in Example 2. As can be seen the electrical variables *vdd! & vss!* declared for supply inheritance are assigned not with static numerical values, but with unique hierarchical references each representing a node, usually but not necessarily at the top most hierarchy.

```
connectmodule E2L_inhconn (Ain, Dout);
  input Ain; electrical Ain;                    // electrical input
  output Dout; \logic Dout;                     // logic output
// Inherited vdd! and vss!
  electrical
     (* integer inh_conn_prop_name="vdd";
        integer inh_conn_def_value="cds_globals.\\vdd! "; *) \vdd! ;
  electrical
     (* integer inh_conn_prop_name="vss";
        integer inh_conn_def_value="cds_globals.\\vss! "; *) \vss! ;
// INSTANCE PARAMETERS:
 parameter real vsup_min=0.5 from (0:inf); // min supply for normal operation
// scaled input/output levels/thresholds (0 maps to Vref, 1 maps to vdd-vss):
 parameter real vthi=2/3   from (0:1);      // frac. for high tresh (def=2/3)
 parameter real vtlo=1/3   from (0:vthi);   // frac. for low tresh (def=1/3)
 parameter real vtol=(vthi-vtlo)/4  from (0:(vthi-vtlo)/2]; // frac. for vtol
 parameter real vtlox=vtlo+vtol from (vtlo:vthi);  // lo to X state threshold
…
 parameter real vsup_off=vsup_min-vtol from (0:vsup_min); //shut-off supply
// LOCAL VARIABLES:
 reg Dreg;               // output register
 …
real Vds;               // supply voltage
 real Vas;               // input voltage
//================================================================
=============
 initial begin
…
   Dreg=1'b0;            // initial level
…
 end
…
 analog begin
   Vds = V(\vdd! ,\vss! );
   Vas = V(Ain,\vss! );
   …
 end
endmodule
```

Example 2. VAMS code snippet of supply inherited CM definition

Existing tool flows cannot automatically identify the different voltage domains and hence one needs the following procedure to use this for power managed designs.

1. Identify all voltage domains in the system.

2. Create unique CM definition per domain by differentiating the supply node reference between the different domains as shown below and defining a unique logic discipline (say logicavdd & logicdvdd) for its logical signals.

   cds_globals.\\vdd! will be replaced by *testbench.ams_avdd & testbench.ams_dvdd* for different CMs representing the two domains; and correspondingly *cds_globals.\\vss!* will be replaced by *testbench.ams_avss & testbench.ams_dvss* for different CMs representing two domains; where testbench is the top most design element of the simulation setup, ams_avdd, ams_dvdd, ams_avss and ams_dvss are the top most net names corresponding to different supplies and grounds of the two domains.

3. Based on the design knowledge classify the different cells, cell instances, instance terminals and nets among one of the identified logic domains corresponding to its power supply voltage / power domain. One practical method is to use the backend physical design tool that has the knowledge of different voltage / power domains in the design.

4. Pursue with compilation, elaboration and simulation; now the setup can comprehend the difference between different voltage domains.

The above described method can enable dynamic supply sensitivity during AMS co-simulation, but is limited to comprehend the voltage domains as identified earlier and follow the dynamics of the identified hierarchical electrical node. This identified node will now behave like a global connectivity and will override any physical electrical supply connectivity that exists in the design netlist (power connected gate level for the digital and SPICE/spectre for the transistor level). CPF/UPF based CM in effect achieves the same purpose as supply inherited CM except that the supply voltage/power domain information is coming not from an electrical net (could be global; may or may not be connected to all design elements, but has to be connected to the supply stimuli in electrical domain) but from CPF/UPF definitions of voltage/power domains. In both cases the information from such a global definition in CM overrides that dynamically available through physical electrical supply connectivity during simulation. The domain definition in CPF/UPF is more logical in nature. Hence particularly both of them are not very useful in AMS co-simulation with a power connected gate level abstraction.

C. *Supply sensitive CM*

A true and fully supply sensitive (SS) CM is illustrated in Example 3. As can be seen the only different to the inherited CM is the supply sensitivity information as are highlighted. This also requires each logical design element in the design to have the same supply sensitivity information but assigned to its local supply terminal or net, an example is provided below.

```
input VCC;
input VDD;
input (* integer supplySensitivity = "VCC" ; integer groundSensitivity = "VSS" ; *) a_3p0v;
input (* integer supplySensitivity = "VDD" ; integer groundSensitivity = "VSS" ; *) a_3p0v;
```

The discipline resolution has to comprehend and resolve various scenarios including falling back to default sensitivity information to a global when there is no valid information available from design, handling bidirectional signals spanning across disciplines, handling real and Boolean (logical) type signals in logical domain. Functional models of all design elements including standard cells, I/O cells, ABMOD should have an embedded supply sensitivity information as discussed above [6]. While it is easier to comply with this requirement when they are under design, it would be difficult to migrate an existing design to support SS CM. A practical approach requires a custom automation to comprehend the domain sensitivity information for all signals in a design from existing design artefacts including specifications, design intent in IPXACT or CPF/UPF, and/or information in design schematic front end and automatically include the supply sensitivity information in all functional models including ABMODs. The authors have implemented this using executable scripts in Perl and are collaborating with select commercial EDA vendors to support such features in a formal tool flow [9].

1) Coexistence of DMS & AMS

When all the ABMODs use only logical and electrical constructs with no real number then SS CM works

```
connectmodule E2L_ss  (Ain, Dout);
  input Ain; electrical Ain;                    // electrical input
  output Dout; \logic Dout;                      // logic output
// Supply Sensitivity attributes
  electrical (* integer supplySensitivity = "cds_globals.\\vdd! " ; *) \vdd! ;
  electrical (* integer groundSensitivity = "cds_globals.\\vss! " ; *) \vss! ;
// INSTANCE PARAMETERS:
…
initial begin
…
end
…
 analog begin
   …
 end
endmodule
```

Example 3. VAMS code snippet of fully supply sensitive CM definition seamlessly irrespective of mixing and matching of ABMODs and TL netlists for different parts of the designs. When the real number ABMODs are used to enable DMS, the co-existence of RN, electrical and conventional logic data types cause some practical difficulties in using SS CM. Currently tool flows only support signals and sensitive supply to be of same data type (*REAL* signal sensitive to supply of *REAL* type and *ELECTRICAL* signal sensitive to supply of *ELECTRICAL* type), which we call as homogeneous SS CM. It is difficult to avoid heterogeneity when the system has real number ABMODs using Boolean logical (For example, Verilog *wire* and VHDL *std_logic*) data type for logic signals, and RN data type (For example, VHDL *real* and Verilog *wreal*) for analog behavioural signals, and all of digital functionality in standard Boolean logical models. Existing tool flows don't support full heterogeneity with real type signals sensitive to electrical supplies or vice-versa. But a partial heterogeneity is supported in Cadence® Incisiv™ tool flow for AMS co-simulation, by automating a merged CM insertion in the supply path if the logical domain uses real type supplies and real/logical signals. This supports logical/real signals to be sensitive to real type supplies. It just requires a minor change in the SS CM definition to define the supplies and grounds in wreal type and consistent / appropriate use of the related signals / variables in the CM definition. The heterogeneity may be achieved either by CM redefinition or by certain algorithmic changes to the discipline resolution to optimize for hierarchical CM placement, which is not yet implemented in any commercial tool flow. The impact of the choice is also discussed in Section III.C.4) below.

2) Some practical tips in using SS CM

Some practical issues in using SS CM are identified and listed below.

a) Applying logical operation on the supply: This is mainly done to comprehend power awareness (PA) in gate level simulations (GLS). Use of electrical supplies in all logical design entities enable ease of AMS co-simulation with SS CM, it adds unnecessary co-simulation overhead for essentially digital RTL/GL simulations. While use of real supplies in all logical design entities enable consistent and seamless DMS and AMS co-simulation flows, such usage with PA functions require insertion of real-to-logical (R2L) CM in the supply path in each logical design element i.e., gate. But it becomes practically difficult due to a cyclic dependency caused by the supply sensitivity requirement on this CM itself. While this is not supported in existing flows, even if it is supported hypothetically it will result in a huge blow up in the number of CMs which may cause severe performance degradation in terms of simulation rum time. A practical solution for this would be to either rewrite these functions in real data type (can be limited to AMS only by using conditional compiler directives); or the bulk/n-well/substrate terminals (where applicable, for example VBBNW, VBBPW, etc.) of standard cells for supply sensitivity by keeping them real type while using the supplies for PA function by keeping them logical or vice-versa. Where these are impractical, one can safely strip-off the PA function only in AMS mode by using specific conditional compiler directives as the SS CMs on the signal path just do the job of PA for the peripheral logic (1st level only) with the limitation of missing PA for all deeper logic. But this may be acceptable given all the full PA-GL DMS coverage of all AMS co-simulation test-suites at insignificant additional effort or run time.

b) Use of out of module references (OOMR): This is commonly done to enable assertions on TL net that is internal to an analog module. A practical solution for this would be to use respective access functions or built in tasks (like Cadence's *CDS_GET_ANALOG_VALUE*, Synopsys's *$SNPS_GET_VOLT* and euivalent constructs) that avoid CM insertion on OOMRs. Use of embedded PSL assertions in Spectre netlist or Verilog-A/VAMS based self-checking concepts would also help in this regard [10][11][12].

3) Superiority of SS CM over other existing solutions

Static CMs require significant verification overhead to get some supply dependency while it can never comprehend dynamic sensitivity. Supply inherited CMs require significant verification overhead (effort in setup) to enable its use in a power managed design, without formal ways to address the gaps between intent and implementation. Since the supply sensitivity information is usually available at design (IP) level and SS CM provides a practical way to include them at right stage, it frees up the verification stage from any additional effort and results in a seamless and clean verification solution with no additional efforts / steps that may cause quality closure concerns. Thus SS CM is found to be superior and most suited for seamless AMS co-simulation.

4) Performance impact and scope for improvements

Actual placements of CMs depend on the placement of SS constructs and the depth of design hierarchy. This will result in exorbitant amount of CMs are placed in designs with deeper hierarchy with SS constructs in the leaf cells, for example with GL abstraction for digital portion. Even for a single signal sensitive to a given supply fanning out to multiple loads, the SS CM will be placed at each load. In addition, the way in which heterogeneity is achieved will also have additional CMs in the supply path. Especially for a real signal sensitive to a supply defined as electrical in the leaf cell there may be an E2R CM needed on the supply path. For a GL abstraction this can also result in an insertion of huge number of CMs. These together can cause severe performance degradation in terms of longer simulation run time. This may be overcome with enhancements to discipline resolution algorithm for merging the fan out nodes both in signal and supply paths, which are not yet available in any commercial EDA vendor tool flows. While simple fan out optimization can handle E2L paths, L2E path optimization is not straight forward which is illustrated in Figure 7. The optimization should also take into account the actual power supply connectivity.

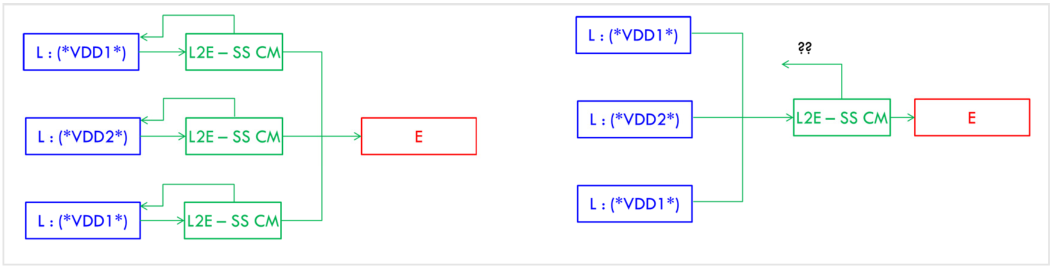

Figure 7. Fanout signal causing sub-optimal SS CM insertion

D. Compararative summary of CMs

A brief, concise summary of the comparison of the different CM flavours discussed is presented in Table I.

Table I. Comparative summary of different types of CMs

S. No	*Requirements*	*Static CM*	*Supply inherited CM*	*Supply sensitive CM*	*Power aware (CPF/UPF)*
1	Power awareness	Inherently not supported.	Dynamic with global.	Dynamic with actual.	Dynamic with global.
2	Multiple voltage and power domains	Static support using custom setup requiring high effort.	Dynamic, not connectivity aware.	Dynamic & connectivity aware.	Dynamic, not connectivity aware.
3	Setup effort	Negligible for designs with single supply. High for power managed (PM) designs.	High in defining disciplines. Custom automation possible.	Low. Initial effort spread across each design element.	None. Reuses existing PA RTL simulation flow setup.
4	GL simulation concerns	None.	None.	Performance issue due to CM blow-up.	CPF may be stripped which has SS information.
5	Migration of legacy designs	Low effort. Same as new design.	Moderate effort. Same as new design.	High effort. Custom automation possible.	Very high effort, if no CPF/UPF available.
6	Recommended design flavours & stage	Non-PM designs.	PM designs at mature stage. Designs not having SS information.	PM design at early stage. Designs having SS information.	PM designs with CPF/UPF, with no GL AMS requirement.
7	Sandwitch AMS configuration	No additional effort.	Additional discipline handling required.	No additional effort.	Not evaluated.
8	Improvements needed	String parameter support for *vsup*.	None.	Heterogeneity & Hierarchical CM optimisation.	GL-AMS simulation support.

IV. CONCLUSIONS

In this paper, the authors have presented the necessity of CMs (IEs) for AMS co-simulation, different flavours of these CMs, their relative strengths and weaknesses. They have established the necessity of supply sensitivity and explained different ways of achieving the same. While many of these techniques are known and existing, their applicability to SoC level AMS co-simulation is not straight forward. The authors have presented practical automation methods to make the use of different CM flavours and enable supply sensitivity easier. The superiority of SS CM is established in this paper in the context of power management and power awareness support. Practical methods to intercept SS CM in new designs and migrating existing designs are discussed.

ACKNOWLEDGMENT

We are thankful to Jagdish C Rao, Nikhil Chandrakant Sangani of Texas Instruments (TI) for their support and motivation. We acknowledge the specific technical contributions and support of Badrinarayanan Zanwar, Bhanuprakash R and Michael Womac of Cadence Design Systems (CDS) towards the studies of several flavours of CMs for AMS co-simulation. We are thankful to Abhijeet Kolpekwar, Chong Chi, Ron Vogelsong, Dan Cline, Michael Dolan, Andre Baguenier and Vishwajeet Betai of CDS for their untiring support on AMS co-simulation methodology, technical discussions, brainstorming and formal tool support. We thank the contributions of Kalyan Ram Hampapuram, Aswani Kumar Golla, and Penchalkumar Gajula of TI for their support on several aspects of backend physical design flow and tools.

REFERENCES

[1] Buss, D.D., Chatterjee, A., Efland, T.R., Evans, B., Goodpaster, H.D., Haroun, B.S., Hellums, J.R., Krenik, W.R., Morton, A., Schichijo, H., Tsai, C.-Y., Vrotsos, T.R. 2000. DSP & analog SOC integration in the Internet era. IEEE Emerging Tech. Symp.: Broadband, Wireless Internet Access. Apr. 2000, 1-5. DOI=http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=916520

[2] Moloney, A. 2006. Power Supply Management – Principles, Problems, and Parts. Analog Dialogue, Vol 40. Analog Devices Inc. May 2006. DOI=http://www.analog.com/library/analogdialogue/archives/40-05/power.pdf

[3] Manninger, M. 2007. Power Management for Portable Devices. Proc. of 33rd ESSIRC. 2007, 167-173.
DOI=http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4430272

[4] Poluri, B., Lele, A., Kedia, R., Balasubramanian, L. 2014. Unified flow using CPF derived Inherited Connect Modules for Digital and Analog Mixed Signal verification. Designer Track, 51st DAC. 2014.

[5] Accellera Systems Initiative. 2014. Verilog-AMS language reference manual, V2.4. 2014.
DOI=http://www.accellera.org/downloads/standards/v-ams/VAMS-LRM-2-4.pdf

[6] AMS Supply-Sensitive Connect Modules Application Note, Product Version 5.3, July 2004, Cadence.
DOI=http://sourcelink.cadence.com/docs/files/Application_Notes/2007/AMS_Supply_Sensitive_Connect_Modules_Application_Note.pdf

[7] Frey, P., O'Riordan, D. 2000. Verilog-AMS: Mixed-Signal Simulation and Cross Domain Connect Modules. Proc. of IEEE/ACM Intl. Workshop on BMAS. 2000, 103-108. DOI=http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=888372

[8] Zinke, O. 2002. Bi-directional mixed signal connection modules for automatic insertion. Proc. of IEEE/ACM Intl. Workshop on BMAS. 2002, 102-107. DOI=http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1291066

[9] Siddiqui, S. 2014. Efficient methods for analog mixed signal verification: Efficient and fully automated connect module handling. Bachelors Intern thesis. IIT Madras, Chennai & Texas Instruments (India) Pvt. Ltd., Bangalore, India. July 2014.

[10] Balasubramanian, L., Matta, H., Jeevanandam, S. 2012. A user friendly and intuitive infrastructure that enables efficient analog mixed-signal verification using assertions based self checking analog designs. CDN Live India, Bangalore, India. Oct. 2012. DOI=http://www.cadence.com/cdnlive/library/documents/2012/IN/3.3_TI_ABV.pdf

[11] Sundar, P., Balasubramanian, L., Tare, S., Chandramohan, S., Jiang, C. 2011. Enabling Efficient AMS Co-simulation of Mixed-signal SoC with Analog and Power Management Integration. SNUG India, Bangalore, India. Jun. 2011. DOI=https://www.synopsys.com/news/pubs/snug/india2011/FC3.3_TI_paper.pdf

[12] Sundar, P., Balasubramanian, L., Fischer, T.W. 2011. Assertion based self-checking of analog circuits for circuit verification and model validation in SPICE and co-simulation environments. Accellera DV Con, San Jose, USA. Mar. 2011. DOI=http://www.dvcon.org/2011/proceedings/papers/02_2.pdf