Power Aware Models:

Overcoming barriers in Power Aware Simulation

Mohit Jain, STMicroelectronics (mohit-jain.crd@st.com)

Amit Singh, STMicroelectronics (amit-ftm.singh@st.com)

J.S.S.S.Bharath, STMicroelectronics (jsss.bharath@st.com)

Amit Srivastava, Mentor Graphics (amit_srivastava@mentor.com)

Bharti Jain, Mentor Graphics (bharti_jain@mentor.com)

***Abstract*— The increasing presence of power management at earlier stages in the design cycle and its effect on functionality demands simulations at RTL stage to be power aware to ensure correctness of the power management. The presence of variety of pre-implemented complex IPs (hard macros) in a SoC often lack appropriate power aware simulation models, forcing users to rely on gate level or analog simulations for complete power aware verification. This limits the effectiveness of power aware simulations at RTL. Moreover, the presence of power management may introduce new IPs that act as power sources, supplying power to the supply network, additionally requiring power aware simulation models. In this paper, we will demonstrate how to extend existing models with power aware behavior and also their integration in the UPF environment for more accurate power aware simulations at RTL.**

Keywords—Power Aware Simulation; Hard Macros; Analog Macros; UPF; Liberty; Voltage Regulators; Power Sources

I. Introduction

Energy efficient systems depend on sophisticated power management strategies to reduce the static and dynamic power consumption. In order to achieve maximum gains the consideration of power management starts as early as in the system design phase itself. These strategies result in complex power management architecture in the design, which if not verified correctly, can result in potential bugs in the system. These bugs are difficult to detect in traditional simulations when it is done at RTL phase as the power management architecture is still absent in HDL design. Hence, they require more advanced power aware simulations to mimic the effect of power architecture at RTL and expose design issues related to it. At RTL stage, the power management architecture is present in abstract form captured by power intent specification called Unified Power Format (UPF). The UPF is an IEEE 1801 standard [1] which defines commands and semantics to enable early verification of power management. Hence, verification tools, using information from UPF, can perform Power Aware Simulation at RTL phase and detect complex bugs related to power management early in the design cycle thereby reducing the time to market.

The power aware simulation becomes a challenge when it is done for Low Power SoC based designs. These designs integrate a variety of multi-rail hard macros whose supply related behavior has an impact on low power simulations in their own way. Such behavior may not be defined by UPF and needs to be accurately modeled. Hence, power aware simulation requires simulation models embedded with power aware behavior for hard macros. The complexity of behavioral models of hard macro causes problems in extending the traditional behavior models with power aware functionality. Also, limitations of HDL's to capture power management information like power states, voltage information results in loss of significant details required for effective power aware verification. Moreover, adding power aware behavior in the model results in changes to the interface of the model for connecting supplies to the models. Therefore, users need to do some special handling for integration in non-power aware simulations. These challenges force users to defer the power aware simulations to a later stage when the design contains entire supply network implemented. This happens as late as fully PG connected gate level netlist or the spice netlist. The simulations done at that stage are extremely slow and any issues are hard to debug. This forces users to make a tradeoff with the time and verification quality.

The paper describes the concept of power aware models for hard macros and usage of these models at RTL verification phase for Power Aware Simulations. The power aware simulations are more comprehensive for verifying the power behavior at SoC level. The paper will demonstrate how to use existing non-PA models provided by IP vendors for use in power aware simulation. The paper will also provide guidelines with the help of real application scenarios, which can be used to construct more accurate power aware models at RTL using standard UPF-HDL interface.

II. Power Aware Simulation with UPF

The emergence of UPF has empowered verification tools to perform power aware simulations early in the design stage even before the implementation process has taken place. In a UPF based flow, a designer expresses the power intent in a side file, called UPF file, which is used in conjunction with the RTL design by the verification tools. The UPF is designed in such a way that the same UPF can be used by implementation tools to insert the power management architecture in the design.

The UPF standard defines commands to create abstract supply network. Any UPF compliant simulation tool can interpret the commands and create a virtual supply network and automatically connect them to logic requiring supply. The UPF has commands which enable the placement of power management cells like isolation, level shifter and repeaters at appropriate places in the user design. There are commands that can convert regular register present in the design to retention registers. The standard also defines the default simulation behavior of these special cells which can be used to simulate them even before they get implemented. Similarly, UPF enables connection of supplies to all the logic present in the design which can also be used to simulate corruption behavior. These semantics allow an early verification of the power management and its interactions with the design functionality. Hence, exposing hard to find bugs early in the design flow thereby reducing the debugging time and cost.

III. Challenges with Power Aware Simulation of SoCs

Low Power SOCs integrate a variety of multi-rail hard macros like IO pads which propagate the supply and signals from analog to digital world. These IO PADs are not actually part of the digital design. They are analog / mixed signal components and the digital supplies are not available to them. To add to the complexity, the analog supplies driving them may not be available during RTL and would be added further down the implementation process.

Multi-rail memories, PLLs, sensors etc. have dedicated supply for various blocks to allow power saving during inactive mode and faster wake-up during power up. The power aware semantics of these blocks may not be defined in the UPF. There may be various reasons for this like legacy models having only HDL description or inability of UPF to model complex sequencing behaviors.

In addition, there are mixed signal IPs such as USB, DDR, DPHY, etc. These IPs comprise of analog macros, digital controllers and full custom blocks resulting in multi-layer hierarchies and pose a different set of challenges in managing its power behavior. They may or may not have supply sequencing which needs to be properly modelled. Some of its logic may be powered by an internal supply derived from some external supply and may possibly depend upon some control inputs which needs to be modelled properly. All these IPs support low power modes and need supply sequencing in a specific order which needs to be verified by SOC designers.

Power sources require some special attention as they generate supplies which are significant only in a power aware simulation. They are generally modeled in HDL for providing stimulus and driving the supply network. The only defines the simulation behavior of power switches. The more advanced macros like voltage regulators are not expressed in UPF. This requires users to provide HDL models for regulators. Also, lack of proper methodology and additional information required by UPF, the integration process becomes cumbersome and suffers in accuracy of the simulation.

Connecting power supplies during integration of various IPs in a SoC has its own set of challenges like connecting supplies in different forms from various sources, verifying proper sequencing, proper synchronization and conversions required for compatibility between various models.

IV. Existing Verification Models and Their Limitations

The simulation of SoC having hard macros at RTL relies on simulation models which describe the behavior. These models are provided by IP providers and are developed in HDL's like Verilog or System Verilog. As an analog behavior is described in a digital language, careful choice between optimism and pessimism is made to make the behavior as close to the design as possible. Depending upon the functionality they are divided into three types:

A. Simulation models without power management behavior (non-PA HDL Model)

The non-PA HDL Model contains basic functionality without power management behavior. This makes them more suited for non-PA simulations and are less complex than their PA counterparts. They are easy to integrate in the design as they contain pins which can be easily connected in the design.

Using non-power aware models for hard macros doesn't go well with a power aware simulation. Since the power aware functionality is missing in these models, they depend on UPF for power aware semantics. However, since the simulation models are typically developed in HDL which uses constructs that are not conducive to synthesis, the default semantics of UPF do not come into play and instead result in inconsistent behavior of hard macro due to corruption of internal variables and states. As a result, users have to use some tool specific extensions to manually disable the UPF semantics on the models which are non-power aware. This enables UPF behavior for the rest of the design but lack of power aware behavior for the IP causes incomplete low power simulation or significant loss of accuracy.

B. Simulation models with power management behavior (PA- HDL Model)

The PA HDL models contain normal design functionality along with power management behavior. The power management architecture results in additional supplies at the interface of the model. In a non-PA simulation, the extra supplies pose problems in integration process at RTL as the supplies are not present in the design. This forces users to do some special handling by connecting dummy supplies with default values.

In a UPF based simulation, the integration process is much cleaner. This is due to the predefined semantics in UPF defines that results in disabling of default UPF behavior when the model contains supplies which is connected by UPF supplies. This helps in avoiding re-implementation of power management. However, it puts the burden on users to ensure that the HDL models of hard macros are accurately modeling the power aware behavior. The complexity of the behavioral models of hard macro causes problems in extending the traditional non-power aware behavior models with power aware functionality. Also, for the analog and mixed signal blocks it is not possible to embed the power ware functionality at RTL stage as the model itself may not be complete.

Moreover, the limitations of HDL to capture crucial power management information like power states, voltage information results in loss of significant details required for effective power aware verification. Also, the HDL has limitations in differentiating between various kinds of supplies like power, ground, pwell, nwell, deeppwell, deepnwell etc. Different supplies impact PA simulation in different ways and hence it is necessary to differentiate and handle them accordingly.

C. Analog Models

The analog models capture the most accurate representation of the hard IP. They are generally written in Spice and hence depend on Spice simulations for verification. The accuracy comes at a great cost, the simulations are extremely slow and any bugs discovered at that stage is costly to debug and fix. The slowness of analog simulation forces users to validate only selected set of protocols. This often results in incomplete verification and increases the risk involved in low power designs. To add to that, there may be combinations of various IPs like soft macros, hard macros, power sources in the SoC and hence the power aware verification models with embedded power aware behavior and soft macros with power aware behavior defined in UPF need to work in conjunction with each other and hence need to be integrated properly together.

All these traditional HDL models lack the necessary information related to power management needed for the UPF based simulation at RTL phase thereby requiring additional information in the form of Power Aware Models.

V. Power Aware Models

Power Aware Models overcome the barriers in low power simulation by complementing the existing verification models with power management information at RTL to enable more accurate power aware simulation. This is accomplished by combining the behavior modeling capability of HDL languages and abstract power intent representation capability of UPF or liberty. Usage of power aware models has two aspects:

A. Components of power aware models

The Power aware model is a combination of two components, the power management interface and the power management behavior.

1) Power Management Interface

The power management interface contains information related to the power management architecture at the interface of the IP. This information is typically not present in UPF or Liberty instead of HDL descriptions. The interface information is used by the top level UPF to make proper integration of the hard IP.

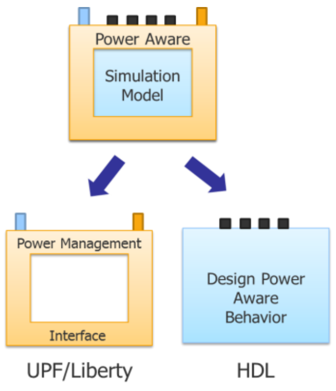


Figure 1: Power Aware Model components

The following information about power management interface is necessary for the IP integration:

a) Related Supplies

The IP integrator needs to ensure whether there is a need to have isolation/level shifting cells at the boundary of the IP. Since, the hard IPs are typically considered as black box and lack any RTL description. The integration process depends on pin level information about driver and receiver supplies with the IP. This information is specified by related supply attributes defined on the pins of the hard IP.

b) Supply Information

The supply pins at the IP boundary needs to be connected to correct supplies. The connection requires information about the pg_type property of the pin. The pg_type attribute indicates the function of the supply pin as to how it powers the regions within the IP. This information can help the tool to make appropriate translation of supply values coming from UPF.

c) Power states

Any low power IP may operate in different power modes. These power modes are typically represented in the form of power state information in the power intent specification files. The knowledge of power states can help the IP integrator check the integration doesn't contradict IP's power modes.

d) Interface Protection Cells

Sometimes, the IPs already contains the power management cells (e.g. isolation/level shifters) inside them at the interface. This information needs to be conveyed to the IP integrator so that the integration process doesn't result in redundant placement of these cells.
The information about power management interface is expressed in any of the following formats:

a) UPF

The UPF 2.0 standard allows capturing the details of power management interface in the form of UPF files. This is demonstrated in the DVCon 2012 paper, titled: "Low Power SoC Verification: IP Reuse and Hierarchical Composition using UPF"[2]. The latest UPF 2.1 standard formalized the concept of Power Models by introducing new commands dedicated to model the interface information, as described in DVCon 2014 paper, titled: "Stepping into UPF 2.1 world: Easy solution to complex Power Aware Verification" [3]. The UPF 2.1 introduced two new commands begin_power_model and end_power_model to encapsulate a set of UPF commands in the form of power models.

The following UPF commands can be used to capture the details of power management interface:

Power Management Information	**UPF Commands**	**Example**

Related Supply	`set_port_attributes`
	`-related_power_net`
	`-related_ground_net`
	`-related_bias_pin`
	
	`set_port_attributes`
	`-driver_supply_set`
	`-receiver_supply_set`
		`set_port_attributes hardIPInst \`
		`-elements { datain } \`
		`-related_power_net VDD \`
		`-related_ground_net VSS`

Supply Information	`create_supply_port`
	`create_supply_set`
		`create_supply_port VDD`
		`create_supply_set IP_SS \`
		`-function { power VDD } …`

Power States	`add_power_state`
	`create_pst`
	`add_pst_state`
		`add_power_state PD_HardIP \`
		`-state ON { \`
		`-logic_expr { \`
		` PD_HardIP.primary == ON \`
		`}\`
		`}`

Interface Protection cells	`set_isolation`
	`set_level_shifter`
	`set_repeater`
		`set_isolation PD_HardIP \`
		`-applies_to outputs`

Example

```
#---------------
#hard_ip.upf
#---------------

# Power Model for Hard IP
begin_power_model hardMacro

#Power Domains for Hard IP
create_power_domain pd_hardIP \
  -include_scope \
  -supply { backup_ssh } \
  -supply { primary }

#Related Supply Constraints
set_port_attributes -domain pd_hardIP \
  -applies_to outputs \
  -driver_supply pd_hardIP.primary
set_port_attributes -ports portA \
  -driver_supply pd_hardIP.backup_ssh
set_port_attributes -domain pd_hardIP \
  -applies_to inputs \
  -receiver_supply pd_hardIP.primary

#Retention Constraints
set_retention_elements critical_regs \
  -elements { reg_a reg_b } \
  -retention_purpose required

#Internal switchable supply
create_power_switch
# Isolation/level shifter and retention cells
set_isolation ...
set_level_shifter ...
set_retention ...

# System states for hard IP
add_power_state pd_hardIP ...

# ... Other Constraints ...

end_power_model
```

Although UPF provides more flexibility and compatibility in representing the power interface, the tools are still in the process of developing some of these capabilities. Hence, the IP providers are slow in adopting UPF 2.1 commands in representing the power management interface.

b) Liberty

The information about interface of the power management can also be represented in the form of liberty attributes. There are special attributes in liberty which can be used by tools to identify the information about power management interface. The support for liberty is not as comprehensive as UPF but IP providers are already providing the necessary information along with the IP for use with other static tools. So, there is less overhead involved in reusing the information already available for simulation at RTL. However, this requires tools to understand and interpret the liberty specification along with making liberty files available at RTL stage.

Power Management Information	Liberty Attributes	Example
Related Supply	`related_power_pin` `related_ground_pin`	`pin(IN) { direction : input; related_power_pin : VDD; related_ground_pin : VSS; … }`
Supply Pins	`pg_pin` `pg_type`	`pg_pin(VDD) { pg_type : primary_power; … }`
Power States	Not Available	Not Available
Interface Protection cells	`is_isolated`	`pin (OUT) { is_isolated : true ;`

Example:

```
#---------------
#hard_ip.lib
#---------------

# Liberty Model for Hard IP

cell (Hard_IP) {
    is_macro_cell : true;
    pg_pin(VDD) {
        pg_type : primary_power;
        direction : input;
    }
    pg_pin(TVDD) {
        pg_type : backup_power;
        direction : input;
    }
    pg_pin (VSS) {
        pg_type : primary_ground;
        direction : input;
    }
    pin (IN) {
         direction : input;
         related_power_pin : VDD;
         related_ground_pin : VSS;
    }
    pin (OUT) {
         direction : output;
         power_down_function : "!VDD + !TVDD + VSS";
         related_power_pin : VDD;
         related_ground_pin : VSS;
         is_isolated : true;

    }
}
```

2) Power Management Behavior

The power management behavior is captured in HDL descriptions and often involves reuse of traditional HDL models that are shipped with the IP.

a) Non-PA Behavioral Model

The non-pa behavioral model can automatically be extended to behavioral model by using related supply information to corrupt the boundary pins. This provides approximate power aware behavior for IPs which have only basic power management capabilities without any retention capability. If the IP has more advanced power management and involves complex sequencing protocols then there is a need to have a more accurate power aware HDL model.

b) Allpins Model

Allpins models have the power pins at the port level and the effect of the supplies on the outputs programmed. These are traditional power aware HDL models that are used at Gate level or a later stage when netlist contains full connection of supplies.

c) PA Behavioral Model

The power aware behavioral model is developed in such a way that, the power behavior is inside the model and will be visible when the UPF connections are made. The PA behavioral model has the following characteristics

- The power pins are not present in the portlist.
- The input power supplies are declared as registers and are initialized to their definitive logical values.
- If the IP has any output supplies, they are declared as registers / wires and contain the functional intent in them.

Generally a PA behavioral model is derived from an allpins model by removing the supplies from the port level and declaring them as reg / wire.

d) Voltage-Aware PA Behavioral Model

The voltage aware PA behavioral model is similar to PA behavioral model but instead of using normal wires to represent supply, they use UPF's supply_net_type objects to model supplies. This enables them to have knowledge of voltage information in the supplies. There is a standard UPF HDL interface available via SV and VHDL package functions which can be used to create models and apply voltage values on them. These models provide more accurate representation of power management architecture, especially modeling the power sources. Use of this model allows PA Simulation to discover bugs related to multiple voltages early in the design flow. However, the voltage aware models are still in infancy and not available for current hard IPs. It is still very early to witness any live examples of these models. The potential of capturing additional voltage representation puts them at very high in the list of future development of power aware models.

B. Integration in SoC Environment

The integration of power aware models is straightforward and happens automatically without much interference of the user. The user needs to ensure that both components of the power model are available to the verification tool and the tool understands the UPF or liberty attributes specific to low power.

VI. POWER AWARE MODEL EXAMPLES

A. Analog Macros

Some Mixed signal IP’s viz. PLL[4], Oscillators etc. have separate analog and digital supplies. These supplies could be individually switchable. Each supply may impact the IP behavior in different ways, the example chosen is a PLL. A Phase Locked Loop is a mixed signal circuit capable of generating a stable and reliable high frequency clock from a low frequency jittery clock, now such an analog circuit requires analog power supplies. There are some approximations done while modeling analog supplies i.e. when the analog supply has ramped to the level then it is depicted as a logic H, when the analog supply has reached its ground level then it is depicted as a logic L.

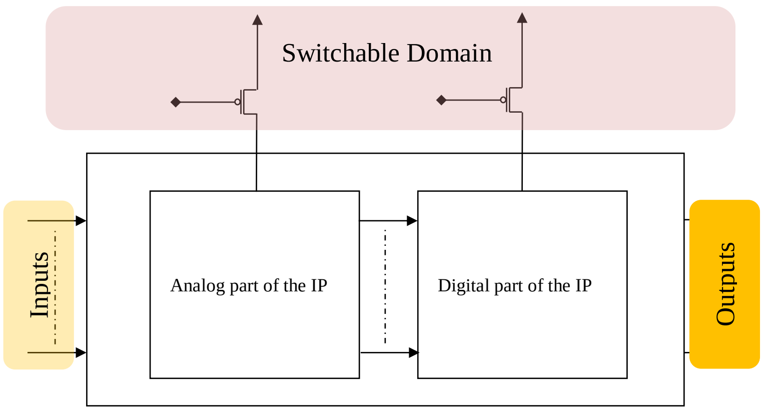


Figure 2: A typical analog macro with some analog and digital part together and interacting in one IP interface

1) Using non-pa behavioral model and power intent from liberty

a) Power Management Interface

The power supplies to these IP’s are purely driven by externally switchable supplies and no other functionality is present inside the model, hence these supplies become part of the model interface and standard UPF connection guidelines are followed which is depicted in the examples and the results

b) Power Management Behavior

The impact of power supplies is captured in the liberty files through the power_down_function attribute. The HDL model does not contain any power management information and has only basic macro functionality.

PA Information in Liberty

```
pg_pin(avdd) {
    pg_type : primary_power;
}
pg_pin(dvdd) {
    pg_type : primary_power;
}
pin(REFANA) {
    …
    power_down_function :”!dvdd+!advv+dvss+avss”;
    …
}
```

UPF Connections

```
connect_supply_net avdd_n \
  -ports { hm_inst/avdd }
connect_supply_net dvdd_nm \
  -ports { hm_inst/dvdd }
```

Non-PA Behavioral Model

```
module ana_mac( … ip1, … );
…
  //valid macro functionality.
  //Non power aware model
….
endmodule
```

Figure3: Power management interface and behavior

c) Simulation Results

As can be seen from the figure 4, the IP recovers as soon as the supplies are recovered. The major limitation of this approach is that, the actual design behavior is not seen and hence not verified and could be a potential cause for failures

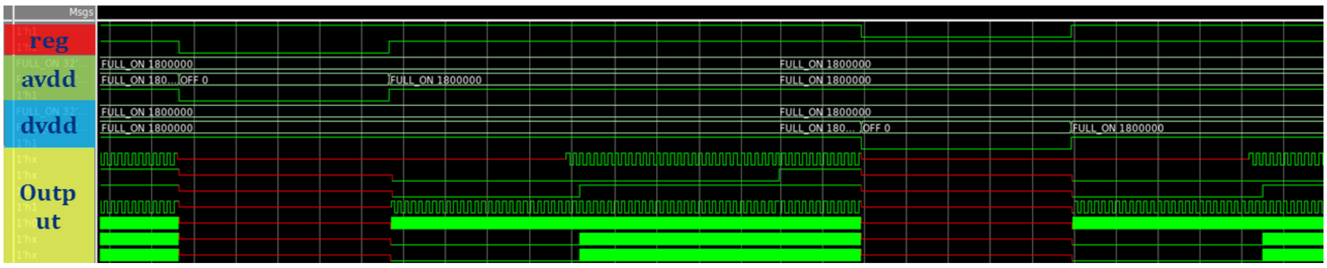


Figure 4: Simulation with non-power aware behavioral model, with power intent taken from liberty

2) Using power aware behavioral model and liberty for interface

a) Power Management Interface

The power management interface is similar to case I, except that it may not contain power_down_function attribute.

b) Power Management Behavior

These IP’s have the analog and digital blocks interacting with each other and hence the behavior has a strong dependency on supply (both analog and digital) availability. In most of the cases, the IP would have functionally correct “outputs” after a protocol is followed, viz. resetting the IP after supply availability.

PA Information in Liberty

```
pg_pin(avdd) {
    pg_type : primary_power;
}
pg_pin(dvdd) {
    pg_type : primary_power;
}
```

UPF Connections

```
connect_supply_net avdd_n \
  -ports { hm_inst/avdd }
connect_supply_net dvdd_nm \
  -ports { hm_inst/dvdd }
```

PA Behavioral Model

```
module ana_mac( … ip1, … );
…
reg avdd, dvdd, avss, dvss;
always @(avdd or dvdd or avss or dvss)
begin
 if(avdd === 1’b1 && dvdd === 1’b1 && avss
=== 1’b0 && dvss === 1’b0) begin
  //valid macro functionality…
 end
 else begin
  // Invalid supplies, outputs are
corrupted or pulled L..
 end
end
```

Figure 5: Power management interface for an IP with analog and digital supplies

c) Simulation results

The result shown depicts a behavior in which any of the digital or analog supply if pulled to L, all the outputs are pulled L, but after revamping the supplies to their credible levels, the startup protocol needs to be re-initiated

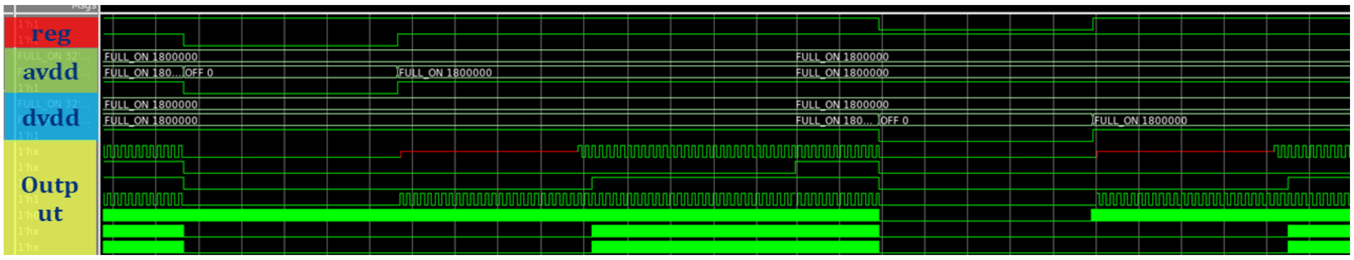


Figure 6: Simulation with power aware behavioral model

B. *Macros with Internal Switching*

Some of the IP's have a power switch embedded inside the IP boundary for powering off some of the logic. Such architectures are required for a granular control of the power supplied to these IP's. The input supply to this switch is typically driven from a switchable supply. In some cases there could be some logic that is outside the IP but having dependency on the outputs generated by the IP and hence would be working on the same voltage level as the one which is internally switched. As a generic guideline the output of the switch of the IP (i.e. internally switched supply) and the supply driving (i.e. externally switched supply) the logic outside the IP are connected together. It is important that there is no misalignment in the ramp up and ramp down of the macro internally switchable supply port and the UPF supply net with which it is shared.

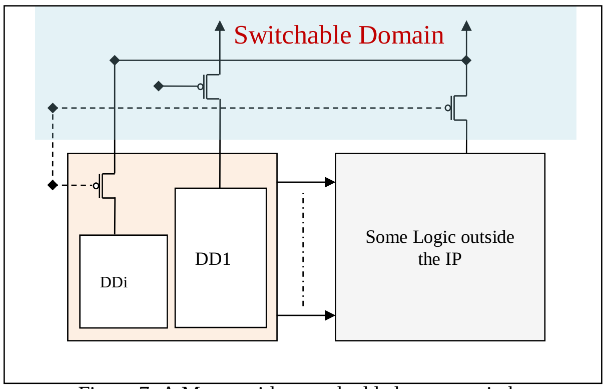


Figure 7: A Macro with an embedded power switch

1) Power Management Interface and behavior

The liberty model requires some attributes to indicate that the supply under consideration is a switchable supply. The supplies needs to be declared as a register and initialized to their respective values so that the model works fine in the context of non-power aware simulations. In the figure 8, the example depicts a macro with switch embedded in it.

PA Information in Liberty

```
pg_pin(vdd) {
    pg_type : primary_power;
}
pg_pin(vdd1) {
    pg_type : primary_power;
}
pg_pin(vddi) {
    pg_type : internal_power;
    direction : internal;
    switch_function : ctrl;
    pg_function : vdd;
}
pin(ctrl) {
    direction : input;
    switch_pin : true;
}
pin(outp) {
    power_down_function : "!vddi +
vss";
```

PA Behavioral Model

```
module mac_int_sw( … sw_ctrl … );
…
input sw_ctrl;
reg vdd, vdd1, vddi;
wire w_vddi;
always @(w_vddi) vddi = w_vddi;
initial vdd = 1'b1; vdd1 = 1'b1;
assign w_vddi = sw_ctrl === 1'b1 ?
  vdd : sw_ctrl === 1'b0 ? 1'b0 : 1'bx;
... Macro Functionality ...
endmodule
```

UPF Connections

```
connect_supply_net vdd_n_lv \
  -ports { hm_inst/vdd }
connect_supply_net gnd_snet \
  -ports { hm_inst/vss }
```

Figure 8: Liberty, verilog model and typical UPF connections for a macro with an embedded switch

2) Simulation Results

Figure 9 shows the UPF architecture built around a macro with internal (power) switch, it also shows the connectivity of the external switch and internal switch (which needs to be synchronous) and their associated power supplies.

The example below describes a scenario in which the switch that is internally switched and the switch that is present outside the IP interface are connected together and hence the switch controls are to be synchronized for proper functioning.

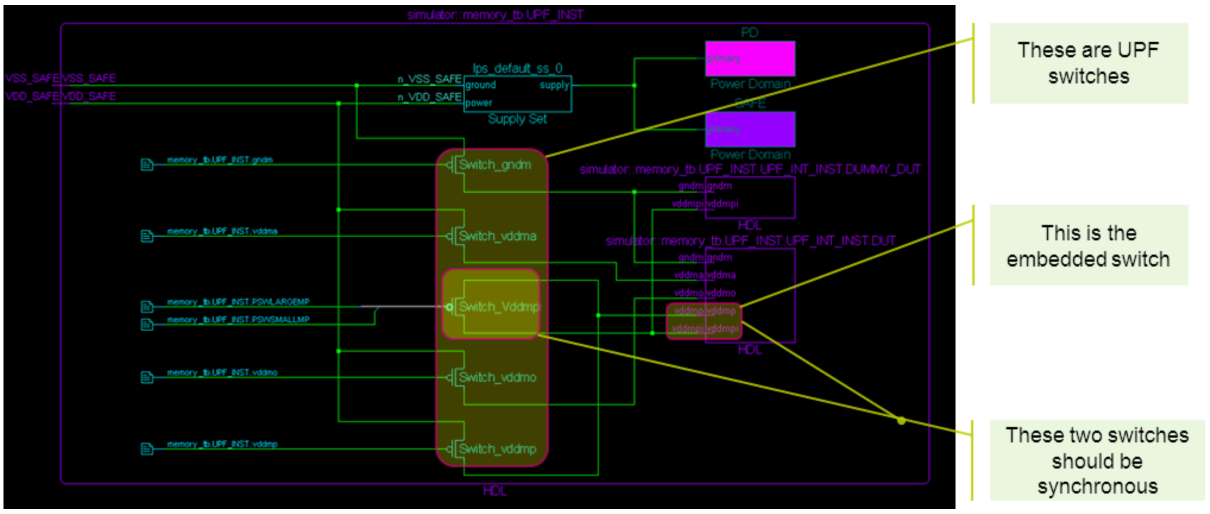
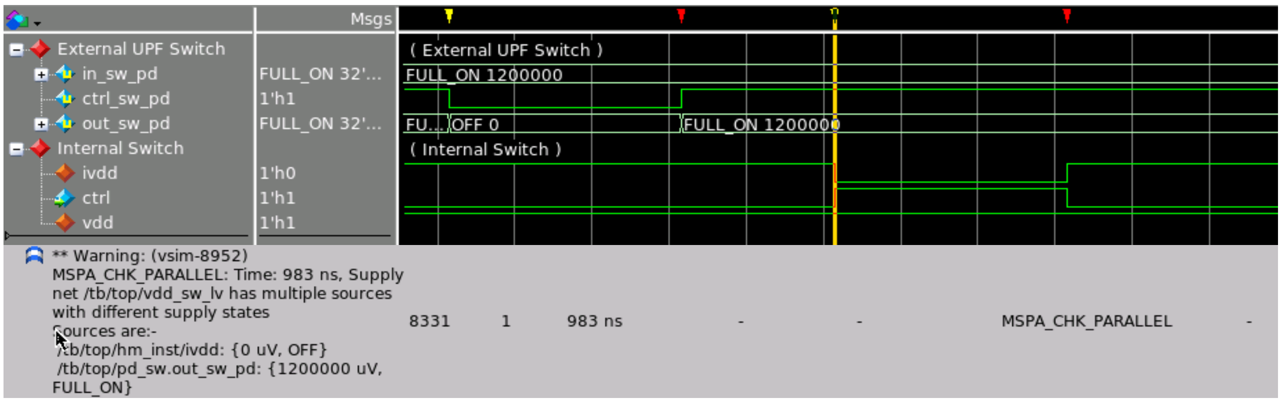


Figure 9: Typical scenario in which the embedded switch and the UPF switch are supplied with a yet another switchable supply

Figure 10 shows the case in which the controls to both the switches (internal and external) are out of sync, the simulator issues a warning indicating that a UPF net is driven by multiples sources with different states. When the switches are synchronous the UPF net will be driven to a desired state.



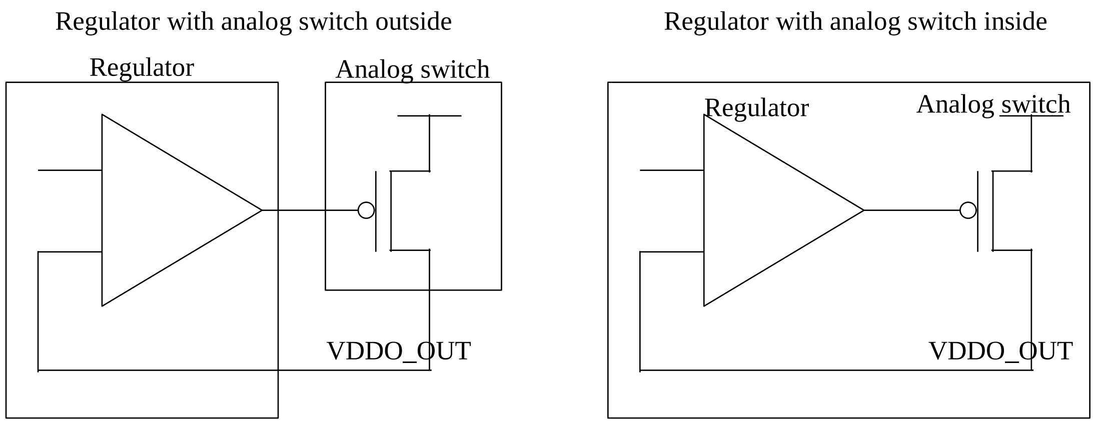
	Msgs
External UPF Switch	
in_sw_pd	FULL_ON 32'...
ctrl_sw_pd	1'h1
out_sw_pd	FULL_ON 32'...
Internal Switch	
ivdd	1'h0
ctrl	1'h1
vdd	1'h1

```
** Warning: (vsim-8952)
MSPA_CHK_PARALLEL: Time: 983 ns, Supply
net /tb/top/vdd_sw_lv has multiple sources
with different supply states
Sources are:-
 /tb/top/hm_inst/ivdd: {0 uV, OFF}
 /tb/top/pd_sw.out_sw_pd: {1200000 uV,
FULL_ON}                                   8331    1    983 ns    -    -    MSPA_CHK_PARALLEL    -
```

Figure 10: Example depicting a scenario in which the switches are not synchronized for which the tool logs a synchronization error

C. Voltage Regulator

A voltage regulator [5] is an analog circuit which takes a raw input supply and generates a regulated and monitored supply with proper voltage level and drive capability. A voltage regulator typically consists of a control block, precision references, comparator and an analog switch. Typical function of the regulator is that, the comparator generated the difference between the output supply (or a part of it) with a reference supply, this difference signal is translated to the switch control to appropriately turn on or off the analog swtich. The regulated supplies of a voltage regulator are typically driven by analog switch (with the control of the switch governed by the regulator), with their presence inside / outside the IP boundary. The regulated supply can be bypassed by an external source. These externally driven supplies are required (to be sensed) by the regulator for sanity checks viz. threshold etc. The behavioral models of these macros are Power Aware and support both the modes of operation. Specific Liberty attributes required for targeted functionality



Regulator with analog switch outside
Regulator
Analog switch
VDDO_OUT
Regulator with analog switch inside
Regulator
Analog switch
VDDO_OUT

Figure 11: Voltage regulator with the analog switch presence inside and outside IP interface

1) PowerManagement Interface

The UPF net connected to the output supply should have a resolution of ONE_HOT because the UPF net will be driven exclusively by the voltage regulator or from external i.e from the UPF environment. The UPF net connecting the output of analog switch and the regulator should have a resolution of one_hot because the UPF net will be driven exclusively driven by the analog switch or from external i.e from the UPF environment

As described in figure 12, if the power block outside the regulator interface, the supply port is analogous to any input supply i.e this port will always be in a reception mode. If the power block inside the regulator interface, the supply port is a derived supply and hence is an internal_power, as it can drive or can be driven the direction is inout.

PA Information in Liberty for VREG

```
is_macro_cell : true;
pg_pin(vdd_sw_outside){
   voltage_name : "vdd_switch_outside";
   pg_type      : primary_power;
}
pg_pin(vdd_sw_inside){
   voltage_name : "vdd_switch_inside";
   pg_type      : "internal_power";
   direction    : inout;
}
```

UPF Connections (Analog Switch Inside)

```
create_supply_net net_sw_inside
-domain vreg_domain -resolve one_hot

connect_supply_net net_sw_inside
-ports vreg_inst/vdd_sw_inside
```

UPF Connections (Analog Switch Outside)

```
create_supply_net net_sw_outside
-domain vreg_domain -resolve one_hot
connect_supply_net net_sw_outside
-ports vreg_inst/vdd_sw_outside
connect_supply_net net_sw_outside
-ports ana_mac_sw_inst/vdd_driver
```

Figure 12: Power management interface for a regulator with analog switch inside or outside the IP interface

2) Power Management behavior

The PA (behavioral) model of the regulator is developed depending on the architecture of the regulator as described below:

a) Voltage regulators with Analog switch outside

In this type of regulator, the regulated supply port is driven from the outside in all modes viz. regulation or bypass. The value at the output supply port is required for the regulator as a feedback for controlling the analog switch. This value is also required for status signal generation which indicates that the output voltage level is within the supported specification. To cater this requirement the PA (behavioral) should be capable of receiving a value from the outside and hence a 1'bz is driven on to the supply port to enable the external driving to take precedence.

PA Behavioral Model

```
assign VGATE = ALLCONDITIONS_GOOD ? REGULATION_MODE ? 1’b1 : BYPASS_MODE ? 1’b0 : 1’bx :
1’bx : 1’bx;
assign VDDO_OUT = ALLCONDITIONS_GOOD ? REGULATION_OR_BYPASS_MODE ? 1’bz : 1’bx : 1’bx;
```

Figure 13: Verilog code for a regulated supply, with the analog switch outside the IP interface

b) Voltage regulators with analog switch inside

In this type of regulator, the regulated supply port is driven from the IP in regulation mode and driven from outside in bypass mode. The value at the output supply port is required in bypass mode only for status signal generation which indicates that the output voltage level is within the supported specification. To cater this requirement the PA (behavioral) should be capable of receiving a value from the outside in bypass mode and hence a 1’bz is driven on to the supply port to enable the external driving to take precedence.

In regulation mode 1’b1 is driven on to the supply port indicating a valid supply availability.

PA Behavioral Model

```
assign VGATE = ALLCONDITIONS_GOOD ? REGULATION_MODE ? 1’b1 : BYPASS_MODE ? 1’b0 : 1’bx :
1’bx : 1’bx;
assign VDDO_OUT = ALLCONDITIONS_GOOD ? REGULATION_MODE ? 1’b1 : BYPASS_MODE ? 1’bz :
1’bx : 1’bx : 1’bx
```

Figure 14: Verilog code for a regulated supply, with the analog switch inside the IP interface

3) Simulation Results

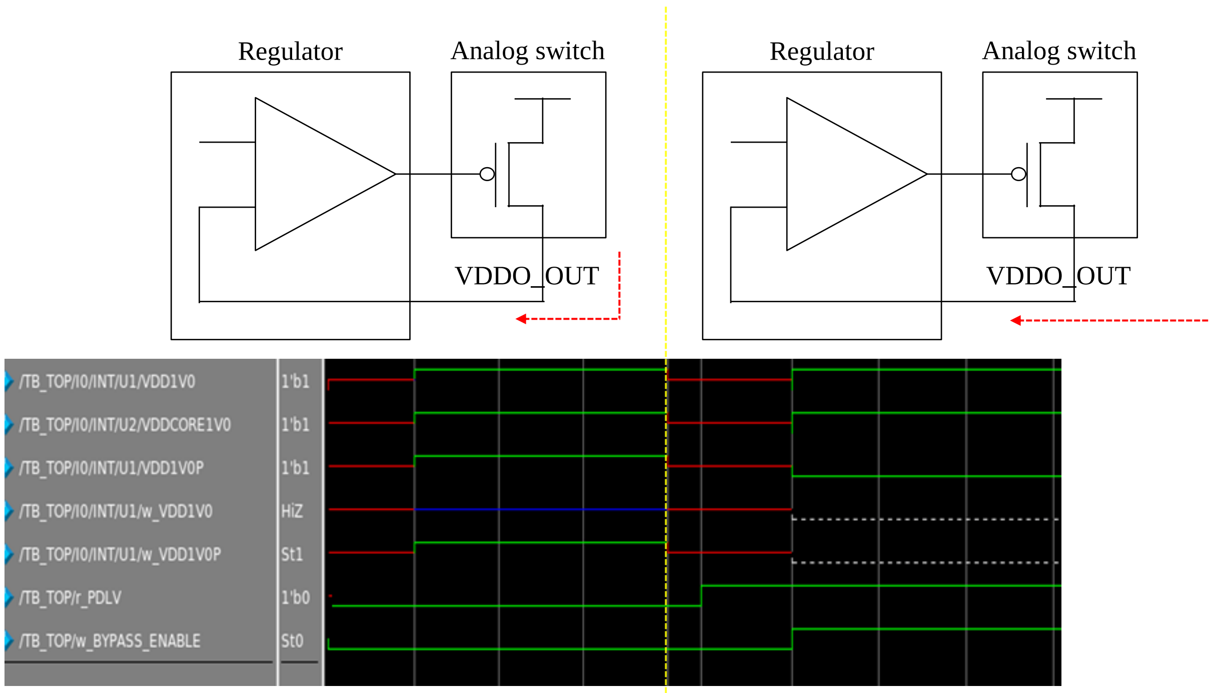


Figure15: Simulation results for voltage regulator with analog switch outside the IP interface

In the figure 15, w_VDD1V0 is indicative of what is being driven from the behavioral, as can be seen w_VDD1V0 is driven a Z from the behavioral model, but a H is seen on the output supply port which is driven by the analog switch which is outside the IP.

Similarly, there is another case in which the analog switch presence is inside the IP interface, the results will be similar to what is seen the waveform excepting that everything is happening inside the IP interface.

VII. Conclusion

The power aware models comprising of power management interface and power management behavior enable more accurate power aware simulations at RTL phase. This allows users to catch complicated functional bugs early in the design flow thereby reducing the need for costly re-spins. In this paper, we have demonstrated how to use existing models and convert them to power aware models that can be used at RTL phase for power aware simulations. We have used live examples and existing liberty information for demonstrating the simulation at RTL phase. In future, we plan to explore Voltage Aware models and use of latest UPF commands and methodology for representing power management interface.

References

[1] IEEE Std 1801™-2013 for Design and Verification of Low Power Integrated Circuits. IEEE Computer Society, 29 May 2013.

[2] Amit Srivastava, Rudra Mukherjee, Erich Marschner, Chuck Seeley and Sorin Dobre : "Low Power SoC Verification: IP Reuse and Hierarchical Composition using UPF", DVCon 2012.

[3] Amit Srivastava, Madhur Bhargava : "Stepping into UPF 2.1 world: Easy solution to complex Power Aware Verification", DVCon 2014.

[4] Behzad Razavi, "Monolithic Phase-Locked Loops and Clock Recovery Circuits: Theory and Design"

[5] G. A. Rincon-Mora and P. E. Allen, "Study and design of low drop-out regulators," 1996

[6] G. Eason, B. Noble, and I.N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," Phil. Trans. Roy. Soc. London, vol. A247, pp. 529-551, April 1955. (*references*)