INTERPRETING UPF FOR A MIXED-SIGNAL DESIGN UNDER TEST

Kenneth Bakalar and Eric Jeandeau
Mentor Graphics Corporation

***Abstract*—IEEE Std 1801-2009 (1) defines the Unified Power Format (UPF) for specifying power distribution and control information required by digital RTL and gate models to represent the structure and behavior of designs with active power management. We propose an interpretation of UPF for designs that include analog and mixed signal elements coded in Verilog-AMS, VHDL-AMS, or SPICE. No changes to the UPF syntax or file are required. We offer as proof of concept a complete implementation and a demonstration of its use in a sample case. The implementation consists of a modified elaborator and an extensible set of power-aware interface elements.**

I. INTRODUCTION

The digital register-transfer level (RTL) abstraction incorporates the unstated assumption that sufficient power is always available for the system to operate correctly after simulation time zero. The increasing importance of power consumption in electronic systems invalidates this assumption and creates a need for a model of active power management that supplements the RTL abstraction (2).

UPF provides the necessary notation for modeling active power control of digital RTL designs. In contrast, low-level electrical designs—traditionally coded in SPICE and, more recently, SPICE combined with the mixed-signal languages VHDL-AMS and Verilog-AMS—model the power supply and power control explicitly. Electrical models are fundamentally dependent on appropriate power for correct operation, and in this they are unlike functional models at the RTL.

Two new problems are salient in analog and mixed-signal (AMS) practice that do not arise in homogeneous digital or electrical design hierarchies. The first problem arises at the boundary between abstractions, when a composite hierarchy combines and connects elements from the electrical and digital domains. The second occurs because the power sources and power control defined by UPF that supplement an RTL hierarchy must be interpreted to include the power sources and controls of the AMS hierarchical descendants as well.

To resolve these issues we propose, a method for providing UPF-controlled, SPICE-level power to those AMS instances that require it, and a method for defining signal connect elements (connect modules) that are sensitive to the supply set of the drivers and/or receivers of a port.

The use model for UPF in an AMS design described in the following paragraphs provides a guiding context for the design and implementation we describe in the net section.

We assume the engineer begins with a power-aware design described by a homogeneous digital hierarchy and an accompanying UPF file. An instance of an AMS model (coded in SPICE, Verilog-AMS, or VHDL-AMS) with electrical ports will be substituted for an existing digital instance in the digital design. The AMS model has electrical ports that correspond one-for-one to the ports of the digital instance that will be replaced. The translation of information to and from the electrical ports should be transparent to the instantiating digital circuit. The translation should be sensitive to the supply set of the UPF domains containing the drivers and/or receivers of the digital signals.

The AMS model also has additional electrical ports that supply power to the analog sub-circuit within the model. The analog power supplies connected to the ports must be synchronized with the power state of the UPF power domain of the instance. Alternatively, the model may contain the root supply driver for a power function of the enclosing power domain—that is, the model acts as a producer rather than a consumer of power. In that case, the port must be interpreted as the root of a UPF power net.

II. SIGNAL CONNECT ELEMENTS

Digital nets may be connected to electrical nets in a composite hierarchy. The flow of information across the connection may be digital-to-analog, analog-to-digital, or bidirectional. An additional active element, which we will call a *signal connect element*, must be interposed at every connection between the analog and digital sides.

A. *Classification of signal connect elements*

If the flow of information is from the digital side to the analog side, such an element is termed a *digital-to-analog connect element*. The new element is dynamically driven by the digital signal. The connect element models the digital signal as a voltage and impedance that changes depending on the signal logic value.

If the flow of information is from the analog side to the digital side, then the interposed element is an *analog-to-digital connect element.* It measures some characteristic of the analog side (for example, the voltage at the connected node) and interprets the measured value as a logic value (‘0’, ‘1’, or ‘X’ to represent ‘unknown’), which it then transmits to the digital side.

If the flow of information is bidirectional, then both a source and a sensor must be added. These may be combined in a single element[1].

In the Verilog-AMS modeling language, signal connect elements are called connect modules. In other contexts they are called hyper models, boundary elements, analog-to-digital converters, digital-to-analog converters, and bidirectional converters.

In a UPF regime, the interpretation of, say, an analog voltage as a logic value or, conversely, the interpretation of a logic state as a voltage source must take into account the additional information provided or needed by the UPF concerning the power state of the digital side of the connection. For example, in order to accurately represent the value ‘1’, a digital-to-analog signal connect element might generate a voltage scaled to the difference between the primary power and primary ground of the power domain containing the digital driver. A signal connect element with this additional capacity will be called a *power sensitive* signal connect element.

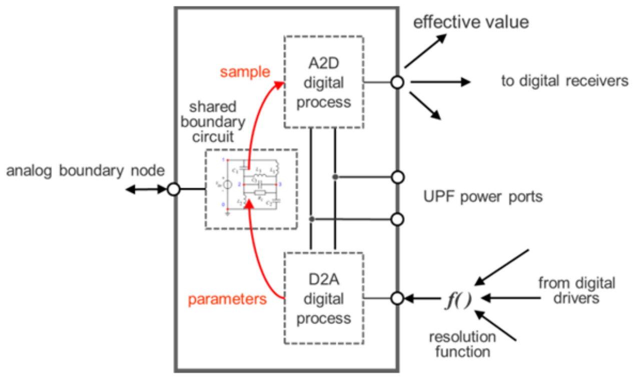


Fig. 1. General model of a power sensitive, bidirectional signal connect element

B. Selection of a signal connect element

Digital and analog elements are connected using port, net, or node connections or *associations*. In the present context we reserve the word *net* to designate the equipotential point modeled by a series of such associations starting at a locally-declared *root element* and descending through the design hierarchy tree from instance to instance. We will call the SPICE nodes, VHDL signals and terminals, and Verilog nets that participate in a net the *nodes* of the net. We will also define here the terms *net type* and *net mode*. Net type and mode are used in selecting the appropriate signal connect element for a given net.

[1] In Verilog-AMS, they are required to be so combined.

At certain places in the design hierarchy a digital node of a net is read by a process. These elements are the digital receivers of the net. At certain places processes will assign new values to digital nodes of the net. These are the drivers of the net. The same element may serve simultaneously as receiver and driver.

A *characterizing association* is an association between an electrical node and a digital node within a net. The elaborator of a mixed-signal simulator identifies the *mode* of each characterizing association. If the dataflow is from the digital side to the analog side, the mode is D2A; if from the analog to the digital side, A2D; and if bidirectional, BIDIR.

If a net contains a characterizing association then the net is a *mixed-signal net*. The *net type* of a mixed-signal net is a function of the types of its nodes[2]. The *net mode* of a mixed signal net is determined by the modes of its characterizing associations. If its characterizing associations are all of mode D2A, then the net mode of the mixed-signal net is D2A; if they are all of mode A2D, then the net mode of the mixed-signal net is A2D; otherwise the net mode of the mixed-signal net is BIDIR.

Any candidate signal connect element must have port or interface nodes compatible with the net type and mode of the net with which it is associated. Different models (or the same model with different parameters) may be assigned to nets with identical type and mode based on other user-defined criteria—for example, the particular sub-hierarchy containing the root element of the net.

C. Coding a power sensitive signal connect element

Power sensitive signal connect elements follow the UPF conventions for an RTL model without implicit power control. The models have one signal, one electrical terminal, and a pair (power and ground) of the standard UPF *supply_net_type*. In each case, the behavior of the connect element can be customized with generic parameters. The use of the attributes conforms to the UPF standard with regard to attributing ports for automatic connection.

The A2D connect element *A2Real* in Fig. 2 and Fig. 3 uses a straightforward algorithm to check the input analog voltage against its previous value. If the value changes by as much as the *tt* parameter, then a new digital value is driven on to *outp*. If the power state of either power net is *OFF*, the output is unconditionally driven to *voff*.

In the D2A connect element *Logic2A* in Fig. 4 and Fig. 5, the primary power port is used to detect the digital power state. If the state is *ON*, then the output analog signal is ramped-up to a voltage level equivalent to the supply voltage level. Otherwise, it is set to fall to the *OFF* value.

A mixed-signal simulator should include a library of power sensitive signal connect elements for digital signals of *std_logic* (Verilog logic), *bit*, and *real* type. The extension to other types (e.g., records or arrays) is straightforward.

[2] In Verilog-AMS, the system of disciplines serves as a surrogate for the types.

```
Library ieee;
  use ieee.electrical_systems.all;
  use ieee.upf.all;
entity A2Real is
  generic(
    tt   :  real := 1.0e-3;
    voff :  real := 0.0);
  port(
    terminal inp :  electrical;
    signal outp  :  out real;
    signal power :  in supply_net_type;
    signal ground:  in supply_net_type);
    attribute UPF_pg_TYPE : string;
    attribute UPF_pg_type of
       power : Signal is "primary_power";
    attribute UPF_pg_type of
       ground : Signal is "primary_ground";
begin
   ASSERT tt > 0.0
   REPORT "Tt=" & real'image(tt) &
    " but must be postive."
   SEVERITY FAILURE;
end entity A2Real;
```

Fig. 2. VHDL-AMS entity for an electrical to real (A2D) connect element

```
architecture ams of A2Real is
  quantity vinp across inp;
begin
  p : process
   variable pstate, gstate: net_state;
   variable scopy: real;
  begin
  pstate := get_supply_state(power);
  gstate := get_supply_state(ground);
  if ((pstate = FULL_ON or pstate = PARTIAL_ON) AND
       (gstate = FULL_ON or gstate = PARTIAL_ON)) then
      scopy := vinp;
    else
      scopy := voff;
    end if;
    outp <= scopy;
    wait on vinp'above(scopy+tt),
       vinp'above(scopy-tt),power, ground
  end process p;
  break on scopy;
end architecture ams;
```

Fig. 3. VHDL-AMS architecture for an electrical to logic (A2D) connect element

```
library ieee;
  use ieee.electrical_systems.all;
  use ieee.upf.all;
  use ieee.std_logic_1164.all;

entity Logic2A is

  generic( trise           :  real := 1.0e-12;
           tfall           :  real := 1.0e-12;
           vx_as_previous  :  boolean := true;
           vx              :  real := 0.0;
           voff            :  real := 0.0);

  port( terminal outp      :  electrical;
        signal inp         :  IN std_logic;
        signal power       :  IN supply_net_type;
        signal ground      :  IN supply_net_type);

     attribute UPF_pg_TYPE : string;
     attribute UPF_pg_type of
        power : Signal is "primary_power";
     attribute UPF_pg_type of
        ground : Signal is "primary_ground";

end entity Logic2A;
```

Fig. 4. VHDL-AMS entity for a logic to electrical connect element

```
architecture ams of Logic2A is
  quantity vout across iout through outp;

  type rt is array(std_logic) of REAL;
  signal reff : Real := 0.0;
  signal veff : Real := 0.0;

  function r_table(val : Std_logic) return Real is
    begin
      case val is
        when 'U'|'X'|'0'|'1'      => return ron;
        when 'W'|'L'|'H'          => return rwk;
        when  'Z'|'-'             => return rhiz;
      end case;
  end function r_table;

  function V_table(val: Std_logic;vhigh, vlow :real)
    return Real is
    begin
      case val is
        when '1'|'H'      => return vhigh;
        when '0'|'L'      => return vlow;
        when 'OTHERS      => return vx;
      end case;
  end function V_table;

  begin

    vout == veff'ramp(tt) + iout * reff'ramp(tt);

    p : process(inp,power,ground)
       variable psupply, gsupply : real;
          variable pstate, gstate : net_state;
    begin
       pstate := get_supply_state(power);
       psupply := get_supply_voltage(power);
       gstate := get_supply_state(ground);
       gsupply := get_supply_voltage(ground);
       if pstate >= PARTIAL_ON AND gstate >= PARTIAL_ON
       then
          reff <= r_table(inp);
          veff <= v_table(inp,psupply,gsupply);
       else
          reff <= 0.0;
          veff <= voff;
      end if;
    end process p;
end architecture ams;
```

Fig. 5. VHDL-AMS architecture for a logic to electrical connect element

III. Power Pins and Power Connect Elements

An element of an AMS hierarchy may have analog pins that supply analog power or that require connection to an analog power supply. If the element is instantiated in a pure analog context, the power connections are created by association in the same manner as ordinary signal connections. However, if the element is instantiated from a digital RTL context governed by a UPF-based power description, then a problem arises. Since power is implicit in the RTL code, there are no explicit power nets to associate with the analog power pins. In our implementation, the problem is resolved during the process of assembling the composite hierarchy (*elaboration*).

A. *Connecting UPF power to analog pins*

If the instance is a consumer of power, power supplies must be connected to each electrical power/ground pair (*pg pair*) during elaboration. One common option is to connect the pg pair to a globally accessible SPICE source element. However, in a design with power defined by UPF, the pg pair must be connected to a power supply that is dynamically responsive to the controls described by UPF. Our interpretation requires the elaborator to interpose a *power-to-*

electrical connect element (P2E) for this purpose. An example of a P2E coded in VHDL-AMS[3] appears in Fig. 6.

```
library ieee;
  use ieee.electrical_systems.all;
  use ieee.upf.all;

entity P2E is
  generic (
    Voff, Vund  : Real := 0.0;
    trise, tfall : Real := 50.0e-12);
  port (
    signal upfin : in supply_net_type;
    terminal vdd, vss : electrical);
end entity P2E;

architecture ams of P2E is
    signal s : real := 0.0;
    quantity vds across ids through vdd to vss;
begin
  p: process(upfin)
      variable current_state, last_state: net_state :=
get_supply_state(upfin);
  begin
    last_state := current_state;
    current_state := get_supply_state(upfin);
    case current_state is
      when FULL_ON =>
        s <= get_supply_voltage(upfin);
      when PARTIAL_ON =>
        if last_state = FULL_ON then
          s <= Voff;
        else
          s <= get_supply_voltage(upfin);
        end if;
      when OFF =>
        s <= Voff;
      when OTHERS =>
        s <= Vund;
    end case;
  end process p;

  If s = Voff use
    ids == s'ramp(trise,tfall)/1.0e9;
  else
    vds == s'ramp(trise,tfall);
  end use;
   break on s;
end architecture ams;
```

Fig. 6. VHDL-AMS model for a P2E

The P2E converts the digital input *upfin* of the UPF-defined *supply_net_type* to an analog voltage. The digital state of *upfin* indicates whether the digital power port is powered up or powered down. The state controls the voltage and impedance of the branch from vdd to vds, which will be used to power the analog blocks. The conditional equation defining the branch (the *if…use*) exhibits high impedance when it is turned off. The ramps ensure a gradual transition in either case.

B. *Connecting an analog power supply to a UPF supply function*

If the instance will provide the *root supply driver* of the power function of a UPF power set (see (1) 5.1 “Supply network creation”), then the elaborator must interpose an *electrical-to-power connect element* (E2P) between a pg pair and the corresponding UPF power function. An example of an E2P appears in Fig. 7.

[1] The examples are coded in VHDL-AMS rather than Verilog-AMS because Verilog-AMS has no way to represent the structure of the UPF power type supply_net_type. The design elements themselves may be in any digital or mixed-signal HDL, including Verilog-AMS.

```
Library ieee;
  use ieee.electrical_systems.all;
  use ieee.upf.all;

entity E2P is
  generic (voff : real := 0.2;
           von  : real := 0.5;
           eps  : real := 1.0e-3);

  port (signal upfout : out supply_net_type;
        terminal vdd, vss : electrical);
end entity E2P;

architecture ams of E2P is

  quantity vds across vdd to vss;

  signal sds : real := 0.0;

  begin

    upf : process(vds'above(sds+eps),vds'above(sds-eps))
      variable returnSupply : boolean;
    begin
      sds <= vds;
      if sds < voff then
        returnSupply := supply_off("upfout");
      elsif sds > von then
        returnSupply := supply_on("upfout",vds);
      else
        returnSupply := supply_partial_on("upfout",vds);
      end if;
    end process upf;

end architecture ams;
```

Fig. 7. VHDL-AMS model for An E2P

The voltage difference *vds* between *vdd* and *vss* will drive the digital output *upfout*. If *vds* exceeds a supplied threshold *von*, then *upfout* will be powered up to *vds* and the state will be set to *ON* using the UPF *supply_on* function. If *vds* is below a supplied threshold *voff*, then *upfout* will be powered down and the state will be set to *OFF* using the *supply_off* function. If *vds* is between the thresholds, then *upfout* will be given the voltage *vds* and the state *PARTIAL_ON*.

C. *Marking analog pins as pg pairs*

Analog or AMS blocks may require power that will be supplied by P2E connect elements inserted at elaboration time.

There is no automatic way to recognize the power pins in the block interface. The pg pairs must be decorated with attributes of type *UPF_pg_type* that distinguish their function (Fig. 8). This use of *UPF_pg_type* is a new but consistent interpretation of the UPF standard.

```
module analog_source (
   (*UPF_pg_type = "primary_power"*)
      output electrical Vdd,
   (*UPF_pg_type = "primary_ground"*)
      output electrical Vss,
    input IN_CTRL,
    output electrical OUT_PULSE );
   real ctrl_enable;
   analog begin
       ctrl_enable = 0;
       if (IN_CTRL === 1) ctrl_enable = 1;
           V(OUT_PULSE) <+ transition
      (ctrl_enable? 3.3:0.0, 0.2e-6, 0.5e-6, 0.5e-6);
   end
endmodule
```

Fig. 8. *UPF_pg_type* attributes of the ports of a Verilog-AMS module

A similar mechanism provides a way to label the pg pins of a SPICE block.

IV. An Example of a Mixed-Signal DUT

The extension of UPF to the mixed-signal domain described in the preceding sections has been implemented in the Mentor Graphics® Questa® ADMS mixed-signal simulator (3). We will illustrate this methodology using a very simple and artificial design to keep the focus on power management.

A. *Principles of operation*

The design under test, along with its test stimulus and power supplies, is illustrated in Fig. 9.

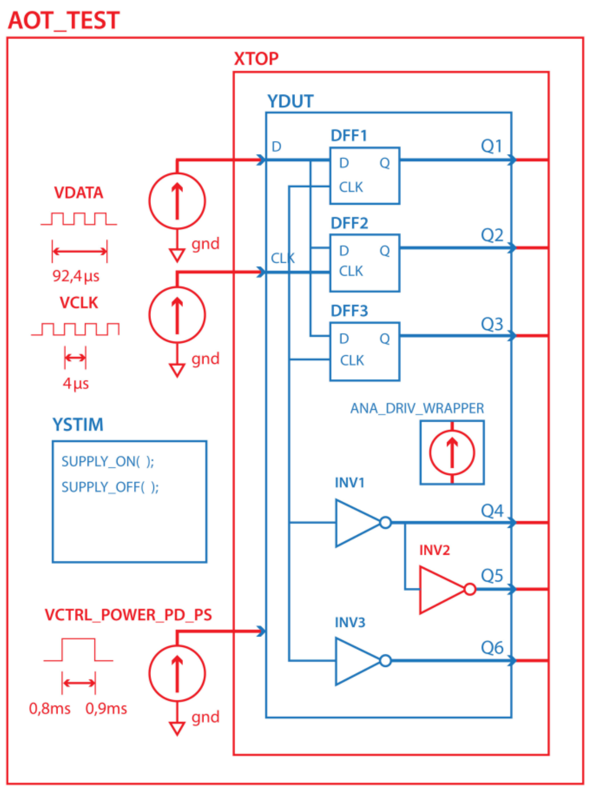


Fig. 9. The design under test with its testbench

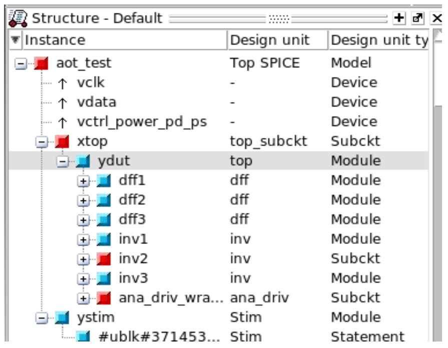
The top level of the hierarchy is coded in SPICE with a digital Verilog design under test *YDUT* instantiated within a SPICE subcircuit *XTOP*. The choice of top level language is arbitrary, but only a single UPF root is allowed by the implementation.

There are two types of cells, either D flip-flops (coded in digital Verilog) or inverters (digital Verilog for instances *INV1* and *INV3*, SPICE for *INV2*).

The flip-flops are clocked by the SPICE source *VCLK* (250 KHz). They sample data from analog SPICE source *VDATA* (the binary pattern “1100101” repeated every 92 µs).

The cell outputs (Q1-Q6) may be monitored to verify whether the corresponding power supply is turned ON (cell output ‘0’ or ‘1’) or turned OFF (cell output ‘X’).

The design hierarchy appears in the simulator “structure” window as illustrated Fig. 10. The implicitly inserted power and signal connect modules are not visible by default.


Structure - Default

Instance	Design unit	Design unit ty
aot_test	Top SPICE	Model
vclk	-	Device
vdata	-	Device
vctrl_power_pd_ps	-	Device
xtop	top_subckt	Subckt
ydut	top	Module
dff1	dff	Module
dff2	dff	Module
dff3	dff	Module
inv1	inv	Module
inv2	inv	Subckt
inv3	inv	Module
ana_driv_wra...	ana_driv	Subckt
ystim	Stim	Module
#ublk#371453...	Stim	Statement

Fig. 10. The design hierarchy, as it appears in the simulator GUI

B. *The power domains*

The Verilog instance *YSTIM* will control the UPF power domains assigned to the elements of *YDUT*. Four power domains are superimposed on the DUT, as illustrated in Fig. 11.

- *PD_TOP* controls digital cells *DFF1* and *INV1* and is driven from the digital block *YSTIM* using supply_on and supply_off UPF function calls.
- *PD_ECO* controls digital cell *DFF2* and *SPICE* block *INV2* and is driven from *YSTIM* using supply_on and supply_off function calls.
- *PD_ANADRIV* controls digital cell *DFF3* and is driven by a SPICE source in subcircuit *ANA_DRIV*.
- *PD_PS* controls digital cell *INV3* and is driven by the SPICE source *VCTRL_POWER_PD_PS*.

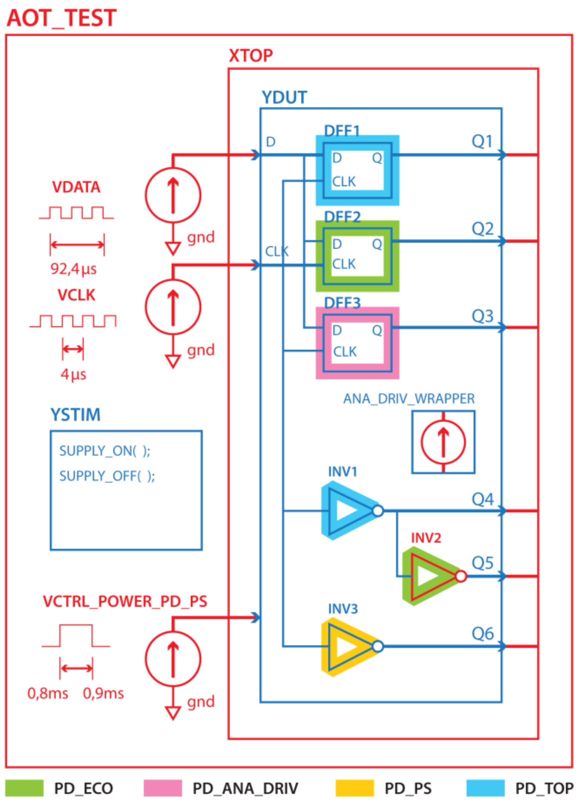


Fig. 11. The power domains of the DUT

```
`timescale 1 µs/1 µs
import UPF::*;

module Stim();

initial begin
     supply_on("/AoT_test/XTOP/YDUT/gnd_port", 0.0);
// from 0 to 100µs, All Power Domains OFF
     #100;
// test 1 (100 to 400µs): Power Domain PD_Top
     /// PD_TOP turned ON = 5.0
     supply_on("/AoT_test/XTOP/YDUT/pwr_port", 5.0);
     #100;
// test 2 (200 to 500µs): Power Domain PD_Top AND PD_ECO
     /// from 200 to 250ns, PD_ECO turned ON = 5.0 V
     supply_on("/AoT_test/XTOP/YDUT/pwr_port_eco", 5.0);
     #50;
     /// from 250 to 300ns, PD_ECO turned ON = 0.5 V
     supply_on("/AoT_test/XTOP/YDUT/pwr_port_eco", 0.5);
     #50;
     /// from 300 to 350ns, PD_ECO turned ON =3.3 V
     supply_on("/AoT_test/XTOP/YDUT/pwr_port_eco", 3.3);
     #50;
     /// from 350µs , PD_ECO turned OFF
     supply_off("/AoT_test/XTOP/YDUT/pwr_port_eco");
     #50;
     /// from 400µs , PD_TOP turned OFF
     supply_off("/AoT_test/XTOP/YDUT/pwr_port");
     #50;
// test 3 (500µs to 700µs): Power Domain PD_ANADRIV
     /// The analog source in ANA_DRIV rises, using SPICE
     /// "pwl (0 0 500µs 0 501µs 5 600µs 5 601µs 0)"
     #250;
// test 4 (700µs to 1000µs):
//Power Domain PD_PS via power swtich PD_PS_SW
     ///  At 700µs, the supply port that is the
     /// input to the switch is turned ON = 5.0 V
     supply_on("/AoT_test/XTOP/YDUT/pwr_port_PS", 5.0);
     ///  PD_PS_SW is controlled by
     /// the analog source VCTRL_POWER_PD_PS.
     /// The control is switched on at 800µs
     /// and off again at 900µs, using SPICE
     /// "pwl(0 0 800u 0 801u 5 900u 5 901u 0)""
     end
endmodule
```

Fig. 12. Testbench stimulus illustrating the sequence of demonstration tests

The chronogram in Fig. 13 illustrates the power control sequence imposed by the testbench module *YSTIM* of Fig. 12. the analog source *ana_drive*, and the UPF switch controlled by the analog source *VCTRL_POWER_PD_PS*. Four tests are performed in sequence. The tests, and their results, are detailed in subsequent sections.

C. The UPF definition

Fig. 14 shows the UPF file that defines the power topology of the DUT. The scope is limited by the *set_design_top* and *set_scope* commands to the top of the digital design under test. Each of the four power domains are described in turn. All domains are in the *top* scope for simplicity. They all share the *primary_ground_net*, *gnd_net*, declared in domain *pd_top*.

The *pd_top* domain is controlled exclusively from the testbench. The analog power supply *ana_driv* is instantiated in top, and must be explicitly excluded from control by the *-exclude_elements* argument.

Domain *pd_eco* includes an isolation strategy for signal *en_iso* of the element *dff2*. The isolation element is powered by *pwr_net*, declared in the *pd_top* section. The net, *pwr_net*, is connected to the power port, *pwr_port*, of *pd_eco*, which is turned on by the testbench at 100 µs and stays on for the remainder of simulation.

Domain *pd_PS* contains a power switch that controls its primary power input. The switch is exercised from the testbench in the example.

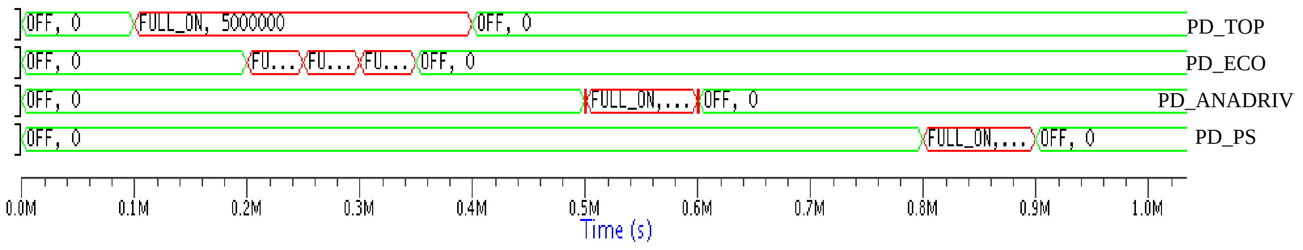


Fig. 13. Chronogram of the switching regime for the power domains (the signals displayed are of the record type supply_net_type defined by the UPF standard. The “500000” is in units of microvolts)

```
upf_version 2.0
set_design_top top
set_scope .

#####################
#pd_top Power Domain
#####################

create_power_domain pd_top -elements {.} \
    -exclude_elements { ana_driv1 }

#####################
#pd_eco Power Domain
#####################

create_power_domain pd_eco -elements { dff2 inv2 }

set_isolation  iso_strategy1 -domain pd_eco \
    -isolation_supply_set pd_top.primary \
    -elements { dff2 } \
    -clamp_value 0 \
    -isolation_signal { en_iso }

#####################
#pd_driven_from_analog Power Domain
#####################

create_power_domain pd_anadriv -elements { dff3 }
create_supply_net pwr_net_anadriv -domain pd_anadriv

connect_supply_net pwr_net_anadriv  \
    -pg_type primary_power -cells { ana_driv }

create_supply_set ana_ss \
    -function {power pwr_net_anadriv} \
    -function {ground pd_top.primary.ground}

associate_supply_set ana_ss -handle pd_anadriv.primary

#####################
#pd_PS Power Domain
#####################

create_power_domain pd_PS    -elements { inv3 }
create_supply_port pwr_port_PS -domain pd_PS
create_supply_net pwr_net_PS -domain pd_PS
create_supply_net switched_net_PS -domain pd_PS
connect_supply_net pwr_net_PS -ports { pwr_port_PS }

create_power_switch pd_PS_sw \
    -domain pd_PS \
    -output_supply_port { out_sw_pd_PS switched_net_PS } \
    -input_supply_port { in_sw_pd_PS pwr_net_PS } \
    -control_port { ctrl_sw_pd_PS CTRL_POWER_PD_PS } \
    -on_state { normal_working in_sw_pd_PS {ctrl_sw_pd_PS
} } \
    -off_state { off_state {!ctrl_sw_pd_PS} }

create_supply_set switched_ss \
    -function {power switched_net_PS} \
    -function {ground pd_top.primary.ground}
associate_supply_set switched_ss -handle pd_PS.primary
```

Fig. 14. The UPF file for the example

V. The Demonstration Tests

A. *Test 1*

The results of test 1 are illustrated in Fig. 15. At 100 μs power domain *pd_top* turns on and the outputs of *dff1* and *inv1* become valid. Both *dff1* and *inv1* are digital modules governed by normal UPF implicit rules.

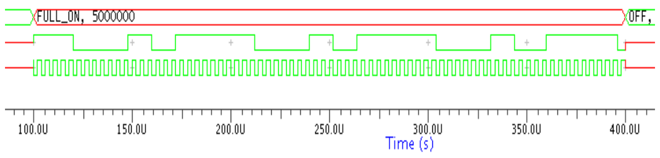


Fig. 15. Test 1—waveforms are from top to bottom: *pwr_port*, *dff1:Q* output, *inv1:outp* output

B. *Test 2*

Test 2 modulates primary power in *PD_ECO*, which contains *DFF2* and *INV2*. The effects of the varying power supply voltage (first 5 V, then 0.5 V, then 3.3 V) are visible on the output of the analog inverter *INV2* (Fig. 16). DFF2 is unaffected.

INV2 uses power supplied by P2E boundary elements inserted at elaboration time.

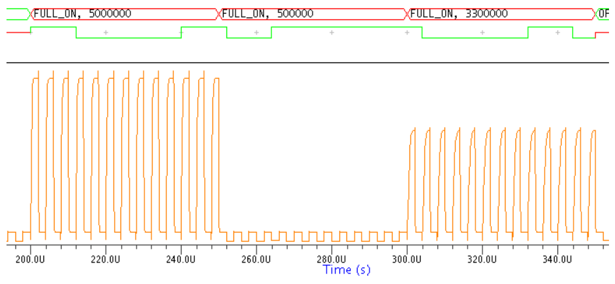


Fig. 16. Test 2—waveforms are from top to bottom: *pwr_port*, *dff2:Q* output, *inv2* output

C. *Test 3*

A SPICE sub-circuit *ana_driv* provides primary power to domain *PD_ANADRIV*. Digital module DFF3 is in *PD_ANADRIV*. As ana_driv ramps up to 5 V the primary power for *PD_ANADRIV* enters *PARTIAL_ON* state (the small red rectangles on the waveform in Fig. 17), and then *FULL_ON* state. The output of *DFF3* appears only when power is *FULL_ON*.

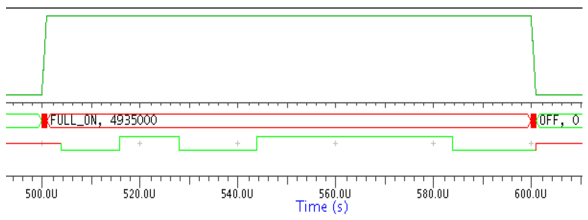


Fig. 17. Test 3—waveforms are from top to bottom: *ana_driv* output, *pwr_net_anadriv*, *dff3:Q* output

D. Test 4

Supply port *pwr_port_PS* was turned on at 700 µs, which is the input to the switch defined at the end of Fig. 14. The power domain *PD_PS* is governed by the output of the switch. The switch is controlled by an independent analog source *ctrl_power_pd_PS*, which rises at 800 µs and falls again at 900 µs. The output of the switch is the primary power for *PD_PS*, which contains *inv3*. In consequence, the output of *inv3* is valid only in that interval (Fig. 18).

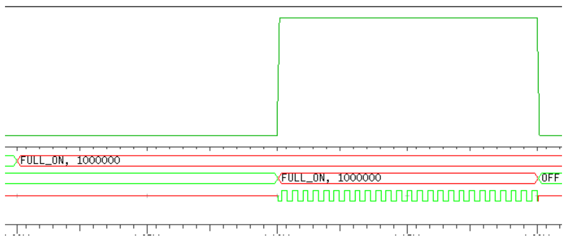


Fig. 18. Test 4—waveforms from top to bottom are: *ctrl_power_pd_PS*, *pwr_port_PS*, *switched_net_PS*, *inv3:outp*

VI. CONCLUSION

The interpretation of UPF here proposed and illustrated satisfies the additional requirements exposed when the design flow includes the addition of SPICE, VHDL-AMS, or Verilog-AMS in a digital design that uses UPF to represent the power characteristics of the RTL portions of the hierarchy. The important extensions are two: a method for providing UPF-controlled, SPICE-level power to those AMS instances that require it, and a method for defining signal connect elements (connect modules) that are sensitive to the power state of the enclosing UPF power domain.

The methodology has been implemented in the Mentor Graphics Questa ADMS mixed-signal simulator. Mentor Questa SIM users are currently experimenting with our mixed-signal UPF extension for large digital designs (upwards of 20 power domains) that were originally coded in SystemVerilog and VHDL.

VII. REFERENCES

[1] Institute of Electonic and Electrical Engineers. The IEEE Standard for Design and Verification of Low Power Integrated Circuits (IEEE Std 1801-2009) . 2009.

[2] Flynn, David, Rob Aitken, Alan Gibbons, and Kaijian Shi. Low Power Methodology Manual: for System-on-chip Design. New York, NY : Springer, 2007.

[3] Mentor Graphics Corporation. Questa ADMS User's Guide 13.1. Wilsonville, OR : Mentor Graphics Corporation, 2013.