A Framework for AMS VIP development with UVM and Verilog-AMS

Mike Bartley, CEO, TVS
Jeganath Gandhi R, Project Lead, TVS
Agenda

• Feature extraction from AMS Design Specification.
• DUT components: Digital RTL and Analog AMS model.
• VIP development plan from the extracted feature list and digital and analog design partition.
• UVM based VIP components including MS BFM to interact with Analog interfaces.
• Real value monitor based checks.
• Coverage Collection and Signoff.
Introduction

• A reusable, scalable methodology to improve AMS design verification.
• Leverage UVM’s built-in automation and testbench capabilities to AMS design verification.
• The methodology scales to SOC level verification as well as being reusable and portable at the IP level. Supports, module-to-system and project-to-project reuse.
• Does not intend to replace Analog Verification, but aid Analog IP verification early in design cycle.
• This paper presents the technique with an example Power Management IC (PMIC) DUT.
PMIC

• The PMIC detects presence of Battery and powers different peripherals of a mobile platform.
• The PMIC includes a Power management state machine in the digital domain and multiple LDOs and SMPS in the analog domain.
• Comparators and ADCs form the interface between 2 domains.
• State change is based on current battery voltage level, generating system resets, clocks, responding to interrupts from the host processor.
• FSM controls signals to enable the different voltage regulators, configuring them in low power mode, based on the.
• Digital component interacts with the system on Chip with different Interface protocols, while the analog component interacts with different regulators, bias generators and comparators.
Feature Extraction

• Driven by the functional specification.
• Feature List used as a metric for test development, checker development, coverage modeling and verification signoff.
• PMIC Examples
  – When battery voltage is below 1.5v the state machine is power cut state and none of the LDOs are enabled
  – When battery voltage is above 1.5v and below 3.0v the state machine is Low power and VCLP alone is enabled to power the low power domain.
  – Full Power state is when batter voltage is above 3.0v
  – VChi is enabled only when in Full power state
The AMS DUT

• The difference in an AMS Verification flow: the continuous discipline seen in the Analog components used as models.

• In our example of a PMIC, the LDO output voltage is a function of the control signals from the Digital Sub chip, the bias currents from the bias generator, the reference voltage.

• The modelling techniques employed to develop the bias generator, reference generator, the Voltage regulator involving the ramp timings, delays modelled are important in designing an accurate Self-Checking VIP.

• This continuous behaviour has to be taken into account while sampling the analog output in the VIP monitor.

• while measuring a frequency of an analog signal, we might have to drop few samples immediately after power up, depending on the models settling time.
include "disciplines.vams"

Module por(vbat, vrefin, gnd, vlow, vhigh, vclp_vref, vclp_iref);

// Parameters
// Threshold below which vlow goes high and above which vlow goes low.
parameter real vlow_threshold = 1.5;
// Threshold below which vhigh goes low and above which vhigh goes high.
parameter real vhigh_threshold = 3.5;
// Threshold below which vhigh goes low and above which vhigh goes high.
parameter real vref_input_p = 1.5;

// Define input/output
Input vbat, vrefin, gnd;
//vbat: voltage at the battery terminal.
//vrefin: Input reference voltage for the comparator
//gnd: ground terminal
Output vlow, vhigh, vclp_vref, vclp_iref;
//vlow: 1 - battery voltage lower than low voltage threshold
//vhigh: 1 - battery voltage higher than minimum value for full chip operation

//Define port types
Wreal vbat, vrefin, gnd;
Wreal vclp_vref, vclp_iref;

Reg vlow, vhigh;

Real vclp_vref_r, vclp_iref_r;
initial begin
  vclp_vref_r = 0.0;
  vclp_iref_r = 0.0;
  vlow = 1;
  vhigh = 0;
end

always @(vrefin, gnd, vbat) begin
  // perform logical assignment of the control outputs based on input battery voltage and reference and ground inputs.
  if (vrefin == vref_input_p & gnd == 0.0) begin
    if (vbat > vlow_threshold) begin
      // When the battery level is higher than the vlow threshold, the device is just out of low power.
      vlow = 0;
      vhigh = 0;
      vclp_vref_r = 1.0;
      vclp_iref_r = 0.1;
    end
    else if (vbat < vlow_threshold) begin
      // When the battery level is lower than the vlow threshold, this indicates a loss of power.
      vlow = 1;
      vhigh = 0;
      vclp_vref_r = 0.0;
      vclp_iref_r = 0.0;
    end
  end
  else begin
    //Check interconnection or reference voltage check failure
    vlow = 'bx;
    vhigh = 'bx;
    vclp_vref_r = 0.0;
    vclp_iref_r = 0.0;
  end
end

assign vclp_iref = vclp_iref_r;
assign vclp_vref = vclp_vref_r;
endmodule
The UVM environment

- A UVM test bench is composed of reusable verification components in System Verilog.
  - transaction item, driver, sequencer, monitor, agent and the encapsulating environment
- The difference in the digital VIP and an AMS VIP is the interaction of the monitor and driver with the analog components as detailed in next few slides.
The UVM environment
MS Driver

• Interact between the Digital components of the VIP and the Analog components of the DUT
• MS driver is not containing Analog signal processing, but treats the signals as discrete-time real value numbers
• The actual D/A conversion to the electrical signal is done in the Verilog-AMS domain by means of real-valued-modelling (RVM) techniques.
### Class por_item extends uvm_sequence_item:

```plaintext
rand real vbat;
rand real vref;
rand real iref;
constraint c1 { vbat inside { [0.0:5.0] }; }
constraint c1 { vref inside { [0.0:0.1] }; }
constraint c1 { iref inside { [0.0:0.1] }; }
```

...<uvm factory registration>...

endclass :por_item

### Interface por_intf():

```plaintext
real vbat;
real vref;
real iref;
endinterface: por_intf
```

### Class por_driver extends uvm_driver#(por_item):

```plaintext
Virtual por_intf por_if;
...
<uvm factory registration, phases>...

Task run_phase(uvm_phase phase);
fork
forever begin
seq_item_port.get_next_item(req);
por_if.vbat = req.vbat;
por_if.vref = req.vref;
por_if.iref = req.iref;
$cast(rsp, req.clone());
seq_item_port.item_done(rsp);
end
join
endtask :run_phase
Endclass :por_driver
Connect Modules

• The connect modules form the interface between the real valued UVM environment and wreal/electrical analog disciplines.

• Connect modules are used to connect the continuous and discrete disciplines (mixed nets) of the design hierarchy together.

• A connect module defines the conversion of logic for a specific types of disciplines that it bridges. (E.g. : real to electrical).

• Connect modules are not instantiated if the UVM environment interacts only with the discrete domain.
connectmodule electoreal (ana_in, dis_out);
  input ana_in;
  electrical ana_in; //input electrical
  output dis_out;
  wreal dis_out; //output wreal
  ddiscrete dis_out; //discrete domain
  parameter real vdelta=1.8/64 from (0:inf); // voltage delta
  parameter real vtol=vdelta/4 from (0:vdelta); // voltage tolerance
  parameter real ttol=10p from (0:1m]; // time tolerance

  real dis_reg; //real register for A to D wreal conversion
  assign dis_out = dis_reg;
  always @(absdelta(V( ana_in ), vdelta, ttol, vtol))
    dis_reg = V( ana_in );

endmodule
Sampling and Checking

• The self-checking mechanisms, assertions work on the samples received from the MS monitor.

• For example, we might want to have a self-checking mechanism on a VCO output, which is a continuous analog waveform.

• Existing language constructs doesn’t allow us to have a continuous frequency check assertion on an analog signal. Solution is to convert the value to discrete real values, perform some computations on the values as the sample code below
virtual task clock_monitor();
var time prev_time;
var real ldo_output;
var real peak = 1.0;
var real trough = 0.0;
var real cur_sample;
var real prev_sample;
var real prev_2_sample;
var bit first_peak = 0;

@(posedge vco_if.clock)
forever begin
  @(posedge vco_if.clock)
  cur_sample = vco_if.ldo_out;
  if(cur_sample == peak) begin
    first_peak = 1;
  end else if(first_peak == 1) begin
    // Frequency Monitor
    if(cur_sample < prev_sample && prev_sample > prev_2_sample) begin
      peak_q.push_back(prev_time);
    end
    if(cur_sample > prev_sample && prev_sample < prev_2_sample)
      trough_q.push_back(prev_time);

    prev_2_sample = prev_sample;
    prev_sample = cur_sample;
    prev_time = $time;
    if(peak_q.size() != 0 && trough_q.size !=0)
      begin
        clk_period = (trough_q[0] > peak_q[0]) ? 
          (trough_q[0] - peak_q[0]) : (peak_q[0] - 
          trough_q[0]);
        if(!(clk_period > exp_period - err_margin) 
          && (clk_period < exp_period + err_margin))
          uvm_report_error("ERR_PERIOD", 
            $psprintf("Unexpected Time Period :: %0t - 
            Expected :: %0t", clk_period, exp_period));
      end
  end
endtask : clock_monitor
Coverage Collection

- As with any verification effort, the verification target has to be set or measured with metrics.
- In case of an AMS environment, this is done at the higher layer of the VIP, making it completely transparent to the analog components.
- The example Battery voltage coverage along with its trim value to test a VAMS comparator with use of system Verilog RVM is shown below
MS Coverage Model

```vhdl
logic [1:0] vbat_trim;
real vbat;

covergroup comparator_cg @(posedge vco_if.clock);

vbatvalues : coverpoint vbat {
  bins power_cut = {[0.1:0.5]};
  bins low = {[0.6:1.5]};
  bins full_power = {[1.6:5.0]};
}

Comp_cross : cross vbat_trim, vbatvalues;
endgroup
```
IP and SOC Level

- VIP follows the UVM methodology which allows to scale to different levels of design.
- Use transaction-Level modelling of all communication between verification components including MS interface.
- Ensures verification IP can be easily ported by either connecting to IP level interface or to a hierarchical interface at the SOC.
- Interface mechanism helps control where the VIP hooks up to the DUT.
- Example: The scoreboard should be taken care not to operate on cycle by cycle basis, but ONLY TLM. This ensures the self-checking mechanism is available even when the VIP is configured as PASSIVE in a SOC environment.
Advantages

• Apart from verifying the Full chip with Analog and Digital domains early in the design cycle, this flow can be used as proof concept for the Analog architecture while the design evolves.

• UVM and Metrics based verification helps to bring the methodology based verification to AMS designs and thus reduce cost and increase efficiency.

• Leverage flexibility of Standardized verification methodology, UVM.
Suggestions and Cons

• This methodology should co-exist with a strong module level verification of the Analog components, focusing on its parameters and analog behaviors.

• Full Chip integration should still be checked at the net-list level.

• When the accuracy of model is important and we use electrical discipline (rather than real / wreal), Efficient when significant portion of the design is Digital and few analog modules. Not efficient for example, when the digital part just an register interface.
Questions

Finalize slide set with questions slide