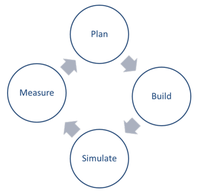
UVM Based Approach To Model Validation For SV-RNM Behavioral Models

Donald Lewis and Courtney Fricano
Analog Devices, Inc. 3 Technology Way, Norwood, MA 02062

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

Problem Statement

Applying a metric driven verification (MDV) flow to complex Analog-Mixed Signal designs is critical for successful verification.

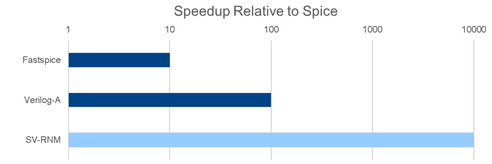


MDV Flow

A typical MDV flow requires a large number of simulations for:

- Nightly Regressions
- Coverage
- Vplan Annotation

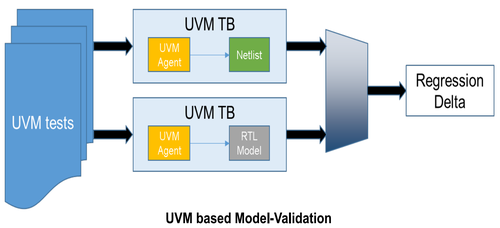
AMS designs simulate slowly, so in order to run enough simulations to achieve the goals of MDV, the speedup of SV-RNM models is often required.



The validation of these SV-RNM models is essential to the success of the verification effort. Model validation is required to ensure that functional simulations of the top level system accurately reflect the state of the design.

Solution

This work presents a model validation methodology that uses a UVM block level testbench to simulate both the analog netlist as well as the SV-RNM model. UVM is used to allow for high quality block level verification and easy re-use at the system level (typically also a UVM testbench).



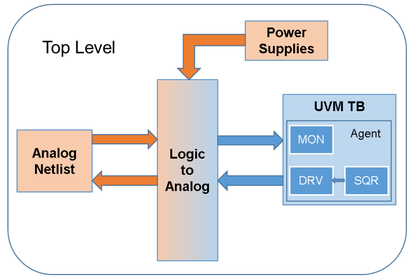
UVM based Model-Validation

The UVM testbench is designed to verify all of the functionality required of the SV model. The testbench flags an error if either the netlist or the model violates any of the checks. Running tests through the testbench with both the netlist and the model allows for a comparison of results to determine how well netlist and model match.

Implementation

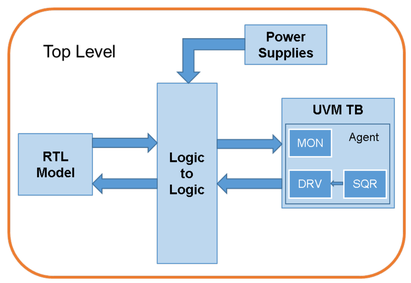
Implementation at the block level requires two testbench configurations: one for the analog netlist verification and one for the SV model verification.

The first configuration requires a co-simulation environment: the analog netlist and power supplies are simulated with an analog solver, and the UVM SystemVerilog testbench requires a digital solver. An example co-simulation testbench architecture is shown below. This structure allows for either simulator to be the master and avoids the need for a nested Verilog co-simulation, which may not be supported by the analog simulator.



Analog Configuration

The testbench must also be able to run as all digital for the SV model simulations. In this configuration, models for analog blocks and interfaces are replaced with their digital equivalents.



Digital Configuration

Digital Interface Checker Example

Assertions on the interface signals can be used to ensure block level interoperability with digital connections by checking timing specifications. For the checks, ranges of valid operation are used to provide a tolerance for mismatch between design and model.

```
property sva_prop_tsbh_check;
       time pdm33_3v_change;
       disable iff(!resetb_3v || (tm_set === 'h5) ||
                     fault_code === 3'h4 || !rstb)
       @(negedge tb_clk)
          ($rose(pdm33_3v)) |->
             (((pdm33_changed - pdm33_timer) >= min_tsbh) &&
              (pdm33_3v == pdm33_set_value));
endproperty : sva_prop_tsbh_check
```

Analog Checker Example

Analog functionality that is required for the SV real number model must be checked in UVM TB. Features or functionality that are not required to be modeled can remain as analog checkers or can be brought into the UVM TB.

```
//check for glitch pulses
if(this.ctl_en === 1) begin
   // count increments based on voltage level
   if(evdd_in > ovlo_e_th && this.ctl_evddcompen == 1)
      evdd_ov = evdd_ov + inc;
   else
      evdd_ov = 0;
   if(evdd_in < uvlo_e_th && this.ctl_evddcompen == 1)
      evdd_uv = evdd_uv + inc;
   else
      evdd_uv = 0;
   //...
end

// check multiple thresholds to determine if an error must be flagged
if (evdd_max > 64 || evdd_ov > 100 || evdd_uv > 100 || evdd_min > 64)
    evdd_err_req = 1;
//...

// check multiple thresholds to determine if an error can be flagged to
complete the window
evdd_err_valid = (evdd_max >= 10 || evdd_ov >= 12 ||
                  evdd_uv >= 12 || evdd_min >= 10);
//...

// when TB sees error flag
if (!evdd_err_valid )
   `uvm_error(get_type_name(),"evdd_err flagged before glitch timeout")
// if no error flag is seen
if(evdd_err_req)
   `uvm_error(get_type_name(),"evdd_err not latched after glitch pulse")
```

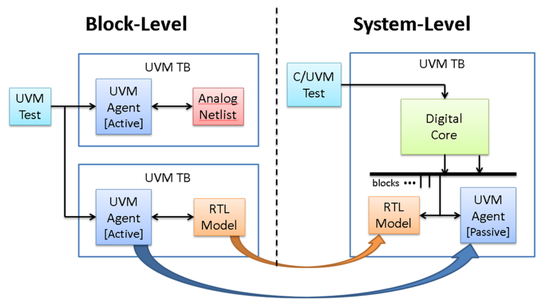
Validation

Once the checks are created, testbench is ready to be used for model validation. The following features allow for an automated high-quality validation methodology:

- Automated sourcing of analog netlist and SV-RNM model
- Regular regressions of same stimulus and tests provided to analog and digital testbench
- Automated messaging when model and design are out of sync

System-Level Integration

Use of UVM allows for re-use of monitors and drivers from the block level at the system level. This re-use mitigates the impact of the overhead associated with the block level UVM TB development. Additionally, the re-use improves the quality of the system level verification.



System Level Re-use

Results

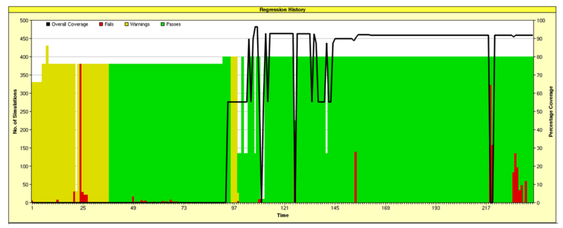
This methodology was successfully used to validate multiple new AMS blocks on a large dual-core SoC. These AMS blocks were critical to the SoC's functional safety features, and had no functional bugs after tapeout.

This verification effort exposed several real issues with both the circuit designs and the SV real number models. This methodology also enabled the use of UVM and MDV techniques for these AMS blocks in a large number of simulations.

	Design Type	Bugs Found	Simulations Run
AMS Block #1	new block	13	64595
AMS Block #2	modified	3	14675
AMS Block #3	modified	3	4886
System-Level	n/a	4	34271

Example design and model issues that were found include:

- Test mode implementation mismatch
- Oscillator trim implementation mismatch
- Reset pin polarity incorrect at system level
- Signal handshaking error between AMS and digital block found at system level



Regression History: Mismatches between the model and design shown as fails in red

Conclusions

The approach for model validation presented in this paper had the following benefits:

- Thorough functional validation of models
- Quick feedback on model or design changes via automated regression testing
- Use of modern MDV methods on AMS blocks
- Re-use of block level UVM components for easy integration into system level environment
- Generic, simulator independent approach

Acknowledgements

Thanks to the following Analog Devices employees for their feedback and contributions to this paper: John Mackintosh, David Brownell, and Mark Valley