2024 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

SAN JOSE, CA, USA
MARCH 4-7, 2024

UVM Testbench Automation for AMS Designs

Jonathan David – Innophase Inc.

Henry Chang – Designer's Guide Consulting, Inc.

accellera SYSTEMS INITIATIVE

Outline

- The Analog TB generation problem
- Designing a standard TB
- A generic UVM agent design
- Automating TB construction
- Results
- Resources

Why Long delays
until first test of mixed-signal designs?

Per design:

- Select test approach
- Build components
  - stimulate inputs
  - observe outputs
- Assemble
  - DUT
  - test components
- Manage
  - Error reports
  - Test checking

Result:

- Wild variations
  - Per design
  - Per verifier or team
  - Per company
- More TB time => Less Testing
  - Test quality/quantity suffers

Importance:

- Early System level
- Control Sequencing

- A Faster way?

2024 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

SAN JOSE, CA, USA
MARCH 4-7, 2024

Designing a Standard Test-bench

Select a standard framework

Design test-bench approach adaptable to any design.

accellera SYSTEMS INITIATIVE

UVM: the Incumbent Framework
(for digital design verification)

Pro

- Available training material
- digital team (probably) uses
  - Possible help source
- Built in:
  - Messaging (Fatal, Err, Warn, Info)
  - Phasing
  - Sequence management
  - Randomization
  - Test parallelization

Con

- Complicated
- Available training focused on digital bus transactions.
- Debugging also complicated

Alternatives?

Models-in-Minutes for the System or Chip-Level Test-bench?

- MiM generates models
- MiM generates block-level testbenches
  - Model validation
  - SystemVerilog RNM / AMS model vs Schematic
- TB easy to use
  - small designs
  - model testing and validation

- Limited above block-level IP top, chip-level
- Poor fit where:
  - Many sub-function types
  - 100+ control signals
  - Multiple test scenarios

- Not selected as full design framework

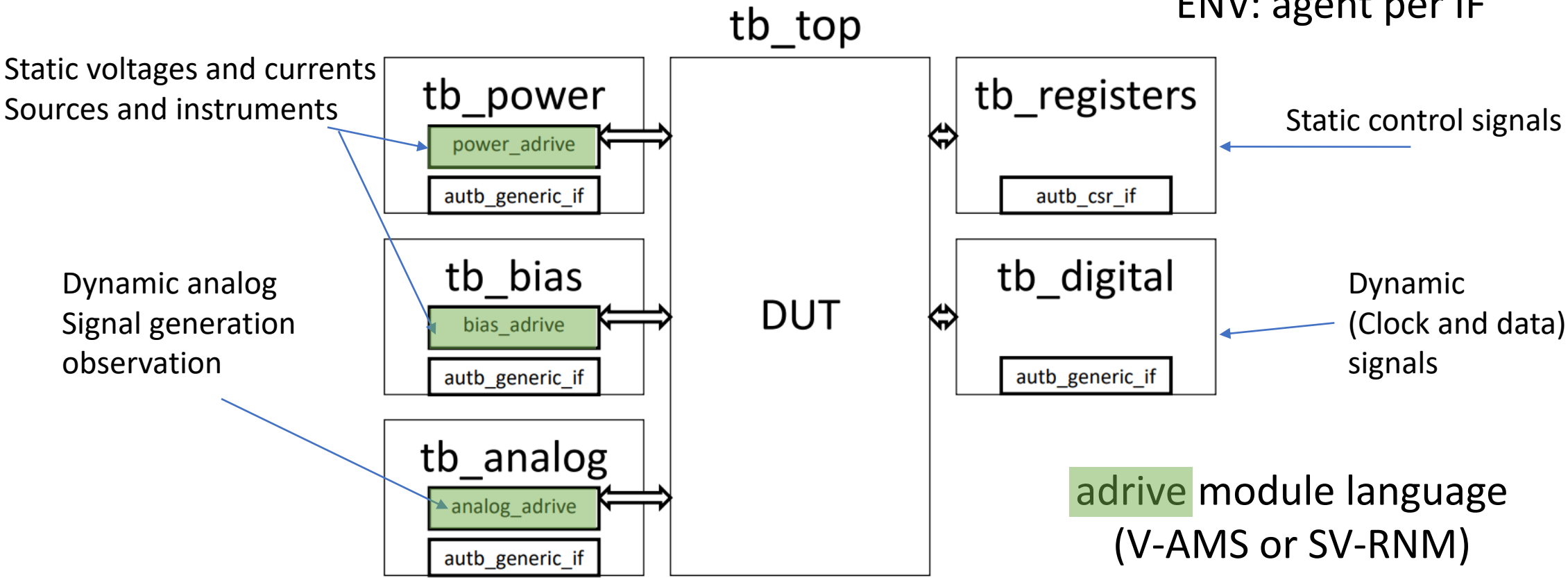
Approach: Manage Complexity
Standardize, Abstract, Automate

- Standardize:
  - Reusable standard agents
  - Stimulus generation patterns
  - Observation collection patterns
- Abstract:
  - Single file to edit
  - “Simple” data structure
  - Available Editor Support
- Automate:
  - Python
  - Jinja template rendering
- Agents:
  - Autb_generic agent
  - Autb_csr_agent
- Templated TB top with
  - Power, Bias, (other) Analog
  - Register, (other) Digital
- Python dictionary
  - NextedText format+ Python code and Jinja2 for code templates
- Python scripts
  - Jinja Templates

Standard AUTB Construction

TB: Adapter block per IF

ENV: agent per IF



adrive module language
(V-AMS or SV-RNM)
adapted based on dut

2024 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

SAN JOSE, CA, USA
MARCH 4-7, 2024

A Generic UVM Agent Design

Since we select UVM can we abstract away some complexity?

accellera SYSTEMS INITIATIVE

AUTB Generic IF

Interface -> sequence_item -> driver, monitor, agent

Flexibility: Key Requirement

```
1  // interface for the autb_generic agent.
2  interface autb_generic_if();
3      string settings[string], observations[string];
4      event sample_trigger;
5      string sequence_name, design_state_name;
6  endinterface
```

- Associative arrays
- String functions for get put
  - String <->logic variable
  - String <-> real variables
- String index
  - easy access to values by name.

- Modify per case:
  - Sequences
  - Adapter module

AUTB Generic Sequence Item (base)

Contains: same associative arrays adds housekeeping variables

```
1  // sends control values to tb in name, value pairs in a hash,
2  //  with values converted to string for sending
3  class autb_generic_seq_item extends uvm_sequence_item;…
5      `uvm_object_utils(autb_generic_seq_item)…
10     string settings[string];
11     string observations[string];
12     string design_state_name;…
14     extern function new(string name = "autb_generic_seq_item");
15     extern function void do_copy(uvm_object rhs);
16     extern function bit
17         do_compare(uvm_object rhs, uvm_comparer comparer);
18     extern function string convert2string();
19     extern function void do_print(uvm_printer printer);
20     extern function void do_record(uvm_recorder recorder);
21 endclass:autb_generic_seq_item
```

Base: simple string comparison (similar for convert2string, copy etc)

```
42 function bit autb_generic_seq_item::do_compare(
43   uvm_object rhs, uvm_comparer comparer);
44   autb_generic_seq_item rhs_;
45   string i;
46   bit eq = 1;
47   if(!$cast(rhs_, rhs)) begin
48     `uvm_error("do_compare", "cast of rhs object failed")
49     return 0;
50   end
51       eq &=  super.do_compare(rhs, comparer);
52   foreach (rhs_.settings[i])
53       eq &= settings[i] == rhs_.settings[i];
54   foreach (rhs_.observations[i])
55       eq &= observations[i] == rhs_.observations[i];
56   eq &= design_state_name == rhs_.design_state_name;
57   return eq;
58 endfunction:do_compare
```

AUTB Generic Driver

Passes sequence item to the interface

```
42  task autb_generic_driver::run_phase(uvm_phase phase);
43    autb_generic_seq_item req;
44    autb_generic_seq_item rsp;
45    int psel_index;
46    string setting;
47    forever
48      begin
49        seq_item_port.get_next_item(req);
50        `uvm_info($sformatf("%s_DRIVER",
51          this.get_full_name().toupper()),
52          $sformatf("Starting sequence %s",
53          req.get_name() ),UVM_LOW)
54        foreach (req.settings[setting])
55          m_vif.settings[setting] = req.settings[setting];
56        m_vif.sequence_name = req.get_name();
57        seq_item_port.item_done();
58      end
59  endtask: run_phase
```

Monitor is similar, and collects if changes, and puts them in an item.

- Here we chose to copy each element in the array to the if
- This allows unchanged elements in the interface to persist
- Separate sequences to manage subsets of variables simply.

AUTB CSR Interface and Agent

- Similar to generic agent
- Associative arrays:
  - *int* indexed by *string.*
- Because Many registers
  - conversions (int<->string) might affect performance
- Adds delay function
  - mimics bus transaction delays.

```
1   //interface for the autb_csr agent
2   interface autb_csr_if();
3       int readonly[string], writable[string];
4       string sequence_name, design_state_name;
5       int delay_us, delay_ns;
6   >   task delay(input int us_delay=1, ns_delay=0);…
18  endinterface
```

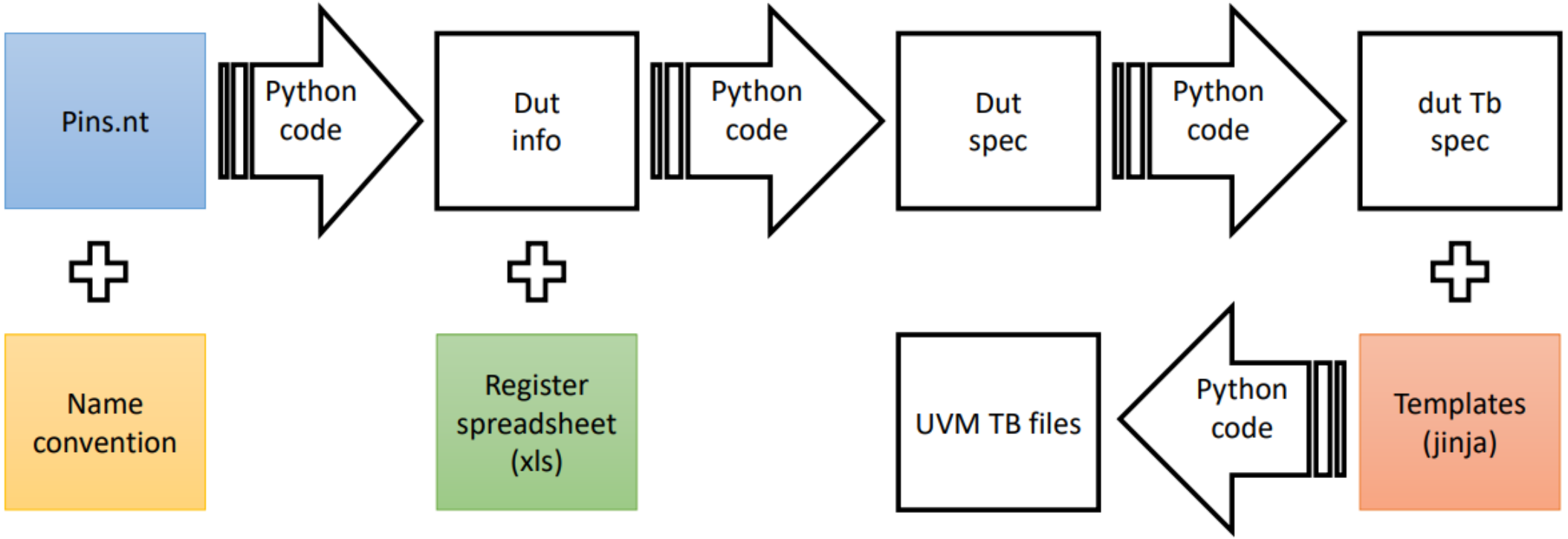
2024
DESIGN AND VERIFICATION™
DVCON
CONFERENCE AND EXHIBITION
UNITED STATES
SAN JOSE, CA, USA
MARCH 4-7, 2024

Automating Testbench Creation

accellera
SYSTEMS INITIATIVE

Automation Work Flow

Colored boxes show the inputs needed



Example Jinja Template

Data structure(dictionary) passed to template render engine
Included: dictionaries, lists, variables with their values
variety of functions for output interpolation

```
1   //SystemVerilog HDL for "{{library_name}}", "{{name}}" "svlog"
2   module {{ name }} ( {% set comma=joiner(', ') %}{% for port,portitem in ports.items()  %}{{ comma() }}
3       {{portitem['dir']}} interconnect {%if portitem['msb'] %}[{{portitem['msb']}}:{{portitem['lsb']}}]
4   {%- else %}     {%endif%}   {{port}} {% endfor %}
5   );
6   import uvm_pkg::uvm_config_db;
7   autb_generic_if pwr_if(); // interface contains an associative array of string with string index settings
8   initial begin
9       uvm_config_db#(virtual autb_generic_if)::set(null, "uvm_test_top", "power_vif", pwr_if);
10  end
11  {{name}}_adrive power_adrive({% set comma=joiner(', ') %}{% for port,portitem in ports.items()  %}{{ comma() }}
12      .{{port}}({{port}}){%endfor%});
13  bit sample_trigger;
```

loop

test

Dict var

Simple var

Function defined

Function used

Demo Design Data Structure

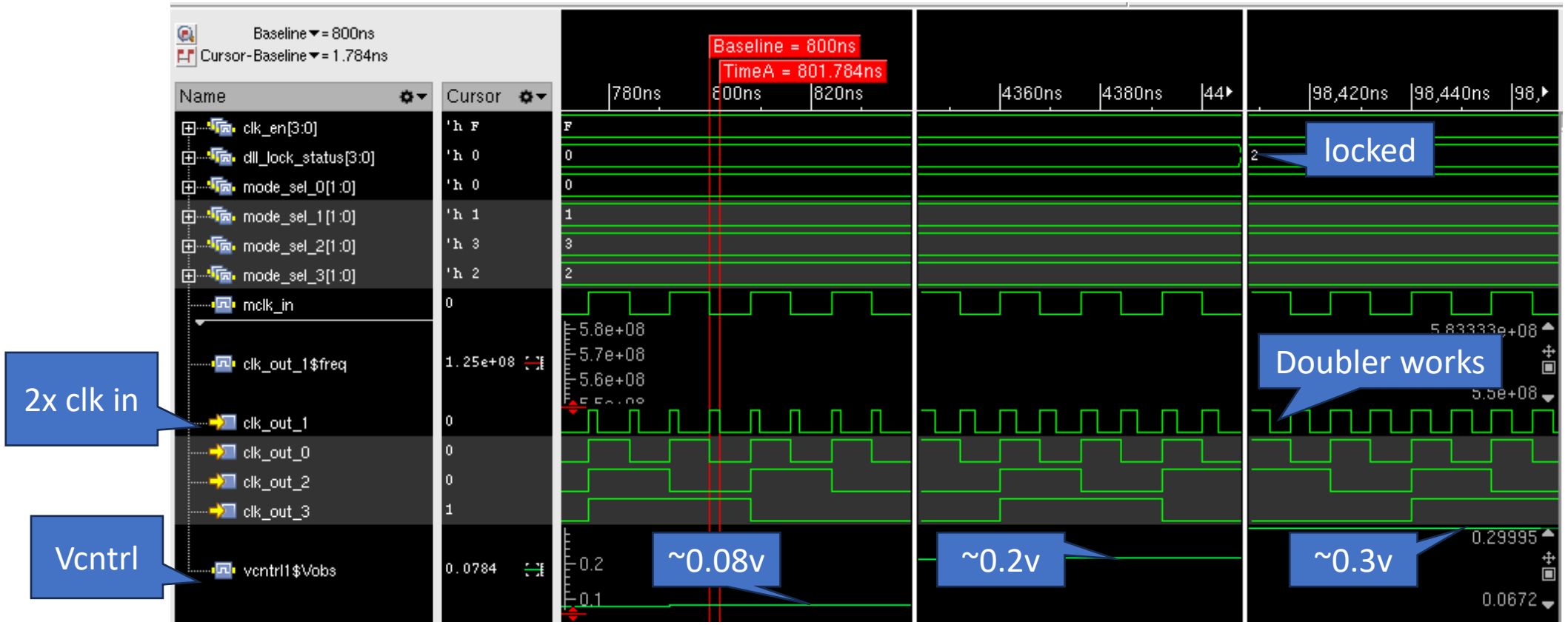
NestedText format – some repeated sections are folded.

```
library: RFDV_scratch
name: simple_uvm_testcase
ports:
    AVDD1P8:
        direction: input
        porttype: power
        supply_nom: 1.8
        supply_type: volts
    AVSS:
        direction: input
        porttype: power
        supply_type: ground
    bg_enable:
        direction: input
        porttype: register
        reg_info:
            name: bg_enable
            blockid: top
            width: 1
            default: 'b0
            on_value: 'b1
            description: bias_blk enable
    clk_en:
        direction: input
        lsb: 0
        msb: 3
        porttype: register
    > reg_info: …
    clk_out_0: …
    clk_out_1: …
    clk_out_2: …
    clk_out_3:
        direction: output
        porttype: digital
        digital_type: clock
    dll_lock_status: …
    mclk_in: …
    mode_sel_0: …
    mode_sel_1: …
    mode_sel_2: …
    mode_sel_3: …
    vcntrl0:
        direction: output
        porttype: analog
        analog_type: dc
    vcntrl1: …
    vcntrl2: …
    vcntrl3: …
tests:
    sanity_test:
        sequences:
            top_on: register
        body: #100us;
```

Simple Data modification here drives UVM file generation.

Example Design Simulation

(Folded) DLL operation at startup locking locked



Results

Effort level of single person – seems to be productive use of time.

- First iteration
  - Initial development of flow, agents, templates and code
  - 3 weeks from dut netlist to first test working.
- Second iteration
  - Simplified and polished the flow
  - Another 3 weeks from DUT netlist to full startup test working.
- Third iteration (and additional)
  - Simple flow reuse with minor tweaks
  - 1 day to build a simple example design and testbench and test
  - (testcase for simulator issue)

Future Plans

- Package code for command line use
- Add results checking
  - Enabling randomization.
- Provide generalized support for dynamic analog signals.

Resources

- Agent and template Code examples at https://github.com/jbdavid-inno/analog-uvm-tb .
- Example DUT with generated files are published at https://github.com/jbdavid-inno/simple_uvm_testcase .
- MiM (designers-guide.com)
- Visual Studio Code(Microsoft)
- Jinja.palletsprojects.com
- Verification Academy
- Google/Bing search
- NestedText.org
- Python.org
- IEEE.org

2024 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES
SAN JOSE, CA, USA
MARCH 4-7, 2024

Questions?

Jonathan's Email: jdavid@innophaseinc.com

Henry's Email: henry@designers-guide.com

accellera SYSTEMS INITIATIVE