2022 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

Accellera UVM-AMS Standard Update

UVM-AMS Working Group

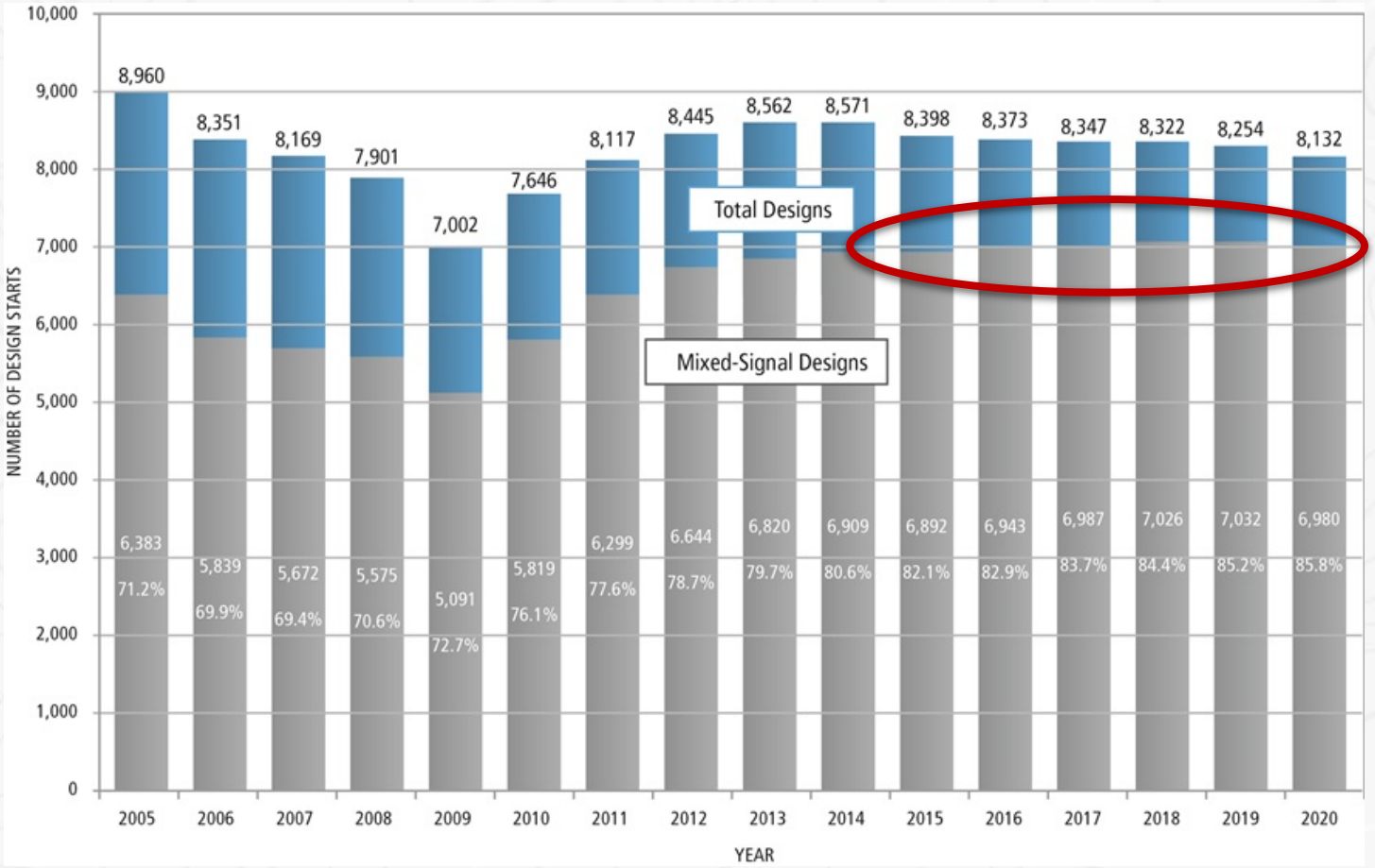
Tom Fitzpatrick, Siemens EDA

Tim Pylant, Cadence Design Systems

accellera SYSTEMS INITIATIVE

Mixed-Signal is Dominant

Need for more advanced, **standard** methodologies for modular, scalable, and reusable coverage-driven mixed-signal verification



85% of design starts are mixed signal

IBS: Mixed-signal design starts as percent of total

Working Group Participants

Terminology

AMS Simulation and Verification

Co-Simulation

Event-Driven

Modular, Scalable, Reusable & Interoperable Verification IP, Stimulus and TB Components

UVM

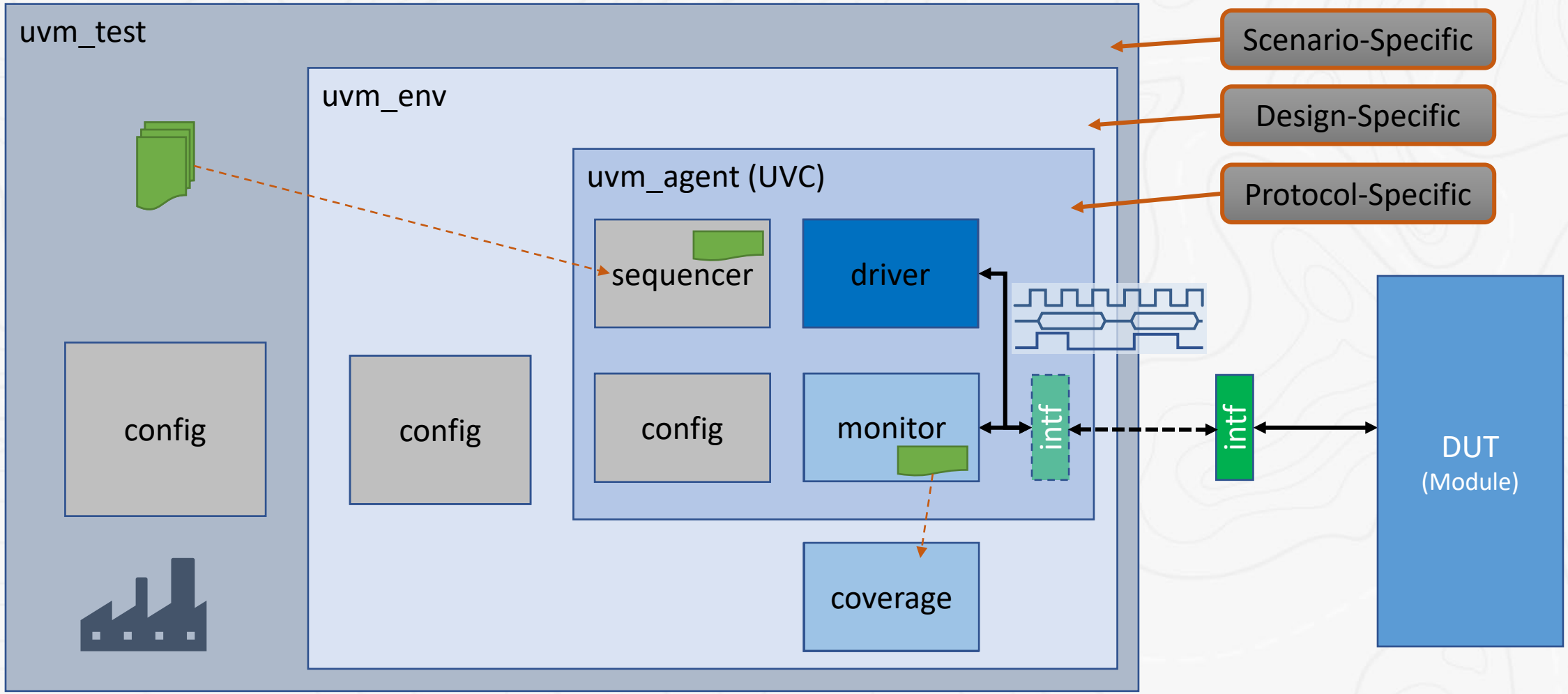
UVM-AMS

Comprehensive Unified AMS Verification Methodology based on UVM

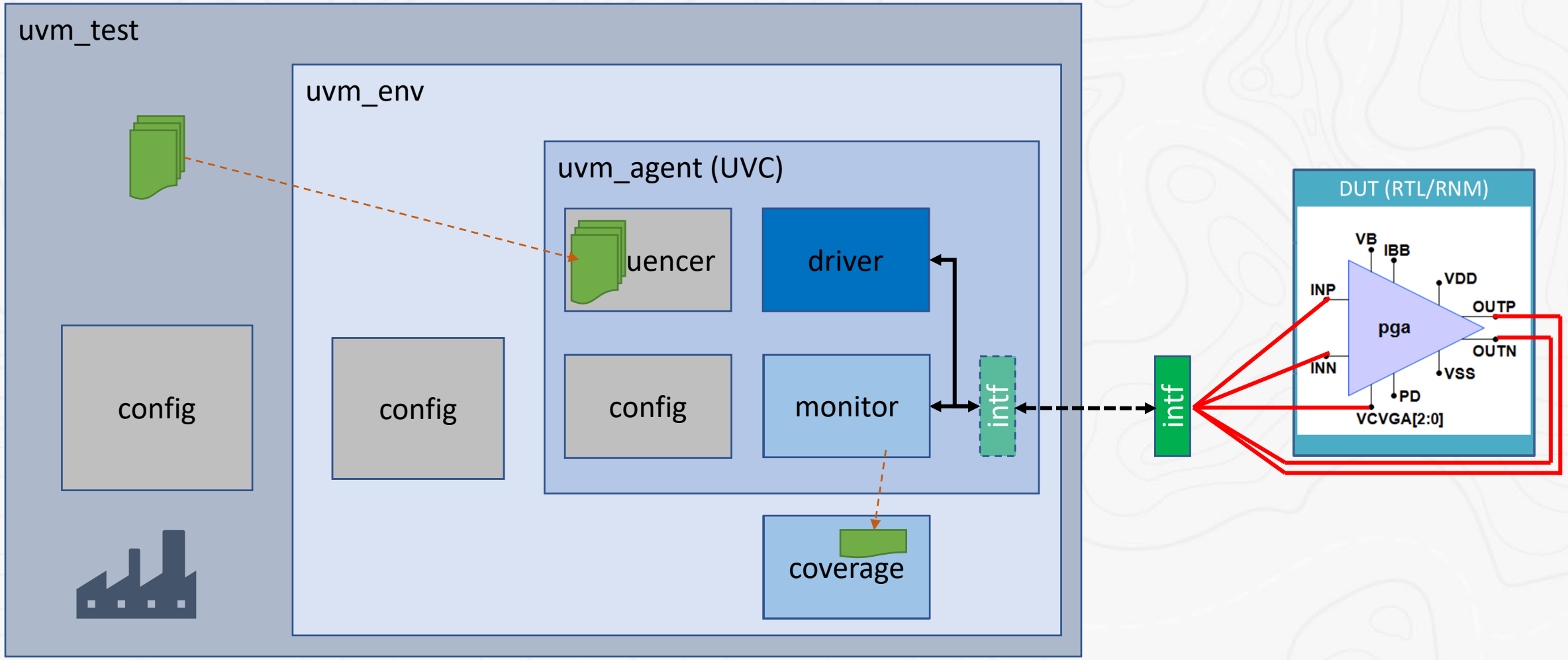
Unified AMS Methodology Based on UVM

- Define a set of class-based extensions to UVM and an accompanying set of components and/or packages in SystemVerilog and/or Verilog-AMS
  - stimulus, tests, sequences, components and analysis functions
- Improve analog/mixed-signal (AMS) and digital mixed-signal (DMS) verification
- Define a framework for the creation of analog/mixed-signal verification components and test benches by introducing extensions to digital centric verification IP
- Based on existing standards
- Promote the use of the UVM-AMS standard among the verification community

UVM Basics

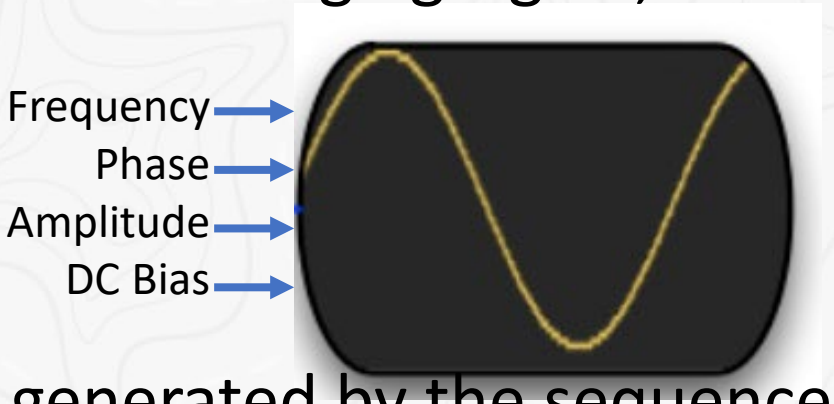

uvm_test
uvm_env
uvm_agent (UVC)
sequencer
driver
config
config
config
monitor
coverage
intf
intf
DUT
(Module)
Scenario-Specific
Design-Specific
Protocol-Specific

UVM for Analog?



Generating and Driving Continuously Changing Analog Signals

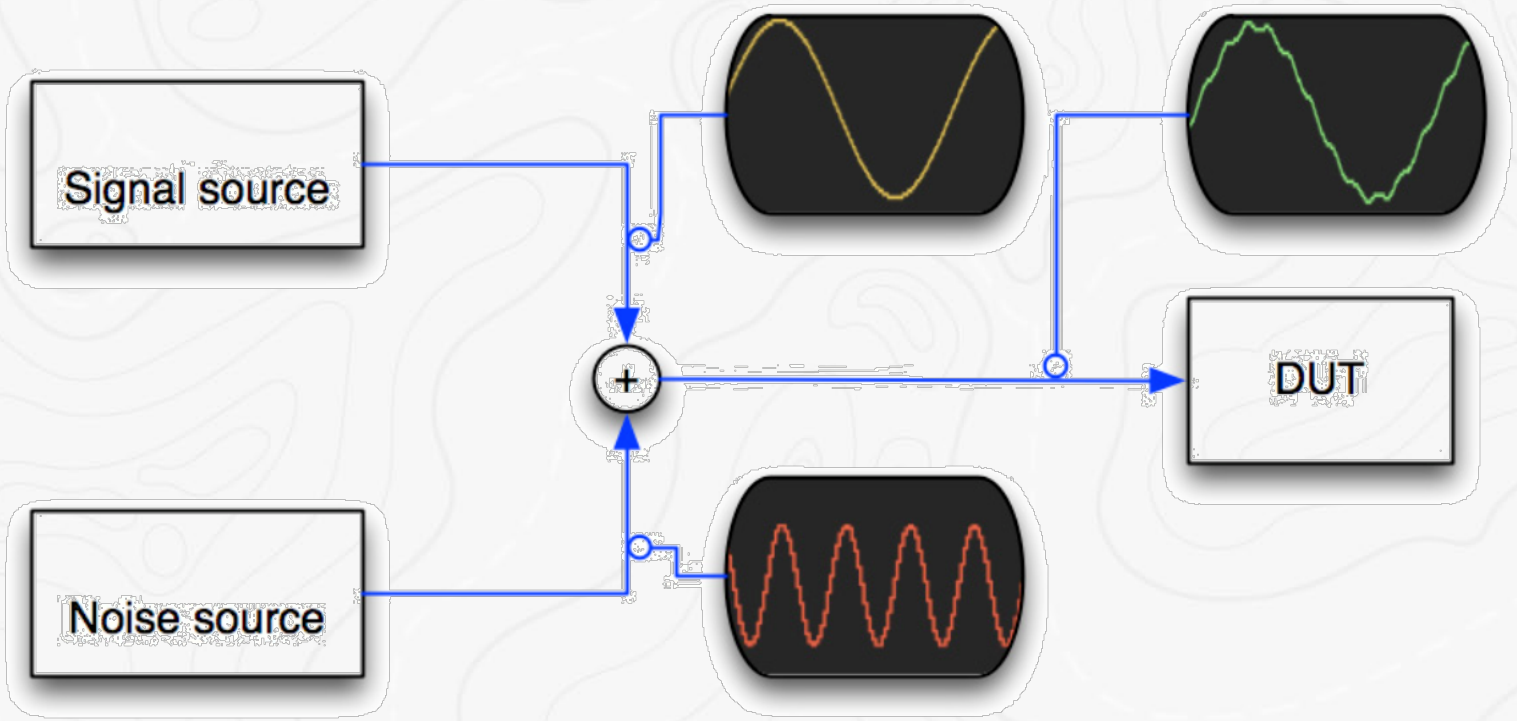
- An analog signal that is not simple DC or a slow changing signal, needs to be a periodic waveform like a sine wave or a sawtooth, or some composition of such sources.



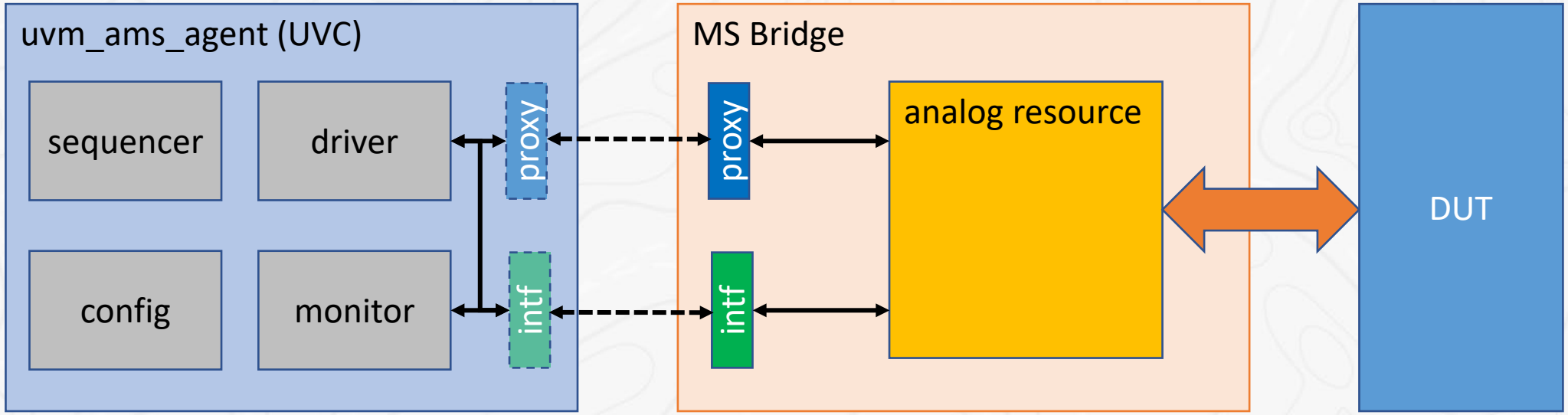
- The properties of the analog signal being driven are controlled by real values, generated by the sequencer
- A UVM sequence_item contains fields for all the control parameters.
- The driver converts the transaction to a setting for the signal generator.
- Sequence items provide simple control interface for the test writer, making it easy to control the generated signal on the fly.

Multiple Signal Generators

- Ability to combine several signal generators to achieve complex input signals. Modulation and noise injection are common cases

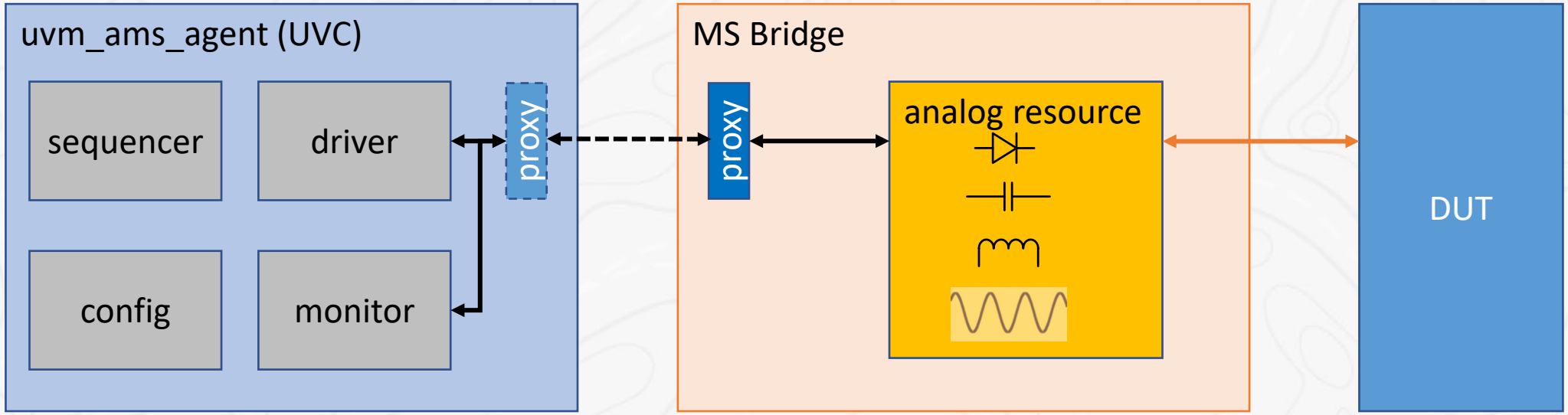


Overall UVM-AMS Methodology



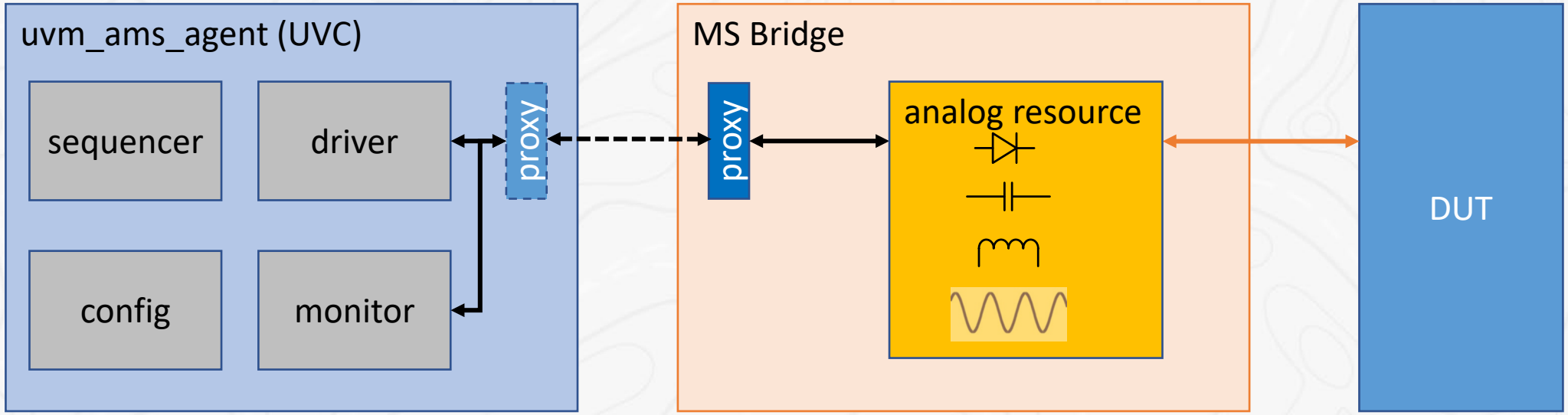
- MS Bridge is the proposed layer that sits between the UVC and the (A)MS DUT
- MS Bridge is a SV module that consists of a proxy API, SV interface, and an analog resource module
- The ‘proxy’ is an API that conveys analog attributes between the UVC and the MS Bridge
- The SV ‘intf’ passes digital/discrete signal values (logic, real, nettype/RNM) between UVC and MS Bridge
- Both ‘proxy’ and ‘intf’ can be used together or individually
- The analog resource (SV, Verilog or Verilog-AMS)
  - Communication layer between intf/proxy and the ports of DUT
  - Uses the analog attributes from proxy to generate continuously changing values (e.g. ramping voltage supply, electrically modeling drive strengths or cap/res loading, etc.)

UVM-AMS Analog Resource



- MS testbench may require the behavior and presence of analog components that a typical UVM-RTL testbench could not include. These could be:
  - Capacitors, Resistors, Inductors, Diodes, current/voltage sources etc. Or a complex passive network for multiple DUT pins.
  - A piece of Verilog-AMS code
  - Such components will be used to model the analog behavior of PADs, lossy transmission lines, loads/impedances, or any other voltage/current conditioning required to accurately model the signals connecting to the ports of DUT
  - Those components can be placed inside the analog resource to be controlled by proxy.

UVM-AMS Analog Resource (cont.)



- Proxy is an API used to interact with analog resource to perform the following
  - Pull electrical values such as voltage, current, component values.
  - Push values such as capacitance value.
  - Event generation
  - Arbitrary sampling of a continuous signal to update a variable in the proxy.
- The analog resource would have the same number of ports as the DUT for a one-to-one connectivity between the ports of analog resource and the DUT
- The API between the proxy and the analog resource must be written using Verilog-AMS language constructs which supports all analog resource views (VAMS, SV, etc.)

Proxy “hook-up”

Proxy Template (API)

```
virtual class pga_bridge_proxy;
…
  pure virtual function void setCapacitance(…);
  pure virtual function real getCapacitance(…);
…
endclass
```

Implement

Proxy instance in MS Bridge module

```
module pga_bridge(…);
…
pga_bridge_core #(…) core (…); // AMS model
…
class proxy extends pga_bridge_proxy;
  function void config_wave(input real ampl, bias, phase, freq);
    core.ampl_in  = ampl;
    core.bias_in  = bias;
    core.phase_in = phase;
    core.freq_in  = freq;
  endfunction
endclass

proxy bridge_proxy = new();
…
endmodule
```

Instance of analog resource

Passes values to analog resource to “program” waveform

UVM config setting

```
module tb;
…
  pga_bridge pga_bridge(.OUTP, .OUTN, .INP, .INN, .VCVGA, .VB, .IBB, .VDD, .VSS, .PD);
…
  initial begin
    uvm_config_db#(pga_bridge_proxy)::set(null,"*pga_uvc*","bridge_proxy",pga_bridge.bridge_proxy);
    run_test();
  end
endmodule
```

Proxy ←→ Analog Resource

MS Bridge

```
class proxy … ;
  function void config_wave(…);
    core.ampl_in  = ampl;
    core.bias_in  = bias;
    core.phase_in = phase;
    core.freq_in  = freq;
  endfunction
  function void get_measure(...);
    ampl  = core.ampl_out;
    bias  = core.bias_out;
    phase = core.phase_out;
    freq  = core.freq_out;
  endfunction

  real min, max;
```

Push

Pull

Monitored

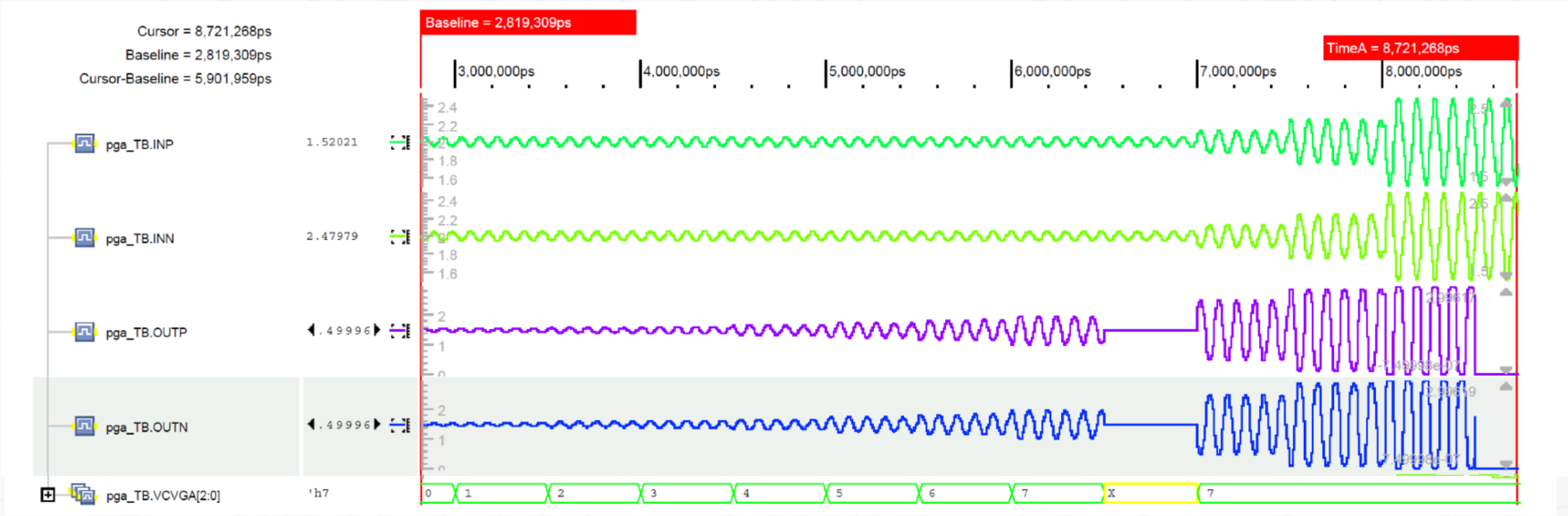
```
assign bridge_proxy.min = core.min_a;
assign bridge_proxy.max = core.max_a;
```

```
pga_bridge_core (...);
  ...
  real ampl_in;
  real bias_in;
  real phase_in;
  real freq_in;

  analog begin
    vsin = (ampl_in * sin(`M_TWO_PI * freq_in * $abstime);
    ...
  end

  wreal  ampl_out;
  wreal  bias_out;
  wreal  freq_out;
  wreal  phase_out;

  Vsig = V(sig);
  if (Vsig > max_a)
    max_a = Vsig;
  else if (Vsig < min_a)
    min_a = Vsig;
```

If target is different its seen as a D2A event

Interpolated value

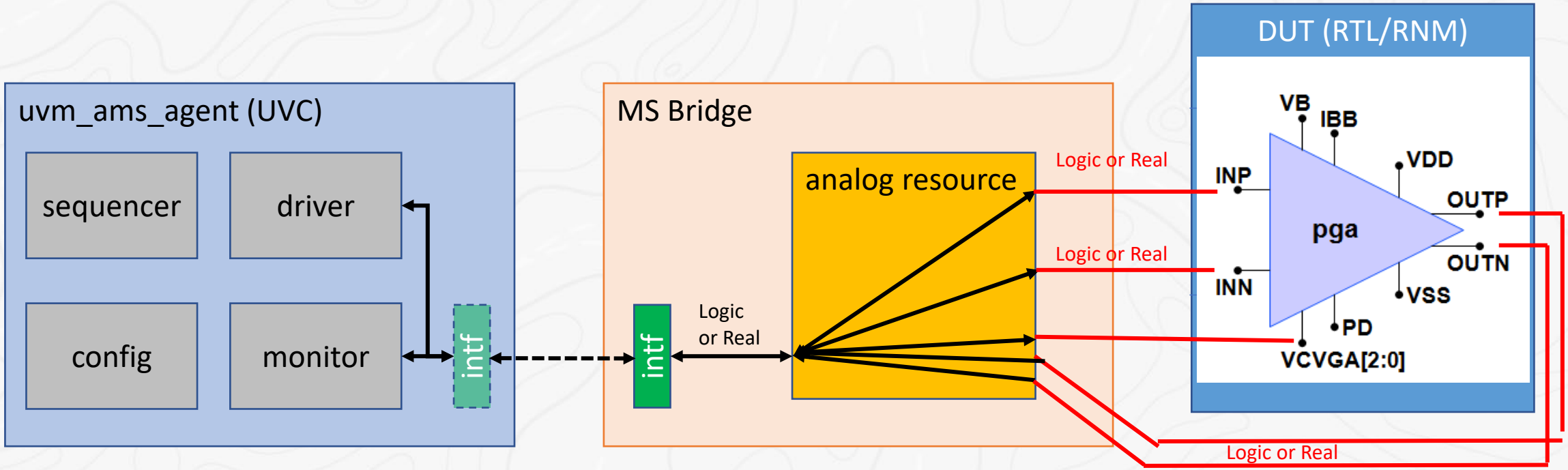
Analog generates update


Cursor = 8,721,268ps
Baseline = 2,819,309ps
Cursor-Baseline = 5,901,959ps
Baseline = 2,819,309ps
TimeA = 8,721,268ps
3,000,000ps
4,000,000ps
5,000,000ps
6,000,000ps
7,000,000ps
8,000,000ps
pga_TB.INP
1.52021
pga_TB.INN
2.47979
pga_TB.OUTP
.49996
pga_TB.OUTN
.49996
pga_TB.VCVGA[2:0]
'h7
0
1
2
3
4
5
6
7
X
7

Model of Programmable Gain Amp Ports in SV

```
module pga import rnm_pkg::*; (
  input  logic[2:0] VCVGA;  // digital control voltage
  input  logic PD;          // powerdown control
  output real_net OUTP,OUTN;// differential output
  input  real_net INP,INN;  // differential input
  input  real_net VB,IBB;   // required bias inputs
  input  real_net VDD,VSS;  // power supplies
);
```

Doubler Example in UVM-AMS (RTL/RNM DUT)

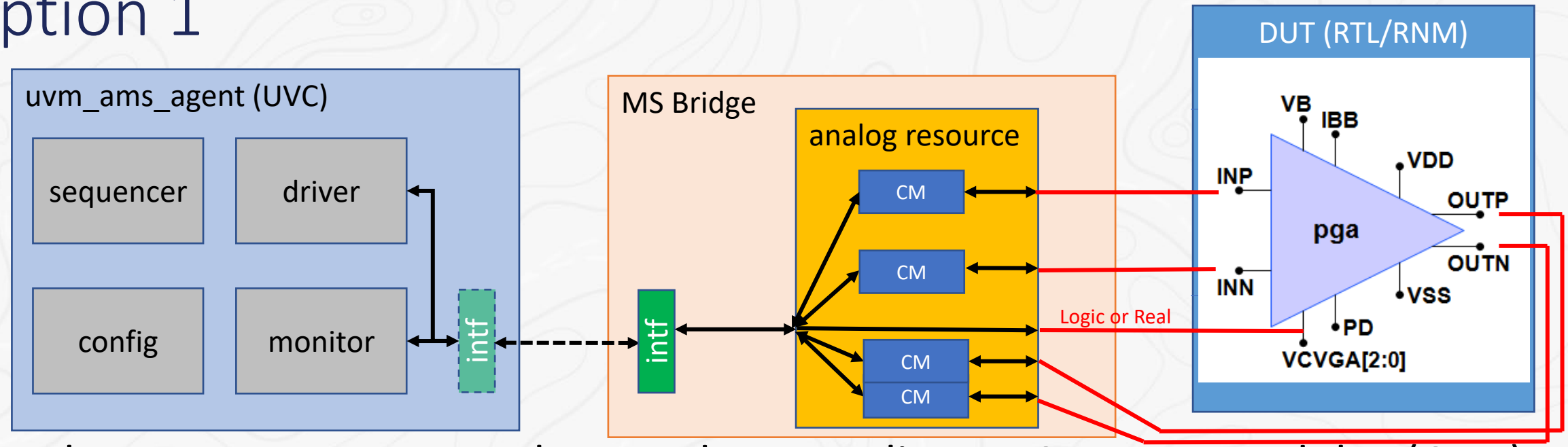


- Analog resource simply acts as pass through between intf and DUT.
  - DUT ports should follow the language rules of the analog resource typically Verilog-AMS.

Model of Programmable Gain Amp Ports in SV

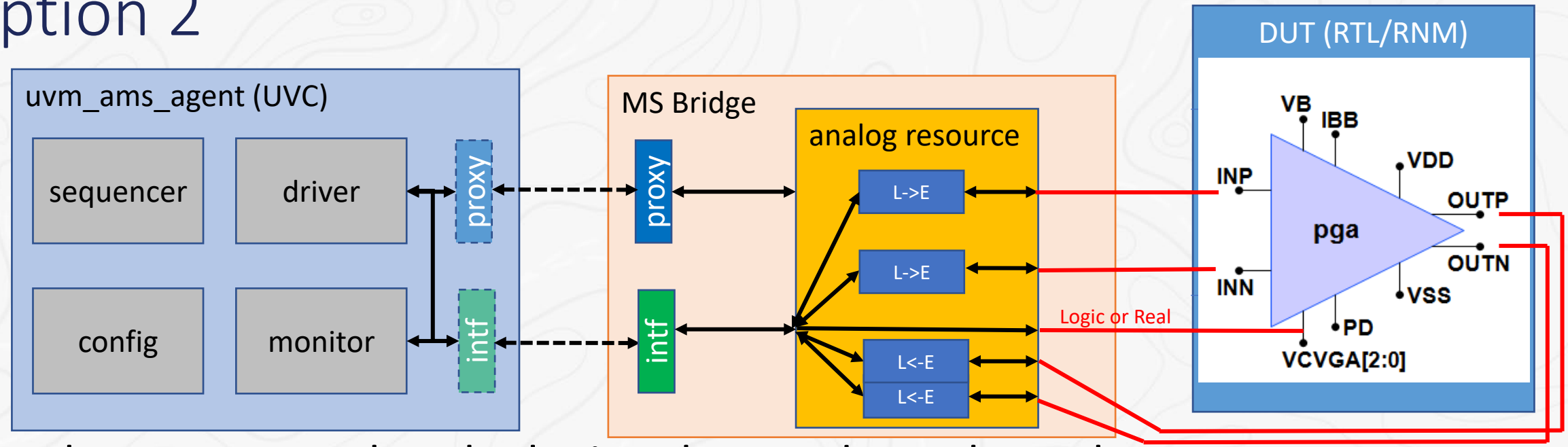
```
module pga ( OUTP,OUTN, INP,INN, VCVGA, VB,IBB, VDD,VSS, PD );
  output OUTP,OUTN;          // differential output
  electrical OUTP,OUTN;
  input INP,INN;             // differential input
  electrical INP,INN;
  input wire [2:0] VCVGA; // digital control voltage
  input VB,IBB;              // required bias inputs
  electrical VB,IBB;
  input VDD,VSS;             // power supplies
  electrical VDD,VSS;
  input wire PD;           // powerdown control
```

Doubler Example in UVM-AMS (VAMS/SPICE DUT) Option 1



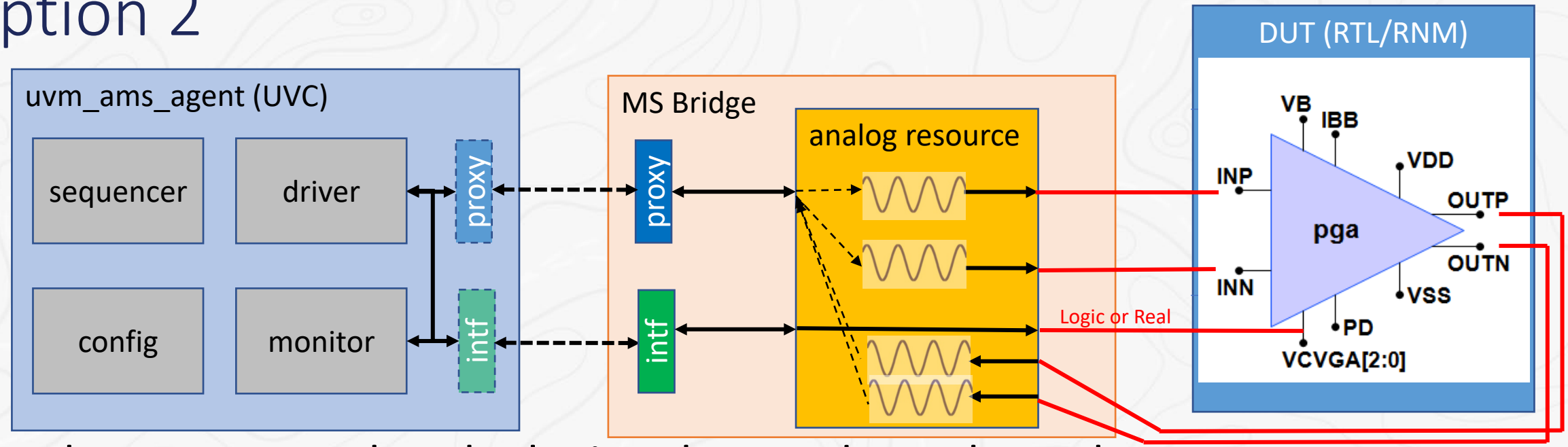
- Analog resource acts as a short and users relies on Connect Modules (CMs) inserted by the simulator to interface between analog and digital domains
  - Simple to use but many non-standard requirements; Supply connection, DRS etc.
  - No fine control on the analog resources ‘electrical’ interface.
  - DUT/CM supplies generated by another MS Bridge or source.

Doubler Example in UVM-AMS (VAMS/SPICE DUT) Option 2



- Analog resource takes the logic values and uses bespoke code L<->E to translate this to the analog signal
  - Proxy used to control analog resource L<->E setup
  - Optionally same UVC/MS Bridge could generate Supplies for DUT/L<->E
  - Optionally dedicated UVC/MS Bridge could generate Supplies for DUT/L<->E

Doubler Example in UVM-AMS (VAMS/SPICE DUT) Option 2

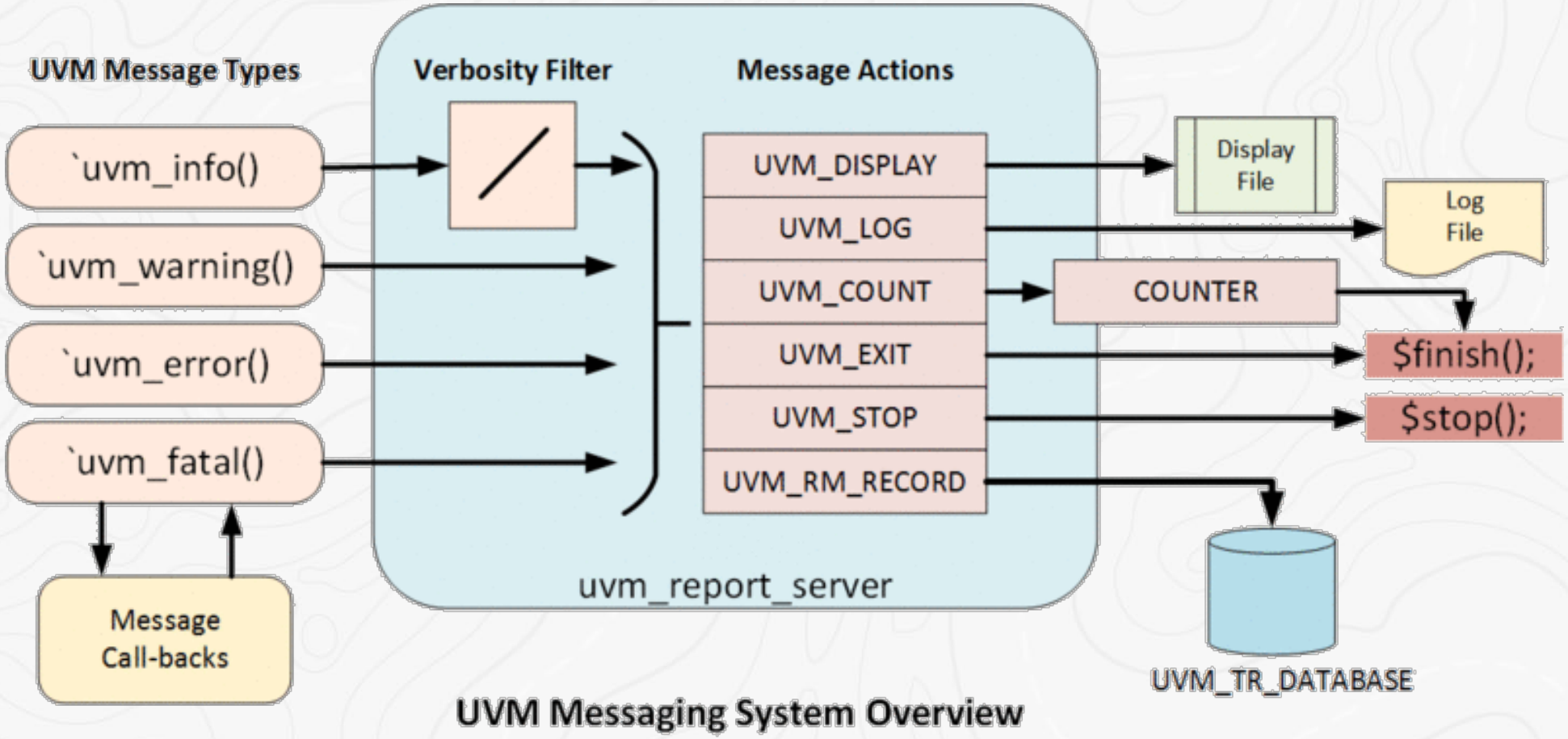


- Analog resource takes the logic values and uses bespoke code L<->E to translate this to the analog signal
  - Proxy used to control analog resource L<->E setup
  - Optionally same UVC/MS Bridge could generate Supplies for DUT/L<->E
  - Optionally dedicated UVC/MS Bridge could generate Supplies for DUT/L<->E

Messages for Debug and Error Reporting

- Debugging activity inside a large environment with many UVCs is critical.
- Need to report:
  - Errors
  - Debug
  - Progress
- Messages need to be categorized via severity:
  - Fatal, Error, Warning, Info
- Need to link actions with messages
  - Stop simulation on fatal or after four errors
  - Summarize number of messages reported
- Need a different mechanism than simulator messages to avoid filtering effects

UVM Messaging System



UVM Messaging System Overview

UVM Messaging from Analog Resource

- UVM Reporting macros not supported in Verilog-AMS modules.
- Take advantage of up-scoping to provide solution. (1364-2001 LRM)
- `include "uvm_ams.vamsh“ in Verilog-AMS file (analog resource)
  - localparams to define UVM Verbosity levels as integers to match UVM enum
- `include "uvm_ams.svh“ in SV file (MS Bridge)
  - Void functions that wrap `uvm_*() reporting macros into functions of the same name
- Within a digital block of a Verilog-AMS file users call; uvm_[info|warning|error|fatal](…)
  - Up scoping means it find the function in the MS Bridge file
- Within analog block, many solutions so here is one (calling of digital functions not allowed)
  - Set string value and toggle integer
  - Use absdelta to trigger on toggle and read string to call up-scoping function

UVM Message – Analog block

VAMS

```
localparam string uvm_path = $sformat(uvm_path,”%m”);
localparam string message = $sformat("The Current is above the threshold @ %eA",I_PLUS);
uvm_info(P__TYPE,message,UVM_MEDIUM,uvm_path);
```

SV Bridge

```
function void uvm_info(string id, string message, int verbosity_level, string uvm_path);
`uvm_info_context(id,message,uvm_verbosity'(verbosity_level),uvm_root::get().find(uvm_path))
endfunction: uvm_info
```

- Hold UVM component hierarchy path string in proxy class via get_full_name()
- Use *_context reporting macros to direct message to relevant component

```
UVM_INFO ../../include/uvm_ams.svh(26) @ 52001.098068ns: uvm_test_top.env.v_agent [i_bridge]
The Current is above the threshold @ 1.178812e+00A
```

FAQs & Wrap-Up

- What value will the UVM-AMS standard bring to the community?
- The UVM-AMS WG envisions the availability of an industry-agreed analog/mixed-signal verification methodology based on its planned UVM-AMS standard. This will encourage support by tool and IP providers, offering ready-to-use analog/mixed-signal verification IP that can be integrated easily into a UVM-AMS testbench. It will raise the productivity and quality of analog/mixed-signal verification across projects and applications, thanks to the reuse of proven verification components and stimuli.

FAQs & Wrap-Up

- I currently use Interface Verification IP which only supports digital signals. Should I replace them with AMS Verification IP?
- The objective of the UVM-AMS standard is to introduce dedicated capabilities that enable making AMS extensions to digital-centric verification IP. As UVM offers the foundation technology, we could benefit from using testbench configuration, factory-based component overrides, and virtual interfaces to seamlessly insert AMS components, signals and analysis in an existing digital-centric verification IP. Obviously, this extension heavily depends on the flexibility and configurability offered in the digital VIP itself.

FAQs & Wrap-Up

- Will the UVM-AMS Working Group develop specific AMS verification IP?
- The primary objective of the UVM-AMS Working Group is to develop a standard that forms the framework to develop AMS verification components or testbenches. As such, the working group itself will not develop these elements. The internal functionality to drive, monitor, or analyze AMS signals should be developed by the verification team. However, to explain how to create an AMS verification IP or testbenches to the AMS verification community, it is expected that there will be basic examples available as part of the deployment of the UVM-AMS standard.

Conclusions

- There is a need for more advanced, standard methodologies for scalable, reusable and metric-driven mixed-signal (AMS/DMS) verification
- The UVM-AMS proposal addresses the gaps in current verification methodology standards
- Extend UVM class-based approach to seamlessly support the module-based approach (MS Bridge) needed for mixed-signal verification
  - Targeting analog/mixed-signal contents (RNM, electrical/SPICE)
  - Application and extension of existing UVM concepts and components
    - Sequencer, Driver, Monitor
    - MS Bridge / Analog resources
    - UVM Messaging System

2022 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

Questions?

accellera SYSTEMS INITIATIVE