

# SystemVerilog Assertions Verification with SVAUnit

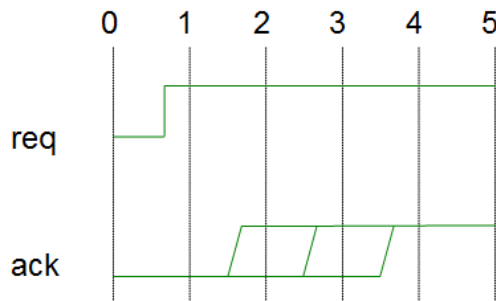
Ionuț Ciocîrlan  
Andra Radu

AMIQ Consulting

# Agenda

1. SystemVerilog Assertions (SVAs)
2. About SVAUnit
3. SVAUnit example
4. SVAUnit Infrastructure
5. Conclusions
6. Q&A

# SystemVerilog Assertions (SVAs)



After the rise of request signal, the acknowledge signal should be asserted no later than 3 clocks cycles.

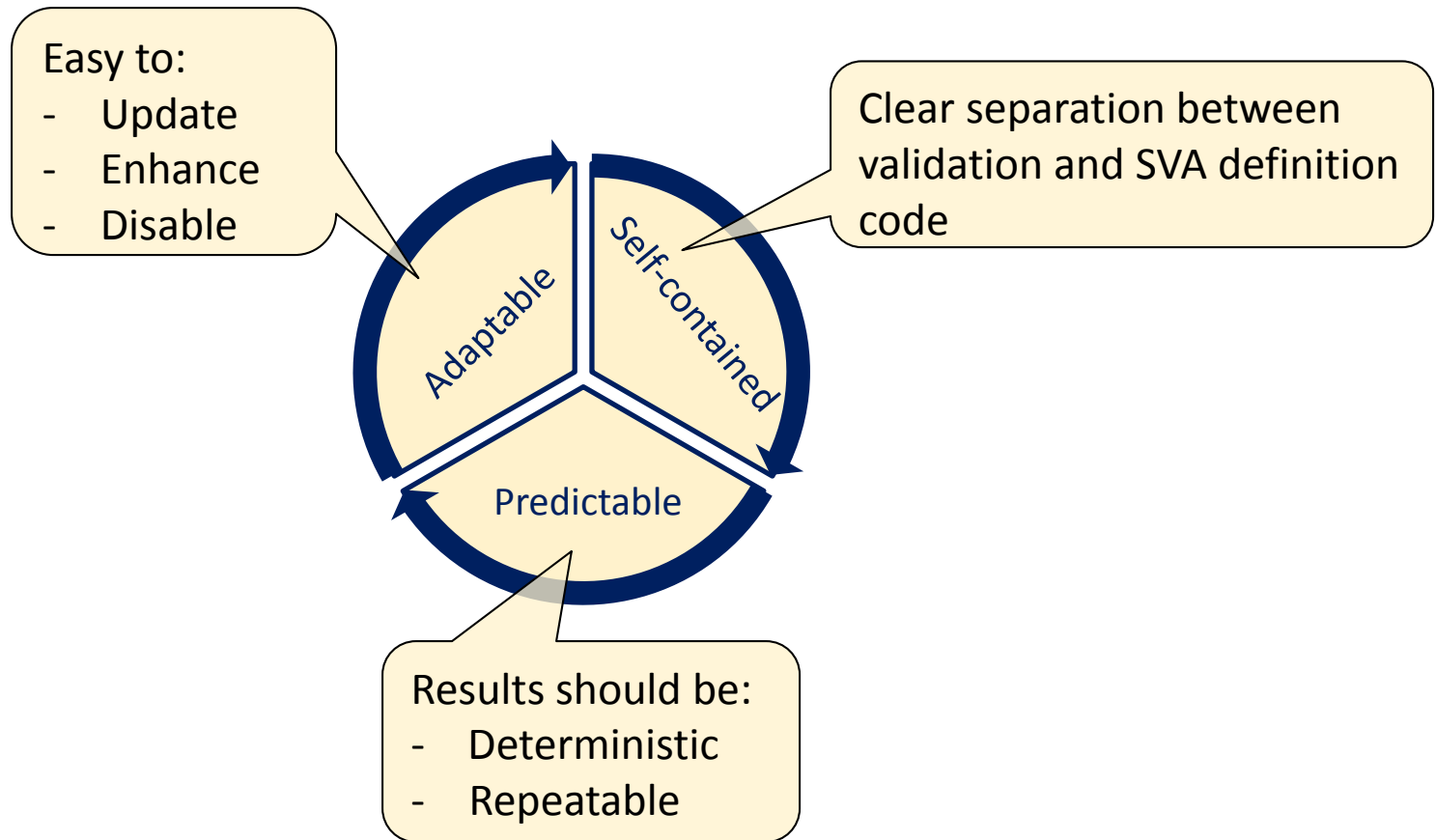
- What is an assertion?

A check against the specification of a design that it never violates.

- Why use SVA?

Powerful feature, flexible, measurable and with a concise syntax.

# SVA Verification Challenges

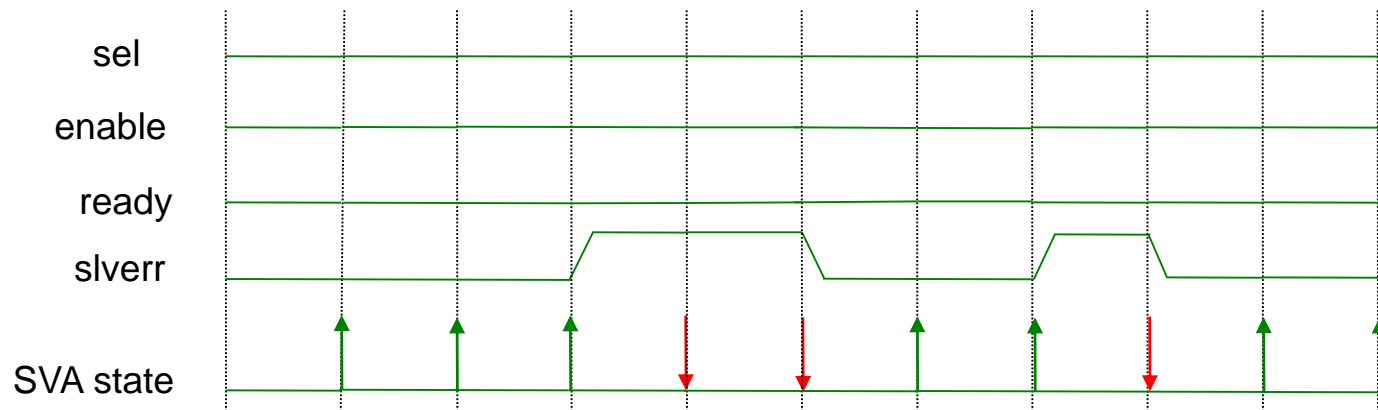


# Introducing SVAUnit

- Structured framework for Unit Testing for SVAs
- Allows the user to decouple the SVA definition from its validation code
- UVM compliant package written in SystemVerilog
- Encapsulate each SVA testing scenario inside an unit test
- Easily controlled and supervised using a simple API

# Hands-on example - What to check

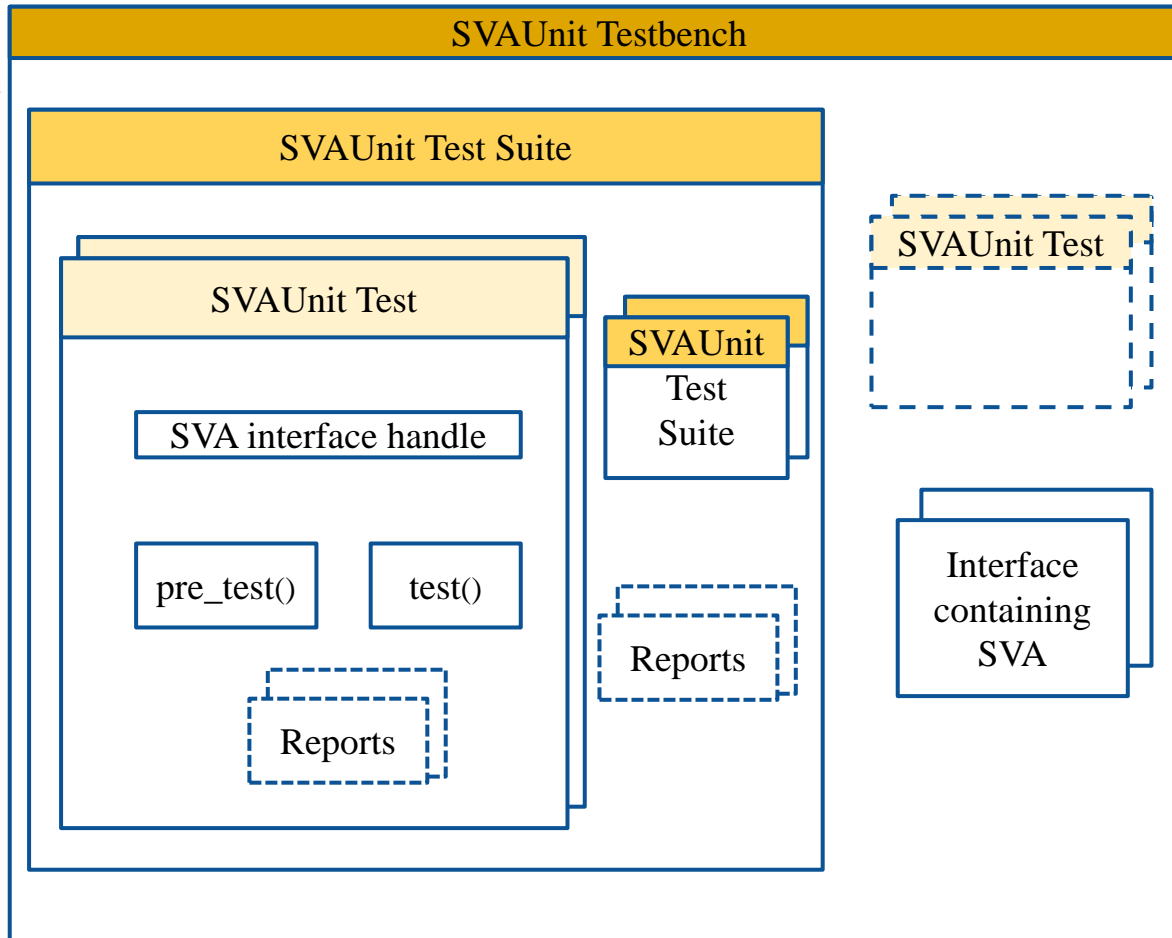
“slverr signal should be 0 if no slave is selected or when transfer is not enabled or when slave is not ready to respond”



```
interface an_if (input clk);  
  property an_sva_property;  
    @(posedge clk)  
    !sel or !enable or !ready |-> !slverr;  
  endproperty  
  AN_SVA: assert property (an_sva_property) else  
    `uvm_error("an_sva_property", "AN_SVA failed")  
endinterface
```

↑ - SVA succeeded  
↓ - SVA failed

# SVAUnit Environment Architecture



- **SVAUnit Testbench**
  - Enables SVAUnit
  - Instantiates SVA interface
  - Starts test
- **SVAUnit Test**
  - Contains the SVA scenario
- **SVAUnit Test Suite**
  - Test and test suite container

# Example of SVAUnit Testbench

```
module top;
  // Instantiate the SVAUnit framework
  `SVAUNIT_UTILS

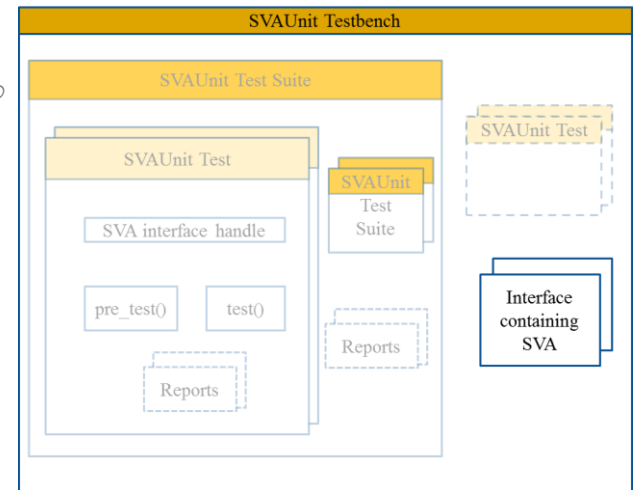
  ...

  // Instantiate the interface with the SVAs we want to test
  an_if dut_if(.clk(clock));

  initial begin
    // Register the interface with the uvm_config_db
    uvm_config_db#(virtual an_if)::
      set(uvm_root::get(), "*", "VIF", dut_if);

    // Start the scenarios
    run_test();
  end

  ...
endmodule
```





# Example of SVAUnit Test

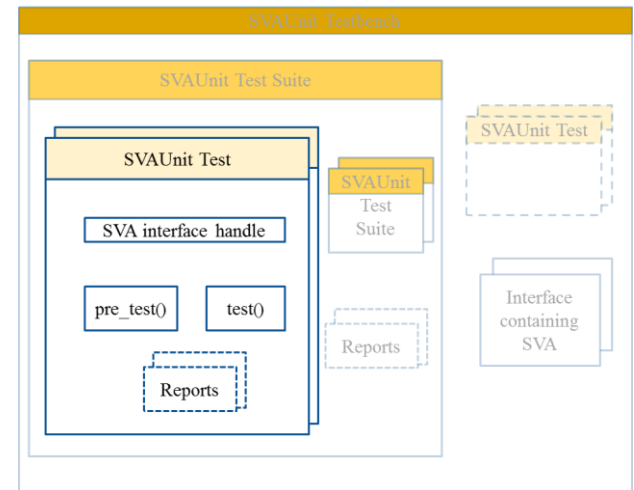
```
class ut1 extends svaunit_test;
  // The virtual interface used to drive the signals
  virtual an_if vif;

  function void build_phase(input uvm_phase phase);
    // Retrieve the interface handle from the uvm_config_db
    if (!uvm_config_db#(virtual an_if)::get(this, "", "vif", vif))
      `uvm_fatal("UT1_NO_VIF_ERR", "SVA interface is not set!")

    // Test will run by default;
    disable_test();
  endfunction

  task pre_test();
    // Initialize signals
  endtask

  task test();
    // Create scenarios for AN_SVA
  endtask
endclass
```



# Example of test() method

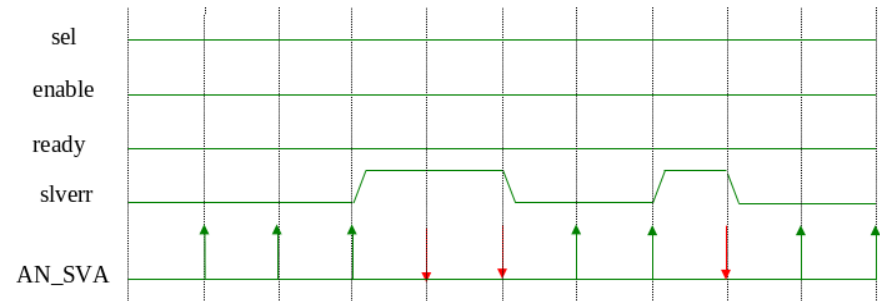
```
task test();
    // Create scenarios for AN_SVA
    disable_all_assertions();
    enable_assertion("AN_SVA");

    repeat(3) begin
        @(posedge vif.clk);
        fail_if_sva_not_succeeded("AN_SVA", "The assertion should have succeeded");
    end

    // Trigger the error scenario
    vif.slverr = 1'b1;

    repeat(2) begin
        @(posedge vif.clk);
        fail_if_sva_succeeded("AN_SVA", "The assertion should have failed");
    end

    vif.slverr = 1'b0;
    ...
endtask
```

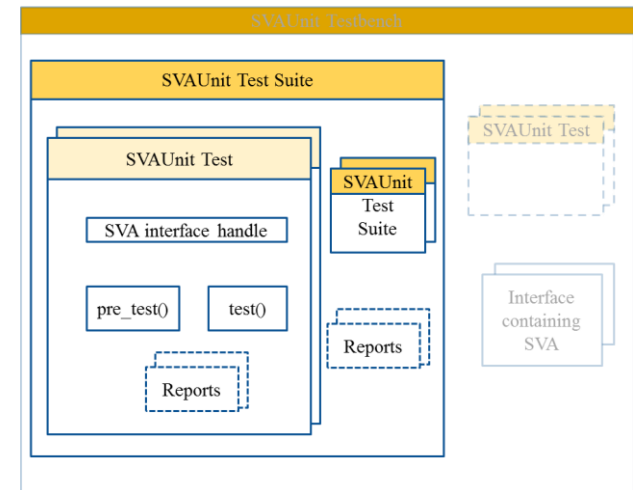


# Example of SVAUnit Test Suite

```
class uts extends svaunit_test_suite;
  // Instantiate the SVAUnit tests
  ut1 ut1;
  ...
  ut10 ut10;

function void build_phase(input uvm_phase phase);
  ut1 = ut1::type_id::create("ut1", this);
  ...
  ut10 = ut10::type_id::create("ut10", this);

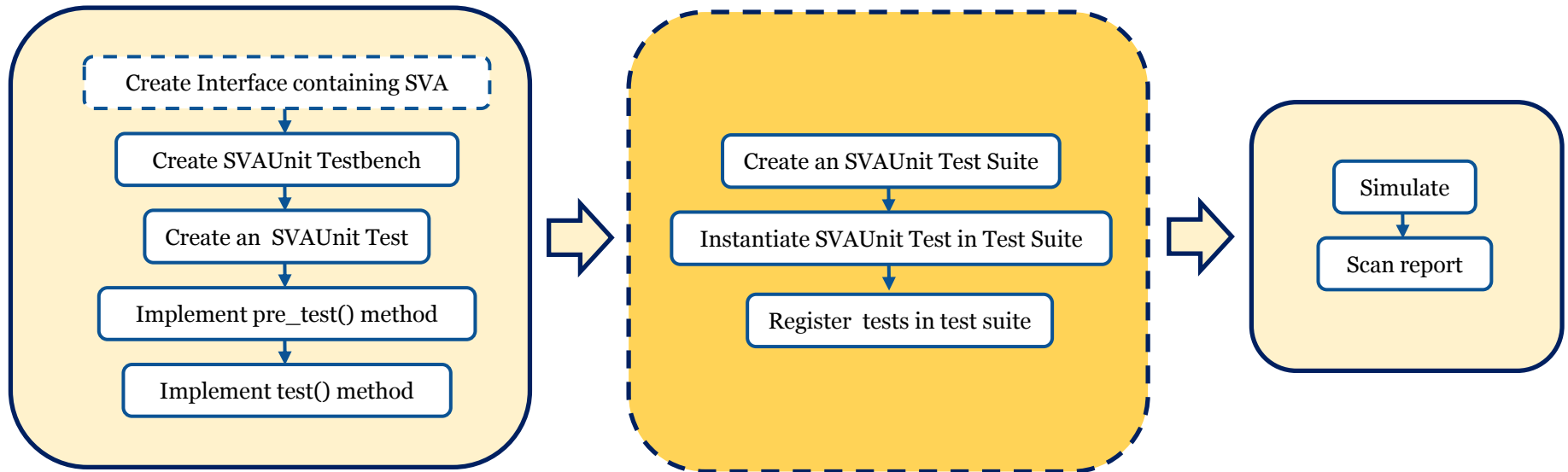
  // Register tests in suite
  add_test(ut1);
  ...
  add_test(ut10);
endfunction
endclass
```



# SVAUnit Test API

Control	<ul style="list-style-type: none"><li>• <code>disable_all_assertions();</code></li><li>• <code>enable_assertion(sva_name);</code></li><li>• <code>enable_all_assertions();</code></li><li>• . . .</li></ul>
Check	<ul style="list-style-type: none"><li>• <code>fail_if_sva_does_not_exists(sva_name, error_msg);</code></li><li>• <code>pass_if_sva_not_succeeded(sva_name, error_msg);</code></li><li>• <code>pass/fail_if(expression, error_msg);</code></li><li>• . . .</li></ul>
Report	<ul style="list-style-type: none"><li>• <code>print_status();</code></li><li>• <code>print_sva();</code></li><li>• <code>print_report();</code></li><li>• . . .</li></ul>

# SVAUnit Flow



# Error reporting

Name of SVAUnit check

SVAUnit test path

```
UVM_ERROR @ 55000 ns [SVAUNIT_FAIL_IF_SVA_SUCCEEDED_ERR]: [x_z_suite.addr_x_z_test::x_z_addr_ut  
AMIQ_APB_ILLEGAL_ADDR_VALUE_ERR] The assertion should have failed
```

Name of SVA under test

Custom error message

# Hierarchy report

```
UVM_INFO @ 56000 ns [protocol_ts]:  
  protocol_ts  
    protocol_ts.protocol_test1  
    protocol_ts.protocol_test2  
    protocol_ts.x_z_suite  
      x_z_suite.addr_x_z_test  
      x_z_suite.slverr_x_z_test  
      x_z_suite.sel_x_z_test  
      x_z_suite.write_x_z_test  
      x_z_suite.strb_x_z_test  
      x_z_suite.prot_x_z_test  
      x_z_suite.enable_x_z_test  
      x_z_suite.ready_x_z_test
```

# Test scenarios exercised

```
----- protocol_ts test suite : Status statistics -----  
  
* protocol_ts FAIL (2/3 test cases PASSED)  
  * protocol_ts.x_z_suite FAIL (0/8 test cases PASSED)  
    protocol_ts.protocol_test2 PASS (13/13 assertions PASSED)  
    protocol_ts.protocol_test1 PASS (13/13 assertions PASSED)  
  
UVM_INFO @ 56000 ns [protocol_ts]:  
  
    3/3 Tests ran during simulation  
  
        protocol_ts.x_z_suite  
        protocol_ts.protocol_test2  
        protocol_ts.protocol_test1
```



# SVA and checks exercised

```
----- protocol_ts test suite : SVA and checks statistics -----  
  
  AMIQ_APB_ILLEGAL_SEL_TRANSITION_TR_PHASES_ERR   13/13 checks PASSED  
    SVAUNIT_FAIL_IF_SVA_SUCCEEDED_ERR 1/1 times PASSED  
    SVAUNIT_FAIL_IF_SVA_NOT_SUCCEEDED_ERR 2/2 times PASSED  
    SVAUNIT_FAIL_IF_SVA_DOES_NOT_EXISTS_ERR 7/7 times PASSED  
    SVAUNIT_PASS_IF_SVA_IS_ENABLE_ERR 3/3 times PASSED  
  
  AMIQ_APB_ILLEGAL_SEL_TRANSITION_DURING_TRANSFER_ERR 13/13 checks PASSED  
    SVAUNIT_FAIL_IF_SVA_NOT_SUCCEEDED_ERR 1/1 times PASSED  
    SVAUNIT_FAIL_IF_SVA_SUCCEEDED_ERR 2/2 times PASSED  
    SVAUNIT_FAIL_IF_SVA_DOES_NOT_EXISTS_ERR 7/7 times PASSED  
    SVAUNIT_PASS_IF_SVA_IS_ENABLE_ERR 3/3 times PASSED
```

# Tools integration

The screenshot displays three overlapping windows from Cadence's simulation ecosystem:

- Console - SimVision:** Shows a text-based log of simulation results. Key messages include: "1 of 1 Tests ran during simulation", "1 of 1 SVA were tested into uts", and "4 of 20 Checks were used". It also lists several SVA (System Verilog Assertions) that failed or passed.
- Waveform 1 - SimVision:** A timing diagram showing signals like 'clk', 'slvrr', 'sel', 'enable', and 'ready' over time. A cursor is positioned at 21,000ps. A red box highlights a failure event at 21,000ps.
- Incisive Enterprise Manager:** A management interface showing a "Sessions Table" with two sessions. The table includes columns for Session Status, Session Name, Progress, and Total.

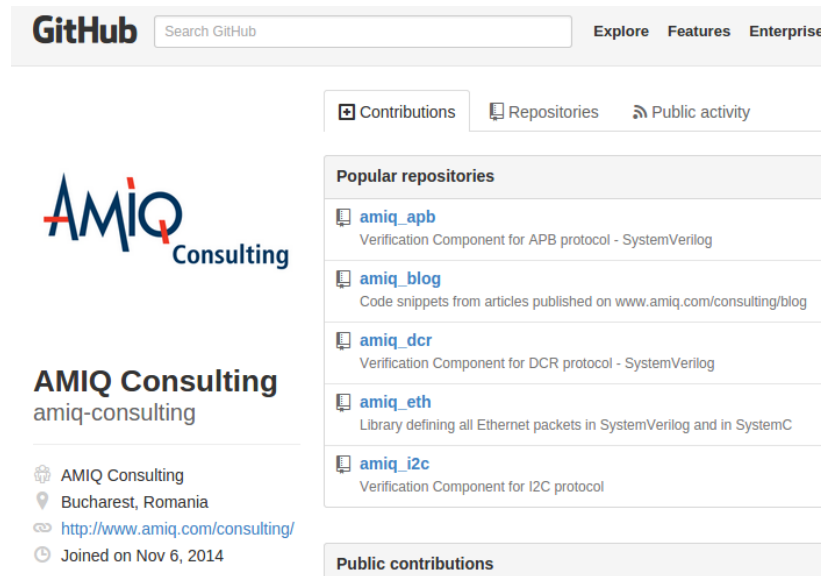
Session Status	Session Name	Progress	P	F	R	W	O	Total
0	apb_ms_ex_req.andra.socianu.15_03_03_13_57_36_2607	4.167%	2	133%	0	0	0	6.1100%
1	apb_ms_ex_req.andra.socianu.15_03_03_14_03_41_3967	2.140%	2	160%	0	0	0	5.1100%

- Simulation tools:
  - All major simulators
- Regression simulation tools:
  - VManager

# Conclusions

- SVAUnit decouples the checking logic from SVA definition code
- Safety net for eventual code refactoring
- Can also be used as self-checking documentation on how SVAs work
- Quick learning curve
- Easy-to-use and flexible API
- Speeds up verification closure
- Boosts verification quality

# Availability



**GitHub** Search GitHub Explore Features Enterprise

Contributions Repositories Public activity

**AMIQ Consulting**

**AMIQ Consulting**  
amiq-consulting

AMIQ Consulting  
Bucharest, Romania  
<http://www.amiq.com/consulting/>  
Joined on Nov 6, 2014

**Popular repositories**

- amiq\_apb**  
Verification Component for APB protocol - SystemVerilog
- amiq\_blog**  
Code snippets from articles published on [www.amiq.com/consulting/blog](http://www.amiq.com/consulting/blog)
- amiq\_dcr**  
Verification Component for DCR protocol - SystemVerilog
- amiq\_eth**  
Library defining all Ethernet packets in SystemVerilog and in SystemC
- amiq\_i2c**  
Verification Component for I2C protocol

**Public contributions**

- SVAUnit is an open-source package released by AMIQ Consulting
- We provide:
  - SystemVerilog and simulator integration code
  - AMBA-APB assertion package
  - Code templates and examples
  - HTML documentation for API

<https://github.com/amiq-consulting/svaunit>

Q&A

?

Thank you!