A Cross-domain Heterogeneous ABV-Library for Mixed-signal Virtual Prototypes in SystemC/AMS

Muhammad Hassan[1] (muhammad.hassan@dfki.de)

Thilo Vörtler[2], Karsten Einwich[2]

Rolf Drechsler[1,3]

Daniel Große[1,4]

[1]Cyber-Physical Systems, DFKI GmbH, Bremen, Germany

[2]COSEDA Technologies GmbH, Dresden, Germany

[3]Institute of Computer Science, University of Bremen, Bremen, Germany

[4]Institute of Complex Systems, Johannes Kepler University, Linz, Germany

AUTOASSERT

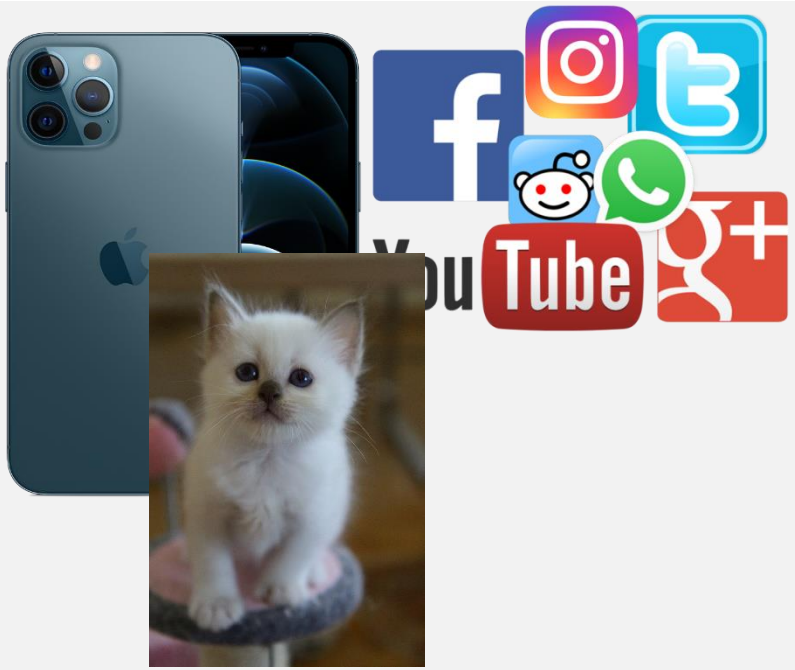
Bundesministerium für Bildung und Forschung

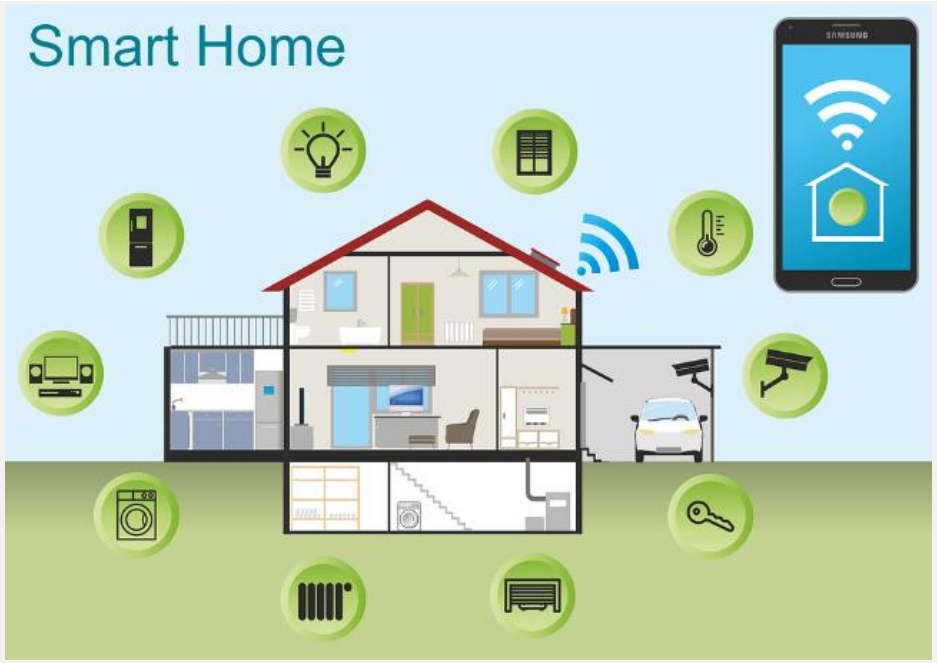
16ME0117

Agenda

- Motivation
- Challenges
- Contribution
- Conclusion and outlook

Smart Devices





Software

WiFi
Sensors
4G/5G

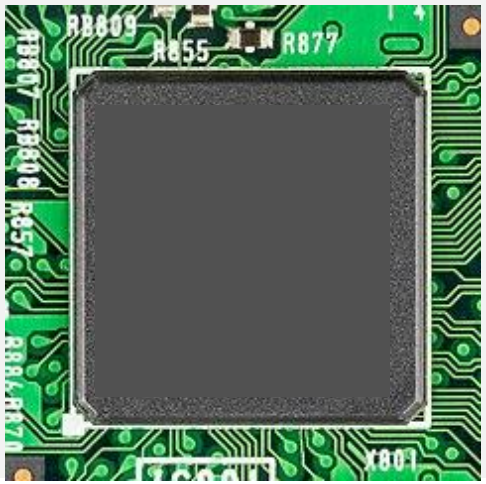
Heterogeneous systems

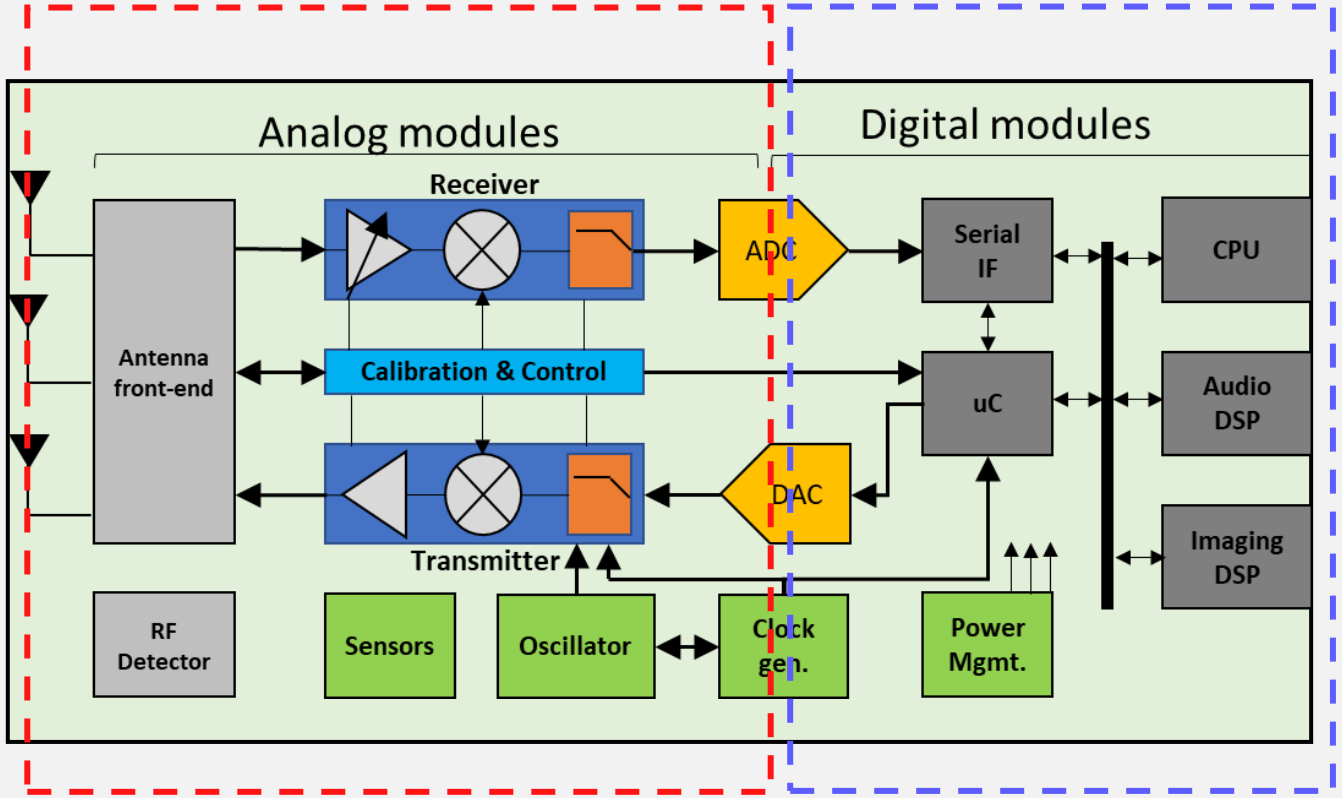
Analog + Digital

Processors
Accelerators
Memories

Heterogeneous System on Chip

- Single IC
- Feature rich

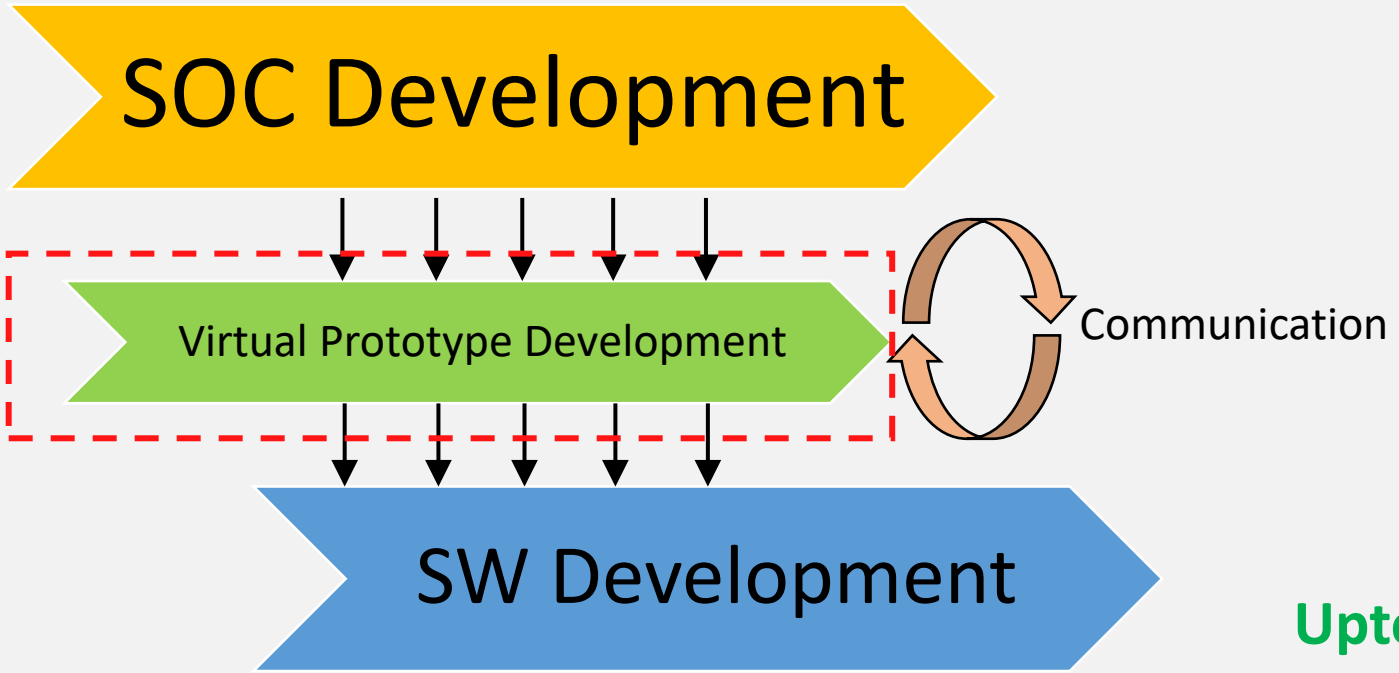




- High performance
- Efficient

Design and verification of SOCs in timely manner?

Virtual Prototypes



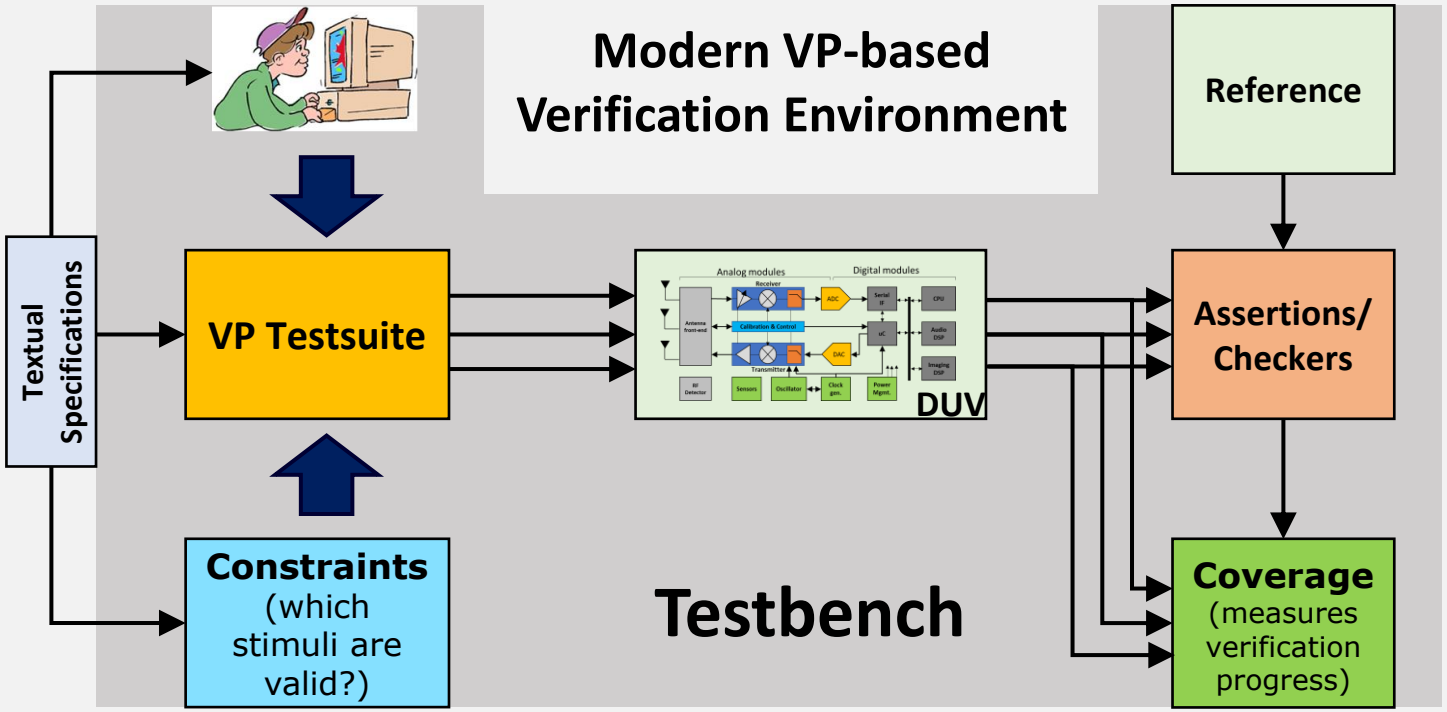
Upto 12 months saved[1]

Abstract SW models of HW
SystemC/ AMS
HW/SW co-design
Golden reference

Functional correctness?

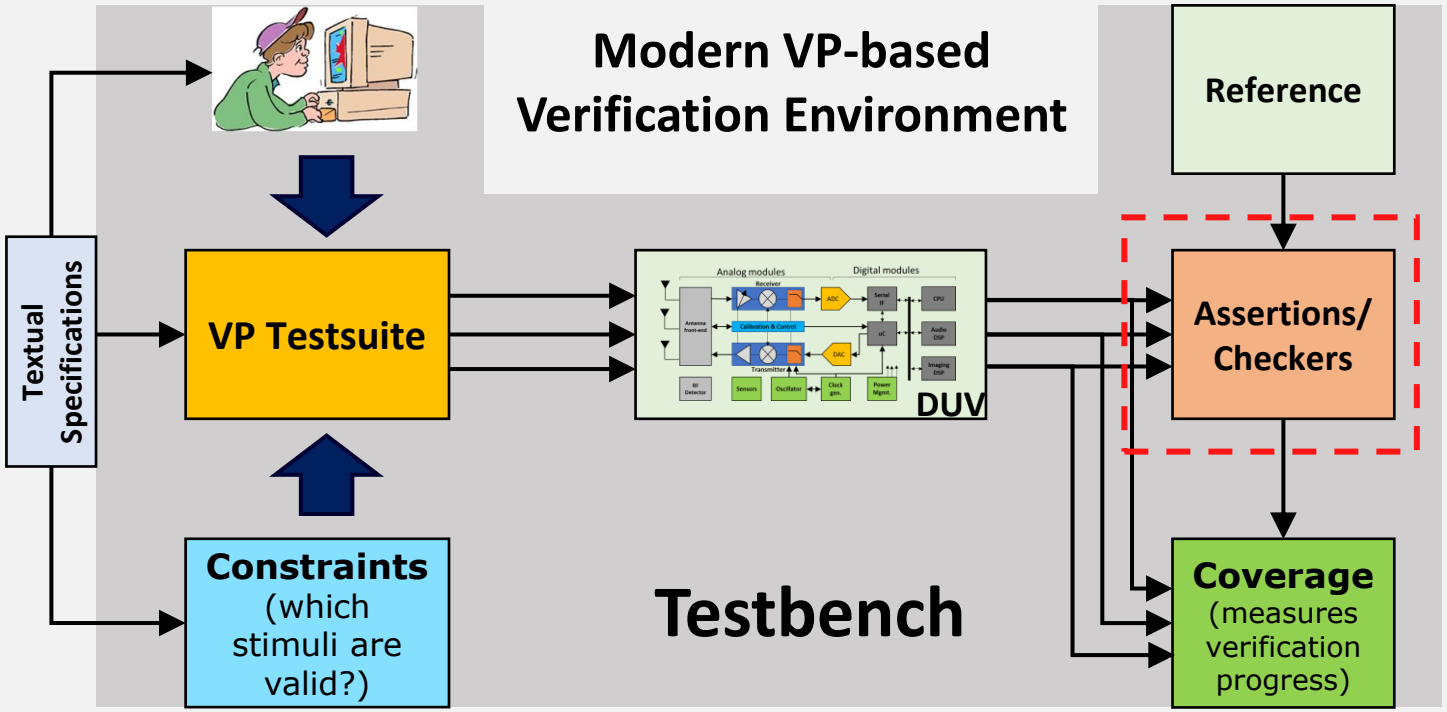
1. https://armkeil.blob.core.windows.net/developer/Files/pdf/white-paper/virtual-prototyping-soc-design.pdf

Modern VP-based Verification Environment



Poor quality Testsuite and VP

AMS VP Verification – Challenge



Availability of system-level assertions library

- Expressiveness to represent complexity
- Compatibility to SystemC/AMS and TLM
- Capturing complex interaction, e.g., analog – digital

Big problem!

Practical system-level assertions library!

Contribution: System-level Assertions Library

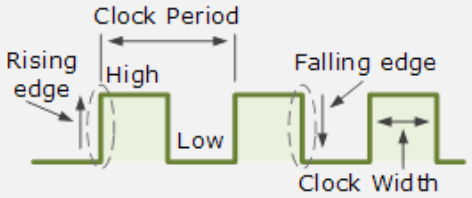
- System-level assertions library for heterogeneous systems
  - SystemC/AMS + TLM
  - Follows SystemVerilog Assertions (SVA) closely
- Application Programming Interface
  - Intuitive
  - User-friendly
  - Expressive
- Complex behaviors
  - Analog-to-digital
  - Digital-to-analog …
- Software/Hardware interactions
- Industrial Case Studies
  - ARM V8 based CPU using ARM Fast Models
  - Temperature control system

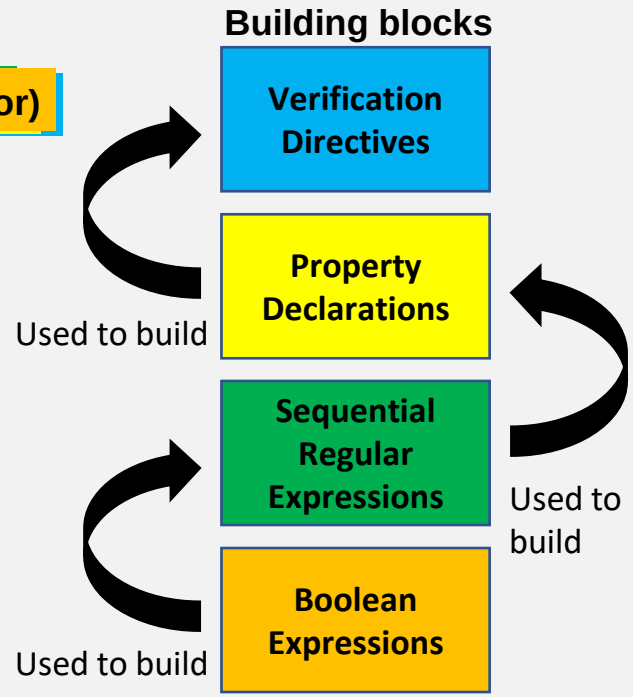
Outcome: High-quality verified DUV

SystemVerilog Assertions Standard

- SystemVerilog Assertions (SVA)

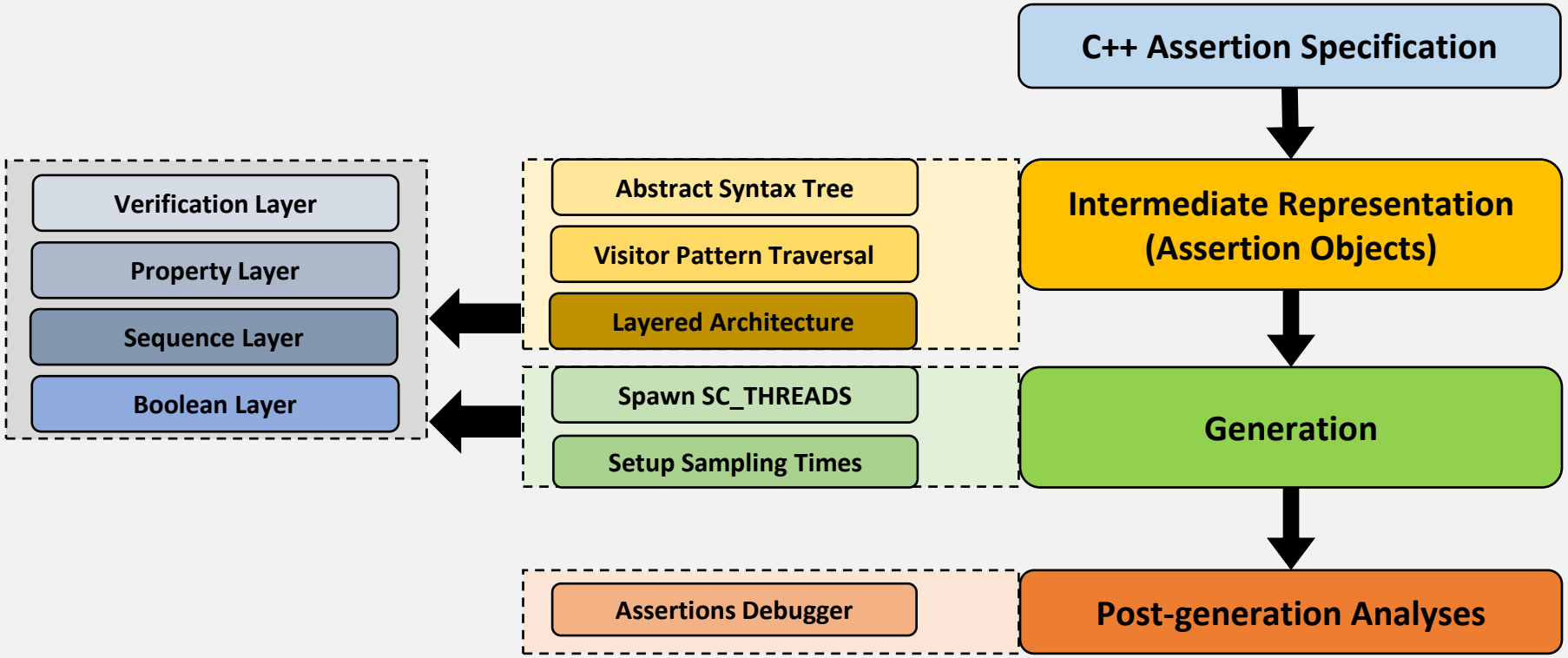
```
assert property (@posedge clock) req |-> gnt ##1 (done && !error)
```





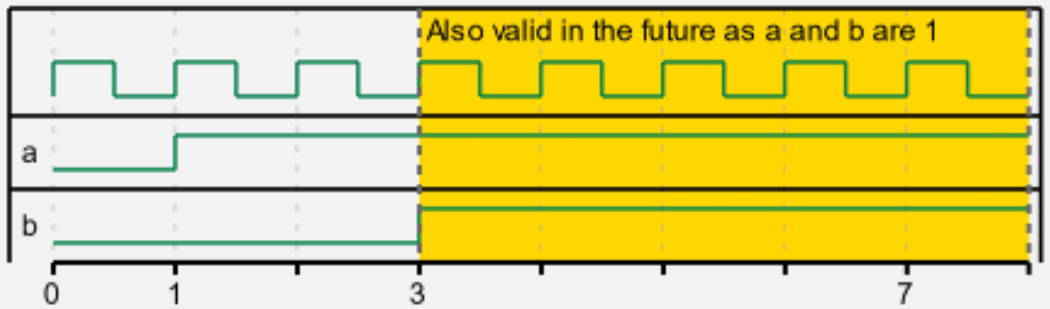
- Operators (arithmetic, relational, implication ...)
- Implication operator
  - Pre-conditions (antecedent)
  - Post-conditions (consequent)
- Mainly for digital systems!

System-level Assertions Library - Architecture



Assertion Example

- Required behavior



  - b is true 2 cycles after a is true

- SystemVerilog assertion

```
sequence a_b;
   @ (posedge clk)
      a ##2 b;
endsequence
assert property (a_b);
```

- System-level assertions library

```
1. sequence a_b = a | delay(2) | b;
2. property example_prop = a_b;
3. example_prop.default_sampling(clk.posedge_event());
4. ASSERT_PROPERTY (example_prop);
```

Boolean Layer and Lambda Functions

- Boolean layer
  - Describes the atomic behavior of signals
  - Lambda functions are supported
    - Define complex functionality inside an assertion

```
1. expr<std::function<int(void)>> lambda { []() {
2.             return calc_average();
3. } };
4. auto expr_lambda = lambda < 3.0;
5. ASSERT_PROPERTY (expr_lambda);
```

Operator	Name
+= -= /= *= &= |=	Binary assignment operators
< <= > >=	Binary relational operators
+ - ∗ =	Binary arithmetic operators
&& || == !=	Binary logical operators
+ - ! ++ –	Unary operators

Supported boolean operators

Sequence Layer and More …

- Sequence layer
  - Builds on top of boolean layer
  - Specifies temporal relationship between boolean expressions
  - Support for *delay ()* and *repeat ()*

```
1. // delay (value) and repeat (value)
2. sequence seqr1 = expr_a | delay (3) | expr_b | repeat (2)
3. //repeat (min_value, max_value)
4. sequence seqr2 = expr_a | delay (2,3) | expr_b | repeat (1,3)
```

Operator	Description
delay	Specifies delay from current sampling point until the next
and	Sequence *and* operation
or	Sequence *or* operation
repeat	Repetition operator
|	Sequence continue operator

Supported sequence operators

Property Layer and Implications

- Property layer
  - General behaviors to be specified
  - An implication built up from several sequences

antecedent **- > ∗** consequent
where **- > ∗** is overlapping implication operator

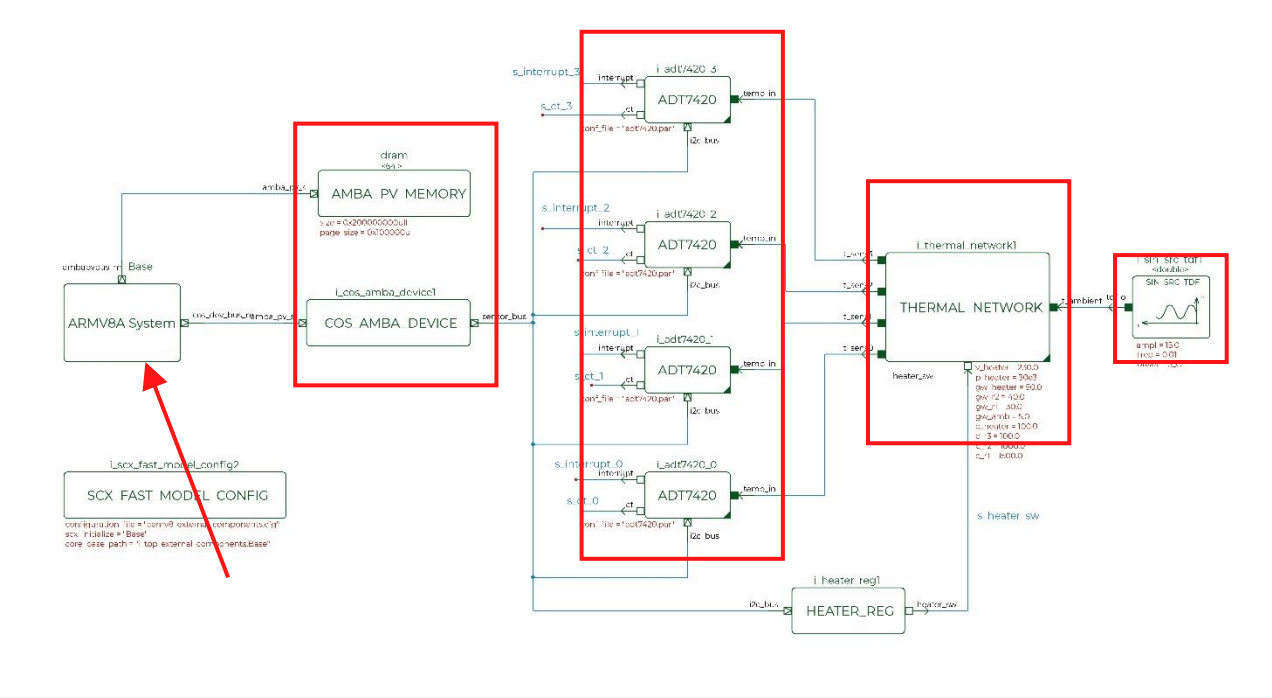
```
1. expr expr1 = a_int < 42;
2. sequence expr2 = true | delay(1,2) | expr_a + expr_b;
3. property non_overlap = expr1 ->* expr2;
4. ASSERT_PROPERTY (non_overlap);
```

Not supported – non-overlapping assertion!*

* One can use delay(…) operator

Experimental Evaluation – Case Study

- Temperature control system
  - Software + digital hardware + analog
  - SystemC/AMS + TLM
    - Time dataflow model
    - Electrical linear network
  - An ARM V8 based CPU using ARM Fast Models – Linux OS + SW
  - 4 ADT7420 temperature sensors – discrete event model
  - AMBA bus to connect temperature sensors and ARM processor
  - An environment model (Thermal_Network)
  - A heater model



Experimental Evaluation – Scenario

- Booting a Linux operating system on the ARM processor,
- A control SW is executed on top of Linux. The control SW continuously measures (monitors) the temperature sensor output,
- If the SW detects that the temperature value falls below a programmed threshold value, it switches the heater to ON state,
- Otherwise, when the temperature exceeds a certain programmed threshold, the heater is switched to OFF state

Assertion description

When the temperature of Room 1 *t_r1* (SystemC TDF signal) is above the threshold *t_threshold* (SW-controlled TLM register value), the heater has to be switched off (*heater_sw*) within 1 ms.

Experimental Evaluation – Scenario to Assertion

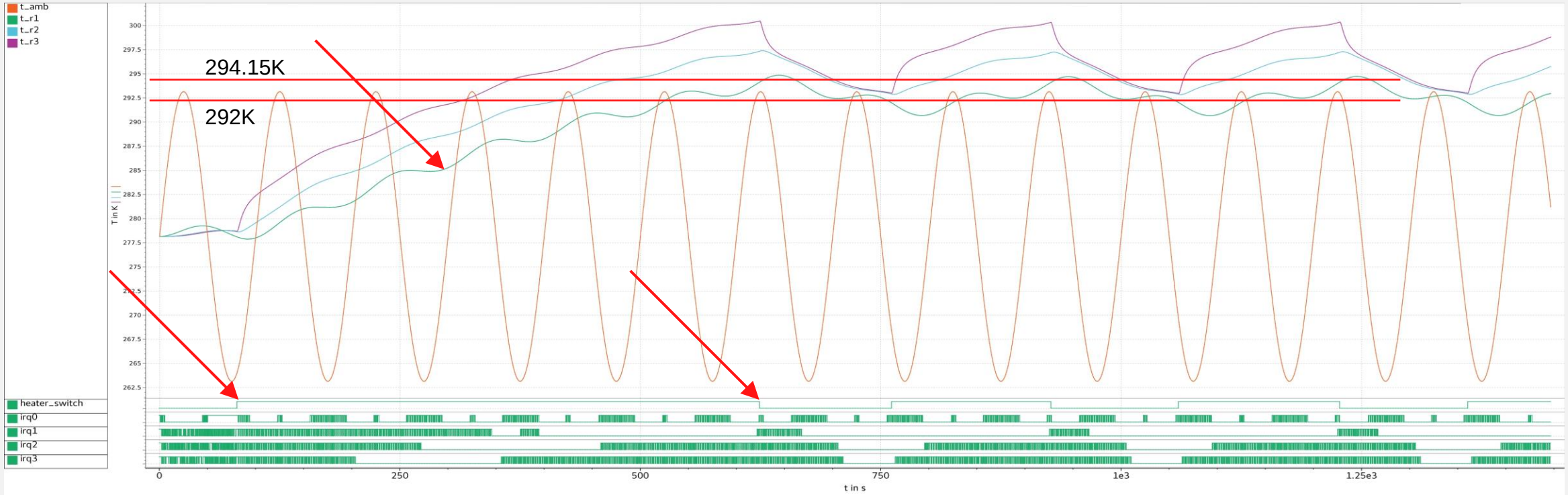
Assertion description

When the temperature of Room 1 *t_r1* (SystemC TDF signal) is above the threshold *t_threshold* (SW-controlled TLM register value), the heater has to be switched off (*heater_sw*) within 1 ms.

```
1. auto heater_off = (t_r1 > t_threshhold) ->* (true | delay(1_SC_MS) | (heater_sw==false));
2. heater_off.default_sampling(1_SC_MS);
```

Experimental Evaluation – Simulation Results

```
1. auto heater_off = (t_r1 > t_threshhold) ->* (true | delay(1_SC_MS) | (heater_sw==false));
2. heater_off.default_sampling(1_SC_MS);
```



Conclusion

- System-level assertions library for heterogeneous systems
- Application Programming Interface
- Complex behaviors
- Software/Hardware interactions
- Industrial Case Studies
  - ARM V8 based CPU using ARM Fast Models
  - Temperature control system

Outcome: High-quality verified DUV

A Cross-domain Heterogeneous ABV-Library for Mixed-signal Virtual Prototypes in SystemC/AMS

Muhammad Hassan[1] (muhammad.hassan@dfki.de)

Thilo Vörtler[2], Karsten Einwich[2]

Rolf Drechsler[1,3]

Daniel Große[1,4]

[1]Cyber-Physical Systems, DFKI GmbH, Bremen, Germany

[2]COSEDA Technologies GmbH, Dresden, Germany

[3]Institute of Computer Science, University of Bremen, Bremen, Germany

[4]Institute of Complex Systems, Johannes Kepler University, Linz, Germany

AUTOASSERT

Bundesministerium für Bildung und Forschung

16ME0117