

AMT2.0 - Qualitative and Quantitative Trace Analysis with Extended Signal Temporal Logic

TACAS 2018

Dejan Ničković

AIT Austrian Institute of Technology

Olivier Lebeltel, Oded Maler, Dogan Ulus

VERIMAG

Thomas Ferrère,

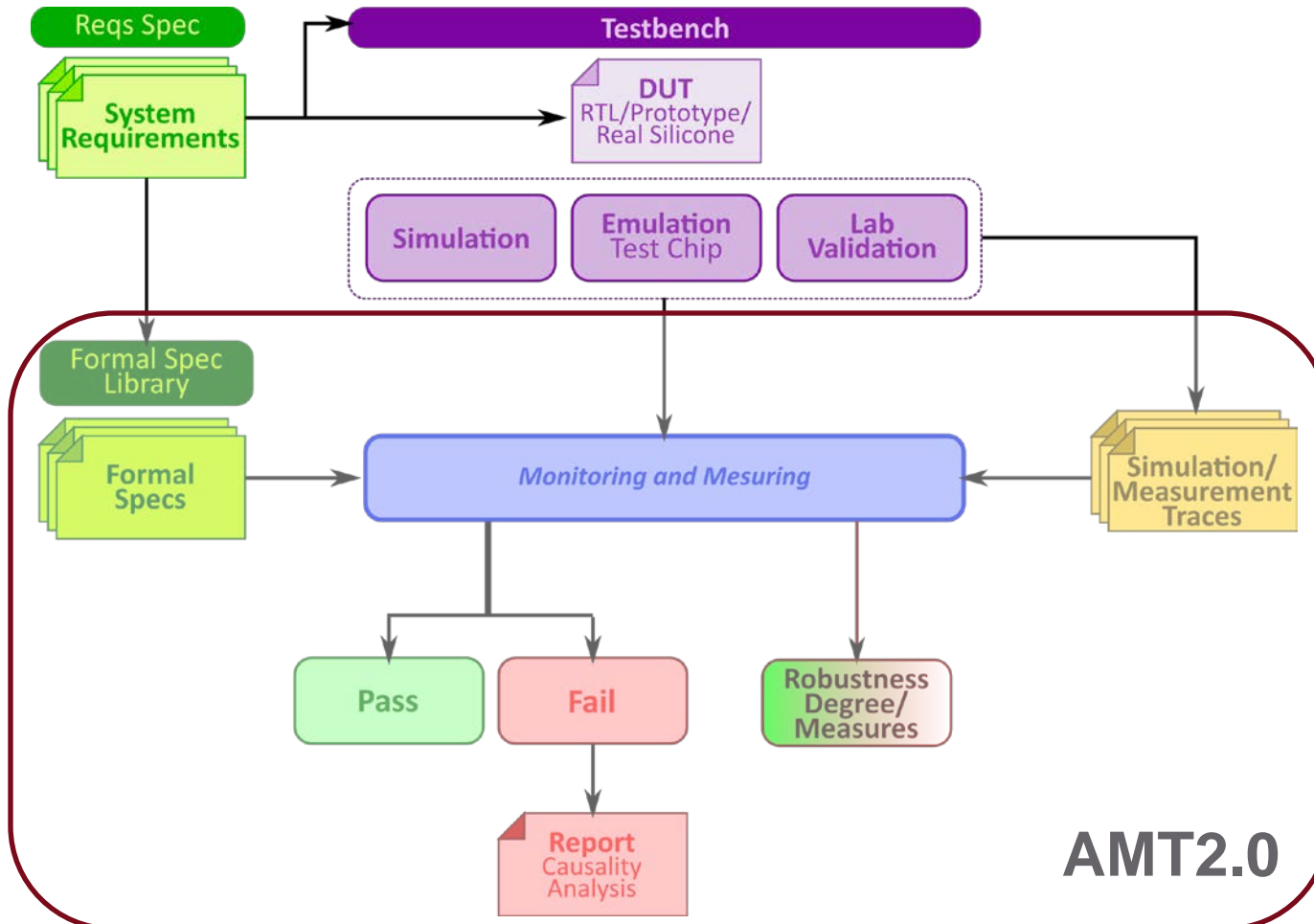
IST Austria

Introduction

- Mixed-signal increasingly important in safety-critical applications
 - Automotive, avionics, medical...
 - Sensors ↔ Controllers ↔ Actuators

- V&V is a challenge
 - Simulation-based testing a common approach
 - **Property-based analysis of simulation traces**

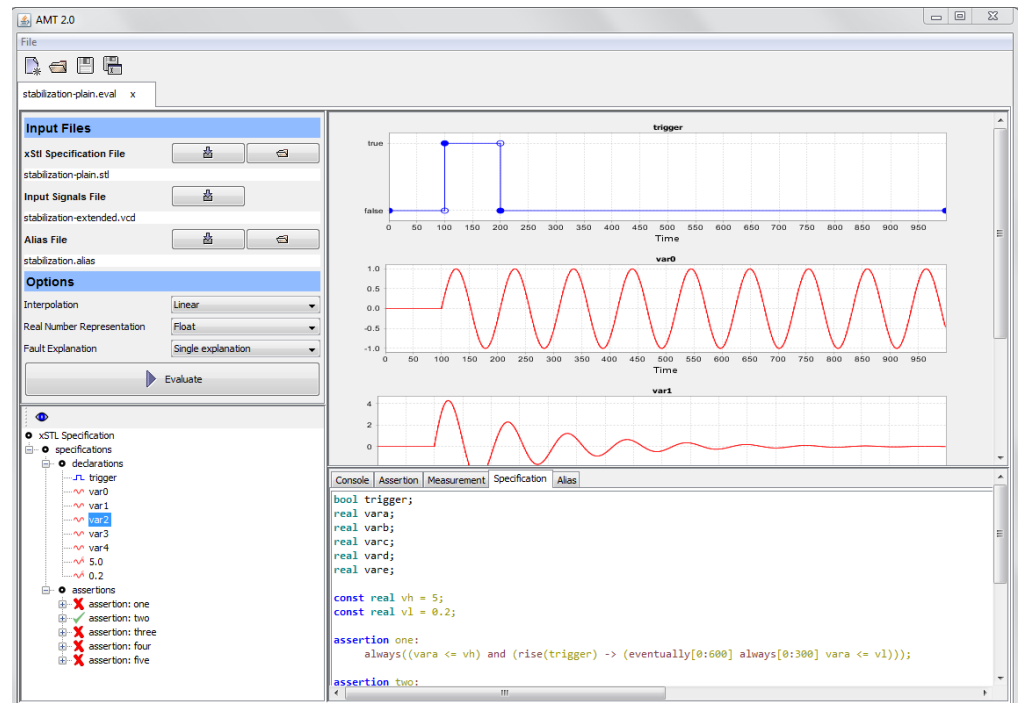
Property-based Monitoring Technology



- Rigorous
- Not ambiguous
- Automatic
- Scalable
- Reusable

AMT2.0 Highlights

- Extended Signal Temporal Logic
 - Signal Temporal Logic
 - Timed Regular Expressions
 - Measurement specifications
 - Offline qualitative monitors
 - Trace diagnostics
 - Fault explanations
 - Property-driven measurements
-
- Tool functionality via two examples
 - Bounded stabilization property
 - Clock jitter property



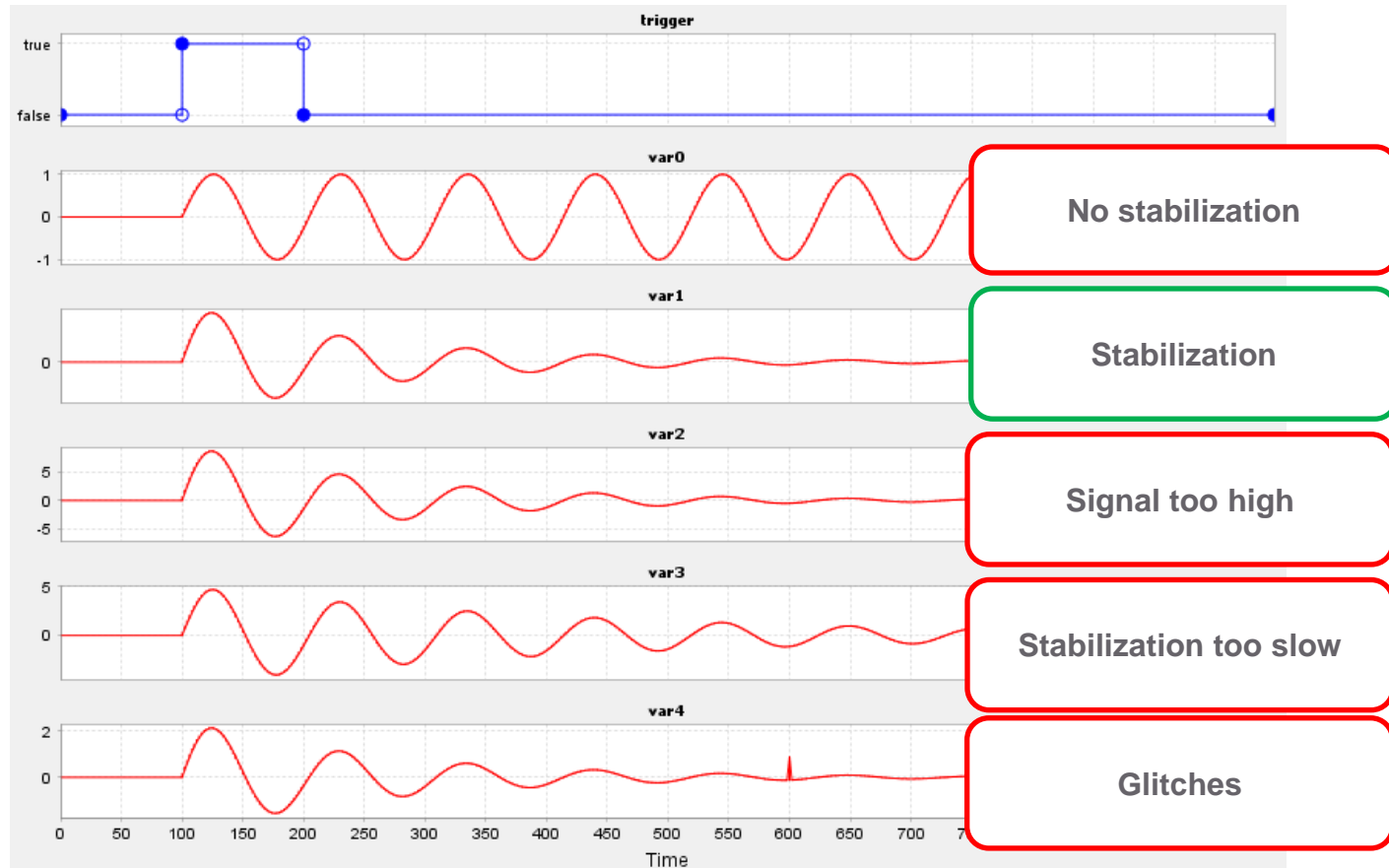
Bounded Stabilization Property

Informal Requirement

This requirement specifies conditions that need to hold for a bounded stabilization requirement. At every rising edge of the boolean **trigger**, the analog signal **var** is allowed to oscillate under the following conditions:

- **var** must always remain below **5V** ; and
- **var** must within **600s** go below **0.2V** , and continuously remain under that threshold for at least **300s**.

Simulation Traces



Formalization of the Requirement in xSTL

```

bool trigger ;
real var0;
...
real var5;
const real vh = 5;
const real vl = 0.2;

```

Variable and constant
declarations

```

template bool stabilization ( bool tg , real x, real vhigh , real vlow ) {
  bool result = ((x <= vhigh) and (rise(tg) ->
    (eventually[0:600] always[0:300] x <= vlow)));
  return result ;
}

```

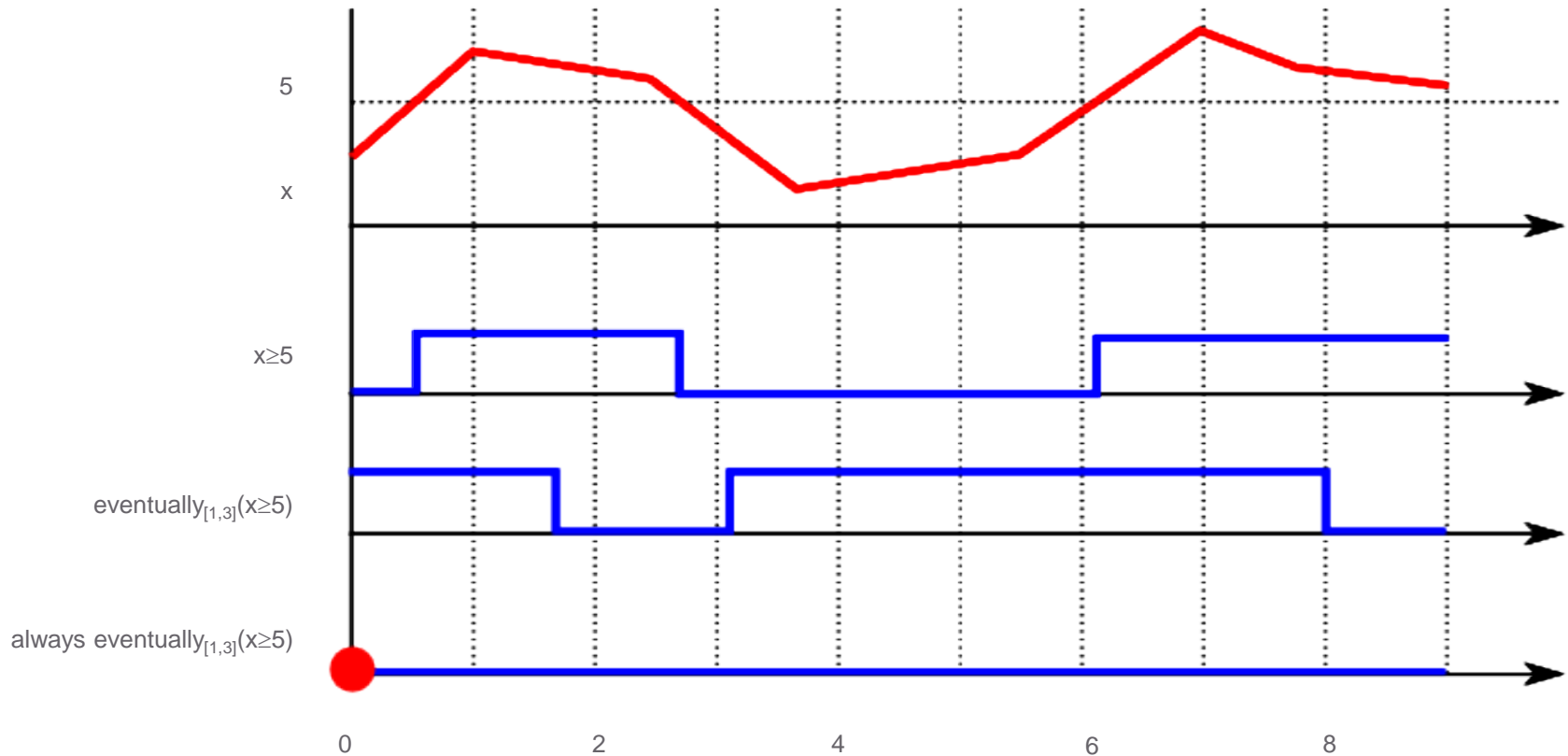
Property templates

```

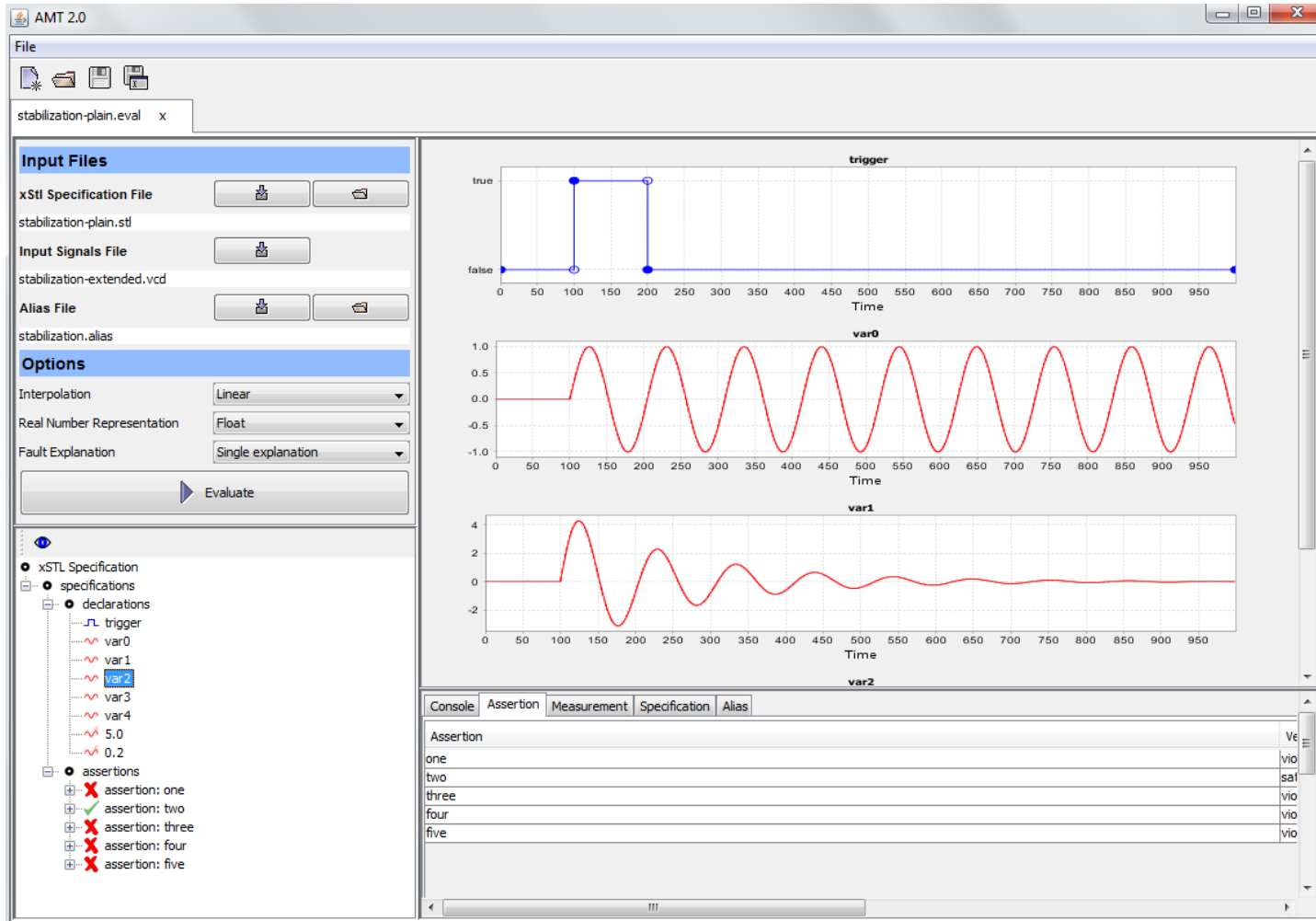
assertion one:
  always ( stabilization ( trigger , var0 , vh , vl));
...
assertion five :
  always ( stabilization ( trigger , var5 , vh , vl));

```


Property Evaluation – Offline Marking



Property Evaluation



The screenshot displays the AMT 2.0 software interface for property evaluation. The window title is "AMT 2.0" and the active file is "stabilization-plain.eval".

Input Files:

- xStl Specification File: stabilization-plain.stl
- Input Signals File: stabilization-extended.vcd
- Alias File: stabilization.alias

Options:

- Interpolation: Linear
- Real Number Representation: Float
- Fault Explanation: Single explanation

Tree View:

- xSTL Specification
 - specifications
 - declarations
 - trigger
 - var0
 - var1
 - var2
 - var3
 - var4
 - 5,0
 - 0,2
 - assertions
 - assertion: one (failed)
 - assertion: two (passed)
 - assertion: three (failed)
 - assertion: four (failed)
 - assertion: five (failed)

Waveform Plots:

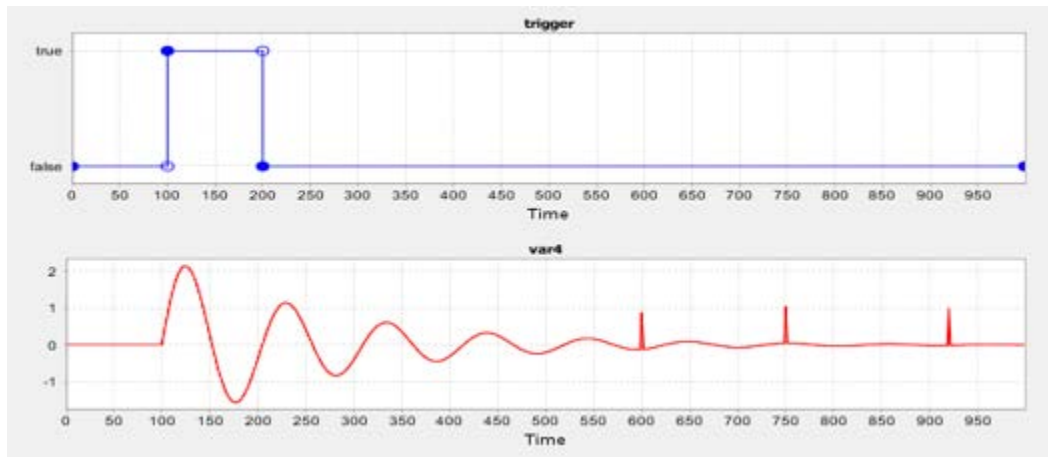
- trigger:** A blue square wave signal that is true from time 100 to 200 and false otherwise.
- var0:** A red sine wave oscillating between -1.0 and 1.0, starting at time 100.
- var1:** A red signal that starts at 0, peaks at 4 around time 120, and then decays towards 0.
- var2:** A red signal that starts at 0, peaks at 4 around time 120, and then decays towards 0.

Console/Assertion Table:

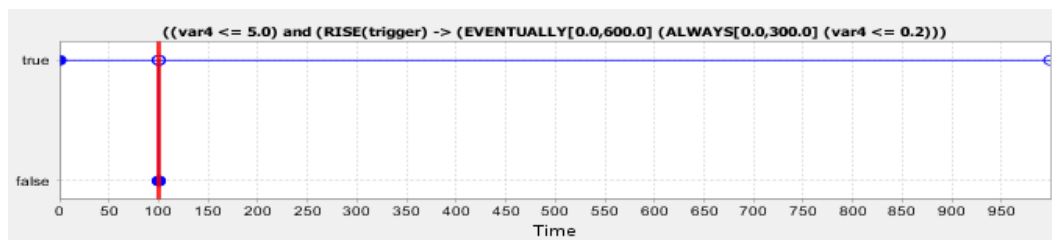
Assertion	Measurement	Specification	Alias
one			vio
two			sat
three			vio
four			vio
five			vio

Trace Diagnostics

- We focus on signals **trigger** and **var4**

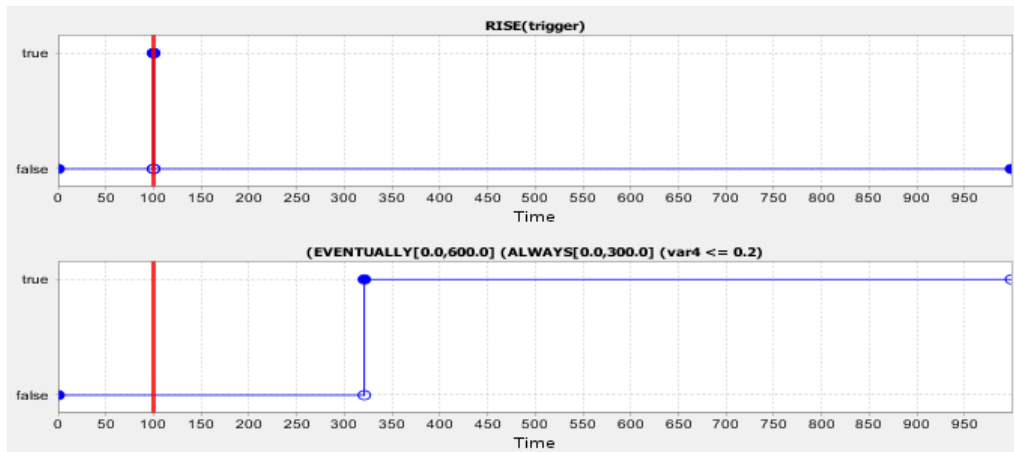


- Assertion violated because top formula violated at time 100s

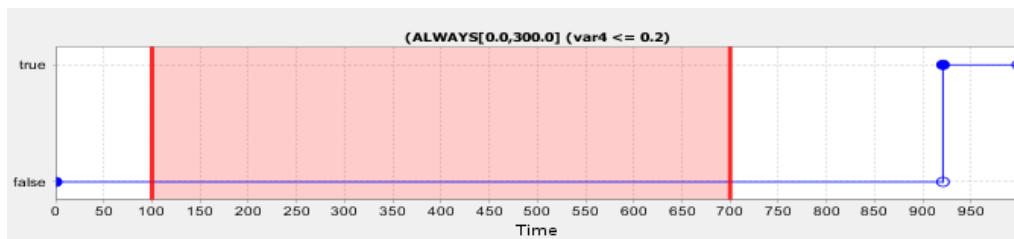


Trace Diagnostics

- Top formula violated at time 100s because **trigger** is at its rising edge at time 100s, but the future obligation **eventually[0:600]always[0:300] (var4 <= 0.2)** is not met

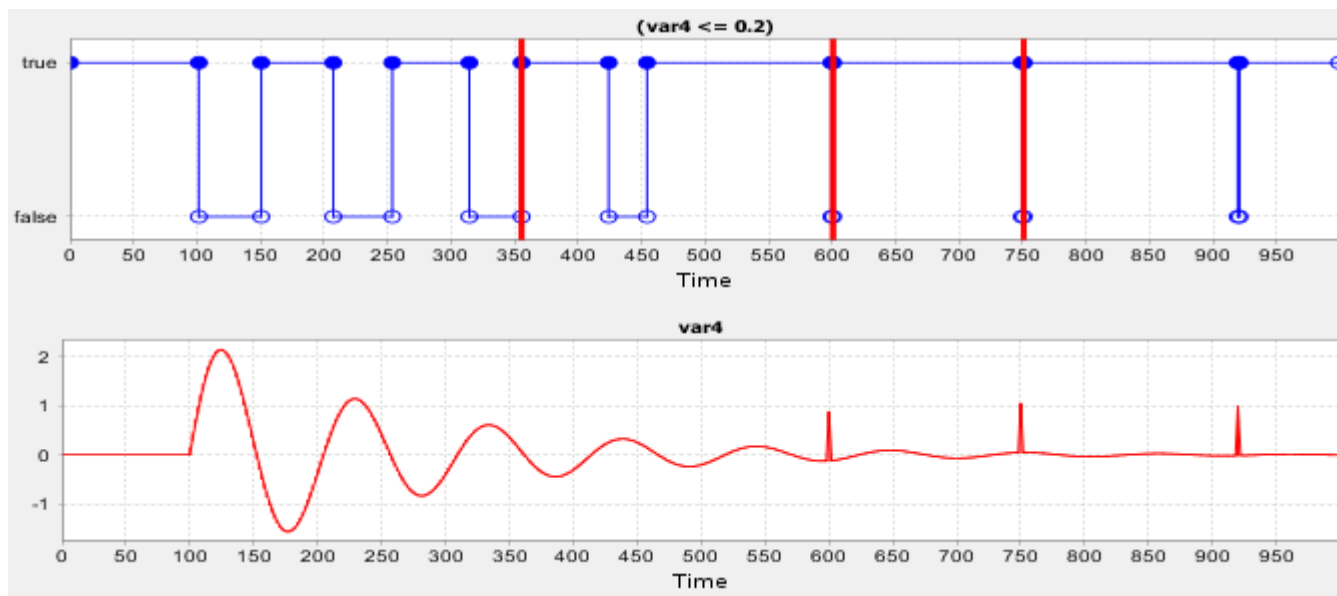


- because there is not time in [100s,700s] from which **var4** stays continuously below 0.2 for at least 300s



Trace Diagnostics

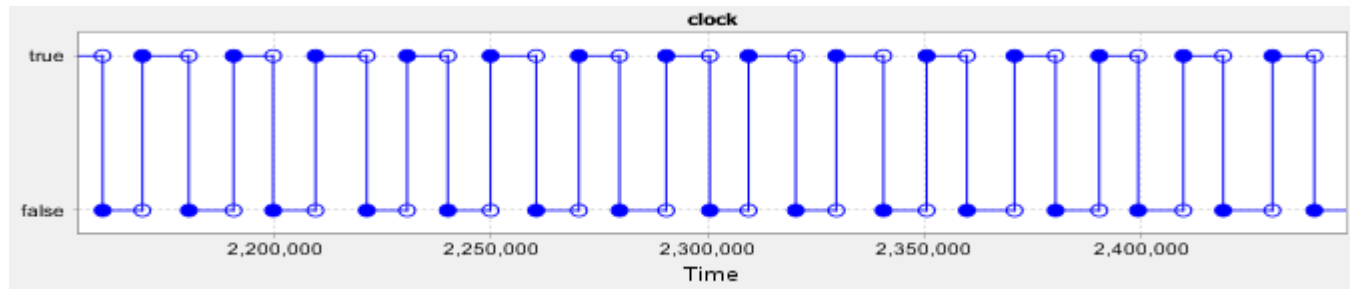
- there is not time in [100s,700s] from which **var4** stays continuously below 0.2 for at least 300s because var4 goes above 0.2 at regular intervals smaller than 300s because of glitched – for instance at times 350s, 600s and 750s



Clock Jitter Property

Informal Requirement and Input Signal

This requirement species a digital clock jitter pattern to measure. Given a continuous-time Boolean-valued signal **clock**, a clock period is defined as a segment that starts with the rising edge of the clock and ends with its consecutive rising edge. The measurement specification requires measuring the **duration** of all the **clock periods** matched within the clock signal.



Formalization of the Requirement in xSTL

```
bool clock;
bool nclock = not clock;
```

```
measurement jitter_clock_period {
    pattern clock_period = start(clock):clock:nclock:start(clock);
    measure duration(clock_period);
}
```

Mesurement specification
With TRE

Property Evaluation

AMT 2.0

File: stabilization-plain.eval x dock-jitter.eval x

Input Files

xStl Specification File: clock-jitter.stl

Input Signals File: clock.vcd

Alias File: clock-jitter.alias

Options

Interpolation: Linear

Real Number Representation: Float

Fault Explanation: None

Evaluate

duration (start(clock):clock:not(clock):start(clock))

Number of matches

Duration

Total number of segments: 299
Maximum value: 21877.0
Minimum value: 17241.0
Average value: 20002.046822742475

duration (start(clock):{clock:not(clock)}[19000.0,21000.0]:start(clock))

Number of matches

Duration

Total number of segments: 243
Maximum value: 20995.0
Minimum value: 19017.0
Average value: 19995.897119341564

Console | Assertion | Measurement | Specification | Alias

Name	Type	Number of Segments	Maximum Value
jitter_clock_period	duration (start(clock):clock:not(clock):start(...	299	21877.0
jitter_clock_period_c	duration (start(clock):{clock:not(clock)}[19...	243	20995.0
jitter_clock_period	duration (start(clock):clock:not(clock):start(...	299	21877.0
jitter_clock_period_c	duration (start(clock):{clock:not(clock)}[19...	243	20995.0

Tree view:

- xSTL Specification
 - specifications
 - declarations
 - measurements
 - jitter_clock_period
 - jitter_clock_period
 - patterns
 - measures
 - measure
 - jitter_clock_period_c
 - jitter_clock_period_c
 - patterns
 - def

Summary and Additional Insights

AMT2.0 Algorithms

- Offline monitoring algorithm with full STL semantics (including events)
 - *Oded Maler, Dejan Nickovic: Monitoring properties of analog and mixed-signal circuits. STTT 15(3): 247-268 (2013)*
- Timed regular expressions matching
 - *Dogan Ulus, Thomas Ferrère, Eugene Asarin, Oded Maler: Timed Pattern Matching. FORMATS 2014: 222-236*
- Timed regular expressions measurements
 - *Thomas Ferrère, Oded Maler, Dejan Nickovic, Dogan Ulus: Measuring with Timed Patterns. CAV (2) 2015: 322-337*
- Trace diagnostics for STL
 - *Thomas Ferrère, Oded Maler, Dejan Nickovic: Trace Diagnostics Using Temporal Implicants. ATVA 2015: 241-258*

xSTL – Combining STL and TRE

- STL formula within TRE pattern
 - *Implicit*
 - *Example: (not clock and reg):clock*
- TRE pattern within STL formula
 - *Explicit projection operators **match_begin** and **match_end***
 - *Example: match_end(not clock:clock) -> eventually reg*

AMT2.0 Features - Summary

- New specification language
 - *STL + TRE*
 - *Easier specifications*
 - *Declaration of typed variables and constants*
 - *Reusable property templates*
 - *Measurement specifications*
- Trace diagnostics with temporal implicants
 - Small and hierarchical explanations of violations
- Continuous signal interpolation and interpretation
 - Linear and step interpolation
 - Reals as floats or rationals
- Tool portability
 - Java implementation
- Delay with the release

Thank you!