Hybrid RELTL for Analog-Mixed Signals

Stefano Tonetta

joint work with

Alessandro Cimatti and Marco Roveri

FBK-irst, Trento, Italy
{cimatti,roveri,tonettas}@fbk.eu

FAC Workshop, Snowbird, 14 July 2011
Assertion-based design for AMS

- Design of IC more and more complex.
- Integration of digital and analog block is a main issue.
- Verification techniques for digital systems does not work for system-level logic verification.
- Most of bugs in misunderstanding/incomplete/inconsistent properties on the interfaces among digital and analog blocks.
- In Software Engineering jargon, these are requirements faults/errors.
- Necessary a precise specification of assertions and assumptions.
- Standard languages for discrete circuits assertions such as PSL (Sugar, ForSpec, ...).
  - RELTL as core temporal logic.
  - It combines Linear-time Temporal Logic (LTL) and Regular Expressions.
- HDLs extended with continuous variables and differential equations.
We need a logic that
- represents temporal constraints
- includes predicates over derivatives
- includes predicates over discrete changes
- can be analyzed symbolically and automatically.

Our solution:
1. **HRELTL** logic:
   - extends RELTL (Linear-time Temporal Logic with Regular Expressions) with hybrid aspects;
   - interpreted over hybrid traces;
   - predicates over derivatives in continuous evolutions;
   - predicates over discrete steps.
2. Reduction of satisfiability problem for a linear fragment to an equi-satisfiable problem for RELTL.
   - allows the re-use of validation techniques for RELTL.
Outline

1. From discrete to hybrid RELTL
2. HRELTL for AMS
3. SMT-based analysis
4. Conclusions
LTL

- Propositions: \( p_1, p_2, \ldots \) bits or Boolean predicates.
- Boolean combinations: and, or, not, implies.
- Temporal operators: next, eventually, always, until.
- Examples:
  - safety
    always (not (\( p_1 \) and \( p_2 \)))
  - response to an impulse
    always (\( p_1 \) implies eventually \( p_2 \))
  - response to permanent holding
    always (always \( p_1 \) implies eventually \( p_2 \))
  - response to persistence
    (always eventually \( p_1 \)) implies eventually \( p_2 \)
  - fairness
    always eventually \( p_1 \)
Regular expressions:
- Repetition: \( r_1[^*n] \) (\( n = 0 \) means empty sequence)
- Concatenation: \( r_1; r_2 \).
- Fusion: \( r_1 : r_2 \).
- Or: \( r_1|r_2 \).
- And: \( r_1\&\&r_2 \).
- Non-matching and \( r_1\&\&r_2 \).

Suffix operators:
- Suffix implication: \( r \midrightarrow \phi \).
- Suffix conjunction: \( r \diamondrightarrow \phi \).

Allows responses to sequences:
\[
\text{always } \left( \{ p_1; p_2[^*]; p_3 \} \midrightarrow \text{eventually } p_4 \right)
\]

Reaches \( \omega \)-regular expressiveness:
\[
\{ \text{true}; p \}[^*] \diamondrightarrow \text{true}.
\]
From discrete to hybrid traces

<table>
<thead>
<tr>
<th>discrete trace</th>
<th>hybrid trace</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="discrete_trace" /></td>
<td><img src="image2.png" alt="hybrid_trace" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>continuous signal</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image3.png" alt="continuous_signal" /></td>
</tr>
</tbody>
</table>

- **HRELTL** = RELTL interpreted over hybrid traces with:
  - continuous variables
  - arithmetic predicates with next and derivatives
Interpretation of continuous predicates

- Required features to guarantee the well-defined interpretation of the continuous predicates:
  - interval-based logic
    - in a semantics based on time-points, $x \leq 0 \text{ until } x > 0$ would be unsatisfiable (if $x$ is continuous);
  - both open intervals and time-points:
    - $x < 0$ requires right-open intervals.
    - $x > 0$ requires left-open intervals.
    - $x = 0$ requires time points.
  - finite variability:
    - we must guarantee that the continuous behaviors can be sampled enough to have a uniform interpretation of the predicates;
  - sampling invariance:
    - the interpretation of formulas does not depend on the sampling.
- arbitrary interpretation of next terms over continuous evolution.
Outline

1. From discrete to hybrid RELTL
2. HRELTL for AMS
3. SMT-based analysis
4. Conclusions
AMS assertions in HRELTL

Examples taken from the web:

1. always((a < 10 and b) implies c)
2. always((0 ≤ a ≤ 5) implies (−275 ≤ DER(a) ≤ 275))
3. always(a > 5 implies ((a ≥ 4.5 and b ≥ 4.5) until (b < 4.5))
4. always(a > 4.5 implies −0.1 ≤ b − c ≤ 0.1)
Oscillator

-- v is a continuous variable
VAR v: continuous;
-- v does not jump
-- during discrete changes
CONSTRAINT
G ( STEP -> next(v)=v)
-- oscillating behavior
CONSTRAINT
G F ( v>0 ) & G F (v<0)
-- inconsistent scenario
CONSTRAINT
G (v !=0)
VAR
v: continuous; t: continuous;
-- initial condition
CONSTRAINT
t=0 & v=-1000 & der(v)>0
-- switching behavior
CONSTRAINT
G (der(v)>0 -> ( (der(v)>=18 & der(v)<=22 & t<100) U (t=100 & X (t=0 & der(v)<0)))) & 
G (der(v)<0 -> ( (der(v)>=-22 & der(v)<=-18 & t<100) U (t=100 & X (t=0 & der(v)>0))))
-- the property
CONSTRAINT
! G (v>=-2000 & v<=2000)
-- Assumptions:
-- v does not jump during discrete changes
CONSTRAINT
G ( STEP -> next(v)=v)
-- t can be reset only after 100
CONSTRAINT
G (t<100 -> ( STEP -> next(t)=t))
-- t is a timer
CONSTRAINT
G (der(t)=1)
Switched capacitor II
Outline

1. From discrete to hybrid RELTL
2. HRELTL for AMS
3. SMT-based analysis
4. Conclusions
Equi-satisfiable discretization

- Satisfiability is undecidable.
- Discretize and apply infinite-state model checking.

\[
\begin{align*}
\text{HRELTL} & \quad \xrightarrow{\tau} \quad \text{RELTL} \\
\text{RELTL} & \quad \text{(with SMT constraints)}
\end{align*}
\]

The translation \( \tau \) of a generic HRELTL formula is defined as:

\[
\tau(\phi) := \psi_l \land \psi_{\text{DER}} \land \psi_{\text{PRED}_\phi} \land \psi_{V_D} \land \tau'(\phi).
\]

Theorem

\( \phi \) and \( \tau(\phi) \) are equi-satisfiable.
SMT-based analysis

1. convert hybrid formula into discrete $\phi$
2. build a fair transition system $S_\phi$
3. check whether the language accepted by $S_\phi$ is not empty.

Example:

11 boolean variables
2 real variables
4 fairness conditions

BMC (with fairness)

$k = 4$
$< 1$ second

SAT

K-induction + predicate abs.

$k = 6, 14$ predicates
$< 1$ second

UNSAT
Outline

1. From discrete to hybrid RELTL
2. HRELTL for AMS
3. SMT-based analysis
4. Conclusions
Final remarks

- Techniques integrated on top of NuSMV.
- GUI with timed-trace viewer.
- **OTHELLO** = Object Temporal Hybrid expressions Linear-time temporal Logic
  - Example:
  
  The train trip shall issue an emergency brake command, which shall not be revoked until the train has reached standstill and the driver has acknowledged the trip (ETCS SRS Sec. 3.13.8.2)

  for all \( t \) of type \( \text{Train} \) \( (t.\text{trip} \implies (t.\text{emergency\_brake \ until (t.\text{speed} = 0 \ and \ t.\text{driver.\ ack}}))) \)

- Result of the industrial project EuRailCheck (European Railway Agency) and the project OthelloPlay (winner of the SEIF 2010 MSR award).
- Validated by railway experts to formalize the requirements of the European Train Control System.
Future directions

- Integration with SMT techniques for hybrid system verification (see talk of Sergio Mover at CAV).
- Integration with testing and ATPG.
- Validation of hybrid regular expressions.
- Non-linear continuous signals.
- SMT-based representation of digital encoding of real data.
Thanks for your attention