Last Values of Quantities in VHDL-AMS

We point out to some aspects that concern access to values of quantities at the last analog solution point. We start with some modeling examples where such an access seems to make sense. Afterwards possibly handling in VHDL-AMS is discussed.

1 Examples

In the subsequent examples, we use t_n as the current analog simulation point (ASP) and t_{n-1} as the previous time point.

Preisach model

The Preisach model of hysteresis bases on the relay hysteron with (time-dependent) input *x(t)* and output *y(t)*. Let t_n the current time point and t_{n-1} the last time point we can express the input-output relationship of the building block by

$$y(t_n) = \begin{cases} 1 & if \quad \beta \le x(t_n) \quad or \quad (\alpha < x(t_n) < \beta \quad and \quad y(t_{n-1}) = 1) \\ 0 & if \quad x(t_n) \le \alpha \quad or \quad (\alpha < x(t_n) < \beta \quad and \quad y(t_{n-1}) = 0) \end{cases}$$

with an initialization $y(t_0) = 0$ or $y(t_0) = 1$ if $(\alpha < x(t_0) < \beta$ (see also [1]). The Preisach model is an approach to describe for instance ferromagnetic behavior.

Inline Integration

Inline integration offers an opportunity to reduce the simulation effort in some cases. The basic idea is to insert discretization formulae directly into the differential algebraic equation model and afterwards make use of the potential to simplify the model in a preprocessing step (see for instance [2, 3, 4]). If the modeler can access the value of an analog quantity at the last analog solution point he can carry out this step during establishing a model

Example: The constitutive relation of an inductance $v = L \cdot \frac{di}{dt}$ is replaced by
$i(t_n) = i(t_{n-1}) + \frac{t_n - t_{n-1}}{L}$ v(t_n).

Recursive convolution

Recursive convolution is similar to the inline integration. In this case, we determine $y(t) = \int_0^t x(\tau) \cdot g(t-\tau)d\tau$ where the weight function is given by the sum $g(t) = \sum_{i=1}^n g_i(t)$ with special $g_i(t) = \alpha_i \cdot e^{-\beta_i \cdot t}$. Thus, $y(t) = \sum_{i=1}^n y_i(t)$ with $y_i(t) = \int_0^t x(\tau) \cdot g_i(t-\tau)d\tau$.

We can now recursively determine all $y_i(t_n) = \int_{t_{n-1}}^{t_n} x(\tau) \cdot g_i(t_n-\tau)d\tau + \int_0^{t_{n-1}} x(\tau) \cdot g_i(t_n-\tau)d\tau$

using $\int_0^{t_{n-1}} x(\tau) \cdot g(t_n-\tau)d\tau = \int_0^{t_{n-1}} x(\tau) \cdot \alpha_i \cdot e^{-\beta_i \cdot (t_n - t_{n-1} + t_{n-1} - \tau)}d\tau = \alpha_i \cdot e^{-\beta_i \cdot (t_n - t_{n-1})} \cdot y_i(t_{n-1})$.

Thus, we exactly get $y_i(t_n) = \int_{t_{n-1}}^{t_n} x(\tau) \cdot \alpha_i \cdot e^{-\beta_i \cdot (t_n-\tau)}d\tau + \alpha_i \cdot e^{-\beta_i \cdot (t_n - t_{n-1})} \cdot y_i(t_{n-1})$. If we determine the first summand using for instance trapezoidal rule we get

$$y_i(t_n) = \frac{\alpha_i \cdot \left(x(t_n) + x(t_{n-1}) \cdot e^{-\beta_i \cdot (t_n - t_{n-1})}\right)}{2} \cdot (t_n - t_{n-1}) + \alpha_i \cdot e^{-\beta_i \cdot (t_n - t_{n-1})} \cdot y_i(t_{n-1})$$

Recursive convolution can be applied to model lines (see for instance [5]).

Thermo-electrical modeling

Access to solutions at the last time-point might also help to simplify electro-thermal simulation. We assume that the electro-thermal problem is described by the following vector-valued equations

$$\begin{aligned} f_{el}(x'_{el}(t), x_{el}(t), x_{th}(t)) &= 0 \\ f_{th}(x'_{th}(t), x_{th}(t), x_{el}(t)) &= 0 \end{aligned} \qquad (1)$$

with the functions f_{el} and f_{th} that characterize the electrical and thermal part resp. x_{el} and x_{th} are unknows that describe the electrical and thermal behavior resp. (for instance currents/voltages and temperatures). In general, time constants that characterize the dynamic behavior of the electrical part are much smaller than those of the thermal part. Thus, is seem that the equation can be decoupled with only small loss of accuracy using instead of (1) the equations

$$\begin{aligned} f_{el}(x'_{el}(t_n), x_{el}(t_n), x_{th}(t_{n-1})) &= 0 \\ f_{th}(x'_{th}(t_n), x_{th}(t_n), x_{el}(t_{n-1})) &= 0 \end{aligned} \qquad (2)$$

If the Jacobians of both parts w.r.t. x_{el} and x_{th} are regular it can be expected that this approach reduces numerical problems and improves the simulation speed.

2 Situation in VHDL

We try to discuss whether VHDL provides features to access the last values of time-discrete signals at the last digital solution point.

Attribute LAST_VALUE

S'LAST_VALUE provides the last value of a signal S (see [6], p. 247): "For a signal S, if an event has occurred on S in any simulation cycle, S'LAST_VALUE returns the value of S prior to the update of S in the last simulation cycle in which an event occurred; otherwise, S'LAST_VALUE returns the current value of S."

It is in general expected that S'LAST_VALUE delivers the value of S at the previous time point where a digital solution was determined. However, this might not be the case if the updates occur in more than one delta cycle at a discrete time point.

Postponed process

Postponed processes are characterized by the following features ([6], section 11.3, [7], p. 727-730)

- A postponed process is triggered in the same way as a normal process. However, it waits until the last delta cycle at the current digital simulation time. It is executed after all non-postponed processes are ready at this time point.
- Also in the initialization phase a postponed process is executed after all non-postponed processes.
- A postponed process accesses the values of signals that are the "final" values at the current simulated time.
- The postponed processes must not change signals with delta delay.

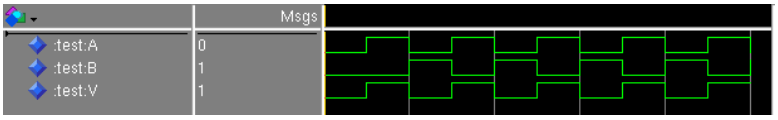
Example

A value can be assigned to a (shared) variable in a postponed process. The value can be set in a postponed process and be used in another process afterwards.

```
entity TEST is
end entity TEST;

architecture A0 of TEST is
    signal          A : BIT;
    signal          B : BIT;
    shared variable V : BIT;
begin
    A <= not A after 1 ms;
P1: postponed process is
    begin
      V := A;
      wait on A;
    end process P1;

P2: process (A) is
    begin
      B <= V;
    end process P2;
 end architecture A0;
```



3 Possibly VHDL-AMS extensions

The examples from section 1 could be applied if an access to the final value of a quantity at the last analog solution point is possible.

Attribute 'LAST_VALUE

This could be supported by a 'LAST_VALUE attribute that can be applied on a quantity Q. Q'LAST_VALUE should deliver the value of Q at the last analog solution point (not the last value during an iteration at the current time point).

The introduction of such an attribute would require some effort in the elaboration and execution phase as for instance

- Determination of the quantities with 'LAST_VALUE attribute
- Update of the values of these quantities after finishing every analog solution point

Postponed procedural statement

In the WG meeting on March 12, 2014, Ernst Christen presented the intention to provide postponed procedural statements in the VHDL-AMS revision. The minimum demand resulting from examples of section 1 would be

- To allow the assignment of values to shared variables of type REAL in postponed procedural statements,
- To avoid type conflicts unprotected types for (these) shared variables should be allowed,
- To clarify handling of postponed procedural statements in the elaboration phase (necessary to avoid conflicts during the initialization of shared variables).

Initialization of the mentioned shared variables could be done in postponed processes.

Example for initialization (without postponed procedural statement)

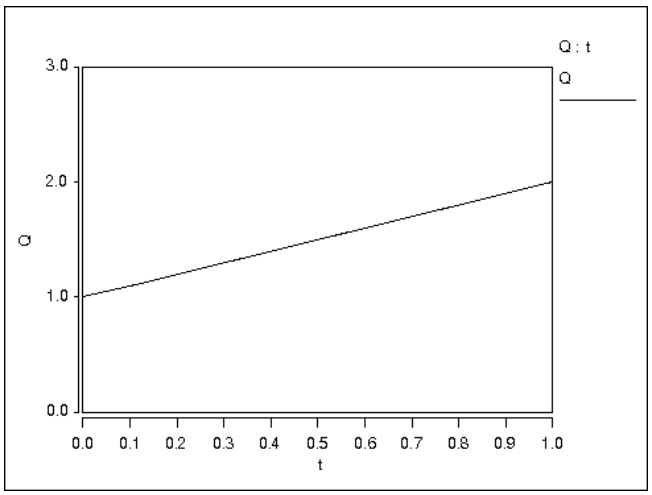
```
entity TEST is end entity TEST;

architecture A0 of TEST is
   quantity         Q      : REAL ;
   shared  variable Q_LAST : REAL ;
begin


INITIALIZATION: postponed process is
    begin
      Q_LAST := 1.0;
      wait;
    end process INITIALIZATION;

    Q == Q_LAST + NOW;

--P2: procedural is
--    begin
--      A := Q_LAST + NOW;
--    end procedural P2;
end architecture A0;
```



Comment

Simulation programs should also support

- procedural statements
- shared variables
- postponed processes

in order to support modeling approaches as pointed out in section 1. Consequences for small signal frequency analysis must be checked.

References

[1] Preisach model of hysteresis.
http://en.wikipedia.org/wiki/Preisach_model_of_hysteresis

{2] H. Elmquist, M. Otter, and F. E. Cellier: Inline Integration: A New Mixed Symbolic/Numeric Approach for Solving Differential Algebraic Equation Systems. Proc ESIM '95, 1995.
http://www.inf.ethz.ch/personal/fcellier/Pubs/OO/esm_95.pdf

{3] F. E. Cellier and E. Kofman: Continuous System Simulation. Springer, 2010. –see for instance section 10.6

[4] H. Elmquist, S.-E. Mattson, and H. Olsson: New Methods for Hardware-in-the Loop Simulation of Stiff Models. Proc. 2nd International Modelica Conference 2002, pages 59-64.
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.17.3783&rep=rep1&type=pdf

[5] Tuyen V. Nguyen: Recursive Convolution and Discrete Time Domain Simulation of Lossy Coupled Transmission Lines. IEEE Trans. CAD, vol. 13, no 10, October 1994, pages 1301-1305.
http://dx.doi.org/10.1109/43.317472

[6] IEEE Standard VHDL Language Reference Manual (IEEE Std 1076-2008)
IEEE Computer Society, January 2009.

[7] Peter J. Ashenden, Gregory D. Peterson and Darrell A. Teegarden: The System Designer's Guide to VHDL-AMS. Morgan Kaufmann Publishers - Elsevier, 2003.

[8] Peter J. Ashenden and Jim Lewis: VHDL-2008: Just the new Stuff. Morgan Kaufmann Publishers - Elsevier, 2008.

Ha/2014-02-28