Analysis and Simulation of Variable Structure Systems Using Bond Graphs and Inline Integration

François E. Cellier
Institute of Computational Science
ETH Zürich
CH-8092 Zürich, Switzerland
FCellier@Inf.ETHZ.CH

Matthias Krebs
DaimlerChrysler AG
HPC G202, Mercedesstr. 143/5
D-70327 Untertürkheim, Germany
Matthias.Krebs@DaimlerChrysler.Com

Keywords: computational causality, hybrid modeling, numerical simulation of discontinuous systems, object-oriented modeling, switching events.

Abstract

In this paper, a new methodology is being discussed that aids the numerical simulation of discontinuous systems involving switching events.

Many object-oriented models containing switches require a change in the selection of state variables as a function of the switch position. Such systems are classified as variable-structure systems. Their simulation leads to a division by zero at switching time.

Until now, such systems were usually simulated using non-ideal switches. However, this approach invariably leads to stiff models. Also, the simulation results obtained in this fashion may be too inaccurate.

This paper proposes a different approach, involving inline integration, that tackles the problem without requiring the introduction of spurious components.

1. INTRODUCTION

Physical systems that involve switching events frequently cause problems in their simulation. To demonstrate these problems, let us consider a simple electrical circuit containing an electrical switch. The circuit diagram is shown in figure 1.

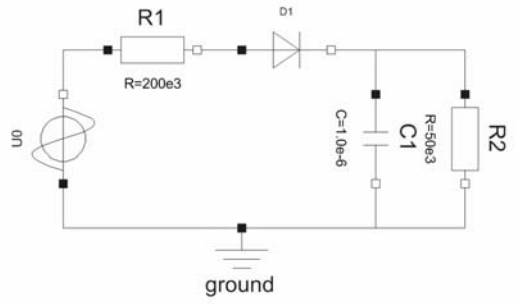


Figure 1. Circuit diagram of switching circuit

A sinusoidal input voltage of 10V and 50Hz was chosen. The diode causes the switching events. The circuit was modeled in Dymola [3] using the standard electrical library, and could be simulated without any problems. The simulation results are depicted in figure 2.

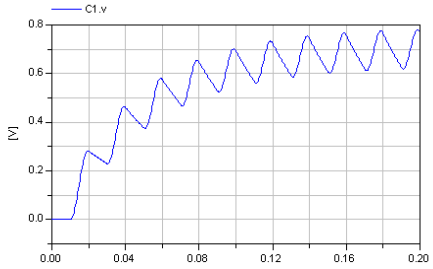


Figure 2. Simulation results of circuit simulation

To demonstrate that switching circuits can lead to numerical difficulties during simulation, we modified the circuit slightly. To this end, we replaced the source resistor by an inductor. The modified circuit is shown in figure 3.

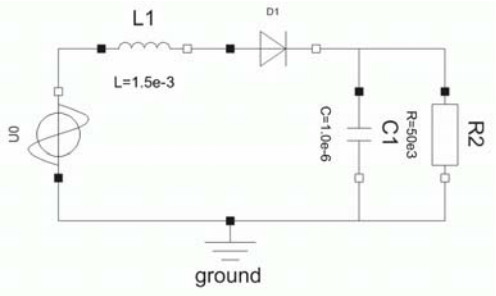


Figure 3. Circuit diagram of modified switching circuit

This circuit could not be simulated using an ideal diode. The simulation died with a division by zero. The reasons for this division by zero shall become clear in due course.

These problems occur frequently in the simulation of switching models, and therefore, Dymola replaces by

default the ideal switch by a non-ideal switch, in which the resistance of the closed switch is being increased to 1.0e-5 Ω, whereas the conductance of the closed switch is being increased to 1.0e-5 mho. Using the modified switch model, the circuit can be simulated without any problems, but the resulting circuit is very stiff, requiring a stiff system solver, and the simulation results may be inaccurate, as shall be demonstrated in due course.

2. BOND GRAPH MODELING OF SWITCHING CIRCUITS

In order to understand what went wrong in the modified circuit, it may be advantageous to look at the bond graph models of the two circuits. Figure 4 shows the bond graph model of the original circuit. The circuit was modeled in Dymola using its BondLib library [2].

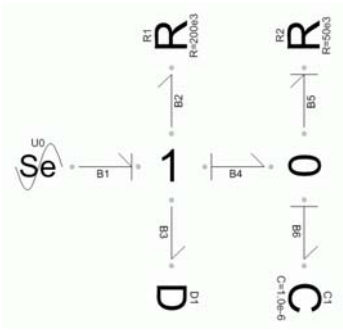


Figure 4. Bond graph of switching circuit

The D-element represents an ideal diode. For enhanced convenience and compactness of the graphical representation, BondLib offers a number of basic bond graph elements that are not contained in the standard set. The ideal diode represents an internally modulated ideal switch element.

It can be recognized that the computational causality of the diode and of the source resistor are not completed by the normal causality assignment, i.e., these two elements form an algebraic loop. In BondLib, the modeler can choose between causal and a-causal bonds. Dymola is perfectly capable of determining the computational causality of all equations on its own, i.e., a-causal bonds can be used always, but the authors of this paper recommend to use causal bonds whenever possible for increased readability and interpretability of the bond graph.

In contrast, when the source resistor is replaced by an inductor, as shown in figure 5, the preferred (integral) causality of the inductor determines the causality also on the diode.

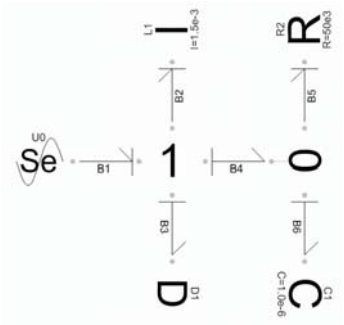


Figure 5. Bond graph of modified switching circuit

It turns out that this was precisely the problem, as shown below. Whenever the computational causality on a switching element is fixed, a division by zero will occur during the simulation. In contrast, when all switching elements are contained within algebraic loops, the simulation will not exhibit any such difficulties.

3. THE COMPUTATIONAL CAUSALITY OF SWITCHING ELEMENTS

An ideal switch can be modeled as follows:

$$0 = sw \cdot f + (1 - sw) \cdot e$$

where *sw* is a binary variable that assumes a value of 1.0, when the switch is open, and a value of 0.0, when the switch is closed. Notice that, in Dymola, equations are a-causal by default. They are only causalized in the process of compilation.

If the causality stroke of the bond to which the diode is attached is fixed as in figure 5, the switch needs to compute the effort, i.e.:

$$e = \frac{sw}{sw - 1} \cdot f$$

This equation can be simulated without any problems as long as the switch is closed. Yet, a division by zero is obtained as soon as the switch opens.

If the causality stroke of the bond to which the diode is attached is fixed in the opposite position, the switch needs to compute the flow, i.e.:

$$f = \frac{sw - 1}{sw} \cdot e$$

This equation can be simulated without any problems as long as the switch is open. Yet, a division by zero is obtained as soon as the switch closes. Hence a diode that is

attached to a causal bond will invariably lead to a division by zero in one of the two switch positions.

If the ideal switch is replaced by a non-ideal switch including small but non-vanishing leakage resistors, the division by zero is replaced by a division by a small number. Hence some variables will temporarily assume large values. Since the analytical solution of the ideal model does not contain any variables assuming large values, these artificial transients will die out quickly, i.e., the resulting model is stiff and becomes increasingly stiff, as the leakage resistors are made smaller.

Let us now look at the equations generated from the circuit containing a resistor instead of an inductor. We can compute two variables at once:

$$U_0 = 10 \cdot sin(100\pi \cdot t)$$

$$i_{R2} = \frac{u_C}{R_2}$$

since u_C is a state variable. Then we have an algebraic loop in three equations and three unknowns:

$$0 = u_{R1} - R_1 \cdot i_D$$

$$0 = sw \cdot i_D + (1 - sw) \cdot u_D$$

$$0 = U_0 - u_D - u_{R1} - u_C$$

and finally, we have two more equations that can be computed once the algebraic loop has been solved:

$$i_C = i_D - i_{R2}$$

$$\frac{du_C}{dt} = \frac{i_C}{C}$$

Using the tearing method [4], we choose i_D as the tearing variable and the switch equation as the residual equation, and we substitute the other variables from the other two equations. In this way, we obtain a set of causalized equations:

$$i_D = \frac{(sw - 1) \cdot (U_0 - u_C)}{sw + (sw - 1) \cdot R_1}$$

$$u_{R1} = R_1 \cdot i_D$$

$$u_D = U_0 - u_{R1} - u_C$$

It can be easily verified that this set of equations can be computed correctly in both switch positions, i.e., does not lead to a division by zero ever. Dymola performs the required symbolic formula transformation automatically at compile time.

It was shown by Krebs [7] that:

- A necessary condition for simulatability of the switch equations is that all switch elements are included in algebraic loops.
- If the causality strokes of all bonds attached to switch elements can be moved independently of each other, the condition is also sufficient.
- Sometimes it is not necessary that all switches can change their causality independently, because it happens frequently in switching models that not all combinations of switch positions are physically meaningful.

4. INLINE INTEGRATION AND COMPUTATIONAL CAUSALITY

The preferred (integral) causality associated with capacitors and inductors is a relic from the times, when most continuous system models were integrated using explicit integration algorithms. When an implicit integration algorithm is being used, there is numerically no difference any longer between numerical integration and numerical differentiation [6].

There may still be a computational advantage in using an explicit state-space form, although even that advantage gets blurred by the use of the tearing method [4]. In the end, the number of state variables doesn't truly matter. What matters is the number of tearing variables that need to be selected to break all algebraic loops, as these will be the iteration variables in the Newton iteration that must be carried out at each implicit integration step.

One way to preserve the explicit equation structure as much as possible is by the use of inline integration [5]. Inline integration inserts the implicit equation describing the integration algorithm symbolically into the set of model equations, thereby eliminating the strict separation between model equations and solver equations, which again is a relic from old times, when the symbolic formulae manipulation capabilities built into model compilers were poor.

Let us demonstrate the approach by means of the modified circuit while inlining the integrator used by the inductor. Partial inlining leads to a mixed-mode integration scheme [1].

For simplicity, we shall use the implicit backward Euler algorithm [1] for inlining the inductor. Thus, we now have two separate equations for the inductor:

$$u_L = L \cdot di_D$$

$$i_D = pre(i_D) + h \cdot di_D$$

where the first equation represents the model equation, and the second equation is the inlined solver equation. i_L and di_L are now two separate independent algebraic variables, and h

is the integration step size. We added one more equation (the solver equation) and one more unknown (i_D) to the set of equations.

Hence we now have an algebraic loop in four equations and four unknowns:

$$0 = u_L - L \cdot di_D$$

$$0 = i_D - pre(i_D) - h \cdot di_D$$

$$0 = sw \cdot i_D + (1 - sw) \cdot u_D$$

$$0 = U_0 - u_D - u_L - u_C$$

This time, we choose again the switch equation as the residual equation, but now, we select u_D as the tearing variable, since we prefer to compute i_D from the solver equation to avoid an unnecessary division by h.

Solving the residual equation for the tearing variable, we obtain the following replacement equation:

$$u_D = \frac{sw}{(sw - 1) \cdot L - sw \cdot h} \cdot \left[L \cdot pre(i_D) + h \cdot (u_C - U_0) \right]$$

which is again valid in both switch positions. Once we have computed u_D, we can compute the other three variables from the original equations:

$$u_L = U_0 - u_D - u_C$$

$$di_D = \frac{u_L}{L}$$

$$i_D = pre(i_D) + h \cdot di_D$$

The bond graph helps us identify, which storage elements fix the causality at any of the switch elements. The corresponding integrators will need to be inlined in order to remove the constraint on the causality at the switch. Hence the bond graph tells us immediately, which of the integrators need to be inlined.

If the causality constraint of a switch cannot be removed, because the causality is fixed by a source element rather than by a storage element, this means that the model is non-physical. In that case, we either short-circuit a voltage source or disconnect a current source by throwing the switch. Clearly, this makes no physical sense.

5. IMPLEMENTATION IN DYMOLA

Dymola offers inline integration as one of its features. Unfortunately, the implementation, as it is currently being offered, does not solve the problem for several reasons.

1. An earlier version of Dymola offered mixed-mode simulation as an option. The modeler could choose which integrators should be inlined. This feature was removed again by Dynasim, as the company decided that the feature was too difficult to use. Hence in the current version, either all integrators are being inlined or none.

2. Even if both integrators are being inlined using Dymola´s "inline" compiler switch, the division by zero remains. This is due to the way, how the inline algorithm has been implemented in Dymola. Whereas Dymola knows that switch equations that appear inside an algebraic loop must be chosen as residual equations, it does not do so in the context of inline integration. It still chooses the solver equation as the residual equation, and thereby, the division of zero in the switch equation remains, because all loop equations, except for residual equations, retain fixed causality.

3. Not all integration algorithms offered by Dymola were implemented to deal with discrete events correctly. The ones that do are DASSL and LSODAR. Yet, neither of those two algorithms is currently available as an inlined algorithm. Therefore, inline integration, as currently implemented in Dymola, cannot be used for simulating switching circuits.

For these reasons, the inlining algorithm had to be implemented manually, which was problematic also, as shall be shown in due course.

Figure 6 depicts the circuit with the inductor once more, this time using a special model, *I2*, implementing the inlined inductance element. In this model, the inlining algorithm is programmed manually, rather than relying on the inlining algorithm offered by Dymola through a compiler switch.

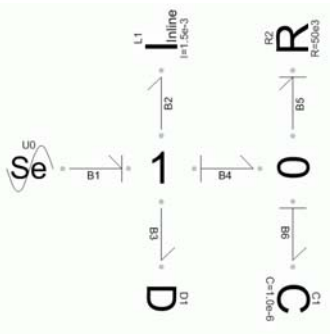


Figure 6. Bond graph of inlined switching circuit

The model of the inlined inductance element is shown in figure 7.

```
model I2 "Inlined bondgraphic inductance"
  extends BondLib.Interfaces.PassiveOnePort;
  Real df;
  Real f_pre;
  Real f_last;
  Boolean SSignal;
  parameter Real I=1 "Bondgraphic Inductance";
  parameter Real h=0.0001;

initial equation
 f_last = 0;

equation
  e = I*df;
  f = f_pre + h*df;

  SSignal = sample(0,h);

  when SSignal then
    f_pre = f_last;
  end when;
  when not SSignal then
    f_last = pre(f);
  end when;

end I2;
```

Figure 7. Model of inlined inductance element

In this model, Dymola´s built-in *pre()* operator could not be used, because the solver equation forms part of the algebraic loop, and Dymola doesn't currently support the inclusion of discrete variables in algebraic loops. Hence all loop variables had to be made continuous variables.

Furthermore, the only one of its internal variables that Dymola propagates out to the modeler is the variable *Time*. The modeler cannot make use of the internal step size that Dymola employs. Thus, the inlined inductor was implemented using a fixed-step algorithm, because otherwise, its step-size control algorithm would also have to be user implemented.

Finally, the algorithm is not aware when an event is taking place. Hence the algorithm will integrate blissfully across events, which is numerically problematic.

6. SWITCHING EVENTS AND SIMULATION ACCURACY

To check the accuracy of the simulation results, we also modeled the circuit using a regular inductor and a leaking diode, using the method that is advertised in Dymola. The circuit is shown in Figure 8.

The leaking diode uses a default value of R_0 = *1e-5* Ω for the resistance value of the closed switch, and one of G_0 = *1e-5 mho* for the conductance of the open switch.

Figure 9 compares the voltage across the capacitor as a function of time using the inlined switching circuit of Figure 6 with the leaking switching circuit of Figure 8.

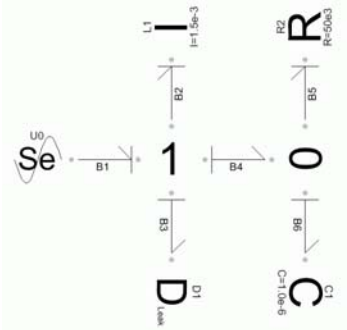


Figure 8. Bond graph of circuit with leaking diode

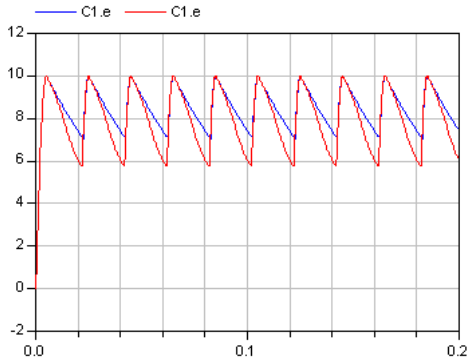


Figure 9. Comparison of simulation results

There is a large difference between the two curves. The curve with the larger oscillation was produced by the leaking diode circuit, whereas the curve with the smaller oscillation was produced by the inlined circuit.

In order to judge the quality of the results obtained, the leakage resistance, R_0, and the leakage conductance, G_0, were reduced to *1e-10* Ω and *1e-10 mho*, respectively. Now, the simulation results of the leaking circuit are no longer visibly distinguishable from those of the inlined circuit. Hence the inlined solution represents more correctly the behavior of the ideal switching circuit.

Although the default leakage resistors of the diode model offered in Dymola´s standard electrical library of *1e-5* Ω and *1e-5 mho*, respectively, seem to be very small, the resulting simulation results are very different from those of the ideal switching circuit. Using fudge parameters is clearly a dubious undertaking, as they burden the modeler with selecting appropriate values of parameters, for which no physical interpretation can be offered. Since most modelers do not understand the significance of these parameters, they will almost invariably rely of the default values provided in the standard library, and thereby may receive vastly incorrect simulation results.

7. CONCLUSIONS

In this paper, we have introduced a new method for dealing symbolically and numerically with variable structure systems, i.e., with systems that change their state variables at event times.

Such systems are encountered frequently in practice, especially in the mechanical domain. A typical example might be the shift-gear box of a car or the ejection seat in a military aircraft [8].

The variable structure system was analyzed using the bond graph approach, and the causality constraint of storage elements that determine the causality of neighboring switches was removed using the inlining technique [5].

The new algorithm was implemented in Dymola, and it was shown that avoiding the fudge parameter approach of leaking diodes helps in avoiding numerical simulation errors.

ACKNOWLEGMENTS

The algorithm presented in this paper was developed by Matthias Krebs during his exchange student visit with the University of Arizona. The exchange visit was financially supported in part by the DAAD (Deutscher Akademischer Austauschdienst) under the program IAS (Integriertes Auslandstudium), in part by Moving Pictures, Inc. on a research grant, and in its final part by a teching assistantship with Dr. Larry C. Schooley for the "ECE220 Basic Circuits" course. These financial contributions are gratefully acknowledged.

The authors also wish to express their gratitude to Dirk Zimmer, who helped with the implementation and debugging of the inlined inductance model in Dymola.

REFERENCES

[1] Cellier, F.E. and E. Kofman (2006), *Continuous System Simulation*, Springer-Verlag, New York.

[2] Cellier, F.E. and R. McBride (2003), "Object-oriented Modeling of Complex Physical Systems Using the Dymola Bond-graph Library," *Proc. ICBGM'03, 6th SCS Intl. Conf. on Bond Graph Modeling and Simulation*, Orlando, Florida, pp. 157-162.

[3] Dynasim AB (2006), *Dymola Users' Manual*, VersiTon 6.0, Lund, Sweden.

[4] Elmqvist H. and M. Otter (1994), "Methods for Tearing Systems of Equations in Object-oriented Modeling," *Proc. ESM'94, SCS European Simulation MultiConference*, Barcelona, Spain, pp. 326-332.

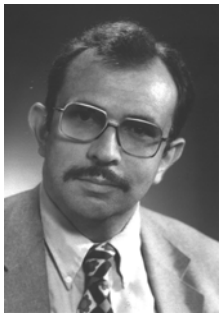
[5] Elmqvist, H., M. Otter, and F.E. Cellier (1995), "Inline Integration: A New Mixed Symbolic/Numeric Approach for Solving Differential-Algebraic Equation Systems," *Proc. ESM'95, SCS European Simulation MultiConference*, Prague, Czech Republic, pp.xxiii-xxxiv.

[6] Hu, L.A. (1991), *DBDF: An Implicit Numerical Differentiation Algorithm for Integrated Circuit Simulation*, MS Thesis, Dept. of Electr. & Comp. Engr., University of Arizona, Tucson, AZ.

[7] Krebs, M. (1997), *Modeling of Conditional Index Changes*, MS Thesis, Dept. of Electr. & Comp. Engr., University of Arizona, Tucson, AZ.

[8] Shiva, A. (2004), *Modeling Switching Networks Using Bond Graph Technique*, MS Thesis, Dept. of Aerospace & Mechanical Engr., University of Arizona, Tucson, AZ.

BIOGRAPHIES



François E. Cellier received his BS degree in electrical engineering in 1972, his MS degree in automatic control in 1973, and his PhD degree in technical sciences in 1979, all from the Swiss Federal Institute of Technology (ETH) Zurich. Dr. Cellier worked at the University of Arizona as professor of Electrical and Computer Engineering from 1984 until 2005. He recently returned to his home country of Switzerland. Dr. Cellier's main scientific interests concern modeling and simulation methodologies, and the design of advanced software systems for simulation, computer-aided modeling, and computer-aided design. Dr. Cellier has authored or co-authored more than 200 technical publications, and he has edited several books. He published a textbook on Continuous System Modeling in 1991 and a second textbook on Continuous System Simulation in 2006, both with Springer-Verlag, New York. He served as general chair or program chair of many international conferences, and served 2004-2006 as president of the Society for Modeling and Simulation International.

Matthias Krebs received his MS degree in Electrical and Computer Engineering in 1997 from the University of Arizona and his "Diplomingenieur" degree in "Technische Kybernetik" in 1998 from the University of Stuttgart. Since 1998 he works in the research and development departments of DaimlerChrysler AG. Mr.Krebs's research interests are focused on control, modeling, simulation, and identification applied to automobiles. His projects include Fuel Assistant for commercial vehicles, Predictive Cruise Control (PCC), series development of the Adaptive Cruise Control (ACC) system "Distronic," and identification of driver properties, such as driver workload, drowsiness or driver awareness.