Anmerkungen zur Modellierung hybrider dynamischer Systeme

Joachim Haase[1], Ewald Hessel[2]

[1] Fraunhofer-Institut für Integrierte Schaltungen/Institutsteil Entwurfsautomatisierung Dresden

[2] Hella KGaA Hueck & Co. Lippstadt

joachim.haase@eas.iis.fraunhofer.de, ewald.hessel@hella.com

Hybride dynamische Systeme kombinieren zeitkontinuierliches und zeitdiskretes Verhalten. Sie stellen ein adäquates Modellierungskonzept für gemischt analog-digitale Systeme dar, die mit Verhaltenbeschreibungssprachen wie VHDL-AMS, Verilog-AMS und auch Modelica beschrieben werden können. Im Beitrag werden unter anderem numerische Aspekte, die bei der Modellerstellung in diesem Zusammenhang zu beachten sind, diskutiert.

1 Einleitung und Problembeschreibung

Hybride dynamische Systeme sind in den 1990er Jahren als Modellierungskonzept zur Beschreibung von Systemen mit zeitkontinuierlichem und zeitdiskretem Verhalten ausführlich untersucht worden. Ausgangspunkt ist dabei ein Zustandsautomat. Durch die einzelnen Zustände (Moden) wird ein unterschiedliches zeitkontinuierliches Verhalten (flow) festgelegt. Außerdem wird beschrieben, unter welchen Bedingungen ein Übergang von einer Mode in einen andere erfolgt (guards) und wie Anfangswerte für die Fortsetzung des kontinuierlichen Verhaltens in der neuen Mode zu bestimmen sind (set action).

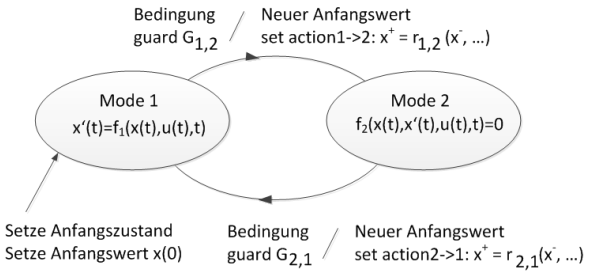

Abbildung 1. Hybrides dynamisches System

Abbildung 1 zeigt ein Beispiel für ein solches System. Eigenschaften spezieller derartiger Systeme sind unter dem Gesichtspunkt einer formalen Analyse ausführlich theoretisch untersucht worden. Im Zuge der Beschäftigung mit Untersuchungen zu cyber-physischen Systemen (cyber-physical systems) hat dieses Konzept erneut an Interesse gewonnen [1], stellt es doch einen systematischen Zugang zur Beschreibung gemischt analog-digitaler Systeme – auch für die Modellierung mit Verhaltensbeschreibungssprachen wie VHDL-AMS, Verilog-AMS und Modelica – dar.

2 Formale Beschreibung

Auf die Wiedergabe einer strengen mathematischen Definition hybrider dynamischer Systeme soll an dieser Stelle verzichtet werden. Die für die Modellbildung im Folgenden benötigten wesentlichen Aspekte sollen nur kurz referiert und diskutiert werden. Die Darlegungen stützen sich auf [2, 3].

Ein hybrides dynamisches System ist durch eine endliche Menge Q diskreter Moden (auch als diskrete Zustände bezeichnet) beschrieben. Jeder Mode $i \epsilon Q$ kann ein Definitionsbereich $D_i \subseteq R^{n_i}$ zugeordnet werden. Die ganzzahligen Werte $n_i < \infty$ können für die einzelnen Moden prinzipiell unterschiedliche Werte annehmen. Das zeitkontinuierliche Verhalten in einer Mode wird allgemein durch eine differentielle Inklusion F (flow map) mit $x'(t) \in F_i(x(t), t)$ mit $x(t) \in D_i$ beschrieben. Dargestellt werden kann eine derartige Inklusion beispielsweise durch ein explizites oder implizites Differentialgleichungssystem. Die bei den F_i zu berücksichtigende Abhängigkeit von der Zeit t ergibt sich z.B. als Folge von zeitabhängigen Eingangsgrößen $u: R^+ \cup \{0\} \rightarrow U \subseteq R^{n_U}$. Der Ausgang $y(t) \in Y \subseteq R^{n_Y}$ des hybriden dynamischen Systems wird durch u und den Verlauf x der aktuellen Mode festgelegt.

Mögliche Übergänge von einer Mode in eine andere können durch einen Graphen erfaßt werden, dessen Knoten die Menge Q der Moden bildet. Die Übergänge sind gerichtete Kanten in diesem Graphen. Für eine Kante (edge) e sei $s(e)$ die Ausgangsmode und $z(e)$ die Zielmode (target). Ein durch eine Kante e festgelegter Übergang wird aktiviert, wenn eine Bedingung (guard) $G_{s(e),z(e)}(x, u, t)$ erfüllt ist. Ferner wird durch den Übergang zur Zeit $t = t_a$ bestimmt

wie aus dem Wert $x(t_a)$ in $D_{s(e)}$ bei Eintreten einer Bedingung zur Kantenaktivierung ein Wert $x^+(t_a)$ in $D_{z(e)}$ abgeleitet wird (set action).

In der Initialisierungsphase zur Zeit $t = 0$ und beim Übergang von einer Mode in eine andere zur Zeit t_a muss gesichert sein, dass es sich beim Startwert *x(0)* bzw. bei $x^+(t_a)$ um konsistente Anfangswerte [4] für den weiteren Fluss handelt. Für explizite Differentialgleichungssysteme der Form $x'(t) = f(x(t), u(t), t)$ ist die Bestimmung konsistenter Anfangswerte in der Regel evident. Schwieriger ist sie bei impliziten Differentialgleichungssystemen der Form $f(x(t), x'(t), u(t), t) = 0$ oft dann, wenn die Komponenten von *x*, deren Ableitungen zur Beschreibung benötigt werden, keine Zustandsgrößen zur Charakterisierung des Flusses sind. Die Bestimmung konsistenter Anfangswerte ist selbst bei linearen Systemen der Form $A \cdot x'(t) + B \cdot x(t) = C \cdot u(t)$ mit singulärer Koeffizientenmatrix *A* u.U. aufwendig [5]. So ist es nicht weiter verwunderlich, dass im Mittelpunkt theoretisch orientierte Publikationen zu hybriden dynamischen Systemen Flüsse stehen, die mit expliziten Differentialgleichungen beschrieben werden. Bei der Modellerstellung ist aber auch sonst sicherzustellen, dass beim Übergang von einer Mode in eine andere konsistente Anfangswerte für die weitere Simulation zur Verfügung gestellt werden [6].

Schwierig kann sich auch die Bestimmung von Anfangswerten für die weitere Simulation gestalten, wenn beim Übergang von einer Mode *i* zu einer Mode *j* mit den zugehörigen Definitionsbereichen $D_i \subseteq R^{n_i}$ bzw. $D_j \subseteq R^{n_j}$ die Dimensionen n_i und n_j verschieden sind. Simulationsprogramme, die mit Verhaltensbeschreibungssprachen wie VHDL-AMS und Verilog-AMS arbeiten, unterstützen diesen Fall nicht. Die Anzahl der Unbekannten des zu lösenden Differentialgleichungssystems in den unterschiedlichen Modi ändert sich nicht. Eine Änderung der Anzahl der Unbekannten und damit der Dimension des Differentialgleichungssystems kann beispielsweise sinnvoll sein, wenn eine Änderung der Modellgenauigkeit oder der Modellstruktur während einer Simulation erforderlich ist [7]. Die Vorgabe neuer Anfangswerte beim Übergang von einer Mode in eine andere ist auch hier ein durch den Modellersteller zu lösendes Problem.

Damit das Verhalten eines hybriden dynamischen Systems determiniert ist, dürfen die Bedingungen, die zur Aktivierung eines Übergangs in eine andere Mode führen für verschiedene von der aktuellen Mode wegführende Kanten nicht gleichzeitig erfüllt sein [3].

Die Bedingungen zum Übergang zwischen den Moden dürfen unter Berücksichtigung neu zu setzender Anfangswerte in der neuen Mode nicht dazu führen, dass in dem Graph, dessen Knoten die Moden bilden, Schleifen existieren, in denen für bestimmte *x* Werte alle Bedingungen, die den Kanten einer solchen Schleife zugeordnet sind, erfüllt sind.

Ferner müssen Bedingungen angegeben werden, wie die Anfangswerte *x(0)* für die einzelnen Moden zur Anfangszeit $t = 0$ zu bestimmen sind. Die Mode zur Anfangszeit kann – soweit möglich – vorgegeben werden oder ergibt sich auf Grund des Wertes *u(0)* des Eingangssignals.

3 Beispiele

Zahlreiche Modellbeschreibungen lassen sich unter Verwendung des Konzepts hybrider dynamischer Systeme erstellen. Beispiele dafür sind Ladungspumpen (charge pumps), Anschläge, Lose, Integratoren mit Begrenzung u.a. Einige Beispiele sollen kurz skizziert werden.

Modell für idealen Generator

Für Systemuntersuchungen elektrischer Bordnetze werden Modelle für geregelte Generatoren benötigt. Für diesen Zweck existieren eine Reihe ausgefeilter Modelle. Für erste Untersuchungen stehen aber häufig nur Kennlinien zur Verfügung, die den maximalen Generatorstrom in Abhängigkeit von Drehzahl und Temperatur beschreiben.

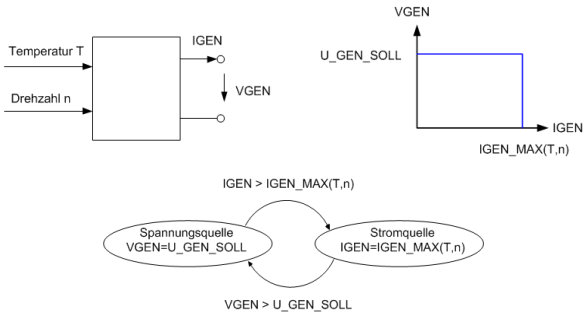

Abbildung 2. Modell für einen idealen Generator

In Abbildung 2 ist das Vorgehen skizziert. Das Bild zeigt das ideale Verhalten an den Klemmen des Generators und die Beschreibung durch einen hybriden (dynamischen) Automaten. Die Abhängigkeit des maximalen Generatorstroms von Temperatur und

Drehzahl IGEN_MAX(T, n) ergibt sich dabei aus dem Generatorkennlinienfeld. Welche Mode sich zur Anfangszeit *t=0* einstellt, ergibt sich aus dem Netzwerk, in dem das Modell verwendet wird.

Memristor

Dass das Konzept auch für neuere Entwicklungen wie beispielsweise bei der Modellerstellung für Memristoren auf Halbleiterbasis Vorteile bietet, soll im Folgenden kurz skizziert werden.

Ein Memristor ist das von L.O. Chua 1971 beschriebene fehlende Netzwerkelement, bei dem das Verhalten eines Zweipols durch eine Beziehung $g(\varphi, q) = 0$ zwischen Magnetfluss φ und Ladung q beschrieben wird [8]. Für ein stromgesteuertes Memristor-Ein-Tor (mit einer Zustandsgröße $x \epsilon R$) wird das Verhalten von Klemmenspannung v und –strom i beschrieben durch [9]

$$x'(t) = f(x(t), i(t), t)$$
$$v(t) = R(x(t), i(t), t) \cdot i(i)$$

R ist dabei die sogenannte Memristanz. Das Interesse an diesem Konzept ist vor wenigen Jahren gewachsen als Memristoren mit einem Dünnschichtverbund realisiert werden konnten und sich damit ein Weg für neuartige integrierte Speicherstrukturen zu eröffnen schien. Das mathematische Modell eines derartigen Memristors, das an dieser Stelle kurz diskutiert werden soll, führt auf eine Beschreibung der Form [10]

$$w'(t) = \mu_V \cdot \frac{R_{ON}}{D} \cdot i(t)$$
$$v(t) = \left(R_{ON} \cdot \frac{w(t)}{D} + R_{OFF} \cdot \left(1 - \frac{w(t)}{D}\right)\right) \cdot i(t)$$

Die Gesamtbreite der aktiv zu beeinflussenden Schicht ist D. w kann Werte zwischen 0 und D annehmen, aber keine Werte außerhalb des Intervalls [0, D]. Damit ergeben sich für die Memristanz in Abhängigkeit von w Werte im Intervall $[R_{ON}, R_{OFF}]$. μ_V charakterisiert die Ladungsträgermobilität in der Schicht der Breite D. Die erste der beiden Gleichungen beschreibt eine lineare Drift der Ladungsträger.

Um sicherzustellen, dass die Werte von w das Intervall [0, D] nicht verlassen, werden verschiedendste Fensterfunktionen vorgeschlagen, mit denen die erste der beiden obigen Gleichungen multipliziert wird. Durch derartige Fensterfunktionen soll erreicht werden, dass $w'(0) = w'(D) = 0$ bei $i(t) < 0$ bzw. $i(t) > 0$ gesichert wird und dass bei Strömen $i(t) > 0$ und $i(t) < 0$ die w-Werte 0 bzw. D wieder verlassen werden können. Diese Fensterfunktionen dienen darüberhinaus einer nichtinearen Beschreibung des Driftverhaltens [11].

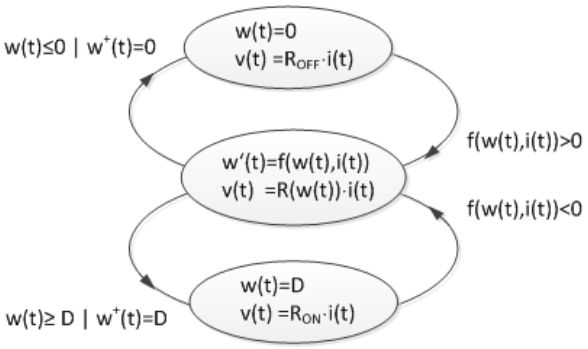

Abbildung 3. Memristormodell mit $w(t) \in [0, D]$

Abbildung 3 zeigt für ein stromgesteuertes Memristor-Ein-Tor mit einer Dünnschichtanordnung ein hybrides dynamisches Modell, das die Einhaltung der Bedingung $w \epsilon [0, D]$ sichert, wobei keine besonderen Anforderungen an die Fensterfunktion gestellt werden. Aus der Vorgabe des Anfangswertes *w(0)* ergibt sich die Mode zur Zeit $t = 0$.

Springender Ball

Das folgende Differentialgleichungssystem liefert eine (sehr) einfache Beschreibung eines springenden Balles:

$$s'(t) = v(t)$$
$$v'(t) = -g$$

s ist die Höhe des Balles über dem Boden, v die Fallgeschwindigkeit und g die Gravitationskonstante.

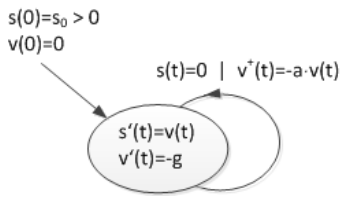

Abbildung 4. Springender Ball

Wenn der Ball fällt und auf den Boden zur Zeit t_b $(s(t_b) = 0)$ mit der Geschwindigkeit $v(t_b)$ stößt (bump event), springt der Ball mit der Geschwindigkeit $v^+(t_b) = -a \cdot v(t_b)$ mit $0 < a < 1$ zurück. Das heißt, der Stoß ist unelastisch. Die Anfangshöhe des Balles sei $s(0) = s_0 > 0$ und die Anfangsgeschwin-

digkeit $v(0) = 0$. Abbildung 4 zeigt das auf dieser Beschreibung basierende hybride dynamische System.

Der Anfangszustand ist bei diesem Beispiel eindeutig charakterisiert. Der springende Ball verdeutlicht ein mögliches Problem, das bei der Beschreibung mit hybriden dynamischen Systemen auftreten kann.

Es kann leicht gezeigt werden (siehe z.B. [12]), dass der Ball zum ersten Mal nach der Zeit $t_1 = \sqrt{\frac{2 \cdot s_0}{g}}$ auf den Boden auftrifft. Die Zeit zwischen dem zweiten und ersten Auftreffen ist $2a\sqrt{\frac{2 \cdot s_0}{g}}$ und zwischen dem dritten und zweiten ist $2a^2\sqrt{\frac{2 \cdot s_0}{g}}$. Die Zeiten zwischen dem Auftreffen des Balles werden immer kleiner. Zeitpunkt des k-ten Auftreffens ist $t_k = 2 \cdot \sqrt{\frac{2 \cdot s_0}{g}} \cdot \left(\frac{1}{2} + a \cdot \frac{1-a^{k-1}}{1-a}\right)$. Bis zur Zeit $t_\infty = \frac{1+a}{1-a} \cdot \sqrt{\frac{2 \cdot s_0}{g}}$ beschreibt das hybride dynamische System aus Abbildung 4 das Verhalten. Die Zeit t_∞ wird Zeno-Zeit genannt. Charakteristisch ist in diesem Fall, dass vor der Zeno-Zeit die Differenzen zwischen den Zustandswechseln immer kleiner werden, unendlich viele Zustandswechsel stattfinden und die Zeno-Zeit endlich ist. Damit ist das Verhalten des Systems nach dieser Zeit formal nicht beschrieben. Im Allgemeinen wird dieses Problem durch Einführung einer Post-Zeno-Mode gelöst, mit der das Verhalten nach Erreichen des Zeno-Punktes beschrieben wird (siehe [13, 14]). Das heißt, Abbildung 4 beschreibt das Verhalten für die Simulation nicht vollständig.

4 Numerische Aspekte

Für die Simulation hybrider dynamischer Systeme existiert eine Reihe von Simulationswerkzeugen. Goebel u.a. [15] verweisen auf Matlab/Simulink, Modelica-basierte Simulatoren, Ptolemy u.a. Zu ergänzen sind in dieser Zusammsnstellung Simulatoren für gemischt analog-digitale Systeme (AMS systems–analog mixed signal systems), die VHDL-AMS- und/oder Verilog-AMS [16, 17] unterstützen. Unterstützt werden muss auf Grund der in Abschnitt 2 beschriebenen Merkmale hybrider dynamischer Systeme

- Vorgabe von Anfangswerten oder Bedingungen für die Bestimmung von Anfangswerten in der Initialisierungsphase
- Beschreibung des Verhaltens in den einzelnen Moden unter Verwendung von expliziten Differentialgleichungssystemen oder differential-algebraischen Gleichungssystemen
- Erkennen von Bedingungen (guards), die zum Übergang von einer Mode in eine andere führen
- Vorgabe neuer Anfangswerte nach dem Übergang von einer Mode in eine andere
- Zugriff auf die aktuelle Mode, in der sich ein Modell bei der Simulation befindet.

Unter praktischem Gesichtspunkt müssen die Modelle robust sein. Iterationsschleifen zwischen unterschiedlichen Moden in der Initialisierungsphase und bei der Simulation im Zeitbereich, immer kürzer werdende Umschaltzeiten zwischen den einzelnen Zuständen, z.B. Zeno-Effekt beim fallenden Ball, und das Verlassen des Definitionsbereiches für den Fluss in einer Mode müssen vermieden werden. Außerdem muss das Eintreten der Bedingungen für den Übergang in eine andere Mode (guard) sicher erkannt werden. Es ist sicherzustellen, dass die in Abschnitt 2 skizzierten Anforderungen für hybride dynamische Bedingungen erfüllt sind. Dabei ist die begrenzte numerische Genauigkeit bei der Ermittlung der Lösungen für differential-algebraische Gleichungssysten in Folge der Diskretisierung zu berücksichtigen. Bedingungen, die auf die Gleichheit mit Null testen, sind nicht zu realisieren. Sie müssen durch die Abfrage, ob Werte größer oder kleiner Null (oder ein entsprechend kleiner ε-Wert) sind, in geeigneter Weise ersetzt werden.

Besonders Simulatoren, die gemischt analog-digitale Verhaltensbeschreibungen unterstützen, sind für die Simulation hybrider dynamischer Systeme geeignet [18]. Die Moden, in der sich Modelle befinden, werden unter Verwendung zeitdiskreter Signale gespeichert. Den Fluss in einer Mode beschreiben zeitkontinuierliche Verläufe. Zur Aktivierung der Bedingungen (guards) zum Übergang von einer Mode in eine andere werden zu diskreten Zeitpunkten Ereignisse generiert. Diese können aus diskreten Signalen oder dem Über- oder Unterschreiten von Schwellwerten für zeitkontinuierliche Verläufe abgeleitet werden. VHDL-AMS stellt dafür das ABOVE-Attribute zur Verfügung, Verilog-AMS die cross() und above() Funktionen. Zum Setzen von Anfangswerten in der Initialisierungsphase und nach Übergang von einer Mode in eine andere verfügen Verhaltensbeschrei-

bungssprachen für analog-digitale Systeme über entsprechende Sprachkonstrukte wie z.B. die break-Anweisung in VHDL-AMS.

Werden die Bedingungen nur zu diskreten Zeitpunkten ausgewertet, können die Wechsel zwischen zwei Moden nicht unendlich nahe beienander liegen. Der Zeno-Effekt wird – zumindest theoretisch – vermieden. Kritisch erweist sich bei der Erstellung von Modellen die sichere Aktivierung der Bedingungen, wenn diese aus dem Über- oder Unterschreiten von Schwellwerten abgeleitet werden. Folgende Maßnahmen erweisen sich als hilfreich, wenn in diesem Zusammenhang mit einem erstellten Modell Schwierigkeiten auftreten

- Wird in Bedingungen (für unterschiedliche Kanten des unter Verwendung der Moden gebildeten Graphen) sowohl das Über- als auch das Unterschreiten eines Schwellwertes abgefragt, sollte diese Abfrage durch Kontrolle des Über- oder Unterschreitens des Schwellwertes + oder – einem kleinen Wert ε ersetzt werden.
- Abfrage der Bedingungen in vorgegebenen Zeitabständen. Das Modell wird dadurch in der Regel langsamer, aber auch weniger störanfällig.
- Einfügen eines Tests, ob der Definitionsbereich des Flusses für die aktuelle Mode verlassen wird, und Abfrage der Bedingungen an den Kanten, die vom Knoten mit der aktuellen Mode wegführen.
- Der Anfangswert, der beim Übergang von einer Mode in eine andere gesetzt wird, sollte im Innern und nicht auf dem Rand des neuen Definitionsbereiches liegen.

Im Übrigen sind die in Abschnitt 2 skizzierten Bedingungen einzuhalten.

5 Modellaustausch

In [19] wird eine allgemeine Modellschnittstelle - Functional Mock-up Interface for Model Exchange (FMI) - beschrieben. Diese Modellschnittstelle ist vordergründig für den Austausch von Modellen zwischen Modelica-basierten Simulatoren entwickelt worden. Sie eröffnet aber auch eine Möglichkeit unter Verwendung von Modelica entwickelte Modelle in andere Simulationsumgebungen zu integrieren. Durch die Schnittstelle werden C-Funktionen und deren Funktionalität zur Beschreibung des Klemmenverhaltens eines Modells definiert. Das FMI for Model Exchange definiert Funktionen zur Beschreibung spezieller dynamischer Systeme. Die Flüsse in den einzelnen Moden werden unter Verwendung expliziter Differentialgleichungssysteme beschrieben (Funktion fmiGetDerivatives). Für die Auswertung der Bedingungen (guards) zum Übergang von einer Mode in eine andere gibt es die Funktion fmiGetEventIndicators. Diese Funktion wertet Nulldurchgänge spezieller zeitkontinuierlicher Verläufe (event indicators) und im Voraus ermittelte Ereigniszeitpunkte T_{next} aus. Nach Auftreten eines Ereignisses werden ggf. eine neue Mode und neue Anfangswerte für die Ermittlung des Flusses in der neuen Mode mit der Funktion fmiEventUpdate bestimmt. Für die Bestimmung von Anfangswerten in der Initialisierungsphase gibt es die Funktion fmiInitializeModel. Funktionen stehen zum Lesen der Eingangsgrößen u des Modells und zum Bestimmen der Ausgangsgrößen y zur Verfügung. Unter Verwendung dieser Grundfunktionen können hybride dynamische Systeme in anderen Beschreibungssprachen modelliert werden. VHDL-AMS-basierte Simulatoren müssen dazu die Verwendung reellwertiger quantity-Vektoren in „simultaneous statements“, das ABOVE-Attribute zum Erkennen der Nulldurchgänge der „event indicators“ und das Setzen von Anfangswerten in der Initialisierungphase und nach dem Übergang von einer Mode in eine andere mit der break-Anweisung unterstützen. Nicht zuletzt muss eine (simulatorspezifische) C-Schnittstelle für den Funktionsaufruf unterstützt werden. Es wurden eine Reihe kleinerer Beispiele umgesetzt, mit denen gezeigt werden konnte, dass dieses Vorgehen möglich ist.

Da sich aber beispielsweise der VHDL-AMS-Simulationszyklus von der durch das FMI-Dokument beschriebenen Aufrufreihenfolge unterscheidet, ist in vielen Fällen, in denen nur Interesse an einer Zeitbereichssimulation besteht, einem anderen Weg der Vorzug zu geben. Bei einem FMI-basierten Modell ist gesichert, dass die Flussgrößen, die in den einzelnen Moden ermittelt werden, nicht in ein gemeinsames Gleichungssystem eines zu simulierenden Gesamtsystems einfließen. Das Modell wird mit einem Löser zur Auswertung der Flüsse in den Moden verknüpft. Eine übersetzte Version des Modells kann dann für unterschiedliche Rechnerplattformen zur Verfügung gestellt werden. Für den Simulator, in den das Modell eingebunden wird, muss ein C-Interface existieren.

6 Literatur

[1] Lee, E.A.; Seshia, S.A. *Introduction to Embedded Systems.* LeeSeshia.org, 2011.

[2] Goebel, R.; Sanfelice, R. G.; Teel, A. R.: *Hybrid Dynamical Systems:Modeling, Stability, and Robustness*. Princeton University Press, 2012.

[3] Abate, A.; D'Innocenzo, A.; Di Benedetto, M.D.; Sastry, S.: Unterstanding Deadlock and Livelock Behaviors in Hybrid Control Systems. In: *Nonlinear Analysis:Hybrid Systems*, vol. 3, vssue 2, May 2009, pp. 150-162.

[4] Leimkuhler, B.; Petzold, L.R.; Gear, C.W.: Approximation Methods for the Consistent Initialization of Differential-Algebraic Equations. In: *SIAM J. Numer. Anal.,* vol. 28, issue 1, pp. 205-226, 1991.

[5] Reißig, G.; Boche, H.; Barton, P.I.: On Inconsistent Initial Condidions for Linear Time-Invariant Differential-Algebraic Equations. In: *IEEE Trans. Circuits Syst. I,* vol. 49, issue 11, pp. 1646-1648, November 2002.

[6] Haase, J.: Rules for Analog and Mixed-Signal VHDL-AMS Modeling. In: Grimm, C. (Hrsg): *Languages for System Specification*. Springer, 2004.

[7] Bastian, J.; Clauß, C.; Enge-Rosenblatt, O.; Schneider, P.: MOSILAB – a Modelica solver for multiphysics problems with structural variability. In: *Proc. 1st Conference on Multiphysics Simulation – Advanced Methods for Industrial Engineering*. Bonn, 12 S., June 22-23, 2010.

[8] Chua, L.O.: Memristor – the Missing Circuit Element. In: *IEEE Trans. Circuit Theory*, vol. 18, issue 5, pp. 507-519, September 1971.

[9] Chua, L.O.; Kang, S.M.: Memristive Devices and Systems. In: *Proc. of the IEEE*, vol. 64, issue 2, pp. 209-223, February 1976.

[10] Strukov, D.V.; Snider, G.S.; Stewart, D.R.; Williams, R.S.: The Missing Memristor Found. In: *Nature*, vol. 453, pp. 80-83, May 2008.

[11] Kvatinsky, S.; Friedman, E.G.; Kolodny, A.; Weiser, U.C.: TEAM: ThrEshold Adaptive Memristor Model. In: *IEEE Trans. Circuits Syst. I*, vol. 60, issue 1, pp. 211-221, January 2013.

[12] Oltean, V.E.: On Simulation of Zeno Hybrid Systems. In: *Rev. Roum. Sci. Techn.. – Électrotechn. et Énerg,* vol. 52, issue 2, pp. 229-239, 2007. Online: http://revue.elth.pub.ro/upload/285662art08.pdf

[13] Zhang, F.; Yeddanapudi, M.; Mosterman, P.J.: Zero-Crossing Location and Detection Algorithms For Hybrid System Simulation. In: *Proc. of the 17th IFAC World Congress*, 2008. DOI: 10.3182/20080706-5-KR-1001.01346

[14] Zheng, H.: *Simulating Zeno Hybrid Systems Beyond Their Zeno Points*. Technical Report No. UCB/EECS-2006-114, September 8, 2006.

[15] Goebel, R.; Sanfelice, R.G.; Teel, A.: Hybrid dynamical systems. In: IEEE Trans. *Control Systems,* vol. 29, issue 2, pp. 28-93, April 2009. DOI: 10.1109/MCS.2008.931718

[16] *Behavioral languages-Part 6: VHDL Analog and Mixed Signal Extensions: IEEE Std 1076.1 IEC 61691-6 Edition 1.0 2009-12*. DOI: 10.1109/IEEESTD. 2009.5464492.

[17] *Verilog-AMS Language Reference Manual*: *Version 2.3.1*. Accelera, June 2009.

[18] Nutaro, J.; Kuruganti, P.T.; Protopopescu, V.; Shankar, M.: The Split System Approach to Managing Time in Simulations of Hybrid Systems having Continuous and Discrete Event Components. In: *Simulation*, vol. 88, issue 3, pp. 281-298, 2012.

[19] *FMI for Model Exchange and Co-Simulation 2.0* URL: https://www.fmi-standard.org/downloads