Analyse zeitbewerteter Bedingung/Ereignis-Systeme mittels Echtzeitautomaten-Tools [1]

S. Kowalewski, J. Preußig, S. Engell
Lehrstuhl für Anlagensteuerungstechnik, Fachbereich Chemietechnik,
Universität Dortmund, D-44221 Dortmund
Tel.: 0231/755-5128, Fax: 0231/755-5129
Email: {stefan | joerg | engell}@ast.chemietechnik.uni-dortmund.de

R. Huuck, Y. Lakhnech, L. Urbina
Lehrstuhl für Softwaretechnologie, Institut für Informatik und praktische Mathematik,
Christian-Albrechts-Universität Kiel, D-24105 Kiel
Tel.: 0431/5604-87 , Fax: 0431/5661-43
Email: {rh | yl | lu}@informatik.uni-kiel.de

Kurzfassung: Der Beitrag beschreibt, wie aus zeitbewerteten Bedingung/Ereignis-Systemen bestehende, modular aufgebaute Modelle mit Hilfe von Werkzeugen für Echtzeitautomaten automatisch analysiert werden können. Dies geschieht, indem eine Übersetzungsvorschrift angewendet wird, die einem zeitbewerteten Bedingung/Ereignis-System einen Echtzeitautomaten mit äquivalentem Verhalten zuordnet. Der Ansatz wird anhand des aus der Literatur bekannten Bahnschranken-Beispiels illustriert. Als Werkzeug wird KRONOS eingesetzt.

Stichworte: Analyse von Echtzeitsystemen, Bedingung/Ereignis-Systeme, Echtzeitautomaten, KRONOS.

---

[1] Die vorgestellte Forschungsarbeit wird von der DFG im Rahmen des Schwerpunktprogramms KONDISK (Analyse und Synthese kontinuierlich-diskreter technischer Systeme) gefördert.

1 Einleitung

Die Analyse von Echtzeitsystemen wird zur Zeit sowohl in der Theoretischen Informatik als auch in der Systemtheorie als Ingenieurwissenschaft intensiv untersucht. Die relativ junge Erkenntnis, daß wesentliche Eigenschaften der zunehmend an technischer Bedeutung gewinnenden Systeme mit gemischt-diskret/kontinuierlicher Dynamik (sogenannter *hybrider Systeme*) algorithmisch im allgemeinen nur auf der Basis einer Approximation durch zeitbewertete diskrete Modelle möglich ist, verschafft diesen Modellen eine neue Aktualität.

Von Ingenieursseite wurden unter anderem *zeitbewertete Bedingung/Ereignis-Systeme* (abgekürzt: *ZBES*) als Beschreibungsform für Echtzeitsysteme vorgeschlagen [KoPr96]. Sie bauen auf den von Sreenivas und Krogh eingeführten, ursprünglich rein diskreten Bedingung/Ereignis-Systemen (B/E-Systemen) auf [SrKr91], mit denen die aus der kontinuierlichen Systemtheorie vertraute Darstellung eines Gesamtsystems durch verschaltete Blöcke möglich ist. Dieser Ansatz ist vor allem bei der Beschreibung komplexer technischer Systeme von Vorteil und findet in jüngster Zeit zunehmende Akzeptanz (siehe z. B. [GRE94], [HaRa95], [Hol96], [EnHo96]). Eine Einführung in die B/E-Systeme findet sich im Tutoriumsbeitrag [KE97] in diesem Band.

In der Informatik ist der *Echtzeitautomat* (hier mit *EZA* abgekürzt) ein geläufiges Beschreibungsmittel [AlDi94]. EZA stellen eine Erweiterung der endlichen Automaten um reellwertige Variablen dar. Diese repräsentieren Uhren, von deren Ständen diskrete Zustandsübergänge des Automaten abhängig gemacht werden können. Zur Analyse von EZA wurden in der Informatik mehrere leistungsfähige Rechnerwerkzeuge entwickelt, von denen HyTech [www1], KRONOS [www2], und UPPAAL [www3] die bekanntesten sind.

Die Verfügbarkeit dieser Tools bildete die Motivation, ihre Eignung zur Analyse von ZBES zu untersuchen. Dazu waren zunächst die beiden Modellformen hinsichtlich ihrer Expressivität zu vergleichen. Dieser Vergleich wird in [HLU+97] beschrieben und hat als wichtigstes Ergebnis, daß EZA und ZBES im wesentlichen die gleiche Modellierungsmächtigkeit besitzen. In [HLU+97] wird darüberhinaus eine einfach zu realisierende Übersetzungsvorschrift angegeben, mit der jedem ZBES ein äquivalenter EZA zugeordnet werden kann. Auf der Basis dieser Transformation ist es nun leicht möglich, beliebige ZBES-Modelle in EZA-Darstellung zu bringen und Spezifikationen

für die ZBES (zum Beispiel in einer Temporalen Logik formulierte Eigenschaften) mit den genannten Tools zu analysieren. Wie dies geschieht, wird in diesem Beitrag näher beschrieben. Dabei soll der Schwerpunkt nicht auf der formalen Darstellung liegen. Stattdessen werden die zugrundeliegenden Prinzipien anhand eines durchgängigen Beispiels, dem aus der Literatur bekannten „Bahnschrankenproblem“, veranschaulicht. Als Werkzeug zur Analyse dieses Beispiels wird KRONOS benutzt.

Der Beitrag gliedert sich folgendermaßen. Im folgenden Abschnitt 2 wird zunächst das Beispiel kurz eingeführt. Die Abschnitte 3 und 4 fassen die Grundlagen der beiden Modellformen ZBES und EZA zusammen und illustrieren beide Darstellungsformen mit Hilfe des Beispiels. Die Transformation von ZBES in EZA wird in Abschnitt 5 dargestellt. Abschnitt 6 zeigt die Resultate der Analyse mit KRONOS. Der Beitrag schließt mit einer Zusammenfassung und Diskussion der Ergebnisse.

2 Das Beispiel

Die Vorgehensweise wird an dem sogenannten Bahnschranken-Beispiel erläutert, das ursprünglich wohl von Leveson und Stolzy [LeSt87] stammt und heute zu einem Standardbeispiel für Methoden zur Sicherheitsanalyse von Echtzeitsystemen geworden ist. Auch im Rahmen dieser Tagungsreihe wurde es schon aufgegriffen [Thu95]. Die Problemstellung ist folgende: Eine Steuerung hat die Aufgabe, eine Bahnschranke bei Durchfahrt eines Zuges rechtzeitig zu schließen und danach wieder zu öffnen. Sowohl das Herannahen eines Zuges auf eine bestimmte Distanz als auch das Entfernen des Zuges werden der Steuerung über Sensorsignale mitgeteilt. Es wird angenommen, daß zwischen dem Signalisieren eines herannahenden Zuges mindestens zwei Zeiteinheiten vergehen, bevor der Zug den Übergang erreicht, und daß er nach höchstens fünf Zeiteinheiten den Übergangsbereich wieder verläßt. Die Schranken benötigen maximal eine Zeiteinheit, um zu sich zu schließen, und zwischen einer und zwei Zeiteinheiten, um sich zu öffnen. Die Fragestellung bei der Analyse ist nun, ob eine gegebene Realisierung der Steuerung, zu der eine bestimmte Wartezeit gehört, die die Steuerung zwischen dem Erkennen eines sich nähernden Zuges und dem Schließen der Schranken wartet, in jedem Fall verhindert, daß ein Zug den Übergangsbereich durchfährt, ohne daß die Schranken geschlossen sind. Dabei muß sinnvollerweise die Lösung ausgeschlossen werden, daß die Steuerung die Schranken ständig geschlossen läßt.

3 Zeitbewertete Bedingung/Ereignis-Systeme

Die Grundidee des B/E-System-Paradigmas (siehe auch [EK97] in diesem Band) besteht darin, das Zusammenwirken von Teilsystemen, die ein diskretes oder hybrides Gesamtsystem bilden, durch zwei Klassen von Signalen zu beschreiben: *Bedingungs-* und *Ereignissignale*. Bedingungssignale übertragen Informationen über aktuelle Zustände in der Umgebung eines Systems. Aus ihrem Wert können Bedingungen für den Wechsel von Zuständen abgeleitet werden. Mit anderen Worten: Bedingungssignale ermöglichen oder verhindern Zustandsübergänge. Dagegen beschreiben Ereignissignale gerade stattfindende Übergänge und sind in der Lage, Zustandswechsel zu erzwingen. Die Unterscheidung zwischen dem Ermöglichen bzw. Verhindern und dem Erzwingen von Ereignissen entspricht der in der Technik üblichen Anschauung und findet sich auch in anderen Darstellungsformen (z. B. Grafcet [DaAl92]) oder Begriffen (vgl. RS- und D-Flipflops [Fas88]) wieder. Formal gesehen sind Bedingungssignale abschnittsweise konstante Funktionen, die die Zeit in eine endliche Menge von symbolischen Bedingungen abbilden. An den Sprungstellen gilt der rechtsseitige Grenzwert als Funktionswert. Ereignissignale sind Zeitfunktionen, die nur zu diskreten Zeitpunkten Werte aus einer endlichen Menge von Ereignissen annehmen und sonst den Wert 0 besitzen.

Bei der Modellierung mit B/E-Systemen können die zu beschreibenden Systeme in Blockdiagrammen dargestellt werden, in denen die einzelnen Blöcke Teile des Gesamtsystems darstellen und über die beiden genannten Signalarten kommunizieren. Im allgemeinen besitzt ein B/E-System ein Bedingungseingangssignal $u(t)$, ein Bedingungsausgangssignal $y(t)$, ein Ereigniseingangssignal $v(t)$ und ein Ereignisausgangssignal $z(t)$. Ihre Wertebereiche werden mit U, Y, V bzw. Z bezeichnet. Einzelne Signale können entfallen, wenn ihr Wertebereich nur aus einem einzigen Symbol besteht. In Abb. 1 ist das B/E-System-Blockdiagramm für das Bahnschrankenbeispiel zu sehen. Man erkennt, daß das Gesamtsystem in die Blöcke *Zug*, *Steuerung* und *Schranke* sowie drei Uhren für das Zeitverhalten dieser drei Teilsysteme aufgeteilt werden kann. Die Signale sind durch Pfeile dargestellt, wobei Ereignissignale durch ein blitzartiges Symbol und Bedingungssignale durch ein stufenartiges Symbol gekennzeichnet sind. Es ist zu erkennen, daß der Zug der Steuerung über ein Ereignissignal mit den Werten „*kommt*“ und „*entfernt sich*“ mitteilt, daß er sich dem Übergangsbereich nähert bzw. ihn verlassen hat. Die Steuerung kann ihrerseits im System *Schranke* über ein Ereignissignal das Öffnen bzw. Schließen erzwingen.

Stellvertretend für alle drei Uhren sind nur die Signale zwischen der Schranke und der dazugehörigen Uhr (t_S steht für den Stand dieser Uhr) angegeben. Diese werden weiter unten bei der Beschreibung der B/E-Uhren erläutert.

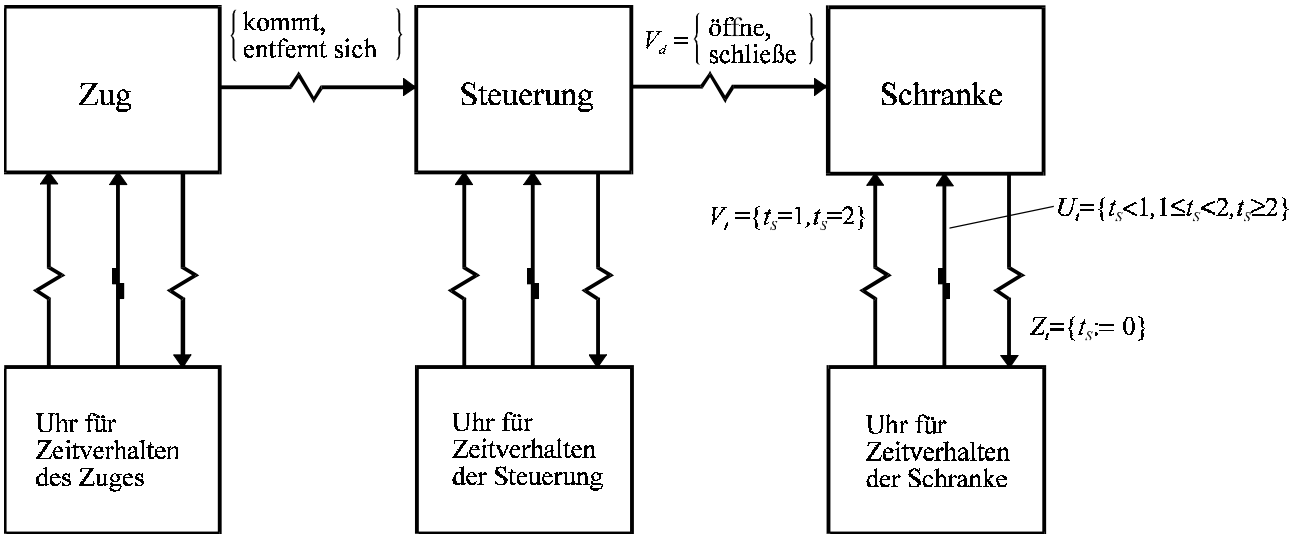

Abbildung 1: Blockschaltbild aus B/E-Systemen für das Bahnschranken-Beispiel.

Zur Beschreibung von Echtzeitsystemen werden also zwei Klassen von Blöcken benötigt: diskrete B/E-Systeme (z. B. *Zug*, *Steuerung* und *Schranke*) und B/E-Uhren (z. B. für das Zeitverhalten dieser Systeme). Die innere Dynamik diskrete B/E-Systeme wird mit Hilfe einer endlichen Menge X von diskreten Zustände dargestellt. Die möglichen Zustandstrajektorien $x(t)$ und Ausgangssignalverläufe werden abhängig vom Anfangszustand und den Eingangssignalverläufen durch die folgenden Funktionen bestimmt:

Zustandsübergangsfunktion: $x(t) \in f(x(t^-), u(t^-), v(t)), \qquad x(0) = x_0 \in X$,

Bedingungsausgabefunktion: $y(t) = g(x(t), u(t))$,

Ereignisausgabefunktion: $z(t) = h(x(t^-), x(t), v(t))$,

wobei $x(t^-)$ eine Abkürzung für $lim_{\Delta t \to 0}\, x(t-\Delta t)$ ist (für $u(t^-)$ gilt entsprechendes).

Man erkennt, daß der Folgezustand neben dem aktuellen Zustand von beiden Eingangssignalen abhängt. Die Bedingungsausgabe wird über den aktuellen Zustand und das Bedingungseingangssignal gebildet, und die Ereignisausgabe hängt davon ab,

ob sich der Zustand vom Zeitpunkt t^- zum Zeitpunkt t geändert hat und welchen Wert das Ereigniseingabesignal hat.

Für das System *Schranke* aus dem Beispiel sehen die Wertemengen und die *f*- und *h*-Funktion wie unten angegeben aus. Bei den Wertemengen für die Ereignissignale $v(t)$ und $z(t)$ ist zu beachten, daß die 0 konventionsgemäß nicht explizit aufgeführt wird. Bei der Angabe der Funktionstabellen werden zur Abkürzung zwei Konventionen getroffen: (1) Ereignisse und Bedingungen werden durch das „Don't-Care"-Symbol „-" ersetzt, wenn die Funktion für jeden beliebigen, sinnvollen Wert des Ereignisses bzw. der Bedingung den gleichen Funktionswert hat. (2) Für nicht explizit angegebene Signalkombinationen im Argument der *f*-Funktion kann das System stets im momentanen Zustand verharren.

Wertemengen:

- $X = \{offen, schließend, geschlossen, öffnend\}$
- $U = U_t = \{t_S<1, 1\leq t_S<2, t_S\geq 2\}$
- $V = V_t \times V_d = \{t_S=1, t_S=2\} \times \{öffne, schließe\}$
- $Z = Z_t = \{t_S:=0\}$

***f*-Funktion:**

{*offen*}	=	*f* (offen, -, (-, 0))
{*schließend*}	=	*f* (offen, -, (-, *schließe*))
{*schließend*, *geschlossen*}	=	*f* (schließend, $t_S<1$, (0, -))
{*geschlossen*}	=	*f* (geschlossen, -, (-, 0))
{*öffnend*}	=	*f* (geschlossen, -, (-, *öffne*))
{*öffnend*}	=	*f* (öffnend, $t_1<1$, (0, -))
{*öffnend*}	=	*f* (öffnend, $t_1<1$, ($t_S=1$, -))
{*öffnend*, *offen*}	=	*f* (öffnend, $1\leq t_S<2$, (0, -))

***h*-Funktion:**

$\{t_S:=0\}$	=	*h* (*offen*, *schließend*, (-, -))
$\{t_S:=0\}$	=	*h* (*geschlossen*, *öffnend*, (-, -))

B/E-Uhren (oder *B/E-Timer*) wurden in [EKK+95] eingeführt und in [HLU+97] dahingehend modifiziert, daß Uhren statt wie ursprünglich nur einem nun mehrere Schwellwerte haben können. Dies diente dazu, die Modellierungsmächtigkeit an die EZA anzupassen, die beliebige Vergleiche der Uhrenstände mit ganzzahligen

Zahlenwerten gestatten. Ein B/E-Timer θ ist demnach ein B/E-System dessen Verhalten durch einen Vektor $T_\theta = (T_\theta^1,...,T_\theta^n)$ gegeben ist. Die Elemente des Vektors sind ganzen Zahlen in aufsteigender Reihenfolge und werden *Schwellwerte* genannt. In Abhängigkeit von seinem Eingangsereignissignal s_v, aus dem die Zeitpunkte hervorgehen, zu denen der Timer zurückgesetzt wird, erzeugt der Timer ein Ausgangsbedingungssignal s_y und ein Ausgangsereignissignal s_z. Das Signal s_y gibt zu jedem Zeitpunkt darüber Auskunft, in welchem der möglichen Intervalle $[T_\theta^i, T_\theta^{i+1})$, $[0, T_\theta^1)$ oder $[T_\theta^n, \infty)$ der Wert des Timers sich augenblicklich befindet. Das Signal s_z gibt zu jedem Zeitpunkt, zu dem der Timer einen seiner Schwellwerte T_θ^i erreicht, ein Ereignis „$t_\theta = T_\theta^i$“ aus. Der Wert eines Timers wächst linear mit der globalen Zeit und beim Zurücksetzen wird der Wert auf 0 eingestellt. Formal werden Timer über eine *Timerfunktion* definiert. Eine Timerfunktion $\tau_\theta(s_v)$: [0, ∞) → [0, ∞) ist gegeben durch $\tau_\theta(s_v)(t) = t - t_i$, wobei t_i der jeweils letzte Zeitpunkt ist, zu dem der Timer zurückgesetzt wurde. Zum Anfangszeitpunkt t_0 startet die Timerfunktion mit einem beliebigen, aber festen Abstand Δ oberhalb des größten Schwellwertes, damit keine Schwellwertereignisse ausgegeben werden, bevor der Timer das erste Mal zurückgesetzt worden ist: $\tau_\theta(s_v)(t_0) = T_\theta^n + \Delta$.

Anstelle einer formalen Definition eines B/E-Timers wird hier das Verhalten anhand eines beispielhaften Zeitverlaufes der Timerfunktion und der entsprechenden Ein- und Ausgabesignale des Timers für das Zeitverhalten der Schranke in Abb. 2 illustriert. Dieser Timer ist durch den Schwellwertvektor T_S=(1, 2) definiert, da sich die Schranke in mindestens einer Zeiteinheit schließen kann und zwischen einer bis zwei Zeiteinheiten wieder öffnen kann.

4 Echtzeitautomaten

Echtzeitautomaten wurden von Alur und Dill [AlDi94] eingeführt und stellen eine Erweiterung der endlichen Automaten über unendlichen Zeichenfolgen dar. Hinzugefügt wurden reelle Variablen, die Uhren darstellen und gleichmäßig ansteigen. Die Uhren können durch Übergänge des Automaten zurückgesetzt und gestartet werden,und die Übergänge können von den Uhrenständen abhängig sein.

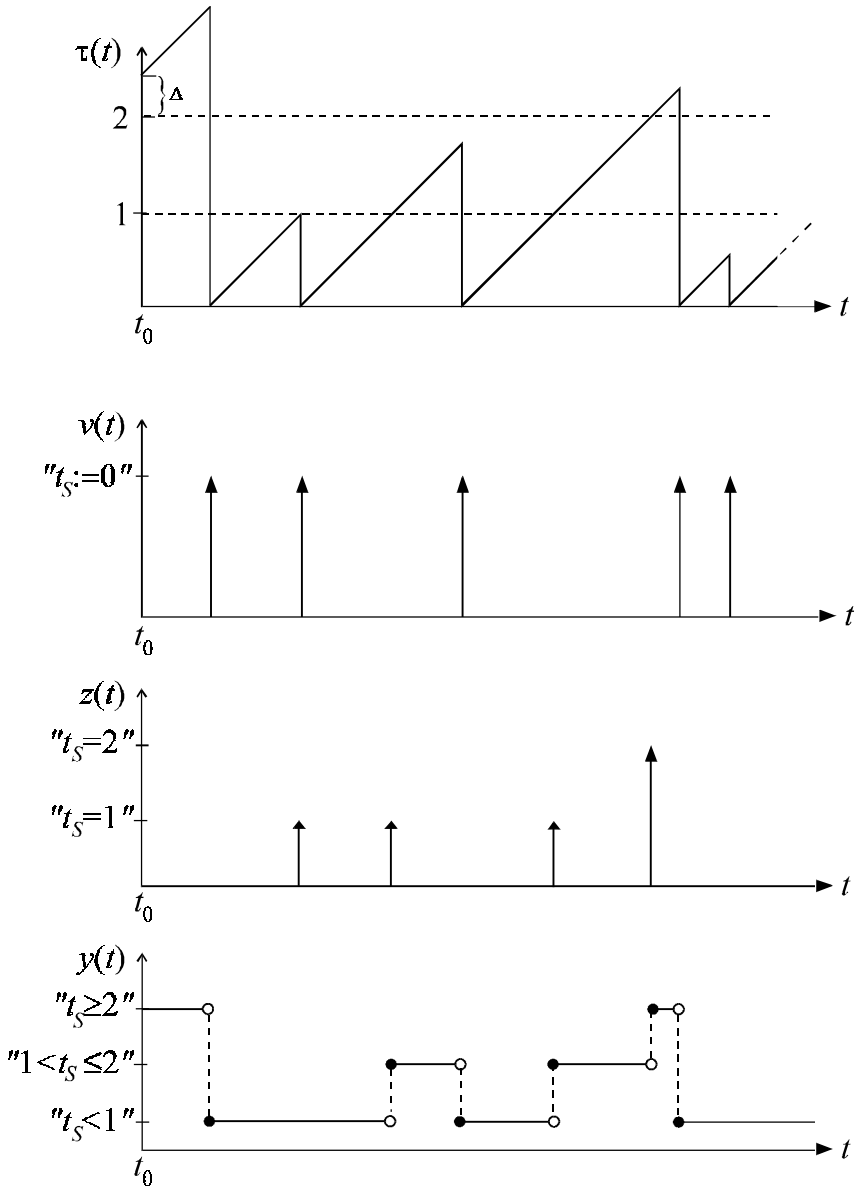

Abbildung 2: Beispielhafter Zeitverlauf der B/E-Timer-Signale für die Schranke.

Abbildung 3 zeigt einen EZA für das System *Schranke* aus dem Beispiel. Dieser hat eine Uhr, die von der Variablen *y* repräsentiert wird. Bei den Übergängen von „*offen*“ nach „*schließend*“ und von „*geschlossen*“ nach „*öffnend*“ wird die Uhr zurückgesetzt, was durch den Ausdruck „*y*:=0“ symbolisiert wird. Die Übergänge von „*schließend*“ nach „*geschlossen*“ und von „*öffnend*“ nach „*offen*“ können nur genommen werden, wenn $y<1$ bzw, $1<y<2$ gilt. Auf diese Weise wird die Dauer des Schließens bzw. des Öffnens modelliert.

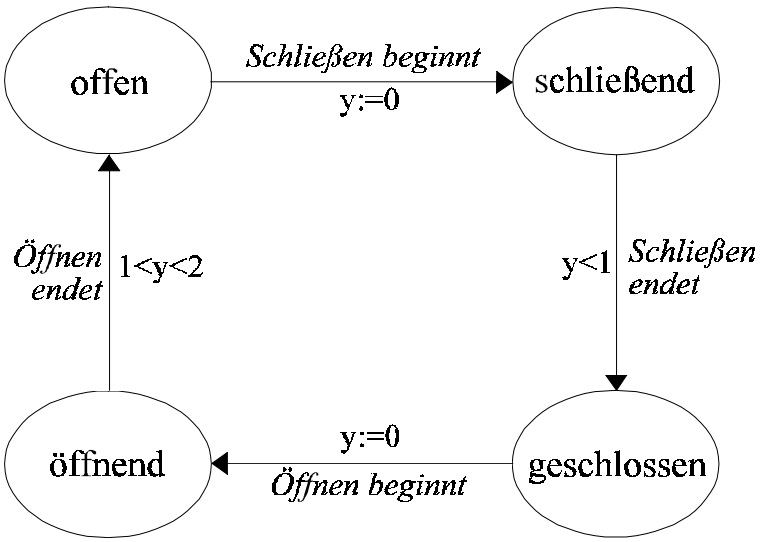

Abbildung 3: Echtzeitautomat für die Schranke

5 Transformation der beiden Modelle

Das Beispiel der Schranke in den beiden vorigen Abschnitten macht deutlich, daß die Unterschiede zwischen den beiden Modellformen nicht so groß sind: Sowohl im ZBES als auch im EZA können die gleichen diskreten Zustände genommen werden, und in beiden Fällen gibt es genau eine Uhr. Das Hauptproblem bei der Transformation bestand darin, die bei den B/E-Systemen gegebene Möglichkeit des Erzwingens durch ein Zeitereignis in den EZA nachzubilden. Um dies auf eine möglichst einfache theoretische Grundlage zu stellen, wurde eine besondere Klasse von EZA betrachtet, nämlich die sogenannten Safety Timed Automata [HNS+94]. Diese zeichnen sich dadurch aus, daß jeder diskrete Zustand im Hinblick auf die Akzeptanz von Zeitwörtern (hier: unendliche Folgen von Paaren aus einem Symbol und einem Zeitpunkt) ein akzeptierender Zustand ist. Hinsichtlich der Semantik wurde vereinbart, daß das Verbleiben in einem diskreten Zustand explizit durch eine *Schleife* (engl. *self-loop*) angegeben sein muß. Das Erzwingen eines Übergangs kann dann dadurch zum Ausdruck gebracht werden, daß eine Zeitbedingung nur einem Übergangspfeil zu einem anderen Zustand zugeordnet ist, nicht aber der Schleife des aktuellen Zustandes. Die Übersetzung eines ZBES in einen EZA geschieht nun im Prinzip so, daß jeder Wertekombination $(x, u, v) \in X \times U \times V$ alle aus der Zustandsübergangsfunktion des ZBES hervorgehenden Übergänge einschließlich der Schleifen zugeordnet werden. In Abb. 4 ist zur Veranschaulichung der auf diese Weise aus dem ZBES *Schranke*

hervorgehende EZA dargestellt. Jeder Übergang ist mit einem Tripel markiert, das aus dem Uhrenprädikat (ergibt sich aus den Signalwerten von der B/E-Uhr zum diskretem B/E-System), der Menge der zurückzusetzenden Uhren (ergibt sich aus der Ereignisausgabe vom diskreten B/E-System zur B/E-Uhr) und dem Wert v_d (ergibt sich aus dem Ereigniseingabesignal des diskreten B/E-Systems von seiner Umgebung) besteht. Das Symbol „-“ hat die analoge Bedeutung wie bei den ZBES.

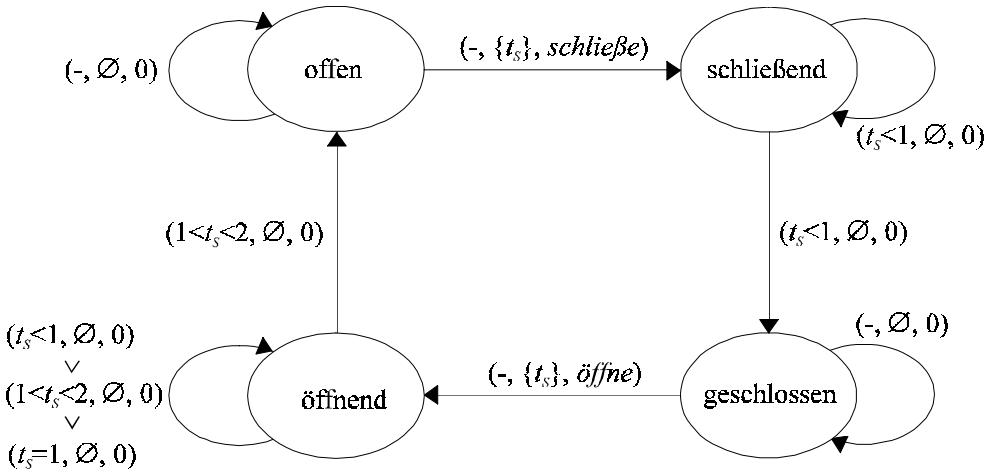

Abbildung 4: Übersetzung des ZBES für die Schranke in einen Echtzeitautomaten.

6 Analyseergebnisse

Die Analyse des ZBES-Modells des Bahnschrankenbeispiels mit KRONOS erforderte zwei vorbereitende Schritte, die zusätzlich zur Anwendung der Transformationsvorschrift ausgeführt werden mußten. Da die Übersetzung bisher nur für einzelne ZBES untersucht wurde und nicht das Zusammenwirken umfaßt, mußten die sechs ZBES aus Abb. 1 zunächst mit Hilfe der bekannten Kompositionsregeln für B/E-Systeme zusammengefaßt werden. Nach der Übersetzung dieses Systems in einen EZA mußte die oben eingeführte Beschreibung mit expliziten Schleifen in eine Darstellung mit Invarianten übertragen werden, da KRONOS nur EZA mit dieser Semantik verarbeiten kann. Invarianten sind logische Ausdrücke über die Uhrenvariablen, die diskreten Zuständen zugeordnet sind und erfüllt sein müssen, solange sich das System in dem entsprechenden Zustand befindet. Sobald die Invariante ungültig wird, muß der Zustand verlassen werden. Auch auf diese Weise kann das Erzwingen der B/E-System nachgebildet werden. Dieser letzte Schritt vor der Analyse mußte noch per Hand

durchgeführt werden. Eine direkte Übersetzung von ZBES in EZA mit Invarianten ist aber in Bearbeitung.

In der Abb. 5 ist die aus den beschriebenen Schritten resultierende Eingabedatei für KRONOS zu sehen. (Ein alternatives Modell kann dem KRONOS-Handbuch [www2] entnommen werden, in dem das gleiche Beispiel behandelt wird). Von den bei der Komposition theoretisch entstehenden 64 Zuständen sind alle schon im rein diskreten, nicht-zeitbewerteten Fall nicht einnehmbaren Zustände weggelassen. Auf der rechten Seite sind die einzelnen Zustände näher erläutert. Der Bezeichnung liegt dabei eine Unterteilung der Zugstrecke in drei Abschnitte zugrunde: Abschnitt 1 ist der direkte Übergangsbereich, Abschnitt 2 ist der zu beiden Seiten an den Übergang angrenzende Bereich, der durch die beiden Sensoren für das Annähern und Entfernen begrenzt wird. Abschnitt 3 bedeutet, der Zug ist außerhalb von Abschnitt 2.

Kopf des Eingabefiles: *Anzahl der Zustände, Transitionen und Uhren.*

```
#states 12
#trans 20
#clocks 3 X Y Z
```

Zustand 0*: Zug noch vor dem Einfahr-Sensor (Abschnitt 3), Schranke offen.*

```
state: 0
invar: true
trans: true => IN2_APPROACH_0; reset { X, Z }; goto 1
```

Zustand 1: *Ankommender Zug wurde signalisiert (Abschnitt 2), Schranke offen.*

```
state: 1
invar: X<5 and Z<=1
trans: X<5 and Z=1 => IN2_0_LOWER; reset { Y }; goto 2
```

Zustand 2: *Ankommender Zug wurde signalisiert (Abschnitt 2), Schranke schließt sich.*

```
state: 2
invar: X<5 and Y<1 and Z>1
trans:
X<5 and Z>1 and Y<1 => IN2_0_0; reset { }; goto 3
2<X and X<5 and Z>1 and Y<1 => IN1_IN_0; reset { }; goto 4
2<X and X<5 and Z>1 and Y<1 => IN1_IN_0; reset { }; goto 5
```

Zustand 3: *Ankommender Zug wurde signalisiert (Abschnitt 2), Schranke geschlossen.*

```
state: 3
invar: X<5 and Z>1
trans: 2<X and X<5 and Z>1 => IN1_IN_0; reset { }; goto 5
```

Abbildung 5: KRONOS-Eingabefile für das Bahnschranken-Beispiel (*wird fortgesetzt*).

Zustand 4: *Zug im Übergangsbereich (Abschnitt 1), Schranke schließt sich.*

```
state: 4
invar: 2<X and X<5 and Z>1 and Y<1
trans:
2<X and X<5 and Z>1 and Y<1 => IN1_0_0; reset { }; goto 5
2<X and X<5 and Z>1 and Y<1 => IN2_OUT_0; reset { }; goto 6
2<X and X<5 and Z>1 and Y<1 => IN2_OUT_0; reset { }; goto 7
```

Zustand 5: *Zug im Übergangsbereich (Abschnitt 1), Schranke geschlossen.*

```
state: 5
invar: 2<X and X<5 and Z>1
trans:
2<X and X<5 and Z>1 => IN2_OUT_0; reset { }; goto 7
```

Zustand 6: *Zug verläßt Übergangsbereich (Abschnitt 2), Schranke schließt sich.*

```
state: 6
invar: 2<X and X<5 and Z>1 and Y<1
trans:
2<X and X<5 and Z>1 and Y<1 => IN2_0_0; reset { }; goto 7
2<X and X<5 and Z>1 and Y<1 => IN3_EXIT_0; reset { Z }; goto 8
2<X and X<5 and Z>1 and Y<1 => IN3_EXIT_0; reset { Z }; goto 9
```

Zustand 7: *Zug verläßt Übergangsbereich (Abschnitt 2), Schranke geschlossen.*

```
state: 7
invar: 2<X and X<5 and Z>1
trans: 2<X and X<5 and Z>1 => IN3_EXIT_0; reset { Z }; goto 9
```

Zustand 8: *Entfernen des Zuges signalisiert (Abschnitt 3), Schranke schließt sich.*

```
state: 8
invar: Z<1 and Y<1
trans: Z<1 and Y<1 => IN3_0_0; reset { }; goto 9
```

Zustand 9: *Entfernen des Zuges signalisiert (Abschnitt 3), Schranke geschlossen.*

```
state: 9
invar: Z<1
trans: Z<1 => IN3_0_RAISE; reset { Y }; goto 10
```

Zustand 10: *Entfernen des Zuges signalisiert (Abschnitt 3), Schranke öffnet sich.*

```
state: 10
invar: Y<2
trans:
Y<2 => IN2_APPROACH_0; reset { X, Z }; goto 11
Y>1 and Y<2 => IN3_0_0; reset { }; goto 0
Z<1 => IN2_APPROACH_0; reset { X, Z }; goto 1
```

Zustand 11: *Entfernen des Zuges signalisiert (Abschnitt 3), Schranke offen.*

```
state: 11
invar: X<5 and Z<1 and Y<2
trans: Z<1 => IN2_0_0; reset { }; goto 1
```

Abbildung 5: KRONOS-Eingabefile für das Bahnschranken-Beispiel (*Fortsetzung*).

Die zu verifizierende Eigenschaft wird in TCTL (Timed Computation-Tree Temporal Logic [HNS+94]) formuliert und besteht aus zwei Teilen, einer Sicherheits- und einer Lebendigkeitsanforderung. Mit der folgenden Sicherheitseigenschaft wird ausgedrückt, daß die Zustände, in denen der Zug sich im Übergangsbereich (also Abschnitt 1) befindet und die Schranke noch nicht vollständig geschlossen ist, nämlich 4, 6 und 8, nicht erreichbar sein dürfen (*init* steht für den Anfangszustand, *impl* bedeutet $\Rightarrow$, *a* heißt $\forall$, und *b* steht für „Box“, also den temporallogischen Operator „*Immer*“):

```
init impl ab not ( state(4) or state(6) or state(8) )
```

Die Lebendigkeitseigenschaft dient dazu, auszuschließen, daß die eine Steuerung als korrekt erkannt wird, die die Schranken nie öffnet. Sie drückt aus, daß immer dann, wenn die Schranke geschlossen ist, sie spätestens nach 7 Zeiteinheiten wieder geöffnet wird, so daß der Straßenverkehr nicht unnötig behindert wird. Dies wird folgendermaßen ausgedrückt (hier steht *d* für „Diamond“, den temporallogischen Operator „*Schließlich*“)).

```
init impl  ( ab ( state(3) or state(5) or state(7) or state(9) )
               impl ad{<=7} ( state(0) or state(1) ))
```

Startet man KRONOS mit dem Eingabefile aus Abb. 5 und der obigen Spezifikation als Eingabe, so erzeugt es eine interne Darstellung des Automaten und führt für diese ein Model-Checking durch, das heißt, es bestimmt anhand einer Erreichbarkeitsanalyse, ob die Anforderungen erfüllt werden oder nicht. Das Ergebnis ist in unserem Fall, daß die fraglichen Zustände nicht erreicht werden können und die Lebendigkeitsanforderung erfüllt ist, die Steuerung also korrekt ist.

7 Zusammenfassung und Diskussion

Es wurde ein Verfahren vorgestellt, mit dem modular aufgebaute ZBES-Modelle technischer Systeme hinsichtlich der Erfülltheit temporallogischer Ausdrücke überprüft werden können. Dazu wird das ZBES in einen äquivalenten EZA übersetzt, der dann mit verfügbaren leistungsfähigen Tools analysiert werden kann. Auf diese Weise wird die Möglichkeit geschaffen, sowohl den Vorteil des modularen Modellierungsansatzes

der B/E-Systeme anzuwenden, als auch die von den Temporallogiken bereitgestellte mächtige Spezifikationssprache auszunutzen.

Unsere zukünftigen Arbeiten werden sich zunächst damit beschäftigen, die Übersetzung kompositional durchzuführen, das heißt, die ZBES-Modelle Modul für Modul zu übersetzen und gleichzeitig das Zusammenwirken in eine Synchronisationsvorschrift für die entstehenden EZA zu transformieren. Auf diese Weise wird die Komposition der ZBES-Teilsysteme als Zwischenschritt unnötig. Damit wird auch die Grundlage geschaffen, reale technische Systeme, die natürlich wesentlich komplexer sind als das hier behandelte Bahnschrankenbeispiel, zu untersuchen. Die Grenze der Anwendbarkeit hinsichtlich der analysierbaren Systemgröße ist noch zu ermitteln. Sie wird sowohl von der Leistungsfähigkeit der verfügbaren Tools abhängen als auch vom Modellierungsaufwand, der aufgrund des modularen Ansatzes aber durch Bereitstellung von Bibliotheken mit wiederverwertbaren Modellen in Grenzen gehalten werden kann. Weitere Arbeiten werden sich mit der direkten kompositionalen Analyse von ZBES und EZA befassen.

Literatur

[AlDi94] R. Alur und D. L. Dill: A Theory of Timed Automata. *Theoretical Comp. Science* **126**, 1994, pp. 183-235.

[DaAl92] R. David und H. Alla: *Petri-Nets and Grafcet.* Prentice-Hall, New York, 1992.

[EKK+95] S. Engell, S. Kowalewski, B. Krogh und J. Preußig: Condition/Event Systems: A Powerful Paradigm for Timed and Untimed Discrete Models of Technical Systems. In Breitenecker u. Husinsky (Eds.): *EUROSIM'95*, Wien Österreich 1995, Elsevier, pp. 421-426.

[EnHo96] S. Engell und I. Hoffmann: Modular Hierarchical Models of Hybrid Systems. *35th IEEE Conf. on Decision and Control*, Kobe, Japan, 1996.

[Fas88] K. H. Fasol: *Binäre Steuerungstechnik*. Springer-Verlag, Berlin, 1988.

[GRE94] H. E. Garcia, A. Ray und R. M. Edwards: A Reconfigurable Hybrid Supervisory System for Process Control. Proc. *33rd IEEE Conf. on Decision and Control*, USA, 1994, pp. 3131-3136.

[HaRa95] H.-M. Hanisch und M. Rausch: Synthesis of Supervisory Controllers Based on a Novel Representations of Condition/Event Systems. Proc.

IEEE Int. Conf. on Systems, Man, and Cybernetics, Vancouver, Kanada, 1995.

[HLU+97] R. Huuck, Y. Lakhnech, L. Urbina, S. Engell, S. Kowalewski, und J. Preußig: Comparing Timed C/E Systems with Timed Automata. Angenommen für den *International Workshop on Hybrid and Real-Time Systems* (*HART'97*), Grenoble, Frankreich, 26.-28.3.1997.

[HNS+94] T. A. Henziger, X. Nicollin, J. Sifakis und S. Yovine: Symbolic Model Checking for Real-Time Systems. *Information and Computation* **111** (2), 1994, pp. 193-244.

[Hol96] L. Holloway: Time Measures and State Maintainability for a Class of Composed Systems. Proc. *Int. Workshop on Discrete Event Systems* (*WODES'96*), Edinburgh, UK, 1996, pp. 24-30.

[KE97] S. Kowalewski und S. Engell: *Bedingung/Ereignis-Systeme und ihre Anwendung zur Verifikation von Steuerungen*. Tutorium des VDI-GMA-Fachausschusses 1.8 auf der EKA'97, in diesem Band.

[LeSt87] N. G. Leveson und J. L. Stolzy: Safety Analysis Using Petri Nets. *IEEE Trans. Software Eng.* **SE-13** 3, 1987, pp. 386-397.

[SrKr91] R. S. Sreenivas und B. H. Krogh: On Condition/Event Systems with Discrete State Realizations. *Discrete Event Dynamic Systems: Theory and Applications* **1** (2), 1991, pp. 209-236.

[Thu95] E. M. Thurner: Formale Beschreibung und Verifikation von Hardwareprotokollen. Tagungsband *4. Fachtagung Entwurf komplexer Automatisierungssysteme* (*EKA 95*), Braunschweig, 1995, pp. 197-215.

[www1] `http://www-cad.EECS.Berkeley.EDU/~tah/HyTech/`

[www2] `http://www.imag.fr/VERIMAG/TEMPORISE/kronos/`

[www3] `http://www.docs.uu.se/docs/rtmv/uppaal/`