Zeitschrift: Bulletin des Schweizerischen Elektrotechnischen Vereins, des Verbandes Schweizerischer Elektrizitätsunternehmen = Bulletin de l'Association suisse des électriciens, de l'Association des entreprises électriques suisses

Herausgeber: Schweizerischer Elektrotechnischer Verein ; Verband Schweizerischer Elektrizitätsunternehmen

Band: 92 (2001)

Heft: 19

Artikel: Korrekte Software für diskrete dynamische Systeme

Autor: Schroeder, Katrin / Fischer, Hans-Dieter

DOI: https://doi.org/10.5169/seals-855756

Korrekte Software für diskrete dynamische Systeme

Korrekte Software[1] ist überall dort von grosser Bedeutung, wo der Schutz grosser Investitionen sowie insbesondere die Sicherheit für Mensch und Umwelt überragenden Stellenwert besitzen. Hierzu zählen Gebiete wie Medizin, Luftfahrt, Eisenbahn, Chemieanlagen oder Kernkraftwerke. Korrekte Software wird eher von Menschen erstellt, die zu Ordnung, Disziplin, Verlässlichkeit und systematischem Arbeiten neigen. Um korrekte Software wirtschaftlich produzieren zu können, finden rechnerunterstützte Werkzeuge stetig zunehmende Verbreitung.

Katrin Schroeder, Hans-Dieter Fischer

Durch die Verwendung solcher rechnerunterstützten Werkzeuge werden insbesondere ermüdende Routinetätigkeiten dem Computer überlassen, der schneller und zuverlässiger auf Vollständigkeit oder Konsistenz prüfen kann als der Mensch. Zudem sind die Resultate reproduzierbar. Wo immer sinnvoll empfiehlt sich daher die Benutzung einschlägiger Werkzeuge während der Softwareentwicklung.

Zu dem als Phasenmodell[2] bekannten Verfahrensmodell der Softwaretechnik werden im vorliegenden Beitrag einige Anmerkungen gemacht, die sich aus der praktischen Anwendung bei der Erzeugung fehlerarmer und fehlertoleranter Software im Laufe der letzten Jahre ergeben haben[3]. Für die Validation der eigentlichen Aufgabenstellung – der so genannten Anforderungsspezifikation – einerseits sowie des fertig integrierten Systems andererseits werden, wie weiter unten erläutert, zueinander «orthogonale» Strategien empfohlen. Solche Strategien können durch (Schutz-)Zielorientierung bzw. durch Anwendungsorientierung umschrieben werden[4]; durch ihre verschiedenartigen Denkansätze unterstützen sie die Unabhängigkeit beider Prüfstrategien. Korrektheitsbeweise[5] werden zukünftig automatisiert mit Werkzeugunterstützung durchgeführt.

Am einfachen Beispiel einer linearen, gewöhnlichen Differenzialgleichung wird das Entwickeln korrekter Software mittels der Wellendigitalmethode[6] und die Verbesserung der Verständlichkeit bei der Verwendung einer graphischen Spezifikation illustriert. Die vorgestellte Wellendigitalmethode kann im Übrigen auch auf nichtlineare partielle Differenzialgleichungen angewendet werden, und es besteht die Aussicht, den vorliegenden Korrektheitsbeweis auf die dazugehörende Software ausdehnen zu können.

Vermehrt Wertschöpfung in der Software

Die vergangenen 30 Jahre sind wohl durch nichts nachhaltiger beeinflusst worden als durch das Erscheinen des Mikroprozessors und sein Vordringen in viele Bereiche des täglichen Lebens. Stellvertretend sei hier nur an die Revolution in der Bürowelt während der achtziger Jahre des vergangenen Jahrhunderts erinnert, an das Aufkommen des rechnerunterstützten Entwerfens und an Steuerungen industrieller Prozesse durch Mikroprozessoren. Internet und Mobilkommunikation sind ohne solche Digitaltechnik erst gar nicht denkbar.

Mit der Digitaltechnik verlagerte sich der Wertschöpfungsprozess von der Hard- zur Software. Es wird immer umfangreichere Software produziert, ohne im gleichen Masse den dazu erforderlichen Erstellungsprozess wirkungsvoll auf Fehlerarmut oder Fehlertoleranz kontrollieren zu können.

Das Verfahrensmodell

In den vergangenen zwei Jahrzehnten sind verschiedene Techniken zur Produktion korrekter Software entwickelt und teilweise erfolgreich angewendet worden. Hierzu zählt insbesondere ein Rahmenplan, nach dem der Entwicklungsprozess begleitet von Prüfungen abläuft.

Das Phasenmodell

Häufig wird dieser Plan auch Phasenmodell oder bei expliziter Darstellung von Zwischenprüfungen auch V-Modell genannt. Alle derartigen Pläne systematisieren die Produktion von Software, indem sie den Lebenszyklus[7] von Software in Zeitabschnitte zerteilen, in denen eine genau bestimmte Teilaufgabe gelöst wird. Die Aussage, eine Software sei korrekt, kann offensichtlich nur dann getroffen werden, wenn eine möglichst lückenlose Aufgabenbeschreibung vorliegt und eine Prüfung ergeben hat, dass die Software alle geforderten Aufgaben erfüllt. Soll die zu entwickelnde Software auch unter geänderten Rahmenbedingungen korrekt funktionieren, dann ist dies in der Anforderungsspezifikation explizit niederzulegen. Entsprechende Vorkehrungen werden dann im Entwurf der Software bereits eingeplant.

In diesem Sinne war der Code, der von den USA nach England portiert wurde (Kasten), korrekt. Die Schwierigkeit liegt offensichtlich darin, dass die Software nicht nur für die Lösung der beschriebenen Aufgaben eingesetzt wurde, sondern auch für andere, die nicht spezifiziert wurden und die für den ursprünglich beabsichtigten Einsatz auch völlig irrelevant sind.

Fehlerhafte Software: ein Beispiel

Die Übertragung einer Software von den USA nach England führte dort zum Systemausfall, weil der Code den nullten Längenkreis nicht korrekt verarbeiten konnte. Die ursprüngliche Anforderungsspezifikation für den Code enthält natürlich nicht den nullten Längenkreis, da er für die USA nicht relevant ist.

Spezifikationssprachen

Um solche Nebeneffekte zu erkennen und ggf. auszuschalten, erzeugt man eine vollständige logische Beschreibung der

Anforderungsspezifikation. Dies wird durch so genannte Spezifikationssprachen erreicht (Kasten). Die Beschreibung von Anforderungen mittels solcher Sprachen begünstigt den Einsatz rechnerunterstützter Werkzeuge beträchtlich, wobei allerdings die Verständlichkeit leidet, insbesondere wenn der Auftraggeber nicht gleichzeitig auch der Realisierer ist.

Beispiel einer Spezifikationssprache

Die nachfolgend dargestellten Programmzeilen zeigen am Beispiel der Sprache XSPEC (zur Verfügung gestellt vom Institut für Sicherheitstechnologie Istec in Garching/D), wie man sich die Syntax von Spezifikationssprachen vorstellen muss.

```
DO
        EXTERNAL_SYSTEM: ext_1
       MESSAGE: msg_from_ext_1
    OR
        EVENT: event_2
    END;
```

Graphische Spezifikationssprachen

Die Verwendung graphischer Spezifikationssprachen kann hier Abhilfe schaffen. Diese Techniken wurden – zumindest in Teilbereichen – bereits erfolgreich erprobt. Sie bedienen sich dabei einer genau definierten abgeschlossenen Menge graphischer Sprachelemente (bzw. Wörter), um eine Programmieraufgabe zu spezifizieren.

Die Idee ist nicht neu: Die Elektrotechnik beispielsweise beschreibt die Wirkungsweise von Schaltungen durch ideale Schaltelemente (z.B. Widerstand, Spule oder Kondensator) mit genau definierten Eigenschaften bezüglich ihres Strom-/Spannungverhaltens und in der digitalen Signalverarbeitung werden abstrakte Algorithmen durch das Vernetzen von idealen Elementen wie etwa Addierer, Multiplizierer oder Speicher visualisiert und somit verständlich dargestellt. Die Leittechnik schliesslich definiert ihre Mess-, Steuerungs-, Regelungs- und Informationsaufgaben in Form von rückwirkungsfreien graphischen Elementen (Grenzwertgeber, Maximalwert-Auswahl usw.).

Der Einsatz graphischer Sprachelemente zur Niederlegung einer Anforderungsspezifikation für eine Software verspricht derzeit das Verständnisproblem zwischen Auftraggeber und Software-Hersteller grundsätzlich lösen zu können und führt auf eine strikt modulare bausteinorientierte Softwarestruktur, die eine automatische Codeerzeugung ermöglicht.

In der Anforderungsspezifikation wird erklärt, welches System «gebaut» werden soll (Absicht). In den in Bearbeitungsabschnitte unterteilten Software-Entwicklungsprozess reihen sich Arbeitsergebnisse seriell aneinander. Absichten werden in der Regel validiert, während Arbeitsergebnisse gegenüber ihren Anforderungen verifiziert werden.

Das Ergebnis eines Abschnitts (Phase) wird in einem Dokument niedergelegt, das die Anforderungen für die darauffolgende Phase zusammenfasst. Dieses Ergebnis wird dann gegenüber den dokumentierten Anforderungen an die Phase verifiziert, um festzustellen, ob das System «richtig gebaut» wird.

Die Phasen brauchen nicht unbedingt zeitlich gegeneinander abgegrenzt zu sein und können sich auch zeitlich überlappen. Hiermit wird ein Vorausschauen ermöglicht – z.B. in Form eines «Rapid Prototyping» –, um zu ergründen, welche Anforderungen zweckmässig an die nächste Phase gestellt werden. Jede Phase wird inhaltlich durch ihr Enddokument abgeschlossen. Der letzte Arbeitsschritt in der Softwareerstellung wird zusätzlich gegenüber der anfänglichen Aufgabenbeschreibung validiert, um abschliessend sicher zu sein, dass auch das richtige System entworfen wurde.

«Orthogonales» Vorgehen bringt Vorteile

Beide Validationen – die der anfänglichen Anforderungsspezifikation und die abschliessende für das erstellte Softwaresystem – sind für korrekte Software von entscheidender Bedeutung. Dazu wird hier ein «orthogonales» Vorgehen empfohlen, das sich dadurch auszeichnet, dass beide Validationsprozesse durch unterschiedliche Denkmuster geprägt sind, um möglichst hohe Unabhängigkeit zu wahren: So kann die Vollständigkeit einer Aufgabenbeschreibung eher durch das Vorgeben von zu erreichenden Zielen (Schutzziele) sichergestellt werden, während die Frage, ob das richtige System gebaut wurde, zweckmässig durch angenommene Anwendungen bzw. Ereignisse beantwortet wird, auf die das zu untersuchende System wirkt, um die Ziele zu erreichen[8].

Während bei der Validation der Anforderungsspezifikation anfänglich Zielorientierung im Vordergrund steht, dominiert schliesslich Anwendungs- bzw. Ereignisorientierung. Man kann beide Paradigmen auch gegenseitig austauschen, wichtig bleibt allerdings, die Unabhängigkeit beider Validationen zu wahren. Zusätzlich können Simulationen von Ereignissen vorgenommen werden, die extrem selten sind und bei Ausfall des konstruierten Systems grossen Schaden anrichten.

Formale Korrektheitsbeweise

Bei extremen Sicherheitsanforderungen wird man in Teilbereichen formale Korrektheitsbeweise verwenden. Diese bedienen sich der Prädikatenlogik[9] und sind mathematisch rigoros. Hierbei bildet man eine logische Formel F aus einer Vorbedingung V, einem Softwaresegment S und einer Nachbedingung P. Diese Formel $F = \{V\}\ S\ \{P\}$ ist «True», falls aus der Einhaltung der Vorbedingung V vor der Ausführung des Segmentes S die Nachbedingung P nach der Ausführung von S als gültig folgt, andernfalls ist die logische Formel «False»[10] [8]. Dem Vorteil des Ausweises eines mathematischen Korrektheitsbeweises steht der Aufwand seiner Erstellung mit den damit verbundenen hohen Kosten gegenüber.

Erfolgreich kann der formale Korrektheitsbeweis nur unter bestimmten Voraussetzungen angewendet werden: Die Software muss klein bzw. strikt modular oder bereits nach festen Regeln aufgebaut sein. In diesen Fällen kann man den Korrektheitsbeweis manuell führen; zukünftig wird man jedoch Werkzeuge zum automatischen Korrektheitsbeweis einsetzen.

Wellendigitalfilter zur numerischen Integration

Für eindimensionale Algorithmen[11], die mit Hilfe von Wellendigitalstrukturen [9–12] entworfen worden sind, ist ein formaler Korrektheitsbeweis der resultierenden Software gelungen [8]. Im Folgenden wird die Methode der numerischen Integration von Differenzialgleichungen mit Wellendigitalstrukturen anhand eines einfachen Beispiels (Differenzialgleichung erster Ordnung) illustriert. Um nicht auf weiterführende Literatur verweisen zu müssen, wird dieses Beispiel sehr ausführlich dargestellt.

Erstellen einer Referenzschaltung

Zur Lösung einer Differenzialgleichung mit Hilfe von Wellendigitalstrukturen muss zunächst eine so genannte Referenzschaltung entworfen werden: So, wie man in der Schaltungsanalyse versucht, eine gegebene elektrische Schaltung äquivalent durch ein System von Differenzialgleichungen zu beschreiben, wird an dieser Stelle in umgekehrter Vorgehensweise zu einer gegebenen Differenzialgleichung eine entsprechende

Schaltungsdarstellung gesucht. Diese graphische Darstellung der Differenzialgleichung wird als Referenzschaltung bezeichnet. Für eine Differenzialgleichung erster Ordnung

$$\alpha \cdot \frac{\mathrm{d}\,y(t)}{\mathrm{d}\,t} + \beta \cdot y(t) = f(t) \qquad (1)$$

erhält man z.B. eine Referenzschaltung, wenn man die Funktion $y(t)$ als Strom und die Konstante β als Widerstand interpretiert. Unter diesen Voraussetzungen hat die Anregungsfunktion $f(t)$ die Dimension einer Spannung und kann somit in der Referenzschaltung als ideale Spannungsquelle dargestellt werden. Entsprechend wird die Konstante α in der Referenzschaltung als Induktivität interpretiert. In Bild 1 ist die resultierende Referenzschaltung abgebildet.

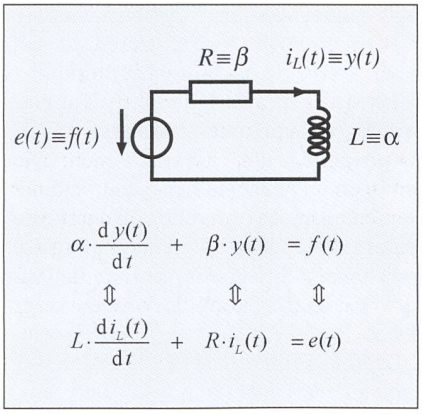

Bild 1 Referenzschaltung der Formel 1

Erstellen der Wellendigitalstruktur

Nachdem die Referenzschaltung entworfen worden ist, muss diese Schaltung in die zugehörige Wellendigitalstruktur (WD-Struktur) überführt werden. Wie schon der Name Wellendigitalstruktur andeutet, beinhaltet diese Umsetzung zwei Schritte:

- die unabhängigen Variablen der Referenzschaltung werden diskretisiert bzw. digitalisiert. Hierbei werden die ursprünglichen Differenzialgleichungen in Differenzengleichungen umgewandelt, welche sich graphisch mit Signalflussdiagrammen darstellen lassen.
- die ermittelten Differenzengleichungen werden durch Wellengrössen ausgedrückt, die – verglichen mit den entsprechenden Strom-Spannungsbeziehungen – häufig eine wesentlich einfachere Realisierung ermöglichen.

Im Folgenden werden auf diese Weise sukzessive alle Elemente der Referenzschaltung in ihre WD-Äquivalente überführt und anschliessend geeignet zusammengesetzt.

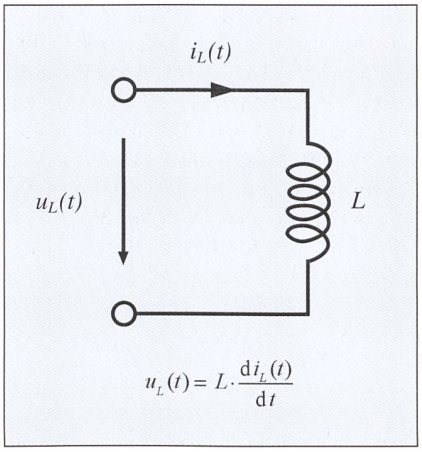

Bild 2 Die ideale Spule

Die Differenzialgleichung und das elektrische Schaltungssymbol sind äquivalente Darstellungsformen der Strom-Spannungsbeziehungen an der idealen Spule.

WD-Äquivalent der idealen Spule

Um eine Differenzengleichung zu entwickeln, die die Strom-Spannungsbeziehungen an der idealen Spule (Bild 2) in Abhängigkeit von diskreten Zeitpunkten wiedergibt, wird die Differenzialgleichung zunächst in Integralform angegeben. Als Integrationsgrenzen werden zwei aufeinander folgende diskrete Zeitpunkte t_{v-1} und t_v gewählt:

$$\int_{i_L(t_{v-1})}^{i_L(t_v)} \mathrm{d}i_L(t) = \frac{1}{L} \cdot \int_{t_{v-1}}^{t_v} u_L(t)\mathrm{d}t \qquad (2)$$

Während sich das Integral auf der linken Seite des Gleichheitszeichens direkt als

$$\int_{i_L(t_{v-1})}^{i_L(t_v)} \mathrm{d}i_L(t) = i_L(t_v) - i_L(t_{v-1}) \qquad (3)$$

angeben lässt, muss das Integral auf der rechten Seite mit Hilfe eines numerischen Integrationsverfahrens approximiert werden. Im Folgenden wird die Trapezregel als Integrationsverfahren gewählt, da sie neben hervorragenden Stabilitätseigenschaften auch den kleinsten Fehlerkoeffizienten unter allen A-stabilen[12] Quadraturformeln aufweist [18]. Unter der Voraussetzung äquidistanter Abtastung mit $t_v = t_{v-1} + T$ gilt

$$\frac{1}{L} \cdot \int_{t_{v-1}}^{t_v} u_L(t)\mathrm{d}t \approx \frac{T}{2 \cdot L} \cdot \left[u_L(t_v) + u_L(t_{v-1})\right] \qquad (4)$$

Der Spulenstrom $i_L(t_v)$ ergibt sich somit näherungsweise zu

$$i_L(t_v) = i_L(t_{v-1}) + \frac{T}{2 \cdot L} \cdot \left[u_L(t_v) + u_L(t_{v-1})\right] \qquad (5)$$

In Bild 3 ist diese Differenzengleichung zusammen mit dem dazugehörigen Signalflussdiagramm dargestellt.

Als nächstes werden die Differenzengleichung und das Signalflussdiagramm der idealen Spule mit Wellengrössen ausgedrückt.

Die so genannten Spannungswellen lassen sich wie folgt aus dem Spulenstrom i_L und der Spulenspannung u_L bestimmen:

$$a_L(t_v) = u_L(t_v) + R_L \cdot i_L(t_v) \text{ und}$$
$$b_L(t_v) = u_L(t_v) - R_L \cdot i_L(t_v) \qquad (6)$$

Entsprechend dem Vorzeichen von i_L ist der Bezugspfeil der Welle a_L stets in das betrachtete Tor hinein gerichtet und der Bezugspfeil der Welle b_L weist aus dem Tor heraus (Bild 4). Der Torbezugswiderstand R_L stellt eine Proportionalitätskonstante zwischen dem Strom i_L und einer Spannung dar. R_L hat physikalisch keine Bedeutung und ist daher prinzipiell frei wählbar. Bei vielen Anwendungen lässt sich eine Spannungswellenbeziehung jedoch durch eine geeignete Wahl des Torbezugswiderstandes stark vereinfachen.

Formt man die Differenzengleichung in Bild 3 so um, dass die Grössen, die von

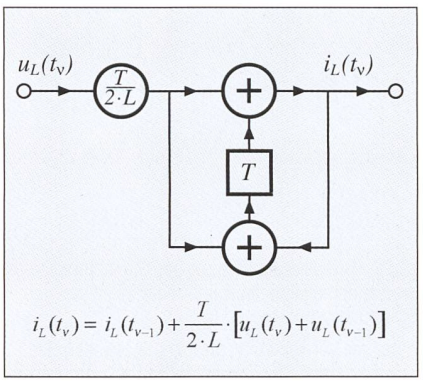

Bild 3 Differenzengleichung und Signalflussdiagramm der idealen Spule in Abhängigkeit von der Spannung u_L und dem Strom i_L

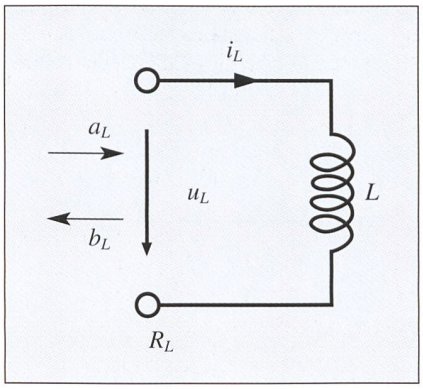

Bild 4 Zur Definition der Spannungswellen

t_v bzw. t_{v-1} abhängen, auf verschiedenen Seiten des Gleichheitszeichens stehen, so können die konstanten Grössen auf beiden Seiten zu 2·L/T zusammengefasst werden:

$$u_L(t_v) - \frac{2 \cdot L}{T} \cdot i_L(t_v) = -\left[(u_L(t_{v-1}) + \frac{2 \cdot L}{T} \cdot i_L(t_{v-1}) \right] \quad (7)$$

Durch die Festlegung $R_L = 2 \cdot L/T$ folgt daraus die einfache Spannungswellenbeziehung

$$u_L(t_v) - R_L \cdot i_L(t_v) = -[u_L(t_{v-1}) + R_L \cdot i_L(t_{v-1})] \quad \text{mit} \quad (8)$$

$$R_L = \frac{2 \cdot L}{T},\ b_L(t_v) = -a_L(t_{v-1})$$

In Bild 5 ist die Differenzengleichung zusammen mit dem zugehörigen Signalflussdiagramm dargestellt. Im Vergleich zu der Realisierung in Bild 3, die neben dem Speicherelement zwei Addierer und einen Multiplizierer erfordert, lässt sich das WD-Äquivalent offensichtlich mit einem wesentlich geringeren Aufwand realisieren.

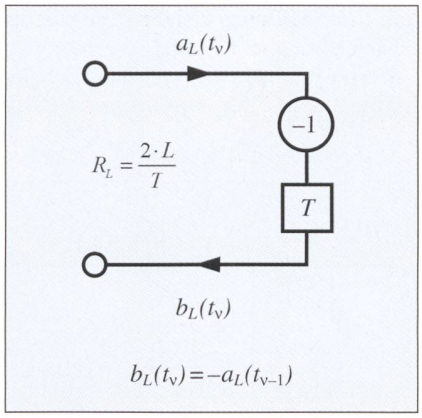

Bild 5 Differenzengleichung und Signalflussdiagramm der idealen Spule in Abhängigkeit von den Spannungswellen a_L und b_L

Durch die Verwendung von Spannungswellen lässt sich die diskrete Darstellung der Spule so weit vereinfachen, dass zu deren Realisierung lediglich ein Speicherelement und ein Invertierer erforderlich sind.

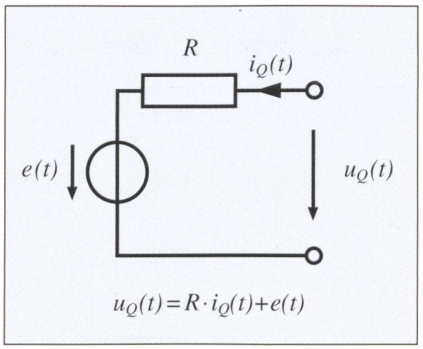

Bild 6 Die Differenzialgleichung und das elektrische Schaltungssymbol sind äquivalente Darstellungsformen der Strom-Spannungsbeziehungen an der realen Spannungsquelle.

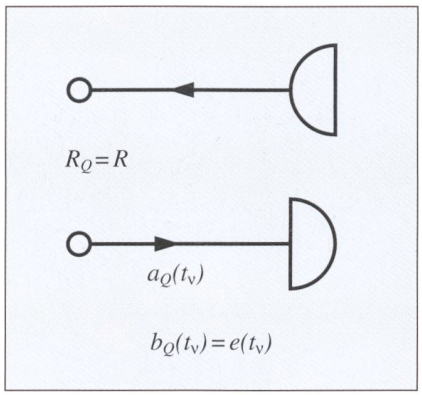

Bild 7 Differenzengleichung und Signalflussdiagramm der realen Spannungsquelle in Abhängigkeit der Spannungswelle b_Q

WD-Äquivalente der idealen Spannungsquelle und des ohmschen Widerstands

Die Wellendigitaldarstellungen der idealen Spannungsquelle und des ohmschen Widerstands lassen sich in einem Schritt ermitteln, wenn man die beiden Elemente zu einer realen Spannungsquelle zusammenfasst (Bild 6).

Da die mathematische Beschreibung der realen Spannungsquelle rein algebraisch ist, gestaltet sich die Bestimmung ihres WD-Äquivalents noch einfacher als bei der idealen Spule. Die Diskretisierung der Zeitvariablen kann in diesem Fall nämlich ohne Anwendung der Trapezregel erfolgen:

$$u_Q(t) = R \cdot i_Q(t) + e(t) \Rightarrow u_Q(t_v) = R \cdot i_Q(t_v) + e(t_v) \quad (9)$$

Um die resultierende diskrete Beziehung mit Hilfe der Spannungswellen

$$a_Q(t_v) = u_Q(t_v) + R_Q \cdot i_Q(t_v) \text{ bzw. } b_Q(t_v) = u_Q(t_v) - R_Q \cdot i_Q(t_v) \quad (10)$$

auszudrücken, setzt man zweckmässigerweise den Torbezugswiderstand R_Q gleich dem Innenwiderstand R der Spannungsquelle und erhält auf diese Weise

$$b_Q(t_v) = u_Q(t_v) - R_Q \cdot i_Q(t_v) = u_Q(t_v) - R \cdot i_Q(t_v) = e(t_v) \quad (11)$$

Die Spannungswelle $a_Q(t_v)$ lässt sich erst dann konkret angeben, wenn das WD-Äquivalent der realen Spannungsquelle mit den Äquivalenten der übrigen Schaltungselemente verbunden worden ist. Das Signalflussdiagramm des WD-Äquivalents der realen Spannungsquelle ist in Bild 7 dargestellt. Die halbkreisförmigen Objekte symbolisieren eine Wellenquelle bzw. -senke.

WD-Äquivalent der gesamten Schaltung

Um die vollständige Referenzschaltung aus Bild 1 in eine WD-Struktur zu überführen, ist neben der Umsetzung der einzelnen Schaltungselemente auch die Übertragung der topologischen Verbindungen (z.B. Reihen- und Parallelschaltungen) erforderlich. Die diskreten Äquivalente dieser Verbindungsstrukturen heissen Adaptoren. Sie werden aus den kirchhoffschen Gleichungen hergeleitet.

Bild 8 zeigt einen Zweitor-Paralleladaptor, der dem WD-Äquivalent einer elektrischen Durchverbindung zwischen zwei Toren entspricht. Der Multiplizierkoeffizient γ wird aus den Torbezugswiderständen der beiden zu verbindenden Elemente ermittelt:

$$\gamma = \frac{R_2 - R_1}{R_2 + R_1} \quad (12)$$

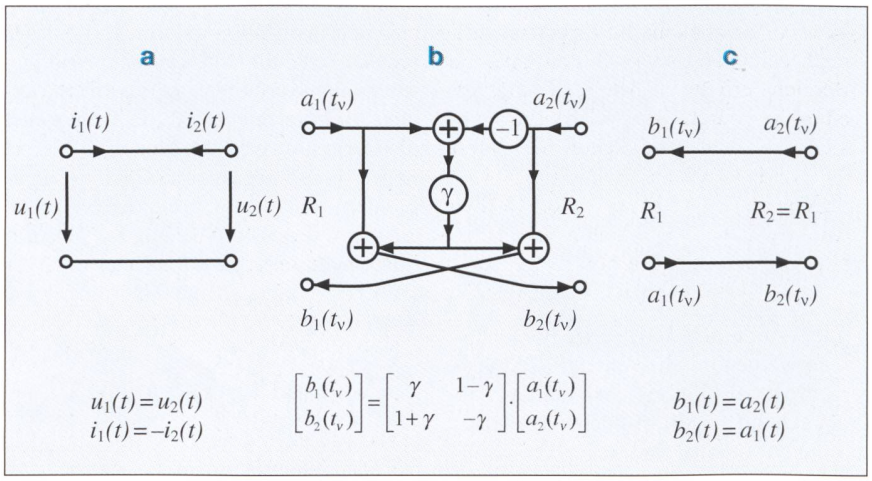

Bild 8 Das WD-Äquivalent der links dargestellten elektrischen Durchverbindung wird Zweitor-Paralleladaptor genannt.

Das Signalflussdiagramm in der mittleren Spalte vereinfacht sich zu dem rechts dargestellten, wenn die Torbezugswiderstände R_1 und R_2 gleich sind.

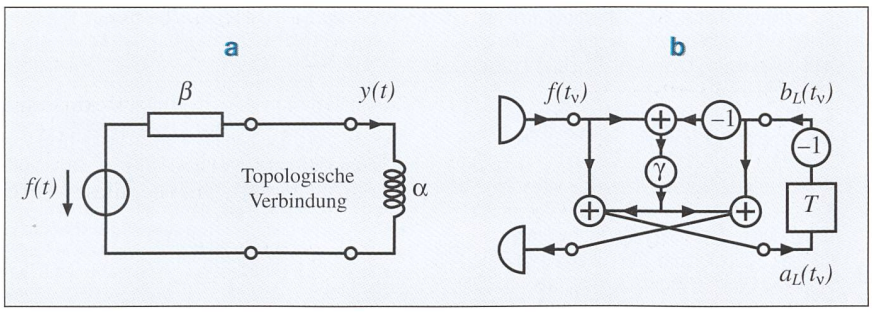

Bild 9 Die Referenzschaltung und die zugehörige WD-Struktur zeigen eine enge Verwandtschaft, da alle Schaltungselemente einzeln in den diskreten Bereich übertragen worden sind.

Bei der Zusammenschaltung von diskreten Äquivalenten, die denselben Torbezugswiderstand besitzen, nimmt γ den Wert null an und die Spannungswellenbeziehungen vereinfachen sich zu

$$b_1(t_v)=a_2(t_v) \text{ bzw. } b_2(t_v)=a_1(t_v) \qquad (13)$$

Das zu diesem Spezialfall gehörende Signalflussdiagramm ist ebenfalls in Bild 8 dargestellt.

Nachdem nun die WD-Äquivalente der drei Teilstücke aus der Referenzschaltung – reale Quelle, topologische Verbindung und ideale Spule – bekannt sind, können diese einfach zu der gesuchten WD-Struktur zusammengesetzt werden. Um den Zusammenhang zwischen den Elementen der Referenzschaltung und denen der WD-Struktur noch einmal bildlich hervorzuheben, ist die Referenzschaltung in Bild 9 um zwei zusätzliche Klemmenpaare ergänzt worden, die die topologische Verbindung kennzeichnen.

Die enge Verwandtschaft zwischen dem diskreten und dem kontinuierlichen System ist der Grund für viele positive Eigenschaften der WD-Strukturen. Beispielsweise bleibt die natürliche Passivität des mit der Referenzschaltung dargestellten physikalischen Systems während des Übergangs in den diskreten Bereich erhalten. Alle Stabilitätsprobleme, die durch die unvermeidlichen numerischen Ungenauigkeiten hervorgerufen werden, können dadurch vermieden werden: WD-Strukturen sind frei von Grenzzyklen[13], sie arbeiten überlaufstabil, und sie sind unempfindlich gegenüber Veränderungen der Multiplizierer-koeffizienten auf Grund begrenzter Wortlängen (weitere positive Eigenschaften der WD-Strukturen sind in [15] wiedergegeben).

Zur Bestimmung der gesuchten Werte der Funktion y berücksichtigt man, dass $y(t_v)$ den diskreten Werten des Spulenstroms $i_L(t_v)$ entspricht. Dieser lässt sich aus den in Formel 6 angegebenen Beziehungen zu

$$i_L(t_v)=\frac{a_L(t_v)-b_L(t_v)}{2\cdot R_L} \qquad (14)$$

bestimmen. Mit $R_L = 2\cdot L/T = 2\cdot\alpha/T$ folgt daraus

$$y(t_v)=\frac{T}{4\cdot\alpha}\cdot\left[a_L(t_v)-b_L(t_v)\right] \qquad (15)$$

An dieser Stelle müsste nun eigentlich darauf eingegangen werden, wie aus der graphischen Spezifikation der Wellendigitalstruktur im Bild 9 auf geeignete Weise ein Algorithmus zur Berechnung der Struktur entwickelt werden kann, der sich mit Hilfe eines Korrektheitsbeweises gegenüber der Spezifikation verifizieren lässt. Da diese Beschreibung jedoch eine intensive Auseinandersetzung mit der Prädikatenlogik erfordern würde, wird hier darauf verzichtet. Der interessierte Leser kann in [13] die Details nachlesen.

Stellvertretend wird die Funktionsfähigkeit der Wellendigitalmethode auf anschauliche Weise im Bild 10 demonstriert, wo die Werte, welche mit dem entworfenen WD-Algorithmus numerisch ermittelt worden sind, der analytischen Lösung der Differenzialgleichung (Formel 5) gegenübergestellt sind. Die drei Kurvenverläufe, welche sich durch eine unterschiedliche Wahl der Abtastzeit T unterscheiden, deuten an, dass bei der Anwendung der Wellendigitalmethode eine beliebige Genauigkeit erreicht werden kann, sofern die Abtastzeit T genügend klein gegenüber der Zeitkonstanten $\tau=\alpha/\beta$ der Differenzialgleichung gewählt wird. Der relative Fehler, der bei der Anwendung der Wellendigitalmethode auftritt, ist proportional zu T^2. Er resultiert aus der Approximation der Integration mit der Trapezregel.

Die Diskussion in diesem Artikel hat sich aus Gründen der mathematischen Einfachheit ausschliesslich auf die Lösung linearer gewöhnlicher Differenzialgleichungen beschränkt. Die Wellendigitalstrukturen sind aber auch schon erfolgreich zur Lösung nichtlinearer partieller Differenzialgleichungen eingesetzt worden [16].

Referenzen

[1] IEC 60880: Software for Computers in the Safety Systems of Nuclear Stations. 1986.

[2] IEC 60987: Programmed Digital Computers Important to Safety for Nuclear Power Stations. 1989-11.

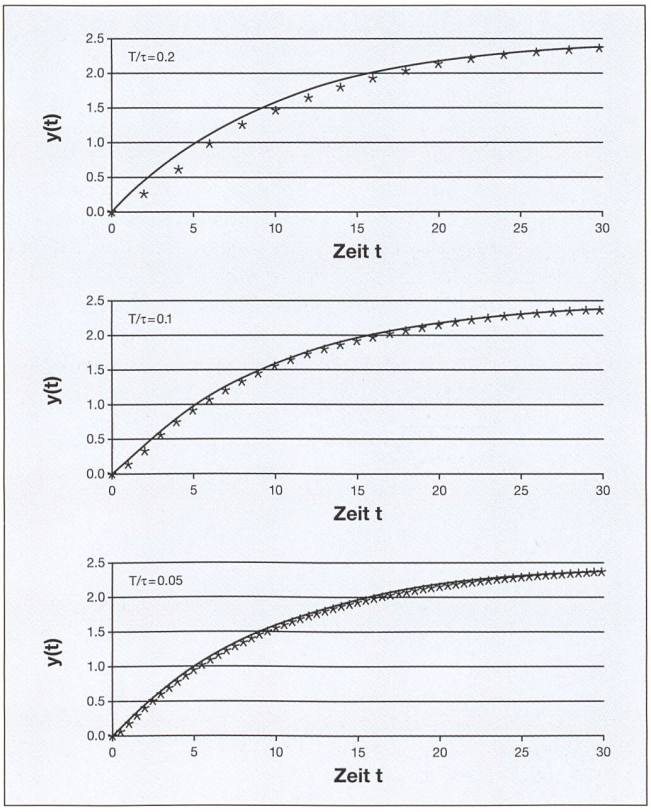

Bild 10 Variation der Abtastzeit T

Mit kleineren Werten der Abtastzeit T nähern sich die durch Sterne gekennzeichneten Ergebnisse der numerischen Berechnung mit Hilfe der Wellendigitalmethode zunehmend besser dem mit einer durchgehenden Linie dargestellten Verlauf der analytischen Lösung an.

[3] IEC 61226: Power Plants – Instrumentation and control systems important for safety-classification. First edition, 1993-05.
[4] ISO 9000: Guidelines for the application of ISO 9001 to the development, supply and maintenance of software. Part 3, 1990-08.
[5] DGQ, NTG: Software-Qualitätssicherung, Aufgaben – Möglichkeiten – Lösungen. 1986
[6] *G. Glöe:* Software-Qualität und Software-Zuverlässigkeit: Eine Herausforderung an die industrielle Informationstechnik. Kolloquium «Software-Qualität» anlässlich der Online 95, Hamburg, 10.02.95.
[7] Eine Herausforderung an die industrielle Informationstechnik. Kolloquium «Software-Qualität» anlässlich der Online 95, Hamburg, 10.02.95.
[8] *P. Liggesmeyer:* Qualitätssicherung softwareintensiver technischer Systeme. Spektrum Akademischer Verlag, Heidelberg, Berlin, 2000, ISBN 3-8274-1085-1.
[9] Bekanntmachungen von Empfehlungen der Reaktor-Sicherheitskommission vom 7. August 1996, Bundesanzeiger Nr. 158a.
[10] Bekanntmachungen von Empfehlungen der Reaktor-Sicherheitskommission vom 29. Oktober 1996, Bundesanzeiger Nr. 214.
[11] *G. Marbach:* Störungen in Computerbussystemen. Nachdruck der Dissertation, ETH Zürich, Nr. 10344, Hartung-Gorre Verlag, Konstanz, 1993, ISBN 3-89191-739-2.
[12] *R. Hellfajer:* Verifikation und Validation paralleler Busse in Leittechnik-Systemen. Dissertation, Ruhr-Universität Bochum, Shaker-Verlag, Aachen, 1998.
[13] *E. Rummert:* Methodik eines formalen Korrektheitsbeweises bei graphisch spezifizierter Software am Beispiel von Wellendigitalfiltern. Dissertation, Ruhr-Universität Bochum, Shaker-Verlag, Aachen, 1998.
[14] *A. Fettweis:* Digital Filter Structures Related to Classical Filter Networks. Archiv für Elektronik und Übertragungstechnik (AEÜ), Band 25, Heft 2, 79-89, 1971.
[15] *A. Fettweis:* Wave Digital Filters: Theory and Practice. Proceedings IEEE, vol. 74, no. 2, Febr. 1986, 270-327.
[16] *A. Fettweis, G. Nitsche:* Numerical Integration of Partial Differenzial Equations by means of Multidimensional Wave Digital Filters. Proceedings, IEEE Int. Symp. Circuits and Systems (ISCAS), 954-957, New Orleans, Louisiana, USA, Mai 1-3, 1990.
[17] *A. Fettweis:* Multidimensional Wave Digital Principles: From Filtering to Numerical Integration. Proceedings, IEEE Conf. on Acoustics, Speech, and Signal Processing (ICASSP), vol. 6, 173-181, Adelaide, Australia, April 19-22, 1994.
[18] *Y. Genin:* A New Approach to the Synthesis of Stiffly Stable Linear Multistep Formulas. IEEE Transactions on Circuit Theory, CT-20, No. 4, 352-360, Juli 1973.
[19] *U. Schöning:* Logik für Informatiker. Spektrum Akademischer Verlag, Heidelberg, 4. Auflage, 1995.

Adresse der Autoren

Fakultät für Elektrotechnik und Informationstechnik, Lehrstuhl für Nachrichtentechnik, Ruhr-Universität Bochum, D-44780 Bochum: Dipl.-Ing. *Katrin Schroeder*, Prof. Dr.-Ing. *Hans-Dieter Fischer*

[1] Unter «korrekter Software» wird eine Software verstanden, deren Funktion mit der Anforderungsspezifikation vollständig übereinstimmt.

[2] Phasenmodell: Arbeitsplan, der den Software-Entwicklungsprozess in seiner Gesamtheit in abgeschlossene Teilaufgaben zerlegt?

[3] «Fehlertolerant» ist mit korrekter Software vereinbar, wenn diese Toleranz-Eigenschaft in der Anforderungsspezifikation berücksichtigt ist. Demgegenüber ist «fehlerarm» nicht mit korrekter Software vereinbar.

[4] Zielorientierung richtet sich dabei nach den letztlich mit der entwickelten Software zu erreichenden Zielen unabhängig von der jeweiligen Anwendung. Beispiel: Eisenbahnsignaltechnik; oberstes Ziel ist der sichere Transport von Menschen von einem Ort A zu einem anderen Ort B, unabhängig davon, ob in diesem oder jenem Gleisabschnitt zwischen A und B beispielsweise Bauarbeiten durchgeführt werden. Anwendungsorientierung richtet sich bei demselben Beispiel dann z.B. auf die zuverlässige Reduktion der Zuggeschwindigkeit innerhalb des Baustellenbereichs.

[5] Korrektheitsbeweise sind i.a. individuell für jedes Programm zu führen. Benutzt man jedoch regelmässige Softwarestrukturen mit wiederverwendbaren Codeteilen, dann hat man das Potenzial, für eine ganze Kategorie von Algorithmen deren korrekte Umsetzung in Software zu zeigen. Dies gelang bis jetzt für eindimensionale Wellendigitalfilter.

[6] Das vorgestellte Verfahren ist für diskrete dynamische Systeme anwendbar, also z.B. für alle Software, die der Simulation technischer Prozesse dient.

[7] Unter dem Lebenszyklus einer Software wird ihre Planung, Entwicklung und Betrieb – zusammen mit erforderlichen Anpassungen, die während des Betriebes erforderlich werden – verstanden.

[8] Beispiele: Beim digitalen Reaktorschutzsystem im Kernkraftwerk Beznau wären unter «Schutzziele» etwa Aktivitätseinschluss, Vermeidung ungewollter Reaktivitätszufuhr oder Sicherstellung der druckführenden Umschliessung zu verstehen. «Angenommene Anwendungen bzw. Ereignisse» wären hingegen – ebenfalls am Bsp. aus dem Bereich der Kernkraftwerke –: zu geringen Filmsiedeabstand vermeiden, Druckabfall sekundärseitig beherrschen, zu geringen bzw. zu hohen Füllstand im Dampferzeuger vermeiden oder Absicherung des Druckgefässes gegen Sprödbruch.

[9] Prädikatenlogik: Axiomensystem für «Prädikate», die als Grundlage eines Beweises dienen, indem man eine Behauptung mittels Beweisregeln auf die Axiome zurückführt [19].

[10] Beispiel für eine logische Formel: die virtuelle Pausenanweisung «wait t». Als Vorbedingung V kann man wählen $V = t_0$; die Nachbedingung ist dann $P = t_0 + t$, wobei angenommen wird, dass die Anweisung «wait t» selbst keine Zeit benötigt.

[11] Eindimensionale Wellendigitalstrukturen weisen in den Speicherelementen zeitliche Verzögerungen auf. Im Gegensatz dazu werden in den Speicherelementen von mehrdimensionalen Strukturen neben den zeitlichen Verzögerungen auch räumliche Verschiebungen durchgeführt. Diese grosse Ähnlichkeit zwischen den mehrdimensionalen und den eindimensionalen WD-Strukturen rechtfertigt somit die Annahme, dass sich der von *E. Rummert* [13] für den eindimensionalen Fall durchgeführte Korrektheitsbeweis auch auf mehrdimensionale Wellendigitalstrukturen erweitern lässt.

[12] Ein Integrationsverfahren ist A-stabil (asymptotisch stabil), wenn die numerische Lösung der Gleichung $dy/dt - p \cdot y = 0$ für alle komplexen p mit negativem Realteil für beliebige Anfangswerte abklingt.

[13] Nimmt ein zunächst von null verschiedenes Eingangssignal von einem bestimmten Zeitpunkt an den Wert null an, so müsste das Ausgangssignal bei unbegrenzter Rechengenauigkeit mit fortschreitender Zeit gegen null konvergieren. Bei begrenzter Wortlänge kann es infolge von Quantisierungsfehlern jedoch vorkommen, dass sich eine periodische Schwingung – der Grenzzyklus – aufbaut.

Le logiciel qui convient aux systèmes dynamiques discrets

Le logiciel adéquat est très important partout où la protection de grands investissements et en particulier la sécurité de l'homme et de l'environnement jouent un rôle prédominant. Ceci englobe des domaines comme la médecine, l'aéronautique, les chemins de fer, les installations chimiques ou les centrales nucléaires. Le logiciel adéquat est plutôt écrit par des gens qui ont tendance à être ordonnés, disciplinés, fiables et à travailler de manière systématique. Afin de produire économiquement du logiciel adéquat, on a de plus en plus recours à des outils assistés par ordinateur.

Sur la base d'un exemple simple d'équation différentielle linéaire ordinaire, le présent article illustre, outre les principes de projet de la méthode numérique dite ondulatoire, l'amélioration de la compréhension dans l'utilisation d'une spécification graphique.