Übersetzung von HyCharts in HDFG *

Christoph Grimm† und Thomas Stauner‡

†Fachbereich Informatik, Johann Wolfgang Goethe-Universität, D-60054 Frankfurt/Main,
Email: grimm@ti.informatik.uni-frankfurt.de

‡Institut für Informatik, Technische Universität München, D-80290 München,
Email: stauner@in.tum.de

Zusammenfassung

Die Spezifikation analoger und analog/digitaler Systeme durch informelle Mittel wie Blockdiagramme und umgangssprachliche Beschreibungen ist durchaus üblich. Derartige Spezifikationen können zu schwerwiegenden Entwurfsfehlern führen, die erst spät – etwa beim Vorliegen des fertigen Produktes – entdeckt werden. Darüberhinaus können informelle Spezifikationen nur schwer als Ausgangsbasis eines automatisierten Entwurfsprozesses dienen. Abhilfe könnte die Spezifikation analog/digitaler Systeme mit Beschreibungstechniken schaffen, die eine formale Syntax und Semantik haben. Dadurch wird effiziente Werkzeugunterstützung möglich und Mißverständnisse können vermieden werden. Insbesondere kann eine Abbildung der Spezifikation auf ausführbare Modelle zum frühzeitigen Experimentieren mit dem System verwendet werden (*rapid prototyping*).

Dieser Beitrag beschreibt die Übersetzung von HyCharts, einer Statechart-ähnlichen Beschreibungstechnik, in hybriden Datenflußgraphen (HDFG), eine blockdiagrammähnliche Darstellungstechnik, die zur Darstellung und Optimierung analog/digitaler Schaltungen geeignet ist. Damit wird ein Schnittstelle zwischen Spezifikationsmethoden aus dem Softwareengineering und Entwurfsmethoden analoger und analog/digitaler Systeme hergestellt.

1 Einleitung

Fehler bei der Erfassung der Anforderungen verursachen nach [1] 50% der Probleme bei ausgelieferten, eingebetteten Systemen, die von Kunden gemeldet werden. Diesen Fehlern könnte durch die Erfassung der Anforderungen mit formalen Notationen, sowie durch die Fähigkeit, Prototypen in einer frühen Entwicklungsphase zu erstellen, begegnet werden. Während zum Prototyping und zur Spezifikation digitaler Systeme auf ausgereifte Werkzeuge zurückgegriffen werden kann, befinden sich Methoden zur Spezifikation und zum Rapid Prototyping analoger und analog/digitaler Schaltungen und Systeme noch im Entwicklungsstadium.

Obwohl Werkzeuge wie MATLAB/Stateflow [12] oder MatrixX/BetterState [7] die Spezifikation und Modellierung analoger Funktionen erlauben, und auch dazu verwendet werden, wird von diesen Werkzeugen kein durchgängiger Weg zu einer analogen Schaltung unterstützt. Dagegen wird von Werkzeugen, die das Rapid-Prototyping analoger und analog/digitaler Schaltungen

*Diese Arbeit wurde von der DFG unter den Kennziffern Br 887/9 und Wa 357/14 im Rahmen des Schwerpunktprogramms *Entwurf und Entwurfsmethodik eingebetteter Systeme* unterstützt.

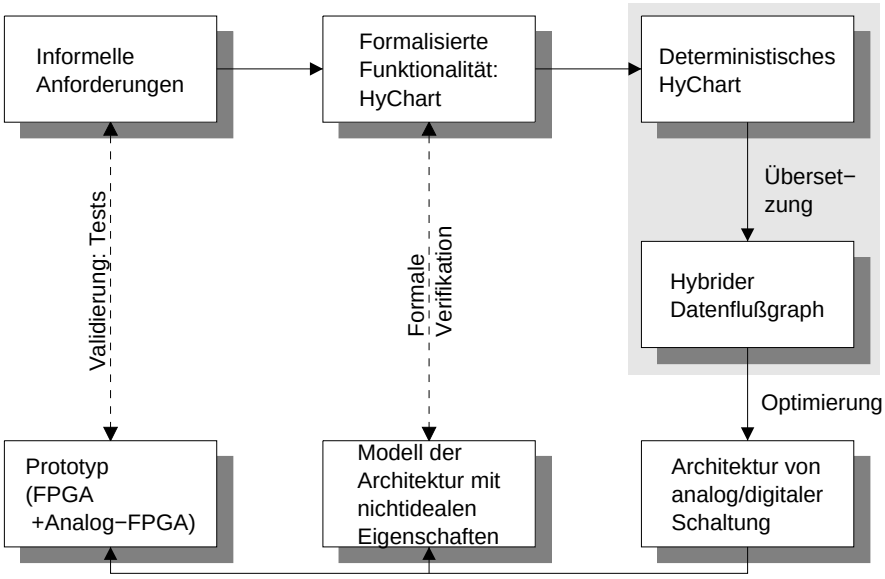


Abbildung 1: Einsatz der Übersetzung im Entwicklungsprozeß.

unterstützen, wie etwa [2, 8], bereits eine Blockdiagramm-Darstellung einer möglichst effizienten Architektur vorausgesetzt. Ein Prototyping aus abstrakten Spezifikationen ist damit nicht möglich.

In diesem Beitrag wird eine Übersetzung von deterministischen *HyChart* Spezifikationen in *hybride Datenflußgraphen (HDFG)* vorgestellt. Die Übersetzung ermöglicht es, analog/digitale Systeme visuell und abstrakt durch HyCharts zu spezifizieren und dann auf der Basis von HDFG zu optimieren und auf analog/digitale Schaltungen abzubilden [9]. Bild 1 gibt einen Überblick möglicher Anwendungen der Übersetzung: Zum einen bietet sie die Möglichkeit die mit HyCharts erfaßten Anforderungen durch einen Prototypen zu validieren. Dazu muß das HyChart deterministisch gemacht werden. Zum anderen kann ein Feinentwurf, der nach einer Reihe von Entwurfsschritten auf dem deterministischen HyChart und nach manuellen Optimierungen an der synthetisierten Schaltungsarchitektur entsteht, gegenüber den erfaßten Anforderungen verifiziert werden.

Die Schwierigkeit der Übersetzung liegt darin, daß HyCharts [5] – ähnlich wie MatrixX und Simulink – auf zwei unterschiedlichen Beschreibungstechniken basieren: Eine (HyACharts) für die Spezifikation von Datenfluß (oder Architektur), und eine zweite (HySCharts) zur Darstellung des Kontrollflusses des Systems. Dagegen werden in Hybriden Datenflußgraphen hybride Systeme homogen durch Datenfluß beschrieben, um von der Struktur der Spezifikation abstrahieren und die Architektur optimieren zu können.

Im folgenden werden zunächst die Beschreibungstechniken HyChart und HDFG beschrieben. In Abschnitt 4.1 wird das Prinzip der Übersetzung von HyACharts erläutert. In Abschnitt 4.2 wird die Übersetzung von HySCharts beschrieben.

2 HyCharts

HyCharts [6, 5] kombinieren zwei grafische Beschreibungstechniken, *HyA Charts* und *HySCharts*. Sie betrachten ein System als Netzwerk von Komponenten, die über gerichtete Kanäle zeitsyn-

chron kommunizieren. HyACharts dienen zur Spezifikation der Systemarchitektur, mit HySCharts wird das Verhalten der Komponenten eines hybriden Systems spezifiziert. HySCharts können als eine Erweiterung der Statechart Variante ROOMcharts [10] betrachtet werden. Sie erweitern diese um *Aktivitäten* zur Beschreibung kontinuierlichen Verhaltens.

Den durch HySCharts spezifizierten Komponenten liegt das Maschinenmodell von Abb. 2 zugrunde. Es besteht aus einem kombinatorischen (oder diskretem) Teil ($Com^\dagger$), einem kontinuierlichen Teil (Ana), einer Feedback Schleife mit einer infinitesimalen Verzögerung (Lim_z), und einer Projektion ($Out^\dagger$). Die Feedback Schleife und Lim_z modellieren den *Zustand* der Maschine. Zu jedem Zeitpunkt t kann die Komponente dadurch auf die empfangene Eingabe und die produzierte Ausgabe "genau vor" t zurückgreifen. Der kombinatorische Teil steuert den analogen Teil und erlaubt diskrete Manipulationen der Zustandsvariablen. Er enthält keinen Speicher. Abhängig von der gegenwärtigen Eingabe und dem zurückgeführten Zustand gibt er nicht-deterministisch und instantan, d.h. ohne zeitlicher Verzögerung, den Folgezustand an. Der analoge Teil benutzt den Folgezustand um eine Aktivität auszuwählen, die die kontinuierliche Entwicklung der Variablen der Komponente spezifiziert. Zudem gibt der Folgezustand den Startwert für die Aktivität vor. Der kombinatorische Teil kann nur zu isolierten Zeitpunkten einen Folgestand bestimmen, der sich vom zurückgeführten Zustand unterscheidet. Während der Intervalle zwischen diesen Zeitpunkten ist er untätig, entsprechend ändert sich die auswählte Aktivität während des Intervalls nicht, sofern nicht die externe Eingabe springt.

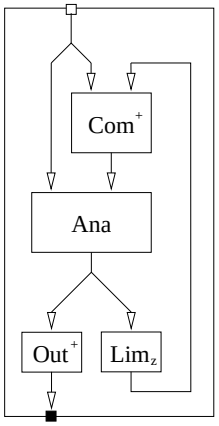


Abbildung 2: HyAChart des hybriden Maschinenmodells.

Der analoge Teil legt das Ein-/Ausgabeverhalten der Maschine fest wann immer der kombinatorische Teil untätig ist. Er kann immer dann eine neue Aktivität wählen, wenn sich seine Eingabe von der Umgebung oder vom kombinatorischen Teil diskret ändert.

Der Zustand $\kappa.\sigma$ der Maschine besteht aus dem Kontrollzustand κ und dem Datenzustand σ. Neben den Werten von privaten Variablen enthält der Datenzustand die Belegung der Ausgabevariablen und eine zeitverzögerte Kopie der Werte der Eingabevariablen. Die in Zeit erweiterte Projektion $Out^\dagger$ selektiert die Belegung der Ausgabevariablen und macht diesen Teil den Datenzustands über die Ausgabekanäle nach außen hin sichtbar. Die im Zustand enthaltenen Kopien der Eingabevariablen speichern zu jedem Zeitpunkt t die externe Eingabe genau vor t. Durch diese verzögerte Kopie der alten Eingabe kann der kombinatorische Teil diskrete Änderungen in der Eingabe erkennen und darauf reagieren. Der analoge Teil aktualisiert diesen Teil des Datenzustands, die Verzögerung erfolgt durch Lim_z.

Aus syntaktischer Sicht sind sowohl HyACharts als auch HySCharts hierarchische Graphen, bzw. können im Fall von HySCharts zu hierarchischen Graphen umgeformt werden. Die Semantik beider HyChart Varianten wir daher definiert indem den Operatoren, aus denen die hierarchischen Graphen aufgebaut sind, Bedeutung zugewiesen wird. Bei HyACharts wird eine *multiplikative Semantik* für die Graphen verwendet. Alle Knoten im Graphen sind gleichzeitig aktiv und entsprechen Komponenten, die Kanten definieren die Kommunikationskanäle zwischen den Komponenten. HyACharts spezifizieren daher den Datenfluß zwischen den enthaltenen Komponenten. Für HySCharts wird eine *additive Semantik* benutzt. Dabei kann jeweils nur ein Knoten im Graphen aktiv sein. Die Knoten entsprechen Automatenzuständen (Kontrollzuständen). Die Kanten beschreiben wie die Kontrolle von einem aktiven Knoten zum nächsten wechselt und entsprechen den Transitionen. Die HySChart definiert somit eine Zustandsübergangsrelation, nämlich den kombinatorischen Teil der hybriden Maschine. Der analoge Teil der Maschine wird nach syntaktischen Transformationen, die allein die Hierarchie der Zustände

in der ursprünglichen HySChart berücksichtigen, mit Hilfe einer additiven Semantik ebenfalls aus der HySChart abgeleitet. HyCharts basieren auf folgenden Operatoren auf hierarchischen Graphen: $\star$ (unabhängige Komposition von Knoten), ; (sequentielle Komposition von Knoten), $\uparrow$ (Feedback) sowie $\bullet\!\!<$ (Verzweigung von Kanten), $>\!\!\bullet$ (Zusammenführen von Kanten) und $\diagdown\!\!\diagup$ (Vertauschen von Kanten).

Abb. 2 enthält bereits ein Beispiel für eine HyAChart. Sie definiert den Datenfluß in der hybriden Maschine. Abb. 3 zeigt die HySChart für einen Sägezahngenerator. Sie enthält zwei Zustände, `up` und `down`. In `up` wird die aufsteigenden Flanke des Signals durch Aktivität `rise` festgelegt, in `down` die absteigende. Wenn das Signal einen gewissen Grenzwert erreicht hat, erfolgt ein Zustandswechsel über `r2f` bzw. `f2r`. Die Aktivitäts und Transitionsnamen beziehen sich dabei auf Prädikate über den Eingangs- und Zustandsvariablen, die explizit angeben werden müssen. Im Abschnitt 4.3 werden wir näher auf das HySChart und seine Übersetzung in HDFG eingehen.

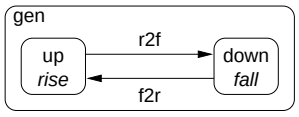


Abbildung 3: HySChart eines Sägezahngenerators.

3 Hybride Datenflußgraphen (HDFG)

Hybride Datenflußgraphen (HDFG [4, 3]) sind eine semiformale, graphische, homogene Darstellung hybrider Systeme. Sie beschreiben ein System durch die graphische Angabe der Zustandsänderungs bzw. -übergangsfunktion, die auf inneren Zuständen operiert. Innere Zustände werden durch spezielle Kanten E_z beschrieben, die eine elementare Verzögerung darstellen. Knoten, die direkt auf inneren Zuständen operieren, werden als *Zustandsknoten* bezeichnet, andere Knoten werden als *kombinatorische Knoten* bezeichnet.

Alle Zustandsknoten besitzen eine *Aktivierungsregel*, die bestimmt, zu welchem Zeitpunkt oder unter welcher Bedingung eine Zustandsänderung stattfindet. Zur Beschreibung heterogener Systeme können zum Beispiel die folgenden Aktivierungsregeln in einem HDFG kombiniert werden:

- Kontinuierliche Änderungen können mit der Aktivierungsregel a_{DESS} beschrieben werden, die zu einer kontinuierlichen Berechnung führt.
- Zeitdiskret spezifizierte Systemteile ("DTSS") können mit der Aktivierungsregel a_{DTSS} beschrieben werden, die zu einer regelmäßigen Aktivierung des Knotens sorgt.
- Ereignisdiskrete Systemteile ("DEVS") können mit Teilgraphen der Aktivierungsregel a_{DEVS} beschrieben werden. Diese Aktivierungsregel aktiviert den Knoten bei einer Änderung an einer seiner Eingangskanten.

HDFG-Knoten N und HDFG-Kanten E können zu HDFG verknüpft werden, solange die entstehende Struktur keine kombinatorischen Zyklen enthält; dies kann sichergestellt werden, wenn jeder Zyklen mindestens eine Kante aus E_z oder einen Knoten der Funktion *intdt* enhält. Die Semantik eines HDFG ergibt sich aus der Struktur des HDFG, sowie den Funktionen und Aktivierungsregeln seiner Knoten (Abb. 4).

Operational kann die Semantik auch wie folgt erklärt werden: Zu Beginn (t_1) sei ein initialer Zustand durch eine Belegung aller Zustandskanten mit Werten oder Segmenten bekannt. Zu einem Zeitpunkt $t_2 > t_1$ werde mindestens ein Knoten des HDFG aktiv. Die Struktur des HDFG beschreibt eine Verkettung von Funktionen zur Berechnung seiner Ausgabe aus Werten der

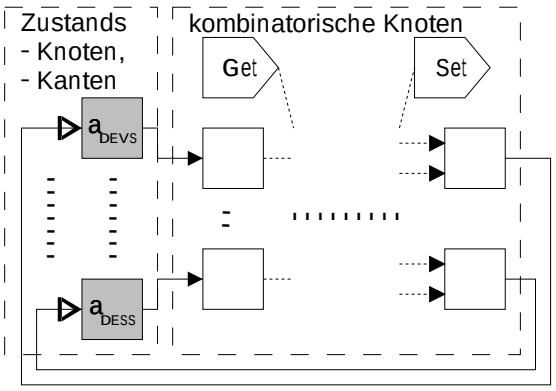


Abbildung 4: Darstellung hybrider Systeme mit HDFG.

Vergangenheit bzw. aktuellen Eingaben. Die Verkettung der Funktionen ist azyklisch. Hieraus ergibt sich ein neuer Zustand, der beibehalten wird, bis wieder ein Knoten aktiv wird.

Zwei spezielle Funktionen f – *select* und *iterate* – erlauben die funktionale Darstellung von Kontrollfluß: *iterate* führt eine Fixpunktiteration auf einem als Parameter gegebenen HDFG durch, und *select* erlaubt die Auswahl eines Signals aus einer Menge von Eingabesignalen.

HDFG können hierarchisch strukturiert werden. In diesem Fall wird die Funktion eines Knotens durch einen HDFG beschrieben.

4 Übersetzung von HyCharts in HDFG

Die hier vorgestellte Übersetzung setzt folgende Eigenschaften des HyChart voraus: Wichtigste Voraussetzung ist, daß die vorliegenden HySCharts *deterministisches Verhalten* spezifizieren, da HDFG die Darstellung von nichtdeterministischem Verhalten nicht erlauben. Weiterhin wird vorausgesetzt, daß alle elementaren Bestandteile in HyACharts und HySCharts, d.h. Bestandteile die nicht in weitere HyACharts oder HySCharts zerlegt werden, durch HDFG beschrieben werden. Das betrifft insbesondere Aktivitäten und Transitionsbedingungen in HySCharts. Außerdem wird vorausgesetzt, daß jede Feedbackschleife in einem HyAChart (und in Subcharts) eine Verzögerung enthält. Darüber hinaus sind Schleifen ohne Verzögerung erlaubt, wenn ein eindeutiger Fixpunkt für alle Mengen von initialen Werten existiert. Um sicherzustellen, daß der resultierende HDFG eine wohldefinierte Semantik besitzt, müssen Schleifen eine Integration oder eine infinitesimale Verzögerung Lim_z, wie z.B. im HyChart-Maschinenmodell (Abb. 2), enthalten. Die Eindeutigkeit des Fixpunktes stellt sicher, daß das resultierende HyAChart deterministisch ist, wenn alle seine Komponenten dies sind. Die zu übersetzenden HyACharts sollen außerdem keine Zusammenführungs Operatoren ($\succ\!\bullet$) enthalten. Diese synchronisieren zwei Kanäle von HyACharts (ähnlich der Synchronisation in Petri-Netzen). Tatsächlich wurden diese Kanten bis jetzt noch in keiner Anwendung von HyACharts verwendet.

4.1 Übersetzung von HyACharts

Da sowohl HyACharts als auch HDFG auf dem Datenflußprinzip beruhen, ist die Übersetzung HyACharts in HDFG unkompliziert. Wir skizzieren daher das Prinzip. Ausgehend von der

textuellen Darstellung eines HyAChart mit den in Abschnitt 2 eingeführten Operatoren, kann die Übersetzung in HDFG durch strukturelle Induktion definiert werden.

- Wie oben angenommen, ist jeder elementare Knoten eines HyAChart entweder durch ein HySChart oder einen HDFG gegeben. Im ersten Fall ist der HDFG die Übersetzung des Knotens. Im zweiten Fall wird das Übersetzungsverfahren auf das HyAChart für das Maschinenmodell (Abb. 2) angewandt. Der kombinatorische und analoge Teil der Maschine sind dabei durch das HySChart definiert. Auf die Übersetzung der Komponenten des Maschinenmodells gehen wir im folgenden Beispiel sowie in Abschnitt 4.2 ein.
- Die Verknüpfung von Knoten mit unabhängiger Komposition, sequentieller Komposition und Feedback wird übersetzt in einen HDFG, der die Knoten in der gleichen Art und Weise verknüpft. Die unabhängige Komposition entspricht damit der parallelen Komposition von Komponenten.
- Die Übersetzung von Kantenverzweigung und Kantenvertauschung ergibt einen HDFG, dessen Ein/Ausgangsports entsprechend verbunden sind. Im Fall der Verzweigung sind also alle Ausgangsports an den Eingangsport angeschlossen und im Fall der Vertauschung erfolgt die Verbindung "über Kreuz".

Alle HDFGs, die aus dieser Übersetzung resultieren, haben zunächst die Aktivierungsregel a_{DESS}.

Beispiel: Übersetzung des Maschinenmodells. Der sich aus der Übersetzung des HyACharts des Maschinenmodells ergebende HDFG ist in Bild 5, rechts, dargestellt. Zur besseren Übersicht wurde der resultierende HDFG hierarchisch flach dargestellt. $C(Com^{\dagger})$ ist die Übersetzung des kombinatorischen Teils, und $C(Ana)$ die Übersetzung des analogen Teils. Com und Ana sind durch die HySChart spezifiziert. Ihre Übersetzung wird in Abschnitt 4.2 definiert. Links unten befindet sich die Übersetzung von $Out^{\dagger}$. In diesem wird das Tupel aus Kontroll/Datenzustand in seine Teile zerlegt. Rechts unten ist die Übersetzung der unendlich kleinen Verzögerung Lim_z, und zwar eine Kante aus der Menge E_z (Aktivierungsregel a_{DESS}). Der initiale Wert z wird als Attribut neben die Kantenspitze notiert. Die HDFG für $Out^{\dagger}$ und Lim_z haben beide die Aktivierungsregel a_{DESS}.

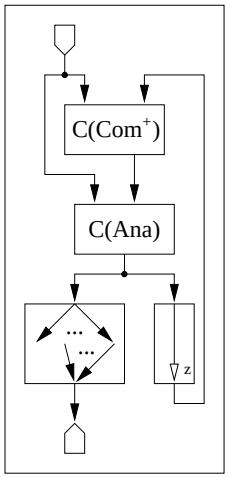


Abbildung 5: HDFG des hybriden Maschinenmodells.

Außer den Kanten, die die externen Ein/Ausgaben transportieren, sind alle Kanten Träger von Werten des Typs $\mathbb{N} \times \mathcal{S}$. Dabei kodiert $\mathbb{N}$ den Kontrollzustandsraum und $\mathcal{S}$ den Datenzustandsraum. An der Schnittstelle von $C(Com^{\dagger})$ und $C(Ana)$ wird dieses Tupel in Kontroll- und Datenzustand zerlegt, da dies die Definition von $C(Com^{\dagger})$ und $C(Ana)$ im nächsten Abschnitt erleichtert.

4.2 Übersetzung von HySCharts

4.2.1 Der kombinatorische Teil

Bevor wir die Übersetzung einiger Schlüsselkonzepte erklären, müssen wir kurz darauf eingehen, wie der kombinatorische Teil aus der HySChart abgeleitet wird. Details hierzu sind in [5] beschrieben. Auf der Ebene der Semantik ist der kombinatorische Teil eine Zustandsübergangsfunktion. Er wird durch einen hierarchischen Graphen definiert, der wiederum aus dem HySChart abgeleitet ist. Die Ableitung kann dabei als Makroexpansion aufgefaßt werden: Komplexe HySChart Konstrukte, wie etwa Zustände, werden zu Teilgraphen expandiert. Der Teilgraph, in den ein primitiver Zustand übersetzt wird, wird dabei nicht durch einen Namen,

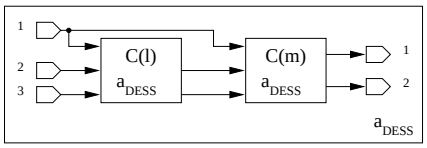


Abbildung 6: HDFG für additive sequentielle Komposition.

sondern durch eine Nummer in dem Subgraphen, in dem er vorkommt, identifiziert. Der hierarchische Graph für einen HySChart Zustand n mit l Unterzuständen besteht also aus l Teilgraphen, die mit den Nummern $1, \ldots, l$ angesprochen werden können. Das Zusammenfassen von zwei Zuständen n und m mit l bzw. h Unterzuständen zu einem hierarchischen Zustand auf Ebene der HySChart erfordert auf der Ebene des hierarchischen Graphen eine Verschiebung im Namensraum. Der entstehende Graph $n \star m$ enthält $l + h$ Teilgraphen, die die Unterzuständen von n und m modellieren. Die Unterzustände von n können auch weiterhin mit den Nummern $1, \ldots, l$ angesprochen werden, der Namensraum für die Unterzustände von m wird in $n \star m$ aber um l verschoben, so daß die Unterzustände von m in der Komposition $n \star m$ die Nummern $l + 1, \ldots, l + h$ erhalten. Der Operator für die unabhängige Komposition $\star$ besorgt diese Verschiebung.[1] Die anderen Operatoren sind kompatibel dazu definiert. So führt z.B. der Vertauschungsoperator eine Umnummerierung im Namensraum aus.

Während die HySChart Zustände zu Teilgraphen expandiert werden, werden Transitionsbedingungen ("*action guards*") und die damit verknüpften Aktionen ("*action bodies*") zu Knoten in dem hierarchischen Graphen. Auf der Ebene der Semantik werden die Knoten zu Funktionen; der Graph gibt an, wie diese Funktionen verknüpft sind. Ein Tupel aus Datenzustand und einer Nummer, dem Kontrollzustand, der einen der Zustände der HySChart referenziert, wählen einen Pfad durch den Graphen aus. Dieser Pfad entspricht einer spezifischen Verknüpfung der einzelnen Funktionen, und resultiert in einem neuen Tupel aus Daten- und Kontrollzustand.

Übersetzung des hierarchischen Graphen mit additiver Semantik. Die Übersetzung des kombinatorischen Teils kann wieder durch Induktion über die Struktur des aus dem HySChart erzeugten hierarchischen Graphen definiert werden. Im folgenden geben wir die Übersetzung für die sequentielle Komposition und die unabhängige Komposition an.[2]

- $n = l; m$ (sequentielle Komposition). Abb. 6 zeigt den entsprechenden HDFG. Die Übersetzung in HDFG verwendet den ersten Eingangs-Port des HDFG-Knotens für die externe Eingabe (Typ $\mathcal{I}$), den Zweiten zur Übertragung des Datenzustands (Typ $\mathcal{S}$) und den Dritten für den Kontroll-Zustand (Typ $\mathbb{N}$). Der Erste Ausgangs-Port gibt den nächsten Daten-Zustand aus, der Zweite trägt den nächsten Kontroll-Zustand. Da alle Aktionen in einem HySChart auf den selben Eingaben operieren, wird die Eingabe für beide Knoten kopiert. Die Übersetzung entspricht damit im Wesentlichen der sequentiellen Komposition der Übersetzung von l mit der von m.
- $n = l \star m$ (unabhängige Komposition), wobei Teilgraph l g Eingangs- und h Ausgangs-Ports besitzt. Abb. 7 zeigt den HDFG für $C(l \star m)$. In diesem HDFG arbeiten $C(l)$ und $C(m)$ parallel, und zwei Auswahl-Knoten wählen die Ausgabe von $C(l)$ or $C(m)$ als Ausgabe des entsprechenden hierarchischen Knotens, abhängig vom erreichten Kontrollzustand (an Port 3). Der gepunktet umrandete Bereich in Bild 7 zeigt die Teile des übersetzten HDFG, die die oben beschriebene Verschiebung des Namensraumes durchführen und m anwenden.

[1] Dieses Vorgehen ähnelt der DeBruijn Notation für Lambda-Terme.

[2] Hierbei wird eine zeitbehaftete Semantik der Graphoperatoren vorausgesetzt, vgl. [11].

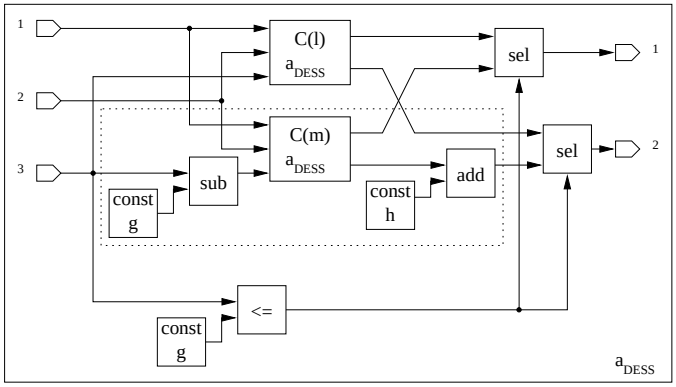


Abbildung 7: HDFG der disjunkten Summe.

Die Übersetzung der verbleibenden Operatoren wird in [11] angegeben. Die Übersetzung des Feedbacks stützt sich dabei auf die *iterate* Knoten von HDFG ab. Hier wird ein Teilgraph so lange wiederholt durchlaufen, bis der neue Kontrollzustand auf einen anderen Teilgraphen verweist.

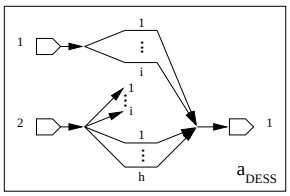
Optimierungen. Alle HDFG-Knoten, die aus der Übersetzung resultieren, sind durch die Aktivierungsregel a_{DESS} ständig aktiv. Für Knoten, die ausschließlich vom Kontrollzustand abhängen bzw. diesen manipulieren, kann alternativ die Aktivierungsregel a_{DEVS} verwendet werden, ohne das Verhalten des HDFG zu ändern.

4.2.2 Der analoge Teil

Der analoge Teil *Ana* beschreibt, wie sich die Variablen ändern, solange keine diskrete Transition in dem HySChart möglich ist, und aktualisiert die verzögerte Kopie der externen Eingabe. Ähnlich wie *Com* wird auch für *Ana* ein hierarchischer Graph aus der HySChart abgeleitet, allerdings mit anderen Transformationsregeln (vgl. [5]). Die Regeln stellen sicher, daß der hierarchische Graph aus einer unabhängigen Komposition von sequentiell komponierten Aktivitäten besteht. Erstes Element der sequentiell komponierten Aktivitäten ist dabei immer die Aktivität *upd*, die die Kopie der externen Eingaben aktualisiert. Ebenso wie bei dem kombinatorischen Teil wird der hierarchische Graph für den analogen Teil in der in Zeit erweiterten additiven Semantik interpretiert. Folglich können dieselben Übersetzungsregeln für die unabhängige Komposition und die sequentielle Komposition angewendet werden. Weitere Operatoren enthält der Graph für *Ana* nicht.

In der Übersetzung setzen wir voraus, daß die einzelnen Aktivitäten durch HDFG beschrieben werden. Der HDFG der Aktivität *upd* ist in Abb. 8, links, dargestellt. Man beachte, daß der erste Eingangsport in diesem Graph die externen Eingaben, und der Zweite den Datenzustand überträgt. Im Graphen wird der Datenzustand aufgeteilt in die verzögerten Eingaben (Kanäle 1 bis i), die nicht weiter verwendet werden, und die übrigen internen und externen Variablen (Kanäle 1 bis h). Diese werden mit den externen Eingaben zu einem Tupel verbunden und bilden den neuen Datenzustand.

Alle Aktivitäten arbeiten allein auf dem Datenzustand. Um die obige Übersetzung der (additiven) Operatoren auf den Graphen für den analogen Teil anwenden zu können, müssen die HDFGs für die Aktivitäten daher um einen Port für den Kontrollzustand erweitert werden. Abb. 8, rechts, zeigt diese Erweiterung. Da Aktivitäten den Kontrollzustand nicht ändern, mo-



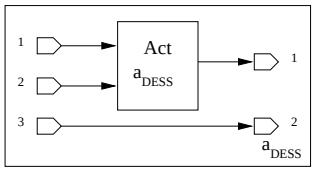
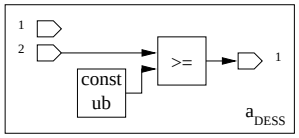


Abbildung 8: HDFG der Aktivität *upd* sowie die Erweiterung der Aktivitäten durch den Kontrollzustand.

difiziert der Graph in der Abbildung die Eingabe auf dem dritten Port nicht. Der HDFG für $C(Ana)$ ergibt sich nun aus der Anwendung der obigen Übersetzung auf den hierarchischen Graphen für Ana.

4.3 Beispiel



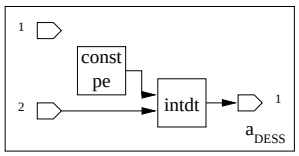


Abbildung 9: Transitionsbedingung für $r2f$ und Aktivität *rise* als HDFG.

Als Beispiel verwenden wir das HySChart für den Sägezahngenerator aus Abb. 3. Der Sägezahngenerator hat keine externen Eingänge, keine privaten Variablen und nur eine reellwertige Ausgabe, y. Damit ist der Datenzustandsraum $\mathcal{S} = \mathbb{R}$. Die Transitionsbedingung für $r2f$ sowie die Aktivität *rise* sind durch die HDFG in Abb. 9 beschrieben. Der erste Eingang dieser zwei HDFG entspricht der externen Eingabe und ist hier deshalb nicht verbunden. Die Transitionsbedingung von $r2f$ ist ein HDFG, der das Prädikat $y \geq ub$ ausdrückt. Die Aktion von $r2f$ ist die Identität auf $\mathcal{S}$, d.h. die Aktion läßt y unverändert. Aktivität *rise* ist definiert als $y = \int pe\, dt$. Durch das Maschinenmodell ist nämlich der Eingangsport des HDFG von *rise* indirekt mit seinem Ausgangsport via Lim_z und $Com^\dagger$ verbunden, solange der gegenwärtige Kontrollzustand up ist. Solange keine Transitionsbedingung $true$ ist, ist $Com^\dagger$ die identische Abbildung, und *rise* erhält somit seine eigene Ausgabe mit einer infinitisimalen Verzögerung. Die HDFG-Funktion *intdt* berechnet dann aus dem vergangenen Zustand sowie dem Wert pe den Wert $y = \int pe\, dt$.

Die Transitionsbedingung von $f2r$ sowie die Aktivität $fall$ seien ebenfalls durch HDFG gegeben und entsprechen $y \leq lb$ bzw. $y = \int ne\, dt$. Die Aktion von $f2r$ ist wiederum die Identität. ub, lb, pe und ne sind Konstanten. Der Startzustand des HySCharts sei up mit $y = 0$.

Bei der Bestimmung des kombinatorischen Teils aus dem HySChart wird der Zustand up mit der Zahl 1 codiert; der Startzustand ist somit $z = (1, 0)$, wobei die zweite Zahl dieses Tupels der Anfangswert von y ist. Das HySChart kann wie in [5] beschrieben auf einen additiv interpretierten hierarchischen Graphen für den kombinatorischen Teil und einen weiteren für den analogen Teil reduziert werden. Der Graph für den kombinatorischen Teil enthält 22 Operatoren, die jeweils bei der Übersetzung einen HDFG erzeugen. Wegen dieser Größe kann der resultierende HDFG hier nicht angeben werden. Stattdessen wird die Übersetzung des analogen Teils, die nach dem selben Prinzip arbeitet, aber einfacher ist, beschrieben.

Der analoge Teil des HySCharts wird durch $Ana = rise \star fall$ beschrieben. Der entsprechende HDFG ist in Abb. 10 dargestellt. Zur Übersetzung wird $C(l)$ in Abb. 7 durch $C(rise)$ und $C(m)$ durch $C(fall)$ ersetzt. Die Konstanten g und h sind beide 1. Die Ausgabe von *rise* wird ausgewählt, wenn der Kontrollzustand 1 ist (Knoten up im HySChart), andernfalls wird die Ausgabe von $fall$ ausgewählt ($down$ im HySChart).

Den HDFG für das gesamte HySChart erhält man durch Ersetzung von $C(Com^\dagger)$ und $C(Ana)$

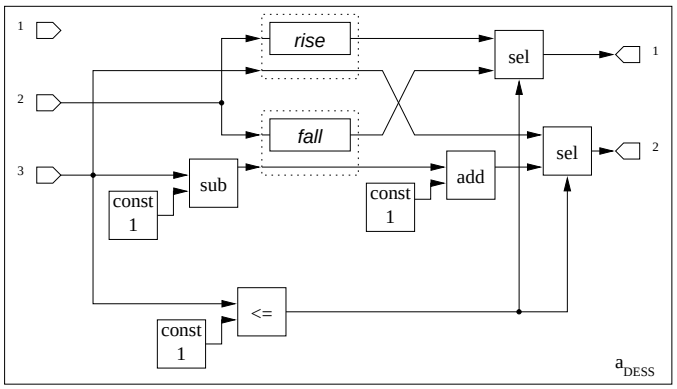


Abbildung 10: HDFG des analogen Teils des Sägezahngenerators.

im übersetzten Maschinenmodell (Abb. 5) durch die aus der Übersetzung resultierenden HDFG.

5 Zusammenfassung, Ausblick

In diesem Beitrag wurde eine Möglichkeit vorgestellt abstrakte, Statechart-ähnliche, visuelle Beschreibungstechniken in eine sehr einfache und homogene, datenflußbasierte Darstellung, nämlich HDFG, zu übersetzen. Diese erlauben insbesondere die systemweite Optimierung, etwa zur Codeerzeugung oder zur Synthese digitaler oder analoger Schaltungen.

In weiteren Arbeiten soll die praktische Durchführbarkeit des Ansatzes anhand einer Fallstudie evaluiert werden; gegebenfalls soll auch an Optimierungen gearbeitet werden, die die Komplexität der erzeugten HDFG reduzieren.

Literatur

[1] M. Broy. Requirements engineering for embedded systems. In *Proc. of FemSys'97*, 1997.

[2] S. Donnay, K. Swings, G. Gielen, W. Sansen, W. Kruiskamp, and D. Leenaerts. A Methodology for Analog Design Automation in Mixed-Signal ASICs. In *The European Design Automation Conference (EURO-DAC)*, pages 530–534, Paris, France, February 1994.

[3] C. Grimm. *Hybride Datenflussgraphen und ihre Anwendung beim Entwurf analog/digitaler Systeme.* PhD Thesis, J. W. Goethe-Universität (submitted), 1999.

[4] C. Grimm and K. Waldschmidt. Repartitioning and technology-mapping of electronic hybrid systems. In *Design, Automation and Test in Europe '98 (DATE)*, Paris, France, February 1998.

[5] R. Grosu and T. Stauner. Modular and visual specification of hybrid systems – an introduction to HyCharts. Technical Report TUM-I9801, Technische Universität München, 1998.

[6] R. Grosu, T. Stauner, and M. Broy. A modular visual model for hybrid systems. In *Proc. of Formal Techniques in Real-Time and Fault-Tolerant Systems (FTRTFT'98)*, LNCS 1486. Springer-Verlag, 1998.

[7] Integrated Systems Inc. Integrated Systems: Products - BetterState. `http://www.isi.com/Products/BetterState/`, 1999.

[8] I. Kanakis, C. Grimm, P. Oehler, and K. Waldschmidt. Rapid Prototyping of Mixed-Signal Systems with KANDIS. In *Int. Workshop on Logic and Architecture Synthesis (IWLAS)*, September 1997.

[9] P. Oehler, C. Grimm, and K. Waldschmidt. KANDIS - A Tool for Construction of Mixed Analog/Digital Systems. In *European Design Automation Conference*, Brighton, UK, September 1995.

[10] B. Selic, G. Gullekson, and P. T. Ward. *Real-Time Object-Oriented Modeling.* John Wiley & Sons Ltd, Chichester, 1994.

[11] T. Stauner and C. Grimm. Prototyping of hybrid systems – from HyCharts to hybrid data-flow graphs. In *Proc. of Workshop on Distributed Systems (WDS'99)*, ENTCS 28, to appear. Elsevier Science, 1999.

[12] The MathWorks Inc. Stateflow. `http://www.mathworks.com/products/stateflow/`, 1999.