Modellaustausch zwischen Steuergeräte-Entwicklungstools auf Basis einheitlicher grafischer Blockbibliotheken

Achim Wohnhaas, debis Systemhaus und Rainer Moser, FKFS

Zusammenfassung

Die Zielsetzung im Projekt MSR-MEGMA ist die Erarbeitung eines Bibliothek-Standards zum grafischen Modellaustausch zwischen Steuergeräteentwicklungs-Tools.
MSR (Manufacturer Supplier Relationship) ist ein firmenübergreifendes Komitee zur Unterstützung gemeinsamer Entwicklungen zwischen Automobilherstellern und Zulieferern von Elektroniksystemen.
Es werden die verschiedenen Möglichkeiten des Modellaustauschs und der Modellkopplung durch Datenaustausch vorgestellt. Desweiteren wird die im MSR-MEGMA für den Entwicklungsprozeß von Steuergerätesoftware verfolgte Vorgehensweise des grafischen Modellaustauschs (MEGMA) auf Basis einer einheitlichen, grafischen Blockbibliothek im Detail erläutert. Außerdem werden technische Besonderheiten der betrachteten Entwicklungstools und deren Einflüsse auf die Konvertierbarkeit der Modelle aufgezeigt.
Der Fokus richtet sich dabei auf Funktionen und Strukturen im speziellen Umfeld der Funktionsentwicklung für Steuergeräte. Anhand eines einfachen Beispiels wird ein erster konzeptioneller Vorschlag zum Modellaustausch vorgestellt.

Ausgangssituation und Zielsetzung

Die Entwicklung elektronischer Funktionen in Kraftfahrzeugen nimmt einen zunehmend hohen Stellenwert in der Automobilentwicklung ein. Der Umfang, die Komplexität und der Vernetzungsgrad der Elektronik steigen sehr stark an. Zukünftige Fahrzeuge werden sich in steigendem Maße durch Softwarefunktionen von ihren Wettbewerbern abheben und herstellerspezifische Eigenschaften werden per Software ins Fahrzeug gebracht. Aus dieser Situation heraus wird der Einsatz rechnergestützter Methoden im Entwicklungsprozeß von Steuergerätesoftware unumgänglich. Eine Neuordnung der Zusammenarbeit zwischen Elektronik-Zulieferer und Hersteller ist erforderlich. Von den Entwicklungstools ist gefordert, daß sie diesen firmenübergreifend verteilten Entwicklungsprozeß sowohl von der technischen Seite der Schnittstellen als auch aus Sicht der Prozeßorganisation optimal unterstützen.
Im Arbeitskreis MSR-MEGMA soll die Schnittstellenproblematik zwischen den Tools für die Entwicklung von Steuergerätesoftware auf Basis einer einheitlichen Blockbibliothek gelöst werden.

Toolkopplung und Toolschnittstellen

Kopplungen und Schnittstellen zwischen Entwicklungstools sind eine große Forderung der Entwickler und eine ebenso große Herausforderung an die Toolhersteller. Die Notwendigkeit unterschiedliche Tools einzusetzen ergibt sich durch

- Unzulänglichkeiten der Tools den gesamten Entwicklungsprozeß durchgängig zu unterstützen,

- die notwendige Zusammenarbeit verschiedener Entwicklungsabteilungen mit ihren gewachsenen Strukturen (Tools, Know-How und Modelle),

- die gemeinsame Entwicklungsarbeit von Hersteller und Zulieferer mit unterschiedlichen Tools,

- den Wunsch für jede Entwicklungsaufgabe das geeignetste Tool einzusetzen.

Die Toolhersteller stehen diesen Wünschen und Forderungen aufgeschlossen gegenüber. Eine perfekte und lückenlose Schnittstellenlösung kann es schon wegen der teils sehr unterschiedlichen Arbeitsweise und den spezifischen Features der Tools nicht geben, durch die sich die Tools im Wettbewerb behaupten müssen.
Es gibt jedoch einige bereits funktionierende und einsatzgeprüfte Möglichkeiten für Toolkopplung und Modellaustausch, die im folgenden kurz dargestellt werden sollen. **Bild 1** stellt einige Möglichkeiten symbolisiert dar.

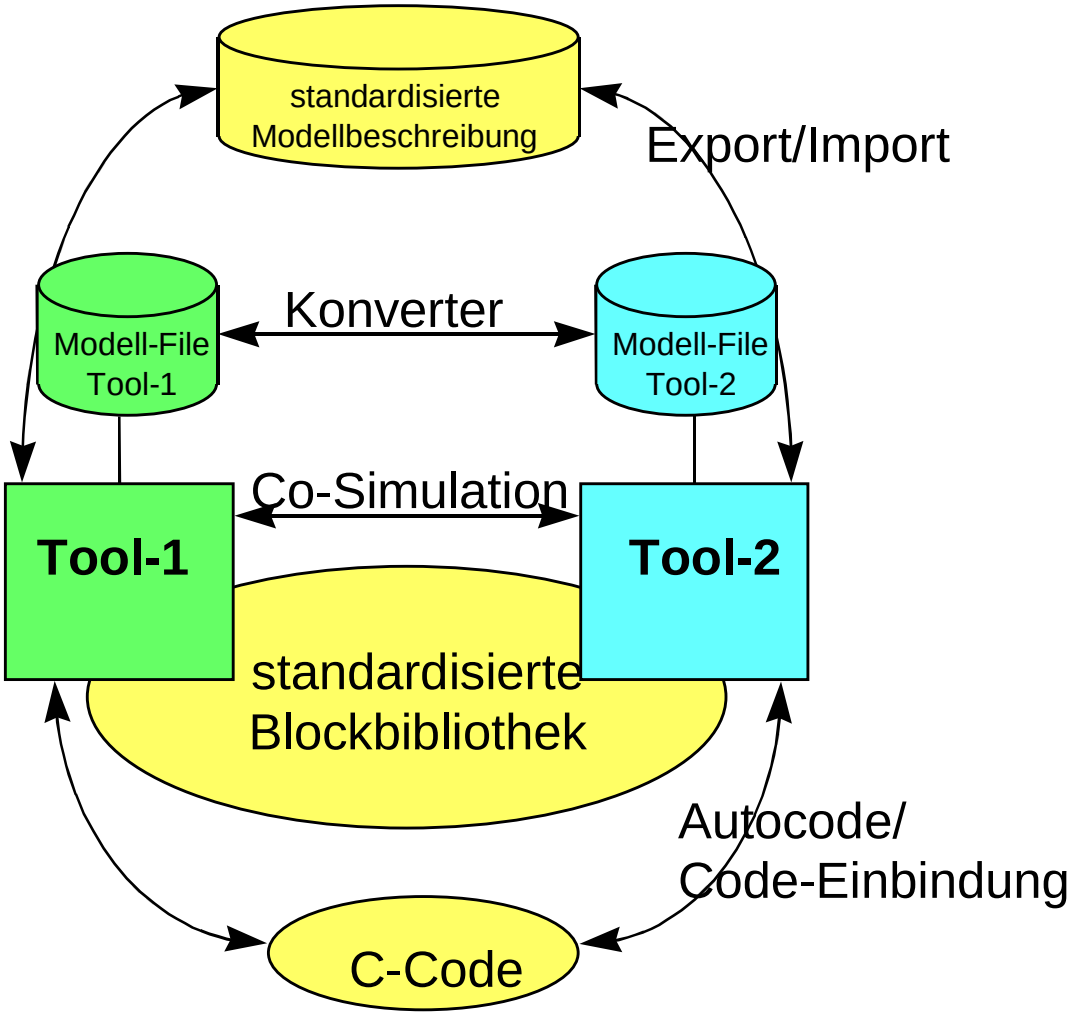


Bild 1: Konvertierungen und Schnittstellen in Entwicklungstools

Co-Simulation (Backplane)

Simulation eines über mehrere Plattformen verteilten Modells. Der Datenaustausch erfolgt beispielsweise über Shared Memory oder über ein Netzwerk. Es gibt sowohl von Drittanbietern angefertigte als auch bereits in den Tools als Feature integrierte Lösungen.

Kopplung mit Remote Function Calls

Aufruf von numerischen Funktionen oder numerischen Solvern in anderen Tools.

Code-Generierung und Code-Einbindung

Die in Frage kommenden Entwicklungstools bieten automatisierte Codegenerierung und Möglichkeiten zur Einbindung von Code als Block im Modell. Es können damit Funktionen, die im Quell-Tool entwickelt wurden, im Ziel-Tool eingesetzt werden. Änderungen oder Weiterentwicklungen der eingebundenen Funktion erfordern den zeitaufwendigen Umweg über das Quell-Tool mit neuer Code-Generierung.

Offline-Konverter

Konvertertools setzen auf den proprietären Modellbeschreibungssprachen der Tools auf und konvertieren eine Beschreibung in diejenige des Ziel-Tools. Für Tools deren Modellspeicherung in einem geschlossenen Format (binär oder in Datenbanken) erfolgt, kann dieser Konverter nur vom Toolhersteller selbst erstellt und verfügbar gemacht werden. Einige Konvertertools sind als „Unsupported Tools“ bereits verfügbar, decken jedoch nur einen eingeschränkten Funktionsumfang ab.

Standardisierte Beschreibungssprache (VHDL-AMS)

Die Tools unterstützen eine standardisierte Beschreibungssprache und bieten einen Import und Export in dieses Format, das die vollständige Beschreibung von Modell und Grafik/Netzliste enthält. Dies wurde bisher von den Toolherstellern nicht realisiert.
Probleme bereiten Funktionen und Strukturen die im Ziel-Tool nicht verfügbar sind. Diese müssen entweder von einem besonders intelligenten Importfilter oder in manueller Nacharbeit umgesetzt werden.

Modellaustausch über standardisierte Blockbibliothek

Modellaustausch auf Basis einer standardisierten Entwicklungsbibliothek mit entsprechenden Konvertierungsmöglichkeiten, die in den Tools integriert sind.
Diese Variante wird vom MSR-MEGMA verfolgt und ist im folgenden Abschnitt beschrieben.

Modellaustausch über eine standardisierte Blockbibliothek

Das Ziel im MSR-MEGMA Projekt ist die Spezifikation einer in allen Entwicklungstools implementierten, einheitlichen Blockbibliothek und der Modellaustausch auf Basis dieser Standardbibliothek. Diese Bibliothek und damit auch die Austauschbarkeit der Modelle beschränkt sich auf physikalische Steuergerätefunktionen (vgl **Bild 2**). Implementierungsinformationen (Abbildung der physikalischen Daten auf Datentypen der Steuergeräte), sowie Modelle der Regelstrecke werden nicht betrachtet.

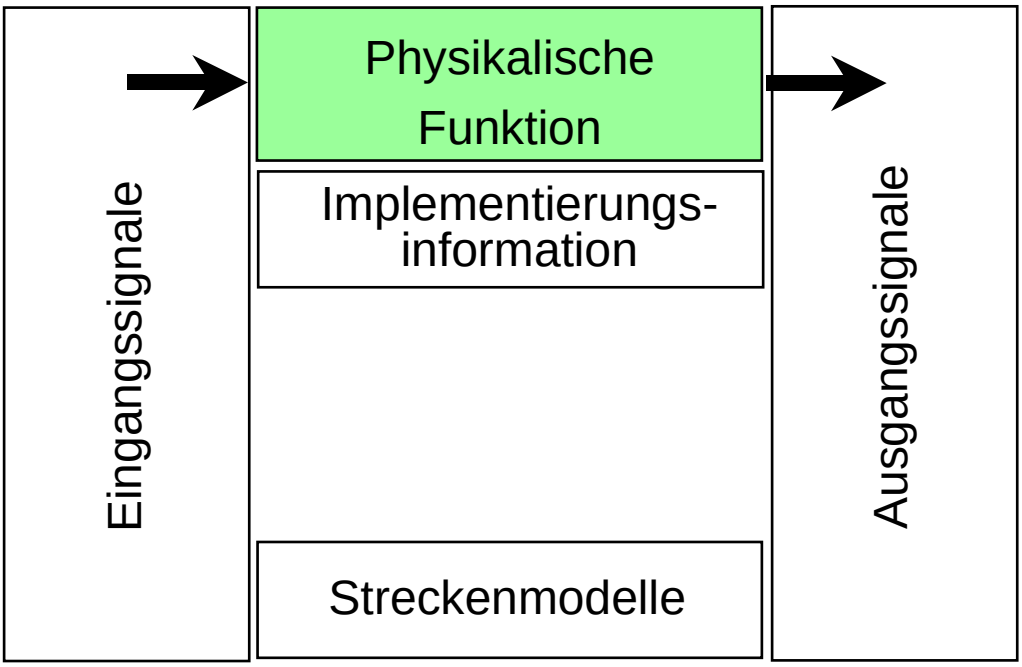


Bild 2: Standardbibliothek im MSR-MEGMA

Der Grundgedanke dieser Vorgehensweise ist, daß Funktionsblöcke mit verifiziert identischer Funktionalität und einheitlichen Schnittstellen einfach per Netzliste und Blockreferenzen in ein anderes Tool übertragbar sind.
Probleme bereiten beispielsweise spezielle Kontrollstrukturen und sonstige Besonderheiten und Features der Tools, auf die in einem späteren Abschnitt eingegangen wird.
Der kritische Leser fragt sich spätestens jetzt, weshalb nicht eine der bereits verfügbaren Methoden über ein spezielles Konvertierungstool eingesetzt und weiterentwickelt wird.

Ein wesentliches Argument sind die wohl speziellen Anforderungen der Elektronikentwicklung an die Entwicklungstools, die u.a. eine exakte funktionale Übereinstimmung nach einer Konvertierung voraussetzen. Auf einige dieser Besonderheiten wird im nachfolgenden Abschnitt eingegangen. Außer der besseren Konvertierbarkeit gibt es noch einige weitergehende Gründe für das genannte Vorgehen.

- Die in der Standardbibliothek spezifizierten Blöcke orientieren sich an den Erfordernissen im Kfz-Elektronikumfeld und werden ohnehin von den Entwicklern benötigt, sofern sie nicht in den Tools direkt verfügbar sind.

- Alle Entwickler arbeiten mit denselben Basisfunktionen. Heterogenitäten innerhalb einer Firma oder sogar einer Entwicklungsmannschaft werden vermieden.

- Aufbau und Struktur der Software werden dadurch zwangsläufig sehr ähnlich.

- Der Wiedererkennungswert der grafischen Repräsentation ist höher. Zusätzliche Modellierungsrichtlinien schaffen eine weitergehende Einheitlichkeit und verbessern diesen Effekt.

- Eine manuelle Konvertierung von Funktionen in andere Tools ist durch die grafische Ähnlichkeit praktikabel ohne Experte im Qell- und Ziel-Tool zu sein.

Die Standardbibliothek und ein Konzept zum Modellaustausch werden von einem Expertenteam erarbeitet, bestehend aus Vertretern der Firmen Robert Bosch GmbH, BMW AG, Daimler-Chrysler, Hella KG und Siemens AG. Die zunächst betrachteten Tools sind ASCET-SD, MATLAB/SIMULINK und MATRIX$_X$/SYSTEMBUILD.

Unterschiede und Besonderheiten der Tools

Die betrachteten Tools weisen einen hohen Überdeckungsgrad an ähnlichen Funktionen auf, der jedoch keine Aussage darüber zuläßt, wie groß die Unterschiede tatsächlich sind. Häufig sind die feinen Unterschiede auf den ersten Blick gar nicht sichtbar.

Modelleigenschaften

Die folgenden Modelleigenschaften sind bei der Entwicklung von SG-Software wesentlich:

- Es werden nur skalare Signale/Größen benötigt,
- SG-Software ist zeitdiskret,
- Hierarchien werden zur Strukturierung (grafisch und funktional) eingesetzt,
- Datentypisierung ist im späteren Steuergerätecode erforderlich,
- Teilmodelle mit mehreren Zeitrastern sind üblich,
- Zustandsautomaten zur Beschreibung reaktiver Systemteile sind erforderlich,

Im MSR-MEGMA wurde hierzu folgendes festgelegt:

Da Datentypen nicht in allen betrachteten Tools verfügbar sind, werden diese später gesondert und im Hinblick auf zukünftige Verfügbarkeit betrachtet.
Zustandsautomaten sind in den Tools ebenfalls sehr unterschiedlich gelöst und werden in einem Arbeitskreis gesondert untersucht.

Hierarchiebildung und Strukturierung

Im folgenden sind die verschiedenen Hierarchieformen der Tools und deren Eigenschaften aufgelistet:

- SYSTEMBUILD:
  - Hierarchielement *SuperBlock*: Zentrales Hierarchieelement in SYSTEMBUILD, das je nach Typ unterschiedliche Zeitattribute besitzt. Ein *SuperBlock* ist oberstes Element eines Modells.

Typ	Zeitattribute
Continuous	Zeitkontinuierliches Raster, vererbt Zeitraster an Subblöcke (Basisblöcke, andere *SuperBlocks*), instantiierbar
Discrete	Zeitdiskretes Raster, vererbt Zeitraster
Procedure	Erbt Zeitraster von *SuperBlock* bzw. *Condition Block*, erzeugt Funktion, referenzierbar (function-call)
Trigger	Externe Rastervorgabe (flankengesteuert)
Enable	Eigenes Zeitraster, Externe Aktivierung des *SuperBlocks* (wertgesteuert)

  - Hierarchieelement *Condition block*: Hierarchieelement für *Procedure-Superblocks*, die über Bedingungen angestoßen werden. Erbt das Zeitraster.

- ASCET-SD:
  - *Hierarchy*: Grafische Zusammenfassung von Blöcken ohne zeitliche oder funktionale Eigenschaften.
  - *Module*: Einfach instantiierbar mit verschiedenen diskreten Zeitrastern.
  - *Class*: Mehrfach instantiierbar mit verschiedenen diskreten Zeitrastern.

- SIMULINK:
  - *Subsystem*: Grafische Zusammenfassung von Basisblöcken oder anderen *Subsystems* ohne zeitliche oder funktionale Eigenschaften.

Kontrollstrukturen

Steuergerätesoftware läßt sich häufig mit einfachen Kontrollflußelementen wie Schalter (Switch) beschreiben. Komplexe Kontrollstrukturen wie sie in den Tools verfügbar sind, ermöglichen gegebenenfalls eine elegante oder effiziente Beschreibung, tragen aber nicht zur einfachen Austauschbarkeit und Transparenz der Modelle bei. Dabei handelt es sich meist um softwarenahe Konstrukte wie IF-THEN-ELSE oder SWITCH-CASE. Es sind jedoch gerade die Kontrollstrukturen, hinter denen sich eine Tool-Philosophie verbirgt über die sich ein Tool vom Wettbewerber abhebt. Der Entwickler, der sich ausschließlich in einem Tool bewegt, schätzt diese Vorteile und nutzt bzw. benötigt sie sogar zwingend, um bestimmte Funktionalität zu erreichen. Verbietet man den Einsatz dieser toolspezifischen Möglichkeiten wird es Akzeptanzprobleme bei den Entwicklern geben. Dieser Zielkonflikt ist durch Überzeugungsarbeit zu überwinden, da eine vollständig automatisierte Austauschbarkeit ohne gewisse Einschränkungen aus heutiger Sicht nicht möglich scheint.
Ein einfaches Beispiel zeigt **Bild 3**, in dem sich die Unterschiede in der Grafik zwischen SYSTEMBUILD und ASCET-SD offenbaren.

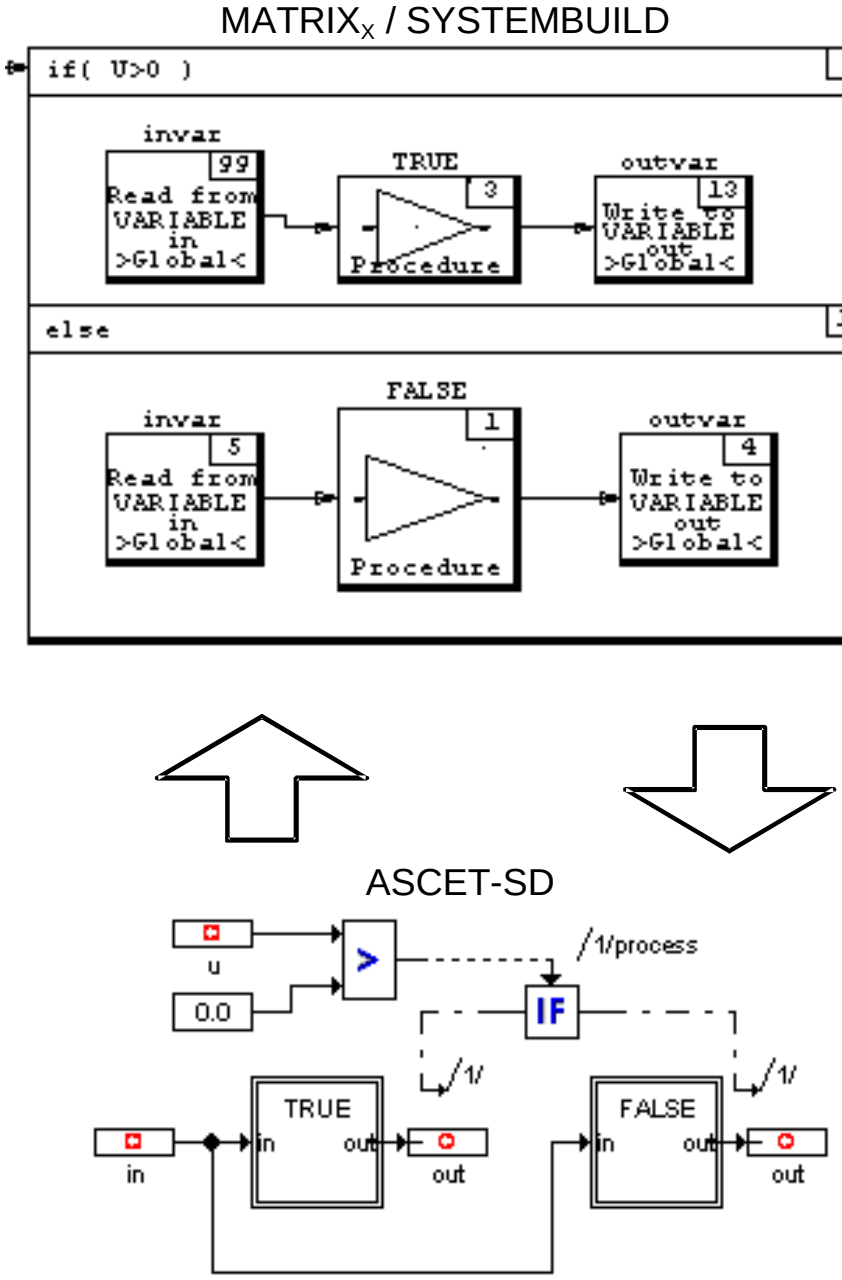


Bild 3: Grafische Unterschiede bei Kontrollstrukturen zwischen den verschiedenen Tools (If-Then-Else-Block).

Varianten und konfigurierbare Blöcke

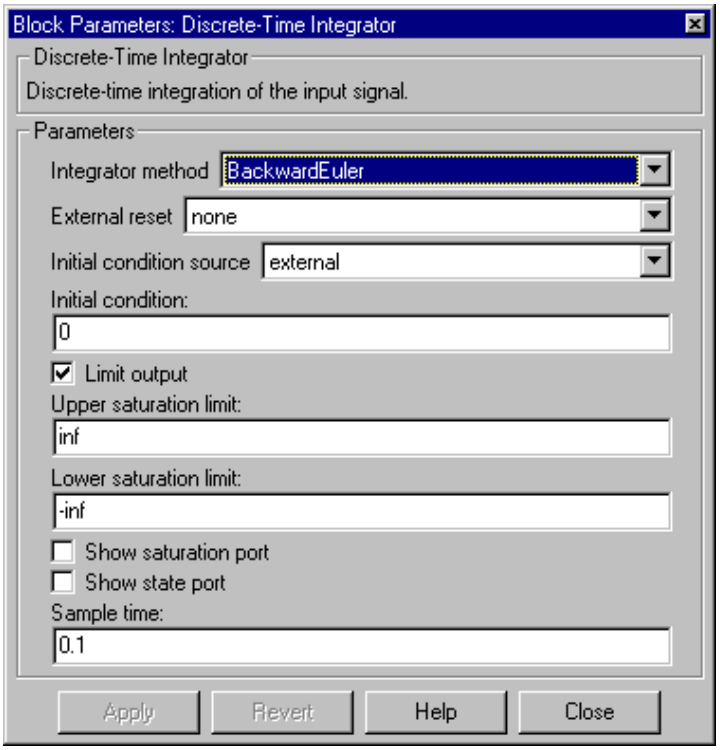
Es ist Tool-Philosophie ob überwiegend mit statischen oder konfigurierbaren Blökken gearbeitet wird. Existieren im einen Tool mehrere Varianten einer Funktion als statische Blöcke, so exisitiert im anderen Tool ein generischer Block der für jede Variante entsprechend konfigurierbar ist.

Signalfluß

Wesentlich für die Transparenz der Modelle ist die durchgängige Sichtbarkeit und Verfolgbarkeit der Signalflüsse im Modell. Insbesondere müssen in der SG-Software alle applizierbaren Parameter im Signalfluß und damit an der Schnittstelle zu Basis-

blöcken auftauchen. In SIMULINK beispielsweise arbeiten viele Blöcke mit internen Parametern, die in der Steuergeräte-Software als applizierbare Parameter benötigt werden
Diese Blöcke sollen durch entsprechend erweiterte Blöcke aus der Standardbibliothek ersetzt werden.

SIMULINK



ASCET-SD

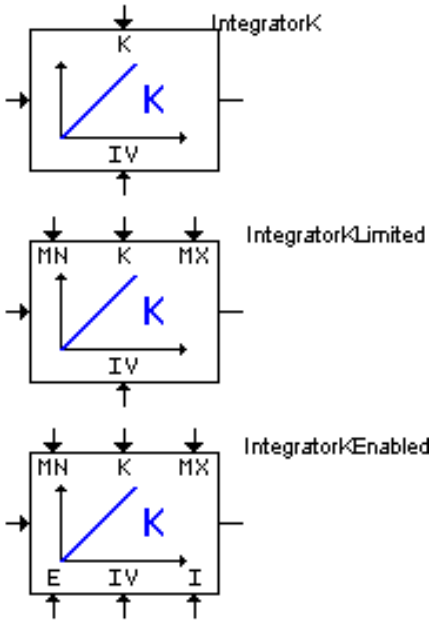


Bild 4: Abbildung der Varianten verschiedener Blöcke durch interne Parameter bzw. externe Signaleingänge (Simulink-Interface und Varianten in ASCET-SD).

Timing und Berechnungsreihenfolge

Im Timing zeigt sich die unterschiedliche Philosophie und die Flexibilität der Tools sehr deutlich. In SYSTEMBUILD wird das Zeitverhalten am SuperBlock festgemacht. In SIMULINK sind interne Parameter der Blockelemente für das Timing konfigurierbar und in ASCET-SD hat der Entwickler alle Freiheiten. Letzteres gilt insbesondere auch für die Festlegung der Berechnungsreihenfolge, die automatisiert datengetrieben oder manuell beliebig veränderbar sein kann. ASCET-SD arbeitet dabei von den Ausgängen zu den Eingängen rückwärts, d.h. ein Ausgang fordert die Berechnung aller Blöcke, die seinen Eingang beeinflussen an. Im Gegensatz dazu arbeitet SIMULINK datengetrieben (vom Eingangssignal zum Ausgangssignal) und bietet dem Anwender keine Eingriffsmöglichkeit in die Berechnungsreihenfolge. In SYSTEMBUILD steht dafür beispielsweise ein SequencerBar zur Verfügung, der eine grafisch sichtbare Trennung der einzelnen Berechnungsblöcke darstellt.
Im Detail ergeben sich Timing-Unterschiede durch unterschiedliche werkzeuginterne Berechnungsreihenfolgen (vgl. **Bild 5**). Zum Beispiel werden in SIMULINK und SYSTEMBUILD zuerst die Ausgänge aktualisiert, bevor die internen Zustände neu berechnet werden. Dies kann zu unterschiedlichen Simulationsergebnissen führen,

da in ASCET-SD diese Berechnungsreihenfolge vom Anwender angepaßt werden kann. Dies erschwert einen automatisierten Austausch der verschiedenen Modelle.

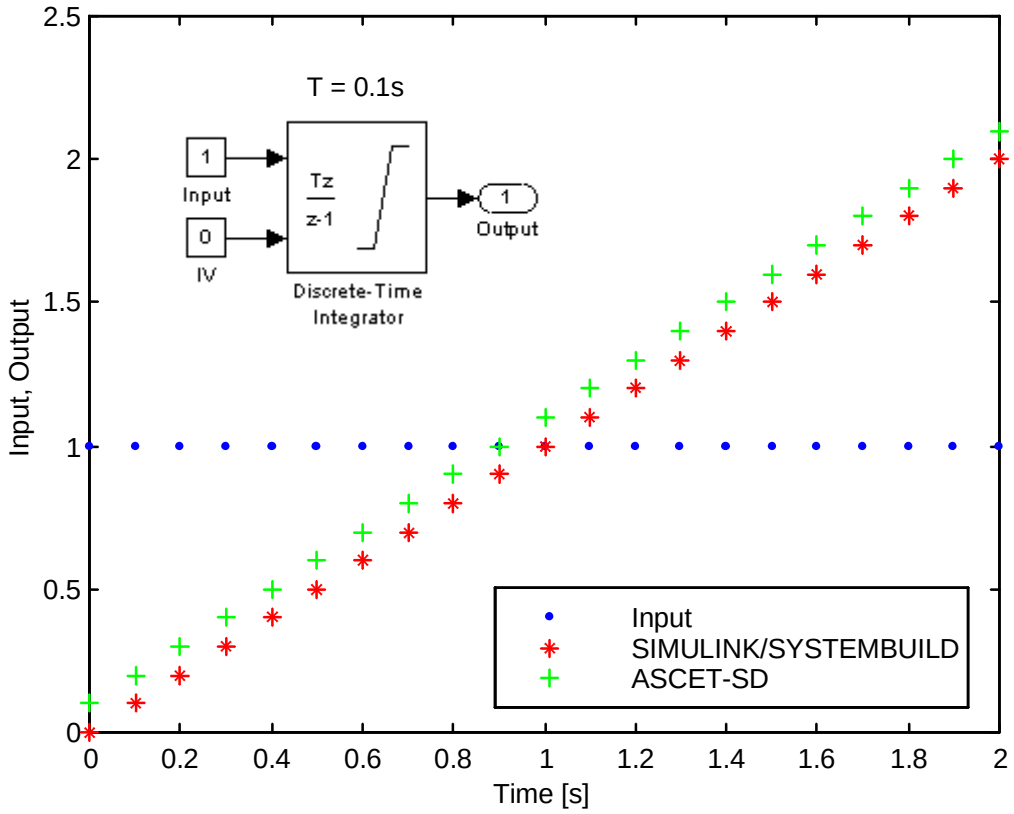


Bild 5: Timing-Unterschied beim Integrator aus den Standardbibliotheken der verschiedenen Werkzeuge.

Parameter

Aus Sicht der Steuergeräteentwicklung werden durch applizierbare Parameter unterschiedliche Steuergarätevarianten realisiert. Im Hinblick auf den Modellaustausch müssen diese Parameter als Teile des Modells übertragen werden. Hierbei sind Standardschnittstellen (z.B. SGML basiert) zwischen den Tools gefragt.
Funktionale Unterschiede bestehen hauptsächlich in der Art der Berechnung der Ausgänge (lineare Interpolation bzw. Tabellensuche) und dem Verhalten außerhalb des definierten Eingangswertebereichs (lineare Extrapolation, Begrenzung) von Kennlinien und Kennfeldern.

Konzept des MSR-MEGMA zum Modellaustausch

Anforderungen

Die Anforderungen an einen grafischen Modellaustausch wurden wie folgt festgelegt:

- Hin- und Rückkonvertierung ohne funktionale Verluste
  (Quell-Tool -> Ziel-Tool -> Quell-Tool)
- Unidirektionale Konvertierung ohne grafische Verluste bei Basisblöcken
  (Menge der unterstützten Basisblöcke wird von MSR-MEGMA festgelegt)
  Hohe Wiedererkennbarkeit der Funktion ist gewünscht.
- Einschränkungen unterstützter Kontrollstrukturen
  (Untermenge tooleigener Kontrollelemente/-strukturen werden von MSR-MEGMA phasenweise festgelegt und unterstützt)
  Es werden keine MSR-eigenen Kontrollelemente spezifiziert

Lösungsansatz Phase 1

Der automatisierte Modellaustausch stützt sich aus der oben genannten Motivation heraus auf eine Blockbibliothek, die alle benötigten Elemente (Basisblöcke) zur Beschreibung von Steuergerätefunktionen enthält. Die prinzipielle Funktionsweise ist wie folgt beschrieben:

- Eine Netzliste beschreibt die Lage und Verbindung der Elemente.
- Eine Abbildungsvorschrift bildet ein Element im Quell-Tool auf ein funktional identisches Element im Ziel-Tool ab.
- Unbekannte Blöcke werden als Dummy mit entsprechenden Schnittstellen konvertiert.
- Unterschiedliche Varianten der Elemente werden über ein Varianten-Mapping behandelt.
- Es gibt eine Vorschrift zur Nachbildung des Zeitverhaltens.
- Ein Verfahren deckt die Behandlung hierarchischer Strukturen ab.
- Zulässige Kontrollstrukturen sind in der ersten Phase nur Schalter (Switch, Mux). Komplexe Kontrollstrukturen müssen daher noch manuell übertragen werden.

Blockbibliothek

Eine Bibliothek der MSR-Automotive-Blöcke wurde vom MSR-MEGMA Expertenteam spezifiziert. Ziel ist die Abbildung eines Elements im Quell-Tool auf ein funktional identisches Element im Ziel-Tool.
Der Entwickler hat damit zukünftig in den betrachteten Tools eine MSR-Automotive Standardbibliothek zur Verfügung, mit der er seine Steuergerätefunktionen aufbauen kann. Ebenso kann er mit den standardmäßig im Tool verfügbaren Elementen arbeiten, sofern diese in der vom MSR-MEGMA festgelegten Menge für einen Modellaustausch zugelassen sind.

Der Arbeitskreis MSR-MEGMA empfiehlt die Herausgabe von Modellierungsrichtlinien, die der Entwickler beachten muß, um sowohl den Modellaustausch zu sichern als auch den Wiedererkennungseffekt zu steigern.

Ein Varianten-Mapping verringert die Anzahl der MSR-Blöcke und ermöglicht die Nutzung der tooleigenen Elemente (vgl. **Bild 6**).

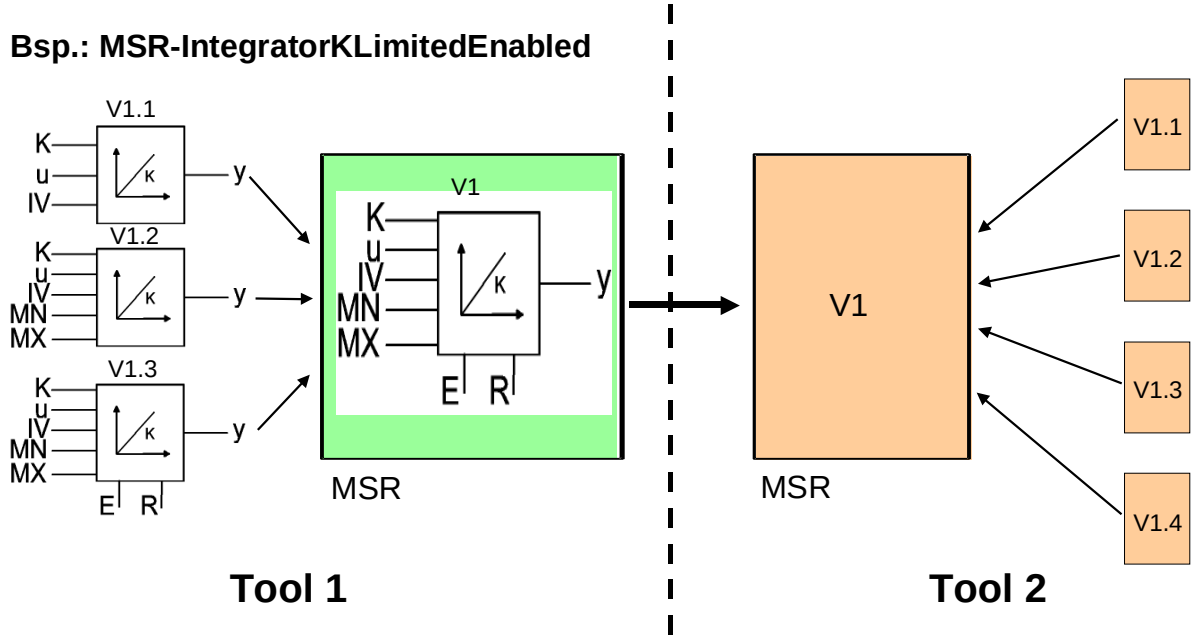


Bild 6: Variantenmapping beim Modellaustausch, am Beispiel von 3 Integratorvarianten.

Zeitverhalten und Hierarchien

Die oben erläuterten Unterschiede der Tools im Zeitverhalten sind eine Herausforderung an einen automatisierten Modellaustausch bei identischer Funktionalität.
Daneben verlangt die unterschiedliche Behandlung der Zeitraster (Abtastraster, sample time) in den einzelnen Tools auch teilweise eine andere Strukturierung des Modells (z.B. SYSTEMBUILD):

- In SYSTEMBUILD erfolgt eine Ordnung nach Zeitrastern (Superblock). Funktionen mit unterschiedlichen Zeitrastern liegen daher in getrennten Superblöcken. Externe Trigger sind ebenfalls möglich.

- In ASCET-SD sind Zeitraster beliebig an Ausgänge und speichernde Elemente gebunden. Rückwärtiges Verfolgen der Struktur bis zur nächsten (davor liegenden) Speicherzelle ergibt ein Raster und wird beispielsweise in SYSTEMBUILD in einen Superblock gewandelt.
- Zeitraster in SIMULINK sind an Eingangssignale und speichernde Elemente gebunden oder durch externe Trigger vorgegeben.

Die entsprechenden hierarchischen Strukturen der Tools werden aufeinander abgebildet. So ergeben sich aus *SuperBlocks* in SYSTEMBUILD *Subsystems* in SIMULINK und *grafische Hierarchien* in ASCET-SD. Die Verwendung von Klassen wurde in ASCET-SD ausgeschlossen.

Konvertierungsbeispiel Leerlaufregler

Die Konvertierungsregeln wurden manuell an einem Beispiel nachvollzogen. Das Ergebnis entsprach den Erwartungen. Es konnte in allen Werkzeugen funktionale Gleichheit erzielt werden. Die grafische Präsentation war in den Tools teilweise sehr ähnlich und entsprach dem erwünschten Wiedererkennungswert (**Bild 7 und 8**).

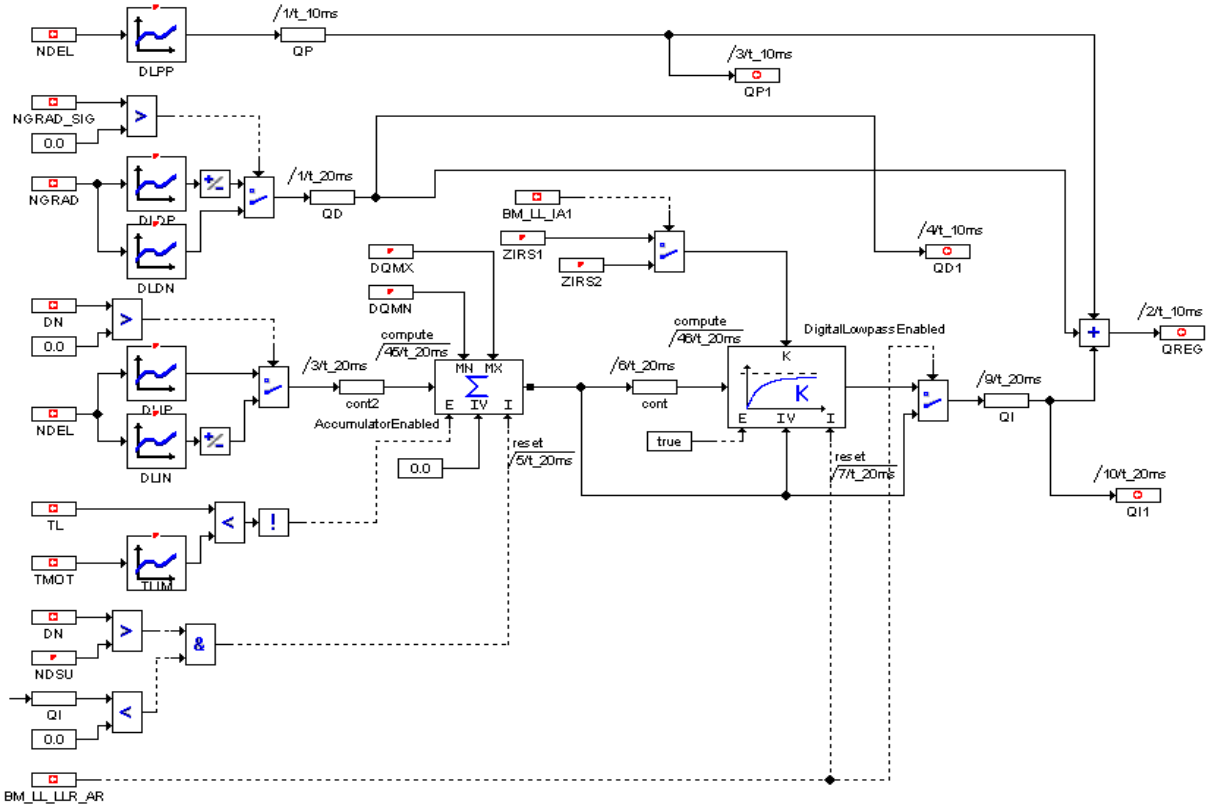


Bild 7: Leerlaufregler in ASCET-SD.

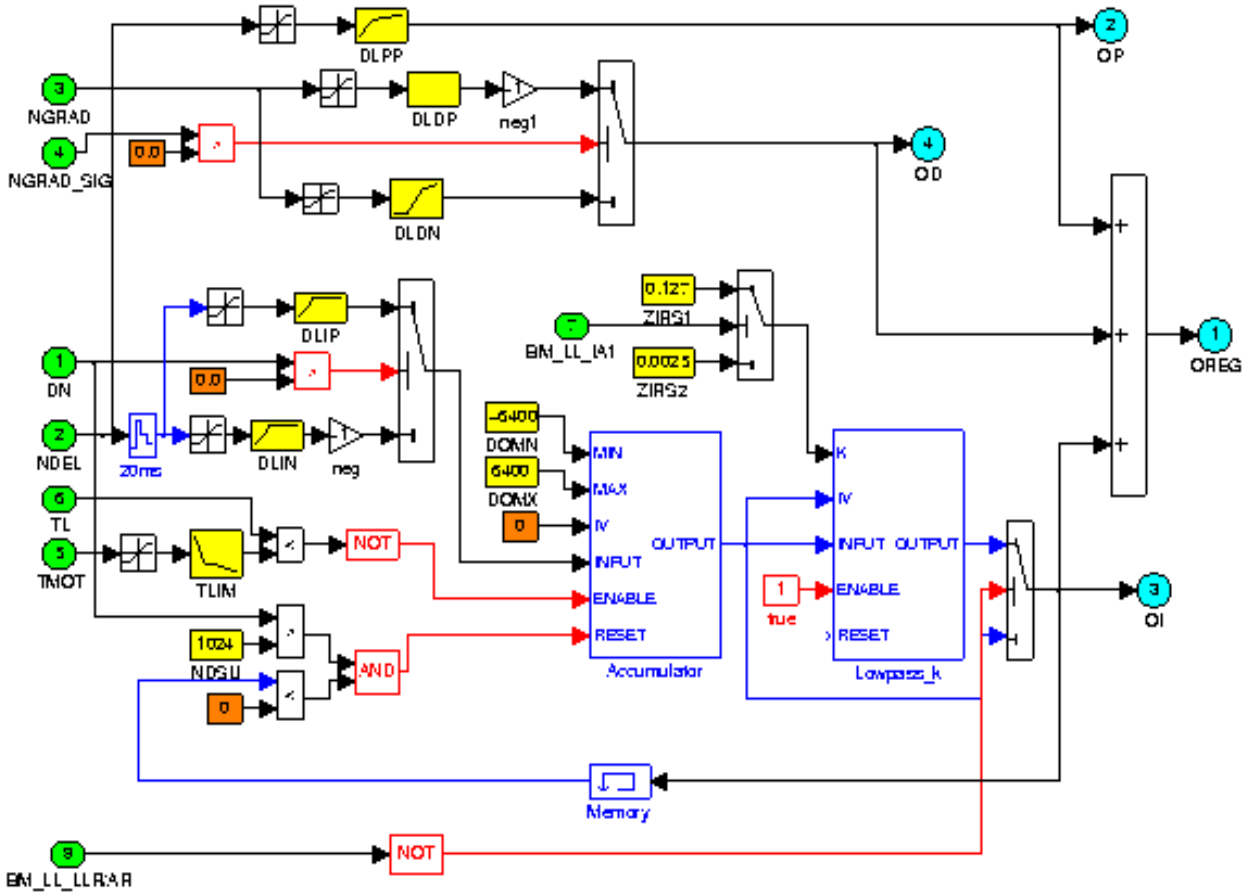


Bild 8: Leerlaufreglerbeispiel von ASCET-SD nach SIMULINK konvertiert.

Ausblick

Der Arbeitskreis MSR-MEGMA wird die Realisierung der ersten Phase des Modellaustausches weiter begleiten. Die Toolhersteller haben ihre Bereitschaft zur Unterstützung signalisiert. Es folgen die Schritte:

- Feinspezifikation der Bibliothekselemente (MSR-MEGMA und Toolhersteller)
- Implementierung der Bibliotheken in den Tools (z.B. Toolhersteller)
- Spezifikation eines Austauschformats (z.B. auf Basis von SGML)
- Einbau der Import/Export-Funktionen in den Tools (Toolhersteller)
- Erstellung von Modellierungsrichtlinien und Kochrezepten zur manuellen Konvertierung von Kontrollstrukturen (MSR-MEGMA)
- Erweiterung übertragbarer Kontrollstrukturen
- Berücksichtigung der Datentypen

Literaturverzeichnis

/1/ E. Perenthaler, B. Weichel, T. Hirth, P. Rauleder (1998): MSR-Standards in der Praxis; VDI-Berichte 1415, 1998.

/2/ http://www.msr-wg.de

/3/ SYSTEMBUILD User's Guide, Integrated Systems Inc., 3260 Jay Street, Santa Clara (CA), 1996.

/4/ ASCET-SD, User's Guide, ETAS GmbH&Co.KG, Borsigstraße 10, 70469 Stuttgart, 1996.

/5/ Using SIMULINK, The MathWorks, Inc., 24 Prime Park Way, Natick (MA), 1996.