

SystemC-AMS basierte Modellierung, Simulation und HiL Echtzeitsimulation für Anwendungen der Automobilelektronik

Karsten Einwich, Thomas Arndt; Fraunhofer IIS/EAS

karsten.einwich@eas.iis.fraunhofer.de, thomas.arndt@eas.iis.fraunhofer.de

Zusammenfassung

Elektronische Komponenten in der Automobiltechnik sind immer mehr mit den Komponenten anderer Domänen vernetzt. Damit werden deren Spezifikation und Entwurf zu einer wachsenden Herausforderung. Dieser Beitrag stellt eine auf SystemC/SystemC-AMS basierende Modellierungs- und Simulationsmethodik vor, mit deren Hilfe diesen neuen Anforderungen begegnet werden kann.

1 Einführung

Elektronik spielt in der Automobiltechnik eine immer größer werdende Rolle. Sie ermöglicht neben der Einführung neuer Sicherheits- und Komfortfunktionen die Reduzierung des Schadstoffausstoßes und des Energieverbrauchs sowie signifikante Kosteneinsparungen. Um diese Ziele zu erreichen, muss die elektronische Hard- und Software immer enger mit den nichtelektronischen (z.B. mechanischen) Komponenten verwoben werden. Da Elektronik und im speziellen Software im Produktionsprozess extrem kostengünstig sind, wird mehr und mehr versucht, teure mechanische bzw. hydraulische Komponenten zu vereinfachen und die dadurch entstehenden Nachteile durch Elektronik zu kompensieren. Speziell durch die Software wird zusätzlich eine hohe Flexibilität und Konfigurierbarkeit erreicht. Dies hat zur Konsequenz, dass im Entwicklungsprozess die einzelnen Komponenten immer weniger getrennt voneinander betrachtet werden können. Die Systemfunktionalität ergibt sich zunehmend aus einer immer engeren Interaktion von nicht elektrischen (z.B. mechanischen) und elektronischen Hard- und Softwarekomponenten. Diese enge Verknüpfung wird für den Entwurfsprozess zunehmend zu einer Herausforderung. Da speziell in der Automobiltechnik die Entwicklung des Gesamtsystems (Auto) über viele Firmen verteilt erfolgt, ist diese enge Interaktion schwer zu handhaben. Klassisch besteht die Zulieferkette vom Halbleiterhersteller (TIER2) über den Komponentenhersteller (TIER1) zum Autoproduzenten (OEM). Außerdem müssen verschiedene Fachgebiete zusammenarbeiten, wobei die Schnittstelle zwischen diesen immer mehr verwischt, d.h. immer schwerer klar zu spezifizieren ist. Damit gewinnen Simulations- und Modellierungstechniken, welche eine Gesamtsystembetrachtung und einen Austausch zwischen verschiedenen Fachgebieten und Firmen erlauben eine wachsende Bedeutung. Solche Techniken ermöglichen zum einen die gemeinsame Erarbeitung der Komponentenspezifikationen als auch die Entwicklung der Hard- und Software unter Berücksichtigung der Gesamtsystemfunktionalität. In diesem Beitrag möchten wir solch eine vielversprechende Methode vorstellen und die Möglichkeiten, die diese bietet diskutieren. Diese beruht auf der Hardwarebeschreibungssprache SystemC und deren Erweiterung zur Modellierung analogen Verhaltens SystemC-AMS.

2 SystemC / SystemC-AMS

SystemC [1] ist eine Hardwarebeschreibungssprache welche auf der Programmiersprache C++ aufbaut. Somit ist SystemC eine Definition von C++ Klassen und Funktionen welche die Beschreibung der Struktur und des Verhaltens von Hardwarekomponenten erlauben. SystemC wird von der Accellera Systems Initiative – einer Industrievereinigung – entwickelt und standardisiert. Somit ist SystemC auch ein IEEE Standard (IEEE 1666). Der Fokus von SystemC liegt in der Beschreibung von komplexen digitalen Hard-/Softwaresystemen auf höheren Abstraktionsebenen. SystemC-AMS [2,3,4] – auch standardisiert von Accellera – erweitert SystemC mit der Möglichkeit analoges und gemischt analog/digitales Verhalten auf hohen Abstraktionsebenen zu beschreiben. Ziel ist es die Beschreibung und Simulation des Gesamtsystems bestehend aus analoger und digitaler Hard- und Software in seiner Umgebung zu ermöglichen. Dabei sollen komplette Anwendungsszenarien mit sinnvollem Aufwand simulierbar sein. Damit ist der Hauptanwendungsbereich von SystemC/SystemC-AMS die Erstellung einer ausführbaren Spezifikation, der Architekturentwurf und die Bereitstellung eines Referenzmodells zur Stimulierung und Überprüfung von implementierten Komponenten. Um vor allem die erforderliche Simulationsgeschwindigkeit zu erreichen, setzt SystemC/SystemC-AMS auf das Konzept von verschiedenen sogenannten Berechnungsmodellen (Models of Computation - MoC). Dabei ist die Idee, dass die unterschiedlichen Systemkomponenten mit dem für sie besten Beschreibungsmittel hinsichtlich Aufwand, erforderlicher Genauigkeit und Simulationsgeschwindigkeit beschrieben werden und damit dann mit optimierten Algorithmen berechnet werden können. Damit lässt sich für Systemebenenbetrachtungen eine Beschleunigung der Simulation um mehrere Größenordnungen im Vergleich zu konventionellen Verhaltensbeschreibungssprachen erreichen. Dadurch wird eine funktionale Gesamtsystemsimulation von komplexen Anwendungsszenarien möglich. Für SystemC und SystemC-AMS wird eine Open Source „proof-of-concept“ Implementierung zur Verfügung gestellt, wobei die Lizenzbedingungen so gestaltet sind, das eine kommerzielle Nutzung möglich ist. Zusätzlich stehen zahlreiche kommerzielle Implementierungen zur Verfügung, welche sich an der Open Source Implementierung orientieren bzw. diese für die entsprechenden Werkzeuge anpassen um z.B. komfortable Debugmöglichkeiten oder die Integration in andere Simulatoren (z.B. VHDL oder Verilog) zu ermöglichen. Dadurch, dass SystemC und SystemC-AMS auf der weit verbreiteten Programmiersprache C++ beruhen, ergeben sich zahlreiche Möglichkeiten, die unter anderem für die Anwendung in der Automobilelektronik interessant sind. Im folgenden Abschnitt werden solche Anwendungsszenarien diskutiert.

3 Anwendungsszenarien

3.1 Ausführbare Spezifikation

Wie im Abschnitt 1 beschrieben, interagiert die elektronische Hard- und Software immer enger mit z.B. den mechanischen Komponenten. Eine unabhängige Spezifikation von Hard- und Software, bzw. Elektronik und nicht Elektronik ist immer schwerer umzusetzen. Die traditionelle Papier-basierte Spezifikation wird immer schwerer erstellbar, sie ist mehr und mehr unvollständig bzw. enthält Fehler welche oft erst während der Systemintegration feststellbar sind. Eine ausführbare Spezifikation ersetzt nicht die Papier basierte Spezifikation, wir aber mehr und mehr ein unentbehrliches Hilfsmittel zur Erstellung dieser. Sie erlaubt es das Zusammenspiel der Elektronik mit den anderen Komponenten schon im Spezifikationsprozess zu verifizieren. Damit lassen sich viele teure Spezifikationsfehler vermeiden und Missverständnisse wie sie z.B. zwischen unterschiedlichen Fachgebieten auf Grund unterschiedlicher Terminologie oder Definitionen immer wieder auftreten rechtzeitig ausräumen.

Damit eine ausführbare Spezifikation erfolgreich verwendet werden kann, muss es mit ihr möglich sein Anwendungsszenarien mit der Umgebung bzw. mit vereinfachten

Umgebungsmodellen in akzeptabler Zeit zu simulieren. Dabei muss die Funktionalität und das Zeitverhalten soweit es die Funktion beeinflusst korrekt abgebildet werden, dagegen ist die Berücksichtigung von parasitären Effekten im Allgemeinen nicht erforderlich. Da die Spezifikation als Kommunikationsplattform zwischen unterschiedlichen Gruppen genutzt werden soll, muss sie von diesen auch ohne spezialisierte Werkzeuge ausführbar bzw. in deren Werkzeuge integrierbar sein.

3.2 Architekturentwurf

Ähnlich den Anforderungen zur ausführbaren Spezifikation, müssen auch beim Architekturentwurf möglichst komplette Anwendungsszenarien simuliert werden um unterschiedliche Architekturvarianten bewerten und somit optimieren zu können. Zusätzlich muss es möglich sein, das Modell punktuell zu verfeinern um z.B. die Überlastung von Bussystemen oder die Verletzung von Echtzeitbedingungen zu detektieren.

3.3 Entwicklung eingebetteter Software

Fast alle aktuellen integrierten Komponenten enthalten mehr oder weniger komplexe Prozessoren und Controller. Diese übernehmen Aufgaben wie z.B. die Ablaufsteuerung, Kalibrierung, Regelung bis hin zur intelligenten Verarbeitung von Daten bzw. der Ansteuerung von Aktoren. Da die erforderliche eingebettete Software sehr eng mit der Hardware interagiert ist eine Entwicklung unabhängig von dieser nicht möglich. Da zudem die Softwareentwicklung einen immer erheblicher werdenden Zeitaufwand beansprucht, ist ein Start dieser nach Verfügbarkeit der Hardware inakzeptabel. Zusätzlich ist die Beobachtbarkeit interner Knoten in der Regel nicht ohne weiteres möglich. Aus diesen Gründen bringt der Softwareentwurf mit Hilfe einer virtuellen Plattform eine Reihe von Vorteilen. Voraussetzung dafür ist, dass das Modell schnell genug simuliert und sich in die Softwareentwurfsumgebung einbinden und debuggen lässt. Wobei für den Softwareentwurf bis auf den Prozessor häufig sehr vereinfachte Modelle verwendet werden können. Allerdings sind dafür die Anforderungen an die Simulationsgeschwindigkeit sehr hoch, da Softwareentwickler eine interaktive Arbeitsweise erwarten.

3.4 Referenzmodell

Bei der Implementierung von integrierten elektronischen Komponenten ist die Verifikation eine große Herausforderung. Zum einen ist es schwer relevante Stimuli für die Subkomponenten zu erzeugen und zum anderen ist es nicht einfach die Antwort der Komponente zu bewerten. Damit gestaltet sich speziell der Aufbau von Regressionstests, wie sie zur Sicherstellung der Entwurfsqualität dringend erforderlich sind sehr schwierig. Ein Ausweg, ist ein Referenzmodell, wie es z.B. eine ausführbare Spezifikation oder das Architekturmodell darstellt. Aktuelle Verifikationsmethoden wie z.B. UVM (Universal Verification Methodology) setzen das Vorhandensein einer Referenz voraus. Da die Modelle für die ausführbare Spezifikation schon während deren Erstellung überprüft wurden, ist es wünschenswert diese als auch die Verifikationsszenarien zur Stimulierung als auch als Referenz für die Komponentenimplementierung weiter zu verwenden. Dazu muss es möglich sein, diese Modelle in die entsprechenden Entwurfswerkzeuge und Simulatoren einzubinden.

3.5 Kundenmodell

Typisch für Systeme der Automobiltechnik ist, dass die Einzelkomponenten von verschiedenen Zulieferern entwickelt werden. So geht die typische Wertschöpfungskette vom Halbleiterhersteller (TIER2) zum Komponentenhersteller (TIER1) bis zum Automobilhersteller (OEM). Für die gemeinsame Erarbeitung der Spezifikation wird es immer wichtiger, Modelle auszutauschen und diese im jeweiligen Gesamtsystemkontext Anwendungsszenarien zu verifizieren. Damit ist es möglich virtuell eine Systemintegration vor dem festschreiben der Spezifikation und somit weit vor der Verfügbarkeit der Hardware vorzunehmen. Da ausreichend detaillierte Modelle in der Regel immer Implementierungsdetails und

Algorithmen enthalten, welche zur „Intellectual Property – IP“ zählen, ist aus kommerzieller Sicht, ein Modellaustausch zwischen unterschiedlichen Unternehmen nur akzeptabel, wenn die innere Struktur und die Algorithmen entsprechend geschützt sind, d.h. das Modell nicht mehr über den inneren Aufbau preisgibt, als es die später verfügbare Hardware tut. D.h., das Modell muss für den Kunden eine „Blackbox“ sein welches er nur wie die reale Hardware über die äußeren Klemmen ansteuern kann (bzw. bei dem er nur vom Zulieferer gewollte interne Knoten beobachten kann). Zusätzlich muss das Modell in die Simulationsumgebung des Kunden einbindbar sein und sollte keine zusätzlichen Lizenzen von anderen Simulationswerkzeugen erfordern.

4 Technische Lösungsansätze

4.1 Modellierungstechniken

Ein großer Vorteil von SystemC/SystemC-AMS ist die parallele Anwendbarkeit verschiedener Modellierungs- und Simulationstechniken und die Möglichkeit SystemC/SystemC-AMS um weitere zu erweitern. SystemC/SystemC-AMS ist in Schichten aufgebaut. SystemC hat ein sogenanntes generisches Model of Computation, d.h. aufbauend auf Basiselemente wie z.B. Prozesse und Ereignisse sind unterschiedliche Berechnungsmodelle und darauf aufbauend verschiedene Modellierungstechniken definiert. Eine wichtige Technik ist die sogenannte TLM (Transaction Level Modeling) [5] Modellierung. Diese wird zur Modellierung von digitaler Kommunikation – z.B. der Kommunikation eines Prozessors mit seinem Speicher oder Peripherie eingesetzt. Diese Kommunikation erfolgt in der Regel über ein Bussystem bestehend aus Daten-, Adressbus und Steuerleitungen. Klassisch wird dieses digitale Verhalten mittels ereignisgesteuerter Modellierung/Simulation (was eigentlich auch schon eine starke Abstraktion des eigentlich analogen elektrischen Verhaltens darstellt) abgebildet. Wenn dabei z.B. ein Zugriff des Prozessors auf seinen Speicher erfolgt, werden taktgesteuert die entsprechenden Steuer- Adress- und Datensignale angelegt. Während der ereignisgesteuerten Simulation wird für jeden Signalwechsel ein Ereignis erzeugt und bei jeder Taktflanke erfolgt ein sogenannter Kontextswitch, d.h. ein Wechsel von einem zu einem anderen. Bei der TLM Modellierung wird ein solcher Buszugriff auf einen Funktionsaufruf abgebildet – welcher einen Wert vom Speicher liest bzw. schreibt, d.h. den Wert einer Variable im Speichermodell liest bzw. setzt. Da solch einem Funktionsaufruf eine Zeit annotiert werden kann, lässt sich das Verhalten auch zeitlich sehr genau an das ereignisgesteuerte Modell anpassen. Da rechentechnisch die Realisierung eines Funktionsaufrufs um Größenordnungen weniger aufwendig ist, sind durch TLM Modellierung je nach Detaillierungsgrad Simulationsgeschwindigkeitsgewinne um den Faktor 10.000 möglich. Dabei ist, wenn das zeitliche Verhalten nur ungefähr abgebildet werden muss, der Modellierungsaufwand wesentlich geringer.

Ein ähnliches Konzept verfolgt SystemC-AMS zur Modellierung von analogem Verhalten. SystemC-AMS erweitert SystemC um Berechnungsmodelle, welche speziell zur abstrakten Modellierung von analog und gemischt analog-digitalen (mixed-signal) Verhalten geeignet sind. Dabei spielt die Datenflussmodellierung eine besondere Rolle. SystemC-AMS enthält dazu das sogenannte Timed Dataflow (TDF) Berechnungsmodell. Ähnlich wie bei TLM erlaubt dieses Berechnungsmodell die Ausnutzung der abstrakten Modellierung zur Steigerung der Simulationsgeschwindigkeit um Größenordnungen. Bei der Datenflusssimulation werden Sample vom Eingang gelesen prozessiert und die Ergebnissample auf den Ausgang geschrieben. Damit lässt sich sehr einfach z.B. das Verhalten analoger Signalverarbeitung bestehend aus Verstärker, Filter und A/D-Wandler abbilden. Ein Verstärker ist z.B. im einfachsten Fall die Multiplikation des Eingangswert mit einem Faktor und evtl. einer Begrenzung des Ausgangswertes. Der Ausgangswert wird dann z.B. im Filter weiterverarbeitet, wobei die Rückwirkung für die funktionale Modellierung vernachlässigbar ist. Solch abstraktes Verhalten lässt sich auch mit klassischen

Verhaltensbeschreibungssprachen wie Verilog-AMS/VHDL-AMS beschreiben – da diese dann aber auf einen allgemeinen nichtlinearen Gleichungslöser abgebildet werden müssen, ist die Simulation um Größenordnungen langsamer als eine Datenflusssimulation. Somit werden die Modellierungsmöglichkeiten soweit wie möglich eingeschränkt, um das Verhalten auf einen möglichst effizienten Berechnungsalgorithmus abbilden zu können. Durch diese Vorgehensweise ist eine sehr hohe Simulationsgeschwindigkeit erreichbar, welche um Größenordnungen über der von klassischen Werkzeugen liegen kann.

4.2 Softwaremodellierung

Ein großer Vorteil von SystemC ist die homogene Integration und die Möglichkeit Software auf verschiedenen Abstraktionsebenen zu modellieren. So kann Software über ein Prozessormodell – z.B. einem sogenannten Instruktionsetsimulator als Binary eingebunden werden. Speziell zum Entwurf der Algorithmen kann die Software auch direkt in das Modell kompiliert werden. Da SystemC auf C++ beruht, ist das ganz einfach möglich. Dabei können dann Zugriffe auf externe Komponenten auf TLM Transaktionen abgebildet werden, wodurch die Software dann mit dem Hardwaremodell kommuniziert. Um das zeitliche Verhalten annähernd abzubilden, kann einzelnen Codeabschnitten eine Zeit annotiert werden. Die Vorteile dieses Ansatzes sind die einfache Debugbarkeit und die extrem schnelle Simulation. Da das Softwaremodell über eine TLM Schnittstelle mit der Hardware kommuniziert, ist es möglich dieses durch den oben beschriebenen Instruktionsetsimulator zu ersetzen, wodurch dann das Zeitverhalten taktgenau abgebildet wird.

4.3 Werkzeugintegration

Viele Simulationswerkzeuge erlauben es Modelle über C-Schnittstellen einzubinden. Da SystemC/SystemC-AMS auf C++ (was ein Superset von C ist) beruht ist eine Einbindung einfach möglich. Dies ist speziell sehr einfach möglich, da eine Open Source und somit lizenzfreie Implementierung für SystemC und SystemC-AMS zur Verfügung steht. Die Einbindung erfolgt mit Hilfe einer Koppelbibliothek, welche die Synchronisation der Zeitachsen und die Konvertierung der Datentypen realisiert. Die SystemC/SystemC-AMS Modelle werden dann zusammen mit der Koppelbibliothek je nach Betriebssystem zu einem shared object oder einer dll gelinkt, welche dann dynamisch von dem entsprechenden Simulationswerkzeug geladen werden kann. Damit kann das Modell vorkompiliert weitergeben werden, womit der oben beschriebene IP-Schutz realisiert werden kann.

4.4 Hardware in the Loop Simulation

Wie bereits beschrieben, lässt sich mit SystemC/SystemC-AMS die Simulationsgeschwindigkeit sehr gut optimieren. Somit ist es für bestimmte Modelle sogar möglich, diese in Echtzeit zu simulieren. Somit werden sogenannte Hardware in the Loop Simulationen möglich [6]. Damit wird die sich im Entwurf befindliche Systemkomponente mit bereits existierender Hardware – z.B. der Systemumgebung verbunden. Somit kann das Systemverhalten schon in Echtzeit vor Verfügbarkeit der Hardware und auch schon in der Spezifikationsphase getestet werden. Es existieren verschiedene Hardwareplattformen zur Durchführung solcher Simulationen. So bietet z.B. die Firma dSpace Plattformen auf der Basis von Intel und PowerPC Prozessoren an. Die Programme für diese Plattformen werden mit nahezu vollständig C++ implementierten Compilern übersetzt. Da die Open Source Bibliotheken von SystemC/SystemC-AMS als Quelltext vorliegen, lassen diese sich so anpassen, dass sie für die Hardware in the Loop Plattformen compilierbar sind. Zusammen mit einer Koppelbibliothek, welche vor allem für die Synchronisation der simulierten Zeit mit der realen Zeit zuständig ist, lassen sich Modelle erzeugen, welche auf die Plattformen herunterladbar sind.

5 Werkzeugunterstützung

Der SystemC/SystemC-AMS Standard definiert nur die Klassen, Makros und Funktionen zur Beschreibung des Systemverhaltens. Die Open Sourcebibliotheken stellen eine Referenzimplementierung frei zur Verfügung. Damit ist es schon möglich Systeme zu beschreiben und zu simulieren. Allerdings passiert dies in diesem Fall nur textbasiert und ist somit sehr aufwendig und fehleranfällig. Um den Modellierungsprozess effizient zu gestalten, gibt es für SystemC und inzwischen auch für SystemC-AMS Werkzeuge z.B. [7,8], die z.B. eine grafische Eingabe (Schaltplaneditor), die Modellgenerierung und die Generierung von Testumgebungen erlauben. Außerdem enthalten solche Werkzeuge auch komfortable Debugmöglichkeiten für die verschiedenen Modellierungsmöglichkeiten (z.B. Debuggen von TLM Transaktionen, Waveform viewer mit leistungsfähigem Postprocessing). Solche Werkzeuge liefern auch Bibliotheken mit vorgefertigten Modellen, welche dann mit Hilfe des Schaltplaneditors verbunden werden können. Auch die Bibliotheken welche für die oben erwähnten Möglichkeiten wie die Werkzeugintegration und Hardware in the Loop Simulation notwendig sind werden zur Verfügung gestellt. Trotzdem generieren diese Werkzeuge letztlich C++ Quelltexte bzw. stellen C++ Bibliotheken zur Verfügung, so dass auch bei deren Verwendung die oben beschriebenen Möglichkeiten bis auf wenige Ausnahmen nicht eingeschränkt werden. Somit machen diese Werkzeuge die komplexen Modellierungs- und Simulationsmöglichkeiten auch für nicht Spezialisten nutzbar.

6 Zusammenfassung

Es wurde eine Modellierungs- und Simulationsmethodik vorgestellt, welche Lösungsansätze für Probleme welche bei der Spezifikation und dem Entwurf elektronischer Hardware-Softwarekomponenten der Automobiltechnik bestehen, bietet. Da immer mehr die Notwendigkeit besteht, dass Spezifikation und Entwurf disziplin- und firmenübergreifend erfolgen muss bietet die auf SystemC/SystemC-AMS basierende Methodik interessante Möglichkeiten. Dies sind z.B. angepasste Modellierungstechniken welche es erlauben Modelle so zu optimieren, dass eine Gesamtsystemsimulation kompletter Anwendungsszenarien möglich wird, die Einbindung von Modellen in verschiedene Simulationswerkzeuge, ein IP geschützter Modellaustausch, Hardwaremodelle für die Softwareentwicklung und die Weiternutzung der Modelle für die Echtzeit Hardware in the Loop Simulation. Um die recht komplexe Methodik auch für nicht Spezialisten anwendbar zu machen stehen inzwischen komfortable Werkzeuge zur Verfügung.

Literatur

- [1] "IEEE Standard for Standard SystemC Language Reference Manual." IEEE Std 1666 2011 (Revision of IEEE Std 1666 2005) (2012): 1–638.
- [2] "Standard SystemC AMS extensions Language Reference Manual", Open SystemC Initiative, March 8 2010
- [3] "Standard SystemC AMS extensions User's Guide", Open SystemC Initiative, March 8 2010
- [4] Einwich, K. "SystemC AMS for the design of complex analog mixed signal SoC's: Presentation held at edaWorkshop 2009, Dresden, Germany, May 26-28, 2009". 2009.
- [5] Ghenassia, Frank. "Transaction Level Modeling with SystemC: TLM Concepts and Applications for Embedded Systems." Boston, MA, 2005.
- [6] Voit, Florian, "Echtzeitsimulation von Electronic-System-Level-Modellen", ASIM 2011
- [7] www.systemc-ams.eas.iis.fraunhofer.de
- [8] <http://www.synopsys.com/Systems/ArchitectureDesign/pages/PlatformArchitect.aspx>