Ein abstraktes SystemC-Modell zur Analyse und Leistungsabschätzung des europäischen Zugsicherungssystems ETCS

Benjamin Beichler, Alexander Nitsch, Frank Golatowski, Christian Haubelt

Institut für Angewandte Mikroelektronik und Datentechnik
Universität Rostock
Richard Wagner Str. 31
18119 Rostock-Warnemünde
{benjamin.beichler; alexander.nitsch;
frank.golatowski; christian.haubelt} @uni-rostock.de

Abstract: In diesem Beitrag wird ein SystemC-Modell der Geschwindigkeits- und Abstandsüberwachung aus dem European Train Control System (ETCS) vorgestellt. Dieses Modell dient als Ausgangspunkt für die frühzeitige Abschätzung der Leistungsfähigkeit des Systems und die in der Berechnung entstehenden Datenmengen. Hierfür wurde eine neuartige Methode entwickelt, welche es bei minimalen Anpassungen am Anwendungsmodell erlaubt, schnell unterschiedliche Entwurfsalternativen zu explorieren. Dabei werden SystemC-Prozesse in Abhängigkeit von Scheduling-Entscheidungen gestartet, deren Kommunikationsverhalten aufgezeichnet und anschließend Ausführungs- und Kommunikationszeiten simuliert.

1 Einleitung

Der europäische Bahnsektor ist durch unterschiedliche Sicherungssysteme gekennzeichnet, was den grenzüberschreitenden Verkehr von Schienenfahrzeugen durch Inkompatibilitäten erschwert. Zur Verbesserung der Interoperabilität ist das European Train Control System (ETCS) als einheitliches europäisches Zugsicherungssystem entstanden. Neben den vorrangigen Zielen, die Kosten für den Betrieb und die Instandhaltung zu reduzieren sowie Interoperabilität zu schaffen, hat das ETCS zum Ziel, die Streckenkapazität und -geschwindigkeit weiter zu erhöhen. Dazu ist es notwendig, eine Vielzahl von Sensor-, Strecken- sowie Verkehrsdaten zu erfassen und zu verarbeiten. Als sicherheitskritisches Cyber-Physikalisches-System unterliegt das Zugsteuerungssystem strengen Echtzeitanforderungen, was eine sehr hohe Rechenleistung der Onboard-Unit voraussetzt. Die Anwendung von Multicore-Technologien bietet eine Möglichkeit, das weiter ansteigende Datenaufkommen durch parallele Datenverarbeitung zu beherrschen. Ein wesentlicher Bestandteil des ETCS-Standards ist die Geschwindigkeits- und Abstandsüberwachung. Für diese komplexe Komponente wird mit der Modellierungssprache SystemC ein abstraktes, ausführbares Systemmodell entwickelt, wodurch gleichzeitig dieser Teil der ETCS-Anforderungsspezifikation formalisiert wird. Dieser Ansatz unterscheidet sich von

bereits verfügbaren Modellen, da er Performance-Abschätzungen für Hardware-/Softwaresysteme des Zugüberwachungsrechners zu einem sehr frühen Zeitpunkt des Systementwurfs gestattet. Die entwickelte Methode erlaubt bei minimalen Anpassungen am Anwendungsmodell, schnell unterschiedliche Entwurfsalternativen zu explorieren. Dabei werden SystemC-Prozesse in Abhängigkeit von Scheduling-Entscheidungen gestartet, deren Kommunikationsverhalten aufgezeichnet und anschließend Ausführungs- und Kommunikationszeiten simuliert. Mit der Verwendung der Open-Source-Modellierungssprache SystemC sind die Ergebnisse dieser Arbeit in das BMBF/ITEA-Projekt openETCS eingeflossen. Das Projekt hat zum Ziel, Open-Source-Modelle und Werkzeuge für das Zugsicherungssystem ETCS zu entwickeln, um eine Langzeitverfügbarkeit und Wartbarkeit zukünftiger Systeme zu gewährleisten. Der Rest des Beitrages strukturiert sich wie folgt: Im zweiten Abschnitt werden verwandte Arbeiten diskutiert. Abschnitt drei behandelt die grundlegenden Zusammenhänge der Geschwindigkeits- und Abstandsüberwachung des ETCS-Standards bevor im vierten Teil die Performance-Evaluationsmethodik vorgestellt wird. Der fünfte Abschnitt dieser Publikation zeigt die Implementierungsdetails des Anwendungsmodells zur Geschwindigkeits- und Abstandsüberwachung. Abschnitt sechs stellt die gewonnen Ergebnisse der Performance-Evaluation des Anwendungsmodells vor. Eine Zusammenfassung erfolgt in Abschnitt sieben.

2 Stand der Technik

Seit der Verabschiedung des ETCS-Standards wurden mehrere Arbeiten veröffentlicht, in denen verschiedene Aspekte der ETCS-Spezifikation untersucht wurden. Viele der Arbeiten beschäftigen sich mit den Echtzeiteigenschaften und der Ausfallsicherheit der Kommunikationsverbindung zwischen Zug und streckenseitigen Einrichtungen. In [zHHS13, JzHS98, ZH05, HJU05] werden verschiedene Petri-Netz-Erweiterungen genutzt, um die funktionalen Eigenschaften sowie stochastische Sicherheiten der Kommunikation zu untersuchen. Die Modellierung und Berechnung der Abstands- und Geschwindigkeitsüberwachung von ETCS wurde in [BT11, Fri10] bearbeitet, wobei sich diese Arbeiten auf die funktionalen Eigenschaften der Berechnung konzentrieren und eine anwendungsspezifische Modellierungsmethodik verwenden. Im Rahmen dieser Publikation wird die in [ACC12] standardisierte Modellierungssprache SystemC verwendet, um einerseits die funktionalen Aspekte zu beschreiben und andererseits auch eine Analyse der Performance des Berechnungssystems vorzunehmen. Im Rahmen der Performance- und Energieverbrauchsanalyse wurden für SystemC-Modelle einige Methoden entwickelt. Auf dem abstrakten *Electronic System Level* (ESL) werden beispielsweise Simulations-Traces [WHL05] zur Abschätzung von Performance oder Leistungsaufnahme genutzt. Diese Arbeit nutzt einen ähnlichen Ansatz wie die in [SFH+06, SHT09] vorgestellten Virtual Processing Components (VPC). Im Gegensatz zu den genannten Publikationen wird die Modellierung des Anwendungsmodells nicht auf aktororientierten Modellen umgesetzt, sondern auf Anwendungsmodellen, die direkt SystemC benutzen. Um Performance-Analysen auf der abstrakten System-Ebene zuzulassen, wurden die Zeitannotationen mit wenigen Veränderungen am Anwendungsmodell umgesetzt. Außerdem wurde die Konformität zur Referenzimplementierung des SystemC-Simulators beibehalten. Der Ansatz aus

[ASH+11] zeigt einige Ähnlichkeiten zur vorgestellten Methode, in dem ein zyklenapproximiertes Verfahren mit TLM-Modellen vorgestellt wird. Die Kalibrierung des Modells wird in [ASH+11] sehr feingranular durch eine Analyse des Quellcodes mit Hilfe eines Compilers vorgenommen, was allerdings eine Neukompilierung nach jeder Änderung der Konfiguration nach sich zieht. Die Methodik in diesem Beitrag arbeitet auf einem abstrakteren Niveau und benötigt keine Neukompilierung.

3 Geschwindigkeits- und Abstandsüberwachung im ETCS

Auf Grund der niedrigen Haftreibung zwischen Rad und Gleis benötigt das Bremsen eines Schienenfahrzeuges besondere Aufmerksamkeit. Im Vergleich zu einem PKW hat ein Zug eine viel geringere Bremsverzögerung und damit einen sehr langen Bremsweg, welcher teilweise im Kilometerbereich liegt. Da Gefahrenstellen deshalb nicht einsehbar sind, wird der Triebfahrzeugführer vom ETCS unterstützt, welches die Überwachung der genauen Geschwindigkeit und Position eines Zuges sicherstellt. Dadurch werden zum einen Fahrinformationen und zum anderen sicherheitskritische Bremsbefehle generiert, um eine größtmögliche Fahrsicherheit zu gewährleisten. Das Kapitel 3.13. *„Geschwindigkeits- und Abstandsüberwachung“* der Anforderungsspezifikation des ETCS-Standards [UNI12] beschreibt die dazu notwendigen Funktionseinheiten und Berechnungen.

Abbildung 1: Geschwindigkeits- und Abstandsüberwachung im ETCS, Funktionseinheiten: Acceleration due to Gradient **(AdtG)**, Calculation of Deceleration **(CoD)**, Calculation of Brake Build Up Time **(CoBBUT)**, Most restrictive Speed Profile **(MrSP)**, Determination of Supervised Targets **(DoST)**, Determination of Brake Deceleration Curves **(DoBDC)**, Supervision Limits **(SL)**, Speed and Distance Monitoring Commands **(SaDMC)**; die Datenabhängigkeiten sind in Abbildung 4 dargestellt

Wie in Abbildung 1 zu sehen, werden für die Berechnungen zugseitige und streckenseitige Eingangsparameter sowie Odometriedaten (momentane Geschwindigkeit/Position) benötigt. Die zugseitigen Eingaben sind jene Daten, welche der Zug lokal zur Verfügung stellt. Diese beinhalten im Wesentlichen: Bremsmodelle, Status der Bremsen, Korrekturfaktoren für Bremsmodelle, zugspezifische Parameter, das Traktionsmodell und feste Werte. Für die Berechnung des Bremsweges muss die Leistungsfähigkeit der Bremsen vorliegen. Ihre mathematische Entsprechung kann zum einen in Treppenfunktionen und zum anderen in den branchenüblichen Bremshundertstel dargestellt sein. Der Standard sieht vier verschiedene Bremsarten vor. Neben den grundsätzlich immer vorhandenen pneumatischen Bremsen können zusätzlich noch Nutzbremsen, Wirbelstrombremsen oder Magnetschienenbremsen vorhanden sein, die in

unterschiedlichen Kombinationen zur Anwendung kommen. Die konkrete Bremskombination hängt darüber hinaus vom aktuellen Zustand der Bremse ab. Ein weiterer wichtiger Aspekt ist die Berücksichtigung der Zeitspanne, die vergeht, bis eine Bremse ihre vollständige Bremskraft entfaltet. Im Betrieb können die Bremsmodelle drei unterschiedlichen Verwendungszwecken dienen. Im Notfall wird die vollständig verfügbare Bremsleistung berechnet, was den sicheren Betrieb eines Zuges garantiert. In den anderen beiden Fällen werden nicht alle Eingangsparameter und Korrekturfaktoren verwendet, weil es sich nicht um sicherheitskritische Berechnungen handelt. Korrekturfaktoren für Bremsmodelle sind notwendig, weil sich das Bremsverhalten je nach Streckenzustand (trocken, nass oder vereist) verändert. Die Bremsbeschleunigung unterliegt einer statistischen Streuung, welche vom Zughersteller über empirische Testszenarien ermittelt und mit einer statistischen Sicherheit angegeben wird. Zusätzlich sind die Zuglänge, die maximale Geschwindigkeit und Achslast sowie die Zugkategorie von großer Bedeutung. Die Zugkategorie bestimmt in diesem Fall, welche Geschwindigkeitsrestriktion zur Anwendung kommt. Das Traktionsmodell beschreibt den Einfluss der Traktionsbeschleunigung des Motors auf die Bremswirkung eines Triebfahrzeugs, wenn eine automatische Zwangsbremsung erfolgt, da die Traktion nicht beliebig schnell deaktiviert werden kann. In der ETCS-Spezifikation werden die erforderlichen Algorithmen beschrieben. Für die Algorithmen sind verschiedene Konstanten, sogenannte *„Fixed Values“* festgelegt. Beispielsweise wird dort die nominale Reaktionszeit des Triebfahrzeugführers festgeschrieben.

Die streckenseitigen Eingangsinformationen sind der Fahrbefehl, streckenseitige und andere Geschwindigkeitsbeschränkungen, das Höhenprofil und sonstige Streckenbeschaffenheiten sowie streckenspezifische Vorgabewerte. Der sogenannte Fahrbefehl, engl. „Movement Authority“, ist die Erlaubnis eines Zuges, sich in einem bestimmten Streckenabschnitt bewegen zu dürfen. Aus der Beschaffenheit der Schiene und der Umgebung ergeben sich verschiedene Geschwindigkeitseinschränkungen. Diese lassen sich in statische und dynamische Beschränkungen einteilen und werden von den streckenseitigen Einrichtungen an den Zug übermittelt. Die statischen Geschwindigkeitsrestriktionen ergeben sich aus der Geometrie des Streckenabschnitts, die dynamischen lassen sich auf vorübergehende Unsicherheiten auf der Strecke, der möglichen Zugkategorie sowie einer spezifischen Achslast zurückführen. Im Höhenprofil sind Informationen über Gefälle und Steigungen der Strecke enthalten. Neben den Streckeneigenschaften, aus denen Geschwindigkeitsbeschränkungen resultieren, gibt es noch andere Streckenbeschaffenheiten, die den Zugbetrieb direkt beeinflussen. Es können sich Einschränkungen für die Nutzung von Spezialbremsen ergeben. Darüber hinaus können weitere Meldungen existieren, wie zum Beispiel Hinweise auf verringerte Haftung in gewissen Streckenabschnitten. Streckenspezifische Vorgabewerte regeln im Allgemeinen die Standard-Werte für verschiedene Berechnungen oder Einstellungen und können für jede Strecke unterschiedlich definiert sein.

Für die Berechnung der Bremsbefehle und Führerstandsanzeigen sind im ETCS-Standard verschiedene Module beschrieben (Abbildung 1), die im Folgenden kurz dargestellt werden. Das Modul *„Acceleration due to Gradient“ (AdtG)* errechnet die durch ein Höhenprofil der Strecke verursachte Beschleunigung des Zuges. Notwendige

Operationen in dem Modul sind die Berechnung einer Zuglängenkompensation und die Berechnung der Beschleunigung anhand der Zugmasse, wobei die Rotationsmasse der Räder eine wichtige Rolle spielen. Das Modul *„Calculation of Deceleration“ (CoD)* fast die Bremsmodelldaten für die Notbremse und die verschiedenen Einflüsse auf die Beschleunigungsfunktionen zu einer Funktion zusammen, woraus sich eine positionsabhängige Bremsbeschleunigungsfunktion ergibt. Die Zeitspanne, die die Bremsen benötigen, um ihre vollständige Bremskraft zu entfalten, wird in dem Model *„Calculation of Brake Build Up Time (CoBBUT)* realisiert. Auf Grund der Vielzahl unterschiedlicher Geschwindigkeitsbeschränkungen besteht die Notwendigkeit, zu jeder Position die jeweils restriktivste Höchstgeschwindigkeit auszugeben. Diese Funktionalität wird im Modul *„Most restrictive Speed Profile“ (MrSP)* erzielt. Die Geschwindigkeits- und Abstandsüberwachung muss simultan verschiedene Ziele, sogenannte *„Targets“* überwachen. Das Modul *„Determination of Supervised Targets“ (DoST)* unterhält eine Liste dieser Ziele mit einer genauen Position und der dazu assoziierten Geschwindigkeit. Bremskurven sind in der Geschwindigkeits- und Abstandsüberwachung sehr wichtige Elemente. Sie ermöglichen die Vorhersage des Bremsweges in Abhängigkeit von einer bestimmten Geschwindigkeit und werden im Modul *„Determination of Brake Deceleration Curves“ (DoBDC)* berechnet. Die im Fokus dieser Arbeit stehende Bremskurve ist die *„Emergency Brake Deceleration Curve“ (EBD)*. Sie repräsentiert die Bremskurve im Falle einer Notbremsung. Die Informationen aus allen vorhergehenden Funktionaleinheiten werden im Modul *„Supervision Limits“ (SL)* genutzt, um die nötigen Daten zum Einleiten von Maßnahmen der Geschwindigkeits- und Abstandsüberwachung zu errechnen, wofür im Standard verschiedene Überwachungsgrenzen definiert sind. Als letzte Instanz der Geschwindigkeits- und Abstandsüberwachung berechnet das Modul *„Speed and Distance Monitoring Commands“ (SaDMC)}* konkrete Zugbefehle, wie automatische Bremsungen, aber auch die Anzeigen für den Triebfahrzeugführer. Im Wesentlichen werden die überwachten Ziele mit der aktuellen Position und Geschwindigkeit verglichen und anschließend geeignete Maßnahmen eingeleitet.

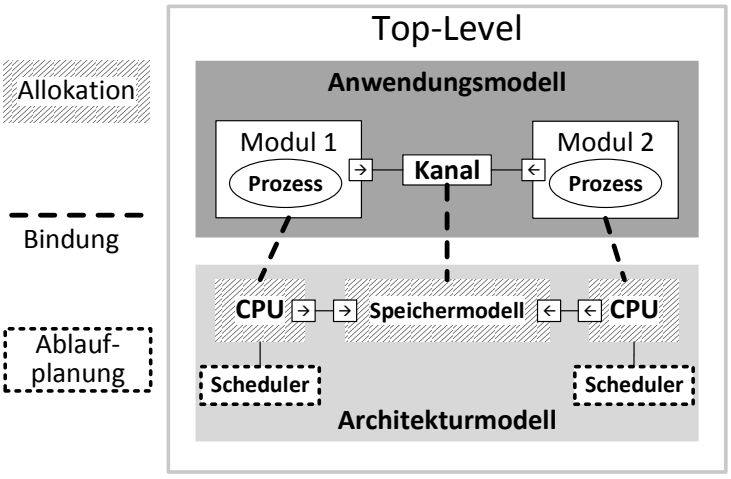

Abbildung 2: Mapping des Anwendungsmodells auf ein Architekturmodell

4 Performance-Evaluation

Im Entwicklungsprozess von eingebetteten Systemen ist es wichtig, schon frühzeitig Abschätzungen der Leistungsfähigkeit des gesamten Systems durchführen zu können. Dies ist insbesondere nötig, um eine Bewertung der getroffenen Entwurfsentscheidungen vornehmen zu können. Da die Entscheidungen auf Systemebene den größten potentiellen Einfluss auf die spätere Leistungsfähigkeit haben, ist es das Ziel, mit der vorgestellten Methode Aussagen über die Leistungsfähigkeit bereits auf dieser Ebene treffen zu können. Die Modellierungssprache SystemC hat sich als de-facto Standard für die Beschreibung und Simulation von Hardware-/Software-Systemen entwickelt. Durch die Flexibilität von SystemC in der Verwendung auf verschiedensten Abstraktionsebenen eignet es sich besonders auch für abstrakte Systembeschreibungen.

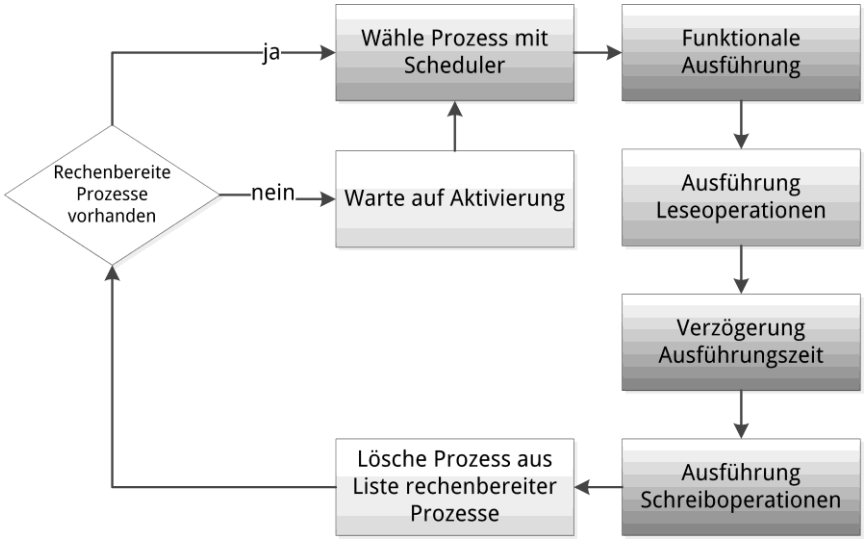

Abbildung 3: Simulationsschleife zur Performance-Evaluation

Für die Performance-Analyse eines Anwendungsmodells wird ein separates abstraktes Architekturmodell entwickelt, welches die Aspekte der Berechnungs- und Kommunikationsdauer simuliert. Die Abbildung 2 zeigt eine solche Kombination aus Anwendungs- und Architekturmodell. Dadurch können die Entwurfsentscheidungen der Allokation, Bindung, und Ablaufplanung einzelner Komponenten modelliert werden. Die Entwurfsentscheidungen werden dabei auf verschiedene programmatische Aspekte abgebildet:

- Allokation: Erzeugung von Objekt-Instanzen
- Bindung: Aufruf von Objekt-Methoden auf den Architektur-komponenten
- Ablaufplanung: Erzeugung eines Scheduler-Objektes, welches aus der Liste rechenbereiter Prozesse den auszuführenden Prozess auswählt

Nach der Elaboration dieses Top-Level-Modells wird während der Simulation innerhalb jeder abstrakten Recheneinheit die in Abbildung 3 gezeigte Schleife ausgeführt. Die ursprünglichen verhaltensbeschreibenden SystemC-Prozesse werden nicht direkt vom SystemC-Simulatorkern ausgeführt, sondern innerhalb der Recheneinheiten. Nach der Aktivierung eines SystemC-Prozesses des Anwendungsmodells durch ein Eingangssignal wird dieser nicht direkt ausgeführt, sondern bis zur geplanten Ausführung der Berechnungseinheit verzögert. Nach der Auswahl durch den Scheduler erfolgt die funktionale Ausführung des Anwendungsprozesses, in welcher gleichzeitig die Kommunikationsvorgänge des Anwendungsprozesses aufgezeichnet werden. Eine Blockierung des Prozesses unterstützt die derzeitige Implementierung nicht, weshalb das vorgestellte Verfahren momentan lediglich `SC_METHOD`s, die über `sc_signal`s kommunizieren, als Anwendungsmodell unterstützt. Während der funktionalen Ausführung schreitet die Simulationszeit nicht voran. Nach dieser funktionalen Phase werden die einzelnen Phasen des Prozesses zeitlich annotiert. In der ersten Phase werden alle Lesevorgänge simuliert, indem die aufgezeichneten lesenden Kommunikationsvorgänge in Blocking-TLM-Transaktionen übersetzt und innerhalb des Prozesses der Berechnungseinheit ausgeführt werden. Darauf folgt die Simulation der Ausführungszeit des Anwendungsprozesses, die im Wesentlichen einen `wait()`-Aufruf enthält. Im Anschluss werden die aufgezeichneten schreibenden Kommunikationsvorgänge in Blocking-TLM-Transaktionen übersetzt und ausgeführt. Erst innerhalb dieser Phase werden die erzeugten Ergebnisse, welche auf die Kommunikationskanäle geschrieben wurden, sichtbar und können weitere Prozesse aktivieren. Die dafür nötigen Änderungen am Anwendungsmodell sind so gestaltet, dass einerseits möglichst viele Freiheiten in der Benutzung von SystemC gestattet sind und andererseits möglichst wenige Änderungen für die Performance-Evaluation im Anwendungsmodell nötig sind.

```
...
SC_CTOR(test_module){
SC_METHOD_ANNOTATED(compute,10,SC_NS);
Sensitive << in1;
...
```

Listing 1: Änderung im Anwendungsmodell

Die beiden wesentlichen Änderungen sind die Änderungen der Kommunikationskanäle zwischen den Modulen und die Deklaration der verhaltensbeschreibenden Prozesse. Die Ersetzung der Kommunikationskanäle findet im Top-Level des Modells statt und ist durch die klare Trennung zwischen Interface und Implementierung dieser Komponenten einfach vorzunehmen. Durch die neuen Kanäle ist das Aufzeichnen der Kommunikationsaufrufe leicht möglich. Die Änderung der Deklaration von Prozessen in Anwendungsmodulen ist im Listing 1 zu sehen. Neben der Änderung des Makronamens wird als zusätzlicher Parameter eine Zeitkonstante übergeben. Diese wird als Ausführungszeit dieses Prozesses genutzt. Neben einer statischen Zeitangabe ist prinzipiell auch die Übergabe eines `sc_time`-Objektes möglich, um eine variable Ausführungszeit zu ermöglichen. Dieses Makro erlaubt dadurch eine gesteuerte Ausführung der Prozesse. Offensichtlich kann es bei diesem Ansatz zu Ungenauigkeiten in der Simulation des Zeitverhaltens kommen, da Datenwerten in der funktionalen Ausführung früher gelesen werden, als dies in einer echten Implementierung der Fall wäre, in welcher die Daten erst nach der Ausführung der lesenden TLM-Transaktionen

vorliegen würden. Diese zeitliche Entkopplung zwischen funktionaler und zeitlicher Simulation lässt aber eine schnelle und einfache Analyse der dynamischen Kommunikationseinflüsse der Komponenten zu. Die weitere angestrebte Verwendung von non-blocking-TLM bei gleichzeitiger Verwendung von *„temporal decoupling“* kann das Konzept noch sinnvoll erweitern.

5 Anwendungsmodell für die Geschwindigkeits- und Abstandsüberwachung im ETCS

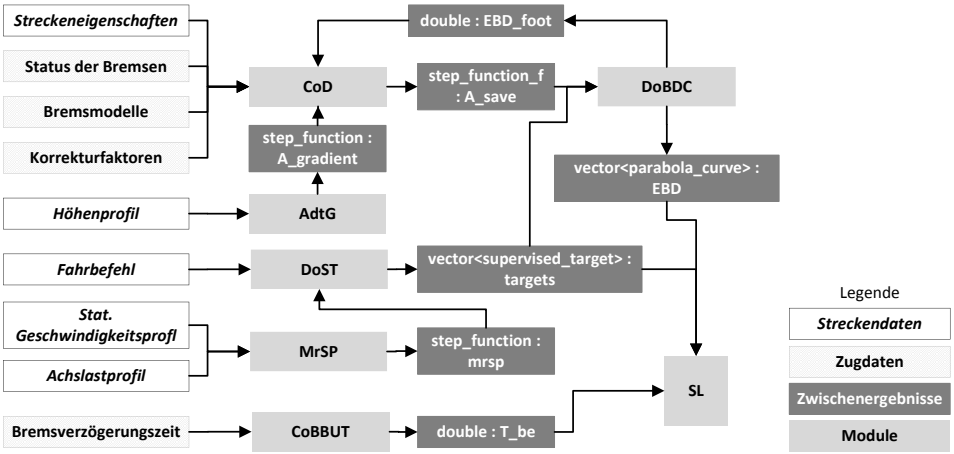

Abbildung 4: Übersicht der SystemC-Module

Dieser Abschnitt beschreibt den konkreten Aufbau und die Umsetzung des entwickelten SystemC-Modells. Die Abbildung 4 zeigt die im Rahmen dieses Beitrags implementierten Module sowie deren Datenabhängigkeiten. Die Implementierung der Module orientiert sich stark an der ETCS-Spezifikation [UNI12]. Alle vorgegebenen Partitionierungen, Benennungen, und Datentypen der Komponenten des Anwendungssystems werden aus der Spezifikation übernommen.

Es ergibt sich außerdem der Bedarf an speziellen Datentypen für die Berechnung in der Geschwindigkeits- und Abstandsüberwachung. Dies sind Repräsentationen für abschnittsweise definierte Funktionen. Für die spezielle Anwendung werden drei Typen von Funktionen benötigt und innerhalb von C++-Klassen implementiert:

- abschnittsweise definierte konstante Funktionen (Treppenfunktion) mit einer unabhängigen Variable: `step_function`
- abschnittsweise definierte konstante Funktionen (Treppenfunktion) mit zwei unabhängigen Variablen: `step_function_f`
- abschnittsweise definierte parabelförmige Funktion: `parabola_curve`

Im Wesentlichen werden alle diese Datenstrukturen mit dem assoziativen Container `std::map` umgesetzt. Als Schlüssel werden die Funktionsargumente des Beginns eines Abschnittes genutzt. Die gespeicherten Daten sind die Funktionswerte. Für Treppenfunktionen mit zwei unabhängigen Variablen werden als Funktionswerte Treppenfunktionen mit einer unabhängigen Variable gespeichert. Die Klasse `parabola_curve` speichert für jeden Abschnitt die Werte s_{begin}, v_{begin} und a.

Die Funktionswerte eines Abschnitts werden anhand des Geschwindigkeits-Weg-Gesetzes der gleichmäßig beschleunigten Bewegung berechnet, also mit der Formel:

$$v = \sqrt{2 \cdot a \cdot (s - s_{begin}) + v_{begin}^2}$$

Eine der Herausforderungen der Algorithmen der Geschwindigkeits- und Abstandsüberwachungen ist die Verarbeitung dieser Datentypen. In vielen Fällen ist es nötig, über Teile des Definitionsbereiches der Funktionen zu iterieren, weshalb verschiedene Hilfsfunktionen für diese Aufgaben implementiert wurden.

Es wurden darüber hinaus folgende generelle Entscheidungen für die SystemC-Modellierung des gesamten Anwendungsmodells getroffen:

- alle Prozesse innerhalb des Modells werden als Method-Prozesse modelliert
- alle Verbindungen zwischen den Modulen des Modells werden mit Signalen realisiert
- das System wird nur über die Dateneingänge aktiviert

Mit diesen Entscheidungen modelliert dieses Anwendungsmodell ein reaktives System, welches permanent auf Eingangswerte reagiert und keinen gemeinsamen Takt besitzt. Das Verhalten des Systems wird nur mit SystemC-Method-Prozessen realisiert, um die Datentransformationen zu beschreiben und effektiv zu simulieren. Diese Vorgehensweise erlaubt eine kompakte abstrakte Beschreibung des Systems, welche aber keine zeitliche Annotation der Vorgänge vornimmt.

6 Testergebnisse

Anhand der in dieser Publikation vorgestellten Methodik zur Performance-Evaluierung wurden mit dem Anwendungsmodell zur Geschwindigkeits- und Abstandsüberwachung von ETCS verschiedene Messungen durchgeführt. In Abbildung 5 sind die drei Szenarien mit Allokations-, Bindungs-, und Ablaufplanungsentscheidungen zu sehen. In allen Architekturen wurden ein FIFO-Scheduler sowie ein zentraler Speicher verwendet. Es wurde nur die Anzahl der Berechnungseinheiten und die nötige Bindung der Aufgaben an die Berechnungseinheit variiert.

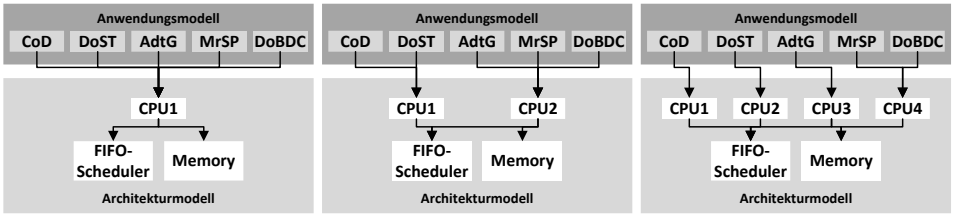

Abbildung 5: Getestete Architekturen für das Anwendungsmodell

Für die Kalibrierung des Modells wurde das Anwendungsmodell mit der Laufzeitanalyse-Programmsammlung Valgrind untersucht, welche in [NS07] vorgestellt wird. Die dadurch erreichte grobe Schätzung der benötigten Maschinenzyklen wurde mit einer Maschinenzyklusdauer von 1 ns kombiniert, welches mit einem Testdatensatz die Simulationsergebnisse in Tabelle 1 erzeugt hat. Es wurde ein zentraler Speicher mit einer einfachen Implementierung des TLM-Blocking-Interfaces genutzt, welcher aber durch beliebige andere TLM-Modelle ersetzt werden kann.

Anzahl CPU	Ausführungszeit [ms]	Deltazyklen
1	11,027	273
2	10,813	298
4	14,686	303

Tabelle 1: Ergebnisse der Performance-Evaluation

Die Tabelle 1 zeigt die ermittelten Werte für einen kompletten Berechnungszyklus der Geschwindigkeits- und Abstandsüberwachung. Durch den erhöhten Kommunikations- und Simulationsaufwand mit mehreren Recheneinheiten steigt die Anzahl der Deltazyklen mit steigender Anzahl der Berechnungseinheiten. Im Gegensatz zur möglicherweise vermuteten Verkürzung der simulierten Berechnungszeit mit mehreren Berechnungseinheiten steigt die Berechnungszeit mit 4 Berechnungseinheiten im Vergleich zur Verwendung einer Berechnungseinheit. In diesem konkreten Beispiel lassen sich diese Werte durch den FIFO-Scheduler erklären, welcher keine Datenabhängigkeiten beachtet, sodass einige Berechnungen unnötigerweise mehrfach ausgeführt werden müssen. Für messbare Ergebnisse der Simulationsperformance wurde die gleiche Berechnung zyklisch bis zu einer Simulationszeit von 100 Sekunden ausgeführt. Auf einem Linux-System mit 2,26 GHz Intel Core2Duo Prozessor beendet das reine Anwendungsmodell die Simulation nach 11,22 s, wohingegen die Simulation mit Performance-Evaluation nach 42,87 s endet. Diese zu erwartende Verlängerung entsteht im Wesentlichen durch die hinzukommenden Kontext-Wechsel der Prozesse in den Berechnungseinheiten sowie den zusätzlich auszuführenden Programmteilen der Performance-Evaluation, die innerhalb des reinen Anwendungsmodells durch die Verwendung der `SC_METHOD`-Prozesse entfallen.

7 Zusammenfassung

Im Rahmen dieser Arbeit konnte ein Modell für die Berechnung der Geschwindigkeits- und Abstandsüberwachung erstellt werden, wodurch eine abstrakte und ausführbare Darstellung der ETCS-Spezifikation entstanden ist. Auf der Grundlage dieses SystemC-Modells ist es möglich, durch die erfolgte Analyse der Spezifikation Ungenauigkeiten und potentielle Fehlerquellen zu identifizieren. Die durchgeführten Tests haben gezeigt, dass dieser Ansatz Performance-Abschätzungen für Hardware-/Softwaresysteme des Zugüberwachungsrechners zu einem sehr frühen Zeitpunkt des Systementwurfs ermöglicht. Obwohl die momentane Implementierung der Performance-Evaluation noch Einschränkungen aufweist, erlaubt die entwickelte Methode bei minimalen Anpassungen am Anwendungsmodell, schnell unterschiedliche Entwurfsalternativen zu explorieren.

Danksagung

Diese Arbeit wurde teilweise vom BMBF im Rahmen des ITEA-Projekts "openETCS" unter dem Kennzeichen 01IS12021K gefördert.

Literaturverzeichnis

[ACC12] Design Automation, Standards Committee und Ieee Computer: *IEEE Std 1666-2011, IEEE Standard for Standard SystemC® Language Reference Manual*, Band 2011. 2012, ISBN 9780738168012.

[ASH+11] S. Abdi, G. Schirner, Yonghyun Hwang, D.D. Gajski und Lochi Yu: *Automatic TLM Generation for Early Validation of Multicore Systems*. Design Test of Computers, IEEE, 28(3):10–19, May 2011, ISSN 0740-7475.

[BT11] B.Vincze und G. Tarnai: *Development and analysis of train brake curve calculation methods with complex simulation.* Advances in Electrical and Electronic Engineering, 5(1-2):174–177, 2011.

[Fri10] B. Friman: *An algorithm for braking curve calculations in ERTMS train protection systems*. Advanced Train Control Systems, Seite 65, 2010.

[HJU05] H. Hermanns, D.N. Jansen und Y.S. Usenko: *A comparative reliability analysis of ETCS train radio communications*, February 2005. AVACS Technical Report No. 2.

[JzHS98] L. Jansen, M. M. zu Hörste und E. Schnieder: *Technical issues in modelling the European Train Control System*. Proceedings of the workshop on practical use of coloured Petri Nets and Design /CPN 1998, Seiten 103–115, 1998.

[NS07] N. Nethercote und J. Seward: *Valgrind: a framework for heavyweight dynamic binary instrumentation*. ACM Sigplan Notices, 2007

[SFH+06] M. Streubühr, J. Falk, Ch. Haubelt, J. Teich, R. Dorsch und T. Schlipf: *Task-accurate performance modeling in SystemC for real-time multi-processor architectures*. In: Design, Automation and Test in Europe, 2006. DATE '06. Proceedings, Band 1, Seiten 2 pp.-, 2006.

[SHT09] M. Streubühr, Ch. Haubelt und J. Teich: *System level performance simulation for heterogeneous multi-processor architectures*. Workshop on Rapid Simulation and Performance Evaluation: Methods and Tools (Rapido), 2009.

[UNI12] UNISIG: SUBSET-026 – *System Requirements Specification*. SRS 3.3.0, ERA, 2012.

[WHL05] T. Wild, A. Herkersdorf und G. Y. Lee: *TAPES—Trace-based architecture performance evaluation with SystemC*. Design Automation for Embedded Systems, 10(2-3):157–179, September 2005, ISSN 0929-5585.

[ZH05] A. Zimmermann und G. Hommel: *Towards modeling and evaluation of ETCS real-time communication and operation*. Journal of Systems and Software, 77(1):47–54, 2005.

[zHHS13] M.M. zu Hörste, H. Hungar und E. Schnieder: *Modelling Functionality of Train ControlSystems using Petri Nets*. Towards a Formal Methods Body of Knowledge for Railway Control and Safety Systems, Seite 46, 2013.