Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Ausarbeitung Anwendungen 2 SS2011

Florian Bartols

Restbussimulation von Time-Triggered Ethernet in Automotive Anwendungen: Verwandte Arbeiten

Fakultät Technik und Informatik
Department Informatik

Florian Bartols

Thema der Ausarbeitung Anwendungen 2
SS2011
Restbussimulation von Time-Triggered Ethernet in Automotive Anwendungen: Vergleichbare Arbeiten

Stichworte
Echtzeit Ethernet, TTEthernet, Restbussimulation, Bussysteme, Automotive Anwendungen

Kurzzusammenfassung
Die zunehmende Komplexität automobiler Netzwerke und deren hohe Bandbreitenanforderungen erfordert neue Konzepte in der Vernetzung. TTEthernet ist ein geeigneter Kandidat diese Problemstellung zu lösen. Um Frühzeitig im Entwicklungsprozess Steuergeräte zu testen hat sich die Restbussimulation als adäquates Mittel etabliert und wird im Projekt für das TTEthernet-Protokoll erarbeitet. Diese Arbeit gibt einen Überblick über verwandte Projekte und Arbeiten und führt Gemeinsamkeiten und Unterschiede auf.

Title of the paper
Clustersimulation of Time-Triggered Ethernet in context of Automotive Engineering: Related Work

Keywords
Real-time Ethernet, TTEthernet, Cluster simulation, Bussystems, Automotive Applications

Abstract
The increasing complexity of automotive networks and their high bandwidth requirements will demand new concepts in networking. TTEthernet is a suitable candidate to solve this problem. In order to allow testing of control units in an early point in the development process, cluster simulation has been established as an adequate manner and therefore it is developed for the TTEthernet protocol in this project. This paper gives an overview of related works and projects and lists similarities and differences.

Inhaltsverzeichnis

1 Einleitung

Moderne Kraftfahrzeuge sind mit ihrem hohen Anteil an elektronischen Komponenten mittlerweile zu sehr komplexen, verteilten Systemen angewachsen. Sie verfügen teilweise über 70 Steuergeräte, die miteinander über das Bordnetzwerk vernetzt sind und tauschen ca. 2500 verschiedene Nachrichten aus (vgl. Marscholik und Subke, 2007; Navet u. a., 2005). Diese Nachrichten haben unterschiedliche Anwendungsfälle und verschiedene Anforderungen in der Bandbreite, Übertragungssicherheit und dem Echtzeitverhalten, sodass sich für jede Anwendung ein eigenes Bussystem etabliert hat (vgl. Zimmermann und Schmidgall, 2011). Die aus diesem Grund resultierende hohe Komplexität ist mit den aktuellen Bussystemen eigentlich nicht mehr beherrschbar (vgl. Badstübner, 2008); zusätzlich wird sie durch neue Anwendungen ansteigen. Daher müssen neue Ideen und Konzepte in der Vernetzung von Steuergeräten in Automobilen Einzug halten, die dieses Problem lösen. Weiterhin sind neue, z. B. kamerabasierende Fahrerassistenzsysteme, die hohe Anforderungen an Bandbreite und Echtzeitfähigkeit (vgl. Steinbach, 2011) haben, zudem mit aktueller Technik nicht realisierbar, da sie entweder nicht über genügend Bandbreite (FlexRay oder CAN) verfügen oder keine Echtzeitfähigkeit besitzen (MOST). TTEthernet (vgl. Steiner, 2008) schafft den Spagat, hohe Bandbreiten bei gleichzeitiger Einhaltung von Echtzeiteigenschaften zu bieten und ist damit ein geeigneter Kandidat zukünftig im Automobil verwendet zu werden, um die vorgestellten Probleme zu lösen. Derzeit gibt es Prototypenprojekte, die den Einsatz von TTEthernet im Automobil evaluieren (vgl. Platzer, 2011).

Die hohe Anzahl an elektronischen Komponenten im PKW und dessen Entwicklung haben einen großen Einfluss auf die Gesamtkosten im Automobil. Lag der Anteil 1985 noch bei ca. 3 %, liegt dieser mittlerweile bei über 20% (vgl. Elektroniknet.de, 2008). Dieser Trend wird sich in Zukunft noch fortsetzen. Um ihm entgegen zu wirken, müssen die Kosten in der Entwicklung gering gehalten werden, indem frühzeitig im Entwicklungsprozess getestet wird (vgl. Riegraf u. a., 2007). Je später Fehler entdeckt werden, umso kostenintensiver ist deren Beseitigung. Ein gängiges Mittel dabei ist die Restbussimulation, bei der fehlende Komponenten durch eine Softwaresimulation ersetzt werden. Während der Entwicklung des „Schluckspechts“ (vgl. Team Schluckspecht), einem Forschungsfahrzeug mit Elektroantrieb der Hochschule Offenburg, das die längste Strecke mit einer Akkuladung gefahren ist, wurde die Restbussimulation erfolgreich eingesetzt (vgl. Ruth und Neuffer, 2011). Das zeigt, dass dieses Werkzeug eine zukunftsträchtige Technologie in der Entwicklung von Automobilen ist. Für das TTEthernetprotokoll ist noch

kein geeignetes Werkzeug zur Restbussimulation vorhanden, sodass es zu einem interessanten Forschungsgebiet gehört, da hier andere Anforderungen gelten als bei Bussystemen (vgl. Bartols, 2011). So müssen z. B. Synchronisationsmechanismen und verschieden Nachrichtenklassen unterstützt werden.

1.1 Motivation und Zielsetzung

Auf dem Gebiet der Entwicklung von Automotive-Anwendungen gibt es eine Vielzahl von Themen, die mit dem Projektthema „Restbussimulation von Time-Triggered Ethernet“ in Verbindung stehen. Dabei kann auf ein breites Angebot von vorhandenen Tools zurückgegriffen werden sowie auf Ergebnisse der Forschung. Ziel dieser Arbeit ist es, verwandte Arbeiten vorzustellen und den Zusammenhang zum Projekt zu verdeutlichen. Dabei soll eine gezielte Abgrenzung zum eigenen Projekt helfen, die Unterschiede herauszustellen und somit überprüft werden, welches Wissen der Arbeiten genutzt werden kann.

1.2 Inhaltlicher Aufbau der Ausarbeitung

Der nachfolgende Bereich 2 auf der nächsten Seite stellt verwandte Arbeiten im einzelnen vor, fasst diese zusammen und prüft sie auf Relevanz zum eigenen Projekt. Die Arbeiten werden zum Schluss im Abschnitt 3 auf Seite 10 zum eigenen Projekt abgegrenzt und es wird ein Fazit gezogen.

2 Verwandte Arbeiten

In diesem Abschnitt werden die für das vorgestellte Thema relevanten verwandten Arbeiten vorgestellt. Es wird dabei in drei unterschiedlichen Kategorien unterschieden. Zuerst wird dabei auf Restbussimulatoren aktueller Bussysteme eingegangen. Anschließend werden Prinzipien der WCET-Analyse erklärt und abschließend Modellierung von Software in der Entwicklung von Automotive-Komponenten beschrieben.

2.1 Restbussimulation der State-of-the-Art Bussysteme

Die Restbussimulation ist in der Entwicklung von Automotiveanwendungen ein gängiges Werkzeug. Sowohl im kommerziellen Bereich, als auch in der Forschung wird an diesem Themengebiet gearbeitet.

2.1.1 Kommerzielle Produkte

Gerade im kommerziellen Bereich ist die Entwicklung von Restbussimulatoren ein attraktives Umfeld für Werkzeughersteller im Softwareentwicklungsprozess. Problematisch daran ist, dass Interna der verschiedenen Produkte nicht veröffentlicht werden, um sich so abzuheben von anderen Herstellern. In dieser Ausarbeitung soll deshalb nur auf drei kommerzielle Produkte eingegangen werden. Der grundlegende Aufbau einer Restbussimulation ist in der Abbildung 2.1 auf der nächsten Seite zu sehen und besteht in der Regel aus drei Hauptkomponenten: Ein Host-Computer zur Konfiguration, dem Restbussimulator und dem Node-under-Test (NuT). Der Host-Computer besteht aus einer normalen, x86-basierten Workstation mit der entsprechenden Konfigurationssoftware. Der Restbussimulator ist mit einem Mikrocontroller versehen, auf dem die Restbussimulation mit harten Echtzeiteigenschaften ausgeführt wird. Die Produkte unterscheiden sich in der Regel durch Konfiguration, deren Software und der verwendeten Hardware. Außerdem ist die Unterstützung der Fahrzeugnetzwerksysteme produktspezifisch. Nachfolgend sind drei Produkte und deren Beschreibung aufgelistet:

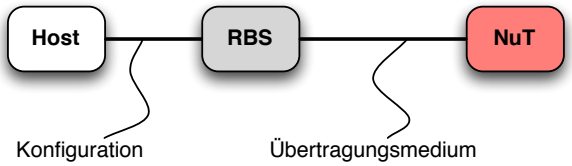

Abbildung 2.1: *Genereller Aufbau einer Restbussimulation*

CANoe ist ein Tool des Herstellers Vector Informatik (vgl. Vector Informatik) und zeichnet sich durch hohe Konfigurierbarkeit und dem damit verbundenen flexiblen Einsatz in der Entwicklung von Steuergeräten aus. So ist dieses Werkzeug in vielen Stadien der Entwicklung einsetzbar. Bei der Konzeption des Netzwerks kann im Vornherein mittels einer Kommunikationsmatrix und einer reinen Softwaresimulation festgestellt werden, ob dieses System den Anforderungen entspricht. Dieser Schritt wird in der Regel durch den OEM-Automobilhersteller durchgeführt. Anschließend können einzelne fertige Knoten in diese Simulation eingebunden werden. Als Modellierungswerkzeug/-Schnittstelle der simulierten Knoten kann auf MATLAB/Simulink/Stateflow oder der selbst entwickelten Programmiersprache CAPL zurückgegriffen werden. Weiterhin steht eine große Auswahl an Analysemethoden bereit, damit die Simulations- und Testergebnisse über Graphen und Grafiken visualisiert werden können. Zu Diagnosezwecken kann es zudem eingesetzt werden, um während der Verwendung des Netzwerkes Informationen zu sammeln und „On-the-fly“ Daten zu generieren. Der modulare Aufbau erlaubt es weiterhin, das Werkzeug mit neuen Hardwarekomponenten einfach zu erweitern, um andere Bussysteme zu unterstützen. Derzeit werden als Bussysteme CAN, Lin, Most, FlexRay, standard Ethernet und J1708 unterstützt.

FlexXCon wird vom Hersteller Eberspächer entwickelt (vgl. Eberspächer). Dieses Tool unterstützt die Bussysteme CAN, Lin und FlexRay. Zusätzlich zu einer Restbussimulation wird die Konfiguration von FlexRay zu FlexRay Gateways ermöglicht zum zwei getrennte Netzwerke zu verbinden. Weiterhin kann es auch dazu verwendet werden, Signale während des Einsatzes zu manipulieren. Dem Entwickler wird die Konfiguration dahingehend erleichtert, dass es möglich ist in Ansi-C Funktionen zu schreiben. Eine Besonderheit ist der Einsatz der Restbussimulation ohne Konfigurations-PC. Dabei wird die Konfiguration erstellt und auf den Simulator geladen. Da er auch für den Einsatz im Automobil entworfen ist, lassen sich so Steuergeräte unter nahezu realen Umgebungsumständen testen.

FlexRay CCM unterstützt die Bussysteme FlexRay und CAN und wird vom Hersteller IXXAT entwickelt (vgl. IXXAT Automation GmbH). Auch dieses Gerät lässt sich autonom betreiben und ist damit geeignet als ein CAN/FlexRay-Gateway zu fungieren. Durch eine präzise lokale Uhr ist dieses Tool zudem zur Analyse von CAN- und FlexRay-Botschaften

verwendbar. Die Restbussimulation wird mit erweiterbaren Softwaremodulen unter harten Echtzeiteigenschaften auf einem Microcontroller ausgeführt und kann mittels Python-Skripten oder User-Code direkt konfiguriert werden. In der Automobilentwicklung bei BMW wird dieses Tool mittlerweile produktiv eingesetzt (vgl. v. Häfen, 2008).

Im nachfolgenden Bereich werden wissenschaftliche Arbeiten zum Thema Restbussimulation erläutert, die zu den Interna des entwickelten Produktes mehr Informationen liefern.

2.1.2 Wissenschaftliche Arbeiten

Die Restbussimulation im wissenschaftlichen Umfeld ist durch die große Anzahl an bereits vorhandenen Tools aktueller Bussysteme sehr geprägt, sodass diese zwar erfolgreich eingesetzt werden, allerdings nicht selber entwickelt werden müssen. Die beiden nachfolgenden Arbeiten befassen sich zum einen mit einer Synchronisierung und zum anderen mit einer Simulation eines Time-Triggered Protokolls, ähnlich des FlexRay-Busses. Zuerst wird eine Arbeit des C-Labs Paderborn und anschließend eine Dissertation der TU-Wien vorgestellt.

Synchronisation eines SystemC-RBS mit einem HiL FlexRay Netzwerk

In dieser Arbeit des C-Labs Paderborn wird ein Sychronisierungsmechanismus beschrieben, mit dem es möglich ist, einen nicht echtzeitfähigen Restbussimulator mit einem FlexRay-Netzwerk (vgl. Defo u. a., 2011) zu synchronisieren. Dieses wurde durch eine Implementierung eines speziellen Synchronisationsbuffers ermöglicht, der als Schnittstelle zwischen dem Bus und der Restbussimulation liegt. Die Architektur des Simulators basiert auf standard PC-Hardware und SystemC (vgl. Open SystemC Initiative) als Simulationsumgebung. Die Anbindung an das „reale“ Netzwerk wird durch die Verwendung einer FlexRay PCI-Card hergestellt. Da SystemC als Simulationssprache nicht echtzeitfähig ist und die Datenverarbeitungsgeschwindigkeit der Hardware in der Regel höher ist, kann es vorkommen, dass nicht alle Daten, die zum RBS gesendet werden verarbeitet werden können. Das Ziel der Synchronisierung ist, diese Überladung zu vermeiden damit die Simulation dennoch performant ausgeführt wird. Mit Hilfe eines Downsamplingverfahrens, das ähnliche Daten herausfiltert, die das Verhalten eines Signalverlaufs (z. B. die Einwirkung einer Kraft auf einen Gegenstand über eine Zeitspanne) nicht größer beeinflussen, wird so Last vom Simulator genommen und die Synchronisierung eingehalten. Spitzen, also Daten, die den Verlauf deutlich beeinflussen, werden trotzdem an den RBS weitergeleitet. Durch dieses Verhalten ergeben sich allerdings geringfügige Änderungen des Verlaufs, die in Kauf genommen werden, da dieses Verfahren nur zu Funktionsüberprüfung eingesetzt wird. Anhand eines Drive-by-Wire Demonstrators wurde die Funktionsfähigkeit untersucht, in dem die vom Rad auf das Lenkrad übertragene Kraft analysiert wurde.

Clustersimulation in Time-Triggered Real-Time Systems

Diese Arbeit beschreibt die komplette Entwicklung eines Restbussimulators für den TTP/C-Bus und ist als Dissertation und Papier von der TU-Wien veröffentlicht worden (vgl. Galla, 1999; Galla und Pallierer, 1999). TTP/C (vgl. Elmenreich und Ipp, 2003) ist ein Bussystem, welches zu der Time-Triggered Architecture gehört. Damit verfügt es über eine zeitgesteuerte Übertragung und ist vom Anwendungsgebiet mit dem FlexRay-Bus vergleichbar. Knoten, also die Teilnehmer im Netzwerk, dieses Protokolls werden in zwei Bereiche unterteilt, die mittels „Communication Network Interface“ (in der Regel Dualportmemory) miteinander verbunden sind. Das Hostsubsystem ist für die Ausführung der eigentlichen Applikation verantwortlich, während das Communicationsubsystem das Übertragen der Daten übernimmt. Die Restbussimulation erfolgt in dieser Arbeit konform zum generellen Aufbau eines Knotens und kann daher auch in zwei Teile getrennt werden. Die Simulation der Knoten im Hostteil führt dazu, dass die Funktionalität auf einem Knoten abgebildet werden muss. Durch die geringe Rechenkapazität ist es erforderlich, dass Komplexität vereinfacht wird und von der eigentlichen Funktionalität abstrahiert wird. Das Scheduling der simulierten Knoten läuft zudem synchron zu dem des Busses, sodass jeder seinen eigenen Zeitslot hat, in dem er ausgeführt wird. Dabei muss überprüft werden, ob die Zeit ausreicht, den Knoten zu simulieren. Ein gängiges Mittel dafür ist die Worst-Case-Execution-Time (WCET) Analyse (siehe Abschnitt 2.2), sodass sich folgende Formel ergibt:

$$\tau_{csim}^{WCET} < t_i^{Uebertragungzeitpunkt} - t_i^{Aufrufezeitpunkt} \tag{2.1}$$

Diese besagt, dass die Worst-Case Ausführungszeit nicht länger dauern darf als die Differenz zwischen dem Übertragungszeitpunkt einer Nachricht und dem Zeitpunkt, an dem die Funktion des simulierten Knoten aufgerufen wird. Erfolgreich eingesetzt wurde dieser Restbussimulator bei der Entwicklung eines Steer-by-Wire-Prototypen, bei dem die Entwicklung der einzelnen Komponenten dezentral durchgeführt wurde. Somit konnten alle Beteiligten im Vornherein ihre Geräte auf Funktionalität prüfen.

2.2 Worst-Case-Execution-Time Analyse

Restbussimulatoren müssen die Ausführung der simulierten Knoten innerhalb bestimmter Grenzen garantieren, damit der Versendezeitpunkt einer Nachricht eingehalten wird, bzw. benötigte Daten rechtzeitig zur Verfügung stehen. Die Worst-Case-Execution-Time Analyse ist ein Verfahren zur Bestimmung der Worst-Case Ausführungszeit von Programmen und findet Verwendung in der Verifikation von Programmen mit harten Echtzeiteigenschaften und ist damit auch ein entsprechendes Werkzeug im Automotivebereich.

Einen Überblick über gängige Methoden, Werkzeuge und Problemstellungen an die WCET-Analyse liefert die Arbeit von Wilhelm et al. (vgl. Wilhelm u. a., 2008). Die Problemstellung

einer solchen Analyse ist die anfallende Komplexität. So hängt die Ausführungszeit unmittelbar von den Eingabedaten ab, denn je nach Eingabewert werden nur bestimmte Bereiche im Code erreicht und so gibt es mehrere Pfade durch die Funktion. Diese Daten sind eventuell bei der Bestimmung noch nicht bekannt, sodass Zeiten eventuell unerkannt bleiben. Ein weiteres Problem ist die hohe Komplexität moderner Prozessoren, deren aktueller Zustand nicht unmittelbar vorhergesagt werden kann. So verändert sich der Status des Caches kontinuierlich zur Laufzeit und hat bei mehreren Ausführungen immer andere Daten und Befehle zwischengespeichert. Das führt dazu, dass bei einem Cache-hit andere Zeiten anfallen, als bei einem Cache-miss. Bei der Bestimmung gibt es zwei gängige Verfahren. Bei der **statischen Analyse** wird kein Code ausgeführt sondern es findet eine Analyse anhand des Quellcodes statt. Dabei wird ein Kontrollflussgraph mithilfe von Werkzeugen erstellt, der das Verhalten widerspiegelt. Einzelne Codeblöcke können dann auf das zeitliche Verhalten analysiert werden, sodass die Gesamtzeit z. B. durch Addition der Blöcke unter Rücksichtnahme des Ausführungspfades im Kontrollflussgraphen errechnet werden kann. Dagegen wird bei **Messungen** Code mit einer gewissen Menge an Eingabedaten ausgeführt und die Zeit gemessen. Die Ausführung kann auf der eigentlichen Hardware oder auf speziellen Simulatoren gestartet werden. Die Eingabedaten können z. B. über die Spezifikation der Software bestimmt werden. Das Problem von Messungen ist, dass nur eine Abschätzung der WCET erreicht wird, weil die genaue Spezifikation der Software nicht vorliegt und so nicht alle Bereiche getestet werden können. Das führt dazu, dass die WCET mitunter zu klein bestimmt wird, während statische Analysen in der Regel zu einer Überbestimmung der WCET neigen, jedoch garantierte Grenzen der Ausführungszeit aufzeigen. Ein Vorteil von Messungen ist, dass es mittlerweile die Möglichkeit gibt eine WCET-Bestimmung ohne dem Vorhandensein von Source- oder Binary-Code durchzuführen (vgl. Marref und Betts, 2011) auf dessen Beschreibung im Angesicht des AW2-Rahmens verzichtet wird.

Die Arbeit von Frank et al. (vgl. Frank u. a., 2008) beschreibt mögliche Tools in der Entwicklung von Automotive Software. Zum einen wird hier auf mögliche Softwaresimulationen eingegangen, zum anderen die WCET-Analyse während des Entwicklungsprozesses beschrieben. Es wird auf ein Tool hingewiesen, welches bereits erfolgreich bei der Validierung von A380-Software verwendet wurde und damit auch zum Einsatzfeld in der Automotiveentwicklung zählt. Zusätzlich wurde auf Verbesserungen und Erweiterungen hingewiesen, die die Kosten für eine WCET-Analyse positiv beeinflussen. So wird beispielsweise erklärt, dass eine intelligente Speicherverwaltung die Vorhersagbarkeit von Cachezuständen positiv beeinflussen kann und damit das Ergebnis der WCET-Analyse genauer wird.

2.3 Modellierung von Automotive-Software

Eine Restbussimulation und das Verhalten der simulierten Knoten muss weiterhin einfach und intuitiv zu konfigurieren sein. In der Regel wird dieses durch die Verwendung von grafischen Modellierungstools ermöglicht.

AutoMoDe (vgl. Bauer u. a., 2005) ist ein wissenschaftliches Projekt das sich mit der Modellierung von Automotivesoftware und dessen besonderen Eigenschaften beschäftigt. So werden zeit- und ereignisgesteuerte Aufgaben gleichermaßen betrachtet. Diese Arbeit definiert für jeden Entwicklungsschritt eine Abstraktionsschicht, in der bestimmte Diagramme verwendet werden. Die „Functional Analysis Architecture“ ist die abstrakteste Schicht und beschreibt die Funktionalitäten, die entweder in Software oder in Hardware realisiert werden. Auf dieser Schicht werden „System Structure“ Diagramme verwendet, die im Prinzip dem Komponenten-Diagramm des UML entspricht. In der nachfolgenden „Functional Design Architecture“ wird das Verhalten einer bestimmten Aufgabe mit Hilfe von „Data Flow“ und „Mode Transition“ Diagrammen beschrieben. Die zur eigentlichen Implementierung nächste Schicht entspricht der „Logical“ und „Technical Architecture“. Auf diesen Bereichen wird die Kommunikation via „Cluster Communication“ Diagrammen repräsentiert, die Zeitpunkte und Datentypen zwischen einzelnen Komponenten der Übertragung definiert. Die Integrierung dieser Spezifikation in ein Werkzeug ist mit dem AutoFocus2-Programm realisiert worden.

2.4 Zusammenfassung und Diskussion der Arbeiten

Dieser Abschnitt fasst die vorgestellten Arbeiten zusammen und stellt die Kernaussagen heraus. Zusätzlich wird die Relevanz zur eigenen Arbeit diskutiert.

Die Restbussimulation wird sowohl in der Forschung als auch im kommerziellen Rahmen der Automobilentwicklung eingesetzt. Für die „State-of-the-Art“-Bussysteme (CAN, MOST, LIN, FlexRay) gibt es bereits eine Menge an Tools, die den Entwicklungsprozess unterstützen. Problematisch an dieser Stelle ist jedoch, dass die Produktinterna nicht veröffentlicht werden, da sich die Hersteller voneinander abgrenzen wollen. Die generelle Durchführung unterscheidet sich allerdings nicht, da der Aufbau mit einem Konfigurator, einem Simulator und einem „NuT“ identisch ist. Dieser Aufbau ist in sofern sinnvoll, da anspruchsvolle Aufgaben wie das Darstellen eines GUI nicht vom eigentlichen Simulator gemacht werden muss. Die Produkte haben in der Hinsicht auf das Projekt Einfluss auf die generelle Durchführung einer Restbussimulation. Zusätzlich können Aspekte des User-Interfaces übernommen werden, da z. B. CANoe ein etabliertes Tool ist.

In der Arbeit des C-Labs Paderborn ist beschrieben, wie eine nicht echtzeitfähige, PC-basierte Simulationsumgebung mit einem echtzeitfähigen Bus mittels speziellen Synchronisationsbuffer verbunden werden kann. In diesem Buffer wird ein Downsamplingverfahren angewendet, dass ähnliche Daten herausfiltert und somit Last von der Simulationsumgebung nimmt. Eine dadurch resultierende minimale Änderung im Signalverlauf wird in Kauf genommen, da der Simulator nur zur Funktionsprüfung verwendet wird. Für die im Projekt entwickelte Restbussimulation hat diese Arbeit nur geringe Relevanz, da keine Simulationsumgebung verwendet wird. Das Verhalten der simulierten Knoten wird im Vornherein modelliert und dann für die Restbussimulation kompiliert. Ausgeführt wird die Simulation dann auf einem Linux-basierten PC mit Echtzeitkernel.

Der entwickelte Restbussimulator der TU-Wien unterstützt das TTP/C-Protokoll und hält sich vom Aufbau her an der generellen Struktur eines TTP/C-Knoten. Das heißt es wird zwischen einem Kommunikations- und einem Hostsubsystem unterschieden. Die simulierten Knoten werden im Hostteil ausgeführt und werden synchron zum Bussprotokoll gescheduled. Die WCET der simulierten Knoten muss bestimmt werden, um zu garantieren, dass die Ausführung vollendet ist, bevor die Nachricht versendet wird. Diese Arbeit hat für das Projekt trotz der Veröffentlichung im Jahr 1999 eine sehr hohe Relevanz. Die Prinzipien des TTP/C-Busses entsprechen generell denen einer TT-Nachricht des TTEthernet (vgl. Bartols, 2011). Zusätzlich ist das synchronisierte Scheduling der simulierten Knoten sehr interessant, da der TTEthernet-Stack die Möglichkeit bietet synchron zum Protokoll Tasks auszuführen.

Einen großen Überblick über Probleme, Lösungsansätze und Tools in der WCET-Analyse findet man in der Arbeit von Wilhelm et al. Ein großes Problem dabei ist die hohe Komplexität moderner Computer, sodass Fachwissen über die Architektur der Hardware gefordert ist. Die Ausführungszeit korreliert weiterhin mit den Eingabedaten. Um die WCET bestimmen zu können gibt es zwei Möglichkeiten: Statische Methoden analysieren den Quellcode und zeigen garantierte Grenzen auf, während Messungen den Code auf einem Simulator oder auf der Hardware ausführen und nur Abschätzungen ermöglichen. Da die WCET-Analyse in der Regel zur Verifikation von Echtzeitsystemen eingesetzt wird, soll im Vornherein bei der Erzeugung des Code von simulierten Knoten auf dessen Ausführungszeit geachtet werden. Im Bezug zum Projekt können dazu die Prinzipien einer statischen Analyse helfen.

Die Arbeit des AutoMoDe-Projektes zeigt, wie eine Modellierung von Software im Automobilbereich in verschiedenen Schichten durchgeführt werden kann. Dazu wurden drei Abstraktionsschichten definiert, die in der Tiefe eine immer konkretere Beschreibung der Software zulassen. Die Diagramme erinnern teilweise an UML haben allerdings eine eigenen Syntax. Diese Arbeit macht deutlich, dass eine modellbasierte Entwicklung auch im Rahmen der Automobilentwicklung Einzug hält und damit den besonderen Anforderungen gerecht wird. Im Projekt muss derzeit evaluiert werden, mit welchen Werkzeugen eine Modellierung der simulierten Knoten möglich ist.

3 Abgrenzung und Fazit

Diese Arbeit wird mit einer Abgrenzung zur eigenen Arbeit und einem Fazit der vergleichbaren Arbeiten abgeschlossen.

3.1 Abgrenzung zur eigenen Arbeit

Dieser Abschnitt soll die Differenzen und Gemeinsamkeiten des Projektes zu den verwandten Arbeiten herausstellen und basiert auf der Struktur des letzten Kapitels. Zuerst wird ein Abgrenzung zu kommerziellen Produkten und wissenschaftlichen Arbeiten vorgenommen. Anschließend wird auf die Modellierung eingegangen und zuletzt die WCET-Analyse diskutiert.

3.1.1 Kommerzielle Produkte und wissenschaftliche Arbeiten

Zunächst sollen die Gemeinsamkeiten zum Projekt nahe gelegt werden. Das Hauptaugenmerk liegt auf dem gemeinsamen Prinzip des Aufbaus der Restbussimulation. Die Teilung der Konfiguration/Analyse und Ausführung auf separate Teilnehmer hat den Vorteil der effizienten Nutzung von Ressourcen. So kann die aufwendige Erstellung einer Konfiguration und Auswertung der Analysedaten mithilfe von grafischen Tools durchgeführt werden, während die Simulation ohne grafische Umgebung nicht dadurch beeinflusst wird. Fehlerhaftes Verhalten durch direkte Verwendung eines GUI kann so vermieden werden. Eine weitere Gemeinsamkeit spiegelt sich in dem Verhalten des Schedulings wider. So wird die Funktionalität abstrahiert und vereinfacht. Das Scheduling der simulierten Knoten wird im Projekt auch synchron zum Zyklus des TTEthernet durchgeführt, denn der Protokollstack erlaubt das Schedulen von Funktionen innerhalb des TTEthernet Zyklusses.

Zu den Unterschieden gehört die Plattform, auf der die Simulation ausgeführt wird. Im Projekt soll diese im Gegensatz auf einer x86-basierten Plattform mit einem Echtzeit-Linuxkernel ausgeführt werden. Damit soll erreicht werden, dass ein genaues Arbeiten der Restbussimulation weiterhin möglich ist und niederpriore Tätigkeiten keinen Einfluss auf die Simulation haben. Da nur TTEthernet als Netzwerkprotokoll unterstützt werden soll, entfällt ein weiteres aufwendiges Betreiben weiterer Systeme und wird durch die Verwendung eines Softwarestacks von TTTech

realisiert, sodass es möglich ist auf standard PC-Hardware zurückzugreifen. Da die Simulation der einzelnen Knoten mit Echtzeitpriorität im Linuxkernel ausgeführt werden, muss weiterhin keine aufwendige Synchronisierung über spezielle Synchronisationsbuffer durchgeführt werden. Dieses erleichtert die Implementierung erheblich.

3.1.2 Worst-Case-Execution-Time Analyse

Die WCET-Analyse wird in der Regel zur Verifikation von Software mit harten Echtzeiteigenschaften verwendet und wird dementsprechend nach der Entwicklung eingesetzt. Im Projekt sollen die Prinzipien einer statischen Analyse während der Entwicklung und Erzeugung von Code der simulierten Knoten verwendet werden. So soll Code möglichst seriell und linear auf einem Pfad ausgeführt werden, denn es hat sich gezeigt, dass mehrere Pfade und eventuelle Schleifen die Ausführungszeit jittern lässt. Dementsprechend muss der Code linear erzeugt werden.

3.1.3 Modellierung von Automotive Software

Im Vergleich zur der vorgestellten Arbeit kann in diesem Projekt auf ein aufwändiges Modellieren der Software verzichtet werden. Wichtig ist vor allem, die Abbildung der Funktionen innerhalb des TTEthernetzyklusses, damit modelliert werden kann, zu welchem Zeitpunkt welche Funktion ausgeführt werden soll. Da die Restbussimulation in der Regel stark von der eigentlichen Anwendung abstrahiert, muss zum großen Teil nur auf Inhalte der versendeten Nachrichten reagiert werden. Bei einer „Open-Loop“-Simulation, dem einfachen Stimulieren ohne Reaktion auf Rückgabewerte vom NuT, kann generell auf aufwändige Regelkreise verzichtet werden. Bei der „Closed-Loop“-Variante, bei der in Abhängigkeit der Rückgabewerte mit angepassten Daten reagiert wird, kann mit Hilfe von Automaten ein Verhalten gut abgebildet werden.

3.2 Fazit verwandter Arbeiten

Diese Literaturrecherche hat interessante Arbeiten erbracht dessen Ideen teilweise komplett, teilweise abgeändert im Projekt verwendet werden können. In Hinsicht des Vergleichbare Arbeiten Kapitels der Masterarbeit sollte zusätzlich noch auf Arbeiten der reinen Softwaresimulation von Netzwerken eingegangen werden, da z. B. CANoe als Tool diese Möglichkeit bereitstellt. Generell ist mit dieser Arbeit die Literaturrecherche nicht beendet und wird stetig fortgesetzt.

Literaturverzeichnis

[Badstübner 2008] BADSTÜBNER, Jens: Kollaps im Bordnetz: Schluss mit Can, Lin und Flexray. In: *KFZ-Betrieb* (2008), Nr. 17, S. 68–70

[Bartols 2011] BARTOLS, Florian: *Restbussimulation von Time-Triggered Ethernet in Automotive Anwendungen – AW1 Ausarbeitung*. Februar 2011. – Bericht

[Bauer u. a. 2005] BAUER, Andreas ; BROY, Manfred ; ROMBERG, Jan ; SCHÄTZ, Bernhard ; BRAUN, Peter ; FREUND, Ulrich ; MATA, Núria ; SANDNER, Robert ; ZIEGENBEIN, Dirk: AutoMoDe — Notations, Methods, and Tools for Model-Based Development of Automotive Software. In: *Proceedings of the SAE 2005 World Congress* Bd. 1921. Detroit, MI : Society of Automotive Engineers, April 2005. – ISBN ISBN 0-7680-1566-9

[Defo u. a. 2011] DEFO, Gilles B. ; MUELLER, Wolfgang ; ROMMEL, Heinrich: Synchronisierung eines SystemC Restbus-Simulators mit einem Hardware-In-the-Loop FlexRay Netzwerk. In: *14. Workshop: Methoden und Beschreibungssprachen zur Modellierung und Verifikation von Schaltungen und Systemen*, Februar 2011. – to appear

[Eberspächer] EBERSPÄCHER: *FlexXCon - Restbussimulation*. Eberspächer. – URL `http://www.eberspaecher-electronics.com/loesungen/rbs-gateway/restbussimulation.html`. – Zugriffsdatum: 2011-18-08

[Elektroniknet.de 2008] ELEKTRONIKNET.DE: *Markt für Autoelektronik weiterhin mit ungebrochenem Wachstum*. Februar 2008. – URL `http://www.elektroniknet.de/automotive/technik-know-how/prozesse-standards-und-qualitaet/article/741/0/Markt_fuer_Autoelektronik_weiterhin_mit_ungebrochenem_Wachstum/`. – Zugriffsdatum: 2011-16-08

[Elmenreich und Ipp 2003] ELMENREICH, Wilfried ; IPP, Richard: Introduction to TTP/C and TTP/A. In: *Proceedings of the Workshop on Time-Triggered and Real-Time Communication Systems*, 2003, S. 1–9. – eingeladen; Vortrag: Workshop on Time-Triggered and Real-Time Communication Systems, Manno, Switzerland; 2003-12-02

[Frank u. a. 2008] FRANK, E. ; WILHELM, R. ; ERNST, R. ; SANGIOVANNI-VINCENTELLI, A. ; DI NATALE, M.: Methods, tools and standards for the analysis, evaluation and design of modern automotive architectures. In: *Proceedings of the conference on Design, automation*

and test in Europe. New York, NY, USA : ACM, 2008 (DATE '08), S. 659–663. – URL http://doi.acm.org/10.1145/1403375.1403536. – ISBN 978-3-9810801-3-1

[Galla 1999] GALLA, Thomas M.: *Cluster Simulation in Time-Triggered Real-Time Systems*. Wien, TU Wien, Dissertation, Dezember 1999

[Galla und Pallierer 1999] GALLA, Thomas M. ; PALLIERER, Roman: Cluster simulation-support for distributed development of hard real-time systems using TDMA-based communication. In: *Proceedings of the 11th Euromicro Conference on Real-Time Systems, 1999.*, Juni 1999, S. 150–157

[v. Häfen 2008] HÄFEN, Robert v.: *Restbussimulation für FlexRay-Netzwerke*. Ixxat. April 2008. – URL http://www.ixxat.de/article_flexray_restbus_apr08_de.html. – Zugriffsdatum: 2011-08-27

[IXXAT Automation GmbH] IXXAT AUTOMATION GMBH: *FlexRay CCM*. Ixxat. – URL http://www.ixxat.de/flexray_ccm_de.html. – Zugriffsdatum: 2011-08-28

[Marref und Betts 2011] MARREF, Amine ; BETTS, Adam: Accurate Measurement-Based WCET Analysis in the Absence of Source and Binary Code. In: *14th IEEE International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing (ISORC)*, März 2011, S. 127–135. – ISSN 1555-0885

[Marscholik und Subke 2007] MARSCHOLIK, Christoph ; SUBKE, Peter: *Datenkommunikation im Automobil: Grundlagen, Bussysteme, Protokolle und Anwendungen*. Heidelberg : Hüthig, 2007 (Hüthig Praxis). – ISBN 3-7785-2969-2

[Navet u. a. 2005] NAVET, Nicolas ; SONG, Yeqiong ; SIMONOT-LION, Françoise ; WILWERT, Cédric: Trends in Automotive Communication Systems. In: *Proceedings of the IEEE* 93 (2005), Juni, Nr. 6, S. 1204–1223. – ISSN 0018-9219

[Open SystemC Initiative] OPEN SYSTEMC INITIATIVE: . – URL http://www.systemc.org/home/. – Zugriffsdatum: 2011-08-18

[Platzer 2011] PLATZER, Petra: *Dr. Markus Plankensteiner im Interview über TTEthernet: "Ethernet-Lösung für hochverfügbare Anwendungen"*. http://www.sps-magazin.de/. Mai 2011. – URL http://www.sps-magazin.de/?inc=artikel/article_show&nr=61739. – Zugriffsdatum: 2011-08-18. – Interview

[Riegraf u. a. 2007] RIEGRAF, Thomas ; BEHH, Siegfried ; KRAUS, Stefan: *Effizientes Testen in der Automobilelektronik - Von der Simulation bis zur Diagnose*. Vector Informatik. August 2007. – URL http://www.vector.com/portal/medien/cmc/press/PND/Testen_ATZ_200708_PressArticle_DE.pdf. – Zugriffsdatum: 2010-12-20

[Ruth und Neuffer 2011] RUTH, Heiko ; NEUFFER, Jochen: *Elektromobilität auf dem Vormarsch*. Vector Informatik. Juli 2011. – URL http://www.vector.com/portal/medien/cmc/press/Vector/Schluckspecht_EletronikAutomotive_201107_PressArticle_DE.pdf. – Zugriffsdatum: 2011-08-15

[Steinbach 2011] STEINBACH, Till: *Echtzeit-Ethernet für Anwendungen im Automobil: Metriken und deren simulationsbasierte Evaluierung am Beispiel von TTEthernet*. Hamburg, Hochschule für Angewandte Wissenschaften Hamburg, Masterthesis, Februar 2011

[Steiner 2008] STEINER, Wilfried: *TTEthernet Specification*. TTTech Computertechnik AG. November 2008. – URL http://www.tttech.com

[Team Schluckspecht] TEAM SCHLUCKSPECHT: *Schluckspecht-Projekt*. – URL http://www.schluckspecht.net/. – Zugriffsdatum: 2011-16-08

[Vector Informatik] VECTOR INFORMATIK: *CANoe - Restbussimulation*. Vektor Informatik. – URL http://www.vector.com/vi_canoe_de.html. – Zugriffsdatum: 2010-12-28

[Wilhelm u. a. 2008] WILHELM, Reinhard ; ENGBLOM, Jakob ; ERMEDAHL, Andreas ; HOLSTI, Niklas ; THESING, Stephan ; WHALLEY, David ; BERNAT, Guillem ; FERDINAND, Christian ; HECKMANN, Reinhold ; MITRA, Tulika ; MUELLER, Frank ; PUAUT, Isabelle ; PUSCHNER, Peter ; STASCHULAT, Jan ; STENSTRÖM, Per: The worst-case execution-time problem - An Overview of Methods and Survey of Tools. In: *ACM Trans. Embed. Comput. Syst.* 7 (2008), Mai, S. 36:1–36:53. – ISSN 1539-9087

[Zimmermann und Schmidgall 2011] ZIMMERMANN, Werner ; SCHMIDGALL, Ralf: *Bussysteme in der Fahrzeugtechnik - 4. aktualisierte Auflage*. Vieweg + Teubner, 2011. – ISBN 978-3-8348-0907-0