



Hochschule für Angewandte Wissenschaften Hamburg  
*Hamburg University of Applied Sciences*

## **Ausarbeitung - Anwendungen 1 WS2011-2012**

Friedrich Groß

Hardware/Software Co-Design für Real-time Ethernet  
basierte Steuergeräte

**Friedrich Groß**

**Thema der Ausarbeitung - Anwendungen 1**

**WS2011-2012**

Hardware/Software Co-Design für Real-time Ethernet basierte Steuergeräte

**Stichworte**

TTEthernet, Time-Triggered Ethernet, Real-time Ethernet, Hardware Software Co-Design, Automotive Anwendungen

**Kurzzusammenfassung**

In dieser Arbeit wird ein neues Entwicklungskonzept für Automobil-Steuergeräte vorgestellt. Dabei geht es darum Steuergeräte gekoppelt an eine Kommunikationstechnologie in Hardware/Software Co-Design zu entwickeln. Dies bedeutet, dass die Logik in parallele und sequenzielle Logik aufgeteilt wird. Es soll für jeden Anwendungsfall eine optimale Partitionierung der Module möglich sein, sodass Steuergeräte in Zukunft höchste Anforderungen mit geringem Ressourcenbedarf erfüllen können. Die Aufteilung der Logik in sequenzielle und parallele Logik ist dabei vielversprechend, wofür in dieser Arbeit erste Grundsteine und Überlegungen für die zukünftigen Arbeiten getätigt werden. Als Kommunikationstechnologie wird in dieser Arbeit Time-Triggered Ethernet favorisiert, welches eine echtzeiterweiterung des Ethernet-Protokolls ist.

**Title of the paper**

Hardware/Software Co-Design for Real-time Ethernet-based controllers

**Keywords**

TTEthernet, Time-Triggered Ethernet, Real-time Ethernet, Automotive Applications, Hardware Software Co-Design

**Abstract**

In this work, a new development concept for automotive control units is presented. It's about to couple control units to a communications technology with use of hardware / software co-design to develop. It should be possible for each application the optimal partitioning of the modules, so that controllers can meet the highest requirements in the future with minimal resource requirements. The breakdown of the logic in sequential and parallel logic is promising, are what made this work first foundation stones and considerations for future work. As communication technology in this work Time-Triggered Ethernet is favored, which is a real-time Ethernet.

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>1</b>
1.1	Motivation . . . . .	1
1.1.1	Warum Real-time Ethernet . . . . .	2
1.1.2	Warum Hardware/Software Co-Design . . . . .	3
1.2	Inhaltlicher Aufbau der Ausarbeitung . . . . .	4
<b>2</b>	<b>Technologieüberblick</b>	<b>5</b>
2.1	Time-Triggered Ethernet . . . . .	5
2.2	Hardware/Software Co-Design . . . . .	7
<b>3</b>	<b>Problemstellung und Zielsetzung</b>	<b>9</b>
3.1	Steuergeräteentwicklung in Zukunft . . . . .	9
3.1.1	Beispiel 1: Elektronisches Stabilitätsprogramm . . . . .	10
3.1.2	Beispiel 2: Multimediasystem . . . . .	10
3.1.3	Entscheidungskriterien für die Partitionierung . . . . .	11
<b>4</b>	<b>Ausblick und Zusammenfassung</b>	<b>12</b>
4.1	Ausblick auf weiteres Vorgehen . . . . .	12
4.2	Zusammenfassung . . . . .	12
	<b>Abkürzungsverzeichnis</b>	<b>13</b>
	<b>Literaturverzeichnis</b>	<b>14</b>

# 1 Einleitung

In dieser Arbeit geht es darum die Motivation hinter meinem Masterthema darzulegen, eine Einführung in die Technologie zu bekommen und die Problemstellung zu beschreiben. Weiterhin werden erste grundlegende Anforderungen aufgestellt, Risiken des Projekts erkannt und ein Ausblick auf zukünftige Arbeiten gegeben.

## 1.1 Motivation

In einem Automobil werden Bus-Systeme für die Kommunikation zwischen den einzelnen Steuergeräten, Sensoren und Aktoren verwendet. Die steigende Anzahl an Fahrerassistenzsystemen führt zu einer steigenden Anzahl an Steuergeräten, was wiederum zu einer komplexeren Kommunikation und einem erhöhten Bandbreitenbedarf führt. Insbesondere Systeme, wie Rückfahrkamera, Multimediadienste in Kopfstützen, Verkehrsschilder- oder Spurerkennung steigern den Bandbreitenbedarf in einem Automobil. Ein Teil der Kommunikation muss Echtzeitbedingungen erfüllen, was bedeutet, dass Daten innerhalb einer bestimmten Zeit garantiert übertragen werden müssen. Dies ist insbesondere bei sicherheitsrelevanten Systemen der Fall.

Im Jahr 2000 wurde von einigen Automobil- und Elektronikherstellern das FlexRay-Konsortium gegründet, wodurch ein neues Bussystem namens FlexRay entwickelt wurde (vgl. FlexRay Consortium). FlexRay ist echtzeitfähig und liefert eine Bandbreite von 10Mbit/s. 2007 wurde das erste Fahrzeug ausgeliefert, welches FlexRay als weiteres Bussystem verwendet (vgl. Böke, 2008). Da die Bandbreite von FlexRay für Bild- und Multimediaübertragungen nicht reicht und die Echtzeitfähigkeit nicht überall benötigt wird, werden weitere Bus-Systeme, die für ihren Einsatzzweck optimiert sind, in einem Fahrzeug verwendet. Beispielsweise wird der MOST-Bus (vgl. MOST Cooperation) für Übertragungen von Multimediadaten im Fahrzeug eingesetzt und der kostengünstige LIN-Bus (vgl. LIN-Administration) für abgesetzte Sensoren und Aktoren z. B. in Türen.

Der Einsatz von verschiedenen Bus-Systemen steigert die Komplexität der Steuergeräte und erfordert teilweise verschiedene Schnittstellen und Gateways in einem Steuergerät. „Eigentlich ist das Bordnetz im Gesamtfahrzeug bereits heute nicht mehr vernünftig zu beherrschen“ und „Die Komplexität steigt weiter“ (Badstübner, 2008, BMW Group Forschung und Technik).

Abbildung 1.1 zeigt einen Maybach aus dem Jahr 2002, in dem damals schon 76 Steuergeräte eingebaut wurden. Abbildung 1.2 zeigt die durchschnittliche Anzahl der Kommunikationsknotenpunkte in einem Fahrzeug. Dort sieht man, dass im Jahr 2013 in europäischen Autos (blaue Linie) durchschnittlich über 30 Knotenpunkte erwartet werden. Ein einheitlicheres, gleichzeitig schnelles und echtzeitfähiges Bus-System würde das steigende Bandbreitenproblem lösen und die Komplexität verringern bzw. in Zukunft nicht weiter ansteigen lassen.

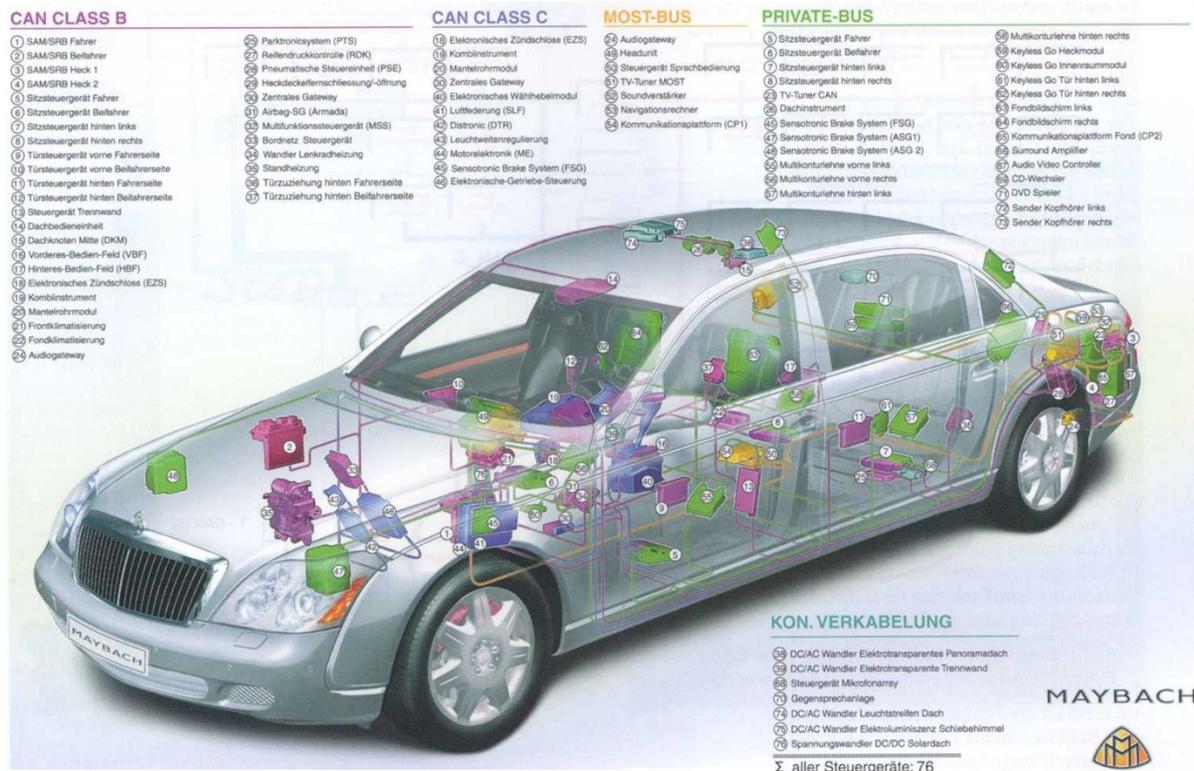
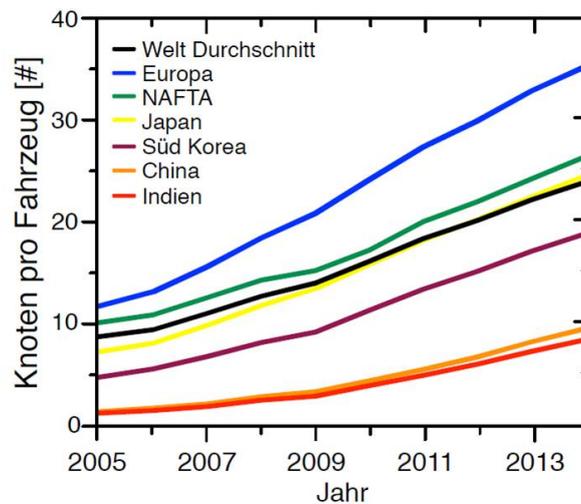


Abbildung 1.1: Maybach Steuergeräte (Daimler AG, 2002)

### 1.1.1 Warum Real-time Ethernet

Ethernet ist eine weit verbreitete Kommunikationstechnologie, die jedoch durch ihren eventbasierten Ansatz in zeitkritischen Systemen lange Zeit nicht eingesetzt wurde. Durch die Entwicklungen in den vergangenen Jahren sind einige echtzeitfähige Varianten des Ethernet-Protokolls entwickelt worden.

Unter dem Begriff Real-time Ethernet werden alle echtzeitfähigen Ethernet-Protokoll-Erweiterungen zusammengefasst. Real-time Ethernet-Protokolle gelten als Lösung für den steigenden Kommunikationsbedarf im Backbone-Netz eines Automobils. Die Hauptvorteile



**Abbildung 1.2:** Durchschnittliche Anzahl Knotenpunkte in einem Fahrzeug (vgl. Bruckmeier, 2010)

liegen in der Skalierbarkeit von Bandbreite und Übertragungsmedium. Weitere Vorteile sind, dass viele Informatikabsolventen mit Standard-Ethernet vertraut sind und die Hardware und Leitungen durch die hohen Stückzahlen günstig sind.

Time-Triggered Ethernet (TTEthernet) ist ein konkretes Real-time Ethernet und gilt als Kandidat für die schrittweise Ablösung von alten Bussystemen im Backbone-Netz eines Automobils. Vorarbeiten haben gezeigt, dass TTEthernet eine Erfolg versprechende Technologie für die Kommunikation in Fahrzeugen ist (vgl. Steinbach u. a., 2010). TTEthernet wurde an der Technischen Universität Wien entwickelt und wird aktuell von der Firma TTTech (vgl. TTTech Computertechnik AG) vermarktet. Dieses Bussystem wird derzeit nicht im Automobil eingesetzt, es befindet sich praktisch in der „Evaluierungsphase“ (vgl. SAE - AS-2D Time Triggered Systems and Architecture Committee, 2009).

Diese Arbeit konzentriert sich auf den Einsatz des TTEthernet-Protokolls.

### 1.1.2 Warum Hardware/Software Co-Design

Ziel des Hardware/Software Co-Design, ist es Funktionen in parallele Logikblöcke und sequenzielle CPU-Logik aufzuteilen. Ein rein softwarebasierter TTEthernet-Stack für einen Mikrocontroller wurde im CoRE-Team bereits entwickelt (vgl. Müller, 2011). Dieser Software-Stack kommt jedoch bei hohen Bandbreiten an seine Grenzen, sodass nicht genug CPU-Ressourcen für eine Anwendung übrig bleiben. In dem Software-Stack werden alle Teile der TTEthernet Spezifikation sequenziell von der CPU ausgeführt, obwohl sich einige Teile parallelisiert abarbeiten lassen könnten, was jedoch mit einem Mikrocontroller nicht möglich ist.

In dieser Arbeit wird auf HW/SW Co-Design gesetzt, da sich hiermit der TTEthernet-Stack in Mikrocontroller basierte Software und paralleler Logik aufgeteilt werden kann, sodass die parallele Logik als Beschleuniger-Module für die Mikrocontroller arbeiten kann. Zudem können unerwünschte Nachrichtenklassen in der parallelen Logik gefiltert werden, sodass diese keinen Interrupts an der CPU auslösen und diese bei anderen Berechnungen unterbrechen. Bei der reinen Mikrocontroller Lösung wird bei jeder Nachricht, egal welche Priorität diese hat oder ob diese für dieses Gerät bestimmt war, ein Interrupt ausgelöst und der Mikroprozessor muss diese Nachricht auf Priorität und Zugehörigkeit prüfen. Diese Schwächen einer Mikrocontroller-Entwicklung werden bei einer HW/SW Co-Design Entwicklung unterbunden.

## 1.2 Inhaltlicher Aufbau der Ausarbeitung

In Kapitel 2 auf der nächsten Seite werden für das Lesen dieser Arbeit wichtige Grundlagen vermittelt. Dabei wird das Thema Time-Triggered Ethernet und Hardware/Software Co-Design eingeführt.

Im Kapitel 3 auf Seite 9 wird die Problemstellung und Zielsetzung dieser Arbeit beschrieben. Dabei wird ein Entwicklungskonzept für zukünftige Steuergeräte vorgestellt und dieses anhand von Beispielen erläutert.

Zum Schluss dieser Arbeit gibt es in Kapitel 4 auf Seite 12 einen Ausblick auf das weitere Vorgehen und eine Zusammenfassung dieser Arbeit.

## 2 Technologieüberblick

### 2.1 Time-Triggered Ethernet

Time-Triggered Ethernet (im Folgenden TTEthernet) ist eine Echtzeiterweiterung für das Ethernet. Es ermöglicht u.a. das zeitgesteuerte Kommunizieren innerhalb eines Netzwerks, wobei für die zeitgesteuerten Nachrichten eine deterministische Paketlaufzeit durch das Netzwerk garantiert wird. Um dieses Verhalten zu ermöglichen, werden spezielle TTEthernet-Switches zwischen Netzwerkteilnehmern benötigt. In anderen Bereichen des Netzwerks können normale Switches verwendet werden.

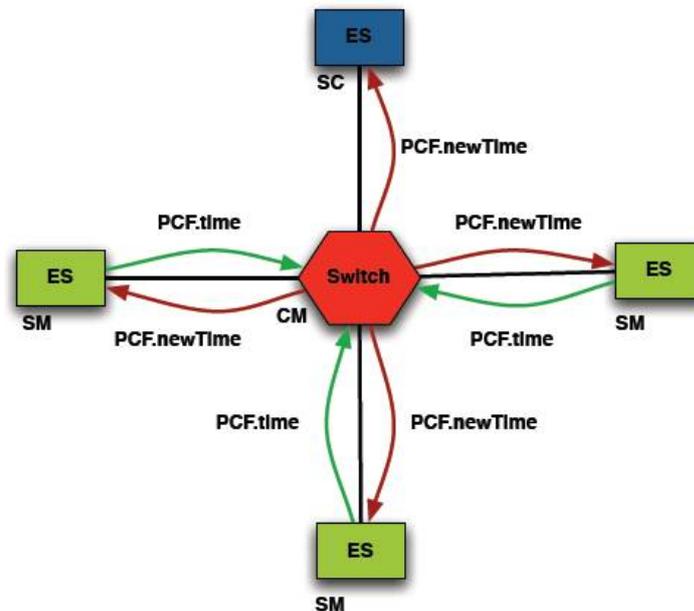
Das TTEthernet-Protokoll unterstützt drei Nachrichtenklassen, die im Folgenden beschrieben sind:

**Time-Triggered-Traffic (TT)** wird für den zeitgesteuerten Datenaustausch verwendet. TT-Frames haben kurze Paketlaufzeit mit einem niedrigen Jitter, was für ein deterministisches Verhalten sorgt. TT-Frames werden von allen Netzwerkgeräten mit höchster Priorität behandelt. Diese Nachrichtenklasse wird für zeitkritische Echtzeitkommunikation verwendet.

**Rate-Constrained-Traffic (RC)** wird für weniger zeitkritische Echtzeitkommunikation verwendet. Die Echtzeitfähigkeit wird durch eine vorher festgelegte garantierte Bandbreite realisiert. Der Nachrichtenaustausch erfolgt eventbasiert und muss wie bei den TT-Nachrichten durch einen speziellen Switch geleitet werden. Diese Nachrichtenklasse entspricht dem AFDX-Protokoll-Standard (vgl. AIM GmbH).

**Best-Effort-Traffic (BE)** entspricht dem Standard Ethernet Verkehr. Nachrichten dieser Klasse werden mit niedrigster Priorität behandelt, sodass eine Übertragung einzelner Frames nicht garantiert werden kann.

Um das zeitgesteuerte Kommunizieren zu ermöglichen, ist eine globale Uhr notwendig, über deren Zeit alle Netzwerkteilnehmer informiert werden müssen. Aus dem Grund beinhaltet die TTEthernet-Spezifikation eine Zeitsynchronisation. Die Zeitsynchronisation wird durch eine Zwei-Wege-Synchronisation realisiert, in der den Netzwerkteilnehmern unterschiedliche Rollen zugeteilt werden, die im Folgenden beschrieben werden und in Abbildung 2.1 veranschaulicht sind:



**Abbildung 2.1:** Beispiel: Synchronisation im TTEthernet (Quelle: Bartols, 2010)

ES: Endsystem - SC: Synchronisation Client - SM: Synchronisation Master -  
CM: Compression Master - PCF: Protocol Control Frame

**Synchronisation-Master (SM)** leiten die Synchronisation ein, in dem PCF-Frames versendet werden. Das PCF-Frame beinhaltet die aktuelle Zeit des SM. Diese Rolle wird mehreren Teilnehmern zugeteilt, um das Netzwerk fehlertolerant zu halten.

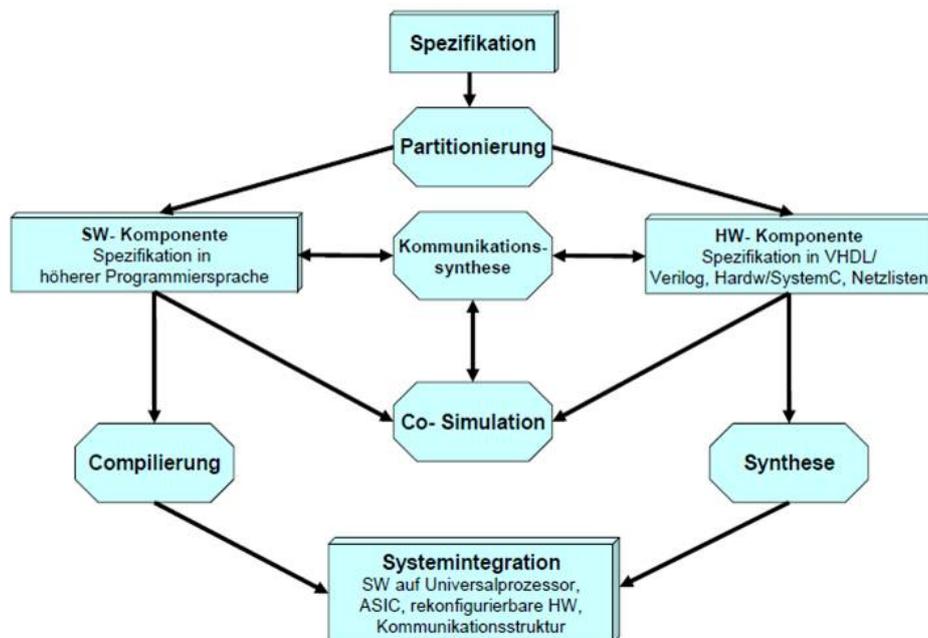
**Compression-Master (CM)** sammeln eintreffende PCF-Frames, errechnen aus den empfangenen Zeiten eine neue Durchschnittszeit, synchronisieren sich auf diese Zeit und leiten diese an alle Teilnehmer weiter. Diese Rolle wird zentralen Switches zugeteilt.

**Synchronisation-Client (SC)** synchronisieren sich auf ein eintreffendes PCF-Frame und leiten diese ggf. an andere Teilnehmer weiter. Vor dem Weiterleiten wird auf die im PCF-Frame eingetragene Zeit die eigene Verzögerung aufaddiert. Diese Rolle wird Endsystemen und Switches die nicht CM sind zugeteilt.

Ein PCF-Frame (Protocol Control Frames) wird u.a. für den Austausch von Zeitstempeln für die Synchronisation verwendet.

## 2.2 Hardware/Software Co-Design

Beim Hardware/Software Co-Design wird die Entwicklung einer Logik in Hardware und Software aufgeteilt, sodass die Hardware- und Softwareelemente miteinander interagieren und gemeinsam ein Problem lösen. Entwickelt wird auf einem Gerät, das parallele und sequenzielle Ressourcen anbietet, z. B. auf einem FPGA oder einem ASIC. Diese Arbeit beschränkt sich auf die Entwicklung auf einem FPGA, der einen oder mehrere Mikroprozessoren enthält.



**Abbildung 2.2:** Ablaufschema Hardware/Software Co-Design (Michael Hübner, 2009)

Abbildung 2.2 zeigt das Ablaufschema mit den Entwicklungsschritten beim Hardware Software Co-Design. Im Folgenden werden die einzelnen Schritte erläutert:

**Spezifikation** In diesem Schritt wird das System spezifiziert, in Module aufgeteilt und deren Anforderungen wie z. B. Performance und Jitter festgelegt.

**Partitionierung** Im Schritt Partitionierung wird die Entscheidung getroffen, welche Module in Hardware und welche in Software umgesetzt werden sollen.

**SW-Komponenten** Im Schritt SW-Komponenten werden die Softwarekomponenten in einer höheren Programmiersprache z. B. C spezifiziert und implementiert.

**HW-Komponenten** Im Schritt HW-Komponenten werden die Hardware Komponenten in einer Beschreibungssprache z. B. VHDL oder Verilog spezifiziert und implementiert.

**Kommunikationssynthese** Im Schritt Kommunikationssynthese werden die Schnittstellen (z. B. Register) zwischen den Software- und Hardwaremodulen festgelegt.

**Co-Simulation** In der Co-Simulation werden die Hardware- und Softwaremodule in deren Funktion simuliert und dabei auch festgestellt, ob die Schnittstellen miteinander kompatibel sind. Dies geschieht z. B. mit dem Werkzeug Matlab/Simulink.

**Compilierung** In diesem Schritt wird Maschinencode für eine spezifische CPU aus einer höheren Sprache übersetzt.

**Synthese** Bei der Synthese wird aus einer Hardware-Beschreibungssprache ein konkreter Schaltplan für ein Hardwaremodul erstellt.

**Systemintegration** In der Systemintegration erfolgt dann das Zusammenspiel der kompilierten und synthetisierten Komponenten.

# 3 Problemstellung und Zielsetzung

## 3.1 Steuergeräteentwicklung in Zukunft

Das Ziel der zukünftigen Arbeiten ist es, eine neue Steuergerätearchitektur für den Automobilbereich zu konzipieren und zu entwickeln. Dabei soll der Fokus auf die Kommunikationsmodule des Steuergerätes gelegt werden. Als Kommunikationsmittel soll ein Real-time Ethernet Bus verwendet werden und in einer konkreten Entwicklung das Time-Triggered Ethernet (TTEthernet) Protokoll implementiert werden.

Dabei soll ein Baukastensystem entwickelt werden, dass es ermöglicht Module eines Kommunikationsstacks in Hardware, Software oder gar nicht zu implementieren. Es soll möglich sein für ein konkretes Steuergerät ein optimales Partitionierungsprofil anzulegen. Um dies besser zu beschreiben, folgen in den nächsten zwei Abschnitten Beispiele, bei denen konkrete Partitionierungsprofile angelegt werden. Für diese Beispiele wird der Kommunikationsstack ebenfalls beispielhaft in vier Module aufgeteilt:

**Zeitsynchronisation Modul** Zuständig für die Time-Triggered Ethernet spezifische Zeitsynchronisation. Erfordert einen zyklischen Timer, dessen Geschwindigkeit und Position im Zyklus angepasst werden kann.

**Time-Triggered Modul** Zuständig für die Abarbeitung von zeitgesteuerten Daten. Dieses Modul muss ein Schedule implementieren, dass zu versendende TT-Nachrichten enthält und darauf achtet, dass zu diesen Zeitpunkten keine anderen Daten versendet werden.

**Rate-Constrained Modul** Zuständig für die Abarbeitung von Echtzeitdaten, denen eine bestimmte Bandbreite garantiert wird. Dieses Modul muss einen Mechanismus implementieren, das erkennt wenn Bandbreitengrenzen erreicht werden und an Grenzbereichen Best-Effort Daten beim Versenden von Rate-Constrained Daten verdrängt.

**Best-Effort Modul** Zuständig für die Abarbeitung von eventbasierten zeitunkritischen Daten. Diese Daten werden versendet, wenn die Bedingungen von TT- und RC-Nachrichten nicht verletzt werden.

Die Entscheidungskriterien für die Partitionierung werden im Abschnitt 3.1.3 beschrieben.

### 3.1.1 Beispiel 1: Elektronisches Stabilitätsprogramm

Angenommen ein Steuergerät für ein ESP-System (Elektronisches Stabilitätsprogramm) muss zeitkritische Daten in Echtzeit mit anderen Kommunikationsteilnehmern austauschen, so wäre das Partitionierungsprofil sinnvoll, das in Abbildung 3.1 dargestellt wird. Bei diesem Profil wird das Zeitsynchronisation Modul und das Time-Triggered Modul in Hardware realisiert werden, das Rate-Constrained Modul nicht implementiert und das Best-Effort Modul in Software implementiert um z. B. Diagnose-Nachrichten verschicken zu können. Das Zeitsynchronisation-Modul und das Time-Triggered Modul sind hier im Profil mit enthalten, da ein ESP-System sicherheitsrelevante zeitkritische Daten austauschen muss. Das Rate-Constrained Modul wird nicht benötigt, da keine unkritischen Echtzeitdaten wie z. B. Medienströme ausgetauscht werden sollen.

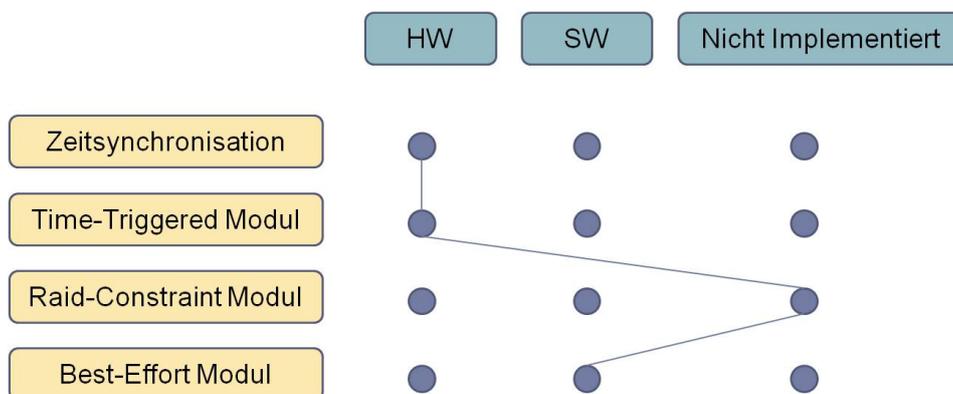
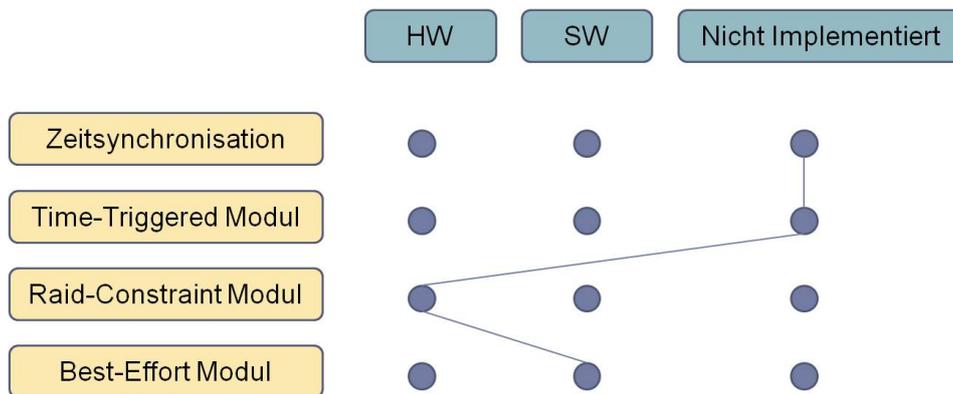


Abbildung 3.1: Partitionierungsprofil ESP

### 3.1.2 Beispiel 2: Multimediasystem

Im zweiten Beispiel geht es um die Steuerung eines Multimediasystems. Angenommen ein Multimediasystem muss zeitkritische, aber nicht sicherheitsrelevante Echtzeitdaten, z. B. ein Videostream über den Kommunikations-Bus verschicken können. So wäre die Nachrichtenklasse Rate-Constrained für diese Art von Daten die richtige Variante. Das zugehörte Partitionierungsprofil wird in Abbildung 3.2 dargestellt. In diesem Profil wird das Zeitsynchronisation- und Time-Triggered Modul nicht implementiert, da keine sicherheitskritischen Daten übertragen werden müssen. Das Best-Effort Modul wird in Software realisiert, um ebenfalls Diagnosedaten oder sogar Internetdaten übertragen zu können.



**Abbildung 3.2:** Partitionierungsprofil Multimediasystem

### 3.1.3 Entscheidungskriterien für die Partitionierung

Die Entscheidung welches Modul in Hardware oder in Software implementiert werden soll wird im Folgenden an den Modulen *Zeitsynchronisation* und *Time-Triggered Modul* erklärt.

**Zeitsynchronisation** Eine Implementierung der Zeitsynchronisation in Hardware führt dazu, dass die Synchronisation genauer vorgenommen werden kann, da diese nicht von jitter-behafteten Interrupts beeinflusst wird, wie es bei einer Softwarelösung der Fall wäre. Auf der anderen Seite würde eine Softwareimplementierung parallele Logik-Ressourcen auf dem FPGA oder ASIC sparen.

**Time-Triggered Modul** Eine Hardware-Implementierung würde hier dazu führen, dass Zeitpunkte, an denen Pakete versendet werden sollen, genauer getroffen werden, da bei einer Softwarelösung ebenfalls jitterbehaftete Interrupts zum Einsatz kommen würden. In Hardware ist hier ein geringer Jitter zu erwarten, was dazu führen würde, dass bei Empfängern kleinere Empfangsfenster eingestellt werden können und der Schedule enger gepackt werden kann. Hier würde eine Softwarelösung ebenfalls dazu führen, dass Logik-Ressourcen eingespart werden.

Welches Modul nun in Hardware oder in Software partitioniert hängt von den Anforderungen des Netzwerks und von der Größe der Anwendung ab. Verlangt ein Netzwerk einen besonders kleinen Jitter im Netzwerk, so spricht das für eine Hardware-Lastige Partitionierung. Wird die CPU von einer Anwendung nur kaum belastet, so könnte sich eine Software-Lastige Partitionierung lohnen, weil dabei parallele Logik-Ressourcen gespart werden und somit ein kleinerer günstigerer FPGA oder ASIC verwendet werden kann.

Am Ende muss an den Anforderungen und an den Kosten die richtige Partitionierung abgewogen werden.

# 4 Ausblick und Zusammenfassung

## 4.1 Ausblick auf weiteres Vorgehen

Im weiteren Vorgehen werden verwandte Arbeiten recherchiert und deren Ergebnisse durchleuchtet. Es wird dabei geprüft, ob bereits gewonnene Erfahrungen in meine Arbeit mit einfließen können.

Auf der praktischen Seite werden Entwicklungsmethodiken analysiert, wie z. B. SystemC, Matlab/Simulink oder ISE Design Suit. Hieraus wird die für diese Arbeit optimale Methode dann ausgewählt.

Außerdem wird nach der optimalen Entwicklungsplattform recherchiert (Xilinx / Altera) und nach Hardware-Bibliotheken, die für diese Arbeit hilfreich sein könnten.

Ein großes Risiko in diesem Projekt ist, dass zum aktuellen Zeitpunkt der Aufwand für diese Arbeit schwer abschätzbar ist. Ein weiteres Risiko ist, dass beim Einsatz von innovativen Entwicklungsmethoden, wie SystemC, es kaum Tool-Unterstützung gibt, diese jedoch optimal für die Entwicklung von interagerter paralleler- und sequenzieller Logik ist.

## 4.2 Zusammenfassung

In dieser Arbeit wurde ein Entwicklungskonzept für Automobil-Steuergeräte eingeführt. Dabei ging es darum Steuergeräte gekoppelt an eine Kommunikationstechnologie in Hardware/Software Co-Design zu entwickeln. Dies bedeutet, dass die Logik in parallele und sequenzielle Logik aufgeteilt wird.

Es soll in Zukunft für jeden Anwendungsfall eine optimale Partitionierung der Module möglich sein, sodass Steuergeräte in Zukunft höchste Anforderungen mit geringem Ressourcenbedarf erfüllen können. Die Aufteilung der Logik in sequenzielle und parallele Logik ist dabei vielversprechend, wofür in dieser Arbeit erste Grundsteine und Überlegungen für die zukünftigen Arbeiten getätigt werden.

# Abkürzungsverzeichnis

<i>μC</i>	Mikrocontroller
<i>AFDX</i>	Avionics Full Duplex Switched Ethernet
<i>ASIC</i>	Application-specific integrated circuit
<i>BE</i>	Best-Effort
<i>CM</i>	Compression Master
<i>CoRE</i>	Communication over Real-time Ethernet
<i>CPU</i>	Central processing unit
<i>ES</i>	Endsystem
<i>ESP</i>	Elektronisches Stabilitätsprogramm
<i>FPGA</i>	Field Programmable Gate Array
<i>HW</i>	Hardware
<i>LIN</i>	Local Interconnect Network
<i>MOST</i>	Media Oriented Systems Transport
<i>NAFTA</i>	North American Free Trade Agreement
<i>PCF</i>	Protocol Control Frame
<i>RC</i>	Rate-Constrained
<i>SM</i>	Synchronisation Master
<i>SW</i>	Software
<i>TT</i>	Time-Triggered
<i>TTEthernet</i>	Time-Triggered Ethernet
<i>VHDL</i>	Very High Speed Integrated Circuit Hardware Description Language

# Literaturverzeichnis

- [AIM GmbH ] AIM GMBH: *Avionics Databus Solutions*. – URL <http://www.afdx.com/>. – Zugriffsdatum: 2010-02-24
- [Badstübner 2008] BADSTÜBNER, Jens: Kollaps im Bordnetz: Schluss mit Can, Lin und Flexray. In: *KFZ-Betrieb* (2008), Nr. 17, S. 68–70
- [Bartols 2010] BARTOLS, Florian: *Leistungsmessung von Time-Triggered Ethernet Komponenten unter harten Echtzeitbedingungen mithilfe modifizierter Linux-Treiber*. Hamburg, HAW Hamburg, Bachelorthesis, Juli 2010. – Bachelorthesis
- [Bruckmeier 2010] BRUCKMEIER, Robert: *Ethernet for Automotive Applications*. Vortrag. Juni 2010. – URL [http://www.freescale.com/files/ftf\\_2010/Americas/WBnr\\_FTF10\\_AUT\\_F0558.pdf](http://www.freescale.com/files/ftf_2010/Americas/WBnr_FTF10_AUT_F0558.pdf). – Zugriffsdatum: 2012-02-02
- [Böke 2008] BÖKE, Dr. C.: FlexRay-Entwicklungen routiniert im Griff. In: *EE Select Automotive* (2008)
- [Daimler AG 2002] DAIMLER AG: Der neue Maybach. In: *ATZ/MTZ Sonderheft* (2002), S. 125
- [FlexRay Consortium ] FLEXRAY CONSORTIUM: *FlexRay*. – URL <http://flexray.com/>. – Zugriffsdatum: 2011-02-07
- [LIN-Administration ] LIN-ADMINISTRATION: *Local Interconnect Network*. – URL <http://www.lin-subbus.org/>. – Zugriffsdatum: 2011-01-06
- [Michael Hübner 2009] MICHAEL HÜBNER, Jürgen B.: *Hardware-Software Codesign*. Folien. Oktober 2009. – URL [http://jegyzetkalauz.hu/5\\_Szemeszter/villany/hard-software\\_Codesign/HSC\\_WS09\\_Kap\\_1\\_Einfuehrung\\_V1.pdf](http://jegyzetkalauz.hu/5_Szemeszter/villany/hard-software_Codesign/HSC_WS09_Kap_1_Einfuehrung_V1.pdf). – Zugriffsdatum: 2012-02-02
- [MOST Cooperation ] MOST COOPERATION: *Media Oriented Systems Transport*. – URL <http://www.mostcooperation.com/>. – Zugriffsdatum: 2011-01-06
- [Müller 2011] MÜLLER, Kai: *Time-Triggered Ethernet für eingebettete Systeme: Design, Umsetzung und Validierung einer echtzeitfähigen Netzwerkstack-Architektur*. 2011

- 
- [SAE - AS-2D Time Triggered Systems and Architecture Committee 2009] SAE - AS-2D TIME TRIGGERED SYSTEMS AND ARCHITECTURE COMMITTEE: *Time-Triggered Ethernet (AS 6802)*. 2009. – URL <http://www.sae.org>. – Zugriffsdatum: 2010-12-11
- [Steinbach u. a. 2010] STEINBACH, Till ; KORF, Franz ; SCHMIDT, Thomas C.: Comparing Time-Triggered Ethernet with FlexRay: An Evaluation of Competing Approaches to Real-time for In-Vehicle Networks. In: *8th IEEE Intern. Workshop on Factory Communication Systems*. Piscataway, New Jersey : IEEE Press, Mai 2010, S. 199–202
- [TTTech Computertechnik AG] TTTECH COMPUTERTECHNIK AG: . – URL <http://www.tttech.com>. – Zugriffsdatum: 2011-01-17