Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Projektbericht - WiSe 09/10

Till Steinbach

Eine Plattform für die eventbasierte Simulation von time-triggered Ethernet Netzwerken

Fakultät Technik und Informatik
Department Informatik

Till Steinbach
Projektbericht - WiSe 09/10
Eine Plattform für die eventbasierte Simulation von time-triggered Ethernet Netzwerken

Kurzzusammenfassung
Eine entscheidende Grundlage für die Bewertung von Ethernet-basierten Vermittungsinfrastrukturen im KFZ bilden Metriken. Auf ihrer Basis können Abschätzungen zum Erfolg im technologischen wie wirtschaftlichen Bereich durchgeführt werden. Mit einer simulationsbasierten Evaluationsstrategie lassen sich neue Konzepte für Vermittlungsinfrastrukturen zuverlässig validieren. Dieser Bericht beschreibt die Arbeiten, die während der zweiten Projektphase des Masterprojektes im Bereich Planung, Erstellung und Überprüfung einer Simulationsumgebung durchgeführt wurden. Dieser Bericht richtet sich an Leser, die bereits mit der Thematik vertraut sind. Dokumente für einen Einstieg in die Thematik sind an verschiedenen Stellen referenziert.

Betreuende Prüfer
Prof. Dr. Franz Korf, Prof. Dr. Thomas Schmidt

Inhaltsverzeichnis

Kapitel 1

Einleitung

Dieser Projektbericht stellt die Projektplanung und Ergebnisse des zweisemestrigen Masterprojektes nach dem zweiten Semster vor.

1.1 Projektplanung und Zielsetzung

Nachdem im ersten Projektsemester im Wesentlichen ein mathematisch-analytisches Framework zur Evaluierung und zum Vergleich der time-triggered Ethernet-Technologie mit dem FlexRay Feldbus (vgl. FlexRay Consortium, 2005) erstellt wurde, fokussiert das zweite Projektsemester auf die Erstellung einer Simulationsplattform für die simulationsbasierte Evaluierung.
Die eventbasierte Simulation soll dabei die Analyseergebnisse bestätigen und insbesondere für umfangreiche Topologien, in denen das analytische Modell sehr groß wird, eingesetzt werden.
Das analytische Modell berücksichtigt bisher allein den Einfluss von time-triggered Kommunikation. In der Simulationsumgebung sollen darüber hinaus alle drei Nachrichtenklassen von TTEthernet (vgl. Steiner, 2008) — *time-triggered*, *rate-constrained* und *best-effort* — implementiert werden.
Simulation und analytische Modelle sollten sich bei der Evaluierung von Kommunikationsnetzwerken ergänzen. Während mit der Simulation besser das reale Verhalten abgebildet werden kann, da sie meist besser in der Lage ist, komplexere Abläufe nachzustellen, liefert das analytische Modell die für den Echtzeit-Bereich besonders relevanten Randfälle (vgl. Navet u. a., 2010). Es ist schwierig, in der Simulation genau die Konstellationen zu treffen, welche z.B. zur maximalen Latenz führen. Egal wie lange simuliert wird, es besteht stets die Gefahr, dass die Simulation Situationen nicht abdeckt, die zu unvorhergesehenen Ergebnissen führen. Durch genaues Planen der Simulation — insbesondere durch das Einbeziehen von Erkenntnissen aus dem analytischen Modell — kann das Risiko der schlechten Simulationsabdeckung zwar reduziert, aber nicht vollständig eliminiert werden.

1.2 Problemstellung

Eine entscheidende Grundlage für die Bewertung von Ethernet-basierten Vermittungsinfrastrukturen im KFZ bilden Metriken. Auf ihrer Basis können Abschätzungen zum Erfolg im technologischen wie wirtschaftlichen Bereich durchgeführt werden. Metriken bilden die Grundlage für eine objektive Sichtweise auf die Technologie. Für Verbesserungen sowohl im Protokoll als auch in der Topologie bilden sie eine objektive Grundlage.
Mit einer simulationsbasierten Evaluationsstrategie lassen sich neue Konzepte für Vermittlungsinfrastrukturen zuverlässig validieren. Die Vorteile dieses simulationsbasierten Ansatzes liegen dabei in der kurzen Einrichtungszeit der Simulationsumgebung und der deutlich wirtschaftlicheren Durchführung, insbesondere bei großen Infrastrukturen, gegenüber Tests auf echter Hardware. Effekte, die durch unterschiedliche Parametrisierung und umfangreiche Topologien entstehen, lassen sich auf echter Hardware nur mit hohen Kosten und großem Zeiteinsatz nachweisen.
Simulationsbasierte Evaluationsstrategien für zukünftige Vermittlungsinfrastrukturen im KFZ können die Entwicklung unterstützen. So lassen sich mit den Werkzeugen der Simulation bereits in einem frühen Entwicklungsstadium spätere Eigenschaften überprüfen und sichtbar machen.

1.3 Rückblick Masterprojekt 1. Semester

Das erste Semester des Masterprojektes hatte drei Ziele: Die Erstellung eines analytischen Modells zum Vergleich von Time-Triggered Ethernet und FlexRay, die Vorbereitungen für die Implementierung in der eventbasierten Simulation sowie die Analyse der TTEthernet-Toolchain von TTTech.
Mit Hilfe des analytischen Modells konnte auf Basis mathematischer Methodik gezeigt werden, dass es möglich ist, ein System mit einem vollständig ausgelasteten FlexRay-Bus in ein System, welches auf Time-Triggered Ethernet basiert, zu überführen. Die Arbeit zum Vergleich von TTEthernet und FlexRay wurde gesondert veröffentlicht (vgl. Steinbach u. a., 2010). Das analytische Modell kann auf verschiedene Topologien angewendet werden und soll zum Vergleich mit den Ergebnissen aus der Simulation verwendet werden.
Als Vorbereitung auf die in diesem Semester angestrebte Implementierung wurde im 1. Semester des Masterprojektes die OMNeT++-Simulationsumgebung (vgl. OMNeT++ Community, b) ausgewählt. Als Framework wurde das INET-Framework (vgl. OMNeT++ Community, a) für OMNeT++ ausgewählt und analysiert. Das INET-Framework bietet bereits weite Teile der Implementierung des OSI-Modells und wird die Basis für die Implementierung bilden.
Die Analyse der TTEthernet Toolchain bildet die Vorarbeit für die Konfiguration der Simulation. Um möglichst effizient simulieren zu können, sollen für die Einrichtung der Simulation

die selben Werkzeuge zum Einsatz kommen, welche auch für die Konfiguration der realen Hardware verwendet werden.
Die Ergebnisse des ersten Teils des Masterprojektes können detailliert im 1. Projektbericht nachgelesen werden (vgl. Steinbach, 2009).

1.4 Projektplanung und Zielsetzung

Das Masterprojekt ist über zwei Semester organisiert. Das Gesamtziel über beide Semester ist, eine Bewertungsgrundlage für die Time-Triggered Ethernet Technologie zu schaffen, auf deren Basis weitere Evaluationen aufbauen können. Für das zweite Semester sind die folgenden Tätigkeiten geplant:

1.4.1 Entwurf und Implementierung der TTEthernet-Komponenten in der eventbasierten Simulation

Die Implementierung der TTEthernet-Komponenten und des Protokolls sind der umfangreichste Teil des zweiten Masterprojektes. Es ist eine Modulare Implementierung geplant, welche unabhängig getestet werden kann. Dabei soll zunächst nur die Funktionalität hergestellt werden, welche TTEthernet benötigt. Es soll zunächst nur das TTEthernet-Protokoll selbst realisiert werden und erst in einem späteren Schritt das Synchronisierungskonzept. Da in der Simulation eine globale Zeit zur Verfügung steht, kann die Synchronisierung vorerst emuliert werden.

1.4.2 Konfiguration der Simulation auf Basis des TTTech-Konfigurationsformates

Damit die Simulation mit den vorhandenen Werkzeugen von TTTech konfiguriert werden kann, soll die Simulation auf dem TTTech-Konfigurationsformat aufsetzen. Damit ist es direkt möglich, reale Netzwerkkonfigurationen in der Simulation zu verwenden. Das Konfigurationsformat ist XML basiert und verwendet das Ecore-Metamodell (vgl. Eclipse Foundation).

1.4.3 Vergleich der Simulation mit analytischen Ergebnissen

Die Ergebnisse der Simulation sollen mit den analytischen Ergebnissen aus dem ersten Projektteil (vgl. Steinbach u. a., 2010) verglichen werden. Dafür wird die Topologie der analytischen Vergleichsarbeit in der Simulation nachgestellt. Da für die Bandbreitenmessungen sehr große Schedules erstellt werden müssen, ist zunächst der Vergleich der Latenz zwischen analytischem Modell und Simulation geplant. Mit den geeigneten Konfigurationswerk-

zeugen lassen sich dann auch umfangreiche Schedules simulieren, um die Bandbreite zu messen.

1.4.4 Vergleich der Simulation mit Hardwaremessungen

Inzwischen wurde das TTEthernet-Evaluationssystem an der HAW in Betrieb genommen. Der Bachelor Student Florian Bartols entwickelt eine Messmethode, mit der der TTEthernet-Switch mit Standardkomponenten und einem Realtime-Linux-Kernel ausgemessen werden kann. Durch Simulation des Versuchsaufbaus sollen die Messergebnisse verglichen und überprüft werden. Damit kann die Genauigkeit des Verfahrens besser eingeschätzt werden.

Kapitel 2

Die Simulationsplattform

In diesem Abschnitt wird kurz die der Implementierung zugrundeliegende Simulationsplattform, bestehend aus OMNeT++ (OMNeT++ Community, b) und dem INET-Framework (OMNeT++ Community, a) vorgestellt. Weitere Informationen zur Plattform können dem 1. Projektbericht (vgl. Steinbach, 2009) entnommen werden.

Omnet++ ist eine Simulationsplattform für die eventbasierte Simulation. Insbesondere im Bereich der Simulation von Netzwerktechnologien hat sich um OMNeT++ eine Community gebildet.
OMNeT++ ist quelloffen und für den akademischen Einsatz kostenlos. Die Komponenten sind in C++ entwickelt. Mit Version 4 ist OMNeT++ vollständig in die Eclipse-Plattform integriert und bietet umfangreiche Werkzeuge für die Konfiguration und Auswertung von Simulationen. Die während der Implementierung eingesetzte Version 4.0 enthielt noch einige Fehler in den Konfigurations- und Auswertungsmodulen. Die meisten dieser Fehler wurden in der kürzlich erschienenen Version 4.1 jedoch behoben.
Die Simulation mit diskreten Events ist für Netzwerke gut geeignet, weil die Paketweiterleitung einfach in Ereignisse überführt werden kann. Für die Simulation interessieren nur wenige Zeitpunkte, wie das Versenden oder Empfangen eines Paketes. Dies reduziert die Zustände im System.

Das INET-Framework ist eine Erweiterung von OMNeT++, welche der Simulationsplattform Funktionalität im Bereich der Netzwerkprotokolle zur Verfügung stellt.
Das INET-Framework bietet Module vom physikalischen Layer bis zur Applikationsschicht des OSI-Modells. Für die Implementierung der TTEthernet-Komponenten sind besonders die Elemente des Layer 2 interessant. Als Grundlage für die TTEthernet-Switchimplementierung dient der INET-Switch. Aber auch auf dem physikalischen Layer stellt das Framework Komponenten zur Verfügung, welche übernommen werden können, darunter z.B. das Kabelmodell, welches bereits die Konfiguration von Kabellängen und Verlustraten erlaubt.

Kapitel 3

Architektur

In diesem Abschnitt wird die Architektur der Implementierung vorgestellt.

3.1 Framework-Komponenten

Das Framework besteht aus drei übereinander liegenden Komponenten (siehe Abb. 3.1). **OMNeT++** bildet als Werkzeug für Event basierte Simulationen die Basis für das Simulationsframework von TTEthernet. Das **INET-Framework** für OMNeT++ liefert Standardkomponenten für die Layer 1-3. Auf Basis des INET-Framework wird die **TTEthernet-Implementierung** aufgesetzt. Sie enthält die speziellen TTEthernet-Komponenten.
Zum vollständigen Framework gehören zudem die Konfigurationswerkzeuge für die Simulation. Hier werden, sofern verfügbar, die Tools der TTEthernet-Toolchain von TTTech (vgl. TTTech Computertechnik AG) eingesetzt.

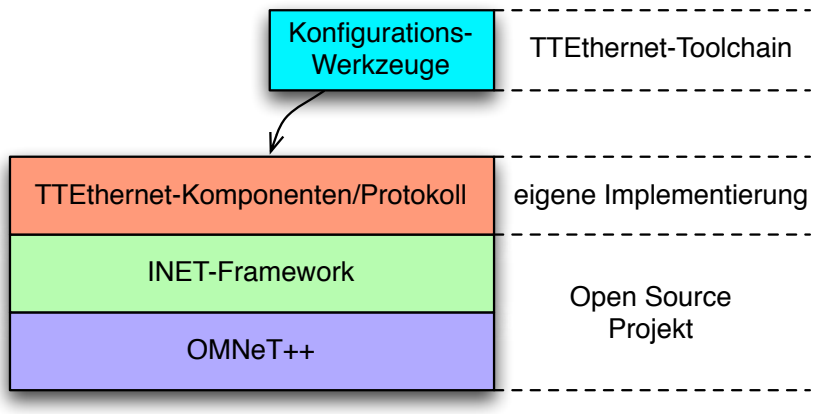

Abbildung 3.1: Komponenten der Simulation

3.2 Module

Die Architektur für die TTEthernet-Komponenten in der Simulation sieht eine starke Modularisierung vor. Die Komponenten von TTEthernet — verschiedene Switche und Hosts — verfügen über gemeinsame und gerätespezifische Funktionalität. Üblicherweise wird in OMNeT++ Vererbung verwendet, um Funktionalität in verschiedenen Implementierungen gemeinsam zu verwenden. Für die Implementierung von TTEthernet ist dieses Vorgehen jedoch nicht sinnvoll, da es zu Mehrfachvererbungen führen würde. Die hier vorgestellte Architektur setzt dagegen auf wiederverwendbare Submodule, welche über definierte Interfaces kommunizieren und vollständig gekapselt sind (siehe Abbildung 3.2). Die Hauptlogik, welche für Switch und Host gleich ist, ist im TTELogic-Modul implementiert. Das Aktivitätsdiagramm (siehe Abbildung 3.3) zeigt vereinfacht die Abläufe, die im Logic-Modul implementiert sind. Das TTETimer-Modul implementiert den Takt des Gerätes in der gewünschten Präzision. Details zur Implementierung des Taktes sind in Abschnitt 4.1.1 beschrieben.

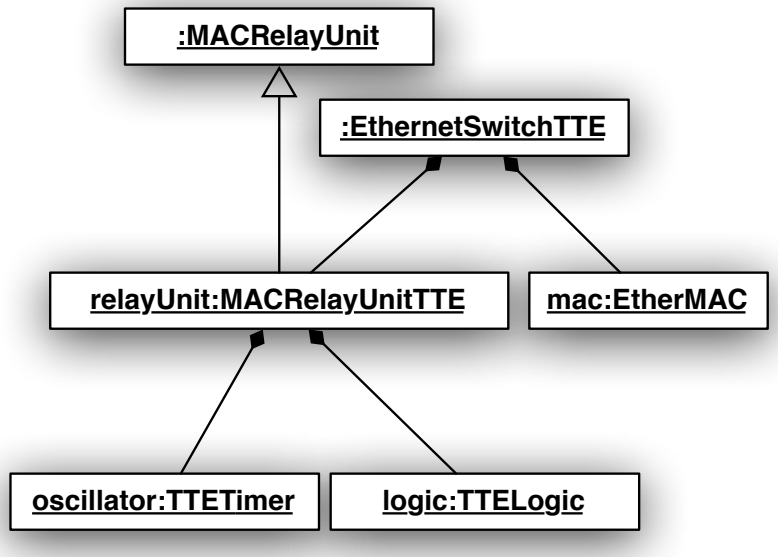

Abbildung 3.2: Klassendiagramm der TTEthernet-Switchimplementierung

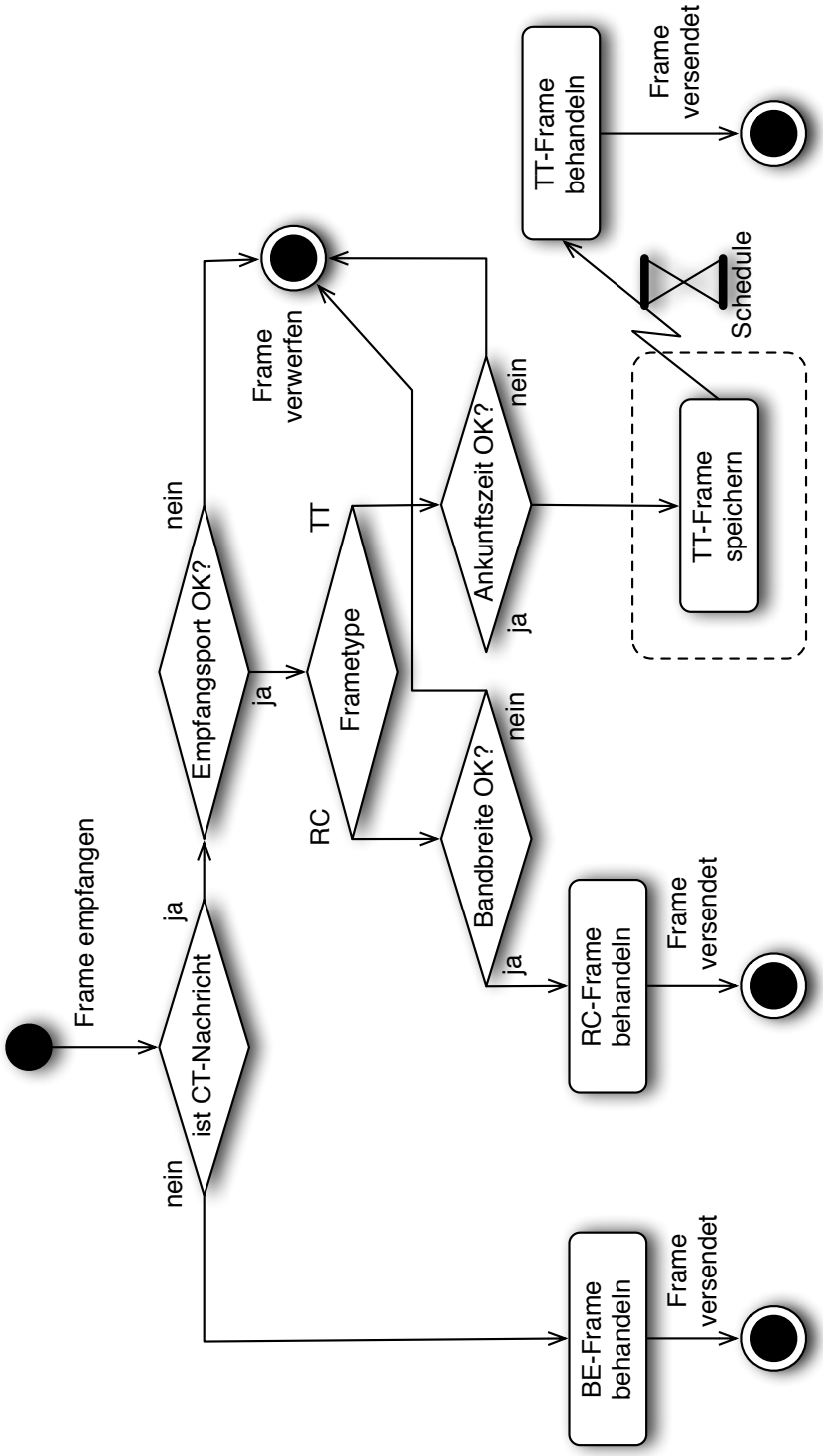

Abbildung 3.3: Aktivitätsdiagramm der TTEthernet-Switchimplementierung

Kapitel 4

Umsetzung und Ergebnisse

Dieser Abschnitt stellt den implementierten Prototyp und erste Simulationsergebnisse vor.

4.1 Prototyp

Auf Basis der vorgestellten Architektur (siehe Abschnitt 3) wurde der TTEthernet-Switch und ein Prototyp des TTEthernet-Hosts für OMNeT++ implementiert. Abbildung 4.1 zeigt die Simulationsanwendung. Eine besondere Herausforderung bestand dabei in der Implementierung des Systemtaktes und der Switch-Konfiguration.

4.1.1 Systemtakt

Da time-triggered Ethernet auf der Synchronisation der Teilnehmer beruht, ist es wichtig, die Taktabweichungen der einzelnen Teilnehmer simulieren zu können. Jedes Gerät im TTEthernet Netzwerk — Switche und Hosts — hat daher seine eigene Systemzeit. Eine globale Zeit wird über das Synchronisationsprotokoll hergestellt. Um in der Simulation den Einfluss der Taktungenauigkeiten jedes Teilnehmers zu berücksichtigen, muss der Systemtakt mit simuliert werden.
Die naheliegende Lösung für die Simulation des Systemtaktes ist, den Takt als ein Modul zu integrieren und damit jeden Schlag des Timers als ein Event zu betrachten. Diese Implementierung ist am genauesten in der Abbildung des Systemtaktes. Eine kurze Überschlagsrechnung zeigt jedoch, dass dies zu einer sehr großen Anzahl von Events führt, die in der Simulation abgearbeitet werden müssen. Bei einem Systemtakt von wenigen Megahertz ergeben sich schon mehrere Millionen Events pro Sekunde pro Gerät. Dies verlangsamt die Simulation erheblich.
Bei der Implementierung einer Simulation muss stets auf den Konflikt zwischen einem genauen Ergebnis und einer langen Simulationszeit geachtet werden. Je präziser ein Objekt

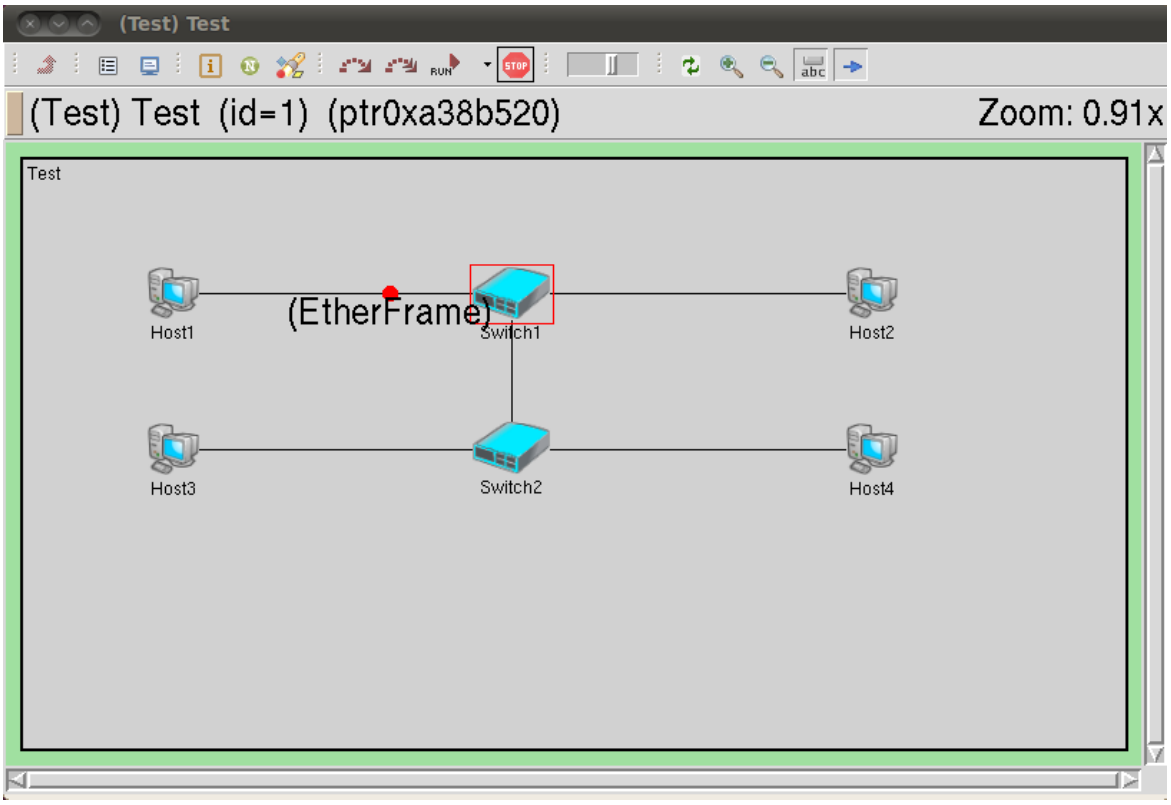

Abbildung 4.1: Screenshot der Simulation

simuliert wird, desto länger braucht üblicherweise die Simulation. Eine grobe Simulation des Objektes kann dagegen meist mit einer kurzen Simulationszeit erreicht werden.
Um detaillierte Aussagen über das Synchronisationsverhalten machen zu können, wird eine präzise Simulation des Systemtaktes benötigt. Da das vorrangige Ziel dieses Projektes jedoch die Simulation des Gesamtsystemverhaltens ist, ist eine weniger genaue, jedoch deutlich performantere Implementierung des Taktes akzeptabel.
Für eine performante Implementierung des Taktes wird nicht mehr jeder Takt einzeln simuliert, sondern eine konfigurierbare Anzahl gleichzeitig. Die Länge eines Taktes ist über diese Zeitspanne konstant. Im nächsten Intervall kann die Taktlänge dann verändert werden. Solange das Intervall kleiner als der Nachrichtenzyklus ist, ist das Ergebnis zwischen der genauen und der performanten Implementierung nahezu gleich.
Durch die modulare Architektur (siehe Abschnitt 3.2) kann die Timer-Komponente stets ausgetauscht werden. Der Nutzer der Simulation kann sich also entscheiden, ob er eine genaue oder performante Simulation des Taktes benötigt.

4.1.2 Konfiguration

Ein zentraler Aspekt bei der Implementierung des Prototyps war die Konfiguration. Die Konfiguration einer TTEthernet Komponente ist deutlich umfangreicher als die Konfiguration eines einfachen Ethernet-Switches. Es muss neben allgemeinen Parametern der komplette Schedule eingetragen werden. TTTech hat hierfür ein eigenes XML-Schema entwickelt, was auf einer abstrakten Ebene die Konfiguration definiert. Daraus werden dann für die einzelnen Komponenten Konfigurationsdateien abgeleitet.
Um die Werkzeuge von TTTech bei der Konfiguration der Simulation verwenden zu können und um simulierte Konfigurationen später direkt auf echter Hardware einzusetzen, sollte die Simulation auf dem TTTech XML-Schema aufsetzen. Leider stand zu beginn der Implementierung das XML-Schema noch nicht zur Verfügung, so dass die Implementierung mit einer provisorischen eigenen XML-Konfiguration begonnen wurde. Inzwischen liegt an der HAW eine erste Version des XML-Schemas vor, und es konnte damit begonnen werden, die provisorische Konfiguration gegen das TTTech-Schema auszutauschen. Da das Schema jedoch noch nicht vollständig definiert ist und es noch einige Unklarheiten bezüglich der Definition gibt, welche im Kontakt mit TTTech bisher noch nicht vollständig ausgeräumt werden konnten, ist der Umstellungsprozess bisher noch nicht komplett abgeschlossen.
Für einfache XML-Konfigurationen bietet OMNeT++ bereits ein XML-Framework an. Da das Format aber komplexe Strukturen wie z.B. Verweise verwendet, wurden der Xerces-C++ XML-Parser (vgl. Apache Software Foundation, b) und der Xalan-C++ XLST-Processor (vgl. Apache Software Foundation, a) des Apache-Projektes für die XML-Verarbeitung verwendet. Eine beispielhafte Konfigurationsdatei ist in Listing 4.1 gezeigt.

Listing 4.1: Device-Konfiguration in XML

```
<ds:Switch xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.tttech.com/Schema/TTEthernet/Device_Specification_V1"
    xmlns:sys="http://www.tttech.com/Schema/TTEthernet/System_Specification_V0"
        xmlns:ds="http://www.tttech.com/Schema/TTEthernet/Device_Specification_V1"
        name="DS_SW_sw0_0" refDeviceMapping="Net1.device_target_mapping#//@deviceMappings/@deviceMapping[name= DTM_sw0_0 ]"
        signature="">
        <ds:config ctMarker="03:04:05:06" ctMask="ff:ff:ff:ff" name="sw0"/>
        <ds:bagAccounts>
                <ds:incomingBagAccount bag="100us" jitter="100us" name="incomingRC1"/>
                <ds:outgoingBagAccount bag="100us" jitter="100us" name="outgoingRC1"/>
        </ds:bagAccounts>
        <ds:vlSchedules name="Schedule1" scheduleStartAfterSync="0us">
                <ds:vlSchedule
                        refVirtualLink="sys:TTVirtualLink␣Test.system_specification#//@virtualLinks/@virtualLink[vlid= VL_TT_0 ]">
                        <ds:incoming bufferDepth="TBD?" receiveWindowEnd="1100us" receiveWindowStart="1000us"
                                xsi:type="ds:TTIncoming">
                                <ds:refInPort>
                                        Test.system_specification#//@devices/@device[name= sw0_0 ]/@port[name= P_sw0_0_P0 ]
                                </ds:refInPort>
                                <ds:outgoing refPeriod="P_1_1000" sendWindowEnd="1100us" sendWindowStart="1100us"
                                        xsi:type="ds:TTOutgoing">
                                        <ds:refOutPort>
                                                Test.system_specification#//@devices/@device[name= sw0_0 ]/@port[name=
                                                        P_sw0_0_P1 ]
                                        </ds:refOutPort>
                                        <ds:refOutPort>
                                                Test.system_specification#//@devices/@device[name= sw0_0 ]/@port[name=
                                                        P_sw0_0_P2 ]
                                        </ds:refOutPort>
                                </ds:outgoing>
                        </ds:incoming>
                </ds:vlSchedule>
                <ds:vlSchedule
                        refVirtualLink="sys:RCVirtualLink␣Test.system_specification#//@virtualLinks/@virtualLink[vlid= VL_RC_0 ]">
                        <ds:incoming bufferDepth="TBD?"
                                refBagAccount="#//@BagAccounts/@incomingBagAccount[name=incomingRC1]"
                                xsi:type="ds:RCIncoming">
                                <ds:refInPort>
                                        Test.system_specification#//@devices/@device[name= sw0_0 ]/@port[name= P_sw0_0_P0 ]
                                </ds:refInPort>
                                <ds:outgoing refBagAccount="#//@BagAccounts/@outgoingBagAccount[name=outgoingRC1]"
                                        refOutPort="TBD?"
                                        xsi:type="ds:RCOutgoing">
                                        <ds:refOutPort>
                                                Test.system_specification#//@devices/@device[name= sw0_0 ]/@port[name=
                                                        P_sw0_0_P2 ]
                                        </ds:refOutPort>
                                </ds:outgoing>
                        </ds:incoming>
                </ds:vlSchedule>
        </ds:vlSchedules>
        <ds:bestEffortRouting/>
</ds:Switch>
```

4.2 Simulationsergebnisse

4.2.1 Allgemein

Bei den bisher durchgeführten Tests zeigt sich bereits, dass der Prototyp das gewünschte Verhalten implementiert. Das statische Routing der time-triggered Pakete funktioniert zuverlässig. Beim parallelen Versenden von Nachrichten auf mehreren Ports entsteht keine zeitliche Differenz — die Nachrichten werden zeitgleich versendet. Time-triggered Nachrichten, die außerhalb ihres Zeitfensters oder auf einem falschen Port eintreffen, werden abgewiesen. Bereits in den ersten Tests zeigte sich, dass in der Simulation ein Fehler in der Konfiguration äußerst schnell gefunden werden kann. Dazu tragen die Fehlermeldungen der Simulation

bei, welche auf der Konsole ausgegeben werden und dem Benutzer helfen, Fehler wie eine falsche Belegung der Ports oder ein falsch berechneter Zeitplan in der Konfiguration zu finden.

4.2.2 Vergleich mit analytischen Ergebnissen

Bereits im vergangenen Semester wurde ein analytisches Framework erstellt, mit dem sich TTEthernet-Metriken berechnen lassen. Das Framework wurde bereits eingesetzt, um ausgewählte Metriken von TTEthernet mit denen von FlexRay zu vergleichen (vgl. Steinbach u. a., 2010). Dieses Framework ist ebenfalls geeignet, um die Ergebnisse der Simulation zu überprüfen und zu bewerten. Dazu wurde sowohl die im Vergleich verwendete Topologie als auch eine Topologie mit nur einem Switch simuliert. Tabelle 4.1 stellt die Ergebnisse dar. Die Werte für die analytische Rechnung sind gerundet, da bei der Berechnung das Delay für den TTEthernet-Switch nur Mikrosekunden genau vorlag. Die Werte sind damit als nahezu gleich anzusehen.

Tabelle 4.1: Vergleich der Latenz zwischen Simulation und analytischem Modell

	Simulation	**Modell**
Latenz min. Framegröße	$23{,}8\mu s$	$24\mu s$
Latenz max. Framegröße	$371{,}7\mu s$	$372\mu s$

Hier wurde bisher noch nicht die Bandbreite im analytischen Modell mit der Simulation verglichen, dies wird nachgeholt, sobald die Werkzeuge zur Schedule-Erstellung vorliegen. Ohne die Werkzeuge ist die Erstellung solch großer Schedules aufwendig. Der Jitter wurde noch nicht verglichen, da der Einfluss des Synchronisationsprotokolls in der Simulation bisher noch fehlt und daher ein sinnvoller Vergleich noch nicht möglich ist. Dies wird mit der späteren Implementierung des Synchronisationsprotokolls durchgeführt.

4.2.3 Vergleich mit Hardware-Messungen

Der Student Florian Bartols entwickelt derzeit im Rahmen seiner Bachelorarbeit ein Messverfahren, welches auf Zeitstempeln im Echtzeitlinux-Treiber basiert, um die Verzögerung im TTEthernet-Switch zu messen (vgl. Bartols, 2010). Erste Vergleiche zwischen der Simulation und den Messergebnissen zeigen eine minimale Abweichung von wenigen zehntel Mikrosekunden (vgl. Tabelle 4.2). Die simulierte Latenz ist dabei leicht höher als die gemessene Latenz. Diese geringe Abweichung lässt sich auf die Messungenauigkeit des Echtzeit-Linux-basierten Verfahrens zurückführen.
Gemessen wurde auf einer Topologie mit einem Switch. Der Schedule enthält ein Delay von $350\mu s$.

Tabelle 4.2: Vergleich der Latenz zwischen Simulation und Hardware-Messung

	Simulation	**Messung**
Latenz min. Framegröße	$355{,}4\mu s$	$355\mu s$
Latenz max. Framegröße	$471{,}6\mu s$	$471\mu s$

4.2.4 Performance der Simulation

Die Simulationen wurden auf Ubuntu Linux in einer virtuellen Maschine mit 1GB RAM und 2 Cores durchgeführt. Dabei wurden mit einem einfachen Schedule — abhängig vom Detaillierungsgrad der Aufzeichnung — 2 bis 16 Simulationssekunden pro Echtzeitsekunde erreicht. Dies stimmt optimistisch, dass mit leistungsfähiger Hardware und ohne die Indirektion der virtuellen Maschine eine Simulation in Echtzeit auch von umfangreicheren Topologien möglich ist. Dies ist wichtig, wenn reale Applikationen mit einem simulierten Netzwerk getestet werden sollen, oder das Framework zur Restbussimulation — also zur Simulation von Netzwerkteilen zusammen mit realer Hardware — eingesetzt wird.

Kapitel 5

Fazit

5.1 Zusammenfassung

In diesem Projekt wurde eine Umsetzung des TTEthernet-Protokolls in OMNeT++ geplant, im Prototyp implementiert und evaluiert. In der Planungsphase wurde bereits darauf geachtet, dass Komponenten in verschiedenen Teilen der Simulation wiederverwendbar einzusetzen sind (siehe Abschnitt 3.2). Dies reduzierte den Implementierungsaufwand und hilft zukünftig, den Programmcode zu warten.
Bei der Implementierung musste insbesondere auf eine flexible Umsetzung der Simulation des Systemtaktes geachtet werden. Durch die modulare Architektur der Implementierung ist es mit der erarbeiteten Lösung nun möglich, je nach Bedarf eine genaue Simulation des Systemtaktes oder eine Simulation mit kurzer Simulationszeit durchzuführen (siehe Abschnitt 4.1.1).
Im Bereich der Konfiguration wurde versucht, das Konfigurationsschema von TTTech zu übernehmen. Dies ermöglicht die Verwendung der TTEtherner-Toolchain während der Planung und Konfiguration der Simulation. Das Konfigurationsformat ist noch nicht vollständig in der Simulation implementiert, da es noch nicht in der finalen Version vorliegt.

5.2 Bewertung

Während der Implementierung zeigte sich, dass OMNeT++ mit dem INET-Framework sehr gut geeignet ist, um eine Simulation von TTEthernet umzusetzen. Auf Basis des hier vorgestellten Prototyps können nun weitere Module wie Paketgeneratoren, das Synchronisationsprotokoll oder charakteristische Fahrzeuganwendungen implementiert werden. Durch die modulare Architektur kann der Quellcode aus dem Prototyp an vielen Stellen der Simulation wiederverwendet werden. Zudem kann die Simulation Vorlagen für die Implementierung

von TTEthernet-spezifischen Algorithmen liefern, welche z.B. für die Implementierung des TTEthernet-Stacks benötigt werden.

5.3 Ausblick

Bisher wird die Konfiguration der Simulation nur teilweise aus dem XML-Schema generiert. Zu den Parametern, die bisher genutzt werden, zählen vor allem die Einstellungen der Switche und die Schedules. Im nächsten Schritt soll auch die Topologie, welche in OMNeT++ in sogenannten Network-Description-Files vorliegen muss, aus dem XML-Schema generiert werden.
Zur Auswertung der Simulationsergebnisse legt OMNeT++ Dateien mit den Ergebnissen der Simulationsdurchläufe an. Aus diesen können dann Berichte generiert werden, welche die Ergebnisse aufbereiten. Auch das Erstellen der Berichte soll automatisiert durchführbar sein. Dazu ist es notwendig, dass die Simulationsumgebung Informationen darüber erhält, welche Metriken von Interesse für den Benutzer sind. Es ist geplant, diese Information ebenfalls an das XML-Schema zu binden. Dafür ist es notwendig, eine geeignete Form der Annotation für die Metriken im Schema zu definieren.

Literaturverzeichnis

[Apache Software Foundation a] APACHE SOFTWARE FOUNDATION: *Xalan-C++ XSLT-Processor*. – URL http://xml.apache.org/xalan-c/

[Apache Software Foundation b] APACHE SOFTWARE FOUNDATION: *Xerces-C++ XML-Parser*. – URL http://xerces.apache.org/xerces-c/

[Bartols 2010] BARTOLS, Florian: *Leistungsmessung von Time-Triggered Ethernet Komponenten unter harten Echtzeitbedingungen mithilfe modifizierter Linux-Treiber*. July 2010. – Bachelorarbeit, Bachelorthesis

[Eclipse Foundation] ECLIPSE FOUNDATION: *Eclipse Modeling Framework*. – URL http://www.eclipse.org/modeling/emf/

[FlexRay Consortium 2005] FLEXRAY CONSORTIUM: Protocol Specification / FlexRay Consortium. Stuttgart, Germany, Dec 2005 (2.1). – Specification

[Navet u. a. 2010] NAVET, Nicolas ; MONOT, Aurélien ; MIGGE, Jörn: *Frame latency evaluation: when simulation and analysis alone are not enough*. 8th IEEE International Workshop on Factory Communication Systems - Industry Day. 05 2010

[OMNeT++ Community a] OMNET++ COMMUNITY: *INET Framework for OMNeT++ 4.0*. – URL http://inet.omnetpp.org/

[OMNeT++ Community b] OMNET++ COMMUNITY: *OMNeT++ 4.0*. – URL http://www.omnetpp.org

[Steinbach 2009] STEINBACH, Till: *Ethernet als Bus für Echtzeitanwendungen im Automobil - Projektbericht*. Sep 2009. – URL http://papers.till-steinbach.de/s-ebeap-09.pdf. – Bericht

[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 (WFCS 2010)*. Piscataway, NJ, USA : IEEE Press, May 2010, S. 199–202

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

[TTTech Computertechnik AG] TTTECH COMPUTERTECHNIK AG: . – URL `http://www.tttech.com`