Hochschule für Telekommunikation Leipzig

Echtzeiteigenschaften von Feldbussen am Beispiel des Controller Area Networks

Sebastian Lammermann

Hochschule für Telekommunikation Leipzig

Echtzeiteigenschaften von Feldbussen am Beispiel des Controller Area Networks

Masterarbeit

Verfasser:	Sebastian Lammermann
	Matrikelnummer 074111
Erstprüfer:	Prof. Dr. Michael Meßollen
Zweitprüfer:	Prof. Dr. Thomas Meier
Fertigstellung:	30. Oktober 2009
Kontakt:	sebastian@lammermann.eu

Published under Creative Commons Licence

Diesem Dokument liegt eine CD-ROM bei.

Alternativ finden Sie den Inhalt des Datenträgers auch in Intranet der Hochschule für Telekommunikation Leipzig.

Inhalt

Abbildungsverzeichnis

1 Vorwort und Hinweise

Die vorliegende Masterarbeit zum Thema *„Echtzeiteigenschaften von Feldbussen am Beispiel des Controller Area Networks“* entstand im Rahmen des Lehrgebiets „Echtzeitsysteme“ im Sommersemester 2009 und im Wintersemester 2009/2010 an der Hochschule für Telekommunikation Leipzig.

Abkürzungen werden in dieser Masterarbeit in der Regel mit Artikel, sowie ggf. mit Genitiv-s versehen. Um einen besseren Lesefluss zu ermöglichen, wird eine eingeführte Abbreviatur nicht durchgehend verwendet, sondern stellenweise die ausgeschriebene Variante bevorzugt.

Im vorliegenden Text wird sowohl für die Transkription als auch für die Translation des griechischen Buchstabens 'Φ', außer bei Namen, ausschließlich die offizielle neugriechische Übersetzung 'F' verwendet. Damit wird die Schreibweise des Dudens in einigen Fällen missachtet. Da dieser allerdings 1998 seine Monopolstellung bei der Rechtschreibung verloren hat und nun vielmehr Richtlinien beinhaltet, sei hiermit auf diese Hinwegsetzung aufmerksam gemacht.

2 Einleitung

Ziel dieser Masterarbeit ist die Entwicklung eines Laborversuchs für das Lehrgebiet „Echtzeitsysteme“. Um diesen Versuch einerseits anschaulich gestalten zu können und gleichzeitig die Kosten zu minimieren, wird hierfür auf ein Feldbussystem, das *Controller Area Network* (*CAN*), zurückgegriffen.

Als „Hauptprodukt“ dieser Arbeit wird insbesondere die für die Durchführung des Labors benötigte Versuchsanleitung betrachtet. Diese beinhaltet alle notwendigen Informationen zur Vorbereitung, sowie sechs Teilversuche, die weitgehend aufeinander aufbauen. Die Aufgaben sind so gestaltet, dass der Gesamtversuch von einer Zweiergruppe innerhalb eines halben Tages (ca. vier Stunden) durchgeführt werden kann. Hinzu kommt jeweils ungefähr der gleiche Zeitaufwand für Vor- bzw. Nachbereitung. Für alle Teilversuche gilt, dass der Wissenserwerb der Studierenden im Vordergrund steht und aus diesem Grund auf komplizierte und langwierige Hard- und Softwarekonfigurationen verzichtet wird.

Darüber hinaus bietet das Dokument zusätzliche Informationen zum Versuch, wie Hintergründe, die Lösungen der Versuchsaufgaben oder Testberichte.

Diese Masterarbeit gliedert sich insgesamt in neun Kapitel:

- **Kapitel 3** enthält eine Erläuterung der Arbeiten, die im Vorfeld dieser Thesis durchgeführt worden sind. Des Weiteren beschäftigt sich dieses Kapitel mit der Aufgabenstellung, sowie den selbst definierten Versuchszielen.
- **Kapitel 4** bildet die Anleitung zum entwickelten Laborversuch. Dieses Kapitel stellt den Kern der Masterarbeit dar und wird außerdem als separates Dokument veröffentlicht. Aus diesem Grund besitzt Kapitel 4 beispielsweise eine eigene Einleitung, ein eigenes Vorwort sowie einen Anhang. Den Studierenden soll einzig dieser Teil der Arbeit ausgehändigt werden.
- **Kapitel 5** liefert die Hintergründe zu den einzelnen Versuchen. Hier wird erläutert, warum der jeweilige Versuch gewählt worden ist und welche Ziele mit ihm erreicht werden sollen.
- **Kapitel 6** bietet eine „Musterlösung“ zu allen Laborversuchen. Diese richtet sich v. a. an das Laborpersonal um die von den Versuchsgruppen eingereichten schriftlichen Ausarbeitungen bewerten zu können.
- **Kapitel 7** berichtet von den durchgeführten Testläufen und zeigt, welche Konsequenzen deren Resultate für die Entwicklung der Laboraufgaben hatten.
- **Kapitel 8** verdeutlicht die Probleme, die während dieser Masterarbeit aufgetreten sind, und zeigt deren Lösungen auf.
- **Kapitel 9** enthält das Nachwort dieses Dokuments.

3 Arbeiten im Vorfeld, Aufgabenstellung und Versuchsziele

3.1 Arbeiten im Vorfeld

Bereits im Sommersemester 2008 wurde in Erwägung gezogen für das Lehrgebiet „Echtzeitsysteme“ einen Laborversuch zu entwickeln, um den Studierenden die Materie durch praktische Anwendung näher zu bringen. Dass hierfür ein Feldbussystem angedacht wurde, ergab sich aus dem Umstand, dass diese zum Einen einfach zu errichten sind und darüber hinaus der Datenverkehr, und damit auch die Echtzeiteigenschaft, gut visualisiert werden kann.

Der CAN-Bus wurde gewählt, da er einerseits in der Industrie, insbesondere im Fahrzeugbau und in der Automatisierung, weit verbreitet ist und andererseits durch sein Alter einen ausgereiften und bewährten Standard darstellt. Ferner sind CAN-Komponenten auf dem Markt hoch verfügbar und vergleichsweise günstig.

Um zunächst das nötige Know-How zu erlangen, wurde im Vorfeld dieser Thesis eine Projektarbeit mit dem Titel „Controller Area Network“ verfasst [Lam208]. Diese beleuchtet alle theoretischen Aspekte des CANs, erläutert die Netzwerktopologie, die Funktionsweise des Systems auf den unteren beiden OSI-Schichten und das Buszugriffsverfahren. Des Weiteren werden die Echtzeiteigenschaften der Netzwerktechnologie analysiert.

3.2 Aufgabenstellung

Die Aufgabenstellung dieser Masterarbeit ist bewusst so gestaltet, dass sie Spielräume für eine vergleichsweise freie Gestaltung lässt. Im Vorfeld festgelegt wurde, dass ein Laborversuch für das Lehrgebiet „Echtzeitsysteme“ zu entwerfen ist. Für die Realisierung soll ein Feldbus dienen, wobei das CAN sich, aufgrund der weiten Verbreitung, der preiswerten Komponenten, der gereiften Technologie und der verfügbaren Literatur, hierfür am ehesten anbietet. Die genaue Themenformulierung lautete schließlich „Echtzeiteigenschaften von Feldbussen am Beispiel des Controller Area Networks“, was damit auch dem Titel dieser Arbeit entspricht.

Zwar enthält die Aufgabenstellung keine detaillierten Voraussetzungen für den zu erstellenden Laborversuch, einige Bedingungen werden aber dennoch gestellt. So soll der Umgang mit dem CAN erlernt werden, die Echtzeiteigenschaften des Systems aber im Fokus des Versuchs stehen. Um diese zu verdeutlichen, wird gleichzeitig empfohlen den Vergleich mit einer nicht echtzeitfähigen Netzwerktechnologie (z. B. Ethernet) in einem Teilversuch zu behandeln. Außerdem muss der finanzielle Gesamtaufwand im Rahmen des hierfür genehmigten Budgets von 2500 € liegen. Auf einen Blick lauten die Anforderungen der Aufgabenstellung an den Laborversuch daher:

- Die Echtzeiteigenschaften sollen im Mittelpunkt des Versuchs stehen.
- Es wird vorausgesetzt, dass das Controller Area Network als Basistechnologie für den Versuch verwendet wird.
- Eine zweite, nicht echtzeitfähige Netzwerktechnologie sollte zum Vergleich ebenfalls behandelt werden.
- Die Gesamtkosten des Versuchs dürfen 2500 € nicht überschreiten.

3.3 Versuchsziele

Da die Aufgabenstellung dieser Mastarbeit relativ große Spielräume bei der Gestaltung erlaubt, gilt es zu Beginn der Entwicklung des Laborversuches zunächst Ziele zu definieren, anhand de-

rer sich der Fortschritt und die Qualität des Versuchs bewerten lassen.

Diese Masterarbeit zielt auf die Entwicklung eines Laborversuchs, was bedeutet, dass die zu erstellende Versuchsanleitung als das „Hauptprodukt“ dieser Thesis zu betrachten ist. Die Anleitung soll, zusätzlich zur Aufführung in dieser Arbeit, als separates Dokument erscheinen und muss daher so gestaltet sein, dass sie auch für sich alleine konsistent ist. Daher gelten für die Anleitung folgende Bedingungen:

- Die Versuchsanleitung verfügt über eine separate Einleitung, ein eigenes Vorwort etc.
- Die Aufgabenstellungen nehmen keinen Bezug auf Kapitel außerhalb der Anleitung.
- Ein Anhang wird ggf. bereits den Kapiteln der Versuchsanleitung zugeordnet.

Da das vorhandene Budget lediglich für die Ausstattung eines Versuchsplatzes reicht und die Versuchsdurchführung nur in Zweiergruppen sinnvoll ist, muss der Laborversuch ferner so gestaltet sein, dass er von den Gruppen verteilt über das Semester, z. B. wochenweise, durchgeführt werden kann. Damit sind folgende Anforderungen verbunden:

- Die Versuchsanleitung muss ausreichend theoretische Kenntnisse vermitteln, damit der Versuch ggf. am Anfang des Semesters durchgeführt werden kann, bevor das CAN in der Vorlesung behandelt worden ist.
- Die Versuche müssen so gestaltet sein, dass die Studierenden die Aufgaben im Labor selbstständig lösen können.
- Die verfügbare Hard- und Software muss so dokumentiert sein, dass sie von den VersuchsteilnehmerInnen eigenständig konfiguriert und betrieben werden kann.

Um den Aufwand des Laborversuchs mit dem Curriculum des Masterstudiengangs zu vereinbaren, ist ferner die Bearbeitungsdauer zu begrenzen. Die Durchführung des Versuchs sollte aus diesem Grund in einem zumutbaren Zeitraum möglich sein. Für die Vor- und Nachbereitung ist jeweils noch einmal die gleiche Zeitspanne einzuplanen. Daraus resultieren folgende Anforderungen:

- Die Laboraufgaben sollten innerhalb eines Vor- bzw. Nachmittags zu lösen sein, was ca. vier Zeitstunden[1] entspricht.
- Die Vorbereitungszeit sollte ebenfalls bei ca. vier Stunden liegen, wobei die Bearbeitungsdauer in diesem Fall vom Wissensstand der Studierenden abhängt.
- Die Ergebnisse sollen schriftlich ausgewertet werden. Die sollte ebenfalls innerhalb von

1 Jeweils vier Stunden als durchschnittliche Dauer für die Vorbereitung, die Durchführung und die Nachbereitung sind ein Zeitraum, der sich erfahrungsgemäß noch neben dem regulären Curriculum im Masterstudium bewältigen lässt.

vier Stunden zu schaffen sein.

Letztlich ist zwar das Thema dieser Masterarbeit die Entwicklung eines Laborversuchs, jedoch darf nicht vergessen werden, welches Ziel der Laborversuch wiederum verfolgt: die Wissensvermittlung an die StudentInnen! Das Wichtigste ist daher sicherzustellen, dass der zu entwickelnde Versuch nicht den bloßen Selbstzweck erfüllt, sondern die TeilnehmerInnen in ihrem Studium voranbringt. Der Erkenntnisgewinn bei der Versuchsdurchführung hat somit oberste Priorität. Für die Entwicklung der Versuchsaufgaben sind aus diesem Grund folgende Voraussetzungen nötig:

- Bei jedem Versuch muss der Lerneffekt im Mittelpunkt stehen.
- Die Versuche sollten möglichst aufeinander aufbauen, damit kurz zuvor erworbenes Wissen von den Studierenden sofort wieder angewandt werden kann.
- Um den Lerneffekt zu garantieren, gilt es die Motivation der StudentInnen während der Durchführung aufrechtzuerhalten. Dies bedeutet, dass beispielsweise zeitraubende Konfigurationsvorgänge vermieden oder Hilfestellungen angeboten werden.

4 Versuchsanleitung

4.1 Einleitung

Diese Versuchsanleitung zum *Controller Area Network* (*CAN*) ist Bestandteil einer Masterarbeit mit dem Titel *„Echtzeiteigenschaften von Feldbussen am Beispiel des Controller Area Networks“*. Diese entstand im Sommersemester 2009 bzw. im Wintersemester 2009/2010 für das Lehrgebiet „Echtzeitsysteme“ an der Hochschule für Telekommunikation Leipzig.

Dieses Dokument gliedert sich in mehrere Kapitel:

- **Kapitel 2** bildet ein Vorwort zu diesem Versuch.
- **Kapitel 3** liefert Ihnen grundlegendes theoretisches Hintergrundwissen, das zur Durchführung des Versuchs behilflich sein kann. Sollten Sie die Materie bereits kennen, ist dieses Kapitel ggf. als optional zu betrachten.
- **Kapitel 4** stellt alle in diesem Versuch verwendeten Geräte vor.
- **Kapitel 5** beinhaltet alle notwendigen Schritte zur Vorbereitung des Versuchs.
- Die **Kapitel 6 bis 11** stellen die empfohlenen Versuchsaufgaben, und damit den Hauptteil dieser Anleitung, dar.
- Schließlich beinhaltet **Kapitel 12** den Anhang dieses Dokuments.

Gelegentlich finden sich in dieser Versuchsanleitung „Infokästen“ wie diesen. Sie bieten Tipps und Hinweise zur Konfiguration der Geräte und könnten dabei behilflich sein, den einen oder anderen Fehler zu vermeiden.

Abkürzungen werden in dieser Versuchsanleitung in der Regel mit Artikel, sowie ggf. mit Genitiv-s versehen. Um einen besseren Lesefluss zu ermöglichen, wird eine eingeführte Abbreviatur nicht durchgehend verwendet, sondern stellenweise die ausgeschriebene Variante bevorzugt.

Im vorliegenden Text wird sowohl für die Transkription als auch für die Translation des griechischen Buchstabens 'Φ', außer bei Namen, ausschließlich die offizielle neugriechische Übersetzung 'F' verwendet. Damit wird die Schreibweise des Dudens in einigen Fällen missachtet. Da dieser allerdings 1998 seine Monopolstellung bei der Rechtschreibung verloren hat und nun vielmehr Richtlinien beinhaltet, sei hiermit auf diese Hinwegsetzung aufmerksam gemacht.

4.2 Vorwort

Liebe Studentin,
lieber Student,

herzlich Willkommen zu diesem Laborversuch. Sicherlich haben Sie in der Vergangenheit schon öfters von *Feldbussen*, *Echtzeitsystemen* oder sogar dem *Controller Area Network* (*CAN*) gehört. Vielleicht haben Sie das Thema bereits theoretisch in Vorlesungen und Seminaren behandelt, kennen es aus Fachzeitschriften oder Büchern. Evtl. hatten Sie CAN-Komponenten auch schon einmal selbst in der Hand, z. B. bei einem Automobil. Aber wie auch immer Sie zu diesem Versuch gefunden haben, ich freue mich, dass Sie sich für das Thema interessieren.

Bitte nehmen Sie sich ein wenig Zeit – bevor Sie sich auf die Versuchsaufgaben stürzen – und arbeiten Sie diese Anleitung in Ruhe durch. Wenn Sie ein paar Minuten in die Vorbereitung des Labors investieren, verspreche ich Ihnen, dass Sie nicht nur schneller vorankommen, sondern auch mehr verstehen und am Ende sagen können, dass der Versuch Sie in Ihrem Studium weitergebracht hat.

Als Studierende des Masterstudiengangs werden an Sie selbstverständlich höhere Anforderungen gestellt als im Bachelorstudium. Dazu gehört auch, dass Sie in der Lage sind im Labor

selbstständig zu arbeiten. Bitte verstehen Sie diese Versuchsanleitung daher nicht als einen vorgeschriebenen Ablaufplan, den Sie abarbeiten und am Schluss auswerten. Er ist vielmehr als Vorschlag zu betrachten. Welche Versuche Sie wann, wie lange und in welcher Reihenfolge durchführen, bleibt grundsätzlich Ihnen überlassen. Natürlich können Sie sich ebenso dafür entscheiden, sich überhaupt nicht an dieser Anleitung zu orientieren. Wichtig ist lediglich, dass Sie wissenschaftlich arbeiten und einen ausreichend hohen Anspruch an Ihre Arbeit stellen. Werten Sie dieses Labor daher bitte so aus, wie Sie es für richtig halten und wie es von Ihnen erwartet wird.

Um Ihnen die Versuchsdurchführung nicht unnötig schwer zu machen, finden Sie in diesem Dokument gelegentlich Hilfestellungen. Diese beschränken sich jedoch insbesondere auf die Konfiguration der Geräte und der Software. Die an Sie gestellten Aufgaben müssen Sie selbstverständlich eigenständig, bzw. in Zusammenarbeit mit Ihren KommilitonInnen, lösen.

Ich wünsche Ihnen bei diesem Laborversuch viel Erfolg, und selbstverständlich auch viel Spaß!

Sebastian Lammermann
Im Oktober 2009

4.3 Versuchsablauf

Der Ablauf dieses Laborversuchs gliedert sich in drei Teile: *Vorbereitung*, *Durchführung* und *Nachbereitung*.

Um die Versuche durchführen zu können, benötigen Sie zunächst ausreichende Kenntnisse über die CAN-Technologie (sowie über das Ethernet, welches Ihnen aber bereits bekannt sein müsste). Sollten Sie das Thema noch nicht in Vorlesungen und Seminaren behandelt haben oder von Ihren außercurricalen Aktivitäten kennen, so bietet Ihnen das folgende Kapitel hierzu ausreichend Informationen. Alternativ können Sie auch auf tiefergehende Fachliteratur zurückgreifen, die Sie in den Leipziger Bibliotheken oder im Internet erhalten (z. B. [Lam208] [Etsc00][Jans00][ISO103][ISO203][ISO306][ISO404][ISO507]). Sind Sie bereits mit dem CAN und seiner Funktionsweise vertraut, betrachten Sie das folgende Kapitel bitte als optional.

Ist die Versuchsvorbereitung abgeschlossen, können Sie mit der Durchführung beginnen. Betrachten Sie den in dieser Anleitung ausgearbeiteten Versuchsablauf bitte als Vorschlag, an den Sie nicht gebunden sind. Er stellt einen Leitfaden dar, mit dem Sie durch praktische Anwendung des CANs alle wichtigen Erkenntnisse erlangen. Sollten Sie es vorziehen sich

anderweitig mit dem Echtzeit- bzw. CAN-Themenkomplex zu beschäftigen, so steht Ihnen dies ausdrücklich frei.

Nachdem Sie den Laborversuch durchgeführt haben, wird von Ihnen eine Nachbereitung erbeten. Diese erfolgt schriftlich in Form einer wissenschaftlichen Arbeit. Unabhängig davon, ob Sie sich am Leitfaden orientieren oder nicht, sollten Sie alle Fragen und Aufgabenstellungen nach bestem Wissen und Gewissen beantworten bzw. lösen. Bitte sprechen Sie mit dem zuständigen Laborpersonal ab, bis wann die Einreichung der Ausarbeitung erfolgen sollte und welcher Umfang von Ihnen erwartet wird.

4.4 Hintergrundinformationen zum CAN-Bus

4.4.1 Allgemeines zu Feldbussen

Insbesondere in der Industrie werden seit der Einführung der Elektronik in diesem Bereich Bussysteme benötigt, die Sensoren, Antriebe und andere so genannte *Feldgeräte* miteinander vernetzen [Gruh01]. Bis in die 1980er Jahre war es üblich, jedes Gerät mit jedem anderen direkt zu verbinden (*Parallelverkabelung*), und dies oftmals noch mittels Analogtechnik. Da schließlich die Verkabelung immer komplexer wurde und größere räumliche Ausmaße annahm, begann man mit der Entwicklung von digitalen seriellen Bussystemen, den *Feldbussen.*

Im Unterschied zur Parallelverkabelung verfügt ein Feldbus in der Regel nur über eine einzige Leitung, die allerdings mit allen Feldgeräten verbunden ist. Hierdurch wird einerseits der Verkabelungsaufwand erheblich reduziert und andererseits ist es so möglich, alle Geräte mit einer gemeinsamen Steuerungseinheit zu regeln. In der Praxis bietet ein Feldbus gegenüber den konventionellen Systemen ferner die Vorteile, dass sich Inbetriebnahme und Wartung vereinfachen sowie Kosten senken lassen.

Feldbusse bauen in der Regel auf dem *OSI-Schichtenmodell* auf, beschreiben diese aber üblicherweise höchstens bis zur Vermittlungsschicht (Schicht 3) [Jans00]. Dadurch, dass der Bus seriell ist und die Kommunikation von mehr als zwei Geräten ermöglicht, ergibt sich darüber hinaus die Notwendigkeit, dass die Nachrichten adressierbar sind. Um in einer industriellen Umgebung eingesetzt werden zu können, muss ein Feldbus im Übrigen bestimmte Leistungsmerkmale aufweisen, wie z. B. *Echtzeiteigenschaften*, eine *hohe Zuverlässigkeit* und eine *geringe Störempfindlichkeit.* Anzustreben ist außerdem eine Verträglichkeit mit anderen Feldbussystemen (*Interoperabilität*) um verschiedene Technologien mit geringem Aufwand miteinander zu verbinden.

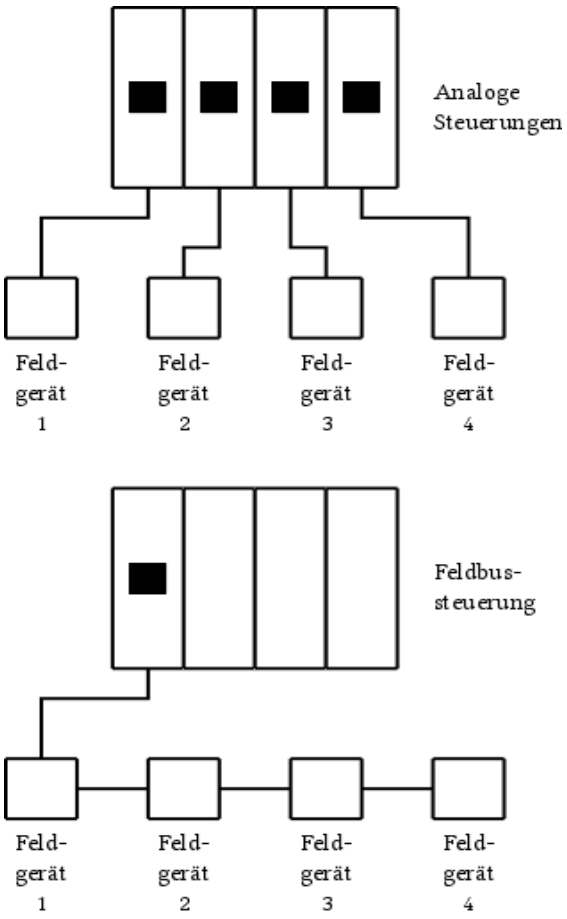

Abb. 1: Vergleich von konventioneller paralleler (oben) und serieller Feldbusverkabelung (unten)

4.4.2 Entstehung des Controller Area Networks

Als Mitte des 20. Jahrhunderts zum ersten Mal Elektronik Einzug in den Kraftfahrzeugbau erhielt, wurden Sensoren, Steuereinheiten und andere – bis dahin ausschließlich analoge – Geräte zunächst direkt (also parallel) miteinander verkabelt. In den folgenden Jahrzehnten gewann die Bordelektronik mehr und mehr an Bedeutung, was letztlich dazu führte, dass die im Automobil verlegten Kabel Längen von bis zu 2 km und eine Masse von über 100 kg erreichten [Enge00].

1983 begann bei der Robert Bosch GmbH (BOSCH) in Stuttgart die Entwicklung des CAN-Busses. Ziel war zunächst die Entwicklung eines digitalen Feldbusses zur Steuerung des Antriebsstrangs[2], der den Anforderungen im Kraftfahrzeugbereich gerecht wird und gleichzeitig einen Großteil der bisherigen Leitungen einspart. Zwei Jahre später wurde die erste

2 Zum Antriebsstrang eines Kraftfahrzeugs zählen alle Komponenten, die für die Übertragung des Drehmoments vom Motor auf die Räder zuständig sind. Hierzu gehören z. B. Antriebswellen, Getriebe, Kupplungen etc.

CAN-Spezifikation vorgestellt und die Entwicklung von geeigneten Chips begann in Kooperation mit Intel. 1988 erschienen die ersten Serienchips auf dem Markt.

Die damalige Daimler-Benz AG war der erste Automobilkonzern, welcher mit der Entwicklung einer CAN-basierten Vernetzung der Fahrzeugelektronik anfing. Binnen weniger Jahre setzte sich BOSCHs – komplett in Hardware implementiertes [Maye06] – Controller Area Network bei allen namhaften Kraftfahrzeugherstellern durch und verdrängte die bis dahin populäre Analogtechnik. 1993 wurde der Feldbus außerdem im ISO-Standard Nummer 11898 erstmals international genormt, welcher seitdem mehrfach aktualisiert worden ist.

Seit den 1990er Jahren findet das CAN auch vermehrt außerhalb des Kraftfahrzeugbereichs Anwendung. Zu nennen sind hier insbesondere die industrielle Automatisierung und die Medizintechnik, in der ähnliche Anforderungen an ein Bussystem gestellt werden wie im Fahrzeugbau.

4.4.3 Netzwerktopologie

Der CAN-Bus weist fysikalisch in der Regel eine mit Abschlusswiderständen terminierte Linienstruktur (*Bus*) auf. Es ist jedoch auch möglich, eine Doppelsterntopologie zu verwenden, bei denen die Netzknoten an Sternverbinder an den Endpunkten der Leitung angeschlossen sind. Als Medium dient üblicherweise eine verdrillte Kupferdoppelader (*Twisted Pair*) oder ein anderes Paar von Kupferadern.

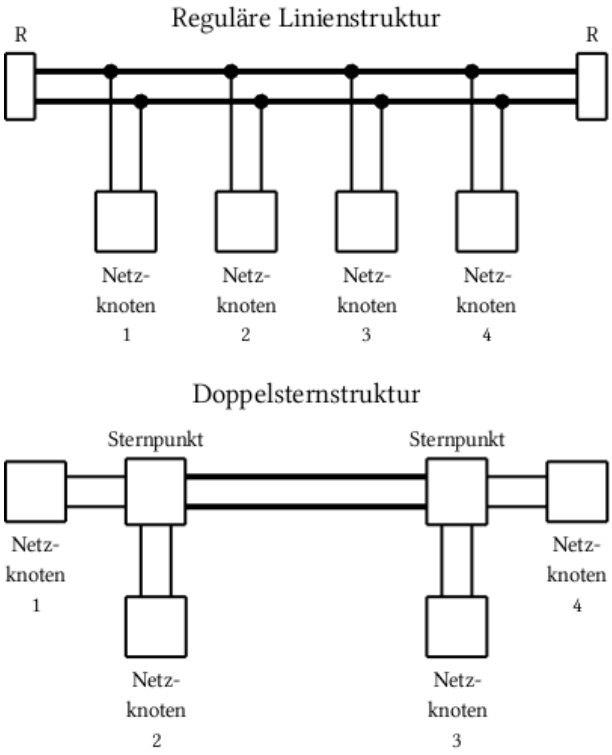

Abb. 2: Vergleich von klassischer Linienstruktur (oben)
und Doppelsternstruktur (unten)

Die möglichen Leitungslängen hängen vom Betriebsmodus des Netzes ab. Generell wird zwischen folgenden zwei Modi unterschieden:

- Der *High-Speed-Modus* bietet eine Übertragungsrate von bis zu 1 Mbit/s und ermöglicht den Anschluss von bis zu 30 Netzknoten. Die Leitungslänge ist in diesem Modus jedoch auf maximal 40 m beschränkt [ISO203]. Gedacht ist dieser Modus insbesondere für räumlich kleine Bereiche in denen hohe Datenraten nötig sind, wie z. B. beim Motorraumbus.
- Im *Low-Speed-Modus* hingegen beträgt die maximale Übertragungsrate 125 kbit/s und es können ca. 20 Knoten mit dem Bus verbunden werden. Die maximale Buslänge beträgt hier ca. 500 m [ISO306]. Der große Vorteil des Low-Speed-Modus liegt in der Ausfallsicherheit, da auch im *Eindrahtmodus*[3] – aufgrund des größeren Spannungshubs – dominante und rezessive Bits unterschieden werden können.

Um die Faktoren für den Aufbau eines CANs zu vereinheitlichen, lassen sich folgende Rahmenkriterien festlegen: Ein Netzwerk mit der maximalen Gesamtlänge von 40 m verbindet ca. 20 Knoten miteinander, wobei der maximale Abstand zweier benachbarter Knoten höchs-

3 Als Eindrahtmodus gilt sowohl der Kurzschluss der Adern gegen Masse und untereinander, als auch mit der Versorgungsspannung.

tens 20 m betragen darf.

4.4.4 Leitungscode

Als Leitungscode verwendet der CAN-Bus *Non-Return to Zero* (*NRZ*). Grundsätzlich sind bei diesem Code die Daten binär codiert und die beiden logischen Zustände durch unterschiedliche (und hier unipolare) Spannungswerte repräsentiert. Der Signalpegel bleibt während der gesamten Bitzeit konstant und fällt nicht auf einen Basiswert zurück, wie beispielsweise beim *Return-to-Zero-Code* (*RZ*). Außerdem verfügt er über kein codiertes Taktsignal, wie z. B. der *Manchestercode*, wodurch für die Darstellung eines Bits lediglich ein Zeitabschnitt benötigt wird. Dies macht ihn zum einfachsten möglichen Leitungscode, allerdings wird eine externe Synchronisation (!) voraussetzt.

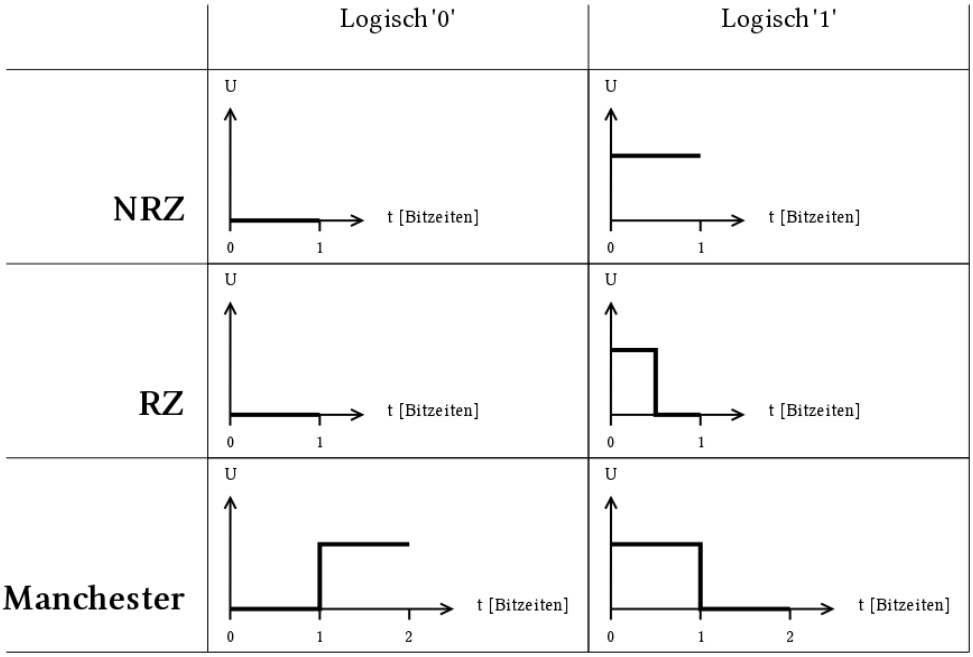

Abb. 3: Vergleich von NRZ-, RZ- und Manchesterleitungscodierung

4.4.5 Buszugriffsverfahren

Um den Zugriff auf das Medium zu steuern, verwendet das CAN einen relativ simplen Algorithmus namens *Carrier Sense Multiple Access with Collision Resolution* (*CSMA/CR*)[4], welcher eine Modifikation von CSMA darstellt.

CSMA besteht aus zwei Verfahren: Das erste (*Carrier Sense, CD*) beschreibt, dass ein Netzknoten vor dem Senden von Daten prüft, ob ein Trägersignal verfügbar ist. Ist dieser Träger

4 Engl. in etwa: „Trägererkennung und Mehrfachzugriff mit Kollisionsauflösung“.

anwesend, ist das Gerät folglich an ein Netzwerk angeschlossen. Im nächsten Schritt wird nun überprüft, ob bereits eine Kommunikation auf dem Bus stattfindet, wodurch ein Senden der eigenen Nachricht nicht möglich wäre. Ist der Bus für eine bestimmte Periode (Interframe Space) frei, kann mit der Datenübermittlung begonnen werden.

Das zweite Verfahren (*Multiple Access, MA*) beschreibt die Tatsache, dass das Netzwerk einen logischen Bus bildet. Dies bedeutet, dass jeder Netzknoten auf das Medium zugreifen kann und dass jede Nachricht von allen anderen mit dem Bus verbundenen Geräten empfangen wird. Weiterverarbeitet werden allerdings einzig die Daten, die an den eigenen Knoten adressiert sind.

Da die Datenübertragung beim CAN unkoordiniert und asynchron erfolgt, kann es passieren, dass zwei oder mehr Geräte gleichzeitig versuchen Daten zu senden. In diesem Fall kommt es zu einem Übertragungskonflikt bzw. zu einer Kollision. Um den CSMA-Algorithmus zu verbessern und die Effizienz des CANs zu steigern, wurde daher das zusätzliche Verfahren der Kollisionsauflösung (*Collision Resolution, CR*) eingeführt. Beginnen mehrere Knoten gleichzeitig mit dem Senden von Daten, wird während einer Auswahlfase (*Arbitrierung*) entschieden, welches Gerät sich durchsetzt und am Ende weiter senden darf.

Um die Kollision zu lösen, verfügt jede zu sendende Nachricht über ein *Nachrichtenarbitrierungsfeld*, bestehend aus Nachrichtenidentifier sowie RTR-Bit[5], welches von jedem beteiligten Netzknoten bitweise auf den Bus geschaltet wird [Etsc00]. Die Nachricht mit dem dominantesten Wert in diesem Feld „gewinnt“ den Konflikt und der entsprechende Netzknoten sendet schließlich als einziger seine Nachricht. Das Procedere funktioniert wie folgt:

Starten mehrere Netzknoten auf einem bis dahin ruhigen Bus eine Datenübertragung, beginnen sie diese mit dem Aufschalten eines dominanten Pegels auf den Bus (Start-of-Frame-Bit, SOF). Während der Übertragung des Nachrichtenarbitrierungsfeldes und des RTR-Bits wechselt der Pegel meist zwischen dominant und rezessiv, je nachdem welche logische Bitfolge übertragen wird. Da beim CAN die Grundregel gilt, dass ein dominanter einen rezessiven Pegel überschreibt, ist irgendwann während der Übertragung der Zeitpunkt erreicht, an dem das rezessive Bit des einen durch das dominante Bit des anderen Netzknotens überlagert wird. Durch den ständigen Vergleich der gesendeten und empfangenen Daten erkennt der „unterlege“ Netzknoten, dass der von ihm übertragene rezessive Pegel nicht auf dem Bus anliegt und beendet seinen Übertragungsversuch.

5 Siehe: *4.4.6 Rahmentypen.*

Beispiel:

Zwei Netzknoten, **A** und **B**, greifen gleichzeitig auf den Bus zu. Sie signalisieren beide als erstes mit dem Aufschalten eines dominanten SOF-Bits, dass sie den Bus belegen. Die Bits 2 bis 4 sind bei **A** und **B** identisch: zunächst wird der Pegel rezessiv und dann zweimal dominant. Bis zu diesem Punkt gleichen sich die Rahmen der beiden Nachrichten und der Konflikt kann nicht gelöst werden.

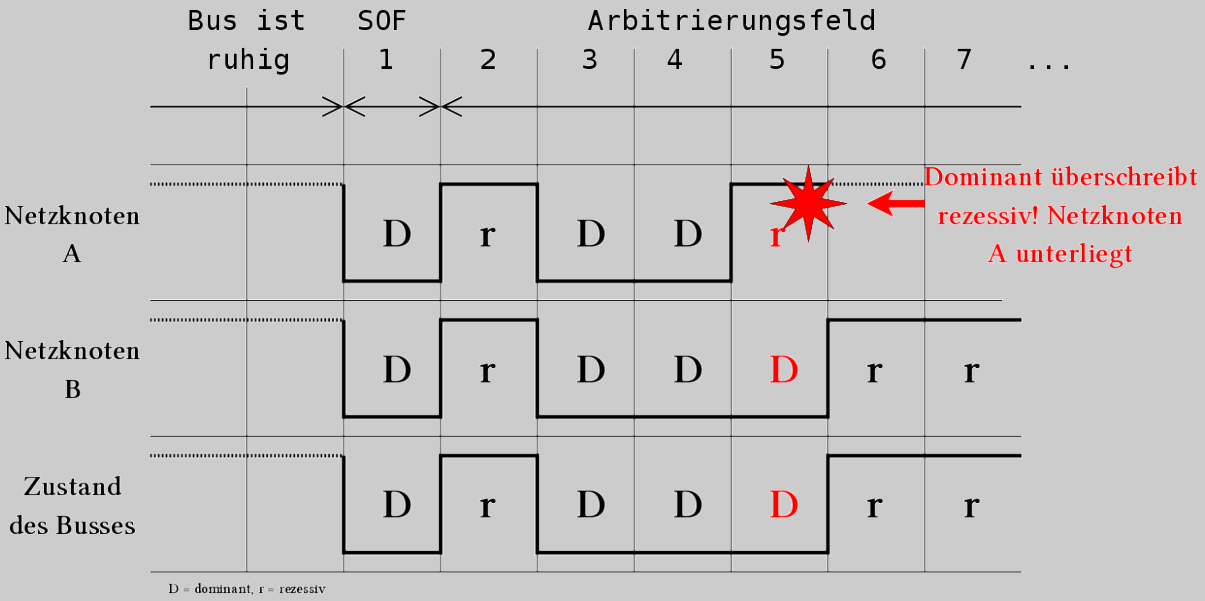

Abb. 4: Netzknoten B gewinnt eine Kollision gegen Netzknoten A

Bei Bit 5 fällt die Entscheidung: Während Netzknoten **A** einen rezessiven Pegel sendet, ist der von **B** dominant. Gemäß der Regel „dominant überschreibt rezessiv“ nimmt der Bus einen dominanten Pegel an. Durch den Vergleich der gesendeten und der empfangenen Daten merkt **A** nun, dass diese nicht übereinstimmen. Eine „höherwertige“ Nachricht muss also ebenfalls gerade auf dem Bus übertragen werden. Daraufhin beendet Netzknoten **A** seinen Übertragungsversuch und kehrt in den Empfangsmodus zurück. Er wird zu einem späteren Zeitpunkt erneut versuchen, seine Nachricht zu senden. Netzknoten **B** sendet hingegen weiter seinen Datenrahmen.

Da einzig die „unterlegenen“ Netzknoten die Kollision bemerken, hat das CSMA/CR-Verfahren den Vorteil, dass die Nachricht mit dem höchsten Wert im Arbitrierungsfeld zerstörungsfrei übertragen wird. Außerdem erlaubt dies eine Priorisierung der Nachrichten, was essenziell für den Echtzeitbetrieb ist.

4.4.6 Rahmentypen

Auf der CAN-Sicherungsschicht existieren fünf unterschiedliche Arten von Datenrahmen. Zwei davon, der *Data Frame* und der *Remote Frame*, dienen der Übertragung von Nachrichten. Zwei weitere, der *Error Frame* und der *Overload Frame*, sind für die Meldung von Fehlern gedacht. Schließlich existiert noch der *Interframe Space* zwischen den anderen Rahmen.

I Data Frame

Aufgabe des Data Frames ist es, reguläre Nachrichten von einem Sender zu mindestens einem Empfänger zu übertragen. Der Datenrahmen, ohne Stopfbits, ist wie folgt aufgebaut:

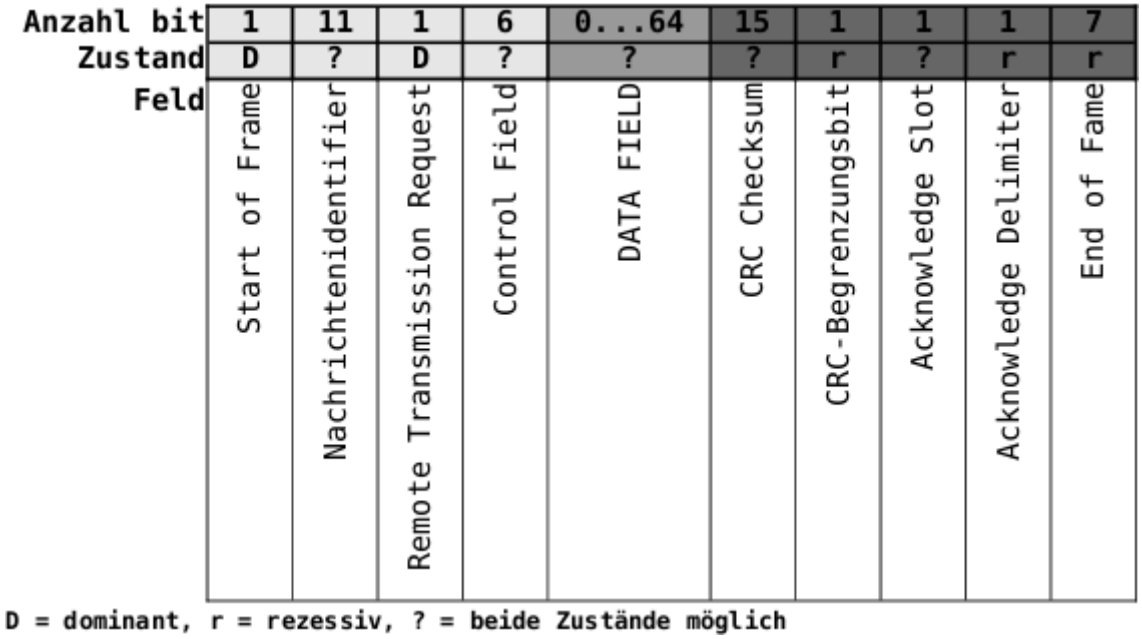

Abb. 5: Aufbau des CAN Data Frames

- **Start of Frame (SOF)**: Das SOF-Bit bildet die Telegrammanfangskennung. Durch Aufschalten eines dominanten Pegels auf den Bus wird anderen Knoten signalisiert, dass eine Nachricht folgt.
- **Nachrichtenidentifier (ID)**: Der Identifier kennzeichnet den Sender eindeutig und bestimmt die Priorität der Nachricht. Im Standardformat ist die ID 11 bit lang.
- **Remote-Transmission-Request-Bit (RTR)**: Das RTR-Bit ermöglicht die Unterscheidung eines Data Frames von einem Remote Frame. Bei einem Data Frame ist das Bit immer dominant und damit höher priorisiert als der entsprechende Remote Frame.
- **Control Field (CF)**: Dieses Steuerfeld beschreibt die Länge der nachfolgenden Payload.
- **Data Field (DF)**: In diesem Datenfeld ist die Payload des Datenrahmens enthalten. Es kann bis zu 8 Byte aufnehmen.

- **Prüfsumme (CRC)**: Damit ein Empfänger prüfen kann, ob eine empfangene Nachricht durch Störungen verfälscht worden ist, enthalten die Datenrahmen eine auf der *zyklischen Redundanzprüfung* (*Cyclic Redundancy Check*, *CRC*) basierte Prüfsumme [Will93].
- **Acknowledge Field (ACK)**: Das Acknowledge Field wird zur Bestätigung der fehlerfreien Nachrichtenübermittlung verwendet.
- **End of Frame (EOF)**: Ein normaler Data Frame wird mit einem 7 bit langen rezessiven Pegel beendet.

Zusätzlich zum normalen Rahmenformat existiert außerdem der *Extended Frame*. Hierbei stehen insgesamt 29 bit für den Nachrichtenidentifier zur Verfügung, aufgeteilt in zwei Felder.

II Remote Frame

Neben dem normalen Data Frame existiert ferner der Remote Frame. Er wird von einem Datenempfänger an einen Datensender verschickt und enthält die Aufforderung eine bestimmte Nachricht zu übermitteln.

Der Aufbau eines Remote Frames entspricht weitestgehend dem des Data Frames, allerdings weist er einige Besonderheiten auf. Der markanteste Unterschied ist der rezessive Zustand des RTR-Bits. Hierdurch unterliegt ein Remote Frame während der Arbitrierung automatisch einem Data Frame mit dem gleichen Identifier. Dadurch wird sichergestellt, dass die Antwort auf eine Nachrichtenanforderung gegenüber dessen Anfrage immer priorisiert wird und es somit nicht zu einer unnötigen Auslastung des Busses kommt.

Ein weiteres spezielles Merkmal ist das Control Field bzw. das Data Field. Das Data Field ist immer leer, enthält also 0 Byte. Der Wert des Control Fields hingegen entspricht der erwarteten Länge des Data Fields im Antwortpaket (!) und ist somit in der Regel größer als 0.

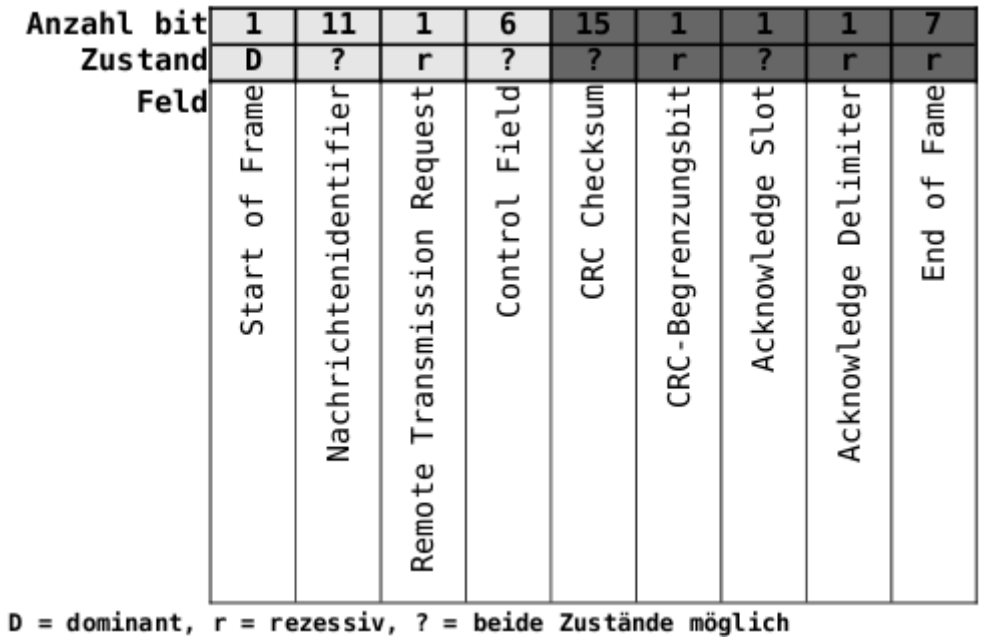

Abb. 6: Aufbau des CAN Remote Frames

Analog zum Data Frame existiert auch beim Remote Frame das erweiterte Rahmenformat mit einer 29 bit langen ID.

III Error Frame

Ist ein Data Frame oder Remote Frame gestört[6], erfolgt der Abbruch des Sendeprozesses mittels eines Error Frames. Ein solcher Datenrahmen unterscheidet sich im Aufbau grundlegend vom Data Frame:

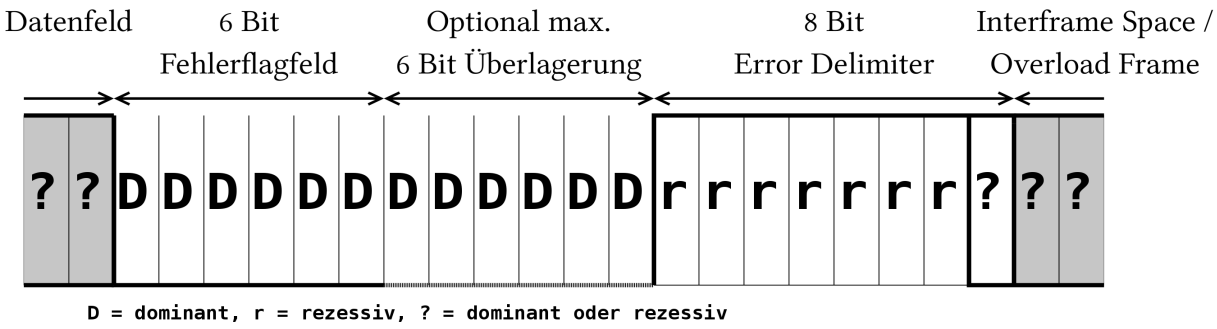

Abb. 7: Aufbau des CAN Error Frames

- **Fehlerflagfeld**: Das Fehlerflagfeld besteht aus 6 dominanten Bits. Einerseits werden hierdurch alle ggf. auf dem Bus gesendeten Daten überschrieben und andererseits wird eine Codeverletzung herbeigeführt. Falls mehrere Knoten gleichzeitig oder dicht

6 Fehler treten bspw. auf, wenn die Prüfsumme oder das Rahmenformat inkorrekt sind, das Acknowledgement nicht gesendet oder die Bitcodierungsregel nicht eingehalten wird.

aufeinander folgend einen Fehler melden, können sich deren Fehlerflagfelder auch überlagern, wodurch maximal 12 dominante Bits hintereinander gesendet werden. Da durch das Prinzip des Bitstuffings maximal fünf Bits gleicher Polarität aufeinander folgen dürfen, bemerken alle teilnehmenden Netzknoten diese Codeverletzung und verwerfen den entsprechenden Datenrahmen.

- **Error Delimiter**: Der Error Delimiter besteht aus einer Folge von 8 rezessiven Bits und bildet den Abschluss des Error Frames. Da es sein kann, dass mehrere Netzknoten kurz hintereinander Fehlerflagfelder senden, wird zunächst nur das erste rezessive Bit auf den Bus geschaltet. Bleibt der Bus dominant, weiß das Gerät, dass andere Knoten ebenfalls einen Fehler melden. So ist es möglich abzuschätzen, ob das Gerät den Fehler zuerst erkannt hat, was bei der Fehlereingrenzung behilflich sein kann. Sobald der Bus tatsächlich den rezessiven Zustand annimmt, werden die letzten 7 Bits gesendet.

Damit ein fehlerhaftes Gerät nicht das gesamte Netzwerk gefährdet, sind CAN-Knoten in der Lage sich ggf. selbst abzuschalten. Je nachdem wie viele selbst verursachte Fehler von einem Knoten erkannt wurden, existieren drei Betriebszustände:

- **Error Active**: Der Knoten nimmt normal an der Kommunikation teil und meldet ggf. auch Fehler.
- **Error Passive**: Der Netzknoten ist nicht voll kommunikationsfähig. Er darf zwar prinzipiell senden und empfangen, jedoch nicht mehr als erstes Fehlerflags verschicken. Außerdem muss er vor dem erneuten Senden einer Nachricht mindestens 8 Bitzeiten abwarten um die Kommunikation der fehlerfreien Knoten nicht zu stören.
- **Bus Off**: Die Fehlerrate des Netzknotens ist so hoch, dass er komplett vom Bus getrennt wird.

IV Overload Frame

Der Overload Frame ist eine spezielle Art des Error Frames. Dieses Überlasttelegramm wird verschickt, wenn der nachfolgende Datenrahmen erst nach einer Verzögerung gesendet werden soll. Dies ist einerseits nötig, wenn während des Interframe Spaces ein dominantes Bit erkannt wird. Andererseits kann das letzte Bit eines vorhergehenden Error Frames oder Overload Frames dominant gesetzt sein um das Übertragen des Rahmens zu forcieren. Es dürfen maximal zwei Overload Frames hintereinander gesendet werden.

Im Aufbau unterscheidet sich der Overload Frame von Error Frame nur insofern, dass im

vorangehenden Rahmen das letzte Bit dominant gesetzt ist.

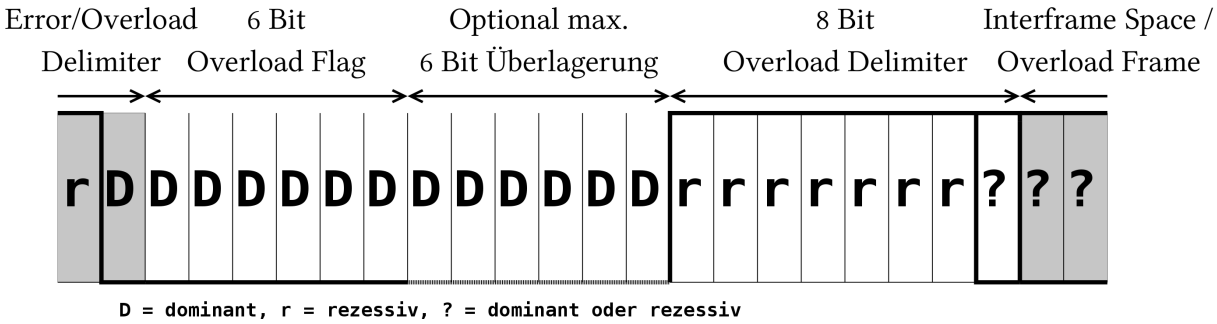

Abb. 8: Aufbau des CAN Overload Frames

V Interframe Space

Der Standard des Controller Area Networks schreibt vor, dass es zwischen zwei Data Frames bzw. Remote Frames einen Interframe Space geben muss. Dieser besteht zunächst aus drei rezessiven Bits, die nur durch einen Error Frame oder Overload Frame überschrieben werden dürfen. Nach diesem Mindestabstand der Rahmen folgt entweder sofort der nächste Data Frame, oder der Bus bleibt so lange im rezessiven Ruhezustand bis wieder Daten gesendet werden müssen.

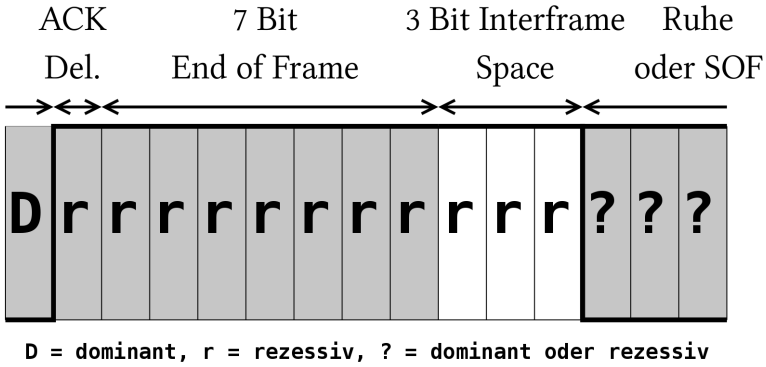

Abb. 9: Aufbau des CAN Interframe Spaces

4.4.7 CAN in der Praxis

In diesem Laborversuch wird das CAN in einer Laboranordnung verwendet, die Ihnen den Aufbau und die Eigenschaften des Feldbusses verdeutlichen soll. Damit sind selbstverständlich weder die Geräte noch die Netztopologie typisch für einen Anwendungsfall.

Wahrscheinlich ist Ihnen das Controller Area Network in der Praxis aber bereits begegnet, ohne dass Sie dies bemerkt haben. Überlegen Sie, ob Ihnen beispielsweise aus einem Automobil

einer der folgenden Anwendungsfälle bekannt ist:

- **Drehzahlmessung**: Jedes Fahrzeug verfügt über einen Drehzahlmesser, der in der Regel an einer der Achsen befestigt ist. Dieses Messgerät überträgt ständig – also im Abstand weniger Millisekunden – den aktuellen Messwert an ein Anzeigegerät, dass sich meist auf der Instrumententafel des Autos befindet. Obwohl die meisten Fahrzeuge aus optischen Gründen Anzeigen im Analogdesign verwenden, ist der dargestellte Wert damit digital.
- **Innenraumbeleuchtung**: Moderne Automobile besitzen heutzutage ein ausgeklügeltes Lichtkonzept. Steigen Sie in ein solches Fahrzeug, leuchten beispielsweise zuerst alle Lampen hell auf. Schließen Sie dann die Tür, wird das Licht leicht gedimmt. Sobald Sie wiederum die Scheinwerfer einschalten, wird die Innenraumbeleuchtung deaktiviert, die Instrumententafel leuchtet auf und auch das Display des Autoradios wird erhellt. Diese und ähnliche Vorgänge werden über das CAN gesteuert. Alle Komponenten, in diesem Fall die Lampen und Anzeigen, erhalten z. B. vom Lichtschalter und den Türen Statusinformationen und reagieren entsprechend.
- **Airbag**: Ihnen hoffentlich aus eigener Erfahrung noch unbekannt ist die Funktionsweise des Airbags. Prallt das Fahrzeug auf etwas, registriert ein Sensor in der Front des Automobils dies und sendet eine Nachricht an die Steuereinheit der Sicherheitsvorrichtung. Diese löst wiederum das Schutzsystem aus und bläst Gas in den Airbag.

4.5 Verwendete Geräte

Beim CAN-Laborversuch stehen Ihnen verschiedene Geräte zur Verfügung. Dieser Abschnitt stellt Ihnen zunächst alle Komponenten vor, die in dieser Anleitung erwähnt werden. In jedem Teilversuch ist dann separat aufgelistet, welche Geräte sich brauchen und wie Sie diese konfigurieren. Sollten Sie nicht alle vorgeschlagenen Versuche durchführen, könnte es sein, dass Sie nicht alle Geräte benötigen. Andererseits müssen Sie ggf. auf weitere Komponenten zurückgreifen, sofern Sie es vorziehen sich anderweitig mit der CAN-Technologie auseinanderzusetzen. Sprechen Sie in diesem Fall bitte vorher mit der zuständigen Laborkraft, damit Ihnen die entsprechende Technik zur Verfügung gestellt werden kann.

Folgende Komponenten sind für den Laborversuch vorgesehen:

- **CAN-Flachbandkabel**: Ihnen stehen zwei CAN-Flachbandkabel mit insgesamt vier 120-Ω-Terminatoren zur Verfügung. Das Kabel und die Widerstände sind mit neunpoligen D-Sub-Buchsen bzw. -Steckern ausgestattet.

- **D-Sub-Stecker**: Um mit einem Oszilloskop auf den Bus zugreifen zu können, liegt den Kabeln außerdem ein neunpoliger D-Sub-Stecker bei, an dessen Adern Sie Signale abnehmen können.
- **USB-CAN-Gateway**: Die drei USB-CAN-Gateways der Firma Sys-Tec kombinieren jeweils einen CAN-Knoten mit einem CAN-Transceiver. Jedes Modul bietet insgesamt zwei Schnittstellen: eine Mini-USB-Buchse für den direkten Anschluss an einen PC, sowie einen neunpoligen D-Sub-Stecker für den Anschluss an ein CAN-Flachbandkabel.
- **Ethernet-CAN-Gateway**: Die beiden Ethernet-CAN-Gateways, ebenfalls von der Firma Sys-Tec, dienen dazu zwei CANs mittels eines Ethernets zu „verbinden". Um dies zu ermöglichen, existieren an den Modulen jeweils eine RJ45-LAN- und eine CAN-Schnittstelle. Letztere ist doppelt ausgelegt um einerseits mittels D-Sub-Stecker den Anschluss an ein Flachbandkabel zu erlauben. Andererseits besteht die Möglichkeit die entsprechenden CAN-Adern direkt anzuschließen. Ferner ist eine EIA-232-Schnittstelle (serielle Schnittstelle) vorhanden um das Gateway mittels Nullmodemkabel direkt vom PC aus zu konfigurieren. Die Ethernet-CAN-Gateways werden jeweils durch ein 12-V-Netzteil mit Spannung versorgt.
- **Nullmodemkabel**: Für die (Erst-)Konfiguration der Ethernet-CAN-Gateways benötigen Sie ein Nullmodemkabel.
- **10Base-T-Hub**: Damit der Datenverkehr im Ethernet mitgeschnitten werden kann, steht ferner ein 10Base-T-Hub zur Verfügung.
- **Ethernetkabel**: Für den Aufbau des LANs liegen diverse Patchkabel bereit.
- **Computer**: Sie benötigen für den Laborversuch insgesamt zwei Personalcomputer. Jeder Rechner muss mit dem Betriebssystem *Microsoft Windows XP* ausgestattet sein und über ausreichend Schnittstellen (mindestens 1x Ethernet, 3x USB, 1x EIA-232) verfügen. Um Software zu installieren, benötigen Sie ggf. Administrationsrechte.
- **Oszilloskop**: Für die Analyse der OSI-Schicht 1 steht Ihnen des Weiteren ein digitales Oszilloskop zur Verfügung.
- **Software**: Um die Laborversuche durchführen zu können, benötigen Sie diverse Treiber und Programme. Sollten diese auf den PCs noch nicht installiert sein, finden Sie sie auf der beiliegenden CD-ROM oder im Intranet der Hochschule.

Abb. 10: Fotografie der wichtigsten Komponenten: Ethernet-Gateways, Nullmodemkabel, Ethernet-Hub, CAN-Bus-Kabel, USB-Gateways, Terminatoren, D-Sub-Stecker

4.6 Vorbereitung des Versuchs

I Installation der Software

Sie finden auf der beiliegenden CD-ROM bzw. in der Archivdatei im Intranet zwei Verzeichnisse. Das Verzeichnis `./Manuals/` enthält die Bedienungsanleitungen zu den USB- bzw. Ethernet-CAN-Gateways in deutscher und englischer Sprache. Darüber hinaus steht Ihnen eine Dokumentation der CAN-APIs zur Verfügung, sofern Sie in die CAN-Programmierung einsteigen möchten.

Im Verzeichnis `./Software/` sind hingegen alle Programme und Treiber hinterlegt, die Sie für den Laborversuch benötigen. Am einfachsten ist es vorab auf beiden PCs die komplette Software zu installieren. Zusammengefasst ist folgende Software enthalten:

- Software für die CANopen-API-Programmierung (sollten Sie installieren, benötigen Sie bei diesem Versuch aber nicht)

- Treiber und Konfigurationssoftware
- `PCANview` für die einfache Generierung und Auswertung von CAN-Nachrichten
- `CAN-REport` für eine detailliertere Auswertung von CAN-Nachrichten
- `Wireshark` (oder wahlweise `Ethereal`) für die Paketanalyse im Ethernet
- `PackETH` für die Erzeugung von Datenverkehr im Ethernet
- `Traffic Graph` für die Visualisierung der Netzwerklast

Darüber hinaus benötigen Sie für diesen Versuch die Software `Hyperterminal`, welche dem Betriebssystem bereits beiliegt.

II Rücksetzung der Geräte

Schließen Sie zunächst die USB-CAN-Gateways an einen der beiden PCs an und installieren Sie alle drei Module. Jedes Modul enthält zwei logische Geräte, so dass Sie insgesamt sechsmal automatisch die Treiber suchen lassen müssen. Sind die Geräte korrekt installiert, leuchtet die rote LED jeweils konstant.

Es folgt die Konfiguration. Öffnen Sie hierfür die Systemsteuerung des Betriebssystems und starten Sie das Programm `USB-CANmodul Control`. Weisen Sie jedem Modul über den Button „`Change`“ eine eigene `Device Nr.` zu, wobei der Wert zwischen `0` und `254` liegen muss.

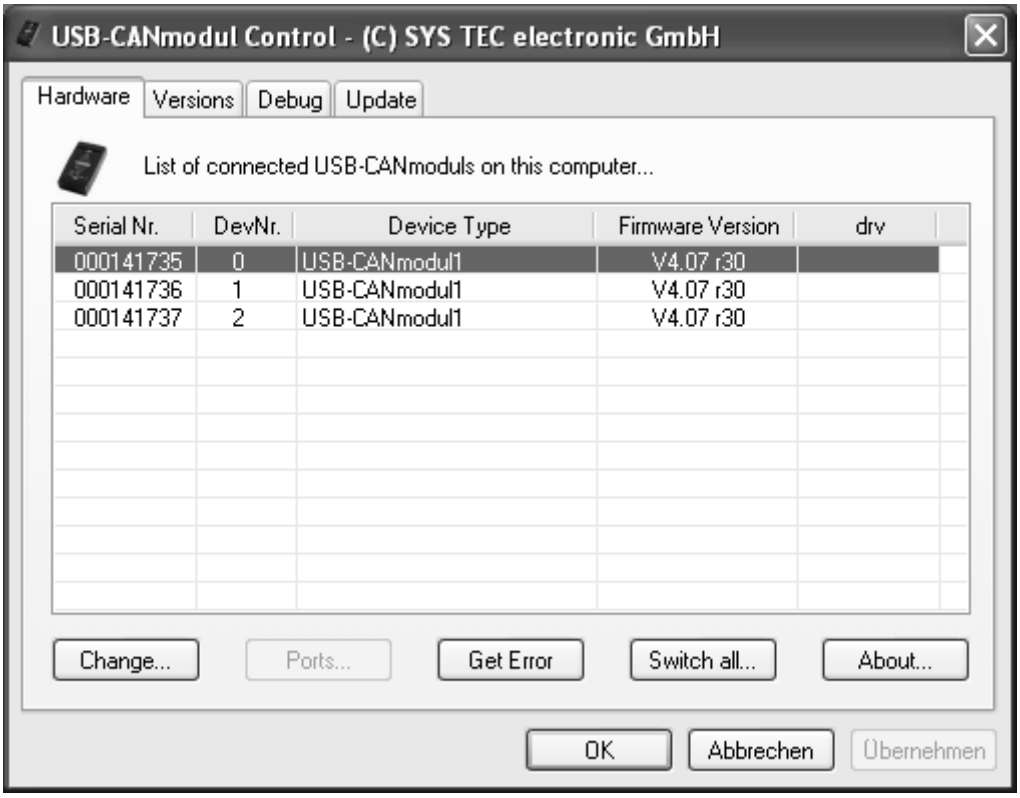

Abb. 11: Fenster des Programms USB-CANmodul Control in der Systemsteuerung

Nehmen Sie sich als nächstes die beiden Ethernet-CAN-Gateways vor. Schließen Sie hierfür das erste Modul per Nullmodemkabel an einen der PCs an und stellen Sie sicher, dass sich alle vier Schalter am Gateway in der unteren Position befinden. Wählen Sie sich dann mit der Software `Hyperterminal` mit 9600 bit/s ein. Ihnen stehen nun folgende Befehle zur Verfügung:

- `ls` Zeigt den Inhalt des aktuellen Verzeichnisses an
- `cd` Wechselt das Verzeichnis
- write Legt eine neue Datei an
- `rm` Löscht eine Datei
- `sync` Schreibt eine Datei in den ROM-Speicher des Gerätes
- reset Startet das Gerät neu
- `mkif` Legt eine neue Gerätekonfiguration an
- `ipcfg` Konfiguriert die Netzwerkschnittstelle
- version Zeigt die aktuelle Version der installierten Firmware an

Zuerst sollten Sie die Netzwerkeinstellungen vornehmen. Verwenden Sie hierfür den Befehl `ipcfg <IP-Adresse> <Netzmaske> <Gateway>`. Eine reguläre Konfiguration könnte demnach z. B. wie folgt aussehen:

```
ipcfg 192.168.2.2 255.255.255.0 192.168.2.1
```

Mit der Eingabe des Befehls `ipcfg` ohne Parameter können Sie jederzeit Ihre Einstellungen überprüfen.

Um eine Kommunikation mit dem Labor-PC zu erlauben, muss auf den Ethernet-CAN-Gateways ein TCP- bzw. UDP-Server gestartet werden. Einen neuen TCP-Server legen Sie mit folgendem Befehl an:

```
mkif tcpserv
```

Dem ersten Server wird die Nummer `0` zugeordnet. Überprüfen Sie daher die Einstellungen mit dem Befehl:

```
/if/tcpserv0
```

Achten Sie darauf, dass das Protokoll auf dem Port `8234` kommuniziert.

Möchten Sie lieber einen UDP-Server verwenden, oder beide gleichzeitig, können Sie diesen mit folgendem Befehl anlegen:

```
mkif udpserv
```

Analog zum TCP-Server können Sie die Konfiguration mit folgendem Befehl überprüfen:

```
/if/udpserv0
```

War die IP-Konfiguration erfolgreich, können Sie weitere Einstellungen am Gerät nun alternativ auch im Terminal via `telnet <IP-Adresse>` *vornehmen.*

Als nächstes folgt die Einstellung der CAN-Schnittstelle. Aufgrund eines Fehlers im System der Gateways ist es zwar auf diesem Wege möglich die aktuelle Konfiguration zu löschen, ein neues Interface kann aber nicht immer angelegt werden. Für den Fall, dass dies bei Ihnen funktioniert, finden Sie im Folgenden die beiden Befehle für das Löschen der aktuellen und das Erstellen einer neuen Konfiguration:

```
rm /if/can0
```

```
mkif can
```

Generell können Sie mit dem Befehl `/if/can0` die aktuelle Konfiguration der CAN-Schnittstelle überprüfen. Stellen Sie nun die CAN-ID, die Übertragungsrate und den Status der Schnittstelle mit den folgenden Parametern ein: `/if/can0 <on/off> canid:<Hexadezimalwert> baud:<Dezimalwert>`. Eine Konfiguration könnte z. B. wie folgt aussehen:

```
/if/can0 on canid:05 baud:4
```

Für die Werte der Übertragungsrate gilt: Je niedriger der Wert ist, desto höher ist die

Bitrate. `0` ist gleichbedeutend mit 1 Mbit/s, `1` mit 800 kbit/s, `2` mit 500 kbit/s usw.

Wiederholen Sie zum Abschluss den gesamten Vorgang mit dem zweiten Gerät.

Leider ist das Betriebssystem der Ethernet-CAN-Gateways nicht vollkommen stabil. Bei der Konfiguration der Geräte könnte es daher gelegentlich zu einem plötzlichen Neustart kommen, wodurch Ihre aktuellen Einstellungen verloren gehen würden. Um die entsprechenden Arbeitsschritte einzusparen, können Sie die obigen Befehle auch in eine Textdatei schreiben, direkt an das Gerät senden und dort in den ROM schreiben. Setzen Sie hierfür den Schalter `DEFT` *nach oben und geben Sie im* `Hyperterminal` *nacheinander die folgenden Befehle ein:*

```
reset
cd save
rm rc
write rc
```

Klicken Sie nun im Menü auf „`Übertragung > Textdatei senden`*“ und wählen Sie Ihre Datei aus. Beenden Sie die Übertragung mit dem Befehl* `Ctrl+D`*. Geben Sie nun noch*

```
sync
```

ein, setzen Sie den Schalter wieder zurück in die untere Position und starten Sie das Gateway neu.

Beispiele finden Sie im Anhang dieser Versuchsanleitung.

4.7 Versuch 1: Inbetriebnahme der USB-Gateways

Ziel dieses ersten Versuchs ist, dass Sie sich mit dem CAN-Bus im Allgemeinen vertraut machen und erste CAN-Nachrichten übertragen. Hierfür wird zunächst ein relativ simpler Versuchsaufbau verwendet.

Schließen Sie zwei der USB-CAN-Gateways an einen der beiden PCs an und verbinden Sie die beiden Module via CAN-Flachbandkabel.

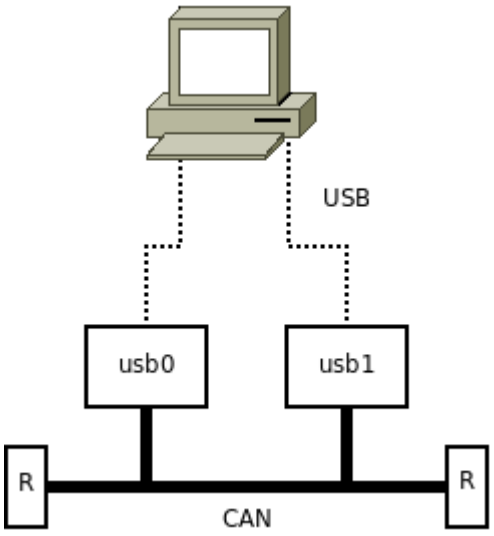

Abb. 12: Schema des ersten Versuchsaufbaus

Starten Sie nun das Programm `PCANview`. Im ersten Fenster müssen Sie zwei Werte einstellen, die `Device Nr.` des ersten Gateways (ja nachdem was Sie eingestellt haben, ist dies z. B. die `0`) und die Übertragungsrate (z. B. 1 Mbit/s). Die Einstellungen im zweiten Fenster sind zunächst nicht relevant, Sie brauchen diese daher lediglich zu bestätigen. Die rote LED sollte danach nicht mehr leuchten.

Starten Sie `PCANview` ein weiteres Mal und wiederholen Sie den Vorgang mit dem zweiten USB-Modul. Achten Sie auch hierbei drauf, dass Sie die korrekte `Device Nr.` wählen und die Übertragungsrate mit der des ersten Gateways identisch ist.

Erstellen Sie nun die erste zu übertragende Nachricht. Wählen Sie hierfür im Fenster eines der beiden Module im Menü „`Transmit > New`“, geben Sie eine ID ein und wählen Sie im Datenteil bis zu acht beliebige Byte. Die Periode sollten Sie zu Beginn auf `0` setzen, wodurch die Nachricht nicht automatisch verschickt wird.

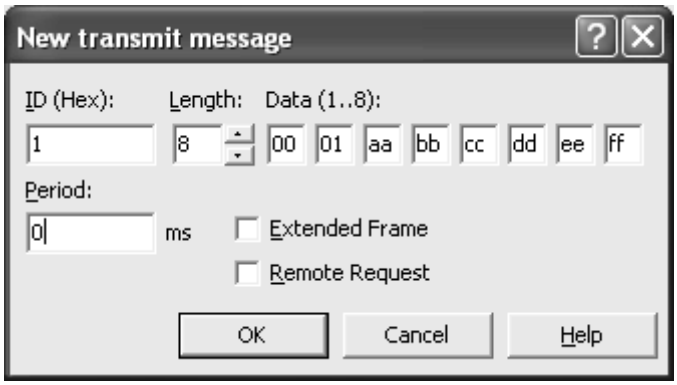

Abb. 13: Nachrichtenerstellungsfenster in der Software PCAN-view

In der unteren Hälfte des Moduls sehen Sie nun die von Ihnen angelegte Nachricht. Zum

Versenden müssen Sie diese auswählen und die Leertaste drücken. **Was stellen Sie fest?**

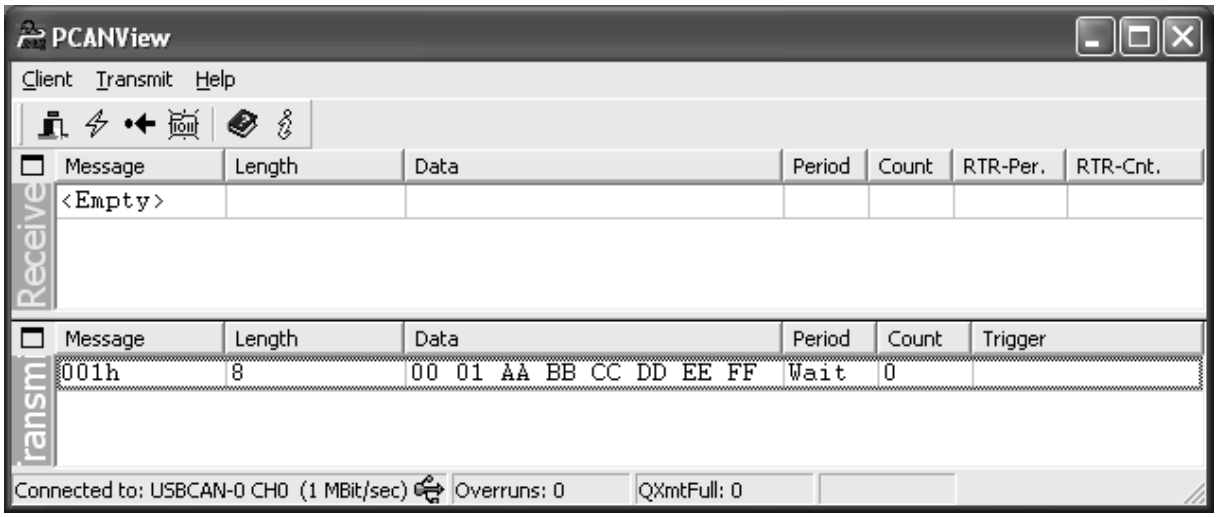

Abb. 14: Hauptfensters der Software PCANview

> *Das Controller Area Network ist lediglich bis zur OSI-Schicht 2 beschrieben. Welche Daten Sie übertragen, ist in diesem Laborversuch demnach irrelevant. Um unterschiedliche Nachrichten besser auseinanderhalten zu können, empfiehlt es sich jedoch die Bytes so zu verwenden, dass sie sich leicht zuordnen lassen. Verwenden Sie z. B. das erste Byte für die Geräte-ID der Nachrichtenquelle und das zweite Byte für das „Ziel". Eine Nachricht mit dem Inhalt* `00 01 AA BB CC DD EE FF` *würde demnach von Modul 0 an Modul 1 geschickt.*

Erweitern Sie nun den Versuch. Wählen Sie beispielsweise eine andere Periode und/oder erstellen Sie eine weitere Nachricht. Versenden Sie außerdem Nachrichten mit dem anderen Modul und nehmen Sie auch das dritte Gateway in Betrieb. **Was stellen Sie fest?**

Wählen Sie nun für die USB-Gateways unterschiedliche CAN-Bitraten. **Was passiert, wenn Sie Nachrichten versenden?**

Probieren Sie außerdem aus was geschieht, wenn Sie bei der Erstellung von Nachrichten Haken bei `Extended Frame` bzw. `Remote Request` setzen. **Wofür werden diese Nachrichtentypen verwendet?**

4.8 Versuch 2: Blick auf Layer 1

Ziel dieses Versuches ist, dass Sie das Controller Area Network auch auf OSI-Schicht 1 kennen lernen.

Schließen Sie hierfür zunächst wieder zwei der USB-CAN-Gateways an einen der beiden PCs an und verbinden Sie die Module ebenfalls mit dem terminierten CAN-Flachbandkabel. Um mit dem Oszilloskop die Signale vom Bus verarbeiten zu können, benötigen Sie ferner den D-Sub-Stecker. Schließen Sie diesen ebenfalls an das Flachbandkabel an und konnektieren Sie das Oszilloskop wiederum mit den freistehenden Adern 3 und 4.

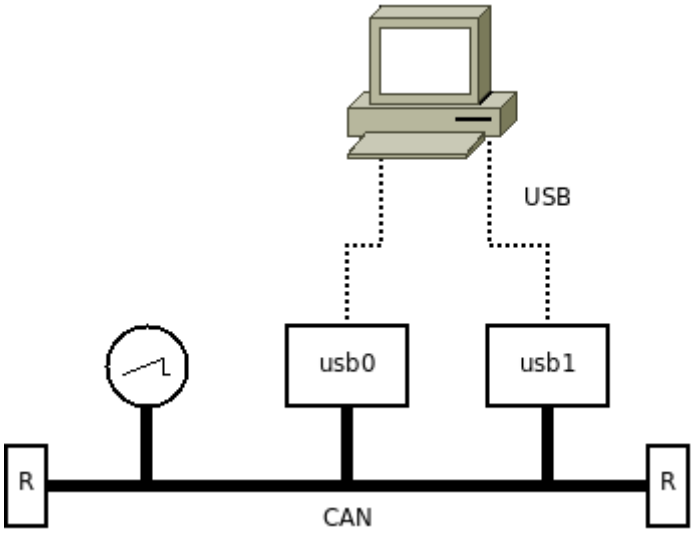

Abb. 15: Schema des zweiten Versuchsaufbaus

Starten Sie, wie bereits im ersten Versuch, das Programm `PCANview` einmal für jedes Gateway. Wählen Sie zunächst eine niedrige Bitrate, z. B. 125 kbit/s. Senden Sie von dem einen zum anderen Modul vorerst *kurze* CAN-Nachrichten, beispielsweise mit dem Datenteil `AA AA`, im Abstand von 50 ms. **Was ist auf dem Oszilloskop (im Digitalmodus) zu sehen?**

Variieren Sie nun die Bytes und die Länge der CAN-Nachricht, sowie die Bitrate. **Was fällt Ihnen auf?**

Bitte beantworten Sie außerdem die folgenden Fragen bzw. lösen Sie die folgenden Aufgaben:

- **Wie groß ist die Spannungsdifferenz der beiden Zustände? Wie groß müsste sie sein?**
- **In welchem Modus wird der Bus betrieben? Welchen Modus gibt es außerdem?**
- **Welcher logische Zustand besitzt welchen Pegel?**
- **Im rezessiven Zustand bzw. im Ruhezustand liegt der Spannungspegel laut Standard bei 2,5 V. Warum können Sie diesen Wert nicht direkt messen?**
- **Weisen Sie nach, dass beim CAN das Prinzip des *Bitstuffings* verwendet wird.**
- **Finden Sie heraus, welche Ader den Pegel CAN_H bzw. CAN_L führt.**
- **Entfernen Sie zuerst einen, dann beide Terminatoren. Ließe sich der CAN-Bus**

auch ohne Endwiderstand betreiben?

- **Was passiert, wenn Sie die Adern 3 und 4 kurzschließen?**

4.9 Versuch 3: Inbetriebnahme der Ethernetgateways

Nun, da Sie erste Erfahrungen mit dem Controller Area Network gemacht haben, ist es an der Zeit einen Schritt weiter zu gehen. Ziel dieses dritten Versuchs ist es daher ein komplexeres Netzwerk aufzubauen und eine weitere Auswertungsmethode zu verwenden.

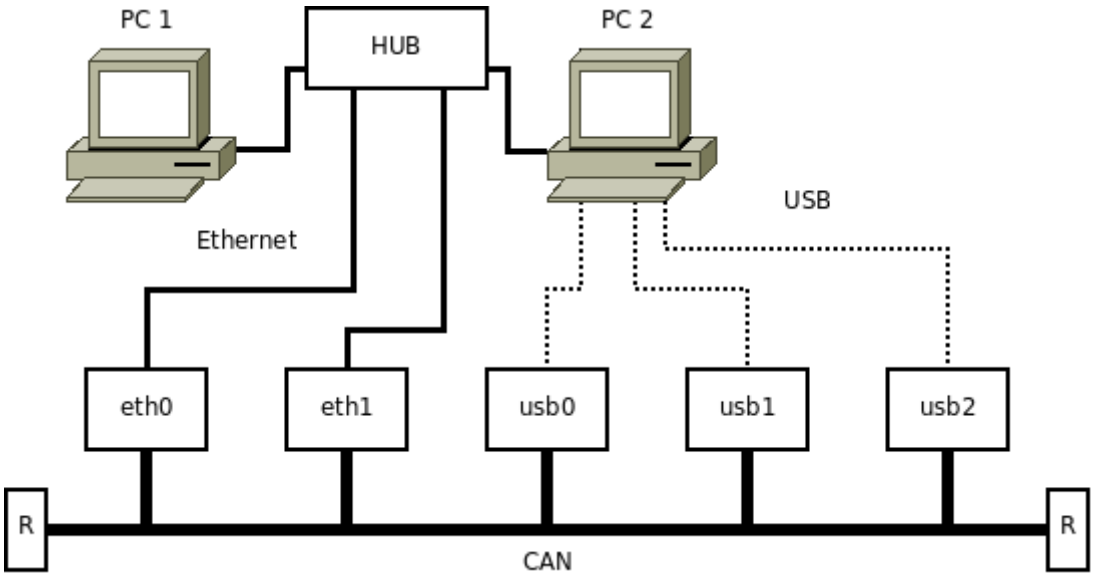

Abb. 16: Schema des dritten Versuchsaufbaus

Verbinden Sie zunächst die drei USB-CAN-Gateways mit einem der beiden PCs (nachfolgend *PC 2* genannt, analog dazu heißt der andere Rechner *PC 1*) sowie mit dem terminierten CAN-Flachbandkabel. An das Flachbandkabel schließen Sie dann außerdem die beiden Ethernet-CAN-Gateways an. Zum Schluss nutzen Sie den Ethernethub um die beiden PCs und die Ethernet-CAN-Gateways in einem LAN zu vereinigen. Sollten Sie die Gateways LAN- und CAN-seitig noch nicht konfiguriert haben, holen Sie dies bitte nach.

Verwenden Sie in diesem Laborversuch generell nur einen Hub und keinen Switch! Aufgrund eines Fehlers versuchen die beiden Ethernet-CAN-Gateways ansonsten ständig eine einmalig versendete Nachricht erneut zuzustellen, wodurch der Bus ausgelastet wird. Außerdem ermöglicht Ihnen ein Hub den Mitschnitt von Ethernetpaketen im LAN.

Starten Sie nun die Software `CAN-REport` und wählen Sie das erste Ethernetmodul als Datenquelle aus. Klicken Sie hierfür im Menü zunächst auf „`Connection > CAN-Interface`“.

Es öffnet sich ein neues Fenster. Wählen Sie als `CAN-Interface` den „`SYS TEC Wrapper`“ aus. Als `Channel` wählen Sie Kanal `0`. Klicken Sie danach auf die Schaltfläche „`Run Wrapper Config Tool`“.

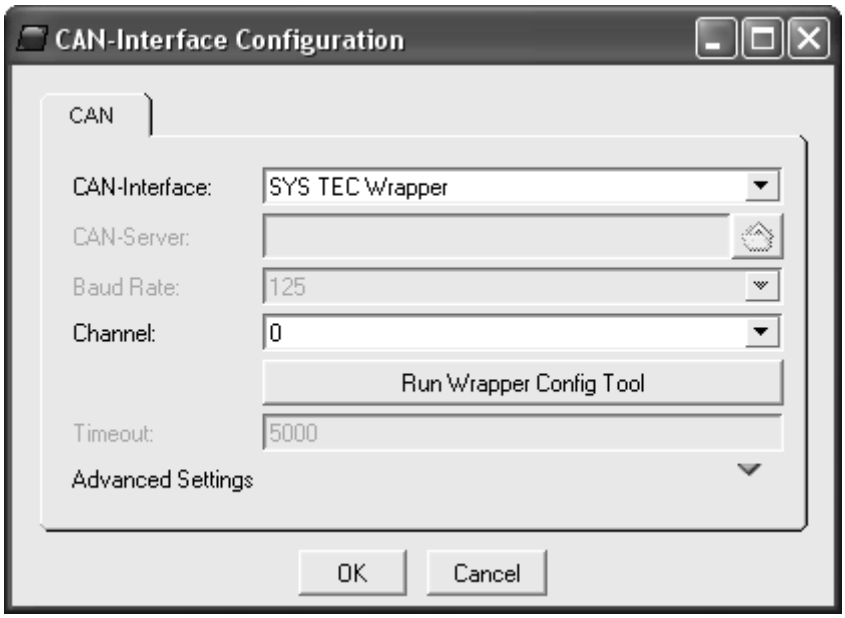

Abb. 17: Fenster für die Schnittstellenkonfiguration in der Software CAN-RE-port

Im neuen Fenster setzen Sie den Wert der `Instanz` ebenfalls auf `0`. Als `Hardware` wählen Sie das `CAN-Ethernet-Gateway` aus. Klicken Sie dann auf die Schaltfläche „`Eigenschaften`“.

Abb. 18: Fenster für die Hardwareeinstellungen in der Software CAN-RE-port

In diesem dritten Fenster tragen Sie zuerst die `IP-Adresse` des Gerätes ein. Stellen Sie ferner

sicher, dass der `Port` korrekt eingetragen ist (die Vorgabe ist `8234`) und dass Sie das richtige Protokoll gewählt haben.

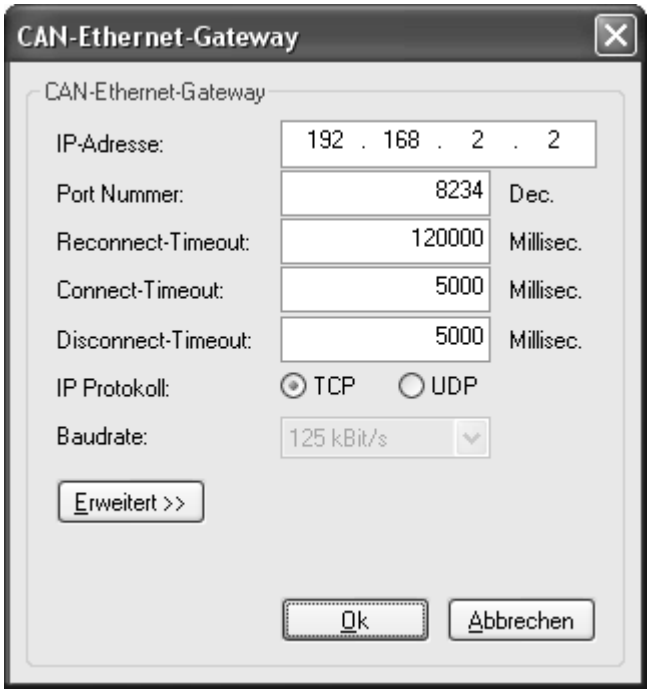

Abb. 19: Fenster für die Gatewayeinstellungen in der Software CAN-REport

Ist das Gateway fertig konfiguriert, kann der Mitschnitt des Datenverkehrs beginnen. Klicken Sie hierfür im Menü der Software `CAN-REport` auf „`Connection > Connect`“.

Erzeugen Sie nun am anderen PC mit der Software `PCANview` und den drei USB-Modulen Datenverkehr. Achten Sie dabei darauf, dass alle fünf CAN-Gateways mit der selben Bitrate betrieben werden.

Ist alles korrekt eingestellt, werden im Hauptfenster von `CAN-REport` nun CAN-Frames dargestellt. **Was wird Ihnen angezeigt? Welche Darstellungsoptionen stehen Ihnen zur Verfügung?**

Wählen Sie dann im Menü unter „`View > View Mode`“ den `Object View` aus. **Wo besteht der Unterschied zum `Trace View`? Welcher Modus lässt sich für welchen Zweck verwenden?**

Versenden Sie schließlich auch aus der Software `CAN-REport` heraus Nachrichten. Wählen Sie hierfür einen der unteren Reiter (T1 bis T10) und geben Sie die entsprechenden Werte für ID, Länge, Datenbytes und Wiederholungszeit ein. Durch Aktivierung bzw. Deaktivierung der Schaltfläche „`Send`“ lässt sich die Übertragung ein- bzw. ausschalten.

Variieren Sie Nachrichtenlänge und -anzahl, Periode und ID der Nachrichten. **Was stellen Sie fest?**

4.10 Versuch 4: Fehlererzeugung im CAN

Sie sind mittlerweile in der Lage ein Controller Area Network aufzubauen und zu betreiben. Bisher haben Sie feststellen können, wie sich das Netzwerk bei korrekter Konfiguration verhält. In diesem Versuch gilt es nun herauszufinden, was geschieht wenn Fehler auftreten. Zu diesem Zweck bauen Sie bitte im ersten Schritt das Netzwerk genauso auf wie im vorangegangenen Teilversuch.

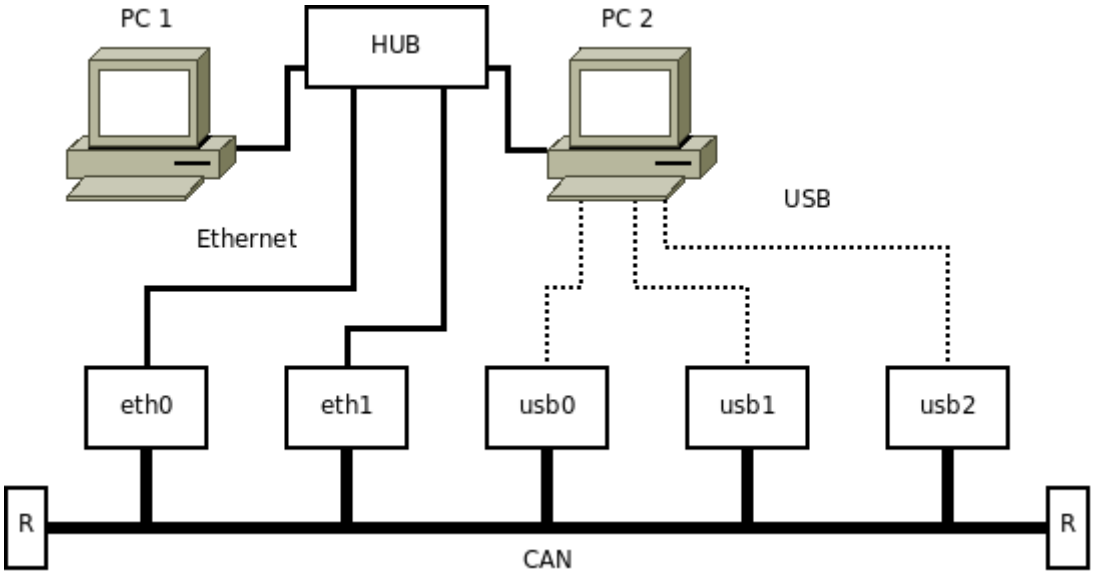

Abb. 20: Schema des vierten Versuchsaufbaus

I Unterschiedliche Bitraten

Beschäftigen Sie sich zunächst mit der Übertragungsrate: Bisher wurden alle Module mit der selben Bitrate betrieben. Betreiben Sie nun *einen* der CAN-Knoten in einem anderen Geschwindigkeitsmodus und übertragen Sie von diesem vorerst *eine* Nachricht. **Was stellen Sie fest? Sind Ihre Ergebnisse identisch mit denen des ersten Versuchs?**

Versuchen Sie nun von einem anderen Modul aus Daten zu übertragen. **Was wird gesendet bzw. empfangen? Welche Fehler treten auf und wie unterscheiden sich diese? Was passiert, wenn Sie das Modul mit der abweichenden Bitrate trennen? Gibt es generell eine Möglichkeit in einem Controller Area Network Module mit unterschiedlichen Übertragungsraten zu betreiben?**

> ⚠ *Tritt beim Betrieb der USB-CAN-Gateways ein Fehler auf, so schaltet die Software* `PCANview` *automatisch wieder in den Betriebsmodus zurück. Welcher Fehler zuletzt aufgetreten ist, wird Ihnen jedoch rechts unten angezeigt.*

II Auslastung des Busses

Versuchen Sie nun herauszufinden wie sich der CAN-Bus unter Last verhält. Stellen Sie hierfür die Übertragungsraten auf allen Modulen vorerst auf 1 Mbit/s ein. Starten Sie danach die Software `CAN-REport` und verbinden Sie diese mit einem der beiden Ethernet-CAN-Gateways. Verwenden Sie schließlich `PCANview` um mit allen drei USB-CAN-Gateways jeweils eine 8 Byte lange Nachricht an ein anderes CAN-Modul zu versenden. Setzen Sie die Periode auf 1 ms um möglichst viel Datenverkehr zu produzieren.

Schaltet Sie nun in `CAN-REport` in den `Object View` um eine aussagekräftige Verkehrsstatistik zu erhalten. **Was stellen Sie fest?**

Sobald Sie den Ansichtsmodus wechseln, wird die Verkehrsstatistik wieder auf Null gesetzt. Diesen Effekt können Sie auch herbeiführen, wenn Sie im Menü auf „`Edit > Clear Log`“ klicken.

Beachten Sie, dass die Werte der Counter mit großer Wahrscheinlichkeit eine leichte Abweichung aufweisen werden. Dies ist darauf zurückzuführen, dass die Aufzeichnung des Traffics quasi zu einem zufälligen Zeitpunkt beginnt und somit nicht exakt die gleiche Anzahl an CAN-Rahmen erfasst werden kann. Die Differenz der Werte sollte aber unter 100 Zählern liegen.

Setzen Sie nun die Übertragungsrate auf 125 kbit/s herab und starten Sie den Vorgang erneut. Senden Sie dabei mit den USB-Modulen *genau die gleichen* Nachrichten wie zuvor. Setzen Sie den Wert der Periode außerdem wieder auf 1 ms. Vergleichen Sie die Messwerte mit denen, die Sie zuvor erhalten haben. **Was stellen Sie fest? Könnte sich diese Eigenschaft des CAN-Busses sinnvoll nutzen lassen?**

III Verwendung der gleichen ID

Verändern Sie nun die Nachrichten, die die USB-Gateways versenden. Statt sich gegenseitig Nachrichten zuzusenden, werden diese nun alle an die gleiche ID geschickt. Belassen Sie die Bitrate zunächst bei 125 kbit/s. **Was stellen Sie fest? Erklären Sie, warum diese Konfiguration zu Fehlern führt**.

Setzen Sie danach die Übertragungsrate wieder auf 1 Mbit/s hoch und starten Sie den Vorgang erneut. **Treten Fehler auf? Wenn ja, warum?**

Beantworten Sie abschließend bitte noch die folgenden Fragen:

- **Warum kann ein Knoten nur die Signale „seiner“ Übertragungsrate detektieren?**
- **Ist es sinnvoll, wenn ein Sender Nachrichten mit mehreren Identifiern verschicken kann?**
- **Ist es sinnvoll, wenn mehrere Sender Nachrichten mit dem gleichen Identifier verschicken können?**

4.11 Versuch 5: Inbetriebnahme der Ethernetbrücke

In diesem fünften Teilversuch werden Sie nun zwei Controller Area Networks aufbauen und diese mittels Ethernet „verbinden“. Ziel ist es zunächst grundsätzlich das Prinzip der Überbrückung kennen zu lernen.

Schließen Sie hierfür die drei USB-CAN-Gateways per USB an einen der beiden PCs an. Verbinden Sie dann, falls noch nicht geschehen, die beiden Ethernet-CAN-Gateways und die beiden PCs mit dem Ethernethub um ein LAN aufzubauen. Letztlich benötigen Sie noch die beiden terminierten Flachbandkabel für zwei CAN-Busse. Verbinden Sie jedes Kabel mit jeweils einem Ethernet-CAN-Gateway und einem bzw. zwei USB-Modulen, wie in nachfolgender Grafik dargestellt:

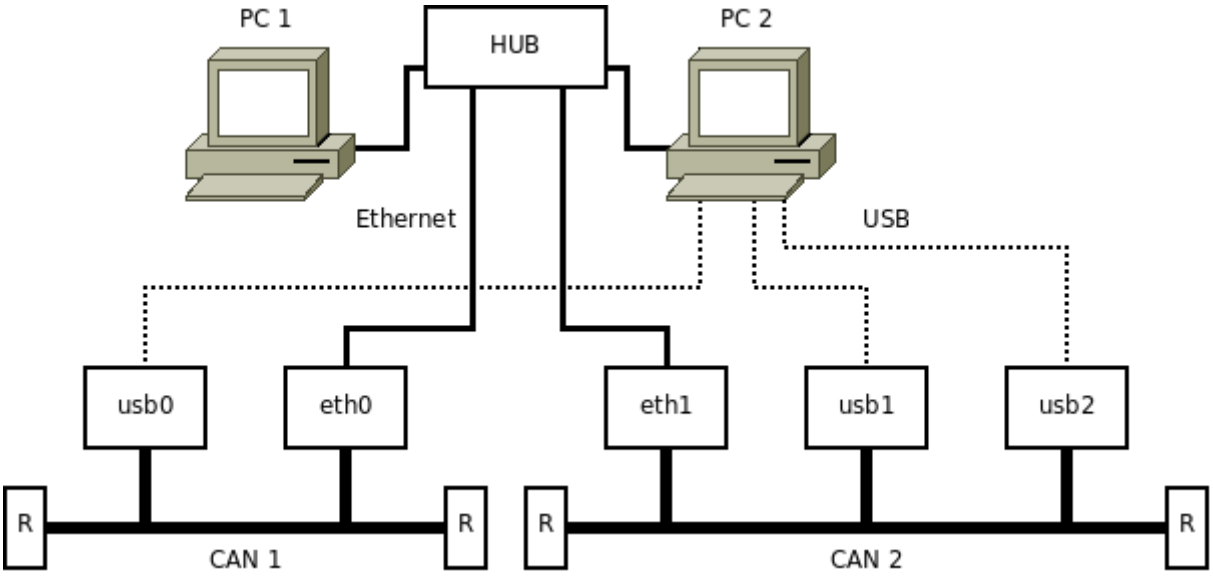

Abb. 21: Schema des fünften Versuchsaufbaus

Wählen Sie für jeden der beiden CAN-Busse zunächst eine Übertragungsrate von 125 kbit/s.

Stellen Sie mit dem Befehl `/if/tcpserv0` sicher, dass ein TCP-Server auf beiden Ethernetmodulen läuft. Ist dies nicht der Fall, starten Sie diesen bitte. Legen Sie nun auf jedem Gateway zusätzlich einen TCP-Client an. Nutzen Sie hierfür zunächst folgenden Befehl:

```
mkif tcp
```

Legen Sie nun die IP-Adresse des *Ziel*servers und dessen Port fest und schalten Sie die Verbindung frei. Verwenden Sie hierfür den Befehl `/if/tcp0/ to:<IP-Adresse>:<Port> on alive:1 reco:1`. Haben Sie die Standardeinstellungen verwendet, sieht die Konfiguration für das erste Modul demnach wie folgt aus:

```
/if/tcp0 to:192.168.2.3:8234 on alive:1 reco:1
```

Ein Fehler im Modul verhindert, dass Sie einen stabilen UDP-Client starten können. Sollten Sie versuchen dies zu tun, wird sich das Gateway mit hoher Wahrscheinlichkeit neu starten und alle neuen Einstellungen wieder löschen. Verwenden Sie aus diesem Grund besser das TCP. Möchten Sie dennoch versuchen diesen Teilversuch mit dem UDP durchzuführen, so erstellen und konfigurieren Sie den Client, analog zum TCP, mit folgenden Befehlen:

```
mkif udp
/if/udp0 to:<IP-Adresse>:<Port> on alive:1 reco:1
```

Testen Sie nun ob die Brücke funktioniert, indem Sie mit der Software `PCANview` eine CAN-Nachricht über diese schicken. Erhalten Sie eine Antwort, starten Sie darüber hinaus die Software `Wireshark` auf dem anderen PC (*PC 1*) und schneiden Sie den Datenverkehr im LAN mit. Filtern Sie, um nur relevante Pakete angezeigt zu bekommen, mit dem Befehl `ip.addr == <IP-Adresse>` den Datenstrom. Standardmäßig entspricht der Filter damit folgender Zeile:

```
ip.addr == 192.168.2.2 || ip.addr == 192.168.2.3
```

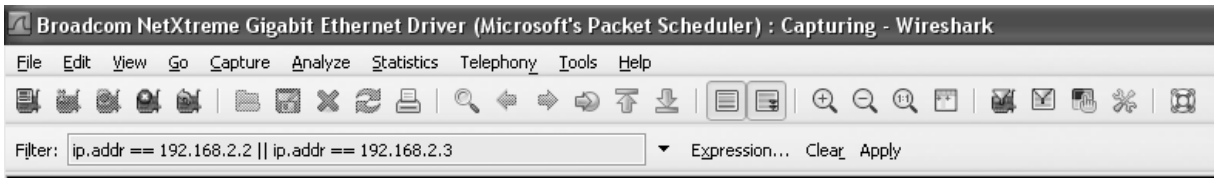

Abb. 22: Feld für die Filtereinstellung in der Software Wireshark

Analysieren Sie die Ethernetpakete. **Wann werden zwischen den beiden Ethernet-CAN-Gateways Nachrichten ausgetauscht? Welchen Zweck erfüllen diese Nachrichten? Finden Sie die ursprüngliche CAN-Nachricht? Finden Sie ggf. weitere Parameter?**

Variieren Sie die Nachricht und versenden Sie CAN-Frames von mehreren USB-Modulen gleichzeitig. Setzen Sie die Sendeperiode auf 1 ms. **Was stellen Sie fest?**

Ändern Sie nun im gesamten ersten CAN-Segment (*CAN 1* in der Grafik) die Bitrate auf 1 Mbit/s. Übertragen Sie danach erneut CAN-Nachrichten über das LAN. **Was stellen Sie fest?**

Finden Sie eine Erklärung.

Erzeugen Sie schließlich eine hohe Last auf dem CAN-Bus. **Treten Fehler auf? Wenn ja, welche?**

Falls der UDP-Server bei Ihnen stabil läuft, wiederholen Sie schließlich den Teilversuch mit dem UDP. Löschen Sie hierfür zunächst auf beiden Ethernet-CAN-Gateways den TCP-Client (nicht den TCP-Server!). Vergleichen Sie schließlich Ihre Ergebnisse mit denen der UDP-Messung. **Was stellen Sie fest? Welche Unterschiede bestehen zwischen den beiden Varianten?**

4.12 Versuch 6: Fehlererzeugung im Ethernet

Sie haben bereits die Ethernetbrücke in Betrieb nehmen können und außerdem festgestellt, wie CAN-Nachrichten über das LAN transportiert werden. Ziel dieses letzten Teilversuchs ist es nun herauszufinden wo die Schwächen des Ethernets liegen.

Verändern Sie den grundsätzlichen Versuchsaufbau nicht. Er sollte demnach folgender Abbildung entsprechen:

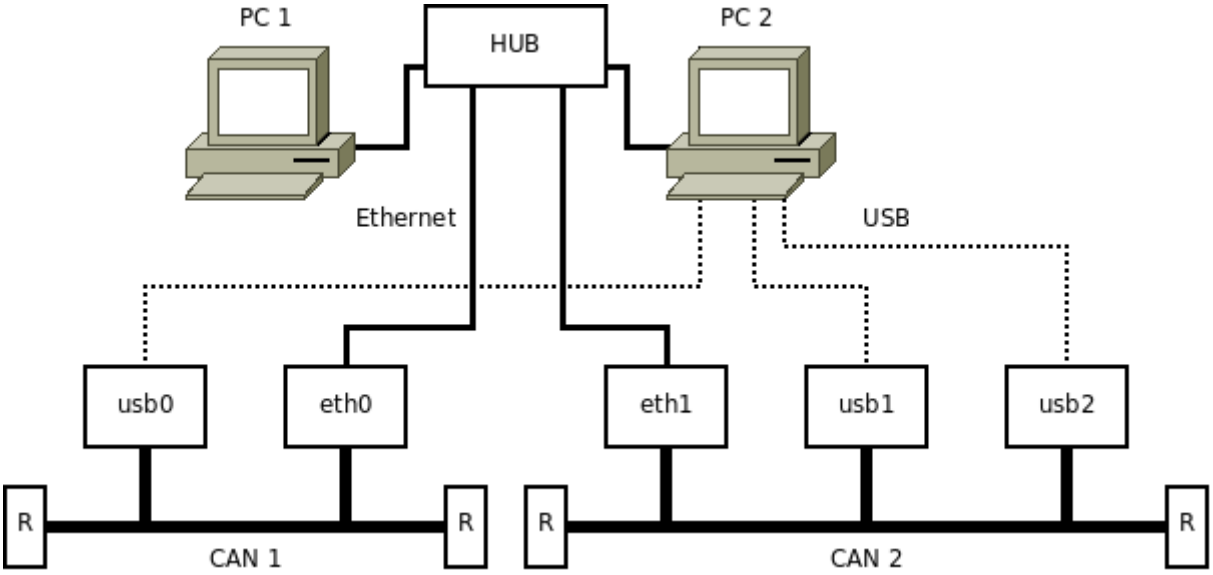

Abb. 23: Schema des sechsten Versuchsaufbaus

Setzen Sie die Bitrate in beiden CANs auf 1 Mbit/s. Behalten Sie die TCP-Einstellungen auf den Ethernet-CAN-Gateways bei. Löschen Sie ggf. die UDP-Clients.

Starten Sie danach auf *PC 2* die Software `PackETH` um Last im LAN zu generieren. Klicken Sie hier als erstes auf die Schaltfläche „`Interface`“ und wählen Sie die Netzwerkkarte aus, mit der der Rechner mit dem Ethernet verbunden ist.

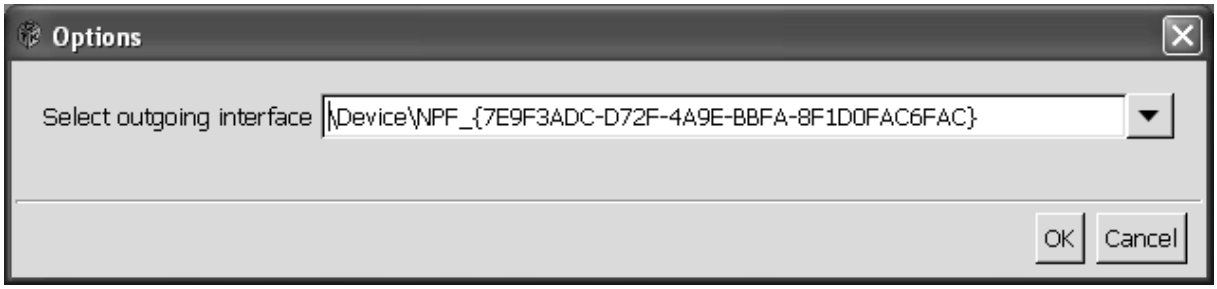

Abb. 24: Fenster für die Auswahl der Netzwerkschnittstelle in der Software PackETH

Geben Sie im zweiten Schritt im Hauptfenster unter der Rubrik „`Link Layer`“ die MAC-Adressen der Netzwerkkarten der beiden Computer ein. Achten Sie hierbei darauf, dass dies in der Form `00:11:22:33:44:55` geschieht, weil andere Schreibweisen nicht verarbeitet werden können.

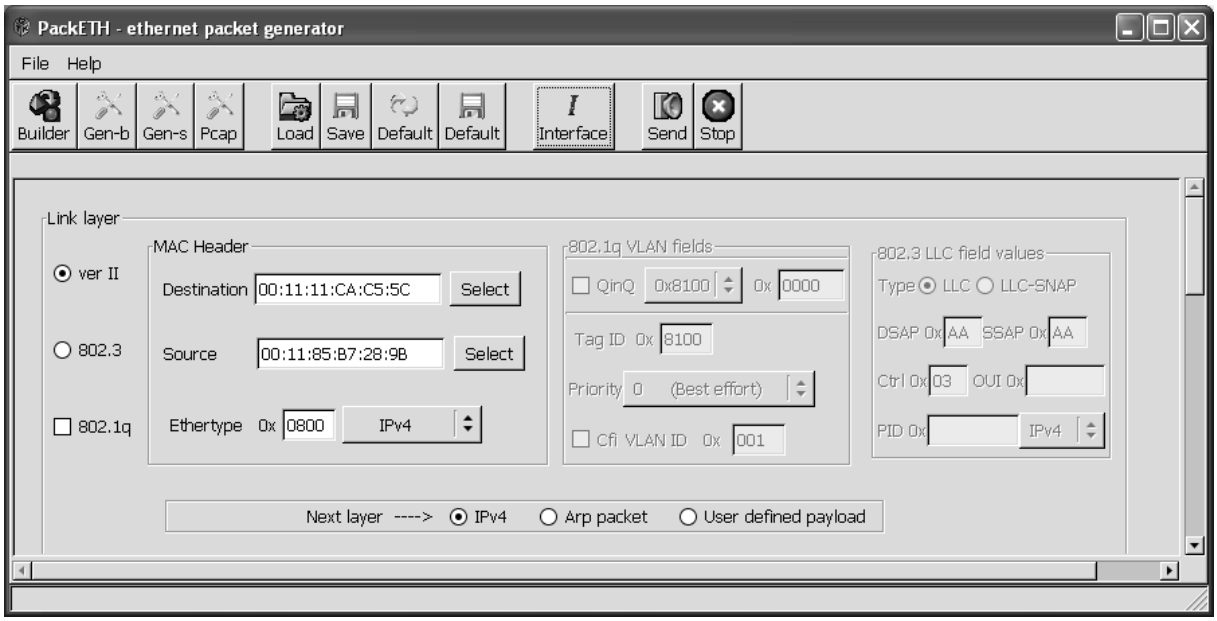

Abb. 25: Rubrik für die MAC-Einstellungen im PackETH-Hauptfenster

Als drittes müssen in der Rubrik „`IPv4 Data`“ die IP-Adressen der beiden PCs ergänzt werden. Hier ist die übliche Dezimalschreibweise zu verwenden.

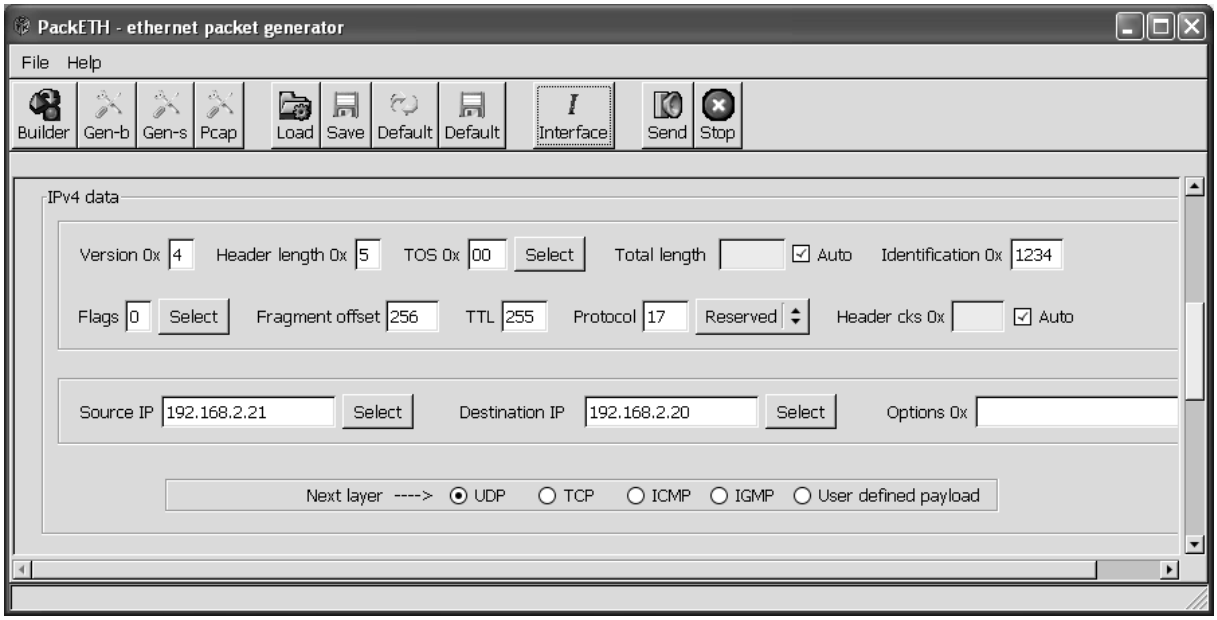

Abb. 26: Rubrik für die IP-Einstellungen im PackETH-Hauptfenster

> ⚠ *Falls Sie die MAC- und IP-Adressen nicht kennen, starten Sie in Windows XP ein Terminal und geben Sie den Befehl* `ipconfig -all` *ein. So werden Ihnen alle relevanten Informationen angezeigt.*

Ergänzen Sie danach noch die Daten in der Rubrik „`UDP Data`“. Wählen Sie hier zunächst den Quell- und den Zielport aus, z. B. jeweils Port `80`. Legen Sie am Schluss noch eine Payload fest. Diese muss aus Hexadezimalwerten bestehen und sollte, um die Vergleichbarkeit Ihrer Ergebnisse zu gewährleisten, 64 Byte groß sein.

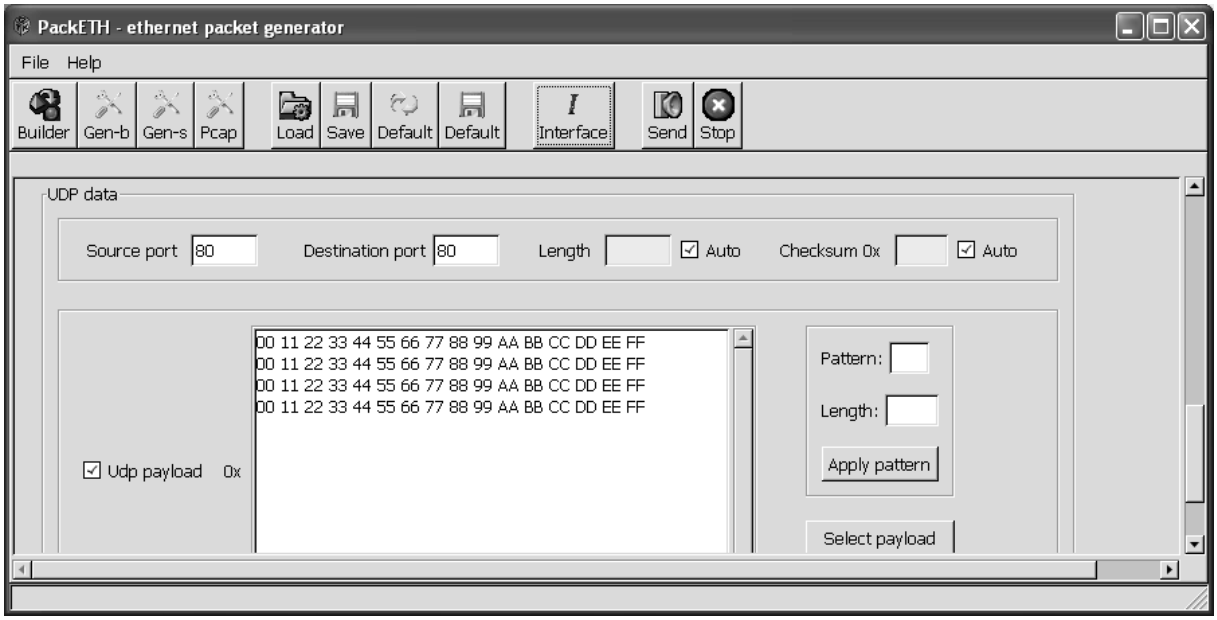

Abb. 27: Rubrik für die UDP-Einstellungen im PackETH-Hauptfenster

Klicken Sie nun noch auf die Schaltfläche „`Gen-b`“ und stellen Sie den Wert der zu sendenden Pakete auf Unendlich (Haken bei „`Infinite`“). Setzen Sie die Verzögerung zwischen den Paketen zunächst auf 1000 µs.

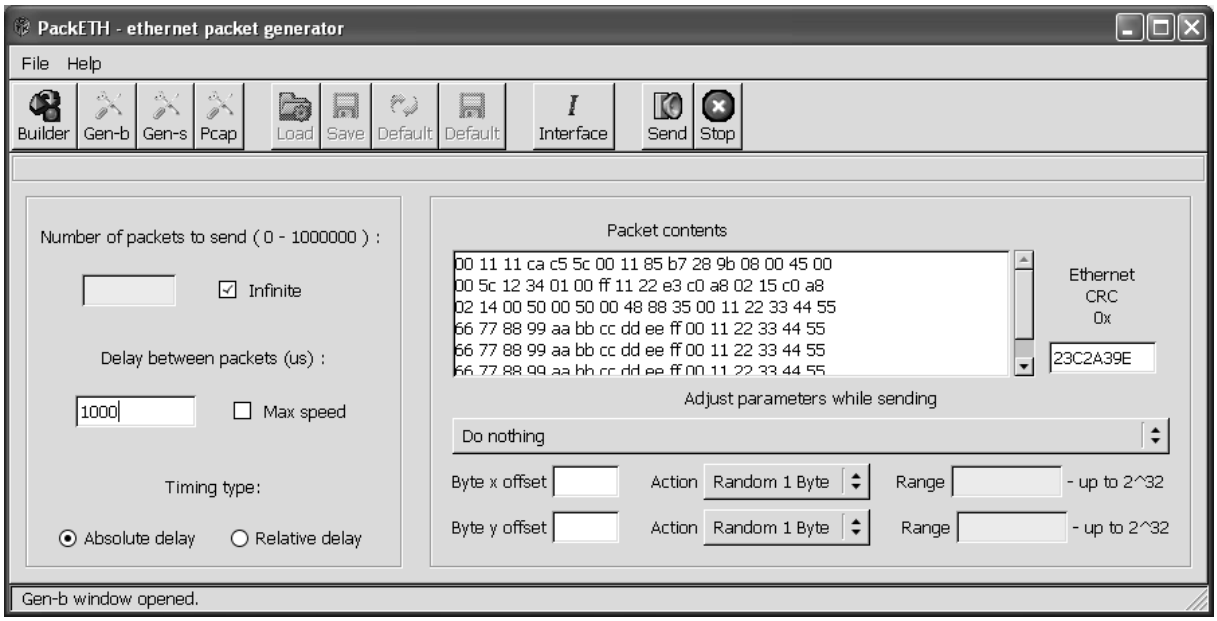

Abb. 28: Fenster für die Paketeinstellungen in der Software PackETH

Beim Neustart der Software `PackETH` *gehen alle Einstellungen verloren. Speichern Sie diese daher ab, wenn Sie vorhaben den Versuch zu unterbrechen und zu einem späteren Zeitpunkt fortzuführen.*

Um die aktuelle Busauslastung zu messen, starten Sie bitte ferner die Software `Traffic Graph` auf *PC 1*.

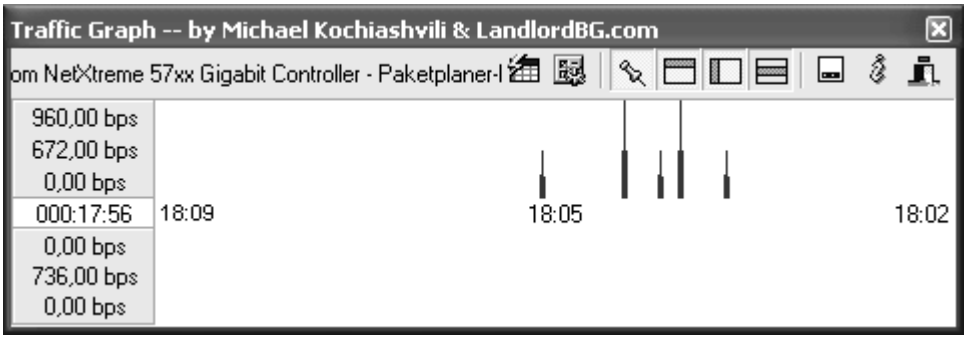

Abb. 29: Hauptfenster der Software Traffic Graph

Beginnen Sie mit der Nachrichtenübertragung über die Ethernetbrücke. Schicken Sie hierfür im Abstand von 1 ms 8 Byte große CAN-Frames vom Gateway *usb0* aus über die Brücke. **Wie hoch ist die gemessene Last im Ethernet?**

> ⚠ *Die Software* `Traffic Graph` *ist nicht in der Lage sehr niedrige Busauslastungen anzuzeigen. Falls keine Messwerte geliefert werden, greifen Sie für die erste Messung auf den „*`IO Graph`*“ der Software* `Wireshark` *zurück.*

Starten Sie parallel die Datenübertragung der Software `PackETH`, indem Sie auf „`Send`“ klicken. **Wie hoch ist die gemessene Last im Ethernet jetzt?**

Reduzieren Sie nun in `PackETH` schrittweise die Verzögerung zwischen den Paketen. Beachten Sie bitte, dass Sie vor der Veränderung des Wertes auf „`Stop`“ klicken müssen um die aktuelle Übertragung zu beenden. **Ab welchem Verzögerungswert bemerken Sie Veränderungen? Wie machen sich diese Veränderungen bemerkbar? Wie hoch ist die Netzlast in diesem Moment? Können Sie die Verzögerung noch weiter reduzieren?**

Beantworten Sie zum Schluss bitte noch folgende Fragen bzw. lösen Sie folgende Aufgaben, bevor Sie Ihre Ergebnisse auswerten:

- **Warum erreichen Sie keine LAN-Auslastung von 10 Mbit/s bzw. 100 %?**
- **Eignet sich ein Ethernet für den Echtzeitbetrieb? Erklären Sie, warum dies so ist.**
- **Für welche Zwecke lässt sich eine Ethernet-CAN-Brücke in einer Echtzeitumgebung verwenden? Wo liegen die Vor- und Nachteile dieser Technologie?**
- **Welche Bedingungen müssen geschaffen werden um die Ethernet-CAN-Brücke in einer Echtzeitumgebung einzusetzen?**
- **Nennen Sie Vor- und Nachteile von Ethernet und CAN.**

- **Nennen Sie die wesentlichen Unterschiede zwischen einem regulären Ethernet- und einem regulären CAN-Frame.**

4.13 Anhang

I Konfigurationsdatei von eth0, Versuch 3 und 4

```
siocfg 9600
ipcfg 192.168.2.2 255.255.255.0 192.168.2.1
ipaccept
mkif led
/if/led0/fin +l +hd1
mkif can
mkif tcpserv
/if/can0 bus:0 on baud:0 canid:05
```

II Konfigurationsdatei von eth1, Versuch 3 und 4

```
siocfg 9600
ipcfg 192.168.2.3 255.255.255.0 192.168.2.1
ipaccept
mkif led
/if/led0/fin +l +hd1
mkif can
mkif tcpserv
/if/can0 bus:0 on baud:0 canid:06
```

III Konfigurationsdatei von eth0, Versuch 5, TCP

```
siocfg 9600
ipcfg 192.168.2.2 255.255.255.0 192.168.2.1
ipaccept
mkif led
/if/led0/fin +l +hd1
mkif can
mkif tcpserv
```

```
mkif tcp
/if/tcp0 to:192.168.2.3:8234 on alive:1 reco:1
/if/can0 bus:0 on baud:4 canid:05
```

IV Konfigurationsdatei von eth1, Versuch 5, TCP

```
siocfg 9600
ipcfg 192.168.2.3 255.255.255.0 192.168.2.1
ipaccept
mkif led
/if/led0/fin +l +hd1
mkif can
mkif tcpserv
mkif tcp
/if/tcp0 to:192.168.2.2:8234 on alive:1 reco:1
/if/can0 bus:0 on baud:4 canid:06
```

V Konfigurationsdatei von eth0, Versuch 5, UDP

```
siocfg 9600
ipcfg 192.168.2.2 255.255.255.0 192.168.2.1
ipaccept
mkif led
/if/led0/fin +l +hd1
mkif can
mkif udpserv
mkif udp
/if/udp0 to:192.168.2.3:8234 on alive:1 reco:1
/if/can0 bus:0 on baud:4 canid:05
```

VI Konfigurationsdatei von eth1, Versuch 5, UDP

```
siocfg 9600
ipcfg 192.168.2.3 255.255.255.0 192.168.2.1
ipaccept
mkif led
```

```
/if/led0/fin +l +hd1
mkif can
mkif udpserv
mkif udp
/if/udp0 to:192.168.2.2:8234 on alive:1 reco:1
/if/can0 bus:0 on baud:4 canid:06
```

VII Konfigurationsdatei von eth0, Versuch 6

```
siocfg 9600
ipcfg 192.168.2.2 255.255.255.0 192.168.2.1
ipaccept
mkif led
/if/led0/fin +l +hd1
mkif can
mkif tcpserv
mkif tcp
/if/tcp0 to:192.168.2.3:8234 on alive:1 reco:1
/if/can0 bus:0 on baud:0 canid:05
```

VIII Konfigurationsdatei von eth1, Versuch 6

```
siocfg 9600
ipcfg 192.168.2.3 255.255.255.0 192.168.2.1
ipaccept
mkif led
/if/led0/fin +l +hd1
mkif can
mkif tcpserv
mkif tcp
/if/tcp0 to:192.168.2.2:8234 on alive:1 reco:1
/if/can0 bus:0 on baud:0 canid:06
```

5 Hintergrund des Laborversuchs

5.1 Allgemeines

Der Fokus bei der Entwicklung des Laborversuchs wurde insbesondere auf den Mehrwert der Aufgaben gelegt, welcher eindeutig der Wissenserwerb der Studierenden ist. Dies bringt bestimmte Anforderungen an die Aufgabenstellungen mit sich. So musste beispielsweise dafür Sorge getragen werden, dass die Teilversuche einerseits modular gestaltet sind, das neu erworbene Wissen aus der vorherigen Aufgabe also möglichst sofort wieder abgerufen werden sollte. Andererseits sollen auch möglichst alle Aspekte des CANs bzw. seiner Echtzeiteigenschaften betrachtet werden. Gleichzeitig durften die Aufgaben nicht zu viel Zeit für die Bearbeitung in Anspruch nehmen, mussten verständlich verfasst sein und sollten keine unnötigen Hürden enthalten. Insbesondere auf letzteren Aspekt wurde Wert gelegt um die VersuchsteilnehmerInnen nicht zu demotivieren[7]. Aus diesem Grund wurden einerseits „Infokästen“ mit wertvollen Tipps für die Versuchsdurchführung eingebaut und darüber hinaus komplexere Konfigurationsabläufe detailliert erläutert.

Die nachfolgenden Kapitel widmen sich den Hintergründen der einzelnen Teilversuche. Sie zeigen auf, welche Ziele der jeweilige Versuch verfolgt und ggf. in welcher Beziehung er zu den anderen steht. Darüber hinaus verdeutlichen die Abschnitte, welche Kenntnisse und Fähigkeiten die Studierenden bei der Versuchsdurchführung erwerben sollen. Daher eignen sich diese Kapitel auch für das Laborpersonal um evtl. einzelnen Versuchsgruppen Hilfestellungen zu bieten.

5.2 Hintergrund zu Versuch 1

Ziel des ersten Versuchs ist es die Studierenden zunächst mit der Hard- und Software des CANs vertraut zu machen. Hierfür wird das denkbar einfachste Netzwerk aufgebaut, bestehend aus dem Bus, zwei CAN-Knoten und einem PC zum Senden und Empfangen von Nachrichten. Bei der Erstkonfiguration der Komponenten werden die VersuchsteilnehmerInnen weitestgehend „an die Hand genommen“, damit sie nicht wertvolle Versuchszeit mit der Suche nach den richtigen Einstellungen vergeuden und sich schnell erste Erfolgserlebnisse einstellen.

Viel weiter als über das Senden und Empfangen von Nachrichten wird vorerst nicht hinausgegangen. Lediglich der erste „grobe“ Fehler, die Verwendung unterschiedlicher Bitraten, wird absichtlich generiert um den Studierenden in den folgenden Versuchen ggf. eine mühse-

7 Weitere Qualitätskriterien, siehe: *7 Test des Versuchs.*

lige Fehlersuche zu ersparen. Außerdem lernen die TeilnehmerInnen den `Extended Frame` und den `Remote Frame` kennen, was erste weitergehende theoretische Kenntnisse voraussetzt.

Nach Abschluss dieses Versuchs verfügen die Studierenden über folgende Fähigkeiten:

- Aufbau eines Controller Area Networks
- Umgang mit der Software `PCANview`
- Senden und Empfangen von CAN-Nachrichten

Außerdem erwerben sie hier folgende Kenntnisse:

- Alle CAN-Knoten müssen mit der gleichen Bitrate betrieben werden
- Der `Extended Frame` ermöglicht die Verwendung einer längeren ID
- Der `Remote Frame` wird für die Anforderung einer Nachricht verwendet

5.3 Hintergrund zu Versuch 2

Im zweiten Versuch werden die Studierenden aufgefordert das Controller Area Network auf der OSI-Schicht 1 zu betrachten. Ziel ist, dass die VersuchsteilnehmerInnen ein besseres Verständnis für die Funktionsweise des Feldbussystems erlangen.

Der bisherige Versuchsaufbau wird weitgehend beibehalten und lediglich um ein digitales Oszilloskop ergänzt. Um eine korrekte Darstellung auf dem Gerät zu ermöglichen, empfiehlt die Anleitung den Studierenden zunächst die Verwendung einer niedrigen Bitrate (125 kbit/s), einer niederratigen Periode (50 ms) und einer kleinen Payload (2 Byte). Mit diesen Einstellungen sind auf dem Bildschirm, auch durch die Verwendung der Auto-Set-Funktion, die übertragenen Bits deutlich zu erkennen. Durch Variationen der Payload, der Nachrichtenlänge und der Bitrate können die VersuchsteilnehmerInnen zunächst Veränderungen des Signals „live“ erleben.

Am Schluss des Teilversuchs folgen einige Fragen und Aufgaben, die die Studierenden mit etwas tiefergehenden Kenntnissen beantworten bzw. lösen können. Diese betreffen drei Bereiche: die Spannungspegel und Betriebsmodi, das Prinzip des Bitstuffings, sowie die fysikalische Fehlertoleranz des Busses.

Durch diesen Teilversuch erhalten die Studierenden als Fähigkeit zumindest eine Auffrischung im Umgang mit dem Oszilloskop und erlernen wie sich Datenrahmen visualisieren lassen.

Außerdem erwerben sie nachfolgende Kenntnisse:

- Generelle Funktionsweise des CANs auf OSI-Layer 1

- Unterschiede zwischen den beiden Betriebsmodi des CANs
- Nach fünf gleichen Bits wird ein Stopfbit eingefügt
- Das CAN ließe sich teilweise auch ohne Terminatoren betreiben
- Die im Standard festgelegten Spannungspegel sind als Mindestanforderungen zu betrachten

5.4 Hintergrund zu Versuch 3

Der dritte Teilversuch widmet sich größtenteils der Inbetriebnahme der restlichen Hardwarekomponenten, insbesondere der Ethernet-CAN-Gateways, und dem Umgang mit der Software `CAN-REport`. Ziel des Versuchs ist, dass die Studierenden genug Know-How sammeln um in den nachfolgenden Teilen die Echtzeiteigenschaften des Feldbusses zu analysieren.

Zunächst wird der bisherige Versuchsaufbau stark erweitert. So befinden sich nicht mehr nur die drei USB-Gateways am Bus, sondern auch die beiden Ethernetmodule. Außerdem wird mittels eines Hubs ein Local Area Network zwischen den Ethernetmodulen und den beiden PCs aufgebaut.

Steht der Versuchsaufbau, so folgt als nächstes die Einbindung einer der beiden Ethernet-CAN-Gateways als Datenquelle für die Software `CAN-REport`. Da dieser Vorgang nicht ganz trivial ist, zeigt die Anleitung ausführlich, wie dies zu geschehen hat.

War auch die Einbindung erfolgreich, beginnt der zweite Teil des Versuchs, bei dem insbesondere die Auseinandersetzung mit dem neuen Programm im Vordergrund steht. Die Studierenden betrachten zunächst die unterschiedlichen Darstellungsoptionen von CAN-Nachrichten, analysieren und vergleichen die beiden zur Verfügung stehenden Darstellungen und bewerten diese.

Am Schluss erlernen die TeilnehmerInnen noch, wie aus `CAN-REport` heraus Datenrahmen verschickt werden können.

Am Ende dieses Versuchs besitzen die Studierenden nachfolgende neue Fähigkeiten:

- Aufbau und Inbetriebnahme der Ethernet-CAN-Gateways
- Umgang mit der Software `CAN-REport`
- Analyse von CAN-Nachrichten
- Analyse des CAN-Datenverkehrs

Außerdem erwerben sie hier folgende Kenntnisse:

- Einzelne CAN-Frames lassen sich mit dem „`Trace View`“ analysieren

- Der „`Object View`“ eignet sich für die Erstellung einer Verkehrsstatistik

Evtl. stellen die Studierenden außerdem bereits in diesem Teilversuch fest, dass das CAN Echtzeiteigenschaften besitzt

5.5 Hintergrund zu Versuch 4

Die Studierenden sind mittlerweile in der Lage ein komplexeres Controller Area Network aufzubauen, zu betreiben und zu analysieren. Von diesem Kenntnisstand macht der vierte Teilversuch nun Gebrauch um künstlich Fehler im Netzwerk zu erzeugen und sich mit diesen auseinanderzusetzen. Er hat damit den Status eines „Hauptversuchs“, bei dem der Erkenntnisgewinn für die VersuchsteilnehmerInnen am höchsten sein dürfte.

Grundsätzlich beschäftigt sich dieser Teilversuch mit drei Bereichen: der Verwendung von unterschiedlichen Bitraten, dem Verhalten des Busses bei hoher Last und der Vergabe von gleichen IDs. Die Anordnung aus dem vorangegangenen Versuch wird beibehalten.

I Unterschiedliche Bitraten

Die Verwendung von unterschiedlichen Bitraten kennen die StudentInnen bereits aus dem ersten Teilversuch. Lag der Schwerpunkt dort allerdings noch auf der Vermeidung eines „Flüchtigkeitsfehlers“ und der damit verbundenen Demotivierung, beschäftigt sich dieser Versuch nun ausführlicher mit der Problematik.

Am Anfang wird der Fehler aus dem ersten Versuch erneut erzeugt um den Einstieg in die Aufgabenstellung zu erleichtern. Danach folgt allerdings eine komplexere Fehleranalyse, die theoretische Kenntnisse der Betriebsmodi voraussetzt. Haben sich die Teilnehmenden in ihrer Versuchsvorbereitung nicht mit dem Thema auseinandergesetzt, müssen sie dies spätestens jetzt nachholen.

II Auslastung des Busses

Eine der wichtigsten Erkenntnisse des gesamten Laborversuchs ist, wie sich das CAN bei hoher Netzwerklast verhält. Um diese Erkenntnis zu erlangen, senden die Studierenden von allen drei USB-Modulen aus im Abstand von 1 ms CAN-Nachrichten mit jeweils 8 Byte Payload. Beträgt die Übertragungsrate auf dem Bus 1 Mbit/s, ist zunächst keine Besonderheit festzustellen. Re-

duzieren die StudentInnen allerdings die Bitrate auf 125 kbit/s, dann treten die Echtzeiteigenschaften des Controller Area Networks deutlich sichtbar zu Tage.

Durch die Auseinandersetzung mit ihren Messergebnissen wird den VersuchsteilnehmerInnen klar, dass das CAN über ein Priorisierungsverfahren für Nachrichten verfügt. Haben sich die Studierenden ausreichend vorbereitet, sind sie in der Lage daraus zu schließen, dass dies eine essenzielle Bedingung für den Betrieb in einer Echtzeitumgebung ist.

III Verwendung der gleichen ID

Die Studierenden beschäftigen sich im dritten Teil dieses Versuchs mit den Problemen die auftreten, wenn mehrere Geräte Nachrichten mit dem gleichen Identifier aussenden. Da die Teilnehmenden nun bereits das Priorisierungsverfahren des CANs kennen und aus der Vorbereitung wissen wie der Buszugriff geregelt wird, gilt es bei dieser Aufgabe das Wissen über die grundsätzliche Funktionsweise des Feldbusses abzurunden.

Senden die Studierenden gleichzeitig von mehreren Knoten aus Nachrichten mit der gleichen ID, treten Fehler auf, die bereits aus der Verwendung unterschiedlicher Bitraten bekannt sind. Um die gestellten Aufgaben zu lösen bzw. die Fragen zu beantworten, muss von den VersuchsteilnehmerInnen nun selbstständig der Zusammenhang zwischen den Messergebnissen und dem Arbitrierungsverfahren hergestellt werden.

Nach Beendigung des gesamten Versuchs verfügen die Studierenden über folgende Kenntnisse:

- Der Betrieb mit verschiedenen Bitraten ist nicht möglich, da aufgrund der unterschiedlichen Bitzeiten das Signal nicht korrekt interpretiert werden kann
- Es existieren drei Betriebsmodi: Normal, Error Passive und Bus Off
- Der Zwischenspeicher des CANs ermöglicht es, dass Nachrichten nach einer Fehlerbeseitigung doch noch übermittelt werden können
- Datenrahmen mit niedrigen IDs werden priorisiert, was dazu führt, dass bei hoher Buslast „weniger wichtige“ Nachrichten zurückgehalten werden
- Das CAN ist, bei richtiger Konfiguration, für den Echtzeitbetrieb geeignet
- Ein Sender kann Nachrichten mit mehreren IDs verschicken
- Nachrichten mit einer bestimmten ID dürfen nur von einem einzigen Sender verschickt werden

5.6 Hintergrund zu Versuch 5

Ähnlich wie in Versuch 3 dient dieser fünfte Versuch in erster Linie dem Erwerb von weiterem Know-How. Ziel ist, dass die Studierenden am Ende in der Lage sind zwei Controller Area Networks mittels einer Ethernetbrücke zu verbinden und den Datenverkehr im LAN zu analysieren.

Um dies zu ermöglichen, wird das bestehende CAN zunächst in zwei Segmente geteilt, die zwar über die gleiche Konfiguration verfügen, jedoch fysikalisch unabhängig sind. Die Ethernet-CAN-Gateways werden nun so konfiguriert, dass sie über das LAN eine Verbindung aufbauen und CAN-Daten von dem einen Segment ins andere tunneln können.

Sind die Geräte aufgebaut und korrekt eingestellt, setzen sich die Studierenden im ersten Schritt mit den Ethernetpaketen auseinander. Sie erkennen, dass einerseits die getunnelten Daten im LAN übertragen werden und andererseits ein Paketaustausch zur Aufrechterhaltung der Verbindung stattfindet. Eine Erkenntnis ist, dass die getunnelten Pakete, die mit der Software `Wireshark` betrachtet werden, neben den Nutzdaten noch weitere Parameter des CAN-Rahmens enthalten.

Die nächste Aufgabe sieht vor, dass die Studierenden die beiden CAN-Segmente mit unterschiedlichen Übertragungsraten betreiben. Da die Netze fysikalisch getrennt sind und eine getunnelte Nachricht im anderen Segment zu den dort herrschenden Bedingungen übertragen wird, treten hier keine Fehler auf. Erzeugen die VersuchsteilnehmerInnen jedoch eine hohe Last auf dem CAN, werden Nachrichten mit niedriger ID ggf. zurückgehalten. Diese Eigenschaft des Feldbusses, stellen die StudentInnen fest, bleibt auch in diesem Fall erhalten.

Optional sieht die Aufgabenstellung vor, dass der Teilversuch mit dem UDP (statt TCP) durchgeführt werden kann. Sollte dies technisch realisierbar sein, weil die Ethernet-CAN-Gateways stabil funktionieren, könnten die Studierenden weitere Kenntnisse erwerben. So ist bei dieser Konfiguration festzustellen, dass das an sich verbindungslose UDP optional dennoch eine „Verbindung“ aufbaut, um eine Kommunikation zwischen den beiden Ethernetmodulen am Leben zu erhalten.

Nach Abschluss des vorletzten Versuchs verfügen die Studierenden über nachfolgende Fähigkeiten:

- Einrichtung und Betrieb einer Ethernet-CAN-Brücke
- Einrichtung einer TCP-Verbindung
- Ggf. Einrichtung einer UDP-Verbindung
- Umgang mit der Software `Wireshark`

Außerdem erwerben sie in diesem Teilversuch folgende Kenntnisse:

- Die Ethernet-CAN-Gateways bauen eine Verbindung auf und halten diese aufrecht
- CAN-Daten werden über das Ethernet getunnelt
- Die Übertragungsraten in unterschiedlichen CAN-Segmenten müssen nicht gleich sein um Nachrichten tunneln zu können

5.7 Hintergrund zu Versuch 6

Die Studierenden sind mittlerweile in der Lage ein komplexes CAN aufzubauen und zu betreiben. Außerdem haben sie gelernt, wie sich zwei fysikalisch getrennte Controller Area Networks durch eine Ethernetbrücke zu einem logischen Netz verbinden lassen. Ziel des letzten Versuchs ist es nun, das Verhalten des Ethernets an seiner Kapazitätsgrenze kennen zu lernen und Rückschlüsse auf den Betrieb unter Echtzeitbedingungen zu ziehen. Gemeinsam mit dem vierten Versuch liefert diese Aufgabenstellung die meisten Erkenntnisse im Bereich der Echtzeitsysteme. Dieser Teil ist daher ebenfalls als ein „Hauptversuch“ zu betrachten.

Die Versuchsanordnung bleibt unverändert, es werden also zwei separat betriebene CANs durch das Ethernet logisch verbunden. Dafür werden zwei neue Programme eingeführt: `PackETH` für die Generierung von Datenpaketen und `Traffic Graph` für die Visualisierung der Auslastung im LAN.

Während `Traffic Graph` eine selbsterklärende Analysesoftware darstellt, wird den VersuchsteilnehmerInnen für die Konfiguration von `PackETH` eine genaue Anleitung mitgegeben. Dies ist notwendig, da die Bedienung des Programms nicht trivial ist und teilweise bestimmte Schreibweisen eingehalten werden müssen um die Paketerzeugung überhaupt starten zu können. Da, wie in den anderen Teilversuchen auch, der Erkenntnisgewinn im Vordergrund steht, soll dadurch vermieden werden, dass die StudentInnen unnötigen Aufwand betreiben um den Umgang mit der Software zu erlernen.

Haben die VersuchsteilnehmerInnen die Software und alle Geräte fertig konfiguriert, beginnen die Messungen. Während der gesamten Zeit werden von einem USB-CAN-Gateway aus über die Ethernetbrücke im Abstand von 1 ms Nachrichten mit einer 8 Byte großen Payload verschickt. Nachdem die erste Auslastungsmessung erfolgt ist und somit ein Vergleichswert existiert, starten die Studierenden den Paketgenerator. Nach und nach wird die Verzögerung zwischen den erzeugten Paketen verringert und damit die Last auf dem Ethernet erhöht. Ab einer Auslastung von ca. 60 % werden die Studierenden feststellen, dass das LAN nicht mehr regelmäßig CAN-Nachrichten zwischen den beiden Segmenten überträgt. Wird die

Last weiter erhöht, bricht die Verbindung schließlich komplett ab.

In jedem Fall werden die Messergebnisse zeigen, dass die theoretische Maximalauslastung des LANs von 10 Mbit/s noch nicht erreicht worden ist und trotzdem nicht alle Pakete gesendet werden können. Auch wenn viele der Studierenden dieses Fänomen wahrscheinlich zum ersten Mal bewusst beobachten, sollte die Funktionsweise des Ethernets bereits bekannt sein. Aus diesem Grund wird erwartet, dass die StudentInnen ihre Beobachtungen erklären können.

In den abschließenden Fragen und Aufgaben beschäftigen sich die TeilnehmerInnen noch einmal ausführlich mit dem Ethernet unter dem Aspekt der Echtzeitkommunikation. Außerdem werden Ethernet- und CAN-Rahmen miteinander verglichen, so dass spätestens hier deutlich wird, dass das Ethernet über keine typischen Eigenschaften einer echtzeitfähigen Netzwerktechnologie verfügt.

Nach Beendigung des letzten Versuchs verfügen die Studierenden über folgende neue Fähigkeiten:

- Umgang mit der Software `PackETH`
- Umgang mit der Software `Traffic Graph`

Außerdem besitzen sie nun folgende Kenntnisse:

- Ab einer Auslastung von ca. 60 % kommt es beim Ethernet erstmals zu nicht vorhersehbaren „Rückstellungen“ von Paketen
- Die Maximalauslastung von 10 Mbit/s wird im Ethernet in der Praxis nicht erreicht
- Das Ethernet ist grundsätzlich keine echtzeitfähige Netzwerktechnologie
- Ethernet- und CAN-Rahmen unterscheiden sich in vielen wichtigen Punkten

6 Lösungen

6.1 Allgemeines

Die nachfolgenden Abschnitte bieten eine „Musterlösung“ für alle Versuche, sowie für ggf. gestellte Fragen und Aufgaben. Alle Messungen wurden mehrfach durchgeführt um die Richtigkeit der Ergebnisse zu gewährleisten. Dennoch kann eine Veränderung in der Konfiguration der Geräte oder im Versuchsaufbau immer zu Abweichungen führen.

Diese Lösungen sind als Vergleich für das Laborpersonal gedacht. Sie sollen den Studierenden nicht ausgehändigt werden um ein eigenständiges Arbeiten zu garantieren.

6.2 Lösungen von Versuch 1

Der erste Teilversuch zielt darauf ab die Studierenden zunächst mit der Software `PCANview` und dem Senden bzw. Empfangen von CAN-Nachrichten vertraut zu machen. Hierfür werden vorerst zwei der USB-CAN-Gateways in Betrieb genommen und *eine* Nachricht von Modul *usb0* an *usb1* gesendet. Bei korrekter Konfiguration ist festzustellen: Die Nachricht erscheint bei *usb0* im Fenster „`Transmit`“, während die selbe Nachricht bei *usb1* im Fenster „Receive“ gelistet wird.

Nimmt man das dritte Gateway (*usb2*) in Betrieb, empfängt dieses ebenfalls die von *usb0* gesendeten CAN-Frames. Wird die Periode verändert und die Nachricht von *usb0* folglich im definierten Abstand immer wieder gesendet, empfangen *usb1* und *usb2* diese auch regelmäßig. Die Spalten „Count“ und „Period“ zeigen hierbei die Anzahl der Nachrichten respektive den zeitlichen Abstand zwischen diesen in Millisekunden an.

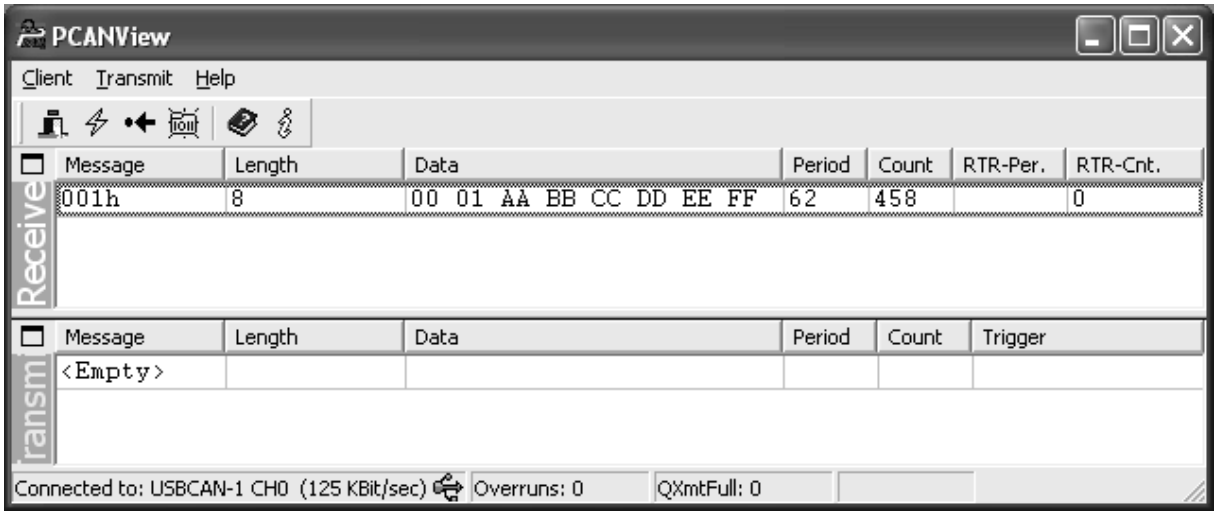

Abb. 30: PCANview-Fenster, das zeigt, dass 458 Nachrichten empfangen wurden und seit der letzten Nachricht 62 ms vergangen sind

Verändert man den Betriebsmodus der USB-Module so, dass sie alle mit unterschiedlichen Bitraten betrieben werden, ist ein Datenaustausch nicht möglich. In diesem Fall werden die Nachrichten zwar versendet, jedoch nicht empfangen. Gleichzeitig melden alle drei Module den Fehler „`Bus Heavy`“ bzw. „`Bus Off`“. Dies ist darauf zurückzuführen, dass, aufgrund der unterschiedlichen Bitraten, die Nachrichten der jeweils anderen Gateways nicht richtig interpretiert werden können (falsche Bitdauer). Mehr zu diesem Thema findet sich im vierten Versuch.

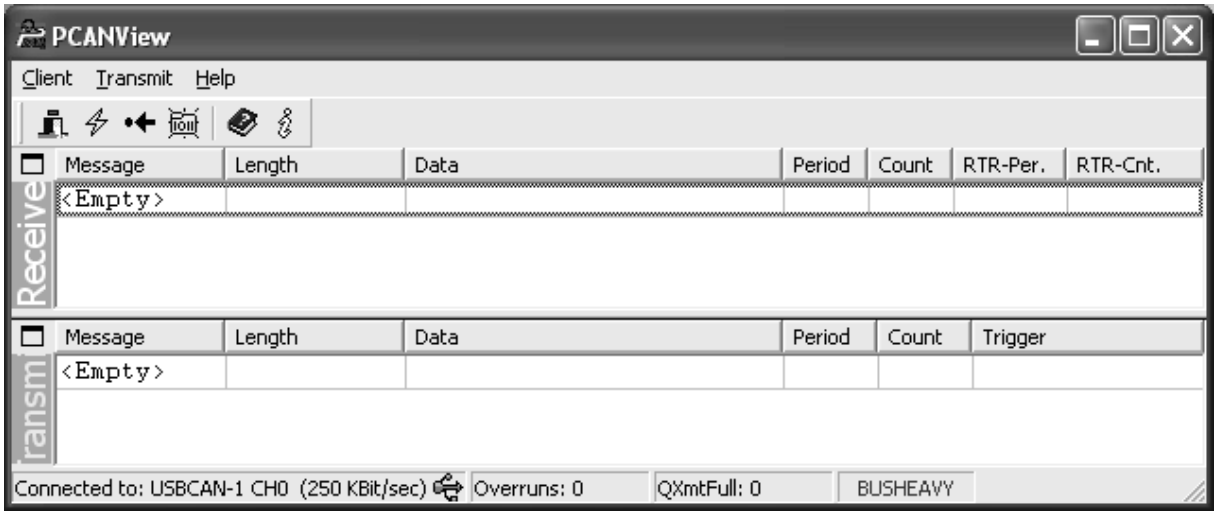

Abb. 31: PCANview-Fenster des Empfängers mit der Fehlermeldung „Bus Heavy“

Setzt man in `PCANview` bei der Nachrichtenerstellung den Haken bei „`Extended Frame`“, so wird das erweiterte Rahmenformat aktiviert. Es ist nun möglich IDs mit bis zu 29 bit Länge zu vergeben.

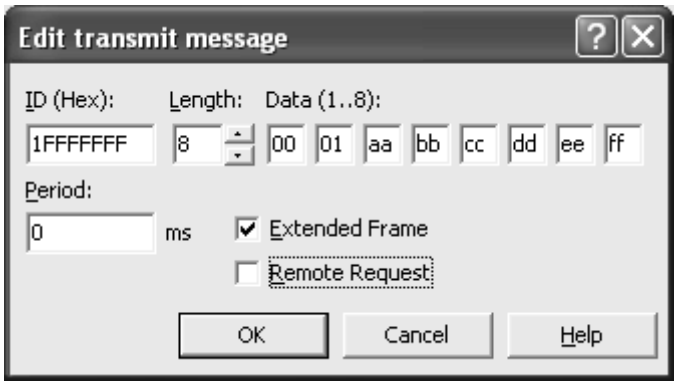

Abb. 32: Nachrichtenerstellung in PCANview mit der höchstmöglichen ID

Sendet man statt eines normal Datenrahmens einen „`Remote Frame`“, enthält der Frame keine Payload. Stattdessen wird das RTR-Bit aktiviert um zu signalisieren, dass eine Nachrichtenanforderung folgt. Die Software `PCANview` visualisiert `Remote Frames` anders als `Data Frames`: im Datenteil wird ein `Remote Request` angezeigt, und an Stelle der normalen Periode und des normalen Zählers werden separate Spalten für „`RTR-`Period“und „`RTR-`Count“ genutzt.

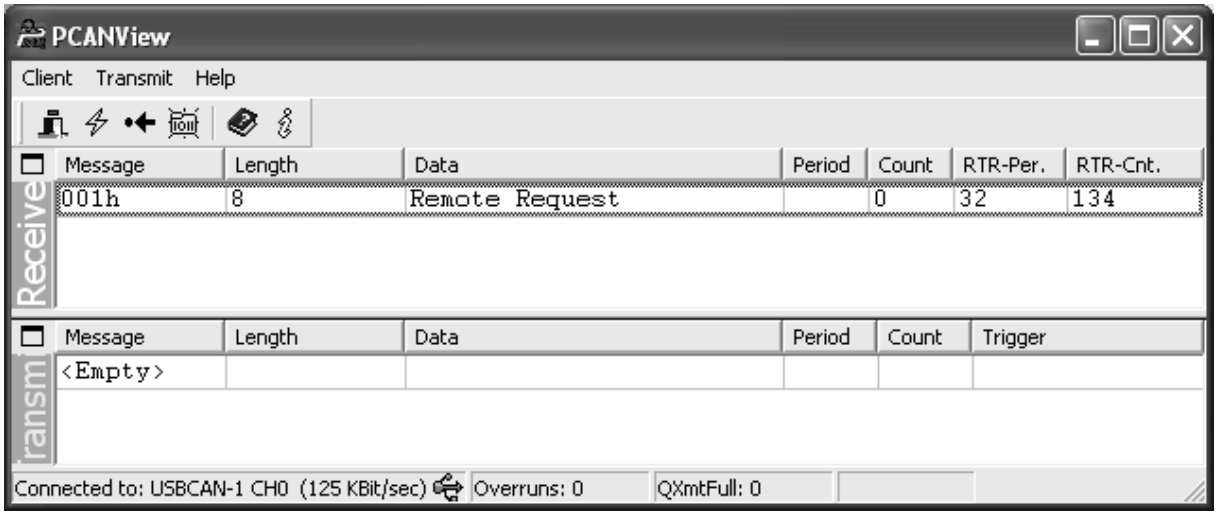

Abb. 33: PCANview-Fenster beim Empfang von Remote Frames

6.3 Lösungen von Versuch 2

Im zweiten Teilversuch widmen sich die Studierenden der Bitübertragungsschicht um durch eine direkte Visualisierung der übertragenen Daten die Funktionsweise des CANs besser zu verstehen. Hierfür wird der bisherige Versuchsaufbau um ein digitales Oszilloskop ergänzt.

Werden in `PCANview` alle Einstellungen so vorgenommen wie empfohlen, also im Abstand von 50 ms bei einer Bitrate von 125 kbit/s 2 Byte lange Nachrichten verschickt, erhält man über die Auto-Set-Funktion des Oszilloskops zunächst ein Bild, das weitestgehend der folgenden Skizze entspricht:

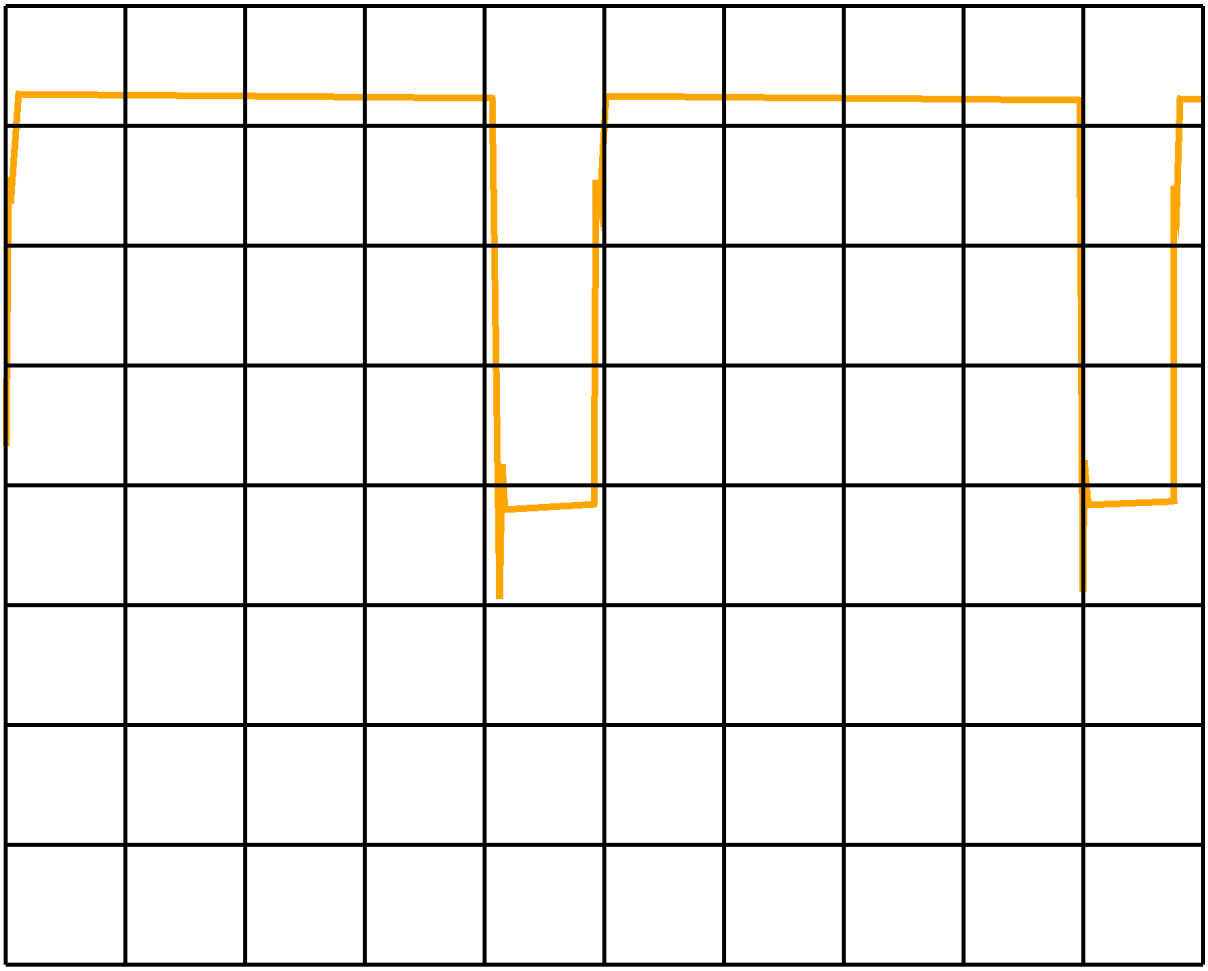

Abb. 34: Skizze der zu erwartenden Darstellung am Oszilloskop (Div = 1 V, TB = 10 µs)

Deutlich zu erkennen sind die Rechteckimpulse des CANs auf dem Bus. Variiert man nun die Länge des Datenrahmens, sowie einzelne Bytes in diesem, lässt sich feststellen, dass dies unmittelbar Auswirkungen auf das dargestellte Signal hat.

Generell lässt sich folgendes festhalten: Erhöht man die Bitrate, ist es einerseits schwieriger das Signal zu detektieren. Andererseits ist zu erkennen, dass die Signalqualität abnimmt und die Rechteckimpulse unsauberer werden. Dies ist auf die erhöhte Signalfrequenz zurückzuführen.

Nachfolgend finden sich die weiteren Antworten und Lösungen zum zweiten Versuch:

- *Spannungsdifferenz*: Die Spannungsdifferenz (ΔV) der beiden Zustände beträgt in der Regel ca. 3 V. Laut Standard wird eine Differenz von mindestens 2 V empfohlen.
- *Betriebsmodus*: Der CAN-Bus wird in allen Teilversuchen im High-Speed-Modus betrieben, da die USB-CAN-Gateways nur in diesem arbeiten können. Zu erkennen ist dies u. a. an der Spannungsdifferenz. Im Low-Speed-Modus, der ebenfalls existiert, beträgt diese gemäß Standard 7,2 V.
- *Pegel*: Im rezessiven Zustand bzw. im Ruhezustand beträgt der gemessene Signalpegel 3

V. Dominante Bits verfügen über einen Pegel von 0 V.

- *Messung des Pegels*: Der laut ISO-Standard definierte Signalpegel von 2,5 V im Ruhezustand bzw. rezessiven Zustand wird in diesem Teilversuch nicht gemessen, da er dem Spannungswert der Ader gegen Masse entspricht. Um die Darstellung der Rechteckimpulse zu erhalten, werden in diesem Versuch jedoch die Spannungspegel der beiden Adern verglichen, wodurch lediglich die Spannungsdifferenz angezeigt wird.
- *Bitstuffing*: Es lässt sich relativ schnell nachweisen, dass das CAN vom Prinzip des Bitstuffings Gebrauch macht, wenn man einen Pegel für längere Zeit anlegt. Am einfachsten ist dies möglich, indem die Payload komplett dominant bzw. rezessiv gesetzt wird (also `00 00 00 00 00 00 00 00` bzw. `FF FF FF FF FF FF FF FF`). Auf dem Oszilloskop ist dann sofort zu erkennen, dass jedes sechste Bit im Nutzdatenteil den jeweils inversen Zustand annimmt.
- *Pegel der Adern*: Misst man die Adern 3 bzw. 4 jeweils gegen Masse (Ader 1 oder 2), so fällt auf, dass die Rechteckimpulse entweder einen positiven oder einen negativen Spannungspegel aufweisen. Ader 3 entspricht demnach CAN_L, während Ader 4 den Pegel von CAN_H führt.
- *Betrieb ohne Terminatoren*: Je nach Versuchsaufbau unterscheiden sich hier die Messergebnisse. In den meisten Fällen ist der Betrieb mit einem Endwiderstand noch ohne Weiteres möglich. Fehlen beide Widerstände, bricht die Datenübertragung jedoch ab.
- *Kurzschluss*: Werden die Adern 3 und 4 kurzgeschlossen, so ist keine Nachrichtenübertragung mehr möglich. Die Netzknoten melden hier jeweils den Fehler „`Bus Off`“.

6.4 Lösungen von Versuch 3

Im dritten Versuch werden die restlichen Komponenten, insbesondere die Ethernet-CAN-Gateways, sowie die Software `CAN-REport` in Betrieb genommen. Um dies zu ermöglichen, ist ein etwas komplexerer Versuchsaufbau und die einmalige Konfiguration der Software notwendig.

Wird mittels der drei USB-Module Datenverkehr auf dem Bus erzeugt, so stellt die Software `CAN-REport` diesen zunächst im „`Trace View`“ dar. Behält man die Standardeinstellungen bei, zeigt das Programm zu jedem CAN-Frame den Zeitpunkt der Detektierung, die Nachrichten-ID, die Rahmenart sowie die Payload an.

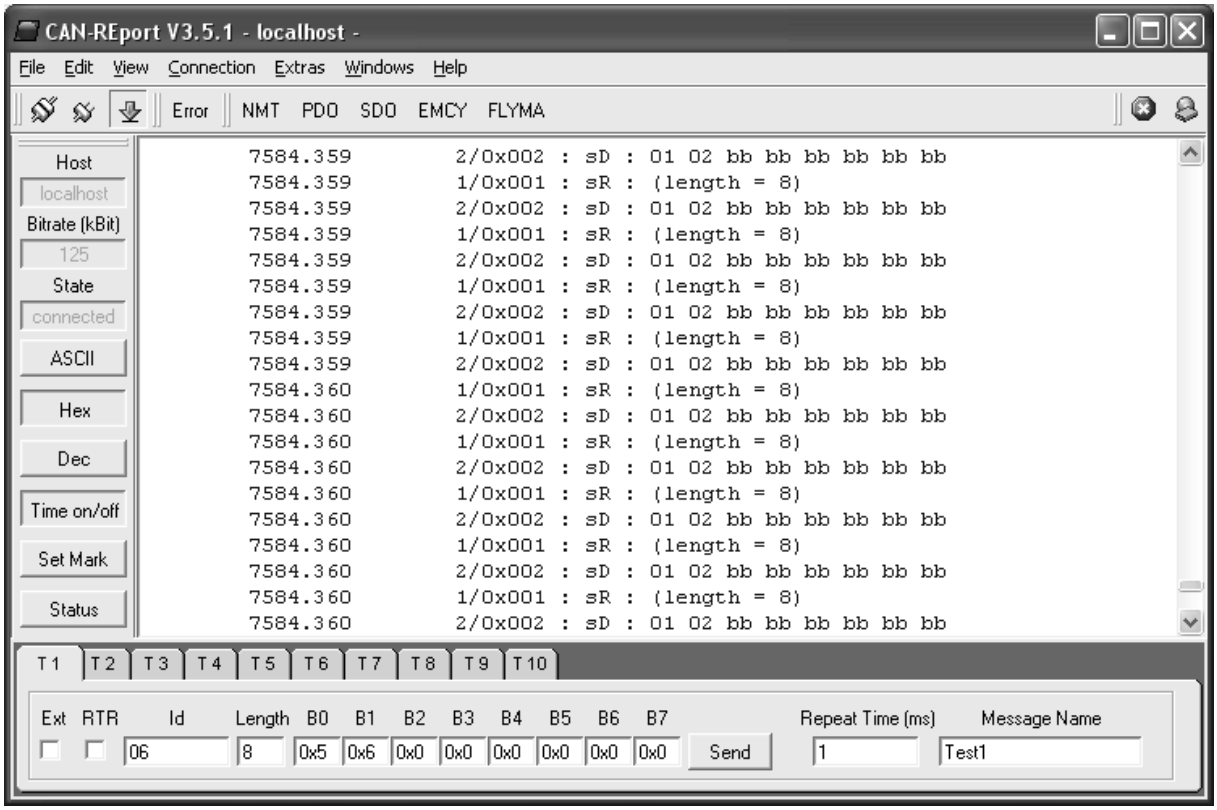

Abb. 35: Fenster des Trace Views in CAN-REport beim Empfang von einfachen Data und Remote Frames

Folgende Einstellungen lassen sich in diesem Modus an der Anzeige vornehmen:

- Visualisierung der Payload in Hexadezimal- und Dezimalwerten, sowie als ASCII-Code
- Aktivierung bzw. Deaktivierung der Zeitanzeige

Der `Trace View` eignet sich insbesondere für die Analyse einzelner Datenrahmen, da hier jede CAN-Nachricht einen eigenen Eintrag erhält. Bei einer hohen Sendefrequenz wird die Liste jedoch schnell unübersichtlich, weshalb sich dieser Modus nur schlecht für statistische Auswertungen eignet.

Im „`Object View`“ liegt der Fokus der Darstellung nicht auf den einzelnen Datenrahmen, sondern auf der Anzahl der empfangenen Pakete. Angezeigt werden hier, ähnlich wie in `PCAN-view`, CAN-ID, Rahmentyp, Nutzlast, Periode und die Anzahl der empfangenen Nachrichten.

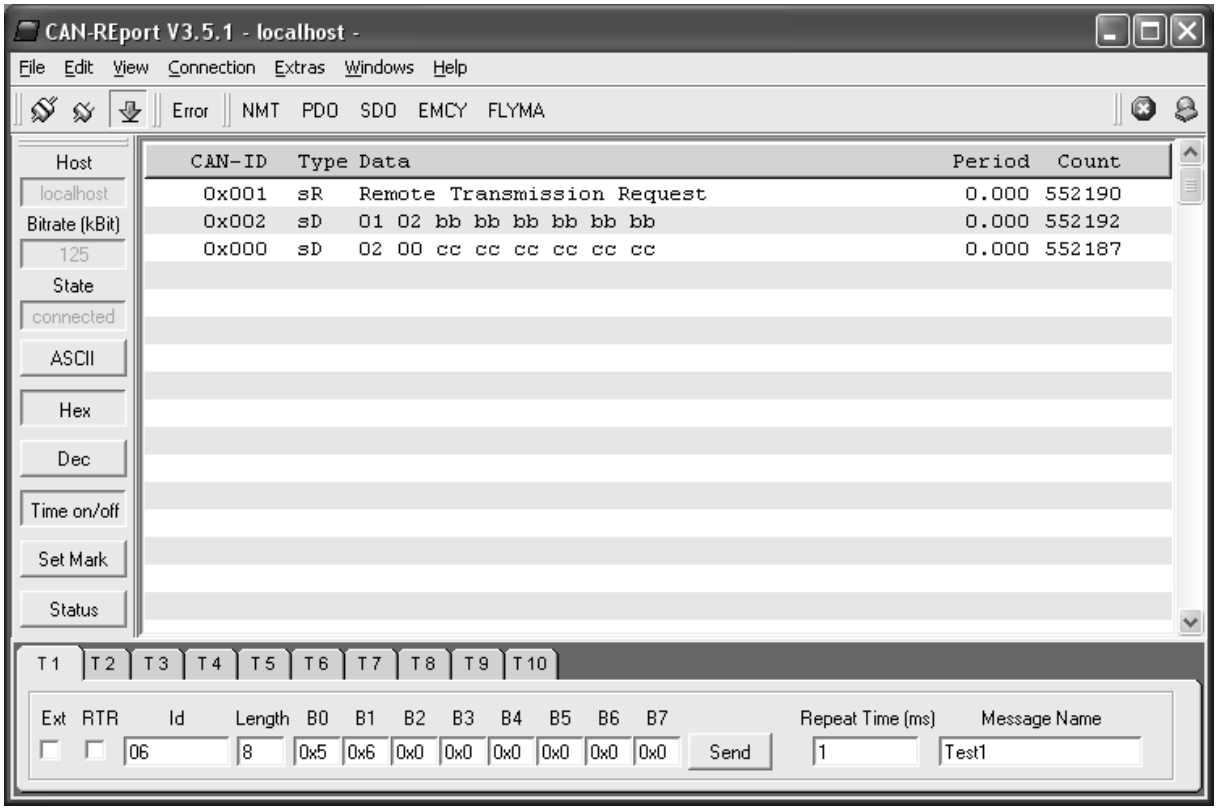

Abb. 36: Fenster des Object Views in CAN-REport beim Empfang von einfachen Data und Remote Frames

Auch in diesem Modus lässt sich die Visualisierung der Nutzlast ändern. Möglich sind Hexadezimal-, Dezimal- und ASCII-Darstellung.

Für die statistische Erfassung des Datenverkehrs ist der `Object View` gut geeignet. Insbesondere der Zähler, der Nachrichten nach der jeweiligen ID unterscheidet, liefert relativ genaue Ergebnisse. Will man die Payload von Nachrichten analysieren, ist dieser Modus hingegen ungeeignet.

Werden auch aus `CAN-REport` heraus Nachrichten versandt, fällt im `Trace View` auf, dass die Frames des Ethernet-CAN-Gateways in rot dargestellt werden. Ansonsten sind die Funktionen weitestgehend analog zu denen von `PCANview`. Der einzige Unterschied ist, dass Änderungen an der Nachricht erst aktiv werden, wenn der Sendevorgang beendet und neu gestartet wird.

Sofern die Übertragungsrate auf 1 Mbit/s gesetzt worden ist und alle vier Gateways Nachrichten mit einer Nutzlast von 8 Byte versenden, könnte man im `Object View` bereits in diesem Teilversuch bemerken, dass mindestens ein Zähler langsamer steigt als die anderen drei. Warum dies so ist, wird im Nachfolgenden Kapitel erläutert.

6.5 Lösungen von Versuch 4

Dieser Teilversuch widmet sich erstmals ausschließlich den Fehlern, die in einem CAN auftreten können. Der Fokus liegt dabei auf drei Bereichen: der Verwendung unterschiedlicher Bitraten, einer (zu) hohen Busauslastung, sowie der Verwendung von gleichen Nachrichtenidentifiern.

I Unterschiedliche Bitraten

Zu Beginn behandelt die Aufgabenstellung die Verwendung von unterschiedlichen Bitraten. Hierfür wird, analog zum ersten Teilversuch, *ein* Knoten mit einer anderen Bitrate betrieben (hier 800 kbit/s) als der Rest (hier 1 Mbit/s) und *eine* CAN-Nachricht verschickt (von dem Knoten mit der niedrigeren Bitrate). Im `PCANview` wird die Nachricht zunächst als gesendet gelistet, jedoch empfangen die anderen Knoten diese nicht. Dafür treten die ersten Fehlermeldungen auf: alle anderen Gateways schalten sofort in den Bus-Off-Mode und übertragen von sich aus keine Nachrichten mehr. Wird der Knoten mit der geringeren Bitrate vom Bus getrennt, ist die gesendete Nachricht „verloren“, erreicht also nie ihren Bestimmungsort.

Führt man dieses etwas abgewandelt durch und sendet eine Nachricht von einem anderen Knoten aus, tritt zunächst ein anderer Fehler auf: alle empfangenden Gateways auf der „richtigen“ Geschwindigkeit schalten lediglich in den Error-Passive-Mode. Erst wenn man, ungeachtet der ersten Fehlermeldung, weiterhin CAN-Nachrichten sendet, so werden die Module in den Bus-Off-Mode versetzt und übertragen selber keine Nachrichten mehr.

Trennt man nun jedoch das Modul mit der geringeren Bitrate von Bus, lässt sich feststellen, dass die zu sendenden Datenrahmen schließlich doch übertragen werden und ihr Ziel erreichen.

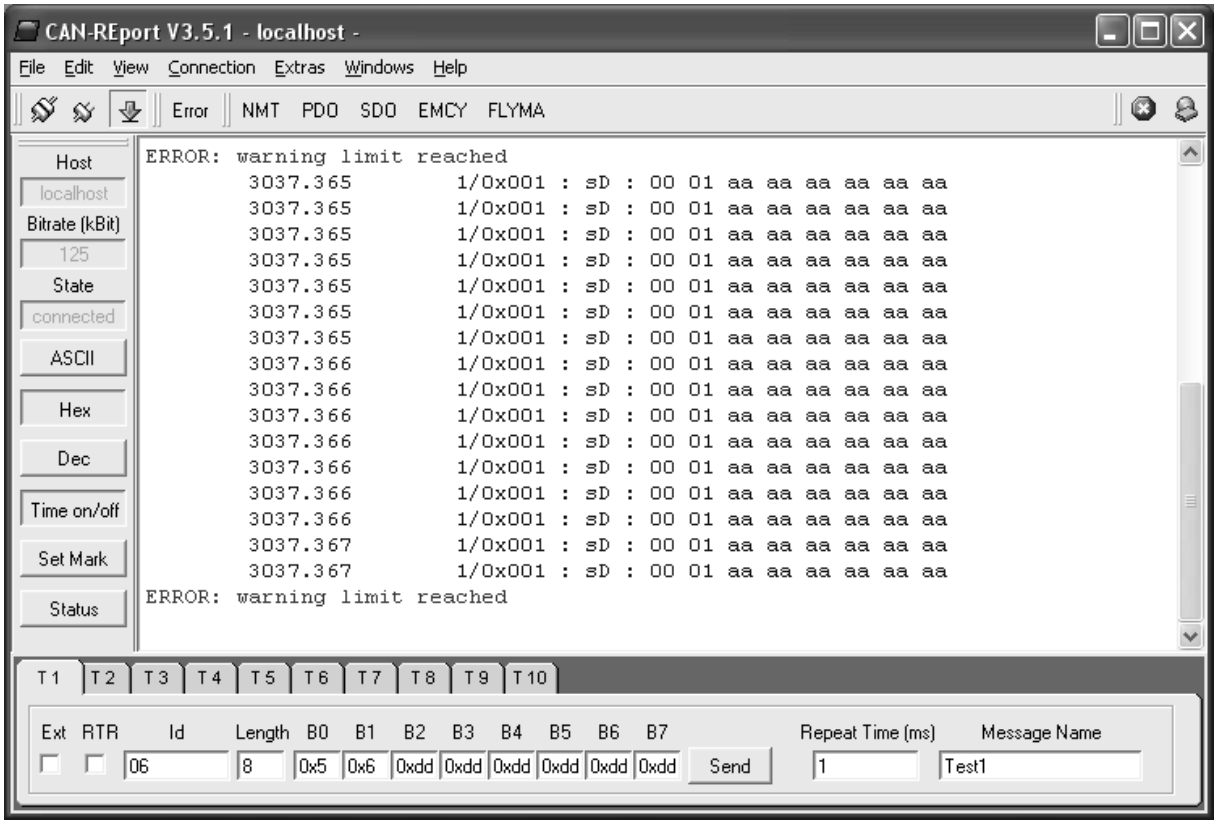

Abb. 37: Gespeicherte CAN-Nachrichten werden übertragen, nachdem ein fehlerhaftes Modul entfernt worden ist

Zusammenfassend lässt sich festhalten:

- Der Betrieb mit unterschiedlichen Bitraten ist im CAN nicht möglich
- Es existieren, neben dem normalen, zwei weitere Betriebsmodi: der Error-Passive-Mode, der einen eingeschränkten Betrieb ermöglicht, und der Bus-Off-Mode, der einen Knoten vom Netzwerk trennt
- Die CAN-Gateways verfügen über einen Zwischenspeicher und sind in der Lage ihre Nachrichten zu übertragen sobald der Bus fehlerfrei ist

II Auslastung des Busses

Im zweiten Teil des Versuchs wird das Verhalten das CANs bei hoher Last analysiert. Hierfür wird im ersten Schritt auf allen drei USB-Gateways eine 8 Byte lange Nachricht erstellt, die jeweils im Abstand von 1 ms gesendet wird. Der Bus wird mit einer Geschwindigkeit von 1 Mbit/s betrieben. Wertet man den Datenverkehr mit dem `Object View` der Software `CAN-REport` aus, so lässt sich feststellen, dass die Werte aller drei Zähler sehr dicht beieinander liegen. Daraus lässt sich folgern, dass der Bus über ausreichende Kapazitäten verfügt um alle Nachrichten zu übertragen.

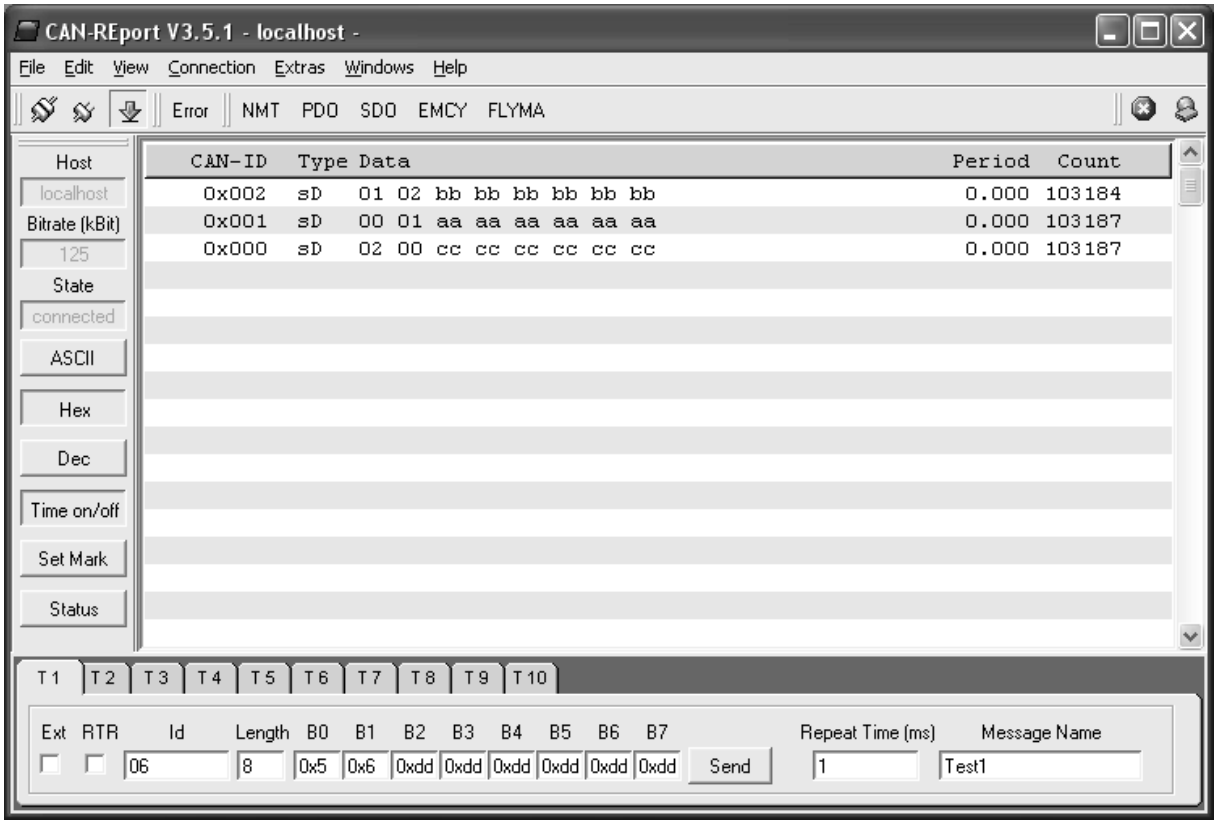

Abb. 38: Datenverkehrsstatistik des CANs bei 1 Mbit/s

Wird das Experiment mit einer Busgeschwindigkeit von lediglich 125 kbit/s, bei ansonsten gleicher Konfiguration, wiederholt, so lässt sich feststellen, dass die Werte der Zähler große Unterschiede aufweisen. Die Nachrichten mit der niedrigsten CAN-ID werden ca. doppelt so häufig übertragen wie Nachrichten mit der nächst höheren. Andere Datenrahmen finden hingegen praktisch nie ihren Weg auf den Bus.

Daraus lässt sich schließen, dass eine Priorisierung der Frames nach ihren Identifikationsnummern erfolgt. Je niedriger die ID eines Datenrahmens ist, desto höher ist die Wahrscheinlichkeit, dass er übertragen wird. Diese Eigenschaft des CAN-Busses ist essenziell für den Einsatz in Echtzeitumgebungen, denn nur so ist es möglich die Reihenfolge der übertragenen Nachrichten zu beeinflussen und eine Schedule zu erstellen.

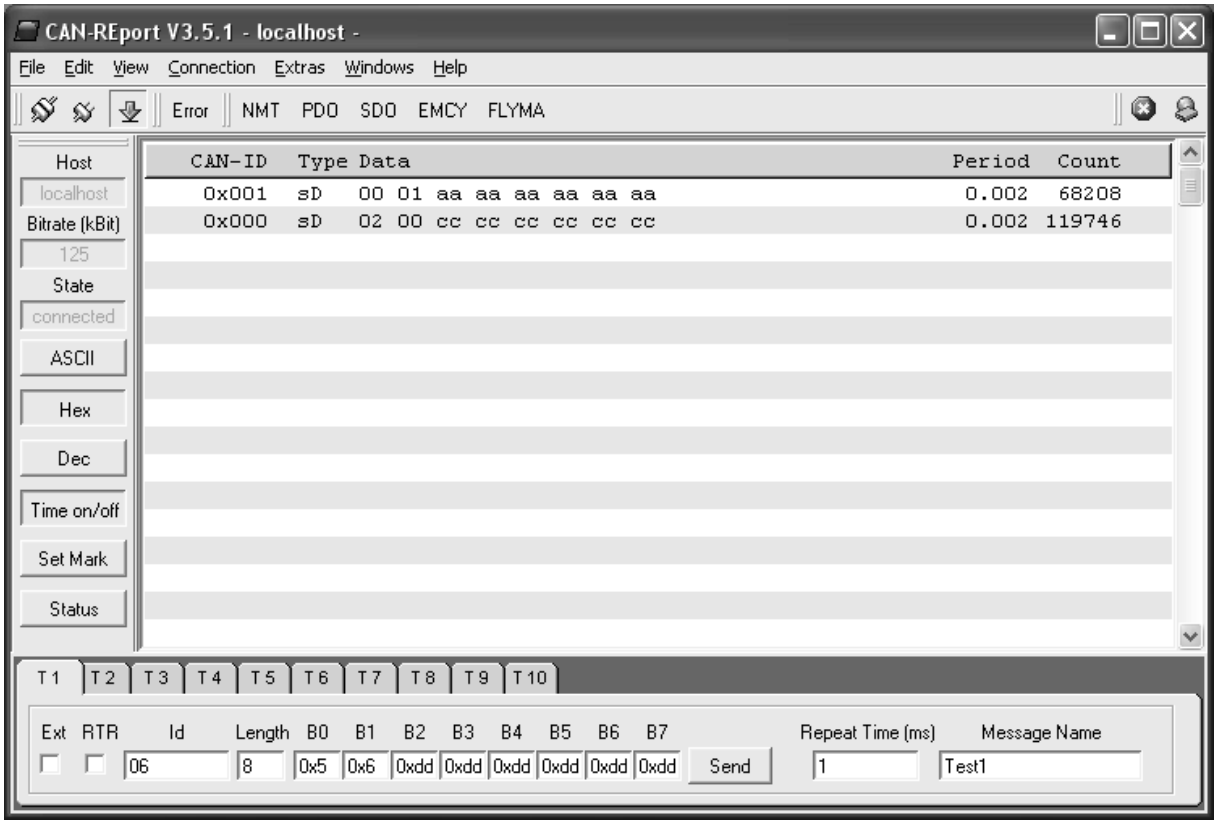

Abb. 39: Datenverkehrsstatistik des CANs bei 125 kbit/s

III Verwendung der gleichen ID

Ändert man schließlich in den CAN-Nachrichten die Identifier so ab, dass alle USB-Module an den gleichen senden, treten bei der Übertragung Fehler auf. Alle Knoten werden in kürzester Zeit in den Bus-Off-Mode versetzt, da aufgrund der einheitlichen IDs das Buszugriffsverfahren ausgehebelt wird. Die einzelnen Knoten erkennen nun nicht mehr, dass gleichzeitig eine andere Nachricht übertragen wird und detektieren demzufolge erst *nach* der Arbitrierung ein Signal auf dem Bus, das nicht dem von ihnen gesendeten entspricht. Dies führt zur Erhöhung des Fehlerzählers und schließlich zur Deaktivierung des Knotens.

Wird die Bitrate wieder auf 1 Mbit/s erhöht, lässt sich das selbe Fänomen feststellen. Die Wahrscheinlichkeit, dass der Bus zu Sendebeginn leer ist, ist allerdings höher, weshalb zunächst mehr Nachrichten übertragen werden und der Fehlerzähler der Module langsamer ansteigt. Letztlich bleibt das Netzwerk aber instabil.

Nachfolgend finden sich die weiteren Antworten und Lösungen zum vierten Versuch:

- *Detektion von Signalen*: Ein Netzknoten kann ein Signal nur erkennen, wenn es mit der korrekten Bitrate übertragen wird. Die Bitzeiten sorgen sonst für eine Fehlinterpretation des Datenstroms.

- *Mehrere IDs und Sender*: Jeder CAN-Knoten kann ohne Weiteres Nachrichten mit unterschiedlichen IDs verschicken. Allerdings ist zu beachten, dass Nachrichten mit der gleichen ID immer vom selben Sender stammen müssen. Es ist demnach nicht sinnvoll, dass mehrere Sender Nachrichten mit der gleichen ID verschicken.

6.6 Lösungen von Versuch 5

Der vorletzte Teilversuch dient in erster Linie der Inbetriebnahme der Ethernetbrücke. Hierfür werden zunächst zwei getrennte Controller Area Networks aufgebaut. Das erste besteht aus einem USB-Modul und einem Ethernet-CAN-Gateway. Das zweite vereint das andere Ethernet-CAN-Gateway und die beiden anderen USB-Module. Alle Geräte sind jedoch so konfiguriert, dass sie auch in einem Netzwerk betrieben werden könnten. Die Übertragungsrate beträgt in beiden CANs zunächst jeweils 125 kbit/s.

Um eine Kommunikation via Ethernet zu ermöglichen, sind beide Ethernet-CAN-Gateways an einen Hub angeschlossen und bilden, gemeinsam mit den Labor-PCs, ein LAN. Auf jedem Gateway läuft jeweils ein TCP-Server und ein TCP-Client, damit CAN-Nachrichten über das LAN getunnelt werden können.

Bei der ersten Messung wird *ein* CAN-Datenrahmen übertragen, der Datenverkehr im LAN mit der Software `Wireshark` mitgeschnitten und analysiert. Dabei fällt zunächst auf, dass zwischen den beiden Gateways zwei unterschiedliche Arten von Paketen ausgetauscht werden. Einerseits sind dies die über das LAN transportierten CAN-Nachrichten (bzw. die dazugehörige Empfangsbestätigungen), sowie ein zyklischer Austausch von Keep-Alive-Paketen um die TCP-Verbindung aufrechtzuerhalten.

Betrachtet man das Paket mit der getunnelten CAN-Nachricht, so ist zu erkennen, dass dieses einen Datenteil beinhaltet. Dabei entsprechen die letzten Bytes der Payload des CAN-Frames. So enthält der nachfolgend dargestellte Datenteil eine 8 Byte große Nutzlast (hier `00 02 AA AA AA AA AA AA`).

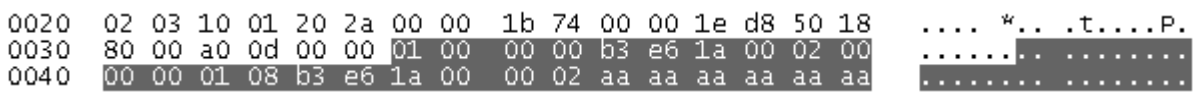

Abb. 40: Der CAN-Datenteil mit einer 8 Byte großen Payload in einem Ethernetframe

Zumindest zwei weitere Parameter sind zu erkennen, wenn die Länge der Payload im CAN und die Nachrichten-ID variiert werden: das achte und neunte Byte des Datenteils beinhalten die ID (hier `00 02`), das 14. Byte hingegen den Wert der Nutzlastlänge (hier `08`).

Senden mehrere Knoten CAN-Nachrichten, so werden alle über das LAN übertragen. Ist

der Bus ausgelastet, werden Nachrichten mit niedrigen IDs nicht oder nur sehr selten übermittelt. Ist der Speicher der Ethernetmodule voll, melden diese einen Fehler und verwerfen weitere Nachrichten.

In jedem Fall ist festzustellen, dass der Datenteil immer so gesendet wird, dass er von dem CAN *weg*transportiert wird, in welchem sich der Quellknoten befindet.

Ändert man nun die Bitrate in *einem* der beiden CAN-Segmente (hier auf 1 Mbit/s) und versendet Datenrahmen, so lässt sich feststellen, dass diese trotzdem in beiden Segmenten empfangen werden können. Dies ist darauf zurückzuführen, dass durch die LAN-Tunnelung lediglich die Nachricht an sich übertragen wird, der empfangende CAN-Knoten diese jedoch erst wieder in einen CAN-Rahmen schreibt und dann auf die lokalen Einstellungen zurückgreift.

Verwendet man statt TCP das UDP, so sind die Erkenntnisse ähnlich. Zwar ist das UDP eigentlich ein verbindungsloses Protokoll, jedoch wird durch die Software der Ethernetmodule trotzdem eine „Verbindung“ aufgebaut und durch in bestimmten Zyklen verschickte Pakete aufrechterhalten. Bricht die „Verbindung“ ab, wird diese automatisch wieder aufgebaut, sofern diese Option aktiviert ist.

Der CAN-Datenteil der UDP-Pakete ist mit denen der TCP-Pakete identisch.

6.7 Lösungen von Versuch 6

Im letzten Teilversuch wird die Versuchsanordnung beibehalten, lediglich evtl. gestartete UDP-Server gilt es wieder zu deaktivieren. Auf *PC 2* wird in diesem Versuch mit `PackETH` allerdings eine neue Software verwendet um Last auf dem Ethernet zu erzeugen.

Zu Beginn der Messung sieht die Versuchsanleitung vor, dass ein USB-Gateway 8 Byte große Nachrichten im Abstand von 1 ms über das LAN schicken soll. Die auf *PC 1* gestartete Analysesoftware `Traffic Graph` ist allerdings nicht in der Lage die durch die Übertragung entstehende Buslast anzuzeigen, da diese anscheinend zu gering ist. Hier hilft der „`IO Graph`“ von `Wireshark`, der eine LAN-Last von ca. 250 kbit/s anzeigt. Da das LAN eine maximale Übertragungsrate von 10 Mbit/s besitzt, entspricht dies einer Auslastung von 2,5 %.

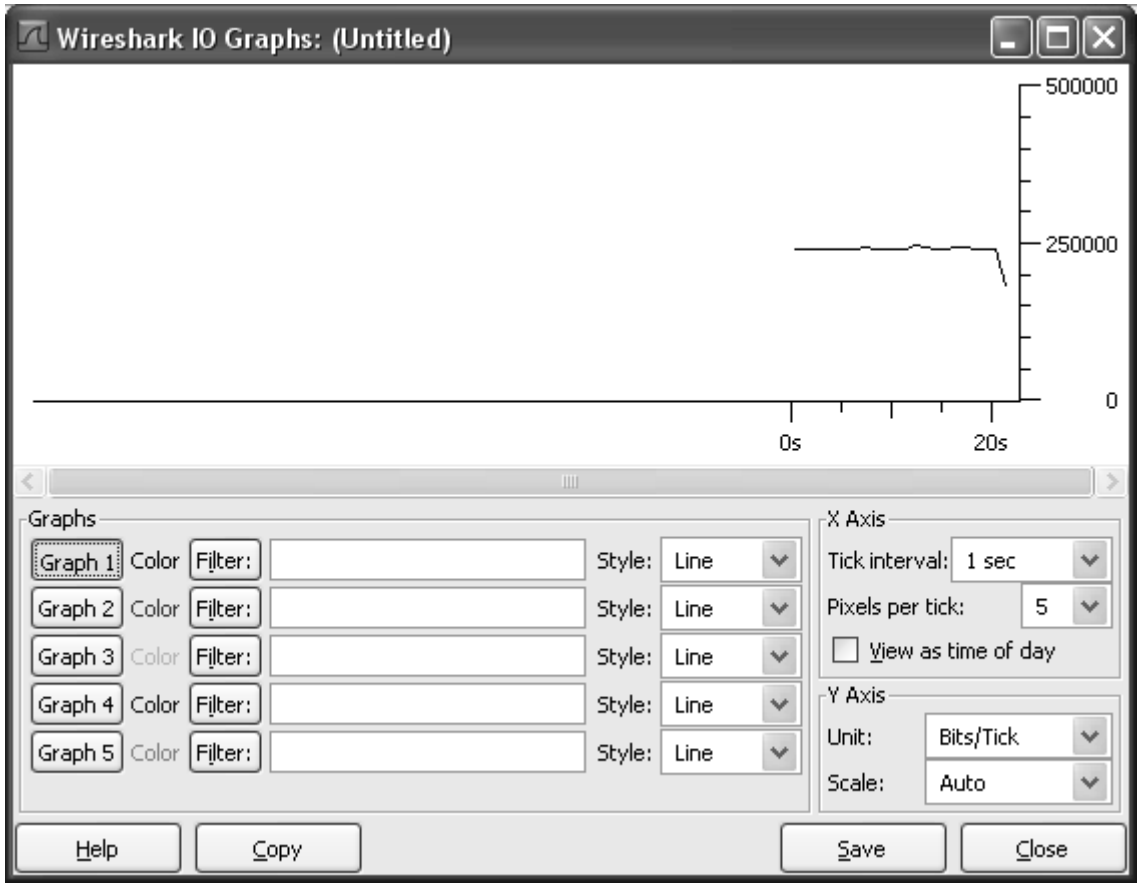

Abb. 41: Auslastung des Ethernets von 250 kbit/s durch ein USB-Gateway

Nun wird zunächst alle 1000 µs ein Paket aus `PackETH` heraus verschickt. Bei den Parametern der Versuchsanleitung ergibt die Messung unter diesen Bedingungen eine Last von 848 kbit/s.

Abb. 42: Auslastung des Ethernets von 848 kbit/s bei einem Paketabstand von 1000 µs

Ab einem Paketabstand von ca. 135 µs sind erstmals Veränderungen im Verhalten der CAN-Gateways zu bemerken. Die LEDs der empfangenden Komponenten blinken nicht mehr gleichmäßig, was bedeutet, dass nicht in regelmäßigen Abständen Datenrahmen gelesen werden. Dies ist darauf zurückzuführen, dass die entsprechenden Pakete nicht mehr gleichmäßig über das Ethernet verschickt werden. Die Auslastung des LANs liegt während dieser Messung bei

ca. 6,3 Mbit/s, was rund 63 % entspricht.

Abb. 43: Auslastung des Ethernets von ca. 6,3 Mbit/s bei einem Paketabstand von 135 µs

Ab einem Paketabstand von ca. 110 µs werden nur noch in unregelmäßigen Abständen getunnelte CAN-Nachrichten verschickt (Auslastung: ca. 7,7 Mbit/s). Sinkt der Paketabstand unter 80 µs, bricht die Übertragung zwischen den CAN-Segmenten de facto zusammen. Das Ethernet führt in diesem Fall ca. 8,2 Mbit/s.

Abb. 44: Auslastung des Ethernets von ca. 8.2 Mbit/s bei einem Paketabstand von 80 µs

Nachfolgend finden sich die weiteren Antworten und Lösungen zum sechsten Versuch:

- *Auslastung*: Theoretisch ist das Ethernet in der Lage Daten mit bis zu 10 Mbit/s zu transportieren. In der Praxis verhindert der nicht deterministische Buszugriffsalgorithmus[8] allerdings, dass dieser Wert erreicht werden kann, da es bei Kollisionen grundsätzlich zu Wartezeiten zwischen den versendeten Paketen kommt.
- *Ethernet und Echtzeitbetrieb*: Das klassische Ethernet eignet sich nicht für den Echtzeitbetrieb. Dies ist einerseits darauf zurückzuführen, dass das Buszugriffsverfahren nicht deterministisch ist, also nicht im Voraus berechnet werden kann, welcher Knoten wann Zugriff auf den Bus hat. In geswitchten Umgebungen wird dieses Problem zwar umgangen, allerdings kennt das Ethernet keine Paketpriorisierung. Ein Scheduling innerhalb der Geräte ist somit nicht ohne Weiteres realisierbar. Weitere Informationen finden sich u. a. in der Literatur [Lam108].

8 *Carrier Sense Multiple Access with Collision Detection* (*CSMA/CD*). Engl. in etwa: „Trägererkennung und Mehrfachzugriff mit Kollisionserkennung“. Siehe: [Lam108].

- *Ethernetbrücke in Echtzeitumgebung*: Die Ethernet-CAN-Brücke lässt sich nur sehr eingeschränkt in Echtzeitumgebungen einsetzen. Verfügt das LAN zwischen den CANs über garantierte Ressourcen, im besten Fall sogar über ein völlig autarkes Netz, ist es unter Umständen für weiche Echtzeitsysteme geeignet. In einer harten Echtzeitumgebung ist der Einsatz aber eher nicht zu empfehlen, da aus oben genannten Gründen das Ethernet grundsätzlich nicht über Echtzeiteigenschaften verfügt.
- *Vor- und Nachteile der Brücke*: Der große Vorteil der Ethernetbrücke ist, dass zwei räumlich getrennte CANs relativ einfach zu einem logischen Netzwerk zusammengeschlossen werden können. Verwendet man beispielsweise einen VPN-Tunnel, ist auch die Kommunikation über das Internet denkbar. Nachteilig ist, wie bereits erwähnt, die fehlende Echtzeiteigenschaft des Ethernets.
- *Vor- und Nachteile von Ethernet und CAN*: Ethernet und CAN verfolgen unterschiedliche Ansätze und sind für verschiedene Anwendungsfälle konzipiert. Als Feldbus kann das CAN in Echtzeitumgebungen eingesetzt werden und ist sehr störungsresistent. Dafür eignet es sich nur für räumlich kleine Netzwerke, bietet eine relativ geringe Übertragungsrate und ist vergleichsweise kompliziert einzurichten. Das Ethernet hingegen ist als paketvermittelte Netzwerktechnologie bestens für den Einsatz in lokalen, aber auch in weiträumigen Netzwerken geeignet. Es bietet eine hohe Übertragungsrate und seine Komponenten sind vergleichsweise günstig und weit verbreitet. Jedoch ist auch das Ethernet, zumindest im unmodifizierten Zustand, nicht für jede Umgebung die beste Lösung. Wie erwähnt ist es nicht echtzeitfähig und auch in anderen Gebieten, wie dem Einsatz als Backbonetechnologie, sollte man von einer Verwendung eher absehen.
- *Vergleich des Rahmenaufbaus*: Die Rahmen von CAN und Ethernet weisen in manchen Teilen eine gewisse Ähnlichkeit auf. So verfügen beide über eine Startsequenz bevor Nutzdaten übertragen werden, über eine Prüfsumme und natürlich über eine Payload bzw. Informationen über deren Länge. Hier enden jedoch bereits die Gemeinsamkeiten. So verfügt ein CAN-Frame über keine klassische Adressierung, lediglich der Nachrichtenidentifier ist im Rahmen enthalten. Da beim CAN die ID bereits auf den Inhalt der Nachricht schließen lässt und nur von einer Quelle stammen kann, ist dies auch nicht nötig. Das Ethernet hingegen beinhaltet sowohl die MAC-Adresse des Senders als auch des Empfängers. Darüber hinaus bietet der Ethernetframe optional die Möglichkeit einen VLAN-Tag einzubinden, um in einem fysikalischen mehrere logische LANs zu betreiben. Ein weiterer markanter unterschied zwischen Ethernet- und CAN-Rahmen

ist außerdem, dass im Ethernet pro Frame eine Nutzlast von bis zu 1500 Byte transportiert werden kann. Im CAN ist diese maximal 8 Byte groß.

7 Test des Versuchs

Um zu überprüfen, ob der entwickelte Laborversuch auch praxistauglich ist und wo ggf. Veränderungen und Verbesserungen vorgenommen werden müssen, wurden mehrere Testläufe durchgeführt. Diese lassen sich in zwei Kategorien aufteilen, die nachfolgend separat betrachtet werden: selbst durchgeführte Tests, sowie Tests mit Versuchsteilnehmern.

I Selbst durchgeführte Tests

Zur Entwicklung eines Laborversuchs gehören selbstverständlich Testläufe, mit denen geprüft wird, ob sich die Versuchsaufgaben überhaupt wie geplant umsetzten lassen. Jeder Teilversuch wurde daher mehrfach durchgeführt und auf folgende Qualitätskriterien hin überprüft:

- Sinnhaftigkeit der Aufgaben
- Realisierbarkeit der Aufgaben mit den eingeplanten technischen Mitteln
- Lösbarkeit der Aufgaben im dafür vorgesehenen Zeitraum
- Verständlichkeit der Aufgabenstellungen
- Wissenserwerb durch die Aufgaben (Lerneffekt)
- Minimierung nicht relevanter Aufwände

Durch diese Kriterien wurde nahezu jeder Teilversuch während des Bearbeitungszeitraumes modifiziert. Der Großteil dieser Optimierungen fand jedoch in der Anfangsfase statt, so dass sie lediglich als Teil des Aufgabenfindungsprozesses zu werten sind.

Nach der Fertigstellung des ersten Entwurfs der Versuchsanleitung wurden die meisten Änderungen am vierten Versuch durchgeführt. So war beispielsweise lange angedacht, dass die Studierenden u. a. überprüfen was passiert, wenn Geräte-IDs mehrfach vergeben werden. Diese Teilaufgabe wurde jedoch verworfen, da nach einigen Tests klar wurde, dass der Erkenntnisgewinn, und damit der Mehrwert des Versuchs, gegen Null tendierte (es wäre in diesem Fall kein Unterschied zum regulären Betrieb festzustellen gewesen). Dafür hätte die Durchführung jedoch ca. 30 Minuten in Anspruch genommen und wäre damit unverhältnismäßig zeitaufwändig gewesen. Zum Ausgleich wurden die anderen Teilaufgaben des Versuchs ein wenig ergänzt.

Ein weiteres Beispiel für Veränderungen in einem fortgeschrittenen Stadium stellt die Verwendung des UDPs dar. Angedacht war, in den Versuchen 5 und 6 dieses an Stelle des TCPs zu verwenden, da die Protokollstruktur einfacher ist. Jedoch verhinderte die Instabilität der Ethernet-CAN-Gateways lange Zeit überhaupt den korrekten Einsatz des verbindungslosen Protokolls. Erst durch das Aufspielen einer angepassten Firmware konnten die gröbsten Probleme behoben werden. Nach wie vor ist das UDP in der Versuchsanleitung lediglich als Option vorgesehen, da mit dem TCP in Tests die besseren Ergebnisse erzielt worden sind.

Als letztes sei noch auf Veränderungen im Softwareportfolio des Laborversuchs aufmerksam gemacht. So erfüllten bestimmte Programme, die zuerst für eine bestimmte Funktion angedacht waren, nicht alle Kriterien und mussten durch andere Anwendungen ersetzt werden. Hierfür ist `Wireshark` ein Beispiel, das nicht nur für die Paketanalyse, sondern auch für die Erstellung von LAN-Verkehrsstatistiken gedacht war. Grundsätzlich ist die Software zwar durch den integrierten „`IO Graph`“ dazu in der Lage, bei hohem Datenverkehrsaufkommen übersteigt der Ressourcenhunger des Programms aber die Kapazitäten des Computers. Aus diesem Grund wurde auf die simple Visualisierungssoftware `Traffic Graph` zurückgegriffen. Ähnlich verhielt es sich beim Paketgenerator für den sechsten Versuch. Hier wurden unterschiedliche Programme ausprobiert und nach einigen Tests verworfen. Lediglich die Anwendung `PackETH` konnte überzeugen, auch wenn sich ihre Konfiguration als relativ kompliziert erwies.

II Tests mit Versuchsteilnehmern

Um einen Testlauf aller Versuchsaufgaben unter realen Bedingungen zu ermöglichen, haben sich freundlicherweise zwei Studenten, Rico Kalbitz und Robert Bärtl[9], dazu bereit erklärt den gesamten Laborversuch durchzuführen. Hierfür wurde den Teilnehmern die zu diesem Zeitpunkt bereits fertige Anleitung ausgehändigt, mit der Bitte den Versuch vorzubereiten.

Der eingeplante Zeitraum für die Durchführung betrug ursprünglich vier Zeitstunden, benötigt wurden im Endeffekt aber lediglich drei. Dies ist darauf zurückzuführen, dass vor Beginn bereits alle Geräte fertig konfiguriert waren und die beiden Studenten während des gesamten Versuchs begleitet wurden.

Bei der Absolvierung der Aufgaben kamen die beiden „Versuchspersonen“ ohne Probleme voran. Es konnten nahezu alle Fragen beantwortet und alle Aufgaben gelöst werden. Auch der Aufbau der Versuchsanordnungen stellte keine Hürde dar. Die beiden Teilnehmer haben

9 Beide waren zu diesem Zeitpunkt bereits Absolventen des Diplom- (Nachrichtentechnik) und des Masterstudiengangs (Informations- und Kommunikationstechnik).

außerdem bei jedem Teilversuch begriffen was sie tun und warum die Aufgaben gestellt worden sind. Schließlich haben „Testfragen“ am Ende des Versuchs ergeben, dass beide einerseits das Funktionsprinzip des CANs verstanden haben und andererseits erfassen konnten, wie ein Echtzeitsystem funktioniert.

Das Feedback zum Laborversuch fiel weitgehend positiv aus. Die Aufgabenstellungen wurden als verständlich und gut strukturiert bewertet, das Niveau als angemessen. Den Versuch als Ganzes beurteilten die Studenten als interessant und erkenntnisreich. Allerdings wurde angeregt ein Kapitel zu schaffen, das praktische Beispiele für CAN-Anwendungsfälle aufzeigt, damit sich die VersuchsteilnehmerInnen besser vorstellen können wie und wo der Feldbus eingesetzt wird. Aus diesem Grund wurde die Versuchsanleitung nachträglich um einen weiteren Abschnitt[10] ergänzt.

8 Probleme im Rahmen der Masterarbeit

Im Rahmen dieser Masterarbeit traten mehrere Probleme auf, die allesamt gelöst oder umgangen werden konnten. Grundsätzlich lassen sich diese in Probleme vor und während der Bearbeitung unterteilen. Um welche Schwierigkeiten es sich jeweils handelte, welche Auswirkungen diese hatten und wie sie schließlich beseitigt werden konnten bzw. was dafür notwendig wäre, sei nachfolgend dargestellt.

I Probleme vor der Bearbeitung

Im Vorfeld dieser Arbeit bereitete insbesondere die Bestellung der benötigten Komponenten Schwierigkeiten. So war zwar relativ schnell klar welche Teile für den Versuchsaufbau benötigt würden, jedoch führte der Hauptlieferant der Deutschen Telekom für elektronische Kleingeräte, die Firma Distrelec, diese nicht. Damit waren fünf weitere Schritte nötig um die gewünschten Komponenten zu erhalten:

1. Suche nach einem geeigneten Händler
2. Erstellung des „Warenkorbs“ bei der Deutschen Telekom
3. Akkreditierung des Händlers durch die Deutschen Telekom
4. Verbuchung der Komponenten als Investition oder Ausgabe
5. Bestellung der Komponenten durch die Deutsche Telekom

10 Siehe: *4.4.7 CAN in der Praxis.*

Während sich die Suche nach einem geeigneten Händler auf wenige Tage beschränkte und auch die Erstellung des „Warenkorbs“ innerhalb von zwei Wochen erledigt war, geriet der Bestellprozess dann bei der Deutschen Telekom ins Stocken. So wurden hier zuerst die CAN-Komponenten als nicht für das Kerngeschäft notwendig erachtet und der „Warenkorb“ mit dieser Begründung abgelehnt. Leider dachte niemand daran diese Information weiterzuleiten, so dass bereits einige Wochen vergingen, bevor überhaupt bekannt wurde, dass noch gar keine Bestellung aufgegeben worden war. Nachdem das Accounting der Deutschen Telekom davon überzeugt werden konnte derartige Bestellungen in Zukunft nicht mehr abzulehnen, hatte die gesamte Prozedur erneut zu erfolgen. Da gerade die Urlaubszeit begonnen hatte, war eine zügige Bearbeitung des Vorgangs nun leider nicht möglich.

Letztlich dauerte es knappe drei Monate bis die benötigten Komponenten die Hochschule erreichten. Dementsprechend konnte die Bearbeitung dieser Masterarbeit auch nur mit Verzögerung beginnen.

An dieser Stelle sei dringend angeraten den Bestellprozess – sowohl seitens der Deutschen Telekom als auch in der Hochschulverwaltung – zu überprüfen und ggf. zu überarbeiten.

II Probleme während der Bearbeitung

Während der Entwicklung des Laborversuchs waren insbesondere hard- und softwaretechnische Probleme zu lösen. Die größten Schwierigkeiten bereiteten hierbei die beiden Ethernet-CAN-Gateways, die sich bei hoher Last im LAN ständig zu quasi zufälligen Zeitpunkten neu gestartet haben. Dies führte zu häufigen Verbindungsabbrüchen, was die Durchführung von Versuch 6 de facto unmöglich machte und andere zumindest erschwerte. Ursache für den Reset war, laut der Firma Sys-Tec, ein Watchdog in der Firmware der Module. Im praktischen Einsatz wäre dieser für die Wiederherstellung einer Verbindung zwischen zwei Gateways zuständig, falls es zum Abbruch der Kommunikation käme. Das Aufspielen einer für den Laborversuch modifizierten Firmware löste schließlich das Problem und ermöglichte einen weitgehend fehlerfreien Betrieb der Ethernetmodule.

Weniger gravierende Probleme im Softwarebereich bestanden des Weiteren beim Programm `CAN-REport`. Zwar funktionierten hier alle wichtigen Funktionen, jedoch wies die Anwendung einige zeitraubende Fehler auf. So waren beispielsweise einzelne Menüeinträge nicht mit den gewünschten Funktionen verknüpft oder führten gar zum Absturz des Programms. Darüber hinaus funktionierte die Abschaltung des Zeitstempels im `Trace View` nur für empfangene CAN-Rahmen, nicht jedoch für diejenigen, die von Messgerät selber

versendet worden sind. Zwar konnte keines dieser Probleme behoben werden, durch konsequente Nichtbenutzung der entsprechenden Softwaremerkmale wurden allerdings diese Fehler umgangen.

Letztlich bereitete noch die Leistungsfähigkeit der Labor-PCs Schwierigkeiten. Vorgesehen war bei der Entwicklung des Laborversuchs zunächst, dass lediglich ein PC verwendet werden solle. Da aber bereits das Versenden von mehreren CAN-Frames den ersten mit Microsoft Windows XP ausgestatteten Rechner an seine Kapazitätsgrenze gebracht hatte, war eine Messung und Auswertung des Datenverkehrs mit dem selben Gerät technisch nicht realisierbar. Freundlicherweise wurde ein weiterer PC für den Laborversuch bereitgestellt, so dass schließlich ein Computer für die Generierung von Datenverkehr und einer für dessen Messung zur Verfügung standen.

9 Nachwort

Blicke ich auf die Entstehung dieser Masterarbeit zurück, so kommen mir zuerst die anfänglichen Schwierigkeiten in den Sinn. Obwohl mich das Thema nach wie vor sehr interessierte, hatte ich, aufgrund der immensen Verzögerung des Bestellvorgangs, ernsthaft in Erwägung gezogen die Bearbeitung abzubrechen und meine Abschlussarbeit einer anderen Problemstellung zu widmen. Letztlich erwies es sich aber als richtige Entscheidung dabei zu bleiben. Die bestellten Komponenten kamen zwar spät, aber immerhin noch rechtzeitig um mit einem festen Zeitplan den Laborversuch dennoch verwirklichen zu können.

Nachdem ich die ersten Wochen v. a. damit verbracht hatte, die Hard- und Software – sowie ihre Mängel – kennen zu lernen, verlief die Entwicklung der Versuchsaufgaben erstaunlich gut. Die festgesetzten Ziele halfen mir einerseits dabei den Überblick zu bewahren und durch die häufigen Tests war es möglich das gewünschte Maß an Qualität zu erreichen.

Bei der Zieldefinition des Laborversuchs griff ich auf meine eigene Erfahrung zurück, denn während meines Diplom- und Masterstudiums hatte ich fast jedes Semester verschiedene Laboraufgaben bearbeitet. Gelernt haben meine KommilitonInnen und ich bei der Durchführung von Laboren immer dann etwas, wenn die Aufgaben durchdacht waren und der Lerneffekt im Vordergrund stand. Versuchen, die lediglich den Selbstzweck erfüllten, konnten wir hingegen nur schwer neue Erkenntnisse abringen und wir hätten unsere Zeit damals lieber sinnvoller in unser Studium investiert. Aus diesem Grund habe ich der *Wissensvermittlung* in meiner Arbeit immer Priorität eingeräumt.

Um die VersuchsteilnehmerInnen nicht zu überfordern, gleichzeitig aber dem Anspruch an

das Masterstudium gerecht zu werden, habe ich die Versuchsreihenfolge so gewählt, dass die Teilversuche aufeinander aufbauen und zuvor erworbenes Wissen möglichst sofort wieder angewandt werden kann. Dabei lassen sich die sechs Aufgaben grob in drei Gruppen gliedern. Die Versuche 1 und 2 dienen der Schaffung von Grundlagen, damit das Funktionsprinzip des CANs verstanden wird. Versuch 3 und 4 ergänzen dieses Basiswissen und runden die Kenntnisse über den Feldbus und seine Eigenschaften ab. Die letzten beiden Teilversuche widmen sich hingegen verstärkt dem Ethernet und verdeutlichen so die Unterschiede zum CAN, v. a. im Bereich der Echtzeitfähigkeit. Der Schwierigkeitsgrad der Aufgaben steigt dabei mit jedem Schritt an und es wird zunehmend tieferes Hintergrundwissen vorausgesetzt. Der Zeitaufwand, den die Studierenden in den Gesamtversuch investieren müssen, sinkt bei guter Vorbereitung signifikant.

Als gute Entscheidung hat es sich erwiesen die entwickelten Aufgaben regelmäßig kritisch zu prüfen. Zwar wurde, nicht zuletzt dank meiner Kommilitonen, immer wieder Verbesserungsbedarf bei den Aufgaben festgestellt. Diese konnten aber jedes Mal sinnvoll optimiert werden, so dass letztlich ein praxistauglicher Versuch entstanden ist.

Abschließend bleibt zu sagen, dass mir die Bearbeitung dieses Themas, trotz der ungewissen Anfangsfase und des teils stressigen Zeitplans, sehr viel Spaß bereitet hat. Es freut mich, dass ich die CAN-Technologie, die ich in der Theorie schon länger kenne, endlich selber anwenden konnte. Darüber hinaus finde ich es fantastisch, dass diese Arbeit meine zukünftigen Kommilitoninnen und Kommilitonen in ihrem Studium voranbringen und damit einen sinnvollen Beitrag zur ihrer Ausbildung leisten wird. Und ich bin guten Mutes, dass dieser Laborversuch bei dem Einen oder der Anderen das Interesse für das Themengebiet der Echtzeitnetze weckt.

Literaturverzeichnis

[Enge00] Engels, H.: "CAN-Bus", Franzis, Poing, 2000, ISBN: 978-3772351464.

[Etsc00] Etschberger, K.: "Controller-Area-Network", Hanser, München, 2000, ISBN: 978-3446194311.

[Gruh01] Gruhler, G.: "Feldbusse und Geräte-Kommunikationssysteme", Franzis, Poing, 2001, ISBN: 978-3772357459.

[ISO103] International Organization for Standardization: "Road vehicles - Controller area network (CAN) - Part 1: Data link layer and physical signalling", ISO, Genève, 2003, Referenznummer: ISO 11898-1:2003(E).

[ISO203] International Organization for Standardization: "Road vehicles - Controller area network (CAN) - Part 2: High-speed medium access unit", ISO, Genève, 2003, Referenznummer: ISO 11898-2:2003(E).

[ISO306] International Organization for Standardization: "Read vehicles - Controller area network (CAN) - Part 3: Low-speed, fault-tolerant, medium-dependent interface", ISO, Genève, 2006, Referenznummer: ISO 11898-3:2006(E).

[ISO404] International Organization for Standardization: "Road vehicles - Controller area network (CAN) - Part 4: Time-triggered communication", ISO, Genève, 2004, Referenznummer: ISO 11898-4:2004(E).

[ISO507] International Organization for Standardization: "Road vehicles - Controller area network (CAN) - Part 5: High-speed medium access unit with low-power mode", ISO, Genève, 2007, Referenznummer: ISO 11898-5:2007(E).

[Jans00] Jansen, W. / Phoenix Contact: "Grundkurs Feldbustechnik", Vogel, Würzburg, 2000, ISBN: 978-3802318139.

[Lam108] Lammermann, S.: "Ethernet as a Real-Time Technology", Hochschule für Telekommunikation Leipzig, Leipzig, 17. Juni 2008, http://www.lammermann.eu/wb/pages/arbeiten/ethernet-as-a-real-time-technology.php.

[Lam208] Lammermann, S.: "Controller Area Network", Hochschule für Telekommunikation Leipzig, Leipzig, 30. November 2008, http://www.lammermann.eu/wb/pages/arbeiten/controller-area-network.php.

[Maye06] Mayer, E.: "Datenkommunikation im Automobil - Teil 2: Sicherer Datenaustausch mit CAN", Elektronik automotive, WEKA Fachmedien, Poing, 08/2006, Seite 34 ff., ISSN: 1614-0125.

[Will93] Williams, R.: "A Painless Guide to CRC Error Detection Algorithms", University of Adelaide, Adelaide, 19. August 1993, http://www.ross.net/crc/download/crc_v3.txt.

Lizenz

Dieses Dokument unterliegt einer Creative-Commons-Lizenz (BY-ND). Zusammenfassung:
Sie dürfen:

- Das Werk bzw. den Inhalt vervielfältigen, verbreiten und öffentlich zugänglich machen.

Zu den folgenden Bedingungen:

- **Namensnennung**: Sie müssen den Namen des Autors/Rechteinhabers in der von ihm festgelegten Weise nennen.
- **Keine Bearbeitung**: Dieses Werk bzw. dieser Inhalt darf nicht bearbeitet, abgewandelt oder in anderer Weise verändert werden.

Wobei gilt:

- **Verzichtserklärung**: Jede der vorgenannten Bedingungen kann aufgehoben werden, sofern Sie die ausdrückliche Einwilligung des Rechteinhabers dazu erhalten.
- **Sonstige Rechte**: Die Lizenz hat keinerlei Einfluss auf die folgenden Rechte:
  - Die gesetzlichen Schranken des Urheberrechts und sonstigen Befugnisse zur privaten Nutzung;
  - Das Urheberpersönlichkeitsrecht des Rechteinhabers;
  - Rechte anderer Personen, entweder am Lizenzgegenstand selber oder bezüglich seiner Verwendung, zum Beispiel Persönlichkeitsrechte abgebildeter Personen.
- **Hinweis**: Im Falle einer Verbreitung müssen Sie anderen alle Lizenzbedingungen mitteilen, die für dieses Werk gelten. Am einfachsten ist es, an entsprechender Stelle unten stehenden Link einzubinden.

Diese „Commons Deed“ ist lediglich eine vereinfachte Zusammenfassung des rechtsverbindlichen Lizenzvertrages in allgemeinverständlicher Sprache. Der genaue Wortlaut des Lizenzvertrages findet sich im Internet unter http://creativecommons.org/licenses/by-nd/3.0/de/.

Besuchen Sie http://www.lammermann.eu für weitere freie Dokumente.

Eidesstattliche Erklärung

Ich versichere, dass die vorliegende Masterarbeit von mir selbstständig verfasst worden ist. Zur Erstellung wurden von mir keine anderen als die angegebenen Quellen und Hilfsmittel verwendet.

_____

Sebastian Lammermann

Leipzig, 30. Oktober 2009