Masterarbeit

Drive-by-Wire System eines Formula Student Electric Boliden

Georg Macher

Institut für Technische Informatik
Technische Universität Graz
Vorstand: O. Univ.-Prof. Dipl.-Ing. Dr. techn. Reinhold Weiß

Betreuer/Begutachter: Ao.Univ.-Prof. Dipl.-Ing. Dr.techn. Eugen Brenner

Graz, im November 2010

Kurzfassung

In der Automobilbranche lässt sich ein verstärkter Trend zur Elektrifizierung der Fahrzeuge erkennen. Der neueste Trend zum elektrifizierten Antriebsstrang stellt nun auch Forderungen an neue voll elektronische Bedienungskonzepte. Diese Konzepte sind unter dem Namen X-by-Wire Systeme zusammengefasst und gegenwärtig einer der wichtigsten Entwicklungsbereiche der Automobilindustrie.

Einen wichtigen Träger neuer Technologien für den Automobilbereich stellt auch der Rennsport dar. Um den Trends der Wirtschaft zu folgen, wurde 2009 eine neue Rennserie der Formula Student gegründet. Die Formula Student Electric ermöglicht es engagierten Studenten, einen rein elektrischen Rennwagen zu konstruieren und ihr erworbenes Wissen in die Praxis umzusetzen.

Diese Arbeit hat ein Drive-by-Wire System für den elektrischen Rennboliden des TU Graz E-Power Racingteams, MaxWheel, zum Ziel. Das in dieser Arbeit vorgestellte Drive-by-Wire System ist Teil der Adaption des elektrifizierten Antriebsstrangs und soll dem Fahrer eine sichere und fehlertolerante Möglichkeit bieten, die zwei Elektromotore im Heck des Fahrzeugs exakt zu dosieren, um das Fahrzeug an seine Grenzbereiche zu fahren.

Diese Arbeit beschreibt alle notwendigen Schritte vom Design zur Implementierung des Drive-by-Wire Systems des Rennboliden. Weiters werden bereits verfügbare Implementierungen von X-by-Wire System genauer betrachtet und auch die rechtlichen Grundlagen zu diesen Systemen vorgestellt.

Die hohen Anforderungen an Ausfallsicherheit und Fehlerredundanz werden mittels geeigneter Entwürfe und strukturierten Vorgehensweisen verwirklicht und mit gängigen Verfahren aus der Automobilbranche verifiziert.

Abstract

The automotive industry is focusing a lot of their efforts towards the advancement of electrical control systems to be used in conjunction with electric powertrains. The latest trend in electrical power-train systems requires the development of electrical operating systems that are capable of utilizing the potential of an electric powertrain.

One of the key development areas recognized by the automotive industry is a concept categorized as Drive-by-Wire systems.

An important carrier of new technologies for the automotive industry is racing. Motorsports provide a venue for automotive companies to develop and test new technologies in a competitive atmosphere under extreme circumstances. To follow industry trends, a new racing series, Formula Student Electric was created in 2009. Formula Student Electric challenges students to design and manufacture a race car with an electric powertrain and apply technical knowledge to a real-life project.

The goal of this project is to develop a Drive-by-Wire system for TU Graz E-Power Racing Team's electric racecar 'MaxWheel'. Through the development and incorporation of a Drive-by-Wire system for the TU Graz E-Power Racing Team, a safe and reliable race vehicle will be constructed utilizing the maximum performance potential of the electric motors.

This work will outline all of the necessary steps from the design process to the implementation of the Drive-by-Wire system in the race car. Additionally, existing examples of Drive-by-Wire systems will be investigated in an effort to get a better understanding of the application and the various design approaches.

Ultimately, the TU Graz E-Power Racing Team's Drive-by-Wire system must be reliable, safe, and be constructed using generic electrical components. The completed system will be tested and verified using methods established and applied in the automotive industry.

EIDESSTATTLICHE ERKLÄRUNG

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Graz,am ...29.November 2010..... .....

(Unterschrift)

Danksagung

Diese Diplomarbeit wurde im Studienjahr 2010 am Institut für Technische Informatik an der Technischen Universität Graz durchgeführt.

Besonderer Dank gilt den Mitgliedern des TU Graz E-Power Teams für die Möglichkeit diese Arbeit überhaupt durchführen zu können und die unvergesslichen Erlebnisse der gemeinsamen Saison 2010.

Den Sponsoren des TU Graz E-Power Teams und vor allem auch meinen Sponsoren (meinen Eltern und meiner Freundin).

Weiterer Dank gilt Herrn Professor Brenner für die Chance diese Arbeit auch wissenschaftlich beleuchten zu können und meiner Mutter und meiner Tante Anna, die das Korrekturlesen dieser Arbeit übernommen haben.

Graz, im November 2010 Georg Macher

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Kapitel 1

Einleitung

In der Automobilbranche lässt sich immer verstärkter ein Trend zur Elektrifizierung der Fahrzeuge erkennen. Verschiedenste Multimediasysteme und elektronische Komfortfunktionen sind in modernen Fahrzeugen gleich wenig weg zu denken wie Fahrassistenzsysteme und elektronische Sicherheitseinrichtungen.
Der neue Trend zum elektrifizierten Antriebsstrang stellt nun auch Forderungen an neue voll elektronische Bedienungskonzepte. Diese Konzepte sind unter dem Namen X-by-Wire Systeme zusammengefasst und gegenwärtig einer der wichtigsten Entwicklungsbereiche der Automobilindustrie.

Schon seit langem ist der Rennsport ein Vorreiter für Neuerungen und Verbesserungen im Automobilsektor. Eine weitere wichtige Vorreiterrolle übernehmen verschiedenste Universitäten und Hochschulen und ihre Forschungsarbeiten und Projekte.
Somit ist der Schritt der Society of Automotive Engineers (SAE) im Jahr 1978 einen internationalen Konstruktionsbewerb für Studenten ins Leben zu rufen nicht weiter verwunderlich. Seit 1998 werden diese Konstruktionsbewerbe der Formula Student Serie auch in Europa ausgetragen und sind mittlerweile, mit über 400 Teams aus aller Welt und hoch technologischen Lösungen in vielen Rennboliden, in der Automobilindustrie und bei Zulieferunternehmen bekannt.

Dem angesprochenen Trend der Automobilindustrie folgend haben die Veranstalter des Formula Student Germany Events 2009 die Formula Student Electric ins Leben gerufen. Bei dieser Serie haben Studenten die Möglichkeit mit modernsten Technologien ein, zu Boliden mit konventionellem Antrieb, konkurrenzfähiges Rennfahrzeug mit reinem Elektroantrieb zu bauen.

Die Idee hinter dieser Serie, ein Elektrofahrzeug zu konstruieren, zu bauen und den Prototypen in einer Rennserie zu fahren, die sowohl technisches Fachwissen als auch Teamgeist und Projektmanagement von den Studenten abverlangt, scheint vollkommen aufgegangen zu sein. Bereits die erste Saison 2010 hat gezeigt, dass Elektrofahrzeuge mit den konventionell angetriebenen Rennboliden der Formula Student mithalten können und darüber hinaus ihr Potential noch nicht vollständig ausgeschöpft haben.

1.1 Motivation

Die Motivation der Industrie ist klar. Dank der Formula Student können in Zusammenarbeit von Universitäten und Partnern der Wirtschaft erworbenes Wissen und Fachkenntnisse der Studenten in die Praxis umgesetzt werden.
Für die Studenten bietet sich die Möglichkeit, die erworbenen Fähigkeiten in einer Rennserie real unter Beweis zu stellen und gleichzeitig auch auf ihr Fachwissen hinzuweisen und Kontakte zu Wirtschaftspartnern zu knüpfen.

Die TU Graz, die bereits seit 2004 mit dem TU Graz Racingteam [1] sehr erfolgreich in der Formula Student Serie vertreten ist, hat sich dieser neuen Herausforderung gestellt und nimmt mit dem TU Graz E-Power Racingteam [2] an der ersten Formula Student Electric Saison 2010 teil.

Der MaxWheel 2010 ist der erste elektrische Rennbolide dieser Serie aus Österreich und ist auf Grundlage des erfolgreichen Vorjahresmodells, des Tankia 2009, mit einem rein elektrischen Antrieb ausgestattet.

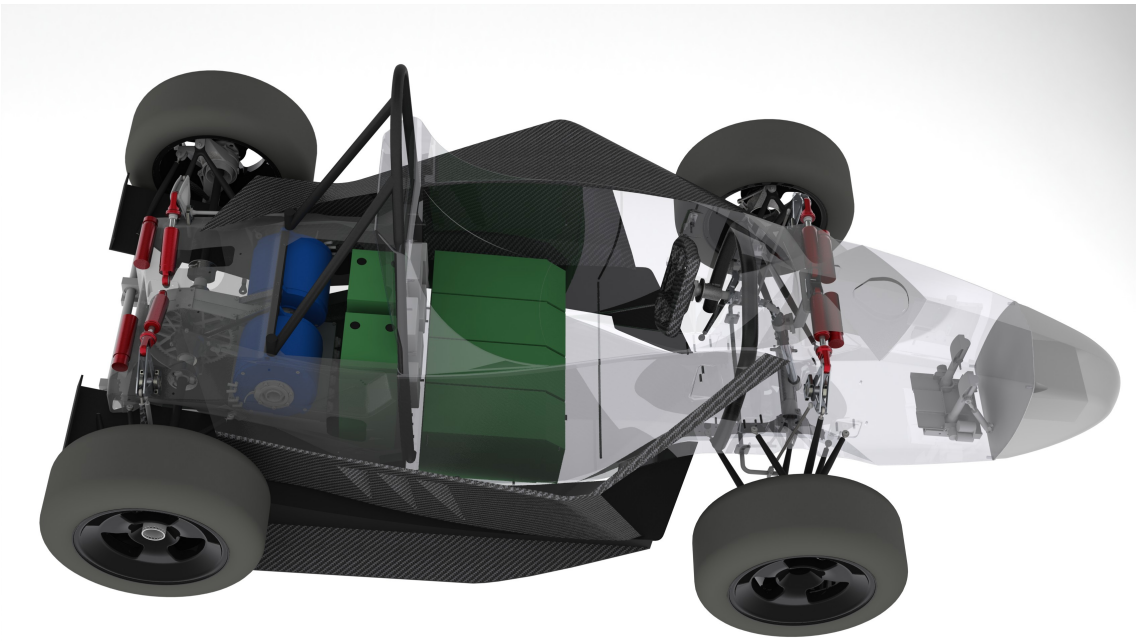


Abbildung 1.1: Rendering des MaxWheel 2010

[1] www.racing.tugraz.at
[2] www.e-power.tugraz.at

1.2 Ziel der Arbeit

Ziel dieser Arbeit ist es, ein Drive-by-Wire System für den elektrischen Rennboliden MaxWheel 2010 zu entwickeln, das den Anforderungen des Reglements, des Rennsports und den hohen Erwartungshaltungen aufgrund der bisherigen Leistungen der Elektroniker des TU Graz Racingteams entspricht.
Wie bei jedem Prototyp ist es auch das Ziel, beim Bau des MaxWheel 2010 einen Prototypen zu entwickeln, aus dem serienreife Technologien entwickelt werden können.
Jedoch handelt es sich bei diesem Fahrzeug um ein Erstjahreswerk und eine Adaption des Vorjahresfahrzeugs, was einigen Spielraum für Verbesserungen für die kommenden Jahre übrig lässt.

Das in dieser Arbeit vorgestellte Drive-by-Wire System ist Teil der Adaption des elektrifizierten Antriebsstrangs und soll dem Fahrer die sichere und fehlertolerante Möglichkeit bieten, die zwei Elektromotore im Heck des Fahrzeugs (siehe Abbildung 1.1) exakt zu dosieren und das Fahrzeug in Grenzbereichen fahrbar zu halten. Gemeinsam mit dem HMI des Lenkraddisplays und des elektronischen Differentials soll es dem Fahrer über das Drive-by-Wire System möglich sein, Beschleunigungsrennen von 0 auf $75m$ unter $3,8s$ zu bestreiten und stationäre Kreisfahrten im Grenzbereich der Reifenhaftung exakt zu dosieren. Ausfallsicherheit und Wartbarkeit stellen essentielle Kriterien dieser Implementierung dar und werden, genau wie alle weiteren Anforderungen, in diesem Dokument näher beschrieben.

1.3 Aufbau des Dokuments

Das Dokument beschreibt in fünf Kapitel das Design und die Implementierung eines Drive-by-Wire Systems für einen Rennboliden.

In Kapitel 2 werden X-by-Wire Systeme genauer beschrieben und bereits bestehende Implementierungen angeführt. Weiters werden auch die rechtlichen Grundlagen, die es für das Entwickeln und Zulassen solcher Systeme für den Straßenverkehr einzuhalten gilt, vorgestellt.

Die hohen Anforderungen an Ausfallsicherheit und Fehlerredundanz, die ein X-by-Wire System aufweisen muss, können nur mittels geeigneter Entwürfe und strukturierter Vorgehensweise verwirklicht werden. Diese Vorgehensweisen und Lösungsansätze werden in Kapitel 3 genauer beschrieben. Dies impliziert den Softwareentwurf genauso wie Hardwarestrukturen. Es werden daher in diesem Kapitel Fehlertoleranzverfahren genauso wie auch unterschiedliche automotive Bussysteme und verschiedene Messprinzipien angeführt.

Die Anforderungen an die Implementierung, Auswahl der Komponenten und eine erste Testimplementierung werden in Kapitel 4 angeführt.

An Hand der Entwurfskriterien aus Kapitel 4 werden dann in Kapitel 5 zwei Implementierungsvarianten vorgestellt. Diese Varianten werden näher beschrieben und miteinander verglichen.

Eine Zusammenfassung der Implementierung und der Ergebnisse sowie Vorschläge für weiterführende Arbeiten sind im Schlusskapitel 6 des Dokuments zu finden.

Kapitel 2

Überblick X-by-Wire Systeme im Automobil-Bereich

Dieses Kapitel soll einen kurzen Überblick darüber geben, welche Systeme unter den Begriff X-by-Wire fallen, wie sich diese Systeme entwickelt haben und welche neuen Anwendungsmöglichkeiten sich aus dem Einsatz dieser Systeme ergeben. Weiters wird unter 2.4 darauf eingegangen, welche rechtlichen Aspekte es beim Entwurf eines X-by-Wire Systems für Straßenfahrzeuge zu beachten gilt. Im anschließenden Abschnitt 2.5 werden kurz alternative Bedienelemente vorgestellt, bevor sich der Abschnitt 2.6 mit bereits bestehenden Implementierungen befasst.

Das Kunstwort X-by-Wire steht für eine Vielzahl an verschiedenen Systemen. Abhängig davon, welche Steuerung vom System übernommen wird, wird das „x" durch den jeweiligen Terminus ersetzt. So gibt es eine Vielzahl an unterschiedlichen gängigen X-by-Wire Systemen:

- Throttle-by-Wire
- Steer-by-Wire
- Brake-by-Wire
- Suspension-by-Wire
- Shift-by-Wire
- Powertrain-by-Wire
- ...

Im weiteren Verlauf dieser Arbeit wird allerdings nur auf die Systeme Throttle-by-Wire, Steer-by-Wire und Brake-by-Wire eingegangen. Die anderen Systeme sind nur angegeben, um einen Überblick zu geben, wie breit gefächert das Einsatzgebiet von X-by-Wire Systemen sein kann.

Die ersten X-by-Wire Systeme stammen aus dem Flugzeugbereich. In modernen Flugzeugen sind alle mechanischen und hydraulischen Flugkontrollsysteme vollständig durch Fly-by-Wire Systeme ersetzt. Bereits 1972 wurde das erste mit Fly-by-Wire ausgestattete Flugzeug, der NASA F-8C Crusader, von der NASA getestet. [25]

In der Zwischenzeit verfügen moderne Automobile über mehr als 80 verbaute Controller, fünf verschiedene Bussysteme und können auf eine größere Rechenleistung als das Apollo Spaceshuttle bei der Mondlandung zurückgreifen. 30 % der gesamten Fahrzeugkosten und 90 % der Innovationen in neuen Fahrzeugen können laut [22] auf elektronische Hardware und Software zurückgeführt werden.

Eines der neueren Ziele der Automobilindustrie ist nun auch die Integration verschiedenster X-by-Wire Systeme ins Fahrzeug. Hauptbeweggründe hierfür sind einerseits Reduktion der Kosten, Verbesserung des Packagings und Integration neuerer, besserer Fahrassistenz Funktionen durch das Wegfallen der mechanischen Verbindungen zwischen Fahrer und Fahrzeug. Hand in Hand mit diesen angestrebten Vorteilen gehen allerdings die höheren Anforderungen an die elektronischen Systeme im Bezug auf Sicherheit und Zuverlässigkeit.

Diese führen dazu, dass sich X-by-Wire Systeme grob in zwei Hauptgruppen einteilen lassen [20]:

1. Systeme mit mechanischer Rückfallebene
2. Systeme ohne mechanische Rückfallebene

2.1 Systeme mit mechanischer Rückfallebene

Diese Systeme sind bereits seit 1986 in Automobilen im Einsatz und repräsentieren die ersten Drive-by-Wire Systeme (z.B. elektrisches Gaspedal). Diese Systeme kennzeichnen sich dadurch, dass im Fehlerfall eine mechanische Rückfallebene existiert und selbst bei Auftreten eines sicherheitsrelevanten Fehlers der Elektronik das Fahrzeug die geforderte „limp home"-Funktion erfüllen kann. Bei dieser Gruppe von X-by-Wire Systemen muss lediglich Augenmerk auf die Erkennung eines Elektronikfehlers gelegt werden und fehlerhafte Komponenten deaktivierbar gestaltet sein, da durch Rückgriff auf die mechanische oder auch hydraulische Komponente ein sicherer Betriebszustand erhalten bleibt. Solche Systeme werden nach dem „fail-save" Prinzip entwickelt. Laut [32] darf folgende Behauptung zur mechanischen Rückfallebene getroffen werden:

> Unter Berücksichtigung der maximal zu erwartenden Belastung und der entsprechenden Auslegung der Belastbarkeit wird z.B. durch Überdimensionierung einem mechanischen Bauteilversagen konstruktiv vorgebeugt. In der Literatur werden keine nennenswerten Unfälle erwähnt, die durch den Ausfall einer mechanischen Lenkanlage verursacht wurden.

2.2 Systeme ohne mechanische Rückfallebene

Systeme ohne mechanische Rückfallebene bedeuten einen sehr großen Evolutionsschritt bei X-by-Wire Systemen. Sie verfügen über keinerlei mechanische Verbindung zwischen dem Bedienelement des Fahrers und dem Fahrzeug. Daher muss fehlertolerante Elektronik verbaut werden, die sehr hohen Sicherheitsanforderungen genügen muss. Zusätzlich ist es auch nötig, dem Fahrer haptische Informationen bereit zu stellen.

Diese Systeme haben grundsätzlich den Vorteil, dass Kosten, Gewicht und Bauraum, die sonst für die Mechanik beansprucht werden, wegfallen. Weiters bieten sie auch die Freiheit, das volle Potential elektrischer Lösungen auszuschöpfen und Fahrassistenzfunktionen zu implementieren, die mit einer mechanischen Verbindung zwischen Bedienelement und Fahrzeug nicht oder nur mit erheblichem Aufwand möglich wären.

Beispiele für solche Funktionen sind:

- Adaptive Cruise Control
- „Stop and Go" Funktionalität
- Spurassistent
- neue Arten des HMI (siehe 2.5)
- diverse Konzepte zur Reduktion des Kraftstoffverbrauchs
- situationsabhängige Lenkübersetzung
- Kompensation von Störeinflüssen (z.B: Kopfsteinpflaster, Seitenwind)
- Reduktion der Schadstoffemission

Einige der wichtigsten Faktoren, die es zu bewältigen gilt, um X-by-Wire Systeme im Automobilmarkt zu etablieren (siehe [25]):

- zuverlässige Energieversorgung
- zuverlässige Aktuatorik
- redundante und fehlertolerante Kommunikationsprotokolle
- fehlerfreie und präzise Sensorik
- standardisierte, zuverlässige Softwareprotokolle und Tools

2.3 X-by-Wire in automotiven Systemen

In diesem Abschnitt werden einige der gängigsten X-by-Wire Systeme im automotiven Bereich und ihre Funktionsweisen aufgelistet und kurz erklärt.

2.3.1 Throttle-by-Wire

Throttle-by-Wire ist oft auch bekannt als „Electronic Throttle Control" und ist das erste im kommerziellen Fahrzeug verbaute und bekannteste X-by-Wire System. Bei diesem System existiert keine mechanische Verbindung (z.B. Seilzug) zwischen Gaspedal und der Einspritzanlage und es stellt den de facto Standard bei fast allen Fahrzeugen neuerer Generation dar. Mittels dieses Systems wurde Einspritzsystemen wie Common Rail und Direct Injection der Weg ins Automobil bereitet, da der Fahrerwunsch einfach von der Motorsteuerung in die jeweils dem Drehzahl- und Leistungsbereich angepasste Einspritzmenge umgerechnet werden kann.

Eine große Vereinfachung bei diesem X-by-Wire System ist, dass das Feedback für den Fahrer relativ einfach und sehr realitätsgetreu über ein Gegenmoment von einer Feder verwirklicht werden kann und dass im Falle eines Defekts der Elektronik das System auf einen sicheren „limp home" Status zurückfallen kann.

Dies vereinfacht die gesamte Struktur des Systems und damit auch die Kosten, da weniger redundante Teilsysteme und kein elektrischer Feedback Kanal benötigt werden. Weiters benötigt dieses X-by-Wire System, im Vergleich zu den anderen, keine Aktoren mit hohem Leistungsbedarf und ist daher einfacher im $12V$ Bordnetz integrierbar.

Wie im Konzept-Bild einer typischen ETC Applikation 2.1 ersichtlich, wird der Fahrerwunsch von 2 bis 3 redundanten Sensoren aufgenommen und der ECU zur weiteren Verarbeitung zugeführt. Die ECU regelt dann die gewünschte Drosselklappenstellung oder Einspritzmenge. Ein, an der Einspritzvorrichtung angebrachter, Sensor liefert der ECU den aktuellen Istzustand und schließt somit den Regelkreis.

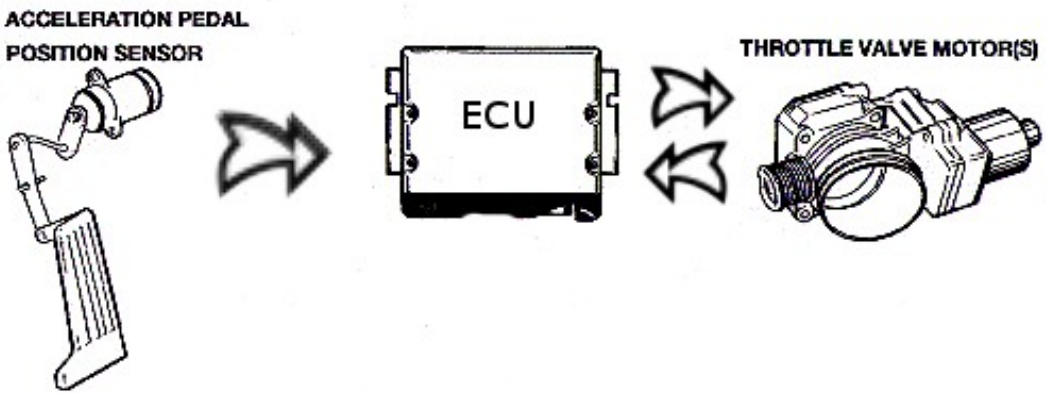


Abbildung 2.1: Throttle-by-Wire Konzept

2.3.2 Steer-by-Wire

Steer-by-Wire Systeme sind komplett mechatronische Systeme ohne Lenksäule. Das heißt, diese Systeme greifen mittels Elektromotore direkt am Lenkgetriebe an und benötigen keine hydraulischen Flüssigkeiten oder Lenkunterstützungspumpen und Lenksäulen mehr. Eine Vorstufe dieser Systeme findet sich unter dem Namen Aktivlenkung bereits in neuen Modellen der BMW 5er und 7er Reihe. Diese Systeme können bereits bis zu einer 4 %igen Treibstoffersparnis führen [14], da sie nur dann Energie benötigen, wenn ein Lenkwunsch vom Fahrer eintrifft. Im Gegensatz dazu werden normale hydraulische Lenkunterstützungen dauernd vom Motor mitbetrieben.

Das Bild 2.2 zeigt sowohl das Grundkonzept als auch die Redundanzstruktur eines Steer-by-Wire Systems für Lenkvorgabe und Rückmeldung. Wie im Grundkonzept ersichtlich besteht auch bei diesem X-by-Wire System keine mechanische Verbindung zwischen HMI und Fahrzeug. Ein Lenkwinkelsensor am Lenkrad nimmt den Fahrerwunsch auf und überträgt diesen zum Steuergerät. Dieses berechnet den optimalen Lenkwinkel und steuert den elektrischen Stellmotor am Lenkgetriebe an. Ein ebenfalls am Lenkgetriebe installierter Sensor nimmt die Umwelteinflüsse, die von der Straße auf das Fahrzeug einwirken, auf

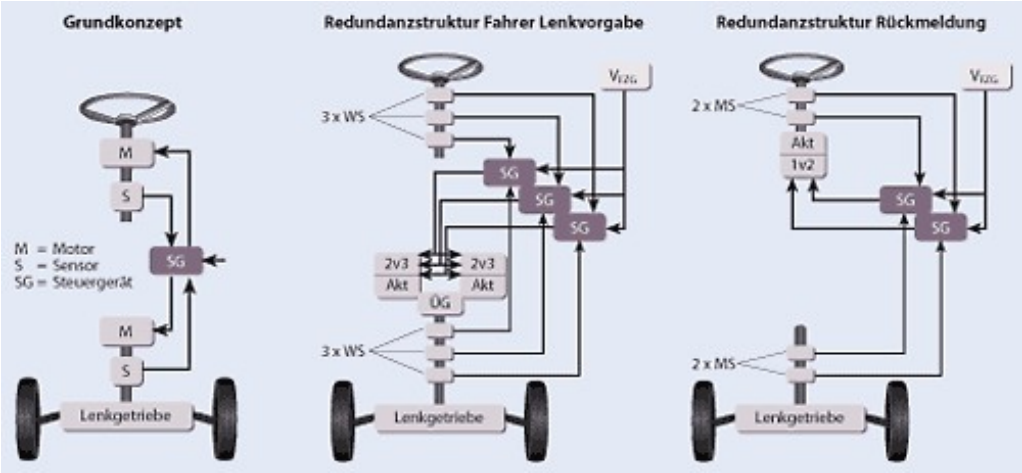


Abbildung 2.2: Funktionsprinzip Steer-by-Wire aus [18]

und stellt diese dem Steuergerät bereit. Über diese Signale wird dann vom Steuergerät das natürlich wirkende Gegenmoment am Lenkrad, ebenfalls mittels eines Elektromotors, erzeugt.

In der Realität ist der Fahrerwunschpfad (2.2 Mitte) dreifach redundant ausgeführt. Vom Lenkwinkelsensor über dessen Anbindung und Versorgung zum Steuergerät bis hin zum Aktuator und den Sensoren am Lenkgetriebe. Dort wird dann eine Mehrheitsentscheidung (2 aus 3) mittels eines Überlagerungsgetriebes (mechanischer Voter) getroffen.

Die Redundanz des Rückmeldungspfads (2.2 Rechts) ist nur doppelt ausgeführt, da ein Ausfall dieses Pfades zwar zu einer Verminderung des Fahrgefühls, nicht aber zu einem Sicherheitsrisiko führen kann. Zu den Messwerten, die zur Generierung des Fahrgefühls aufgenommen werden, zählen neben Lenkeinschlag auch Gegendruck und Fahrbahnbeschaffenheit.

Die bei Steer-by-Wire verbaute Mechatronik kann in der Praxis leichter gebaut werden als eine rein mechanische Lenkung und verbraucht weniger Energie als zum Beispiel eine Servolenkung. Allerdings wird dieser erwähnte Gewichtsvorteil auf Grund der Sicherheitsanforderungen und damit verbundenen Redundanzen wieder wett gemacht. Weiters kann durch Wegfallen der Lenksäule die passive Fahrsicherheit gesteigert und der Bauraum im Motorraum besser genutzt werden. In Verbindung mit ESP kann eine Steer-by-Wire Lenkanlage in kritischen Fahrsituationen auch aktiv gegenlenken und somit die aktive Fahrsicherheit steigern.

Diese Steer-by-Wire Systeme ohne Lenksäule sind allerdings momentan noch nicht gesetzlich erlaubt (siehe 2.4). Das „Quadrasteer" Lenksystem von Delphi kommt gegenwärtig einer vollständigen Implementation des vorher beschriebenen Steer-by-Wire Systems am nächsten, allerdings wirkt dies auf eine aktive Hinterachslenkung und ist nur in wenigen ausgewählten Fahrzeugen der Marke General Motors im Einsatz [14].

2.3.3 Brake-by-Wire

Im Gegensatz zu anderen X-by-Wire Systemen sind reine Brake-by-Wire Systeme (FEB - full electric braking, z.B. Rekuperationsbremsen) in kommerziellen Automobilen noch gar nicht im Einsatz. Hauptursache hierfür ist sicher, dass die Bremsanlage eines Fahrzeugs die sicherheitskritischste Anlage überhaupt darstellt und auch, dass FEB Bremsanlagen wie Rekuperationsbremsen nur Sinn bei Fahrzeugen mit Radnabenmotoren machen.
Weiters kommt hinzu, dass elektrische Bremsanlagen einen hohen Energiebedarf haben, der mit dem gängigen $12V$ Bordnetz nicht unkritisch bereitgestellt werden kann. Im schlimmsten Fall müssen pro Rad $500W$ Energie zur Verfügung stehen, um eine Bremskraft von $30kN$ zu erzeugen [30].

Aufgrund dessen haben sich zwei Hybridsysteme, ein elektrohydraulischer und ein elektromechanischer Hybrid, etabliert. Beide Systeme besitzen ein mechanisch entkoppeltes Pedal und können damit Pedalvibrationen unterdrücken, das Pedalfeedback situationsabhängig variieren und die Fahrsicherheit des Fahrzeuges aktiv erhöhen. Regeneratives Bremsen wie bei FEB Systemen ist in beiden Fällen nicht möglich, da beide Systeme mit einer Reibbremse arbeiten.

Elektrohydraulische Bremse

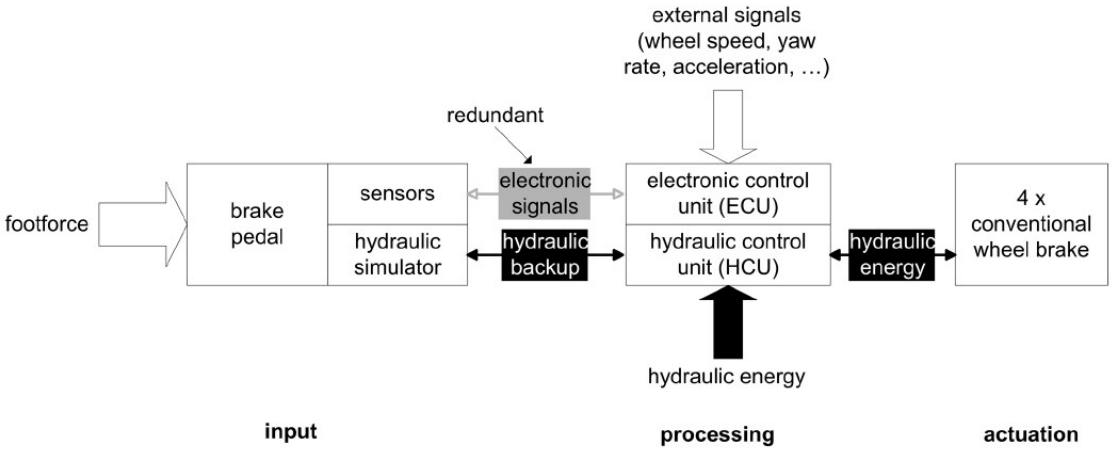


Abbildung 2.3: Systemaufbau einer EHB aus [30]

Die elektrohydraulische Bremse stellt hierbei die erste und bereits in Kundenfahrzeugen verbaute Variante eines Brake-by-Wire Systems mit einer quasi mechanischen Rückfallebene dar. Abbildung 2.3 zeigt das Systemlayout einer solchen elektrohydraulischen Bremse. Dieses Bremssystem besitzt ebenfalls einen für X-by-Wire Systeme typischen Aufbau. Ein mechanisch entkoppeltes Bedienelement (Pedal) mit redundanter Sensorik, redundante elektronische Datenübertragung zu einer fehlertoleranten Kontrolleinheit und getrennt arbeitende redundante Aktuatoren an den zu steuernden Elementen. Weiters verfügt dieses System ebenfalls über einen Rückkanal für die Rückmeldung an den Fahrer.

Das Spezielle an diesem System ist allerdings, dass sowohl die Betätigung der Aktuatoren als auch die Rückmeldung an den Fahrer nicht über elektrische Signale geschieht, sondern über ein darunterliegendes hydraulisches System.
Im Falle eines Fehlers der Elektronik oder Ausfallens der Energieversorgung werden die beiden hydraulischen Kanäle zwischen Hydraulic Control Unit (HCU) und Aktuator sowie HCU und Pedal wieder miteinander verbunden und die Bremse funktioniert wieder wie eine herkömmliche hydraulisch betätigte Bremsanlage.

Elektromechanische Bremse

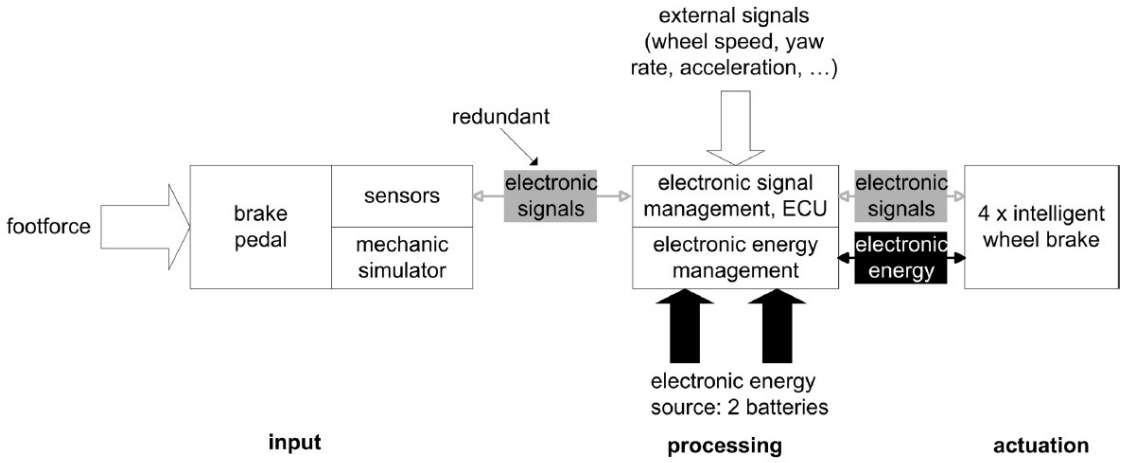


Abbildung 2.4: Systemaufbau einer EMB aus [30]

Die elektromechanische Bremse ist eine Bremsanlage, die völlig ohne Hydraulik arbeitet. Die Bremsfunktion und auch das haptische Feedback werden hierbei mittels Elektromotore verwirklicht. Dieses Bremssystem arbeitet ohne eine mechanische Rückfallebene und muss daher fehlertolerant aufgebaut werden. Dies inkludiert Redundanz in allen Teilsystemen, wie Pedalsensorik, Datenübertragung, Controller Bremsmodul, Aktorsensorik und auch Energieversorgung während des Betriebs. Diese redundante Energieversorgung stellt, wie schon unter 2.3.3 erwähnt, ein größeres Problem dar. Daher kommt dieses Bremssystem auch zumeist nur an der Hinterachse als Feststellbremse und Anfahrhilfe im Winter zum Einsatz.

2.3.4 Suspension-by-Wire

Suspension-by-Wire Systeme sind auch als „Aktives Fahrwerk" bekannt und bereits in einer Entwicklungsvorstufe (aktive Luftfederung) in Oberklassemodellen deutscher Automobilhersteller im Einsatz. Das von Volvo entwickelte „Four C" (Continuously Controlled Chassis Concept) und das von Bose entwickelte aktive Fahrwerk sind neuere Entwicklungsstufen dieses X-by-Wire Systems.

Diese Systeme detektieren Fahrbahnunebenheiten und Schläge, Über-/ Untersteuern und Wanken und adjustieren innerhalb von 5 ms Feder- und Dämpferrate für jedes Rad einzeln neu. Beim System von Volvo geschieht dies mittels elektromagnetisch gesteuerter Ventile, beim Bose System über Linearmotore, die den zusätzlichen Vorteil bieten, die Energie des Schlages in elektrische Energie zurückzuführen.[34]

Wichtigstes Kriterium dieser Gruppe der X-by-Wire Systeme sind sicherlich nicht die hohen Sicherheitsanforderungen, denen Genüge getan werden muss, denn auch mit defektem aktiven Fahrwerk kann ein sicherer Betriebszustand mit eingeschränkter Fahrbarkeit erhalten bleiben. Allerdings benötigt ein vollständiges Suspension-by-Wire System zirka $2kW$ Leistung, welche eindeutig nicht über ein konventionelles $12V$ Bordnetz zur Verfügung steht. Auch hier wird die Forderung an $42V$ Bordspannung im Fahrzeug immer lauter.

2.3.5 Shift-by-Wire

Automatische Schaltroutinen sind aktueller Stand der Technik bei fast allen Fahrzeugen mit Automatikgetrieben. Durch die elektronische Ansteuerung der Getriebe sind Schaltvorgänge in wenigen Hundert Millisekunden, 7% schnellere „Kickdown" Beschleunigungen und sehr viel kürzere Ansprechzeiten des Fahrzeugs möglich. Dieses X-by-Wire System stellt eine geringere Herausfoderung bei der Implemtierung dar, da es nicht vorrangig sicherheitsrelevant ist, im Fehlerfall über eine quasi „limp home" Funktion verfügt, und bereits eine längere Evolution von den ersten Automatikgetrieben bis zum Shift-by-Wire durchlaufen hat.

2.4 Rechtliche Grundlagen zu X-by-Wire Ansätzen im Automobil

Die rechtlichen Grundlagen, die zur Einführung von X-by-Wire Systemen in Kundenfahrzeugen nötig sind, werden in dieser Arbeit nur kurz angeführt und die wichtigsten Normen erwähnt. Näheres Eingehen auf diese Aspekte würden den Rahmen dieser Arbeit sprengen, welche die technischen Aspekte von X-by-Wire Systemen beleuchten soll. Für nähere Informationen zu diesem Teilaspekt wird empfohlen, die Arbeit [32], sowie die diversen Normen ISO26262 und IEC61508 [12] und die verschiedenen Gesetzbücher [8] und [31] zu studieren.
Das im Zuge dieser Arbeit implementierte System stellt eine Art Konzeptstudie dar und unterliegt daher nicht den gesetzlichen Grundlagen, die für ein Kundenfahrzeug Anwendung finden. Allerdings muss dem Reglement der Formula SAE und Formula Student Electric entsprochen werden. Darauf wird im Anhang unter Abschnitt A näher eingegangen.

Fahrzeughersteller sind nicht nur aus moralischen, sondern auch wirtschaftlichen Gründen daran interessiert, sicherheitsrelevante Fahrzeugsysteme mit maximaler Zuverlässigkeit zu bauen und optimalen Schutz für Verkehrsteilnehmer zu gewährleisten.
Auch die Gesetzgeber fordern gewisse Normen und Richtlinien, denen sich die Fahrzeughersteller zu unterwerfen haben.
Das Produkthaftungsgesetz beinhaltet allgemein gehalten die Anforderung an den Her-

steller, sein Produkt so auszulegen, dass durch einen Fehler des Produktes keine Gefahr ausgehen darf.

Die Vertreter der Automobilindustrie und der Gesetzgebung befassen sich gegenwärtig massiv mit einer Modifizierung der gesetzlichen Vorschriften für die Zulassungsfähigkeit von X-by-Wire Systemen. Ein Hauptaspekt dieser Überarbeitung ist die Festlegung von Fehlertoleranz- und Fehlerüberwachungsverfahren, sowie geeigneter Testverfahren zur Absicherung dieser.

Im Hinblick auf die Zulassungsfähigkeit eines X-by-Wire Systems lässt sich zunächst grob vereinfacht sagen, dass reine X-by-Wire Systeme gegenwärtig kaum zulassungsfähig sind. Dies liegt vor allem auch daran, dass die einheitliche ISO 26262 Norm noch nicht vollständig und veröffentlicht ist. X-by-Wire Systeme müssen aber mindestens die gleichen Sicherheitskriterien erfüllen wie das von ihnen ersetzte herkömmliche System. Weiters kann man davon ausgehen, dass diese Forderung nur mit größerem Aufwand bei Hard- und Software verwirklichbar sein wird.

Weitere Informationen, welche hohen rechtlichen Anforderungen die EU an ein Kraftfahrzeug-Bremssystem stellt, finden sich in der Norm 71/320/EWG [7] wieder. Die Anforderungen an ein Lenksystem eines Kraftfahrzeugs enthält die Norm 71/311/EWG [6].

Aus den folgenden Auszügen der StVZO [8] ist ebenfalls ersichtlich, dass rein elektrische Lenkanlagen ohne mechanische Verbindung zwischen Lenkrad und Rädern momentan nicht zulassungsfähig sind.

> § 2.2.2.1 Lenkanlagen dürfen keine rein elektrischen (...) und keine rein pneumatischen Übertragungseinrichtungen haben.

> §5.2.1.1.3 Es muss (...) eine kontinuierliche, gleichbleibende Abhängigkeit zwischen dem Lenkwinkel der Betätigungseinrichtung und dem Lenkwinkel der Räder bestehen.

Allerdings lässt §4.1.6 der StVZO den Spielraum noch offen, dass auch diese Systeme in Zukunft im Automobil Einzug halten können.

> §4.1.6 (...) rein elektrische Übertragungseinrichtungen (...) sind solange verboten, bis die Vorschrift dieser Richtlinie durch spezielle Vorschriften für diese Einrichtungen ergänzt wurden.

Fakt ist, dass der Stand von Wissenschaft und Technik üblicherweise schneller voranschreitet als sich die entsprechende Norm entwickelt. Dies soll sich mit der künftigen Norm für Sicherheit von Straßenfahrzeugen ISO 26262 [12], die Juli 2011 in Kraft treten und die aktuell gültige Norm IEC61508 ablösen soll, ändern.

Nach ISO26262 dürfen nur Produkte in Verkehr gebracht werden, die die Sicherheitserwartungen nach Stand von Wissenschaft und Technik zum Zeitpunkt des In-Verkehr-Bringens erfüllen. Dabei gilt diese Norm für den gesamten Lebenszyklus eines Produkts und umfasst 10 Bände:

1. Band Vokabular
2. Band Management der funktionalen Sicherheit
3. Band Konzeptphase
4. Band Produktentwicklung: Systemebene
5. Band Produktentwicklung: Hardwareebene
6. Band Produktentwicklung: Softwareebene
7. Band Anforderungen an Produktion, Betrieb, Service und Außerbetriebnahme
8. Band Unterstützende Prozesse,Qualifizierung von Tools, Anforderungsmanagement
9. Band ASIL- und sicherheitsorientierte Analysen
10. Band Informativer Leitfaden

2.5 Alternative Bedienungskonzepte

Durch das Wegfallen der mechanischen Verbindungen der Bedienelemente ergibt sich ein sehr großer Freiheitsgrad in Form, Auslegung und Positionierung dieser Bedienelemente. Zwar werden beim Großteil der verschiedenen Arbeiten zum Thema X-by-Wire Systeme konventionelle Betätigungselemente verbaut, dennoch widmen sich auch diesem Teilaspekt der X-by-Wire Systeme einige Forschungsarbeiten.

Ein Grund für das weitere Festhalten an konventionellen Betätigungselementen ist die hohe Kompatibilität zu den üblichen Betätigungselementen und die hohe Akzeptanz und Bekanntheit bei den Benutzern.
Was konventionelle Betätigungselemente vor allem für die Automobilhersteller attraktiv macht ist, dass Kosten gespart und keine Risiken bezüglich der Durchsetzung am Markt eingegangen werden müssen. Bei X-by-Wire Systemen mit mechanischer Rückfallebene (siehe Abschnitt 2.1) sind konventionelle Bedienelemente schon deshalb notwendig, weil die Rückfallebene nur über die Kraft des Fahrers betrieben werden kann.

Der Grund, warum diese alternativen Betätigungselemente hier nur kurz Erwähnung finden und nicht im Zuge dieser Arbeit zum Einsatz kommen, liegt ebenfalls in der Notwendigkeit einer vorgeschriebenen mechanischen Rückfallebene und des expliziten Verbots des Einsatzes solcher Alternativen im Reglement der Formula Student (siehe Anhang A).

2.5.1 Längsführungskonzepte

Ein alternatives Längsführungskonzept ist zum Beispiel der Ansatz aus dem Forschungsfahrzeug F500 Mind von Daimler Chrysler. Hierbei werden anstelle der Pedale für Gas und Bremse zwei Druckplatten mit eingebauten Drucksensoren eingesetzt [37]. Die Betätigung dieser beiden Systeme wird somit ohne Pedalweg nur über Fußkraft geführt. Mittels dieser Lösung werden $12cm$ mehr Beinfreiheit erreicht, welche im Crashfall das Verletzungrisiko weiter minimieren können.

Eine andere Alternative bietet die Kombination von Gas- und Bremspedal in einem Pedal. Dieses kombinierte Pedal, bei dem der Vorderfuss zum Beschleunigen und die Ferse zum Verzögern benutzt wird, hat den Vorteil, dass ein Pedalwechsel bei Notbremsungen nicht mehr notwendig ist und kann dadurch den Anhalteweg um fast $0,2s$ reduzieren.

2.5.2 Querführungskonzepte

Modifizierte Lenkräder, die keine Vollkreisform mehr besitzen, stellen eine erste Alternative der Querführungsbedienung dar. Da ein Umgreifen dank Steer-by-Wire und den damit eingeschränkten Lenkwinkeln nicht mehr notwendig ist, können Lenkräder nur mehr aus Griffstücken bestehen. Ähnlich den Lenkrädern der verschiedensten Formel-Fahrzeuge.

2.5.3 Kombinierte Längs- und Querführungskonzepte

Der überwiegende Anteil an Lösungen in diesem Bereich gehört in die Kategorie Sidestick [37]. Diese Bedienelemente stammen aus der Entwicklung von Kampfflugzeugen oder dem Computerspielebereich.
In Versuchsfahrzeugen wurden Implementierungen mit Force-Feedback-Joysticks eingesetzt. Trotz einiger verschiedener Auslegungsvarianten schnitten diese Varianten im Vergleichstest fast immer schlechter ab als herkömmliche Anordnungen von Bedienelementen. Einzig bei Ausweichmanövern konnte eine bessere Leisungsfähigkeit erbracht werden.

Zusammenfassend lässt sich sagen, dass es eine große Bandbreite an alternativen Bedienkonzepten gibt, diese aber gegenwärtig ihre hohen Zusatzkosten nicht rechtfertigen. Für eine weitere Vertiefung in diesem Themengebiet wird die Arbeit [37] empfohlen.

2.6 Bestehende Implementierungen

Die im folgenden vorgestellten Forschungsarbeiten befassen sich mit Implementierungen ähnlicher Problemstellungen und haben für diese Arbeit als Grundlage gedient.
Es wird kurz auf die einzelnen Arbeiten eingegangen, jedoch wird für ein tieferes Studium der X-by-Wire Systeme ein Studium dieser Arbeiten empfohlen.

2.6.1 X-By-Wire Safety Related Fault Tolerant Systems in Vehicles [5]

Diese Arbeit ist der Abschlussreport zu einem drei Jahre dauernden X-by-Wire Projekt der Technischen Universität Wien in Zusammenarbeit mit einigen Automobilherstellern und -zulieferern.

Ziel des Projekts war es, eine Art Framework für sicherheitsrelevante, fehlertolerante, elektronische X-by-Wire Systeme ohne mechanische Rückfallebene zu erarbeiten.
Das erarbeitete Framework sollte eine Vorbereitungsstufe für eine künftige europaweite Norm für X-by-Wire Systeme sein.

Im Lauf des Projekts wurde ein Prototyp eines Steer-by-Wire Systems ohne mechanische Rückfallebene für ein Versuchsfahrzeug implementiert und analysiert. Hierbei wurde auch darauf eingegangen, dass ein solches System aus wirtschaftlichen Gründen typischerweise aus Massenproduktionsbauteilen aufgebaut wird.

Bei diesem Projekt wurden auch Forschungsergebnisse anderer EC Projekte in die Überlegungen mit einbezogen. Speziell handelt es sich dabei um die MARS Architektur und das verwendete TTP Kommunikationsprotokoll (siehe 3.3.5). Eine Arbeit, die sich mit dem BASEMENT Konzept und einem verteilten Echtzeitsystem für Kraftfahrzeuge beschäftigt, sowie eine DACAPO Arbeit, die sich vorwiegend mit verteilten sicherheitskritischen Steuerungssystemen beschäftigt.

Verschiedene bestehende Ansätze wurden unter den Aspekten eines fehlertoleranten, sicherheitskritischen Systems analysiert. Vom Entwicklungsprozess über Hard- und Softwareanforderungen bis hin zu verfügbaren Designtools wurden Untersuchungen bezüglich Verfügbarkeit und Einsetzbarkeit von X-by-Wire Anwendungen angestellt.

Für diese Arbeit wesentliche erarbeitete Requirements

Aus den Anforderungen des Projektprototypen eines Steer-by-Wire Systems lassen sich folgende Requirements auch für diese Arbeit anwenden:

- die Auflösung des Lenkwinkelsensors muss weniger als 1° Lenkwinkel entsprechen
- das Lenksystem muss einfach und sicher lenkbar bleiben und darf die Steuerbarkeit des Fahrzeugs nicht beeinflussen
- das größte Maß an Sicherheit sollte bereits in früher Designphase das Ziel sein
- das eingesetzte System muss Fehlerinformationen über seinen internen Betriebsstatus bereitstellen
- im Falle eines nicht kritischen Fehlers soll das System weiter funktionieren, aber den Fahrer über diesen Fehler informieren
- die eingesetzte Sensorik soll aus Sicherheitsgründen auf verschiedenen physikalischen Prinzipien basieren
- ein Echtzeit-Bussystem soll verwendet werden
- eine FMEA soll für die Hardware angewandt werden
- Sensorik und Elektronik müssen in einem wasserdichten Gehäuse untergebracht werden

Evaluierungskriterium	Event-Triggered	Time-Triggered
Zuverlässigkeit		einfaches Management
Echtzeitfähigkeit	geringe Delays, limitierte Vorhersagbarkeit	wenig Jitter
Leistungsfähigkeit		wenig System Overhead
Kosten	geringere Leistungsanforderungen	
Wartbarkeit		einfaches Testen, modulare Subsysteme
Entwicklungsprozess		einfaches Testen und Validieren
spezielle Aspekte		bessere EMV

Tabelle 2.1: Event-Triggered vs. Time-Triggered Funktionalität

Evaluierung existierender Ansätze

Die folgenden Tabellen zeigen eine Zusammenfassung der im Dokument evaluierten Ansätze. Die geeigneteren Varianten wurden jeweils über die Aspekte Zuverlässigkeit, Echtzeitfähigkeit, Leistungsfähigkeit, Kosten, Wartbarkeit, Entwicklungsprozess und spezielle Aspekte evaluiert. Dabei gibt es keine Patentlösung für alle Einsatzfälle. Die Auswahl der bevorzugten oder geeigneteren Variante liegt immer noch beim Systementwickler.

Im Folgenden wird auf unterschiedliche Varianten der Softwarefunktionalität (Tabelle 2.1), der Wahl der Fehlertoleranzmethode (Tabelle 2.2), Buszugriffsprinzipien (Tabelle 2.3) und das Design eines fehlertoleranten Knotens (Tabelle 2.5) eingegangen.
Weiters werden zentralisierte Controllarchitekturen und verteilte Controllarchitekturen verglichen (Tabelle 2.4).
Der Unterschied der beiden Ansätze liegt darin, dass bei der zentralisierten Controllerarchitektur der zentraler Rechner und alle für die Steueraufgabe nötigen Sensoren, Aktuatoren und Hilfscontroller am selben zentralen Bus hängen, während beim dezentralisierten Aufbau Sensoren und Aktuatoren zusammengefasst über einen Busknoten am zentralen Bus angeschlossen sind (siehe Bild 2.5).

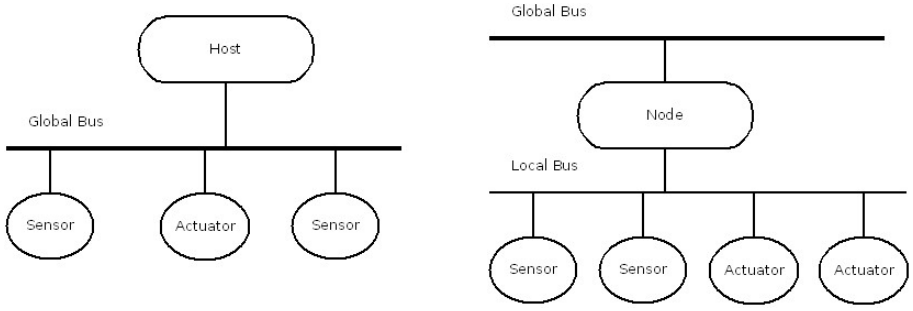


Abbildung 2.5: Unterschied zentralisierte vs. dezentralisierte Controllerarchitektur

Evaluierungskriterium	Systematische Fehlertoleranz	Applikationsspezifische Fehlertoleranz
Zuverlässigkeit	weniger Softwarefehler aufgrund der eigenständigen Applikationen	
Echtzeitfähigkeit		weniger System-Overhead
Leistungsfähigkeit		bessere Effizienz dank Applikationswissen
Kosten	geringere Entwicklungskosten	geringere Hardwarekosten
Entwicklungsprozess	höhere Qualität aufgrund einfacherer Validierung	

Tabelle 2.2: systematische vs. applikationsspezifische Fehlertoleranz

Evaluierungskriterium	Token Passing	CSMA/CD	CSMA/CA	TDMA
Zuverlässigkeit	fair	keine Fehlererkennung		einfache Fehlererkennung
Echtzeitfähigkeit	fair	nicht deterministisch		gut, keine Jitter
Leistungsfähigkeit		schlechte Performance bei hoher Buslast	mittelmäßiger Frame-Overhead	wenig Overhead, gute Performance
Entwicklungsprozess		kein Scheduling	einfaches Scheduling	strenges Scheduling
spezielle Aspekte				bessere elektromagnetische Verträglichkeit

Tabelle 2.3: Buszugriffsprinzipien

Evaluierungskriterium	Zentralisiertes System	Verteilte Systemumgebung
Zuverlässigkeit	wenige Knoten am Bus	mehr Knoten an einem Bus
Echtzeitfähigkeit	mehr Daten am Bus	weniger Daten am Bus, daher geringeres Delay
Leistungsfähigkeit	hohe Datenrate gefordert	geringere Datenrate nötig, da Vor-Ort-Vorverarbeitung
Kosten	ein Businterface pro angeschlossenem Device	

Tabelle 2.4: Zentralisierte vs. verteilte Controllerarchitektur

Evaluierungskriterium	Triple Modular Redundancy	Fail Silent Knoten
Zuverlässigkeit	jeder Einzelmodulfehler wird toleriert	Abdeckung aller Fail-Silent Bedingungen ist kritisch
Leistungsfähigkeit		doppelte Buslast aufgrund der redundanten Systeme
Kosten	dreifache Hardware-ausführung	möglicherweise spezielle Bauteile nötig

Tabelle 2.5: Design der fehlertoleranten Knoten

2.6.2 Brake-By-Wire distributed application Subsystem in the DECOS Integrated Architecture[10]

Die Arbeit beschreibt eine Implementierung einer bestehenden Brake-by-Wire Anwendung eines simulierten Fahrzeugs auf einem DECOS Prototyp Cluster.
Die DECOS-Architektur (Dependabel Embedded Components and Systems) stellt ein Framework für sicherheitskritische Echtzeit Anwendungen auf einem einzelnen verteilten Computersystem dar.
Für die Analyse der Ausfallsicherheit, Genauigkeit und reale Umsetzbarkeit des Systems werden zwei Fahrzeugmodelle herangezogen.

Die Arbeit beschäftigt sich zunächst mit dem eingesetzten Time-Triggered Protocol (TTP, siehe 3.3.5) und der DECOS-Architektur. Im Anschluss daran wird das Fahrzeugmodell und die Hardware genauer beschrieben und analysiert.

DECOS - Dependable Embedded Components and Systems

Ein typischer Aufbau von Fahrzeugelektronik ist, dass genau eine Funktion (z.B. Steer-by-Wire, Brake-by-Wire, elektrische Sitzverstellung) auf einer ECU implementiert wird. Bei „Distributed Embedded Computern", wie dem DECOS-System, werden verschiedene Funktionen von einer ECU ausgeführt. Jede Funktionssoftware kann unabhängig von der anderen entwickelt und designt werden, wie beim typischen Fahrzeugelektronikaufbau. Einzig die vorhandenen Systemressourcen (CPU, Speicher, I/O) und Kommunikationskanäle (Bus) werden unter den Softwarepaketen geteilt.

Die DECOS Architektur ermöglicht somit den Einsatz verschiedenster Software (sicherheitsrelevant und nicht sicherheitsrelevant) von unterschiedlichen Programmierern auf demselben System zu laufen ohne sich gegenseitig zu beeinflussen.
Die einzelnen Softwarepakete werden in sogenannten „Distributed Application Subsystems (DAS)" im gesamten DECOS Netzwerk verteilt abgearbeitet. Um dies zu ermöglichen implementiert das DECOS System fünf „high level services"

1. **Virtual Network (VNET)** stellt time-triggered und event-triggered Kommunikation zu Verfügung
2. **Encapsulated Execution Environment (EEE)** ermöglicht unabhängige Prozessabarbeitung
3. **Gateway (GW)** ermöglicht Informationsaustausch zwischen den VNETs

4. **Fault Tolerance Layer (FTOL)** übernimmt das Management der redundanten Funktionen

5. **Diagnosis Service (DIAG)** ermöglicht Unterscheidung zwischen Hardware-, Software-, transienten, permanenten und intermittierenden Fehlern

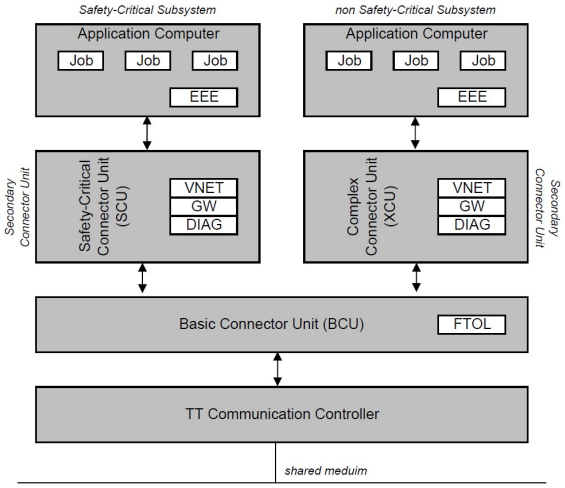


Abbildung 2.6: Aufbau einer DECOS-Komponente aus [10]

Bild 2.6 zeigt den Aufbau einer DECOS-Komponente. Nachrichten werden über das Kommunikationsinterface an die Basic Connector Unit (BCU) übergeben. Diese entscheidet, ob es sich um sicherheitsrelevante oder unkritischere Nachrichten handelt und verteilt diese an die jeweilige Secondary Connector Unit. Bei dieser Secondary Connector Unit handelt es sich entweder um eine Safety-Critical Connector Unit (SCU) oder eine Complex Connector Unit (XCU). SCU und XCU übertragen die Informationen über die jeweiligen VNET Ports an die verteilten ECUs. SCU implementiert nur time-triggered Services, während bei der XCU auch event-triggered Messages übertragen werden können.

Fahrzeugmodell 1

Bei diesem Fahrzeugmodell sind die Wheel Controller nicht direkt mit den Aktuatoren und Sensoren verbunden. Eine Realisierung dieses Modells im realen Fahrzeug hat den Vorteil einer einfachen Verteilung der Wheel Controller Software auf beliebige ECUs, allerdings muss mit höheren Delayzeiten bei der Kommunikation mit den Sensoren und Aktuatoren gerechnet werden. Weiters konnten bei diesem Fahrzeugmodell nicht alle Softwarefunktionen im DECOS System realisiert werden. Die Softwarefunktionen, die im Bild 2.7 mit schwarzem Hintergrund gekennzeichnet sind, mussten ausgelagert werden.

Fahrzeugmodell 2

Fahrzeugmodell 2 dieses Versuchs löst die Probleme, die sich bei Fahrzeugmodell 1 ergeben haben, hat allerdings in einem realen Fahrzeug den Nachteil des höheren Kabelaufwands und fordert, dass die Wheel Controller Software auf ECUs implementiert wird, die mit den einzelnen Radsensoren und Radaktuatoren direkt verbunden sind.

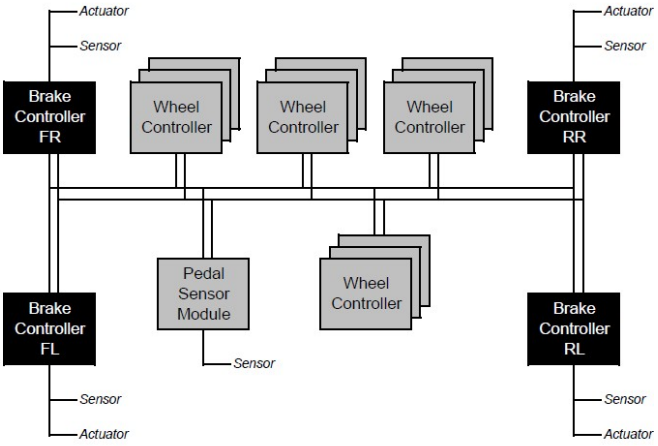


Abbildung 2.7: DECOS Fahrzeugmodell 1 aus [10]

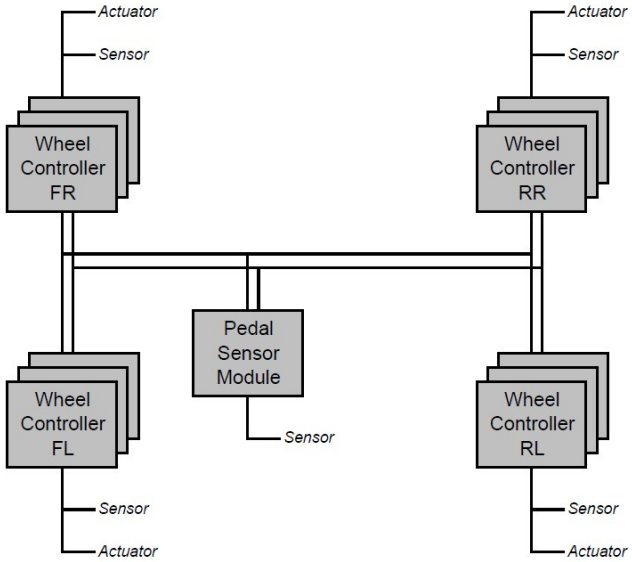


Abbildung 2.8: DECOS Fahrzeugmodell 2 aus [10]

Ergebnisse

Die Versuche zeigen, dass das DECOS System gut funktioniert und das DAS im Fehlerfall richtig reagiert hat, obwohl noch nicht alle „high level services" der DECOS Architektur vollständig implementiert wurden.
Fahrzeugmodell 1 stellt ein sehr dynamisches Brake-by-Wire Modell dar, benötigt aber kurze Responszeiten bei der Übertragung der Sensorsignale.
Fahrzeugmodell 2 bessert diesen Nachteil aus, was aber auf Kosten der Implementierungsfreiheit geht und zu höherem Kabelaufwand führt. Für die Implementierung dieses Fahrzeugmodells in einem realen Fahrzeug würden 12 ECUs benötigt werden.

2.6.3 The X-by-Wire Concept: Time-Triggered Information Exchange and Fail Silence Support by new System Services [9]

Auch dieses Paper präsentiert ein konzeptuelles Modell und Mechanismen der Softwarentwicklung für X-by-Wire Systeme. Ähnlich wie bei der Arbeit [5] soll ebenfalls ein Framework für sicherheitsrelevante und fehlertolerante elektrische X-by-Wire Systeme ohne mechanische Rückfallebene das Ziel der Arbeit sein. Das eingesetzte Übertragungsmedium ist ebenfalls ein TTP/C Bus (siehe 3.3.5), auch in dieser Arbeit wird ein Prototyp implementiert.
Näher eingegangen wird in diesem Paper vor allem auf die Anforderungen an ein Class C Echtzeit-Kommunikationssystem (z.B. TTP/C).

Anforderungen an ein Class C Echtzeit-Kommunikationssystem

Regelmäßigkeit des Informationstransfers Die meisten sicherheitsrelevanten Applikationen sind kontrollflussorientiert. Sie laufen in einer Endlosschleife und arbeiten ihre Tasks, bestehend aus Einlesen von Signalen, Verarbeiten dieser und Berechnen eines Outputs, zyklisch ab. Diese Informationen werden somit in regelmäßigen Abständen über das Kommunikationssystem übertragen. Neben diesen regelmäßigen Übertragungen muss ein Echtzeit-Kommunikationssystem auch noch die Fähigkeit besitzen, unregelmäßige größere Datenlasten zu übertragen (z.B. bei einem Fehlerfall).

Minimale Latenzzeit Als Latenzzeit einer Übertragung wird jene Zeit bezeichnet, die zwischen dem Senden der Nachricht und dem Empfangen der Nachricht vergeht. Die maximal gewünschte Latenzzeit ist von der Applikation abhängig, aber kann mit unter $1ms$ angenommen werden. Die Latenzzeit eines Kommunikationsmediums wird nicht zuletzt durch die Bandbreite und logische Struktur des Protokolls bestimmt. Eine Minimierung und Konstanthaltung der Latenzzeit ist damit von großer Wichtigkeit.

Fehlertoleranz Bei sicherheitsrelevanten Systemen ist ein Erkennen eines Fehlers nicht ausreichend, es muss ein eindeutiger fehlersicherer Zustand erreicht werden. Dies kann nur mittels eines fehlertoleranten Kommunikationsmediums erreicht werden.

Robustheit Das Kommunikationsmedium muss gegenüber elektro-magnetischen Störungen möglichst resistent sein.

Fehlererkennung Fehlerhalfte Busteilnehmer dürfen nicht aufgrund ihres Fehlverhaltens die gesamte Buskommunikation zum Erliegen bringen.

Acknowledgment Für den Sender einer Nachricht ist es zumeist sehr wichtig zu wissen ob diese auch fehlerfrei den Empfänger erreicht hat. Ein spezielles Acknowledgment Schema muss daher im Kommunikationssystem implementiert sein.

Testbarkeit Ein Echtzeit-System soll in seine Einzelsysteme zerlegt testbar sein. Integration neuer Einzelsysteme soll nicht das Verhalten des Gesamtsystems ändern.

2.6.4 Redundancy Management for Drive-by-Wire Computer Systems [27]

Das in dieser Arbeit vorgestellte Drive-by-Wire System basiert auf kommerziell verfügbaren Komponenten, die für den Automobilbereich entwickelt wurden und bereits im Einsatz sind. Die vorgestellte Implementierung besteht aus vier ECUs, die über einen CAN Bus miteinander zu einem „Duo Duplex" System verbunden sind. Bild 2.9 zeigt das Blockschaltbild einer solchen Duo Duplex Struktur. In dieser Struktur arbeiten jeweils zwei ECUs in zwei fail silent Duplex Kanälen. Jede ECU erhält laufend die gleichen Inputdaten wie die Andere und überprüft gleichzeitig den Output seines Gegenübers. Im Falle eines unterschiedlichen Outputs oder eines anderen Fehlers wird auf den anderen Duplex-Kanal umgeschaltet.

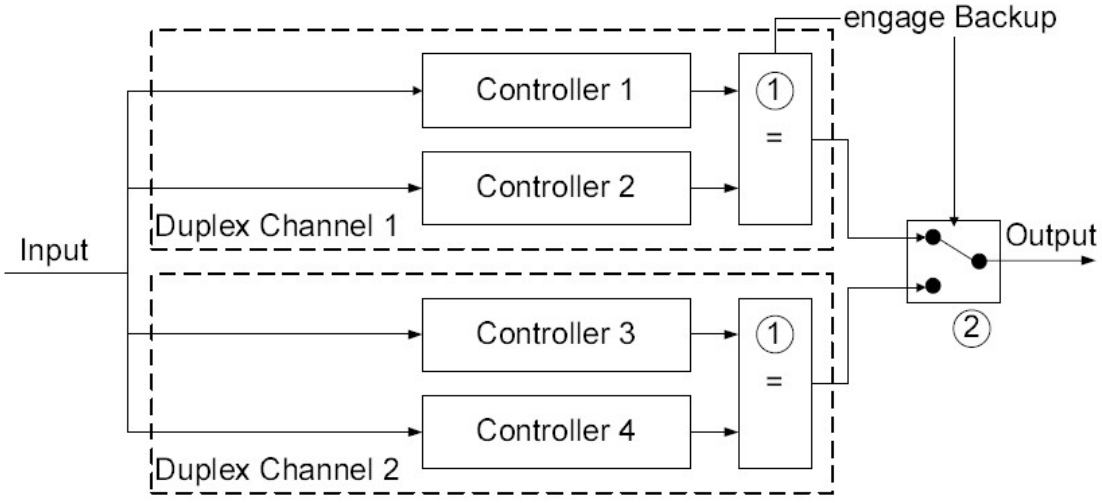


Abbildung 2.9: Blockschaltbild einer Duo Duplex Struktur aus [27]

Ein sehr interessanter Aspekt dieser Arbeit ist, dass auf Komponenten und Systeme aufgebaut wird, die auch für Privatkunden leicht erhältlich und erschwinglich sind.
Weiter wird festgestellt, dass der eingesetzte CAN-Bus (3.3.2) bei geeigneter Wahl der

Nachrichtenpriorität und Buslasten von $10 - 20\%$ als quasi deterministisch angenommen werden kann und damit allen Anforderungen an einen Echtzeitbus entspricht.

Des Weiteren wird speziell der Einsatz von Software zur automatischen C-Code Generierung empfohlen (z.B. dSpace TargetLink oder Simulink), was zur Steigerung der Softwarequalität und gleichzeitige Reduktion der zeitintensiven Tests im Fahrzeug führt. Ein weiterer Vorteil der automatischen Codegenerierung ist auch die einfache Portierbarkeit auf andere Systeme.

2.6.5 Delphi Electronic Throttle Control Systems for Model Year 2000; Driver Features, System Security, and OEM Benefits. ETC for the Mass Market [23]

Das hier vorgestellte elektronische Gaspedal ist eine Entwicklung der Firma Delphi und in einigen SUV Modellen des Baujahrs 2000 im Einsatz.
Das Systems besteht grundsätzlich aus einem einzelnen Microcontroller, der die Bearbeitung der Sensorsignale am Gaspedal übernimmt. Aus Redundanzgründen ist dieser Controller jedoch ein zweites Mal parallel dazu aufgebaut. Das Gaspedal ist, je nach Ausführungswunsch, mit 2 oder 3 redundant arbeitenden Sensoren bestückt (siehe Blockschaltbild 2.10). An der Drosselklappe sind zwei redundante Sensoren, ein Gleichstrommotor und eine Rückstellfeder für die Sicherung der „limp-home" Funktionalität verbaut. Zur Erhöhung der Ausfallsicherheit sind zusätzliche Referenzspannungsversorgungen und Kurzschlusssicherungsschaltungen integriert, die sicherstellen, dass ein defekter Sensor oder Kabelbruch an einem Sensor nicht zu einem Ausfall des Gesamtsystems führen kann.

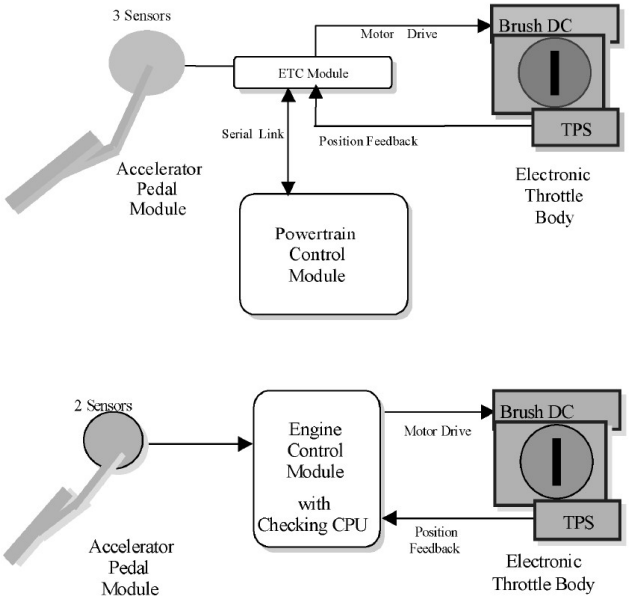


Abbildung 2.10: Blockschaltbild Delphi Electronic Throttle Control System

2.7 Zum Fachgebiet passende Veröffentlichungen

Die hier vorgestellten Veröffentlichungen gelten zwar nicht unbedingt als wissenschaftliche Quellen, waren allerdings sehr informativ zu Beginn der Implementierungsphase und wurden im Laufe der Arbeit zur Klärung gewisser Entscheidungen herangezogen.

2.7.1 Dual Core Microcontrollers

Im Jänner 2010 stellte Texas Instruments einen auf zwei ARM Cores basierenden Dual Core Microcontroller vor. Der TMS570 ist ein in Kooperation mit der Robert Bosch Gmbh entwickelter, speziell für den Einsatz im Automobilbereich designter Microcontroller, der die IEC 61508 SIL3 Norm [11] erfüllt.

In einer weiteren Veröffentlichung im EE Times [35] werden auch noch weitere Dual Core MCUs vorgestellt. Der DCIC9907 von DualCore verfügt über zwei integrierte CAN Schnittstellen und zwei real-time fähige Ethernet MACs.
Der Dual Core MCU der Firma Freescale, der MPC5564xL, arbeitet mit einer 32 bit Power Architektur und erfüllt neben IEC 61508 auch die neue ISO 26262 Norm [15].
Dank dieser Dual Core Architektur ist ein Duplizieren der Komponenten auf Systemebene nicht mehr notwendig und es kann mehr Augenmerk auf die eigentliche Implementation der Hard- und Softwareapplikationen gelegt werden.
Diese Microcontroller verfügen über eine Betriebsart, genannt „Lockstep-Modus", indem sich die beiden CPU Kerne gegenseitig überwachen und somit redundant arbeiten. Auf diesen Microcontrollern sind alle Schlüsselkomponenten redundant ausgeführt und verfügen über eine automatische Hardwarediagnosefunktion. Damit werden die Anforderungen der anspruchsvollsten ASIL- Klasse ohne zusätzliche Hard- oder Software erfüllt [15].

Kapitel 3

Lösungsansätze

In diesem Kapitel wird ein Überblick über mögliche Lösungsansätze und Implementierungsvarianten gegeben.
Abschnitt 3.1 gibt einen Überblick über Microcontroller, die für den Einsatz im Automobil entwickelt wurden und zeigt Besonderheiten, Vor- und Nachteile auf.
Abschnitt 3.2 erklärt verschiedene Messprinzipien und Sensoren, die für den Einsatz in X-by-Wire Systemen in Frage kommen, bevor Abschnitt 3.3 die verschiedenen Busse im Automobil beleuchtet.
Die letzten Abschnitte 3.4.1, 3.4.2 und 3.4.3 behandeln das Thema Fehlertoleranzverfahren und Ausfallsicherheit für Software- und Hardware-Systeme.

3.1 Spezielle Microcontroller für den automotiven Bereich

Wie schon unter 2.7.1 erwähnt gibt es speziell für den X-by-Wire Einsatz in Automobilen entwickelte Mikrocontroller. Die Tabelle C.2 gibt einen kurzen Überblick über die ersten dieser Dual Core MCUs und einige ihrer speziellen Features.
Neben der Dual Core Ausführung haben alle Mikrocontroller gemeinsam, dass sie noch relativ neu und damit selten eingesetzt sind. Dies schlägt natürlich auf die Kosten bei geringen Stückzahlen. Ein weiterer negativer Aspekt erst kürzlich entwickelter Mikrocontroller ist auch der Support an Software und Entwicklungsumgebungen und die geringen Erfahrungswerte mit „speziellen" Verhaltensmustern und Bugs.

3.2 Sensorik und Messprinzip zur Fahrerwunschaufnahme

Sensoren lassen sich vorab grob nach der aufgenommenen Messgröße einteilen:

- geometrisch
- mechanisch
- thermisch
- optisch
- akustisch
- chemisch
- elektrisch
- usw.

In diesem Dokument wird nur auf Sensoren zur Aufnahme geometrischer Messgrößen, speziell nur Sensoren zur Aufnahme von Länge und Winkel, eingegangen. Für weitere Informationen zum Thema Sensoren wird das Buch [21] empfohlen.

Sowohl bei Längen- oder Wegsensoren als auch bei Winkelsensoren werden überwiegend die Prinzipien potentiometrisch und induktiv für analoge Signale und inkremental oder absolut kodiert für digitale Signale verwendet.
Bild 3.1 zeigt eine symbolhafte Darstellung der verschiedenen Prinzipien.

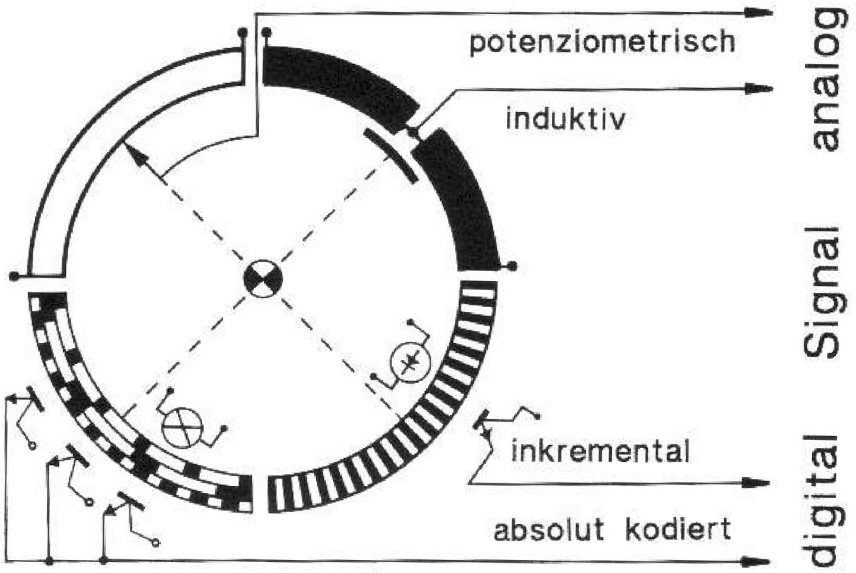


Abbildung 3.1: symbolische Darstellung der verschiedenen Längen oder Winkel Messprinzipien aus [21]

3.2.1 Potentiometrisches Messprinzip

Dieses Prinzip ist das wohl bekannteste und weit verbreitetste unter den Messprinzipien. Potentiometrische Längensensoren werden üblicherweise in einer Spannungsteilerschaltung betrieben. Die längenanaloge Ausgangsspannung des Sensors U_A ist als Funktion der Schleiferstellung s gegeben.

$$U_A = f(s) \tag{3.1}$$

Aktuell eingesetzte Potentiometer werden nicht mehr drahtgewickelt, sondern aus Leitplastik gefertigt. Diese sehr weit verbreiteten Sensoren verfügen über eine sehr hohe Lebensdauer und sind in sehr unterschiedlichen Ausführungsarten verfügbar. Einer der größten Vorteile liegt im einfachen Aufbau, ohne komplizierte Elektronik, was die Sensoren sehr preiswert und einfach anwendbar macht. Ein weiterer positiver Faktor ist, dass die Auflösung des Sensors nicht über den Sensor selbst, sondern nur von der nachgeschalteten, anwendungspezifischen Elektronik abhängig ist. Außerdem sind Potentiometer mechanisch präzise und robust fertigbar und können oft kundenspezifisch ausgelegt werden.

3.2.2 Induktives Messprinzip

Induktive Längensensoren unterscheiden sich von den potentiometrischen äußerlich nur unwesentlich, sie sind ebenso robust aufgebaut, arbeiten allerdings berührungslos und daher verschleißfrei.
Im Inneren des Sensors findet man jedoch ein wesentlich anderes Messprinzip (siehe Bild 3.2). Anstelle der Widerstandsbahn werden hier zwei Spulen als eine induktive Halbbrücke eingesetzt. Statt eines Schleifers wird ein Metall- oder Ferritkern bewegt. Die induktive Halbbrücke wird mit Wechselstrom gespeist und ändert die Ausgangsspannung in Abhängigkeit von der Bewegung des Metall- oder Ferritkerns. Zu einem induktiven Sensor gehört immer eine Auswerteschaltung, die sich entsprechend der jeweiligen Ausführung entweder innerhalb des Sensorgehäuses oder in einem separaten Baustein befindet.

Induktive Sensoren haben eine hohe Auflösung und eignen sich, aufgrund ihrer hohen Linearitätsklasse, gut für anspruchsvolle Messaufgaben. Die Auswerteschaltung kann wahlweise Spannungs- oder Stromsignale bereitstellen und kann in der Empfindlichkeit und der Nullpunkteinstellung justiert werden.

3.2.3 Inkremental-Sensoren

Vereinfacht dargestellt bestehen Inkremental-Sensoren aus einer Codescheibe mit regelmäßig verteilten Schlitzen und einem Impulsgeber. Abhängig von der Auflösung wird pro Umdrehung oder Längendifferenz eine bestimmte Anzahl von Impulsen abgeben. Diese Impulse sind zu zählen und in einen Winkel oder eine Länge umzurechnen (siehe Bild 3.1). Für eine Richtungserkennung ist oft eine zweite versetzte Codescheibe angeordnet. Inkrementelle Sensoren besitzen keinen definierten Bezugspunkt oder Nulllage.

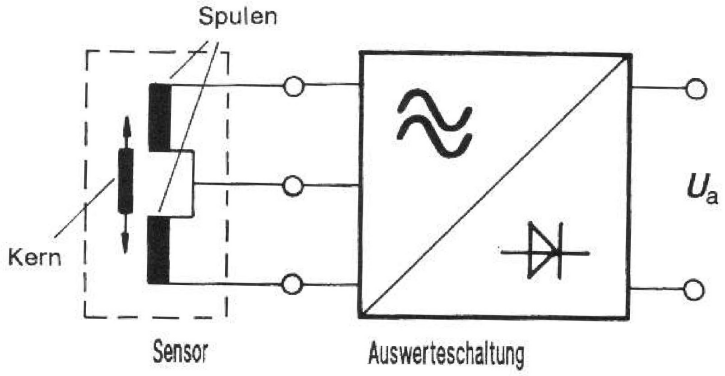


Abbildung 3.2: Schaltungsprinzip induktiver Sensoren aus [21]

3.2.4 Absolut kodierte Sensoren

Vom Prinzip her gleich dem Inkremental-Sensor sind absolut kodierte Sensoren. Bezüglich der Drehrichtungserkennung sind diese jedoch anspruchsloser, da aus dem aktuellen Codesignal bereits die aktuelle absolute Position abgeleitet werden kann. Meist werden für diese Codesignale einfache Tabellencodes (z.B. BCD, Binär-Code, Gray-Code) eingesetzt.

3.2.5 Magnetostriktives Messprinzip

Magnetostriktive Sensoren sind hochgenaue Sensoren, die meist in extremen Umgebungsbedingungen eingesetzt werden. Sie zeichnen sich durch die integrierte Messwertüberwachung, ein hohes Maß an Sicherheit und Geschwindigkeit der Datenübertragung und sehr hohe Schock- und Vibrationsresistenz aus. Diese Sensoren arbeiten nach dem Prinzip der Magnetostriktion.

Die Magnetostriktion ist die Deformation ferromagnetischer Stoffe infolge eines magnetischen Feldes. Beim Anlegen eines magnetischen Feldes an einen Ferromagneten richten sich die Weiss'schen Bezirke gleich aus und verändern damit die Länge des Materials (im Bereich von ca. $2mm/m$ bei hochmagnetostriktiven Werkstoffen).

Im Wesentlichen ist ein solcher Sensor aus drei Einheiten: Wellenleiter, Positionsgeber und Auswerteelektronik aufgebaut (siehe Bild 3.3).
Eingeleitet wird der Messvorgang durch einen kurzen Stromimpuls im Wellenleiter. Dieser Impuls erzeugt ein zirkulares Magnetfeld, das senkrecht zu den Feldlinien des Positionsgebers steht. An der Überlagerungsstelle der Magnetfelder entsteht im Wellenleiter eine elastische Verformung. Diese Verformung löst einen mechanischen Impuls aus, der sich im Wellenleiter fortpflanzt. Diese Torsionswelle wird am Ende des Wellenleiters in ein elektrisches Signal umgesetzt. Die Laufzeit vom Entstehungsort bis zum Signalwandler ist dem Abstand zwischen Signalwandler und Positionsgeber direkt proportional und gibt somit die Messposition wieder.

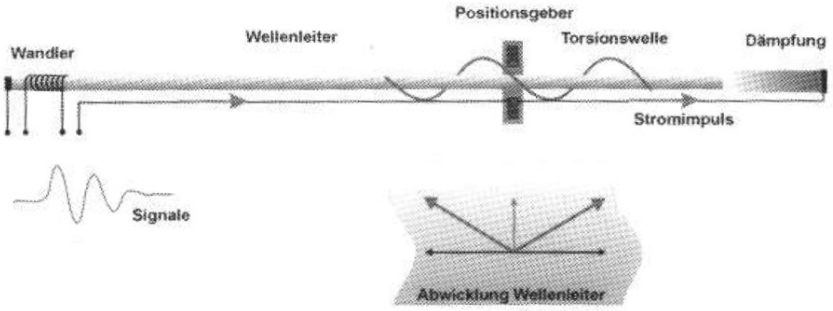


Abbildung 3.3: Prinzipschaltbild eines magnetostriktiven Längensensors aus [21]

3.2.6 Hall-Element Sensoren

Hall-Effekt- oder Hall-Element-Sensoren basieren auf dem Hall-Prinzip. Dieses Prinzip besagt, dass bei einem stromdurchflossenen Leiter in einem stationären Magnetfeld eine Spannung, senkrecht zur Stromfluss- und auch Magnetfeldrichtung, abfällt. Diese Spannung wird Hall- Spannung U_H genannt und lässt sich mit folgender Gleichung berechnen:

$$U_H = A_H \frac{IB}{d} \tag{3.2}$$

Diese Spannung ist proportional der magnetischen Feldstärke B, wodurch durch Anbringen eines Positionsmagneten auf einer drehbaren Welle auf einfachste Weise eine berührungslose Winkelmessung machbar ist. Durch Kombination mehrerer dieser Sensorelemente und Integration der kompletten Signalverarbeitung auf kleinstem Bauraum ist ein hoch auflösender Sensor mit guter Dynamik, einem Messbereich über 360° und großer mechanischer Toleranz verwirklichbar.
A_H ist eine Materialkonstante, auch Hall-Koeffizient oder Hall-Konstante genannt. Materialien mit hoher Empfindlichkeit sind durch eine große Hall-Konstante gekennzeichnet und zumeist Halbleitermaterialien, was einer Massenfertigung in CMOS Technologie entgegenkommt. Weiters kann durch Messen der Hall-Konstante eine Bestimmung der Ladungsträgerdichte eines Materials durchgeführt werden.
Ist der Strom bekannt, kann die magnetische Feldstärke B gemessen werden, wodurch sich Hall-Sensoren für die Magnetfeldmessung eignen.
Wird das Magnetfeld durch eine Spule erzeugt, kann man potentialfrei die Stromstärke I in der Spule messen.
Weiters kann auch über das Hall-Prinzip die Materialschichtdicke d bestimmt werden.
In der Automobilindustrie finden Hall-Sensoren vielfältige Anwendung (z. B. Gurtschloss, Türschloss, Fahrpedal, Bestimmung des Zündzeitpunkts). Hauptvorteil hierbei ist die Unempfindlichkeit gegen Schmutz, Wasser und die schnelle Machbarkeit kundenspezifischer Sonderlösungen.

3.3 Übertragungsmedien und Bussysteme

Im Automobilbereich hat man sich sehr rasch der gängigen Praxis bedient, Kosten und Gewicht für Verkabelung zu sparen und verschiedenste Datenbusse zur Anwendung gebracht. Ohne diese Bussysteme wäre die komplexe Elektronik eines aktuellen Fahrzeugs nicht implementierbar. Im VW Phaeton sind trotz mehrfacher Datenbussysteme $3860m$ Kabel in einem $64kg$ schweren Kabelbaum verbaut [18].
Die hier vorgestellten Busse geben einen Überblick über die wichtigsten Bussyteme im Automobilbereich, deren Anwendungsgebiete und ihre Vor- und Nachteile. Als weiterführende Literatur zu diesem Themengebiet seien hier [3] und [38] empfohlen.

Die Vernetzung im Fahrzeug lässt sich in folgende Kategorien fassen (siehe auch Bild 3.4):

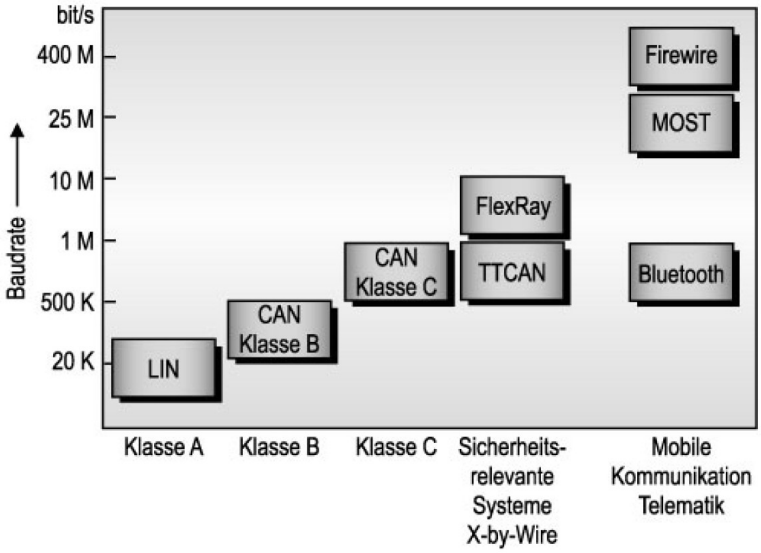


Abbildung 3.4: Kategorisierung der automotiven Bussysteme aus [17]

Klasse A Diese Klasse der Bussysteme wird eigentlich nur für Smart Senors und Smart Aktuators verwendet. Anwendungsbeispiele für Busse dieser Klasse sind Reifendrucksensoren oder auch Regensensoren. Diese Busse kennzeichnen sich durch geringe Kosten und Verkabelungsaufwand sowie niedrige Datenrate. Der bekannteste Vertreter dieser Klasse ist der LIN-Bus (siehe 3.3.1).

Klasse B Busse, die zu dieser Gruppe zählen, werden auch gerne unter Karosseriekommunikationsysteme zusammengefasst. Dies beinhaltet Kommunikation der Systeme, wie Sitzeinstellung, Spiegelverstellung, Klimaanlagen-Sensorik und Zentralverriegelung. Der bekannteste Vertreter dieser Klasse ist der Karosserie CAN (K-CAN), ein CAN Bus (siehe 3.3.2) mit reduzierter Datenrate (125 $Kbit/s$).

Klasse C In die Klasse C fallen Vernetzungsysteme zeitkritischer Funktionen, wie Motor-Management, Regelung der Schadstoffemission und des Antriebsstrangs, sowie auch sicherheitsrelevante Kommunikationsbusse, wie bei X-by-Wire Systemen, ABS, ESP und sonstigen Fahrassistenzprogrammen. Bussysteme dieser Klasse müssen den Anforderungen nach ISO 26262 ASIL D genügen.
An diese Busse werden hohe Anforderungen bezüglich vorhersagbarem Verhalten (hard real time), Fehlertoleranz, guter Wart- und Testbarkeit sowie Unterstützung von verteilten Systemen gestellt. Wichtigste Vertreter dieser Kategorie sind CAN (siehe 3.3.2), Flex Ray (siehe 3.3.4), TTP/C Bus (siehe 3.3.5), TTCAN (siehe 3.3.3).

Multimedia-Busse Mit zunehmender Integration multimedialer Systeme im Automobil sind auch vermehrt Multimedia-Busse im Fahrzeug integriert. Die Anforderungen an diese Kategorie sind verstärkt hohe Bandbreite und konstante Verzögerung während der Datenübertragung. Bekannteste im Fahrzeug eingesetzte Multimedia-Busse sind MOST-Bus (siehe 3.3.6), D2B, FireWire und natürlich auch USB.

In der folgenden Tabelle 3.1 soll zusammengefasst ein Überblick gegeben werden, wodurch sich die Anwendungsgebiete und Anforderungen an die verschiedenen Bussysteme unterscheiden.

Anwendung	**Länge der Botschaften**	**Wiederholungsrate der Botschaften**	**Datenrate**	**Latenzzeit**	**Fehlersicherheit**	**Kosten**
Klasse A	kurz	gering	gering	geringe Anforderung	niedrig	sehr niedrig
Klasse B	kurz	gering	gering	mäßig	hoch	gering
Klasse C	kurz	hoch	hoch	sehr kurz	extrem hoch	mittel
Multimediale Busse	lang	hoch	sehr hoch	mäßig	mäßig	hoch

Tabelle 3.1: Anwendungsbereiche und Anforderungen an automotive Bussysteme

3.3.1 LIN-Bus

LIN steht für „local interconnection network" und ist ein sehr billig zu realisierender, serieller Bus. Typischerweise findet er Anwendung in kleinen Teilnetzwerken von maximal 16 Knoten, bei denen der Ausfall weniger Daten nicht zu Schäden führt. Bild 3.5 zeigt die vereinfacht realisierte physikalische Schicht. Der Bus ist ein Eindraht-Bus und kann mit einer maximalen Datenrate von 20 $kbit/s$ betrieben werden. Übliche Datenraten sind allerdings 2400 bit/s, 9600 bit/s oder 19200 bit/s.

Im Zuge der Spezifikation des LIN-Busses 1999 wurden nicht nur Hardware und Software genormt, sondern auch Hilfsmittel zur Entwicklung eines LIN-Netzwerkes. Durch den Einsatz von Transceivern ist der LIN-Bus jedoch nicht mehr wesentlich günstiger als z.B. der CAN Bus, bietet aber weiterhin die Einsparungsmöglichkeit einer zweiten Busleitung und ist immer noch in modernen Automobilen verbreitet im Einsatz. Das LIN-Protokoll ist zum Protokoll der seriellen PC Schnittstelle RS232C kompatibel und lässt sich damit

so einfach wie kein anderer automobiler Bus ohne zusätzlichen Controller an einen PC anbinden.

Bild 3.6 zeigt den Ablauf einer LIN-Kommunikation. LIN-Kommunikationen sind immer Master-Slave Kommunikationen, d.h. jede Botschaft besteht aus einer Anfrage des Masters, an welche die Antwort des Slaves unmittelbar angefügt wird.

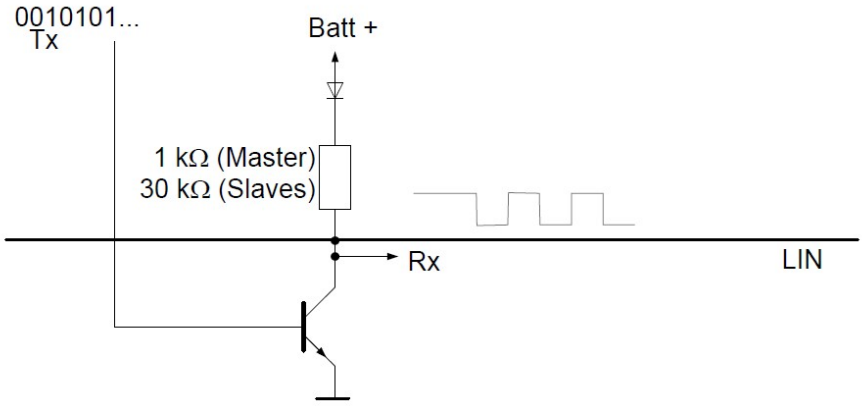


Abbildung 3.5: vereinfachter LIN-Bus Transceiver aus [3]

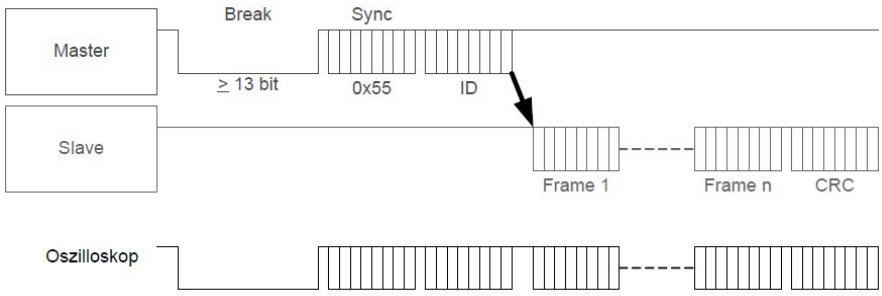


Abbildung 3.6: LIN-Kommunikation aus [3]

3.3.2 CAN-Bus

Der Controller Area Network Bus oder auch CAN-Bus genannte Bus war das erste digitale Bussystem, das sich im Fahrzeug durchsetzte. Entwickelt wurde dieser Bus 1981 von Intel und Bosch und wurde später von der ISO genormt. Dieser Bus findet mittlerweile Anwendung im Automobil-, Industrie- und Automationsbereich. Mittlerweile existieren alternative Implementierungen, stark abweichende physikalische Ausführungen (z.B. Single Wire CAN [J2411]) und viele darauf aufgesetzte Protokollschichten (u.a. J1939 oder Bosch MCNet, CANopen, TTCAN).
Eigenschaften, die der CAN-Bus aufweist:

- differentieller Multimaster Bus
- Bitrate bis 1 $Mbit/s$
- Nachrichten-orientiertes ereignisgesteuertes Protokoll
- hohe Übertragungssicherheit und Datenkonsistenz
- Carrier Sense Multiple Access / Collision Resolution Verfahren (CSMA/CR)
- lineare Topologie oder Sterntopologie

Der CAN-Bus ist die Standard-Netzwerktechnologie des Automobilsektors und nach wie vor der am verbreitest eingesetzte Bus im Fahrzeug. CAN-Controller sind häufig ohnehin in fast allen Mikrocontrollern integriert und somit ist der CAN-Bus ein sehr günstig aufzubauender Bus.

Physikalische CAN-Schicht

Das Übertragungsmedium des CAN-Busses sind zwei verdrillte Kupferleitungen. Die Verdrillung stellt einen sinnvollen Kompromiss zwischen Störfestigkeit und Kosten dar.

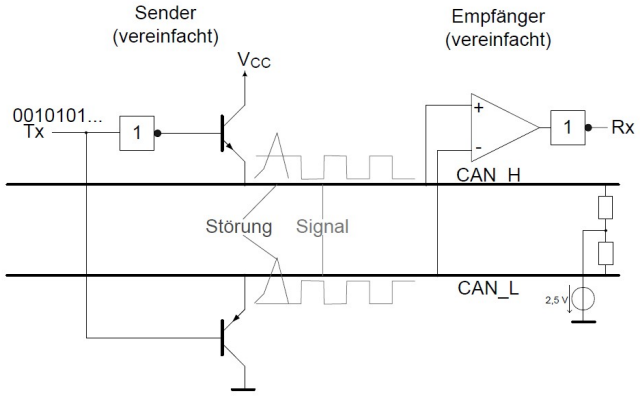


Abbildung 3.7: vereinfachte Darstellung eines CAN-Transceivers aus [3]

In Bild 3.7 wird eine vereinfachte CAN-Transceiver Schaltung dargestellt. Diese Transceiver sind meist in 8-beinige ICs integriert und sind rein für die elektrische Ankopplung zuständig.
Wie aus diesem Bild ersichtlich ist, wird beim Senden einer logischen 1 der Tx- Eingang des Transceivers mit einer Spannung von 5 V angesteuert. Die beiden Transistoren sperren und der Bus hält seine Ruhespannung von $2,5$ V auf beiden Leitungen. Die Spannungsdifferenz der beiden Busleitungen ist 0 V.
Beim Senden einer logischen 0 werden 0 V am Tx- Eingang angelegt, die Spannung an „CAN high" wird erhöht, die Spannung an „CAN low" gesenkt und es resultiert eine Spannungsdifferenz an den beiden Busleitungen.

Der Vorteil dieses Verfahrens liegt beim Auftreten von Störungen. Bei verdrillten Leitungen kann davon ausgegangen werden, dass sich Störungen auf beide Signale etwa gleich auswirken, was bei Differenzsignalen keine Beeinflussung des Signals zur Folge hat.
Stellt sich eine der beiden Leitungen als defekt heraus, kann diese abgeschaltet werden und die verbleibende Leitung gegen Masse weiter betrieben werden.
Weiters besteht die Möglichkeit, den Transceiver in einen stromsparenden Standby-Betrieb zu schalten und über bestimmte Botschaften wieder zu starten.

Die eigentliche Topologie des CAN-Busses ist eine lineare Topologie mit jeweils einem Abschlusswiderstand an den beiden Busenden. Da Leitungsverzweigungen im Kabelbaum allerdings aufwändig sind, hat sich in der Praxis die Sterntopologie als kostengünstige Lösung durchgesetzt. Die Abschlusswiderstände werden auch beim Stern an den beiden weitest entfernten Enden vorgesehen.

Aufbau der Datenpakete

Im CAN Standard sind 4 Arten von Datenpaketen spezifiziert:

- Daten-Frames
- Request-Frames
- Error-Frames
- Overload-Frames

Daten-Frames dienen zur Übertragung der Nutzdaten, Request-Frames beinhalten eine Anfrage nach einem bestimmten Daten-Frame. Diese beiden Datenpakete sind die einzigen Frame-Typen, die in einem störungsfrei arbeitenden CAN Netzwerk auftreten.
In der Praxis jedoch werden Request-Frames kaum eingesetzt, sondern Steuergeräte senden bestimmte Daten in regelmäßigen Zeitabständen, und das Netzwerk arbeitet nicht absolut störungsfrei.

Wird ein Fehler von einem Busteilnehmer erkannt, so teilt er dies den anderen Busteilnehmern über einen Error-Frame mit. Dies geschieht, indem der noch laufende Frame mit einer speziellen Bitsequenz überschrieben wird.

Der Overload-Frame signalisiert anderen Busteilnehmern mit der Übertragung von Daten zu warten, wenn ein Knoten für die Verarbeitung eingehender Nachrichten mehr Zeit benötigt. Dieser Frame ist, wie der Error-Frame, mit einer speziellen Bitsequenz gekennzeichnet, kann aber nur während der Pause zwischen zwei Nachrichten gesendet werden.

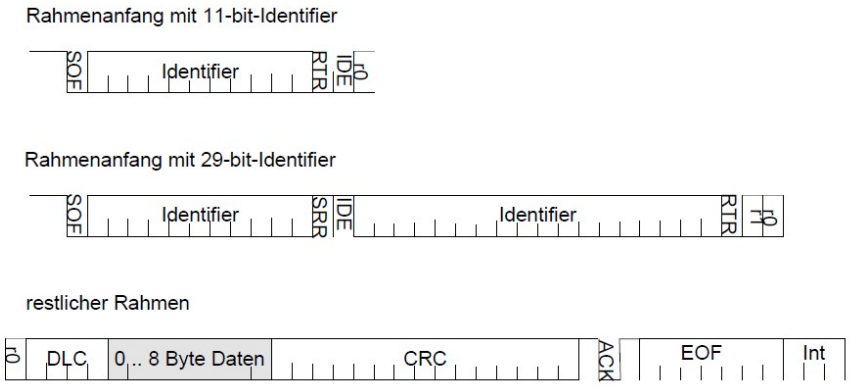


Abbildung 3.8: Aufbau eines CAN-Datenrahmens aus [3]

Solange keine Kommunikation am CAN geführt wird, behält der Bus seine Ruhespannung. Die Ruhespannung entspricht einer logischen 1, die von jedem Busteilnehmer überschrieben werden kann. Damit ist die logische 0 der dominante Pegel.
Bild 3.8 zeigt den Aufbau eines CAN-Datenrahmens. Zu Beginn einer Übertragung wird ein SOF (Start of Frame), eine logische 0, am Bus angelegt. Danach folgen 11 *bit* oder 29 *bit* Message Identifier, je nachdem welcher CAN Standard (2.0A oder 2.0B) eingesetzt wird. Beide Standards sind so entwickelt, dass sie gemeinsam an einem Bus verwendet werden können.
Der CAN Bus ist ein Nachrichten-orientierter Bus, Nachrichten tragen weder Sender- noch Empfängerinformationen und werden von allen Busteilnehmern empfangen. Über Akzeptanzfilter kann dann jeder Knoten entscheiden, ob er die jeweilige Nachricht weiterverarbeitet oder nicht.
Das RTR-Bit (Remote Transmission Request) kennzeichnet einen Request-Frame. Das IDE-Bit (IDentifier Extension) die Verlängerung des Identifiers auf 29 *bit*.
Da die Anzahl der Datenbytes eines Daten-Frames zwischen 0 und 8 *Byte* variieren kann, wird die Länge der übertragenen Nutzdaten im DLC (Data Length Code) angegeben.
Dem DLC folgen die Nutzdaten, der CRC (Cyclic Redundancy Check) und ein ACK-Bit (ACKnowledge), das den korrekten Empfang min. eines Busteilnehmers bestätigt.
Abgeschlossen wird die Kommunikation durch 7 rezessive EOF-Bits (End Of Frame) und einer Mindestpause (IFS, InterFrame Space) bis zum nächsten Kommunikationszyklus.

Beim CAN-Bus handelt es sich um einen Multi-Master Bus, bei dem alle Busteilnehmer gleichberechtigt zu jeder Zeit Daten über den freien Bus übertragen dürfen. Sollten nun gleichzeitig mehrere Busteilnehmer Daten zu senden beginnen, wird eine solche Kollision erkannt und über die Arbitrierung aufgelöst.

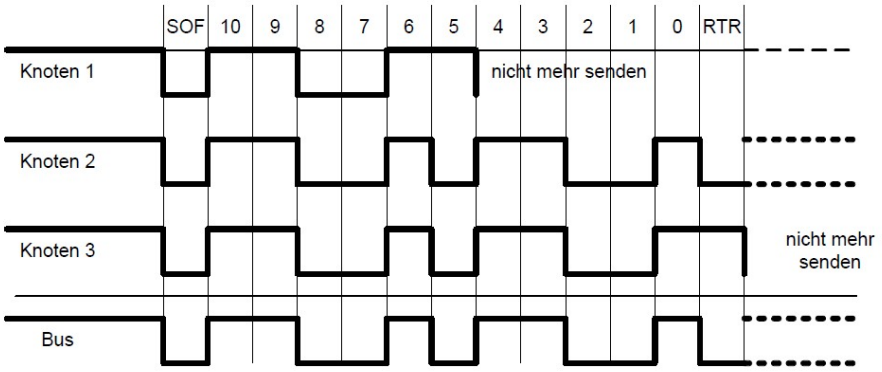


Abbildung 3.9: Arbitrierungsvorgang beim CAN Bus aus [3]

In Abbildung 3.9 sieht man, wie ein solcher Arbitrierungsvorgang abläuft. Jeder Knoten, der Daten übertragen will, überprüft gleichzeitig auch, ob der Bus frei ist und die gewünschten Bits am Bus übertragen werden. Stellt ein Knoten einen Unterschied zwischen seinem Übertragungswunsch und der Übertragung am Bus fest, hört er auf zu senden und beginnt die Kommunikation zu einem späteren Zeitpunkt erneut.
Aufgrund des dominanten 0 Pegels und der Arbitrierungsphase während des Übertragens des Message Identifiers haben Nachrichten mit niedriger ID höhere Priorität und werden ohne Unterbrechung Nachrichten mit niedrigerer Priorität vorgezogen. Damit ist laut [3]

> der CAN nicht für sicherheitsrelevante Anwendungen geeignet, in denen eine Nachricht auf keinen Fall durch andere Nachrichten unterdrückt werden darf.

Da die ereignisgesteuerte Kommunikation keine exakte Aussage über die Sendezeit trifft ist der CAN somit im Worst-Case nicht deterministisch.

Fehlererkennung

Der CAN-Bus wurde für eine hohe Zuverlässigkeit konzipiert und beinhaltet daher eine sehr komplexe und durchdachte Fehlerbehandlung. Ohne nähere Erläuterung, aber vollständig können diese Regelungen in der Dokumentation des Standards nachgelesen werden. Daher wird dieses Thema hier nicht sehr detailliert angeführt.
Nach der CAN Erkennungstrategie können fünf Fehlerarten unterschieden werden:

- Bitfehler
- Formatfehler
- CRC Fehler
- Stuffing Fehler
- fehlendes ACK vom Empfänger

Eine Besonderheit des CAN ist die Warnung eines Busteilnehmers an alle Knoten, sobald er einen Fehler erkennt. Damit nun ein einzelner defekter Knoten nicht den ganzen

Bus blockieren kann, enthält der CAN eine spezielle Strategie, um solche Knoten aus dem Datenverkehr zu entfernen.
Diese Strategie beinhaltet drei Zustände (siehe Bild 3.10), in denen jeder Busteilnehmer sein kann. Jeder Busteilnehmer besitzt zwei Zähler: einen für Sendefehler (TEC, Transmission Error Counter) und einen für Empfangsfehler (REC, Receive Error Counter). Je nach erkannter Fehlersituation werden diese beiden Zähler unterschiedlich inkrementiert und durch erfolgreich gesendete oder empfangene Nachrichten auch wieder dekrementiert. Dies geschieht nach den Regeln in Tabelle 3.2.
Die Zählerstände dieser beiden Counter geben nun an, ob ein Busteilnehmer vollständig aktiv am Bus teilnehmen darf (Error Active Zustand), nur mehr passive Error Frames übertragen darf und damit nur noch seine eigenen Nachrichten als fehlerhaft kennzeichnen kann (Error Passive), oder vollständig vom Bus getrennt wird (Bus Off).

Nachricht fehlerfrei empfangen	$REC = REC - 1$
Nachricht fehlerhaft empfangen	$REC = REC + 1$
Nachricht als Erster fehlerhaft empfangen	$REC = REC + 8$
Nachricht fehlerfrei gesendet	$TEC = TEC - 1$
Nachricht fehlerhaft gesendet	$TEC = TEC + 8$

Tabelle 3.2: Regeln zur Veränderung von TEC und REC

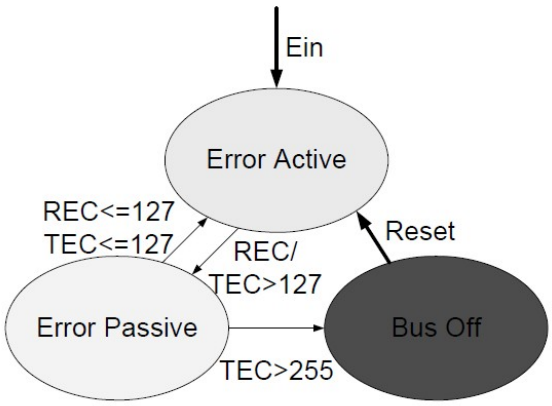


Abbildung 3.10: CAN Fehlerzustandsdiagramm aus [3]

Die Restfehlerwahrscheinlichkeit (Wahrscheinlichkeit, dass ein Fehler unerkannt bleibt) liegt laut [3] bei $4,7 \cdot 10^{-14}$, was bedeuten würde, dass unter realistischen Betriebsbedingungen, ca. alle 2000 Jahre mit einem unerkannten Fehler zu rechnen ist.

3.3.3 Time-Triggered CAN

BMW und Bosch begannen damit, sicherheitsrelevante Systeme mit elektronischer Übertragung zu entwickeln. Im Laufe dessen wurde ein zeitgesteuertes Bussysteme für automobile Anwendungen entwickelt. Dieser „Byteflight" genannte Bus ergänzt den CAN-Bus für typische sicherheitsrelevante Funktionen und ermöglicht eine Datenrate von 10 $Mbit/s$. Für diesen Bus wurde als physikalische Schicht ein Lichtwellenleiter vorgesehen.
Bosch erweiterte parallel dazu auch den CAN-Bus in ähnlicher Weise. Als Resultat entstand der TTCAN (Time-Triggered CAN). Ein Problem bei der Zuteilung von Zeitslots am CAN-Bus ist die Voraussetzung, dass alle Knoten über die gleiche Zeit verfügen. Das CAN Synchronisationsverfahren kann dies nicht mehr ausreichend gewährleisten. Dies führte zur Entwicklung zweier TTCAN Varianten. Eine Variante verzichtet dabei auf eine gemeinsame Zeitbasis, hat allerdings den Vorteil zum CAN Standard kompatibel zu sein. Bei dieser Ausführung werden feste Sendezeiten einfach mit vorhandener Hardware und einer dynamischen Vergabe von Identifiern realisiert.
Die andere Variante arbeitet mit einer gemeinsamen Systemzeit, stellt ein vollwertiges zeitgesteuertes System dar, ist allerdings nicht mehr voll kompatibel zum CAN Standard.

3.3.4 FlexRay

Der FlexRay Bus ist vom FlexRay Konsortium, einem Zusammenschluss der wichtigsten Firmen im Automobilsektor, in den Jahren 2000 bis 2009 entwickelt worden. Der Bus ist robust, skalierbar, deterministisch und fehlertolerant ausgeführt und gehört zu den seriellen digitalen Bussystemen.
Mit der Beendigung der Arbeit des Konsortiums wurde 2009 der aktuell gültige FlexRay Standard Version 3.0 veröffentlicht.
Dieser Bus ist speziell für die nächste Generation der X-by-Wire Anwendungen im Kraftfahrzeug vorgesehen und kann als Standard für diese Anwendungen angesehen werden.
Der Nachfolger des ByteFlight Busses setzt als physikalische Schicht wieder auf kostengünstigere Kupferleitungen mit differenzieller Übertragung und ermöglicht eine Datenrate von 10 $Mbit/s$. Eine Besonderheit ist, dass der Bus parallel geführt über eine zweite physikalische Verbindung verfügt. Diese kann wahlweise als Redundanz oder zur Verdoppelung der Datenrate auf 20 $Mbit/s$ genutzt werden. Die Busstruktur kann wahlweise linear oder auch sternförmig ausgeführt sein.

Jeder Kommunikationszyklus besteht aus einem statischen Segment, einem optionalen dynamischen Segment, einem optionalen Symbolfenster für businterne Zwecke und einer „Network Idle Time"(siehe Abbildung 3.11).

Eine Besonderheit des FlexRay Busses ist, dass jedes statische Segment in feste Zeitslots unterteilt wird, in die genau eine Nachricht passt. Diese Zeitslots bestehen aus einer festen Anzahl an Macroticks und müssen bei allen Busteilnehmern gleich zugeordnet sein. Aufgrund der unterschiedlichen Taktraten der einzelnen Busteilnehmer werden diese Macroticks in weitere Zeiteinheiten unterteilt, genannt Microticks. Diese Microticks und die Anzahl der Microticks per Macrotick können zwischen den einzelnen Busknoten variieren und werden beim regelmäßigen Abgleich der Buszeit angepasst.

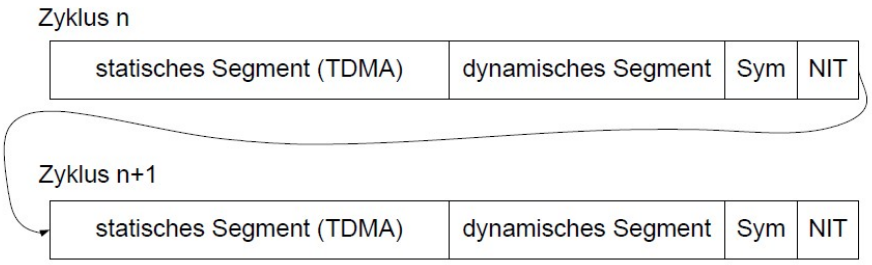


Abbildung 3.11: Kommunikationszyklus des FlexRay aus [3]

Abbildung 3.12 zeigt ein solches, oben beschriebenes statisches Segment. Dynamische Segmente sind diesem Aufbau sehr ähnlich, die Länge der Botschaften ist allerdings nicht festgelegt, muss aber in ein Vielfaches der Minislots passen. Wie aus Abbildung 3.12 auch erkennbar ist, stehen diese Zeitslots auf beiden Buskanälen zur Verfügung. Nachrichten können somit auf beiden Leitungen redundant übertragen werden oder die Datenrate des Busses kann verdoppelt werden, durch paralleles Übertragen unterschiedlicher Nachrichten.

Weiters fällt auf, dass der Datenrahmen dem des CAN-Busses ähnelt. Der wesentliche Unterschied besteht aber darin, dass die Anzahl der Nutzdaten pro Datenrahmen bis zu 254 *Byte* betragen kann und eine erste CRC Prüfung bereits im Header durchgeführt wird.

Ein weiterer Unterschied zum CAN Bus sind die in der FlexRay Spezifikation enthaltenen „Bus Guardians". Diese sind in jedem Knoten implementiert und überwachen während des statischen Nachrichtenteils die Kommunikation. Bei Verstößen gegen die Spezifikation kann der Bus Guardian den Transceiver vom Bus trennen.

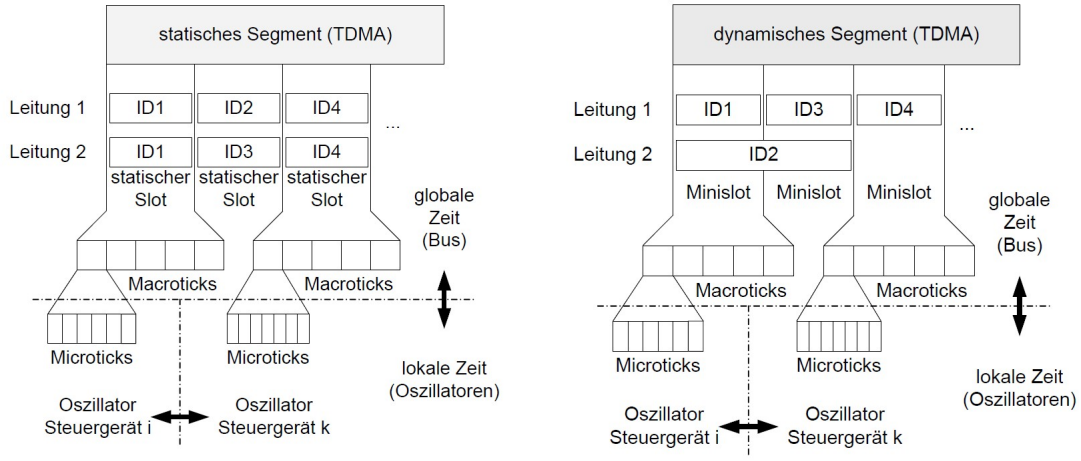


Abbildung 3.12: Segmentaufbau FlexRay aus [3]

3.3.5 TTP/C-Bus

Das Wiener Unternehmen TTTech und die Technische Universität Wien entwickelten einen weiteren zeitgesteuerten Bus für die X-by-Wire Anwendungen, den TTP/C-Bus (Time Triggered Protocoll Class C Bus). Dieser Bus findet Anwendung im Airbus A380, konnte sich aber bisher trotz seiner Vorteile in der Automobilindustrie noch nicht etablieren. Dennoch finden sich einige Implementierungsansätze von X-by-Wire Systemen mit TTP/C-Busnetzwerken (siehe [16] und [5]).

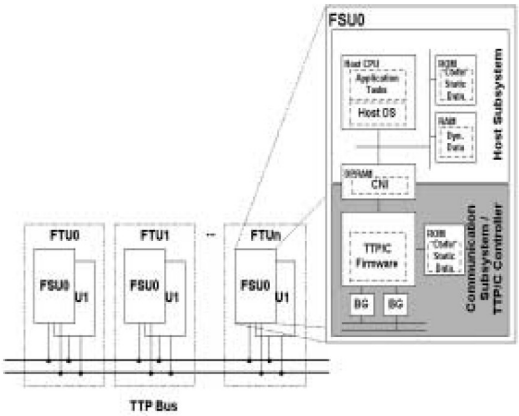


Abbildung 3.13: Aufbau eines TTP/C-Busteilnehmers aus [16]

Den vereinfachten Aufbau eines Echtzeitsystems mit TTP/C-Bus zeigt Bild 3.13. Wie aus dieser Abbildung ersichtlich ist, teilt sich ein solches System in zwei Subsysteme auf. Das Host Subsystem, welches die Echtzeitanwendung exekutiert und das Kommunikationssubsystem, das die Echtzeitkommunikation übernimmt. Als Interface (CNI) zwischen diesen beiden Systemen dient ein Dualport-RAM. Ein TTP/C-Busteilnehmer wird typischerweise als ein einfach redundantes System (Fault Tolerant Unit) ausgeführt und beinhaltet zwei dieser so aufgebauten Systeme, genannt „Fail Silent Unit (FSU)"(siehe 3.24).

Der TTP/C-Bus kann mit verschiedenen physikalischen Medien eingesetzt werden, hat eine Datenübertragungsrate von bis zu 26 $MBit/s$ und wird, wie der FlexRay-Bus, über zwei getrennte physikalische Kanäle ausgeführt. Der Buszugriff erfolgt ebenfalls über das Time Division Multiple Access Verfahren und wird durch unabhängige Bus-Guardians überwacht. Diese stellen sicher, dass jeder Kommunikations-Controller nur im jeweils geplanten Zeitfenster Nachrichten senden kann.

Jeder Busteilnehmer hat somit einen vorbestimmten Zeitslot, in dem er Daten übertragen darf. Die Länge dieser Zeitslots kann von Knoten zu Knoten unterschiedlich sein und muss daher vor Inbetriebnahme abgestimmt werden.
Weiters ist eine globale Zeitbasis und das Einhalten der Zeitslots ein fundamentales Konzept des TTP. Die Einteilung der Zeitslots wird im TTP/C-Controller in der Message Descriptor List (MEDL) gespeichert und ein eigener Startup Node, der keine anderen

Applikationsaufgaben erfüllt, übernimmt den Synchronisationsmechanismus.

Folgende Services werden neben der grundsätzlichen Echtzeitkommunikation ebenfalls von den TTP/C-Controllern übernommen:

- Membership Management
- Reintegration von temporär ausgefallenen Busteilnehmern
- Nachrichtenfehler-Erkennung
- Clock-Synchronisation

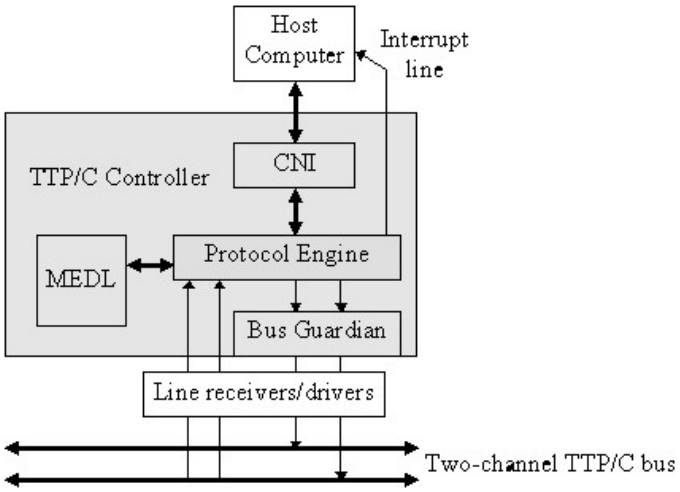


Abbildung 3.14: Aufbau eines TTP Controlers aus [5]

3.3.6 MOST Bus

Der „Media Oriented Systems Transport Bus" oder MOST-Bus ist sowohl als optische Variante, mit 25 $Mbit/s$ und 150 $Mbit/s$ Bandbreite, als auch als elektrische Version, mit 50 $Mbit/s$ Bandbreite, spezifiziert. Dieser Bus stellt den de-facto Standard für multimediale Anwendungen im Automobil dar.
Die optische Variante des Busses wird als Ring ausgeführt und beinhaltet ein Gerät, den Timing-Master, der den Systemtakt vorgibt.

Abbildung 3.15 zeigt den Aufbau des Datenrahmens beim MOST-Bus. Der Datenrahmen besteht aus zyklisch wiederholten Blöcken, die aus jeweils 16 Frames bestehen. Jeder Frame besteht aus 15 Quadlets (4 *bit* Blöcke).
Das erste Quadlet eines Frames ist die Präambel und dient zur Synchronisation. Der darauf folgende Boundary Descriptor gibt die Länge des synchronen und asynchronen Bereichs an. Beide Bereiche zusammen können maximal 480 Bit belegen.
Im synchronen Bereich werden überwiegend Echtzeitdaten wie Audio/Video oder Sensorwerte übertragen, während die Daten des asynchronen Bereiches eine beliebige Länge

haben können und über mehrere Frames fragmentiert werden können (z.B. Kartendaten von DVD zum Navigationssystem). Es könnten auch Dienste wie TCP/IP übertragen werden.
Mit dem Control Frame können Diagnose und Statusnachrichten übertragen werden, diese dienen zur Fehlererkennung.

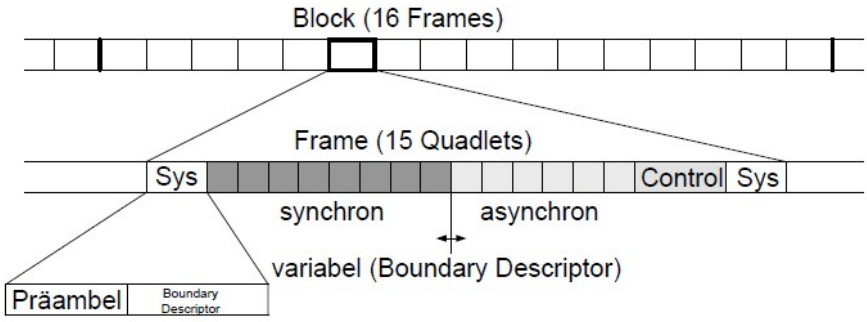


Abbildung 3.15: Datenrahmen Struktur beim MOST-Bus aus [3]

3.4 Zuverlässigkeit und Verfügbarkeit

Elektronische Systeme, die in sicherheitskritischen Bereichen Anwendung finden, müssen die Attribute Sicherheit und Zuverlässigkeit in hohem Maße aufweisen. Dies sind oft unterschiedliche, in Einzelfällen auch widersprüchliche Anforderungen.
Ein zuverlässiges System weist folgende Eigenschaften auf:

- Zuverlässigkeit $R(t)$
- Verfügbarkeit A oder V
- Sicherheit $S(t)$
- Vertraulichkeit
- Verwendbarkeit
- Wiederherstellbarkeit
- Wartbarkeit
- Erweiterbarkeit

Während ein ausfallsicheres System vorrangig die Eigenschaften: Fehlerprävention, Fehlervorhersage, Fehlerbeseitigung und vor allem auch Fehlertoleranz besitzen soll.

Fehlerprävention sollte vor allem in der Designphase vorgesehen werden. Hardwarekomponenten qualitativ hochwertig auswählen, Umgebungstemperaturen beachten oder Reduktion der mechanischen, thermischen und elektrischen Belastung sind hierfür Beispiele.

Für Software müssen die Wahl der Programmiertools, Programmiersprache und auch der Designrules betrachtet werden.

Die Fehlerbeseitigungsphase wird bei ersten Systemtests und Prototypen vorgenommen. Im Automobil-Sektor gängige Tests hierfür sind:

- Funktionstests bei $8, 13.5$ und $16\ V$ bei $-40,\ 25,\ 85°C$
- Hitzetests bei $85°C$ für $16\ h$
- Kältetests bei $-40°C$ für $2\ h$
- Langzeittests bei $85°C$ für $504\ h$
- Temperaturschock- und Temperaturwechseltest

Fehlervorhersage wird mittels eines Modells des Systems für Fehlerfälle durchgeführt und erhöht damit die Fehlertoleranz des Systems. Solche Systemmodelle können simple Blockdiagramme, Markov-Modelle oder Generalized Stochastic Petri Nets (GSPN) Modelle sein.

Fehlertoleranz ist die Eigenschaft eines technischen Systems, seine Funktion aufrecht zu halten, wenn unvorhersagbare Fehler in Hard- oder Software auftreten.

Fehlertolerante Systeme stellen einen kontinuierlichen Betrieb des Systems auch dann sicher, wenn Systemkomponenten oder Teilsysteme fehlerhaft arbeiten oder sogar ausfallen.

Die Fehlervorhersage und Fehlertoleranz umfasst verschiedene präventive Maßnahmen auf unterschiedlichen Ebenen, die in diesem und den folgenden Abschnitten näher besprochen werden. Der folgende Abschnitt befasst sich mit den wichtigsten Ansätzen zur Analyse von Sicherheit und Zuverlässigkeit.

Abschnitt 3.4.3 beschreibt Fehlertoleranzverfahren in Hardware, Abschnitt 3.4.2 in Software. Fehlertoleranz sollte aber auch bei Benutzerschnittstellen in Betracht gezogen werden. Häufig verursachen fehlerhafte Benutzereingaben (menschliches Versagen) unerwünschtes Systemverhalten.
Um verschiedene Implementierungsvarianten und Systemmodelle vergleichen zu können, müssen die oben vorgestellten Systemeigenschaften quantitativ bestimmt werden.

Ausfallrate $\lambda(t)$ wird meist als konstant über die Zeit t angesehen und kann aus Tabellen entnommen werden. Tabelle 3.3 zeigt einige wichtige Ausfallraten. Die Ausfallrate kann für einzelne Bauteile mit $0,1...100\ FIT$, für Baugruppen mit $100...10000\ FIT$ angesetzt werden ($FIT\ =\ failure\ /\ 10^9\ h$).

Zuverlässigkeit $R(t)$ gibt die Wahrscheinlichkeit an, dass die geforderte Funktion unter vorgegebenen Arbeitsbedingungen während einer festgelegten Zeitdauer ausfallsfrei ausgeführt wird. In der Praxis ist der Verlauf der Zuverlässigkeit eine Funktion der Missionsdauer und der Ausfallrate, daher $R(t)$.

$$R(t) = e^{-\lambda t} \tag{3.3}$$

MTTF - Mean Time To Failure gibt die durchschnittliche Einsatzzeit bis zu einem Fehler an

$$MTTF = \int_0^\infty R(t)dt \tag{3.4}$$

MTTR - Mean Time To Repair gibt die durchschnittliche Reparaturzeit an. Diese ist der Kehrwert der Reparaturrate $\mu(t)$, die der Fehlerrate $\lambda(t)$ analog ist und ebenfalls konstant über die Zeit angenommen wird.

$$MTTR = \frac{1}{\mu(t)} \tag{3.5}$$

MTBF - Mean Time Between Failure gibt die durchschnittliche Zeit an, die zwischen dem Auftreten zweier Fehler vergeht. Bei konstanter Ausfallrate ist MTBF der Kehrwert der Ausfallrate.

$$MTBF = \frac{1}{\lambda} \tag{3.6}$$

Verfügbarkeit A gibt die Wahrscheinlichkeit an, dass ein System zu einem bestimmten Zeitpunkt unter den geforderten Bedingungen, die geforderte Funktion ausführt.

$$A = \frac{MTTF}{MTBF} = \frac{MTTF}{MTTF + MTTR} \tag{3.7}$$

Bauteil	**FIT**	**Bauteil**	**FIT**
TTL-SSI	5	bipolarer Transistor	3
CMOS-SSI	3	FET	3
RAM ($< 1\ Mbit$)	20	Power Transistor	40
RAM ($> 1\ Mbit$)	45	Diode	3
EPROM ($< 1\ Mbit$)	10	LED Display oder LCD	15
EPROM ($> 1\ Mbit$)	20	Widerstand	1
Microcontroller	20	Lampe 12 V	500
DSP	40	Lampe 24 V	1000
CMOS Gatter	70	Kondensator	3
Operationsverstärker	3	Schalter (pro Pol)	7
selbstkonstruierte analoge Schaltung	45	Relais	70
Lötkontakt	0,5	Optokoppler	5

$FIT\ =\ failure\ /\ 10^9\ h$

Tabelle 3.3: Einige wichtige Ausfallraten von elektronischen Bauteilen

3.4.1 Methoden zur Analyse von Fehlertoleranzverfahren

Programme zur Qualitätssicherung und Zuverlässigkeitserhöhung beginnen bereits bei der Projektorganisation und Projektplanung und umspannen den gesamten Projektlebenszyklus. Von der Qualifikation von Bauteilen, Analyse von komplexen Prozessen, bis zur kompletten Dokumentation und Wissensdatenbanken. Die Zuverlässigkeits- und Sicherheitsanalyse, als ein Teil dieser Programme, analysiert, welcher Fehler (hazard analysis) sich wie auf ein System auswirkt (accident sequencing) und wie wahrscheinlich dieser Fehler auftritt (quantitative analysis).

Die gängigsten Sicherheitsanalysemethoden sind:

Preliminary Hazard Analysis (PHA) ist der erste Schritt jedes Analyseverfahrens. Dieses Verfahren listet kritische Elemente von Fehlern auf und bestimmt Konsequenzen.

Hazard and Operability Study (HAZOP) ist ein nicht standardisiertes Verfahren, das in einer Liste von Verbesserungs- und Änderungsvorschlägen resultiert. Dieses Verfahren dient vor allem dem vereinfachten Informationsaustausch zwischen Designern unterschiedlicher Systemebenen. Das zu analysierende System wird dabei in Einzelsysteme unterteilt und Abweichungen, die zu Fehlern führen, aufgelistet. Häufig werden hierbei

auch Schlagwörter wie „Not", „No", „More" oder „Less" eingesetzt, eine Standardisierung dieser Methode gibt es allerdings nicht.

Action Error Analysis (AEA) analysiert die Schnittstellen zum Benutzer und hilft Benutzerfehler zu verhindern. Bei der AEA wird eine Liste von Benutzeranweisungen erstellt und diese auf mögliche falsche Anwendung in den einzelnen Schritten untersucht. Nicht mitanalysiert wird allerdings das Verhalten des Benutzers und seine Fehlerkorrekturmaßnahmen.

Cause-Consequence Analysis (CCA) ist eine Kombination aus Fehlerbaumanalyse (FTA) und Ereignisfolgenanalyse (ETA). Diese Methode ist sehr flexibel anwendbar und zeigt die kritischsten Schwachstellen auf, wird allerdings sehr schnell sehr umfangreich und unübersichtlich.

Die nun folgenden drei Sicherheitsanalysemethoden sind die wichtigsten und am häufigsten eingesetzten. Daher werden diese detaillierter beschrieben und für die Sicherheitsanalyse der Implementierungen (Kapitel 5) herangezogen.

Fehlerbaumanalyse (FTA)

Schon bereits bei anderen Analyseverfahren wird eine baumartige Struktur von Fehlerursachen deutlich. Eine Systematisierung dieser Beobachtung stellt die Fehlerbaumanalyse nach DIN25424 dar. Dabei wird ersichtlich, dass ein Fehler in unterschiedlicher Weise von verschiedenen Ursachen abhängen kann. Die „Blätter" des Fehlerbaums sind elementare Ursachen, die sich nicht sinnvoll auf weitere Ursachen zurückführen lassen. Oft genügt eine aus vielen Ursachen, um einen Fehler auftreten zu lassen, dann wiederum sind mehrere gleichzeitige Ereignisse nötig, um einen Fehler zu produzieren. Diese Fallunterscheidungen lassen sich über logische Verknüpfungen differenzieren.
Fehlerbäume können, wie in Abbildung 3.16 dargestellt, qualitativ erstellt sein. Dies wird als Vorbereitung oder Ergänzung zur FMEA gerne gemacht, um Zusammenhänge leichter erkennen zu können, oder kann auch quantitativ aufgebaut sein, um Wahrscheinlichkeiten zu ermitteln. Fehlerbäume sind durch logische Überlegungen einfach aufzustellen, schwieriger ist es allerdings, den Elementarereignissen richtige Wahrscheinlichkeiten zuzuordnen, da diese meist nicht verfügbar oder nur unter bestimmten Testbedingungen gültig sind.

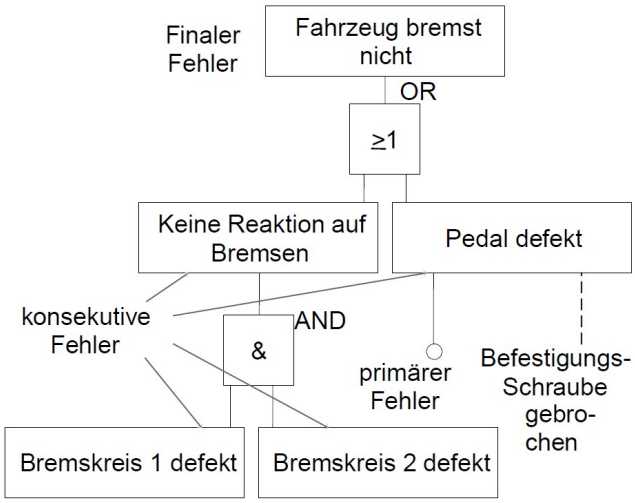


Abbildung 3.16: Beispiel einer Fehlerbaumanalyse aus [3]

Ereignisfolgenanalyse (ETA)

Ausgangspunkt dieser Analyse ist ein konkreter Fehlerfall. Von diesem Fehler ausgehend wird der Schutzmechanismus, der diesen Fehlerfall behandeln soll und meist mehrstufig ist, analysiert. Abhängig von der agierenden Stufe oder dem agierenden Schutzmechanismus kann das System unterschiedliche Endzustände mit unterschiedlichem Gefahrenpotential erreichen. Die Ereignisfolgenanalyse ähnelt der Fehlerbaumanalyse, führt allerdings, entgegengesetzt der FTA, von einem Elementarereignis ausgehend zu den unterschiedlichen Endzuständen. Wie auch bei der FTA ist eine quantitative Analyse mit Wahrscheinlichkeiten an jeder Verzweigung möglich, aber nur schwer zu bestimmen. Abbildung 3.17 zeigt einen Sonderfall. Normalerweise werden bei der ETA laut DIN25419 nur technische Sicherungsmechanismen eingebunden, hier wird auch die Reaktion des Fahrers in Betracht gezogen.

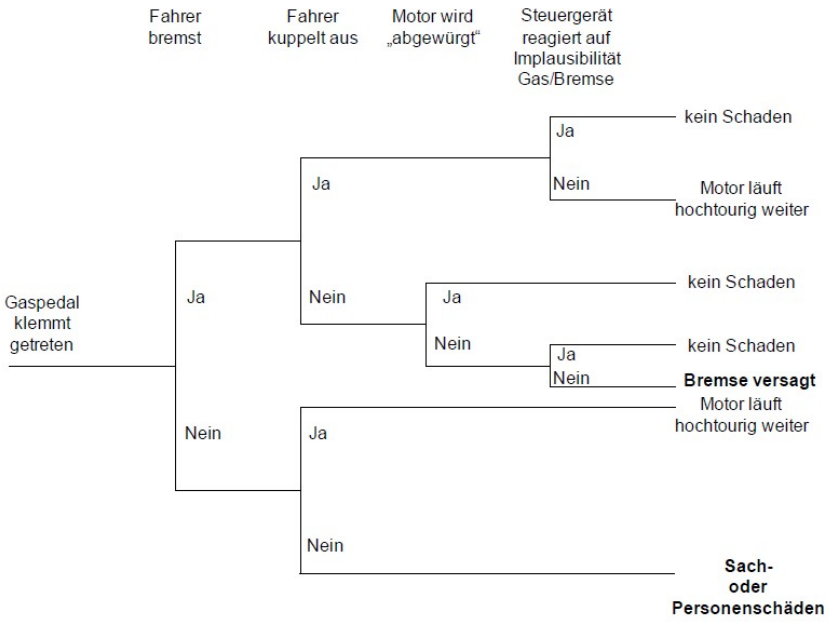


Abbildung 3.17: Beispiel einer Ereignisfolgenanalyse aus [3]

Failure Mode Effect Analysis (FMEA)

Die FMEA ist das mit Abstand wichtigste Verfahren zur Sicherheits- und Zuverlässigkeitsanalyse und ist des Weiteren vorgeschrieben in der Automobilindustrie (DIN EN60812 und ISO/TS 16949). Zu Beginn einer Entwicklung sollten Ausfallsmöglichkeiten eines Systems bewertet werden. Bewertungskriterien bei der FMEA sind die Wahrscheinlichkeit des Ausfalls (P), die Schwere der Auswirkung (S) und die rechtzeitige Erkennbarkeit eines solchen Ausfalls (D). Der erste Schritt liegt darin, möglichst alle Fehlerfälle eines Systems aufzulisten. In der Praxis wird dies in kleinen Gruppen unter Anleitung geschulter Moderatoren durchgeführt. Der nächste Schritt ist die Auflistung aller möglichen Ursachen und Folgen eines jeden Fehlers.

Diese Listen werden dann in eine tabellarische Ansicht gebracht, wie z.B. in Abbildung 3.18, und alle Fehler, Ursachen und Wirkungen quantitativ bewertet. Dabei können die drei Faktoren Wahrscheinlichkeit (P), Schwere (S) und Detektierbarkeit(D) Werte zwischen 1 und 10 annehmen und werden zur Risk Priority Number (RPN) zusammengefasst.

$$RPN = P \cdot S \cdot D \tag{3.8}$$

Die Gewichtung von P, S und D kann wie in Tabelle C.1 erfolgen.
RPN gibt eine Einschätzung, wie kritisch ein Fehler einzustufen ist. Aus Kostengründen werden häufig leichte Fehler riskiert und Gegenmaßnahmen bei gravierenden Fehlern getroffen. Üblicherweise werden Werte der RPN von 250...300 als Schwellwert definiert. Somit lassen sich bereits in der Entwicklungsphase eines Produkts potenzielle Schwachstellen ermitteln und damit frühzeitig Verbesserungsmaßnahmen treffen. Weiters ist es auch möglich, Designalternativen schnell zu vergleichen.

FMEA für Yacht-Autopilot		Aschaffenburg, 14.11.2003			Bewertung				RPN
Funktion	Teilfunktion	Fehlerart	Fehlerfolge	Fehlerursache	P	S	D	RPN	Maßnahme
Schiff sicher von A nach B bringen	Position bestimmen	falsche Position	falscher Kurs	Sonnenwinde	1	5	1	5	
				GPS wird abgeschaltet	1	10	9	90	
				zivile Genauigkeit reduziert	1	5	9	45	
				GPS defekt	1	10	9	90	
				GSP Satellit defekt	1	5	9	45	
	Wind bestimmen	falsche Windrichtung	Drift falsch berechnet	Windmesser ungenau	5	5	1	25	
			Skipperwarnung falsch	schnell drehende Winde	10	9	5	450	...
				zu wenig Wind	10	9	5	450	..
				Windmesser abgeweht	1	9	9	81	
		zu niedrig	Drift falsch berechnet	Windmesser ungenau	5	5	1	25	
				Windmesser abgeweht	1	5	9	45	
		zu hoch		Windmesser ungenau	5	5	1	25	
	Wassertiefe bestimmen	zu niedrig	falscher Seekartenort	Echolot defekt	5	10	9	450	...
				Fischschwarm	5	5	10	250	
		zu hoch		Echolot defekt	5	5	9	225	
	Route berechnen	falsche Route	falsche Ausgabe	falsche Eingabe	5	10	1	50	
				falsche Position	1	10	9	90	
				Spiel in Lenkung	5	10	3	150	
	Ruder einstellen	völlig falsch	völlig falscher Kurs	Abriß	1	10	9	90	
				mech. Defekt	1	10	9	90	
		ungenau	leichte Abweichung	Spiel in Lenkung	5	5	3	75	
		zu stark	leichte Abweichung	zu hohe Motorspannung	5	5	9	225	
		zu schwach	leichte Abweichung	Riemen	9	5	5	225	
		Null	völlig falscher Kurs	Riemen	5	10	9	450	...
	Geschwindigkeit messen	zu niedrig	Abweichung zu vGPS	starke Strömung	10	5	5	250	
				v-Messer defekt	5	5	10	250	
		zu hoch		starke Strömung	10	5	5	250	
				v-Messer defekt	5	5	10	250	
	Skipper warnen	Fehlalarm	Skipper irritiert	Defekt	1	1	10	10	
		gar nicht	Kentern	Defekt	5	10	10	500	Warnsystem doppelt ausle

Abbildung 3.18: Beispiel einer FMEA Tabelle aus [3]

3.4.2 Fehlertoleranz von Software

Fehlertoleranz von Software bezieht sich auf Fehlererkennungs- und Fehlerbehebungsverfahren. Die meisten Softwarefehler sind Designfehler. Diese sind dauernde Fehler, die sowohl im Zeit-, als auch Wertebereich auftreten können. Beispiele dazu sind Programmierfehler, Konfigurationsfehler, falsch ausgelegte Spezifikationen oder auch Probleme, entstanden durch die Komplexität der Software (mehr als 60 MB umfassend in Oberklassefahrzeugen).
Probleme bestehen auch bei der Anwendung von Verfahren zur Sicherheits- und Zuverlässigkeitsanalyse. Die Softwaresparte ist ein relativ neues Feld, in dem diese Verfahren angewandt werden und es mangelt noch an anerkannten und erprobten Methoden. Außerdem sind diese zeit- und kostenintensiv und können oft Echtzeitaspekte kaum oder gar nicht analysieren.
Dennoch sind etwa 90 % der Innovationen im Automobilbereich die Elektronikarchitektur betreffend und ein Großteil davon ist softwarebasierend.

Auf Software-Ebene kann Fehlertoleranz durch folgende Maßnahmen erreicht werden:

- Design-Diversität: verschiedene Implementierungen eines Algorithmus laufen parallel
- Daten-Diversität: die Eingabedaten werden leicht modifiziert mehrfach bearbeitet (z. B. gut gegen Rundungsfehler)
- Temporale Diversität: ein Algorithmus wird mit denselben Daten mehrfach aufgerufen (z. B. gut gegen kurzzeitige Hardwarefehler)
- Recovery Blocks: Kombination aus Hardware und Software Redundanz (vergleichbar mit passiver Redundanz 3.4.3)
- Deadline Mechanismen

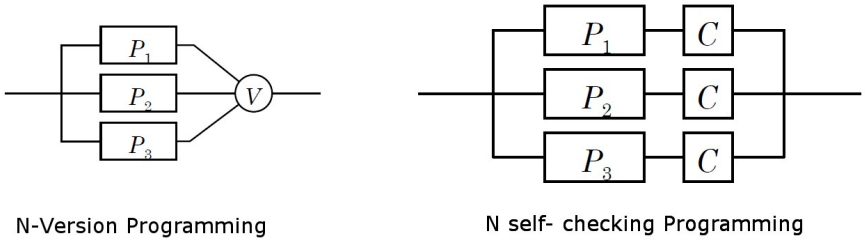


Abbildung 3.19: Implementierungsvarianten mit Design-Diversität

Bild 3.19 zeigt zwei verschiedene Ansätze für Design-Diversität. Beim „N-Version Programming" werden N ungleich implementierte Softwaremodule eingesetzt, die alle die gleiche Aufgabe erfüllen sollen. Ein Voter ermittelt dann aus einem Mehrheitsentscheid die Ausgabe. Dieses Verfahren kann $\frac{(N-1)}{2}$ Modulfehler tolerieren und ist mit aktiver Redundanz in Hardware vergleichbar (siehe 3.4.3).
„N self-checking Programming" arbeitet ebenfalls mit N Versionen eines Softwaremoduls parallel, allerdings ist jedes Modul so ausgeführt, dass es sich selbst überprüft. Dies erspart einen gemeinsamen Voter, der ebenfalls zu Fehlern führen kann.

Programmiersprachen

Die weit verbreitetste Programmiersprache, auch im Automobilsektor, ist C. Zeitkritische Komponenten werden vereinzelt noch in Assembler geschrieben. Diese Sprache spielt aber, wie auch die objektorientierten Programmiersprachen (C++ oder Java), eine untergeordnete Rolle. Wichtige Anforderungen an automotive Softwarecodes sind Zuverlässigkeit und Wiederverwendbarkeit. Raffinierte Programmiertricks und vielschichtige Vererbungen haben besonders in sicherheitskritischen Anwendungen keine Berechtigung. Die Verwendung von C (ohne Einschränkungen) für sicherheitskritische Anwendungen ist jedoch, nicht zuletzt wegen einigen unspezifizierten Eigenheiten, fragwürdig. Solche Fehler werden durch intensive Tests nicht sicher erkannt. Ein Ansatz zur Verbesserung ist die Sprache EC++ (embedded C++), die sich bisher aber nicht durchsetzen konnte, und streng typisierte Sprachen (z.B. Ada, Pascal).
Der Ansatz, der von der Automobilindustrie verfolgt wird, ist, nur ein genau bestimmtes Subset der Sprache C zu verwenden. Mit den unternehmensübergreifenden MISRA C Richtlinien [1] wurde die Programmiersprache C genau auf die Anforderungen der Automobilindustrie zugeschnitten. Diese Richtlinie wird auch von zahlreichen Compilern für fast alle Betriebssystem unterstützt.

Softwaretests

Wie sicherheitsgerichtete Software entworfen und dokumentiert sein soll, wird in Teil 3 der IEC 61508 Norm beschrieben.
Die Norm beschreibt dabei den gesamten Lebenszyklus der Software. Alle Tätigkeiten und

Ergebnisse des Lebenszyklus müssen dokumentiert werden. Basierend auf dem Software Lebenszyklus wird das bekannte V-Modell angewandt und entsprechend der Norm erweitert (siehe Bild 3.20).

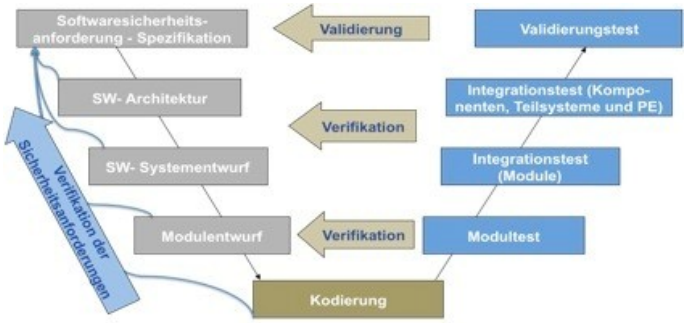


Abbildung 3.20: erweitertes V-Modell laut IEC 61508

Bei der Spezifikation der Softwaresicherheitsanforderungen werden sicherheitsrelevante Funktionen, Schnittstellen zur Hard- und Software beschrieben und es erfolgt eine Trennung der sicherheitsrelevanten und nicht sicherheitsrelevanten Programmteile.
Im zweiten Schritt, dem Entwurf der Software-Architektur, sollten bereits Schnittstellen definiert, Interruptverarbeitung, Zeitanforderungen, Funktionsgraphen und andere Constraints festgelegt werden.
Beim Software-Design werden dann die Softwaremodule näher spezifiziert. Dazu eignen sich Klassendiagramme, Blockdiagramme und/oder Zustandsautomaten.
Die Festlegung der verwendeten Werkzeuge und Wahl der Programmiersprache erfolgt erst im letzten Schritt vor dem eigentlichen Erzeugen des Codes.

Die linke Seite des V-Modells bezeichnet die Tests und Validierungsprozesse, die für die Verifikation des Codes erforderlich sind.
Bei Software-Modultests finden sich Testkonzept und Testfälle. Die Testfälle definieren eine Erwartung und ein Resultat aller relevanten Zweige im Modul. Dabei sollten neben Guttests und Grenzwerttests auch Über- und Unterläufe aller Funktionalitäten geprüft werden. Modultests werden meist als Whitebox-Tests durchgeführt.
Der Integrationstest gliedert sich in zwei Phasen, einmal für die Module und ein zweites Mal für das Teilsystem. Beim Modul-Integrationstest werden mehrere Module und ihr Zusammenspiel untereinander analysiert. Beim eigentlichen Integrationstest selbst wird das Teilsystem auf der Zielhardware getestet. Wichtig ist dabei, dass alle Softwaretests für die selbe Softwareversion durchgeführt werden und jederzeit für diese Version reproduzierbar sind. Diese Tests werden ebenfalls als Whitebox-Tests durchgeführt, jedoch werden einzelne Funktionen schon als abgeschlossene Einheiten ohne deren Interna betrachtet.

Der oberste Punkt auf der linken Seite des V-Modells betrifft die Software-Validierung. Hierbei soll sichergestellt werden, dass das Produkt den Spezifikationen der Sicherheitsanforderungen genügt. Die Validierung soll durch Simulation oder Stimulation folgender Punkte durchgeführt werden:

- Eingangssignale
- unerwünschte Zustände
- Grenzwertanalyse
- Ablaufanalyse
- Performancetest

Der Systemtest wird als Blackbox Test durchgeführt und besteht aus einer großen Anzahl unterschiedlicher Teiltests:

funktionaler Test: ist Hauptbestandteil des Systemtests. Dabei wird geprüft, ob eine geforderte Funktion laut Spezifikation erfüllt wird.

Robustheitstest: hierbei wird die Auslastung von Ressourcen (hauptsächlich Speicher und Rechenzeit) überprüft.

Recovery Test: testet, ob ein Steuergerät nach einer funktionalen Störung wieder in den ordnungsgemäßen Betrieb überführt werden kann.

Benchmark Test: vergleicht Leistungsmerkmale (z.B. Rechenzeit) bestimmter Algorithmen untereinander.

Kompatibilitätstest: stellt sicher, dass ein Steuergerät unter unterschiedlichen Randbedingungen (z.B. Fahrzeug Ausstattungsvarianten) funktioniert.

Usability Test: untersucht die Benutzerfreundlichkeit der Software durch nicht in die Entwicklung involvierte Testpersonen.

Sicherheitstest: prüft überwiegend Sicherheitsaufgaben, wie Wegfahrsperre oder Schutz gegen Tuning.

Dauertest: überprüft Software und Hardware in Versuchsfahrzeugen im normalen Straßenverkehr. Dabei entstehen zufällig ständig neue Situationen, die eventuell bei der Planung nicht bedacht wurden.

Nachteilig am V-Modell ist jedoch eine fehlende Rückkopplung zu frühen Phasen, so dass Fehler bzw. Änderungen erst spät erkannt bzw. berücksichtigt werden können. Dies führt dazu, dass das V-Modell während der Fahrzeugentwicklung mehrmals durchlaufen wird.

Meist wird ein Prototyp mit eingeschränkter Funktionalität entwickelt und in realer Umgebung frühzeitig erprobt. Damit können Defizite früher erkannt werden und der Prototyp wird verwendet, um einen verbesserten Prototypen zu entwickeln. Dieser evolutionäre Zyklus wird wiederholt, bis alle Anforderungen oder Qualitätsziele erfüllt sind. Dieses iterative Vorgehen wird auch als Spiralmodell bezeichnet und ebenfalls von der Automobilindustrie verwendet.
Als vertiefende Literatur zu diesem Abschnitt wird das Buch [29] empfohlen.

Modellbasierte Softwareentwicklung

Modellbasierte Softwareentwicklung ist der neueste Trend in der Automobilindustrie. Dabei wird der Programmcode aus dem Entwicklungswerkzeugen direkt heraus erzeugt. Es gibt zahlreiche solcher Werkzeuge (z.B. Simulink, Easy5, OLT, CASE-Tools), die das Verhalten eines Steuergeräts zunächst als Signalflussplan modellieren. Dieser Flussplan kann einerseits als Dokumentation genutzt werden, ermöglicht andererseits eine Simulation dieser Software und des Steuergeräts in einer sehr frühen Entwicklungsphase. Aus diesem Simulationsmodell lässt sich dann der C-Code für einige bekannte Microcontroller erstellen. Diese Vorgehensweise wird auch als Rapid Control Prototyping (RCP) bezeichnet.
Mit RCP soll die Softwarequalität gesteigert und gleichzeitig die Anzahl der zeitintensiven Fahrzeugtests verringert werden. Die modellbasierte Softwareentwicklung vereinfacht die systematische Bestimmung von Testfällen für die spätere Implementierung und die Verifikation bestimmter Softwareeigenschaften. Weiters werden komplexe Softwaresysteme in relativ einfache Bausteine zerlegt, die unabhängig von einander getestet und in anderen Systemen eingesetzt werden können. Nachteil dieser Methode sind die verminderte Effizienz solcher Codes, schlechte Lesbarkeit des Codes und die Abhängigkeit von eventuell teuren Werkzeugen aufgrund fehlender Standards.

Andere Entwicklungsrichtungen setzen auf UML (Unified Modeling Language). Bisherige Erfahrungen mit dem Einsatz von UML für das Modellieren und Realisieren von X-by-wire Software zeigen aber laut [36] folgende Probleme:

- meist außerordentlich umfangreiche und vermischte Realisierung von Normalverhalten, Konsistenzprüfung und Reaktion auf Fehlverhalten in Statechart-Modellen
- Schwerpunkt der Modellierungsansätze liegt auf statischen Komponentenmodellen, dynamische Rekonfigurationsprozesse lassen sich kaum erfassen
- keine Integration datenflussorientierter Modellierungselemente

3.4.3 Hardware Fehlertoleranzverfahren

Hardwarefehler sind, im Vergleich zu Softwarefehlern, nicht ausschließlich Designfehler. Bei Hardwarefehlern liegen sehr oft Kombinationen aus Designfehlern und/oder physikalischen Fehlern vor, die auch temporär auftreten können. Fehlfunktionen der Hardware lassen sich grob in interne und externe Quellen unterteilen und basieren häufig auf den folgenden Fehlerquellen:

intern	**extern**
• Alterung	• über- oder unterschreiten des Temperaturbereichs
• Design Fehler	• Vibrationen
• Übersprechen	• mechanische Beanspruchungen
• Kontakt- und Lötfehler	• Korrosion
• erhöhte Leistungsaufnahme	• elektromagnetische Entladungen
	• Umwelteinflüsse (Feuchte, Staub, chem. Substanzen)

Tabelle 3.4: Hardwarefehlerquellen

Bei elektronischen Schaltungen wird Fehlertoleranz durch Hinzufügen von Redundanz erreicht. Redundanz liegt immer dann vor, wenn eine Funktion so parallel ausgeführt wird, dass der Ausfall eines Pfades keinen Systemausfall darstellt. Redundanz kann in drei Bereichen betrieben werden.

Informationsredundanz: in dieser Domäne wird Redundanz zum Beispiel durch fehlerkorrigierende Codes oder spezielle robuste Datenstrukturen erreicht. Diese Redundanz findet sich in der Software wieder.

Zeitliche Redundanz: erlaubt es, zeitlich auftretende Fehler zu tolerieren. Dies wird auch in Software realisiert, z.B. durch mehrfaches Berechnen eines Wertes mit unterschiedlichen Algorithmen oder durch mehrfaches Senden einer Nachricht.

Hardware Redundanz

Bei redundanten Strukturen auf Hardwareebene gibt es verschiedene Ansätze. Den Ansatz der aktiven Redundanz, bei dem immer alle parallelen Pfade für die Verarbeitung eingesetzt werden. Und die passive oder „Standby" Redundanz, bei der im Fehlerfall auf den Parallelpfad gewechselt wird. Bei diesem Ansatz kann zusätzlich noch unterschieden werden zwischen „hot standby", hier arbeitet die Ersatzkomponente laufend, und „cold standby", die Ersatzkomponente wird erst im Fehlerfall eingeschaltet.
Inwieweit sich die Komponentenzuverlässigkeit auf die Systemzuverlässigkeit bei verschiedenen Ansätzen auswirkt, ist in Abbildung 3.21 ersichtlich. Aus dem Bild kann entnommen werden, dass die Zuverlässigkeit von nicht fehlertoleranten Systemen (seriellen Systemen) mit zunehmender Komponentenanzahl sinkt, während die Zuverlässigkeit des Systems mit dem Grad an parallel arbeitenden Teilsystemen steigt.

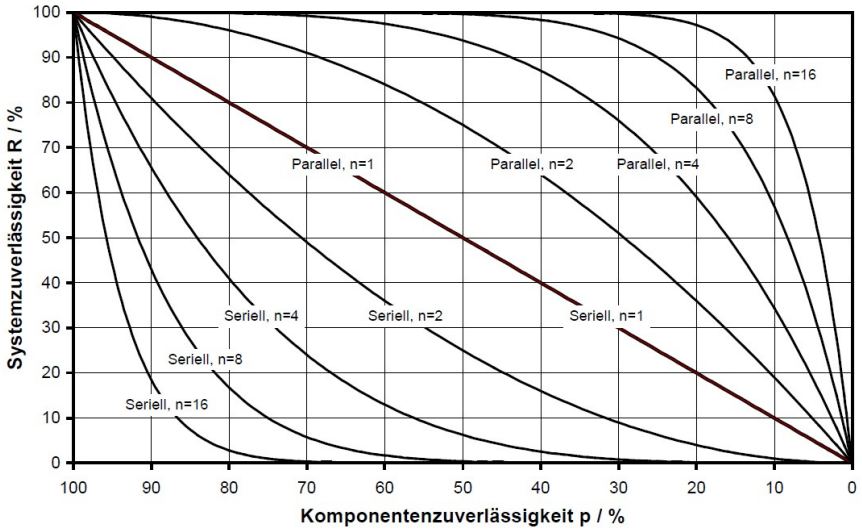


Abbildung 3.21: Zusammenhang zwischen Komponentenzuverlässigkeit und Systemzuverlässigkeit aus [32]

Aktive Redundanz

Abbildung 3.22 zeigt die beiden Varianten aktiver Redundanz. Entweder sind die parallelen Teilsystem direkt zu einem gemeinsamen Ausgang verschaltet, was nur bei Fail Silent Units (FSU) möglich ist. Oder ein Voter ermittelt aus einem Mehrheitsentscheid den richtigen Output. Diese Mehrheitsredundanz wird auch im Luftfahrtbereich häufig eingesetzt, hat aber den erheblichen Nachteil der höheren Kosten, was im Automobilsektor, aufgrund der hohen Stückzahlen, problematisch werden kann.

Laufen z. B. zwei Implementierungen einer Schaltung parallel (Dual Modular Redundancy, DMR), so kann ein Fehler durch Vergleichen der Ausgänge der beiden Komponenten festgestellt, jedoch nicht korrigiert werden. Fügt man eine weitere Instanz den Komponenten hinzu (Triple Modular Redundancy, TMR), so kann der Voter einen Fehler korrigieren. Wird die fehlerhafte Einheit als defekt markiert, ist weiterhin ein Fehler der anderen beiden Teilsysteme erkennbar (wie bei DMR).

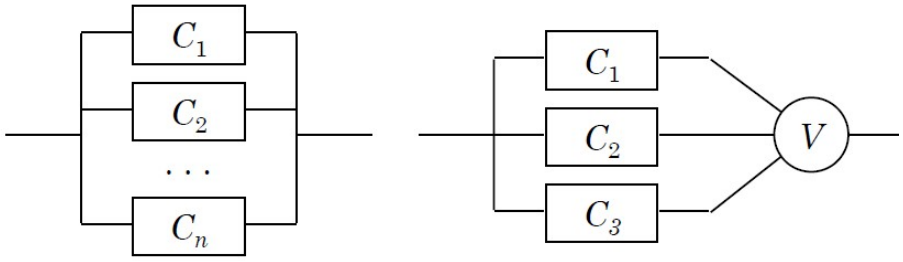


Abbildung 3.22: Varianten der aktiven Redundanz, links: mit Fail Silent Elementen, rechts: TMR mit Voter

Standby Redundanz

Abbildung 3.23 zeigt das Konzept Standby Redundanz. Wie schon vorher erwähnt kann das Reserveelement bereits am Laufen sein (hot standby) oder erst bei Bedarf eingeschaltet werden (cold standby).

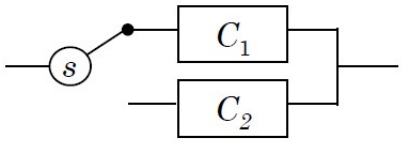


Abbildung 3.23: Standby Redundanz

Um zu verhindern, dass redundante Teilsysteme aufgrund des gleichen Defekts ausfallen, kann man von den Teilsystemen eine möglichst große Unterschiedlichkeit fordern. Dieses Konzept wird diversitäre Redundanz genannt, inkludiert aber einen höheren Aufwand bei Funktionsänderungen oder in Wartungsfällen. Diese Art Redundanz wird sehr verbreitet bei Sensoren und Sensorprinzipien angewandt.

Fail Silent Units (FSU)

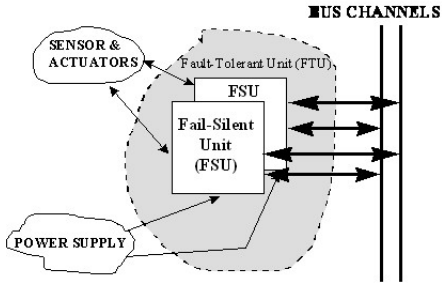


Abbildung 3.24: Fehlertolerantes System bestehend aus zwei FSUs aus [5]

Die Anzahl der redundanten Komponenten t, die notwendig ist, um f Fehler zu tolerieren, ist abhänigig von der Implementierung der redundanten Komponenten selbst.

$t > 3f$ ist notwendig, wenn die redundanten Komponenten kein definiertes Verhalten im Fehlerfall haben und nicht erkennbar ist, welche Komponente fehlerhafte Ergebnisse liefert. Diese Klasse wird „Byzantine Failures" genannt.

$t > 2f$ ist notwendig, wenn alle fehlerfrei arbeitenden Komponenten die fehlerhaften Komponenten identifizieren. „Consistent Failures" nennt man diese Fehlerklasse.

$t > f$ wird benötigt bei Fail Silent Units. FSUs generieren entweder einen richtigen Output oder schalten sich im Fehlerfall ab und generieren gar keinen Output. Diese Klasse benötigt die geringste Anzahl an redundanten Komponenten und ist daher die Grundlage der meisten fehlertoleranten Systeme (FTU). Fehlertolerante Systeme aufgebaut aus FSUs sind relativ einfach aufgebaut- im Vergleich zu den anderen beiden Varianten- und können auch ohne Voter auskommen. Wie ein solches System aufgebaut ist, zeigt Abbildung 3.24.

Vergleich der Redundanzverfahren

Abbildung 3.25 zeigt den Vergleich der unterschiedlichen Redundanzverfahren hinsichtlich ihrer Zuverlässigkeit über die Zeit. Das obere Diagramm zeigt dabei den FSU- gegenüber dem Voting Ansatz. Hierbei wird allerdings davon ausgegangen, dass sich FSUs im Fehlerfall zu 100 % ruhig verhalten und der Voter ebenfalls 100 % fehlerfrei arbeitet. Im unteren Diagramm sieht man einen Vergleich von Standby- und aktiver Redundanz. Auch in diesem Diagramm wird davon ausgegangen, dass sich FSU und der Umschalter zu 100 % dem Fehlerfall entsprechend richtig verhalten. Aus beiden Diagrammen lässt sich erkennen, dass ein System aus zwei parallelen FSUs recht gute Zuverlässigkeit bieten kann.

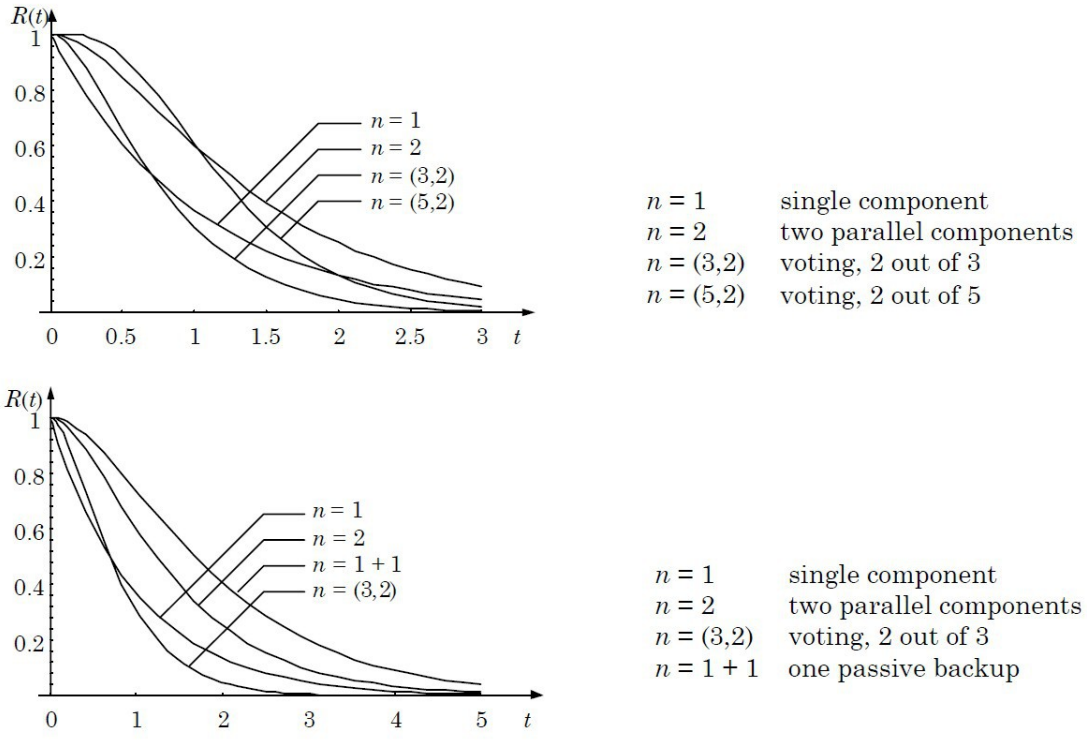


Abbildung 3.25: Vergleich der Hardware Redundanzmöglichkeiten aus [26]

3.4.4 Fehlertoleranz bei Sensoren und Aktuatoren

Fehlertolerante Aktuatoren sind einfach aus mehreren parallelen Aktuatoren aufgebaut. Auch bei Aktuatoren kann entweder aktive oder Standby Redundanz eingesetzt werden, wichtig ist allerdings, dass diese Aktuatoren auf FSUs basieren. Eine Variante dafür wären zum Beispiel mehrere selbsthemmende Motore mit jeweils genügend Moment und ein nicht selbsthemmendes Getriebe („mechanical fail silent"). Auch bei Aktuatoren sollte man diversitäre Redundanz auf Grund unterschiedlicher physikalischer Prinzipien verfolgen.

Bei der Auswahl der Sensoren ist es vorteilhaft auch diversitäre Redundanz einzusetzen. Sensoren an kritischen Elementen (z.B. Gaspedal) sollten so ausgeführt sein, dass zumindest ein Fehler toleriert werden kann. Fehlererkennungsmethoden für Messsignale können sein:

- Limit Checks und Plausibiltätstest bei Einzelsignalen
- Signalmodell-basierend bei periodischen oder stochastischen Signalen
- Prozessmodell-basierend für zwei oder mehr kongruente Signale

Kapitel 4

Entwurf

Das Kapitel 4 dieser Arbeit beinhaltet die Anforderungen, die an die Implementierung gestellt werden und beschreibt die Auswahl der einzelnen Komponenten und Tools. Anhand dieser Entwurfskriterien wird dann im Abschnitt 4.6 eine erste Testimplementierung vorgestellt, mit der die Risikoabschätzung (Abschnitt 4.7), wie im Reglement der Formula Student gefordert, durchgeführt werden kann.
Anhand der festgelegten Entwurfskriterien, der Testimplementierung und durch die, bei der geforderte Risikoabschätzung zusätzlich auftretenden, Anforderungen wird dann im folgenden Kapitel 5 die eigentliche Implementierung im Fahrzeug vorgestellt.

4.1 Implementierungsanforderungen

Die hier besprochenen Implementierungsanforderungen stellen eine Art des Lastenhefts dar, wie es sich zu Beginn der Design- und Implementierungsphase des Projektes dargestellt hat. Ziel dieser Arbeit ist es, ein Drive-by-Wire System in einen Formula Student Boliden zu integrieren, das den Anforderungen des Reglements (Abschnitt A) genügt und ohne größere Modifikationen in das bereits bestehende Design des Rennboliden adaptiert werden kann. Abbildung 4.1 zeigt bereits mögliche Adaptierungen der Pedalerie für Rotationssensoren und Linearpotentiometer an Brems- und Gaspedal.

4.1.1 Sicherheitsanforderungen

Aufgrund der Reglementierung auf Drive-by-Wire Systeme mit mechanischer Rückfallebene, bezogen auf Lenkung und Bremse, stellt nur das Throttle-by-Wire System ein reines X-by-Wire System dar (A.1.1). Brake-by-Wire oder auch Rekuperation über die Elektromotore ist bis zur Hälfte des Pedalweges erlaubt (A.2.1), während die Lenkung eine direkte mechanische Verbindung zu den Rädern der Vorderachse haben muss (A.3.1). Die beiden erlaubten X-by-Wire Systeme (Throttle-by-Wire und Brake-by-Wire) müssen in einer dem Reglement entsprechenden FMEA für FSG und FSA (A.1.2), bzw. einer Risikoanalyse für FSUK (A.2.6) analysiert und dementsprechend implementiert sein. Da es sich bei dem Boliden um ein Rennfahrzeug handelt, das nur bei den entsprechenden Bewerben und auf abgeschlossenen Testgeländen eingesetzt wird, sind keine weiteren gesetzlichen Richtlinien außer den jeweiligen Bewerbsreglements zu beachten.

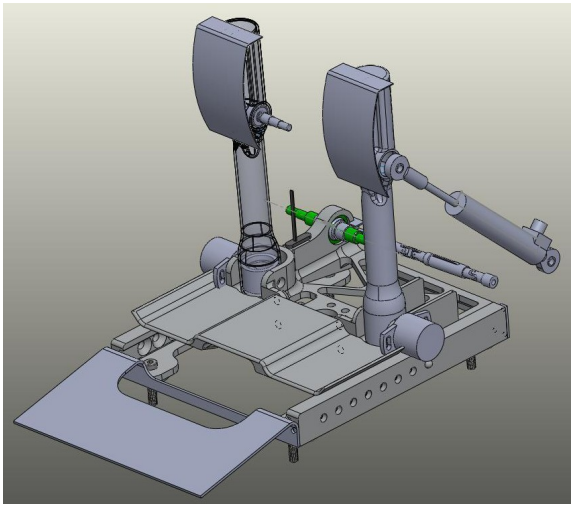


Abbildung 4.1: Pedalerie des MaxWheel 2010

4.1.2 Technische Anforderungen

Die technischen Anforderungen, die an X-by-Wire Systeme vom Reglement der Formula Student gestellt werden, sind bewusst nur sehr rudimentär und geben sehr viel Spielraum für unterschiedliche Implementationsvarianten. Da es sich bei den Bewerben dieser Formel vorrangig um Konstruktionsbewerbe handelt und technisch Neuerungen gefördert werden sollen, sind die einzigen Anforderungen des Reglements:

- der Gaspedalweg muss von zwei Sensoren aufgenommen werden A.1.1
- die Sensorversorgung muss getrennt ausgeführt sein A.2.5
- nur für FSUK: die beiden Sensoren dürfen keine gemeinsamen Komponenten haben A.2.2
- weiters wird empfohlen, die beiden Sensorwerte miteinander zu vergleichen A.1.1

Die Sicherheitsanforderungen an das Chassis, Brems- und Lenkeinrichtung und Fahrersicherheitssysteme des Boliden bewegen sich gleichzeitig auf viel höherem Niveau und zielen vor allem darauf, das Verletzungsrisiko aller Bewerbsteilnehmer zu minimieren. Da der Bolide selbst für diese Arbeit allerdings nur als eine Art Versuchsträger agiert und auf einem Vorjahrsmodell basiert, das diese Sicherheitsanforderungen bereits erfüllt hat, wird in dieser Arbeit nicht näher auf die Sicherheitssysteme und maschinenbaulichen Anforderungen eines Formula Student Boliden eingegangen. Für vertiefende Studien zu diesem Thema wird [19],[28] und [33] empfohlen.

Die nun folgenden technischen Anforderungen sind nicht mehr explizit im FSE Reglement 2010 gefordert, aber, nach Erfahrungen aus dieser Rennserie, sinnvoll und notwendig.

Grundsätzlicher Aufbau des Systems

Hier sind die grundlegendsten und ersten Überlegungen zusammengefasst. Diese Anforderungen sind ein grober Rahmen, der für alle im MaxWheel 2010 verbauten elektrischen Komponenten gilt:

- wasserdichte Gehäuse und Stecker sind zu verwenden (vorzugsweise IP67)
- das Gehäuse soll so klein und leicht wie möglich sein
- Gehäuse und Stecker sollen relative hohe mechanischen Belastungen aushalten
- Platinen und Stecker sind gegen Vibrationen und elektromagnetische Störungen zu schützen
- Stecker und Anschlüsse müssen Zugentlastungen aufweisen
- modularer Aufbau, der das Ersetzen bei Ausfall vereinfacht, ist zu bevorzugen
- schnelle und einfache Verfügbarkeit der verwendeten Bauteile und Einzelkomponenten
- Verwendung von „proven in use" Hard- und Softwarekomponenten
- relevante Fehler müssen dem Fahrer mitgeteilt werden
- der Schaltungsaufwand ist so gering wie möglich zu halten

Zusätzlich gilt speziell für die X-by-Wire Teilsysteme:

- Fahrer muss die Möglichkeit haben, fehlerhafte Systeme zu überstimmen oder zu deaktivieren
- alle Teilsysteme zwischen Gaspedal und Motorsteuerung müssen redundant ausgeführt sein
- die redundanten Pfade dürfen einander unter keinen Umständen beeinflussen (z.B. gemeinsame Voter sind zu vermeiden)

Kommunikationssystem

Von der Wahl des Übertragungsprotokolls ist auch die Wahl des Microcontrollers abhängig. Dieser wiederum beeinflusst die Wahl der Peripherie und Spannungsversorgung. Damit muss das Kommunikationssystem als erste Komponente gewählt werden. Einschränkungen an die Kommunikation werden dabei wenige getroffen:

- Schnittstelle bei allen inkludierten fertigen Zukaufkomponenten integriert
- für automobilen Bereich entwickelter Bus
- ein echtzeitfähiges Bussystem über Kupferleitungen
- einfache Verfügbarkeit der notwendigen Komponenten
- Datenübertragung der Messwerte und zusätzlicher Fehlercodes muss alle 10 ms möglich sein

Sensorik

- die eingesetzte Sensorik soll robust und einfach aufgebaut sein
- unterschiedliche Messprinzipien (siehe 3.4.3 diversitäre Redundanz) sollen verwendet werden
- Limit-Checks und Plausibilitätstest bei den Messsignalen müssen implementiert werden
- Sensoren mit digitalem Ausgang müssen eine Abtastrate $\geq 400\ Hz$ ermöglichen
- der Einsatz von 3 Sensoren pro Bedienelement ist zu bevorzugen
- externer Schaltungsaufwand der Sensorik soll minimal sein

Software

- mit gängiger Programmiersprache geschrieben
- modular aufgebaut und getrennt testbar
- nach Guidelines von MISRA [1] geschrieben
- Debug- und Entwicklungsumgebung oder -tools einfach verfügbar
- einfach geschrieben und schnell rekonfigurierbar
- Diagnosefunktionen sollen implementiert sein
- automatisch generierter Code ist zu bevorzugen, wenn dieser lesbar ist
- diversitäre Programmierung der redundanten Microcontroller ist aufgrund der schnelleren Wartbarkeit zu vermeiden
- „common mode" Fehler aufgrund von Programmierfehlern müssen ausgeschlossen werden können

Spannungsversorgung

- zwei unabhängige Spannungsversorgungen
- jede Spannungsversorgung mit genügend Ausgangsleistung für Sensorik und Verarbeitung im „worst case"
- Diagnoseausgang für jede Spannungsversorgung ist zu implementieren
- Versorgungsspannung ist zwischen 11 V und 14 V anzunehmen
- die Spannungsversorgung darf nicht durch externe Kurzschlüsse an den Sensoren zusammenbrechen

4.1.3 Anforderungen bezüglich Wartbarkeit

Beim Entwurf von Systemen gibt es drei unterschiedliche Systemaspekte:

Perfektionierung zielt darauf ab, ein System inkrementell soweit zu verbessern, bis es die geforderte Zuverlässigkeit erreichen kann. Perfektionierung führt zu einfacheren Konzepten und vermindert die Wahrscheinlichkeit von Designfehlern. Ein perfektioniertes System arbeitet ohne Fehlererkennung oder Fehlersicherheit, ist einfacher zu entwerfen als ein fehlertolerantes System, ist aber nicht für Systeme mit sehr hoher Verfügbarkeit geeignet.

Fehlertoleranz kompliziert ein System und fügt zusätzliche Komponenten hinzu. Fehlertolerante System können Fehler erkennen und auf diese reagieren. Fehler in solchen Systemen führen zu keinem Systemausfall, allerdings sind solche Systeme oft nicht sehr leicht wartbar.

Wartbarkeit eines Systems ist ebenfalls einfacher zu erreichen als Fehlertoleranz. Allerdings ist dies häufig ein Kompromiss zwischen einfacher Systemwartbarkeit und Systemverfügbarkeit.

Für das zu implementierende X-By-Wire System ist zuerst eine Perfektionierung des Systems zu erzielen, so dass die implementierte Schaltung nicht mehr der erste Prototyp mit allen Designfehlern und Kinderkrankheiten ist. Und dann auf ein fehlertolerantes System, das dennoch relativ einfach und schnell wartbar ist, weiter zu entwickeln.

4.2 Wahl des Kommunikationssystems

Wie schon zuvor erwähnt beeinflusst die Wahl des Busses entscheidend die Auswahl des Controllers und der damit verbundenen Peripherie. Somit war der erste Schritt beim Entwurf der Implementierung die Festlegung des Kommunikationssystems. Zur Auswahl stehen dabei eigentlich nur die vorgestellten Bussysteme CAN (3.3.2), TTCAN (3.3.3), FlexRay (3.3.4) und TTP/C-Bus (3.3.5). Da TTP/C-Controller und Bustreiber nicht sehr verbreitet und einfach zu bekommen sind und diese Schnittstelle bei den zur Verfügung stehenden Motorsteuergeräten nicht implementiert ist, steht diese Alternative eigentlich laut den obigen Anforderungen nicht zur Diskussion.
TTCAN, ein Protokoll aufbauend auf dem CAN Bus, der CAN Bus selbst, der nach [30] und [27] unter gewissen Voraussetzungen echtzeitfähig ist und FlexRay sind im Wesentlichen die Alternativen, die die Anforderungen aus 4.1.2 erfüllen.
Die Auswahl des Kommunikationssystems reduziert sich damit auf die Frage:
Ist der Einsatz von FlexRay notwendig, oder können die Echtzeitanforderungen vom CAN erfüllt werden?

Laut [27] kann der CAN Bus für sicherheitsrelevante Anwendungen eingesetzt werden, wenn die Buslast $10 - 20$ % nicht überschreitet und der Standard eingehalten wird. Das ausgewählte Motorsteuergerät von dSpace (MicroAutoBox II) [1] bietet 4 physikalische CAN Schnittstellen an, die im MaxWheel 2010, wie in Abbildung 5.3 ersichtlich, aufgeteilt wurden.

CAN1 ist das primäre Bordnetzwerk. Über diesen Bus kommunizieren der primäre Teil des X-by-Wire Systems (Pedalbox 1), Motorsteuergerät (ECU), Safety Device (überprüft sicheren Betriebszustand aller Komponenten des E-Antriebs), Sensoranbindung im Fahrzeugheck (CAN Knoten), Batteriemanagement und HMI des Lenkrads.

CAN2 bildet den redundanten Teil des X-by-Wire Systems (Pedalbox 2) und der ECU. An dieser Schnittstelle sind sonst keine weiteren Teilnehmer angeschlossen.

CAN3 dient alleinig der Kommunikation zwischen den Motoransteuerungen (Inverter) und der ECU, auch diese Schnittstelle besitzt sonst keine weiteren Teilnehmer.

CAN4 arbeitet mit geringer Übertragungsgeschwindigkeit (500 $kbit/s$) und ermöglicht den Anschluss von autonom arbeitenden Sensoren (z.B. Gierratensensor und Beschleunigungssensorik)

Aus der Abbildung 5.3 ist ersichtlich, dass vor allem die Analyse der Buslast an CAN1 von Interesse ist.
Um die Buslast feststellen zu können, werden zunächst alle auszutauschenden Variablen zu CAN-Botschaften zusammengefasst und die Sendefrequenz dieser Nachrichten festgelegt. Diese Beschreibung der Kommunikationsstruktur wird K-Matrix genannt und unterliegt beim CAN keinem standardisierten Format. Die vollständige K-Matrix findet sich im Anhang C.4, Abbildung 4.2 zeigt aber die wichtigsten Auszüge zur Berechnung der Buslast.

[1] http://www.dspace.de/de/gmb/home/products/hw/micautob.cfm

CAN 1.1	VEHICLE CAN 1	CAN definitions		
		Bitrate	1000 kbit	
		standard Frame	used	
		CAN 2.0B		bus utilization calculations
Level	Message Name	DLC	Transmit Time [ms]	# Messages [s]	# bit [s]
Pedalbox_1 (Little Endian/Intel Dataformat)					
MSG	PB STATUS 1	8	9	112	15120
Batterypack 1..6 (Little Endian/Intel Dataformat)					
MSG	BATTERY PACK VOLTAGES 1-2	8	1000	6	810
MSG	BATTERY PACK VOLTAGES 3-4	8	1000	6	810
MSG	BATTERY PACK VOLTAGES 5-6	8	1000	6	810
MSG	BATTERY PACK VOLTAGES 7-8	8	1000	6	810
MSG	BATTERY PACK VOLTAGES 9-10	8	1000	6	810
MSG	BATTERY PACK VOLTAGES 11-12	8	1000	6	810
MSG	BATTERY PACK TEMPS 1-2	8	1000	6	810
MSG	BATTERY PACK TEMPS 3-4	8	1000	6	810
MSG	BATTERY PACK TEMPS 5-6	8	1000	6	810
CAN Node (Little Endian/Intel Dataformat)					
MSG	CN REAR COMMAND 1	4		1	95
Safety Device (Little Endian/Intel Dataformat)					
MSG	SD COMMAND 1	8	100	10	1350
MSG	SD STATUS 1	8	100	10	1350
MSG	SAFETY_STATUS_2	8			
Steering Wheel (Little Endian/Intel Dataformat)					
MSG	SW TEMPS VOLTAGES	7	1000	1	125
MSG	SW SPEED POWER	3	100	10	850
MSG	SW SOC BATTERY PACKS	6	1000	1	115
MSG	SW ODOMETER	8	1000	1	135
MSG	SW SETTINGS	4	1000	1	95
			Total	201	26525
			bus utilization [%]		2,65

Abbildung 4.2: Auszug aus der K-Matrix CAN1 des MaxWheel 2010

Grundlage für die Berechnung ist die Nachrichtenlänge. Diese ist nominell min. 111 bit lang, wie aus Tabelle 4.1 ersichtlich:

Frameteil	Länge [bit]
Start of Frame (SOF)	1
Arbitrierungsfeld	12
Kontrollfeld (CTRL)	6
Datenfeld (DATA)	0 - 64
Prüfsummenfeld (CRC)	16
Bestätigungsfeld (ACK)	2
End of Frame	7
Pause nach Frame	min. 3
Bits pro Frame	111

Tabelle 4.1: nominelle Länge eines CAN Datenframes (11bit ID)

Es muss allerdings auch noch eingerechnet werden, dass „bit stuffing" das Frame noch verlängert. Nach fünf aufeinander folgenden gleichen Bits muss ein komplementäres Bit eingefügt werden. Dieser Vorgang wirkt auf SOF bis einschließlich dem CRC von Daten- sowie Remote-Frames und dient der Nachsynchronisation der Teilnehmer innerhalb eines Frames. Wie lange der Frame genau wird, ist datenabhängig. Eine Abschätzung der Framelänge l in Abhängigkeit von der Anzahl der Datenbytes n bietet die Formel nach Tindell:

$$l = \frac{34 + 8n}{4} + 44 + 8n \tag{4.1}$$

Die Formel von Tindell kann laut [24] auf die folgende „worst case" Abschätzung vereinfacht werden:

$$l = 55 + 10n \tag{4.2}$$

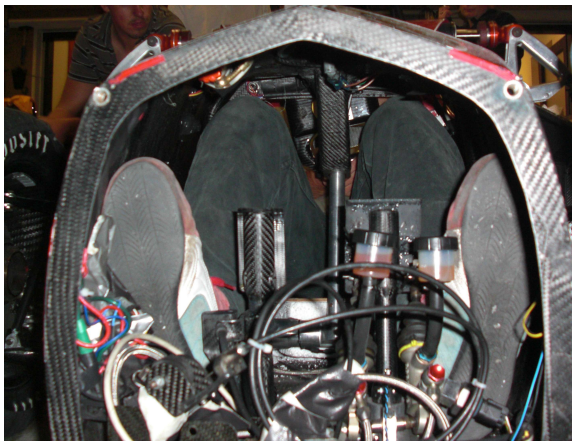
Daraus lässt sich dann die Buslast an CAN1, wie in Abbildung 4.2 ersichtlich, auf $2,65$ % abschätzen.
An CAN2 liegt die Busauslastung sogar bei nur $1,5$ %. Damit sind die Bedingungen für Echtzeitverhalten am CAN erfüllt und dieser Bus kann als Kommunikationssystem der Drive-by-Wire-Systeme des MaxWheel eingesetzt werden.

4.3 Wahl der Sensorik

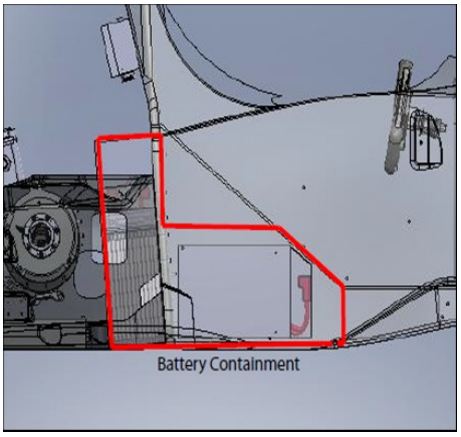
Die letztliche Auswahl der Sensorik beeinflussten vor allem die Aspekte:

1. Verfügbarkeit
2. Bauform und Bauraum
3. Preis
4. Robustheit

Aufgrund des beengten Platzangebotes und der Forderung möglichst einfacher Adaptierbarkeit der bereits bestehenden Pedalerie aus dem Vorjahresfahrzeug konnte ein dritter Sensor pro Pedal nicht eingeplant werden. Dies konnte bereits bei ersten Sitzproben mit geänderter Fahrersitzposition aufgrund der Unterbringung der Batterien festgestellt werden (siehe Bild 4.3).



(a) Sitzprobe mit geänderter Sitzposition



(b) Unterbringung der Batterien und geänderte Sitzposition

Abbildung 4.3: Änderungen in der Fahrersitzposition des MaxWheel 2010: a) Platzangebot Fußraum, b) Batterie Containment unter dem Fahrersitz

Die eingesetzte Sensorik besteht somit nur aus zwei, vom Messprinzip und konstruktiv unterschiedlichen Sensoren der Firma Novotechnik [2]. Bei den Sensoren am Gaspedal

[2] http://www.novotechnik.de/

handelt es sich um ein linear Potentiometer (TEX0050) und einen kontaktlosen, redundanten Winkelsensor (RSC2800). Dieser Winkelsensor wird ebenfalls zur Aufnahme des Lenkwinkels und am Bremspedal verwendet. Der zweite Sensor am Bremspedal sind zwei Drucksensoren der Firma Magneti Marelli Motorsport [3](OPS04). Diese ermitteln den Pedalweg des Bremspedals über die Drücke in den beiden Bremskreisen und ermöglichen gleichzeitig eine Ermittlung der Bremskraftverteilung zwischen hinterem und vorderem Bremskreis.
Alle verwendeten Sensoren zeichnen sich durch rasche Verfügbarkeit, hohe Lebensdauer und relativ geringen Preis aus und wurden für den Automobilbereich entwickelt. Die Ausgangssignale sind jeweils Analogsignale mit linearem Kennlinienverlauf. Abbildung 4.4 zeigt die Lage und den Einbauort der Sensorik.

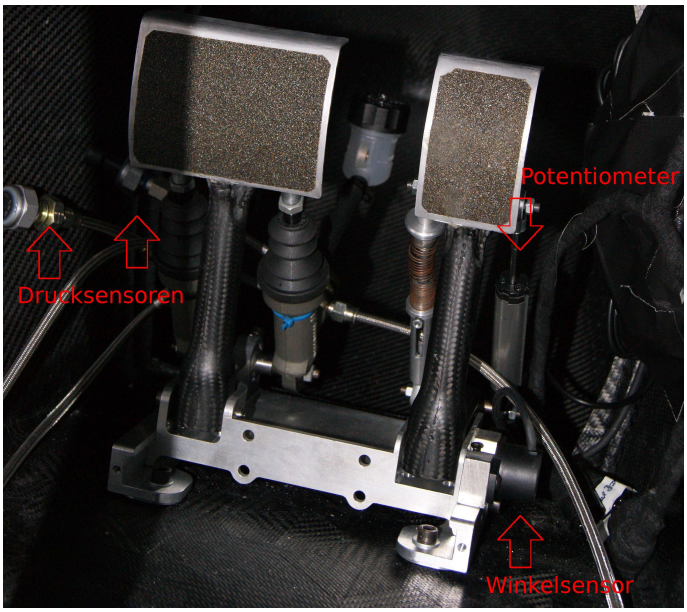


Abbildung 4.4: Pedaleriesensorik im MaxWheel 2010

[3]http://www.magnetimarelli.com/index.htm

4.4 Wahl der Hardwarekomponenten

Aus den Anforderungen an die Implementierung aus 4.1 und den Überlegungen zur Redundanz von 3.4.3 erscheint der Ansatz mit FSUs (siehe 3.4.3) am sinnvollsten. Daraus ergibt sich das folgende Blockdiagramm 4.5, zu dem nun die einzelnen Komponenten zu selektieren sind.

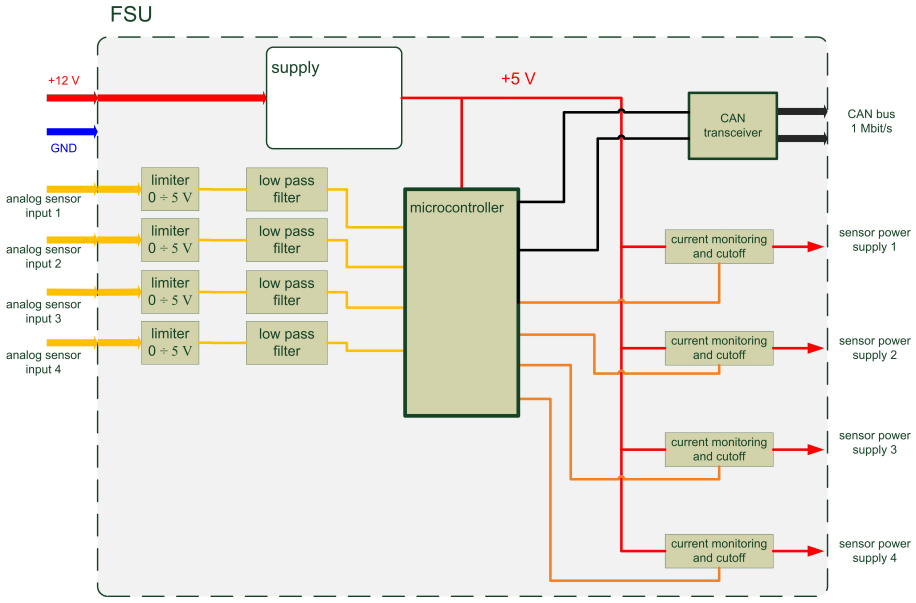


Abbildung 4.5: Entwurfsblockdiagramm des X-by-Wire Systems des MaxWheel 2010

4.4.1 Microcontroller

Bei der Auswahl der Microcontroller wurden vor allem die in Kapitel 2.7.1 vorgestellten DualCore MCUs besonders bevorzugt. Microcontroller, die genau für diese Aufgaben entwickelt wurden und sowohl automotiven Anforderungen erfüllen als auch implementierte Redundanz aufweisen. Jedoch konnte keiner der in Tabelle C.2 gelisteten DualCore Microcontroller die Anforderungen der schnellen und einfachen Verfügbarkeit und raschen Support mit Tools und Entwicklungsumgebungen erfüllen. Bei den meisten MCUs handelt es sich um Neuentwicklungen oder Prototypen, die, wenn überhaupt, nur in einigen Zehntausend Stückzahlen erhältlich sind.

Aufgrund dieser Tatsache und der bereits vorhandenen Erfahrungen, Tools und validierter Programmteile wurden statt eines dieser DualCore MCUs zwei Microcontroller der Serie AT90CANxx der Firma ATMEL [4] parallel eingesetzt.
Die Microcontroller dieser Familie verfügen neben einem integrierten CAN Controller ebenfalls über einen integrierten 10 *bit* ADC und sind im einfach wart- und lötbaren TQFP64 Gehäuse verfügbar.

[4] http://atmel.com/applications/devices_app.asp?AppID=970&source=apps-automotive-microcontrollers

Die benötigte Peripherie dieses Entwurfs beschränkt sich damit für jeden Microcontroller auf:

- einen Quarzoszillator
- einen CAN Transceiver
- eine Vorbehandlung der Analogsignale
- und eine Überwachung der Versorgungsspannungen

4.4.2 Vorbehandlung der Eingangssignale

Die analogen Signale der Sensoren haben allesamt einen Wertebereich von $0 \div 5\ V$ und passen somit zum Eingangsbereich des integrierten ADCs. Um allerdings sicherzustellen, dass erhöhte Spannungen an den Sensoreingängen zu keiner Beschädigung des Microcontrollers führen, müssen alle Eingänge limitiert werden. Um dies und zusätzlich noch eine Entkopplung der Belastung der Sensoren zu erreichen, wird eine relativ einfache Spannungsfolgerschaltung mit einem Operationsverstärker dem Signaleingang vorgeschaltet.
Dieser Spannungslimitierung folgt dann ein einfacher RC Tiefpass. Der Tiefpass soll hochfrequente Störimpulse aus den Signalen filtern und wurde bewusst so einfach gewählt. Der einfache Aufbau erzielt den gewünschten Effekt und ist sehr rasch wartbar und fehlerunanfällig. Die Grenzfrequenz f_g des Tiefpasses wurde auf weniger als die Hälfte der gewünschten Signalabtastrate festgelegt, um dem Nyquist-Shannon Abtasttheorem zu genügen.
Zusätzliche Signalaufbereitungen und -validierungen sind, wie auch eine weitere Tiefpassfilterung, in der Software implementiert.
Geschaltete Binärsignale (z.B. Endpositionsschaltsignale o.Ä.) sind ebenfalls geschützt gegen Störungen und Fehler von außen, realisiert wird dies über Hardware Entpreller (MAX6816 [5]).

4.4.3 Stromüberwachung und -limitierung

Sowohl die Versorgungsspannung als auch jede einzelne Sensorversorgung gilt es zu überwachen und zu limitieren. Die Versorgungsspannung wird dabei vor und nach dem DC-DC Konverter vom Microcontroller überwacht, Fehler oder Probleme werden über den CAN Bus an die angeschlossenen Teilnehmer übermittelt.
Die Überwachung, aber vor allem auch die Limitierung der einzelnen Sensorversorgungen, werden mit Hilfe eines speziellen ICs erledigt. Der Baustein MAX4790 [6] ist für diese Aufgaben konstruiert. Somit ist gewährleistet, dass ein Kurzschluss an einem Sensoranschluss nicht die Funktion der anderen Sensoren bzw. der Auswerteelektronik beeinflussen kann. Der platzsparende IC benötigt kaum externe Beschaltung (wie in Abbildung 4.6 ersichtlich) und eignet sich auch für den automobilen Einsatz.

[5] http://www.maxim-ic.com/datasheet/index.mvp/id/1896
[6] http://www.maxim-ic.com/datasheet/index.mvp/id/3653

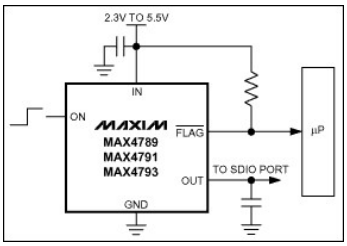


Abbildung 4.6: Strombegrenzer IC MAX4790 und externe Beschaltung (aus dem Datenblatt des MAX4790)

4.4.4 Versorgung

Die Implementierung der Stromversorgung wurde mit einem bereits häufig verbauten DC-DC Konverter der Firma Texas Instruments durchgeführt. Der TPS5410 IC ist für die Anwendungen im Automobil entwickelt, benötigt ebenfalls nur wenig externe Beschaltung (siehe Abbildung 4.7) und zeichnet sich durch hohe Effizienz aus.

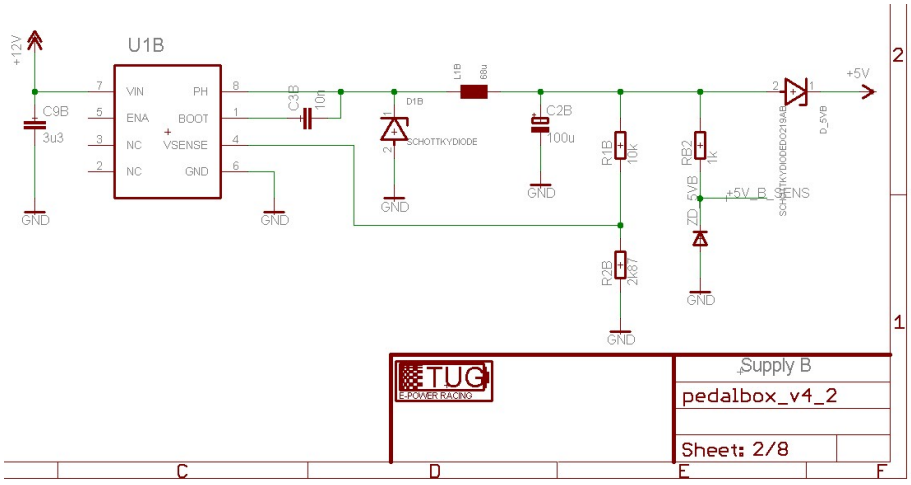


Abbildung 4.7: Schematic der Implementierung der Stromversorgung mit TPS5410

Um die Stromversorgung richtig dimensionieren zu können, ist es notwendig, den Strombedarf der angeschlossenen Sensorik und Elektronik im schlimmsten Fall abzuschätzen und den DC-DC Konverter richtig auszulegen. Die Abschätzung des „Worst-case" Falls ist in Tabelle 4.2 ersichtlich. Mit Hilfe dieser Werte und den geforderten Einsatzrahmenumgebungen lassen sich dann die externen Einzelkomponenten exakt bestimmen. Die notwendigen Formeln und auch wichtige Hinweise zum Design der Leiterplatte findet man im Datenblatt des TPS5410 [7]. Der geforderte Spannungsbereich, für den die Versorgung voll funktionsfähig sein muss, liegt bei 11...14 V und der im Automobilbereich üblichen Umgebungstemperatur von $-40... + 85°C$ und Vibrationen zwischen 10 Hz und 1 kHz.

Aus einem simulierten Belastungstest der Beschaltung, der auf der Herstellerseite des

[7] http://focus.ti.com/docs/prod/folders/print/tps5410.html

ICs [8] online durchgeführt werden kann, ist ersichtlich, dass der IC selbst und die Schottky Diode D1B die höchste Strombelastung und Temperaturbelastung aller Bauteile aushalten müssen und daher die wahrscheinlichsten Gründe für einen Ausfall der Versorgung sein könnten.

Komponente	**max. Strom** $[mA]$
Microcontroller AT90CAN (2 Stück)	60
CAN Transceiver (2 Stück)	40
OPs & Entpreller	30
linear Potentiometer (1 Stück)	10
Winkelsensoren (3 Stück)	120
Drucksensoren (2 Stück)	20
sonstige Hardware	60
abgeschätzte Maximalstromaufnahme	340
zusätzliche Sicherheit für Abschätzungsfehler	100
geschätzter Stromverbrauch für Auslegung	440 mA

Tabelle 4.2: Worst-case-Abschätzung der Stromaufnahme der Einzelkomponenten der Implementierung

4.5 Entwurf der Software

Mit der Auswahl des Microcontrollers wird gleichzeitig auch festgelegt, welche Softwaretools und Programmiersprachen beim Entwurf der Software zur Verfügung stehen. Wie schon unter 4.4.1 erwähnt, wurde die Microcontroller Familie AT90CAN ausgewählt. Für diesen Controller eignet sich die Programmiersprache C sehr gut und wird auch vom Hersteller mit Programmiertools wie AVR Studio [9] unterstützt. Zusätzlich bietet die große AVR Community zahlreiche Compiler [10], Tutorials [11] und Programmbeispiele an, die einen raschen Einstieg ermöglichen.
Des Weiteren stehen über das ATMEL Hochschulprogramm [12] Hardware und Programmiergeräte relativ rasch und günstig zur Verfügung. Ein weiterer Vorteil ist das Vorhandensein von „proven in use" Softwareprogrammen und der Einsatz dieses Microcontrollers in anderen Systemen des MaxWheel 2010.

Abbildung 4.8 zeigt den grundsätzlichen Programmablauf des Drive-by-Wire Systems. Das Programm ist bewusst einfach gehalten und frei von Echtzeitbetriebssystemen oder Multithreading. Das Programm wird in einer Endlosschleife abgearbeitet und kann im normalen Betrieb nicht durch externe Ereignisse gestört werden. Dies reduziert die Buslast, stellt somit die Echtzeitfähigkeit des CAN sicher und verhindert unterschiedliche Verzögerungszeiten beim Signalaustausch über das Anfrage-Antwort Prinzip. Eine Reparametrierung oder Änderung der Kenngrößen kann über CAN Nachrichten damit nur in genau definierten Systemzuständen (z.B. kein Antrieb am Fahrzeug) erreicht werden. Beispiele hierfür sind:

[8] http://www.ti.com/switcherpro
[9] http://www.atmel.com/dyn/products/tools_card.asp?tool_id=2725
[10] http://winavr.sourceforge.net/
[11] http://www.mikrocontroller.net/articles/AVR-GCC-Tutorial
[12] http://www.eproo-student.de/

- Festlegen neuer Pedalkennlinien
- Parametrieren von Sensoren
- Aufheben einzelner Sensorinformationen

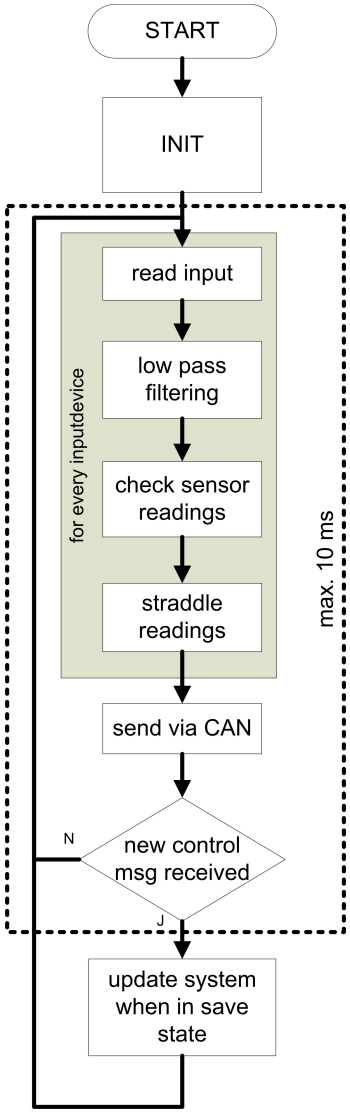


Abbildung 4.8: Flussdiagramm Softwareentwurf

Um Softwarefehler zu erkennen und entgegen zu wirken, sind eine Reihe von Softwaretests (siehe Abschnitt 3.4.2) notwendig. Die Tests der einzelnen Softwaremodule können größtenteils getrennt voneinander auf der Testimplementierung durchgeführt werden, wie auch die späteren Integrationstests der Teilsysteme.
Das Reglement FSUK (A.2.6) fordert des Weiteren die Einhaltung der MISRA C Richtlinien, die ebenfalls zur Verminderung von Softwarefehlern beitragen. Letzte Softwareprobleme und Interface-Schwierigkeiten können vor dem Einsatz im Fahrzeug noch am Prüfstand (Abbildung 4.10) unter Laborbedingungen und ohne Risiko behoben werden.

4.6 Erste Testimplementierung

Die hier vorgestellte erste Testimplementierung dient in erster Linie dazu, Fehler im Schaltungsentwurf zu erkennen und mögliche Schwachstellen der Implementierung festzustellen. Die in Abbildung 4.9 gezeigte Testimplementierung ist weiters auch die erste Evolutionsstufe der geforderten Perfektionierung des Schaltungsentwurfs (4.1.3), die zur einfacheren Erstellung der im Reglement (A.1.2 und A.2.6) geforderten FMEA (siehe 4.7) beiträgt und eventuelle Unstimmigkeiten zwischen Reglement und Schaltungsentwurf schnell aufdeckt.

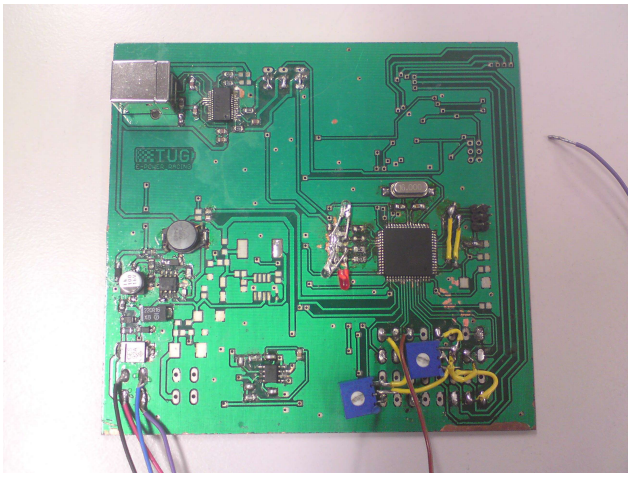


Abbildung 4.9: erste Testimplementierung der X-by-Wire-Elektronik des MaxWheel 2010

Im zweiten Schritt können auf dieser Testimplementierung neue Softwarekomponenten und Ansätze unabhängig vom Fahrzeug und von Testmöglichkeiten rasch am eigens konstruierten Motorprüfstandsaufbau (siehe Abbildung 4.10) getestet werden.

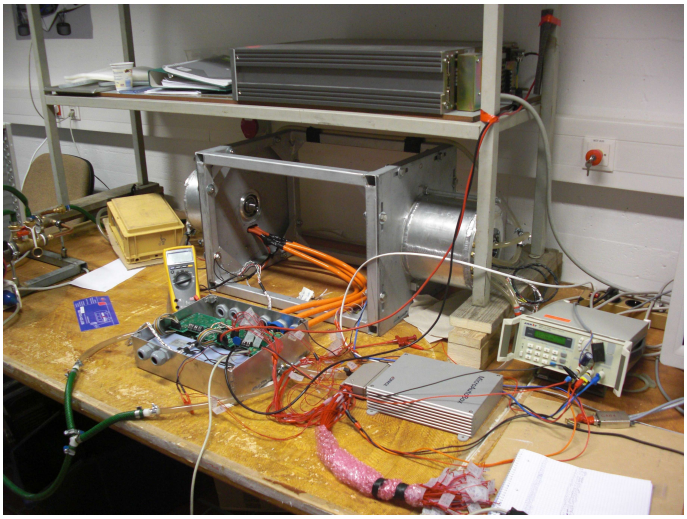


Abbildung 4.10: selbst konstruierter Motorprüfstand

4.7 Reglententsprechende Risikoabschätzung

Wie schon im vorigen Abschnitt erwähnt, ist mit Hilfe der ersten Testimplementierung eine Fehleranalyse entsprechend den gültigen Bewerbsreglements zu erstellen und daraus eventuell entstehende Änderungen der Implementierung vorzunehmen. Die hier vorgestellten FMEA Analysen sind Teil des Formula Student Reglements und nicht von der Art wie in Abschnitt 3.4.1 vorgestellt.

Risk Assessment entsprechend Reglement FSUK A.2

System Element	Keyword	Consequence description	Consequence identifier	Notes
Accelerator Pedal	High	Accelerator pedal output too high for a given position and vehicle accelerators too quickly.	POOR DRIVEABILITY	
	Low	Accelerator pedal output too low for given position and vehicle accelerators too slowly.		
	Random	Vehicle behaviour erratic		
	Early	Vehicle accelerates unexpectedly with driver inside	VEHICLE UNCONTROLLABLE	
	Late	Vehicle still accelerates when pedal no longer pressed		
	Other	Pedal output is open or short circuit	PEDAL MEASUREMENT NOT VALID	
	More, less	N/A		
Pedalbox	High, Low, Random, Early, Late, Other, More, Less	CAN Bus defective	REDUNDANCY EXISTS	
ECU	High	All PWM Outputs remain high, Motordriver Short-Circuit Protection goes into action	VEHICLE FREEWHEELING	
	Low	All PWM Outputs remain low, no power to the engines		

Tabelle 4.3: Auszug der Risk Assessment Tabelle für FSUK

Die Tabelle 4.3 zeigt, welche Aufschlüsselung laut dem Reglement des Englandbewerbs notwendig ist. Hierbei muss lediglich ein Nachweis erbracht werden, dass alle Risiken in den Teilsystemen zwischen angetriebenem Reifen und Bedienelement des Fahrers bedacht und ein definierter Zustand erreicht wird.

Die Risiken sind dabei nach den folgenden Schlagwörtern aufzuteilen:

high	Werte (Spannungen/Ströme/Leistungen/Nachrichten) sind höher als erwartet/gefordert
low	Werte sind niedriger als erwartet/gefordert
random	Werte treten zufällig und unbeeinflussbar auf
early	Werte sind vor ihrem definierten Gültigkeitszeitpunkt verfügbar
late	Werte sind erst nach ihrem definierten Gültigkeitszeitpunkt verfügbar
more	mehr Werte als gefordert stehen zur Verfügung
less	weniger Werte als notwendig stehen zur Verfügung
other	alle anderen Faktoren

Nach dieser Aufteilung muss eine Beschreibung der Konsequenz und des Standes angeben werden, in dem sich das Fahrzeug nach Auftreten dieses Fehlers befindet. Legitime Fahrzeugzustände dabei sind:

poor driveability	kein sicherheitsrelevantes Problem, Fahrzeug kann von trainiertem Fahrer gefahren werden
vehicle uncontrollable	mechanische Bremse funktioniert, Fahrer kann Notstopp durchführen und HV System abschalten
pedal measurement not valid	fehlerhafter Messwert kann festgestellt werden und ein fehlersicherer Zustand kann erreicht werden
redundancy exists	redundantes Teilsystem funktioniert weiter, kein sicherheitsrelevantes Problem
vehicle freewheeling	keine Versorgung des Elektromotors, kein Antrieb, Fahrer kann Notstopp durchführen
vehicle normal operation	alle Systeme funktionieren wie spezifiziert

Anhand dieser Risikoabschätzung und der Systemarchitektur des Fahrzeugs wird dann die technische Abnahme erteilt oder verweigert. Die Systemarchitektur des MaxWheel 2010 ist in den Abbildung 5.1 und 5.2 ersichtlich.

FMEA entsprechend dem FSG Reglement

Die FMEA, die es entsprechend dem FSG Reglement einzureichen gilt (A.1.2), entspricht der unter 3.4.1 vorgestellten FMEA im Wesentlichen, beinhaltet aber keine Abschätzung des Risikopotentials der einzelnen Fehler und keine Lösungsansätze, um den einzelnen Fehlerquellen entgegen zu wirken. Tabelle 4.5 zeigt die wesentlichen Punkte, die es für das X-by-Wire System zu klären gilt.

Beim Aufstellen der FMEA Tabelle wird schnell ersichtlich, dass weitere Anforderungen an das X-by-Wire System gestellt werden müssen:

Aufgrund der Tatsache, dass nur zwei Sensoren pro Pedal verbaut werden können, führen Differenzen zwischen den beiden Messwerten (siehe Tabelle 4.4) zu einem sofortigen Abschalten des Elektroantriebs und damit zu einem Ausfall des Fahrzeugs. Daher würde der Ausfall eines Sensors zu einem Gesamtausfall des redundanten Systems führen und daher das System gegenüber diesem Fehler nicht redundant ausgeführt sein. Ein Sensor muss daher als fehlerhaft kennzeichenbar sein (wie auch in 4.1.2 gefordert). Allerdings können Limitchecks und Plausibiltätstest der beiden Messsignale nicht erkennen, welcher Sensorwert im Fall D in Tabelle 4.4 den richtigen Messwert liefert. Dieser Fall kann mit Endpositionsgebern an den Pedalen erkannt werden und der fehlerhafte Sensor ermittelt werden.
Aufgrund der höheren Anforderung an die Wettbewerbsfähigkeit und Ausfallsicherheit des Boliden muss auch im Fall D ein Ausfall verhindert werden und das Fahrzeug weiterhin voll funktionsfähig bleiben. Die Überprüfung der Richtigkeit des verbleibenden Sensors muss dann über die Endpositionsgeber gewährleistet werden.

Ein weiterer Fall, den es abzusicheren gilt, ist die Verifizierung aktueller Werte. Um dem Fehler eines „steckenden Pedals" entgegenzuwirken, sind die gemessenen Werte mit einem Zeitstempel zu kennzeichnen. Daher ist ein Alive-Counter und ein maximaler Delay zwischen den redundanten Systempfaden zu fordern.

Des Weiteren kann die redundante Ausführung des Lenkwinkelsensors wegfallen. Sensorwerte des Lenkwinkelsensors können mit Hilfe des verbauten Gierratensensors verifiziert werden.

Fall	Sensorwert A	Sensorwert B	Bedingung	Auswirkung ohne Plausibilisierung und Endpositionsgeber	Auswirkung mit Plausibilitätscheck	Auswirkung mit zusätzlichen Endpositionsgebern
A	$0,5...4,5\,V$	$0,5...4,5\,V$	$A = B$	voll funktionsfähig	voll funktionsfähig	voll funktionsfähig
B	$5\,V$	$0,5...4,5\,V$		Abschaltung des Antriebs	fehlerhafter Sensor wird erkannt, funktionsfähig ohne Verifizierung des verbleibenden Sensors	fehlerhafter Sensor wird erkannt, volle Funktion, Verifizierung des verbleibenden Sensors über Endpositionsgeber
C	$0\,V$	$0,5...4,5\,V$		Abschaltung des Antriebs	fehlerhafter Sensor wird erkannt, funktionsfähig ohne Verifizierung des verbleibenden Sensors	fehlerhafter Sensor wird erkannt, volle Funktion, Verifizierung des verbleibenden Sensors über Endpositionsgeber
D	$0,5...4,5\,V$	$0,5...4,5\,V$	$A! = B$	Abschaltung des Antriebs	fehlerhafter Sensor nicht erkennbar, Abschaltung des Antriebs	fehlerhafter Sensor wird erkannt, voll funktionsfähig, Verifizierung des verbleibenden Sensors über Endpositionsgeber

Tabelle 4.4: Auswirkungen unterschiedlicher Sensorwertkombinationen auf verschiedene Implementierungen

Car No.:	**E15**	University:	**Graz University of Technology**				
FMEA No.	**Component /Item**	**Function**	**Failure Mode**	**Failure Cause**	**Failure Effect** local	**Failure Effect** global	**Failure Detection**	**Failure Handling**
6	Torque Encoder	Signaling the pedal position	Sensor 1 and Sensor 2 deliver different position values	Sensor failure in Sensor 1, 2 or both, mechanical linkage failure, wiring failure	divergent input of driver torque request to DBW system	potentially dangerous due to unwanted acceleration if driver request is lower than the determined value	continous monitoring and comparison of both signals on ECU	immediate motor shutdown by ECU
7	Torque Encoder	Signaling the pedal position	Sensor 1 or Sensor 2 signal not plausible	Sensor failure in Sensor 1 or 2, mechanical linkage failure, wiring failure	divergent input of driver torque request to DBW system	potentially dangerous due to unwanted acceleration if driver request is lower than the detremined value	continous monitoring of both signals on ECU	offending sensor is marked as 'invalid' and subsequently ignored.
8	Torque Encoder	Signaling the pedal position	Sensor 1 or Sensor 2 no signal	Sensor failure in Sensor 1 or 2, mechanical linkage failure, wiring failure	divergent input of driver torque request to DBW system	potentially dangerous due to unwanted acceleration if driver request is lower than the detremined value	continous monitoring of both signals on ECU	offending sensor is marked as 'invalid' and subsequently ignored.
16	Torque Encoder	Signaling the pedal position	Sensor 1 or Sensor 2 internal electrical failure	Sensor failure	Sensor may short-circuit Sensor or DBW supply	Potentially dangerous if DBW failure causes unwanted/ uncontrolled acceleration	DBW electronics detect excessive current consumption	DBW electronics shut down the supply for the faulty sensor
28	DBW System	Torque encoder evaluation/ interpretion	system failure	internal failure, supply failure, wiring failure	no up-to-date Torque encoder data available	Potentially dangerous in case of unwanted / uncontrolled acceleration ('stuck pedal' effect)	detection by ECU by either failure receive data or by failure alive-counter	

Tabelle 4.5: Auszug der FMEA Tabelle für FSG

Kapitel 5

Implementierung

> The two systems must not share any components (such as sensors, actuators or electronic control boxes).

Aufgrund dieser Regel im Regelwerk des Formula Student UK Bewerbs [19] und dem Fehlen einer vergleichbaren Restriktion im FSE Reglement [13] können zwei unterschiedliche Implementierungen vorgenommen werden. In den Abschnitten 5.2 und 5.3 werden die beiden Varianten beschrieben.
Unterschiede und Vergleiche der Implementierungen werden in Abschnitt 5.4 angegeben.
Um auch eine bessere Übersicht zu den Strukturen der im Boliden verbauten Systeme zu bekommen, wird im folgenden Abschnitt die Systemarchitektur des MaxWheel 2010 kurz dargestellt.

5.1 Systemarchitektur des MaxWheel 2010

Die Abbildungen in diesem Abschnitt zeigen zusammengefasst den Aufbau der im MaxWheel verbauten Systeme und ihre Verbindungen untereinander.
Dieser Abschnitt dient der Übersicht der Fahrzeugsystemarchitektur, in die das Drive-by-Wire System zu implementieren ist.
Abbildung 5.1 zeigt den funktionalen Zusammenhang aller elektrischen Teilsysteme der Hochvoltelektronik und des Antriebsstrangs.
In Abbildung 5.2 sind alle elektrischen Teilsysteme angeführt, die nicht zur Hochvoltelektronik gehören, der Stromlaufplan des Interlocks und die Sicherheitseinrichtungen im MaxWheel 2010.
Die Aufteilung der CAN Teilnehmer auf die 4 physikalischen CAN Busse des Motorsteuergeräts ist in Abbildung 5.3 ersichtlich.

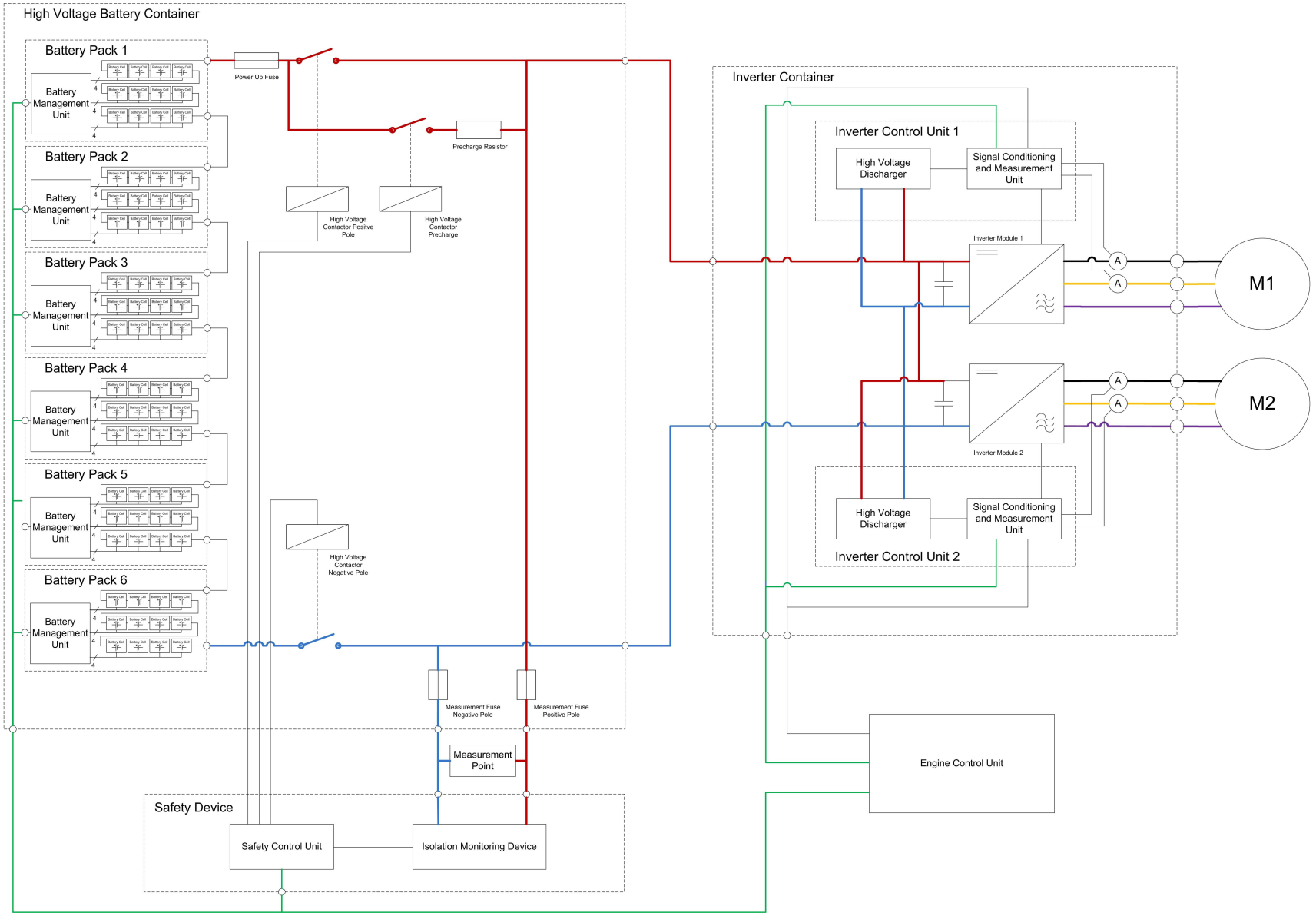


Abbildung 5.1: Tractive System Architektur des MaxWheel 2010

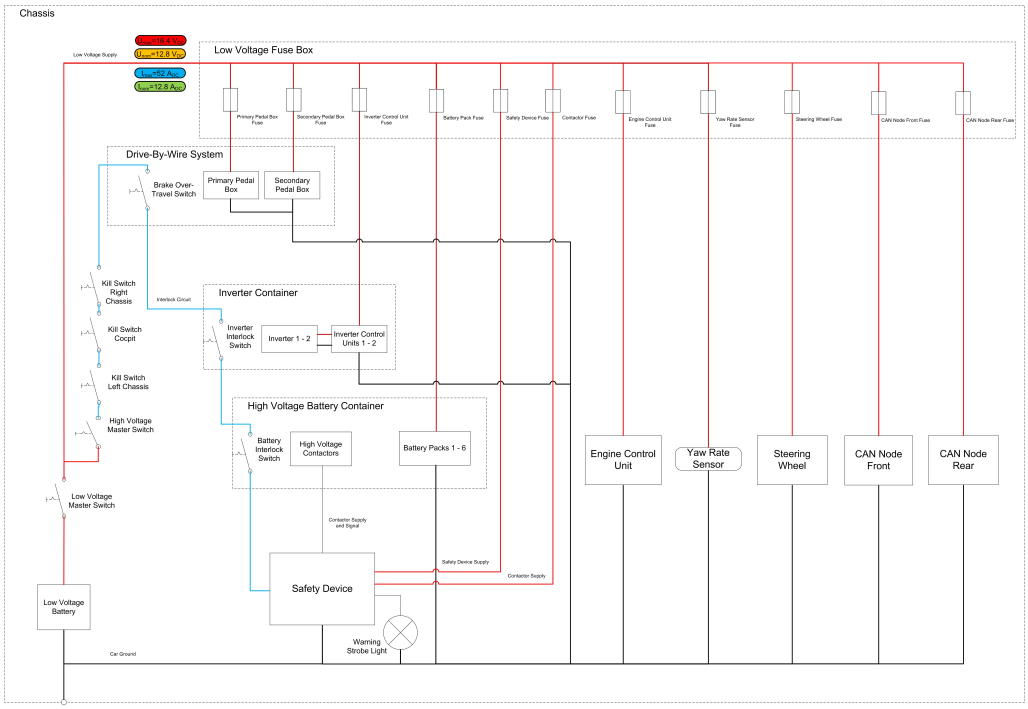


Abbildung 5.2: Architektur des Interlock und LV Systems des MaxWheel 2010

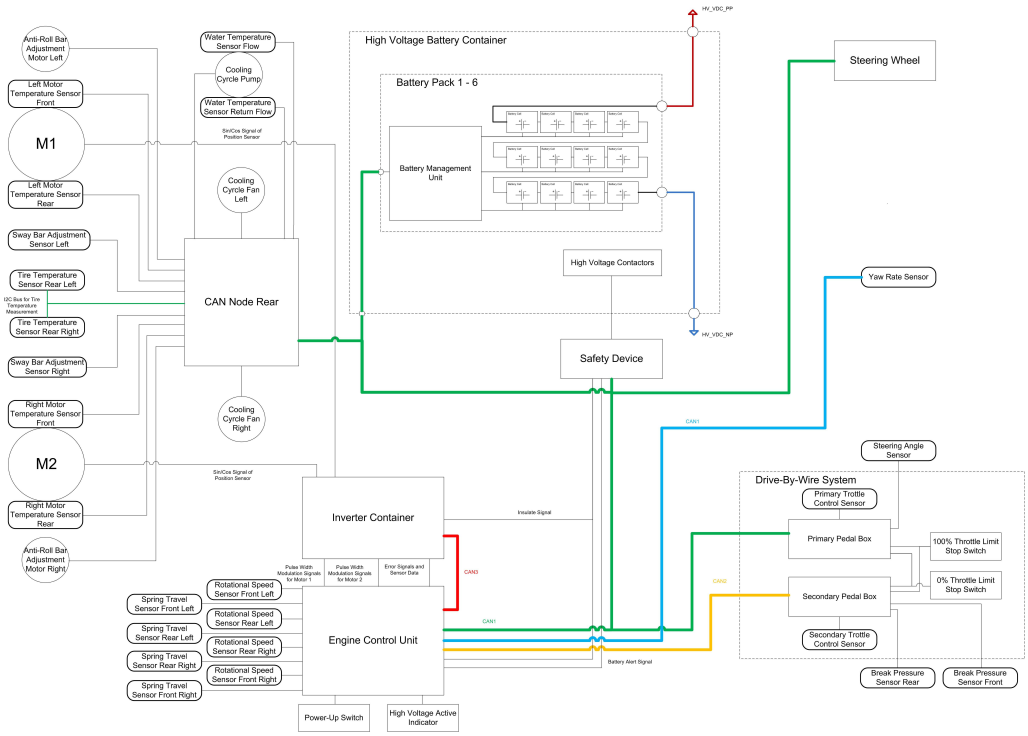


Abbildung 5.3: Sensorsignal- und Buspfade des MaxWheel 2010

5.2 MaxWheel 2010 Drive-by-Wire System Version I

Version I des Drive-by-Wire Systems des MaxWheel 2010 ist zum FSUK Reglement konform und verfügt über keinerlei gemeinsame Komponenten der beiden redundanten Kanäle. Das Systems ist, wie in Abbildung 5.4 skizziert, über zwei baulich getrennte FSU Einheiten realisiert. Das zugehörige Blockschaltbild der FSU Einheiten ist bereits aus dem Systementwurf (Abschnitt 4.4, Abbildung 4.5) gegeben.

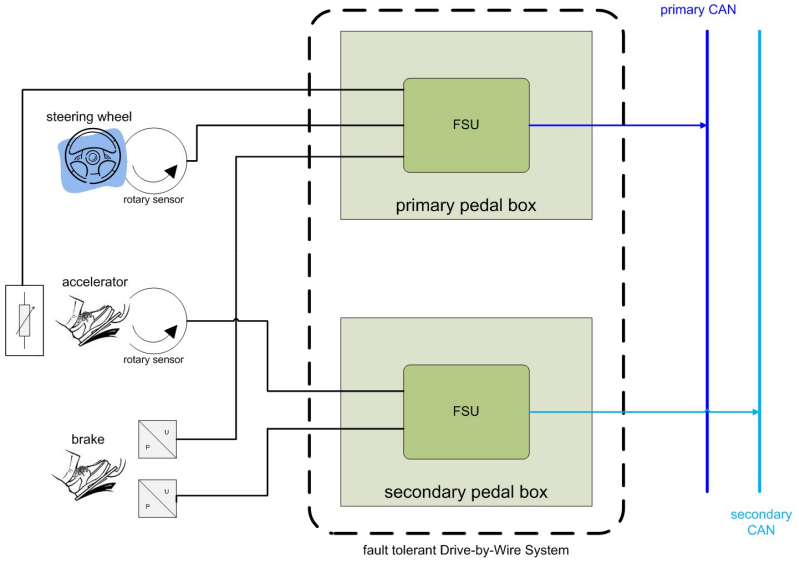


Abbildung 5.4: Blockschaltbild Drive-by-Wire System Version I

5.2.1 FTA des DBW Systems Version I

Der in Abbildung 5.5 dargestellte Fehlerbaum schlüsselt alle aufgetretenen Fehlerursachen einer FSU auf und kann als Reparaturleitfaden eingesetzt werden. Die aufgetretenen Fehlerursachen sind nach der Häufigkeit der Auftritte sortiert. Die häufigsten Ursachen jeder Ebene stehen oben. Der Teilbaum der zweiten FSU des Drive-By-Wire Systems sieht exakt gleich aus und ist wegen der besseren Übersicht zusammengelegt worden.

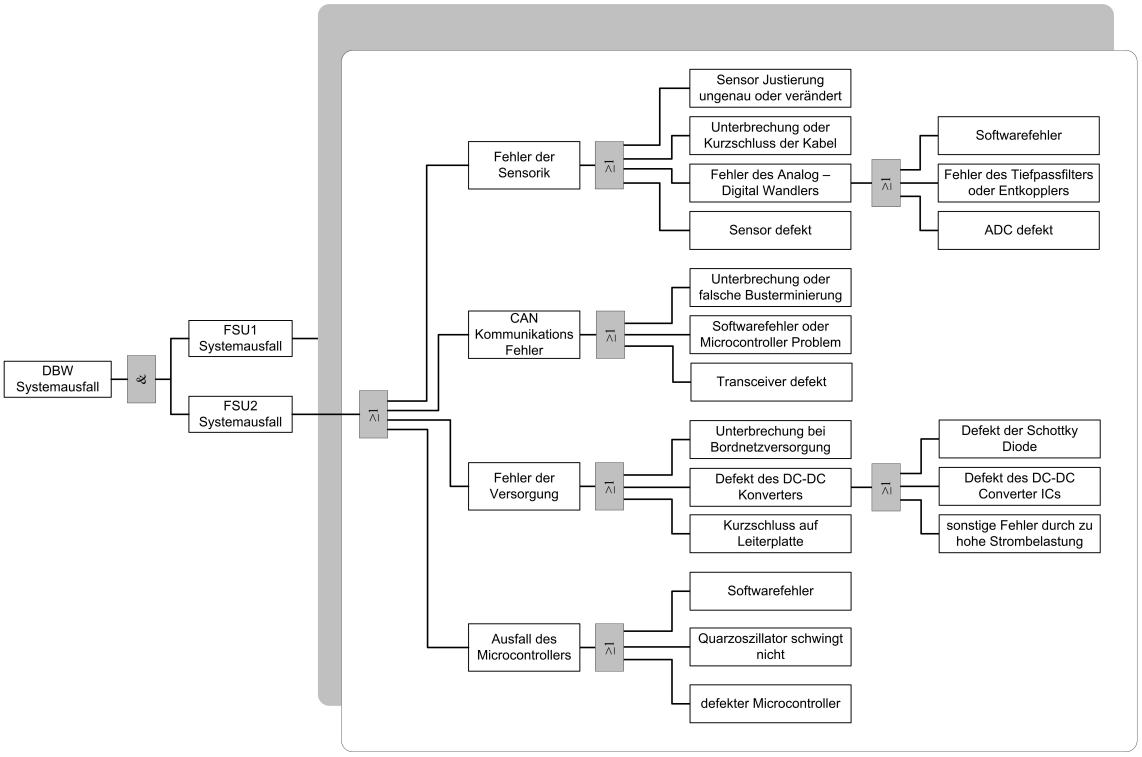


Abbildung 5.5: Fehlerbaum des DBW Systems Version I

5.3 MaxWheel 2010 Drive-by-Wire System Version II

Diese Implementierung des Drive-by-Wire Systems ist ebenfalls über zwei FSU Einheiten, die gemeinsam eine fehlertolerante Einheit bilden, verwirklicht. Die Vorteile dieser Implementierung sind rasch ersichtlich:

- redundante Spannungsversorgung
- entkoppelte Zuführung beider redundanter Sensoren
- Endpositionsschalter zur Verifizierung der Sensorwerte
- leichtere Fehlererkennung über beide Sensorwerte
- alle angeführten Punkte für beide Teilsysteme entkoppelt

Nur diese Variante des DBW Systems erlaubt es auch, den Fall D der Tabelle 4.4 ausfallsfrei abzudecken. Abbildung 5.6 zeigt das Blockschaltbild dieser Implementierungsvariante.

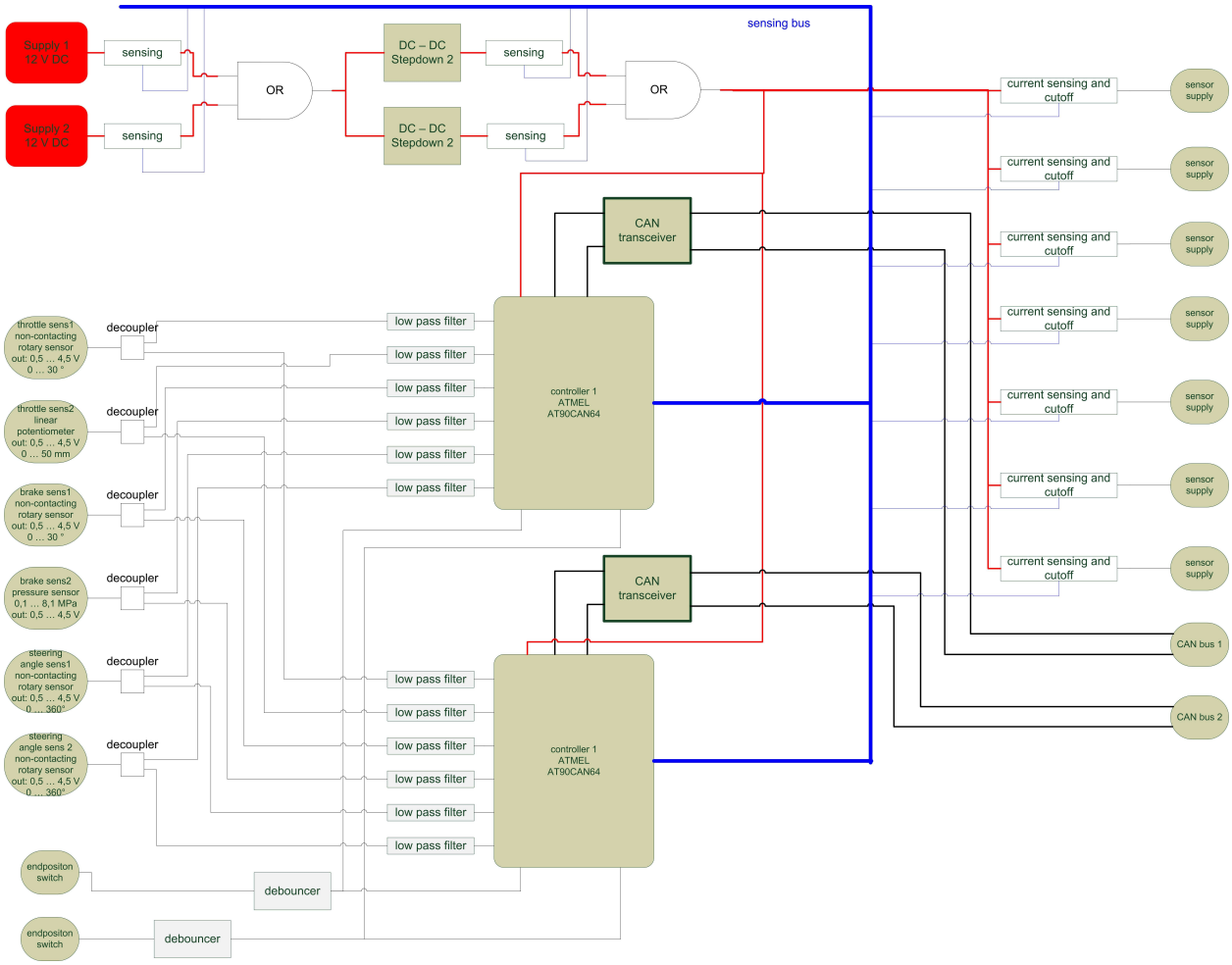


Abbildung 5.6: Blockschaltbild Drive-by-Wire System Version II

5.3.1 FTA des DBW Systems Version II

Auch bei der Betrachtung des Fehlerbaums 5.7 ist festzustellen, dass sich Redundanz auf mehreren Ebenen wiederfindet (gekennzeichnet durch die Schatten) und das Gesamtsystem viel fehlertoleranter als die erste Implementierungsvariante ausfällt.
Auch bei diesem Fehlerbaum sind die Fehlerursachen nach ihrer Auftrittshäufigkeit gereiht. Die häufigsten Fehlerursachen sind jeweils im linken Teilbaum zu finden.

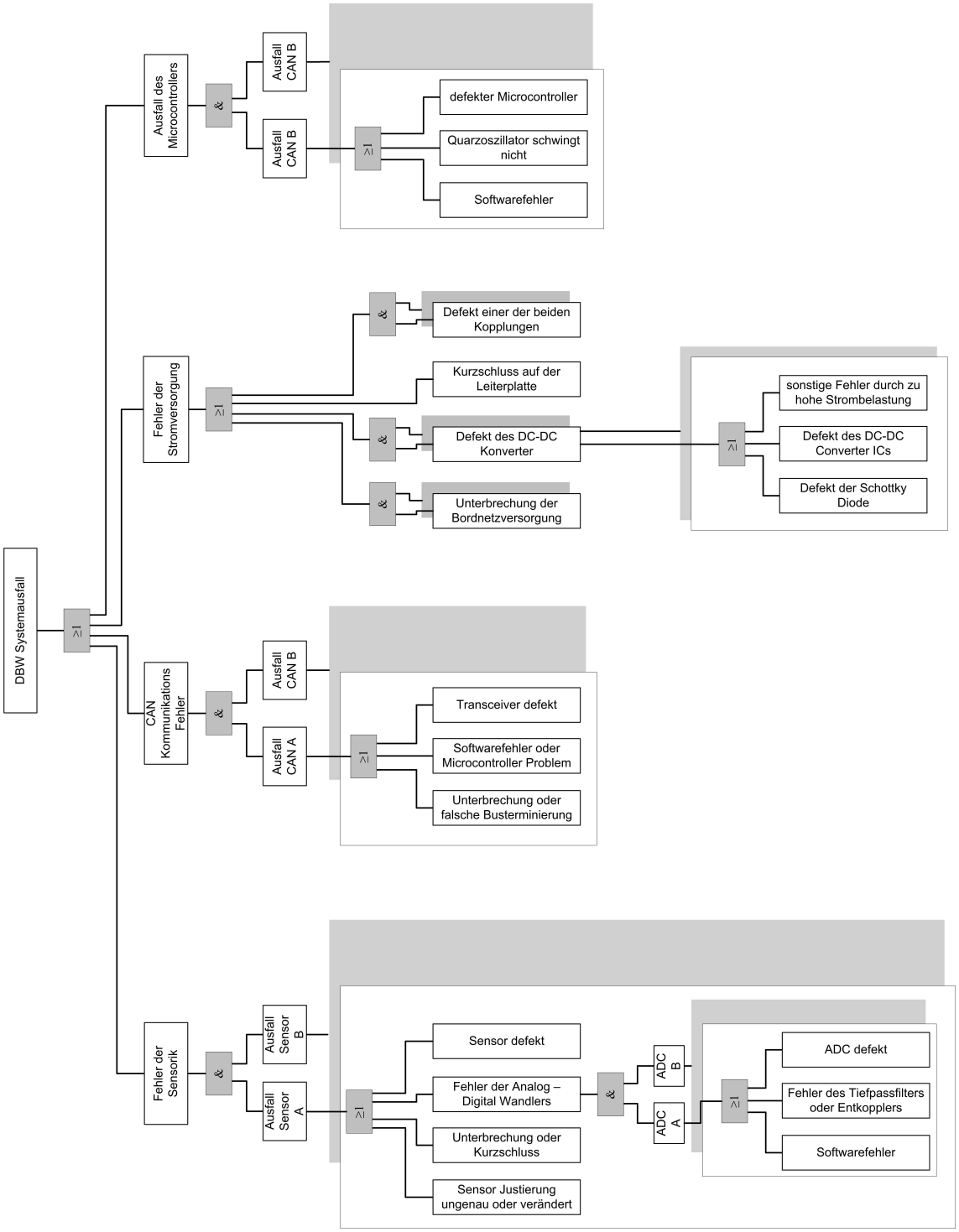


Abbildung 5.7: Fehlerbaum des DBW Systems Version II

5.4 Vergleich der Implementierungsvarianten

Die beiden Blockschaltbilder der Implementierungsvarianten 5.4 und 5.6 lassen bereits erkennen, dass sich Variante II wesentlich von Variante I unterscheidet, was die Fehlertoleranz betrifft. Worin genau der Unterschied zwischen den beiden Implementierungen liegt, soll in diesem Abschnitt geklärt werden.

Zuerst kann man erkennen, dass Variante II aus zwei Implementierungen der Variante I auf einer Platine aufgebaut ist. Welche Vorteile dies bietet, wurde bereits kurz in Abschnitt 5.3 erwähnt, wird hier aber genauer ausgeführt.

1. Redundante Spannungsversorgung

   Die beiden redundanten Spannungsversorgungen stehen beiden Teilsystemen zur Verfügung und können jeweils alleine die Stromlasten des Gesamtsystems übernehmen. Die Verknüpfung der beiden Versorgungsschaltungen ist so implementiert, dass ein Ausfall einer Versorgungseinheit keinen Einfluss auf das Gesamtsystem haben kann und die benötigte Leistung unterbrechungsfrei bereitgestellt wird. Für einen Ausfall der Spannungsversorgung müssten beide Versorgungseinheiten gleichzeitig ausfallen (siehe Fehlerbaum 5.7).

2. Entkoppelte Zuführung beider redundanter Sensoren

   Während bei Variante I jedem Teilsystem nur ein Sensorwert pro Fahrerwunschaufnehmer zur Verfügung steht, stehen bei Variante II alle Sensorwerte jedem Teilsystem entkoppelt zur Verfügung. Dies erlaubt es jedem redundanten Teilsystem, den aufgenommenen Messwert zu verifizieren und eine Plausibilisierung der zu übertragenden Werte durchzuführen.

3. Endpositionsschalter zur Verifizierung der Sensorwerte

   Die zusätzliche Anbringung von Endpositionsschaltern an den Pedalen erlaubt es, die Sensoren während des Betriebs gegebenenfalls nachzujustieren, bzw. bei unterschiedlichen Sensorwerten den fehlerhaften Sensor zu erkennen. Da diese Endpositionsschalter aufgrund des geringen Platzangebotes nicht doppelt aufgebaut werden konnten und wegen des Reglements, das bei Variante I eingehalten werden muss, ist auch diese zusätzliche Einrichtung nur bei Variante II verfügbar.

4. Geringerer Platzbedarf

   Aufgrund der geforderten Aufteilung von Variante I und des nahezu gleichbleibenden Hardwareaufwands beider Implementierungen kann Variante II mit geringerem Platzbedarf eingesetzt werden.

5. Austauschbarkeit der Implementierungen

   Die beiden Implementierungsvarianten sind aufgrund der gleichen Funktion, Sensorik und Pinbelegungen jederzeit gegeneinander austauschbar. Auch der Unterschied

auf Softwareebene betrifft nur einige wenige Funktionen, was zu einer nur unwesentlichen Erhöhung des Testbedarfs führt (siehe 5.8).

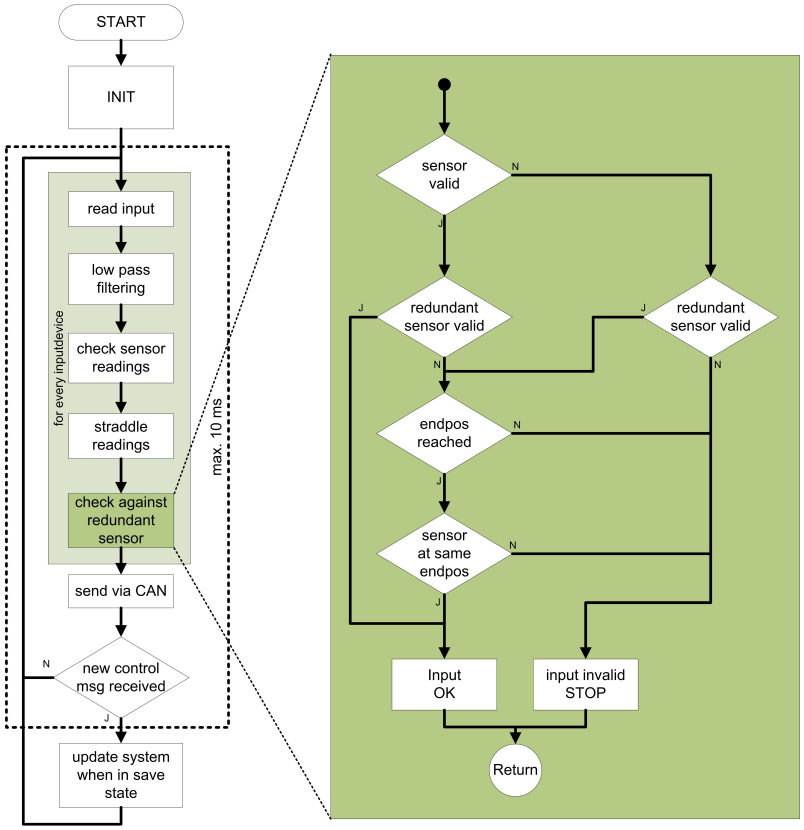


Abbildung 5.8: Software Unterschiede der beiden Implementierungsvarianten

5.4.1 Vergleich mittels System FMEA

Die Unterschiede und Vorteile der Implementierungsvariante II sind aus der System FMEA Tabelle C.3 auch sehr einfach ersichtlich.
Bei dieser FMEA Tabelle wird zur Klassifizierung des Risikopotentials eines Fehlers nur eine Einstufung über die Wahrscheinlichkeit des Fehlers und die Schwere der Auswirkung getroffen. Da aufgrund des Prototypenstatus der Systeme für die meisten Komponenten keine Zuverlässigkeitskenngrößen vorliegen, ist diese Einstufung auch nur sehr rudimentär durchgeführt. Auf eine Klassifizierung der Schwere der Fehlererkennung wird ebenfalls aufgrund des Prototypenstatus und der ständigen Anwesenheit der Entwicklungscrew beim Betrieb des Boliden verzichtet.

Die Einteilung der einzelnen Risikopotentiale erfolgt nach folgenden Punkten:

Wert	**Auftrittswahrscheinlichkeit**	**Wert**	**Schwere der Auswirkung**
1	sehr unwahrscheinlich	1	unerheblich für Renneinsatz
2	unwahrscheinlich	2	
3	Problem bereits einmal aufgetreten	3	wesentliche Einschränkung, Ausfall kann vom Fahrer verhindert werden
4			
5	Problem bereits mehr als einmal aufgetreten	4	
6		5	
7	Problem öfters aufgetreten	6	
8		7	
9		8	
10	Auftritt während Renneinsatz ist zu erwarten	9	
		10	sicherer Ausfall im Rennen

Tabelle 5.1: Einteilung der Risikopotentiale

Aus der FMEA Tabelle C.3 ist ersichtlich, dass Probleme einer der redundanten Spannungsversorgungen oder der Ausfall eines Sensors bei Variante II zu keiner Einschränkung während des Renneinsatzes führen. Diese Fehler werden erkannt und können beim nächsten Fahrzeugservice behoben werden.
Der Ausfall beider Teilsystem hat natürlich bei beiden Implementationsvarianten einen Ausfall des Boliden zur Folge.
Während ein Ausfall eines Teilsystems bei Implementierung I in jedem Fall das aktive Einschreiten des Fahrers erfordert (Schwere der Auswirkung 3), ist dies nur in Einzelfällen bei Implementierung II erforderlich. Weiters sind die Werte der Auftrittswahrscheinlichkeit der einzelnen Fehler für beide Varianten gleich angesetzt worden, was aber schon aufgrund der unterschiedlichen Anordnung im Fahrzeug nicht garantiert werden kann. Wie aber bereits zuvor angemerkt, können qualifizierte Aussagen zu solchen Abschätzungen des Prototypen nicht getroffen werden und sind daher zur besseren Vergleichbarkeit der Implementierung gleichgesetzt worden.

Kapitel 6

Zusammenfassung und Ausblick

In diesem letzten Kapitel des Dokuments sind die Ergebnisse der Arbeit und der Implementierung zusammengefasst. Im Abschnitt 6.2 werden noch kurz Ideen und Änderungsvorschläge angegeben, die bei einer Weiterentwicklung des Projekts bedacht werden sollten.

6.1 Zusammenfassung

Mit fortschreitenden Entwicklungen von fehlertoleranter Elektronik und Softwareentwicklungsmethoden wird auch im Automobilsektor der Einsatz von X-by-Wire Systemen immer verbreiteter. Wie zu Beginn dieses Dokuments kurz angerissen, sind jedoch erhebliche Auflagen und Normen für den Einsatz von X-by-Wire Systemen in Fahrzeugen des Straßenverkehrs einzuhalten. Dies ist einer der Gründe, warum nach wie vor mechanische Rückfallebenen in den meisten Fahrzeugen im Einsatz sind, nichtsdestotrotz werden bereits ehrgeizig Hard- und Softwarekomponenten von Autozulieferern entwickelt.

Hohe Verfügbarkeit und Ausfallsicherheit sind die Hauptmerkmale, die ein X-by-Wire System zu besitzen hat.
Obwohl viele Assistenz- und Fahrsicherheitsprogramme überhaupt nur mit rein elektrischen X-by-Wire Systemen verwirklichbar sind, sind dennoch die Verfügbarkeit und Ausfallsicherheit der mechanischen Lösungen die Mindestanforderungen, die solche Systeme erfüllen müssen.
Neue Reglementierungen und Normen müssen erst ausgearbeitet und fertiggestellt werden, um ein weiteres Fortschreiten dieser Entwicklung zu unterstützen und Systeme ohne mechanische Rückfallebene für den Straßenverkehr zuzulassen.
Auch die Einführung von Fahrzeugen mit rein elektrischem Antrieb weist verstärkt in Richtung von X-by-Wire Applikationen.

Das hier vorgestellte Drive-by-Wire Konzept arbeitet ebenfalls noch mit einer mechanischen Rückfallebene, muss sich allerdings nicht den rechtlichen Vorschriften einer Kundenanwendung beugen. Dieses Drive-by-Wire Konzept ist für Rennboliden mit Elektroantrieb der Formula Student Serie konzipiert und unterliegt nur dem Reglement dieser Serie und den Anforderungen des Rennsporteinsatzes.

Das Ergebnis dieses Projekts ist ein Drive-by-Wire System, das hohe Zuverlässigkeit und Verfügbarkeit mit einfachen Massenelektronikbauteilen erzielt. Beim Design und der Entwicklung ist speziell auf Themen, wie Wartbarkeit, Ausfallsicherheit und Fehlertoleranz eingegangen worden. Das Gesamtsystem ist mit im Automobilsektor etablierten Verfahren, Tools und Methoden konzipiert und in nur 6 Monaten vom Projektstart bis zur Implementierung im Rennboliden fertiggestellt worden.
Das Durchfahren aller gestarteten Endurance Rennen der FSE Saison 2010 sowie das durchwegs positive Feedback der einschlägigen Fachjury bei den Design Events zeigen von hoher Zuverlässigkeit und guter Implementierung des Erstjahresfahrzeugs und des Drive-by-Wire Systems.

6.2 Ausblick

Die vorgestellte Lösung ist ein Erstjahresprojekt und stellt keinesfalls Ansprüche darauf, die letzte Evolutionsstufe des Systems oder die beste mögliche Lösung zu sein.

Im Hinblick auf die bereits veröffentlichten Änderungen des Reglements (A.4) und des Wegfallens des Regulativs, das zur Implementierung zweier verschiedener Lösungen geführt hat (A.2.4), sollten weitere Verbesserungsstufen eher auf Basis des Drive-by-Wire Systems Variante II (5.3) erfolgen.

Das Drive-by-Wire System ist eine relativ gute Lösung für die Adaption in eine Fahrzeugumgebung und Pedaleriebauform, die nicht für diese Anwendung konstruiert ist. Ein erster sinnvoller Schritt ist daher eine neu konstruierte Pedalerie, die für den Einsatz eines Drive-by-Wire Systems ausgelegt ist und keine Adaption bestehender Lösungen. Im Zuge dieses Redesigns würde auch das Implementieren eines dritten Wertaufnehmers für jedes Pedal (ähnlich der Lösung aus [16] oder [23]) Sinn ergeben.

Des Weiteren ist aus Tabelle C.3 bereits ersichtlich, dass eine Implementierung einer wechselseitigen Kontrolle der beiden Microcontroller und ein Algorithmus zur Erkennung von Ausfällen oder Problemen einer MCU von Vorteil ist.
Dies könnte am einfachsten über einen der Spezial MCUs aus C.2 erfolgen, die, bei besserer Verfügbarkeit und Ausgereiftheit, sicher eine bessere Lösung sind.

Auf Softwareebene kann es auch sinnvoll sein, bei einem Wechsel zu speziellen MCUs auf automatische Codegeneration zu setzen und somit die notwendige Testzeit zu reduzieren und anfänglichen Programmierfehlern entgegenzuwirken. Weiters können eventuell so auch bessere Fehlertoleranzen auf Softwareebene erreicht werden.

Aus [26] geht hervor, dass eventuell auch ein Umstieg auf ein System mit passiv redundanten Strukturen sinnvoll sein kann.

Seitens des Gehäuses und der verwendeten Stecker ist weiterhin auf Schutzart IP67 oder besser zu setzen. Allerdings ist eine Verkleinerung der Leiterplatte und damit verbunden des Gehäuses wünschenswert. Auch Steckverbinder, die dem Military Standard entsprechen, sind gegenüber automotiven Standards zu bevorzugen.

Eine Adaption auf einen anderen echtzeitfähigen automotiven Bus stellt ebenfalls eine legitime Weiterentwicklung dieses Projekts dar. In jedem Fall ist dieser Schritt allerdings nur dann zu empfehlen, wenn die gesamte Busstruktur des Rennboliden auf diesen Bus umgestellt wird. Dabei ist vorrangig der FlexRay Bus zu empfehlen, da dieser bereits weiter verbreitet und damit einfacher, rascher und kostengünstiger verfügbar ist. Dies impliziert allerdings den drastischen Schritt eines vollständigen Redesigns aller Busstrukturen im Boliden und ist aufgrund der geringen Buslasten des verwendeten CAN Bus (siehe 4.2) nicht erforderlich.

Eine weitere Entwicklung kann der Entwurf des Drive-by-Wire Systems nach den ISO 26262 Richtlinien sein, sobald diese veröffentlicht sind. Aber auch dies geht weit über die Anforderungen an den Rennboliden und seine Teilsysteme hinaus und ist für diese Rennserie nicht erforderlich.

Anhang A

Reglement

A.1 Reglement Formula Student Electric [13]

A.1.1 3.11.3 Torque Encoder (throttle pedal position sensor)

Drive by wire is permitted. The torque encoder must be actuated by a foot pedal. The foot pedal must return to its original position when not actuated. At least two sensors must be fitted as torque encoder. The purpose of the second sensor is redundancy. Both sensors must have different supply and ground wiring. A plausibility check is recommended to verify that both sensors give the same pedal position.

A.1.2 6.2 Failure Modes and Effects Analysis (FMEA)

Teams must submit a complete failure modes and effects analysis (FMEA) of the tractive system prior to the event. Design judges will have access to the submitted FMEA and will take the FMEA into account for design judging.

A.2 Reglement Formula Student UK Class 1A [19]

A.2.1 B7.1.4 'Brake-by-wire' systems

are allowed for regenerative braking as long as at least 50% of the brake travel activates a mechanical hydraulic system which meets the normal FSAE rules when the regenerative braking system is turned off.

A.2.2 B8.5.2 Throttle Actuation

The use of electronic throttle control (ETC) or "drive-by-wire" is allowed in Class 1A...

A.2.3 B19.1 Drive by Wire systems

Drive-by-wire systems which control the power delivered to the wheels electronically will be allowed in Class 1A. The functioning of such systems must be covered by a risk assessment.

A.2.4 B19.3.2

The two systems must not share any components (such as sensors, actuators or electronic control boxes).

A.2.5 B19.3.4

Any sensors included in these systems must have separate power and ground wiring.

A.2.6 20.3 Risk Assessment content

Where appropriate, the following items must be included in the risk assessment:

- Information on any system used which is allowed for Class 1A but not under FSAE rules (e.g. any system between the driver and supply or absorption of energy at the wheels, drive/brake by wire, etc.)
- FMEA study to be conducted on any drive/brake by wire systems to ensure where practical all failure scenarios have been evaluated and accounted
- Where software is used this should be verified to comply with MISRA Guidelines published by MIRA [http://www.misra.org.uk/] or similar approved guidelines must be followed

A.3 Reglement Formula Hybrid [28]

A.3.1 3.2.4 Steering

The steering system must affect at least two (2) wheels. ... The steering wheel must be mechanically connected to the front wheels, i.e. “steer-by-wire” of the front wheels is prohibited.

A.3.2 3.2.5 Brake Systems

The car must be equipped with a braking system that acts on all four wheels and is operated by a single control. It must have two independent hydraulic circuits such that in the case of a leak or failure at any point in the system, effective braking power is maintained on at least two wheels. ... Up to the first 50% of brake pedal travel may be dedicated to activating regenerative or other advanced braking systems, but the remaining travel must mechanically activate the hydraulic system. Regenerative braking may continue into the latter portion of the pedal travel. ...

A.4 Reglementänderungen Saison 2011 [4]

A.4.1 3.3.1 Brake System Master cylinder actuation

It is allowed to apply a low amount of brake torque to the driven wheels, when the throttle is not actuated. Low means that the applied brake torque is comparable to the brake torque of a combustion engine.

A.4.2 3.11.4 Torque Encoder

At least two sensors have to be used as torque encoder. If an implausibility occurs between the values of these two sensors, the activation of the brake pedal has to shut down the power to the motors completely. If three sensors are used an implausibility check is also necessary, but no special action is required if one of the sensors fails. If two of three sensors fail, the activation of the brake pedal has to cut off the power to the motors completely. The sensors need to have connectors that enable a check of these functions during E-Scrutineering.

A.4.3 6.8 Rain test

The rain test will not be optional any more. Every car will have to pass it. The execution of the test will be modified. Teams have to make sure that water cannot aggregate anywhere in the chassis.

Anhang B

Begriffsbestimmung

B.1 Definitionen

ETC „Electronic Throttle Control" oder E-Gas

ECU Engine Control Unit oder Motorsteuergerät, steuert den Betriebszustand moderner KFZ Motore

EPS Electric Power Steering, Lenksysteme, die direkt mit Elektromotoren ohne Zuhilfenahme hydraulischer Systeme arbeiten

HMI Human Machine Interface, Bedienelement, mit welchem Fahrwünsche an die Fahrzeugelektronik übermittelt werden können

ESP Elektronisches Stabilitätsprogramm, kann in kritschen Fahrsituationen den Fahrer aktiv unterstützen

limp home Funktion Notlaufprogramm, das ein sicheres Verhalten im Fehlerfall garantiert und ein Weiterfahren bis zur nächsten Reparaturmöglichkeit bietet

ADC Analog zu digital Wandler, kann analoge Spannungssignale in digitale Werte überführen

MCU Abkürzung für Microcontroller

FSE Abkürzung für Formula Student Electric

FSG Abkürzung für den Formula Student Bewerb in Deutschland

FSA Abkürzung für den Formula Student Bewerb in Österreich

FSUK Abkürzung für den Formula Student Bewerb in England

Anhang C

Tabellen

Bewertung	Schwere der Auswirkung(S)	Auftretenswahrscheinlichkeit (P) Beschreibung	p(P)	Detektierbarkeit (D) Beschreibung	p(D)
10	Gefährdung, Verstoß gegen Gesetze	Fehler nahezu sicher	> 30 %	keine Endeckung	< 90 %
9	Gefährdung, Verstoß gegen Gesetze möglich	sehr große Anzahl an Fehlern wahrscheinlich	< 30 %	Endeckung unsicher aber möglich	90 %
8	Totaler Funktionsausfall	große Anzahl an Fehlern wahrscheinlich	< 10 %	sehr geringe Wahrscheinlichkeit	
7	Funktion stark eingeschränkt	mäßig große Anzahl an Fehlern	< 5 %	geringe Wahrscheinlichkeit	98 %
6	Ausfall einzelner Hauptfunktionen	mittlere Zahl an Fehlern	< 1 %	mäßig geringe Wahrscheinlichkeit	
5	mäßige Einschränkung des Gebrauchsnutzens	gelegentliche Fehler	$< 0,3$ %	mittlere Wahrscheinlichkeit	
4	wenig Einschränkung des Gebrauchsnutzens	wenige Fehler	$< 500\,ppm$	mäßig hohe Wahrscheinlichkeit	99,7 %
3	geringfügige Einschränkung des Gebrauchsnutzens	sehr wenige Fehler	$< 60\,ppm$	hohe Wahrscheinlichkeit	
2	Auswirkung sehr geringfügig	Fehler selten	$< 7\,ppm$	sehr hohe Wahrscheinlichkeit	99,9 %
1	Auswirkung bleibt von Kunden unbemerkt	Fehler unwahrscheinlich	$< 0,6\,ppm$	nahezu sichere Entdeckung	99,99 %

Tabelle C.1: FMEA Faktorengewichtungstabelle

Hersteller	Typbez.	Prozessor-kern	Clock [MHz]	I/O Spg.	ADC Eingänge	ADC Res.	implem-entierte Kommuni-kationsin-terfaces	GPIO	Speicher	Gehäuse
TI	**TMS570**	ARM 32 bit RISC	160	3.3 V	24	12 bit	CAN, LIN2.0, FlexRay	115	2MB Flash / 160 kB RAM	144 pin LQFP / 337 pin Ball Grid
ST	**SPC560xx**	32bit Flash Microcon-troller	64	5 V	36	10 bit	CAN, LIN, IIC	123	512 kB Flash / 48 kB RAM	144 pin LQFP
DualCore	**DCIC9907**	ARM 946E	128				CAN, Ether-net	109		
Atmel	**AT90SDC10**	secure AVR Master core	30	5 V	0	0	0		8 MB Flash / 6 kB RAM	
Freescale	**MPC564xL**	32 bit e200 Z4	120	3.3 V	36	10 bit	CAN, SPI, LIN, IIC, FlexRay		2 MB Flash / 592 kB RAM	256BGA Emulati-onspacka-ge
Freescale	**MPC5510**	32 bit e200z	80	3.3 V	40	10 bit	CAN,SPI, LIN, IIC, FlexRay	145	1 MB Flash / 80 kB RAM	144 pin LQFP
Maxim	**MAXQ3108**	16 bit RISC MAXQ20	10	3.3 V	0	0	SPI, IIC	21	64 kB Flash / 2 kB RAM	28 pin TS-SOP

Tabelle C.2: Überblick Automotive Dual Core Microcontroller

Systemkomponente		**Drive-by-Wire Variante I**			
Teilsystem	Ausfall von:	Fehlerauswirkung	Schwere der Auswirkung	Auftrittswahrscheinlichkeit	**Klassifikation**
Versorgung	einer Versorgung	Ausfall eines Teilsystems	3	3	9
	beider Versorgungen	Ausfall des Fahrzeugs	10	1	10
DC-DC Konverter	eines DC-DC Konverters	Ausfall eines Teilsystems	3	4	12
	beider DC-DC Konverter	Ausfall des Fahrzeugs	10	2	20
Kommunikation	einer CAN Kommunikation	Ausfall eines Teilsystems	3	3	9
	beider CAN Busse	Ausfall des Fahrzeugs	10	1	10
Sensorik	eines Pedalsensors	Ausfall eines Teilsystems	3	5	15
	beider Sensoren	Ausfall des Fahrzeugs	10	1	10
Microcontroller	eines Microcontrollers	Ausfall eines Teilsystems	3	3	9
	beider Microcontroller	Ausfall des Fahrzeugs	10	1	10
Systemkomponente		**Drive-by-Wire Variante II**			
Teilsystem	Ausfall von:	Fehlerauswirkung	Schwere der Auswirkung	Auftrittswahrscheinlichkeit	**Klassifikation**
Versorgung	einer Versorgung	keine Auswirkungen auf Betriebszustand	1	3	3
	beider Versorgungen	Ausfall des Fahrzeugs	10	1	10
DC-DC Konverter	eines DC-DC Konverters	keine Auswirkungen auf Betriebszustand	1	4	4
	beider DC-DC Konverter	Ausfall des Fahrzeugs	10	2	20
Kommunikation	einer CAN Kommunikation	keine Auswirkungen auf Betriebszustand	3	3	9
	beider CAN Busse	Ausfall des Fahrzeugs	10	1	10
Sensorik	eines Pedalsensors	keine Auswirkungen auf Betriebszustand	1	5	5
	beider Sensoren	Ausfall des Fahrzeugs	10	1	10
Microcontroller	eines Microcontrollers	keine Auswirkungen auf Betriebszustand	3	3	9
	beider Microcontroller	Ausfall des Fahrzeugs	10	1	10

Tabelle C.3: Vergleich der System FMEA der Implementierungsvarianten

CAN 1.1	VEHICLE CAN 1	CAN definitions	
		Bitrate	1000 kbit
		standard Frame	used
		CAN 2.0B specifications	

Level	Naming		LOC		Definition			Comments
	Message Name	Short Name	ID	DLC		Transmit Time [ms]	Max. Delay time [ms]	Comments
	Analog Signal / Bitmap Name	Short Name	Start Bit	Bit Length	Valid range	Resolution	Unit	Comments
	Bit Name	Short Name	Start Bit	Bit Length	Value Definition			Comments
Pedalbox_1 (Little Endian/Intel Dataformat)								
MSG	PB STATUS 1	PB_S1	0x20	8		9	30	
SIG	THROTTLE POSITION	PB_S1_TP		8	10..210			
SIG	STEERING ANGLE	PB_S1_ST	8	8	10..210			
SIG	BRAKE POSITION FRONT	PB_S1_BP	16	8	10..210			
SIG	BRAKE POSITION REAR	PB_S1_RFU	24	8	10..210			
SIG	ALIVE COUNTER	PB_S1_AC	32	16	$0..2^{16}-1$			alive counter inc. with every message
SIG	<	PB_S1_CRC16	48	16	$0..2^{16}-1$			set to 0 for CRC calculation
MSG	PB STATUS 2	PB_S2	0x21					reserved for future use
MSG	PB COMMAND 1	PB_C1	0x22					reserved for future use
MSG	PB COMMAND 2	PB_C2	0x23					reserved for future use
Batterypack 1 (Little Endian/Intel Dataformat)								
MSG	BATTERY PACK 1 VOLTAGES 1-2	BP1_V12	0x100	8		1000		
SIG	VOLTAGE 1 OK	BP1_V12_V1O		8	0..1	1=OK		
SIG	VOLTAGE 2 OK	BP1_V12_V2O	8	8	0..1	1=OK		
SIG	VOLTAGE 1	BP1_V12_V1	16	12	$0..2^{12}-1$	1+x/1024	V	
SIG	VOLTAGE 2	BP1_V12_V1	32	12	$0..2^{12}-1$	1+x/1024	V	
MSG	BATTERY PACK 1 VOLTAGES 3-4	BP1_V34	0x101	8		1000		see BP1_V12
MSG	BATTERY PACK 1 VOLTAGES 5-6	BP1_V56	0x102	8		1000		see BP1_V12
MSG	BATTERY PACK 1 VOLTAGES 7-8	BP1_V78	0x103	8		1000		see BP1_V12
MSG	BATTERY PACK 1 VOLTAGES 9-10	BP1_V910	0x104	8		1000		see BP1_V12

Tabelle C.4: K-Matrix CAN1 des MaxWheel 2010 Teil 1/5

Level	Naming		LOC		Definition			Comments
	Message Name	Short Name	ID	DLC		Transmit Time [ms]	Max. Delay time [ms]	Comments
	Analog Signal / Bitmap Name	Short Name	Start Bit	Bit Length	Valid range	Resolution	Unit	Comments
	Bit Name	Short Name	Start Bit	Bit Length	Value Definition			Comments
MSG	BATTERY PACK 1 VOLTAGES 11-12	BP1_V1112	0x105	8		1000		see BP1_V12
MSG	BATTERY PACK 1 TEMPS 1-2	BP1_T12	0x106	8		1000		
SIG	TEMP 1 OK	BP1_T12_T1O		8	0..1	1=OK		
SIG	TEMP 2 OK	BP1_T12_T2O	8	8	0..1	1=OK		
SIG	TEMP 1	BP1_T12_T1	16	12	$0..2^{12}$-1		°C	
SIG	TEMP 2	BP1_T12_T1	32	12	$0..2^{12}$-1		°C	
MSG	BATTERY PACK 1 TEMPS 3-4	BP1_T34	0x109	8		1000		see BP1_T12
MSG	BATTERY PACK 1 TEMPS 5-6	BP1_T56	0x10A	8		1000		see BP1_T12
MSG	BATTERY PACK 1 TEMP 7	BP1_T7	0x10B	8		1000		see BP1_T12
MSG	BATTERY PACK 2 CAN ID	BP2_xxx	0x200	8		1000		Details see Pack 1
MSG	BATTERY PACK 3 CAN ID	BP3_xxx	0x300	8		1000		Details see Pack 1
MSG	BATTERY PACK 4 CAN ID	BP4_xxx	0x400	8		1000		Details see Pack 1
MSG	BATTERY PACK 5 CAN ID	BP5_xxx	0x500	8		1000		Details see Pack 1
MSG	BATTERY PACK 6 CAN ID	BP6_xxx	0x600	8		1000		Details see Pack 1
CAN Node (Little Endian/Intel Dataformat)								
MSG	CN REAR COMMAND 1	CN_R_C1	0xB0	4				
SIG	PUMP Left	CN_R_PL		8	0..100		%	Pump Left Speed
SIG	PUMP Right	CN_R_PR	8	8	0..100		%	Pump Right Speed
SIG	FAN Left	CN_R_FL	16	8	0..100		%	Fan Left Speed
SIG	FAN Right	CN_R_FR	24	8	0..100		%	Fan Right Speed
Safety Device (Little Endian/Intel Dataformat)								
MSG	SD COMMAND 1	SD_C1	0x70	8		100	350	
SIG	SD COMMAND 1 Bitmap	SD_C1B		16				Command
BIT	Contactor Negative Pole	SD_C1_CNEG		1	0 = OFF, 1 = ON			
BIT	Contactor Precharge	SD_C1_CPRE	1	1	0 = OFF, 1 = ON			
BIT	Contactor Positive Pole	SD_C1_CPOS	2	1	0 = OFF, 1 = ON			
BIT	HV-active Lamp	SD_C1_HVAL	3	1	0 = OFF, 1 = ON			
BIT	Reserved	SD_C1_RFU	4	12	Set to 0 for CRC calculation			reserved for future use

Tabelle C.5: K-Matrix CAN1 des MaxWheel 2010 Teil 2/5

Level	Naming		LOC		Definition			Comments
	Message Name	Short Name	ID	DLC		Transmit Time [ms]	Max. Delay time [ms]	Comments
	Analog Signal / Bitmap Name	Short Name	Start Bit	Bit Length	Valid range	Resolution	Unit	Comments
	Bit Name	Short Name	Start Bit	Bit Length	Value Definition			Comments
SIG	Reserved	SD_C1_RFU	16	16	Set to 0 for CRC calculation			reserved for future use
SIG	Nounce	SD_C1_N	32	16	$0..2^{16}$-1			copy of last SD_S1_N
SIG	CRC16	SD_C1_CRC16	48	16	$0..2^{16}$-1			set to 0 for CRC calculation
MSG	SD COMMAND 2	SD_C2	0x71					reserved for future use
MSG	SD COMMAND 3	SD_C3	0x72					reserved for future use
MSG	SD COMMAND 4	SD_C4	0x73					reserved for future use
MSG	SD STATUS 1	SD_S1	0x74	8		100		General Status
SIG	SD STATUS 1a Bitmap	SD_S1AB		8				General Status
BIT	µC Supply Voltage Error	SD_S1_UC_V		1	1 = Error, current less than approx. 10.8V			
BIT	C Supply Voltage Error	SD_S1_C_V	1	1	1 = Error, current less than approx. 8V			
BIT	CNEG Status	SD_S1_CNEG_ST	2	1	0 = OFF, 1 = ON			
BIT	CNEG Voltage Error	SD_S1_CNEG_V	3	1	1 = Error, current less than approx. 8V			
BIT	CNEG Current Error	SD_S1_CNEG_I	4	1	1 = Error, current less than approx. 0.8A			
BIT	CPRE Status	SD_S1_CPRE_ST	5	1	0 = OFF, 1 = ON			
BIT	CPRE Voltage Error	SD_S1_CPRE_V	6	1	1 = Error, current less than approx. 8V			
BIT	CPRE Current Error	SD_S1_CPRE_I	7	1	1 = Error, current less than approx. 0.8A			
SIG	SD STATUS 1b Bitmap	SD_S1BB	8	8				General Status
BIT	CPOS Status	SD_S1_CPOS_ST	8	1	0 = OFF, 1 = ON			
BIT	CPOS Voltage Error	SD_S1_CPOS_V	9	1	1 = Error, current less than approx. 8V			
BIT	CPOS Current Error	SD_S1_CPOS_I	10	1	1 = Error, current less than approx. 0.8A			
BIT	HVAL Status	SD_S1_HVAL_ST	11	1	0 = OFF, 1 = ON			
BIT	HVAL Voltage Error	SD_S1_HVAL_V	12	1	1 = Error, current less than approx. 8V			
BIT	HVAL Current Error	SD_S1_HVAL_I	13	1	1 = Error, current less than approx. 0.02A			
BIT	IMD NOT OK	SD_S1_IMD_NOK	14	1	1 = Error from IMD or R_{iso} < 100kΩ			
BIT	TRIP Status	SD_S1_TRIP	15	1	1 = Tripped because R_{iso} < 150kΩ			
SIG	SD STATUS 1c Bitmap	SD_S1CB	16	8				General Status
BIT	INTERLOCK State	SD_S1_IL	16	1	1 = Error			
BIT	Reserved	SD_S1_RFU	17	7	Set to 0 for CRC validation			reserved for future use

Tabelle C.6: K-Matrix CAN1 des MaxWheel 2010 Teil 3/5

Level	Naming		LOC		Definition			Comments
	Message Name	Short Name	ID	DLC		Transmit Time [ms]	Max. Delay time [ms]	Comments
	Analog Signal / Bitmap Name	Short Name	Start Bit	Bit Length	Valid range	Resolution	Unit	Comments
	Bit Name	Short Name	Start Bit	Bit Length	Value Definition			Comments
SIG	SD STATUS 1d Bitmap	SD_S1DB	24	8	Set to 0 for CRC calculation			reserved for future use
SIG	Nounce	SD_S1_N	32	16	$0..2^{16}$-1			needed in each command
SIG	CRC16	SD_S1_CRC16	48	16	$0..2^{16}$-1			set to 0 for CRC validation
MSG	SAFETY_STATUS_2	SD_S2	0x75	8				Isolation Resistance
SIG	Isolation Resistance	SD_S2_RISO		32	$0..2^{32}$-1	1	Ohm	Isolation Resistance
SIG	Counter	SD_S2_CNT	32	16	$0..2^{16}$-1			increasing counter
SIG	CRC16	SD_S2_CRC16	48	16	$0..2^{16}$-1			set to 0 for CRC validation
MSG	SAFETY_STATUS_3	SD_S3	0x76					reserved for future use
MSG	SAFETY_STATUS_4	SD_S4	0x77					reserved for future use
Steering Wheel (Little Endian/Intel Dataformat)								
MSG	SW TEMPS VOLTAGES	SW_TV	0x80	7		1000		
SIG	MOTOR Left TEMP	SW_TV_MLT		8	0..255	1	°C	Motor Left Temperature
SIG	MOTOR Right TEMP	SW_TV_MRT	8	8	0..255	1	°C	Motor Right Temperature
SIG	INVERTER Left TEMP	SW_TV_ILT	16	8	0..255	1	°C	Inverter Left Temperature
SIG	INVERTER Right TEMP	SW_TV_IRT	24	8	0..255	1	°C	Inverter Right Temperature
SIG	SOC TOTAL	SW_TV_ST	32	8	0..100	1	%	State of Charge Total
SIG	LOW VOLTAGE	SW_TV_LV	40	8	0..255	0.1	V	Low Voltage System Voltage
SIG	HIGH VOLTAGE	SW_TV_HV	48	8	0..255	2	V	High Voltage System Voltage
MSG	SW SPEED POWER	SW_SP	0x81	3		100		
SIG	VEHICLE SPEED	SW_SP_VS		8	0..255	1	km/h	Vehicle Speed
SIG	POWER PERCENTAGE	SW_SP_PP	8	8	0..100	1	%	Power Percentage
SIG	ECONOMY FACTOR	SW_SP_EF	16	8	0..255	0.1	%/km	Percentage of SOC used per km
MSG	SW SOC BATTERY PACKS	SW_SBP	0x82	6		1000		
SIG	SOC PACK1	SW_SBP_P1		8	0..100	1	%	SOC Battery Pack 1
SIG	SOC PACK2	SW_SBP_P2	8	8	0..100	1	%	SOC Battery Pack 2
SIG	SOC PACK3	SW_SBP_P3	16	8	0..100	1	%	SOC Battery Pack 3
SIG	SOC PACK4	SW_SBP_P4	24	8	0..100	1	%	SOC Battery Pack 4
SIG	SOC PACK5	SW_SBP_P5	32	8	0..100	1	%	SOC Battery Pack 5

Tabelle C.7: K-Matrix CAN1 des MaxWheel 2010 Teil 4/5

Level	Naming		LOC		Definition			Comments
	Message Name	Short Name	ID	DLC		Transmit Time [ms]	Max. Delay time [ms]	Comments
	Analog Signal / Bitmap Name	Short Name	Start Bit	Bit Length	Valid range	Resolution	Unit	Comments
	Bit Name	Short Name	Start Bit	Bit Length	Value Definition			Comments
SIG	SOC PACK6	SW_SBP_P6	40	8	0..100	1	%	SOC Battery Pack 6
MSG	SW ODOMETER	SW_OM	0x83	8		1000		
SIG	ODOMETER	SW_OM_OM		16	$0..2^{16}$-1	0.1	km	Odometer
SIG	TRIP METER	SW_OM_TM	16	16	$0..2^{16}$-1	1	m	Tripmeter
SIG	ENERGY USAGE ODO	SW_OM_EUO	32	16	$0..2^{16}$-1	0.1	kWh	Energy Usage Odometer
SIG	ENERGY USAGE TRIP	SW_OM_EUT	48	16	$0..2^{16}$-1	0.1	kWh	Energy Usage Tripmeter
MSG	SW SETTINGS	SW_S	0x84	4		1000		
SIG	VDC SLIP TARGET	SW_S_VST		8	0..255	0.1	%	Vehicle Dynamics Control Slip Target
SIG	VDC TORQUE VECTOR PARAM1	SW_S_VTP1	8	8	0..255	0.1	%	VDC Torque Vectoring Parameter 1
SIG	VDC TORQUE VECTOR PARAM2	SW_S_VTP2	16	8	0..255	0.1	%	VDC Torque Vectoring Parameter 2
SIG	COOLING SYSTEM OVWR Bitmap	SW_S_CSOB	24	8				
BIT	FAN Left Overwrite	SD_S_FLO	24	1	0=OFF, 1 = ON with 100%			
BIT	FAN Right Overwrite	SD_S_FRO	25	1	0=OFF, 1 = ON with 100%			
BIT	PUMP Left Overwrite	SD_S_PLO	26	1	0=OFF, 1 = ON with 100%			
BIT	PUMP Right Overwrite	SD_S_PRO	27	1	0=OFF, 1 = ON with 100%			
BIT	Reserved	SD_S_RFU	28	3				
BIT	OVERWRITE Enabled	SD_S_OE	31	1	1 = Overwrite enabled			

Tabelle C.8: K-Matrix CAN1 des MaxWheel 2010 Teil 5/5

Literaturverzeichnis

[1] MISRA The Motor Industry Software Reliability Association. *Guidelines for the use of the C language in critical systems*. MISRA, 2004.

[2] Burkhard Bauer. Drive-by-Wire rückt näher. In *Auto & Elektronik*, pages 68–69, 2005.

[3] Prof. Dr.-Ing. Kai Borgeest. *Elektronik in der Fahrzeugtechnik*, pages 73 – 110. Vieweg+Teubner Verlag, 2010.

[4] Formula Student Electric Committee. Planned Major Rules Changes for FSE 2011, 27.08.2010. http://www.formulastudentelectric.de/uploads/media/Planned_Rules_Changes_FSE_2011.pdf.

[5] X-By-Wire Consortium. X-By-Wire Safety Related Fault Tolerant Systems in Vehicles. Internet, accessed 2010-02-12, http://www.vmars.tuwien.ac.at/projects/xbywire/docs/final.doc, 1998.

[6] Rat der Europäischen Union. Richtlinie 70/311/EWG des Rates vom 8. Juni 1970 zur Angleichung der Rechtsvorschriften der Mitgliedstaaten über die Lenkanlagen von Kraftfahrzeugen und Kraftfahrzeuganhängern . Internet, accessed 2010-07-22, http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31999L0007:DE:HTML.

[7] Rat der Europäischen Union. Richtlinie 71/320/EWG des Rates vom 26. Juli 1971 zur Angleichung der Rechtsvorschriften der Mitgliedstaaten über die Bremsanlagen bestimmter Klassen von Kraftfahrzeugen und deren Anhängern. Internet, accessed 2010-07-22, http://europa.eu/legislation_summaries/internal_market/single_market_for_goods/motor_vehicles/motor_vehicles_technical_harmonisation/l21247_de.htm.

[8] Bundesministerium der Justiz Deutschland. Straßenverkehrs-Zulassungs-Ordnung. Internet, accessed 2010-07-22, http://bundesrecht.juris.de/bundesrecht/stvzo/gesamt.pdf, letzte Änderung 2009-04-21.

[9] Elmar Dilger, Thomas Führer, and Bernd Müller. The X-by-Wire Concept: Time-Triggered Information Exchange and Fail Silence Support by new System Services. In *SAE Technical Paper Series, Advances in Safety Technlology (SP-1321)*, pages 141–149, 1998.

[10] Martin Elshuber. Brake-By-Wire distributed Application Subsystem in the DECOS Integrated Architecture. Internet, accessed 2010-02-12, http://www.vmars.tuwien.ac.at/php/pserver/extern/docdetail.php?DID=2393 &viewmode=thesis, 1998.

[11] embedded europe.com. Dual-Core Technology Gives Microcontrollers More Power. Internet, accessed 2010-01-21, http://www.esemagazine.com/index.php?option=com_content&task=view&id=422 &Itemid=3, 2010.

[12] ISO International Organization for Standardization. ISO26262. Internet, accessed 2010-07-16, http://www.iso.org/iso/home.htm, 2010.

[13] Formula Student Electric Committee. Formula Student Electric Rules 2010, 2010. Ver. 1.4.7, http://www.formulastudentelectric.de/uploads/media/FSE_Rules_2010_v1.4.7.pdf.

[14] Randy Frank. X-By-Wire: for Power, X Marks the Spot. In *Ward's Auto Electronics*, pages 26 – 30, 2004.

[15] Freescale Halbleiter Deutschland GmbH. Mikrocontroller der nächsten Generation von Freescale und STMicroelectronics ermöglichen einen Quantensprung bei sicherheitskritischen Automotive-Systemen. Internet, accessed 2010-01-21, http://www.fair-news.de/news/Mikrocontroller+der+naechsten+Generation+von+Freescale +und+STMicroelectronics+erm%F6glichen+einen+Quantensprung+bei+ sicherheitskritischen+Automotive-Systemen/32536.html, 2009.

[16] B. Hedenetz and R. Belschner. Brake-by-Wire without Mechanical Backup by Using a TTP-Communication Network. Society of Automotive Engineers Inc., 1998.

[17] Michael Hinterberger. Bussysteme im Kraftfahrzeug. Internet, accessed 2010-09-06, http://www.ife.tugraz.at/LV/upprog/Pdfs/BUSSYSTEME.pdf, 2007.

[18] I.Ludwig. CAR IT. Internet, accessed 2010-07-16, http://homepages.fh-giessen.de/ hg10013/Lehre/MMS/WS0304/Ludwig/, 2004.

[19] IMechE Incorporating by permission of SAE International. 2010 Formula Student Class 1A Rules, 2010. http://www.formulastudent.com/Libraries/Documents/Class_1A_rules_2010_1.

[20] Rolf Isermann. Falut Tollerant Drive-by-Wire Systems. Technical report, Darmstadt University of Technology, Institut of Automatic Control, 2000.

[21] Helmuth Lemme. *Sensoren in der Praxis*. Franzis Verlag, 1993.

[22] Manfred Broy. Automotive Software and Systems Engineering. Technical report, Technische Universität München, Institut für Informatik, 2006.

[23] D. McKay, G. Nichols, and B. Schreurs. Delphi Electronic Throttle Control Systems for Model Year 2000; Driver Features, System Security, and OEM Benefits. ETC for the Mass Market. In *SAE Technical Paper Series, Electronic Engine Controls 2000 (SP-1500)*, pages 1–11, 2000.

[24] Daniel Müller. Buslastanalyse und -senkung der CAN-Busse im Automobil. Master's thesis, Fachhochschule Furtwangen, 2006.

[25] Christian Coronado Mondragon et al. Managing technology for highly complex critical modular Systems: The case of automotive by-wire systems. *Int. J. Production Economics 118*, 2009.

[26] Stefan Poledna. Vorlesung und Übung Fehlertolerante Systeme. Internet, accessed 2010-08-12, http://www.vmars.tuwien.ac.at/courses/ftol/vo_slides.html, 2006.

[27] O. Rooks et al. *Redundancy Management for Drive-by-Wire Computer Systems*, pages 249 – 262. Springer Berlin / Heidelberg, 2003.

[28] SAE International. 2010 Formula Hybrid Rules, 2010. http://www.formula-hybrid.org/pdf/Formula-Hybrid-2010-Rules.pdf.

[29] Dipl.-Ing. Jörg Schäuffele and Dr.-Ing. Thomas Zurawka. *Automotive Software Engineering*. Vieweg, 2006.

[30] Fred Seidel. X-by-Wire. Technical report, Chemnitz University of Technology, Operating Systems Group, Hauptseminar Transportation Systems, 2009.

[31] Österreichisches Bundeskanzleramt Rechtsinformationssystem. Bundesgesetz vom 23. Juni 1967 über das Kraftfahrwesen (Kraftfahrgesetz 1967 - KFG. 1967) StF: BGBl. Nr. 267/1967. Internet, accessed 2010-07-22, http://www.ris.bka.gv.at/GeltendeFassung/Bundesnormen/10011384/KFG%201967%2c%20Fassung%20vom%2022.07.2010.pdf.

[32] Irina Theis. Das Steer-by-Wire System im Kraftfahrzeug - Analyse der menschlichen Zuverlässigkeit. Master's thesis, Technische Universität München, Lehrstuhl für Ergonomie, 2002.

[33] Michael Trzesniowski. *Rennwagentechnik: Grundlagen, Konstruktion, Komponenten, Systeme*. Vieweg+Teubner Verlag, 2010.

[34] Auto-Motor und Sport. Aktives Fahrwerk von Bose. Internet, accessed 2010-07-19, http://www.auto-motor-und-sport.de/news/aktives-fahrwerk-von-bose-672534.html, 2004.

[35] Martin Whitbread. PRODUCT FEATURE: Dual-cores give MCUs more power. Internet, accessed 2010-01-22, http://www.eetimes.com/electronics-news/4135885/PRODUCT-FEATURE-Dual-cores-give-MCUs-more-power, 2008.

[36] H. Winner et al. Wann kommt By-Wire auch für Bremse und Lenkung. Steuerung und Regelung von Fahrzeugen und Motoren - AUTOREG 2004 : Tagung Wiesloch, 2. und 3. März 2004 / VDI/VDE-Gesellschaft Mess- und Automatisierungstechnik.- Düsseldorf, 2004.

[37] Hermann Winner and Oliver Heuss. X-By-Wire Betätigungselemente - Überblick und Ausblick. Internet, accessed 2010-06-26, http://www.fzd.tu-darmstadt.de/media/fachgebiet_fzd/publikationen_3/2005/2005_winner_heuss_menschundfahrzeug.pdf, 2005.

[38] Werner Zimmermann and Ralf Schmidgall. *Bussysteme in der Fahrzeugtechnik*. Vieweg-Teubner-Verlag, 2008.