Bachelorarbeit - Christian Mentin

Drive-by-Wire Systems des TU Graz Formula Student Electric Boliden 2011

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

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

Inhaltsverzeichnis

Abbildungsverzeichnis

Abstract

Heutzutage sollen Fehler in Hard- und Software erkannt und wenn möglich von Systemen selbstständig korrigiert werden können, um die Auswirkungen eines Ausfalls von kritischen Systemkomponenten gering zu halten. In modernen Fahrzeugen sowie in Fahrzeugen der Formula Student Electric werden die Antriebsstränge durch elektrische oftmals digitale Signale gesteuert. Sogenannte Drive-by-Wire Systeme sind ein wichtiger Bestandteil für die Bewegung des Fahrzeuges, welche nicht nur fehlertolerant sein sollen, sondern auch in der Lage sein müssen Fehler, beispielsweise in der Pedal Sensorik, zu detektieren und bewerten, um eventuell im Betrieb und unterbrechungsfrei auf vorhandene Redundanz zurückgreifen zu können. Um Personenschäden zu verhindern muss in solch kritischen Anwendungen Fail-Safe Operation sichergestellt werden. In dieser Bachelorarbeit werden grundlegende Designprinzipien für Fehlertolerante Systeme diskutiert und die gewonnen Erkenntnisse beim Design der Hard- und Software des Drive-by-Wire Systems des TU Graz Formula Student Electric Boliden 2011praktisch umgesetzt.

1. Einleitung

In der Automobilbranche lässt sich ein klarer Trend erkennen. Wer noch keine Elektro- oder Hybrid Fahrzeuge in seinem Angebot hat, ist zumindest dabei, erste Studienfahrzeuge für Klein- und Kleinstserien zu entwickeln. Alle großen Automobilhersteller sind am Weg serienreife Fahrzeuge zu entwickeln, deren Hauptantrieb ein rein elektrischer ist. BMW lässt schon seit mehr als 3 Jahren 500 Mini-e in den USA ihre Kreise ziehen, um mehr Informationen über rein elektrisch betriebene Fahrzeuge im Alltagsverkehr zu erhalten. Nissan vertreibt schon längere Zeit den Nissan Leaf, Opel den Ampera ab Ende 2011, in Großserie.

Der Weg für die Automobilentwicklung in den nächsten Jahren scheint geebnet für den elektrischen Antrieb.

1.1.Motivation

Doch der Wille allein genügt nicht um ein Elektrofahrzeug mit Serienreife zu entwickeln. Die Entwicklung des Verbrennungsmotors hatte viele Jahre Zeit um den Antrieb zu perfektionieren. Die Ansprüche an den elektrischen Antrieb sind daher nicht geringer. Wartungsfreiheit, Standfestigkeit und große Reichweiten sind die großen Knackpunkte, an denen sich zeigen wird, ob sich der rein elektrische Antrieb am Massenmarkt behaupten kann. Daher sucht und braucht die Industrie junge motivierte Ingenieure und Absolventen technischer Studienrichtungen, welche mit ihrem Knowhow und fachspezifischem Wissen die Entwicklung serienreifer rein elektrisch betriebener Fahrzeuge vorantreiben können.

Abbildung 1.1: Max Wheel 2011

Aus diesem Grund wird die „Formula Student Electric“ von der Industrie auch besonders wertgeschätzt. Ohne Unterstützung und Sponsoring von großen Firmen und Konzernen könnte das TUGraz E-Power Racing Team nicht bestehen.

Die Erfahrung des Vorjahres zeigt, dass nicht nur die Studenten, sondern auch die Industrie, durch die gewonnenen Erfahrungen und Erkenntnisse, enorm profitieren.

1.2.Ziel der Arbeit

Der MaxWheel 2011 sollte sich, wie schon in den Reglements gefordert, deutlich von seinem Vorgänger Modell unterscheiden. Neben etlichen mechanischen Neuerungen wurde auch das elektrische Konzept komplett neu aufgestellt. Zusätzlich zu neuen Controllern, welche in diesem Jahr von Infineon bereitgestellt wurden (XC2000 Series), sind CAN Nodes aus dem Vorjahreskonzept geworfen und neue Einheiten geplant worden.

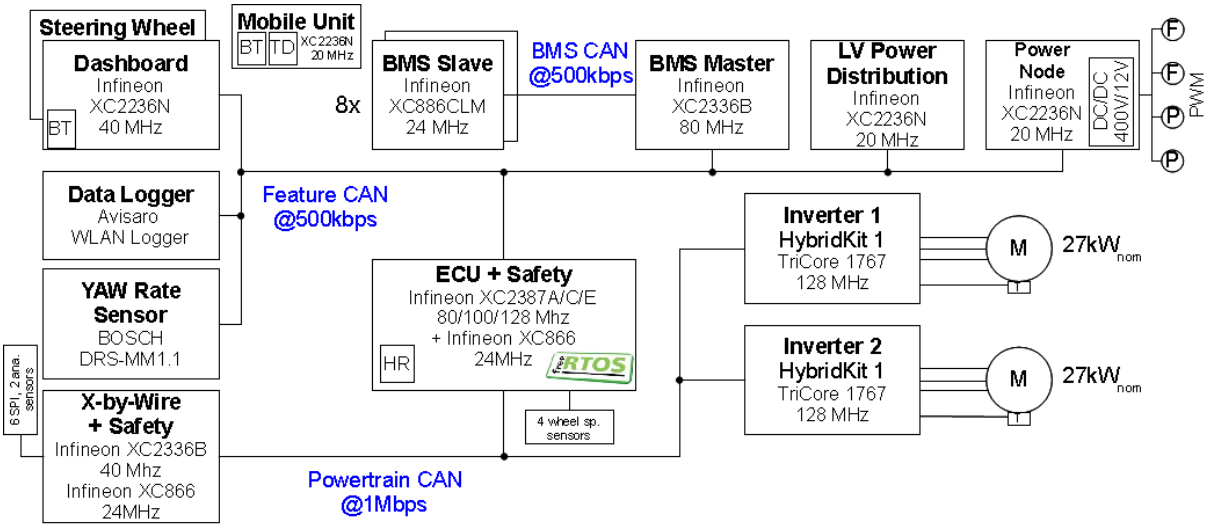


Abbildung 1.2: Max Wheel 2011 – Elektrisches Konzept

Eine für die Bewegung des Fahrzeuges besonders wichtige Einheit, welche Bestandteil dieser Arbeit ist, war die X-by-Wire Box. 6 über SPI angeschlossene, sowie 2 analoge Sensoren sollen Aufschluss über Pedalstellungen sowie Lenkwinkel geben, um anschließend die validierten Werte, über den CAN Bus, zur zentralen Steuereinheit (ECU) zu transferieren. Im Vorjahr war die Box redundant ausgeführt. Im ersten und bisher verwendeten Konzept des Max Wheel 2010 wurden die Messungen an den Sensoren von zwei Boxen getrennt durchgeführt und zur ECU übermittelt. Die Validierung der Sensorwerte selbst wurde jedoch auf der ECU durchgeführt. Dieser Umstand, auch auf Grund des etwas entschärften Reglements, die Redundanz der Pedalboxen betreffend, sollte im Max Wheel 2011 daher grundlegend verändert werden. Die Anforderungen an das Design der X-by-Wire Box 2011 sind im *Punkt 3.1 Grundanforderungen an das Design* detailliert angeführt.

2. Fehlertoleranz in verteilten Systemen

Für eine detaillierte Ausführung der in diesem Kapitel vorgestellten Systeme und Praktiken möchte ich auf die Masterarbeit, meines Vorgängers[1] beim TUGraz E-Power Racing Team verweisen, da eine genauere Ausführung den Rahmen dieser Arbeit in großem Maße übersteigen würde.

Ein Ausfall eines Systems, egal in welchem Anwendungsgebiet, ist für den Endverbraucher immer unangenehm. Daher bringen redundante Systeme einen erheblichen Vorteil, da auf Ersatzsysteme je nach Anwendung, nahtlos, meist ohne großen Aufwand, umgeschalten werden kann.

Im speziellen, für den Automotive Bereich, genügt es natürlich nicht die Entscheidung wann ein System funktioniert und wann es fehlerhaft ist, dem Fahrer zu überlassen. Daher müssen Fehler schon Systemintern erkannt werden und wenn möglich auf Ersatzsysteme umgeschaltet werden. Heutzutage ist es bei Verbrennungsmotoren eine übliche Praktik, dass bei Ausfall von bestimmten Sensoren, oder erreichen von kritischen Temperaturen, der Motor nur mehr Drehzahl- und Leistungsbegrenzt läuft, sodass der Fahrer noch die Möglichkeit hat, die nächste Werkstatt aufzusuchen, ohne den Motor zu beschädigen.

Von einem fehlertoleranten System spricht man, wenn ein korrekter kontinuierlicher Betrieb des Systems auch dann sichergestellt ist, wenn Systemkomponenten oder Teilsysteme fehlerhaft arbeiten oder sogar ausfallen. Fehlertoleranz erhöht insbesondere auch die Zuverlässigkeit eines Systems.

2.1. Arten von Fehlertoleranten Systemen

Prinzipiell unterscheidet man mehrere Arten von Fehlertoleranten Systemen, welche in den nachfolgenden Punkten genauer beschrieben werden. Weiters muss man zwischen Abweichungen (faults), Fehlern (errors) und Ausfall (failures) eines Systems unterscheiden. Aus Faults können Errors, daraus Failures, resultieren. Ein Fault ist immer die Grundursache für einen System Ausfall, jedoch kann dieser durch geschickte Gegenmaßnahmen verhindert werden. [2]

Weiters können Faults in transiente und permanente Faults eingeteilt werden, wobei aus technischer Sicht permanente Fehler in den meisten Fällen leichter zu detektieren sind, als transiente. Diese zwei großen Gruppen von möglichen Faults lassen sich detailliert in folgende Kategorien einteilen:

- Crash Faults: Die betroffene Komponente fällt komplett aus, oder erreicht nie wieder einen validen Zustand.
- Omission Fauls: Die betroffene Komponente führt ihren Service nicht mehr aus.
- Timing Faults: Die Komponente funktioniert nicht immer einwandfrei.
- Byzantine Faults: Zufällige Fehler die spontan und unvorhersehbar auftreten.

2.1.1. Fail-Operational System

Ein Fail-Operational System führt noch immer die erwartete Funktion aus, auch wenn das Hauptsystem ausfällt. Als Beispiel können hier Aufzüge genannt werden, oder ein System das in den 60er Jahren vom U.S. Militär entwickelt wurde, um einen nuklearen Gegenschlag einzuleiten, auch wenn alle wichtigen Personen eliminiert, und jede Kommunikation zu den Raketenstützpunkten abgebrochen wäre.

2.1.2. Fail-Silent System

Ein Fail-Silent System liefert immer einen korrekten Wert solange es keinen Fehler gibt. Im Fehlerfall liefert das System keine Daten.

2.1.3. Fail-Safe System

Ein Fail-Safe System wird im Fehlerfall sicher, wenn es nicht mehr richtig arbeiten kann. Als Beispiel kann hier das Fahrsignal auf den Signalmasten für Zugstrecken genannt werden, da dort im Fehlerfall Sicherheit gegeben sein muss.

2.1.4. Fail-Secure System

Fail-Secure Systeme bieten im Fehlerfall den maximalen Schutz. So werden, im Gegensatz zu Fail-Safe Türen, welche im Fehlerfall geöffnet werden, Fail-Secure Türen verschlossen, mit dem Kompromiss, dass zwar Personen in den Räumlichkeiten eingeschlossen werden können, dafür aber maximaler Schutz gegen z.B.: Diebstahl gegeben ist.

2.1.5. Fail-Passive System

Fail-Passive Systeme erfüllen im Fehlerfall weiter ihre vorgesehen Funktion, so ist es beispielsweise im Flugzeug möglich, dieses zu steuern, auch wenn der Autopilot ausfällt.

2.1.6. Fault-Tolerant System

Tritt ein Fehler auf so kann dieser ohne Leistungsverminderung des gesamten Systems sofort behoben werden. Das System toleriert Abweichungen (in z.B.: Messgrößen) oder Ausfall bestimmter Systemkomponenten bis zu einem gewissen Grad, sodass der Normalbetrieb des Gesamtsystems weiterhin gewährleistet ist.

2.2.Fehlertoleranz in Hardware

Fehlertoleranz kann in Hardware nur durch das Hinzufügen von Redundanz erreicht werden. Dazu gibt es mehrere Konzepte, welche nachfolgend kurz erklärt werden:

- Statisch redundant: mehrere Komponenten sind permanent aktiv.
- Dynamisch redundant: Komponenten werden nach Bedarf (Fehlerfall) aktiviert.
- Hybrid redundant: Kombination von statischer und dynamischer Redundanz.

2.2.1. Statische Hardware Redundanz

Für statische Hardware Redundanz möchte ich hier als Beispiel ein N-Modular Redundancy (NMR) System nennen.

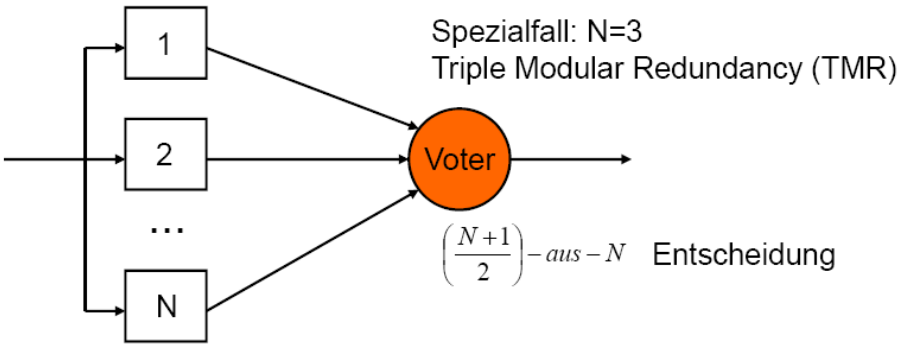


Abbildung 2.1: NMR-System

Das Ziel bei diesem System ist es einen „Mehrheitsentscheid“ durchzuführen. Da alle Komponenten parallel laufen kann der Votierer sowohl permanente als auch transiente Fehler erkennen. Für dieses System müssen mindestens $\frac{N}{2} + 1$ Teilsysteme einen validen Wert liefern.

2.2.2. Dynamische Hardware Redundanz

Der Ablauf bei Auftreten eines Fehlers ist wie folgt:

1. Fehlerdiagnose
    - Erkennung
    - Fehlerlokalisierung
2. Fehlerbeseitigung
    - durch Reparatur der ausgefallenen Einheit
    - Ersatz durch Reserveeinheit
    - Re-Konfiguration
3. Fehlerbehebung
    - Wiederherstellen des Systemzustandes vor dem Ausfall
    - Wiederanlauf

Weiters unterscheidet man bei Dynamischer Redundanz zwischen:

- Hot Standby: Reserveeinheiten laufen mit.
- Cold Standby: Reserveeinheiten laufen nicht mit
- Duplex-System: Diagnose beginnt, wenn der Vergleicher eine Diskrepanz feststellt.

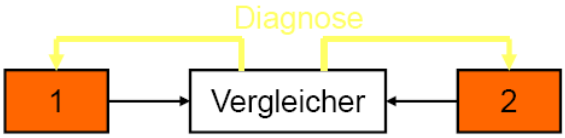


Abbildung 2.2: Duplex System

Abschließend möchte ich kurz die statische, der dynamischen Redundanz gegenüberstellen:

Statische Redundanz	**Dynamische Redundanz**
Vorteile: • transiente und permanente Fehler • unterbrechungsfrei • mit Standardhardware realisierbar (keine eigene Fehlererkennung)	Vorteile: • Momentaner Redundanzgrad einstellbar • Fremd- bzw. gegenseitige Nutzung der Redundanz • Lange Missionszeiten durch „Cold Standby“
Nachteile: • Hardware Aufwand • Synchronisationsproblem • kurze Missionszeiten	Nachteile: • Adaptionszeit • Subsysteme mit Fehlererkennung • Transiente Fehler

Für detaillierte Informationen zu allen erwähnten Systemen möchte ich wiederum auf [1] referenzieren.

2.3.Fehlertoleranz in Software

Da Fahrzeugsysteme immer komplexer werden, und in modernen Fahrzeugen etliche Mega Byte an Software in vielen Controllern laufen, wird Fehlertoleranz in Software auch in Zukunft immer wichtiger werden. Diese kann 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: Eine Kombination aus Hardware und Software Redundanz (vergleichbar mit passiver Redundanz 3.4.3)

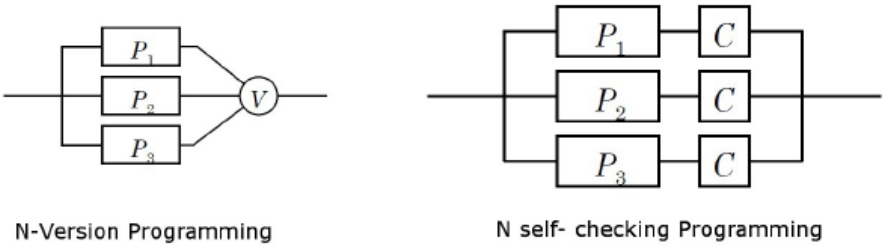


Abbildung 2.3: Realisierungsvarianten von Design-Diversität

Abbildung 2.3 zeigt verschiedene Ansätze, wie verschiedene Varianten von Design Diversität aussehen könnten. Beim N-Version Programming entscheidet ein Voter, vergleichbar mit statischer Hardware Redundanz, durch Mehrheitsentscheid, welchem Ergebnis vertraut wird. Bei N self-checking Programming, kann sich jedes Modul selbst überprüfen. Der Voter, welcher ebenfalls einen Fehler erzeugen könnte, wird in dieser Ausführung nicht benötigt.

Diese Ausführungen haben jedoch einen sehr hohen Entwicklungsaufwand. Vorbeugend, um Softwarefehler zu vermeiden, sollte daher versucht werden sich an bestehende Coding Standards zu halten, und den Programmcode, unter Beachtung bestehender Prinzipien wie z.B.: Datenkapselung und den MISRA Richtlinien, zu gestalten. Zusätzlich sollen häufig Tests (Black/White box Test, Fault injection…) durchgeführt werden und bekannte Verfahren zur Bewertung von Software Sicherheit verwendet werden.

2.3.1. Überwachungssysteme – Supervisory Circuits

Als Spezialfall von sich selbst überwachender Software sei an dieser Stelle das Supervisory Konzept genannt. Die Idee hinter diesem Konzept ist folgende, dass jeder wichtige Rechenschritt einer CPU, überwacht werden soll, um im Fehlerfall darauf reagieren zu können.

Aus diesem Grund werden seit 2010 schon mehrere „Dual Core“ Controller von namhaften Herstellern in Kleinserie speziell für den Automotive Bereich entwickelt, da dort Fehler in Berechnungen gravierende Konsequenzen nach sich ziehen können.

Zusätzlich wurden für Systeme, in denen Sicherheit eine besondere Rolle spielt, Supervisory Controller entwickelt, welche alle wichtigen Funktionen eines Moduls (Betriebsspannungen, Funktion des Hauptcontrollers, Temperatur…) überwachen können, um im Fehlerfall das Gesamtsystem zu resetieren oder komplett abzuschalten. Infineon bietet eigens dafür entwickelte, nach ASIL-D Standard konforme, Safty Controller (CICxxx Series) an.

2.4.Zeit- und Coderedundanz

Nicht nur Hardware- und Software Redundanz spielen eine wichtige Rolle, sondern auch das Übertragen oder Speichern der berechneten Daten mit Redundanz.

Daher sind bei Bussystemen, welche speziell für den Automotive Bereich entwickelt wurden, große Redundanzen in der Codierung des Übertragungsstandards vorhanden, um Fehlern bei der Übertragung vorzubeugen. Im speziellen kann hier, der für den Automotive Bereich entwickelte CAN Bus genannt werden, dessen Nutzdatenrate weit unter der realen Datenrate liegt, da durch Hinzufügen von Redundanz (z.B.: CRC, ACK, Stuff Bits…) Übertragungsfehler detektiert werden können.

Zusätzlich zu Redundanz in verwendeten Codes, kann zeitliche Redundanz ebenfalls verwendet werden, um Übertragungsfehler zu kompensieren. So kann es sinnvoll sein, wenn es das Übertragungssystem zulässt, ein und denselben Wert, in bestimmten Zeitintervallen nochmals zu senden.

2.5.Fehlertoleranz in Benutzerschnittstellen

Nicht nur die bisher aufgeführten Maßnahmen sind wichtig um ein hohes Maß an Ausfallsicherheit zu gewährleisten. Auch die Mensch-Maschine Schnittstelle soll robust gegen Eingabefehler durch den Menschen sein. So wird beispielsweise in der Automobiltechnik schon seit mehreren Jahren an automatischen Bremsassistenten gearbeitet, sodass Fehler in der Benutzereingabe (z.B.: Auffahrunfälle) sofern möglich, ohne Folgen für die Passagiere bleiben.

3. Entwurf des Drive-by-Wire Systems

Das X-by-Wire System könnte zwar aus technischer Sicht als Drive-,Steer- und Break-by-Wire System ausgeführt werden, jedoch wird aus Gründen, welche die Kosten und das Reglement betreffen ein reines Drive-by-Wire Konzept mit zusätzlichen Funktionen realisiert. Bremse sowie Lenkung sind mit mechanischen Komponenten aufgebaut.

Dieses Jahr ist es erstmals zulässig, Energie beim Bremsvorgang zu rekuperieren. Um beurteilen zu können wann wie viel Energie rekuperiert werden kann, soll jeweils ein Bremsdrucksensor in den beiden Bremskreisen angebracht werden, welcher von der X-by-Wire Box, sie wird in dieser Arbeit auch Pedalbox bezeichnet, ausgewertet werden soll.

Nachdem der Max Wheel 2011 von zwei Motoren an der Hinterachse angetrieben wird, und kein mechanisches Differential besitzt, wird auf ein elektronisches Differential implementiert, welches über den Lenkwinkel die Drehzahl der Räder abgleicht. Dazu soll ein Sensor am Lenkgetriebe angebracht werden, welcher ebenfalls von der X-by-Wire Box ermittelt werden soll.

3.1.Grundanforderungen an das Design der X-by-Wire Box

Durch Erfahrungen und Erkenntnissen aus dem Vorjahr wurden die Anforderungen an die Pedalbox 2011, durch Reglements und das E-Power Racing Team neu definiert.

3.1.1. Anforderungen durch das Reglement

Nachfolgend einige wichtige Auszüge aus den Reglements, das Drive-by-Wire System betreffend:

Formula Student Electric **Germany** [3]:

"4.11.4 Torque Encoder (throttle pedal position sensor)

> *Drive by wire is permitted.*
> *…*
> *At least two sensors have to be used as torque encoder.*
> *If an implausibility occurs between the values of these two sensors the power to the motor(s) has to be shut down completely. It is not necessary to completely deactivate the Tractive System, the motor controller(s) shutting down the power to the motor(s) is sufficient. Motor power is allowed to be restored after the driver has selected the sensor that works correctly. Each sensor has to have a separate detachable connector that enables a check of these functions by unplugging it during E-Scrutineering."*

Formula Student Electric **UK** [4]:

"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 (B20) Note: Front wheel steer-by-wire systems will not be allowed as per rule B3.2.4.

…

B19.3 Two independent systems to shut off power

B19.3.1 There must be at least two completely independent systems to shut off power due to the throttle pedal being released or the brake pedal pushed. (This is the equivalent of the current two throttle return springs rule as described in the FSAE rules). The functioning of these systems must be covered by a risk assessment (B20).
B19.3.2 The two systems must not share any components (such as sensors, actuators or electronic control boxes).
B19.3.3 The two systems must be independently demonstrated to the scrutineers before the car will be allowed to run (the Entrant must determine a method to perform this test such that this can be checked quickly).
B19.3.4 Any sensors included in these systems must have separate power and ground wiring.
B19.3.5 Where these systems rely on electrical sensors, these systems must be able to detect open circuit and short circuit faults on any signal wires or sensors such that any fault condition results in all power being turned off being shut down.
B19.3.6 The above regulations are in addition to the brake over travel switch is still required for Class 1A cars and this must shut off all power."

Formula Student Electric **Italien** [5]:

"3.8.2.4 Electrical Throttle Actuation:
When electrical or electronic throttle actuation is used, the throttle actuation system must be of a fail-safe design to assure that any single failure in the mechanical or electrical components of the throttle actuation system will result in the engine returning to idle (IC engine) or having zero torque output (electric motor)…"

Das Reglement in England zeigt, dass es zu Anfang der Saison noch nicht ganz schlüssig war wie die Realisierung des Drive-by-Wire Systems aussehen sollte, da im Reglement von 2 unabhängigen Systemen gesprochen wird, dies jedoch für das Vorhaben, ein rein elektrisches Fahrzeug in Betrieb zu nehmen etwas überzogen scheint.

Da viele der Regeln im Reglement für England einfach kurzerhand aus der Formula Hybrid übernommen wurden, war hier noch einiges an Klärungsbedarf. Schließlich konnten wir nach Beantwortung mehrerer Fragen, die Drive-by-Wire Box so in Betrieb nehmen, wie durch das E-Power-Racing Team geplant.

Wie und mit welchen Komponenten die Drive-by-Wire Box schließlich realisiert wurde, um die Reglements bestmöglich zu erfüllen, ist im Detail unter *Punkt 3.1.2.Anforderungen durch das E-Power Racing Team* und *4.Implementation und Realisierung im Max Wheel 2011* nachzulesen.

3.1.2. Anforderungen durch das E-Power Racing Team

Die Anforderungen an die X-by-Wire Box 2011 wurden wie folgt definiert:

- Eine einzige X-by-Wire Box ausgeführt als Fail Silent- und fehlertolerantes System.
- Die Box soll so klein als möglich und möglichst stabil ausgeführt sein, um im vorderen Teil des Monocoques seitlich neben der Pedalerie Platz zu finden.
- Funktionalität des 16bit Mikrocontrollers muss gewährleistet sein (8bit Supervisory Controller).
- Redundante Messung von Lenkwinkel, Gaspedalstellung mithilfe von mehreren Sensoren mit digitaler Schnittstelle.
    - Messung des Bremsdruckes über analoge ratiometrische Drucksensoren.
- Entscheid über die Richtigkeit der Sensorwerte wird in der X-by-Wire Box durch Software getroffen.
- Übermittlung der validierten Sensorwerte in einem, durch einen Alive-Counter, eindeutig identifizierbaren Packet, an die ECU in 10ms Abständen, mittels eines „Echtzeit“ CAN Busses.
- Bei Ausfall eines Sensors durch beispielsweise Kurzschluss oder Kabelbruch, muss eine einwandfreie Funktion aller weiteren Sensoren gewährleistet werden.
- Separate Stecker und strombegrenzte Ausgänge sollen für jeden einzelnen Sensor vorgesehen werden.
- Alle verwendeten Stecker sollen von einander unterscheidbar sein, eine Zugentlastung besitzen und mit Arretierungen gegen Vibrationen geschützt werden.
- Bei Ausfall mehrerer Sensoren unter der Bedingung, dass keine Redundanz mehr vorhanden ist, entscheidet der Fahrer welcher Sensor weiter verwendet wird.
- Relevante Fehler müssen dem Fahrer mitgeteilt werden
- Optional.: Degressive oder Progressive Pedalkennlinien einstellbar

3.2.X-by-wire Sensoren

Die Sensoren sollten so gewählt werden, sodass diese die nachfolgenden Spezifikationen erfüllen:

- Staub- und Spritzwassergeschützte Gehäuse mindestens nach IP65
- Das Sensor Gehäuse soll so klein und leicht wie möglich sein und hohen Belastungen standhalten

3.2.1. Gaspedalstellung, und Lenkwinkel

Im Vorjahr wurde ein potentiometrisches Messprinzip, durch ein Linearpotentiometer (TEX0050), und ein kontaktloses auf dem Hall-Effekt basierendes rotatorisches Messprinzip (RSC2800), zur Ermittlung der Gaspedalstellung, gewählt. Zur Ermittlung des Lenkwinkels wurde 2010 derselbe kontaktlose Hall-Effekt Sensor verwendet wie zur Messung der Gaspedalstellung.

Die Erfahrung aus dem Vorjahr hat gezeigt, dass die Befestigung des Linear geführten Potentiometers einiges an Platz beanspruchte. Da das Design dieses Jahr noch etwas kompakter ausfallen sollte, musste auf das Linearpotentiometer verzichtet werden.

Daher wurden die an die Sensorik gestellten Anforderungen weiter angepasst:

- Digitale Schnittstelle um die Auswirkung von eingekoppelten Störungen in den Sensorleitungen möglichst gering zu halten
- Redundante Sensorausführung
- Rotatorisches Messprinzip für Lenkwinkel und Gaspedalstellung
    - Die Auflösung soll mindestens 14bit bei 360° Drehwinkel, das entspricht $45{,}5\ \frac{Samples}{Grad}$, betragen
- Montagemöglichkeit mit Flansch
- Wenig mechanischer Verschleiß
- Nach Möglichkeit kein mechanischer Anschlag
- Verfügbarkeit und Kosten

Durch das Anforderungsprofil wurden wir auf die Firma Megatron aufmerksam, welche Winkelsensoren mit verschiedensten Schnittstellen und Montagemöglichkeiten im Angebot hat. Tatsächlich konnten wir einen kontaktlosen Hall-Effekt Sensor mit SPI Schnittstelle (ENS22F) ausfindig machen, der alle genannten Kriterien erfüllt.

Abbildung 3.1: Winkelsensor - ENS22F

An dieser Stelle sei erwähnt, dass wir alle Sensoren von der Firma Megatron durch ein Sponsoring beziehen und somit für das Team wertvolles Geld sparen konnten.

3.2.2. Bremsdruck

Da sich die Bremsflüssigkeit nur in kleinstem Maße komprimieren lässt, sei hier gesagt, dass es Sinn frei wäre auch den Pedalweg des Bremspedals in ähnlicher Auflösung zu messen wie Lenkwinkel oder Gaspedalstellung. Daher genügt es für die Bremskraft, wenn Drücke im Bereich von 0-80bar gemessen werden können.

Im Vorjahr wurde ein Drucksensor von Magneti Marelli (OPS04) mit analogem Ausgangssignal verwendet. Da für den Bremsdruck keine besondere Genauigkeit gefordert ist, können auch Sensoren mit analogem Ausgangssignal verwendet werden.

Durch ein Sponsoring von i2s wurde als Sensor ein Drucksensor (VSP1630) mit ratiometrischem analogem Ausgang gewählt.

Abbildung 3.2: Bremsdrucksensor – VSP16xx

Der Sensor liefert bei einer Versorgungsspannung einen Ausgangsspannungspegel von 0,5V – 4,5V. Dass der Ausgang ratiometrisch ist, sollte sich als äußerst nützlich erweisen. Dies wird aber in Kapitel 4.2 nochmals genauer erläutert.

3.3.Mikroprozessor

Für die Wahl des Mikrocontrollers sollten zwar jene in Kapitel 2.3.1 vorgestellten „Dual Core“ Mikrocontroller verwendet werden jedoch zeigt auch die Erfahrung des TUGraz Racing Teams, dass es während einer Saison aus mehreren Gründen mehr als unvorteilhaft ist, wenn nicht alle Entwickler ähnliche Mikrocontroller benutzen und dieselbe Entwicklungsumgebung zu Verfügung steht. Daher wurde ein generelles Anforderungsprofil für die verwendete CPU Serie des E-Power Racing Teams erstellt:

- Mindestens ein CAN Controller integriert
- Serielle Kanäle für SPI und UART
- 10bit ADC integriert
- Automotive erprobt
- Von Hand lötbare Package Varianten
- Schnell verfügbar
- Kostengünstig

Da es viele Hersteller gibt, die Controller anbieten, welche die oben stehen Anforderungen erfüllen, jedoch Infineon schon vorab ein Sponsoring zusicherte, sollte aus Kostengründen die Entscheidung auf Infineon Controller fallen. Die 16bit Controller der XC2000 Serie bieten alle gewünschten Funktionen und sind in verschiedenen Ausführungen Größe und Peripherie betreffend in kurzer Zeit verfügbar.

Daher wurde speziell an die Anforderungen der X-by-Wire Box angepasst, der XC2336B [6] als Controller gewählt welcher ein LQFP-64 Package besitzt und daher auch mit Hand gut lötbar bleibt. Um das Expose Pad des Controllers auch anlöten zu können, wird anstelle eines SMD Pads, eine größere Durchkontaktierung vorgesehen, welche später mit Lötzinn aufgefüllt werden soll.

Durch die Auswahl dieses Controllers stehen folgende wichtige Features zu Verfügung:

- Bis zu 320kB on-chip Programm Speicher
- Zwei 10-bit A/D Umsetzer
- 16-Channel Capture/Compare Unit
- 5 Timer
- 4 Serielle Kanäle (UART,SPI, LIN, IIC)
- On-Chip CAN Interface mit 2 Kanälen
- Single Power Supply 3,0V - 5,5V
- On-Chip Debug
- …

Um die Reglements zu erfüllen wurde auch ein Supervisor Controller CIC61508 [7], auch aus dem Hause Infineon eingeplant, welcher speziell dafür entwickelt wurde die Funktion des Hauptcontrollers, sowie Betriebsspannungen zu überwachen und den ASIL-D Standard erfüllt. Dieser wird auch über eine weitere SPI Schnittstelle mit dem Hauptcontroller XC2336B verbunden, um diesen überwachen zu können. Dazu ein Applikationsvorschlag der Firma Infineon:

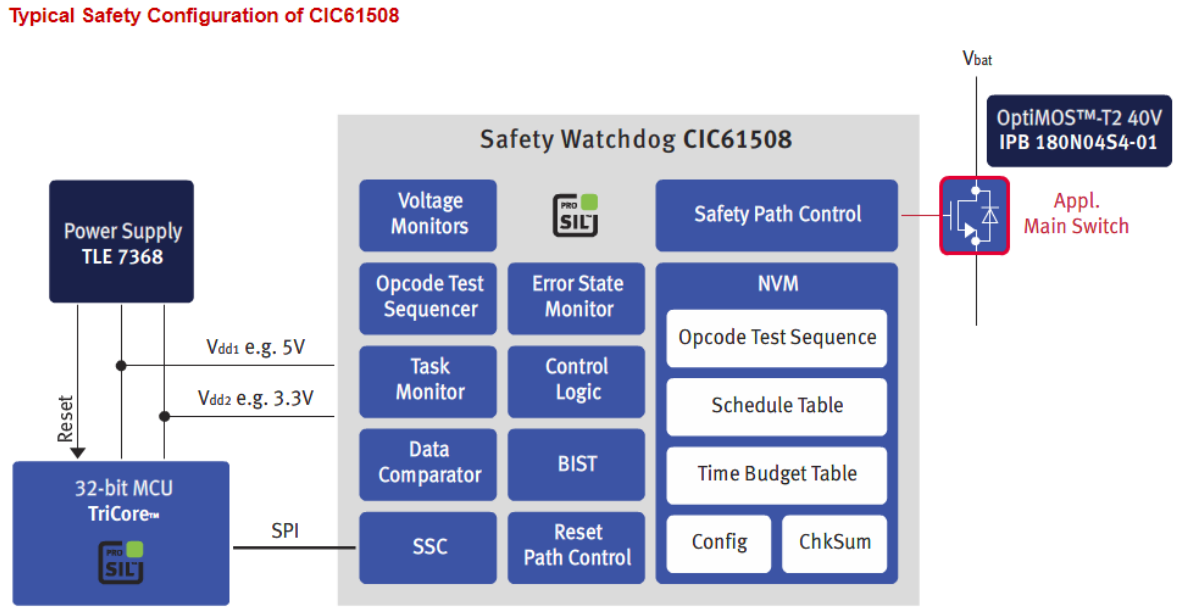


Abbildung 3.3: Applikationsbeispiel für Safety Watchdog - CIC61508

4. Implementation und Realisierung im MaxWheel2011

Nachdem alle wichtigen Sensoren definiert waren, wurde ein Blockschaltbild erstellt, um einen Überblick geben zu können, welche Schnittstellen in welcher Geschwindigkeit zur Realisierung dieses Projekts wirklich notwendig sind.

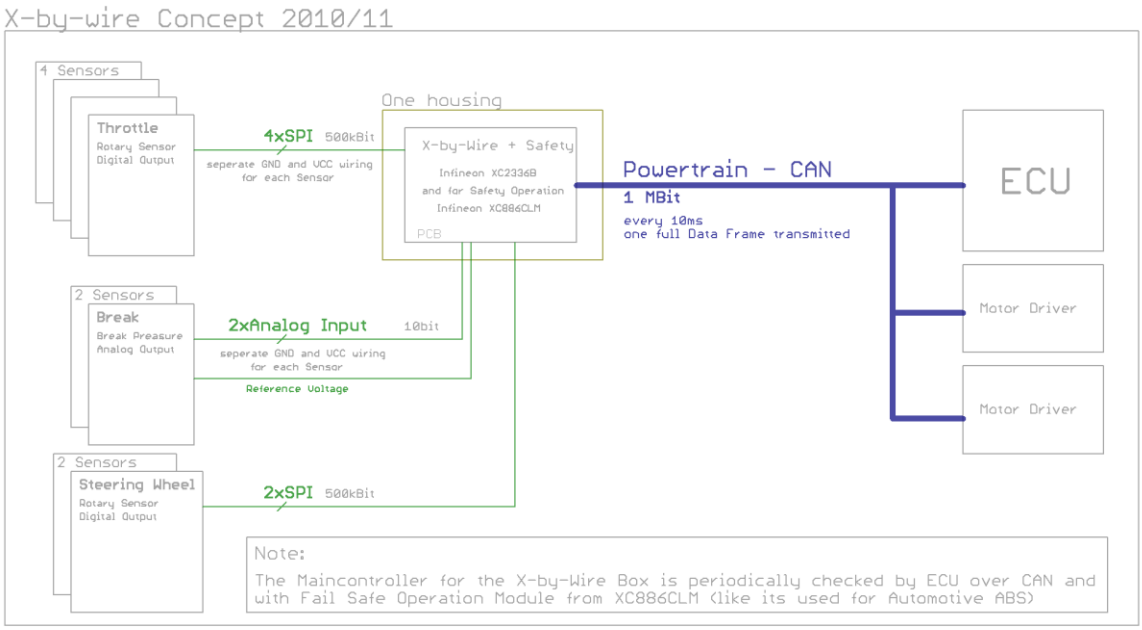


Abbildung 4.1: X-by-Wire Konzept 2010/2011

4.1. Wichtige Komponenten

Bei der X-by-Wire Box spielt auch die Spannungsversorgung der zu designenden Platine eine besondere Rolle. Sie sollte im gesamten 5V und 3,3V Ausgangsspannungen sowie strombegrenzte Ausgänge bereitstellen. Insgesamt sollten 8 Sensoren, mit separater Verkabelung an strombegrenzen Ausgängen angeschlossen werden.

Von den Sensoren ist bekannt, dass jeder der Sensoren maximal 16mA@5V benötigt. Da Infineon ja alle benötigten Bauteile sponsert und auch Multi- Voltage Power Supply IC's im Programm hat wurde mit der Suche dort begonnen.

Tatsächlich konnte ein IC ausfindig gemacht werden, welcher für dieses Projekt nahezu prädestiniert ist. Der TLE6368-G2 [8] stellt nicht nur 2-mal 3,3V(500/350mA) und 1-mal 5V(800mA) als Versorgungsspannung für den Controller bereit, sondern auch 6-mal 5V mit 17mA strombegrenzte Ausgänge. Er beruht auf der Funktionsweise eines Stepdown Wandlers und unterstützt einen großen Eingangsspannungsbereich von 5,5V – 60V.

Abbildung 4.2: TLE6368-G2

Nachdem mit diesem Multi Spannungsregler nicht nur der Hauptcontroller (XC2336B) und der Safety Watchdog Controller (CIC61508) sondern auch alle über SPI angeschlossenen Sensoren versorgt werden können, musste nur noch ein Tracker Baustein für den Analogen Sensor gefunden werden. Da Infineon eine Vielzahl von Tracker IC`s anbietet konnte hier mit Leichtigkeit ein passender ausgesucht werden. Schließlich haben wir uns für den TLE4254GS [9] entschieden, da dieser einen Status Ausgang, sowie einen Enable Eingang besitzt.

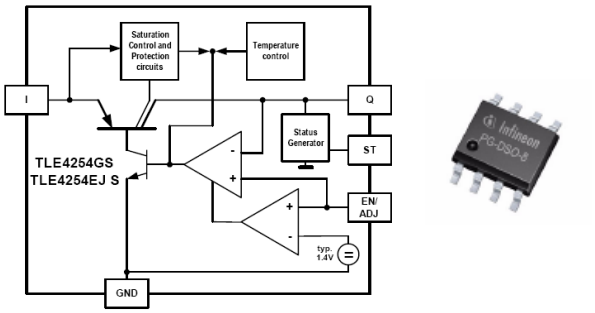


Abbildung 4.3: TLE4254GS

4.2.PCB Design

Beim PCB Design sollten grundlegend folgende Punkte beachtet werden:

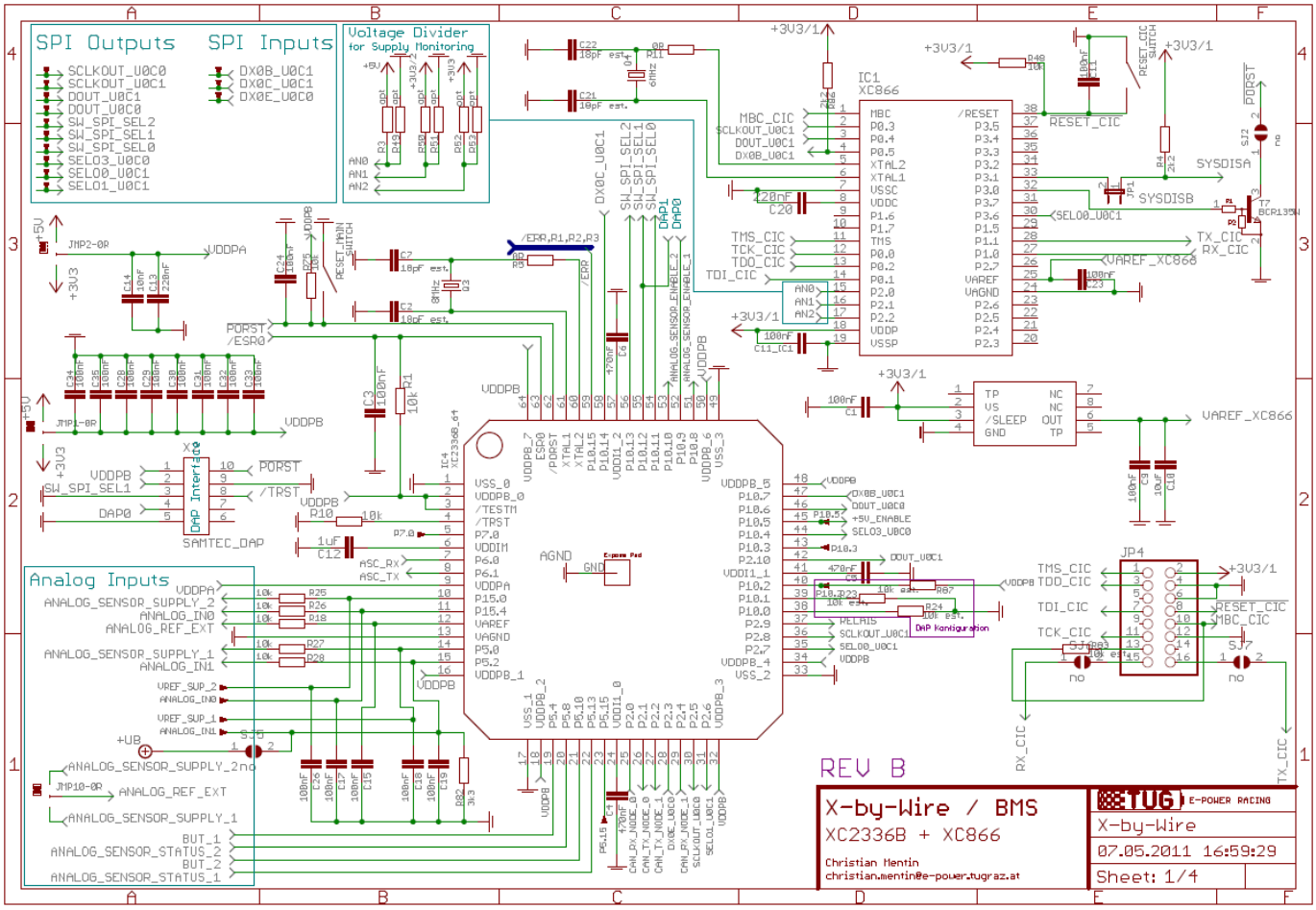
- Um frequenzstabile Operation zu gewährleisten soll ein 8MHz Quarz verwendet werden
- Signalleitungen sollen so kurz als möglich gehalten werden
- Schaltplan übersichtlich halten
- Alle freien Flächen mit GND Planes füllen
- Keine Flaschenhälse in Versorgungsleitungen
- Wenn möglich nur 2 Layer
- Bei der Gehäusewahl sind Bauhöhen der Bauteil und Durchbrüche für die verwendeten Stecker zu berücksichtigen
- Das verwendete Gehäuse kompatibel zur PCB halten (Befestigungslöcher)
- Im Fehlerfall soll die Platine leicht austauschbar sein, daher sollen auch die Verbindungen von den Gehäusesteckern auf die Platine hin steckbar sein
- Genügend Debug LED’s und Messpunkte vorsehen
- And always remember: Do it first time right!

4.2.1. Schaltplan

Um dieses Dokument nicht unnötig in die Länge zu ziehen habe ich versucht, nur die relevantesten Teile des Schaltplanes hier zu dokumentieren.

Vorab möchte ich aber erwähnen, dass das Design mit vielen optionalen Komponenten geplant wurde. So wurde beispielsweise, um auf alle Eventualitäten vorbereitet zu sein, die Versorgungsspannung frei

konfigurierbar (3,3V oder 5V) für nahezu alle Komponenten implementiert, das meiner Meinung nach viele Vorteile bei der Inbetriebnahme und beim Testen der Platine bietet.



Die erste Seite des Schaltplanes zeigt den Main Controller XC2336B(IC4) und dessen Supervisor(IC1). Damit der Safty Controller die Spannungen auf der Platine genau überwachen kann, hat dieser eine eigene externe Spannungsreferenz bekommen, welche rechts im Bild zu sehen ist. Die zu messenden Spannungen werden dann über Widerstandsspannungsteiler bei auf niedrigere Pegel gebracht, um in dem durch die Referenzspannung festgelegten, messbaren Bereich zu bleiben. Durch verschiedene der SYSDISx Ausgänge hat der Safety Controller die Möglichkeit:

1. Den XC2336B zu reseten (SYSDISB)
2. Gezielt Spannungsregler des TLE6368 Multi Voltage Supply IC's im zu reseten.

Die Programmier- und Debugschnittstelle (DAP Interface) ist ein 10poliger Connector, links in der Abbildung. Das Array von Kondensatoren ebenfalls auf der linken Seite sind Stützkondensatoren, welche so nah als möglich an den Versorgungspins des Main-Controllers platziert werden sollen.

Zur Versorgung des Hauptcontrollers sei gesagt, dass dieser intern aus der angelegten Spannung über einen LDO 1,8V für den Core erzeugt, dessen 1,8V nochmal an 3 Pins mit je 470nF gestützt sind. Die 3,3V sind eindeutig als Versorgungsspannung vorzuziehen, da eine niedrigere Betriebsspannung

Abbildung 4.4: Schaltplan Seite 1 – XC2336B + CIC61508

weniger Verlustleistung und eine niedrigere Eigenerwärmung klare Vorteile dieser Variante sind. Da das Analoge Port autonom vom Rest des Controllers versorgt werden kann, wurde für diesen Port als Betriebsspannung 5V gewählt, um analoge Spannungen bis zu 5V messen zu können und auch weitere 5V digital Inputs zu Verfügung zu haben. Zusätzlich wurden die analogen Eingänge mit Tiefpässen beschalten, um hochfrequente Störungen etwas zu dämpfen. Zusätzlich soll versucht werden nicht mit den maximal möglichen 80MHz sondern mit weniger zu arbeiten, da höhere Taktraten auch mehr Verluste bedeuten.

Da die Ausgänge der analogen Bremsdruck Sensoren ratiometrisch sind, sie stehen also in direktem Zusammenhang mit der Versorgungsspannung des Sensors, wurde folgender Trick angewandt. Der Tracker, welcher für die analogen Sensoren verwendet wird, hat einen Adjust Eingang, an dem die am Ausgang einzustellende Spannung vorgegeben werden muss. Daher sollte die Adjust Spannung wieder als Referenzspannung auf den Controller rückgeführt werden, da somit das ratiometrische Verhalten der Ausgangsspannung direkt in die Messung miteinfließt und nicht mehr berücksichtigt werden muss.

Eine interessante Erkenntnis zu den analogen Ports (Port5, 6 und 7) des XC2336B lässt sich unter Punkt 5.1 Fehler und Bugs Hardware nachlesen.

Zur Erläuterung wie die Absicherung der SPI Schnittstellen implementiert wurde, möchte ich folgenden Ausschnitt aus Seite 2 des Schaltplanes heranziehen.

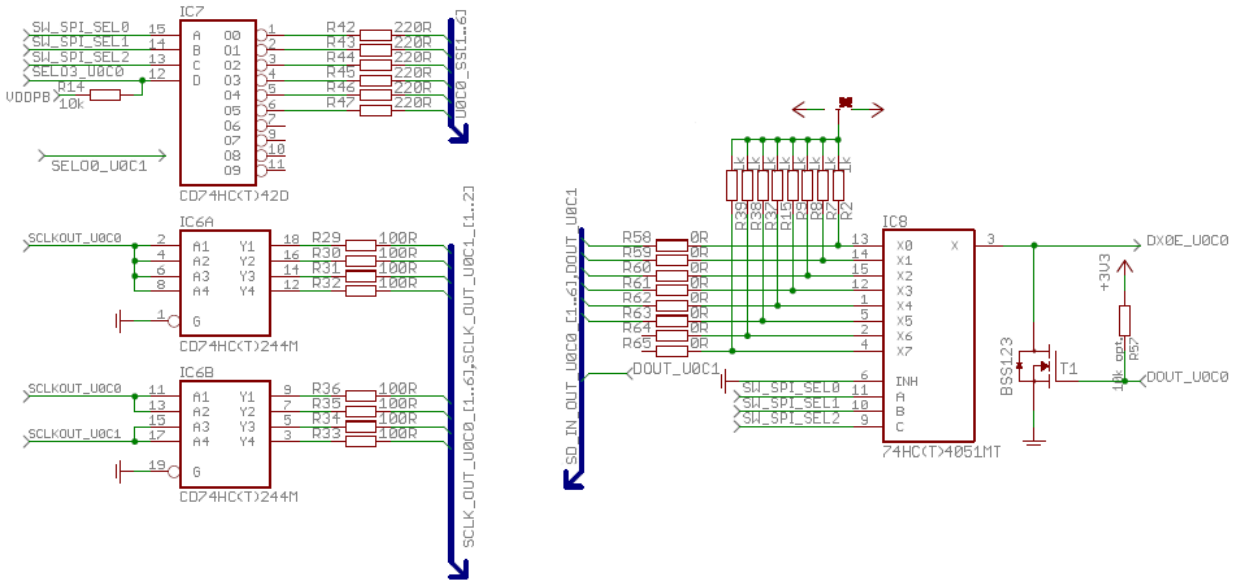


Abbildung 4.5: Schaltplan Teil Seite 2: SPI Interface – kurzschlussfest

Die SPI Schnittstelle basiert auf Basis einer Datenleitung, daher sind DOUT_U0C0 (SPI Data out)und Dx0E_U0C0 (SPI Data In) auf einer einzigen Leitung realisiert. Diese Datenleitung wird durch einen Multiplexer mit allen 6 SPI Sensoren geteilt. Zusätzlich wurde die SPI Schnittstelle zu den Sensoren über verschiedene Treiberbausteine, vom Controller getrennt. Die Ausgangswiderstände sind so dimensioniert, dass diese im Kurzschlussfall (Kurzschluss gegen GND oder +5V) eine Beschädigung des Treiberbausteines verhindern. Eine detaillierte Erklärung lasse ich hier mal außen vor, da durch Einsicht der Datenblätter die Funktionsweise eindeutig nachvollzogen werden kann.

Als letztes möchte ich noch einen kurzen Einblick in die Versorgung der gesamten PCB geben.

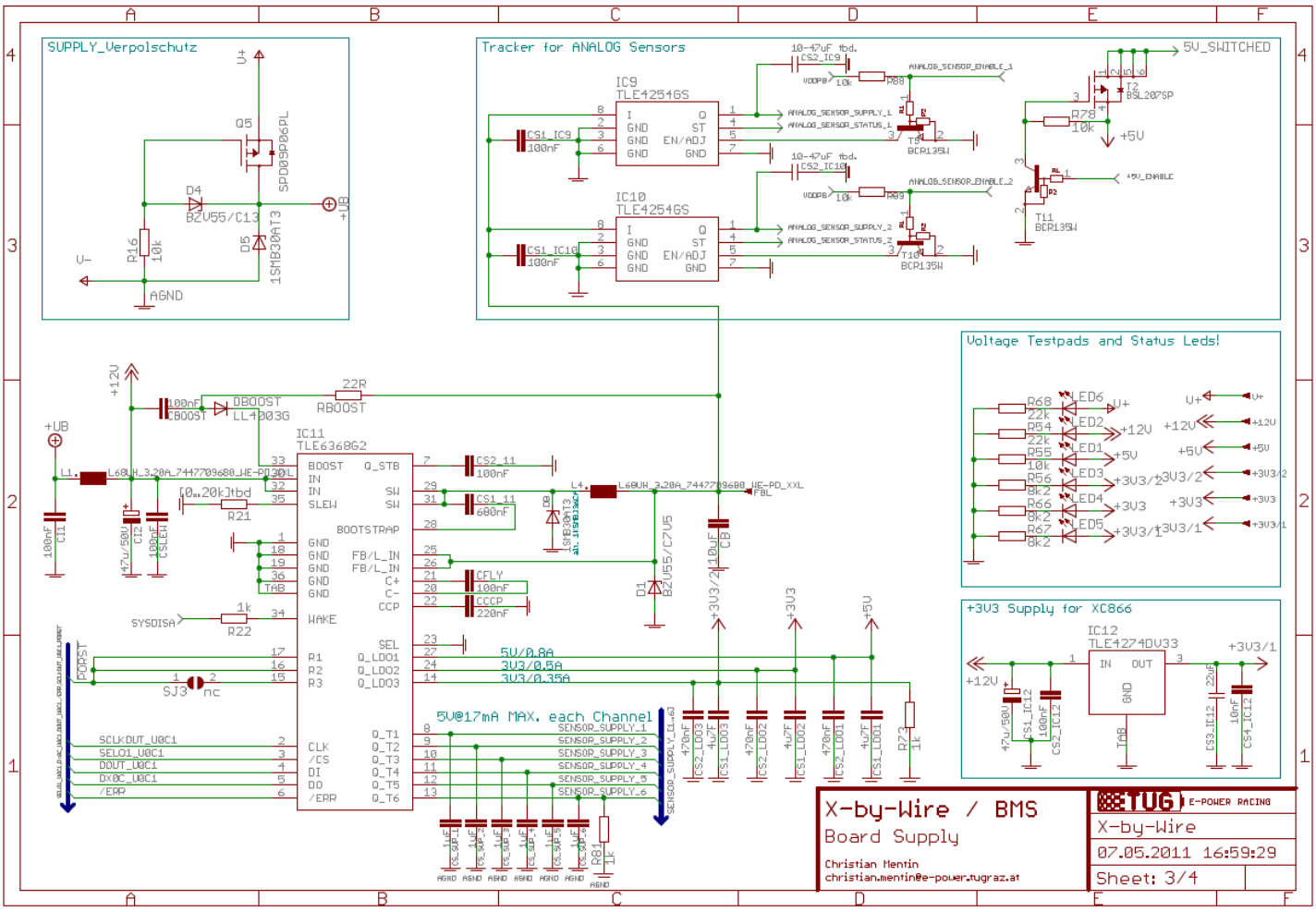


Abbildung 4.6: Schaltplan Seite 3 – PCB Supply

Wie in Abbildung 4.6 zu sehen ist, wurde ein Verpolschutz mit einem P-Kanal MOSFET realisiert, um nicht sofort bei versehentlichem verpoltem Anschluss der Betriebsspannung, das gesamte Board zu beschädigen. Die zusätzlichen 3,3V Supply wurden für den Safty Controller eingebaut, damit dieser, wenn er den Multi Spannungsregler in Reset hält, selbst noch im ON-State verbleiben kann. Zu den beiden Trackern im oberen Bereich der Schaltung sei gesagt, dass diese, in dieser Schaltungsvariante leider nicht das gewünschte in Punkt 4.1 beschriebene Verhalten zeigen können. Da dies ein klarer Fehler im Design des Schaltplanes war, wird im Abschnitt 5.1 nochmals näher darauf eingegangen.

Die 4. Seite des Schaltplanes möchte ich in diesem Dokument nicht einpflegen, da auf dieser nur die Steckverbindungen zu den Gehäusesteckern hin enthalten sind.

4.2.2. Layout

Da es besonders wichtig war ein kompaktes Design zu entwerfen, welches in ein kleines und nicht allzu hohes Gehäuse passt, musste dies auch von Anfang an berücksichtigt werden. Nach dem platzieren der Bauteile in zusammengehörige Gruppen konnte schon eine Gehäusegröße erahnt werden. Daher wurde schon früh versucht ein passendes zu finden. Schließlich habe ich mich aufgrund der kleinen Maße von 123x96x35mm für ein Gehäuse von Hammond (1591XXGSFL) entschieden. Bei dieser Gehäusevariante wäre es möglich, die ausgewählten, für den Motorsport entwickelten Stecker aus der Serie 8STA von Soriau, seitlich und/oder, bei geschickter Platzierung der Bauteile, auch auf der Oberseite des Gehäuses zu integrieren.

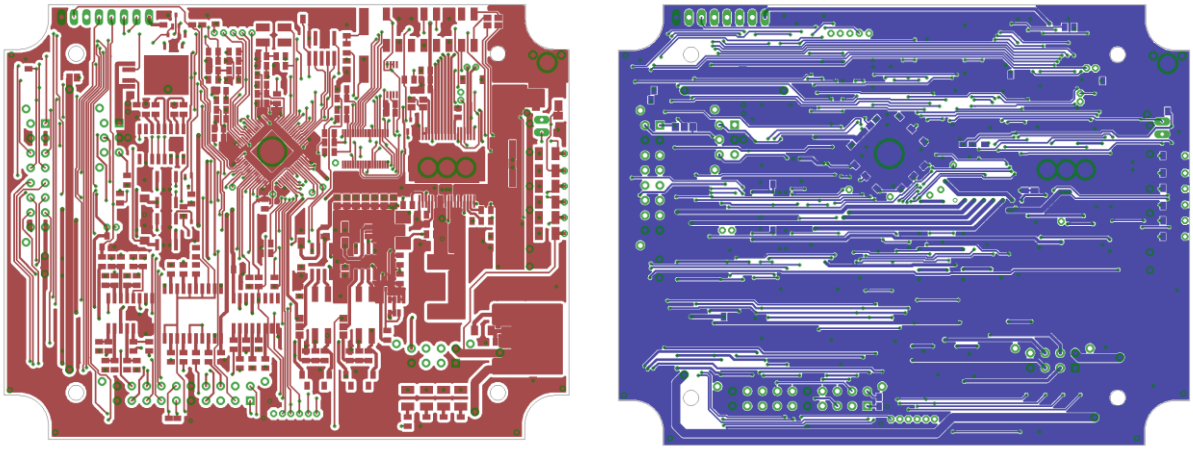


Abbildung 4.7: PCB Layout - Top und Bottom Seite

Nach der Fertigstellung des Layouts wurden die Platinen bei Multi-CB, ebenfalls durch ein Sponsoring, gefertigt. Eine Revision des Layouts später, welche wegen eines Fehlers bei der Nummerierung der Pins am Hauptcontroller durchgeführt werden musste, sah die bestückte Platine schlussendlich so aus

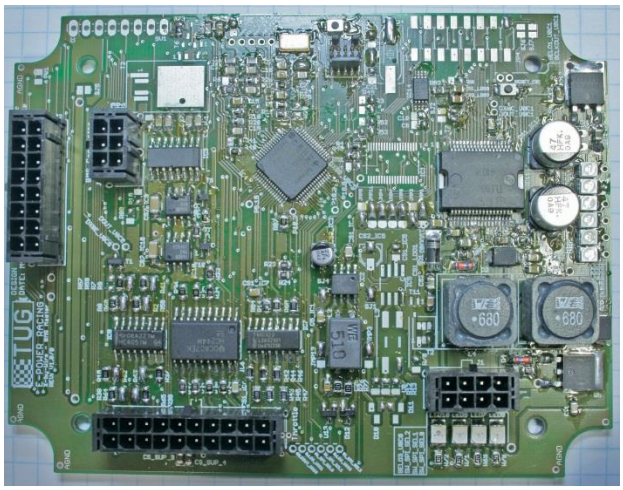


Abbildung 4.8: bestückte Platine der X-by-Wire Box

Aufgrund von zeitlichen Problemen wurde der Safty Controller für die erste Version der X-by-Wire Box noch nicht bestückt. Eine Implementation des Controllers, Hardware sowie Softwareseitig, sollte dann zu einem späteren Zeitpunkt erfolgen.

4.3.Software

Grundsätzlich sollte beim Design der Software besonders auf strukturierte Programmierung wertgelegt werden. Globale Variablen sollten so gut als möglich vermieden, ein Codingstandard von Anfang bis Ende durchgezogen werden. Die Namensgebung der Variablen sollte sowohl einfach als auch „sprechend“ sein, sodass auch jemand den Code verstehen kann, obwohl er mit der Programmierung dieses Codes nichts zu tun hatte.

Wichtige Anforderungen an die Software:

- Das System soll Fail Silent sein
- Die Auswertung der Sensorwerte sollte einfach gehalten werden, Fließkomma Operationen sind aufgrund des Fehlens einer FPU am XC2000 zu vermeiden
- Validierte Sensorwerte müssen alle 10ms über den CAN Bus übertragen werden
- Intelligente Sensorüberwachung, Limit- und Plausibilitätschecks implementieren
- Debug Modus – Anzeige der validen Sensoren am Dashboard
- Auswahl eines, durch den Fahrer am Dashboard selektierten, vertrauenswürdigen Sensor

4.3.1. Datenaufbereitung

Zu detailliert möchte ich in dieser Arbeit nicht auf die konkreten ausprogrammierten Algorithmen eingehen, jedoch möchte ich kurz erläutern wie die prinzipielle Auswertung der Sensorwerte erfolgte. Als Beispiel werde ich einen Sensor, welcher die Gaspedalstellung ermittelt, heranziehen. Daher soll die nachfolgende Grafik kurz den Verlauf der Sensorwerte erklären.

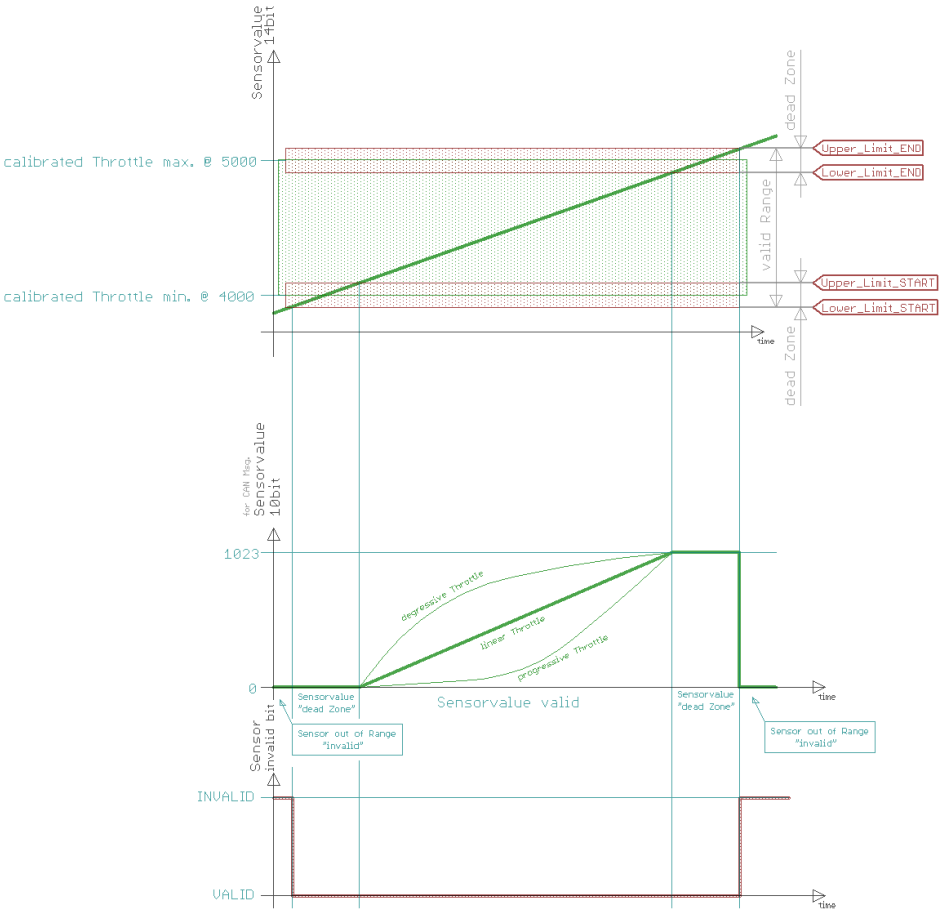


Abbildung 4.9: Datenaufbereitung Gaspedal – Sensor

Das Pedal besitzt ca. 6° Pedalweg, wobei je nach Einbauposition des Sensors, der Winkelsensor immer andere Sensorwerte liefern wird. Daher wurde für die Beispielabbildung ein fiktiver Wert als „Anfangs“ Wert des Pedals, also die unbetätigte Pedalstellung, mit 4000 Samples (14bit = 16383 Samples) angenommen. Das voll durchgetretene Pedal erwirkt am Sensor einen Messwert von 5000 Samples. Dieser Bewegungsbereich ist in der Abbildung grün punktiert dargestellt. Jeder Wert der nicht in diesen Bereich fällt, wäre daher kein valider Sensorwert mehr. Daher wurden zu diesen Grenzwerten noch Limits eingefügt. Die hier sogenannten „Tot“ Zonen des Pedals sollen ein präziseres Steuern des Fahrzeuges insofern erleichtern, dass sich das Fahrzeug nicht schon bei kleinsten Veränderungen an der Pedalstellung in Bewegung setzt (wenn der Fahrer z.B.: seinen Fuß auf das Pedal legt). Genauso sollen Vibrationen (bei Vollgas) und mechanische Verformungen berücksichtigt werden. Wenn der Fahrer das Pedal um mehr als 100% bewegt, z.B.: durch größere Krafteinwirkung, soll dies natürlich auch zu keinem Fehler führen (geforderte Fehlertoleranz). Durch die Tot - Zonen des Pedals wurde dies in recht einfacher Form implementiert, da der „valid Range“ des Sensors dadurch noch ein wenig größer wird. Diese „Tot“ – Zonen sind momentan über Defines im Code implementiert.

Die X-by-Wire Box soll am CAN Bus ein Fail Silent System sein, daher möchte ich nochmals auf Abbildung 4.9 zurückgehen, in welcher ersichtlich wird, wie die ermittelten Sensorwerte auf den Bus umgelegt werden. Über verschiedene Algorithmen soll der aktuelle Sensorwert auf degressive, progressive oder lineare Pedalkennlinien in einen Bereich von 0 – 1023, was einer Auflösung von 10bit pro Sensor entspricht, umgerechnet werden. Befindet sich der Sensor nicht im validen Bereich, so wird dieser mit der Wert 0 am Bus dargestellt (im Debug Mode) bzw. das Invalid Bit des entsprechenden Sensors in der Standard CAN Message gesetzt. Über das Dashboard wird der Fahrer informiert, welcher Sensor ausgefallen ist.

Durch einen Mehrheitsentscheid über alle validen Sensoren mit anschließender Mittelwertbildung werden die vier Sensoren am Gaspedal zu einem Wert zusammengefasst, und als „Throttle Position“ über den CAN Bus zum Hauptsteuergerät übermittelt. Somit konnte wie gefordert ein fehlertolerantes Fail Silent System, welches im Fehlerfall 0 als Wert für die Pedalstellung liefert, realisiert werden.

Da die Datenaufbereitung für Lenkwinkel und Bremsdruck auf selbigem Prinzip aufgebaut ist, möchte ich diese hier nicht mehr genauer ausführen.

4.3.2. “Echtzeit” CAN-BUS

In der Automobil Industrie wurde schon früh begonnen Steuergeräte mit Hilfe von Bussystemen miteinander zu verbinden. Durch die hohe Störsicherheit und die dadurch realisierbare Datenrate wurde dem CAN Bus „Echtzeitfähigkeit“ attestiert. Jedoch ist dies nur der Fall wenn die Buslast nur äußerst gering ist. Da der CAN Bus mittlerweile für das transferieren großer Datenmengen mehr oder weniger „missbraucht“ wird, muss die Echtzeitfähigkeit des Bussystems immer wieder in Frage gestellt werden und für den Einzelfall betrachtet werden. In unserem Fall ist es notwendig für die Regelung der Motoren alle 10ms die aktuellen Werte zur Hauptsteuereinheit (ECU) zu übertragen.

Da auf diesem CAN Bus, dessen Übertragungsgeschwindigkeit 1MBit ist, pro 10ms Zeitschlitz nur 3-6 weitere Nachrichten von der ECU zu den anderen Geräten übertragen werden, ist es möglich den Nachrichten der X-by-Wire Box „Echtzeitfähigkeit“ zu attestieren, da durch Priorisierung der Nachrichten, maximal zwei höher priorisierte Messages gesendet werden, die maximale Verzögerung dadurch <1ms bleibt.

Für genauere Abhandlungen zum Thema Buslastanalyse für CAN Busse im Automobilbereich möchte ich auf eine Masterarbeit[10] verweisen.

5. Fehler und Bugs

Trotz ständiger Kontrollen und Bemühungen jeden Arbeitsschritt nach bestem Wissen und Gewissen durchzuführen, passierten leider einige kleinere Fehler welche in diesem Abschnitt genauer erläutert werden sollen. Weiters sollen für die Saison 2011/2012 bekannte Fehler eliminiert und eventuelle Verbesserungen durchgeführt werden.

5.1.Hardware

Bei der Erstellung der Bauteilbibliothek wurde bei der Beschriftung der Pins des Hauptcontrollers, ein Fehler in der Nummerierung gemacht. Daher mussten auf jeweils 3 Seiten des Controllers 8 Pins ausgekreuzt werden, wobei hier nur die wichtigsten Funktionalitäten implementiert, andere nicht korrekte Verbindungen einfach aufgetrennt wurden. Aus zeitlichen Gründen wurde die erste Platine per Hand adaptiert (Abbildung 5.1), um die Zeit nutzen zu können, bis die überarbeitete Platine produziert wurde.

Abbildung 5.1: “gepatchte” Platine vor der ersten Revision

Ursprünglich sollte es auch möglich sein dass jeder einzelne der Spannungsregler des Multi Spannungsregler IC’s vom Safety Controller resettiert werden kann. Dieses Feature wird dann wohl erst im Herbst nächsten Jahres implementiert werden, da leider darauf vergessen wurde.

Interessant war auch dass das analoge Port des Hauptcontrollers (XC2336B) als reines Eingangsport implementiert ist. Zusätzlich verfügt dieses Eingangsport über keine Zusatzfunktionen wie alle anderen Ports des Controllers, um beispielsweise interne Pullup`s an die Eingänge schalten zu können. Daher müssen diese bei der nächsten Variante der X-by-Wire Box implementiert werden, da Pullup`s für die Status Ausgänge der Tracker für die Bremsdrucksensoren fehlten.

Zur Befestigung der Platine wurden zwar Montagelöcher vorgesehen, diese konnten jedoch leider nicht benutzt werden, da die Positionen nicht ganz korrekt sind. Auf den Bildern der bestückten Platine ist ersichtlich, dass es einige Steckverbinder gibt. Nachdem die vielen Einzeladern im Gehäuse zwar

Platz hatten, drückten diese aber die Platine schon auf den Gehäuseboden, sodass eine Befestigung der Platine über die vorgesehenen Montagelöcher nicht notwendig war. Die Platine wurde daher nur mit doppelseitigem Klebeband in ihrer Position fixiert. Für ein eventuelles Re-Design für die Saison 2011/2012 soll die Platinengröße jedoch noch etwas verringert werden, um das Gehäuse leichter schließen zu können, sowie die Montagelöcher an ihrer richtige Position versetzt werden.

Bei ersten Tests der SPI Schnittstelle, wurde die maximal mögliche Geschwindigkeit der Schnittstelle ermittelt. Da der Sensor mit 7MBit funktionieren sollte wurde die Schnittstelle mit 1MBit getestet. Doch schon bei dieser Geschwindigkeit konnte festgestellt werden, dass übermäßig viele Fehler in der Datenübertragung vom Sensor zum Controller auftraten. Nach einigen Messungen und Recherchen wurde festgestellt dass die nicht ideal gerouteten Leitungen und Kabel zu große parasitäre Kapazitäten besitzen, um solche Geschwindigkeiten erreichen zu können, zumindest mit diesen Ausgangstreibern. Für die nächste Platinenversion soll dieses Verhalten durch andere Treiberbausteine und besser gewählte Leitungsführungen verbessert werden.

5.2.Software

Als größtes Problem in der X-by-Wire Software möchte ich die EEPROM Emulation ansprechen. Da die Controller der XC2000er Serie keinen EEProm besitzen ist es notwendig den internen Flash Speicher als solchen zu benutzen. In diesem Fall sollten Kalibrierdaten und andere selten aber dynamisch änderbare Daten im EEProm abgelegt werden, um beim nächsten Systemstart zu Verfügung zu stehen. Leider konnte aus bisher noch unerklärlichen Gründen die EEProm Emulation zwar gestartet werden, doch benötigt die Initialisierung, welche bei jedem Systemstart durchgeführt werden muss, zwischen 0,5-300 Sekunden! Solange dieses Problem nicht behoben wurde, sind alle Kalibrierdaten fix im Code als Defines verankert, da eine zu lange andauernde Initialisierung auch das gesamte Fahrzeug für diesen Zeitraum lahmlegt.

Zum Debugging wurde eine Serielle Konsole in sehr einfacher Form implementiert, welche es ermöglich alle kalibrierten Parameter, sowie Status der Sensoren und alle nur erdenklichen Werte von den Sensoren abfragen zu können. Um jedoch später im Fahrzeug und auch von der Mobile Unit aus, alle möglichen Informationen von der X-by-Wire Box erhalten zu können, sollten alle Befehle und daraus resultierenden gesendeten Daten, welche momentan in der seriellen Konsole implementiert sind, auch auf CAN Messages umgelegt werden.

Im letzten Stand der Software wurde der Safty Controller noch nicht implementiert. Dies soll aber noch so rasch als möglich während dieser Saison durchgeführt werden.

Quellenverzeichnis:

[1] Georg Macher – Drive-by-Wire System eines Formula Student Electric Boliden

[2] Brian Selic, IBM Staff – Fault tolerance techniques for distributed systems

[3] Formula Student Electric Germany - Rules

[4] Formula Student UK - Rules

[5] Formula Student Italien - Rules

[6] Infineon – XC2336B Datasheet

[7] Infineon - CIC61508

[8] Infineon – TLE6368-G2

[9] Infineon – TLE4254GS

[10] Masterarbeit, Daniel Müller – Buslastanalyse- und senkung der CAN-Busse im Automobil