Abschlussbericht für das Teilvorhaben

Zuverlässigkeit und Fehlertoleranz von Komponenten des Elektroantriebs

durchgeführt im Rahmen des Verbundprojekts

Zuverlässigkeit und Sicherheit von Elektrofahrzeugen (ZuSE)

GEFÖRDERT VOM

Bundesministerium
für Bildung
und Forschung

Bezug:
Bekanntmachung des BMBF
von Richtlinien über die Förderung von
Schlüsseltechnologien für die
Elektromobilität "Energieeffiziente und
sichere Elektromobilität" (STROM 2)

Projektlaufzeit: 1.1.2013 bis 30.6.2016.

Dieser Bericht gibt eine Zusammenfassung der im Teilvorhaben „Zuverlässigkeit und Fehlertoleranz von Komponenten des Elektroantriebs“ des Verbundprojekts ZuSE erzielten Projektergebnisse. Er fasst dazu die Voraussetzungen und die erzielten Teilergebnisse unter Verwendung der Zwischenberichte für die Jahre 2013/14/15 zusammen und ergänzt die bis zum Projektabschluss am 30.6.2016 erzielten Ergebnisse.

Verbundprojektpartner

ZF Friedrichshafen AG
Graf-von-Soden-Platz 1
88038 Friedrichshafen

Verantwortlicher Ansprechpartner

Dr. Roland Geiger
07541-77-7712
roland.geiger@zf.com

Der Bericht greift auf Ergebnisse zurück, die von verschiedenen Personen im Rahmen des Teilvorhabens „Zuverlässigkeit und Fehlertoleranz von Komponenten des Elektroantriebs“ im Projekt ZuSE zusammengefasst wurden.

Folgende Personen seien aufgrund ihrer jeweiligen Rolle für die in diesem Bericht abgehandelten Schwerpunktthemen namentlich genannt:

- Sven Gohl (FKFS / ZF ab 1.7.2015)

- Thomas Riemer (FKFS)

- Bülent Sari (ZF)

Um einen Gesamtüberblick für das Projekt ZuSE zu erlangen benötigt der Leser zusätzlich den entsprechenden Bericht des IVK der Uni Stuttgart.

Inhaltsverzeichnis

1 Einleitung

Sicherheit und Zuverlässigkeit der konventionellen Antriebssysteme haben über viele Jahre, unterstützt durch zahlreiche Maßnahmen in den elektronischen Steuerungen des Antriebs, einen guten und akzeptierten Stand erreicht. Für die Antriebskonzepte der Zukunft, bei denen die Elektrifizierung eine zentrale Rolle einnimmt, sind zusätzliche Anstrengungen erforderlich. Für den Systemanwender müssen, unter Beachtung der Anforderungen an die Systemsicherheit, die Fehlertoleranz und damit die Zuverlässigkeit auf ein notwendiges Maß gebracht werden.

Damit gewinnen die E-Antriebskomponenten von Elektro- und Plug-in-Hybrid-Fahrzeugen an Bedeutung. Gleichzeitig werden technologische Fortschritte für die Steuerung relevant. Für die seien hier insbesondere der Übergang von elektronischen Steuerungen mit Einkernprozessoren hin zu elektronischen Steuerungen mit Mehrkernprozessoren sowie die zunehmend in Hardware vorhandenen Sicherheitsfeatures der Controller genannt. Besonders in Antriebssteuerungen, die bekanntermaßen relativ hohen Anforderungen an die Prozessor-Performance bei gleichzeitig hohen Anforderungen an Zuverlässigkeit unter schwierigen Umgebungsbedingungen wie Hoch-/Tieftemperatur, Vibration, EMV, Feuchtigkeit, Partikel und Vielem mehr unterliegen, macht sich dies besonders bemerkbar.

Grundlegendes Ziel des Teilprojekts war daher, unter Berücksichtigung der Anforderungen an die Systemsicherheit sowie der Besonderheiten der Mehrkernprozessoren, Möglichkeiten zur Erhöhung von Fehlertoleranz und Zuverlässigkeit von Antriebskomponenten in Elektro- und Plug-in-Hybrid- Fahrzeugen aufzuzeigen. Insbesondere wurde damit auch die Analyse von Möglichkeiten zur Beherrschung von E-Fahrzeug-spezifischen neuen Risiken adressiert, die beispielsweise beim Versagen von Komponenten des Elektroantriebs auftreten können. Außerdem stellte sich damit auch die Frage, welche Auswirkungen dies für die Fahrsicherheit nach sich zieht.

Der für das Teilprojekt gewählte Ansatz fokussiert auf den Entwurf von fehlertoleranten Architekturen, für das Vorgehen wurden im Einzelnen folgende Schritte gewählt:

1. Fehlermodelle für E-Antriebskomponenten im Hinblick auf deren Einsatz in Elektro- und Plug-in-Hybrid- Fahrzeugen entwickeln.
2. Mit diesen Modellen verbundene Verfügbarkeits- und Ausfallszenarien und deren Auswirkungen auf die Fahrsicherheit analysieren.
3. Den elektrischen Antrieb im Hinblick auf Zuverlässigkeit und auf das notwendige Fehlertoleranzniveau der elektronischen Steuerung bewerten.
4. Eine fehlertolerante Steuerungsarchitektur für E-Antriebssysteme mittels Anwendung von geeigneten Entwurfsmethoden definieren.
5. Eine systemnahe Simulationsumgebung für die Komponenten der E-Antriebssysteme zur Unterstützung der Arbeiten am Projekt bei ZF und zur Verifikation der Ergebnisse umsetzen und Ergebnisse im Fahrsimulator überprüfen.

ZF befasste sich im Sinne einer anwendungsorientierten Forschung mit dem Thema.

Soweit dies im Projektrahmen möglich war, sollte die Projektbearbeitung mit modernen Simulations- und Entwurfsmethoden unterstützt geschehen. Die Ausgangssituation hierzu war so, dass Simulations- und Entwurfsmethoden in System-, Hardware- und Softwareentwicklung jeweils individuell etabliert waren, dass es allerdings keine übergreifende Toolunterstützung gab. Außerdem waren Aspekte der Sicherheit und Zuverlässigkeit nicht durchgängig auf Architekturebene einbezogen. Für eine systematische Entwicklung technischer Konzepte für Sicherheit oder Zuverlässigkeit ist das aber notwendig.

2 Projektplanung

Die Gesamtplanung für das Projekt ZuSE findet sich im Abschlussbericht der Universität Stuttgart. Die nachfolgende Übersicht stellt den für das Teilprojekt relevanten Ausschnitt der Arbeitsplanung dar und definiert gleichzeitig die abgestimmten Verantwortlichkeiten für die Projektdurchführung und für die Berichterstellung.

Tabelle 1: Relevanter Ausschnitt aus der Arbeitsstruktur ZuSE

AP	AP-Nr	AP-Beschreibung	V
AP 1		**Erfassung von Lastkollektiven für Elektrofahrzeuge**	**IVK**
	AP 1.2	Vorbereitung der Fahrversuche	IVK
	AP 1.5	Auswertung der Fahrversuchsdaten	IVK
AP 2		**Echtzeitsimulation des E-Fahrzeugs und seiner Umgebung**	**IVK**
	AP 2.2	Wechselwirkungen zwischen Antrieb und Fahrdynamik	IVK
AP 4		**Fahrerassistenzsysteme für E-Fahrzeuge**	**IVK**
	AP 4.3	E-Fzg-spez. Eingriffsalgorithmen für Sicherheitsfunktionen	IVK
	AP 4.6	Erweiterte Assistenzsysteme mit Querdynamikeingriff	IVK
AP 5		**Zuverlässigkeit und Fehlertoleranz**	**ZF**
	AP 5.1	Spezifische Ausfallszenarien von Elektroantrieben	ZF
	AP 5.2	Robuste und fehlertolerante Steuerungsarchitektur	ZF
	AP 5.3	Zusatznutzen der fehlertoleranten Systemarchitektur	ZF
	AP 5.4	Realer Elektroantrieb als HW-in-the-Loop	IVK
	AP 5.5	HiL-Erprobung von Assistenzfunktionen mit Antriebseingriff	IVK
AP 6		**Beispielhafte Umsetzung im Fahrsimulator**	**IVK**
	AP 6.4	Absicherung kombinierter Eingriffe im Fahrsimulator	IVK

Der Schwerpunkt der Arbeiten von ZF lag demzufolge im Arbeitspaket 5 „Zuverlässigkeit und Fehlertoleranz“ und da speziell in den Teilarbeitspaketen 5.1 (Spezifische Ausfall-szenarien von Elektroantrieben), 5.2 (Robuste und fehlertolerante Steuerungsarchitektur) sowie 5.3 (Zusatznutzen der fehlertoleranten Systemarchitektur).

Da für die Simulation auf dem HiL in Abstimmung auf Einrichtungen der Uni Stuttgart zurückgegriffen wurde, lagen die Zuständigkeiten für die Teilarbeitspakete 5.4 (Realer Elektroantrieb als HW-in-the-Loop) und 5.5 (HiL-Erprobung von Assistenzfunktionen mit Antriebseingriff) beim IVK der Uni Stuttgart.

Die Arbeitspakete AP1 (Erfassung von Lastkollektiven für Elektrofahrzeuge), AP2 (Echtzeitsimulation des E-Fahrzeugs und seiner Umgebung), AP4 (Fahrerassistenz-

systeme für E-Fahrzeuge) und AP6 (Beispielhafte Umsetzung im Fahrsimulator) lagen gesamthaft in Verantwortung des IVK der Uni Stuttgart. In diesen Arbeitspaketen arbeitete ZF an den jeweils beschriebenen Themen mit, soweit ein Bezug zu Antriebsthemen bestand.

Für das Gesamtprojekt wurde ein 36-monatiger Zeitplan zu Grunde gelegt, aus dem für das Teilvorhaben die nachfolgend abgebildete Planung abgeleitet wurde. Jede farbig hinterlegte Zelle entspricht einem Personenmonat. Die Arbeitspaket-Titel sind aus Darstellungsgründen hier nicht enthalten, sie sind Tabelle 2 zu entnehmen.

Tabelle 2: Ausschnitt aus der zugehörigen Ablaufplanung ZuSE

1 Aktivität einer Person in Summe in diesem Monat

AP-Nr	FKFS Auftr. (PM)	ZF Pers. (PM)	M1									M2									M3									M4
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
AP 1																																						
AP 1.2	1	1																																				
AP 1.5	2	1																																				
AP 2																																						
AP 2.2	2	2																																				
AP 4																																						
AP 4.3		2																																				
AP 4.6		2																																				
AP 5																																						
AP 5.1	12	2																																				
AP 5.2		12																																				
AP 5.3	5	4																																				
AP 5.4	3	1																																				
AP 5.5	3	1																																				
AP 6																																						
AP 6.4	2	3																																				

Rückblickend hat sich gezeigt dass die geplante 3-jährige Bearbeitungszeit operativ auch tatsächlich erforderlich war. Der Projektstart unmittelbar nach einer kurzfristigen Freigabe und Genehmigung durch das BMBF hat dazu geführt, dass sich das Projekt mit dem erforderlichen Personal erst einmal konstituieren musste, wofür ca. ein halbes Jahr erforderlich war. In dieser Phase konnte nur wenig operativ geschehen. Der anfängliche Verzug der Arbeiten und der entsprechend niedrige Mittelverbrauch konnten über den Projektverlauf nicht mehr eingeholt werden und führten in der Konsequenz zu einer kostenneutralen Verlängerung der Projektlaufzeit um ein halbes Jahr. Dies betraf nicht nur das Teilvorhaben sondern das Gesamtprojekt.

Für einen Teil der geplanten Arbeiten wurde das FKFS für die Dauer des Projekts aus folgenden Gründen unterbeauftragt:

1. Es konnte so ein enger Kontakt zur Institutsforschung für das behandelte komplexe Themenfeld sichergestellt werden und auf Fahrzeugebene konnten wichtige Ressourcen der Universität Stuttgart für die Untersuchungen mit verwendet werden.
2. Das FKFS unterstützte mittels der Bindung an die Uni Stuttgart im Forschungsbereich den notwendigen Austausch mit der Uni Stuttgart selbst und mit anderen Forschungseinrichtungen.
3. Die Ausbildung von Fachspezialisten war aufgrund der schwierigen Lage am Arbeitsmarkt dringend geboten. Dies geschah in Form von Doktorarbeiten, die seitens FKFS betreut wurden.

Das FKFS wurde mittels eines Werkvertrags entsprechend der vereinbarten Teilplanung von ZF im Rahmen des Teilprojekts unterbeauftragt.

3 Technische Randbedingungen

In diesem Kapitel werden die wichtigsten Randbedingungen für die Projektdurchführung für die im Teilprojekt durchgeführten Schwerpunktarbeiten beschrieben.

3.1 Funktionale Sicherheit und ISO 26262

Den Arbeiten wurde die ISO 26262 zu Grunde gelegt. Dieser Standard formuliert Anforderungen unter anderem für die Zusammenführung von sicherheitsrelevanten Entwicklungen für Teilsysteme sowie für das Zusammenspiel von Hardware- und Software-Entwicklung für eine elektronische Steuerung (ECU).

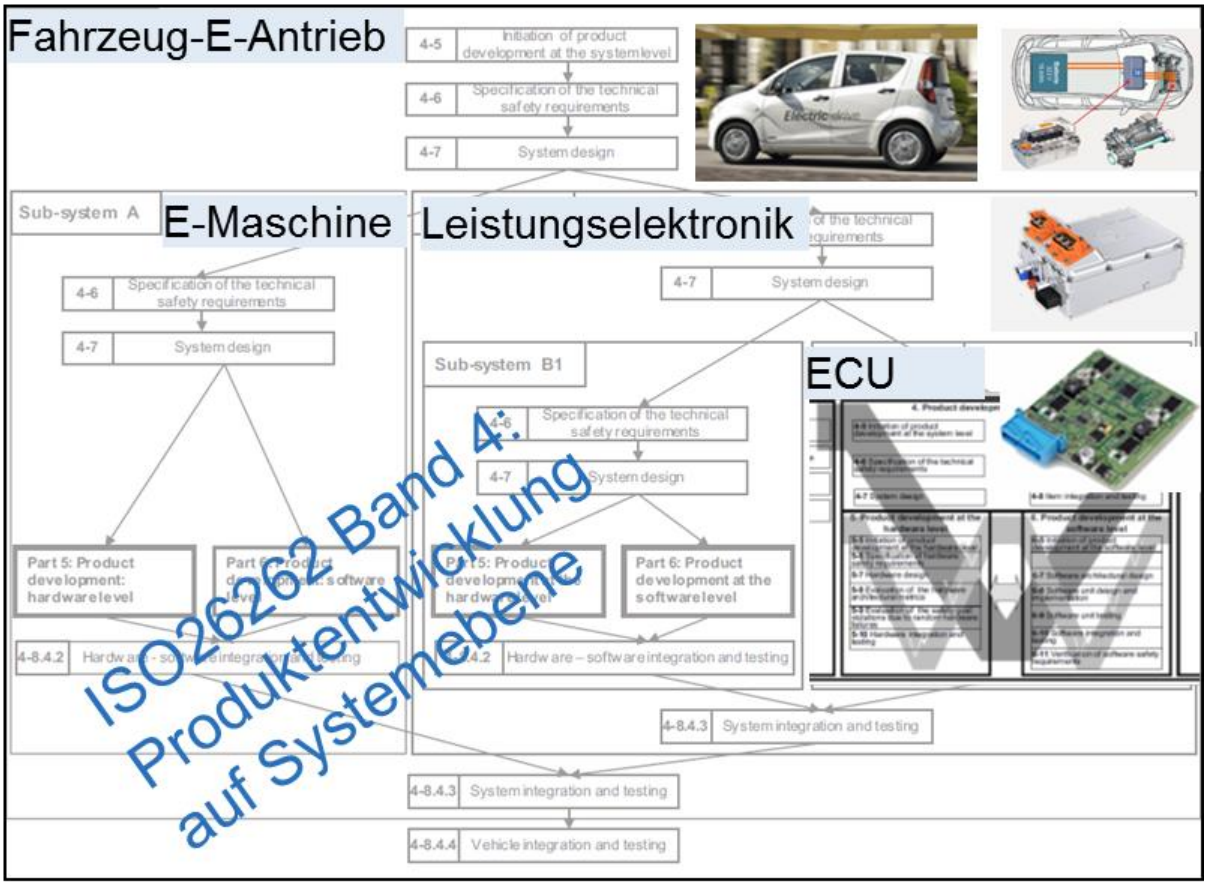


Abbildung 1: Zu Grunde gelegter Entwicklungsprozess

Im Arbeitspaket 5 wurden auf der Basis der relevanten Bände 4 (Systementwicklung), 5 (Hardware-Entwicklung) und 6 (Software-Entwicklung) der ISO 26262 Fehlermodelle für E-Antriebskomponenten identifiziert und die damit verbundenen Ausfallszenarien im Hinblick auf Zuverlässigkeit und Fehlertoleranz bewertet.

Funktionale Sicherheit muss gewährleisten, dass im Falle eines Fehlers von dem System selbst keine Gefährdung ausgeht. Das bedeutet üblicherweise, dass im Fall eines sicherheitsrelevanten Fehlers das System abhängig von der Fehlerschwere und von vorgesehenen Ersatzszenarien entweder abgeschaltet wird, oder wenn dies möglich ist,

mit Einschränkungen weiter betrieben wird. Die Abschaltung und auch sehr einschränkende Ersatzszenarien als Folge von Fehlern können die Verfügbarkeit eines sicherheitsrelevanten Systems erheblich beeinträchtigen. Um also eine möglichst hohe Verfügbarkeit zu erreichen und trotzdem sicher zu sein, gibt es Bedarf für fehlertolerante Systeme.

Es gibt schon länger auch technische Randbedingungen, die eine Toleranz für bestimmte Fehler erzwingen. Das ist immer dann der Fall wenn es zwingend nötig ist, das System trotz eines Fehlers aktiv zu halten, weil z. B. Abschalten dann nicht in einen sicheren Zustand führt. So kann ein elektrischer Lenkeingriff, wenn dieser automatisiert erfolgt, nicht in jedem Systemdesign und jeder Situation einfach abgeschaltet werden. Das Beispiel zeigt, dass der Automatisierungsgrad eine wesentliche Rolle spielt. Tatsächlich ist die zunehmende Automatisierung des Fahrens nur mit mehr Fehlertoleranz möglich.

Die grundlegende Sicherheitsarchitektur wird in vielen Fällen am E-Gas-Konzept ausgerichtet. Es ist ein 3-Ebenen-Konzept und in Abbildung 2 schematisch dargestellt.

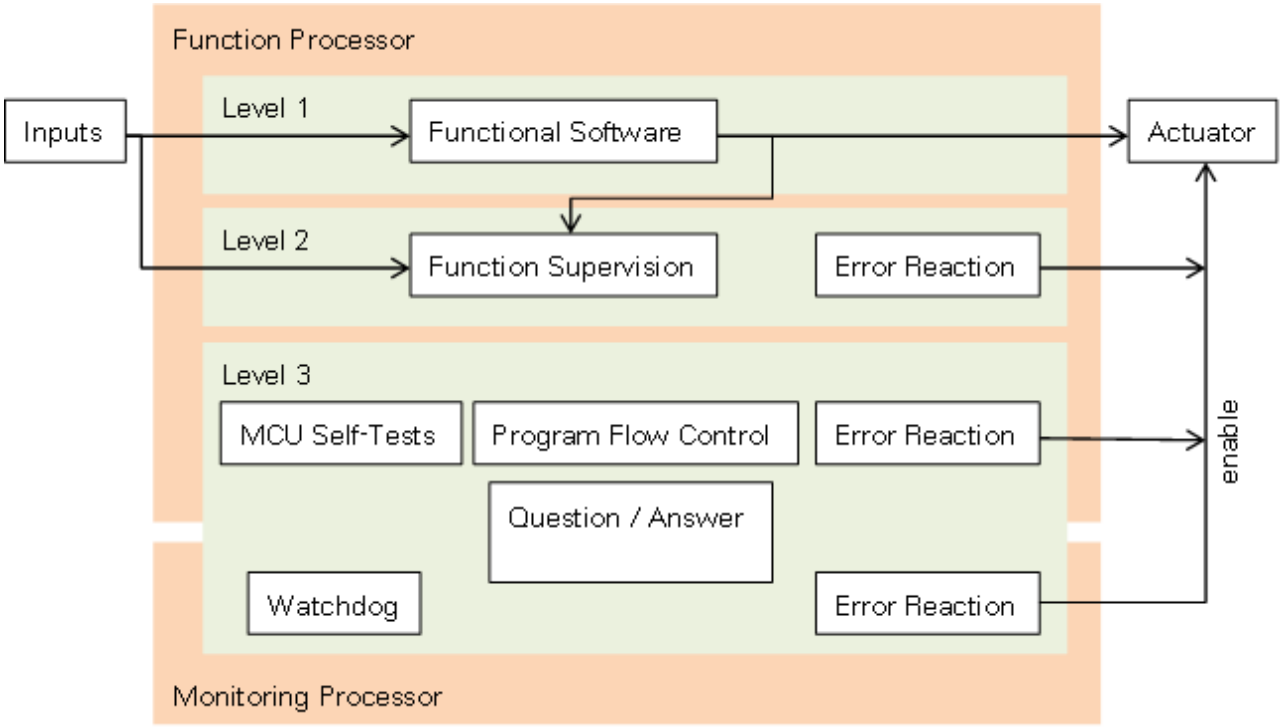


Abbildung 2: E-Gas-Konzept schematisch

Das E-Gas-Konzept führt zu einer hohen Komplexität der Software und stellt hohe Anforderungen an die Trennung zwischen den verschiedenen Softwareanteilen.

3.2 Simulations- und Testsystem für Fahrzeug- und Antriebssystem

Für das Erreichen der Arbeitsergebnisse in den Teilarbeitspaketen 5.1 und 5.3 und deren Verifikation war die Bereitstellung eines Simulations- und Testsystems erforderlich. Mit einem klaren Fokus auf E-Antriebskomponenten musste dieses ermöglichen, Simulationen der Komponenten und im erforderlichen Umfang auch ihrer Umgebung durchzuführen und

den Übergang auf den Fahrsimulator zu unterstützen. Insbesondere musste es mit dem Simulations- und Testsystem auch möglich sein, ein realitätsnahes Steuerungssystem mit den für die Sicherheit und Zuverlässigkeit des Systems notwendigen Basisumfängen zu integrieren.

Abbildung 3: links: Simulations- und Testsystem; rechts: Fahrsimulator

Das Programm CarMaker® der Firma IPG Automotive GmbH diente als Grundlage der entwickelten Simulationsumgebung. Es wurde speziell für die Analyse der Auswirkungen von möglichen Ausfallszenarien verwendet. Funktionale Modelle auf Komponentenebene wurden darüber hinaus mit dem Tool Simulink© der Firma Mathworks erstellt. Daher war es zusätzlich erforderlich, eine Kopplung wie in Abbildung 4 dargestellt zu realisieren.

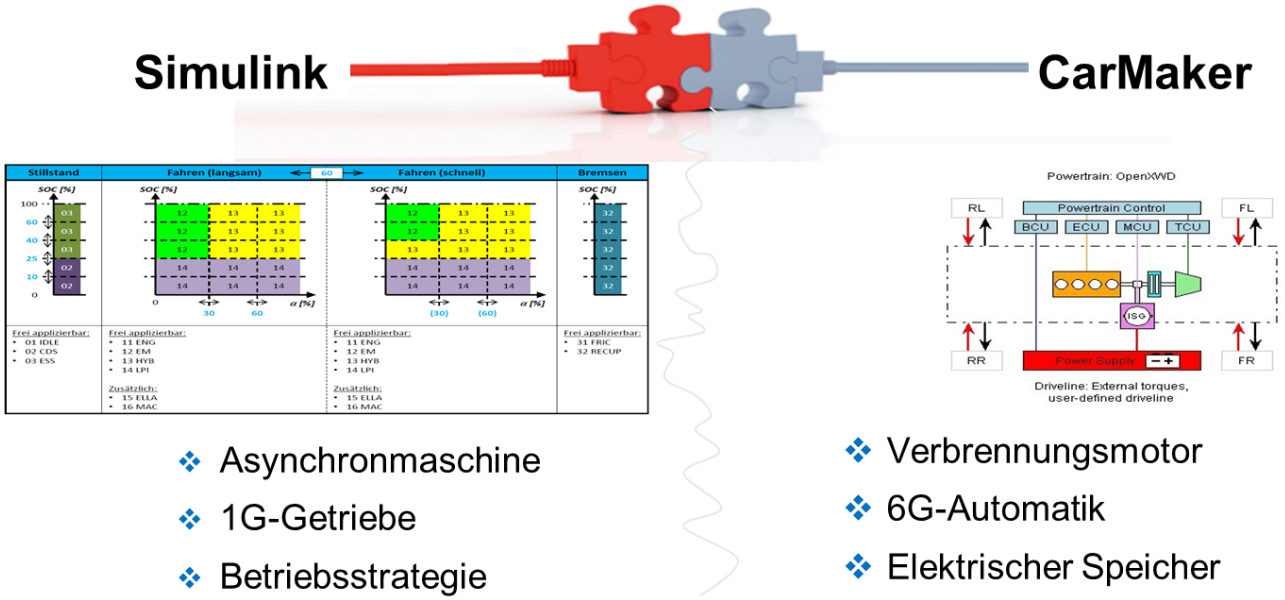


Abbildung 4: Kopplung CarMaker©-Simulink©

4 Vorbereitende Arbeiten

In diesem Kapitel werden vorbereitende Arbeiten beschrieben, die für die Durchführung der Schwerpunktarbeiten im Teilprojekt erforderlich waren.

4.1 Methode zur Modellierung von System-, HW- und SW-Architektur

Im AP 5.2 hat sich im Laufe der Projektbearbeitung herausgestellt, dass es zweckmäßig ist, auf Ergebnisse des Forschungsprojekts SAFE, an dem ZF ebenfalls aktiv beteiligt war, zurückzugreifen. Speziell wurden die im Projekt SAFE weiterentwickelte formale Beschreibung als Erweiterung der EAST-ADL-Beschreibungssprache und die dazu definierte Methode verwendet.

Die SAFE-Methode verbindet die funktional orientierte Systementwicklung mit den sicherheitsrelevanten Elementen einer Systementwicklung entsprechend ISO 262626, wie bereits im Kapitel 3.1 dargestellt. Die Methode bietet eine Möglichkeit zur modellgestützten Architekturentwicklung unter Berücksichtigung der Sicherheitsanforderungen, sie ist als Übersichtsbild in Abbildung 5 dargestellt.

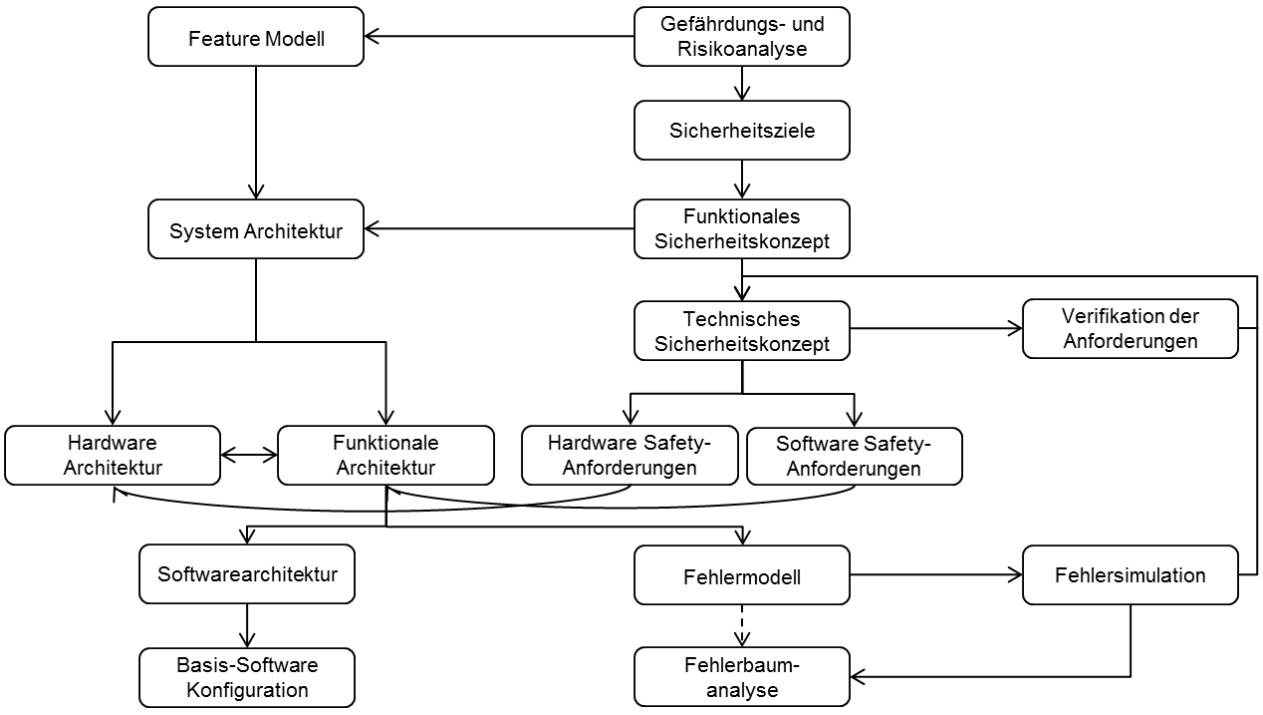


Abbildung 5: Modellgestützte Architekturentwicklung

Die Modellierung besteht aus fünf Clustern, die in Abbildung 6 dargestellt sind.

Speziell wird durch die beiden grünen und das gelbe Cluster, das für die modellbasierte Sicherheitsanalyse steht, sichtbar gemacht, um welche Elemente sich das Projekt SAFE bereits gekümmert hatte.

Insbesondere die Fehlersimulation, hier speziell eingesetzt zur Verifikation der Anforderungen des technischen Sicherheitskonzepts und als blaues Cluster dargestellt, wurde im Zuge der Erweiterungen des hier beschriebenen Teilprojekts ergänzt und angebunden. Die Cluster sollen zunächst grob beschrieben werden.

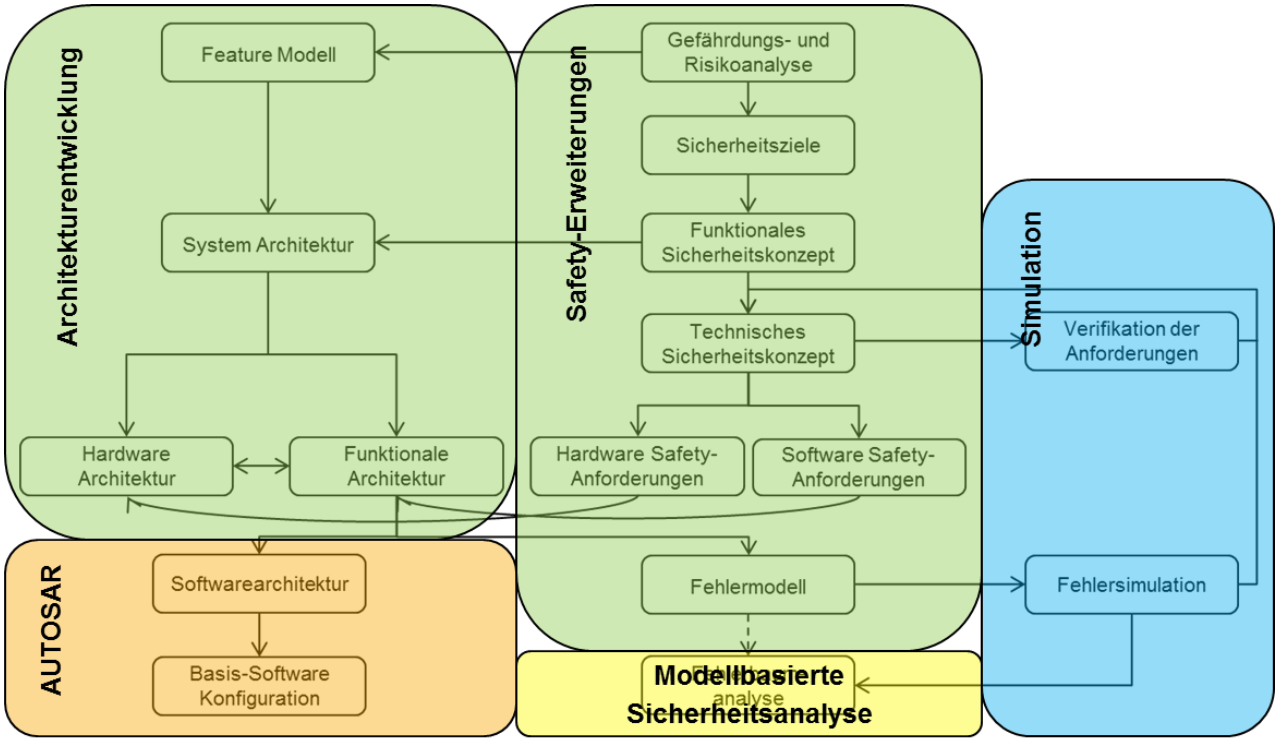


Abbildung 6: Gruppierung der modellgestützten Architekturentwicklung

- Teil eins (grünes Cluster links) kennzeichnet die Architekturentwicklung, ohne aber notwendige Prozesssicherheitsanforderungen zu berücksichtigen. Beinhaltet sind Erstellung des Featuremodells, der Systemarchitektur, sowie der funktionalen Architektur und der Hardwarearchitektur.

- Teil zwei (grünes Cluster rechts) beschreibt die Erweiterungen für funktionale Sicherheit. Hier geht es um die modellbasiert unterstützte Erstellung der Gefahren-und-Risiko-Analyse, den daraus abzuleitenden Sicherheitszielen, sowie das funktionale und technische Sicherheitskonzept. Insbesondere müssen Schnittstellen auf verschiedenen Ebenen zwischen der Architekturentwicklung und Safety-Erweiterungen beschrieben werden.

- Teil drei (oranges Cluster) beschreibt die Infrastruktur. Hier werden die Softwarearchitektur und die Basis-Software Konfiguration entsprechend AUTOSAR definiert. Mit der Anordnung der Softwarearchitektur kommt bereits zum Ausdruck, dass diese nahe an der Softwareumsetzung anzusiedeln ist und inzwischen von AUTOSAR standardisiert ist. Das gilt insbesondere für klassische Steuerungssysteme für PkW-Anwendungen zu denen entsprechende Antriebssteuerungen gehören.

- Teil vier (gelbes Cluster) beschreibt die modellbasierte Sicherheitsanalyse. Hier werden die Fehlerbäume aus den Fehlermodellen automatisch generiert.

- Teil fünf (blaues Cluster) kennzeichnet die Simulation, die aus Konsistenzgründen und zu Verifikationszwecken angebunden wird. Hier geht es um Fehlersimulation und um Verifikation von Anforderungen bereits in frühen Entwicklungsphasen. Dieser Teil wurde prototypisch im Projekt ZuSE unter Verwendung der in Kapitel 3.2 beschriebenen Simulationstools eingeführt.

Die Bilder in Anhang A: Details der angewendeten Methode beschreiben nacheinander die Inhalte und veranschaulichen die Prozessschritte der Cluster eins bis vier im Detail für das modellierte Beispiel eines PHV-Antriebssystems. In dieser Modellierung bestand die wesentliche Vorarbeit

Auf den Inhalt des Cluster fünf (blau) wird in diesem Bericht später eingegangen, da dieser Teil im Rahmen des Teilprojekts für die Simulationen prototypisch ergänzt wurde.

4.2 Analyse relevanter Funktionen und Architekturen

Als Grundlage für die Analyse von Systemeinflüssen auf HW- und SW-Architekturen wurden die Primäreffekte von wichtigen Funktionen im Hinblick auf die wesentlichen marktbestimmenden Kriterien für den Einsatz in Plug-In-Hybrid-/E-Fahrzeugen (PHV / EV) in einer Übersicht zusammengestellt. Tabelle 3 zeigt einen Auszug für PHV.

Tabelle 3: Einfluss von Funktionen auf Marktkriterien für PHV (Auszug)

Einflussmatrix (Auszug, Primäreffekte) Zu operativen Funktionen	Lok. Verbrauch Emissionen	Reichweite für Elektr. Fahren	Dynamik	Komfort
Rekuperieren	X			
Starten und Stoppen des VM	X			
Verschieben des Betriebspunkts des VM	X	X		
Segeln	X	X		
Boosten			X	
Modulieren der Übergänge				X
Elektrisches Anfahren	X	X		
Elektrisches Fahren		X		

Im Fokus stehen dabei Elektronik-Hardware und -Software unter Berücksichtigung der elektrischen Anbindung von Sensoren und Aktuatoren, eingeschränkt auf den betrachteten Umfang für die Steuerung von PHV und EV.

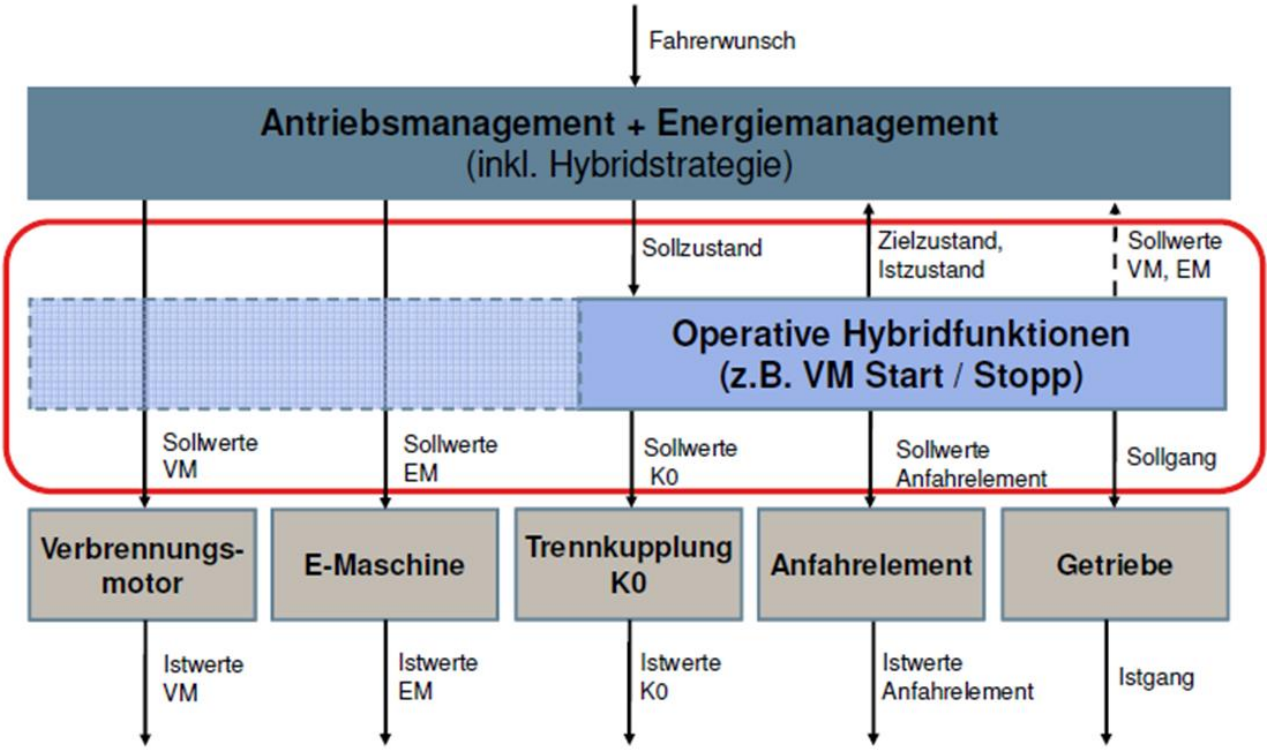


Abbildung 7: Übliche Komponenten eines PHV-Antriebssystems

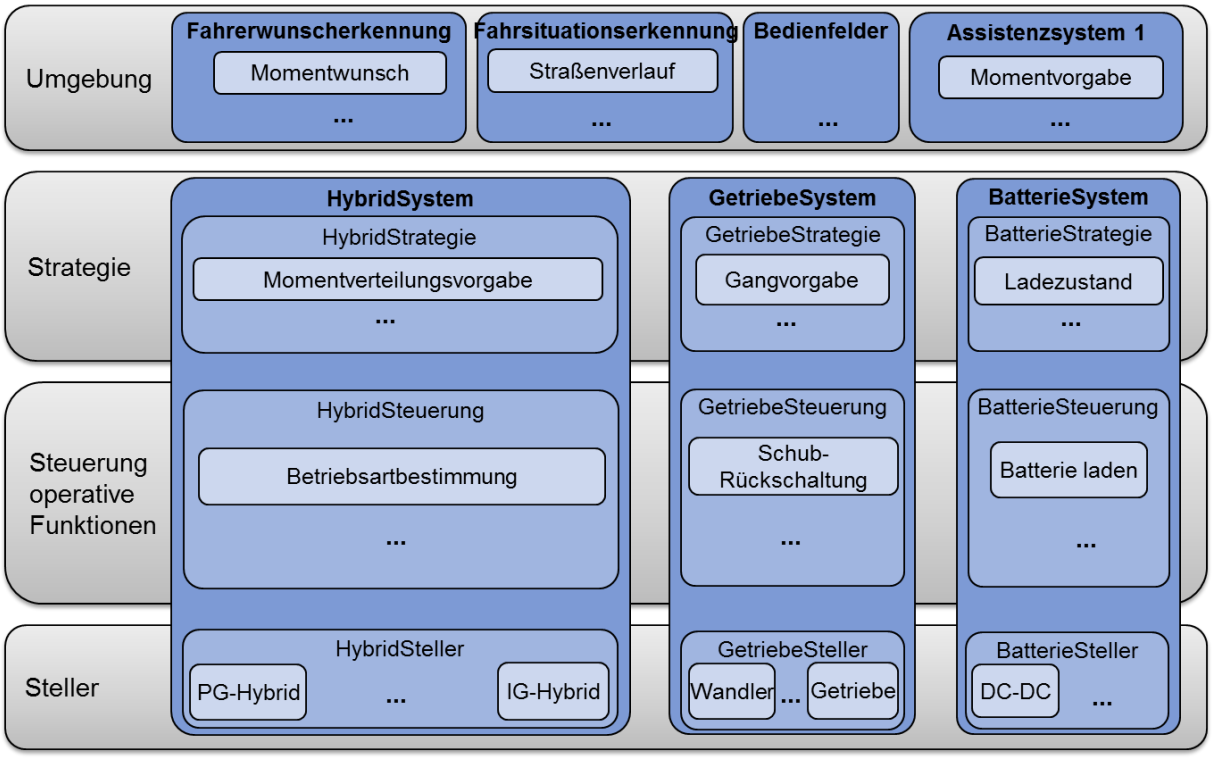


Abbildung 8: Beispielhafte Softwarearchitektur des PHV-Antriebssystems

Insbesondere PHV erfordern viel Koordination der beteiligten Antriebskomponenten im Hinblick auf alle operativen Funktionen. Gleichzeitig ergeben sich durch die Antriebsredundanz Möglichkeiten, die Verfügbarkeit ohne Sicherheitseinschränkungen auf Systemebene zu verbessern.

Verbesserungen sind allerdings nur möglich wenn das Antriebssystem und dessen Steuerung entsprechend ausgelegt sind. Abbildung 7 zeigt grundlegende Komponenten eines PHV unabhängig von der konkreten Funktion und Abbildung 8 beispielhaft die Funktionsarchitektur.

4.3 Softwareverteilung auf Kerne bei Mehrkernprozessoren

Für die grundsätzliche Anwendung von Mehrkernprozessoren können im Hinblick auf Sicherheit und Verfügbarkeit allgemeine Grundsätze, unabhängig von der konkreten Plattform, abgeleitet werden. Hierzu ein Beispiel, welches in den folgenden 3 Bildern illustriert wird. Es legt die Sicherheitsintegritätseinstufung zu Grunde und stellt die 3 wichtigsten Beispiele einander gegenüber, die in der Literatur immer wieder zu finden sind.

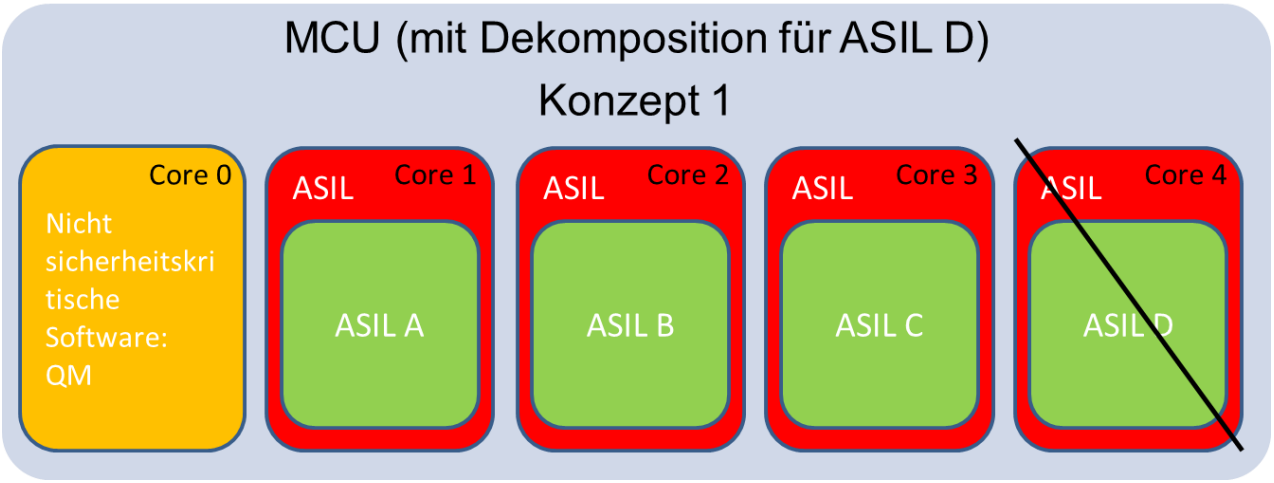


Abbildung 9: Trennung der Kerne nach QM und ASIL-Stufen

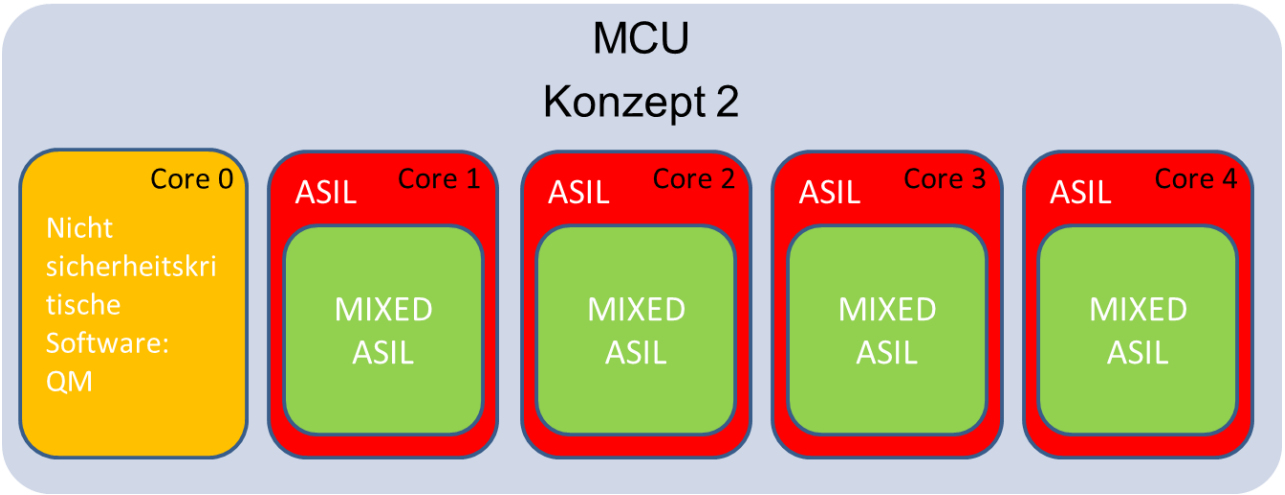


Abbildung 10: Ein Kern QM und alle anderen Kerne mit mixed ASIL

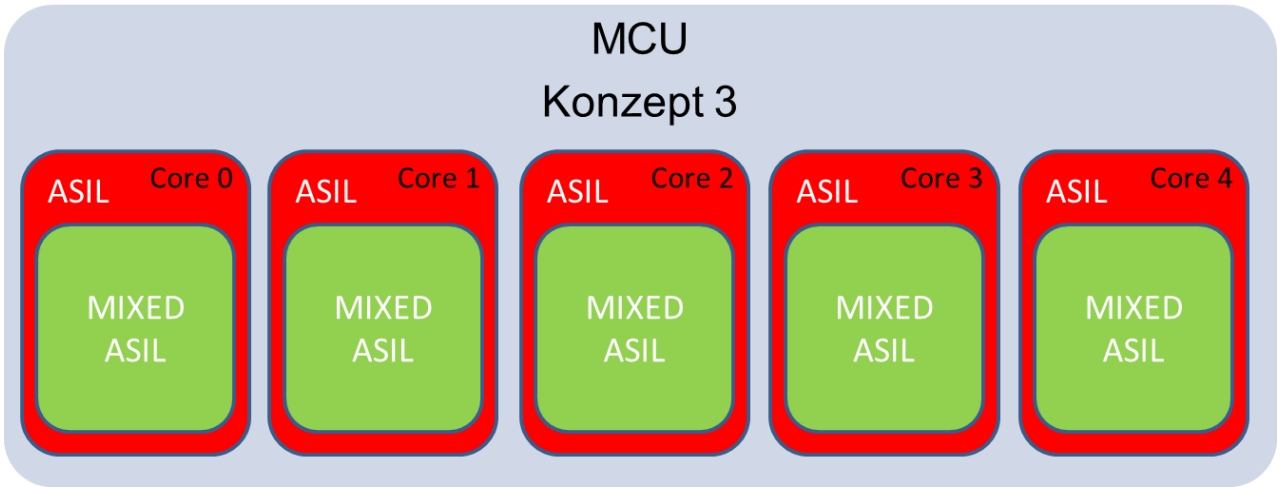


Abbildung 11: Alle Kerne mit mixed ASIL

In Tabelle 4 findet sich ein Vergleich dieser Konzepte.

Tabelle 4: SW-Verteilung abhängig von der Sicherheitsintegritätseinstufung

	Konzept 1 Nur für kleinere oder überschaubare Software verwendbar	**Konzept 2** Wenn QM-Software von der sicherheitsrelevanten gut zu trennen ist	**Konzept 3** Für komplexe Software verwendbar
Vorteile	• Keine Maßnahmen für Rückwirkungsfreiheit (Freedom from Interference) notwendig • Entwicklungsaufwand gering	• QM-Software unabhängig und dafür der Änderungsaufwand gering • Keine Sicherheitsmaßnahmen für QM-Teil erforderlich	• Synchronisierungsaufwand gering
Nachteile	• Synchronisierungsaufwand hoch, wenn die Funktionen voneinander nicht genug unabhängig sind	• Synchronisierungsaufwand für Trennung der sicherheits-kritischen Software von der nicht sicherheitskritischen	• Maßnahmen für Rückwirkungsfreiheit notwendig • Entwicklungsaufwand hoch

Für die hier betrachteten Antriebssysteme kommen wegen der Komplexität nur die Konzepte 2 oder 3 (jeweils mit mixed ASIL) in Frage. Zusätzlich lassen sich Level 1 und Level 2 der Software entsprechend dem E-Gas-Konzept in Verbindung mit zusätzlichen Anforderungen an Fehlerreaktionszeiten und wegen der erforderlichen Datenkonsistenz im Fehlerfall nur schwer trennen. Wenn man dann, was sehr oft der Fall ist, Level 1 nach QM und Level 2 nach dem jeweils mit der entsprechenden Absicherung geforderten ASIL umsetzt, dann resultiert daraus eine schwierige Trennung in QM und mixed ASIL, unter diesen Randbedingungen ist dann nur noch Konzept 3 sinnvoll möglich.

4.4 Beispielhafte E/E-Architektur

In Abbildung 12 ist eine E/E-Architektur für ein PHV dargestellt. Im Falle eines EV reduziert sich der Umfang, die separate Getriebesteuerung kann dann entfallen.

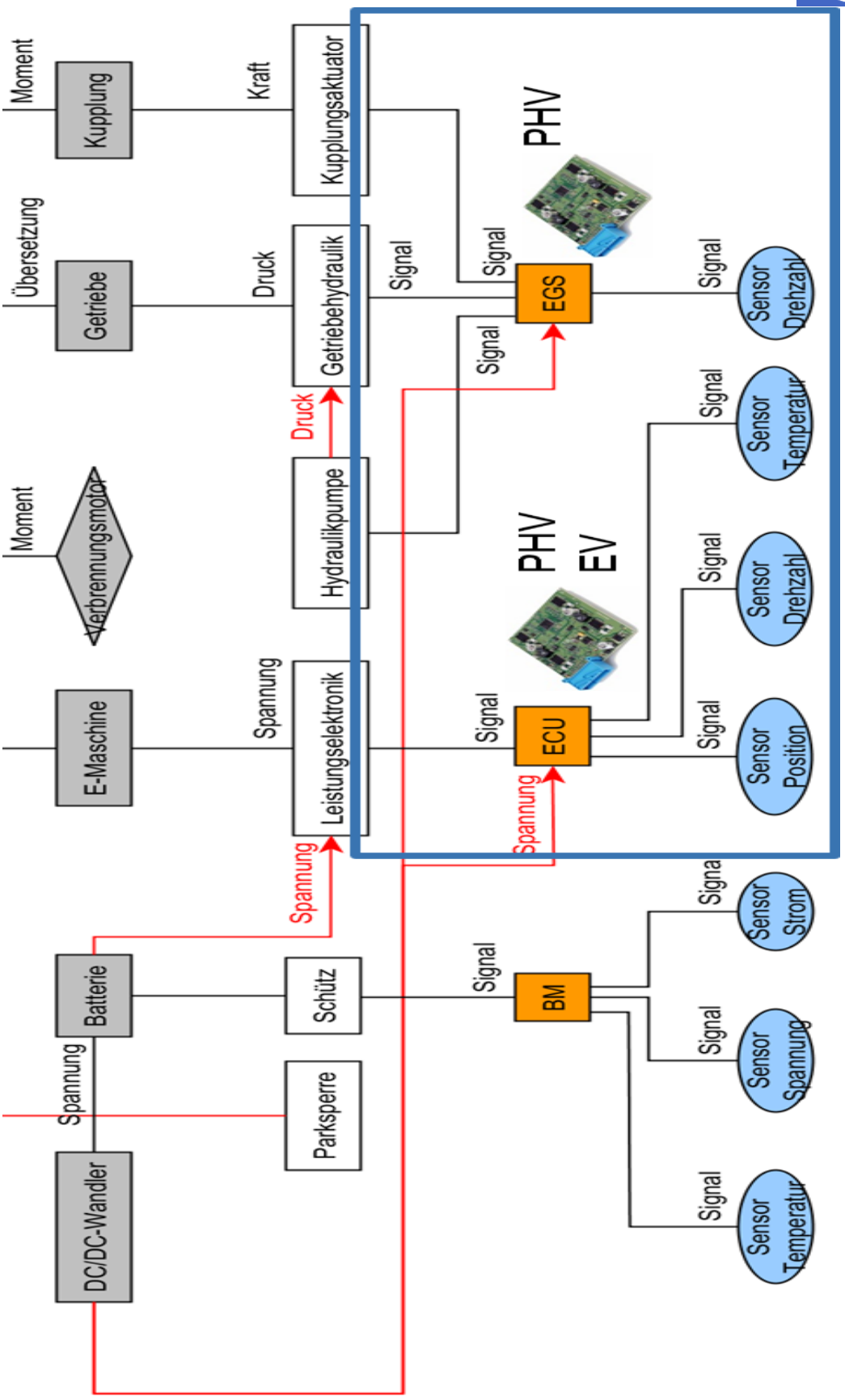


Abbildung 12: Beispielhafte E/E-Architektur des PHV-Antriebssystems

5 Spezifische Ausfallszenarien von Elektroantrieben

Im Zuge der Bearbeitung von Arbeitspaket 5.1 wurden Fehlermodelle für E-Antriebskomponenten identifiziert und die damit verbundenen Ausfallszenarien analysiert. Die Auswirkungen auf Zuverlässigkeit und Fehlertoleranz dienten als Grundlage für die Bewertung des elektrischen Antriebs. Diese Bewertung ermöglichte die Definition von geeigneten Methoden um Auslegung, Entwurf und Realisierung von Steuerungssystemen für die Antriebe zu unterstützen. Als Basis für die Modellierung und Identifikation wurden Fahrversuche mit aktuellen E Fahrzeugen durchgeführt.

5.1 Fehlermodelle und Simulationsumgebung

Die Arbeiten dieses Kapitel umfassen die Identifikation von Fehlermodellen für Komponenten des Elektroantriebs und den Aufbau einer Simulationsumgebung zur Fehleranalyse.

5.1.1 Identifikation der wesentlichen Fehlermodelle von Elektroantrieben

Die im Folgenden zusammengetragenen Komponentenfehler stammen aus der Literatur, Beobachtungen an realen Fahrzeugen und aus analytischen Betrachtungen. In der ISO 26262 wird in Part 5 eine Liste von Fehlern genannt, die es in Abhängigkeit des zu erreichenden Diagnosedeckungsgrads des betrachteten Systems zu bedenken gilt. Die Untersuchung der Funktionsausfälle der elektrischen Fahrzeugflotte des FKFS in Alltagsfahrten und Probandenstudien (Kapitel 5.3.3) ergab ebenfalls für Sicherheit und Zuverlässigkeit relevante Auffälligkeiten. Zudem wurde für die essentiellen Komponenten des elektrischen Antriebsstrangs eine analytische Betrachtung möglicher Fehlerfälle durchgeführt.

Folgende Fehlermodi und Fehlerursachen wurden dabei für die Sensoren herausgearbeitet. Sie werden am Beispiel eines Drehzahlsensors dargestellt:

- Ausfall
    - Leitungsunterbrechung der Spannungsversorgungs-, Erdungs- oder Steuergeräteanschlussleitung.
    - Wenn die Versorgungsspannung unter unteren Grenzwert von ca. 4 V fällt, stellt der Sensor konstante 7mA, was einer Drehzahl von 0,1/min entspricht. Bei weiterem Abfall der Versorgungsspannung fällt der Sensor komplett aus.
    - Bei unbekannter Drehrichtung bspw. beim Anlauf, senden manche Sensortypen kein Signal, wodurch die Bewegung des Systems von der Regelung nicht erfasst werden kann.

- Oszillation

- Elektromagnetisch induzierte Spannungen in den Leitungen.
- Ungleiche Zahnabstände aufgrund von Fertigungstoleranzen oder mechanische Einwirkung führen bei einer Zahn-zu-Zahn-Auswertung zu einer oszillierenden Drehzahl.
- Fehler des digitalen / analogen I/Os des Steuergeräts.

- Out of range
  - Bei Überdrehzahl verkleinert sich die Impulsweite mit steigender Drehzahl. Fällt die Impulsweite unter den unteren Grenzwert wird keine Drehzahl mehr erkannt.
- Vorzeichen
  - Falsche Interpretation der Impulsweiten in der Applikationssoftware.
  - Um 180° verkehrter Einbau des Drehzahlsensors.
- Stuck at
  - Fehler des digitalen / analogen I/Os des Steuergeräts.
- Betrag zu klein
  - Zahn zu Zahn Auswertung der Drehzahl führt bei nicht erkanntem Zahn zu einer kurzzeitigen Halbierung der Drehzahl. Eine Mittelwertbildung über mehrere Zähne reduziert die Auswirkungen nicht erkannter Zähne.
- Stuck in range
  - Falsche Grenzwerte in der Signalaufbereitung oder in einem Regler mit Anti-WindUp-Maßnahme.
- Drift
  - Fehler des digitalen / analogen I/Os des Steuergeräts.
- Time Delay
  - Der Drehzahlsensor weist zwar eine durch seinen Aufbau verursachte, konstante, niedrige Zeitverzögerung auf, eine Veränderung der Zeitverzögerung ist jedoch nicht vorstellbar.

5.1.2 Simulationsumgebung zur Fehleranalyse

Die gesammelten Fehlermodi sind in Fehlermodellen in Simulink® integriert. Das Fehlermodell der Sensoren ist in Abbildung 13 dargestellt. Es wird in die Subsysteme „Fehlerklassen“, „Signalbereich“ und „Signalzuordnung“ untergliedert. Im Subsystem „Signalbereich“ wird der Bereich aller Sensorsignale im laufenden Betrieb ermittelt und als Orientierungsgröße für Fehler wie bspw. „Stuck in range“ verwendet. Das Subsystem „Fehlerklassen“ wendet den vorgegebenen Fehlermodus auf das vorgegebene Sensorsignal an. Anschließend wird das fehlerbehaftete Sensorsignal im Subsystem „Signalzuordnung“ wieder in den Sensor-BUS eingegliedert. Über die Input-Ports des Fehlermodells lässt sich neben dem Signal und dem Fehlermodus auch der Zeitpunkt des Fehlerauftretens steuern.

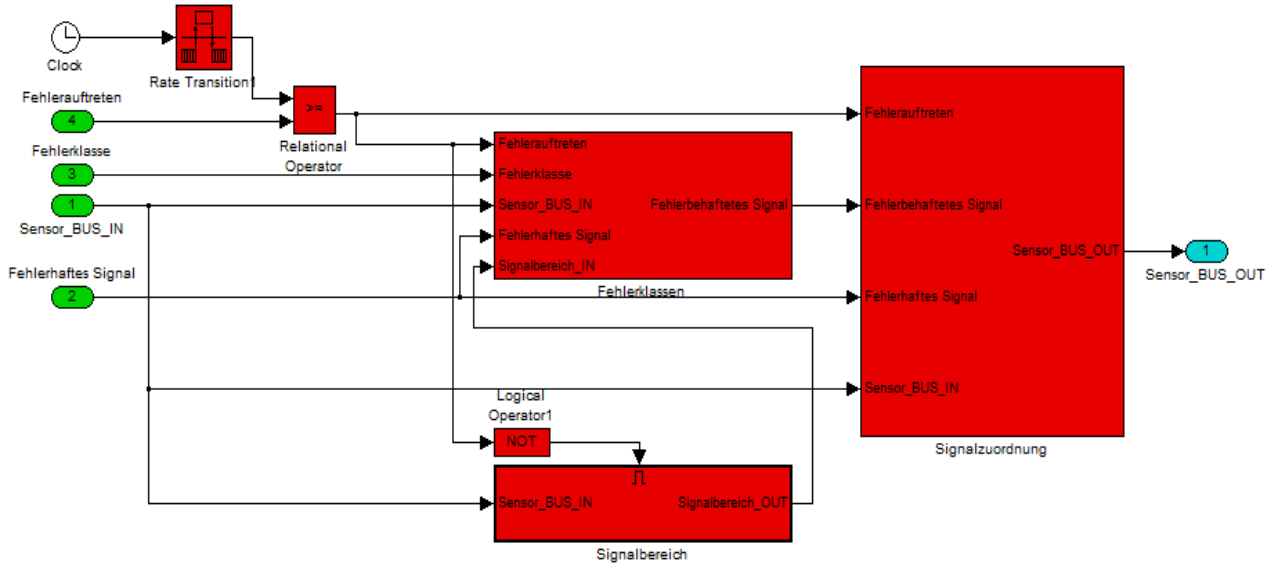


Abbildung 13: Simulink©-Modell zur Fehlermodellierung für Sensoren

Eine mathematische Beschreibung der im Fehlermodell der Sensoren integrierten Fehlermodi ist in Tabelle 5 angegeben.

Tabelle 5: Mathematische Beschreibung des Fehlermodells für Sensoren

Fehlermodus	**Math. Beschreibung für $t>t_{Fehler}$**
Vorzeichen	$y(t) = -u(t)$
Betrag zu groß	$y(t) = k * u(t) \quad \forall\ k > 1$
Betrag zu klein	$y(t) = k * u(t) \quad \forall\ 0 > k < 1$
Ausfall	$y(t) = 0$
Pos. Drift	$y(t) = u(t) + k * \int\int max(u(t<t_{Fehler}))dtdt$
Neg. Drift	$y(t) = u(t) - k * \int\int max(u(t<t_{Fehler}))dtdt$
Stuck in range	$y(t) = min(k * max(u(t<t_{Fehler})) , \ldots$ $max((k-1) * max(u(t<t_{Fehler})) , u(t)))$
Offset	$y(t) = u(t) + k * max(u(t<t_{Fehler}))$
Oscillation	$y(t) = u(t) * (1 + WhiteNoise)$
Stuck at	$y(t) = u(t_{Fehler})$
Time Delay	$y(t) = u(t - t_{Delay})$

5.2 Ausfallszenarien und Kennzahlen

Die Arbeiten dieses Kapitel umfassen das Erstellen von Ausfallszenarien für die jeweils ausgewählte Topologie des Antriebs und das Ermitteln von Kennzahlen der Fehler in Fehlersimulationen.

5.2.1 Gesamtfahrzeugsimulation

Das Programm CarMaker® der Firma IPG Automotive GmbH diente als Grundlage der entwickelten Simulationsumgebung zur Analyse der Auswirkungen von möglichen Ausfallszenarien. Zur Minimierung des Modellierungsausaufwandes wurden das von CarMaker® bereitgestellte Fahrzeug-, Umgebungs- und Fahrermodell verwendet. Zur Untersuchungen der fahrdynamischen Auswirkungen von Ausfallszenarien wurden zwei detaillierte Antriebsstrangmodelle ausgebaut. Das erste untersuchte Fahrzeug wurde definiert als Kleinwagen mit Heckantrieb. Die Topologie umfasst einen rein elektrischen Antriebsstrang mit zentraler E-Maschine in Form einer Asynchronmaschine, ein eingängiges Getriebe und ein Achsdifferential. Das zweite untersuchte Fahrzeug war ein Axle-Split-Hybrid. Die Topologie des ersten Fahrzeuges wurde an der Vorderachse um einen Verbrennungsmotor, ein 6-Gang Automatikgetriebe und ein Achsdifferential ergänzt. Die Komponenten des elektrischen Antriebs wurden in Simulink® modelliert und mit CarMaker® gekoppelt, während die Komponenten des konventionellen Antriebs von

CarMaker® verwendet und parametriert wurden. Für den Axle-Split-Hybrid wurde eine vom FKFS entwickelte, frei parametrierbare Betriebsstrategie verwendet.

Zur Verifizierung des in CarMaker® verwendeten Fahrermodells wurde eine Probandenstudie am Stuttgarter Fahrsimulator durchgeführt. Die Reaktion realer Fahrer auf verschiedene Ausfallszenarien wurde dabei untersucht und mittels der Daten wurden Parameter zur Anpassung des Fahrermodells ermittelt. In Abbildung 14 ist das Ergebnis der anschließenden Verifizierung des CarMaker® Fahrermodells (IPGDriver) mit der Probandenstudie für die relevanten Kenngrößen Lenkradwinkelgeschwindigkeit, Gierrate und Querbeschleunigung dargestellt.

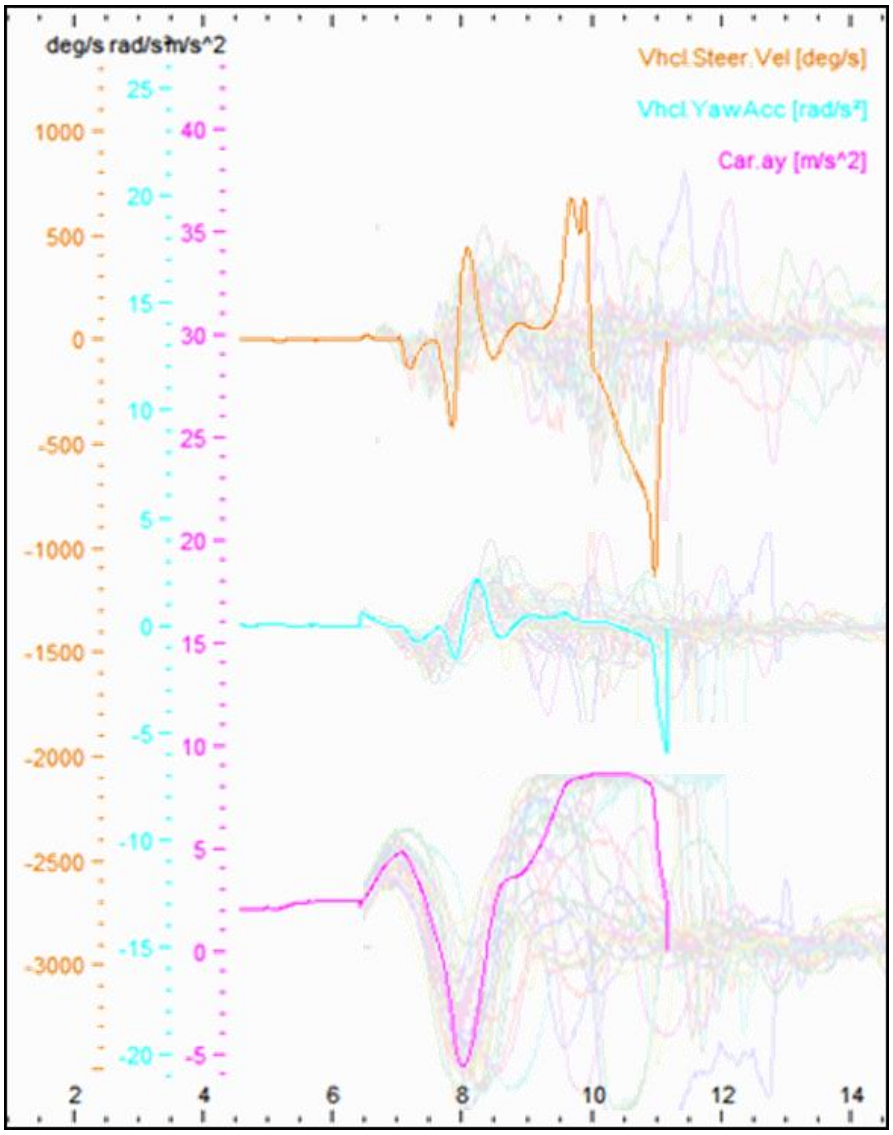


Abbildung 14: Vergleich IPGDriver mit der Probandenstudie

5.2.2 Bewertungsmetrik

Für die Bewertung der Fehlermodi werden, wie bereits während des Projektverlaufs erarbeitet, an die FMEA angelehnte Bewertungskriterien und Metriken verwendet, da diese zur Beurteilung der Fehlerkritikalität weit verbreitet sind und da zudem die Entwickler mit der Anwendung vertraut sind. Die Bewertungskriterien sind in Tabelle 6 angegeben. Sie werden, entgegen einer FMEA, aufgrund der Anwendung in einer frühen Entwicklungsphase, lediglich in 5 Stufen unterteilt.

Tabelle 6: Bewertungskriterien Fehlerkritikalität

Punkte	Bedeutung (B)	Punkte	Auftretens-wahrscheinlichkeit (A)	Punkte	Entdeckungs-wahrscheinlichkeit (E)
5	Nichterfüllung der Sicherheits- und / oder behördlichen Anforderungen	5	Rand- und Störbedingungen unbekannt	5	Keine oder eingeschränkte Erkennung des Fehlers
4	Ausfall oder Beeinträchtigung der Primärfunktion	4	Rand- und Störbedingungen mit keinem bekanntem System vergleichbar	4	Erkennung des Fehlers aber nicht der Fehlerursache
3	Ausfall oder Beeinträchtigung der Sekundärfunktion	3	Rand- und Störbedingungen mit bekanntem System vergleichbar	3	Erkennung der Fehlerursache erst nach genauer Untersuchung
2	Störung	2	Rand- und Störbedingungen mit bekanntem System identisch	2	Sofortige Erkennung des Fehlers, Erkennung der Fehlerursache durch zusätzliche Maßnahmen leicht möglich
1	Keine Auswirkung	1	Auftreten des Fehlers unwahrscheinlich	1	Sofortige Erkennung der Fehlerursache

Aus der Bewertung der drei Faktoren Bedeutung, Auftretenswahrscheinlichkeit und Entdeckungswahrscheinlichkeit ergibt sich als Produkt die Risikoprioritätszahl (RPZ). Ab einer festzulegenden RPZ (hier z. B. 25, beim 10-Punkte-System ist der Wert typischerweise 100) kann von einem dringenden Handlungsbedarf ausgegangen werden.

Um die Eignung der Metrik zu überprüfen, wurden Fehlfunktionen der Sensorsignale simuliert, insbesondere der für die E-Maschinen-Regelung relevanten Sensorsignale. Die Auswirkung (Bedeutung) der Fehlfunktionen ist in Tabelle 7 dargestellt.

Tabelle 7: Bedeutung von Fehlfunktionen der Sensorsignale

	Bzg	Bzk	Vz	Offset	Oscillation	Stuck in range	Stuck at	Time Delay	Ausfall
iEMotPhase (1/2/3)	Mittleres Drehmoment geringfügig zu niedrig. (B = 3)	Mittleres Drehmoment geringfügig zu hoch. (B = 3)	Extrem hohe Drehmoment-schwankungen (>> 160 Nm). Auswirkung abhängig von Fahrzeug-geschwindigkeit (B = 9)	Hochfrequente Schwingung des Drehmoments mit großer Amplitude (insbesondere bei Fahrzeug-geschwindigkeit 60 km/h) (B = 5)	Keine Auswirkung (B = 1)	Hochfrequente Schwingung des Drehmoments. Weist bei best. Fahrzeug-geschwindigkeiten (ca. 60 km/h) Drehmomentspitzen auf. (B = 5)	Hochfrequente Schwingung des Drehmoments. Weist bei best. Fahrzeug-geschwindigkeiten (ca. 60 km/h) Drehmomentspitzen auf. (B = 5)	Keine Auswirkung (B = 1)	Mittleres Drehmoment geringfügig zu hoch. Weist bei best. Fahrzeug-geschwindigkeiten (ca. 60 km/h) Drehmomentspitzen auf. (B = 5)
nEMot	Drehmoment fällt ab auf 0 Nm. (B = 8)	Drehmoment springt abhängig von der Ursprungsdrehzahl (Fahrzeugg-eschwindigkeit kleiner 60 km/h) auf hohen neg. Wert (bis zu -80 Nm) und klingt dann auf 0 Nm ab. (B = 9)	Drehmoment fällt ab auf 0 Nm. Hohes neg. Drehmoment (ca. -60 Nm) wenn Drehzahl unter ca. 4000 1/min fällt. (B = 9)	Drehmoment fällt ab auf 0 Nm. (B = 5)	Führt zu starker Schwingung des Drehmoments mit im Mittel geringer Abweichung. Ausnahme 120 km/h, zu Beginn negativer Mittelwert. (B = 9)	Drehmoment springt abhängig von der Ursprungsdrehzahl (Fahrzeug-geschwindigkeit kleiner 90 km/h) auf hohen neg. Wert (bis zu -80 Nm) und klingt dann auf 0 Nm ab. (B = 9)	Stationärer Betriebspunkt Keine Auswirkung (B = 1)	Keine Auswirkung (B = 1)	Drehmoment springt abhängig von der Ursprungsdrehzahl (Fahrzeug-geschwindigkeit kleiner 30 km/h) auf hohen neg. Wert (bis zu -80 Nm) und klingt dann auf 0 Nm ab. (B = 9)
uBatt	Keine Auswirkung (B = 1)	Keine Auswirkung (B = 1)	Keine Auswirkung (B = 1)	Keine Auswirkung (B = 1)	Hochfrequente Schwingung des Drehmoments mit großer Amplitude (insbesondere bei Fahrzeug-geschwindigkeit 60 km/h) (B = 5)	Keine Auswirkung (B = 1)	Stationärer Betriebspunkt Keine Auswirkung (B = 1)	Keine Auswirkung (B = 1)	Keine Auswirkung (B = 1)

Die Ausfallraten (Auftrittswahrscheinlichkeit) wurden mithilfe von Erfahrungen im Feld oder von Zuverlässigkeitsmodellen ermittelt. Verschiedene Quellen wie z.B. die Siemens Norm SN 29500 oder die MIL-HDBK-217F Notice 2 ermöglichen es mit Tabellen und den ungefähren Einsatzbedingungen eine Ausfallrate zu berechnen. Für beispielsweise den Fall eines Sensorausfalls in Folge schlechter Kontaktierung wird die Ausfallrate λ_P anhand der MIL-HDBK-217F Notice 2 berechnet.

$$\lambda_p = \lambda_b * \pi_T * \pi_K * \pi_Q * \pi_E \, Failures/10^6 Hours$$

λ_b ist die Basis-Fehlerrate, die von der Art der Steckverbindung abhängt. Unter der Annahme einer Rund-Steckverbindung ergibt sich λ_b=0,001. π_T ist der Temperaturfaktor, der von der Temperatur der Steckverbindung abhängt. Die Temperatur der Steckverbindung setzt sich zusammen aus der Umgebungstemperatur und der Erwärmung in Folge des Stromflusses. Unter Annahme einer Temperatur der Steckverbindung von 70°C ergibt sich π_T=2,0. π_K ist der Faktor für die Anzahl der Steck- und Lösezyklen der Steckverbindung pro 1000 Betriebsstunden. Da die Steckverbindung lediglich bei Reparaturen gelöst werden muss, wird die Anzahl der Steck- und Lösezyklen als 0 angenommen und damit ist π_K=1,0. Der Qualitätsfaktor ist π_Q=2,0 und der Umgebungsfaktor π_E ergibt sich aufgrund der mobilen Anwendung am Boden zu π_E=8,0.

$$\lambda_p = 0{,}001 * 2 * 1 * 2 * 8 \, Failures/10^6 Betriebsstunden$$

$$\lambda_p = 0{,}032 \, Failures/10^6 Betriebsstunden$$

Daraus folgt, dass die Auftrittswahrscheinlichkeit dieser Fehlfunktion < 1/10^6 Fehler / Betriebsstunde und somit sehr gering (A=1) ist.

Die Entdeckungswahrscheinlichkeit beim Ausfall des Phasenstromsensors ist üblicherweise sehr hoch (E=1). Neben den drei Phasenstromsensoren ist in der Leistungselektronik meist ein Gleichstromsensor verbaut, über den mithilfe von 2 Phasenströmen der dritte errechnet werden kann. Somit können die gemessenen Phasenströme plausibilisiert und Ausfälle eines Sensors sicher entdeckt werden. Das Signal des ausgefallenen Sensors kann aus den verbleibenden Sensoren errechnet werden wodurch sich keine nennenswerte Funktionsbeeinträchtigung ergibt. Die Entdeckungswahrscheinlichkeit des Ausfalls des Drehzahlsensors ist mäßig (E=5). Zwar kann ein Ausfall zeitnah über den Abfall der Spannung am I/O-Port des Steuergeräts erkannt werden, die Funktion kann jedoch nicht aufrechterhalten werden, da keine redundanten Systeme zur Verfügung stehen.

5.3 Grenzen der Fehlertoleranz

Die Arbeiten dieses Kapitel umfassen das Ermitteln von Grenzen der Fehlertoleranz für Ein- und Mehr-Motorantriebe und Komponenten-Lastkollektive aus Systemlastfällen, um darauf basierend die Zuverlässigkeit von Komponenten zu bewerten.

5.3.1 Fehlertoleranzniveaus

Für die E-Maschinensensoren Drehzahl, Temperatur und Phasenstrom wurden simulativ Fehlertoleranzniveaus bestimmt. Die Grundlage bildete das entwickelte Simulationsmodell des elektrischen Antriebsstranges in Simulink®. Dabei wurde die E-Maschine vollfaktoriell über den Drehzahl- und Drehmomentbereich angesteuert und der Sensorwert variiert. Über die Definition tolerierter Drehmomentabweichung mit dem absoluten Toleranzkriterium von 5 Nm und dem prozentualem von 10 % kann das Fehlertoleranzniveau für statische Betriebspunkte abgeleitet werden.

In Abbildung 15 ist beispielhaft die Fehlertoleranz für die sensorisch erfasste Drehzahl der E-Maschine dargestellt. Die Bezugsdrehzahl beträgt 5000 1/min, das Referenzdrehmoment wird in Schritten von 20 Nm von 0 auf 100 Nm erhöht. Bei einem Referenzdrehmoment von 0 Nm hat eine Verfälschung der Drehzahl keinen Einfluss auf das gestellte Drehmoment, obwohl die Verschiebung der Drehzahl-Drehmomenten-Kennlinie theoretisch zu einer Abweichung des Drehmoments führen müsste. Eine mögliche Ursache kann die Reduktion des Schlupf in Abhängigkeit des Referenzdrehmoments sein. Wird der Fluss bei einem Referenzdrehmoment von 0 Nm vollständig reduziert, hat auch die Verschiebung der Drehzahl-Drehmomenten-Kennlinie keinen Einfluss auf das gestellte Drehmoment. Mit zunehmendem Referenzdrehmoment wird der Bereich der tolerierten Drehzahlabweichung kleiner. Weiterhin steigt die Drehmomentabweichung außerhalb des tolerierten Bereichs stark an.

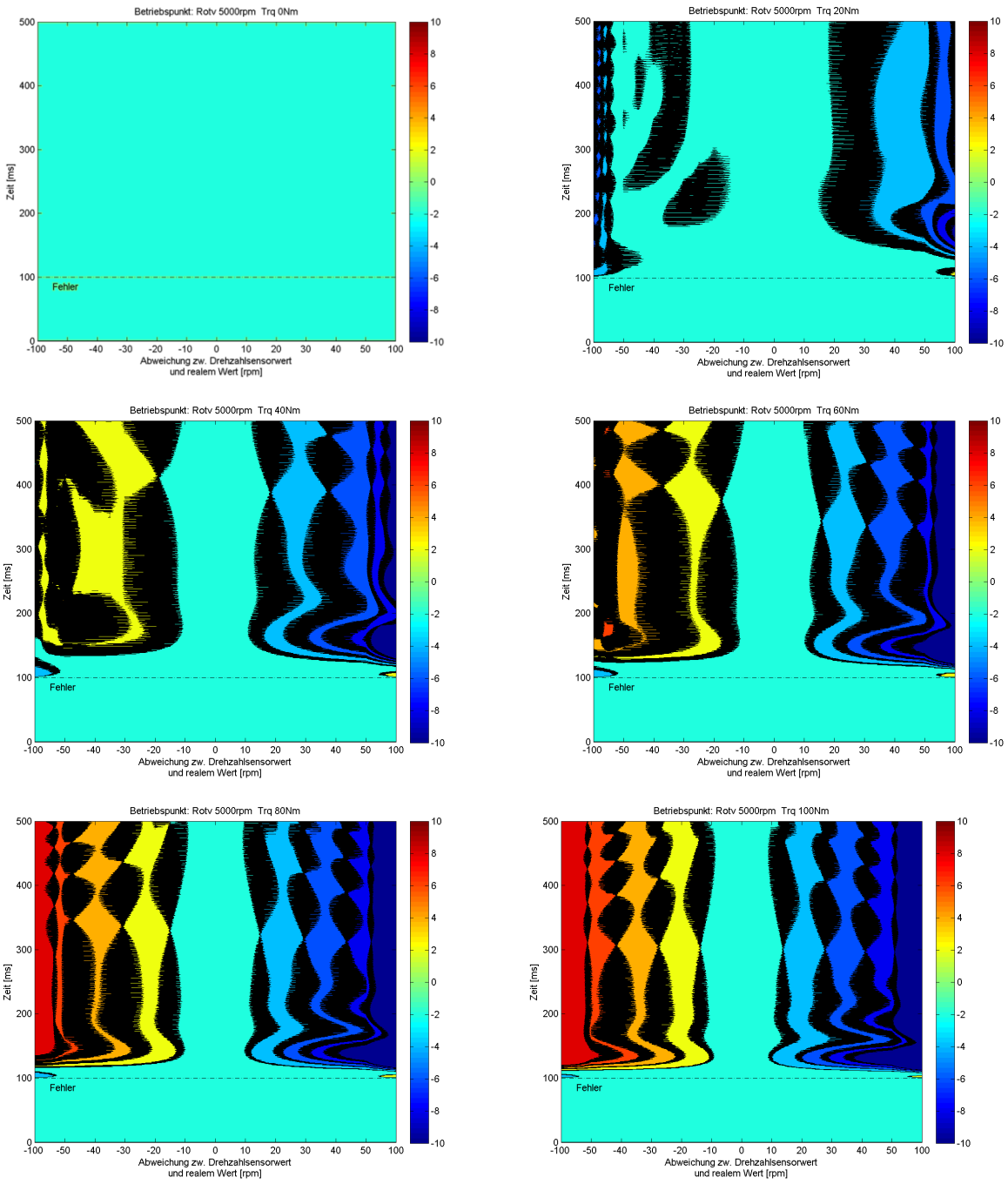


Abbildung 15: Drehmomentabweichung bei 5000 1/min und schrittweise steigendem Referenzdrehmoment (links oben: 0 Nm, rechts unten: 100 Nm)

5.3.2 Fehlerbehandlungsstrategien

Zur Behandlung von Fehlern der Sensoren Drehzahl, Temperatur und Phasenstrom wurden Fehlerbehandlungsstrategien entwickelt. Dabei wurde für das Elektrofahrzeug und für den Axle-Split-Hybrid jeweils eine Systemstruktur entworfen. Mittels dieser wurde anschließend ein Seriell-Parallel-Diagramm aufgestellt, um die identifizierten Kausalitäten im System darzustellen.

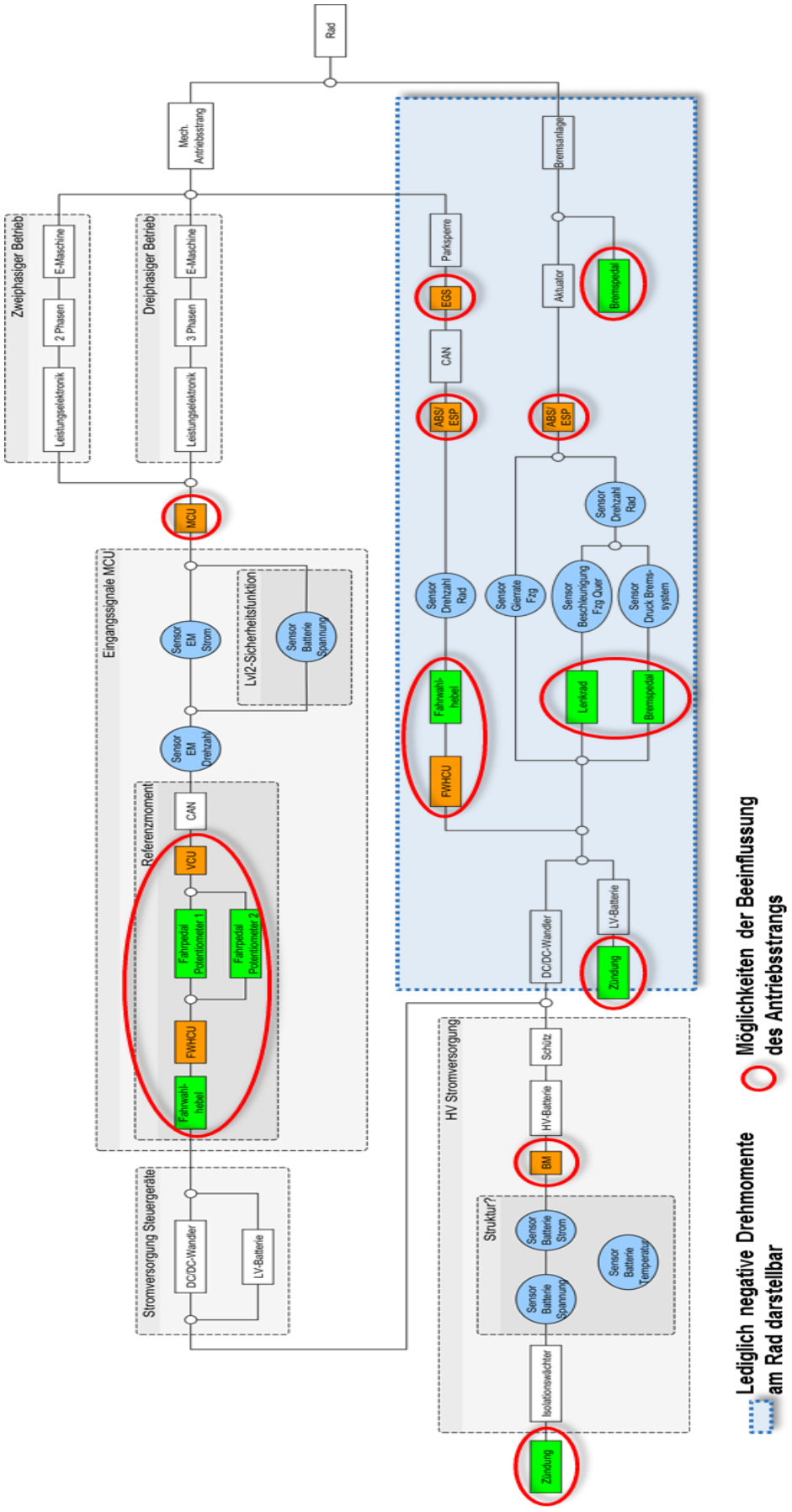


Abbildung 16: Seriell-Parallel-Diagramm des Elektrofahrzeuges

Abbildung 16 zeigt beispielhaft das Seriell-Parallel-Diagramm des drehmomentbildenden Pfads des Elektrofahrzeugs. Wie zu erkennen ist, handelt es sich bei dem Antriebsstrang eines Elektrofahrzeugs um einen weitestgehend seriellen Pfad. Parallel dazu verläuft der Pfad des Bremssystems (blau hinterlegt), das jedoch lediglich Drehmomenten entgegen der Drehrichtung des Rades erzeugen kann. Somit kann mit dem Bremssystem ein Ausfall der Rekuperation, sowie ein fehlerhaftes positives Drehmoment des Antriebsstrangs kompensiert werden. Tritt jedoch ein fehlerhaftes negatives Drehmoment des Antriebsstrangs auf, so ist die Einflussnahme auf die in der seriellen Kette des Antriebsstrangs liegenden Subsysteme begrenzt. Die beeinflussbaren Subsysteme sind rot umrandet. Es handelt sich dabei um die Steuergeräte und die Mensch-Maschine-Schnittstellen (Human-Machine-Interface, HMI).

Die Fehlerbehandlungsstrategie für den Drehzahlsensor ist in Abbildung 17 aufgezeigt. Fällt der Drehzahlsensor aus, so wird der Regelkreis aufgrund der nicht vorhandenen Trennkupplung um den Antriebsstrang erweitert. Als Sensoren werden die Raddrehzahlsensoren der angetriebenen Achse verwendet, deren Signal über das ABS/ESP-Steuergerät auf dem Fahrzeug-CAN bereitgestellt werden. Die Umsetzung wurde per Simulation aufgezeigt.

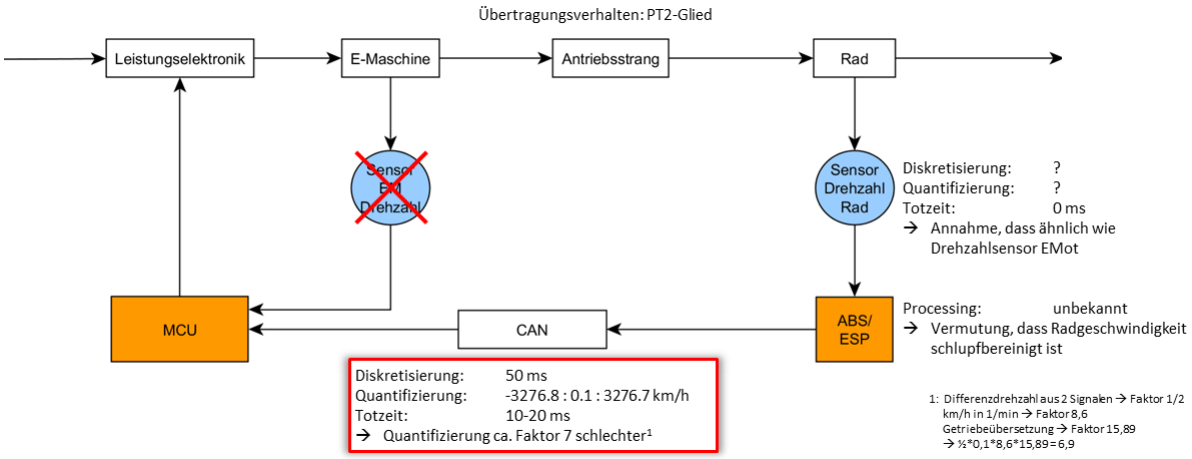


Abbildung 17: Regelkreis E-Maschine mit Raddrehzahlsensoren

Fällt der Temperatursensor aus, so wird der Regelkreis nach Abbildung 18 erweitert. Die Regelstrecke erweitert sich um den Kühlkreislauf der Asynchronmaschine. Als Sensor wird der Temperatursensor des Kühlwassers verwendet, der ebenfalls von der MCU eingelesen wird. Es wurde ein Temperaturmodell des Motors entwickelt, mit dem auf die Motortemperatur zurückgerechnet wird.

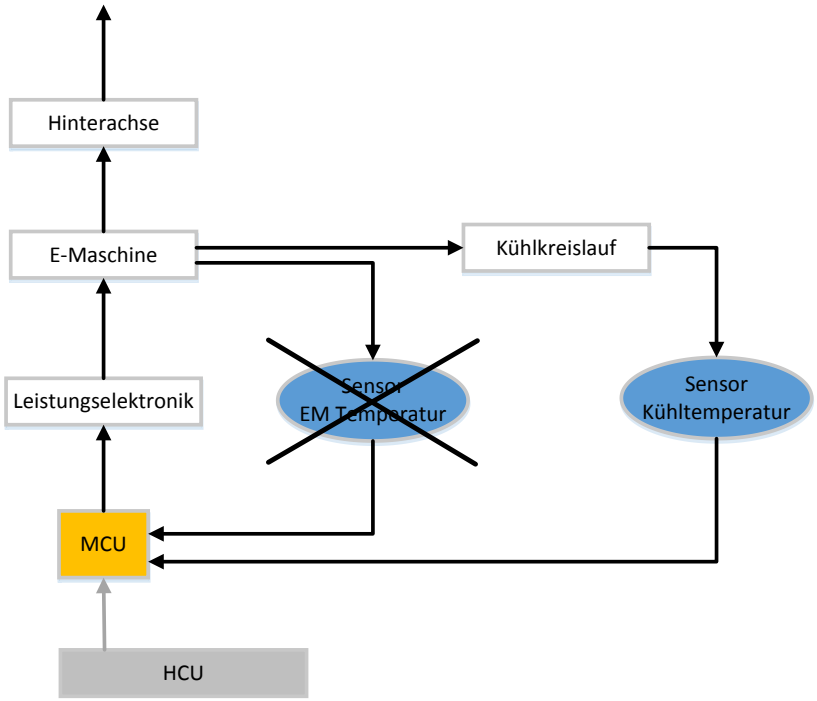


Abbildung 18: Regelkreis Asynchronmaschine mit Kühltemperatursensor

Da bei der Phasenstrommessung keine parallelen Pfade für die einzelnen Phasenstromsensoren existieren, wird beim Ausfall eines Sensors dessen Wert aus den beiden anderen Sensoren errechnet. Da es sich um ein symmetrisches Dreiphasensystem handelt, bei dem die Summe der drei Phasenströme Null ist, ist es ausreichend zwei Phasenströme zu messen.

5.3.3 Komponenten-Lastkollektive

Die Ableitung der Komponenten-Lastkollektive wurde für Elektro- und Hybridfahrzeuge durchgeführt. Dabei wurden für Elektrofahrzeuge Realfahrten als Grundlage verwendet, während für Hybridfahrzeuge ein Ansatz mit Simulation gewählt wurde. Dabei diente die in CarMaker® erstellte Gesamtfahrzeugsimulation als Basis.

Das FKFS unterhält eine Elektrofahrzeugflotte, in Abbildung 19 dargestellt. Die Realfahrten wurden mit diesen Fahrzeugen durchgeführt. Dabei wurden Messungen aus Alltagsfahrten der Mitarbeiter, sowie einer Probandenstudie im Sommer und einer im Winter verwendet. Aufgrund der zwei Probandenstudien konnten neben den Fahrereinflüssen auch klimatische Einflüsse berücksichtigt werden.

Abbildung 19: Überblick der eingesetzten Fahrzeuge

Am FKFS werden seit vielen Jahren Fahrversuchs-Studien mit Normalfahren auf öffentlichen Straßen durchgeführt. Ziel ist es, den Endkundenbetrieb nachzubilden. Entscheidend ist dabei, dass solche Studien eine auf wissenschaftlicher Grundlage nachweisbare Aussagekraft (Signifikanz) aufweisen.

Dies wird erreicht, indem die statistischen Parameter der befahrenen Strecken und des Fahrerkollektivs sowie die Umgebungsbedingungen möglichst exakt das Autofahren in der betrachteten Region repräsentieren.

Der Stuttgart-Rundkurs in Abbildung 20 wurde vor einigen Jahren in Zusammenarbeit mit einem Industriepartner definiert und repräsentiert aufgrund seiner statistischen Parameter das Fahren in Deutschland mit sehr guter Näherung. Er diente als Grundlage sowohl für die Probandenstudien, als auch für die Ableitung der Lastkollektive aus der Simulation.

Die in den Realfahrten aufgezeichneten Antriebsstrang- und Fahrzustandsgrößen dienten als Grundlage um das Verhalten der Fahrzeuge im Normalbetrieb und bei spezifischen Fahrmanövern zu bewerten.

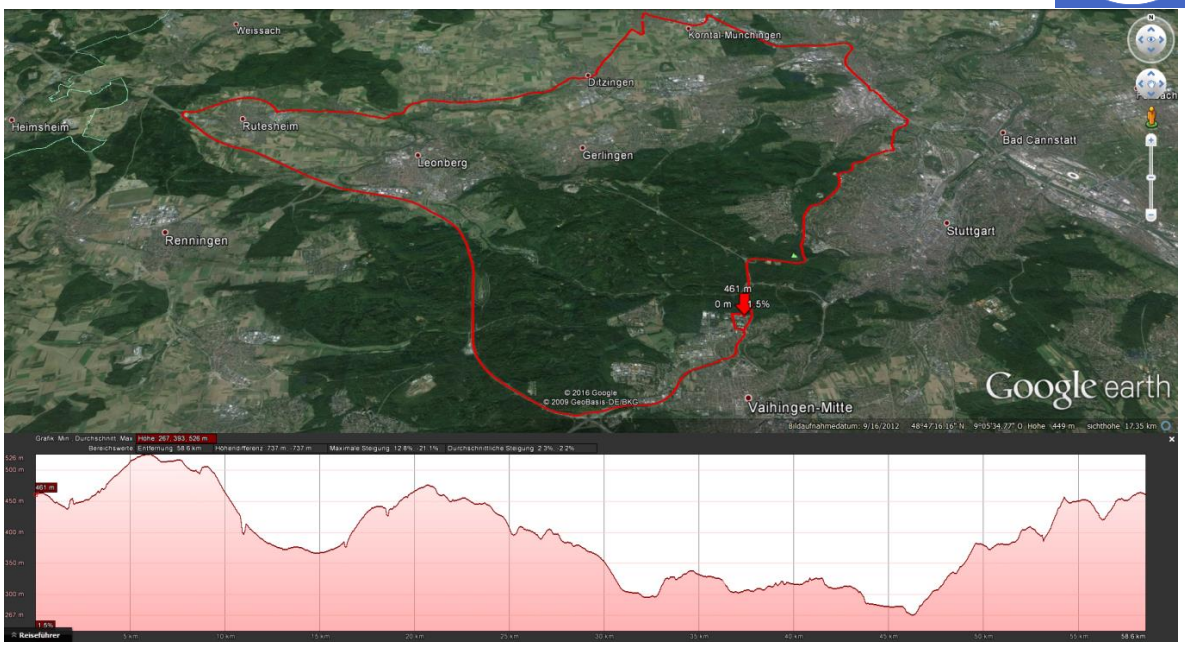


Abbildung 20: FKFS Stuttgart-Rundkurs

Neben der Untersuchung der Antriebsstränge in spezifischen, für Fehlerfälle relevanten, Manövern wurden statistische Auswertungen zu Belastungen von Antriebstrangkomponenten im Alltagsbetrieb und im Rahmen von Probandenstudien durchgeführt. Der Alltagsbetrieb umfasste dabei Fahrten mit den E-Fahrzeugen die unterschiedliche Fahrer im täglichen Betrieb zurücklegten. Dabei wurden größtenteils unterschiedliche Strecken, Streckenlängen und Fahrtdauern gefahren. Demgegenüber stehen die Probandenstudien. Bei diesen wurden die Fahrzeuge auf einem definierten Rundkurs von Probanden bewegt. Dabei wurden die Randbedingungen wie Streckenwahl, Fahrtdauer und Streckenlänge durch die Verwendung des Rundkurses konstant gehalten. Die Unterschiede zwischen den Fahrten beschränkten sich damit auf den Fahrweiseeinfluss der Probanden. Im Rahmen der Probandenstudien wurden ca. 20.000 km Wegstrecke zurückgelegt. Dabei fuhren 42 Probanden jeweils alle an der Studie beteiligten Fahrzeuge. Da sowohl bei der Strecken- als auch die Probandenauswahl eine statistische Durchschnittsverteilung zugrunde gelegt wurde, können anhand der Messdaten dieser Fahrten Aussagen zum durchschnittlichen Nutzungsverhalten von E-Fahrzeugen gemacht werden. Diese Auswertungen geben einen Überblick welche Lastbereiche und Fahrzustände mit welcher Häufigkeit im Alltagsbetrieb bzw. im mittleren Betrieb auftreten. Auf Basis dieser Erkenntnisse wurden die Simulationsmodellierungen auf die relevanten Bereiche eingegrenzt, was im Allgemeinen zu einer höheren Modellgüte führte. Zudem kann die Auftrittshäufigkeit von Fehlerzuständen im Normalbetrieb bewertet werden. Die Ergebnisse dienten als Grundlage für Kapitel 5.1.

6 Robuste und fehlertolerante Steuerungsarchitektur

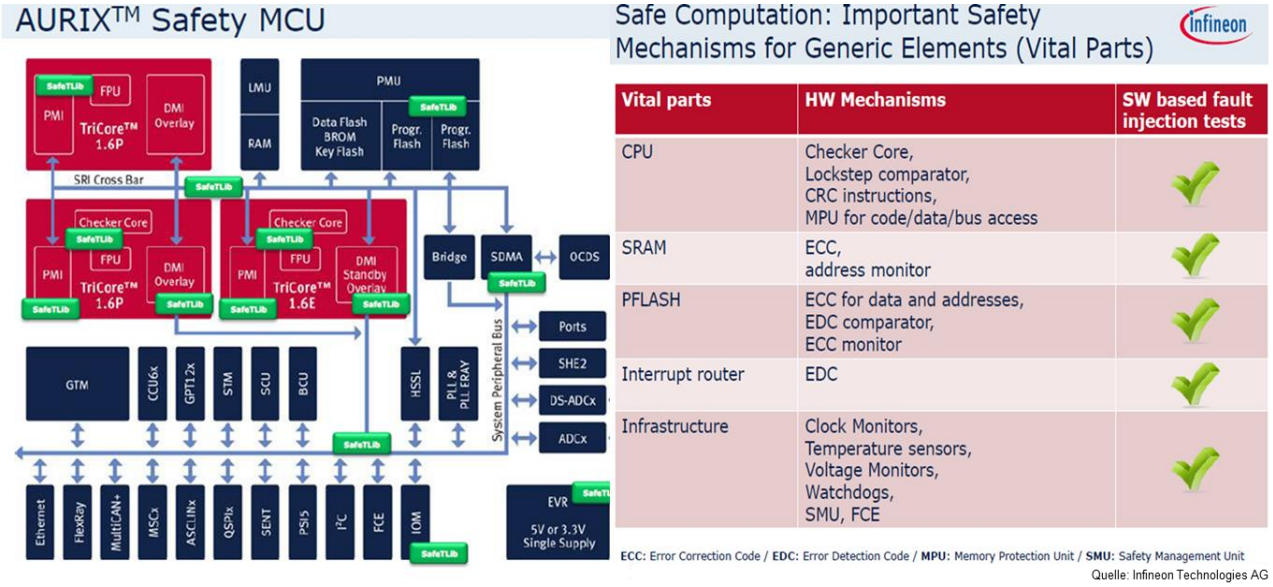
Im Zuge der Bearbeitung von Arbeitspaket 5.2 wurden für EV- und PHV-Antriebssteuerungen im Hinblick auf Fehlertoleranz Hardware- und Software-Architekturen definiert, analysiert und verglichen. Zusätzlich wurden fehlertolerante Betriebsstrategien, die sich aus den diesbezüglichen Systembetrachtungen im Arbeitspaket 5.1 ableiten ließen, simuliert.

6.1 Fehlertolerante Hardwarearchitektur

Die Arbeiten dieses Kapitel umfassen Definition, Analyse und Vergleich von fehlertoleranten Hardwarearchitekturen für Steuergeräte im Antriebsstrang mit wechselseitiger Überwachung.

6.1.1 Aktuelle Prozessoren und Sicherheitsmechanismen

Viele Maßnahmen, die heute zur Absicherung eines Prozessors einschließlich seiner on-Chip-Ressourcen (RAM, Flash, Interrupt, Register, …) erforderlich sind, werden für moderne Controller, zu denen insbesondere die aktuell in die Entwicklung vordringenden Mehrkernprozessoren gehören, vom Halbleiter-Lieferant mit angeboten und so weit wie möglich durch Mechanismen im Prozessor oder mittels geeigneter Peripherie umgesetzt und auch mit Software unterstützt, die in Form von Lizenzrechten erworben werden kann.



Vital parts	HW Mechanisms	SW based fault injection tests
CPU	Checker Core, Lockstep comparator, CRC instructions, MPU for code/data/bus access	✓
SRAM	ECC, address monitor	✓
PFLASH	ECC for data and addresses, EDC comparator, ECC monitor	✓
Interrupt router	EDC	✓
Infrastructure	Clock Monitors, Temperature sensors, Voltage Monitors, Watchdogs, SMU, FCE	✓

Abbildung 21: Fehlerkennung/–korrektur auf µC-Ebene am Beispiel AURIX (Infineon)

Die neuen Prozessorfamilien realisieren weiter verbesserte Maßnahmen zur Fehlererkennung, wie beispielsweise Tests für alle wichtigen Ressourcen wie Timer, Speicher usw., aber auch zur Fehlerkorrektur wie z. B. ECC mit bestimmten Einschränkungen bezüglich der Anzahl verfälschter Bits. Bezogen auf die AURIX-

Prozessorfamilie sind die Prozessorarchitektur und wichtige Hardware- Mechanismen in Abbildung 21 dargestellt. Insbesondere die Fehlerkorrektur-Maßnahmen, angewendet auf verfälschte Dateninhalte von ansonsten funktionsfähigen Speicherzellen, erhöhen die Fehlertoleranz und führen daher zur Verbesserung der Zuverlässigkeit des Prozessors.

Die Prozessor-Architekturen bieten über Speicherschutzmechanismen und über Schutzmaßnahmen für den zeitlichen Ablauf oder die interne Kommunikation, weitere neue Möglichkeiten. Fehlerhafte Zugriffe der Software können erkannt werden, noch bevor sie durch eine unbeabsichtigte Beeinflussung von sicherheitsrelevanten Funktionen zur Verletzung eines Sicherheitsziels führen können. Doch ist es sehr schwierig, diese Ansätze systematisch zur Verbesserung der Verfügbarkeit auf Systemebene zu nutzen. Häufig ersetzen sie nur die Erkennung von Fehlern anhand der Fehlerfolgen durch eine präventive Erkennung. Der Fehler an sich führt aber in Ermangelung von anderen Konzepten dann doch weiterhin zur Systemabschaltung als Fehlerreaktion. Die Speicherschutzmechanismen werden auch für andere Zwecke verwendet, z. B. um die Inhalte von Registern des Prozessors aus Gründen der Datenintegrität vor Verfälschung zu schützen.

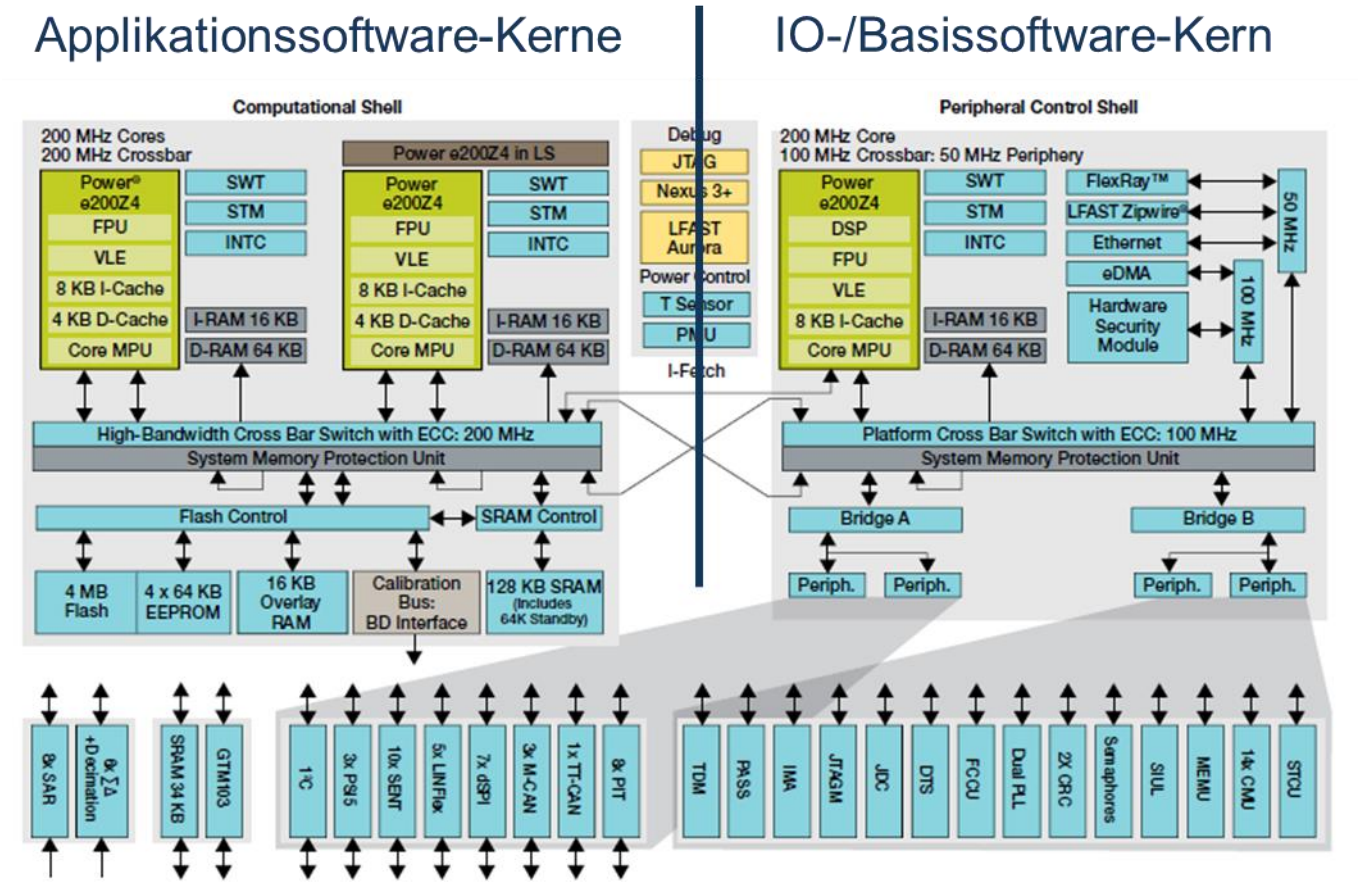


Abbildung 22: Architektur am Beispiel Qorivva MPC5746M (Freescale)

Während die Hardware-Architektur des AURIX von Infineon aus Abbildung 21 dem Anwender eine hohe Flexibilität in der Zuordnung von Controller-Ressourcen zu einzelnen

Prozessorkernen einräumt, legt die Architektur des Qorivva MPC5746M von Freescale insbesondere für die Zugriffe auf IO-/Kommunikation-Ressourcen leicht erkennbar eine Zuordnung zu einem „IO-/Kommunikation-Kern“ nahe, der vermutlich auch die mit der IO-Verarbeitung zusammenhängende Interrupt-Last möglichst komplett übernehmen und so die für Applikationen vorgesehenen Kerne davon befreien soll.

Aufgrund von HW-Limitierungen in der Prozessor-Entwicklung und wegen der für SW benötigten Ressourcen kommen verstärkt Mehrkernprozessoren zum Einsatz. Im Rahmen des Teilprojekts haben wir uns speziell auf die AURIX-Familie der Firma Infineon konzentriert, da entsprechende Prozessoren verfügbar waren. Es sei jedoch darauf hingewiesen, dass es alternative Mehrkernprozessoren von anderen Herstellern wie beispielsweise Freescale oder Renesas gibt, die ebenfalls bestimmte Derivate für den Einsatz in Antriebssteuerungen im Automobilbereich anbieten. Die Einarbeitung in eine neue Prozessorarchitektur erfordert jedoch hohe Aufwände. Nur bei der Infineon-Lösung konnten im Zeitrahmen des Projekts die erforderlichen Synergien für die Arbeit im Projekt ZuSE hergestellt werden.

Bereits bekannte Prozessor-Sicherheitsmechanismen wurden nicht im Detail betrachtet und ausgeführt, da Ergebnisse aus Projekten bekannt sind und entsprechend mit berücksichtigt werden können. Insbesondere aber die kritischen Ressourcen und die neuen Schutzmechanismen (z. B. Memory Protection/Management oder Safety Management Unit) und die Randbedingungen für Software-Verteilung auf Kerne wurden am Beispiel AURIX intensiver untersucht, da sie in der Projektstartphase noch nicht im breiten Einsatz waren.

Kritische Controller-Ressourcen sind in Bezug auf Sicherheit und Verfügbarkeit von besonderem Interesse, daher wurden sie im Zuge der Betrachtungen für die AURIX-Plattform näher betrachtet. Einige Erkenntnisse sind in Abbildung 23 zusammengestellt.

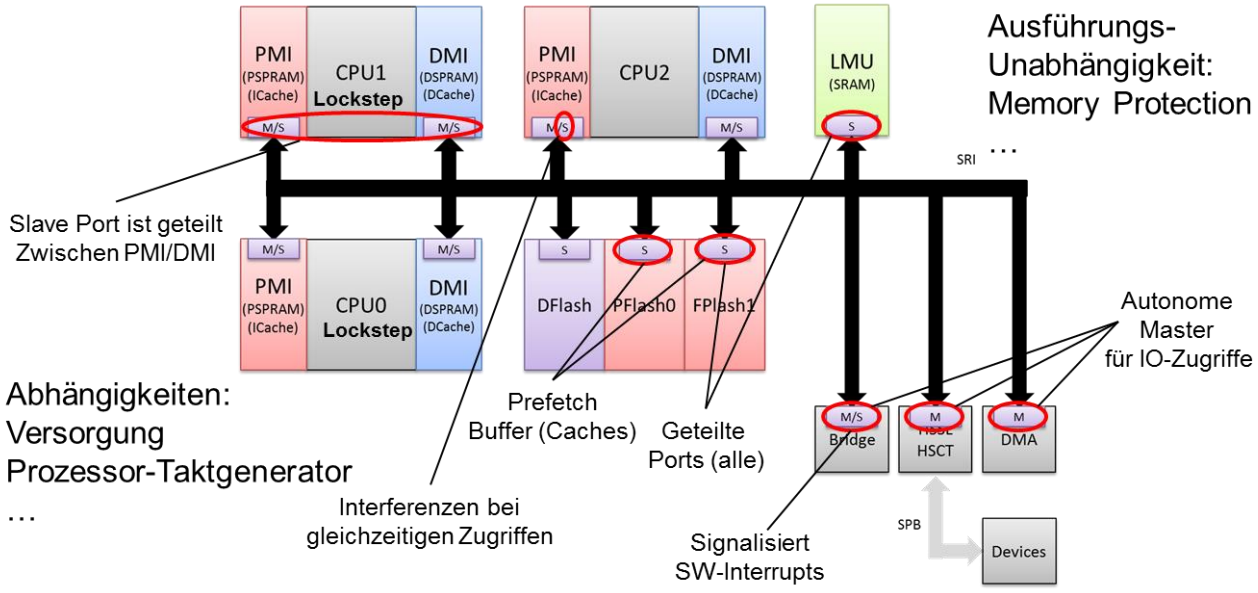


Abbildung 23 Beispiele für kritische Ressourcen der AURIX-Plattform

In Abbildung 24, Abbildung 25 und Abbildung 26 sind Analyseergebnisse zu AURIX-internen Speicherzugriffseinschränkungen, sowie zu den MPU-Mechanismen dargestellt.

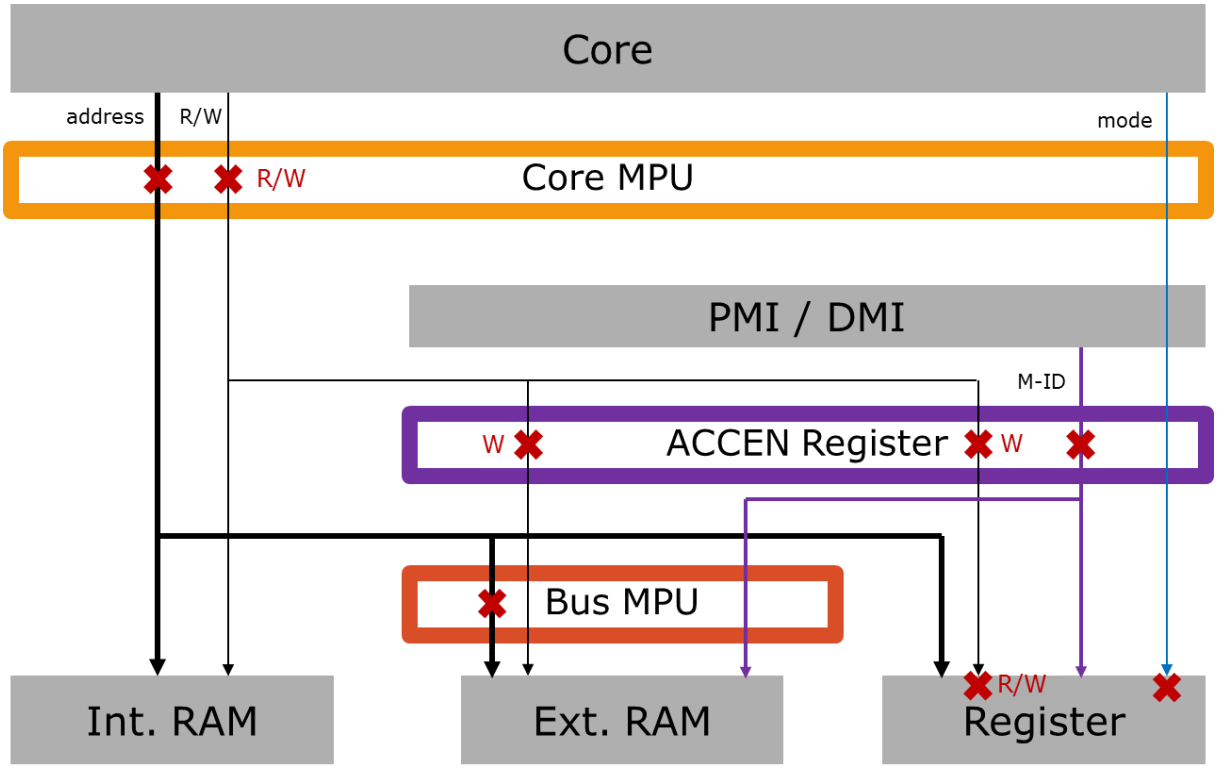


Abbildung 24: Speicher-Zugriffseinschränkungen der AURIX-Plattform mit Core-MPU

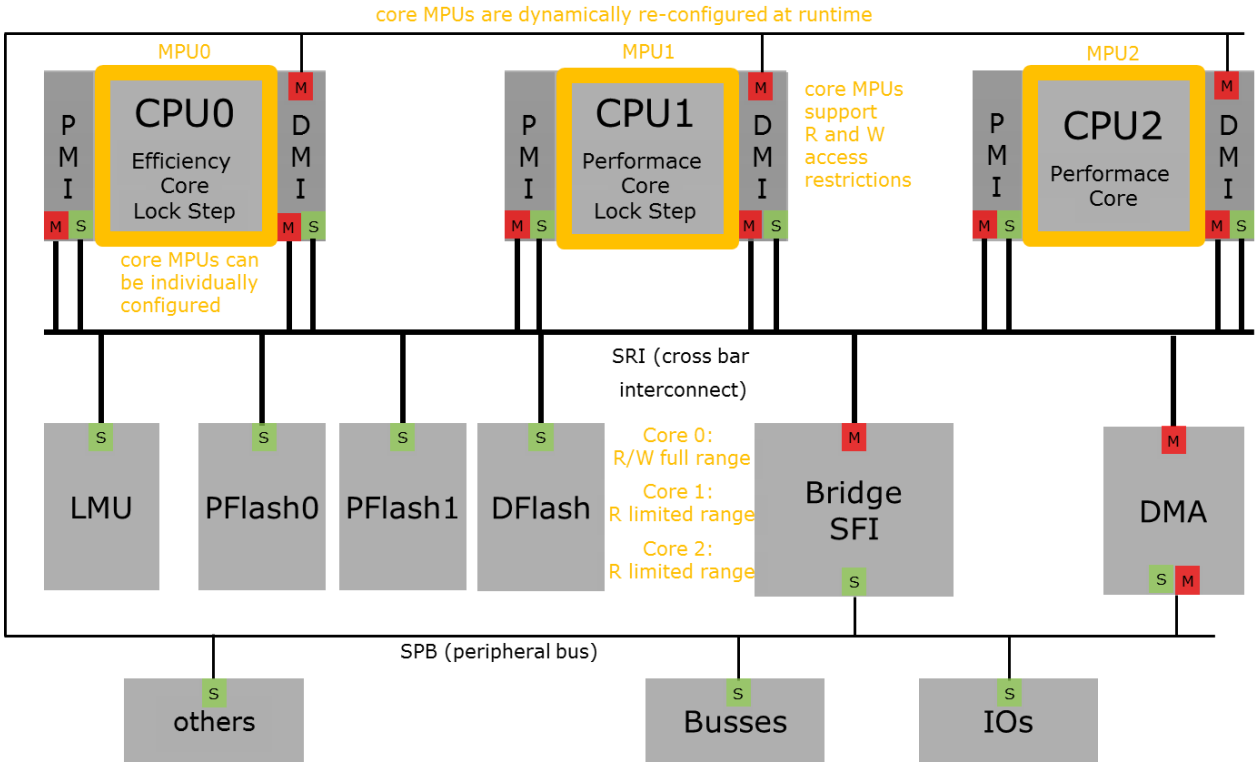


Abbildung 25: Core-MPUs der AURIX-Plattform

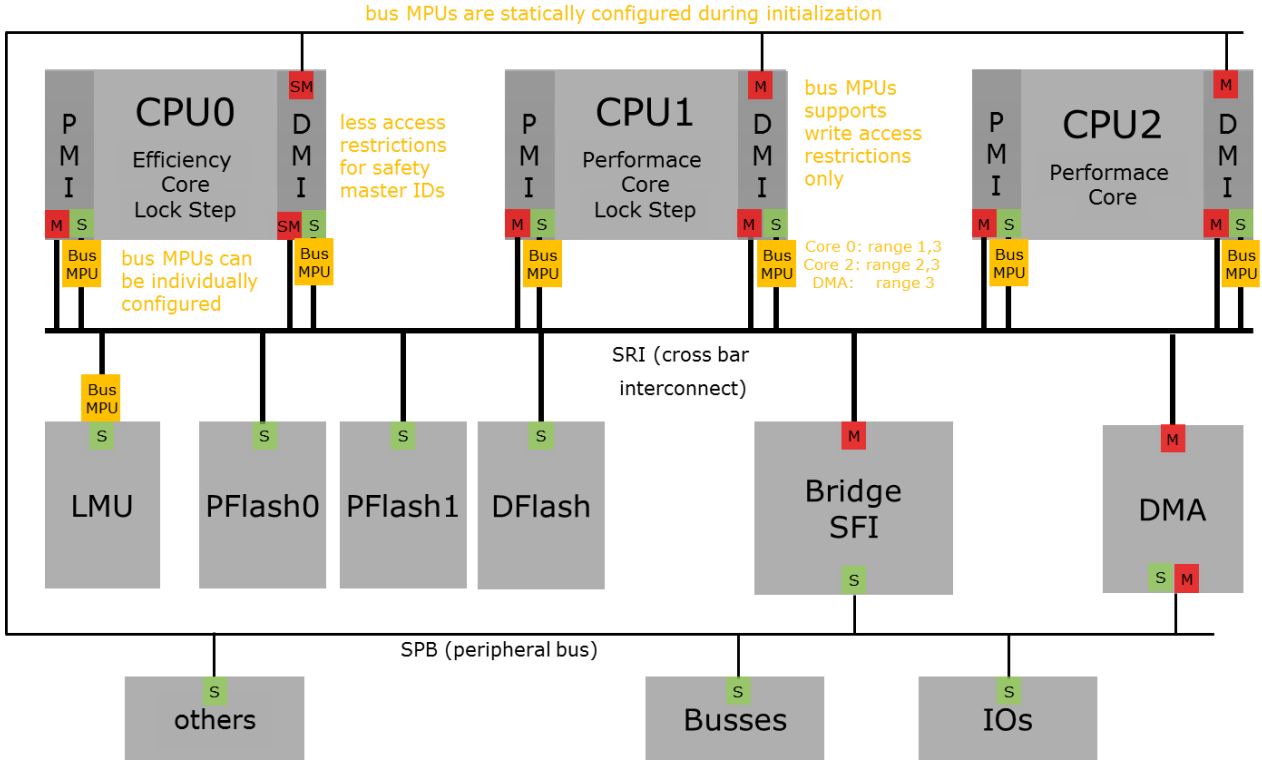


Abbildung 26 Bus-MPUs der AURIX-Plattform

Weitere Analysen wurden durchgeführt um die Möglichkeiten für den Schutz von Registerinhalten und für die Verwendung von Master-IDs für Zugriffsbeschränkungen zu erfassen. Es hat sich gezeigt dass es eine gute Unterstützung für Zugriffsschutz gibt, dass es aber auch Beschränkungen gibt, wie z. B. eine maximale Anzahl an Speicherbereichen.

6.1.2 Hardware-Sicherheitsarchitektur

In der ISO 26262 werden in Part 5 Fehler der Elektronik-Hardware aufgelistet, die in sicherheitsrelevanten Anwendungsfällen in Abhängigkeit des zu erreichenden Diagnosedeckungsgrads des betrachteten Systems zu berücksichtigen sind. Die entsprechenden Fehlermodelle müssen auf Bauteil-/Baugruppenebene für die einzelnen Elemente angesetzt werden und in entsprechende Hardware-Sicherheitsanalysen (z. B. FMEDA) einfließen. Dies setzt die vollständige Definition der Hardware voraus und ist wegen der typisch sehr großen Anzahl an Einzelbauteilen ein sehr aufwändiger Prozess. Große Teile der Analyse sind Plattform-spezifisch. Man muss eine hohe Wiederverwendung der Plattform erreichen, damit sich der Aufwand auf mehrere mit der Plattform bediente Anwendungen verteilen lässt.

Für Systeme mit hohen Anforderungen an Sensorik und Aktuatorik, dazu zählt beispielsweise auch eine Antriebssteuerung, wird die Flexibilität letztlich ausschlaggebend, da sie dem Anwender ermöglicht, die Auslegung bezüglich der wesentlichen Kriterien Sicherheit und Verfügbarkeit freier zu gestalten. Geteilte Ressourcen (wie z. B. eine Spannungsversorgung) stellen generell Quellen für mögliche „common cause“-Fehler dar, die für eine ausreichende Sicherheit des Systems durch geeignete Zusatzmaßnahmen ausgeschlossen werden müssen.

Abbildung 27 zeigt beispielhaft für eine Anwendung die Zuordnung von Peripherie-einheiten zu den Prozessorkernen eines Mehrkernprozessors. Es zeigt sich insbesondere, dass die Ressourcen zwar unterschiedlichen Kernen zugeordnet sind, dass i. W. aber eine Trennung in Digital-/Analog-IO/SPI und Bus-Kommunikation erfolgt. Weiterhin ist ersichtlich, dass der Prozessor auch sicherheitsrelevante Einschränkungen mit sich bringen kann. Hier ist z. B. auf Core 2 nur maximal ASIL B erreichbar, da dieser Kern im Unterschied zu den beiden anderen nicht über eine Lockstep-Absicherung verfügt.

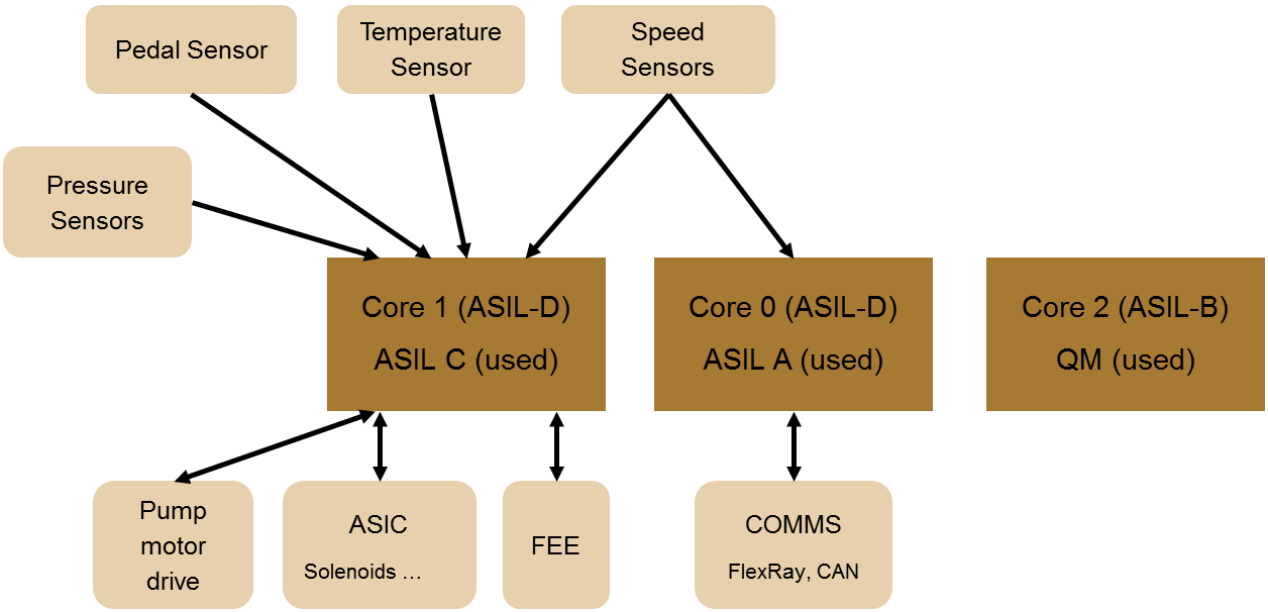


Abbildung 27: Zuordnung von Peripherie-Einheiten zu Kernen

In Antriebssteuerungen hat sich die Fail-Safe-Architektur behauptet weil sie die geforderte Sicherheit zu geringsten Kosten realisiert. Voraussetzung dafür ist, dass die Elektronik Fehler nur hinreichend gut erkennen muss und das System im Fehlerfall in einen sicheren Zustand bringen kann. Der sichere Zustand definiert sich systemabhängig, i. d. R. wird er aber dann eingenommen, wenn die Ausgänge für die Aktuatorik abgeschaltet werden.

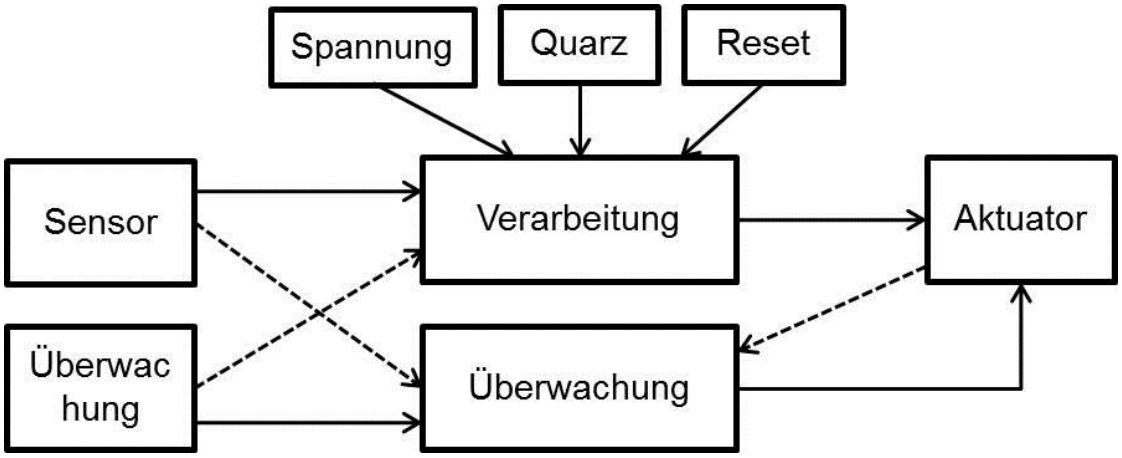


Abbildung 28: Fail-Safe-Systemarchitektur

Das Prinzip, dargestellt in Abbildung 28, besteht darin dass das System mit seinen Hauptkomponenten Sensor, Steuergerät und Aktuator durch unterschiedliche Diagnose-

Funktionen überwacht wird und der Aktuator bei einem Fehler abgeschaltet wird. Faktisch ist es so, dass auch in der Fail-Safe-Architektur wegen gemeinsamer Fehlerursachen eine externe Überwachung hinzukommt. Diese wird in Form eines Watchdog-Bausteins realisiert. In dem Überwachungspfad der Fail-Safe-Architektur wird dazu ein externer Baustein vorgesehen.

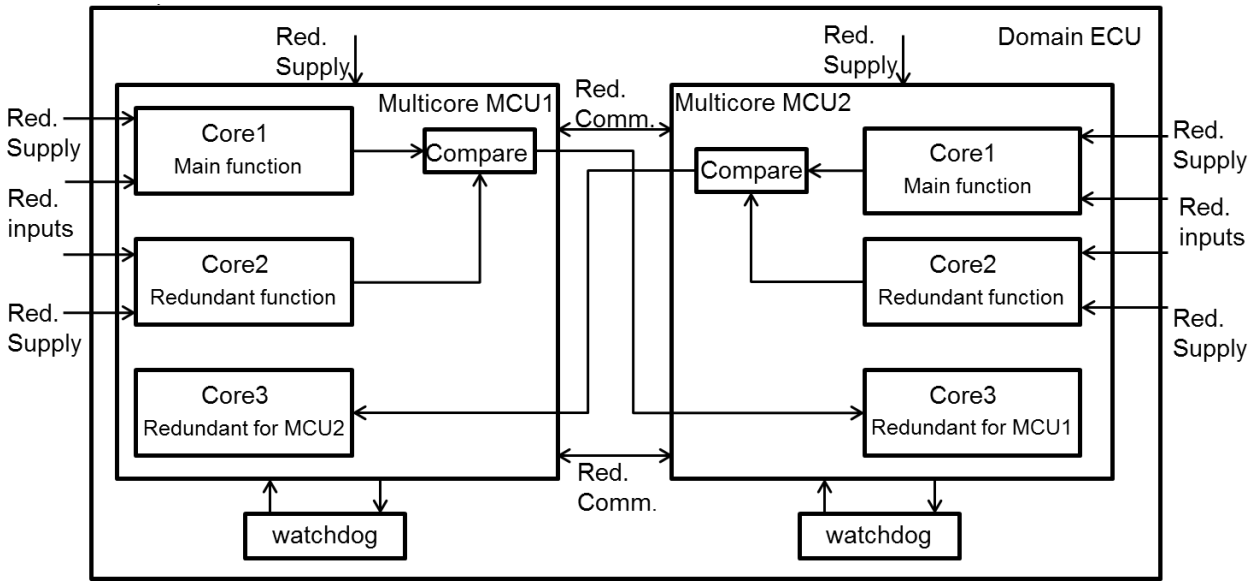
Wenn das Abschalten eines Systems gefährlich ist, weil dadurch eines oder mehrere Sicherheitsziele verletzt werden, müssen für solche Systeme Fehlertoleranzmaßnahmen entwickelt werden, die im Fehlerfall zumindest einen Notbetrieb ermöglichen. Ein solches System bezeichnet man als Fail-Operational-System. Um die Zuverlässigkeit durch Fail-Operational-Konzepte zu verbessern braucht es Konzepte, die üblicherweise mehr Hardware-Redundanz aufweisen als dies in einer Fail-Safe-Architektur der Fall ist.

Eine bekannte Fail-Operational-Architektur ist die 2oo3 (two-out-of-three)-Architektur, bei der mit 2-facher Redundanz entwickelt wird. Diese Architektur ist aufwändig und entsprechend teuer. Sie wird im Luftfahrtbereich eingesetzt weil dort der sichere Zustand nicht durch eine Abschaltung der Aktuatorik erreicht werden kann und weil die Kostensituation wegen der sehr hohen Systemkosten und wegen den eher kleinen Stückzahlen nicht zu einer anderen Entscheidung führt. Darauf basierende Systeme bieten eine hohe Fehlertoleranz und entsprechend eine sehr hohe Verfügbarkeit. Sie bestehen aus drei voll unabhängigen redundanten Elementen vom Sensor bis zum Aktuator. Diese drei unterschiedlichen Pfade werden gegeneinander plausibilisiert. Dadurch ist es möglich, den fehlerhaften Pfad bei einer Störung zu finden und zu isolieren. In einem Fehlerfall kann das System mit den anderen zwei funktionsfähigen Systemen aufrechterhalten werden.

Wie eingangs schon angemerkt kommen Mehrkernprozessoren immer stärker in heutigen Antriebssteuerungen im Automobil zum Einsatz. Es ist im Grunde auch klar, dass eine für die Verwendung im Automobil geeignete Basis-Hardwarearchitektur einen tragfähigen Kompromiss zwischen notwendiger Zuverlässigkeitserhöhung und den dafür aufzuwendenden zusätzlichen Systemkosten erzielen muss. Sie kann bezüglich der notwendigen Hardware-Ressourcen nur zwischen der eingangs beschriebenen Fail-Safe-Architektur und der sehr aufwändigen 2oo3-Architektur liegen. Im Grunde sind solche Zwischenlösungen auch heute schon für bestimmte spezifische Anforderungen entwickelt und im Einsatz, sie können aber nur verwendungsspezifisch optimiert werden.

Im Laufe der Untersuchungen hat sich gezeigt, dass es nicht einfach möglich ist, anhand eines Kriterienkatalogs eine optimale Basis-Hardwarearchitektur zu bestimmen. Im Einzelfall hängt die optimale Hardwarearchitektur stets von den exakten Randbedingungen ab. Mit den zuvor geschilderten Erkenntnissen und der zusätzlich schon beschriebenen Notwendigkeit, Mehrkernprozessoren in die Betrachtung einzubeziehen, wurde die in Abbildung 29 dargestellte Basis-Hardwarearchitektur gefunden. Aus Gründen der Übersichtlichkeit sind darin nicht alle Kommunikationspfade eingezeichnet. Haupteigenschaft dieses Konzepts ist die Verwendung einer Rückfallebene

für den fehlerhaften Prozessor oder Rechenkern. Die sicherheitskritischen Funktionen werden auf dem jeweiligen Prozessor redundant gerechnet und gegeneinander verglichen. Wenn die Ergebnisse nicht übereinstimmen, werden diese Ergebnisse mit dem Ergebnis vom anderen Prozessor verglichen, um den fehlerhaften Pfad rauszufinden. Das System kann danach mit den fehlerfreien Pfaden aktiv weiterarbeiten. Es sei angemerkt, dass der Aufwand immer noch recht hoch ist, denn von drei Prozessorkernen kann nur ein Kern pro Prozessor frei verwendet werden. Man kann also einen Prozessor einsparen, muss aber Steuerungsaufgaben für 2 Teilsysteme haben. Zusätzlich ist ein hoher Aufwand erforderlich um mögliche gemeinsame Fehlerursachen („common causes“), z. B. bei Spannungsversorgung und Taktgenerierung auszuschalten.



Anmerkung: Aus Übersichtlichkeitsgründen sind nicht alle Kommunikationspfade eingezeichnet.

Abbildung 29: Fehlertolerante Hardwarearchitektur

Erst eine spezifische Anwendung für eine konkrete Problemstellung macht es möglich, die Struktur im Hinblick auf Zuverlässigkeit und Kosten zu optimieren. Im Einzelfall kann ggf. auch noch Redundanz entfallen. Gleichzeitig müssen die vorhandenen Abhängigkeiten als mögliche gemeinsame Fehlerursachen und die Mechanismen der Fehlerausbreitung in der Software verstanden werden um die Sicherheitsrisiken richtig einschätzen zu können.

Alle Kerne eines Mehrkernprozessors befinden sich auf einem einzigen Silizium-Chip und teilen typischerweise Versorgungsspannungen, aber auch physikalische Eigenschaften wie das Temperaturniveau. Wenn hier Redundanz erforderlich ist, um das geforderte Sicherheitsniveau einzuhalten, geht es nicht ohne andere ECUs oder andere Controller in die Absicherung einzubeziehen.

6.2 Fehlertolerante Softwarearchitektur

Die Arbeiten dieses Kapitel umfassen Definition, Analyse und Vergleich angepasster, fehlertoleranter Softwarearchitekturen für die Steuerung und Regelung von Elektroantrieben.

6.2.1 Freiheitsgrade in der Softwarearchitektur

Die Freiheiten in der Softwarearchitektur hinsichtlich Sicherheit und Zuverlässigkeit sind gering wenn auf Funktions- und Hardwareebene eine Architektur mit minimaler Redundanz vorgegeben wird. Unter diesen Voraussetzungen muss die Softwarearchitektur vor allem sicherstellen dass die vorgesehene Redundanz wirksam eingesetzt werden kann, um das damit angestrebte Sicherheitsziel zu erfüllen. Dabei geht es vor allem darum, die Software zweckdienlich auf die Kerne eines Mehrkernprozessors zu verteilen und die Möglichkeiten der Hardware und insbesondere des Prozessors geeignet zu nutzen. Zusätzlich wird die Softwareverteilung auch durch das bereits in Kapitel 3.1 dargestellte E-Gas-Konzept eingeschränkt, denn zwischen den verschiedenen Levels muss die Rückwirkungsfreiheit, häufig mit dem englischen Begriff „freedom from interference“ bezeichnet, sichergestellt werden und es wird eine unabhängige Hardwareeinheit, als „Monitoring Processor“ bezeichnet, vorgegeben.

Können die Umgebungs-/Betriebsbedingungen den Funktionsprozessor zu einem Ausfall bringen und damit ein wichtiges Sicherheitsziel verletzen, dann muss diese zusätzliche Hardwareeinheit in der Lage sein den Ausfall zu erkennen und eine kontrollierte Abschaltung der ausgefallenen Einheit vorzunehmen. Will man den Ausfall zusätzlich tolerieren dann muss die Hardwareeinheit darüber hinaus in der Lage sein zumindest ein reduziertes Programm aufrecht zu erhalten.

6.2.2 Vergleich von grundlegenden Softwareverteilszenarien

Legt man die für die sicherheitsrelevanten Antriebsysteme etablierten Schichten-architekturen, z. B. mit Trennung in Applikation (funktionale Ansteuerung und Sicherheits-monitore entsprechend dem E-Gas-Konzept) und Basissoftware (siehe auch AUTOSAR), zu Grunde dann können darauf basierend die grundlegenden Eigenschaften einer Verteilung der Software auf mehrere Kerne analysiert werden.

In diesem Sinne wurden zwei Extremszenarien betrachtet. Zum einen wurde ein horizontaler Schnitt zur Verteilung angelegt, indem die in AUTOSAR definierte RTE als Schnittstelle für eine Verteilung herangezogen wurde (dies entspricht dem in AUTOSAR Release 4.x nahegelegten und von dem spezifizierten Multicore-OS unterstützten Vorgehen). Das Schema eines solchen horizontalen Schnitts ist in Abbildung 30 zu sehen.

Ein gravierender Nachteil dieses Verteil-Szenario ist unmittelbar ersichtlich. Für sämtliche Eingaben und Ausgaben von Daten müssen sich die SW-Anteile auf den verschiedenen Kernen intensiv austauschen und es entstehen große Abhängigkeiten, um z. B. die Datenintegrität oder auch die Fehlerlatenzzeiten innerhalb der Wirkketten des Systems (Input – Verarbeitung in der Applikation – Output) zu beherrschen. Beides sind sehr

wichtige Kriterien im Hinblick auf die funktionale Sicherheit des Systems. Die Möglichkeiten, Redundanz zu nutzen, werden mit dieser Teilung ebenfalls erschwert, da die Monitore im Normalfall in dieser Architektur auch Applikationen darstellen. Die Möglichkeiten für eine Erhöhung der Verfügbarkeit sind somit ebenfalls eingeschränkt.

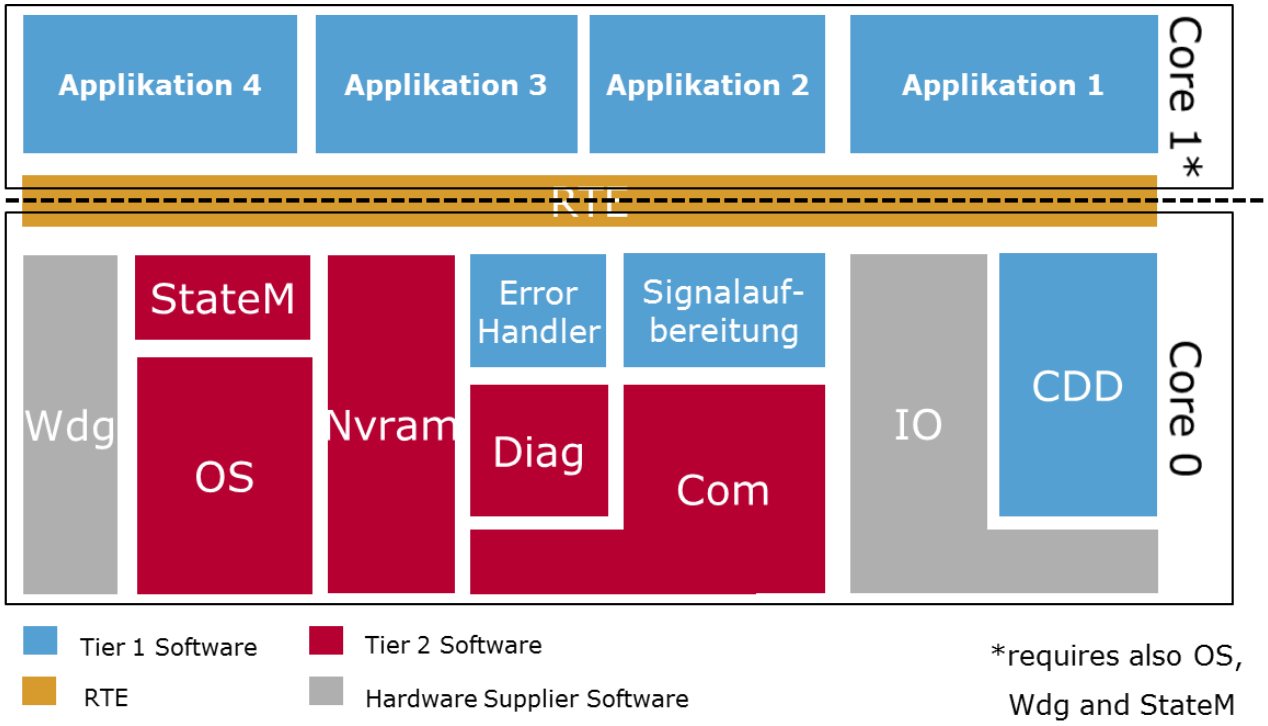


Abbildung 30: Horizontaler Schnitt für die SW-Verteilung

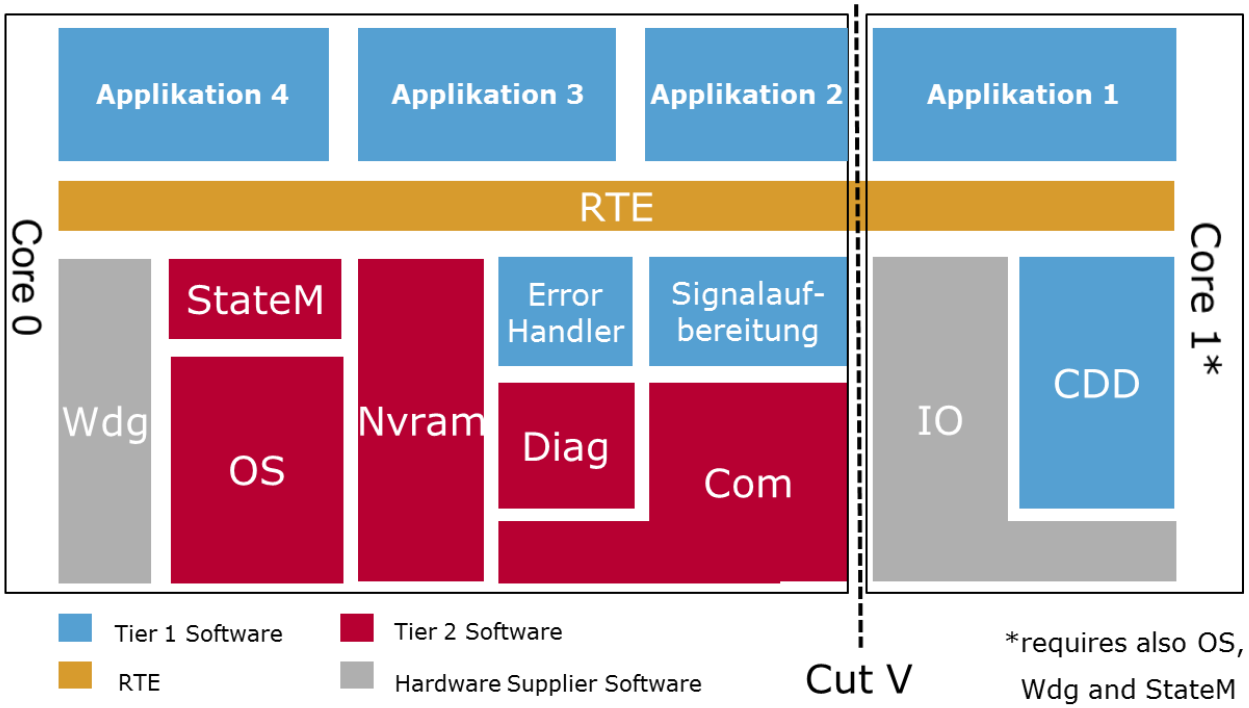


Abbildung 31: Vertikaler Schnitt für die SW-Verteilung

Als zweites Extremszenario wurde ein vertikaler Schnitt in der Software betrachtet, um physikalische Wirkketten in einem Ausführungspfad zusammenzuführen und um dadurch die Notwendigkeit von Kern-übergreifender Synchronisation/Kommunikation zu verringern und so die wesentlichen Nachteile des horizontalen Schnitts zu vermeiden. Das Schema dieses Schnitts ist in Abbildung 31 dargestellt.

Der vertikale Schnitt eignet sich grundsätzlich um die gravierenden Nachteile des horizontalen Schnitts zu vermeiden, er erhöht jedoch den Aufwand dadurch, dass jeder Kern im Extremfall beliebigen Zugriff auf die IO-Ressourcen benötigt.

Als Resümee daraus bleibt festzuhalten dass vertikale Schnitte, eingeschränkt durch ein Abwägen zwischen Gewinn an Autonomie auf den Kernen und dem für die Realisierung der Verteilung benötigten Aufwand eine sinnvolle Verteilung ermöglichen und dass sie für Fehlertoleranz im Sinne möglichst autonomer Teilsysteme eine Basis bieten. Tabelle 8 fasst die wesentlichen Vorteile und Nachteile der beiden Szenarien zusammen.

Tabelle 8: Vorteile und Nachteile der beiden Extremszenarien

	Horizontaler Schnitt
Vorteile	• CPU0 steuert alle Peripherieeinheiten und alle Interrupts. • CPU1 hat nur wenig Interaktion mit der HW und daher keine Unterbrechung der Prozesse durch HW-Interrupts.
Nachteile	• Alle Busdaten und IOs müssen zwischen Kernen ausgetauscht werden. • Zusätzliche Latenzzeiten bei jeder Signalverarbeitung. • Keine Parallelisierung von Prozessketten.
	Vertikaler Schnitt
Vorteile	• Prozessketten können parallelisiert werden. • Datenaustausch zwischen Kernen kann optimiert werden. • Im Bedarfsfall können existierende Applikations-spezifische SW-Anteile angepasst werden.
Nachteile	• Interrupt-Handling und HW-Zugriffe auf beiden Kernen. • Zusätzliche Latenzzeiten bei jeder Signalverarbeitung.

In der praktischen Anwendung findet man oft eine Mischung der beiden Szenarien. Einen vertikalen Schnitt wird man generell ansetzen wenn man die Fehlertoleranz verbessern muss, nur damit erreicht man die dazu notwendige Unabhängigkeit.

6.2.3 Autonomie der Kerne und Datenaustausch zwischen Kernen

Fehlertoleranz ist verbunden mit Redundanz die über möglichst autarke Systemeinheiten erzielt werden muss. Die weitgehende Autonomie der Software auf den Kernen wiederum ist an bestimmte Gegebenheiten geknüpft. Diese sind in Abbildung 32 schematisch dargestellt.

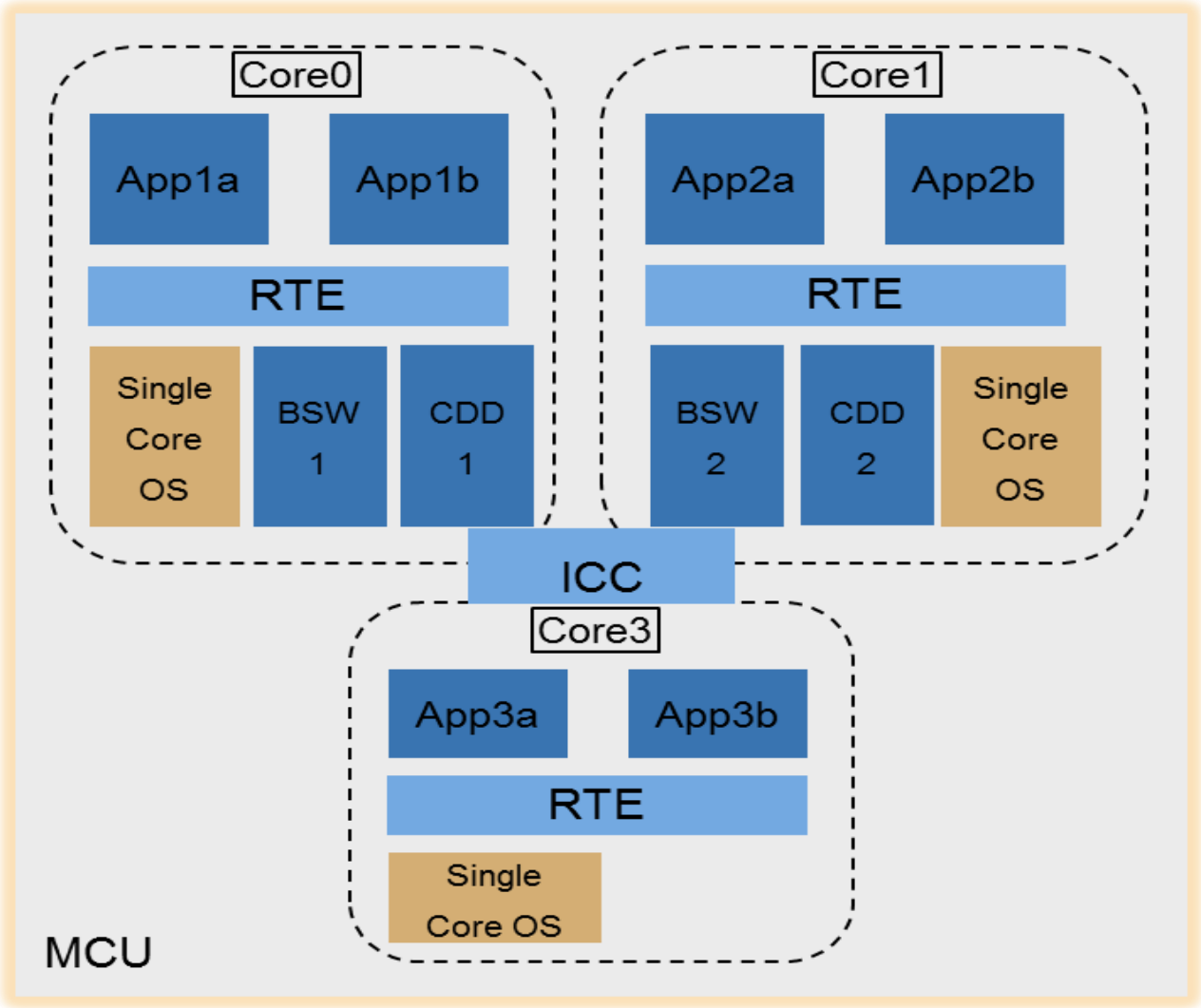


Abbildung 32: SW-Architektur für verbesserte Fehlertoleranz auf 3 Kernen

Die Anforderungen lassen sich folgendermaßen zusammenfassen:

- Jeder Kern hat unabhängigen Zugang zu den Ressourcen, z. B. jeweils ein eigenes OS oder Zugang zu den auf dem Kern benötigten Sensor- und Bussignalen.
- Basissoftware wird auf den Kernen nach Bedarf umgesetzt.
- Applikationssoftware kann parallelisiert und auf verschiedenen Kernen betrieben werden (Redundanz oder Verteilung).
- Sicherstellen der „Freedom from Interference“ zwischen den Kernen ist i.A. notwendig.

Anforderungen an die Kommunikation zwischen Kernen entstehen durch die Verteilung von OS-Applikationen und von OS-Tasks auf Kerne wie in Abbildung 33 dargestellt.

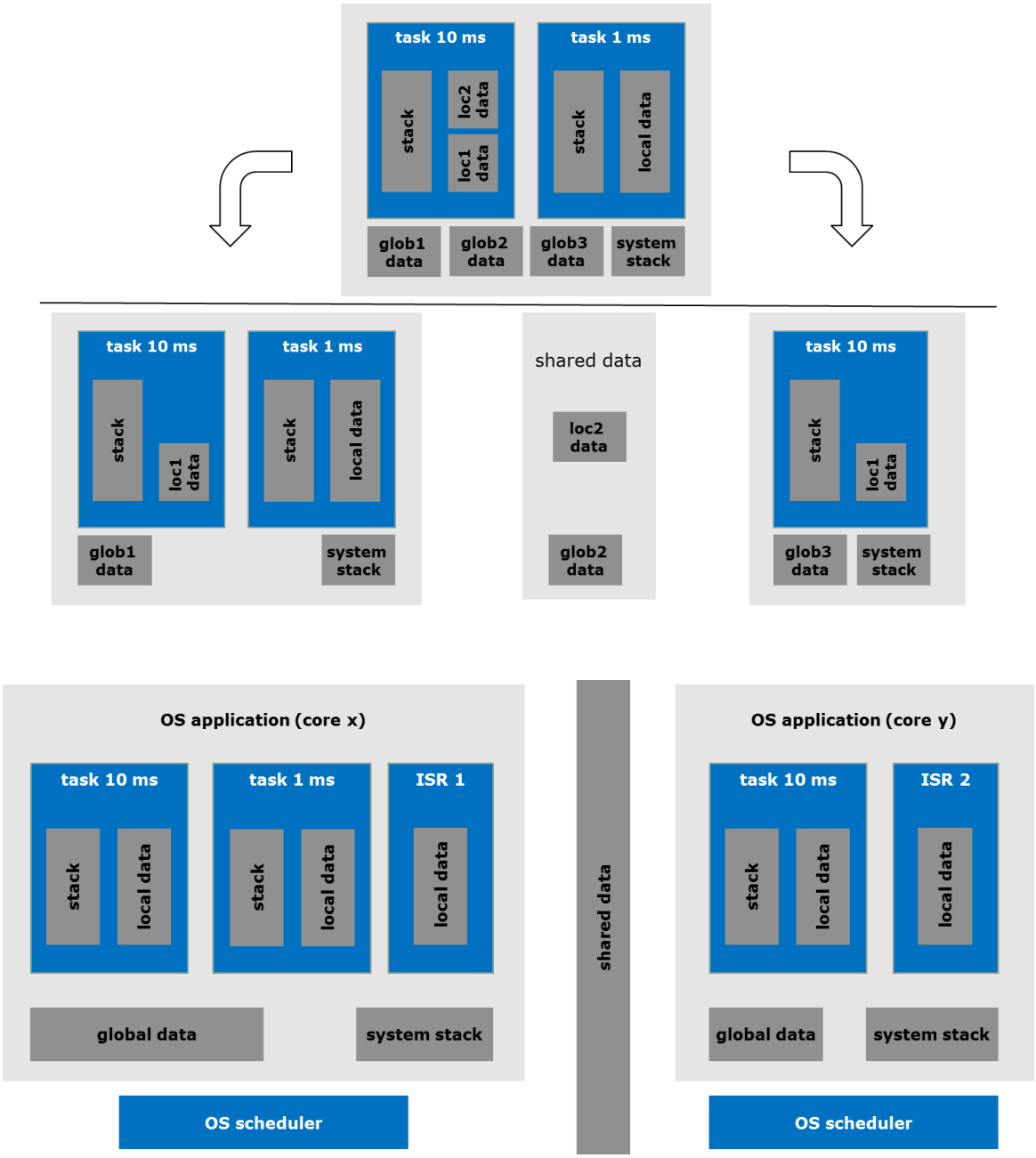


Abbildung 33: Zuordnung von OS-Applications und OS-Tasks auf Kerne

Der Datenaustausch muss für sicherheitsrelevante Informationen abgesichert erfolgen. Der Mechanismus dafür kann zur Erhöhung der Fehlertoleranz in der Kommunikation grundsätzlich um Fehlerkorrekturmechanismen erweitert werden. Das ist dann sinnvoll wenn es auf dem Übertragungsweg zu Verfälschungen kommen kann.

Solche Korrekturen sind zur Absicherung sporadischer Fehler und für bestimmte Fehlerbilder in den Daten bestimmter Speicherobjekte z. B. mittels ECC üblich.

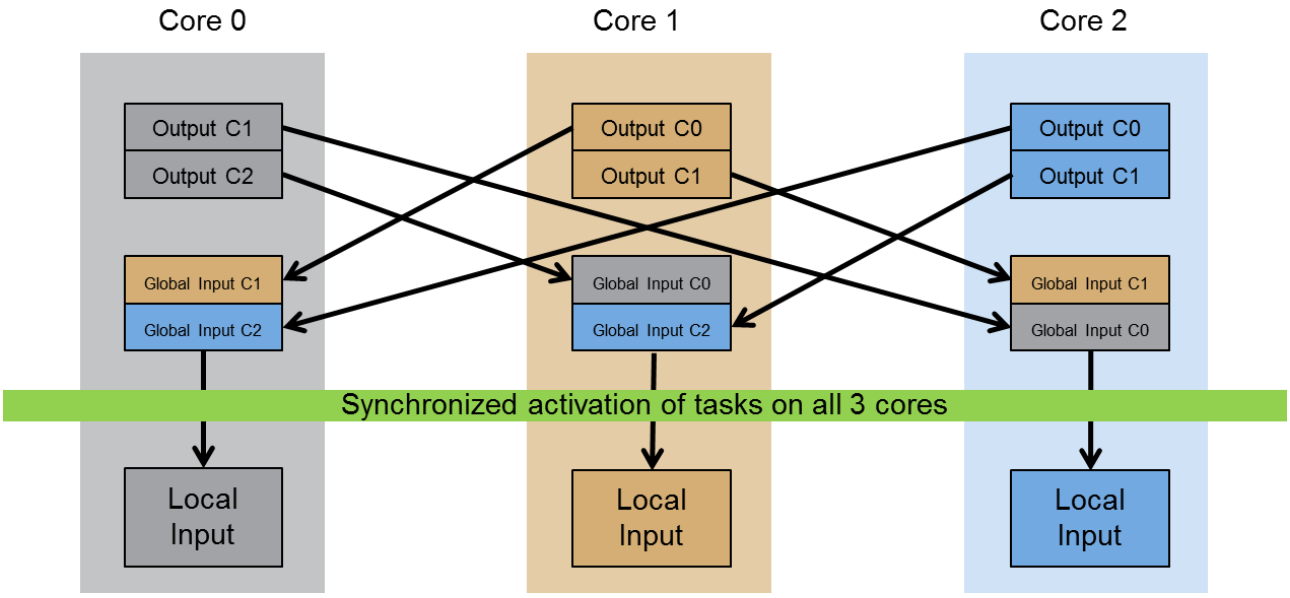


Abbildung 34: Datenaustausch zwischen den Kernen

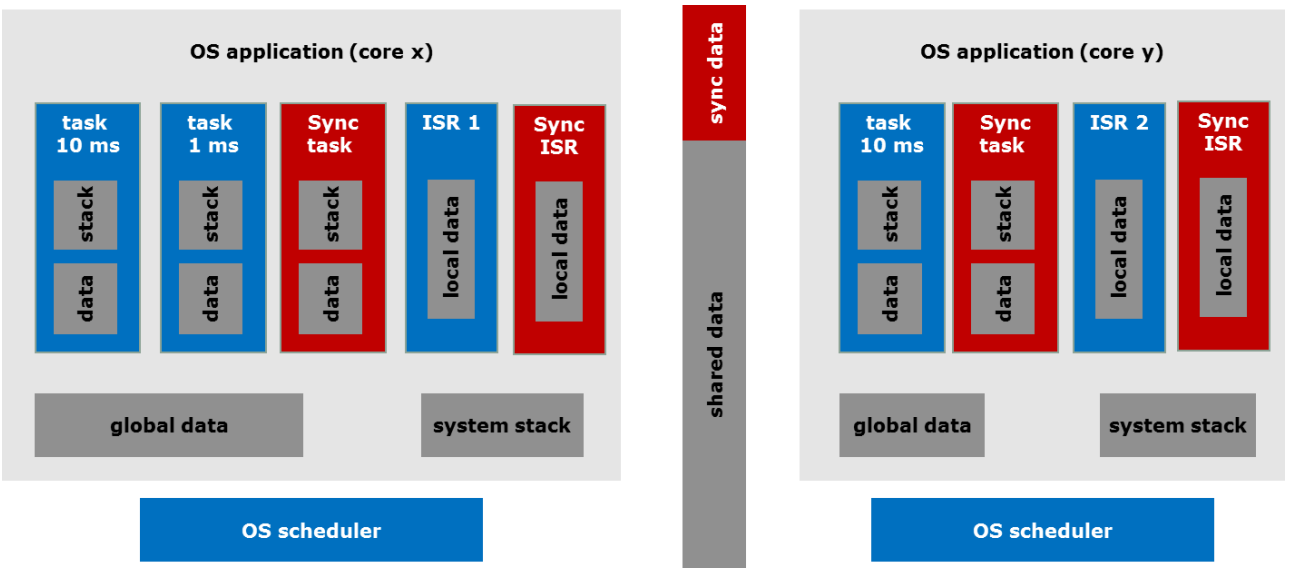


Abbildung 35: OS-Applikationen und zusätzliche Synchronisationsobjekte

In Busprotokollen wie z. B. für die Übertragung von Daten auf dem CAN-Bus gibt es als Methode zur Absicherung eine Fehlererkennung und Wiederholung der Übertragung. Dies beinhaltet ebenfalls Fehlertoleranz, das Vorgehen kann aber kritisch in Bezug auf die geforderten Fehlerlatenzzeiten sein. Die Fehlerlatenzzeiten müssen entsprechend geprüft werden. Typischerweise wird für Steuerungssysteme die über einen Bus miteinander kommunizieren aber jeweils ein hohes Maß an Eigensicherheit verlangt.

Ein wesentliches Element bilden die in Kapitel 6.1.1 beschriebenen Speicherschutz-mechanismen. Um diese mit bestehenden Lösungen nutzen zu können müssen teilweise

Client/Server- zu Sender/Receiver-Kommunikationsschnittstellen umgesetzt werden. Dazu können C/S-Proxy-Komponenten wie in Abbildung 36 skizziert verwendet werden.

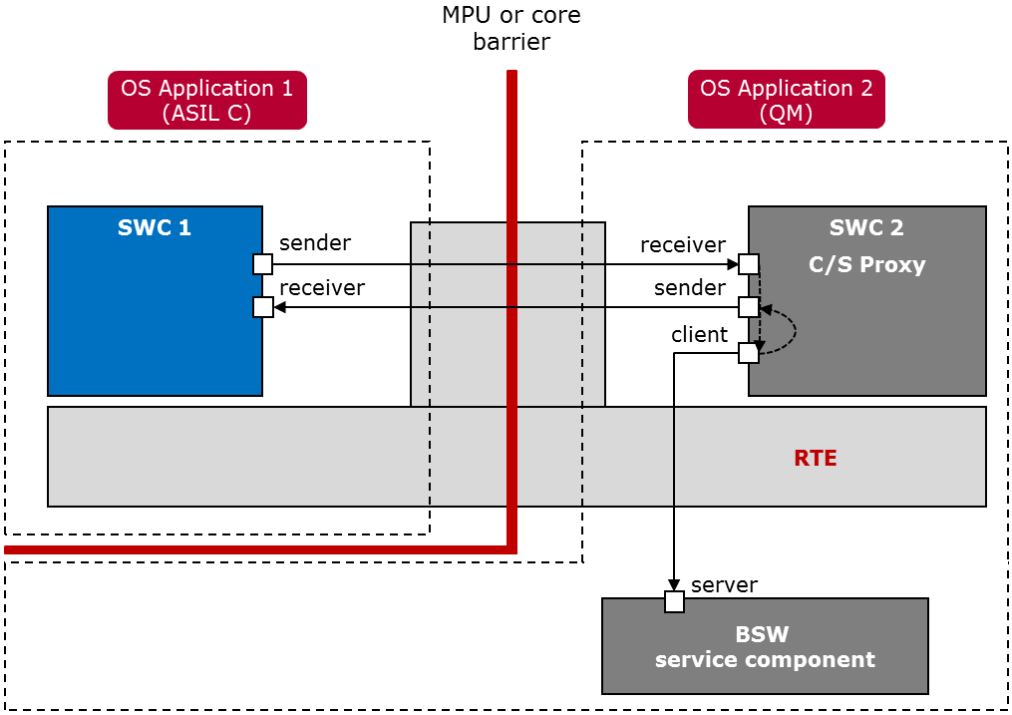


Abbildung 36: Beispiel für Verwendung der MPU (ASIL C ←→ QM)

6.3 Fehlertolerante Betriebsstrategien

Die Arbeiten dieses Kapitel umfassen die Simulation von fehlertoleranten Betriebsstrategien für den Elektrofahrzeug(EV)- und Plug-in-Hybrid(PHV)-Antrieb.

6.3.1 Gesamtsystemarchitektur und betrachtete Fehler

Der Simulation von fehlertoleranten Betriebsstrategien wurden Ausfallszenarien die auf Systemebene mit entsprechenden fehlertoleranten Systemarchitekturansätzen identifiziert wurden, sowie eine fehlertolerante Hardwarearchitektur mit Mehrkernprozessoren und die Analyseergebnisse zur Softwarearchitektur zu Grunde gelegt.

Betrachtet wurden speziell folgende Fehlersituationen

- Ausfall Drehzahlsensor
- Ausfall Temperatursensor
- Ausfall Phasenstrom

Für die PHV wurden die Ausfälle auf Systemebene anhand der PHV-Gesamtsystemarchitektur, dargestellt in Abbildung 37, analysiert und bezüglich der Fehlerauswirkungen simuliert.

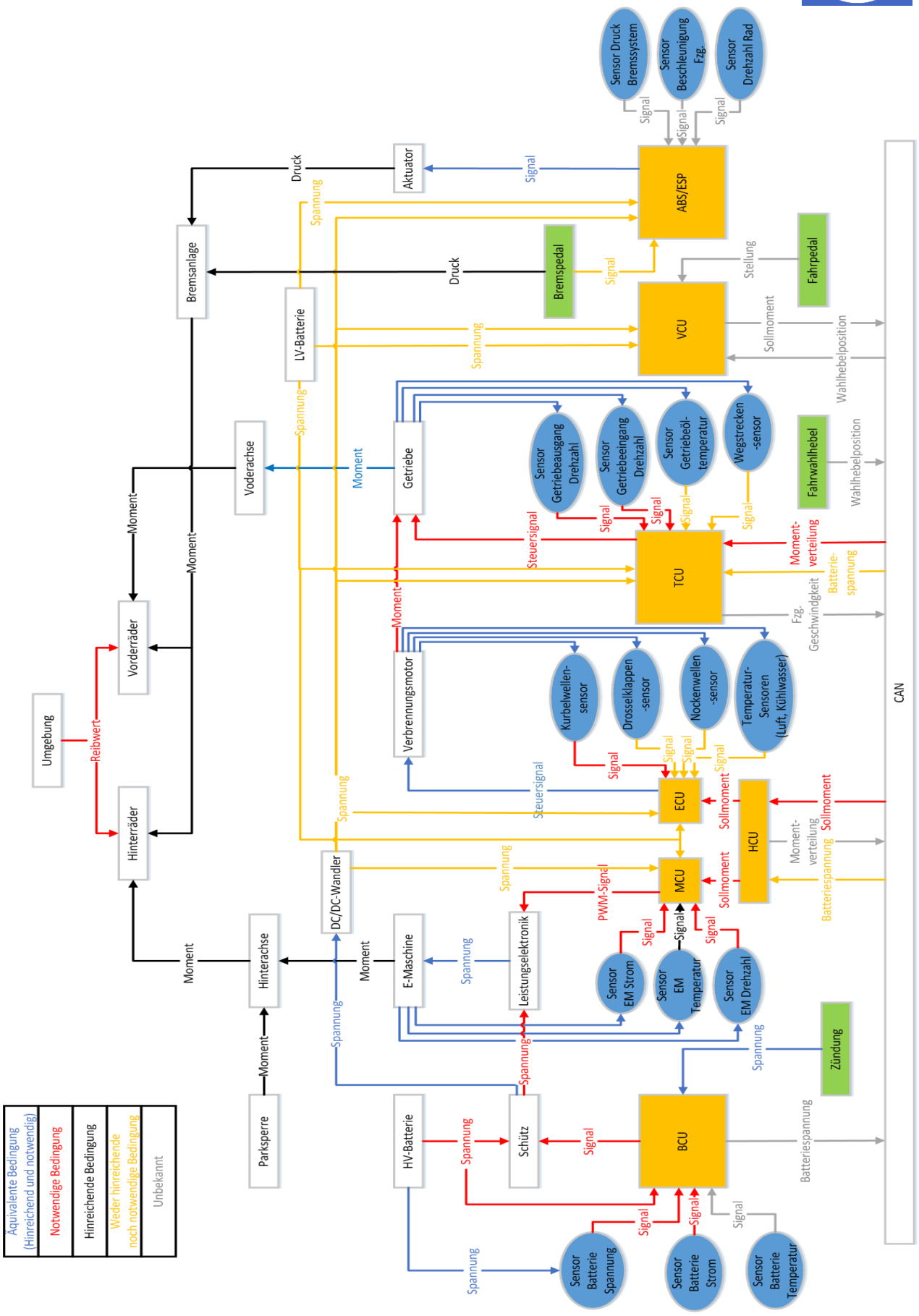


Abbildung 37: Gesamtsystemarchitektur PHV

6.3.2 Simulation fehlertoleranter Betriebsstrategien

Für die Einbeziehung von Softwarearchitektur und Hardwarearchitektur in die Simulation wurde die im Kapitel 4.1 eingeführte Methode verwendet und die Modellierungen entsprechend erweitert. Dazu wurden die fehlertoleranten Architekturen für Hardware und Software (soweit dies in Simulation sinnvoll möglich war) in die im Anhang A illustrierte Eclipse-Entwicklungsumgebung integriert.

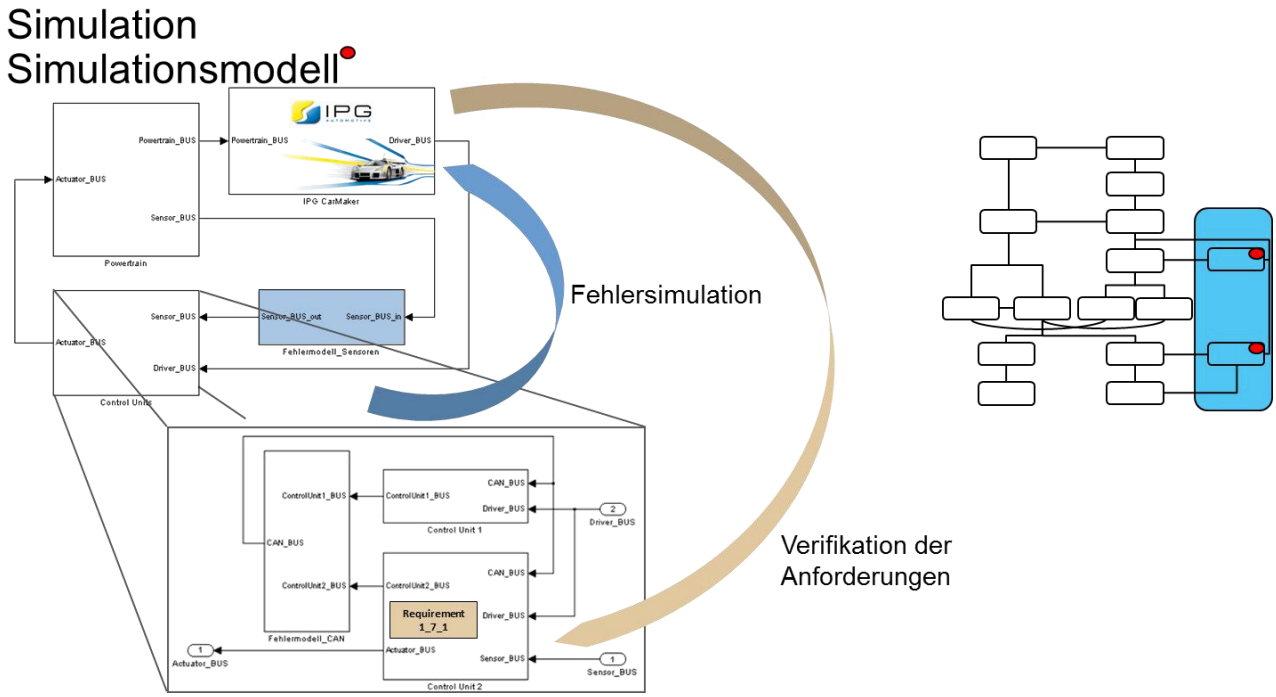


Abbildung 38: Simulationsmodell für Fehlersimulation

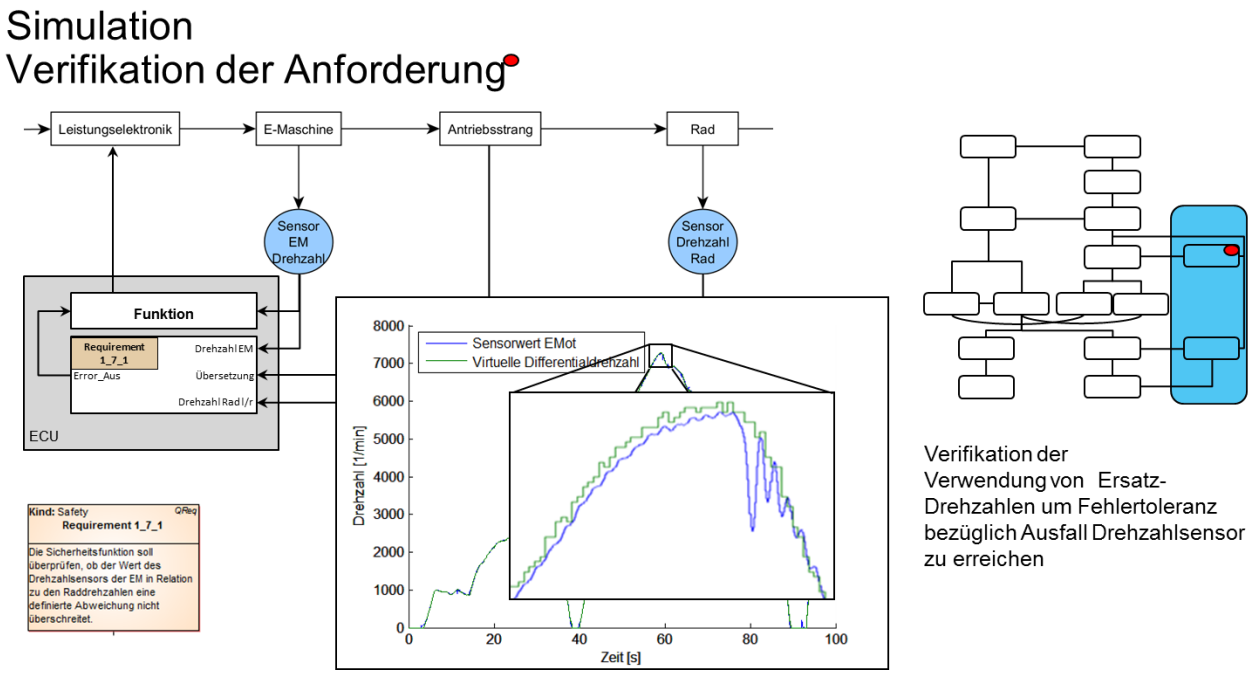


Abbildung 39: Anforderungsverifikation für Fehlertoleranz bei Ausfall Drehzahlsensor

Eine weitere Simulation untersuchte die Auswirkungen eines Zentralisierungsansatzes für die EV-Antriebssteuerung durch Zusammenfassen von E-Motorsteuerung und E-Fahrzeugsteuerung in eine Antriebssteuerung.

Abbildung 40 zeigt das in Simulink© erstellte Simulationsmodell.

Abbildung 41 zeigt die Ausgangsarchitektur und Abbildung 42 die Architektur entsprechend dem Zentralisierungsansatz.

Hier hat sich gezeigt dass zwar über die Plausibilisierung der Eingänge eine Verbesserung der Fehlertoleranz möglich ist, dass aber die Unabhängigkeit für Abschaltungen im Fehlerfall schwieriger wird. Im Ergebnis hat die Zentralisierung ihre Grenzen, beziehungsweise führt sie aus Gründen der Sicherheit quasi wieder zu einer „lokalen Dezentralisierung“. Aus dieser Betrachtung folgte letztlich die fehlertolerante Hardwarearchitektur die in Abbildung 29 dargestellt ist.

Die Umsetzung im Seriensteuergerät war nicht Ziel des Projekts, daher konnten nicht alle Details in die Simulation einbezogen werden. Insbesondere kann somit nicht komplett ausgeschlossen werden, dass es in der Umsetzung noch nicht berücksichtigte Probleme bei der Realisierung der Unabhängigkeit von Softwareanteilen gibt oder dass zugunsten der Kosten weitere Optimierungen erfolgen müssen.

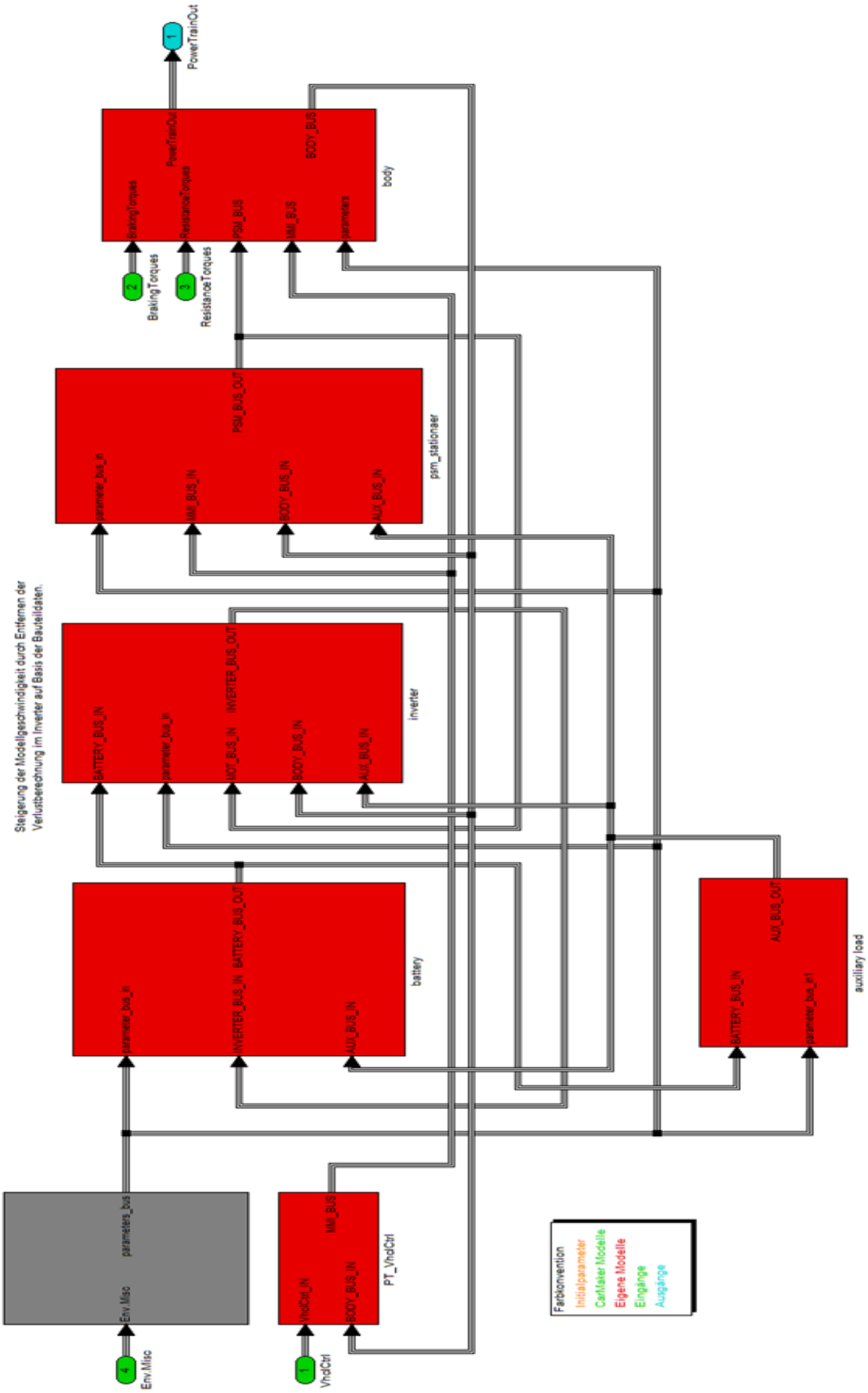


Abbildung 40: EV-Antrieb Simulationsmodell

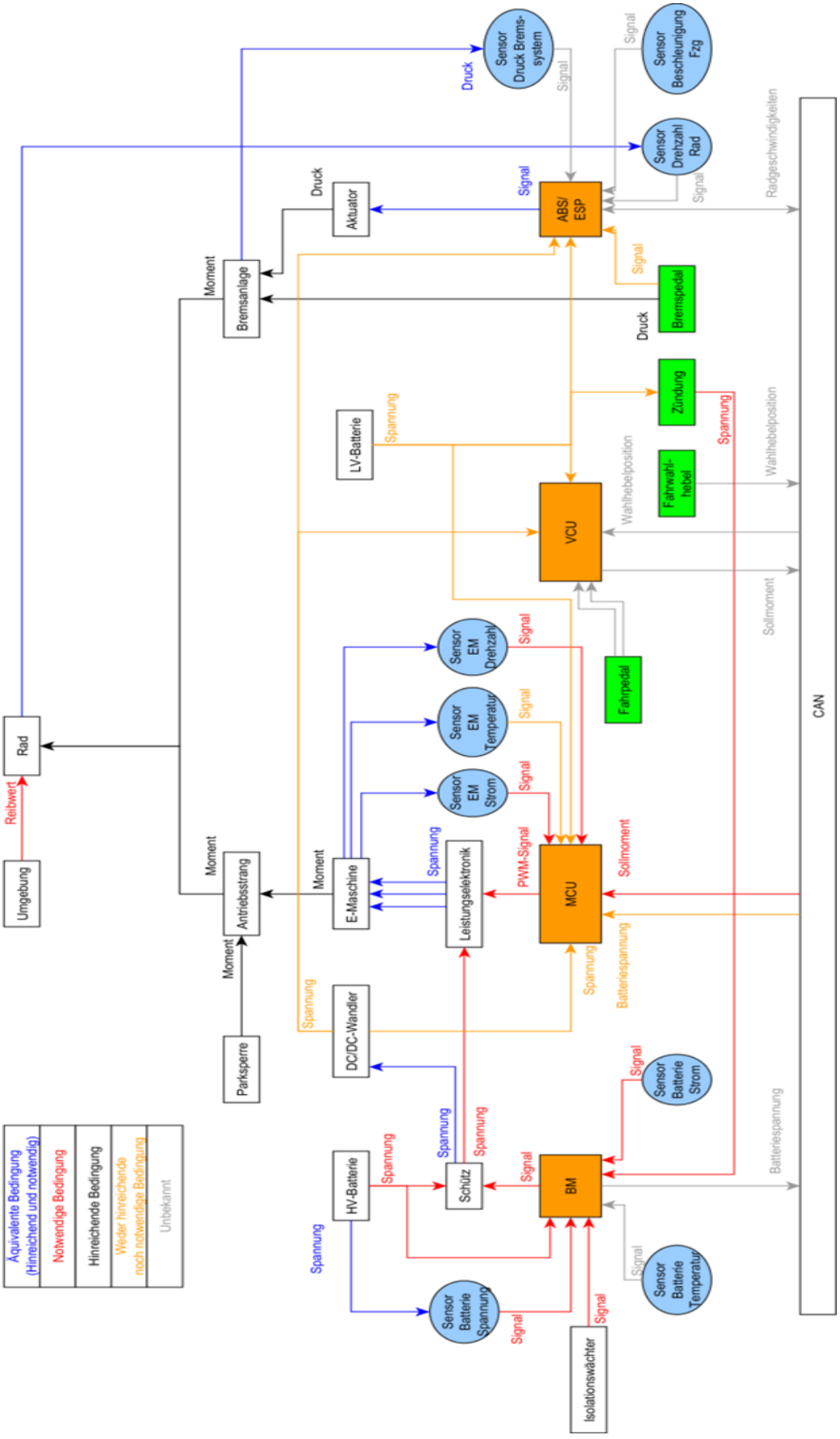


Abbildung 41: EV-Antriebssteuerung mit separatem Fahrzeugrechner

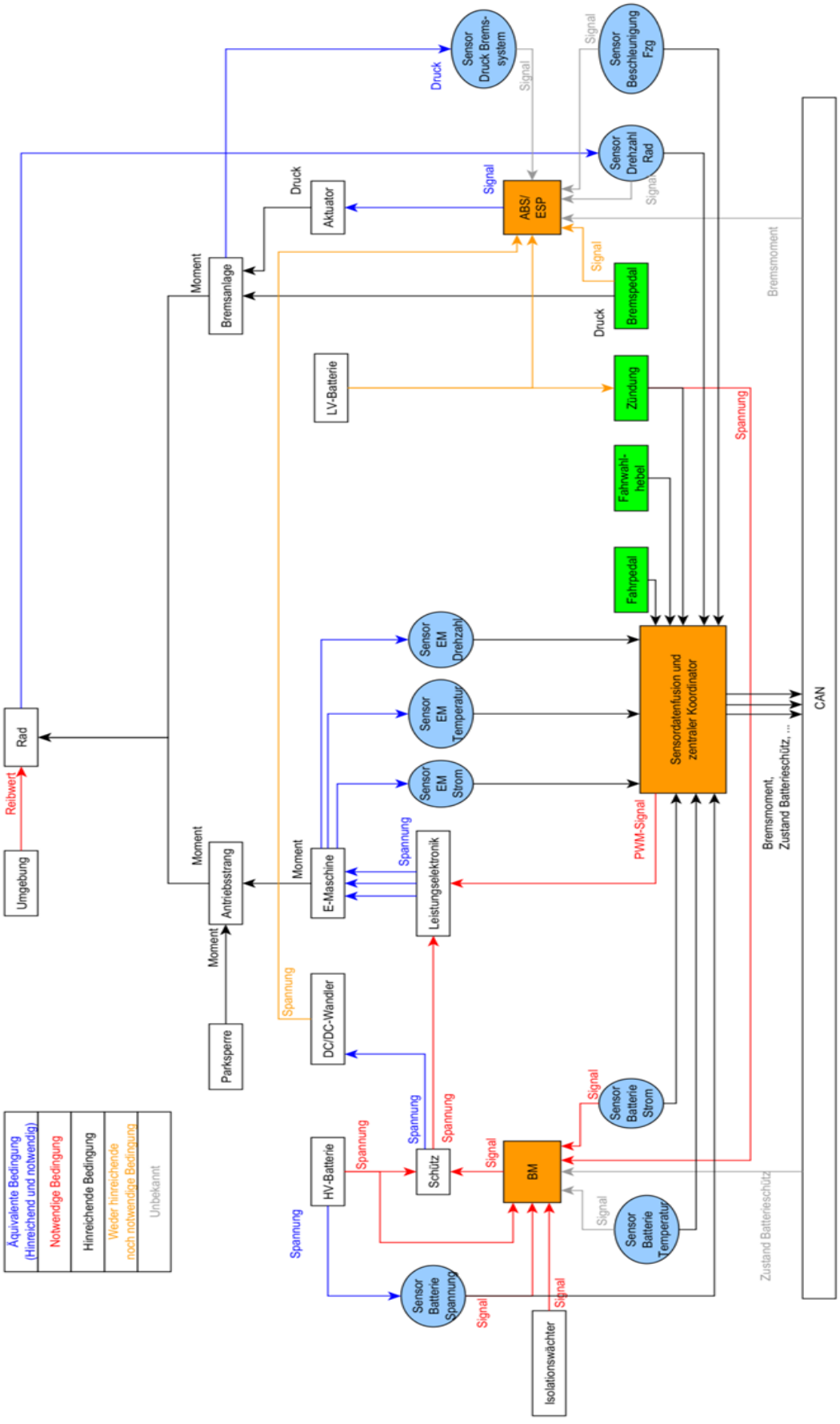


Abbildung 42: EV-Antriebssteuerung zentralisiert

7 Zusatznutzen der fehlertoleranten Systemarchitektur

Im Zuge der Bearbeitung von Arbeitspaket 5.3 wurden die erzielte Systemarchitektur im Hinblick auf ihren Nutzen für die Erhöhung von Zuverlässigkeit und Verfügbarkeit und die HW- und SW-Architekturen im Hinblick auf ihre Eignung für Prüfungen mittels Simulation früh in der Systementwicklung bewertet. Zusätzlich wurden ausgewählte Projektergebnisse auf dem Fahrsimulator der Uni Stuttgart umgesetzt. Dies wird anhand von einigen Beispielen dargestellt.

7.1 Analyse und Beschreibung des Zusatznutzens

Für die Ermittlung der Synergien wurden die in Kapitel 5.3.2 angesprochenen Fehlerbehandlungsstrategien für den Drehzahlsensor, Temperatursensor und die Phasenstromsensoren analysiert. Alle untersuchten Fehler und entwickelten Fehlerbehandlungsstrategien haben Einfluss auf die Regelung der E-Maschine (Antriebsstrangmanagement). Das Ziel ist es somit, eine fehlertolerante Regelung der E-Maschine zu entwickeln.

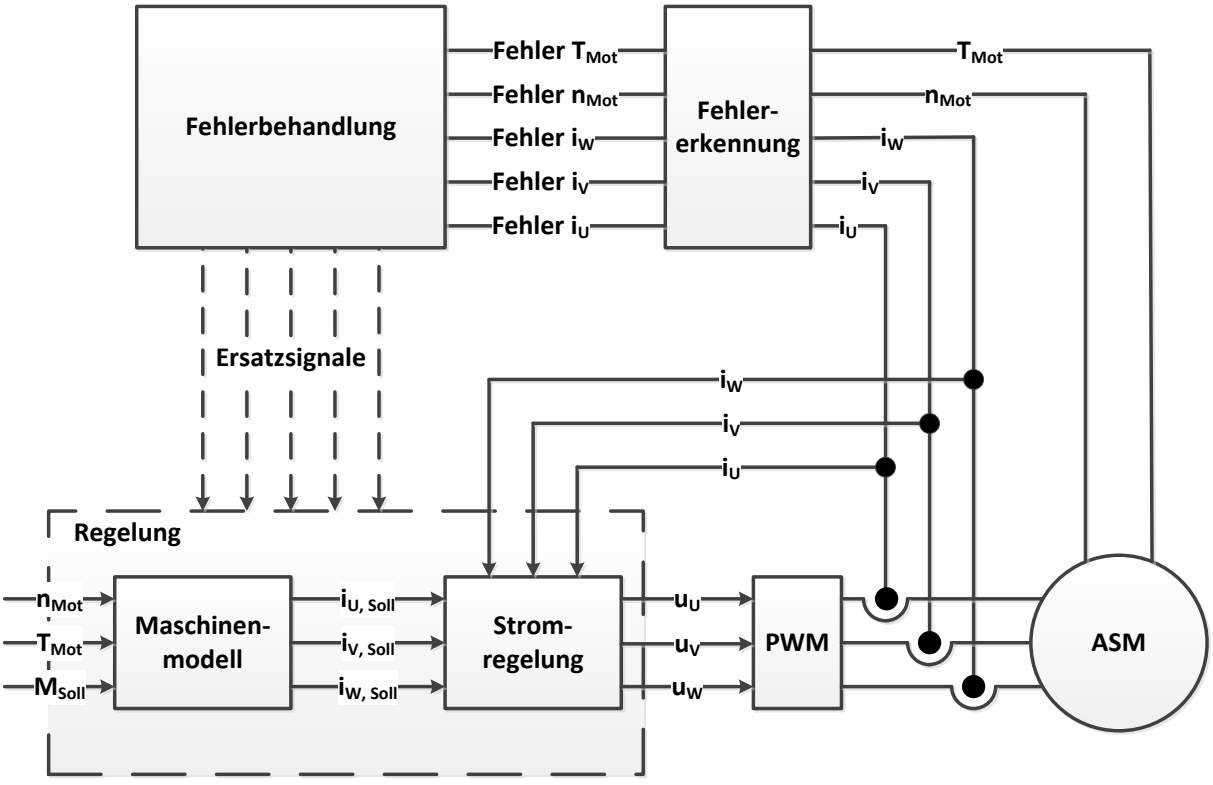


Abbildung 43: Erweiterung der E-Maschinen Regelung

In Abbildung 43 ist eine beispielhafte Erweiterung der bestehenden Regelung aufgezeigt. Mittels einer Fehlererkennung werden fehlerhafte Sensorsignale analysiert und anschließend die entsprechenden Fehlerbehandlungsstrategien aktiviert. Diese generieren Ersatzsignale, die der bestehenden Regelung zugeführt werden.

Es wird bei den Betrachtungen weiterhin ersichtlich, dass keiner der Sensoren gleichwertig durch einen anderen Sensor und eine entsprechende Fehlerbehandlungsstrategie ersetzt werden kann. Es tritt immer eine Verschlechterung in der Dynamik und im maximalen Drehmoment der E-Maschine auf. Garantiert werden kann jedoch der Notlaufbetrieb des elektrischen Antriebsstranges und somit des Fahrzeuges.

7.2 Verifikation des Zusatznutzens

Als Beispiel wird die Erhöhung des Sicherheitsniveaus durch Umgebungsinformationen betrachtet. Zur Ermittlung der Möglichkeiten, die durch den Nutzen der Umfeldsensorik entstehen, soll diese zuerst kurz vorgestellt werden. In Abbildung 44 sind beispielhaft die Bereiche dargestellt, die durch die zur Verfügung stehenden Sensoren abgedeckt werden. Zu nennen sind für den Nahbereich Ultraschallsensoren, für den Fernbereich Radarsysteme, Lidar-/Lasersysteme, Infrarotsysteme und für die optische Erkennung Kamerasysteme. Alle Systeme dienen der Erkennung von Hindernissen, Verkehrshinweisen und anderen Verkehrsteilnehmern im direkten Umfeld des Fahrzeuges. Ein direkter Fehler im elektrischen Antriebsstrang kann somit nicht kompensiert werden, jedoch die entstehenden Folgen können verringert werden, gezeigt im nachfolgenden Beispiel.

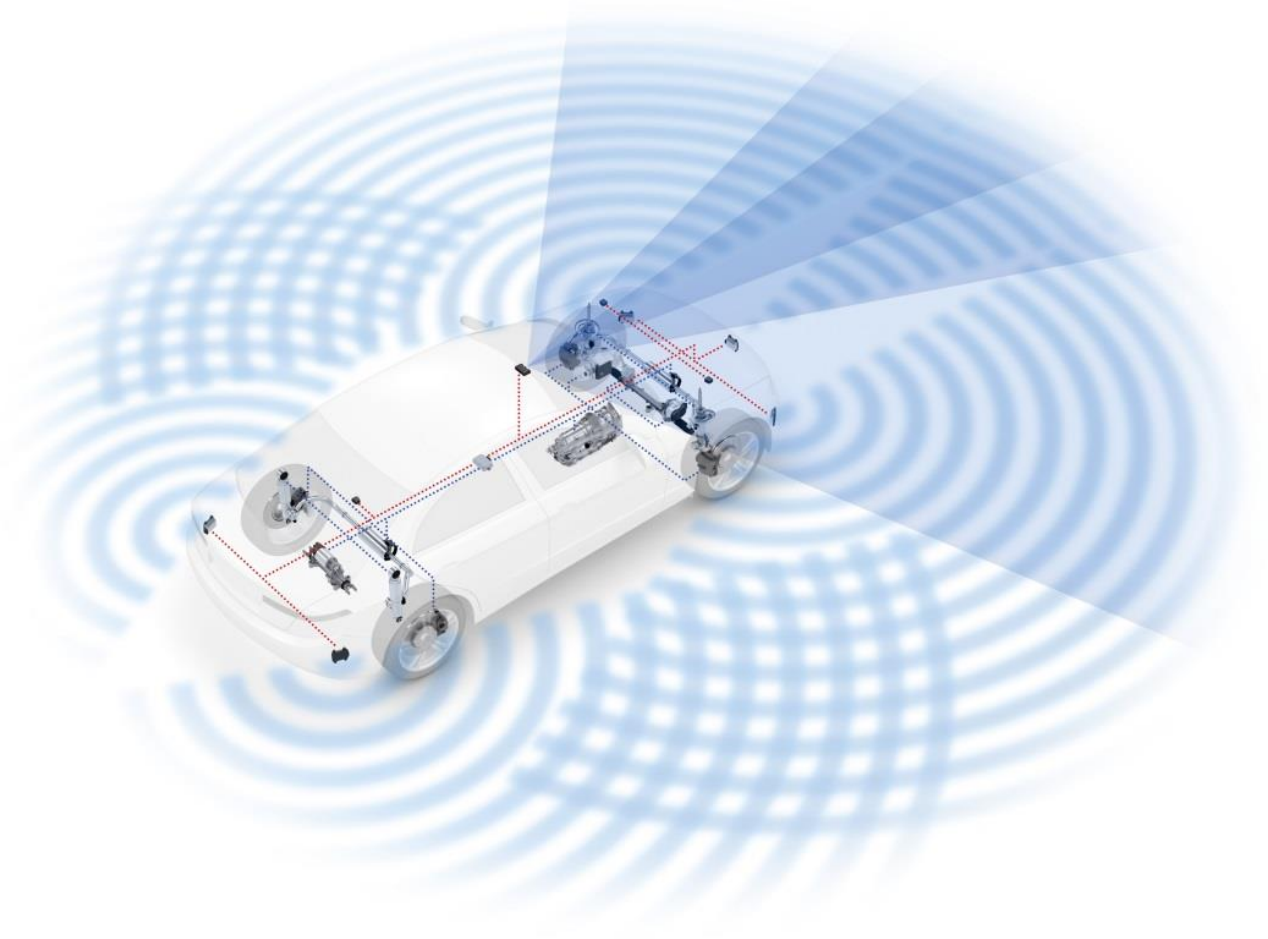


Abbildung 44: Umfeldsensorik im PKW (ZF, 2016)

Es wird die Annahme getroffen, dass ein Sensor fehlerbehaftet ist, oder ausfällt und dies aufgrund der gewählten Fehlerbehandlungsstrategie zu einer Reduzierung der Leistung der E-Maschine führt. Über ein System der Umfeldsensorik, beispielsweise das Radarsystem, wird zusätzlich ein sich schnell näherndes Objekt detektiert. Dies kann ein Lastkraftwagen auf der Autobahn sein, der aufgrund hoher Verkehrsdichte nicht ausweichen kann. Es liegt somit eine kritische Situation vor, die zu einem potentiellen Unfall führen kann. Durch das Erkennen dieser Situation mittels der Umfeldsensorik kann die Reduzierung der Leistung der E-Maschine unterbunden werden, bis eine potentiell sichere Situation vorliegt. Es wird der Schutz der Insassen dabei über den Schutz des elektrischen Antriebsstranges gestellt. Ein zusätzlicher Schaden des elektrischen Antriebsstranges kann nicht ausgeschlossen werden.

7.3 Prüfungen mittels Simulation früh in der Systementwicklung

Die verwendete Methodik schafft eine wichtige Grundlage um Prüfungen mittels Simulation früh in der Systementwicklung durchzuführen.

Die sicherheitskritischen Funktionen eines Fahrzeugs (PHV, EV) können damit in einer Architekturbeschreibungssprache modelliert und die Systemauswirkungen von Fehlern sowie die Wirksamkeit von Maßnahmen zur Erhöhung der Fehlertoleranz früh im Entwicklungsprozess sichtbar gemacht werden.

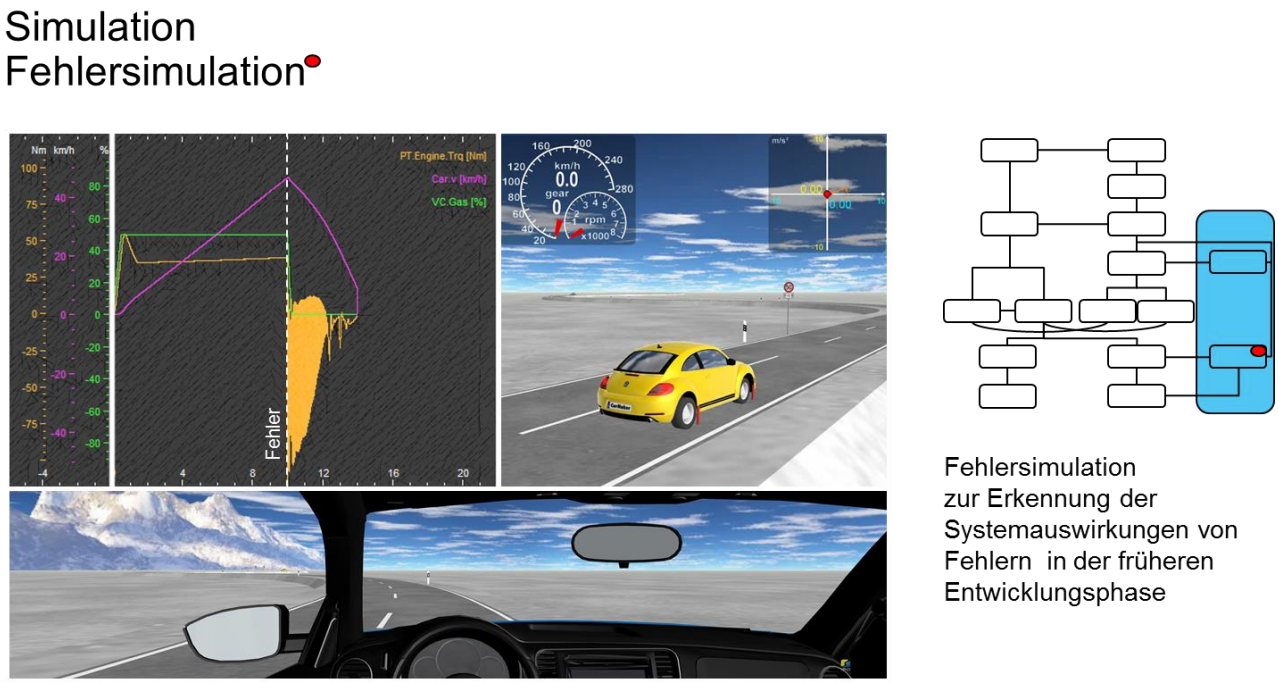


Abbildung 45: Fehlersimulation

Die Anwendung der Methode ermöglicht eine durchgängig modellbasierte und somit effizientere Entwicklung. Damit ist es möglich den Übergang von der Risikoanalyse zu den Sicherheitsanforderungen transparent zu gestalten.

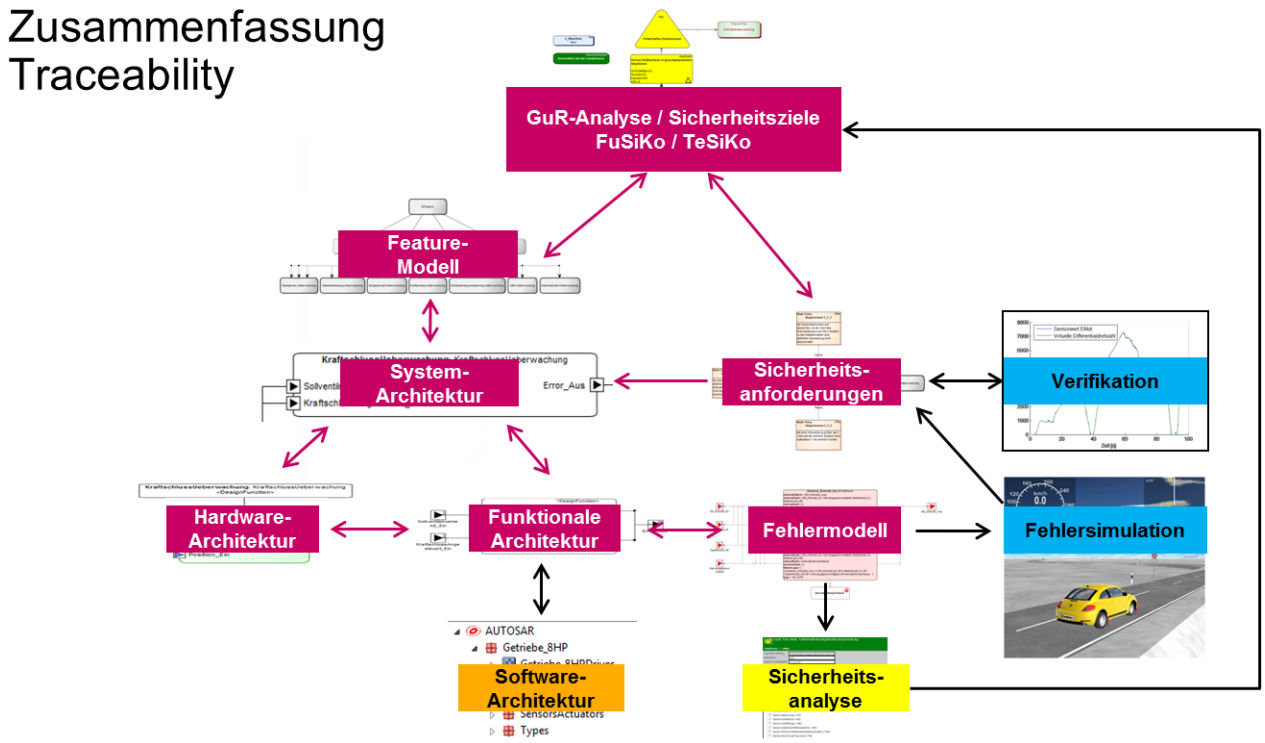


Abbildung 46: Durchgängigkeit (Traceability)

7.4 Umsetzung von Projektergebnissen auf dem Fahrsimulator

Im Rahmen einer Probandenstudie wurden exemplarische Fehlfunktionen des Antriebsstrangs simuliert und daraus Parameter für das Fahrerverhalten in Gefahrensituationen abgeleitet. Ziel war eine bessere Parametrisierung des Fahrermodells von CarMaker©. Durch die Simulation war die Situation für alle Probanden reproduzierbar gleich und die Probandenreaktion somit mit der Reaktion des Fahrermodells vergleichbar. Die Ergebnisse der Studie sind in Kapitel 5.2.1 dargestellt.

8 Schlussbetrachtung

Im Rahmen des Projekts ZuSE wurden Grundlagen für fehlertolerante Architekturen im Zusammenhang mit den für ZF interessanten E-Antriebssystemen für PHV und EV erarbeitet und es wurden verschiedene E/E-System-Verteilszenarien unter Einbeziehung der Möglichkeiten von Mehrkernprozessoren untersucht.

Vor dem Hintergrund der Frage welche Architekturen sich für E-Antriebssysteme im Hinblick auf die geforderte Zuverlässigkeit und Fehlertoleranz eignen und wie sie sich unterscheiden wurden verschiedene Architekturen analysiert und simuliert. Dazu wurden bekannte und relevante Ergebnisse aus den Forschungsprojekten SAFE und MotorBrain berücksichtigt.

Speziell über die SAFE-Methode wurde eine Kopplung des sicherheitsbasierten Entwicklungsprozesses mit der Simulation erreicht, was in Abbildung 47 verdeutlicht ist.

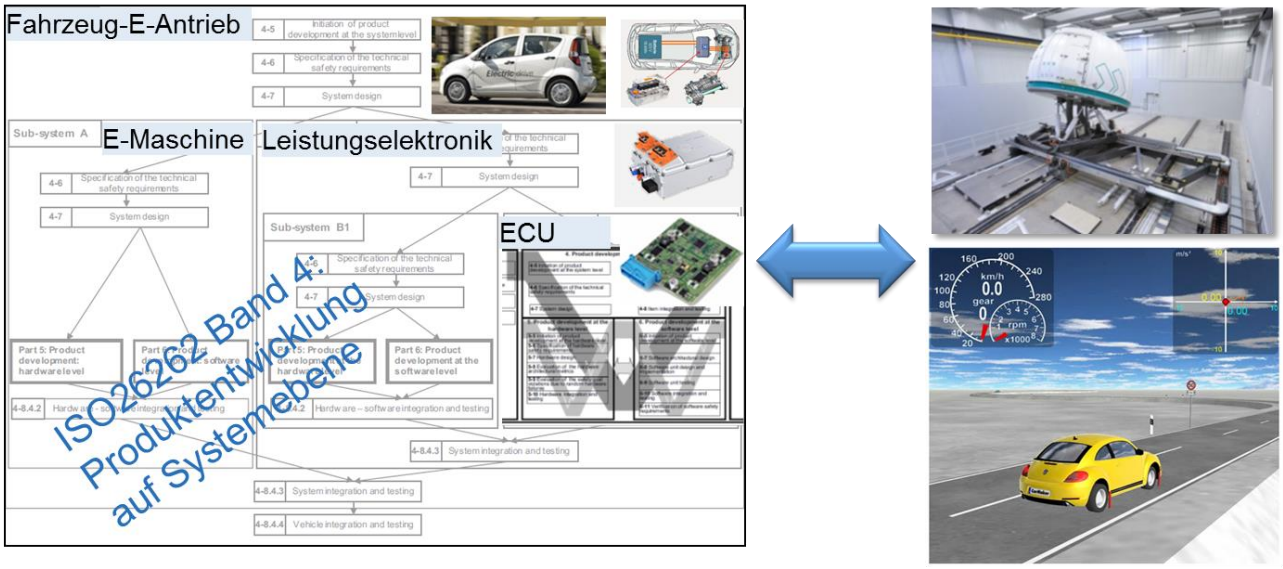


Abbildung 47: Simulation mit Anbindung an die Systementwicklung

Das ursprünglich von der Politik ausgegebene Ziel, bis 2020 in Deutschland eine Million zugelassene Elektrofahrzeuge zu haben, lässt sich bis 2020 wahrscheinlich nicht mehr erreichen. Dennoch nimmt der Anteil an reinen E-Fahrzeugen insbesondere in urbaner Umgebung deutlich erkennbar zu. Bei weltweit führenden Lieferanten wie ZF werden große Anstrengungen unternommen um die E-Mobilität voranzubringen und wichtige Fahrzeughersteller erweitern ihre diesbezüglichen Produktpaletten in teilweise beachtlichem Umfang.

Ebenso nimmt aber auch die Elektrifizierung des konventionellen Antriebs noch weiter zu und es zeichnet sich wie erwartet ab, dass das Fahrzeug mit Hybridantrieb eine nicht

unwesentliche Rolle in einem Übergangsszenario spielen wird. Unter anderem stellt der Betrieb von vielen E-Fahrzeugen große Herausforderungen an die Infrastruktur, die erst noch zu bewältigen sind. Darüber hinaus gibt es ergänzende und teilweise konkurrierende Technologien wie z. B. die Brennstoffzelle oder eine „saubere Verbrennung“, die ebenfalls noch weiter erforscht werden und die allesamt die Mobilitätstechnologien der Zukunft noch beeinflussen können.

Die im Projekt erforschten Methoden tragen somit zur Erreichung der Ziele bei, welche die Bundesregierung zur Umsetzung der Elektromobilität im Inland formuliert hat. Eine Schlüsselrolle nimmt die Anwendung von simulationsgestützten Methoden ein, ohne die eine hinreichende Absicherung von sicherheitsrelevanten Fahrzeugfunktionen und von Überwachungsfunktionen kaum möglich ist. Diesbezüglich wurden im Rahmen der durchgeführten Arbeiten entsprechende exemplarische Umsetzungen und Verifikationen durchgeführt. Einrichtungen der Universität Stuttgart, insbesondere der Fahrsimulator, wurden im Rahmen der Durchführung des Teilvorhabens ebenfalls genutzt.

Zweifellos werden die Elektrifizierung und die Automatisierung in Richtung des autonomen Fahrens weiter voranschreiten. Auch die Digitalisierung und die damit verbundenen Möglichkeiten wie „Softwareupdate on Demand“, werden die Architekturen für zukünftige Antriebssteuerungssysteme beeinflussen. Es muss damit gerechnet werden, dass diese Trends die Anforderungen an Zuverlässigkeit und Fehlertoleranz weiter erhöhen werden. Zusätzlich gibt es Bestrebungen bei den OEMs und den Systemanbietern leistungsstarke Domain-ECUs einzusetzen, um die Steuergerätanzahl im Fahrzeug zu reduzieren. Das wird ebenfalls einen starken Einfluss auf die Architekturen ausüben. Diese Faktoren konnten im Rahmen des ZuSE-Projekts nicht oder nicht ausreichend berücksichtigt werden, sie sind Inhalte zukünftiger Arbeiten zu Fehlertoleranz und Zuverlässigkeit.

Abbildungsverzeichnis

Tabellenverzeichnis

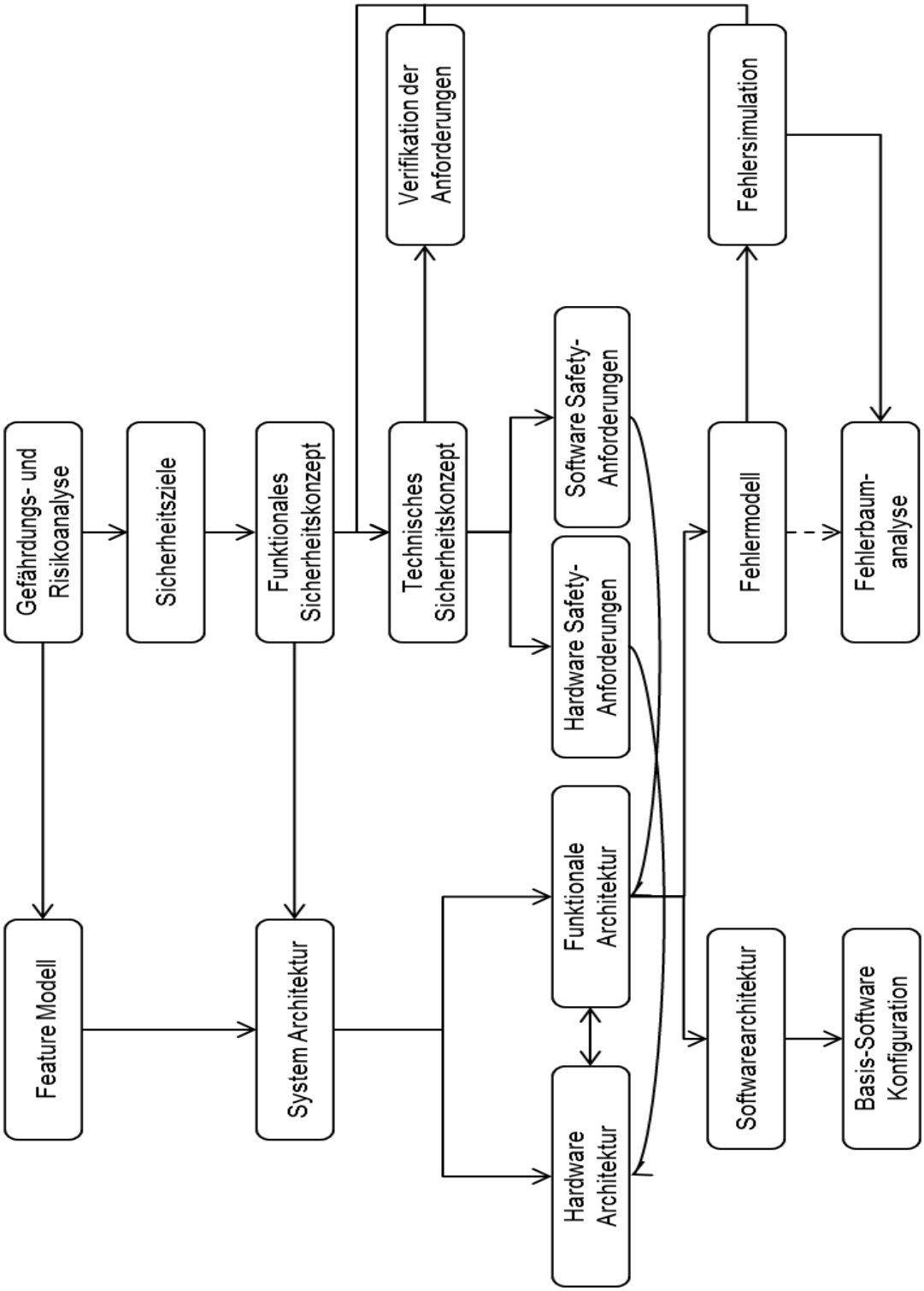
Literaturverzeichnis

Vortrag bei safe.tech 2016: Sari, B.; Gohl, S.; Geiger, R.:
“Ein modellgetriebener Ansatz zur Beherrschung von sicherheitskritischen Funktionen im E-Antrieb”, *safe.tech 2016*

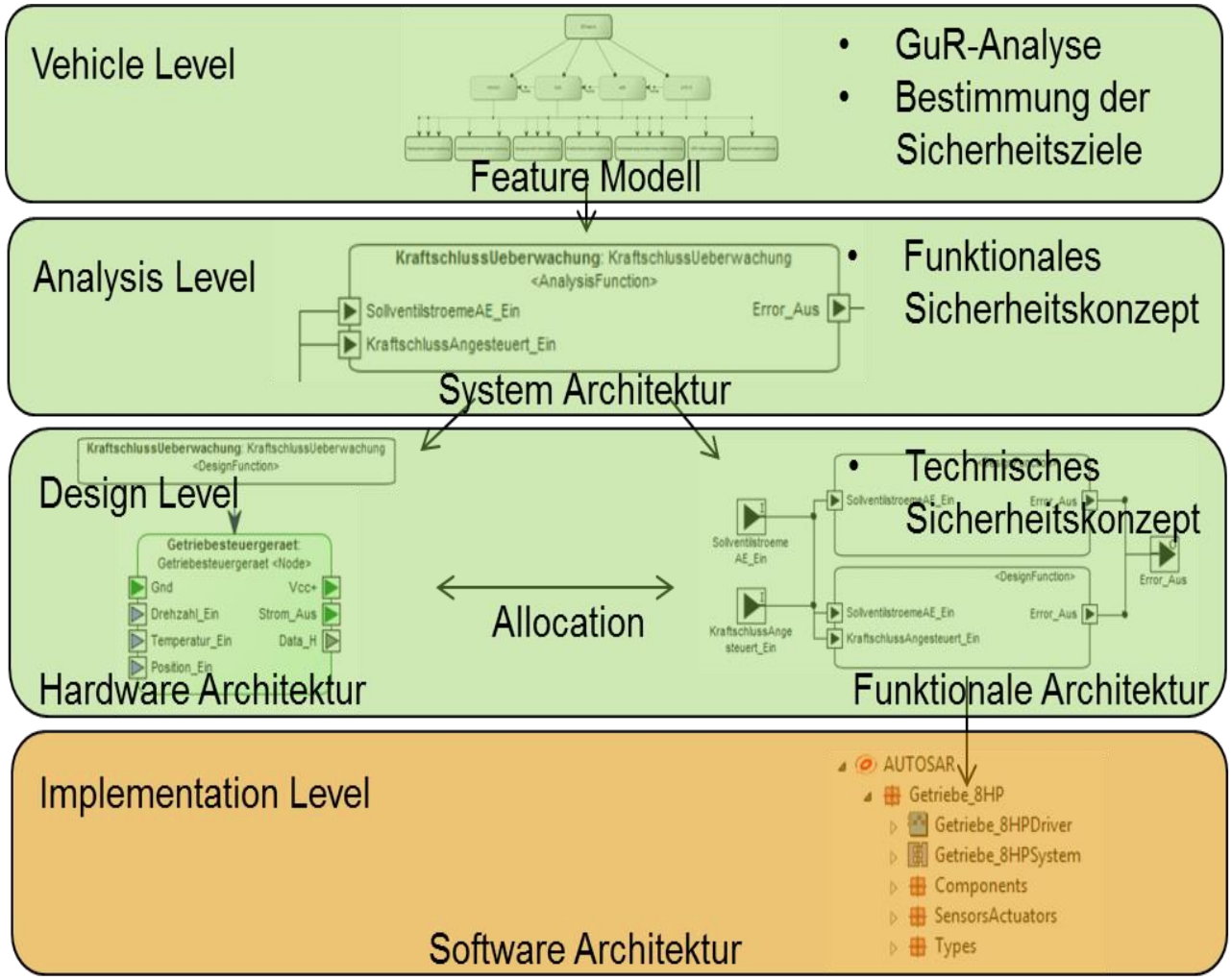
Paper für ESARS-ITEC 2016: Sari, B. and Reuss, H.C.:
“A model-driven approach for the development of safety-critical functions using modified Architecture Description Language (ADL)”, *Electrical Systems for Aircraft, Railway, Ship propulsion and Road Vehicles & International Transportation Electrification Conference (ESARS-ITEC 2016)*.

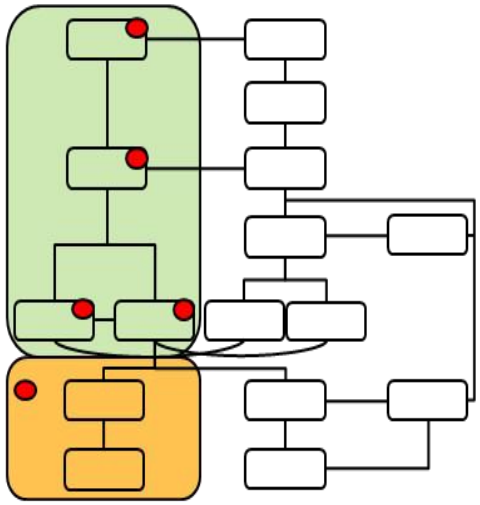
Paper für WCX™17: SAE World Congress Experience (eingereicht, noch im Review):
Sari, B. and Reuss, H.C.:
“A Model-driven Approach for Dependent Failure Analysis in Consideration of Multicore Processors using Modified Architecture Description Language (ADL)”, WCX™ 17: SAE World Congress Experience

Anhang A: Details der angewendeten Methode (SAFE)

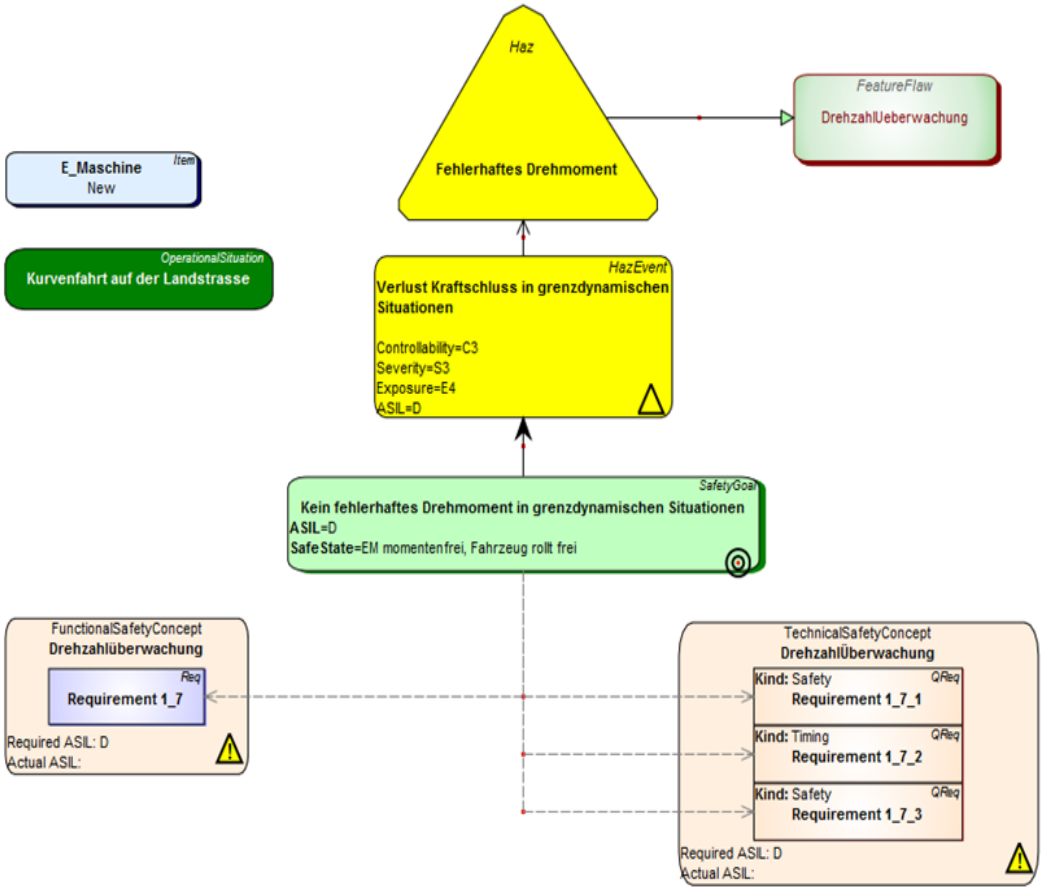


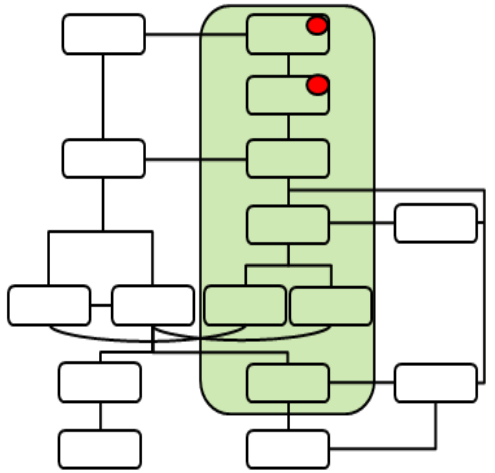
Zur Orientierung markieren in den folgenden Bildern rote Punkte in einer verkleinerten Darstellung die im jeweiligen Bild betrachteten Schritte, eine Erläuterung steht jeweils darunter. Es sei zusätzlich auf Ergebnisse des Forschungsprojekts SAFE verwiesen.



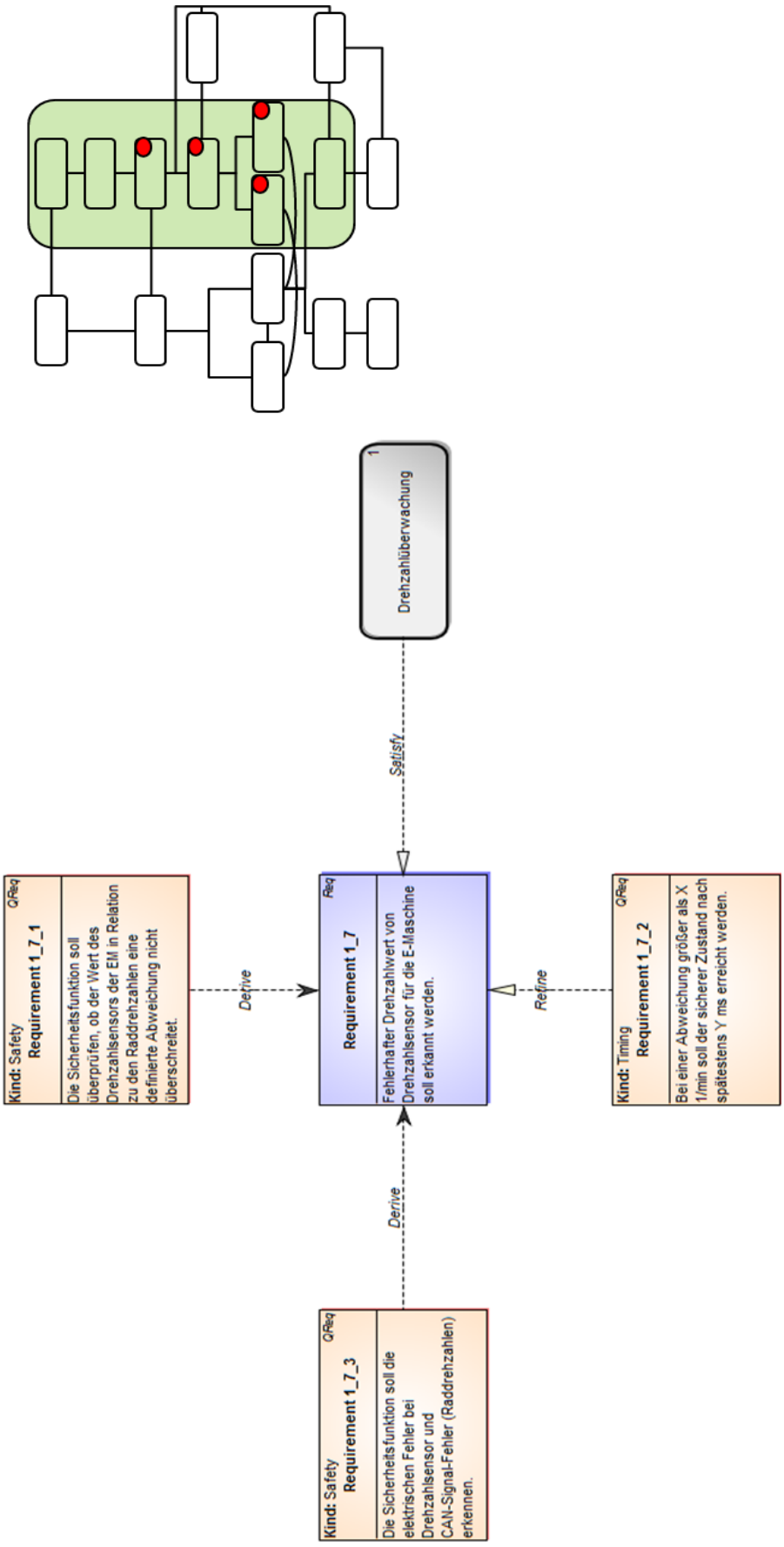


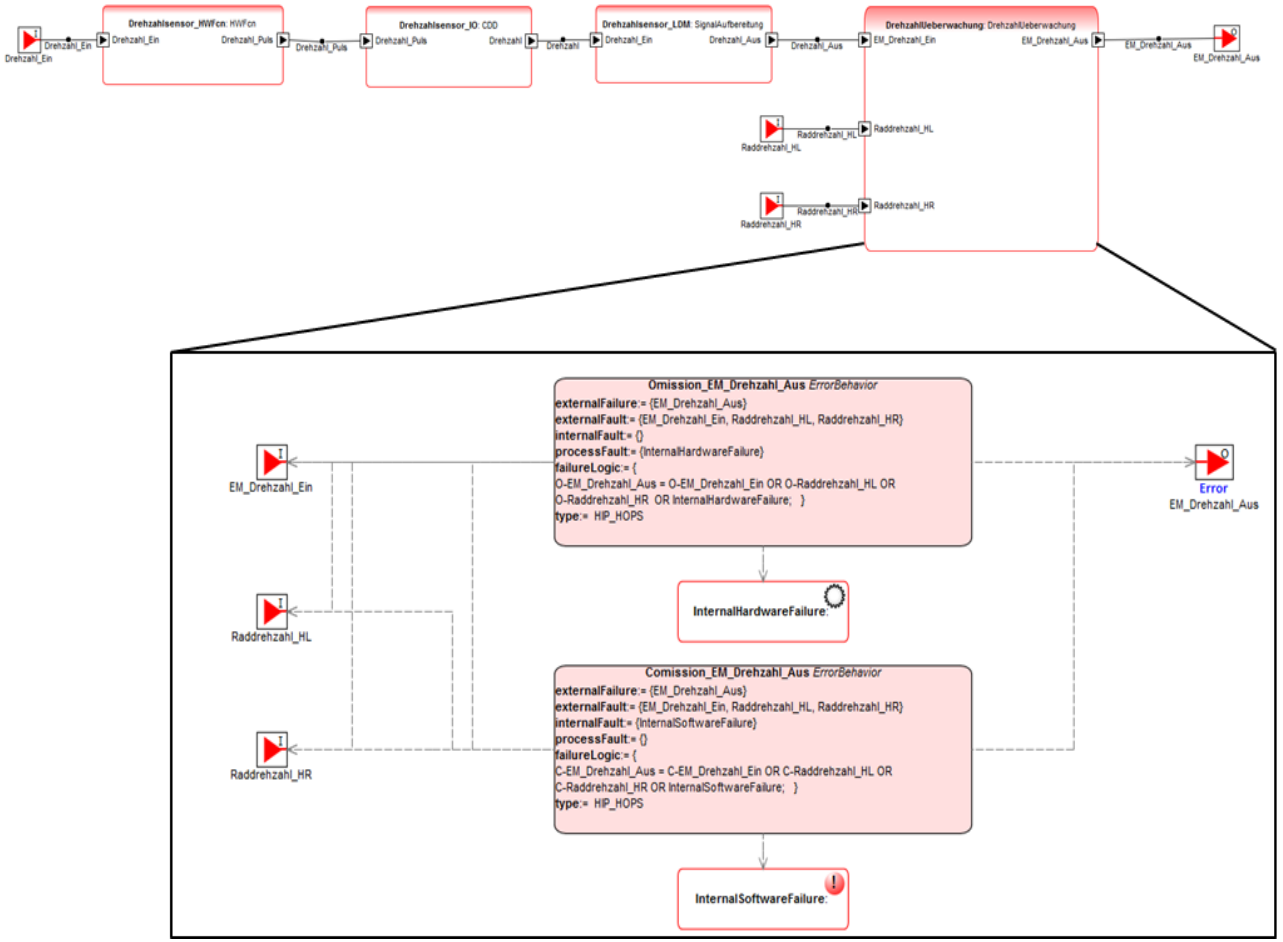
Schrittweise und durchgängige Modellierung von Systemanforderung bis Implementierung

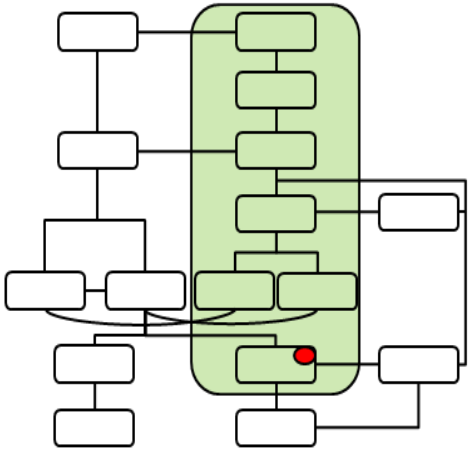




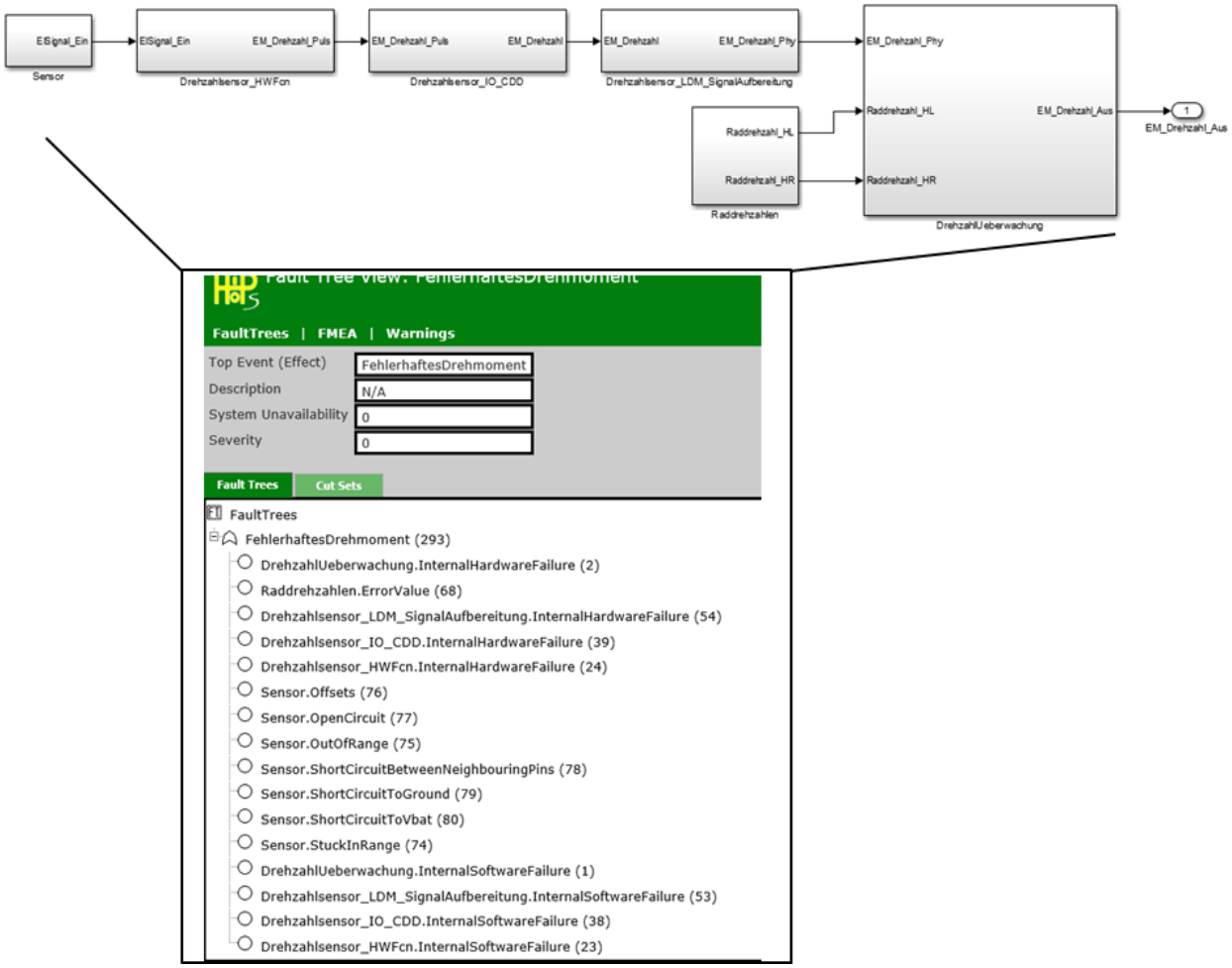
Durchgängige und rückverfolgbare Modellierung von GuR-Analyse bis Sicherheitsanforderungen

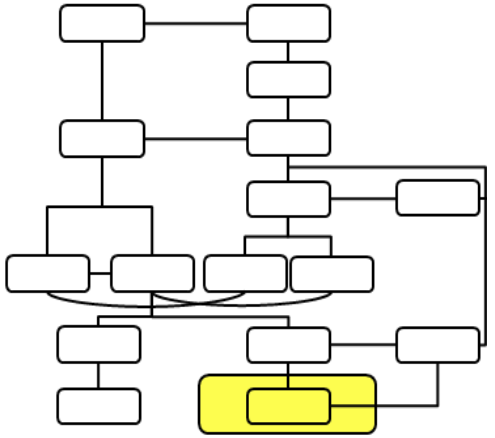

Drehzahlüberwachung
Satisfy
QReq
Kind: Safety
Requirement 1_7_1
Die Sicherheitsfunktion soll überprüfen, ob der Wert des Drehzahlsensors der EM in Relation zu den Raddrehzahlen eine definierte Abweichung nicht überschreitet.
Derive
Req
Requirement 1_7
Fehlerhafter Drehzahlwert von Drehzahlsensor für die E-Maschine soll erkannt werden.
Refine
QReq
Kind: Timing
Requirement 1_7_2
Bei einer Abweichung größer als X 1/min soll der sicherer Zustand nach spätestens Y ms erreicht werden.
Derive
QReq
Kind: Safety
Requirement 1_7_3
Die Sicherheitsfunktion soll die elektrischen Fehler bei Drehzahlsensor und CAN-Signal-Fehler (Raddrehzahlen) erkennen.





Automatisierte Erstellung der Fehlermodelle aus der Funktionalen Architektur





Modellbasierte
Sicherheitsanalyse
von Fehlermodellen

Berichtsblatt

1. ISBN oder ISSN	2. Berichtsart (Schlussbericht oder Veröffentlichung)
Nicht relevant	Schlussbericht

3. Titel
Zuverlässigkeit und Fehlertoleranz von Komponenten des Elektroantriebs

4. Autor(en) [Name(n), Vorname(n)]	5. Abschlussdatum des Vorhabens
Geiger, Roland Gohl, Sven Riemer, Thomas Sari, Bülent	30.6.2016
	6. Veröffentlichungsdatum Nicht relevant
	7. Form der Publikation Nicht relevant
8. Durchführende Institution(en) (Name, Adresse)	9. Ber. Nr. Durchführende Institution
ZF Friedrichshafen AG Graf-von-Soden-Platz 1 88038 Friedrichshafen	Nicht relevant
	10. Förderkennzeichen 16N12547
	11. Seitenzahl 71
12. Fördernde Institution (Name, Adresse)	13. Literaturangaben
Bundesministerium für Bildung und Forschung (BMBF) 53170 Bonn	Keine
	14. Tabellen 8
	15. Abbildungen 47

16. Zusätzliche Angaben
Ergänzender Bericht zum Bericht des IVK der Uni Stuttgart (Prof. Reuss) zum Projekt ZuSE

17. Vorgelegt bei (Titel, Ort, Datum)
TIB Hannover, BMBF Bibliothek Bonn

18. Kurzfassung
Veröffentlichte Untersuchungen über Zuverlässigkeit, Fehlertoleranz und Sicherheit von Fahrzeugen mit Antriebslösungen für Elektro- und Plug-In-Hybrid-Fahrzeuge beschränken sich auf Betrachtungen von Einzelkomponenten oder Einzelfunktionen. Es fehlt an Ansätzen für spezifische Synthese- und Testverfahren auf der Systemebene und es fehlt an effizienten, durchgängigen und mit Simulation unterstützten Entwicklungsmethoden, um die Architektur des Systems und daraus die von Hardware und Software so zu entwerfen dass Anforderungen an die Zielgrößen Zuverlässigkeit, Fehlertoleranz und Sicherheit erfüllt werden. Vor dem Hintergrund der Frage, welche Architekturen sich für E-Antriebssysteme im Hinblick auf die geforderte Zuverlässigkeit und Fehlertoleranz eignen und wie sie sich unterscheiden, wurden System-/Hardware- und Softwarearchitekturen für E-Antriebslösungen unter verschiedenen Aspekten analysiert, bewertet und simuliert. Ein wichtiger Aspekt waren z. B. Multicore-Controller, da diese zeitgleich in ersten Projekten mit hohen Performance-Anforderungen eingeführt wurden. Ergebnis der Untersuchungen sind einige Konzepte die zum Teil erfolgversprechende Grundlagen für eine spätere Umsetzung bieten. Es sind aber weitere Schritte zur Verwendung in einer Serienentwicklung erforderlich. Zusätzlich ist damit zu rechnen, dass schon in Kürze weitere Anforderungen über neue und innovative Zukunftsthemen wie Fahrautomatisierung und Digitalisierung zu berücksichtigen sein werden.

19. Schlagwörter
Fehlertoleranz, Sicherheit, Architektur, Simulation, E-Antrieb, Plug-In-Hybrid, E-Fahrzeug

20. Verlag	21. Preis
Nicht relevant	Nicht relevant

Document Control Sheet

1. ISBN or ISSN Not relevant	2. type of document (e.g. report, publication) Final report
3. title Reliability and fault tolerance of components of the electric drivetrain	
4. author(s) (family name, first name(s)) Geiger, Roland Gohl, Sven Riemer, Thomas Sari, Bülent	5. end of project 30.6.2016
	6. publication date Not relevant
	7. form of publication Not relevant
8. performing organization(s) (name, address) ZF Friedrichshafen AG Graf-von-Soden-Platz 1 88038 Friedrichshafen	9. originator's report no. Not relevant
	10. reference no. 16N12547
	11. no. of pages 71
12. sponsoring agency (name, address) Bundesministerium für Bildung und Forschung (BMBF) 53170 Bonn	13. no. of references None
	14. no. of tables 8
	15. no. of figures 47
16. supplementary notes Supplementing report to the report of University of Stuttgart (IVK, Prof. Reuss) for Project ZuSE	
17. presented at (title, place, date) TIB Hannover, BMBF Bibliothek Bonn	

18. abstract

Publicly available examinations of reliability, fault tolerance and safety of drivetrains for electric and plug-in hybrid vehicles are limited to single components or single functions.

There is a lack of seamless and simulation supported methods on system level, allowing system, hardware and software design in a way that specific requirements on reliability, fault tolerance and safety are considered.

To find an answer for the question which architectures for electric drivetrains controls are suitable to meet the required level of reliability and fault tolerance and what the differences are, certain system-, hardware and software architectures were analyzed, evaluated and simulated.

As example, one important hardware aspect was to consider multicore controller architectures since multicore controllers entered development for systems with high performance demands at the same time.

Main results of the work are several architecture concepts. They partly provide promising basics for future applications, but further investigations for series application will be necessary.

Furthermore it is assumed that additional requirements, derived from new and innovative topics like autonomous driving and digitalization, soon have to be considered as well.

19. keywords

Fault tolerance, Safety, Architecture, Simulation, Electric Drive, Plug-In-Hybrid

20. publisher Not relevant	21. price Not relevant