

INMOVE – Entwicklung eines autarken parallelen Hybridfahrzeugs

Dipl.-Ing. Christian Amsel

Dipl.-Ing. Christian Renner

Dipl.-Ing. Ralf Bady

Institut für Kraftfahrwesen Aachen (ika), RWTH Aachen

Forschungsgesellschaft Kraftfahrwesen mbH Aachen (fka)

INHALTSVERZEICHNIS

- 1 Einleitung
- 2 Architektur des parallelen Hybridfahrzeugs INMOVE
- 3 1. Prototyp
 - 3.1 Offline Simulation mit Matlab/Simulink/Stateflow
 - 3.2 Inbetriebnahme RCP-System auf einem dynamischen Rollenprüfstand
 - 3.3 Betriebsstrategie
 - 3.4 Gang- und Schaltwunscherkennung
 - 3.5 Versuche auf dem Prüfstand
- 4 2. Prototyp
- 5 Ausblick

VDI-Tagung “Innovative Fahrzeugantriebe“

26./27. Oktober 2000, Dresden

INMOVE – Entwicklung eines autarken parallelen Hybridfahrzeugs

INMOVE – Development of an Autark Parallel Hybrid Vehicle

Dipl.-Ing. Christian **Amsel**, Dipl.-Ing. Christian **Renner**, Dipl.-Ing. Ralf **Bady**, Aachen

Zusammenfassung

Im Rahmen des Forschungs- und Entwicklungsprojekts INMOVE wird ein Hybridfahrzeug entwickelt und realisiert. Die Hauptziele dieses Projekts bestehen in der Definition eines Antriebssystems, der Optimierung von Antriebsstrangtechnologien sowie dem Aufbau zweier Demonstrationsfahrzeuge auf Basis des Citroen Berlingo. Zum Betrieb des Fahrzeugs muss eine geeignete Betriebsstrategie entwickelt und erprobt werden. Die parallele Antriebsstruktur wurde ausgewählt, um einen niedrigen Verbrauch mit einer kostengünstigen Struktur zu realisieren. Der Antriebsstrang stellt einen Einwellenparallelhybrid dar, bei dem der Elektromotor auf die Getriebeeingangswelle des manuellen Getriebes wirkt. Die Kupplung ist durch ein elektronisches Kupplungssystem automatisiert. Die Antriebssteuerung und insbesondere die Betriebsstrategie werden durch einen Hybridfahrzeugcontroller realisiert. Im Zuge der Entwicklung der Software für den Hybridfahrzeugcontroller wurden verschiedene Autocodetools eingesetzt. Im ersten Schritt kamen RCP-Tools zum Einsatz mit deren Hilfe die simulierte Betriebsstrategie auf ein RCP-System automatisch implementiert wurde. Im zweiten Schritt kam die automatische Codegenerierung zum Einsatz, um Teilfunktionen der Betriebsstrategie in C-Code umzusetzen.

Summary

The R&D-project INMOVE deals with the development and realization of a hybrid drive system. The main objective of the project is the definition of such a powertrain, the research on optimized technology and finally the prototyping of two demonstrators (Citroen Berlingo), in order to design and evaluate an appropriate hybrid driving strategy. To achieve a good fuel economy with a cost effective solution a parallel hybrid drive has been developed. The drive train is of the "single shaft" configuration, where the electric motor works on the input shaft of the manual gearbox. The clutch is electronically controlled and automated. The overall control, the drive system management and especially the driving strategy are realized by a hybrid vehicle controller. During the development process of the software for the hybrid vehicle controller autocode tools could be used successfully in different steps. In a first step, RCP-tools were used to transfer strategies from the simulation of the complete vehicle. In a second step automatic code generation has been used to implement several sub functions of the operating strategy.

1 Einleitung

Im Rahmen des Projekts INMOVE (Integrated Modular Electric Propulsion System for Parallel Hybrid Vehicles), gefördert durch die Europäische Gemeinschaft im Programm Brite_EURAM III, wird ein paralleler hybrider Antriebsstrang entwickelt und in Prototypenfahrzeugen aufgebaut und getestet.

Auf Basis der grundlegenden Anforderungen

- Fahrleistungen (Beschleunigung, Steigfähigkeit, Höchstgeschwindigkeit)
- Geringe Umweltbeeinflussung (niedrige Emissionen/Verbrauch)
- geringe Systemkomplexität und damit kostengünstige Lösung mit wenig Modifikationen am konventionellen Basisantrieb

wurde das Konzept eines parallelen Hybridfahrzeugs entwickelt, bekannt als „Einwellen-Hybrid“, bei dem das Moment einer Elektromaschine auf die Eingangswelle des Getriebes wirkt. Als Besonderheit ist die Elektromaschine an dem motorabgewandten Teil der Getriebeeingangswelle angeflanscht, so dass eine einfache Integration des Elektromotors möglich ist.

Gegenstand dieses Beitrags ist die Entwicklung und Umsetzung einer Betriebsstrategie für das Fahrzeug, die den autarken, das heißt von einer Nachladung am Netz unabhängigen, Betrieb ermöglicht. Hierzu ist die Elektromaschine in geeigneten Betriebsituationen als Generator zu betreiben, um eine Ladung der Traktionsbatterie zu erzielen.

Die Vorhersage von Fahrleistungen und Verbrauch sowie die Entwicklung der Strategie erfolgte in der Konzeptionsphase mit Hilfe eines komplexen längsdynamischen Simulationsmodells, welches auch die Betrachtung von Detailproblemen, wie zum Beispiel die aktive Synchronisierung der Getriebeeingangswelle durch den Elektromotor während des Schaltens, ermöglicht. Realisiert wurde dieses Modell unter MATLAB/SIMULINK.

Im nächsten Schritt wurde diese Strategie am realen Fahrzeug mittels einer Rapid Control Prototyping (RCP) Hardware getestet, wobei die in der Simulation entwickelten Software Module direkt angewendet werden konnten.

Im letzten Schritt erfolgte die Implementierung der so entwickelten und bereits teilweise ausgetesteten Algorithmen auf den Steuerrechner des Fahrzeugs. Zur Zeit werden die Möglichkeiten und Chancen der Autocode-Generierung im Vergleich zur manuellen Programmierung untersucht.

Vorgestellt werden Simulationsergebnisse im Vergleich zu ersten Messungen am realen Fahrzeug.

2 Architektur des parallelen Hybridfahrzeugs INMOVE

In der Vergangenheit wurden bereits eine Vielzahl von Hybridfahrzeugvarianten untersucht und realisiert. Obwohl noch keine einheitliche Meinung darüber besteht, welche Variante hinsichtlich einer frühen Markteinführung am geeignetsten erscheint, zeichnet sich die Architektur des Parallelhybrids als wahrscheinlichste Variante ab.

Vorteilhaft beim parallelen Hybridfahrzeug gegenüber anderen Strukturen ist der vergleichbar geringere Bauaufwand, teilweise weniger Gewicht und niedrigere Kosten, sowie hohe Wirkungsgrade aufgrund der direkten mechanischen Ankopplung. /1/

Bei bisherigen Lösungen wurde der Elektromotor als zusätzliche Komponente zwischen Verbrennungsmotor und Getriebeeingang hinzugefügt /2/. Diese Anordnung offeriert ein Potenzial hinsichtlich einer platzoptimierten Integration von Kupplung und Elektromotor. Auf diese Weise konnte die des öfteren kritische Länge des gesamten Antriebsstrangs optimiert werden. Andererseits resultiert dieser Anordnung jedoch in aufwendigen Konstruktionsänderungen sowohl am Verbrennungsmotor als auch am Getriebe.

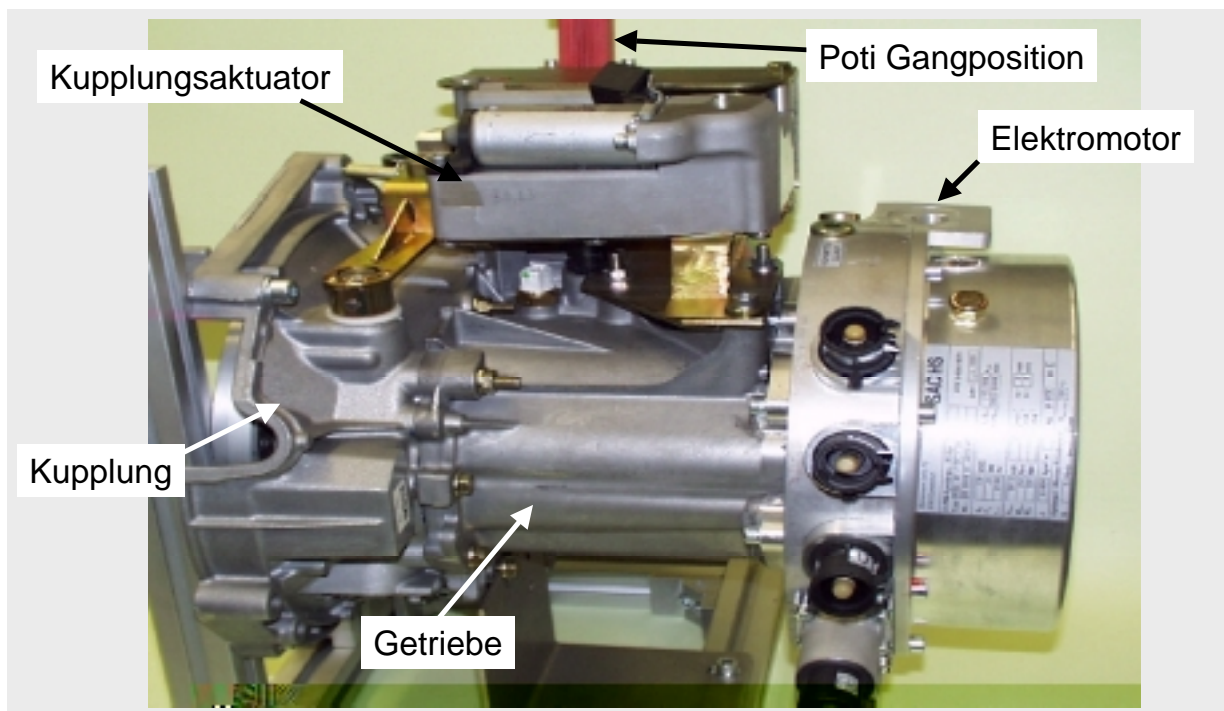


Abb. 2-1: Antriebsstrangstruktur INMOVE

Für das Projekt INMOVE wurde eine alternative Lösung ausgewählt, um die Konstruktions- und Prototypenherstellungskosten zu reduzieren. Der realisierte Antriebsstrang ist in Abb. 2-1 gezeigt. Im Bild links ist das Getriebe mit Kupplungsaktuator und dem Potentiometer zur Erfassung der Gangposition dargestellt. Rechts an das Getriebe angeflanscht befindet sich der Elektromotor. Der Elektromotor ist an die Stelle der normalerweise für den fünften Gang

vorhandene Erweiterungseinheit montiert und ist fest mit der Getriebeeingangswelle verbunden. Durch diese Anordnung werden die Gangstufen auch im elektrischen Betrieb nutzbar. Die fixe Anbindung des Elektromotors an die Getriebeeingangswelle erfordert beim Schalten eine aktive Unterstützung der mechanischen Synchronisierung, da das im Vergleich zu einem normalen Schaltgetriebe zusätzliche Massenträgheitsmoment des Elektromotors entsprechend abzubremesen oder zu beschleunigen ist.

Der Einbau im Fahrzeug und das Fahrzeug selbst sind in Abb. 2-2 und Abb. 2-3 gezeigt.



Abb. 2-2: Versuchsträger

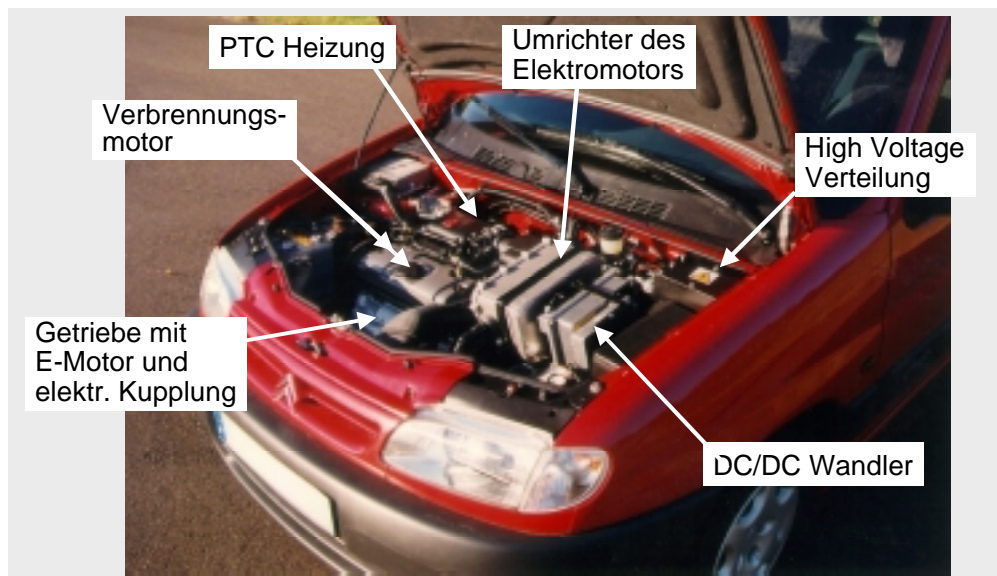


Abb. 2-3: Motorraum

Im einzelnen seien die wesentlichen Fahrzeugdaten aufgeführt:

Leergewicht: 1380 kg
 Beschleunigung: 0-100 km/h 14 s
 Höchstgesch.: 150 km/h
 Antrieb: ICE 1.4 l Otto Motor, 55kW @5300 1/min, 112 Nm @2800 1/min
 permanent-erregter Synchronmotor 25/30 kW (nom./Spitze)
 4-Gang Getriebe, automatisierte Kupplung

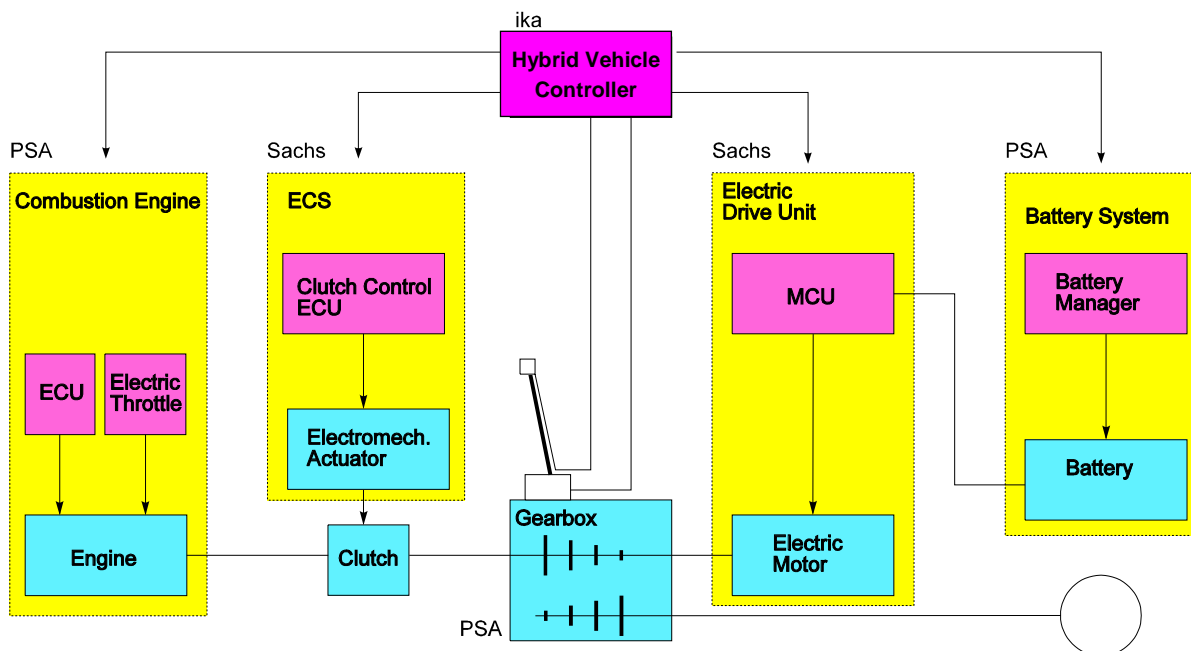


Abb. 2-4: Aufbau und Systemvernetzung INMOVE

In Abb. 2-4 ist der Aufbau und die Systemvernetzung des Hybridfahrzeugs dargestellt. Die zentrale Fahrzeugsteuerung (Hybrid Vehicle Controller) stellt die übergeordnete Steuerungseinheit dar und beinhaltet die Betriebsstrategien. Hierzu erfasst sie die Fahrerwünsche, stellt interne Zustände des Fahrzeugs auf einem Display geeignet dar und steuert alle anderen Systeme über CAN-Bus an. In Abb. 2-5 ist der Hybrid Vehicle Controller dargestellt. Der mit einem E-Gas ausgestattete Verbrennungsmotor erhält von dem Hybrid Vehicle Controller Drehzahl- oder Drehmomentenanforderungen. Das Kupplungssystem (ECS) erhält von dem Hybrid Vehicle Controller die Ganginformation und Drehzahlen (Verbrennungsmotordrehzahl und E-Motordrehzahl) und öffnet und schließt nach der im ECS implementierten Strategie die Kupplung. Die NiCd-Batterie ist mit dem zugehörigen Batteriemanagementsystem (BMS) ebenfalls über CAN mit den übrigen Steuersystemen vernetzt und sendet wesentliche Batteriekenndaten. Der Elektromotor mit Umrichter und Steuergerät (MCU - Motor Control Unit) stellt einen leistungsstarken permanent-erregten Synchronmotor in Außenläufer Bauart dar. Die MCU erhält ebenfalls die relevanten Steuerinformationen (Momenten- und Drehzahlanforderung) über CAN.

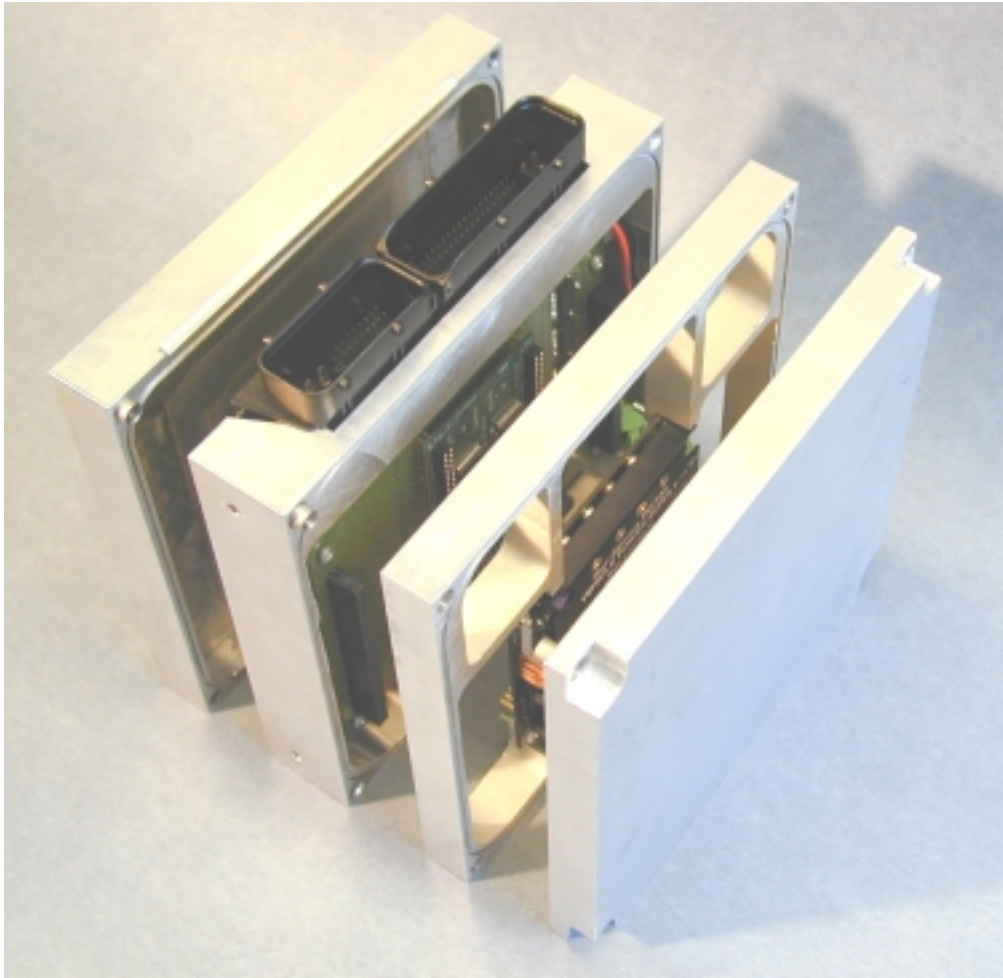


Abb. 2-5: Zentraler Fahrzeugsteuerrechner Smart (ika/fka)

Der Hybrid Vehicle Controller stellt ein modulares Steuerrechnersystem dar, das speziell für die kostengünstige und schnelle Realisierung von Prototypen und Kleinserienprojekten entwickelt wurde. Der Steuerrechner ist in einem leichten Aluminiumgehäuse untergebracht und besteht aus Netzteil, Recheneinheiten sowie I/O-Funktionalitäten. Für das Rechnersystem wurde ein aus der Luftfahrt erprobtes 40W PC/104-Netzteil mit einem Eingangsspannungsbereich von 6..40 V ausgewählt und angepasst. Die Recheneinheit besteht aus dem Mikrocontroller C167 von Infineon mit einem maximalen Speicherausbau von jeweils 2 MB RAM- und Flashspeicher.

Ein Sicherheitsrechner ist über einen internen Bus mit dem C167er verbunden und überwacht dessen Handeln. Im Fehlerfall übernimmt der Sicherheitsrechner die Ansteuerung der I/Os und überführt das Hybridfahrzeug schrittweise kontrolliert in einen sicheren Zustand. Für den Fall, dass der Hauptrechner wieder zur Verfügung steht, kann kontrolliert wieder an ihn übergeben werden. Ein sicherer Betrieb ist somit immer gewährleistet. Eingeschränkte Fahrfunktionen können garantiert werden, um beispielsweise eine Gefahrenstelle verlassen zu können.

Die I/O-Funktionalitäten (Grundausbau: 32 Digital IN, 16 Analog IN, 32 Digital OUT, 4 PWM Out, 2 CAN, 2 RS232) sind über modernste Halbleiter realisiert und erfüllen höchste Automotive-Anforderungen. Die Erweiterbarkeit ist aufgrund des modularen Hardwarekonzepts sehr gut möglich. Neben der erwähnten I/O-Funktionalitäten besteht über ein vorhandenes PC/104-Businterface (IEEE P996.1 Standard for Compact Embedded PC-Modules) die Möglichkeit auf dem Markt verfügbare Module, z.B. GPS- oder GSM-Module, Speichermedien usw. zu integrieren.

3 1.Prototyp

Die Entwicklung bis zur Implementierung der Betriebsstrategie auf den Fahrzeugsteuerrechner (Hybrid Vehicle Controller) erfolgte in drei Schritten:

- 1. Schritt: Offline Simulation mit Matlab/Simulink/Stateflow
- 2. Schritt: Rapid Control Prototyping
- 3. Schritt: Implementierung auf Hybrid Vehicle Controller

Abb. 3-1: Entwicklungsschritte

Im Rahmen der offline Simulation mit Matlab/Simulink/Stateflow wurde unter Vernachlässigung von Extrem- und Fehlerfällen die grundsätzliche Betriebsstrategie entwickelt. Hierdurch konnten verschiedene Strategien erprobt und schnell modifiziert werden.

Im 2. Schritt konnte durch die Methode des Rapid Control Prototyping (RCP) die offline entwickelte Betriebsstrategie am realen Fahrzeug eingesetzt werden. Eine geeignete Rapid Control Prototyping (RCP) Plattform ersetzt das Steuergerät vollständig und simuliert dieses in Echtzeit. In Abb. 3-2 ist der schematische Aufbau einer solchen Plattform dargestellt. Sie zeichnet sich durch eine hohe Modularität der Rechnerkomponenten sowie der I/O-Hardware aus, wodurch sowohl eine hohe Echtzeit als auch eine ausreichende Anzahl von Schnittstellen durch die Möglichkeit der Erweiterbarkeit gegeben ist.

Die Kommunikation zwischen simuliertem Hybrid Vehicle Controller und realem Restfahrzeug erfolgt mit Hilfe des Real-Time Interfaces und der Bediensoftware Control Desk der Fa. dSpace (Abb. 3-3). Echtzeit-C-Code wird durch die Rapid Control Prototyping Plattform generiert und automatisch auf die Echtzeithardware von dSpace geladen. In einem Closed-loop Lauf können nun Parameter, Kennfelder usw. online geändert werden, wodurch bereits in diesem frühen Stadium eine **Vorkalibrierung des Systems** erfolgt.

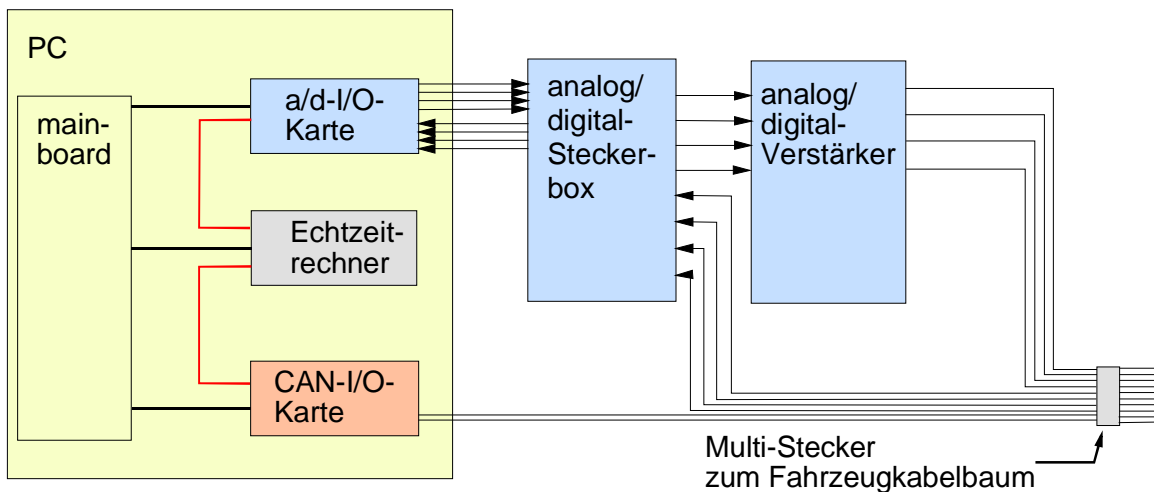


Abb. 3-2: Aufbau Rapid Control Prototyping Plattform

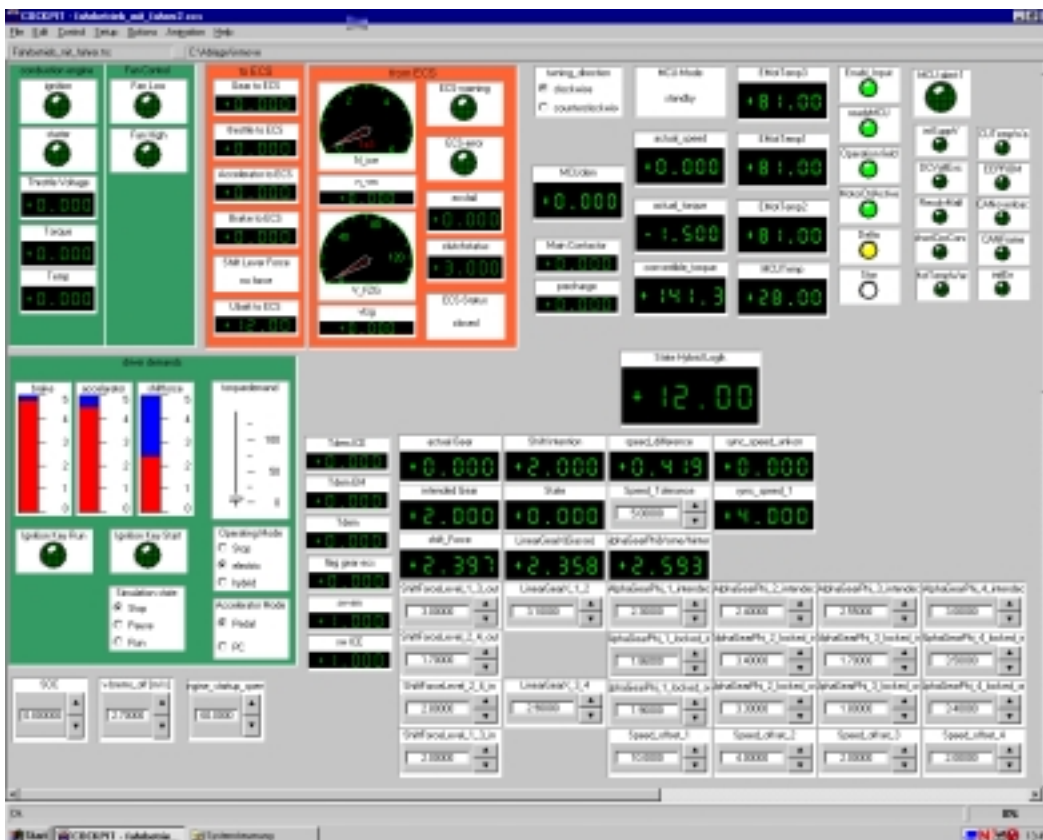


Abb. 3-3: Bedienoberfläche für RCP Plattform

Im letzten Schritt erfolgte die Implementierung auf einen geeigneten Fahrzeugsteuerrechner mit Mikrocontroller (siehe Abb. 2-5), den Hybrid Vehicle Controller, für einen dauerhaften Einsatz im Fahrzeug. Die Codierung erfolgte manuell in C-Code auf Basis des OSEK Betriebssystems.

3.1 Offline Simulation mit Matlab/Simulink/Stateflow

Die Vorhersage von Fahrleistungen und Verbrauch sowie die Entwicklung der Betriebsstrategie erfolgte in der Konzeptionsphase, noch vor der Verfügbarkeit des realen Fahrzeugs oder des Hybrid Vehicle Controllers mit Hilfe eines komplexen längsdynamischen Simulationsmodells. Dieses erlaubt auch die Betrachtung von Detailproblemen, wie zum Beispiel die aktive Synchronisierung der Getriebeeingangswelle durch den Elektromotor während des Schaltens. Realisiert wurde dieses Modell unter MATLAB/SIMULINK/STATEFLOW.

In Abb. 3-4 ist das Simulationsmodell gezeigt. Dargestellt ist die oberste Ebene des Modells. Man erkennt die Struktur des Antriebstrangs sowie den übergeordneten Baustein "Hybrid Vehicle Controller", in dem die Betriebsstrategie und Ansteuerung der Sub-Systeme abgelegt ist.

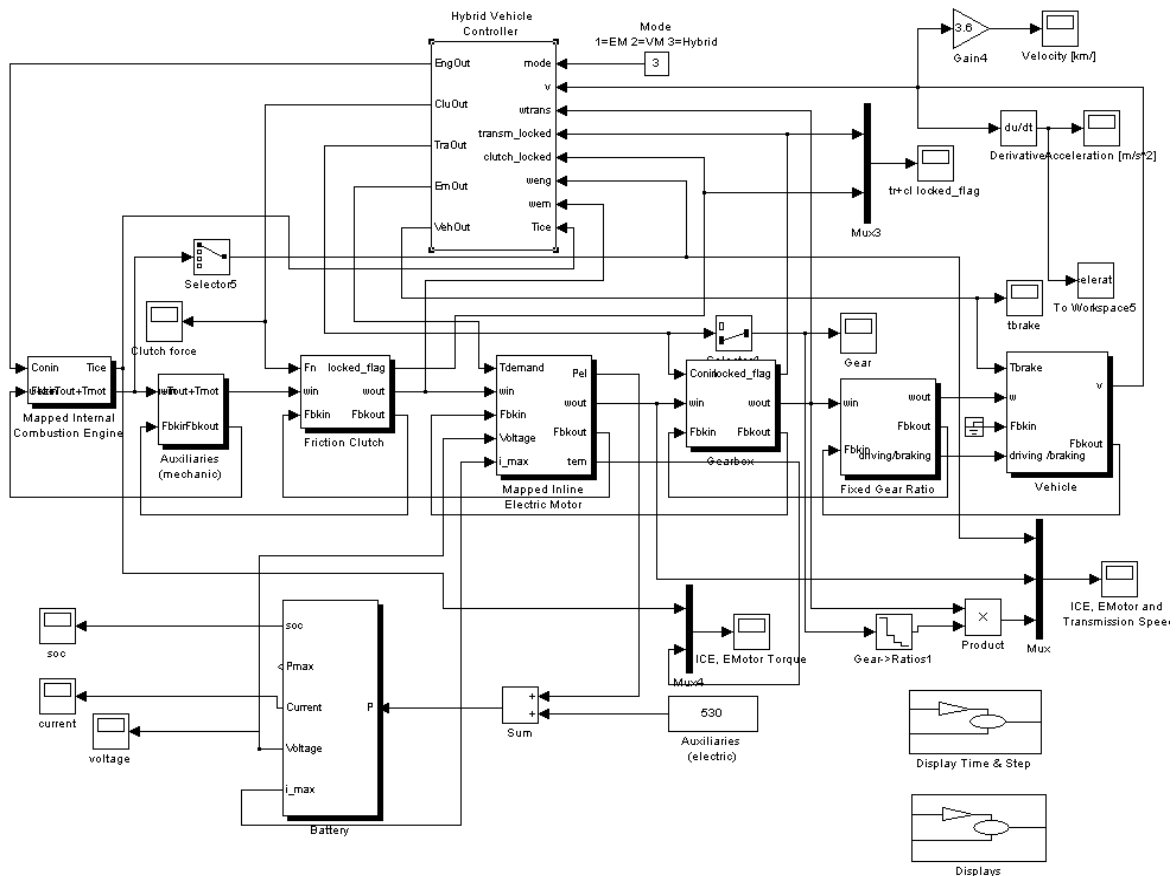


Abb. 3-4: Simulationsmodell in Matlab/Simulink

Ein Ergebnis eines Simulationslaufes ist in Abb. 3-5 gezeigt. Simuliert wurde das Rückschalten vom zweiten in den ersten Gang. Hierbei muss der Elektromotor konzeptbedingt die Getriebeeingangswelle aktiv synchronisieren. Die Elektromaschine kann

dabei mit dem verfügbaren Moment den Drehzahlssprung von ca. 2800 1/min auf 5200 1/min in ca. 0.4 Sekunden bewältigen. In der Realität addieren sich zu dieser Zeit noch der Zeitbedarf zum Erkennen eines Schaltwunsches und die Zeiträume, in denen der Fahrer den Schalthebel vom ursprünglichen Gang in den Leerlauf und aus dem Leerlauf heraus in den Zielgang bewegt.

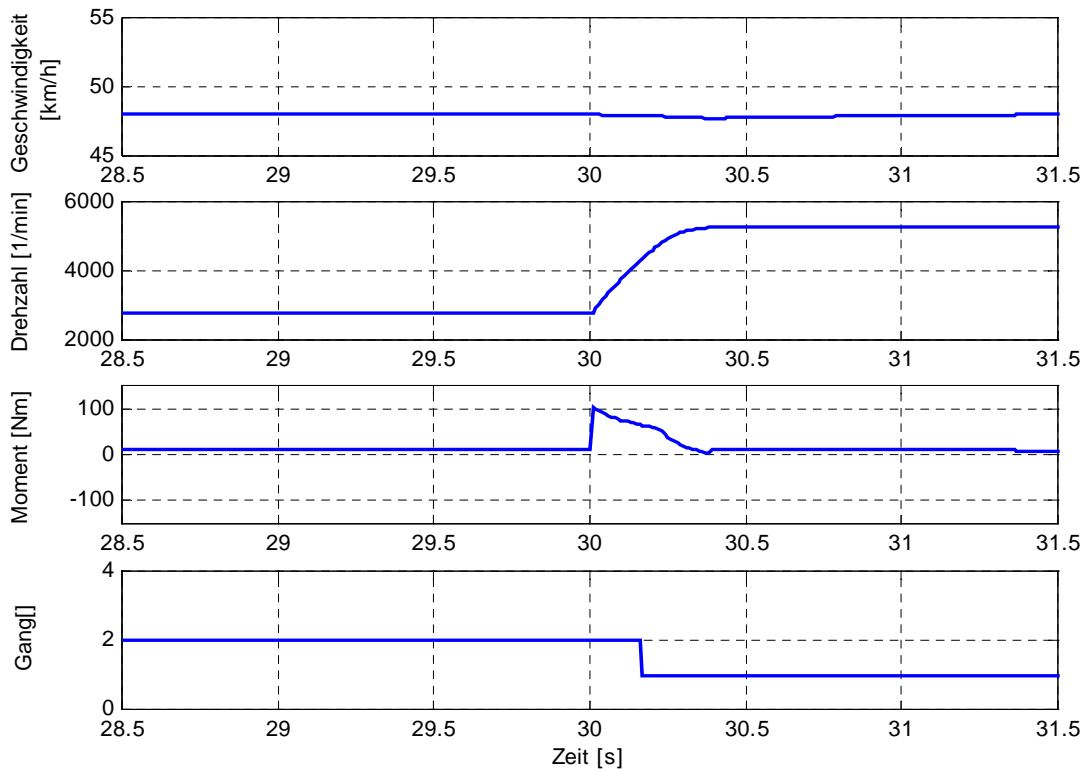


Abb. 3-5: Simulierte Rückschaltung zweiter – erster Gang

3.2 Inbetriebnahme RCP-System auf einem dynamischen Rollenprüfstand

Nach dem mechanischen Aufbau des Prototypen erfolgte die Inbetriebnahme der Sub-Systeme im Fahrzeug auf dem dynamischen Rollenprüfstand des ika. Um das Fahrzeug auch elektrisch über einen beliebig langen Zeitraum betreiben zu können – normalerweise würde die begrenzte Kapazität der Batterie nur kurzzeitigen rein elektrischen Betrieb erlauben – wurde hierbei die Antriebsbatterie von einem modular aufgebauten Batterietest- und Simulationssystem gestützt bzw. ersetzt.

Die Betriebsstrategie wurde mittels einem PC-basierten RCP-System mit digitalen und analogen I/O's sowie einer CAN-Schnittstelle realisiert, welches die Funktionen des „Hybrid Vehicle Controller“ übernimmt. Dazu wurde das Simulink Model der Betriebsstrategie aus der

Längsdynamik-Simulation als Basis verwendet und mit entsprechenden Adaptionen eingesetzt.



Abb. 3-6: Prototyp auf dem dynamischen Rollenprüfstand

3.3 Betriebsstrategie

Die Betriebsstrategie eines Hybridantriebs beinhaltet die logische und zeitliche Abfolge aller Betriebszustände, d.h. wann welche Komponente des Antriebs wie betrieben wird.

Den Kern des Reglermodells bildet die Betriebsstrategie auf Basis der Simulation des gesamten Fahrzeugs. Vor der Anbindung der vorhandenen Strategie an das reale Fahrzeug mussten jedoch einige logische Abläufe angepasst werden. Besonders die Steuerung des Schaltvorgangs gestaltet sich erheblich aufwendiger als in der Simulation. Als Führungsgrößen für die Betriebsstrategie kommen eine Vielzahl von Parametern zum Einsatz. Im wesentlichen sind dies die momentanen Anforderungen des Fahrers, also der von ihm gewünschte Gang, die angeforderte Leistung und der Betriebsmodus. Aber auch die Betriebsparameter des Fahrzeugs, wie der Ladezustand des elektrischen Energiespeichers, die Fahrzeuggeschwindigkeit, die Drehzahlen der Antriebsaggregate sowie die Betriebszustände von Kupplung und Getriebe sind Führungsgrößen für die Betriebsstrategie. Da mehrere Einflussgrößen kombiniert zum Einsatz kommen, handelt es sich um eine mehrdimensionale Betriebsstrategie. Aufgabe des in Abb. 3-7 gezeigten Zustandsdiagramms ist

die Koordination von Elektromotor und Verbrennungsmotor. In ihr findet die Entscheidung statt, ob elektrisch oder im Hybridbetrieb gefahren wird. Hierzu wurden eine Reihe von Zuständen samt Übergangsbedingungen definiert. Die grafische Umsetzung dieser Abläufe in Stateflow ist dabei zugleich die Implementierung, da die Umsetzung des Stateflow-Diagramms in lauffähigen Code für das RCP-System automatisch erfolgt [2], [4].

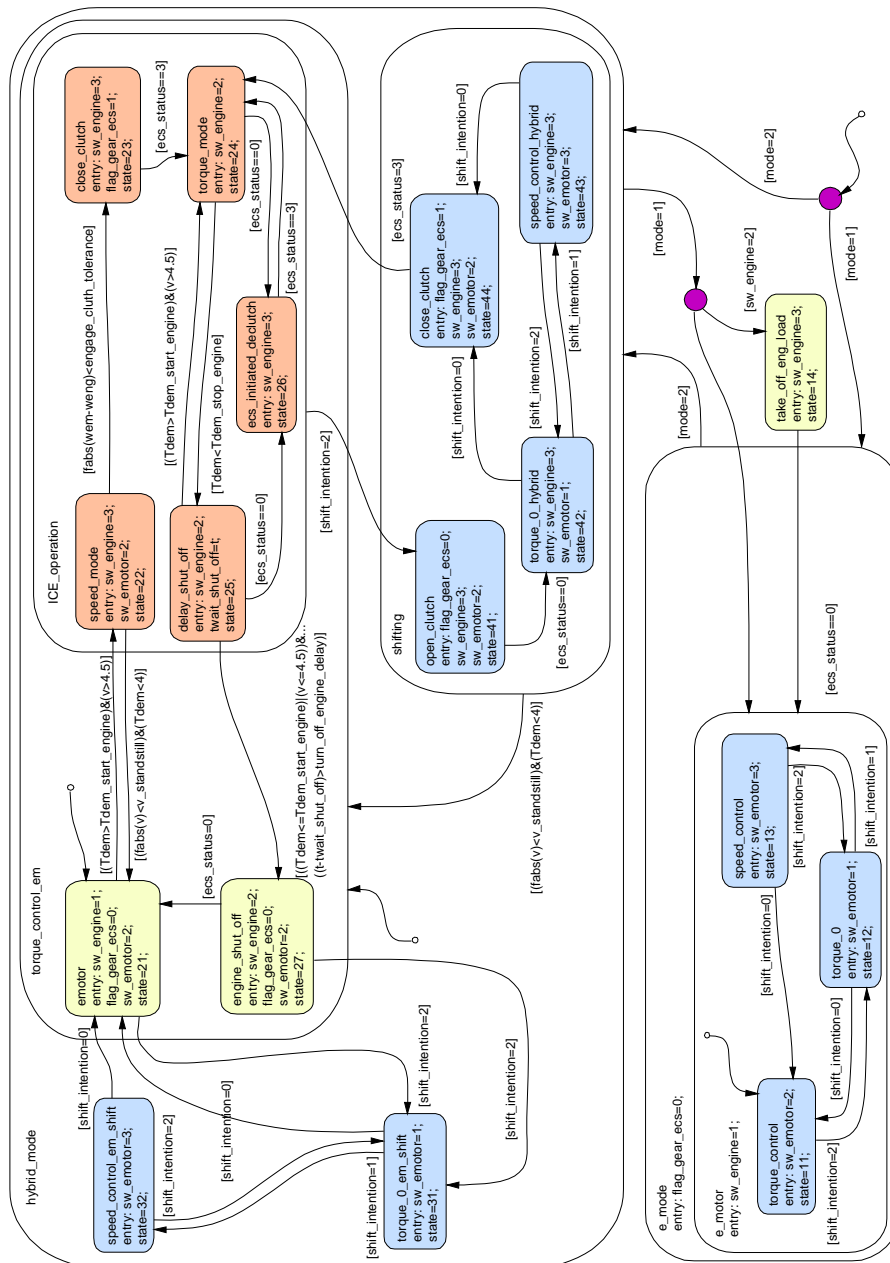


Abb. 3-7: Zustandsdiagramm der Betriebsstrategie in Stateflow

3.4 Gang- und Schaltwunscherkennung

Ein weiteres Teilproblem, was während der RCP-Tests entwickelt und abgestimmt wurde, ist die Gang- und Schaltwunscherkennung. In der Simulation ist der Gang üblicherweise im Testzyklus fest vorgegeben. In der Realität aber wird das Getriebe vom Fahrer betätigt. Das heißt, dass die Steuerung des Fahrzeugs den Schaltwunsch als auch den aktuellen Gang erkennen und die Subsysteme entsprechend den Aktionen des Fahrers ansteuern muss.

Die in Abb. 3-8 gezeigte Logik nutzt als Führungsgrößen die Position der Schaltwelle im Getriebe, die Schaltkraft am Schalthebel sowie ein Signal, das Aufschluss darüber gibt, ob der Synchronisationsvorgang abgeschlossen ist, d.h. die Differenz von Soll- und Ist Drehzahl unterhalb eines Schwellwerts liegt ('sync_ready'). Die Position der Schaltwelle wird über zwei Potentiometer erfasst, die Schaltkraft über einen direkt am Schalthebel angebrachten Kraftsensor. Während des Anstiegs der Potentiometerwerte nach dem Betätigen des Zündschlosses erkennt die Logik Getriebezustände, die nicht dem tatsächlichen Betriebszustand entsprechen. Daher wird die Logik in einem Wartezustand ('start_delay') initialisiert und beginnt mit der eigentlichen Auswertung erst nach 1,5 Sekunden.

Bewegt der Fahrer den Schalthebel vom Leerlauf ('neutral') in Richtung eines Gangs, so ändern sich die Spannungswerte der beiden Potentiometer (AlphaGearPhi und LinearGearX). Somit wird eine der Bedingungen erfüllt, die zu einem Zustand führt, der dem Synchronisationsvorgang ('intended_n') entspricht und die Variable 'shift_intention=1' wird gesetzt. Der Elektromotor und ggf. auch der Verbrennungsmotor werden nun von der Betriebsstrategie in den drehzahlgeregelten Betrieb geschaltet und auf die Drehzahl des gewünschten Gangs synchronisiert.

Kann der Gang eingelegt werden, so wird die Bedingung erfüllt, die zum Zustand des eingelegten Gangs (Gear_n) führt. Dort wird die Variable 'shift_intention=0' gesetzt. Die Antriebsaggregate werden nun von der Betriebsstrategie wieder in den drehmomentgeregelten Betrieb geschaltet und können wieder über das Bedarfsmoment gesteuert werden.

Kann der Gang nicht eingelegt werden, obwohl die Drehzahl der Getriebeeingangswelle leicht oberhalb der synchronen Drehzahl liegt ('sync_ready'), so wird vom Fahrer normalerweise eine erhöhte Betätigungskraft ('ShiftForce') am Schalthebel aufgebracht. Daher wird die Bedingung erfüllt, die zum Zustand des beinahe eingelegten Gangs ('nearly_n') führt und die Variable 'shift_intention=0' wird gesetzt. Von der Betriebsstrategie wird nun das Getriebe lastfrei geschaltet, und der Gang kann eingelegt werden. So wird jetzt die Bedingung erfüllt, die zum Zustand des eingelegten Gangs (Gear_n) führt.

Um den Gang auslegen zu können, muß das Getriebe lastfrei sein. Bewegt der Fahrer den Schalthebel vom eingelegten Gang in Richtung Leerlauf, so ändert sich der Spannungswert des Kraftsensors, und im Zustand 'shift_intention_n' wird das Getriebe lastfrei geschaltet. Nun kann der Gang ausgelegt werden, und die Bedingung, die zum Zustand 'neutral' führt, wird erfüllt.

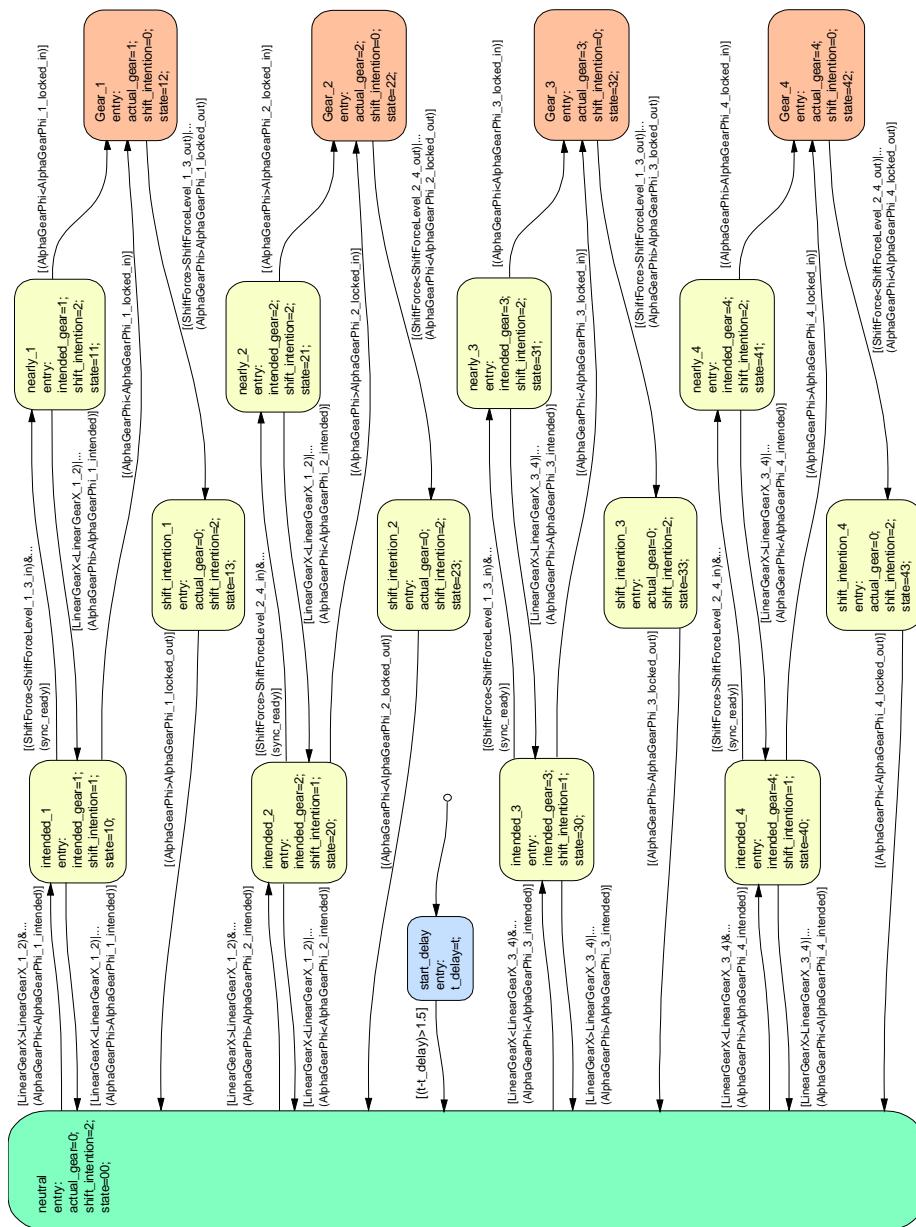


Abb. 3-8: Logik zur Gang- und Schaltwunscherkennung

Um den aktuellen Zustand der Logik in Control Desk überwachen zu können, wurde jeder Zustand nummeriert und als Variable 'state' ausgegeben. So lassen sich eventuelle Fehler der Logik erkennen und anschließend analysieren bzw. abstellen.

Um einen Schaltwunsch des Fahrers und den Betriebszustand des Getriebes bestimmen zu können, werden als Eingangssignale die beschriebenen Werte für Schaltwellenposition und Schaltkraft benötigt. Neben dem Signal, das Aufschluss darüber gibt, ob der Syn-

chronisationsvorgang abgeschlossen ist ('sync ready'), werden zahlreiche Grenzwerte der Potentiometerspannungen zur Abbildung der logischen Zustände des Getriebes benötigt.

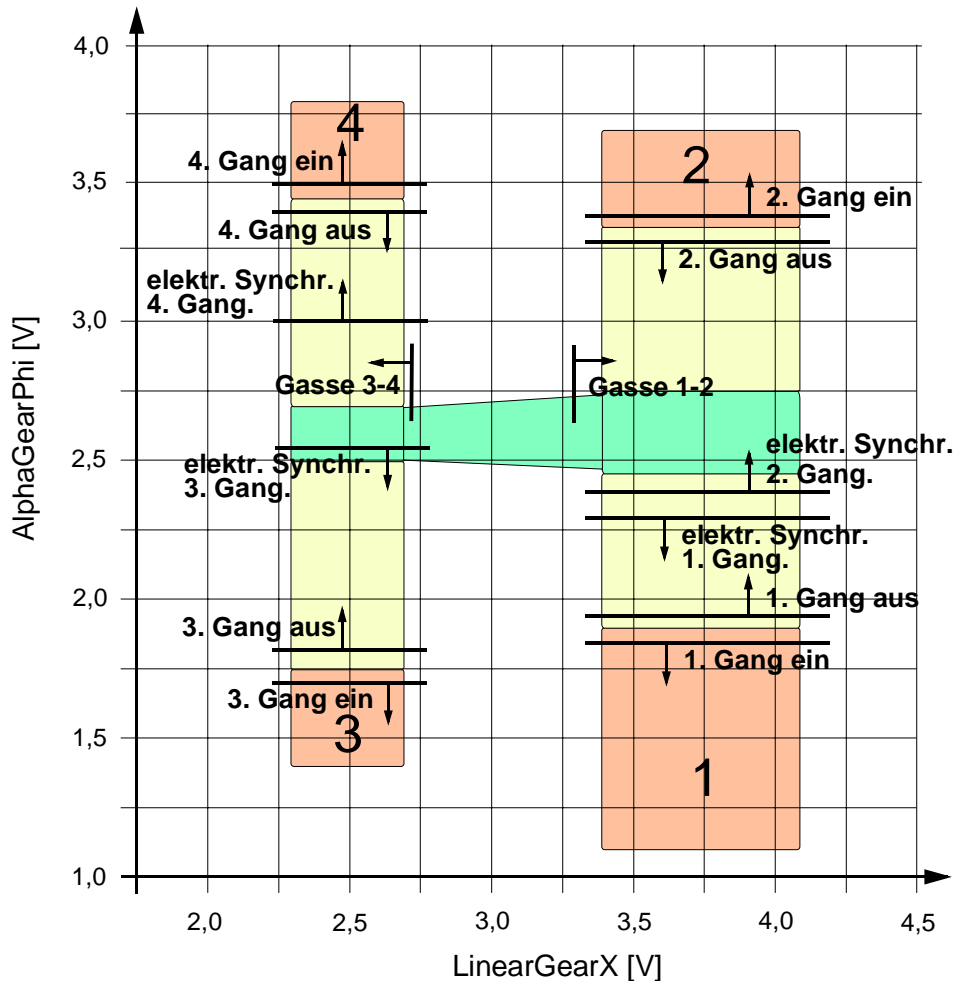


Abb. 3-9: Potentiometerspannungen zur Erfassung der Schaltwellenposition

In Abb. 5-3 sind die Spannungswerte der beiden Potentiometer aufgetragen. Die Werte auf der Querachse entsprechen der Position des Schalthebels in Fahrzeugquerrichtung ('LinearGearX'), die auf der Hochachse der Position des Schalthebels in Fahrzeuglängsrichtung ('AlphaGearPhi'). Daraus ergeben sich die farbig markierten Felder als mechanische Zustände des Getriebes. In den roten Feldern ist der Gang eingelegt, also der Formschluss hergestellt. In den gelben Feldern wird durch die Synchronringe ein Reibmoment auf das Gangrad des entsprechenden Ganges aufgebracht, um die Drehzahlgleichheit zwischen der Getriebeeingangswelle und dem Gangrad herzustellen. In den grünen Feldern wird kein Moment übertragen, das Getriebe befindet sich im Leerlauf.

Die eingetragenen Markierungen stellen die Grenzen zwischen logischen Zuständen dar, die zur Erkennung des Betriebszustands von der Logik ('Gear_ID Shift_ID') als Eingangsgrößen verwendet werden. Der Schaltkraftsensor ist direkt am Schalthebel angebracht. Lediglich

Betätigungskräfte in Fahrzeiglängsrichtung ändern die Ausgangsspannung des Sensors. Der Proportionalitätsfaktor des Sensors beträgt 1,0V/20N, so dass auch kleine Betätigungskräfte erkannt werden.

3.5 Versuche auf dem Prüfstand

Zur Verifikation der oben gezeigten Zustandsdiagramme wurden ausführliche Tests durchgeführt, von denen zwei beispielhafte Manöver vorgestellt werden:

- ein Schaltvorgang vom zweiten in den ersten Gang
- sowie die Zuschaltung des Verbrennungsmotors aus dem elektrischen Betrieb heraus.

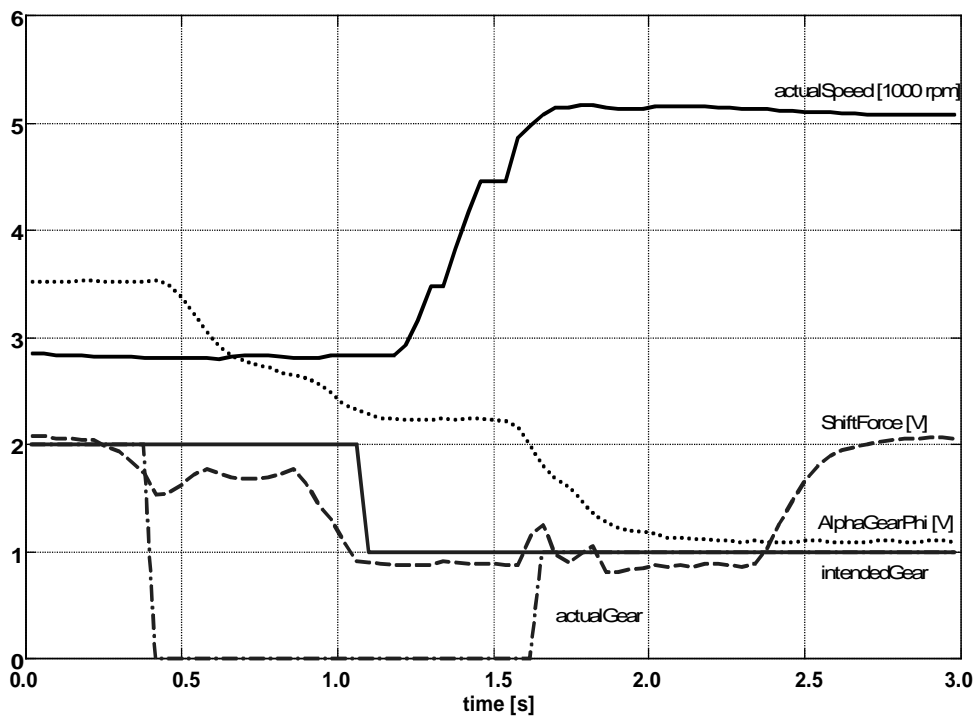


Abb. 3-10: aktive Synchronisation der Getriebeeingangswelle durch den Elektromotor

Ausgangszustand der Messung ist der elektrische Fahrbetrieb im zweiten Gang mit einer Elektromotordrehzahl von ca. 2800 U/min. Ab ca. 0,25 s sinkt der Spannungswert für die Schaltkraft (gestrichelt) ab, da der Fahrer den Schalthebel in Fahrtrichtung betätigt, und der Gang (strich-punktiert) wird ausgelegt. Dies ist auch am Verlauf der Potentiometerspannung für die Schaltwellenposition (punktiert) zu erkennen, da der Wert deutlich absinkt. Die Drehzahl des Elektromotors (durchgezogene Linie) bleibt aber solange unverändert, bis der neue Wunschgang (durchgezogene Linie) erkannt wird. Diese Erkennung basiert auf Werten

der Potentiometerspannung unterhalb von 2,3 V. Jetzt steigt die Drehzahl des Elektromotors auf die berechnete synchrone Drehzahl des ersten Gangs an (ca. 5200 U/min). Bei ca. 1,6 s kann der erste Gang dann eingelegt werden, was durch ein weiteres Absinken der Potentiometerwerte erkannt wird. Ab etwa 2,4 s beginnt der Fahrer, den Schalthebel loszulassen und die Fahrt wird im ersten Gang fortgesetzt. Da der Übersetzungsunterschied zwischen erstem und zweitem Gang der größte ist, und in den ersten Gang bei einer Geschwindigkeit von 48 km/h geschaltet wurde, muss der Elektromotor eine große Drehzahldifferenz von etwa 2400 U/min überbrücken. Daher stellt dieser Schaltvorgang also größere Anforderungen an die aktive Drehzahlsynchronisation, als dies bei Schaltvorgängen in üblichen Betriebspunkten der Fall wäre. So ist auch die relativ große Dauer dieses Vorgangs verständlich.

Der zweite Vorgang, der dargestellt wird, ist das Zuschalten des Verbrennungsmotors (Abb. 3-11).

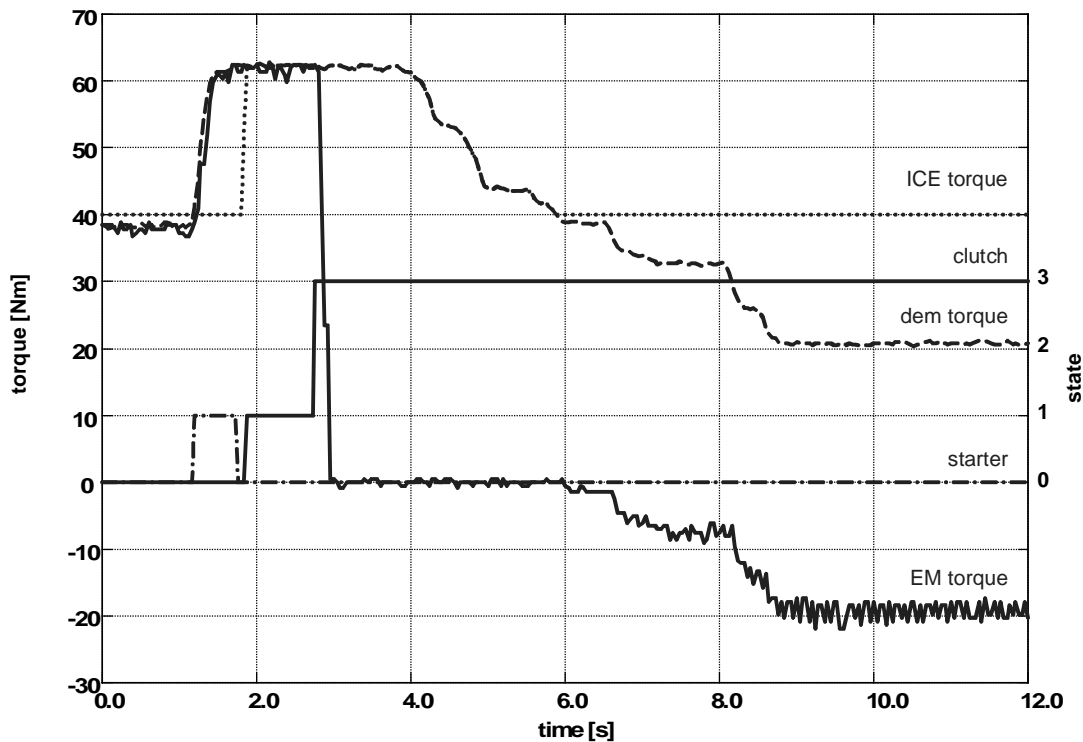


Abb. 3-11: Zuschalten des Verbrennungsmotors und Lastanhebungsbetrieb

Im rein elektrischen Fahrbetrieb, also bei geöffneter Kupplung (durchgezogene Linie bei 0), entspricht das Drehmoment des Elektromotors (durchgezogene Linie) dem Bedarfsmoment (gestrichelt), das durch den Fahrer vorgegeben wird. Die punktierte Kurve ist lediglich die Momentenvorgabe an den Verbrennungsmotor und somit bei nicht geschlossener Kupplung ohne Aussage. Das Bedarfsmoment steigt ab 1,1s deutlich über 40Nm, so dass der Verbrennungsmotor über den Anlasser gestartet wird (strich-punktierte Linie bei eins). Die

Kupplung beginnt den Schließvorgang (durchgezogene Linie springt von Null auf eins) und die Momentenvorgabe an den Verbrennungsmotor wird dem Bedarfsmoment angepasst. Sobald die Kupplung geschlossen (Kupplungszustand gleich drei) ist, wird das Moment des Elektromotors auf null abgesenkt und das Moment des Verbrennungsmotors entspricht dem Bedarfsmoment. Ab ca. 4s sinkt das Bedarfsmoment ab und unterschreitet ab ca. 5,9s den Wert von 40Nm. Da das Moment des Verbrennungsmotors bei 40Nm bleibt, wandelt der Elektromotor die Differenz zwischen Bedarfsmoment und Moment des Verbrennungsmotors in elektrische Energie, durch Aufbringen eines negativen Moments. So wird ein Betrieb des Verbrennungsmotors im Teillastbereich vermieden, und elektrische Energie zum Laden der Batterie erzeugt.

Die Vorteile des Einsatzes der RCP-Hardware zur Realisation der Funktionen der Hybrid Vehicle Controller lagen bei dem hier vorgestellten Einsatzzweck in der Flexibilität des Systems. Beginnend mit der grafischen Programmierung von Algorithmen mittels Zustandsdiagrammen in Stateflow können sie dann automatisch in ein lauffähiges Programm umgesetzt werden. So steht ein leistungsfähiges Werkzeug zur Verfügung, mit dem auch komplexe Abläufe schnell und zuverlässig realisiert und in der Praxis in Bezug auf ihre Funktionalität getestet werden können.

Die so gewonnenen und ausgetesteten Algorithmen wurden für das erste Fahrzeug dann manuell in C-Code umgesetzt, da zu diesem Zeitpunkt noch keine hinreichenden Erfahrungen mit Auto-Code-Generatoren für das Zielsystem, den Infineon C167 Mikrocontroller vorlagen.

4 2.Prototyp:

Im Rahmen des Projekts INMOVE wird zur Zeit ein zweiter Prototyp aufgebaut, in den die Erfahrungen aus dem ersten Fahrzeugprototypen einfließen. Weiterhin kommen zum Teil erweiterte oder weiterentwickelte Komponenten zum Einsatz. So verfügt z.B. der Verbrennungsmotor über ein Steuergerät mit verbessertem E-Gassystem und einem CAN-Interface. Der Elektromotor sowie der zugehörige Umrichter wurden ebenfalls verbessert. Hier standen Produktionskosten und eine Verbesserung des Wirkungsgrads im Vordergrund.

Im Zuge der Einführung von Targetlink der Fa. dSpace soll in diesem Projektabschnitt die automatische Codegenerierung bereits vorhandener Simulationsmodelle aus Matlab/-Simulink direkt auf den Hybrid Vehicle Controller erfolgen.

Die Software des Hybrid Vehicle Controller setzt sich somit aus den folgenden Teilmodulen zusammen:

- Betriebssystem OSEK mit CAN-Treibern /5/
- I/O-Treiber für Steuerrechner

- manuelle programmierte Funktionen der Betriebsstrategie
- Autocode generierte Funktionen der Betriebsstrategie auf Basis von Matlab/Simulink

Die Einbindung der mittels Autocode generierten Funktionen in den bestehenden C-Code geschieht manuell. Am Beispiel der aktuellen Schaltroutine wird diese Vorgehensweise erläutert.

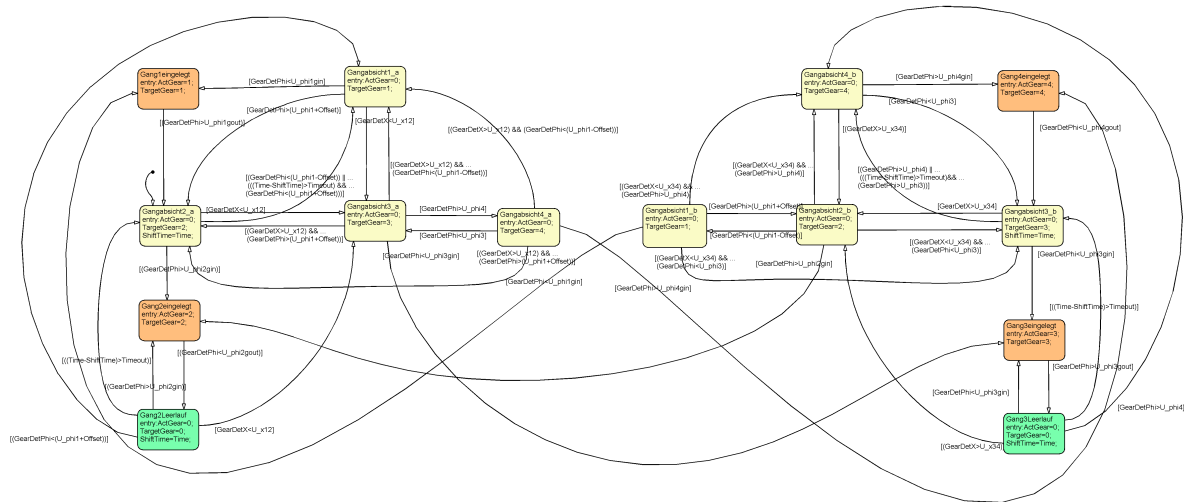


Abb. 4-1: Schaltstrategie des 2. Prototypen

In der unter Abb. 3-8 dargestellte Gangerkennungslogik des ersten Prototypen ergibt sich der erkannte Zielgang ausschließlich aus der Position der Schaltwelle unabhängig von der Vorgeschichte. Hier wird z.B. beim Schalten vom ersten in den zweiten Gang letzterer erst als Zielgang erkannt, wenn der Schalthebel schon über die Leerlaufposition hinaus in Richtung zweiter Gang bewegt worden ist. (Siehe auch Abb. 3-9).

Beim zweiten Fahrzeug (Abb. 4-1) geht auch die Vorgeschichte in die Bestimmung des Zielgangs ein. Somit wird der Zielgang früher erkannt, die Drehzahlsynchronisierung früher erreicht und die Schaltzeit verkürzt. Beispielsweise ist beim Schalten aus dem ersten Gang der zweite Gang als Zielgang standardmäßig eingestellt. Nur bei Betätigung des Schalthebels aus der linken in die rechte Gasse führt dies zu einer neuen Zielgangbestimmung. Die hier dargestellte Zustandsmaschine unterscheidet sich in wesentlichen Punkten von der des ersten Fahrzeugs. Eine manuelle Umsetzung der geänderten Strategie in C-Code wäre sehr aufwendig gewesen, während das Werkzeug der Auto-Code-Generierung hier eine schnelle Umsetzung der Änderungen möglich machte.

Die Erfahrungen aus dem ersten Prototypen beim Einsatz des RCP-Systems haben gezeigt, dass mittels Matlab/Simulink/Stateflow umgesetzte Strategie erfolgreich im realen Fahrzeug eingesetzt werden können. Aufgrund der Verfügbarkeit der Autocodegenerierung für den

Hybrid Vehicle Controller und der Fähigkeit zur Online-Parametrisierung via Applikationstool CANape ist der Einsatz des RCP-Systems für den zweiten Prototypen nicht mehr nötig.

Ein weiterer Grund ist, dass mit den Erfahrungen aus dem ersten Prototypen die Wiederverwendung und Erweiterung vieler der mit dem RCP-Ansatz getesteten Module unter Targetlink möglich ist. Hinzu kommt, dass das Applikationstool CANape eine Kalibrierung online ermöglicht, somit ist eine Vorkalibrierung mit RCP nicht erforderlich.

Weiterhin ermöglicht die Verwendung der Auto-Code-Generierung eine hohe Flexibilität bei der Umsetzung und Evaluierung neuer Strategien, da mit der Beschreibung einer Steueraufgabe anhand von Zustandsdiagrammen gleichzeitig die Implementierung weitgehend gegeben ist. Die autocode-generierten Module müssen manuell eingebunden werden. Hierzu ist in Abb. 4-2 beispielhaft der Rumpf der autocode-generierten Funktion zur Gangerkennung dargestellt.

```
/* *****\
   Model step function of subsystem(s): Gangerkennung
   \***** */
Void Gangerkennung(
  UInt16  Sal_in1      /* LSB: 2^0 OFF: 0 MIN/MAX: 0 .. 65535 */,
  UInt16  Sal_in2      /* LSB: 2^0 OFF: 0 MIN/MAX: 0 .. 65535 */,
  UInt32  Sal_in3      /* LSB: 2^0 OFF: 0 MIN/MAX: 0 .. 4294967295 */,
  UInt8 * Sal_ActGear2 /* pointer to LSB: 2^0 OFF: 0 MIN/MAX: 0 .. 255 */,
  UInt8 * Sal_ActGear3 /* pointer to LSB: 2^0 OFF: 0 MIN/MAX: 0 .. 255 */)
{

Code der Statemaschine

  /* TargetLink outport: Gangerkennung/ActGear2 */
  *Sal_ActGear2 = Sal_Chart_ActGear;

  /* TargetLink outport: Gangerkennung/ActGear3 */
  *Sal_ActGear3 = Sal_Chart_TargetGear;
}
```

Abb. 4-2: C-Code-Rumpf einer autocode-generierten Funktion zur Gangerkennung

Der C-Code dieser Funktion wird mit Hilfe eines Make-File zu den manuell codierten Teilen hinzugelinkt und so innerhalb der Betriebsstrategie verfügbar. Der Funktionsaufruf erfolgt mit den drei Eingangsgrößen Sa1_in1, Sa1_in2, Sa1_in3 (Position der Schaltwelle in x- und y-Richtung sowie Systemzeit). Zum Erhalt der Rückgabewerte an die aufrufende Routine werden zusätzlich noch zwei Zeiger übergeben (Sa1_ActGear2, Sa1_ActGear3), über die der ermittelte aktuelle Gang sowie der erkannte Zielgang übergeben werden. Die Einbindung von autocode-generierten Funktionen ist somit einfach möglich.

Zusammenfassend kann man also feststellen, dass der Entwicklungsprozess durch die Anwendung von automatischer Code-Generierung deutlich beschleunigt werden kann. Zwar ist der Aufwand zur Einarbeitung und Konfiguration des Prozesses zur Auto-Code-Generierung nicht unerheblich – die Tücke steckt hier oftmals im Detail -, wenn der Prozess einmal „steht“, erlaubt er aber relativ einfach und sicher Modifikationen vorzunehmen und auszutesten.

5 Ausblick

In Zukunft ist für den zweiten Prototypen geplant, die automatische Codegenerierung für die gesamte Software einzusetzen. Dazu wird bereits auf der Matlab/Simulink/Targetlink-Ebene der Zugriff auf die Hardwareschnittstellen des Hybrid Vehicle Controllers ermöglicht. Hierzu werden die entsprechenden Module zum Zugriff auf die Schnittstellen (digitale und analoge I/Os, CAN-Bus) erstellt. Die Benutzerschnittstelle in Simulink ist in Abb. 5-1 gezeigt und in Teilen bereits realisiert.

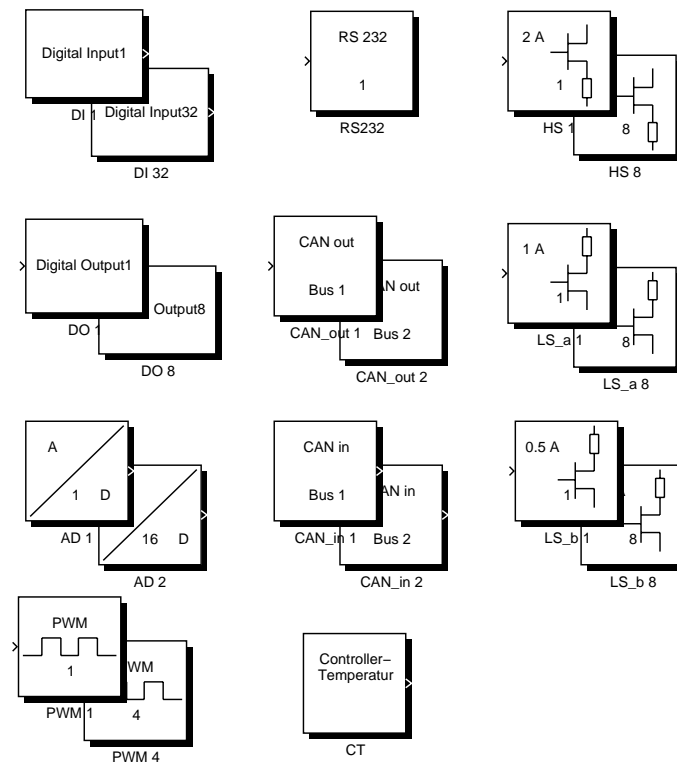


Abb. 5-1: Module zur Ansteuerung der Schnittstellen

Weiterhin ist geplant, die Funktionalitäten des OSEK-Betriebssystems (Task-Konzept, Ressourcenverwaltung, Semaphore, Messagesystem) unter Matlab/Simulink/Targetlink abzubilden. Erste Schritte in diese Richtung sind bereits vollzogen.

6 Literaturverzeichnis

- /1/ Wallentowitz, Bady, „Chancen durch das Elektroauto“, VDI-Tagung "Batterie-, Brennstoffzellen- und Hybridfahrzeuge", Dresden, 17./18. Februar 1998
- /2/ Biermann, Bady, "Hybridantriebe-Strukturvarianten, Betriebsstrategien sowie deren Vor- und Nachteile", 5. Symposium "Elektrische Straßenfahrzeuge", Esslingen, 26./27. März 1998
- /3/ Wallentowitz, Biermann, Bady, Renner, "Strukturvarianten von Hybridantrieben", VDI-Tagung "Hybridantriebe", München, 25./26. Februar 1999,
- /4/ Rühle, Bady, Renner "INMOVE - A Single Shaft Parallel Hybrid Concept", Global Powertrain Congress, Stuttgart, 5.-7. Oktober 1999,
- /5/ Amsel, Renner, Wittek, Brock
"Anwendung des OSEK-Betriebssystems am Beispiel der Betriebsstrategie eines Hybridfahrzeugs", 8. Aachener Kolloquium Fahrzeug- und Motorentechnik 1999, Aachen, 04.-06. Oktober 1999.