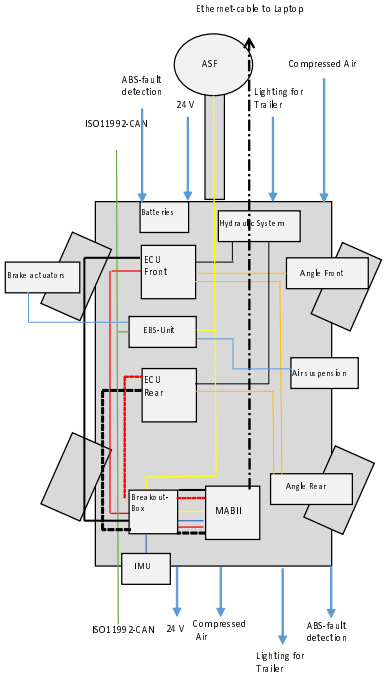
CHALMERS



Real-Time Control Interface for a Steered and Braked Converter Dolly for High Capacity Transport Vehicles

Master's thesis in Automotive Engineering

SEBASTIAN FRANZ
MICHAEL HOFMANN

Department of Applied Mechanics
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden 2015

MASTER'S THESIS IN AUTOMOTIVE ENGINEERING

Real-Time Control Interface for a Steered and Braked Converter Dolly for High Capacity Transport Vehicles

SEBASTIAN FRANZ
MICHAEL HOFMANN

Department of Applied Mechanics
Division of Vehicle Dynamics
Vehicle Dynamics Group
CHALMERS UNIVERSITY OF TECHNOLOGY

Göteborg, Sweden 2015

Real-Time Control Interface for a Steered and Braked Converter Dolly for High Capacity Transport Vehicles
SEBASTIAN FRANZ
MICHAEL HOFMANN

Master's thesis 2015:01
ISSN 1652-8557
Department of Applied Mechanics
Division of Vehicle Dynamics
Vehicle Dynamics Group
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone: +46 (0)31-772 1000

Cover:
Some explanation

Chalmers Reproservice
Göteborg, Sweden 2015

Real-Time Control Interface for a Steered and Braked Converter Dolly for High Capacity Transport Vehicles
Master's thesis in Automotive Engineering
SEBASTIAN FRANZ
MICHAEL HOFMANN
Department of Applied Mechanics
Division of Vehicle Dynamics
Vehicle Dynamics Group
Chalmers University of Technology

Abstract

To improve the dynamic behaviour of high capacity transport vehicles, the second semi-trailer can be equipped with steered and braked axles. By improving the actuation and steering strategy, the combinations performance can be increased drastically.

This thesis modified an existing steered dolly to accommodate a RapidPrototpying system, so that the previously developed steering algorithms can be executed in real-time and evaluated in a hardware-in-the-loop simulation as well as on-track testing further on. The main-outcomes are a ready-to-use interface for algorithm development in MATLAB's Simulink that allows to request steering angles and braking torques individually for each axle and supplies feedback for all relevant parameters (e.g. articulation angle between different units of combination, steering angle of tractor) for their computation in real-time. To ensure safety of the whole combination, especially during track-tests, a supervisor system was implemented to limit the request based on vehicle-dynamic simulations and hardware limitations. Additionally an inertial measurement unit setup was developed and implemented for use on the combination to gather sufficient data for each of the combination's independently moving units dynamic behaviour during testing to calibrate and parametrize the tested algorithms.

In the scope of this thesis the dolly was modified to conduct hardware-in-the-loop simulations. All necessary systems to achieve this were built as prototypes for a workshop test-environment limiting the use-case to a standing-system. Reliable track-proof solutions were developed and commissioned for manufacturing, so that track-testing is possible soon after.

Keywords: High capacity transport vehicles, steered converter dolly, RapidPrototyping, dSpace MicroAutobBoxII, Arduino platform

Contents

List of Figures

List of Tables

Acronyms

ABS Anti-lock braking system. 19, 53

ASF Angular Sensor Fusion. 21, 36, 39, 43

CAN Controler Area Network. i, v, 7, 14, 15, 17–25, 27–32, 36, 39–44, 46, 53

CoG Center of Gravity. 12, 24

DC Descrete Current. 44

EBS Electronic Brake System. 21, 22, 30, 37, 43, 46, 53

ECU Electronic Control Unit. 2, 16, 18, 21, 36, 37, 40, 41, 43, 44, 53

ESC Electronic Stability Control. 1, 53

ETS Electronic Truck Steering. v, 15–18, 21, 30, 31, 35–37, 40, 42–44, 49, 53

FMEA Failure Mode and Effect Analysis. iv–vi, 35, 36, 57, 58

GPS Global Positioning System. 19

GUI Graphical User Interface. v, 45, 56

HCT High Capacity Transport. v, vi, 1, 2, 5–7, 15, 24, 40

HiL Hardware In the Loop. v, 2, 5, 22, 25, 28, 33, 35, 43, 45, 46, 51

HMI Human Machine Interface. 39

I^2C Inter-Integrated Circuit. 23–25

IDE Integrated Development Environment. 31

IMU Inertial Measurment Unit. v, 19, 23–25, 31, 37, 42, 43, 45

IP Internet Protocol. 28

LEMO Swiss company specialized in industrial connector manufacturing. 18

MABII dSpace MicroAutoBoxII. v, 14, 17–22, 24, 25, 27–30, 35, 37, 40, 42, 44–46

OEM Original Equipment Manufacturer. 16

RCP Rapid Control Prototyping. 8

RPN Risk Priority Number. 35

RTI Real-Time Interface. 27, 28, 30

SPI Serial Peripheral Interface. 16, 19, 25, 31

UART Universal Asynchronous Receiver Transmitter. 42, 43

UDP User Datagram Protocol. 18, 19, 25, 28, 29, 46

UML Unified Markup Language. 7

VDS Volvo Dynamic Steering. 23

VMM Vehicle Motion Management. 39

VSE V.S.E. Vehicle Systems Engineering B.V.. 15, 16, 18, 43, 44

VTM Virtual Truck Model. 22, 28, 39, 45, 46, 52

ZIF Zero Insertion Force. v, 18, 27, 56

1 Introduction (3 Seiten)

1.1 Purpose

Heavy goods-transport on the road has constantly increased over the last decades. Coupled with the stricter environmental regulations concerning CO_2-emissions and pollution, the call for more economical transport solution has led to the wider introduction of HCT vehicles. Those truck-trailer combinations have a longer history in geographical areas with low population density, mining and transport within factory sites where rail-road transport is not a viable option but transportation of large volumes and tonnages are called for. The prospects of saving costs on driver's salaries, reduced fuel consumption and decreased costs suggests the introduction of those combinations in other environments as well. Introduction of a new vehicle class leads to many challenges in safety, research and development, and legislation.

The driving behaviour of HCT vehicles is in many ways different to that of standard trucks and needs to be researched in great detail to gain an understanding of the vehicle's dynamic properties, that is equally detailed as it is for other vehicle classes. This will lead to development of better safety and assistance systems and thus reduce threat potential, accidents and fatalities involving this emerging mode of transportation. Different usage patterns of HCT vehicles have to be considered as well, when developing functions for HCT vehicles. For example inner-city use is no prevalent use-case for HCT vehicles, whereas highway safety features and handling properties at higher speeds are prime goals due to high percentage of highway-driving for HCT vehicles. Nevertheless maneuverability for docking is also a development goal.

Besides the technical implementation, socio-economic aspects have to be considered. Legislation has to be adjusted to allow for longer vehicle classes, including new certification processes and driver training. Furthermore infrastructure might have to be modified or reviewed to accommodate the needs and dimensions of extended truck combinations.

The research project in which this thesis is embedded aims to develop an active dolly, meaning that steering will be autonomously conducted by the dolly based on the driving situation at hand and various vehicle parameters (e.g. speed, steering wheel angle). Furtheron braking capabilities are to be implemented to act in a similiar fashion as an Electronic Stability Control (ESC) by creating a yaw-moment countering undesired vehicle movements. This counter-stearing will be achieved through wheel-individual brake-application.

This high-level control algorithm will be executed on a rapid-prototyping system which is linked to and controls the dolly. To supply this connection between the hardware and control-algorithm implemented in the modeling-environment Simulink is the main-task of this thesis.

1.2 Objectives

The main-goals that are supposed to be achieved within this thesis' scope of work are:

- Develop a software interface for the high-level control algorithm implemented in Simulink to be run on a rapid-prototyping system to control the steering system on an active dolly.
- Develop the physical hardware interface with the dolly; establish a suitable environment connection for the rapid-prototyping system on-board of the dolly

- Develop a measuring solution to determine the processing delays in the sensing system as well as the delays introduced by computation and actuator reaction times. Determine and try to minimize these occuring delays.
- Verification of designed systems through different stages of HiL tests
- Develop a safety system that continously monitors the active steering system, prevents malicious inputs and triggers necessary warnings.
- Develop an interface that allows to contiously determine the system's maximum actuation capabilities depending on the system's current properties (axle load, speed, steering angle, yaw-behaviour). Utilize a vehicle dynamics model to find out the critical boundries for HCT vehicles' behaviour.
- prepare on-track testing (design road-ready measuring equipment, start-up procedure, repeatable pre-recorded maneuver decription for automated testing)

1.3 Limitations

To implement the actual high-level algorithm to compute the desired angle for the dolly's steerable axles is not in the scope of this thesis. It was developed in the research project in which this thesis is embedded. Nevertheless to establish an easier insight into the interfaces' parameters, an overview of the structure, in- and outputs of the underlying computational steering model is needed and shall be presented in chapter 3.

The hardware- and low-level control-system of the hydraulic actuators is in place already and thus will not be part of this thesis. It is supplied as turn-key software by the manufacturer and readily available on the dolly's Electronic Control Unit (ECU)s. Substantial modifications are necessary to achieve the desired goals, though. The ECU's software version will be available fully calibrated and parametrized for the dolly at hand and thus provide a reliable working base to build upon.

Braking the dolly's axles individually will not be part of the testing and development scope of this work. Nevertheless the conceptual solution for including this functionalities furtheron will be outlined and a foundation for sending actuation request to these systems will be included in the rapid-protoyping interface block resulting from this thesis.

1.4 Structure of this thesis

In the first two sections of this thesis a brief overview of the legal situation concerning HCT vehicles for different countries, the current state of the art and ongoing research in the field of HCT vehicles shall be presented (chapter 2). Furthermore an introduction to the model, that will be run on the rapid-prototyping system will be given (chapter 3). Those two chapters are meant to give an introduction into the matter and are mainly based on literature review.

In the succeeding chapters the conducted development work will be described in detail. As a loose guiding structure for this work the development process after the V-model (see 2.4) will be referred to. Starting with a description of the utilizied hardware-systems and their interconnections in chapter 4, followed by detailing the different software-tools and

environments running on those hardware-platforms in chapter 5. In chapter 7.3 the measuring concepts and theoretical details for the determination of the overall processing delays in the control-chain will be discussed. As at the planned high speeds and great inertia for testing safety is a major concern, safety functions will be implemented and systematically evaluated. This will be outlined and discussed in chapter 6. After the development work was described the created (sub-)systems will subsequently be validated and peu-à-peu put together and following the V-model evaluated at different stages of integration (chapter 7).

The work closes with a discussion of the results collected during testing and a conclusion where the authors will try to give recommendation for practical implementation and outline future research work in the field.

2 Overview (6-7 Seiten)

2.1 Background of high capacity transport vehicles (HCT vehicles)

In this section first the legal regulations for HCT vehicles in different countries are outlined, following that, the requirements for drivers of HCT vehicles are described. After that, the impact of HCT vehicles on the road infrastructure is looked upon.

In countries of the EU the permitted maximum length of a truck-trailer combination is 18.75 m and its maximum weight is 44 tonnes. There is however the possibility for countries to make exceptions from that rule.[36] For example in Sweden and Finland road trains can be up to 25.25 m long with a maximum weight of 60 tonnes.[37] In several other countries of the EU HCT vehicles are allowed on certain roads for testing purposes. Table 2.1 shows the maximum length and weight of HCT vehicles in different countries.

Table 2.1: HCT vehicles in different countries[37][35][2][17][6]

Country	Max. Length [m]	Max. Weight [t]
EU (general)	18.35	44.00
Sweden	25.25	60.00
Finland	25.25	60.00
Australia	53.50	132.00
USA(trailers without truck)	26.07	59.86
Canada	36.88	63.50
Mexico	31.00	75.50

In Sweden there are attempts to have HCT vehicles that exceed the current dimensions. One example is an A-double combination, which consists of a tractor, two semitrailers and a dolly between the semitrailers. This type of combination was used for the simulations and HiL tests conducted in this project.

There are certain regulations for HCT vehicles in terms of driver requirements. In countries of the European Union a person who drives a HCT vehicles is required to have the CE class drivers license, which allows you to drive a tractor with a mass over 3.5 tonnes with a trailer, whose mass is over 750 kg. To be able to acquire that drivers license the person has to be over 21 years old. [25] On top of that, there have been more requirements for the drivers that took part in trials of HCT vehicles in European countries. For example for the trials in the Netherlands drivers were required to have their drivers license for at least five years, to not have lost it during the last three years and they had to acquire a special certificate. To get that certificate the drivers had to take theoretical as well as practical courses. [1]

When it comes to the impact of road transport on road infrastructure, road wear is the most important effect. The main factor of road wear is the axle load of the vehicles. The following

equation for a load equivalency factor shows how different axle loads impact road wear.

$$\frac{N_{ref}}{N_x} = \left(\frac{W_x}{W_{ref}}\right)^4 \tag{2.1}$$

Where W_x and W_{ref} are axle loads and N_x and N_{ref} are the corresponding numbers of load applications.[15] Equation (2.1) shows that for example doubling the axle load leads to a road wear that is 16 times higher.

Assuming an equal distribution of the load over the different axles, the axle load for a vehicle is calculated as follows:

$$W_x = \frac{m_{tot}}{n_{axl}} \tag{2.2}$$

Where m_{tot} is the total mass of the vehicle and n_{axl} is the numbers of axles. A standard tractor-semitrailer combination has five or six axles and a maximum mass of 40 tonnes. With equation (2.2) the axle load, if fully loaded, for that combination is 8 tonnes/axle and 6.67 tonnes/axle, respectively. An A-double-combination, which has ten or eleven axles and a maximum mass of 80 tonnes, also reaches a maximum axle load of 8 tonnes/axle and 7.27 tonnes/axle, respectively. For a 25.25 m combination, which has nine or ten axles and a maximum mass of 60 tonnes, the maximum axle load is even lower than for the standard combination. The maximum axle load for this case is 6.67 tonnes/axle and 6 tonnes/axle, respectively.
Based on that it can be concluded that high capacity transport vehicles don't lead to higher road wear than standard combinations.

2.2 Ongoing research

There is a lot of research going on concerning high capacity transport vehicles. The main topics are the safety of HCT, their environmental impact as well as economical aspects.

There are many concerns about the safety of HCT in public opinion, so there have been quite a lot studies that investigated this issue. One concern is the extended length of HCT, which causes an increase of time needed to overtake a combination. Studies showed that accident risk for overtaking a long combination is not statistically higher than for standard combinations [3]. For overall safety, the majority of studies show that the use of HCT increases traffic safety due to the reduced number of vehicles on the roads, which is induced by the higher capacity of HCT vehicles. However, HCT might have a higher accident rate for individual vehicles [3]. In another study crash data from the Swedish Traffic Accident Data Acquisition was analyzed. The results showed that there is no higher rate of fatal or severe crashes per traveled vehicle km for HCT vehicles compared to standard vehicle combinations. [5]

To reduce the effects on safety, which might occur with the use of HCT, different safety systems are being developed.

One aim of safety systems for HCT is to reduce the rearward amplification of yaw rate and lateral acceleration. A possibility to achieve that is to use a steerable dolly. In section 3.2 a controller to reduce rearward amplification for high speed is described.
Another problem of HCT is the off-tracking of the last trailer for low speed cornering. Off-tracking means that the trailer doesn't follow the path of the tractor but moves on a radius

that is smaller then the tractors. This issue can also be addressed by using a steerable dolly. In section 3.1 a controller to improve the path-following is described.

Since one of the reasons for using of HCT vehicles is the expected positive influence on the environment compared to standard combinations, several studies that address this topic can be found. Overall this studies showed that with the use of HCT vehicles the environmental impact of goods transportation can be reduced significantly for high loading factors. The fuel consumption is about 15% lower compared to the standard vehicle combinations used in the EU. In the number of trips a reduction of 32% can be seen. This reduction is achieved because two HCT vehicles can carry the same amount of goods as three standard vehicle combinations. [4] However, the environmental impact of HCT vehicles can become negative if due to a high penetration of HCT vehicles goods transportation is shifted from rail to the road. [10]

The introduction of new systems is alway driven by economical aspects, especially in the transportation industry. This VBLBALLABLBA HCT vehicles is their economical advantage. First of all, the transport costs decrease by 1.8-3.4%. Together with the environmental costs, that are lower compared to standard combinations, an overall benefit of HCT vehicles can be seen.[3]

2.3 Market overview for existing solutions

There are some solutions for dollys with steerable axles that exist today. However, the steering angles of the axles is only based on the articulation angle between the dolly and the trailer or truck it is attached to.
One of these solutions is the "active articulated dolly" manufactured by "Fahrzeugwerk Bernard Krone". The front axle of this dolly is steerable. In order to achieve that, the drawbar of the dolly is mechanically connected to the frontaxle and is rotateable. So if the drawbar is turned the wheels of the front axle steer. The transmission of the steering CAN be varied by changing the length of the drawbar, which is adjustable. For safety reasons the steering is mechanically locked automatically depending on the velocity. The main reason for implementing this steering capability is to reduce off-tracking. This dolly is equipped with air suspension, an electronic breaking system as well as a stability system. [14]
Other manufacturers that offer steerable dollys, that work the same way, are "Schmitz Cargobull" and "Kögel Trailer". [21][38]
In order to use the dolly to improve the vehicle dynamics of a combination in a more advanced way, a solution that allows more independently steering is preferable. The dolly used in this project is capable of steering both front and rear axles individually and independent of the articulation angle between the dolly and the trailer in front of it. In section 4.2 this system is described in detail.

2.4 Rapid Control Prototyping

In automotive software development standardized processes are applied to assure structured work. The V-model which derives its name from the characteristic shape constitutes one graphic representations of the sequence of steps towards the finished overall system. It also gives an approximate impression about the level of detail of each of the steps on the vertical axle, as well as chronological progress on the horizontal. A simplified version of this V-model can be seen in

figure 2.1. First the user or customer needs are analyzed and put into a schematic formulation of the system showing the logic dependencies and underlying physical working principles. This is usually done in a semi-standardized fashion utilizing Unified Markup Language (UML) or similiar diagrams, preventing non-specific prose to ensure less room for mis-communication and better readability. Technical decisions and limitations are not yet considered. After the logical system and dependencies have been established they will be analyzed and broken down to actual technical system descriptions defining also which subsystem will take over which function, what parts are to be realised as software or hardware (e.g. filtering, safety functions). The different parts of the technical system are then specified in detail, defining interfaces between different software sub-system, operational states and distribution of functions over sub-systems. The fourth step will define the specifications for the software in details for example: data types, control sequences, decision structures, real-time behaviour. Finally the actual design step will implement these specification with regards to the hardware limitations (RAM/ROM, processing power), computation methods, data handling (detailed format, variable types, utilization of parameters or variables). To compile the system from the component-level sub-system a similar hierachy is used in inverse. The corresponding steps of the project definition and specification phase (left side of the schematic) are supposed to provide test-procedures and goals. If necessary after insufficient test outcomes, another iteration of the previous step will be performed (vertical iterations). A similar sequence of development steps is applied to develop the hardware platform in parallel.[30]

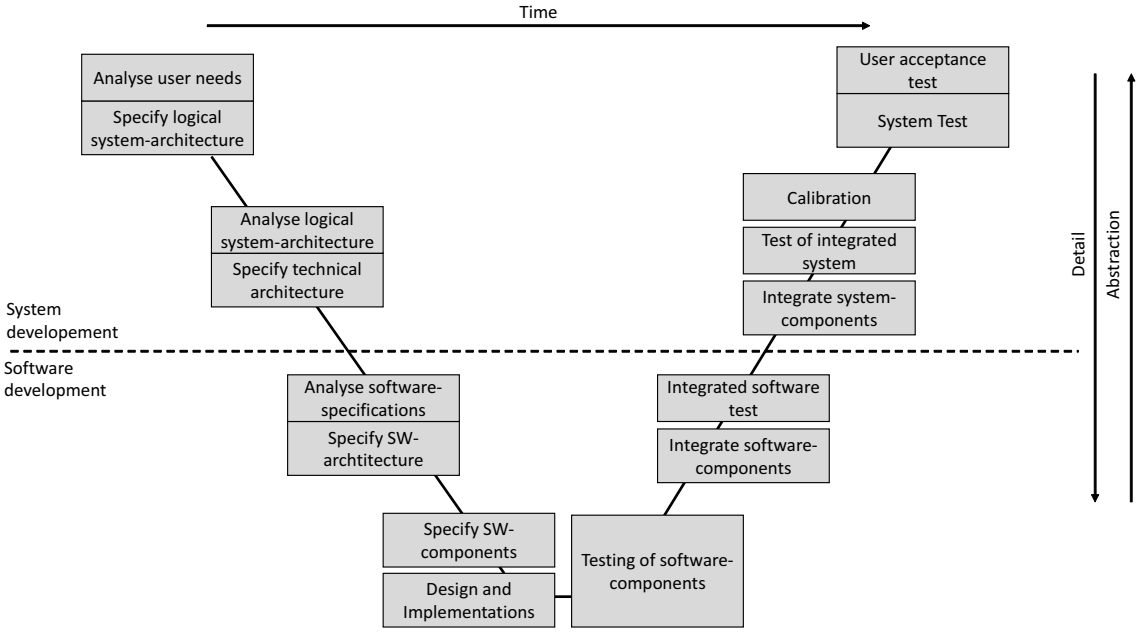


Figure 2.1: Overview for a process in automotive software development after the V-model

The increasing complexity of mechatronic systems and shorter product life-cycles lead to new development methods that made it possible to take 'short-cuts' in the established V-model process. This methods are summarized under the term rapid prototyping - in the case of software functions for mechatronic systems the term Rapid Control Prototyping (RCP) was coined. It is possible to conduct testing and verification with early soft- and hardware versions which are still under development by having the remaining system and environmental influences

simulated by the RCP-platform. In utilizing these methods such as rapid control prototyping and software/hardware-in-the-loop testing, a tremendous decrease in development time can be achieved. It is also possible to validate specifications early in the process, eliminating possible cost-intensive changes in later stages.[29] For the research project in which this theses is embedded it even opens up the possibility of on-road testing, as it eliminates the low-level development steps of the actual implementation and some of the detailed design work. The implementation of the steering algorithm (see section 3) on a stand-alone ECU, hydraulic actuator control, hardware layout, etc. would by far exceed the dimensions of a research project. Applying rapid control prototyping is the only feasible way to establish a functioning on-road prototype vessel.

- V-model
- real-time capability
- abstract high level model to programm ==¿ focus on function development and modelling, no low level coding needed
- time critical processes

2.5 Functionality architecture

As a basis for this project a functionality architecture developed at Volvo GTT was used. The different parts of this architecture are shown in Figure 2.2.

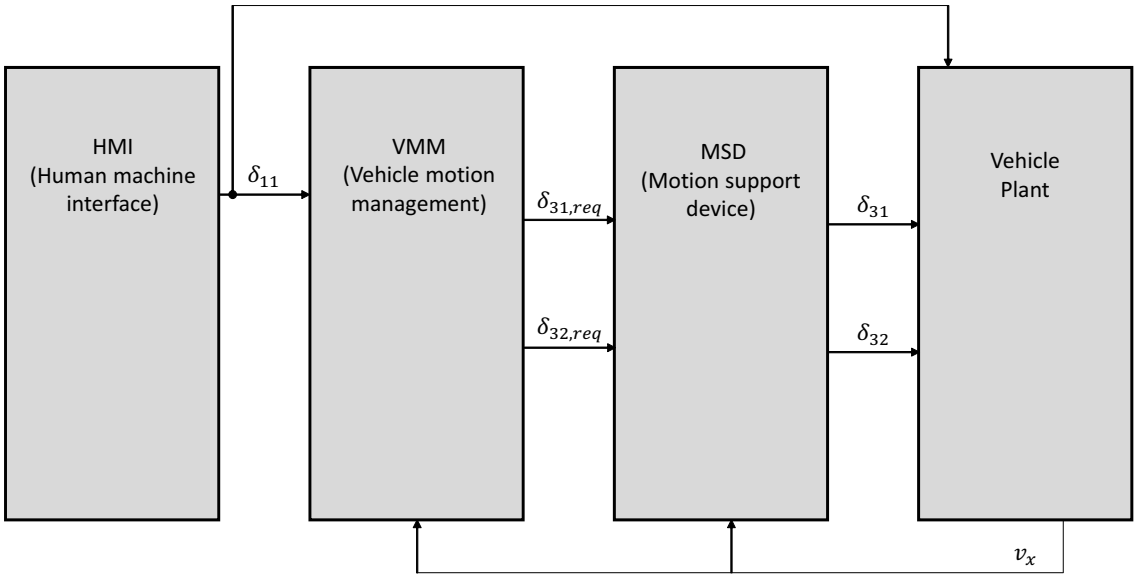


Figure 2.2: Functionality Architecture

The functionality architecture consists of four different parts: The Human machine interface, the vehicle motion management, the motion support devices and the vehicle plant.

The Human machine interface (HMI) is the interface between the driver and the vehicle. The driver can input requests to the vehicle, for example a steering angle via the steering wheel or a torque request via the accelerator pedal. The driver gets feedback from the vehicle about the vehicle's state, for example via the speedometer.

The vehicle motion management (VMM) contains the high-level controllers for the vehicle's motion and stability. It has an time horizon up to one second. As an input it gets the request of the driver from the HMI and transforms these requests into requests for the actuators of the vehicle. These request are then sent to the Motion support device (MSD). From the MSD the VMM receives information about the state of the actuators and the current capabilities.

The MSD contains the actuators and low-level controllers of the vehicle. In there the requests from the VMM are executed. Information about the status of the actuators is sent both to the VMM and the vehicle plant.

The vehicle plant corresponds to the rest of the vehicle.
In different kind of tests, the parts of the functionality architecture are either simulated or real hardware is used. The more advanced the development process is, more and more of the functionality is shifted from simulation to actual hardware.

3 Vehicle Model controllers (3 Seiten)

There are two different controllers used to control the steerable axles of the dolly. One is used for low-speed and one for high-speed maneuvers. The main purpose of the low-speed controller is the reduction of off-tracking. For the high-speed controller the aim is to reduce the rearward amplification.

3.1 Low-Speed controller [18]

The low-speed controller is implemented in path-distance domain η. The path-distance is calculated by integrating the forward speed $v_{x,1}(t)$.
The steering angle of the first axle of the dolly is calculated as follows:

$$\delta_{31}(\eta) = k_1\delta_{11}(\eta - \eta_1) + k_2\Delta\psi_1(\eta - \eta_2) + k_3\Delta\psi_2(\eta - \eta_3) + k_4\Delta\psi_3(\eta - \eta_4) \tag{3.1}$$

where δ_{31} is the steering angle of the dolly's front axle, δ_{11} is the steering angle of the tractor's front axle, $\Delta\psi_1$ the articulation angle between the tractor and the first semitrailer, $\Delta\psi_2$ the articulation angle between the first semitrailer and the dolly, $\Delta\psi_3$ the articulation angle between the dolly and the second semitrailer, η_1 the distance between the tractor's first axle and the dolly's first axle, η_2 the distance between the first articulation joint and the dolly's first axle, η_3 the distance between the second articulation joint and the dolly's first axle, η_4 the distance between the third articulation joint and the dolly's first axle, and k_1, k_2, k_3, k_4 are gains.
The steering angle of the second axle of the dolly is calculated using the steering angle of the first dolly axle as follows:

$$\delta_{32}(\eta) = k_5\delta_{31}(\eta - \eta_5) \tag{3.2}$$

where δ_{32} is the steering angle of the dolly's second axle, η_5 is the distance between the dolly's second and first axle, and k_5 is a gain.
The gains k_1, k_2, k_3, k_4, k_5 were optimized using the particle swarm optimization. Table 3.1 shows the values of the gains after the optimization and the improvement of the controller for different maneuvers.

Table 3.1: Control gains and performance improvement

		Improvement		
Gain	Value	180-deg turn (%)	90-deg turn (%)	S-turn
k_1	-0.2126			
k_2	-0.1567			
k_3	-0.8780	+44.61	+37.75	+45.07
k_4	-0.3143			
k_5	0.9868			

Figure 3.1 shows the implementation of the controller in the functional architecture.

3.1.1 Input parameters

The input parameters for the low-speed controller are:

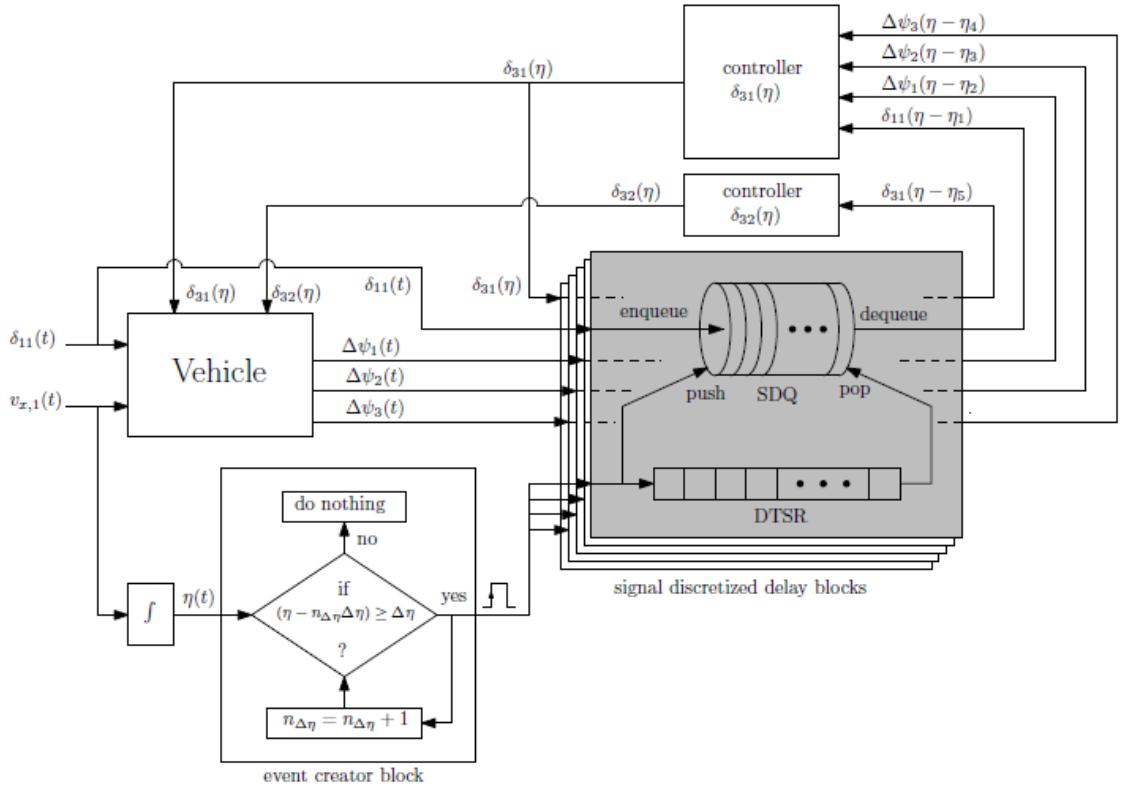


Figure 3.1: Controller implementation in the functional architecture [18]

- The articulation angles between the different units: $\Delta\psi_1$, $\Delta\psi_2$, $\Delta\psi_3$
- The steering angle of the tractor's first axle: δ_{11}
- The distances of the tractor's first axle, the articulation joints of the different units and the dolly's second axle to the dolly's first axle: $\eta_1, \eta_2, \eta_3, \eta_4, \eta_5$
- The vehicle speed for the transformation from time to path-distance domain: v_x

3.2 High-Speed controller [19]

The high-speed controller is implemented using a nonlinear inverse of a dynamic vehicle model. The aim of the controller is to copy the movement of the first unit to the following ones at a certain point. Therefore the movements of the following units need to be delayed by the time it takes them to reach this point. For the dolly unit, this delay τ_{13} is calculated with the vehicle speed v_x and the distance l_{cg13} between the Center of Gravity (CoG) of the tractor and the dolly as follows:

$$\tau_{13} = \frac{l_{cg13}}{v_x(t)} \tag{3.3}$$

With this delay and the lateral acceleration of the tractor a_{y1} a reference signal for the dolly's lateral acceleration is calculated as follows:

$$a_{y3ref}(t) = a_{y1}(t - \tau_{13}) \tag{3.4}$$

This reference signal, together with the steering angle of the tractor, the forward speed of the vehicle and the actual steering angle of the dolly, is used as inputs to an inverse single track model, that is modeled in the Modelica language. The outputs of the model are the request steering angle δ_{dolreq} and the lateral acceleration a_{y3} of the dolly. Figure 3.2 shows a block diagram of that model.

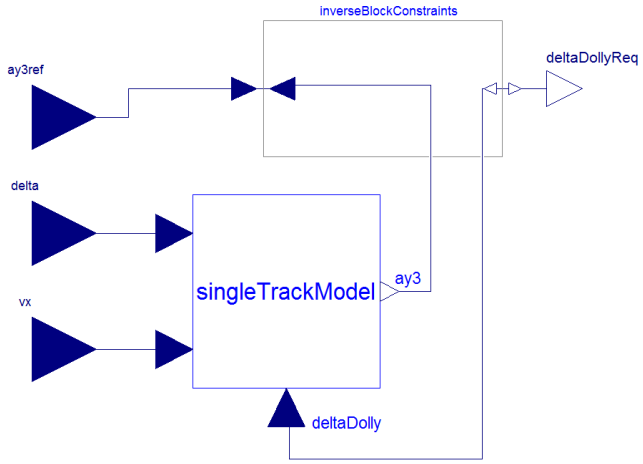


Figure 3.2: Inverse of the single track A-double model [19]

To determine the performance of this controller simulations of a single lane change at 20 meters per second and a sine wave steering input of 0.4 Hz with an amplitude of 0.026 rad were done. As a result the rearward amplification of the yaw rate of the second trailer is 0.95 with this controller compared to 1.49 for a passive dolly. Furthermore the transient off-tracking is improved by 67%.

3.2.1 Input parameters

Figure 3.3 shows the input and outputs of the inverse model controller.

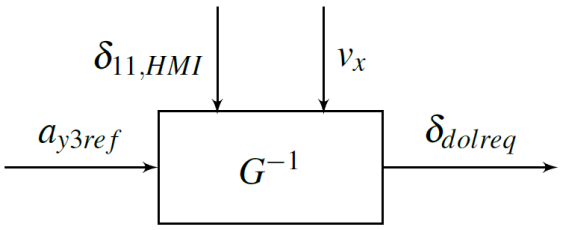


Figure 3.3: Inverse model controller[19]

The input parameters of the high-speed controller are:

- The vehicle speed: v_x

- The steering angle of the tractor's front axle: δ_{11}
- The reference signal of the lateral acceleration of the dolly: a_{y3ref}

3.3 Real-Time implementation of controllers

- this paragraph in close cooperation with MI
- what had to be changed to allow for MABII execution?
- how will feedback loop be handled?
- incorporate measurings?
- simulation step size?
- utilized computational method

3.4 Interface with Real-Time environment

As described in section 3.1 and 3.2 the controllers need different kind of input signals. These signals are received by the Simulink model running on the MABII via CAN. When the controllers are tested on a test track they are implemented in the same Simulink model, so they can get the necessary input signals directly within the model.

In order to guaranty safety at all time the steering angle requests of the controllers are checked before sending them to the dolly. This is done by first sending the requests to a supervisor block in the same Simulink model. The supervisor block compares the steering angle requests with the current capabilities of the system. The functionality of that block is described in detail in section 6.4.

4 Hardware architecture(9 Seiten)

4.1 Hardware overview

As already outlined in chapter 2 the HCT vehicles consists of a truck with two semi-trailers. To achieve the goals of this thesis, some modifications had to be done to the combination as well as equipment mounted in addition to the legacy fixtures. The utilized combination can be seen in figure 4.1, the shown number will be referenced further-on to give easier insight into the implemented systems' mounting points.

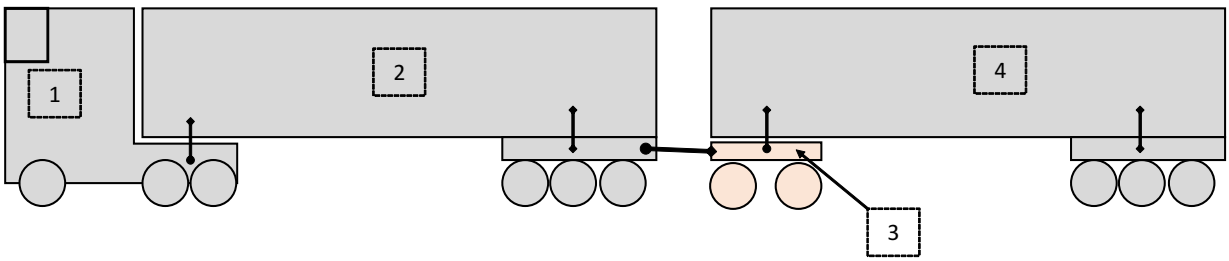


Figure 4.1: Positioning of different sensors on the HCT vehicle

The first semi-trailer's (number 2 in figure 4.1) rear needs to be equipped with a drawbar eye to accommodate the dolly's drawbar. Besides this obvious physical requirement power supply for the dolly and second trailer, pressurized air hoses for brake actuation, e-brake CAN-buses and lighting connectors have to be routed through the first trailer and supplied to the second trailer with a robust enough cable. This solution had to mainly be custom-made, as these appliances usually would not be available in the rear of a semi-trailer as usually no further towed unit is present. These installations would also apply for any passive combination or for the legacy V.S.E. Vehicle Systems Engineering B.V. (VSE) ETS system. Furthermore CAN networks have to be patched through the first semi-trailers (see section 4.5). The Real-Time system will be mounted on the dolly directly (position 3) to ensure short signal pathes and minimize wiring efforts. However the Host-PC will have to sit in the tractor (position 1) to sufficiently protect it against the track environment, which would be impossible if sitting on the dolly directly. This leads to an additional ethernet-cable for the Host-PC connection in the cabin which also needs to be routed through the first semi-trailer. The second semi-trailer does not require these modifications, as it is merely a passively towed unit.

Figure 4.1 also depicts another important aspect. Round bullets on the unit-connecting lines indicate a connection with one rotational degree of freedom (DOF) square bullets mean fixed connection. As CAN be gathered from the figure there are three DOF in the whole combination: between tractor and firstsemi-trailer, between first-semitrailer and the dolly, and between the dolly and the second semi-trailer. As the angle between the different units greatly influences the steering performance of combination, the angles have to be accounted for in the simulation and of course also the steering algorithm. It is thus necessary to measure them and provide the measurings via CAN to the rapid-prototyping system which is described in great detail in section 3.4 for the model-side and in section 4.5 for the hardware details. These articulation angle sensors also need to be wired from their mounting point to the dolly, where they are fed into the RapidPrototyping environment.

- give overview of system on truck
- sketch for where which unit sits (with numbers to be referenced)
- different tasks for Arduino
- make communication diagramm with all protocols (CAN, i2c, Serial Peripheral Interface (SPI) , serial, ethernet) <== this could also be moved to software chapter?

4.2 Utilized dolly system

The utilized dolly by Parator Industri AB (Parator) is equipped with two steerable axles. They are controlled by an after-market solution called ETS supplied by VSE, of which figure 4.2 gives an overview. Their product includes sensors, ECUs and hydraulic systems which come in a ready-to-mount housing, which is placed on the trailer/dolly. This solution is usually sold as a low-speed active steering system for truck-trailer combinations to provide better maneuverability at low speeds in inner-city areas. Besides electrical power and compressed-air (see "2" in the figure) supply there is no connection with the truck in place. This allows for use with many different truck/trailer Original Equipment Manufacturer (OEM), as no insight into proprietary Controler Area Network communication (CAN) is needed. In the original VSE system the two parameters that influence the actuation of the dolly's steering are vehicle speed ("4") and kingpin-deflection ("3"). This deflection is the angle between the truck and trailer, which is measured by an additional kingpin-angle sensor supplied by VSE and mounted in the eye of the kingpin hub. Furthermore every steering-knuckle of the dolly is equipped with an angle sensor to provide appropriate wheel-individual feedback for the VSE control-system. A diagnosis screen is available in the VSE-unit, which allows for relatively simple set-up, calibration and parametrization to be done.[7]

VSE provides assisted steering up to a speed of 25km/h over which intervention until it reaches zero at 55km/h. At this treshhold the steerable axles are locked and thus behave like normal rigid axles. This according to the manufacturer is to ensure stability at higher speeds.[7] Locking the steering at higher speeds leads to a more predictable behaviour for the user and system robustness. However, performance during high speed maneuvers CAN be improved by uniformly steering the dolly as well.[20] The desired demonstration of the algorithm presented in chapter 3 will as outlined in the introduction to this thesis, include steering at higher speeds,

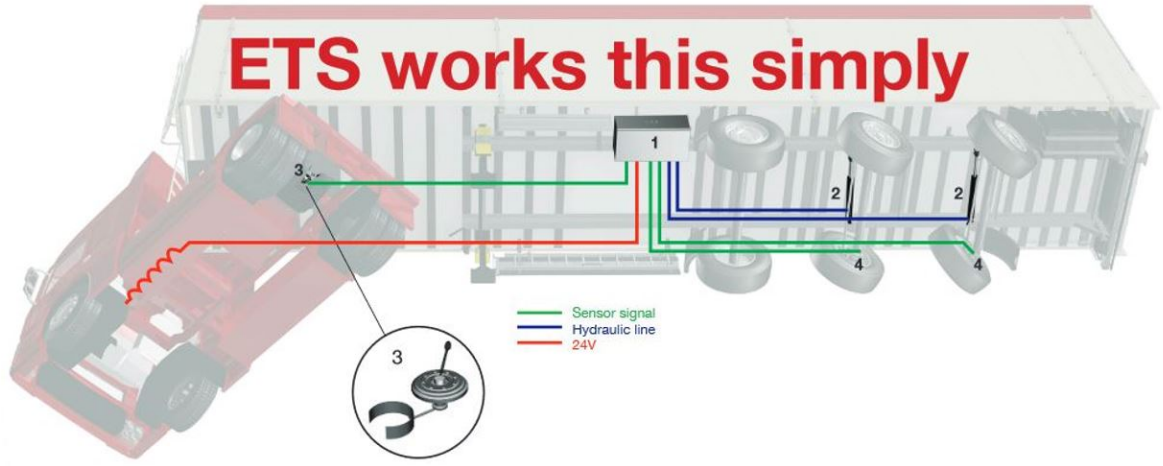


Figure 4.2: Active dolly legacy steering system supplied by VSE[7]

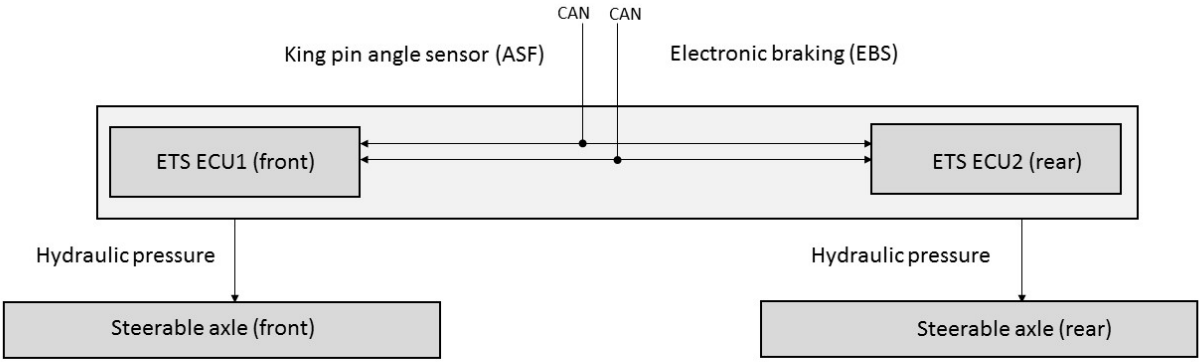


Figure 4.3: System overview for sensors and signal path for ETS legacy system by VSE

thus a work-around had to be established.

Short overview for the dolly, including:

- mechanical properties in short (weight, turning radius, max. tonnage)
- description of function (brake, steer, countersteer, lock at highspeed)
- difference low <=> high speed

- control system by VSE, diagnosis, display connection with truck
- system overview picture/schematics

4.3 Real-Time Environment

To run the utilized steering controller outlined in chapter 3, a MABII by dSpace company is utilized, an embedded computer running on a 900MHz PowerPC CPU with 32MB RAM. The MABII has very compact dimensions of only 200mm x 225 mm x 50mm making it possible to fit it inside the housing locker of the VSE ETS (position 3 in figuure 4.1). It accepts a wide variety of input currents for power supply, is shock resistant to a high degree and equipped with sturdy Swiss company specialized in industrial connector manufacturing (LEMO) connectors; the unit is cast into very robust aluminium housing and has mounting points, so it CAN be utilized in tougher conditions in automotive applications. It boots up almost at the speed of a standard ECU and can run headless ensuring safety, even without the Host-PC to control it. Besides normal digital and analog in- and output pins, CAN, Ethernet and many other standard interfaces are readily available, making for a practical use of these protocols on a high level.

The connection to the various CAN-buses for this thesis' purposes is established via the dSpace ZIF-connector. This is a specially made connector that allows to access all of the 64 in- and output-pins of the MABII and quickly plug and unplug the MABII Ethernet connections with reliable LEMO connectors are used to communicate with the Host-PC which runs logging and start-up sequences during testing. Power-supply is realized from the buffer batteries of the hydraulic steering system of the dolly during testing and a standard laptop charger during bench-testing.

The hardware platform is part of an integrated chain of software tools to develop functions and models in the sense of RapidControlPrototyping. It integrates with MATLAB's Simulink and compiles and uploads executable code of the abstract Simulink-models directly into the MABII's program memory. Furthermore a tool (ControlDesk 4.2) to log and manipulate the executed Simulation running on the MABII is used on the connected Host-PC. This software tools will be detailed in section 5.1 and 5.3.[13]

4.3.1 CAN-bus extension

The MABII comes with a preset number of in- and output ports. The maximum number of CAN-buses, that CAN be connected to the MABII's native controllers is limited to six. As there is no off-the-shelf solution by dSpace for extending the available bus-connections, it was necessary to come up with a gateway solution that allows to patch the needed amount of additional CAN-buses through to the ethernet connection which is also available on the MABII. It should be mentioned that this issue with lacking CAN-buses mainly arises due to the limitation, that buses have to be kept separate as shown in figure 4.6. Without this limitation it would of course be possible to just merge the different physical CAN-bus lines to one line.

A gateway from CAN-protocol to the standardized User Datagram Protocol (UDP) used for communication on ethernet infrastructure was implemented to achieve this. It is a very leight-weight protocol, that is straight-forward to implement and runs well, even with limited

processing power on a microcontroller. For future purposes the broadcasting capabilities of UDP might also proof useful, as many nodes could be connected to this CAN-to-UDP gateway all receiving the broadcasted datagrams for example for visualization on different computers or additional logging outside of the MABII environment. One limitation that was decided on, is to have receiving capabilities only for the MABII to eliminate the need for extensive computation and CAN-matrix handling on the Arduino.

Excursus: The CAN-protocol is the most widespread protocol to allow for communication between different ECUs in the automotive field. Development began at the Robert Bosch GmbH, but is now standardized and enhanced and adjusted for special purposes internationally. Messages are broadcasted by the bus-participants and stamped with their unique identifier, which also doubles as an arbitration token to handle message prioritization. Each bus-participant can listen to all available messages and "only picks from the bus what he needs". The standard message has a size of eight bytes à eight bit, transmitted after the identifier and followed by an ending sequence. Hardware-wise CAN-communication relies on only a pair of twisted wires, where a very robust voltage difference signal is transmitted. [8]

By utilizing the UDP-protocol to acquire data into the simulation, the MABII environment's very convenient possibilty to incorporate CAN database files (.dbc-file)[1], which is the usual way to exchange information and instructions for CAN-networks is no longer available. To decompose the UDP packets into the original signals it was necessary to implement the function of a .dbc-file in the underlying Simulink-model.

All messages that the CAN-gateway is supposed to handle and forward are put into one UDP frame in succession with respective CAN-identifiers and additional spacer bytes between signals. The UDP frame is broadcasted every time a new CAN-message reaches the gateway on one of its CAN-buses. Messages that were not updated with the incoming CAN-message will be kept at their previous value. This is called for, as UDP is a connection-less protocol without any loss-prevention mechanisms like acknowledgment-handling or retransmission of messages. Holding the values if not available ensures, that at least some value is available on the bus. UDP also doesn't have native timestamping, which is why a Global Positioning System (GPS)-shield was introduced to receive the high-precision time to allow for easy synchronizing of logging over different platforms (MABII-environment, IMUs, external CAN) by adding the global time to each send packet.

To allow for easier scalability to add more CAN-buses in the future it was decided to not use the Due's two native CAN-controllers but rely on the external MCP2515 in combination with the already mentioned transceiver (MCP2551). As shown in figure 4.4 it is controlled via SPI . More bus-participants CAN be added conveniently by incorporating additional instances of these two microships and attaching them to a device selection pin (depicted as SS) each.

- daten-konvertierung und packaging zu UDP und von UDP zu Daten in CD!!
- verwendete bibliotheken

[1]Correlates physical human readable/understable signals with units to the actual distribution over the different bytes of a CAN-frame. It allows to "decrypt" the information which is available on a CAN-bus and allows to code and send messages in the respective format that the other bus-participants (ECUs, sensors) expect.

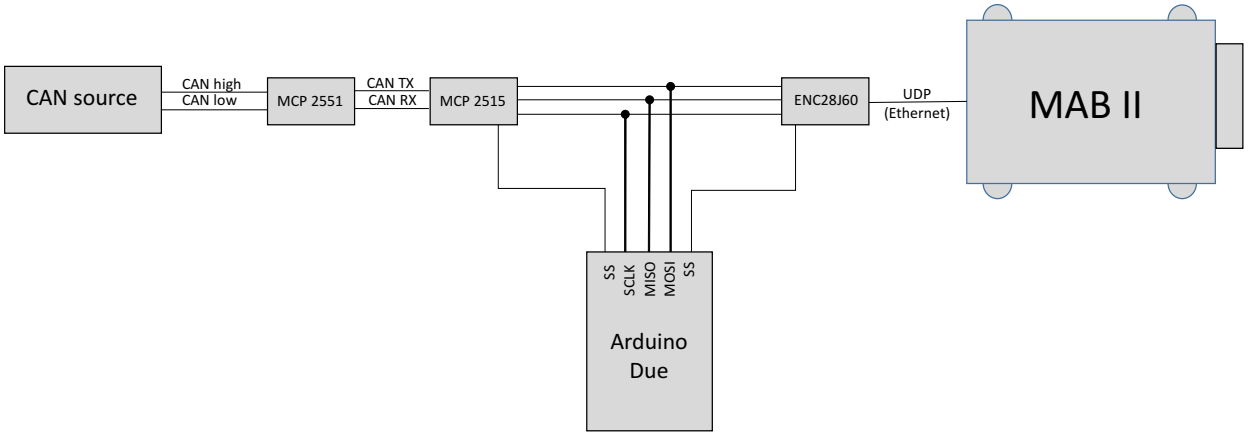


Figure 4.4: Extension of MABII CAN-buses (hardware overview)

4.4 Interfaces and connections with dolly

Figure 4.5 gives an overview of all the in- and outputs of the dolly.

To the first semitrailer the dolly is connected with a standard 7-pin ISO 7638-2 connector. This connector includes 24 V electrical power, a signal for Anti-lock braking system (ABS) fault detection as well as the ISO 11992 CAN. There's a second connector, which includes different pins for different kinds of lighting of the dolly, like the turn indicators and brake light. The dolly also receives compressed air from the first semitrailer. For controlling the MABII an Ethernet connection is established between the dolly and the tractor unit, where a laptop is located. The interfaces of the dolly with the second semitrailer are the same as those to the first one. To control and measure signals of the dolly a connection to the dolly's ETS- and Electronic Brake System (EBS)-system had to be established. To achieve this, different kinds of CAN-buses had to be used. That was necessary because the various systems of the dolly use different CAN-protocols. In total five CAN-buses connect the MABII with the dolly. Two for each of the ETS-ECUs and one for the Angular Sensor Fusion (ASF)- and EBS-Signal.

One of the CAN-buses contains the signals from the EBS-ECUs and the articulation angle sensor, which measures the angle between the dolly and the first semitrailer. It uses the extended CAN-standard. In the original state this CAN is directly routed to the CAN of the two ETS-ECUs (see section 4.2). In order to achieve steering that is both independently from the articulation angle as well as the velocity of the vehicle the connection between the ASF/EBS-CAN and the CAN of the ETS-glsECU was cut. Instead the ASF/EBS-CAN is connected to the MABII. Figure 4.6 shows an illustration of that.

The MABII is connected to the CAN bus of each ETS-ECU. Because of the use of different

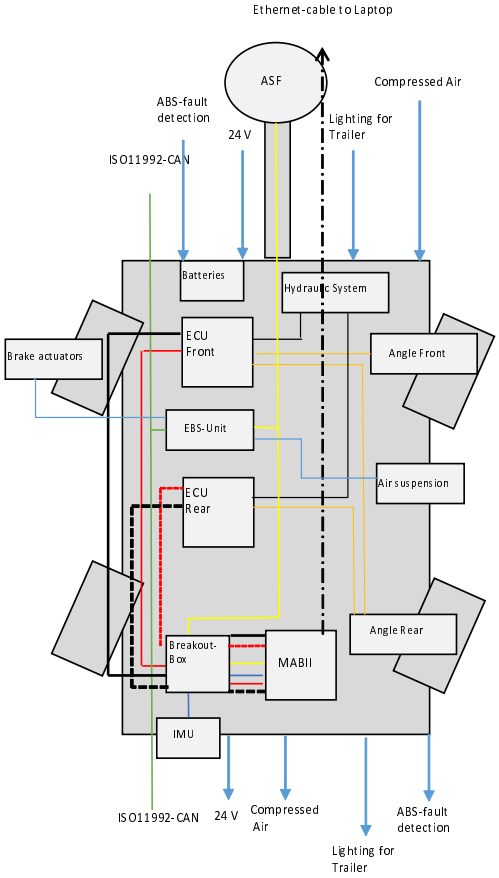


Figure 4.5: Dolly Interfaces

CAN standards of the ASF/EBS-CAN, which uses the extended CAN standard, and the ETS-ECU CAN, which uses the J1939 protocol, two physical CAN ports are needed on the MABII for each ETS-ECU. Those two CANs are merged inside the breakout-box. Figure 4.7 shows the breakout box which is used to as an interface between the different CANs that were used and the MABII. The merging of the two CANs that contain signals for the ETS-ECU can be seen on the right side of the scheme.

Furthermore there is the ISO 11992 CAN bus which is used for communication between tractor and the attached semitrailers. In the current design this CAN is only connected to the EBS-units of the dolly. For future projects it is preferable to connect the MABII to that CAN, too. If the connection is established and a matching dbc-file for the ISO 11992 standard is acquired, there would be different benefits. For one thing more information about the state of the trailers, like for example axle loads and wheel speeds, could be received and used to improve quality of the capability-state of the dolly (see section 6.3), for another thing there would be the possibility to also control breaking and the leveling of the air suspension of the dolly. With this a more sophisticated control of the vehicle dynamics of the dolly CAN be achieved.
Another possibility to get control over the braking and leveling system could be over the diagnosis port of the EBS. This port is usually used for diagnostic purposes. With the matching

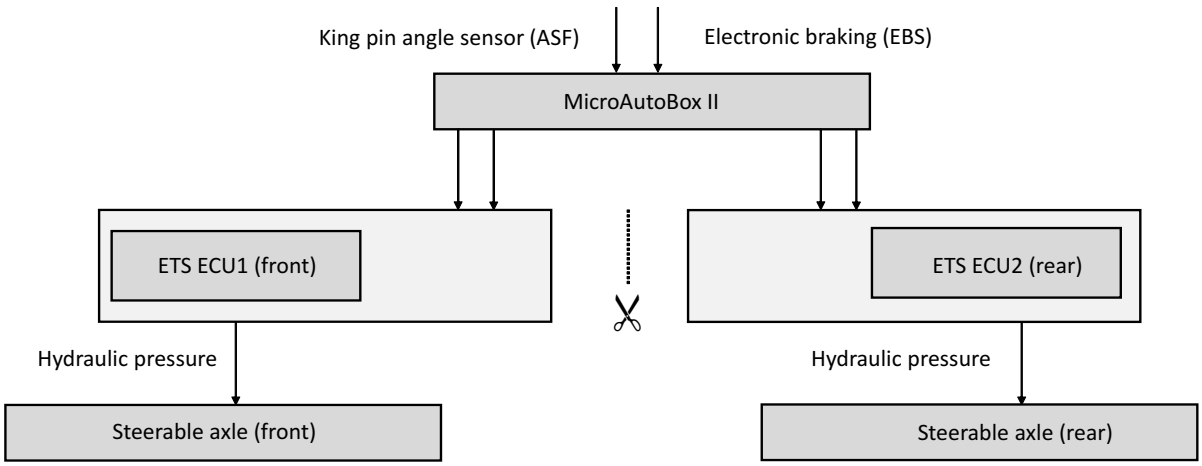


Figure 4.6: Split of ETS-CAN

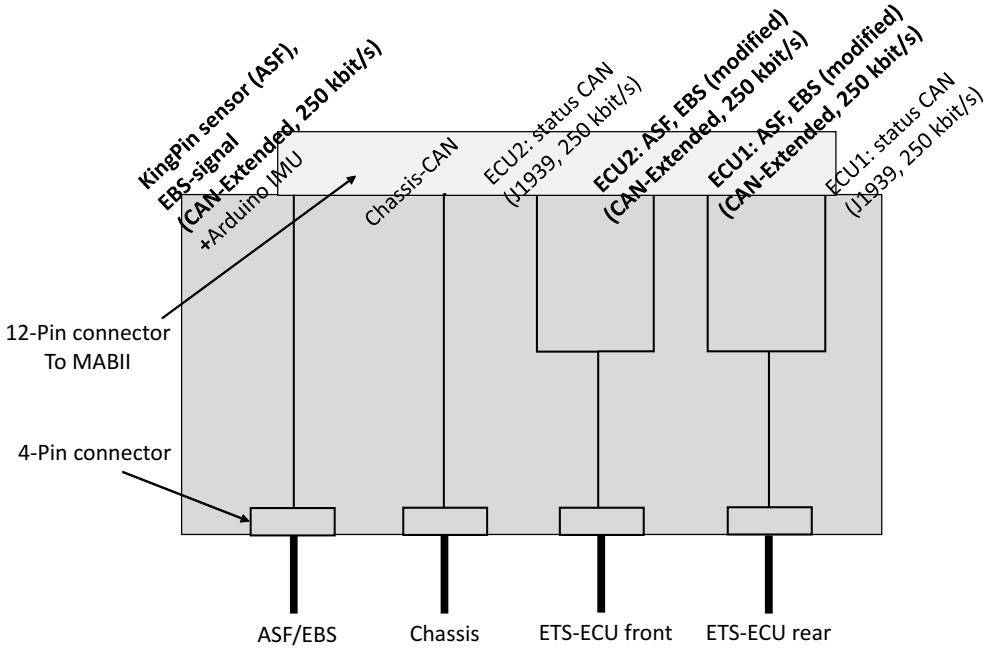


Figure 4.7: Breakout box scheme

diagnostic software the braking and leveling can be controlled manually. If the EBS supplier provides insight into how the diagnostic software communicates with the EBS unit, this port could be used to access the EBS units and the brake and leveling control could be implemented.

- **distinguish between HiL /bench-testing and track-testing!!**

4.5 Interfaces with truck

The vehicle model controllers depend on inputs from the tractor unit(see section **??**). One of the inputs is the steering angle of the tractors front axle, another the current vehicle speed. For track-testing the MABII is connected to the Chassis CAN of the tractor, in order to get those signals. Since the chassis CAN of the tractor is not usually available on trailers, the chassis CAN needs to be accessed somewhere in the tractor and looped through all the way to the dolly. For the HiL -testing the tractor unit is simulated using Virtual Truck Model (VTM) (see section 5.1.2). As already mentioned in section 4.4 the dolly gets power supply and compressed air from the tractor.

- CAN communication with truck
- steering system conncetion (Volvo Dynamic Steering (VDS)!)
- physical CAN path to dolly?

4.6 Measurment Setup

4.6.1 On-board sensors

Sensor	**Measurings**	**Type**	**Mounting point**
Draw bar sensor	Kingpin angle Temperature	External	TBD!!!
EBS	wheel speed	External	wheel brakes
Steering angle sensor	steering angle (first, second axle)	External	steering knuckles

Table 4.1: Available sensors of VSE's ETS

4.6.2 Inertial measurement unit

To determine the processing delays in the control chain (refer to chapter 7.3) as well as logging implementation for verification and analyses purposes a number of inertial measurement units (IMU) where utilizied throughout this thesis' work. The system at hand combined a gyroscope (L3GD20H), and an accelero- and magnetometer (LSM303D) into an IMU put on one circuit board.[26] This one-chip solution allowed for a convenient access to the sensor measurings, as the sensor outputs could be received via Inter-Integrated Circuit-protocol (I^2C) which eliminates the need for a transducer. Furthermore a high-pass filter is integrated into the IMU's accelerometer, which leads to simpler compensation of the immanent drift of the magnetormeter. These units supply the measurings for three axes each at a maximum frequency of 1600Hz for the accelerometer and 757.6Hz for the gyroscope. [33][32]

The IMU-chip was put in a suitable plastic housing and cast into resin, to fixate the IMU securily and prevent movement relative to the truck-frame on which it will is mounted as well as ensuring water-proofing.

Figure 4.8 shows, how the IMU is connected to the Arduino Due utilizing the Inter-Integrated Circuit (I^2C)-interface on the Due. Not pictured are the power supplies (both 5V) for the MCP2515 CAN-transceiver and the IMU itself. To minimize cables that have to be rooted through the combination and avoid problems with cable length for the power cables, it was decided to equip each of these assemblies with a rechargeable battery-pack. This also allows to measure in remote location and collect data locally on the Arduino Due during testing, eliminating the need for CAN-communication completely if needed in the future.

A maximum of two IMUs CAN run on one I^2C line as it is possible to override the last bit of the otherwise hard-coded I^2C address via a hardware jumper. If both I^2C lines of the Due are used, it would be possible to connect up to four IMUs to one Arduino. Though eight IMUs are desired for the whole HCT vehicle, this was outruled as a possibility for this project, due to the limitations of the physical bus-length of the I^2C lines of only a few meters (as the name already suggests I^2C is not meant for long distance communication). As CAN also be seen in the overview figure of the setup on the truck in figure 4.1 this distance would be exceeded with the distance between the different mounting points (15-20m).

In the final measuring setup the assembly shown in figure 4.8 will be present four times in the whole combination, to gather motion data from each independently moving part. Figure 4.1 shows those placements in an overview sketch. The correct placement will be as close as possible to the CoG of the respective unit to have as little influence of the vehicle's rotary motions as possible, which drastically reduces work in data analysis further on. CoG locations CAN be gathered from Volvo's internal database and are available at the test site. Putting two IMUs in each spot allows for averaging and ensure redundancy should one IMU chip fail or produce corrupt outputs, which occured from time to time during the bench-testing of the assembly if the voltage of the preliminary power-supply dropped due to other loads.

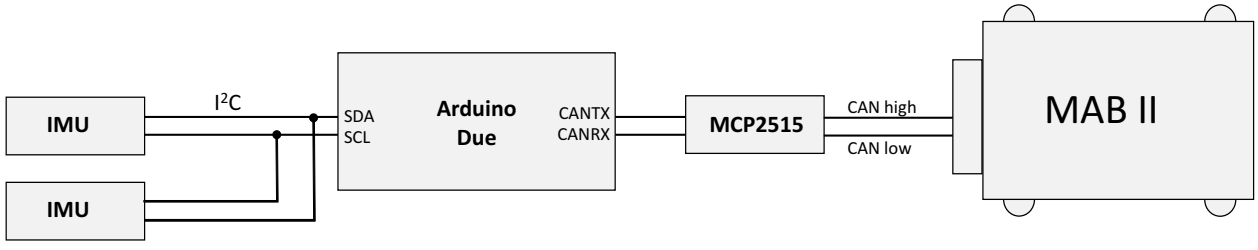


Figure 4.8: Connection of IMU with Arudino Due to MABII

4.6.3 Arduino Due

The Arduino platform series was developed by SmartProjects company in Italy and consists of a micro-controller board with several in- and output pins, and a set of programming tools to program said micro-controller (see also section 5.4). Hardware layouts as well as software of this solution are released under an open-source licence, so changes at any level of detail can be made and many other suppliers are now offering solution that tie in with the Arduino. Furthermore this lead to a broad user base and availability of vast supporting resources. For a number of tasks in this thesis, it was decided to rely on this cost-efficient and flexible microcontroller platform. Namely those purposes are:

- data collection from IMU and transmission over CAN
- CAN-to-UDP gateway
- CAN-to-Serial gateway for HiL -testing
- benchtesting sensor outputs and quick CAN-debugging

SmartProjects offers various Arduino boards mainly differing in number of communication pins, memory size, clock frequency and physical size. It was decided to utilize the to date most powerful Arduino Due, offering 84 MHz of clock-speed on a ARM Cortex-M3 processor, 512kB flash memory for programm-code and a 96kB working SRAM. It is the first Arduino with 32-bit architecture. This was done in order to allow for future use of this platform in other projects and having enough space for different sub-programs on the controller as well as enough processing power to deal with basic signal filtering and parsing at appropriate speeds.

The Due has a native I^2C interface, which was used to gather the measurings from the IMU. It also has a SPI which is needed to access the ethernet controller necessary for the CAN-UDP-gateway (refer to section 4.3.1), serial communication is also handled on hardware level. The availability of these ports in hardware form allow for robust systems and eliminate the need to implement those protocols on software level (bit-banging), which frees up memory resources. In addition to this digital communication possibilities the Due offers an abundance of 54 digital and 12 analog freely configurable I/O-pins. The Due is also the first Arduino to host an on-board CAN-controller, which in this thesis will be used for measurement transfer to the logging system (MABII), eliminating the need for additional hardware for CAN-bus interfacing on the Arduino side.

Besides the power-supply and the IMU-chip a MCP2551 CAN-transceiver by Microchip Technology Inc. was incorporated to take care of the pyhsical layer of CAN-bus communication by converting the digital signals from the Due's CAN-controller to the standardized voltage levels of the CAN. The MCP2551 is capable of different CAN standards and fully ISO-11898 compatible, which makes future use in different environments or as unit for in-vehicle CAN-interfacing feasible.

The Arduino Due was used to determine the delays in the system (see section 7.3.2) as well as a light-weight solution to quickly read CAN-outputs of the various systems during this thesis' work. This proofed to be an easy debugging solution. Additionally it was used to trigger some error codes to test the developed safety mechanisms (see section 7.2.3).

5 Software Architecture (8 Seiten)

5.1 Matlab/Simulink environment

5.1.1 dSpace RTI-blockset

The dSpace RTI-Blockset (Real-time interface) is a plug-in for MATLAB/Simulink that allows you to connect a simulink-model to the different inputs and outputs of the MABII There are Real-Time Interface (RTI)-blocks for CAN, Ethernet, and the analog and digital outputs of the MABII In this project the RTI CAN MultiMessage and the RTI Ethernet (UDP) Blocksets were used.

RTI CAN MultiMessage Blockset

The RTI CAN MultiMessage blocks establish an interface between the physical CAN-Buses of the MABII and the simulink-model running on the MABII There are four different blocks in this blockset:

In the "GeneralSetup" block the paths of the model root and the destination folder for generated files are set.

In the "ControllerSetup" block first the name of the controller has to be set. After that the physical CAN-Bus, that should be used, is set by choosing a module number and a controller number. How the module- and controller numbers have to be set for the different CAN-buses of the MABII is shown in 5.1.

Table 5.1: CAN-layout MABII

CAN	Module number	Controller number	ZIF -Pin CAN-High	ZIF -Pin CAN-Low
CAN1	1	1	c2	c3
CAN2	1	2	b2	b3
CAN3	2	1	B2	B3
CAN4	2	2	A2	A3
CAN5	3	1	P2	P3
CAN6	3	2	N2	N3

After setting the module- and controller number the identifier format has to be set to either standard or extended format, the transceiver type must be chosen between ISO11898-2 and ISO11898-6. ISO11898-2 is used for a high-speed medium access unit and ISO11989-6 for the selective wake-up functionality of a high-speed medium access unit. If needed, a termination resistance of 120 Ohms can be set in the block as well. As a last step the Baud rate of the CAN-bus has to be defined.

In the "Main Block" block a dbc-file is connected to one of the controller blocks, that were created before. A "ControllerSetup" block CAN only be connected to one "Main Block" at a time. When the dbc-file is loaded in the "Main Block", the different messages and signals that are specified in the dbc-file can be chosen as inputs and/or outputs of the "Main block". For signals that should be transmitted it has to be defined when they should be transmitted. There are different ways to specify this. One way is to set a cycle time in the simulink model,

that determines with which frequency the signal is sent out. Another way is to use triggers that trigger the transmission every time a specific event happens. This CAN be set in the tab "Messages" under "Triggering".

RTI Ethernet (UDP) Blockset

The RTI Ethernet (UDP) Blockset was used for two purposes: On the one hand for the HiL -testing (see section 7.5), on the other hand for the CAN-bus extension of the MABII (see section 5.1.3). The RTI Ethernet (UDP) Blockset contains four different blocks:
The "Ethernet_UDP_SETUP" block consists of two pages. In the "Unit" page the main settings for the Ethernet are made. First the interface name has to be chosen. Then the board type has to be selected. There are two types of boards: "ETH Type 1" and "ECU Type 1 ETH". If the Ethernet cable is connected to the Ethernet port of the MABII, "ETH Type 1" has to be chosen. The module number, which has to be set in the next step, must be set to 1 since the used MABII only has one module. After that, the local Internet Protocol (IP) address has to be set.
On the options page of the setup block up to four different sockets CAN be enabled. For each socket a local port number, the remote ip address and the remote port number have to be set.

The "Ethernet_UDP_RX" block is used to receive data over Ethernet. To set up this block the board type, module number and socket number need to be set corresponding to the settings that were made in the "Ethernet UDP Setup" block. Furthermore the maximum message size CAN be defined. Outputs of this block are the received data, the message size and a status. The datatype of the received data is uint32. If the data wasn't transmitted as uint32 it needs to be decoded. In the demo for the Ethernet blocks a block called "DSDecode32" CAN be found. With this block the decoding CAN be done, by choosing the desired output data format. To get the single signals the output of that block just needs to be demuxed. If the data type of the transmitted data was already uint32 the signal CAN be demuxed right away.

The "Ethernet_UDP_TX" block is used to send data over Ethernet. This CAN be done in a similar way as for the receive block. The only difference is that in the transmit block additionally a send timeout CAN be set. The maximum message size must match the port width of the data that is send out. It CAN be calculated by:

$$data\ port\ width = (MaxMessageSize + 3)/4 \tag{5.1}$$

The inputs to the transmit block are the data to transmit, the message size and a constant to enable the transmission. The data format of the data to transmit needs to be uint32. Similar to the receive block a so called "DSEncode" block CAN be found in the demo for this blockset. This block encodes muxed signals of different types in a way, that they CAN be transmitted with the transmit block. In the "DSEncode" block the data type of the different signals, that are put in the block, need to be specified.

The "Ethernet_UDP_INTERRUPT" CAN be used to make the hardware interrupts available as trigger sources.

5.1.2 Virtual Truck Modelling Library

The VTM is a computational framework used within Volvo Trucks to simulate the dynamic behaviours of trucks and combinations. As a library it extends Simulink, where maneuvers,

track layout and the trucks kinematic and dynamic properties are linked together and then computed. This toolbox was used as a base for the simulation of the LVC including the dolly for this project. It is possible to run simulation offline with a predefined maneuver and a given environment as well online where these parameters are fed into the calculation live or as measurings from the real-world environment. For online use all relevant parameters and states CAN be accessed in Simulink or in case of execution on the MABII through ControlDesk as their respective representation of the Simulink variable.

5.1.3 CAN-bus extension

- outline data diagram structure
- show limitations uf UDP (time, reliability, retrans)
- timestamping

5.2 Steering interface

The steering interface block constitutes the interface between the vehicle model controller and the different CANs of the dolly. It is a Simulink model that runs on the MABII Figure 5.1 shows the steering interface block with its in- and outputs.

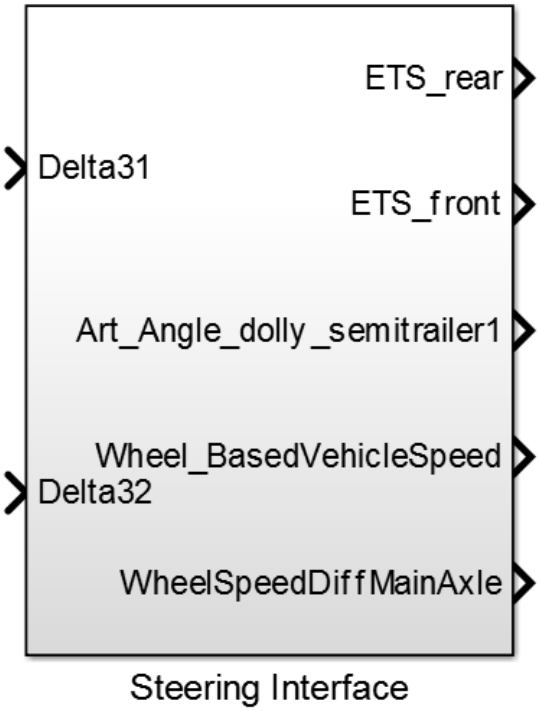


Figure 5.1: Steering interface block

The inputs to the block are:

- Steering angle requests for front axle of the dolly

- Steering angle requests for rear axle of the dolly

The outputs of the block are:

- All signals on the ETS-CAN for both front and rear axle
- The actual articulation angle between the dolly and the first semitrailer
- The vehicle speed signal from the dolly's EBS
- The wheel speed difference from the dolly's EBS

The outputs are used to monitor the dolly's state and for the calculation of the current capabilities (see section 6.3). Figure **??** shows the inside of the steering interface block. It contains RTI CAN MultiMessage blocks (see section **??**).

The software CAN-layout corresponds to the hardware CAN-layout described in section 4.4. There are five different CAN-controllers. Controller1 is used to receive the signals of the ASF-Sensor and the dolly's EBS system. These signals, except for the "Angle", "Wheel_BasedVehicleSpeed" and the "WheelSpeedDiffMainAxle" signals are forwarded to Controller2 and Controller3. Controller2 corresponds to the ETS-CAN of the front axle and Controller3 to the one of the rear axle. The inputs Delta31 and Delta32 are sent to the front ETS-CAN (Controller2) and rear ETS-CAN (Controller3), respectively. For the signals "Wheel_BasedVehicleSpeed" and "WheelSpeedDiffMainAxle" constants with the value zero are forwarded both to Controller2 and Controller3. Controller4 and Controller5 also correspond to the front ETS-CAN and rear ETS-CAN, respectively. As mentioned in section 4.4 this is necessary because different CAN-protocols are used. The signals received from those Controllers, together with the signals from Controller1, that weren't forwarded, are outputs of the steering interface block.

5.3 ControlDesk monitoring environment

The software used to monitor the Simulink model running on the MABII and for logging of data is called dSpace ControlDesk. It also provides the possibility to control the Simulink model running on the MABII, for example by changing parameters like constants in the model during run-time. Therefore it was used for bench-testing the dolly system (see section 7.2) to directly send steering angle request to the ETS-CAN via the steering interface (section 5.2). In ControlDesk the CANs that the MABII is connected to can be monitored as well. This proofed very useful for checking if all systems work correctly during early stages of development.

5.3.1 Maneuver control

Though as mentioned in section 4.3 the MABII is capable of running headless and it would be possible to execute the maneuvers fully independently on start-up, it was decided to rely on manual maneuver triggering to have an extra level of safety. For controlled testing environments with clearly delimited maneuvers, the structure shown in figure 5.3 was introduced and allows to either command a steering request of zero to the dolly or patch through the calculated steering angle that is computed by the steering controller and thus starting the maneuver.

Sending a steering request of zero results in the dolly behaving like a passive un-steered dolly, which is also the fall-back level implemented in hardware in the legacy ETS system. The constant block in the depicted system can be changed manually in ControlDesk to use this as a maneuver start-switch, but of course can also be accessed by the model itself. The supervisor-block can thus shutdown the steering completely on software level, preventing the triggering of the hardware-level error-mode. the hardware implementation moves the hydraulic piston of the steering system back to the middle position, if pressure is not present anymore. In doing so it enters error-mode, which needs to be reset by turning the truck's ignition off and on again. Software shutdown within the environment developed in this thesis, allows reverting back from locked mode to steering mode.

5.3.2 Monitoring and logging

- data-format
- frequency
- synchronizing over different CANs

5.4 Arduino IDE and applications

The Arduino system provides an Integrated Development Environment (IDE) written in Java, providing cross-platform support. It is used to develop the code as well as compiling the code and subsequently uploading it into the microcontroller via the computers serial interface. Within the IDE it is also possible to load some of the officially supported libraries directly. It conveniently is possible to monitor the computer's serial interface as well, which is the most practical way to monitor and debug code that is executed on the Arduino.

Library name	**Purpose**	**Comment**
LSM303	read magnetometer on IMU via I2C	[28]
L3G	read gyro & accelerometer on IMU via I2C	[27]
UIPethernet	control ENC28J60 via SPI	[24]
TinyGPSPlus	acquire and parse GPS signal from EM-506 via serial	v0.94b[22]
mcp_can	implement CAN via MCP2515 and MCP2551 via SPI	[12]
due_can	implement CAN via MCP2515 using Due's CAN-transceiver	[11]

Table 5.2: List of utilized libraries on the Arduino platform

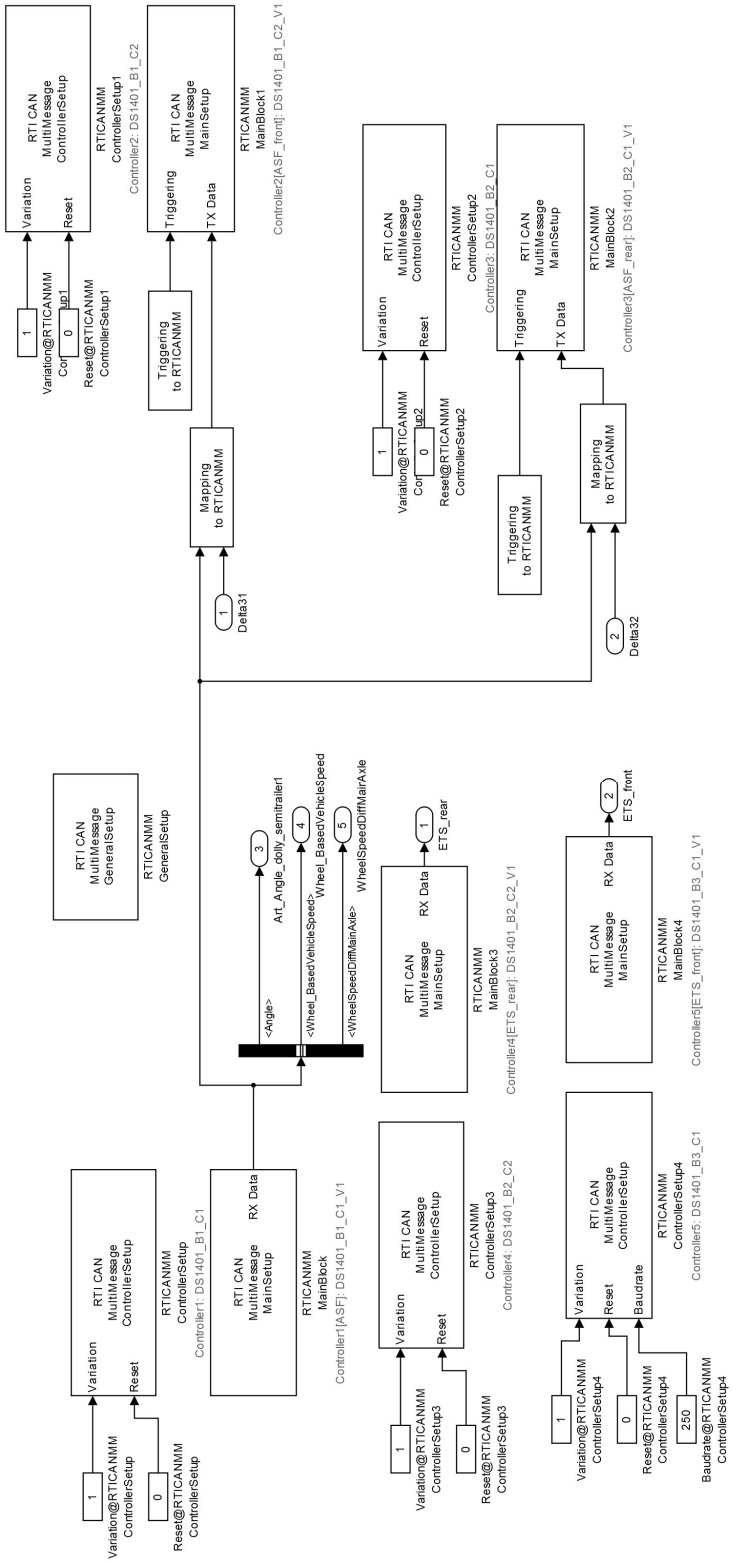


Figure 5.2: Steering interface block CAN layout

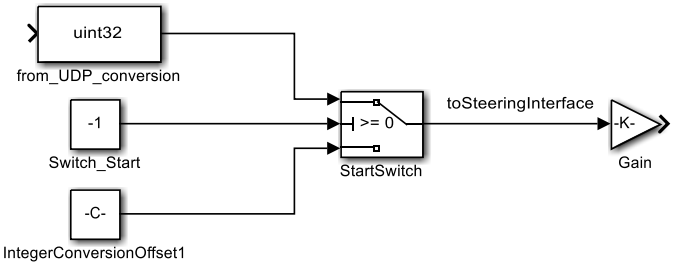


Figure 5.3: Trigger to start maneuver or fall back to safe-mode (e.i commanding a steering angle of zero degree on the dolly); this structure was used in HiL-testing

6 Fault detection and system ability (5 Seiten)

6.1 FMEA

FMEAs were developed by the US military in the late 1940s and later also used by NASA in the Apollo project in the 1960s. A FMEA is used to detect possible failures before they appear. Therefore the FMEA is done in a early stage of a project in order to be able to take the results of the FMEA into consideration when developing a system. In the process every possible failure of a system are taken into consideration.

In a first step a block diagram of the system with all of its inputs, outputs and subsytems was created to gain a complete understanding of the system. Every subsystem was broken down to the lowest level and for those subsystems block diagrams with all the components, inputs and outputs were created as well.

Starting from the block diagrams all potential failure modes were determined for every component of the system respectively subsystem. As a next step the potential effects of these failure modes were determined and the severity of these effects were evaluated on a scale from one to ten, with 1 being the lowest severity and ten being the highest severity. For the evaluation a table that shows how different severities correlate with the numbers was used (see Table A.1). Following this, potential causes for every failure mode were defined. Then the probability of occurrence for each cause was evaluated, also using a scale from one to ten (Table A.2). After that the current control mechanisms, that detect the failure when it should appear, were listed for each failure mechanism and the detectability of the failure mechanism was evaluated using a scale from one to ten (Table A.3). After the severity, probability and detectability were evaluated, a Risk Priority Number (RPN) is calculated as follows:

$$RPN = severity * probability * detectability$$

This RPN is used to identify the failure mechanisms that need to be addressed. After the RPN for every causes is calculated, the RPN are ranked from highest to lowest using a Pareto diagram. The causes with the highest RPN need to be addressed first. Special attention needs to be paid to cases where a high severity number, which means 9 or 10, occurred, even if the RPN isn't as high as for other cases.

In order to lower the RPN for those cases actions need to be decided, that either lower the severity, probability or detectability of them or more than one of those factors. After the performance of these actions the severity, probability and detectability are evaluated again and a new RPN is calculated. If the RPN still is to high a iteration of the process needs to be done. [9] Figure 6.1 shows a screen shot of the FMEA-Excel-worksheet for the first iteration. The consequences of the FMEA conducted for this thesis were:

- Special attention needs to be paid to the mounting of the MABII in the Dolly's ETS-box
- The design of the breakout box was revised
- Bench- and HiL -testing is necessary to validate the vehicle model controllers and to exclude dangerous behavior, that might be caused by delays in the system.
- A start-up checklist for the track testing needs to be created

Item / Function	Potential Failure Mode(s)	Potential Effect(s) of Failure	Sev	Potential Cause(s)/ Mechanism(s) of Failure	Prob	Current Design Controls	Det	RPN	Recommended Action(s)	Responsibility & Target Completion Date	Actions Taken	Nev Sev	Nev Occ	Nev Det	Nev RPN
CAN-Connection															
	No Connection														
		No/Wrong Steering Angle	10	MABII unplugged	2	Check of Connection in ControlDesk	1	20							
			10	Failure of breakout-box	6		5	300							
			10	ECU unplugged	1		1	10							
Signal Angle_front															
	No Signal														
		Wrong Steering Angle	10	Connection problem	1	Error Code in Steering-ECU	1	10							
			10	Failure of Sensor	2	Error Code in Steering-ECU	1	20							
			10	ECU unplugged	1		1	10							
	Wrong Signal														
		Wrong Steering Angle	10	Failure of Sensor	2	Comparison of front/rear sensors	1	20							
	Different														

Figure 6.1: Example of FMEA-worksheet for the first step of the FMEA

6.2 Safety concepts

6.3 Maximum capabilities of the system

The capabilities of the system CAN be divided into two parts. For one thing there are the capabilities that are independent from influences. These are the maximum steering angle and maximum steering rate, which is limited through the ETS-ECU, because of the physical limitation of the hydraulic system. For another thing there are the capabilities that are influenced by the vehicle speed, load of the dolly, state of the hydraulic and electrical system of the dolly, and environmental influences. Depending on those parameters the capabilities are lowered.

- give indicator of maximum angle/angle rate
- describe "algorithm"/lookuptable
- explain underlying physical correlation (ref to MA from)

blablabla maximale rate

6.4 Warning and state-info system

- warnings from dolly ECU
  In the original state, the ETS-ECU monitors all safety critical components of the dolly. In case of a malfunction, an error code is sent via the CAN-Bus and the error message is visible on the diagnose displays of the dolly. Depending on the severity of the error

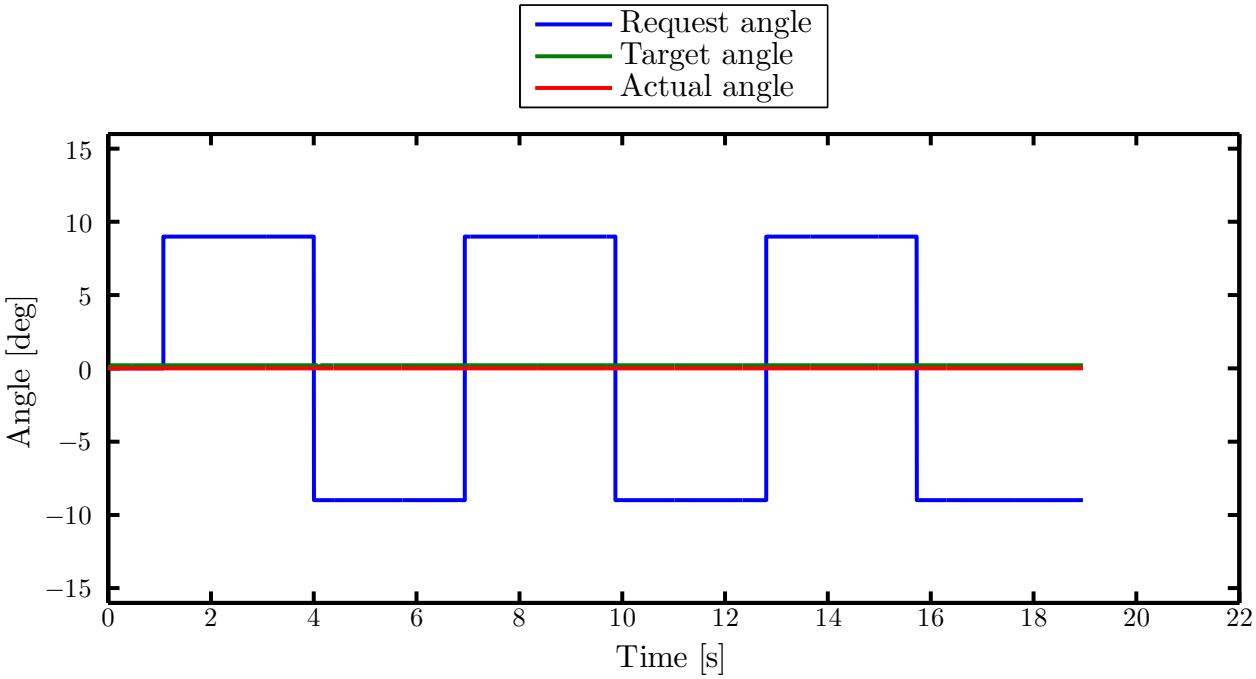


Figure 6.2: Step input 8 degree

the steering is deactivated and the axes steer back to a steering angle of 0 deg. Due to the required modification of the ASF- and speed-signal, the inbuilt supervision of the ETS-ECU is no longer fully functioning. Therefore a additional supervision of the system has to be implemented in the Simulink-model. This supervision considers the current capabilities of the dolly and reduces the requested steering angle or steering rate, if needed.

- warnings from EBS
- warnings from vehicle
- 'own' error codes and warnings (e.g. logging, MABII related, arduino-IMU related)

7 Verification and validation of steering (2-3 Seiten)

7.1 Overview

In different states of the project different kind of tests were performed. As a first step bench tests were done to verify the developed software. Following that, several tests on the actual dolly, with the dolly in standstill and no trailers connected, were made.

7.2 Bench-Testing

7.2.1 VTM maneuver verification

For the testing of the vehicle model controllers described in chapter 3 as a first step all the parts were simulated in Simulink. Therefore all parts of the functionality architecture introduced in section 2.5 are simulated. For the Human Machine Interface (HMI) lookup tables were used for the steering angle and vehicle speed input, so instead of actual driver input a pre-defined maneuver was executed. The Vehicle Motion Management (VMM) consists of the vehicle model controller. The steering actuators in the Motion Support Device (MSD) domain were simulated by implementing a first order filter and a time delay, that represents the delay caused by the hydraulic system. For the vehicle plant the VTM (see section 5.1.2) was used. Figure 7.1 shows how the parts of the functionality architecture are divided to different systems for the simulation of the vehicle model controllers.

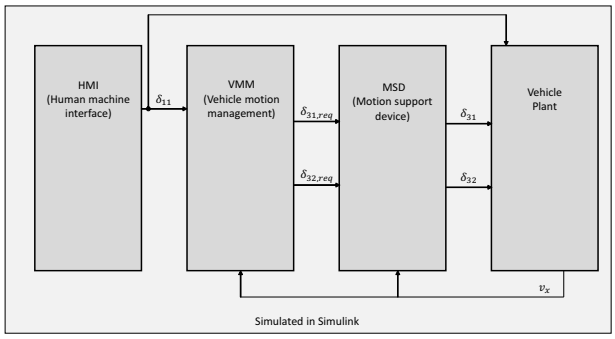


Figure 7.1: Functionality Architecture in simulation

- include simulation results for maneuver of controller being run inside VTM
- perhaps include some picture of the video-output, with the actuated truck
- plots for maneuver (steering input, speed over time)

7.2.2 CAN verification

To test the functionality of the ASF-Sensor as a first step it was just connected to power supply and an Arduino with a attached CAN-shield. After the CAN-message was received

successfully with the Arduino, as a next step the sensor's CAN pins were connected to the MABII. On the MABII a simple model, that only contained one CAN-Controller with the corresponding dbc-file, was loaded. The bus-monitor tool of ControlDesk, that shows all incoming and outgoing signals on the CAN for the corresponding dbc-file, was used to check the sensor signals.

As outlined in section 4.4 the signals generated for each ETS-ECU need to differ to achieve independent steering. The CAN-wiring of the legacy system thus had to be interrupted and split. To ensure that only the correct set of messages was available after the splitting point, Arduino Dues were spliced into the system at all relevant junctions. This was a necessary procedure as gateway-systems between all different CAN-subnets on the dolly are in place and needed to be circumvented, which proofed to be a challenging task operating under the maxim of easy revertability to the original state.

7.2.3 Fault detection system verification

- send faulty inputs
- show plot of faulty input and system reaction

7.3 Processing time evaluation

7.3.1 Background

The desired solution is supposed to operate at any speed. For high speeds a quick processing and transmission time is required to ensure prompt and realtime intervention of the control system based on the measured input signals. If the delays induced by the different components in the complete system are known or CAN be estimated, they can be compensated for in the steering-algorithm running on the rapid-prototyping system.

As outlined in section 4.6 the dolly is equipped with a system to determine the deflection angle of the draw-bar and both the kingpin angles of the HCT combination. The sensors' raw signal is then parsed and filtered in an integrated low-level system which feeds the filtered signals to a private CAN-bus, where it is picked up by the MABII. The filtering operation takes a certain time and thus induces a delay. Furthermore the model running on the rapid-prototyping system needs a certain time to calculate the current desired steering angle for the dollys' wheels. This has to be determined as well. The steering mechanisms on the dolly are also a delay-inducer due to the inertia in the hydro-mechanic system. This is of course unavoidable in any physical system, but when measured can as well to a certain degree be compensated for in the calculations.

7.3.2 Measured input delay

The response time of the dolly's steering system was determined in workshop environment in stand-still without any additional axle loads. Tests were conducted for both axles individually to ensure that maximum power was available preventing further delays through insufficient electrical power or hydraulic pressure. A repeated steering angle in the form of a step input was used to determine the reaction time of the system. The step amplitude was increased over

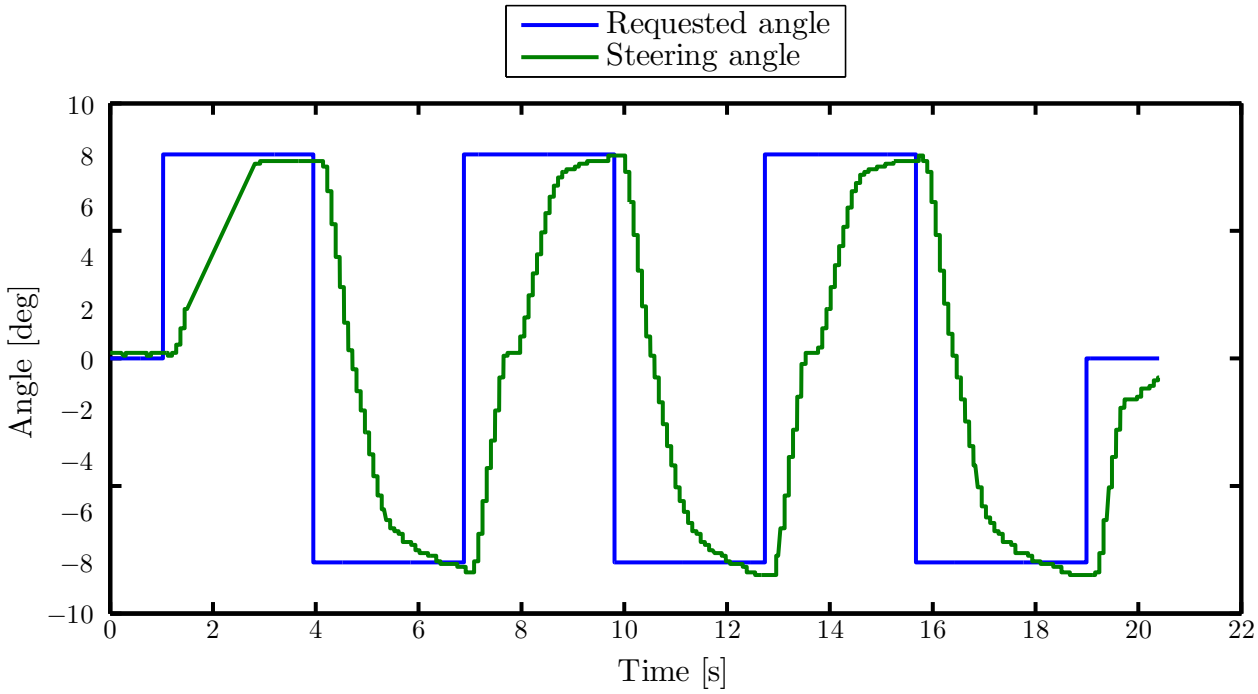


Figure 7.2: Example of step input for delay determination, front axle lifted, 8° amplitude, six repetitions.

different tests to account for possible dependencies on articulation of the steering system. The tests each started and ended with a steering angle of 0°. An example for the test sequence can be seen in figure 7.2. Clearly the difference between the requested angle and the actual articulation of the wheel can be seen. Further discussion of the results can be found in section 8.2.

To eliminate tire friction the investigated axle was suspended in the air in further series of testing by slightly craning up the dolly's front/rear (see figure A.1) to allow for free movement of the tires. An example for the utilized testing routine can be gathered from table 7.1. Between the different steps (double amplitude) a sufficient wait time of 3500ms was used to allow the system to reach the requested steering angle. This waiting time was kept constant throughout all tests.

As also described in section 8.2 additional response tests were conducted utilizing a ramp input. Here a maximum steering amplitude of 15° was used and the slope of the ramp was varied. As the request was not in the form of a step input, no error was triggered like in test #9 and #18 (see table 7.1). The test matrix for this setup is outlined in table 7.2. Besides the changed input-pattern the test- and logging-setup was left unchanged

All testing was automated in ControlDesk using Python scripts to loop through the different tests. The utilized GUI CAN be seen in figure A.3. Logging within ControlDesk was started and stopped manually after each test and automatically exported to MATLAB for parsing and analysis. All data was acquired from the steering knuckle sensors originally present in the ETS.

[1]If a too big of a step is requested from the steering system, VSE's ECU will go into emergency mode. Steering is completely locked, turning the dolly completely passive.

#	Amplitude [°]	Axle	Comment
1	1	front	lift front axle, ETS dead band
2	2	front	
3	3	front	
4	4	front	
5	5	front	
6	6	front	
7	7	front	
8	8	front	
9	9	front	error mode, step too[1]
10	1	rear	lift rear axle, ETS dead band
11	2	rear	
12	3	rear	
13	4	rear	
14	5	rear	
15	6	rear	
16	7	rear	
17	8	rear	
18	9	rear	error mode, step too big[1]

Table 7.1: Test matrix for delay measuring with lifted axle (analogous matrix for non-lifted testing (#19-#36))

#	Amplitude [°]	Rate	Sample time [s]	Comment
1	15	0.01	0.001	
2	15	0.001	0.001	
3	15	0.1	0.001	
4	15	0.3	0.001	
5	15	0.4	0.001	error mode, slope too steep

Table 7.2: Test matrix for delay measurings with ramp input

This sensor's signal is available on the ETS-CAN (also see section 4.2).

It was decided to neglect the delays induced by the CAN communication. The reason for this is, that both the request as well as the measuring are send over the CAN. Which means that the two transmission times are counted with different signs and thus almost exactly cancel out only leaving double the worst case transmission (maximum stuffing bits, worst case arbitration) time as a maximum error of 232μs. This is sufficiently low regarding the fact, that the simulation on the MABII is executed every 0.001s.

7.3.3 Delays in logging measuring chain

To have a complete picture of the occuring delays in the measuring chain all relevant bigger execution blocks of the code running cyclically on the Arduino Due to broadcast the IMU data to the CAN-bus were evaluated for execution time. Figure 7.3 gives an overview, into which functions the code was grouped as well as the respectively used protocol for communication. The output via the serial port (Universal Asynchronous Receiver Transmitter (UART)) for visualization purposes on the computer while logging takes very long compared to the execution

times of the other depicted code blocks.[1]It therefore is disabled while actually logging.

The delays were evaluated by inserting time-stamped variables into the code before and after each examined function's call. Using the Due's *micros*() function a resolution in microseonds CAN be achieved[2] with an accuracy of the inverse of the processors clock frequency (i.e. 11.9ns for the Due). *micros*()' call/execution time was neglected as it is in the magnitude of a couple of dozen nanoseconds. As

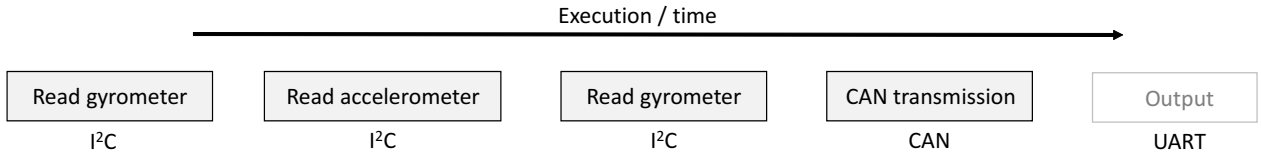


Figure 7.3: Execution blocks for IMU data acquisition on the Arduino Due (overview)

7.4 Vehicle testing

7.4.1 System calibration

The two ECUs on the standard ETS were calibrated to account for the hardware characteristics before handing-over the legacy system to the active converter dolly project. The parameter set included zeroing of hydraulic pressure levels of the steering system and the integration of the EBS-CAN signals as well as the articulation angle sensor CAN (ASF), as it would theoretically be possible to use other sources for the speed-signal needed in the legacy ETS. This calibration ensured a correctly and fully working base-system, which in case of failure could be reverted back to. Further on this parameter sets will not be changed to prevent inconsistent behaviour.

[1]The maximum transmission rate on the UART for the Due is 115200bit/s. Transmitting the payload (3x3 sensor axes, signed integer) only excluding all control bits accumulates to 3 sensors x 3 axes x 16bit = 144bit or 1.25ms

[2]On earlier/smaller Arduinos *micros*() resolution is 4μs!

7.4.2 Actuator tests

Actuator tests to verify working hydraulic system for the steering system and correctly pressurized brake-lines as well as normally running ECUs receiving correct analog sensor-information (voltage levels) within the legacy ETS and CAN-signals are possible with VSE's diagnosis screen, that usually sits in the ETS locker but was patched out for the HiL -tests and during setup in the workshop. As this diagnosis utility ties in directly with the core-functionality it provides a very robust and reliable way to track errors during testing. An example of the home-screen output of the diagnosis display can be seen in figure 7.4. Note that it is in "Alarm Mode", which is the fall-back safety level, where all steering cylinders are locked in middle position (i.e. wheels straight). The ETS reverts to this state, if sensor information is corrupted or deemed inconsistent. This particular example was triggered by the tests described in section 7.3.2.

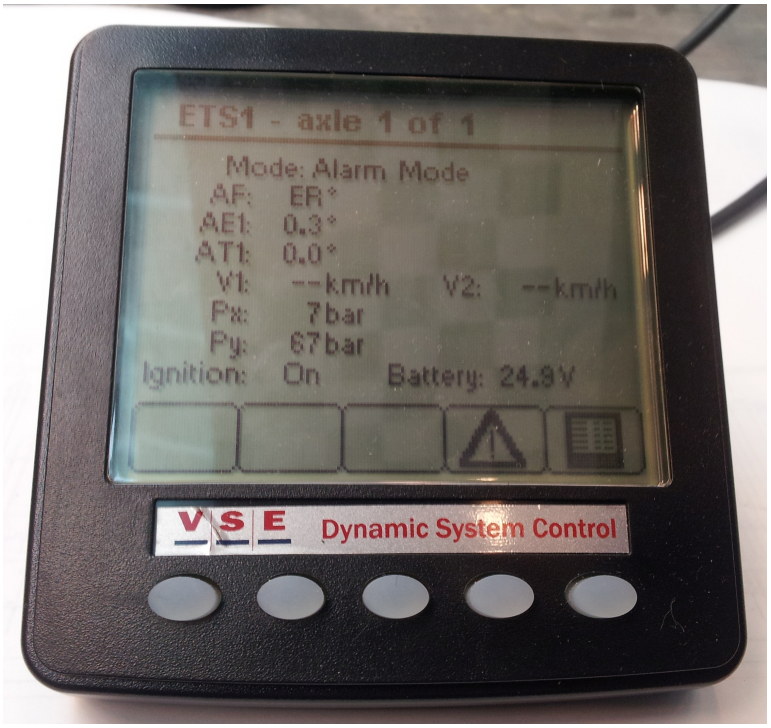


Figure 7.4: ETS' diagnosis sceen patched out of the original ETS-locke rto allow easy access during workshop testing. The display is showing the homescreen.

7.4.3 Algorithm evaluation

7.4.4 Sensor testing

The three angle-sensors to determine the articulation between the different independently moving units of the combinations differ in form-factor as they are mounted either on the fifth-wheel turntable or the king-pin respectively, nonetheless all rely on the same measuring principle and broadcast their gathered data in the same fashion on a private CAN-bus (refer to figure 4.5).

As was laid out in section HIER-REF-FUER-BILD-MIT-WEITERFUEHRPARAMETERN!

some CAN-messages needed to be patched through to the receiving ECU and some had to be changed. It was necessary to run the sensors in bench-tests first to ensure this communication was understood and controllable before implementing it into the final system. The VSE sensors need to be connected to 24V Descrete Current (DC) and then independently setup and broadcast the angle of deflection of their turning-plate. The CAN-signal was picked up with a slim Arduino CAN-setup at first to verify the data. In further tests the CAN-reception was realized n th MABII where subsequently the describes by-pass with modification of selected signals was developed. HIER NOCH EIN BILD VOM SENSOR AM ARDUINO BREADBOARD!!!! Further calibration or testing was not necessary for this sensors, as they are very robust and reliable.

- zero IMUs, make sure paired sensors show the same outputs

7.5 Hardware-in-the-loop testing

- description of HiL
- Functional architecture diagram (MSD replaced with actual dolly)
- Limitations (axle loads, standing still)

VTM includes a lot of processing heavy sub-models (tire models, vehicle parameter sets) which lead to processing power not sufficing to allow for VTM 's execution in the dSpace environment on the MABII. To perform HiL -testing it was thus necessary to split the comptuational load and accomplish real-time data exchange between the hardware controlling system on the MABII and the rest of the simulation which will be run parallely in Simulink in real-time on a standard PC. Though there are dedicated real-time platforms available to achieve real-time execution it was decided to rely on a standard PC to minimize costs and have a lean work-process without extra steps of code conversion to different platforms. Figure 7.5 illustrates the distribution of the HiL -setup's different components over two computers, the MABII and the actual hardware.

The Host-PC fulfills the usual task to start-up the dolly and simulation during HiL testing and log the behaviour of the dolly system and the IMUs. This is done through the ControlDesk GUI which was described in section 5.3 and of which a screenshot can be found in the appendix (figure A.3). It already runs the final version of the control and capturing environment and does not require changes for track testing. It is connected to the MABII via the Host-PC port.

On the Simulation-PC the simulation of the complete rest of the hardware-system is computed in real-time. Mainly truck, trailers, environment and steering behaviour of the truck are part of this model. All these sub-systems are modeled in VTM which runs on this computer in Simulink.

Usually Simulink simulations are computed as fast as possible and then afterwards analysis of the results and system's behaviour is conducted offline. This is an unsuitable approach for this thesis' purposes, as it is important to have real-time execution for this sub-system to provide the actual hardware of the HiL -test with the needed parameters and simulate the environment at the correct instance also considering the hardware's feedback. I.e. the hardware runs in real-time, so the rest of its environment has to as well. Simulink's Desktop Real-Time toolbox provides the functionality to execute models in real-time on a desktop PC as well as

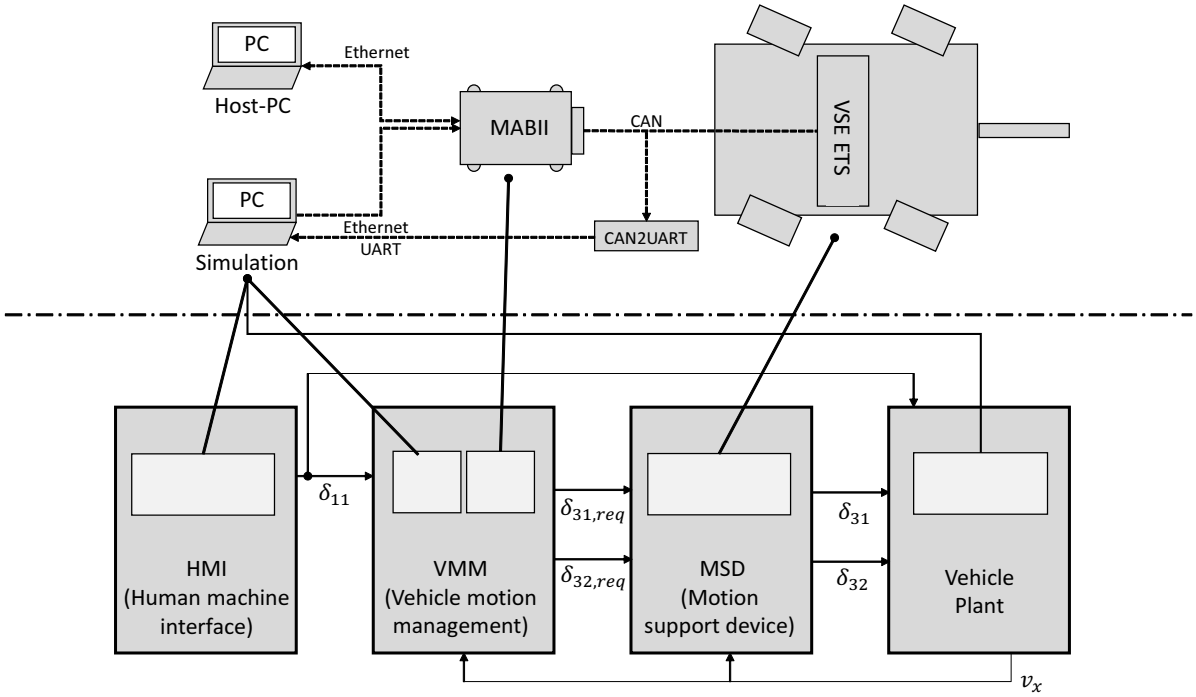


Figure 7.5: Overview of HiL -simulation, distribution of sub-functions over different physical platforms (top) and correlation to Volvo functional architecture (bottom)

support for a variety of hardware systems. This toolbox was used to connect the MABII's ethernet port to the simulation-PC's ethernet port using the UDP protocol. Also part of this toolbox is a sub-packet to access the PC's serial port within the simulation-environment. As it was not possible to receive reliable real-time data via the ethernet-interface which lead to the introduction of the serial interface for the feedback-loop. A similiar setup as decribed in section 4.6.3 was used to realize the conversion of the CAN-messages directly taken from the EBS-CAN to serial output on a Arduino Due. As depicted in figure 4.6 it is essential to to keep the CAN-buses of the two axles separated to ensure individual steering. Thus two MCP2551 CAN-transceivers had to be used for the feedback-loop in the HiL -setup.

Limitations: HiL -testing took place in workshop environment and thus the dolly was not rolling compromising the tire friction compared with a rolling combination. As outlined in section 8.2 this does not increase the reaction time of the system, though. Furthermore the additional axle load was zero for both axles, as it was not possible to increase the load without a trailer in the lab environment at hand. This is also unlikely to change the results, as is suggested in 8.2 the axle-load is no deciding factor in the wheel articulation's performance.

The setup for the HIL-testing can also be represented using the functionality architecture introduced in section 2.5. Compared with the testing of the controllers described in section 7.2, the division of the functionality architecture changed. The MSD are no longer simulated but the actual dolly together with the MABII, which is needed as an interface between the MSD and the other parts of the functionality architecture, is used. The HMI, VMM and vehicle plant are still simulated in Simulink. Figure 7.6 shows how the parts are divided to different systems.

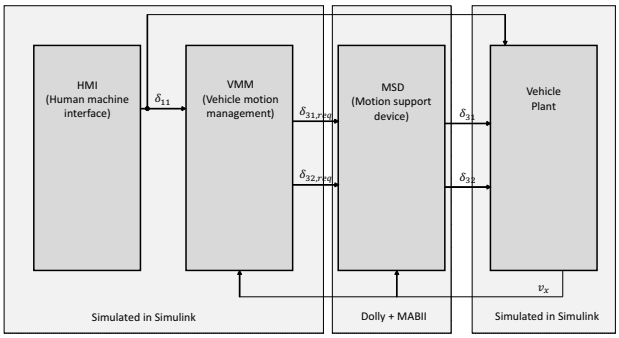


Figure 7.6: Functionality Architecture for HIL-testing

7.5.1 Low-speed controller

The maneuver that was used for testing the Low-speed controller was a 180 degree turn at constant speed. The steering angle of the tractor can be seen in Figure 7.7.

As a first step a very simple controller was used for controlling the steering of the dolly. The

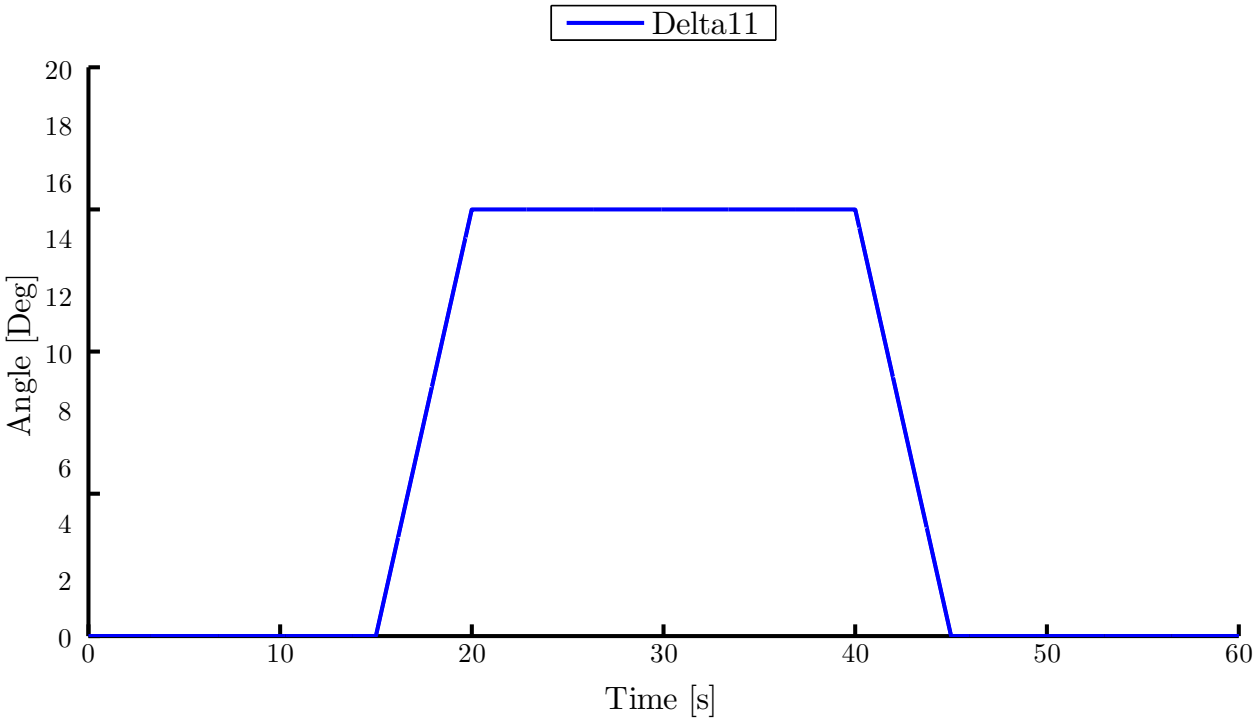


Figure 7.7: HIL test steering angle tractor

controller used the articulation angle between the first semitrailer and the dolly as an input. The output to both axles was this angle multiplied with a gain of 4.

7.5.2 High-speed controller

7.6 Track testing

For testing on a test track the different parts of the functionality architecture (see section 2.5) will be handled as follows: Since a complete vehicle combination will be used for track testing, this corresponds to the vehicle plant of the functionality architecture. The HMI domain is represented by the driver of the vehicle. The VMM is running on the MABII, which works as an interface between the VMM, the dolly and the vehicle. Figure 7.8 shows how the parts are divided to different systems.

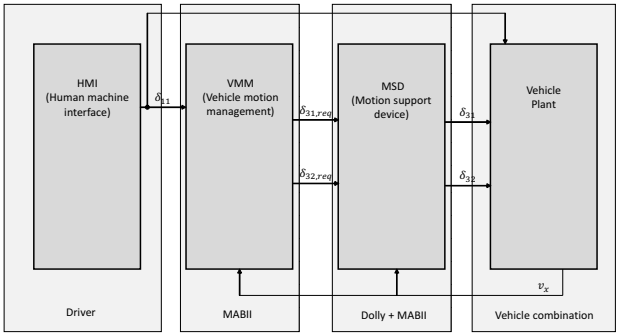


Figure 7.8: Functionality Architecture on track testing

7.6.1 Testmaneuvers

- lit research for standard maneuvers
- sine-wave
- outline critical parts of maneuver
- figure with SA over time
- expected behaviour from simulation

7.6.2 Testenvironment AstaZero

- overview of AZ
- map in appendix?
- restrictions of environment

7.6.3 Testmatrix

- checklist for launch
- parameters that very varied

- different runs
- planned maneuvers

7.6.4 Test setup and instrumentation

- detailed description of placement of sensors, wiring, logging-PC

8 Discussion (3-4 Seiten)

8.1 Results from bench testing

- lessons-learned?
- adaptation for future projects
- what was taken over for further tests?
- what couldnt be simultaed?

8.2 Results from processing time evaluation

Delay is 0.26s for front and 0.3s for rear.

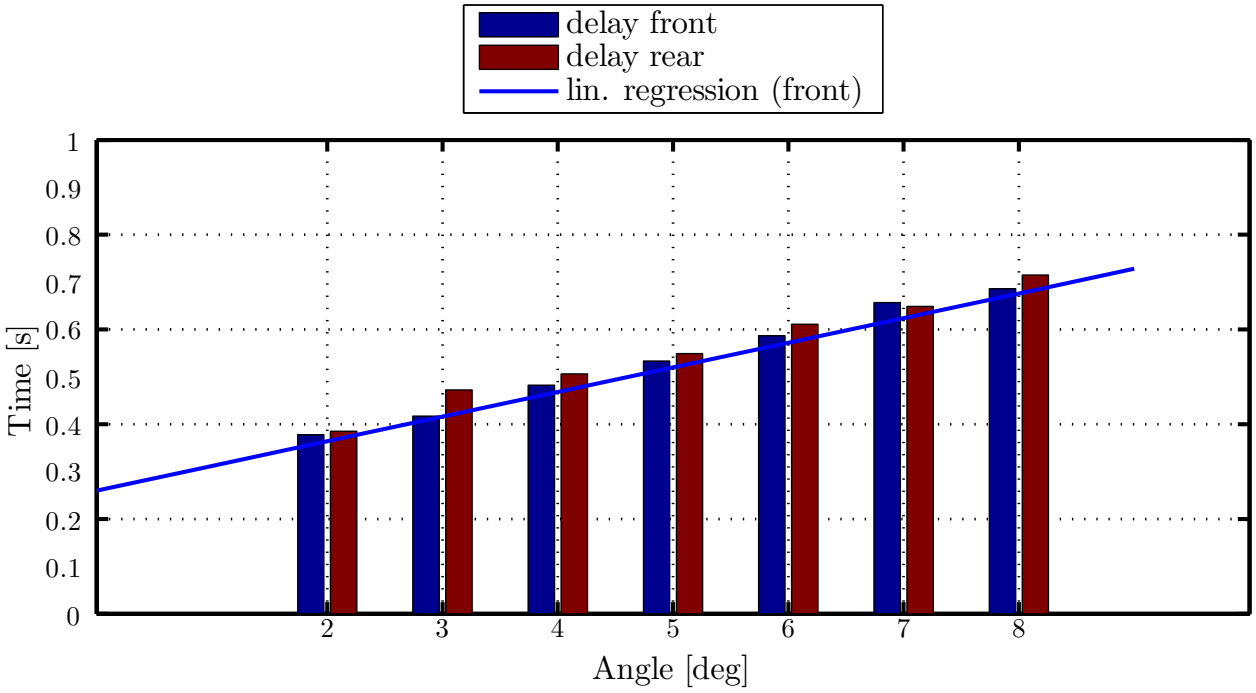


Figure 8.1: Rise time from send out request to achieve a certain steering angle on the dolly.

- diagrams (over amplitude, show time)
- explain different curves
- explain data gathering
- mention ramp input
- show reaction time

Figure 8.2 shows the requested angle, target angle and actual angle of the dolly's front axle over time. As described in section 7.3.2 the requested angle is a step input, in this case of 5 degree.

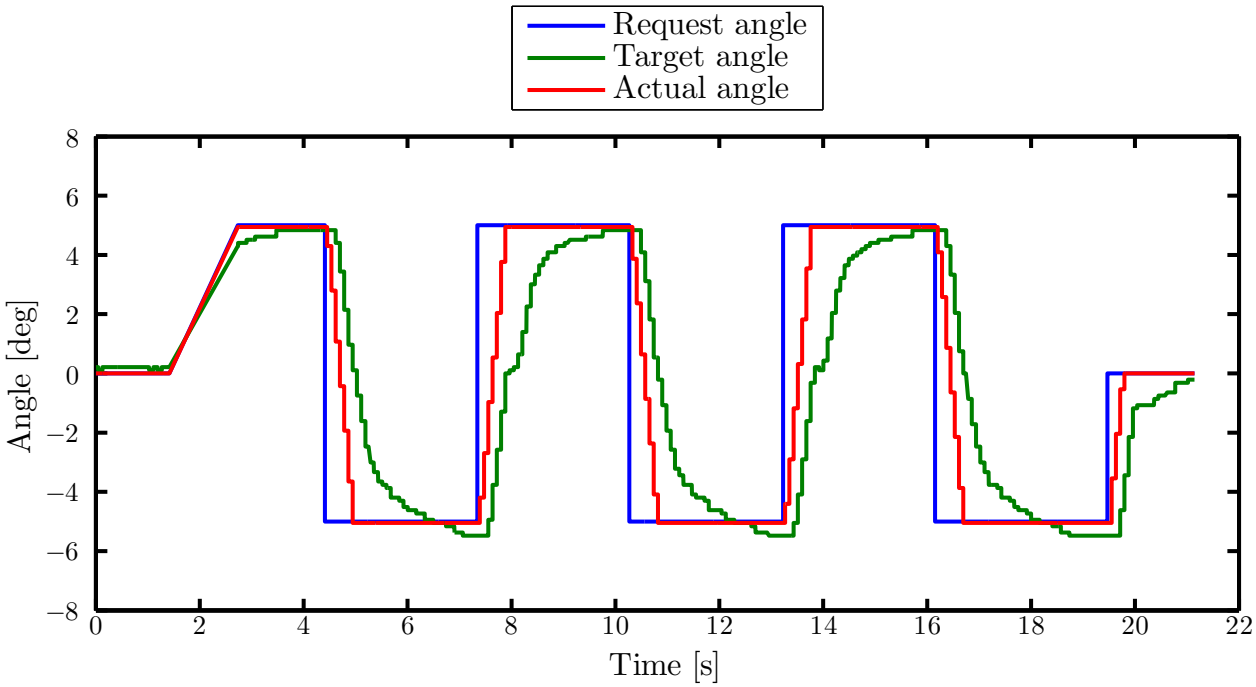


Figure 8.2: Step input 5 degree

The target angle's slope is constant and lower than the slope of the request angle. The actual angle has approximately the same slope as the target angle, but it is delayed. The actual angle tends to overshoot the requested angle for falling slopes. Furthermore the slope of the actual angle is reduced when the actual angle almost reached the request. There is also a saddle point around an angle of 0 degree for rising slopes of the actual angle.

During plot analysis of preliminary results it was discussed, whether it would be possible to by-pass the occuring delay by feeding a ramp-input instead of a step-input into the ETS. It was assumed that there is a low-pass filter in place which would not affect ramp inputs. Subsequent tests with ramp inputs with varying slopes and amplitudes showed that it was indeed possible to eliminate the low-pass filter leading to no delay between the target angle and the request angle. Nevertheless it was not possible to get any faster change than with the request of a step input. It thus can be concluded that the maximum steering rate of $5°/s$ can not be circumvented.

8.3 Results from in vehicle testing

- CAN-analysis
- robustness?
- reliability of safety features

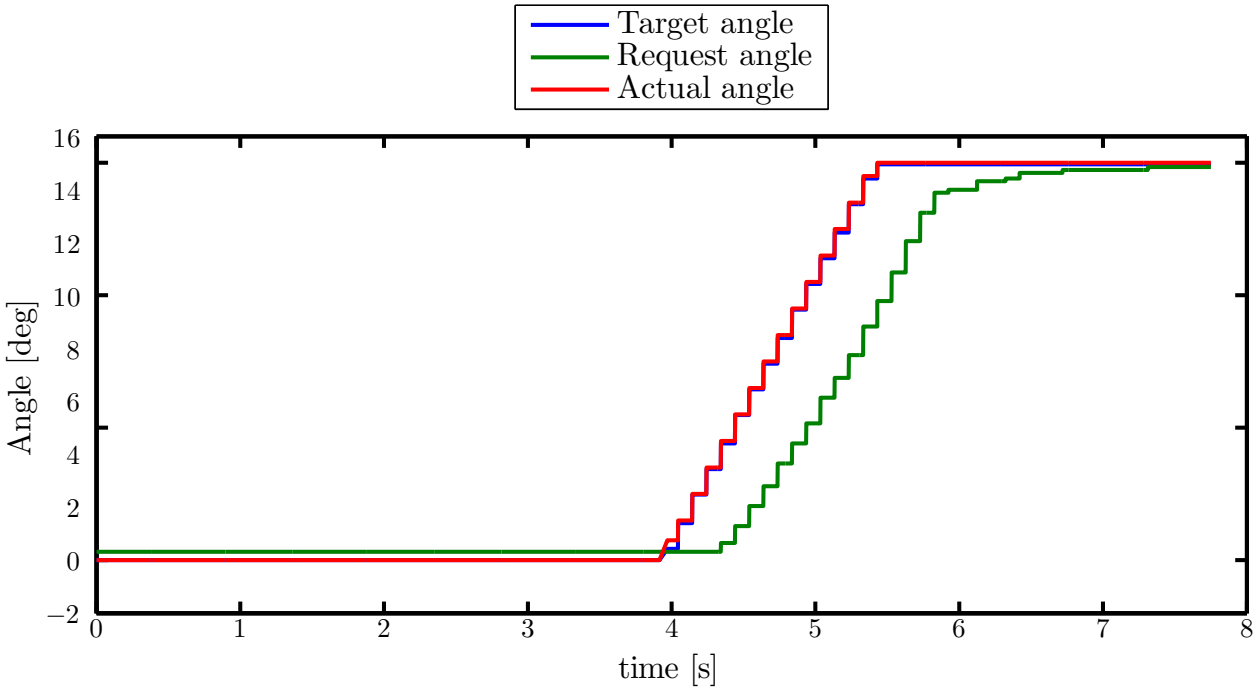


Figure 8.3: Step input with rate limiter

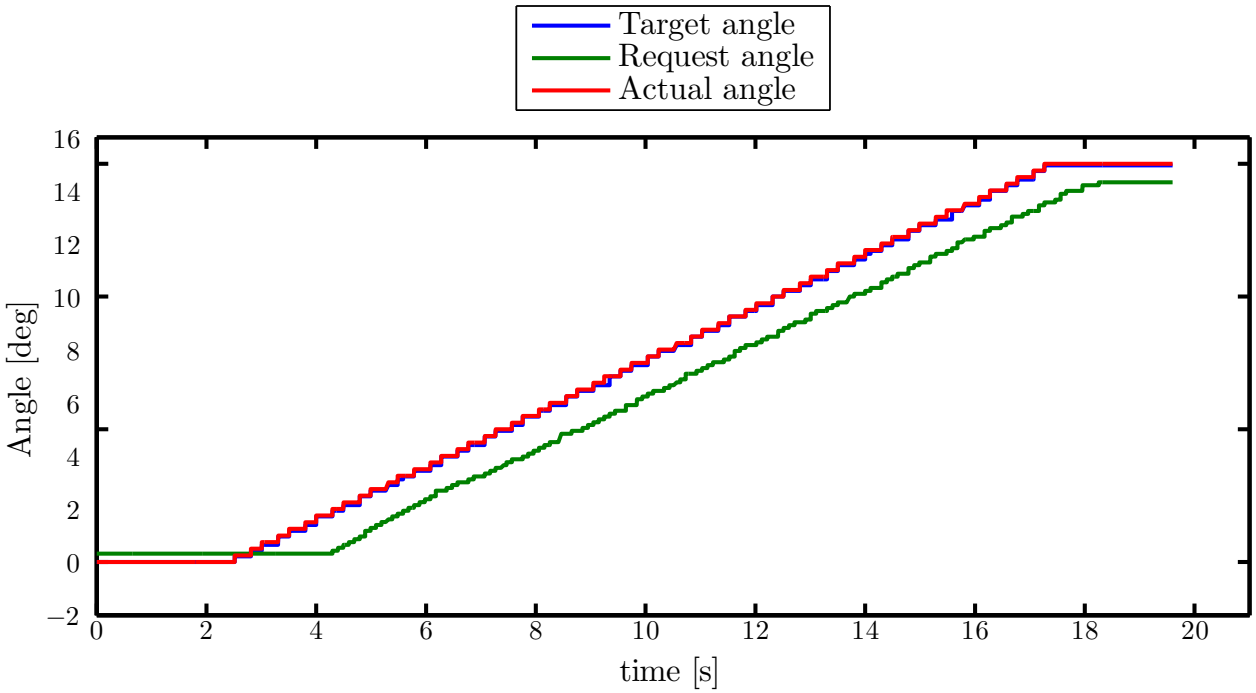


Figure 8.4: Low rate input

8.4 Results from hardware-in-the-loop testing

Figure 8.6 shows the measurements of the ETS-CAN of the front axle of the dolly for the low-speed maneuver described in section 7.5.

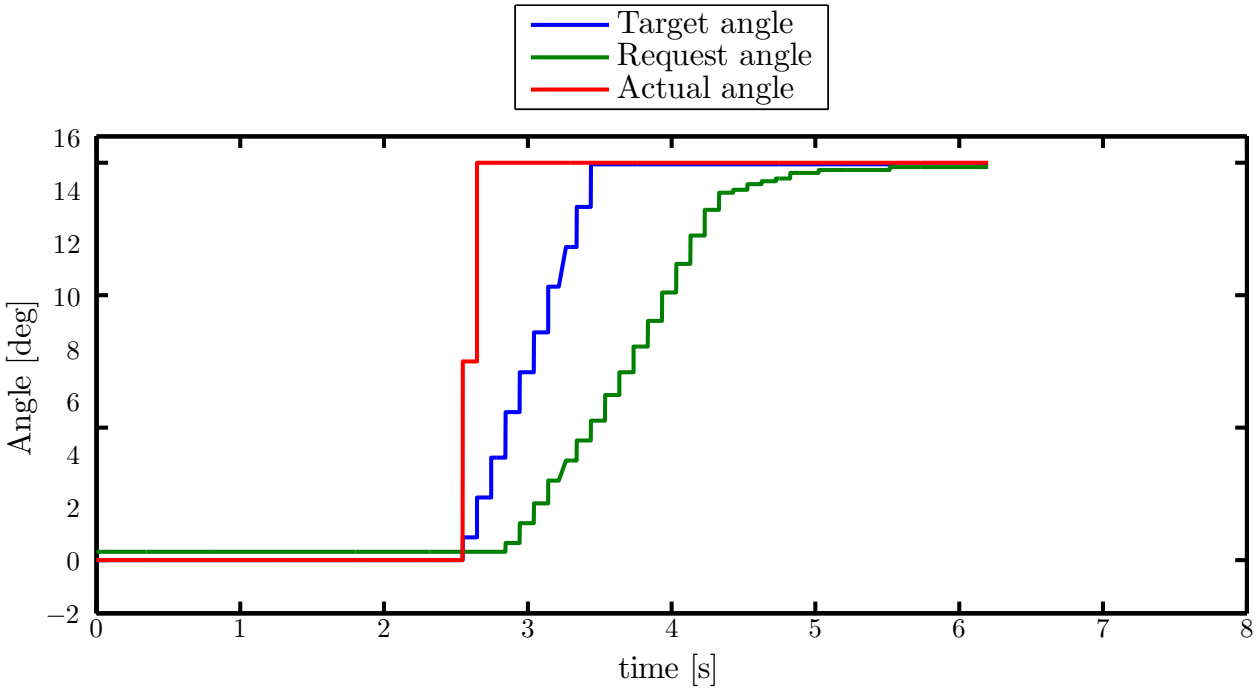


Figure 8.5: High rate input

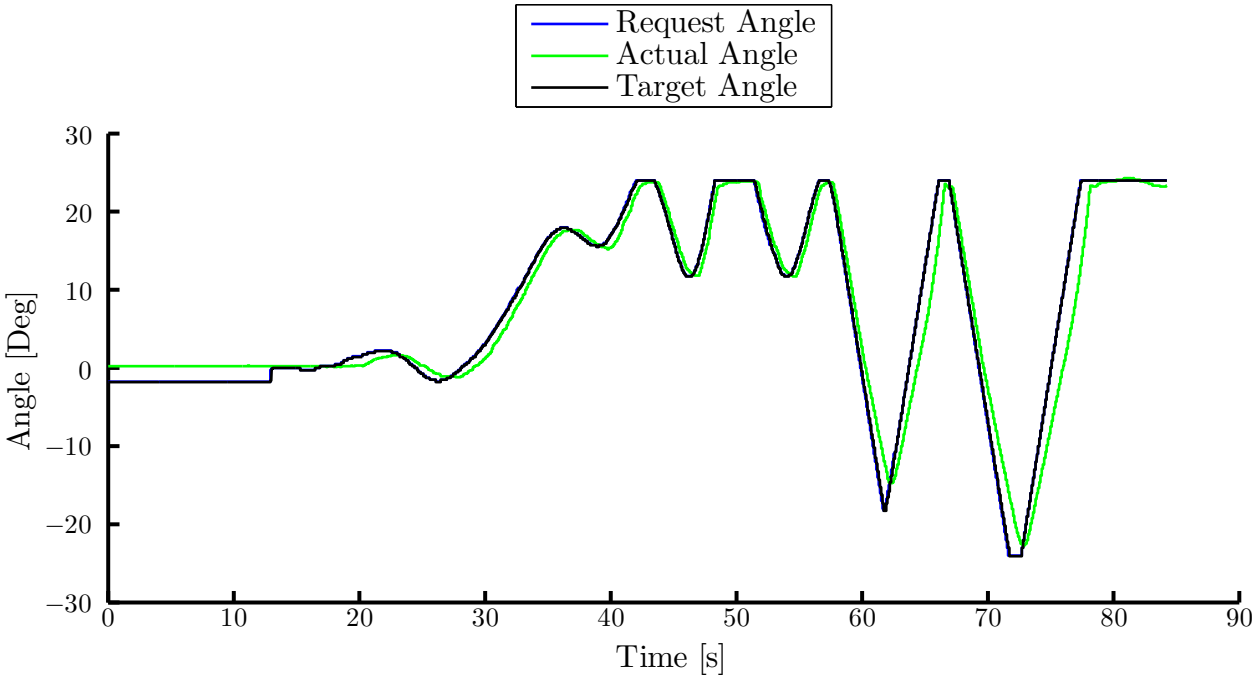


Figure 8.6: HIL-test low-speed front axle

The request angle is the angle that was requested from the controller and then sent from the simulation-PC to the MABII. The target angle is the angle that was calculated by the ETS-ECU and sent as a request to the actuators of the dolly. The actual angle is the angle that the sensors on the dolly's wheels measured. Figure 8.7 shows the same measurements for the rear axle.

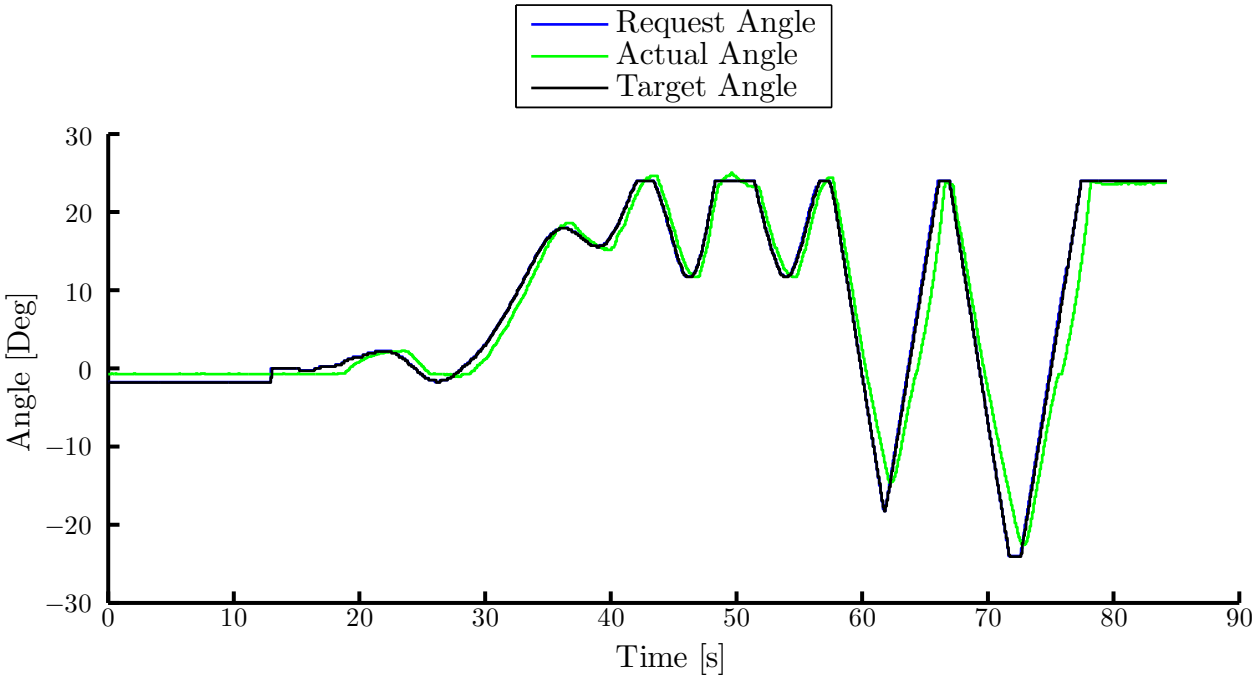


Figure 8.7: HIL-test low-speed rear axle

Since the same requested angle for front and rear is the same, below only the front axle is looked upon. Figure 8.8 shows an extraction of the measurements of the front axle.

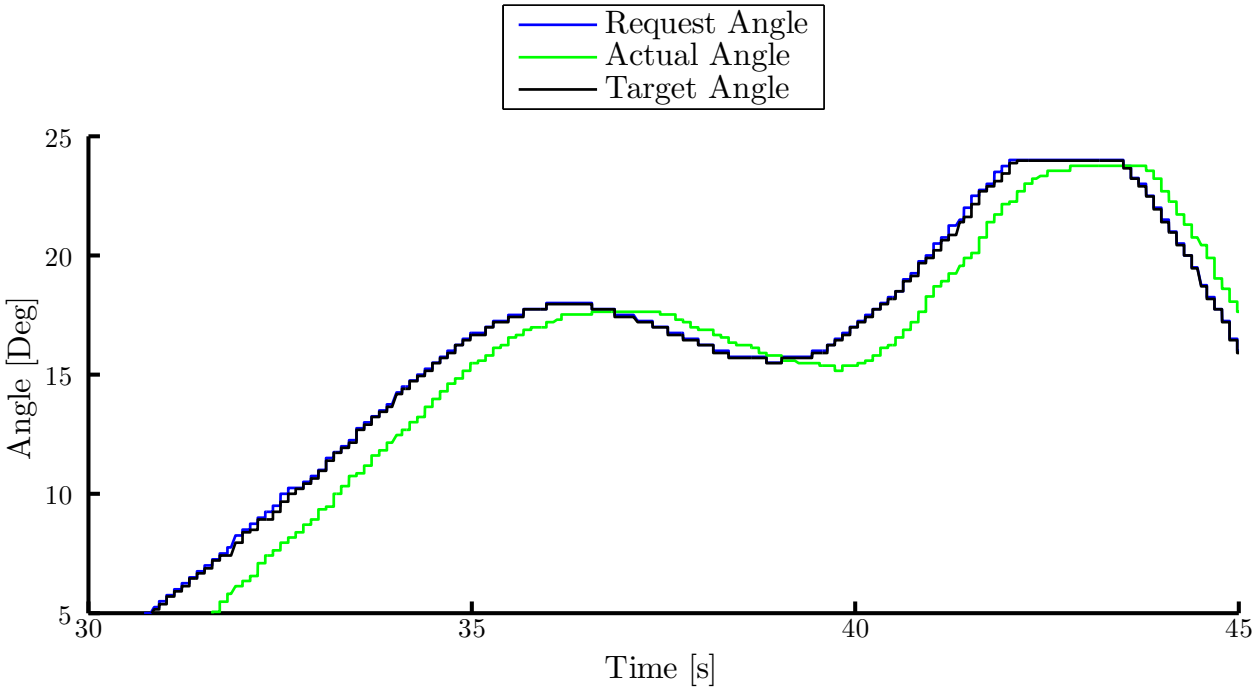


Figure 8.8: Details of HIL-test low-speed front axle

As already mentioned in section 7.3.2 there is a delay between the angle requested of the ETS-ECU and the actual angle of the wheels. This can also be seen in this plot. In comparison to this,

the delay between the angle request that is sent from the MABII to the CAN and the target angle that is sent from the ETS-ECU to the actuators is very small and can therefore be neglected.

To get a picture of the overall results of the HIL-test, Figure 8.9 shows the measurements conducted on the simulation-PC and the measurements done in ControllDesk in the same plot.

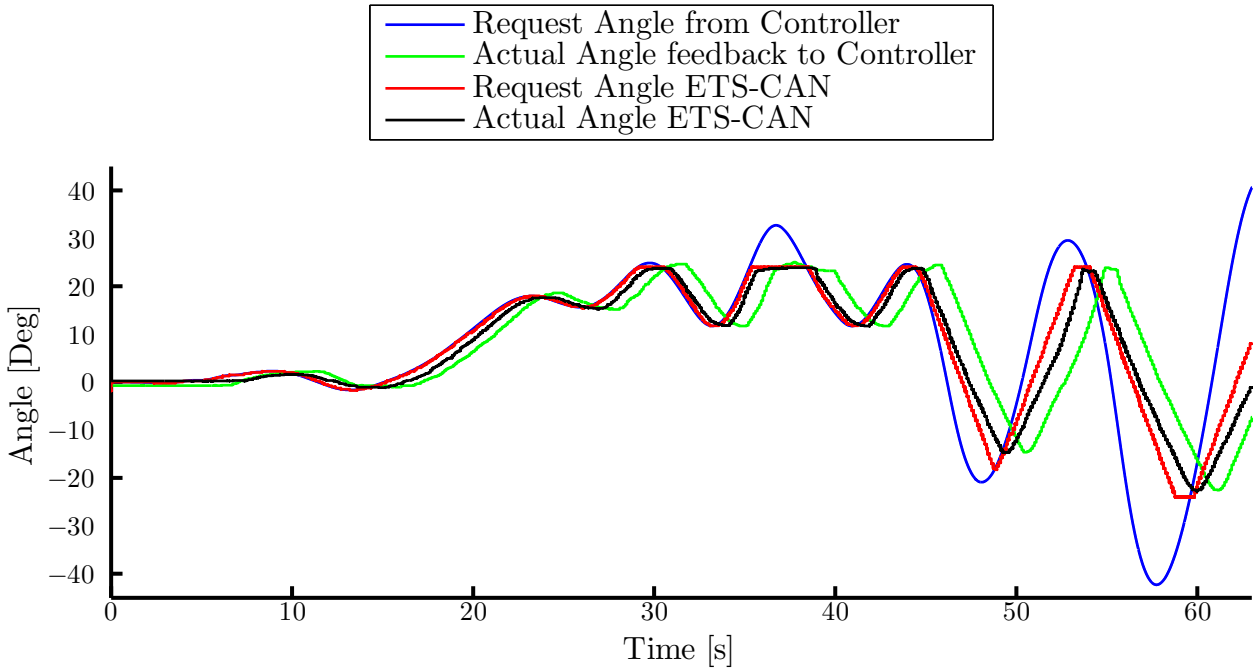


Figure 8.9: HIL-test front axle measurements from VTM and ETS-CAN

For one thing the plot shows the limitation of the angle that is requested from the controller by the supervisor block described in section 6.4. For another thing it shows that there is a relatively high delay between the signal of the actual angle of the dolly from the ETS-CAN and the signal that is feedback to the simulation-PC. This is most likely caused by the use of the serial interface to transmit the signal from the CAN to the simulation-PC. This delay causes a amplification of the articulation angle between the first semitrailer and the dolly, which directly correlates to the steering angle request to the dolly. This amplification can be seen in the end of the measurements.
To get a more detailed view of this delay as well as the limitation of the request angle Figure 8.10 shows a extraction of Figure 8.9.

- compare plots from HiL with simulation

8.5 Comparison

- VTM <=> testing

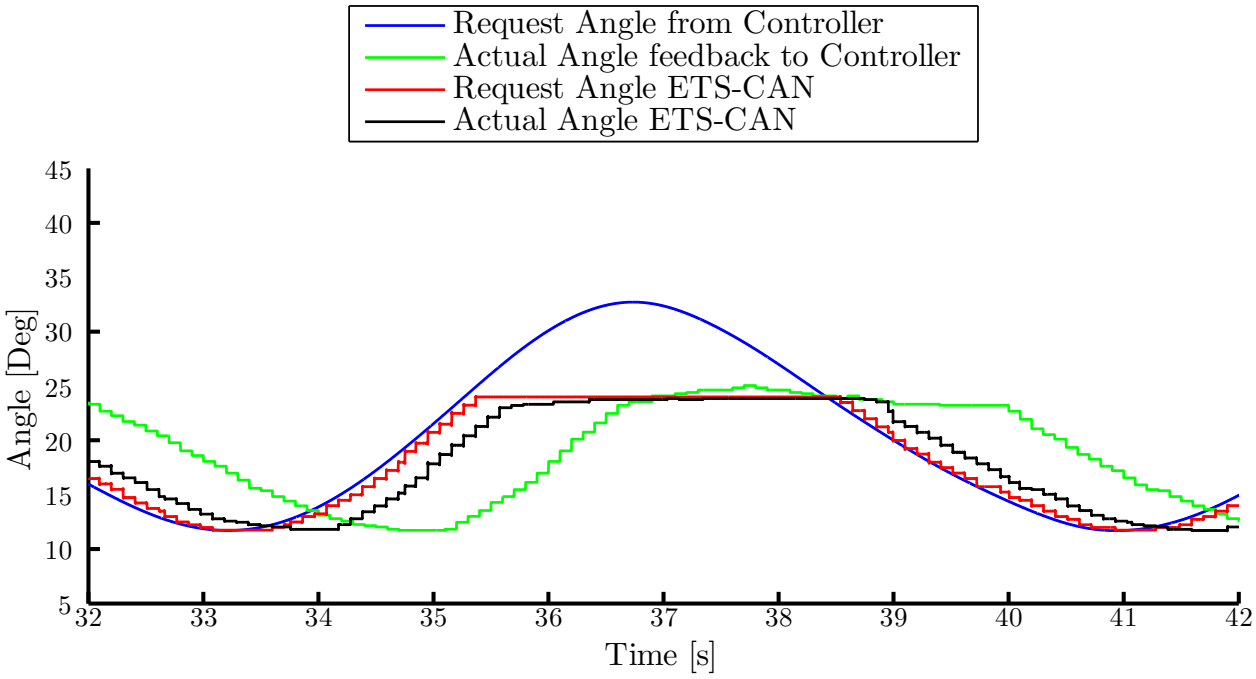


Figure 8.10: Details of HIL-test front axle measurements from VTM and ETS-CAN

9 Conclusion (2 Seiten)

9.1 Recommendation

The mounted systems only give information about the physical state of the combination but leave no possibility to take environmental circumstances into consideration. Sensing for the surroundings, such as radar/LIDAR or cameras can greatly simplify and enhance the results in post-processing for the data logged during testing.

The supervisor-functions have to be extended as well to incorporate a broader use of the combinatation's dynamic data (yaw, acceleration).

For simplified delay-free HiL testing a dedicated real-time system could be utilized, which would also provide the possibility of including environment to a greater detail (e.g. randomly generated road situations) to achieve a very robust verification in safe lab-environment before integrating the dolly in an A-double combination on track.

Direct command of the ECUs with designated request messages would greatly simplify the system, eliminating many work-around solutions that were developed in this thesis. Also directly controlling the hydraulic actuation on a lower control level could benefit the investigated delays by circumventing the added control-layers.

Utilizing electric motors for the steering instead of hydraulic systems could decrease the delays induced by the mechanical mechanism greatly.

9.2 Future Work

The developed solutions can fairly easily be adopted to accommodate wireless network support for logging as well as direct controls, because all main-signals at some are transmitted via IP. The controller is executed robustly on the MABII and runs independently as soon as HiL testing is left behind for track-testing; this ensures safety, as all the supervisor systems are in place before the immanently fragile wireless transmission of the data. Wireless networking support makes the implemented systems on the dolly more accessible for example for convenient and quick debugging, visualization or even app-controlled steering of the dolly in shunting or couplings situations where the driver could maneuver his combination outside of the tractor cabine, greatly enhancing his view-point.

The braking solution that was outlined in this thesis has to be implemented in detail and evaluated with great precautions before taking it to the track. The safest solution would be to request braking torques/brake pressures via the brake ECU's diagnosis interface, as this would leave the original braking CAN-communication intact and prevent a solution similar to the one implemented for the interception and modification of the ETS sensor signals.

An air-suspended leveling system is available on the dolly and CAN be controlled fairly easily over the standardized ISO-11992 CAN, physically available on the 7-pin trailer connector which carries the ABS/EBS signals. It has to be evaluated whether this functionality is available while the combination is moving and of course whether it contributes anything to the enhanced dynamic behaviour of the whole combination.

Wheel individual propulsion via hub-mounted electric/hydraulic motors is already available for cars and leads to a safety system similiar to the working principal of ESC but without the disadvantage of slowing down the vehicle. These implementations are usually called torque

vectoring or active yaw and greatly increase the performance in highly dynamic situations. Of course it is not possible to drive the second semi-trailer independently for extended periods of time as this exceed the output of the trucks alternator and would thus require impossibly large energy-storage in the second semi-trailer. Besides the great advantages that can be expected in dynamic situation, a electronically propelled dolly would allow independent shunting of the decoupled second semi-trailer unit in constricted spaces.

A Appendix

Figure A.1: Dolly craned up slightly to determine delay of steering actuation

Figure A.2: MicroAutoBoxII viewed from the top including cables for ZIF -connector, Host-PC, Simulation-PC and power supply (clockwise, starting on the right)

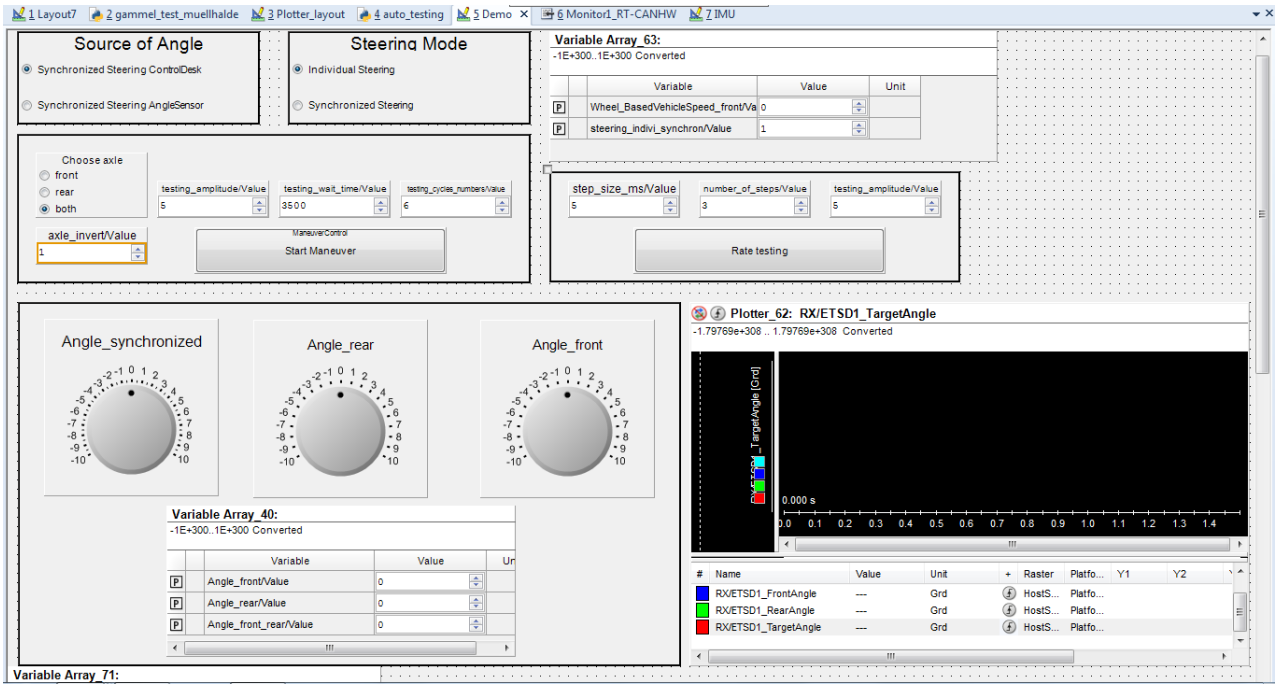


Figure A.3: ControlDesk GUI for basic testing and start-up of the dolly

Table A.1: Severity table for FMEA [16]

Severity	Description	Ranking
Hazardous without warning	Very high severity ranking when a potential failure mode effects safe system operation without warning	10
Hazardous with warning	Very high severity ranking when a potential failure mode affects safe system operation with warning	9
Very High	System inoperable with destructive failure without compromising safety	8
High	System inoperable with equipment damage	7
Moderate	System inoperable with minor damage	6
Low	System inoperable without damage	5
Very Low	System operable with significant degradation of performance	4
Minor	System operable with some degradation of performance	3
Very Minor	System operable with minimal interference	2
None	No effect	1

Table A.2: Probability table for FMEA [16]

Probability	Description	Ranking
Very High: Failure is almost inevitable	>1 in 2	10
	1 in 3	9
High: Repeated failures	1 in 8	8
	1 in 20	7
Moderate: Occasional failures	1 in 80	6
	1 in 400	5
	1 in 2,000	4
Low: Relatively few failures	1 in 15,000	3
	1 in 150,000	2
Remote: Failure is unlikely	<1 in 1,500,000	1

Table A.3: Detectability table for FMEA [16]

Detectability	Description	Ranking
Absolute Uncertainty	Design control cannot detect potential cause/mechanism and subsequent failure mode	10
Very Remote	Very remote chance the design control will detect potential cause/mechanism and subsequent failure mode	9
Remote	Remote chance the design control will detect potential cause/mechanism and subsequent failure mode	8
Very Low	Very low chance the design control will detect potential cause/mechanism and subsequent failure mode	7
Low	Low chance the design control will detect potential cause/mechanism and subsequent failure mode	6
Moderate	Moderate chance the design control will detect potential cause/mechanism and subsequent failure mode	5
Moderately High	Moderately High chance the design control will detect potential cause/mechanism and subsequent failure mode	4
High	High chance the design control will detect potential cause/mechanism and subsequent failure mode	3
Very High	Very high chance the design control will detect potential cause/mechanism and subsequent failure mode	2
Almost Certain	Design control will detect potential cause/mechanism and subsequent failure mode	1

References

[1] L. Aarts et al. Longer and heavier vehicles in the Netherlands: Facts, figures and experiences in the period 1995-2010. *The Hague: Ministerie van Verkeer en Waterstaat (VenW)* (2010).

[2] U. E. P. Agency. *Longer combination vehicles. a glance at clean freight strategies*. 2010. URL: http://www.epa.gov/smartway/forpartners/documents/trucks/techsheets-truck/420f10053.pdf (visited on 02/09/2015).

[3] I. Åkerman and R. Jonsson. European Modular System for road freight transport–experiences and possibilities. *TFK report* (2007).

[4] H. Backman and R. Nordström. Improved performance of European long haulage transport. *Transport Research Institute (TfK), Stockholm* (2002).

[5] A. Bálint et al. "Correlation between truck combination length and injury risk". *Australasian College of Road Safety Conference, 2013, Adelaide, South Australia, Australia*. 2013.

[6] T. R. Board. *REVIEW OF MEXICAN EXPERIENCE WITH THE REGULATION OF LARGE COMMERCIAL MOTOR VEHICLES*. Oct. 2011. URL: http://onlinepubs.trb.org/onlinepubs/nchrp/nchrp_rrd_362.pdf (visited on 02/09/2015).

[7] V. V. S. E. B.V. *Product information ETS for trailers*. 2014.

[8] S. Corrigan. Introduction to the controller area network (CAN). *Texas Instrument, Application Report* (2008).

[9] E. DIN. 60812: 2006-11 (2006) Analysetechniken für die Funktionsfähigkeit von Systemen–Verfahren für die Fehlzustandsart-und-auswirkungsanalyse (FMEA)(IEC 60812: 2006). *Deutsche Fassung EN* **60812** (2006).

[10] C. Doll et al. Long-Term Climate Impacts of the Introduction of Mega-Trucks, Study to the Community of European Railways and Infrastructure Companies (CER). *Fraunhofer ISI (Study Co-Ordinator, Karlsruhe) TRT (Milan), NESTEAR (Gentilly), Fraunhofer-ATL (Nuremberg), Fraunhofer-IML (Dortmund), Karlsruhe* (2009).

[11] GitHub. *collin80/due$_c$an*. 2015. URL: https://github.com/collin80/due_can.

[12] GitHub. *Seeed-Studio/CAN$_B$US$_S$hield*. 2015. URL: https://github.com/Seeed-Studio/CAN_BUS_Shield/.

[13] dSpace GmbH. *MicroAutoBox II - Product Brochure*. 2013. URL: https://www.dspace.com/shared/data/pdf/2013/ProductBrochure_MicroAutoBox-HW_E_ebook.pdf.

[14] F. B. K. GmbH. *Excellent driving characteristics outstanding manoeuvrability - The active articulated dolly*. Sept. 2012. URL: http://www.krone-trailer.com/english/neuheiten/excellent-driving-characteristics-outstanding-manoeuvrability/ (visited on 05/12/2015).

[15] M. Hjort, M. Haraldsson, and J. M. Jansen. *Road Wear from Heavy Vehicles: An Overview*. NVF Committee Vehicles and Transports, Nordiska vägtekniska förbundet, 2008.

[16] U. of Idaho. *FMEA*. URL: http://www.ece.uidaho.edu/ee/classes/EE481S03/FMEA.PDF (visited on 05/25/2015).

[17] A. Infrastructure and Transportation. *Highway Provider View of Long Combination Vehicles*. Mar. 2005. URL: http://www.transportation.alberta.ca/Content/docType59/Production/FAQs%28US%29.pdf (visited on 02/09/2015).

[18] M. M. Islam, L. Laine, and B. Jacobson. "Improve Safety by Optimal Steering Control of a Converter Dolly using Particle Swarm Optimization for Low-Speed Maneuvers".

[19] M. M. Islam, L. Laine, and B. Jacobson. "Inverse Model Control Including Actuator Dynamics for Active Dolly Steering in High Capacity Transport Vehicle".
[20] M. S. Kati et al. "Performance Improvement for A-double Combination by introducing a Smart Dolly". *13th International Heavy Vehicle Transport Technology Symposium, San Luis, Argentina*. 2014.
[21] K. T. G. . C. KG. *Kögel Lang-Lkw: Drei Dollys stehen zur Wahl*. Apr. 11, 2012. URL: http://www.koegel.com/de/news/artikel/datum/2012/04/11/koegel-lang-lkw-drei-dollys-stehen-zur-wahl/ (visited on 05/12/2015).
[22] *mikalhart/TinyGPSPlus*. 2013. URL: https://github.com/mikalhart/TinyGPSPlus/.
[23] P. Nilsson. On Traffic Situation Predictions for Automated Driving of Long Vehicle Combinations (2015).
[24] *ntruchsess/arduino$_u$ip*. 2014. URL: https://github.com/ntruchsess/arduino_uip.
[25] T. E. PARLIAMENT and T. C. O. T. E. UNION. *DIRECTIVE 2006/126/EC*. Dec. 20, 2006. URL: http://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32006L0126&from=DE (visited on 05/07/2015).
[26] PololuCorporation. *AltIMU-10 v4 Gyro, Accelerometer, Compass, and Altimeter (L3GD20H, LSM303D, and LPS25H Carrier)*. 2015 (accessed February 12, 2015). URL: https://www.pololu.com/product/2470/.
[27] *pololu/l3g-arduino*. 2015. URL: https://github.com/pololu/l3g-arduino.
[28] *pololu/lsm303-arduino*. 2015. URL: https://github.com/pololu/lsm303-arduino.
[29] "Rapid Control Prototyping". German. *Rapid Control Prototyping*. Springer Berlin Heidelberg, 2006, pp. 295–318. ISBN: 978-3-540-29524-2. DOI: 10.1007/3-540-29525-9_7. URL: http://dx.doi.org/10.1007/3-540-29525-9_7.
[30] J. Schäuffele and T. Zurawka. *Automotive software engineering: Grundlagen, Prozesse, Methoden und Werkzeuge effizient einsetzen*. Springer-Verlag, 2012.
[31] P. Steenhof, C. Woudsma, and E. Sparling. Greenhouse gas emissions and the surface transport of freight in Canada. *Transportation Research Part D: Transport and Environment* **11**.5 (2006), 369 –376. ISSN: 1361-9209. DOI: http://dx.doi.org/10.1016/j.trd.2006.07.003. URL: http://www.sciencedirect.com/science/article/pii/S1361920906000411.
[32] STMicroelectronics. *L3GD20H - MEMS motion sensor: three-axis digital output gyroscope: Datasheet - production data*. 2013.
[33] STMicroelectronics. *LSM303D - Ultra compact high performance e-Compass 3D accelerometer and 3D magnetometer module: Datasheet — preliminary data*. 2012.
[34] K. Tindell, H. Hansson, and A. J. Wellings. "Analysing real-time communications: controller area network (CAN)". *Real-Time Systems Symposium, 1994., Proceedings.* IEEE. 1994, pp. 259–263.
[35] D. of Transport and M. Roads. *Guideline for Multi-combination Vehicles in Queensland*. July 2013. URL: http://www.tmr.qld.gov.au/~/media/busind/Heavyvehicles/guidepermits/Guideline_mcv_form1.pdf (visited on 02/09/2015).
[36] T. C. O. T. E. UNION. *Council Directive 96/53/EC*. July 25, 1996. URL: http://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:31996L0053&from=en (visited on 02/05/2015).
[37] Vägverket. *Weight and Dimensions for Road Traffic*. URL: http://www.unece.org/fileadmin/DAM/trans/wp24/wp24-presentations/documents/pres08-04.pdf (visited on 02/05/2015).

[38] G Wangrin, B Stürmer, and M Wöhrmann. Technische Erprobung von Fahrzeugkombinationen mit einer Gesamtlänge bis 25, 25m („GigaLiner “). *Abschlussbericht NRW Modellversuch. TÜV Rheinland Kraftfahrt GmbH, Köln* (2009).