Master Thesis

Czech
Technical
University
in Prague

F3

Faculty of Electrical Engineering
Department of Electric Drives and Traction

FPGA Model Development of PMSM for Hardware-in-the-Loop Testing System

Bc. Vojtěch Štěrba

Supervisor: Ing. Tomáš Haubert, Ph.D.
Field of study: Electrical Engineering, Power Engineering and Management
Subfield: Electrical Drives
July 2020

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Štěrba** Jméno: **Vojtěch** Osobní číslo: **457076**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra elektrických pohonů a trakce**

Studijní program: **Elektrotechnika, energetika a management**

Specializace: **Elektrické pohony**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

FPGA Model Development of PMSM for Hardware-in-the-Loop Testing System

Název diplomové práce anglicky:

FPGA Model Development of PMSM for Hardware-in-the-Loop Testing System

Pokyny pro vypracování:

1. Analyze current state in the field of Power Inverter ECU testing.
2. Develop a real-time simulation with hardware-in-the-loop system for Power Inverter ECU.
3. Analyze the current FPGA model of PMSM machine.
4. Improve the PMSM FPGA model and verify it.

Seznam doporučené literatury:

[1] QUANG, Nguyen P. a Jörg-Andreas DITTRICH. Vector Control of Three-Phase AC Machines: System Development in the Practice . Aufl. Berlin, Heidelberg: Springer-Verlag, 2008. ISBN 9783540790280;3540790284;.

[2] WILSON, Peter. Design Recipes for FPGAs: Using Verilog and VHDL [online].Second. GB: Newnes, 2016;2015;. ISBN 9780080971292;0080971296;.

[3] BLYLER, John. Software-Hardware Integration in Automotive Product Development [online].Warrendale: SAE International, 2013. ISBN 0768080525;9780768080520;

Jméno a pracoviště vedoucí(ho) diplomové práce:

Ing. Tomáš Haubert, Ph.D., katedra elektrických pohonů a trakce FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **09.02.2020** Termín odevzdání diplomové práce: **14.08.2020**

Platnost zadání diplomové práce: **30.09.2021**

Ing. Tomáš Haubert, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

.

Datum převzetí zadání	Podpis studenta

Acknowledgements

I would like to express my gratitude to my supervisor Ing. Tomáš Haubert Ph.D. for the greatest support during my studies and opportunity to work on the topic of this thesis. Without his support and big effort I would have never completed this thesis. Furthermore I would like to thank to Bc. Rafael Banzhaf. His knowledge, experience and goodwill allowed this thesis to happen. Great gratitude goes also to all my Czech and German colleagues and friends who helped me during the thesis.

I would like to also thank my family for moral and material assistance since the beginning of my studies. Without them, I would have never gotten so far and I would have never finished the university.

The last acknowledgement goes to my girlfriend Peťa for her never ending patience and support even in the coronavirus breakdown.

“ Exactly!" said Deep Thought. "So once you do know what the question actually is, you’ll know what the answer means.”
Douglas Adams, The Hitchhiker’s Guide to the Galaxy

Declaration

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

In Prague, 5. July 2020

Abstract

This thesis deals with the problematic of Hardware-in-the-Loop (HiL) systems for control units of High Power Inverters. The text describes the structure of the system for simulations of electric drives in real-time and other necessary components of the Hardware-in-the-loop system. A control unit from an electric vehicle without High Power Stage is connected to the HiL system and an FPGA card is used to simulate electric drives. All time-critical tasks are simulated on the FPGA card. Furthermore, all necessary mathematical models are presented in this work.They run on a real-time PC and calculate all the necessary physical quantities.

In the next part of the work I deal directly with HiL settings, connection with the control unit, bus simulations and implementation of the sensor part.

The third part of this work mathematically describes the models that are implemented on the FPGA card. This model is further extended to foreign phenomena that are neglected in the derivation.

At the end of the work I compare the model with real measurement.

Keywords: Mathematical modelling, Hardware-in-the-Loop, PMSM, FPGA, Electric vehicles, Vehicle dynamics, dSPACE

Supervisor: Ing. Tomáš Haubert, Ph.D.
České vysoké učení technické v Praze,
Fakulta elektrotechnická
Technická 1902/2
Praha 6, 166 27

Abstrakt

Tato práce se zabývá problematikou Hardware-in-the-Loop (HiL) systémů pro řídící jednotky výkonových měničů. V textu je popsána struktura systému pro simulaci elektrických pohonů v reálném čase a další nutné součásti Hardware-in-the-loop systému. K HiL systému je připojena řídící jednotka z elektrického vozidla bez výkonnové části a pro simulaci elektrických pohonů je zde použita FPGA karta. Na FPGA kartě jsou simulovány všechny časově kritické úlohy. Dále jsou v této práci představeny nutné matematické modely, které běží na real-time PC a počítají všechny nezbytné fyzikální veličiny.

V další části práce se zaobírám přímo nastavením HiL, propojením s řídící jednotkou, simulací datových sběrnic a implementací sensorové části.

V třetí části této práce jsou matematicky popsány modely, které jsou implementovány na FPGA kartě. Tento model je dále rozšířen, aby zahranoval jevy, které jsou v odvození zanedbány.

V závěru práce je model porovnán s reálným měřením.

Klíčová slova: Matematické modelování, Hardware-in-the-Loop, PMSM, FPGA, Elektrická vozidla, Dynamika vozidel, dSPACE

Překlad názvu: Vývoj PMSM modelu pro FPGA bla bla doplnit — Vývoj PMSM FPGA modelu pro Hardware-in-the-Loop testovací systém

Contents

Figures

Tables

Chapter 1

Introduction

The segment of electric vehicle industry has been significantly growing during past years and there are major challenges ahead of the automotive industry. European cities and governments actively support transitions to low emission systems. As a result, European car producers need to develop electrical or hybrid vehicles.

More over during past years, new systems with autopilots are sales-blockbusters. To develop those autopilot systems, new algorithms with an enormous software complexity have to be used and new hardware topologies must be considered to overcome related issues. In connection to that, car producers must ensure software reliability and safety. For their simplicity, Hardware-in-the-Loop automated testing systems are developed and used.

1.1 Motivation

Electric car consists of many different embedded control units and their connections. Major task is to provide safety and reliability for the costumers. Every minor system in the car has to be tested and verified to ensure its correct behaviour. With never-ending growth of software complexity, automated test system for execution and evaluation are the key milestones. During my studies on CTU FEE I found the problematic around electronic control system, power electronics, real-time simulations very challenging and interesting.

I have been actively working with these systems since my Bachelors thesis and I was given a great opportunity to work and develop new HiL system from scratch. With the gained experience, I tried to improve the system structure.

1.2 Goals of the Master thesis

The main goals of this master thesis are following:

- Analyse current state in the field of Power Inverter ECU testing.
- Develop a real-time simulation with hardware-in-the-loop system for Power Inverter ECU
- Analyse the current FPGA model of PMSM machine.
- Improve the PMSM FPGA model and verify it.

Chapter 2

State of art - Hardware-in-the-Loop systems

2.1 Hardware-in-The-Loop systems

The Hardware-in-The-Loop (HiL) is a real-time operating system of multiple units for testing embedded control units in environmental models. The system consists of the unit under the test (UUT), dynamical models and signals for the simulation of needed environment. The system is operating in the real-time to provide results as close to reality as possible.

According to the best scenario all embedded systems would be tested in real environment, but usage of HiL system can reduce the costs of testing. Moreover, it's also possible to run tests, that would be hardly feasible in the real world. For example crash reactions or usage of the systems beyond the range of defined parameters.

The HiL testing is a last development stage before the UUT is released. This testing should provide an assurance that all systems operate correctly, and the final requirements are fulfilled. [11] [12]

2.2 HiL structure for Power Inverter ECU

This section provides overview of needed components and overall system structure for electrical machines simulations handling.

Hardware-in-the-Loop systems are nowadays built with a different purpose and different goals. The amount of testing scenarios options is depending on the purpose of the UUT and hardware equipment of UUT. This thesis mostly focuses on HiL testing of Power Inverter ECU, therefore usual HiL structure is presented, and possible simulation models are discussed.

Figure 2.1: dSPACE SCALEXIO Processing Unit [1]

2.2.1 Real-Time PC

The main component of the HiL is a real-time running computer. This computer provides a real time simulation of mathematical models with time cycles usually around 1 ms. Real-time computers technology uses mostly standard commercial components for computers. Because of that, system can easily be updated to a newer hardware with higher computing power. Moreover the Real-Time PC is also operated via external User Control Computer. [1]

2.2.2 Embedded Control Unit

Embedded Control Unit (ECU) is the key external component for HiL system. The ECU is the Unit Under Test (UUT). The ECU is usually the same hardware as hardware used in real product. Only additional parts for memory access and calibration are added, therefore it is possible to access internal variable values and easily flash a new software.

2.2.3 Electric Motor Simulation Card

Nowadays Power Inverter switching frequency continuously rises and frequencies around 10 kHz are not uncommon. Therefore, model calculations of corresponding physical reality must be carried with appropriate task rate to ensure that the model would match ECU predictions.

However, for this task period Real-Time PC is not suitable and special hardware for the simulation is used - FPGA cards.

"An FPGA is an integrated circuit (IC) that can be programmed for different algorithms after fabrication. Modern FPGAs consist of up to two million logic cells that can be configured to implement a variety of software algorithms. Although the traditional FPGA design flow is more similar to a regular IC

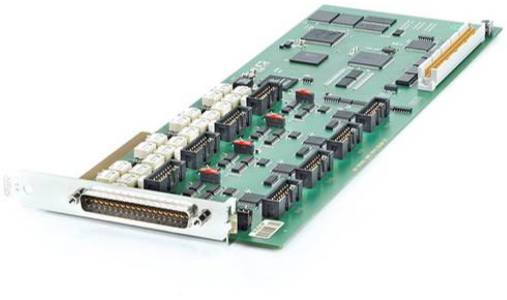


Figure 2.2: dSPACE DS4302 CAN Interface board [1]

than a processor, an FPGA provides significant cost advantages in comparison to an IC development effort and offers the same level of performance in most cases. Another advantage of the FPGA when compared to the IC is its ability to be dynamically reconfigured. This process, which is the same as loading a program in a processor, can affect part or all of the resources available in the FPGA fabric" [13]

2.2.4 Bus Simulation Card

There are usually several ECU's connected in a co-working system (for example car). Those ECUs must cooperate together and therefore bus interface is used. In a HiL simulation usually only one ECU is connected and therefore the rest of bus traffic must be calculated and simulated. Moreover, reactions to bus messages are often considered as a basic test to be carried out with HiL.

Below listed communication protocols are used in automotive industry and therefore are introduced here as an example. [1]

- CAN, CAN FD
- FlexRay
- Automotive Ethernet
- LIN

FlexRay

FlexRay (from Flexible Ray) is an automotive network protocol. This protocol was developed by FlexRay consortium and nowadays major German car

Figure 2.3: FlexRay logo.[2]

manufacturers are using this protocol or used this protocol in history.

Flexray was developed with a goal to be a deterministic, fault-tolerant and high-speed bus system. Transmission speed can be up to 10 Mbits and compared to CAN, it can be used in dual channel configuration to increase the fault-tolerance. Differential twisted pair wiring is used together with characteristic impedance between 80Ω and 110Ω. ECUs can be wired in different topologies from Multi-drop, Star or Hybrid Network. [2]

FlexRay communication cycle is usually around 1 to 5 ms with 4 main parts in. [2]

Nowadays FlexRay is used for example in following companies.

- Audi
- Porsche
- BMW
- Lamborghini

CAN

The Control Area Network is a bus developed in 1986 by Robert Bosch GmbH. This protocol was developed with intention to have a new multi-master protocol with non-destructive arbitration mechanism and furthermore there was no need to have central bus master.

Nowadays, this protocol is widely used in different fields, even though automotive industry is still the biggest utilizer. This protocol can be described as a multi-master serial bus, that can connect various ECUs. Connection is made with two differentialy wired cables, often noted as CAN-high and CAN-low. Usage of differential pair provides possibility to run the bus on different voltage levels and thus to have a higher interference resistance. Characteristic impedance is 120Ω. [14]

Various implementations of CAN higher layer protocols bus have been introduced, for example:

- CANopen - Industrial automation
- milCAN - Military vehicles
- ARINC 812 - Aviation industry
- NMEA 2000 - Marine industry

In automotive industry, standards are handled by each manufacturer.

2.2.5 I/O cards

Nowadays ECU measures and controls multiple systems in reality. To ensure correct entry data for the ECU, additional I/O cards are usually added to HiL system. These IO cards are used e.g. temperature measurement simulations.

Typical I/O card consists of:

- Voltage analogue input/output
- Current analogue input/output
- Voltage digital input/output
- Resistor simulation

Nowadays, manufactures provide lot of specialized cards for various types of simulation. (Gearbox sensors etc.)

2.2.6 Other Hardware Equipment

In addition to all listed hardware, other measurement tools are often connected to a HiL system. Those devices are often used as a measurement, calibration, and prototyping tools. They provide possibility to verify bus traffic with external measurements, they also provide option to online calibrate and measure connected ECU. With those devices timing measurements or algorithms verifications can easily be evaluated.

As an example, calibrations devices that can be used are listed:

- Etas ES891
- Vector VX1135 base module

Furthermore, HiLs are often designed to be easily extended and connectible with other hardware units. Examples of those units are listed below.

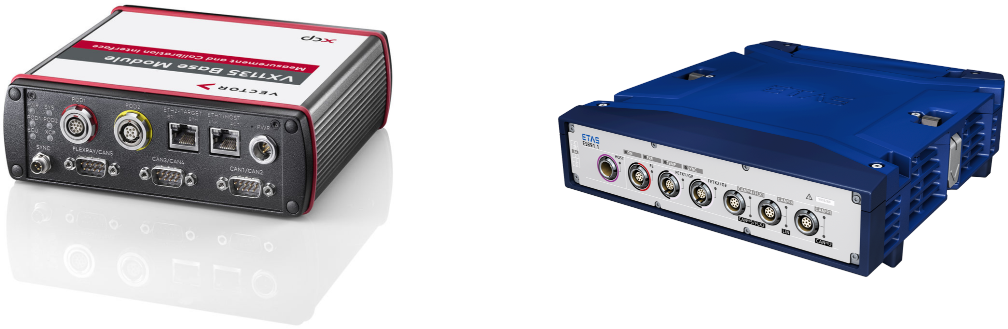


Figure 2.4: Vector VX 1135 [3], Etas ES 891.1 [4]

- Failure Injection Unit
- Oscilloscope carrier
- External bus connectors
- Break-Out panel

2.3 Electric motor simulations providers

This section provides information about HiL system manufacturers for Power Inverters and system overview for electrical machines simulations.

Special hardware for electrical machines simulation must be used. There are several options on the market with different advantages and disadvantages in case of HiL electrical machines control testing. In following sections, various solutions of hardware are presented together with a model solutions.

Main manufacturers for Power Inverters Hardware-in-the-Loop testing systems are listed.

- dSPACE
- ETAS
- Opal-RT

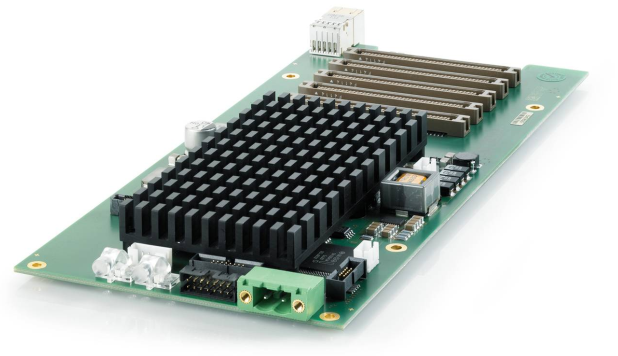


Figure 2.5: dSPACE DS2655 FPGA base board [1]

2.3.1 dSPACE

Signal level testing

This section describes dSACE HiL solutions on signal level testing. In this setup, High power stage is removed from the ECU and only logical signals are present.

There are several hardware solution offered by dSPACE, therefore, following hardware comparison table 2.1 is introduced.According to [1], these FPGA boards are designed for applications that require very fast, high-resolution signal processing, for example:

- Hybrid vehicle applications
- Electric drive applications
- FPGA-based electric drive simulation
- Power electronics simulation
- Electric motor control development

Moreover, every FPGA base board can be extended with I/O modules (DS2655M1, DS2655M2, DS5450 SC) to increase capabilities and amount of channels on the board.

dSPACE offers **RTI FPGA Programming Blockset**, **XSG Electric Components Library** and **XSG AC Motor Control Library** for effortless model implementations. Those components are implemented and developed via Xilinx® System Generator Blockset.

[1] DS6602 is also equipped with additional 4 GB onboard RAM and 36 000 kbit Ultra RAM

	Logic cells	Distributed RAM	Block RAM
DS2655 7K160	162 240 (DSP slices: 600)	2 188 kbit	11 700 kbit
DS2655 7K410	406 720 (DSP slices: 1540)	5 663 kbit	28 620 kbit
DS6601	444 000 (DSP slices: 1700)	5 908 kbit	19,000 kbit
DS6602[1]	1 143 000 (DSP slices: 1968)	9 800 kbit	34,600 kbit

Table 2.1: dSPACE FPGA boards [1]

RTI FPGA Programming Blockse provides RTI blocks for implementing the interface between the FPGA mounted on a dSPACE board and its I/O, and the interface between the dSPACE FPGA board and its computation node.

XSG Electric Components Library contains closed loop models of corresponding electrical machines. These models are calculated with high oversampling rate, where no PWM synchronizations are necessary.[15]

Models of following machines are provided with a corresponding inverter and sensor models [16]

- Permanent Magnet Synchronous Motor
- Brush-less DC motor
- Asynchronous Squirrel cage Induction Motor

High-Power testing

dSPACE is currently providing special system for power inverter testing, where the ECU is not tested on signal level, but the HiL system is built on so called **Hight-Voltage Electronic Load** solution. According to [1], one HV E-load module can be used with voltages up to 800 V and currents 70A/100A (continous/peak).

This module is directly designed for multiple parallel operations to support higher currents. This module must also be used together SCALEXIO unit, in which the model calculations are evaluated. [5]

Therefore, UUT directly operates in almost real environment thanks to which critical testing conditions, that cannot be performed with a real machine, can be evaluated.

2.3.2 ETAS

The ETAS ES5340 PCIe are internal computer cards developed specially for simulation of electrical motors in automotive industry.

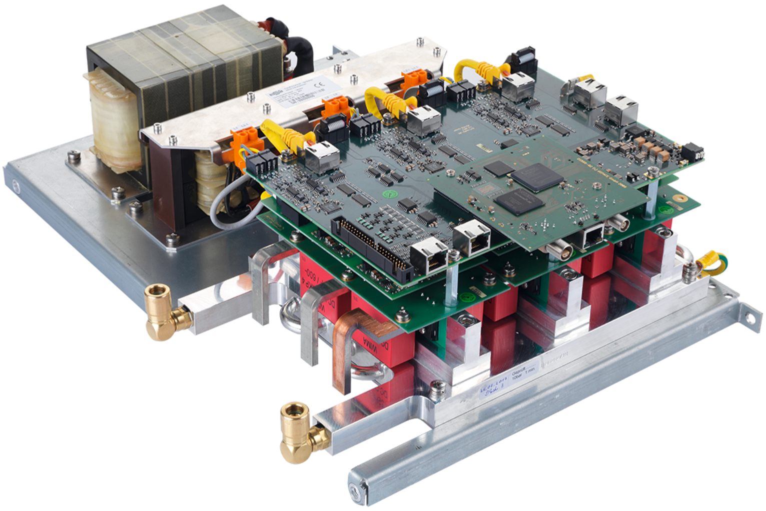


Figure 2.6: dSPACE HV E-load module [5].

The ETAS ES5340 PCIe simulation boards are connected to the control computer motherboard by PCI slots. These cards can be used to simulate the behaviour of electric motors, combustions engines, IGBT stages and electrified power-trains. That means that various motors can be parametrized and simulated.

Simulation cards are connected to PWM gate control signals generated in the tested power inverters. IGBT stage with PMSM is simulated.[17]

According to [17], the main benefits of the ES5340 are following:

- Calculations are carried with a step size of 850 ns.
- Phase current prediction.
- Various sensors can be configured on each output.
- Software models (LABCAR-RTPC) are synchronized with the PWM period by means of a task trigger.

2.3.3 Opal RT

Hardware solution

Opal-RT is providing multiple industrial solution for easy HiL system development, Rapid Control Prototyping together with I/O expansion capabilities. Furthermore, user-friendly automation environment can be used in connection with Opal-RT systems. Following hardware systems are recommended for Power electronics simulations:

Figure 2.7: OP4200 simulator.

Name	Hardware Solution	Logic cells	Distributed RAM	Block RAM
OP4510	Xilinx XC7K325T	326 080	4 000 kbit	16 020 kbit
OP5705	Xilinx XC7VX485T	485 760	8 175 kbit	37,080 kbit

Table 2.2: Opal-RT FPGA solutions [10]

- OP4200
- OP4510
- OP5707

The eFPGASIM suite is used for testing and validation of power electronic systems and covers a wide range of applications, from renewable energy conversion to highly complex multi-modular converters. Developers can use eHS electrical solver to easily bring models to real-time simulation. E-machine simulations are based on Xilinx® Virtex®-7 FPGA. According to [18], it is possible to run basic machine model with minimum task of 100 ns.

Model solution

Opal RT offers their eFPGASIM Power electronix toolbox with Modular Multilevel Converter Models. This tool can be used for easy model development from prepared model or a user can directly access the model structure using Xilinx System Generator for DSP.

Opal RT FPGA Electric Machine Library includes following models:

- Permanent Magnet Synchronous Machine (PMSM, IPM, BLDC, SPM)

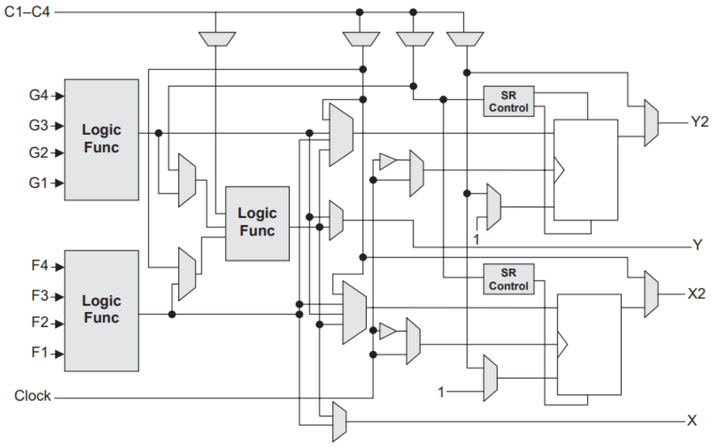


Figure 2.8: Xilinx CLB structure [6]

  - Variable D-Q model
  - Spatial Harmonics model

- Induction Machine (DFIG, DFIM, Squirell Cage Induction Machine)

- Switched Reluctance Machine

Furthermore, Opal-RT provides possibility to import data measurements from Finite Element Analysis (FEA) tool for example JMAG-Studio, Infolytica's MotorSolve or ANSYS Maxwell. FEA is mostly useful, once classical models can't be used. For example due to big rotor asymmetry. [18]

2.4 FPGA

Field Programmable Gate Arrays (FPGAs) are semiconductor devices that are based around a matrix of configurable logic blocks (CLBs) connected via programmable interconnects. FPGAs can be reprogrammed to desired application or functionality requirements after manufacturing. FPGA design mostly fits applications, where the flexibility and re-usability are requirements altogether with high frequency operations 100 MHz. [6]

FPGAs are heterogeneous compute platforms and are made with various parts including Block RAMs, DSP slices and Programmable logic. Because of the selected structure, parallelism and pipelining can be used.

Basic structure of FPGA is following and is usually called Configurable Logic Block /CLB)[7]:

- Look-up table - This element performs logic operations

- Flip-Flop - This register element stores the result of the LUT.
- Wires - These elements connect elements to one another.
- Input/Output (I/O) pads - These physical ports get data in and out of the FPGA.

2.4.1 FPGA design techniques

VHDL

Nowadays, there are several options how to process algorithm device into the FPGA. One of those option is usage of VHDL. VHDL is a hardware description language often used for programming of FPGAs and other electronic integrated circuits.

When synthesizing VHDL language for usage on FPGA, there are three main specific functions to be executed. [6]

1. Mapping - Logic functions mapped onto CLBs.
2. Placement - CLBs placed on FPGA.
3. Routing - Routed connections between CLBs

However, today algorithms are too complex to be manually mapped, and therefore synthesis software is used to turn VHDL into logic functions mappable into the FPGA. Due to the FPGA structure and its limited resources, algorithms size must be considered carefully and optimized.

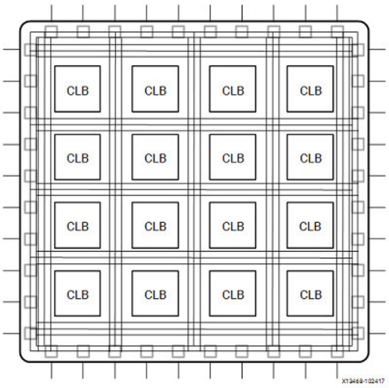


Figure 2.9: FPGA structure. [7]

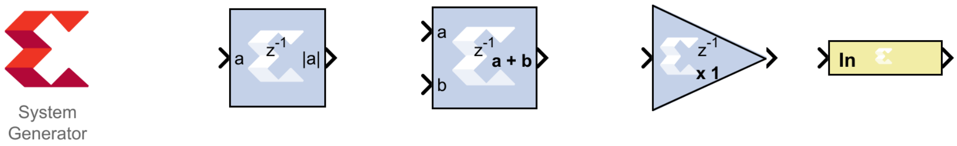


Figure 2.10: System generator for DSP blockset.

System generator for DSP

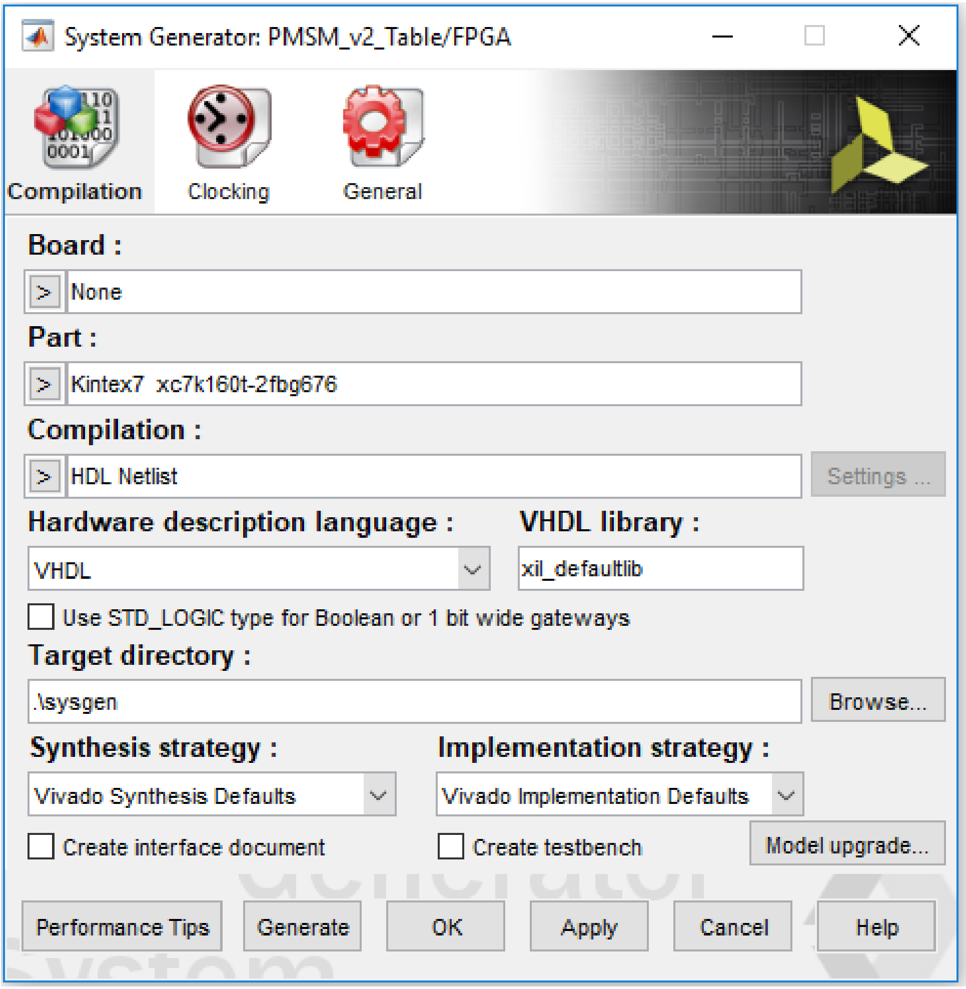
Another option how to program FPGA devices is nowadays System generator for DSP made by Xilinx. System generator for DSP is Matlab Simulink add-on designed to simplify FPGA development with all the benefits of the Simulink environment. In system generator for DSP, more than 100 RTL optimized Simulink blocks are present. Moreover, system design can be easily verified with online Simulink hardware accelerated simulations. This option is used by **dSPACE XSG libraries** and, therefore, FPGA model in this thesis is based on System generator for DSP. All information in the section are taken from [19].

Furthermore, direct code generation to packaged IP or low-level VHDL is possible with Timing and Resource analysis.

Various libraries are prepared for the developer, where some blocks are low-level (bit shifting or delays) and others can be described as high-level implementations (signal processing). Following examples of DSP libraries are listed:

- **Basic Elements** - Standard building blocks for digital logic
- **Control Logic** - Blocks for control circuitry and state machines
- **Data Types** - Blocks that convert data types (includes gateways)
- **Floating-Point** - Blocks that support the Floating-Point data type
- **Math** - Blocks that implement mathematical functions
- **Memory** - Blocks that implement and access memories
- **Tools** - "utility" blocks

System generator supports calculation and design with different number data types such as Boolean, floating point and arbitrary precision fixed-point values. This is in contrast with the fundamental Simulink number representation in double precision floating point and thus conversion blocks (**Gateway In, Gateway Out**) must be used, when DSP is used together with other Simulink blocks. Most of the Xilinx blocks can also deduce



System Generator: PMSM_v2_Table/FPGA

Board : None

Part : Kintex7 xc7k160t-2fbg676

Compilation : HDL Netlist

Hardware description language : VHDL

VHDL library : xil_defaultlib

□ Use STD_LOGIC type for Boolean or 1 bit wide gateways

Target directory : .\sysgen

Synthesis strategy : Vivado Synthesis Defaults

Implementation strategy : Vivado Implementation Defaults

□ Create interface document

□ Create testbench

Figure 2.11: System Generator for DSP, settings view

their data automatically to ensure no precision is lost. Overflow options are saturation, truncation, and overflow error.

Floating point number representation in DSP is in compliance with IEEE-754 standard for floating point arithmetic. User can select from Single, Double or Custom precision type. Single precision is 32 bits long, with 8 bits for the exponent, 24 bits for the fraction. Double precision is 64 bits. In Custom precision, user can freely decide bit width for exponent width and fraction width.

Algorithm functionality logic can easily be verified by Simulink interface. However, design must be furthermore verified with additional tools to make sure it will work on target device. **Timing Analysis** can be used to verify that timing requirements have been met in critical paths. **Resource Analysis** is the tool for verification of resources being used by HDL design and thus that design will fit into the target device.

Chapter 3

HiL development

In this chapter, development of the real-time HiL platform is discussed. All necessary steps for successful real-time simulations are presented. Cable harness for ECU connection is shown, followed by basic SCALEXIO settings. Furthermore, CAN and FlexRay implementations are discussed in connection with corresponding Restbus simulation models. Simulink models are presented.

3.1 Platform description

For the development of the HiL system, dSPACE SCALEXIO system was used. This system was bought by Porsche Engineering Services to provide automated ECU testing. However, there has been delay in the development of the ECU, that was supposed to be connected to HiL. Therefore, I had chance to use this platform for development of models mentioned in this thesis.

The SCALEXIO system in Porsche Engineering is a multicore system originally developed to be working with 3 ECUs. Therefore, this system is equipped with various I/O and BUS capabilities. Moreover, 2 FPGA DS2655 together with 6 DS2655m1 I/O modules are built in the HiL to provide accurate PMSM real-time simulations.

CAN, FlexRay, Lin and Automotive Ethernet bus systems can be simulated on the HiL system.

3.1.1 ECU description

The ECU used in this thesis was originally developed for Porsche Taycan. Special derivation of this ECU with no High Power stage was used. Moreover, Porsche Taycan is already in production since 2019, thus ECU software is finalized.

Figure 3.1: SCALEXIO Rack illustration [1]

In the real vehicle, ECU works with 3 phase PMSM placed in the front axes and can work with DC current up to 300 A. High Voltage system is built on 800 V.

High Power stage is removed and therefore, all logic signals sent from High Power stage to ECU must be simulated to ensure proper behaviour.

Cable Harness

To provide conductive connection between the HiL and ECU, cable harness has to be used. This harness is made on specification of the manufacturer of the ECU and manufacturer of the HiL. Cable harness is terminated with dSPACE hypertac connector on the HiL side and various connectors on the ECU side.

Connections were made for following signal groups:

- Power Supply
- Bus
  - CAN
  - FlexRay
- Temperature sensors
- Current sensors (DC and AC)
- Voltage sensors
- Internal diagnostic
- Gate signals
- Resolver

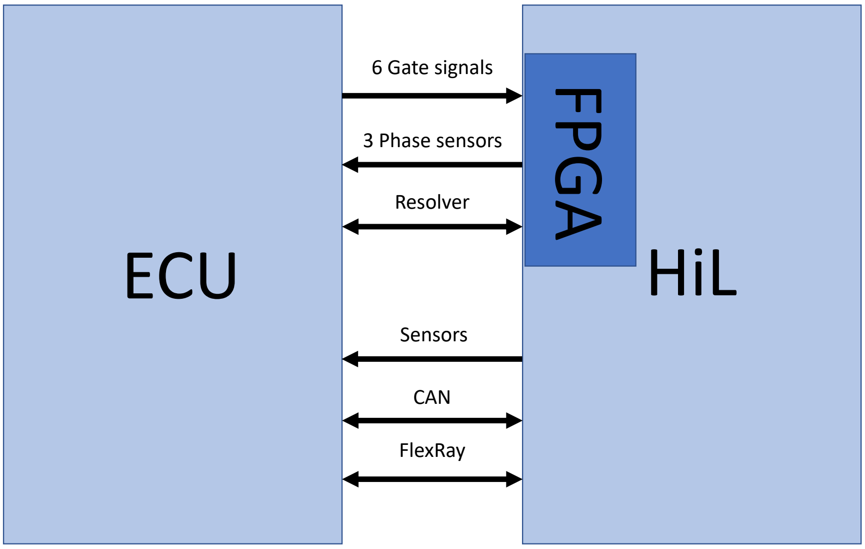


Figure 3.2: Schematic of HiL and ECU connection.

3.1.2 ConfigurationDesk

ConfigurationDesk is developed by dSPACE and is the main integration tool for HiL related processes. It provides graphical interface for implementation of behaviour models. It is possible to view the signal path from ECU through the I/O of the HiL to behaviour model.

When the SCALEXIO platform is connected to control PC, ConfigurationDesk automatically loads hardware description files from Platform and correct mapping on IO function can be finalized.

ConfigurationDesk is divided into 7 main working views, where following parameters can be managed.

Project - view set is general window for project management. Project version with all the changes are listed here, together with all important project related files

Model Function - is view, where creating and configuring of I/O functions in model structure, optimized for Simulink oriented users

Signal Chain - for working with complete signal chains, e.g., external devices, functions, and model port locks

Buses - view set for bus configuration

Tasks - table for tasks configuration

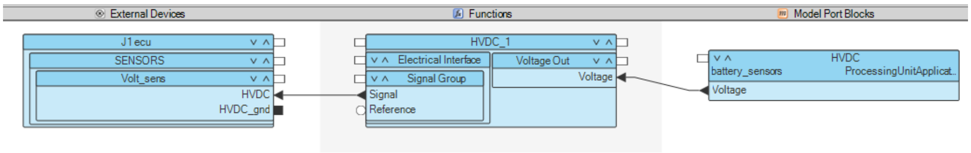


Figure 3.3: Example of ConfigurationDesk connection.

Multiple Models - for managing multiple models and multiple processing unit applications

Builds – for controlling the build process

In the developed simulation, default simulation task is set to $1ms$, with model priority:

In signal chain view, all inputs and outputs mentioned in 3.1.1 are mapped to corresponding I/O pin and afterwards behaviour model is connected. To visualize this chain, following figure 3.3 is enclosed.

3.1.3 FlexRay

To implement FlexRay bus communication into SCALEXIO platform, another dSPACE software must be used. FlexRay Configuration Tool can be used to set up monitoring and simulation of FlexRay nodes.

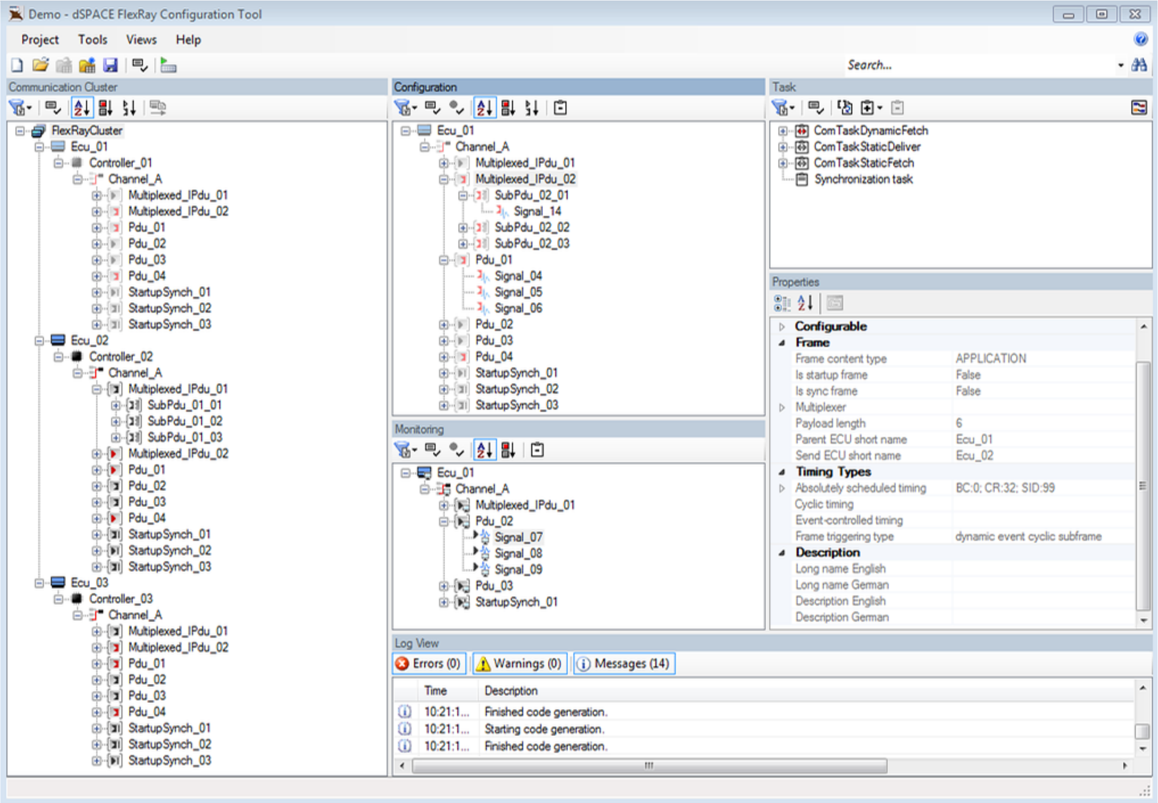
Simulation is set up based on the communication matrix (FIBEX or AUTOSAR XML). In the case of HiL simulations with one concrete ECU, user can set up the simulation by choosing the ECU and all other communication ECUs are directly created.

In the monitoring view, real ECUs connected to the HiL system must be placed in order to have access to received messages. Furthermore, user can add even simulated ECUs to the measuring window to easily verify bus traffic.

In the configuration view, simulated ECUs are entered, and their properties can be easily adjusted. Developer can adjust wake-up frames, signal scaling, minimal, maximal and default value etc.

In the task view, corresponding tasks for signal sending and receiving are created. Where Rx stands for messages sent from platform to ECU, Tx for messages sent from ECU to platform.

After successful set up, developer can generate RTI FlexRay Configuration Blockset for Simulink to have a possibility for behaviour model connection. [8]



Configurable

Frame

Frame content type	APPLICATION
Is startup frame	False
Is sync frame	False
Multiplexer	
Payload length	6
Parent ECU short name	Ecu_01
Send ECU short name	Ecu_02

Timing Types

Absolutely scheduled timing	BC:0; CR:32; SID:99
Cyclic timing	
Event-controlled timing	
Frame triggering type	dynamic event cyclic subframe

Description

Long name English	
Long name German	
Description English	
Description German	

```
Time        Description
10:21:1...  Finished code generation.
10:21:1...  Starting code generation.
10:21:1...  Finished code generation.
```

Figure 3.4: Example of FlexRay Configuration Tool dSPACE [8].

E2E protection

E2E (End-To-End) protection is implemented when data safety exchange must be ensured. When the E2E protection is used, faults in lower layers can be detected, and communication is protected against communication faults, such as random hardware faults introduced by the MCU hardware, communication peripherals, transceivers, communication lines or other communication infrastructure.

The main concept of E2E protection is to add additional parts to the message, where the sending ECU encodes information about the content of the ECU. Receiving ECU then decodes the information and compares it with the content of the message.

In the concept of the E2E, so called message counter is also present. Message counter increments with every sent message. Message counter has a define bit length and when the maximal possible number is reached, message counter is resettled to 0. [20]

CRC. Cyclic Redundancy Check is one of the methods, how to detect faults in the communication. In mathematical way, CRC can be described as a polynomial division. The data is interpreted as the coefficients of a giant polynomial which is divided by a given CRC polynomial. The remainder of this division is the CRC.

"Polynomial division of polynomial $p(x)$ by another polynomial $q(x)$ can be expressed as finding $s(x)$ so that there is $r(x)$ (the remainder polynomial) with a degree of less than the degree of $q(x)$ s:

$$p(x) = s(x) \cdot q(x) + r(x) \tag{3.1}$$

Using this, the set of polynomial congruence classes F2[x]=pCRC(x) can be defi

ned: Each element r(x) within this set is one of the possible remainders and represents all polynomials which leave r(x) as the remainder when divided by pCRC(x). The computation of the CRC is a polynomial division which computes the remainder of our data-polynom (after it's multiplied by x N for technical reasons). So finally, calculating the CRC is defined as finding a polynomial b(x) so that there is an r(x) with a degree of less than N so": [21]

$$a(x) \cdot x^N = b(x) \cdot p_{CRC}(x) + r(x) \tag{3.2}$$

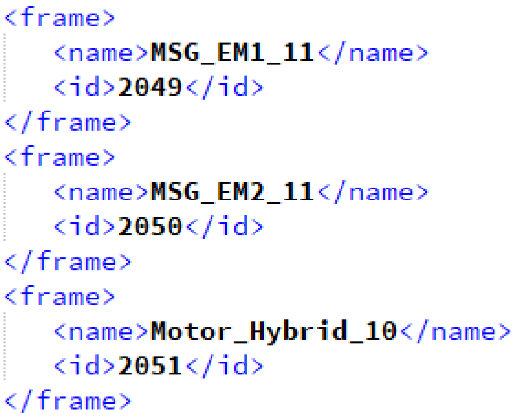
There are several more effective ways how to calculate CRC with bit-oriented algorithms. This is also the way, how the CRC algorithm is implemented

As direct implementation of chosen algorithm cannot be enlisted, algorithm mentioned in [20, p. 15] can be used as a close example of implementation.

- 4-bit counter is used (0 to 15) and counter is incremented with every sent message
- message key for the CRC calculation is 8 bits long.
- Each data, which is protected by a CRC owns a dedicated DataIDList which is deposited on the sender site and all the receiver sites.

Only the important PDUs, where data safety must be ensured, contain the E2E protection. Mostly messages regarding safety (Airbag or Active Discharging), control mode messages (requested control mod, requested torque), or battery management system messages.

To implement the above mentioned, C-code modules can be implemented in FlexRay Configuration Tool to provide access to bit format of the message. Moreover, FlexRay Configuration Tool also provides possibility to implement xml structure, where each PDU has assigned *ID* to be easily identified.



```
<frame>
    <name>MSG_EM1_11</name>
    <id>2049</id>
</frame>
<frame>
    <name>MSG_EM2_11</name>
    <id>2050</id>
</frame>
<frame>
    <name>Motor_Hybrid_10</name>
    <id>2051</id>
</frame>
```

Figure 3.5: Example of xml structure for PDU identification.

3.1.4 CAN

For CAN bus implementation and settings, ConfigurationDesk is used. In the card "Buses" user can upload corresponding bus description file (usually called K-Matrix). According to this description file, user can decide which messages should be simulated and which messages are sent by the real ECU.

User tab "Buses" has various subsection.

- **Bus Configurations** tab provides overview of the bus settings
- **Bus Access Requests** provides possibility to choose on which hardware board and channel is the simulation running
- **Bus Simulation Features** tab is used for setting-up of the simulated messages on the HiL system.
- **Bus Inspection Features** tab is used for setting-up of the receiving messages by the HiL system from ECU.
- **Bus Manipulation Features** tab is used for real-time manipulation in feed-through mode. (This mode is not used with HiL systems)
- **Bus Configuration Function Ports** tab is used for connections of the messages to Simulink models.

3.2 Mathematical models - SCALEXIO

Mathematical model is a system of differential equations describing certain physical aspects. Mathematical model of electrical machine provides possibility to study its behaviour in controlled conditions. In case of HiL testing, usage

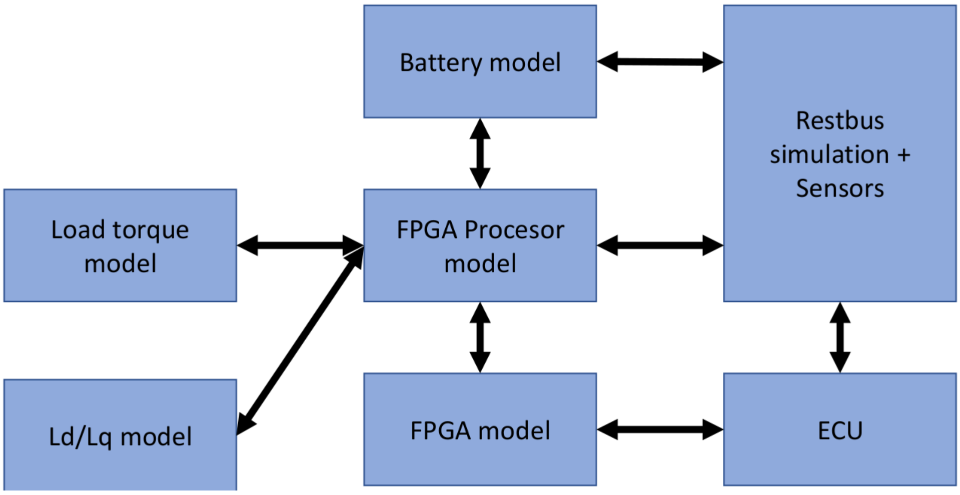


Figure 3.6: Structure and connections of the models.

of mathematical models can help to verify ECUs software and more over, study the system at its physical limits.

To provide overall system structure overview figure 3.6 is present. In this figure, models connection are shown altogether with the ECU. However these connections are only schematic. Full description can be found in section 3.1.1 and 4.1.

All models mentioned in this section are running on the processor board of the SCALEXIO system with task *1 ms*.

3.2.1 FPGA model - processor part

To handle real-time FPGA simulation on the dSPACE SCALEXIO platform, additional model running on the real-time PC must be added. This model works as an interface between FPGA calculations and ControlDesk for model control.

In this model, equivalent blocks for the structure created for FPGA are generated. This provides user option to parametrize the model, visualize its internal values and also connect calculated FPGA signals to other necessary systems.

dSPACE provides special Simulink models blocks to set up communication with the FPGA over the IOCNET (optical connection inside the SCALEXIO system). They are so called "Register blocks" and one block can handle up to 32 bits of data. Various calculations, that can be handled on the real-time pc are not time critical and are pre-evaluated on the real-time pc and then sent to FPGA.

For PMSM sub-model, following parameters can be set up and online manipulated:

- Stator resistance R_s
- L_d and L_q inductance
- Flux of permanent magnets Ψ
- Transformation constant
- Machine topology

Additionally, following signals are received from FPGA card.

- i_a, i_b, i_c
- i_d, i_q
- M_{PMSM}
- u_d, u_q
- u_{emf}

3.2.2 Load torque model

Informations in following section are taken from [12].

As mentioned in section 4.1, electrical transients of the PMSM machine and corresponding mechanic is calculated inside of the FPGA card. These transients change with very short time constants and therefore classical Simulink model with time period around 1 ms is not suitable.

However, M_{load} changes with reasonable time constants and Simulink model for the torque load calculation can be used. Corresponding M_{load} components are described in the following paragraphs.

There are various forces in real car, which differ in the direction and their magnitude. In this subsection simplified load model of the car is described. The drag force, brake force, rolling resistance force is used in the calculation.

The motion of vehicle can be described by various differential equations with a different complexity. There are many simplifications introduced for purposes of this HiL system. To keep the the computing complexity low and also satisfy the precision of simulation, these quantities are taken in context.

- The model is made as a longitudinal. Lateral movements of the vehicle are not calculated, because those quantities do not significantly affect power electronics.
- The vehicle is considered as ideal rigid.
- The resultant force is in the center of gravity

Gravitational force

Any two objects with a mass attract each other with a force called gravitational force. On earth gravity gives weight to physical object. This force is directly proportional to the product of their masses and inversely proportional to the square of the distance. For our proposes when the distance between a vehicle and Earth is from the Earth size point of view constant, we will consider a gravitational field with gravitational constant. Gravitation force is defined as [22]:

$$\vec{F_g} = m \cdot \vec{g} \tag{3.3}$$

Even though in the HiL testing road profile doesn't have to be simulated, in the section 3.2.2, the vehicle was moving on a real track where the track angle α was changing. According to this, following equation can be written as:

$$\vec{F_g} = \vec{F_g} \cdot \sin\alpha = m \cdot \vec{g} \cdot \sin\alpha \tag{3.4}$$

Rolling resistance force

For better simulation results, rolling resistance was added to the load model of the vehicle. According to [22] rolling resistance is force acting in opposite direction of vehicle motion. The magnitude of the force is directly proportional to the normal component of the gravitation force. The equation (3.5) is describing rolling resistance force.

$$\vec{F}_{RollingRes} = \mu_0 \cdot |\vec{F_g}| \cdot cos(\alpha) \tag{3.5}$$

where μ_0 is a friction coefficient.

Friction coefficient is strongly affected by many physical aspects and to simulate a correct behaviour of friction coefficient car velocity, temperature, materials of the wheels and surface, inflation of the tires and many more physical aspects must be taken in consideration. Simulation of all these parameters would take a lot of simulation time and as a result, μ_0 is considered to be constant with value of 0.01. This value is provided by [23]. [12]

Brake pedal

Brake pedal is also simulated in vehicle dynamics model for the needs of HiL testing. Input for brake pedal is stimulated in GUI by slide-bar with value range $0 - 100$. Braking force is then calculated by (3.6)

$$\vec{F}_{Brake} = \vec{Br} \cdot k \tag{3.6}$$

Br is the value stimulated via user interface, k is linear coefficient.

Even though this linear equation doesn't precisely describe non-linear physical phenomena, for the purposes of the HiL testing linear equation is sufficient.

Drag force

A drag force is the resultant force of forces acting opposite to the relative motion of any object moving in respect to the surrounding fluid. In the vehicle dynamic model is drag force implemented by following equation (3.7),

$$\vec{F}_{Air} = \frac{1}{2} \cdot \vec{v}^2 \cdot A \cdot C_D \cdot \rho_{Air} \tag{3.7}$$

where $\vec{v}$ is vehicle velocity relative to fluid, A is the cross section area,C_D is a drag coefficient and ρ_{Air} is in this case density of air. This equation is formally identical to [24] as well as values used. [12]

Resultant force

Resulting load force is calculated as sum of forces described in equations 3.7, 3.6, 3.5.

$$\vec{F}_{Load} = \vec{F}_{Air} + \vec{F}_{RollingRes} + \vec{F}_{Brake} + \vec{F}_g \tag{3.8}$$

$\vec{F}_{Load}$ is afterwards recalculated to appropriate torque with equation 3.10

$$\vec{F}_{Load} = \frac{1}{2} \cdot \vec{v}^2 \cdot A \cdot C_D \cdot \rho_{Air} + \mu_0 \cdot |\vec{F}_g| \cdot cos(\alpha) + k \cdot Br + m \cdot \vec{g} \cdot sin(\alpha) \tag{3.9}$$

$$\vec{M}_{Load} = \vec{F}_{Load} \cdot r \tag{3.10}$$

Where r is the car wheel diameter.

Gear box

In order to keep the PMSM in better efficiency areas, there is a one speed gearbox in the car. Gear box mathematical model is simulated by equation (3.11). This simplified model is implemented for easy recalculation of load torques from side of the vehicle to the side of the EM-machine,

$$M_{Wheel} = M_{EM} \cdot i_i \tag{3.11}$$

where M_{Wheel} is torque on output shaft, M_{EM} is torque on input shaft from PMSM and i_i is a conversion ratio

$$i_i = \frac{\Omega_{in}}{\Omega_i} \tag{3.12}$$

This equation is identical to the equation used in [25].

Parameters fitting

Even with the proposed load torque model, various parameters are unknown or inaccurate. Therefore Least square method of parameter fitting is introduced. This method uses numerical optimization in software Wolfram Mathematica and real vehicle measurement on race track.

Least square method is regression analysis method developed in the early 1800s by Karl Friedrich Gauss, Adrien Marie Legendr. The unknown parameters are estimated by minimizing the sum of the squared deviations between the data and the model. This method can be used for linear models with overdetermined parameters. [26]

Because torque load model is linear model in its parameters in polynomial form. This equation combines equation from load model 3.9 and 4.22 and fully describes movement equation.

$$M_{PMSM1}(t)+M_{PMSM2}(t)-\beta_0 sin(\alpha(t))-\beta_1\omega^2(t)-\beta_3 Br(t)-\beta_4\epsilon(t)-\beta_5 cos(\alpha(t)) = 0 \tag{3.13}$$

where $M_{PMSM1}(t), M_{PMSM2}(t), \alpha(t), \omega(t), \epsilon(t), Br(t)$ are time variant parameters, that can be extracted from a real car measurement, whereas β_i are unknown or problematically measurable parameters. To overcome this issue, equation 3.14 is furthermore adapted to following form, where a new member of equation is added with additional point measurement. Because this overdetermined system is solved by minimization, every member is also square powered. Driving torque is simplified as a sum of driving torques from front a rear machine. $M_{PMSM1}(t) + M_{PMSM2}(t) = M_{PMSM}$.

$$\sum_{t_1}^{t_n}(M_{PMSM}(t) - \beta_0 sin(\alpha(t)) - \beta_1\omega^2(t) - \beta_3 Br(t) - \beta_4\epsilon(t) - \beta_5 cos(\alpha(t)))^2 = 0 \tag{3.14}$$

Using a numerical minimization method, corresponding β_i that fits the system with smallest deviation, is obtained. Fitted model is presented in the end of this thesis.

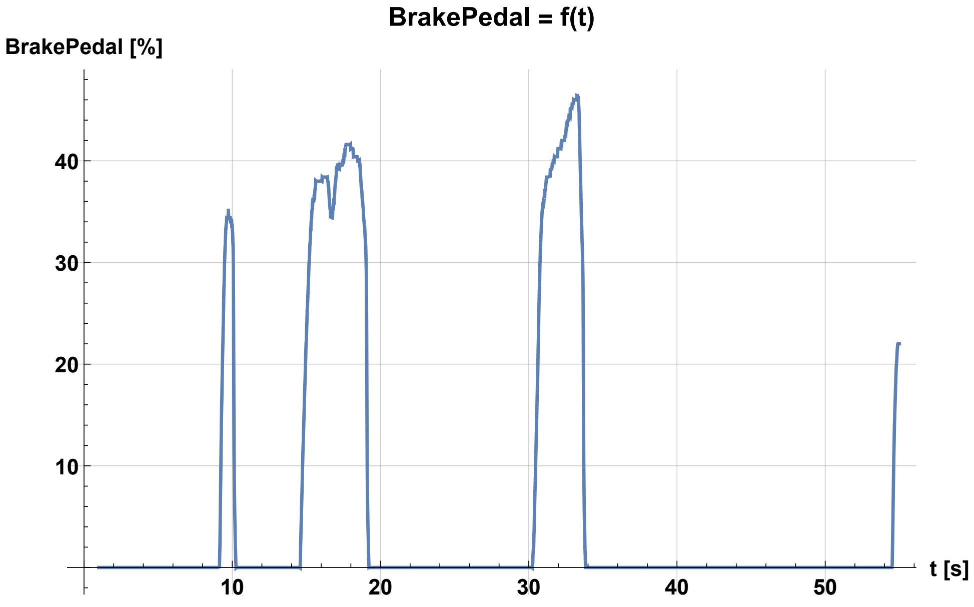


Figure 3.7: Measured brake pedal on race track.

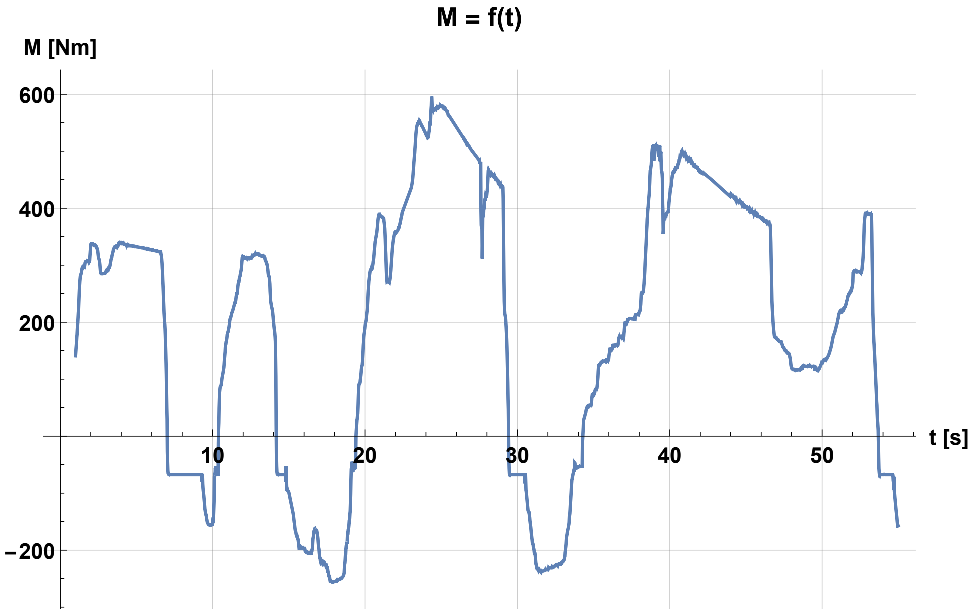


Figure 3.8: M_{PMSM} measured on the race track.

3.2.3 Battery model

Information in following section are taken from [12].

Nowadays, battery system is the most common storage of energy for automotive industry. Majority of electric cars is using battery system to provide electric power for motors. New types of batteries with bigger capacity and energy density are developed. Price of the battery system is continuously decreasing, but the battery is still one of the most expensive parts of an electric car.

There are several mathematical models of batteries with different complexity and accuracy. Electrochemical models can be used to optimize physical aspects of the batteries. However, the system complexity is unnecessarily complicated. [27]

Proposed battery model for PWR-Hil is based on electrical model combining current source and resistor. This battery system was mainly developed to observe the energy flow between the battery and the PMSMs. Another point was to test the power inverter behaviour in connection with battery and not to simulate the battery pack with its full complexity.

Consequently, battery model with equivalent circuit on figure 3.9 was chosen and following assumptions were made.

- Temperature of the battery cell does not change during the simulation. As a result temperature dependent values are considered to be constant.

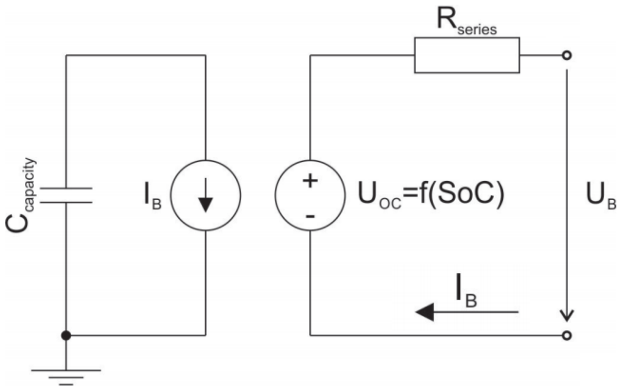


Figure 3.9: Battery cell equivalent circuit.

Property	Value	Units
Nominal Voltage	3.7	$V/cell$
Charge up voltage	4.2	$V/cell$
Cut off voltage	2.8	$V/cell$
Serial Resistance	0.51	$m\Omega$

Table 3.1: Specific values of battery cell

- Transient response of battery cell is not taken in account.
- Self discharging and ageing is not implemented.
- Serial battery resistance is SOC non-dependant.

Terminal voltage of one battery cell is calculated with equation 3.15

$$u_B = u_0 - R_{Series} \cdot i_B \tag{3.15}$$

Specific values of battery cell used in simulation are shown in table 3.1

Battery pack

In order to get the correct voltage on electrical machine, battery cells must be organized into a battery pack. Cells are connected in series and final voltage is described by equation 3.16.

$$u_{BP} = u_{Cell} \cdot n_S \tag{3.16}$$

where u_{BP} is terminal voltage of battery pack, u_{Cell} is voltage of one li-ion cell and n_S is number of cells connected in series. To increase the capacity of the battery pack, cells must be organized in parallel. Due to the same resistance, current splits equally between all of them.

$$i_{Cell} = \frac{i_{BP}}{n_P} \tag{3.17}$$

Where i_{BP} is terminal current of battery pack, n_P is number of packs connected in parallel.

In a real battery system, cells are not stressed equally, because the parameters are not the same. To avoid unnecessary ageing, battery is normally equipped with battery management system. [28]

Open-circuit voltage

Open-circuit voltage changes with the state-of-charge (SOC) of the battery cell and there is a non-linear relation. As a result, open-circuit voltage is implemented as function of SOC. Battery model all together with car dynamic was programmed in Simulink and therefore this battery property was implemented as a look-up table. [28]

Function $U_0(SOC)$ is shown on figure 3.10. When the battery is fully charged, SOC is equal to 1. When the battery is discharged SOC is equal to 0. Data for this model were internally provided and measured in Porsche Engineering Group.

SOC is function of battery current and can be described by equation 3.18,

$$SOC = \frac{1}{Q_C} \int i_B dt \tag{3.18}$$

where Q_C is battery capacity and i_B is a battery current. [28]

Intermediate circuit

Intermediate circuit, which is used in the real power inverter and in the simulation, can be described as intermediate circuit of voltage type. Large capacitor is connected to the battery on one side and to IGBT stage on other side. This capacitor is installed to stabilize voltage and as a source of peak energy. [29]

Nowadays, most of the power inverters are developed as voltage types in automotive industry.

Intermediate circuit is schematically shown on figure 3.11. The u_{out} can be calculated as

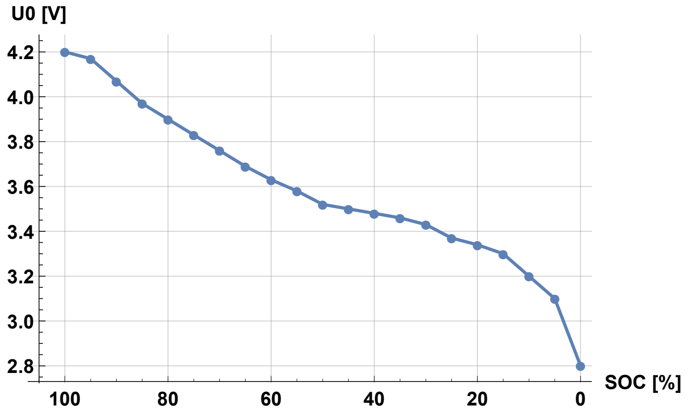


Figure 3.10: Open-circuit voltage

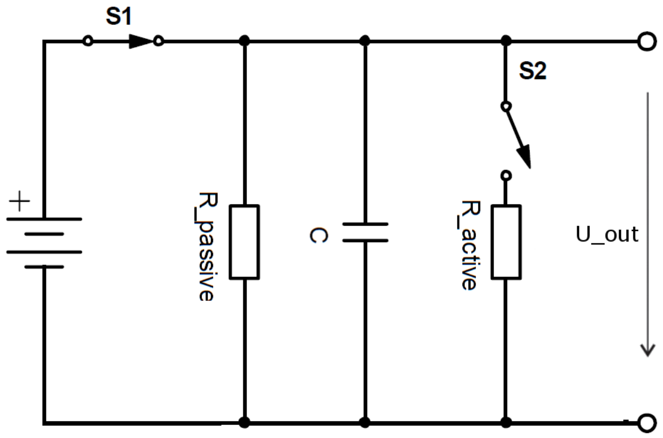


Figure 3.11: Intermediate circuit - active state.

$$0 = u_b(t) - \frac{1}{C} \int i_C dt \tag{3.19}$$

$$\frac{1}{C} \int i_C dt = u_{out} \tag{3.20}$$

Active Discharging

Various conditions must be fulfilled to start active discharging of the intermediate circuit. These conditions are controlled by the power inverter ECU and the external safety computer. Active discharging must be activated in case of a crash or in case of a hardware emergency turn off.

Before the active discharging request is sent, battery contacts must be opened and battery must not be connected to the charger. Additional resistor

τ_{Active}	$0.21s$
$\tau_{Passive}$	$39.48s$

Table 3.2: Time constants of intermediate circuit.

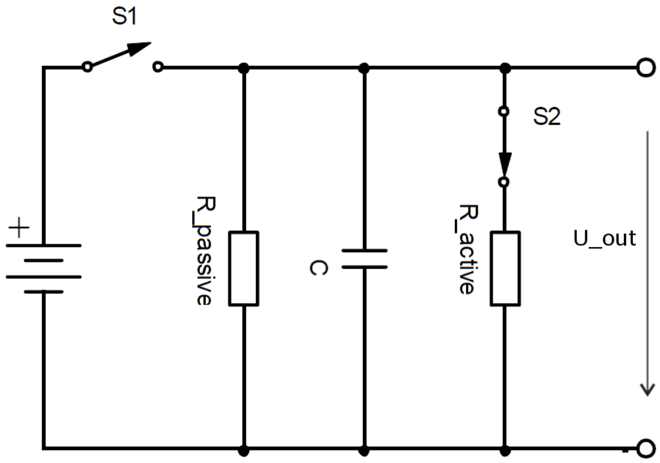


Figure 3.12: Intermediate circuit - active discharging.

for discharging is then connected to the main capacitor and IGBT transistors are closed. According to internal requirements, intermediate circuit must be discharged under $60V$ in 5 seconds under any circumstances.

When the hardware is turned off, discharging of the main capacitor is done only by $R_{passive}$ resistor. Time constants of active and passive discharging are shown in table 3.2

Active discharging is schematically shown on figure 3.12 and can be described with equation,

$$C \cdot \frac{\mathrm{d}u_{out}(t)}{\mathrm{d}t} = -\frac{u_{out}(t)}{R} \tag{3.21}$$

where R is in case of active discharging calculated as parallel combination of R_{Active} and $R_{Passive}$. In case of passive discharging, the resistor value equals to $R_{Passive}$. $u_{out}(0)$ is voltage on capacitor in the moment of disconnection.

When the active discharging is activated by the ECU, big discharging currents are flowing through the resistor. In order not to burn down the discharging resistor, PWM switching is used.

3.2.4 Restbus model

Restbus simulation model provides interface between models and buses. This model collects all signals from various models, manipulate them and afterwards

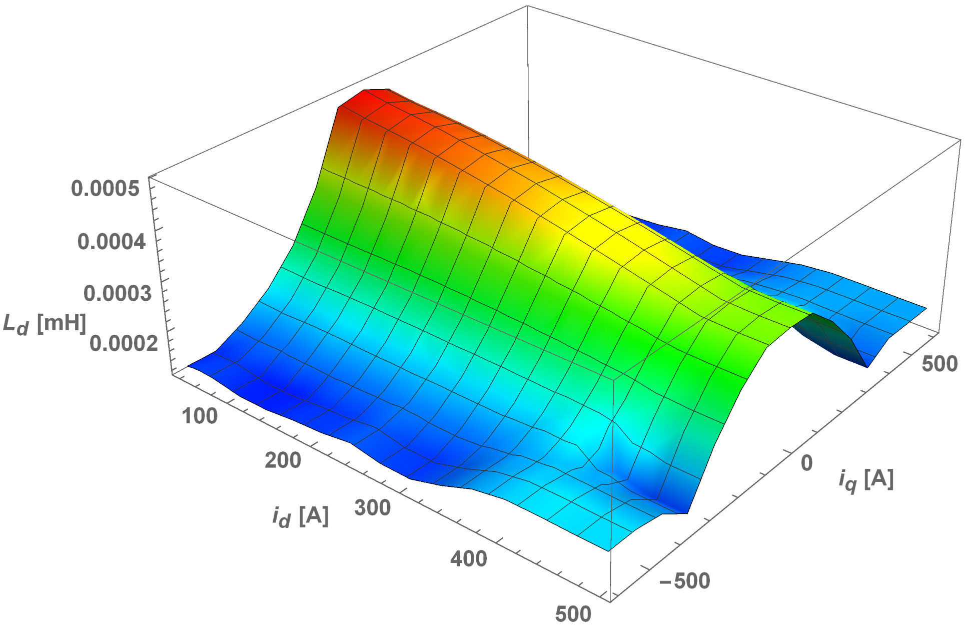


Figure 3.13: L_d map

send the signals to corresponding bus system.

As an example, brake pedal signal can be presented. This signal comes from GUI, afterwards is sent to PMSM model and also to connected ECU.

This model is taken from Porsche Engineering libraries.

3.2.5 Ld/Lq model and sensor simulation

There are various non-linearities in real PMSM machine. Those linearities are neglected during the derivation of the mathematical model in order to get reasonable and solvable set of equation. Derivation of the PMSM model is described in details in section 4.1.2.

One of the neglected features is the magnetic hysteresis and saturation. To improve the overall simulation quality, L_d and L_q are implemented as a look-up table with following formula 3.22 and are presented on figure 3.13

$$L = f(i_d, i_q) \tag{3.22}$$

Additional non-linearities in the model are related to the sensor behaviour and their non-linear curves. Those relations are implemented as a table with saturation to over or under-values.

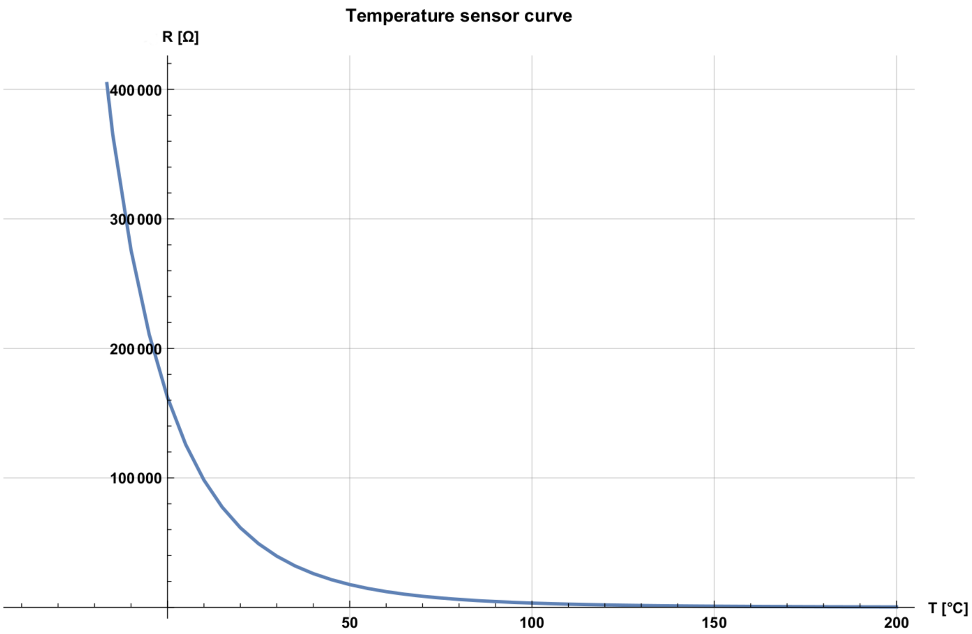


Figure 3.14: Example of temperature sensor curve.

Chapter 4

FPGA model analysis and improvements

This section provides overview of PMSM machine features. Afterwards, mathematical model of synchronous machine is presented in corresponding form for PMSM machine. Model structure on FPGA card is presented altogether with other FPGA needed models for successful HiL simulation. In the end of the chapter, possible model improvements are shown.

4.1 Mathematical models - FPGA

To handle correct simulation for the ECU, every real part, that had been removed form the ECU must be simulated. Also all sensors, PMSM machine, mechanic and Power Inverter models are introduced. There are 6 submodels in the created FPGA model. Each submodel has the mathematical description in following sections. However, to visualize model structure and connections, figure 4.1 is introduced.

As described in section 3.2.1, parametrization is done online via corresponding processor models and is not shown on figure 4.1.

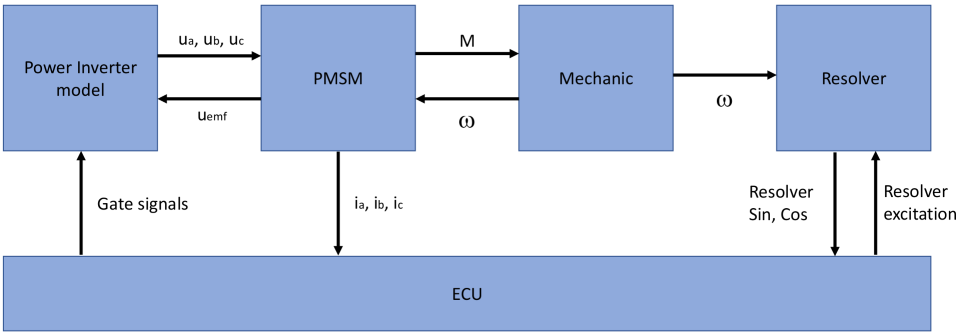


Figure 4.1: Submodels structure on FPGA.

All submodels mentioned in this thesis are generated with dSPACE XSG libraries. This libraries provides prepared models and tools, that can be implemented on FPGA. Corresponding blocks for communication has to be set for model running on processor part and also for model that is running on FPGA side.

Example of generated processor model part and FPGA model part can be found in section 5.

4.1.1 Coordinate transformation

In order to simplify differential equations of three-phase system, space vectors are introduced. These vectors can be represented in Cartesian coordinate systems, which are particularly chosen to suitably render the physical relations of the machines. The orientation on a certain vector for modelling and design of the feedback control loops is generally called vector orientation. [30]

A space vector of a physical quantity in three-phase system, under condition of sinusoidal distribution of winding, is described by equation:

$$\vec{x}_s(t) = K(x_a(t) \cdot e^{0j} + x_b(t) \cdot e^{\frac{2\pi}{3}j} + x_c(t) \cdot e^{\frac{-2\pi}{3}j}) \tag{4.1}$$

This is called $\alpha\beta$ **transformation**. Where K stands for transformation constant. If we rewrite complex number in this equation to real and imaginary part then we consider the real part to be named α and imaginary part β. The winding axis a will be placed in the real axis α and corresponding winding b and c are rotated by 120°, resp. −120° then equation 4.1 can be rewritten to 4.2 matrix form. Element x_0 appears, if system is not harmonically balanced. [30]

$$\begin{bmatrix} x_\alpha \\ x_\beta \\ x_0 \end{bmatrix} = K \cdot \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} x_a \\ x_b \\ x_c \end{bmatrix} \tag{4.2}$$

For a three-phase symmetrical system connected to star, following constrain 4.3 can be added and further simplification is possible.

$$x_a(t) + x_b(t) + x_c(t) = 0 \tag{4.3}$$

$$x_\alpha(t) = \frac{3}{2}Ki_a(t) \tag{4.4}$$

$$x_\beta(t) = K(\frac{\sqrt{3}}{2}i_a(t) + \sqrt{3}i_b(t)) \tag{4.5}$$

$\alpha\beta$ coordination system is fixed system with stator frame. However, in this coordination system, elements still behaves harmonically. In order to transform harmonic system to DC system, **DQ transformation** is introduced with following relations.

$$\begin{bmatrix} x_d(t) \\ x_q(t) \end{bmatrix} = \begin{bmatrix} \cos\theta(t) & \sin\theta(t) \\ -\sin\theta(t) & \cos\theta(t) \end{bmatrix} \begin{bmatrix} x_\alpha(t) \\ x_\beta(t) \end{bmatrix} \tag{4.6}$$

4.1.2 PMSM model

Permanent Magnet Synchronous Machine (PMSM) is an AC electric machine with 3 phase stator winding and the rotor with permanent magnets. A synchronous machine, as the name indicates, rotates at synchronous speed. From a historical point of view, synchronous machines were mostly used as generators of electrical power or motors that were rotating with constant speed and load. However, these machines were built with excitation field, that was generated by electrical current.

Nowadays PMSM machine uses permanent magnets as source of rotor flux and therefore no excitation field has to be used. To manufacture those magnets, rare Earth elements are used (for example SaCo, NdFeB) with high saturation B_r up to 1 T and high coercive force around 8 kA/m. Disadvantage of this solution is impossibility to regulate magnetic flux. There are two arrangements of the permanent magnets on the rotor:

- sinusoidal surface magnet machine, where $L_d = L_q$
- sinusoidal interior magnet machine, where $L_d \neq L_q$

With evolution of the switching elements in latest decades, synchronous motor can be used in as a regulated machine with speed or torque regulation. This solution provides high reliability with very good efficiency.

In comparison with IM, PMSM machines with the same rate power have higher efficiency, lower moment of inertia and lower weight. Disadvantage of PMSM is a smaller heat level, that can not be exceeded. If the temperature exceed approximately 150°C magnets are losing their magnetic characteristics. Also rotor position sensor must be used in order to ensure correct magnetic field orientation. [31] [22]

Because PMSM is a synchronous type of machine, following mathematical model will be derived from the model of a synchronous machine. Following conditions are taken in context:

- Symmetrical rotor winding

- Invariable air gap
- Sinusoidal distribution of windings
- Linear magnetization curve
- Core losses are neglected
- Symmetrical 3 phase power supply

Nowadays PMSM machines are mostly used with position rotor sensor (described in 4.1.3), the field orientation is always ensured and therefore, *DQ transformation* is the most convenient way to describe the machine model

In the following text, mathematical model derivation of synchronous machine is taken from [25] and following equations describing synchronous motor are rewritten.

$$u_d = R_s i_d + \frac{\mathrm{d}\Psi_d}{\mathrm{d}t} - \omega_1 \Psi_q \tag{4.7a}$$
$$u_q = R_s i_q + \frac{\mathrm{d}\Psi_q}{\mathrm{d}t} + \omega_1 \Psi_d \tag{4.7b}$$
$$u_f = R_f i_f + \frac{\mathrm{d}\Psi_f}{\mathrm{d}t} \tag{4.7c}$$
$$u_D = R_D i_D + \frac{\mathrm{d}\Psi_D}{\mathrm{d}t} \tag{4.7d}$$
$$u_Q = R_Q i_Q + \frac{\mathrm{d}\Psi_Q}{\mathrm{d}t} \tag{4.7e}$$
$$\Psi_d = L_d i_d + L_{df} i_f + L_{dD} i_D \tag{4.7f}$$
$$\Psi_q = L_q i_q + L_{qQ} i_f \tag{4.7g}$$
$$\Psi_f = L_{df} i_d + L_f i_f + L_{fD} i_D \tag{4.7h}$$
$$\Psi_D = L_{dD} i_d + L_{fD} i_f + L_D i_D \tag{4.7i}$$
$$\Psi_Q = L_{qQ} i_q + L_Q i_q \tag{4.7j}$$

Where subscripts d,q stands for equivalent stator equation. D,Q stands for rotor damper winding quantities and subscript f stands for quantities of field winding. However the construction of PMSM motor is less complicated and various members can be neglected.

PMSM machine doesn't have damper winding and thus, voltage equations (u_D,u_Q) and flux equations(Ψ_D,Ψ_Q) can be neglected with other mutual inductances ($L_{df}, L_{dD}, L_{qQ}, L_{fD}$). Furthermore field flux in PMSM machine is not generated by electrical current, but is generated by permanent magnets and therefore will be noted as $\Psi_p = const$ and voltage equation u_f together with flux equationΨ_f must be neglected. Because the real axis of the coordinate system is directly orientated to the preferred axis of the pole flux, the quadrature component of permanent magnets is zero.

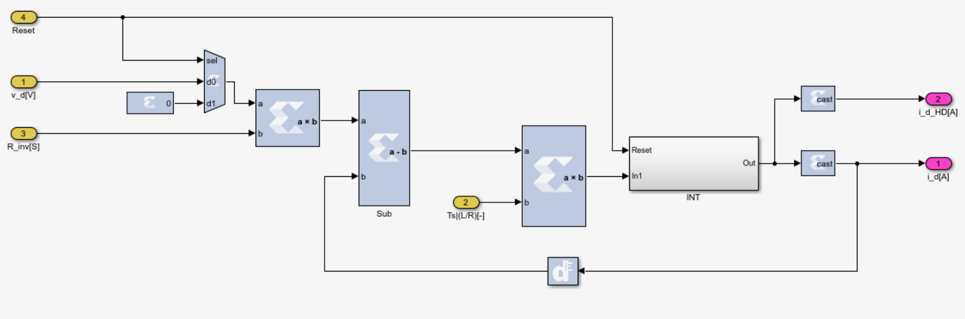


Figure 4.2: dSPACE XSG implementation of equation 4.12

After those simplifications, machine equations can be rewritten in following form.

$$u_d = R_s i_d + \frac{\mathrm{d}\Psi_d}{\mathrm{d}t} - \omega_1 \Psi_q \tag{4.8}$$

$$u_q = R_s i_q + \frac{\mathrm{d}\Psi_q}{\mathrm{d}t} + \omega_1 \Psi_d \tag{4.9}$$

$$\Psi_d = L_d i_d + \Psi_p \tag{4.10}$$

$$\Psi_q = L_q i_q \tag{4.11}$$

Substituting equations 4.10,4.11 into 4.8, 4.9 we get following two equations

$$u_d = R_s i_d + L_d \frac{\mathrm{d}i_d}{\mathrm{d}t} - \omega_s L_q i_q \tag{4.12}$$

$$u_q = R_s i_q + L_q \frac{\mathrm{d}i_q}{\mathrm{d}t} + \omega_s L_d i_d + \omega_s \Psi_p \tag{4.13}$$

From [30], The general torque equation mentioned is taken

$$M_{PMSM} = \frac{3}{2} K(\Psi_d i_q - \Psi_q i_d) \tag{4.14}$$

Substituting equations 4.10,4.11 into 4.14, final torque equation is obtained

$$M_{PMSM} = \frac{3}{2} K(\Psi_p i_q + i_d i_q (L_d - L_q)) \tag{4.15}$$

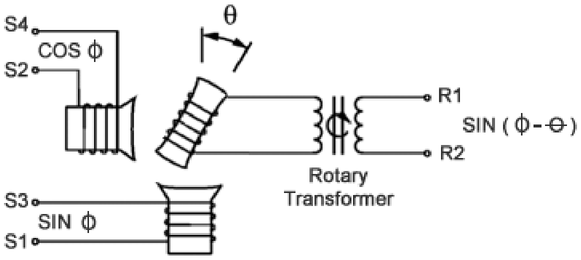


Figure 4.3: Resolver schematic [9].

4.1.3 Resolver model

In order to ensure correct behaviour and control of the machine, rotor position sensor must be used. Even though sensor-less position measurement technologies are developed, in user critical applications, rotor position sensor is often used.

There are several rotor positions sensors options on the market. However in dusty and mechanical demanding environment usage of digital sensors is not preferred option due to the device unreliability. Thus position measurement with analogue device with high endurance is preferred option. In case of the Porsche Taycan, ECU was developed to work with resolver.

Resolver is an electrical transformer device with three windings. One winding is used as a excitation winding usually on rotor and the other two windings are placed on stator with 90° shift. Those stator windings are usually called SIN and COS winding.

In this wiring, excitation winding is fed by ECU with AC voltage with high frequency. In the SIN and COS winding is afterwards induced voltage with corresponding shift based on the mechanical angle.

Induced voltage in SIN and COS winding can be written with following equation. [9], [30]

$$u_{sin} = k \cdot u_{0sin} \cdot \sin(\omega t) \cdot \sin(\theta) \tag{4.16}$$

$$u_{cos} = k \cdot u_{0cos} \cdot \sin(\omega t) \cdot \cos(\theta) \tag{4.17}$$

where k is the transformation ration, u_0 is the amplitude. Using the well-known formula from geometry $\tan\theta = \frac{\sin\theta}{\cos\theta}$, equations 4.17 and 4.16 can be rewritten to following form, where angle θ is obtained. Relations from mentioned equation can be also pictured as figures 4.4 and 4.5. Usually, the rotor position is estimated from the signal envelope.

$$\theta = \arctan \frac{u_{0sin} \cdot \sin(\omega t) \cdot \sin(\theta)}{u_{0cos} \cdot \sin(\omega t) \cdot \cos(\theta)} \tag{4.18}$$

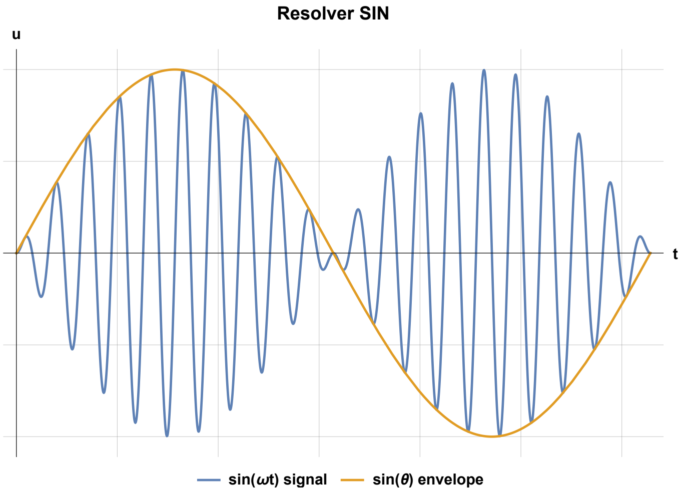


Figure 4.4: Resolver SIN winding.

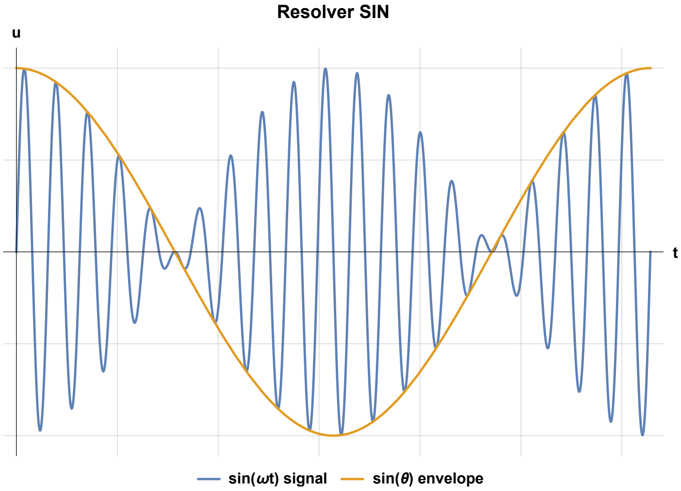


Figure 4.5: Resolver COS winding.

4.1.4 Power Inverter Model

To simulate Power inverter model, dSPACE proposed a model where the logical signals from IO submodels, altogether with Battery voltage, minimal switching currents, Forward Voltage diode and generated u_{emf} is used.

In this submodel, voltage on phase is calculated based on the state of the switch, which can be following:

- Shorten
- HSD
- LSD
- Open Current Positive
- Open Current Negative
- Open, no current

4.1.5 Mechanic model

In the mechanic submodel of the FPGA, revolutions per minute are calculated. In order to have this system fully described movement equation must be introduced. In this submodel torque generated by PMSM together with the load torque stands as input for the model.

According to D'alemberts principle acceleration torque is generated if there's imbalance between load torque M_{Load} from equation 3.10 and driving torque on shaft from equation 4.15. In case that driving torque is higher than load torque ($M - M_L > 0$) machine is accelerating in case that driving torque is the same as load torque ($M - M_L = 0$), machine is in steady state. In the last case, when load torque is higher then driving torque ($M - M_L < 0$), machine is decelerating. This is described by equation (4.19). Equations are formally identical to equations used in [25].

$$M_{PMSM} - M_L = M_{Dyn} \tag{4.19}$$

Where M_{Dyn} is dynamic torque. The dynamic torque can be expressed by equation (4.21)

$$M_{Dyn} = \frac{\mathrm{d}}{\mathrm{d}t}(J\Omega) \tag{4.20}$$

Because the rotational inertia doesn't change in time in our case $J = const$, following simplification is accepted.

$$M_{Dyn} = J \cdot \frac{\mathrm{d}}{\mathrm{d}t}\Omega \tag{4.21}$$

where Ω is mechanical angular speed, J is rotational inertia, M_load is from load torque model described in section 3.2.2. After substitution, resulting equation is 4.22.

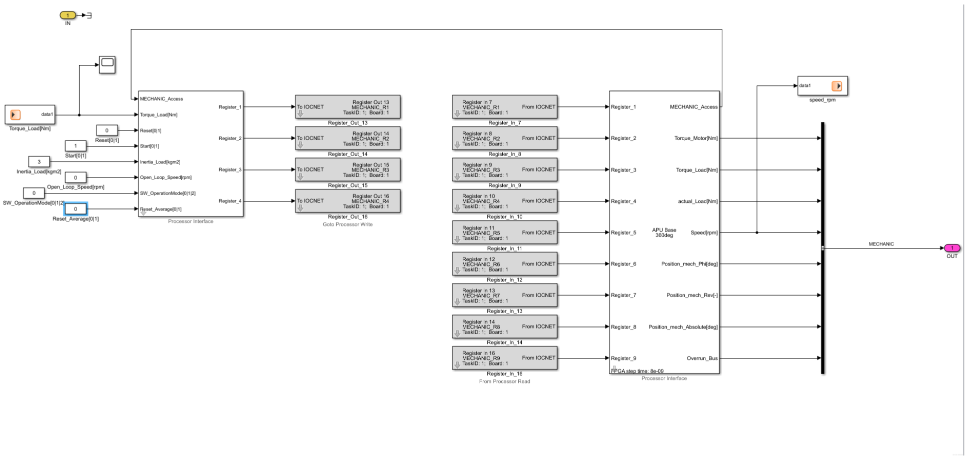


Figure 4.6: dSPACE XSG implementation of mechanic - processor model

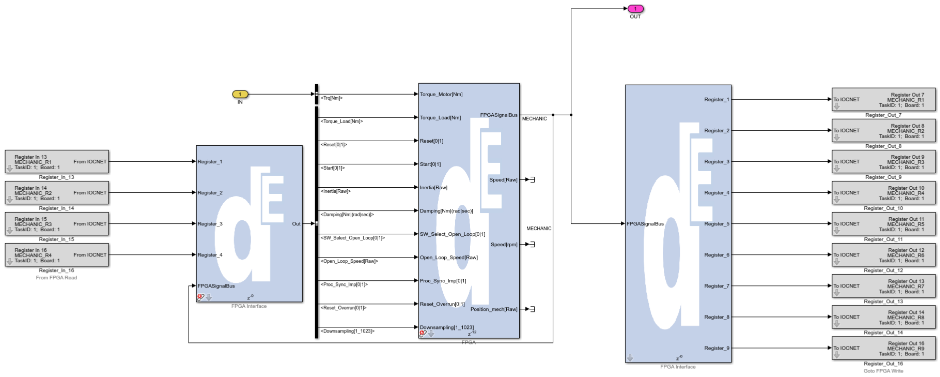


Figure 4.7: dSPACE XSG implementation of mechanic - FPGA model

$$M_{PMSM} - M_{Load} = J \cdot \frac{\mathrm{d}}{\mathrm{d}t}\Omega \tag{4.22}$$

4.1.6 IO model

To make the closed-loop simulation possible, direct connection of ECU and FPGA must take place. As described in 3.1, there is conductive connection between the ECU and the FPGA simulation card. In the model simulation, submodel with following elements was made.

- 7x Digital Input
- 5x Analogue output

Name	Value	Unit
k	0.003	V/A
q	2.5	V
u_{min}	0.5	V
u_{max}	4.5	V

Table 4.1: Common sensor equation values

For the purposes of closed loop simulation, 6 digital inputs were implemented to catch gate switching signal originally for IGBT. In this simulation, only the voltage threshold is specified.

1 digital input and 2 analogue outputs are used as source for excitation resolver field, in details described in section 4.1.3.

3 Analogue outputs are used as current measurement simulation. In the PMSM machine submodel, motor currents are calculated and afterwards sent to the IO submodel. In this submodel, scaling of real sensor must be implemented. Scaling is considered to be linear with following equation.

$$u_x = ki_x + q \tag{4.23}$$

where u_x is the output voltage of simulated sensor on phase x, k is the slope in unit V/A and q is sensor voltage offset. i_x is calculated phase current. Further more minimal voltage value u_{min} and maximal voltage value u_{max} has to be specified. Values implemented in the simulation are in table 4.1. Implemented IO model is shown in chapter 5

4.2 PMSM Model improvements

In following section, possible model improvements will be discussed and decided model improvement implementation will be presented.

Nowadays PMSM has become broadly used type of machine in transport systems. This brings new challenges for engineers in machine control and passengers comfort.

There are various non-linearities, that are neglected during the derivation of the PMSM model 4.1.2. However these non-linearities can strongly affect behaviour of the PMSM machine but the derivated model cannot be used to simulate this behaviour.

For example magnetic saturation of the machine can strongly affects harmonics structure of generated currents and absolute value generated torque, but in the derivation of PMSM model, linear magnetization curve is used.

This issue can be partly overcame by Ld and Lq tables described in 3.2.5 previously in this thesis. This method can only affect absolute value of generated torque, but cannot affect harmonic structure of generated currents.

Other neglected issue is the distribution of windings. In the derivation, this phenomena is neglected. However in the reality, windings are distributed in a slots and this phenomena can cause additional torque ripples.

To achieve better PMSM simulation results in reasonable time requirements, many new methods and model structures are developed. One option for more precise simulation is to use Finite Element Method (FEM). However those simulation are often very time requiring and they are not suitable for a real-time simulation.

In [32] author suggest new hybrid approach, where the table data are generated by FEM and afterwards they are used in FPGA real-time simulation. Model suggested in this work uses hybrid abc-dq frame with tables of linkage fluxes from FEM. This model can simulate all torque ripples, electromagnetic, reluctant and load cogging torque. However for the purposes of this thesis, similar approach was not possible due to unavailability of corresponding FEM model and model complexness on FPGA.

In [33], author suggests to use a loss d-q model of PMSM machine and compare this model to conventional PMSM model. In this model, core saturation effects can be simulated. However, various additional effects as torque ripples are not mathematically described in this model.

In [34], author suggest to use hybrid approach, where the table data are generated by FEM. In this work, model called *Spatial Harmonic* is presented. This model also includes torque and current harmonics contents induced by machine slots and back-EMF spatial configuration.

Model description

For the purposes of this thesis, approach based on [34] is implemented. However, FEM model of corresponding machine is not available and therefore, different approach of data generation is presented. This approach is based on real measurement on the test-bench. Unfortunately due to the COVID-19 crisis I was able to obtain just one measurement from the testbench. However, discussed algorithm can be easily extended.

In this approach, model is extended by variable inductance, which varies according to rotor angle θ and i_d, i_q current. To implement these relations, 3D tables are used. Moreover θ changes with very every calculation step and thus those 3D tables can't be implemented in processor model, but FPGA implementation is preferred. This structure is visualized on figure 4.8 and mathematically described:

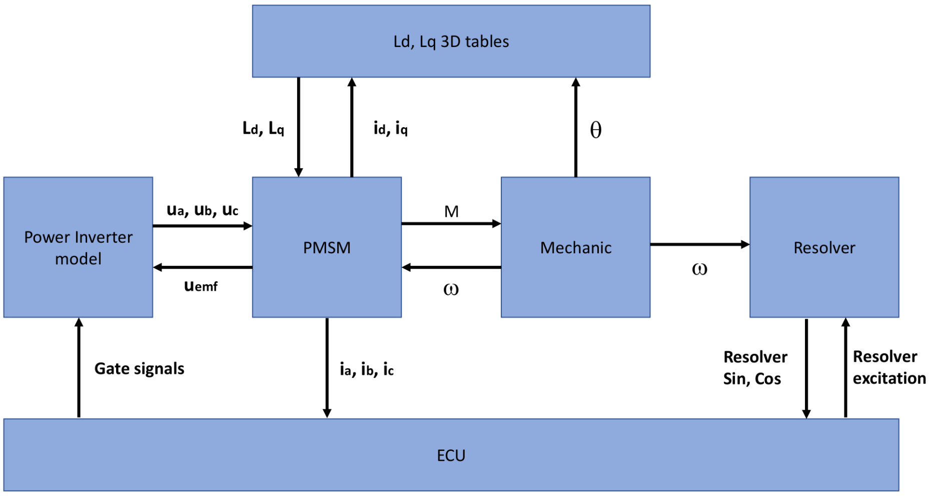


Figure 4.8: Submodels structure on FPGA with 3D tables.

$$L_d = f(i_d, i_q, \theta) \tag{4.24}$$

$$L_q = f(i_d, i_q, \theta) \tag{4.25}$$

To obtain those relations, following mathematical procedure will be presented on a concrete example of test-bench measurement. Quantities i_a,i_b,i_c and M were measured therefore all calculations are counting only with them. On figure 4.9 are measured phase currents, on figure 4.10 is measured torque.

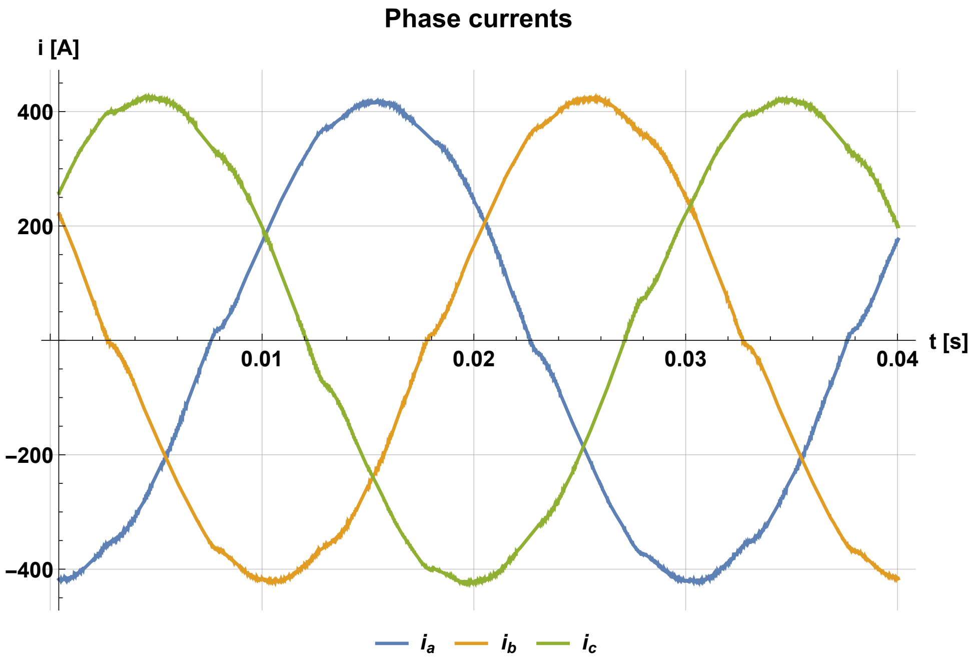


Figure 4.9: Measured phase currents.

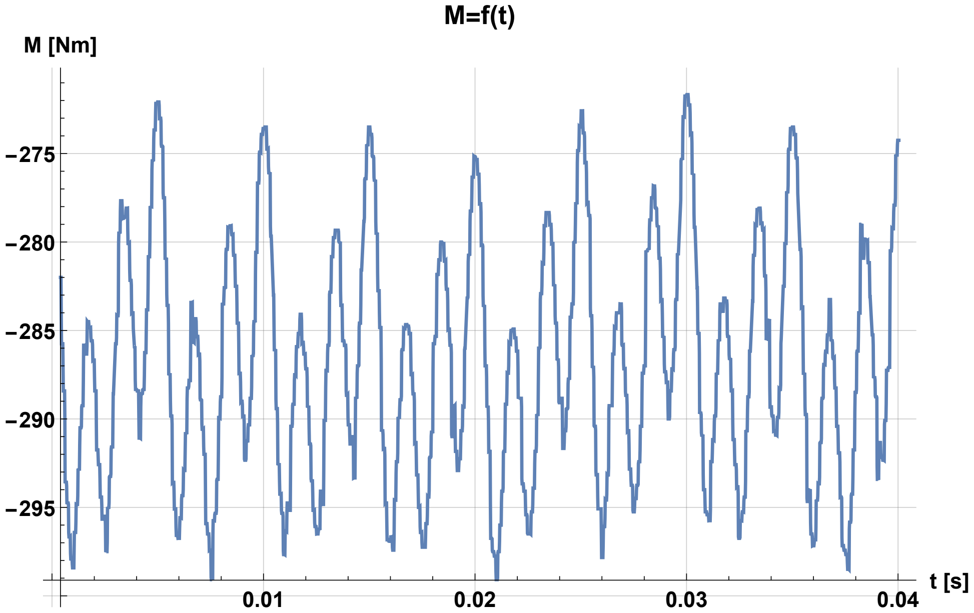


Figure 4.10: Measured torque.

To obtain i_d and i_q values of the measured i_a,i_b,i_c, first phase currents must be recalculated to $\alpha\beta$ frame. This can be easily done using equations 4.4 and 4.5.

However, for transformation of $\alpha\beta$ frame to dq frame requires knowledge of transformation angle θ. This angle was not measured and therefore different approach had to be chosen.

To obtain the angle first corresponding speed of the machine has to be found. This can be done using numerical methods to find best fit for measured phase current. The fit is in form $I_0 \sin(\omega t + \phi)$ and can be seen following figure 4.11. Using this formula, ω can be easily obtained.

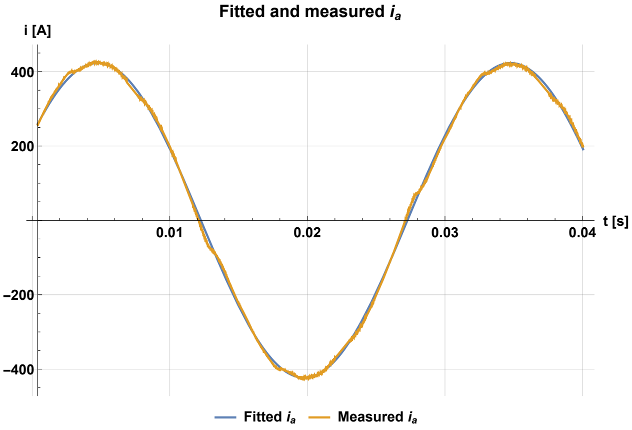


Figure 4.11: Measured and fitted i_a current.

However even though the angle θ can be easily obtained by integration of the angular speed, resolver offset is remains still unknown. To overcome this issue, following algorithm is introduced.

Firstly coordinate transformation $\alpha\beta$ to dq is adjusted to form:

$$\begin{bmatrix} x_d(t) \\ x_q(t) \end{bmatrix} = \begin{bmatrix} \cos(\omega t + \phi) & \sin(\omega t + \phi) \\ -\sin(\omega t + \phi) & \cos(\omega t + \phi) \end{bmatrix} \begin{bmatrix} x_\alpha(t) \\ x_\beta(t) \end{bmatrix} \tag{4.26}$$

where ϕ is the unknown offset parameter. Using the equation torque equation of PMSM in form 4.32, where L_d and L_q are obtained from maps mentioned in 3.2.5.

$$M_{PMSM} = \frac{3}{2} K(\Psi_p i_q + i_d i_q (L_d(i_d, i_q) - L_q(i_d, i_q))) \tag{4.27}$$

Direct solution of this equation system is not possible because its overdeterminess, and therefore, the least square method already described in section 3.2.2 will be used for the search of parameters with smallest deviation. M_{msr} stands for measured torque on the testbench.

$$\sum_{t_1}^{t_n} (M_{msr}(t) - M_{PMSM}(t))^2 = 0 \tag{4.28}$$

This formula is solved by numerical algorithms in Software Wolfram Mathematica. When the model with the smallest deviation is found, corresponding i_d and i_q can be easily obtained and afterwards can be used as boundaries in relation $L = f(i_d, i_q, \theta)$

To obtain data $L = f(\theta)$ without noise, first data are averaged over measured periods afterwards Fourier series equivalent is found using formula 4.31

$$a_n = \frac{2}{P}\int_P M_{msr}(t) \cdot \cos(2\pi t\frac{n}{P})dt \tag{4.29}$$

$$b_n = \frac{2}{P}\int_P M_{msr}) \cdot \sin(2\pi t\frac{n}{P})dt \tag{4.30}$$

$$M_{fourier}(t) = \frac{a_0}{2} + \sum_{n=1}^{N}(a_n \cos(2\pi t\frac{n}{P}) + b_n \sin(2\pi t\frac{n}{P}) \tag{4.31}$$

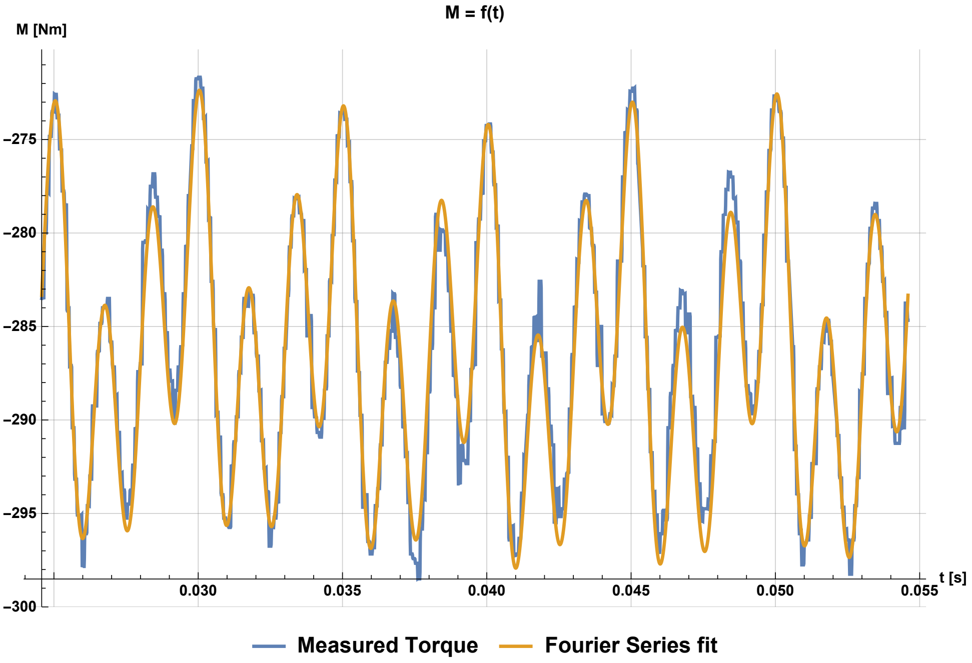


Figure 4.12: Measured and fitted torque.

To obtain value of inductance as a function of time, torque equation of PMSM is rewritten to form:

$$M_{fourier}(t) = \frac{3}{2}K(\Psi_p i_q(t) + i_d(t)i_q(t)L_{diff}(t))) \tag{4.32}$$

where L_{diff} stands for $L_{diff} = L_d - L_q$. In order to get reasonable values without switching frequency, $i_d(t)$ and $i_q(t)$ currents are fitted with linear approximation. Therefore, final $L_{diff}(t)$ is directly proportional to the Fourier series fit of the torque. Also the angular speed is know, thus inductance can be expressed as a function rotor angle.

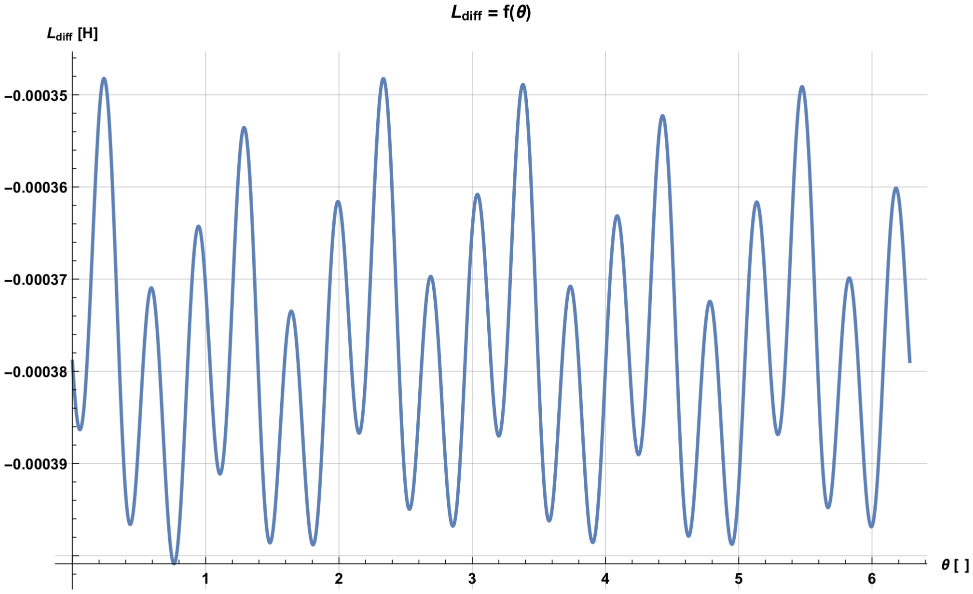


Figure 4.13: L_{diff} as a function of rotor angle.

To calculate L_d and L_q out of L_diff and generate corresponding 3D tables, I have decided to use following algorithm. This algorithm was chosen due to lack of additional informations and measurements. In order to generate the tables, L_{diff} has to be recalculated relative normalized inductance L_{norm}, which mean value is 1. Afterwards 3D table is generated with formula:

$$L_{3D}(i_d, i_q, \theta) = L_{2D}(i_d, iq) \cdot L_{norm}(\theta) \tag{4.33}$$

The equation 4.33 is mentioned in this form because of the lack of additional informations. If additional measurements were present, equation can be rewritten to form 4.34, where $L_{normX}(i_d, i_q, \theta)$ will be valid only in the range in specified i_d and i_q.

$$L_d(i_d, i_q, \theta) = L_d(i_d, iq) \cdot L_{norm1}(i_d, i_q, \theta) \cdot L_{norm2}(i_d, i_q, \theta) \ldots \tag{4.34}$$

The same equation only with different index is used for L_q map. On figure 4.14 can be seen a L_d as a function of i_q and $theta$ rotor angle.

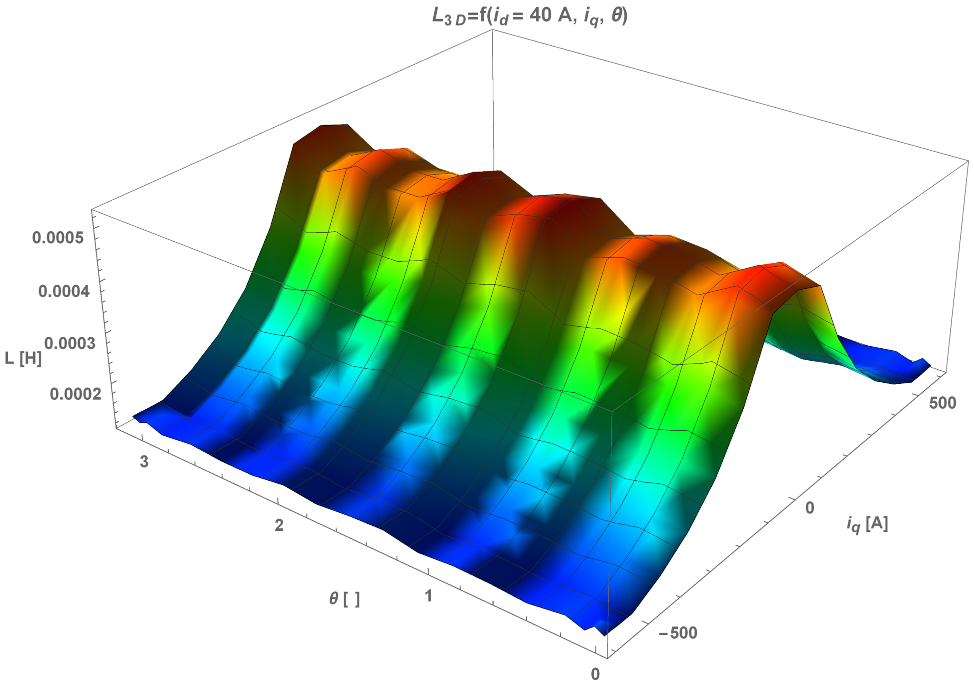


Figure 4.14: L_d as a function of i_q and *theta* rotor angle.

Chapter 5

Results

5.1 Platform configuration and mathematical models

5.1.1 Platform configuration

On figure 5.3, on the left side of the picture, Simulink models implemented in the HiL can be seen. The signal chain is in the middle part of the picture. *DCC_L and DCC_H* stands for DC current sensors simulation, where *DCC_L* is used with currents up to 200 A, *DCC_H* is used with currents form 200 A to 600 A. Corresponding curves are implemented in a Simulink models and afterwards sent to HiL I/O system. Other labels mostly stand for temperature simulations.

On figure 5.4, Basic layout in ControlDesk is presented. This layout was taken from other HiL project therefore, all labels are implemented in German language. The dashboard corresponds to the dashboard in the real Porsche Taycan. All warning signs with RPM and speed of the car can be shown. *KL30* stands for permanent power supply of the unit, meanwhile KL15 stands for the ignition signal.

On figure 5.5, user can see created INCA interface for internal variables. In the section *Overview*, user can see requested mode *vOPR*, operating mode *vSEQNO*, where 1 stands for the torque control. Label *vTMRQ* shows requested torque, meanwhile *vT_MOT_DLVR* stands for measured torque. In section *Temperatures*, measured values from the simulation are shown. In the section *Currents*, all measured currents are shown with calculate i_d *(IM)* and i_q *(IT)* currents. On the right side, active errors are displayed.

On figure 5.6, ConfigurationDesk settings for CAN bus is shown. In this settings, messages sent by *Battery Management System* are present. According to the German nomenclature, *BMS_IstStrom* stands for measured DC current, *BMS_IstSpannung* for measured voltage on the battery, *BMS_Spannung_ZwKr* is voltage measured in the intermediate circuit. *BMS_SOC* is clearly the battery state of charge.

Afterwards, on figure 5.1 measured data transfer on FlexRay bus can be seen. To better demonstrate both message counter and CRC, message counter is multiplied by 10 to have a better scaling.

On figure 5.2, created cable harness is shown. Two Hypertacs connectors are used on the HiL side with 9pins D-Sub connector. 50pins D-Sub connectors with other automotive connectors are used on the side of the ECU.

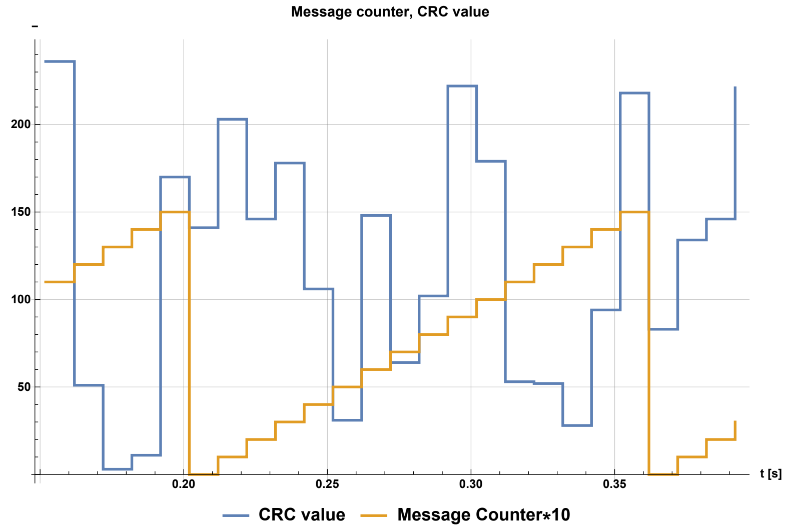


Figure 5.1: Flexray measurement with Message Counter and CRC

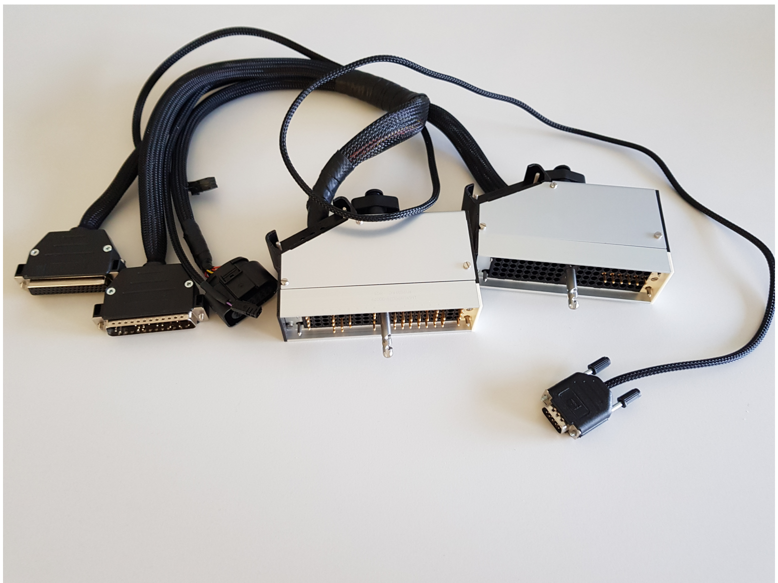


Figure 5.2: Created cable harness.

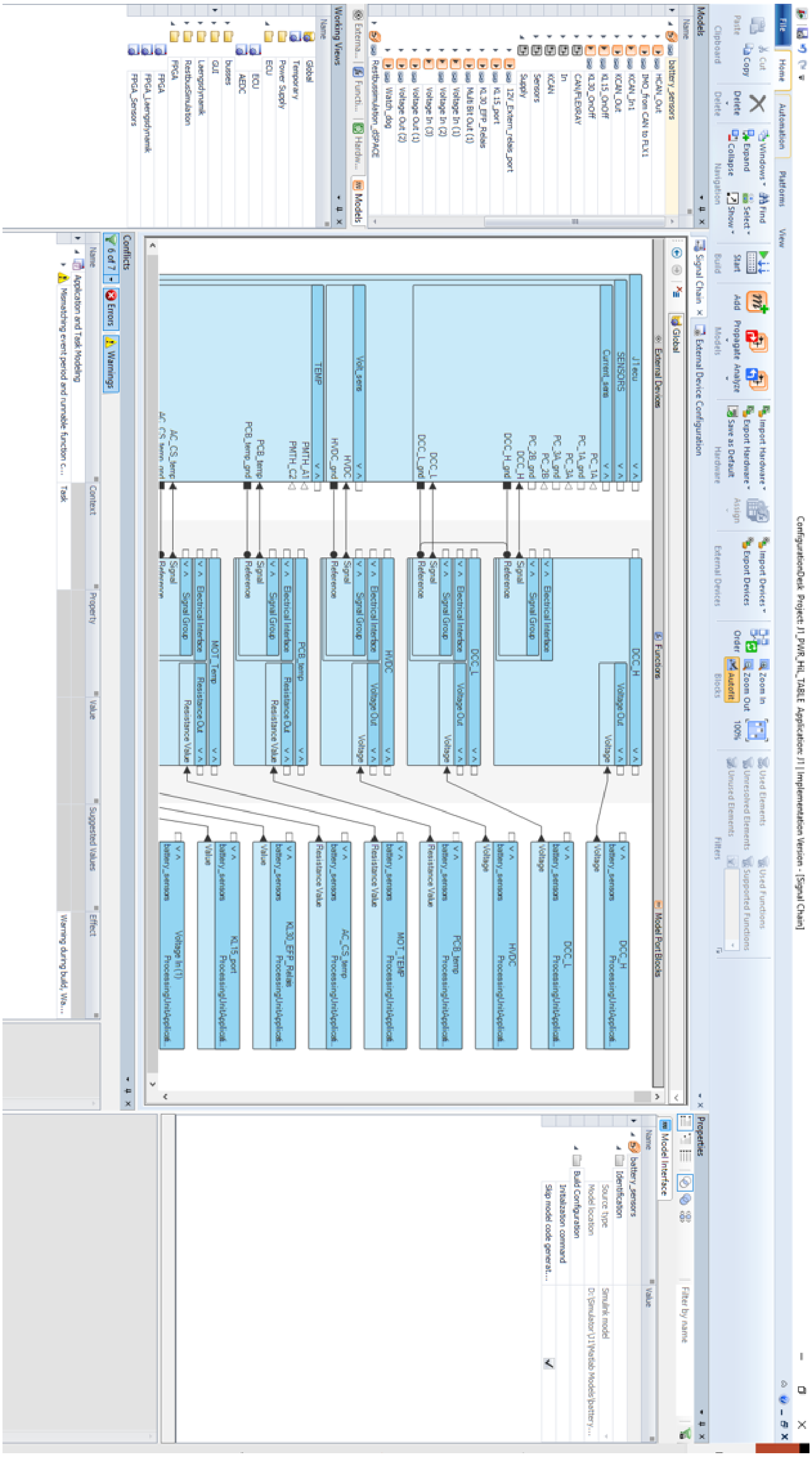


Figure 5.3: Example of created ConfigurationDesk settings.

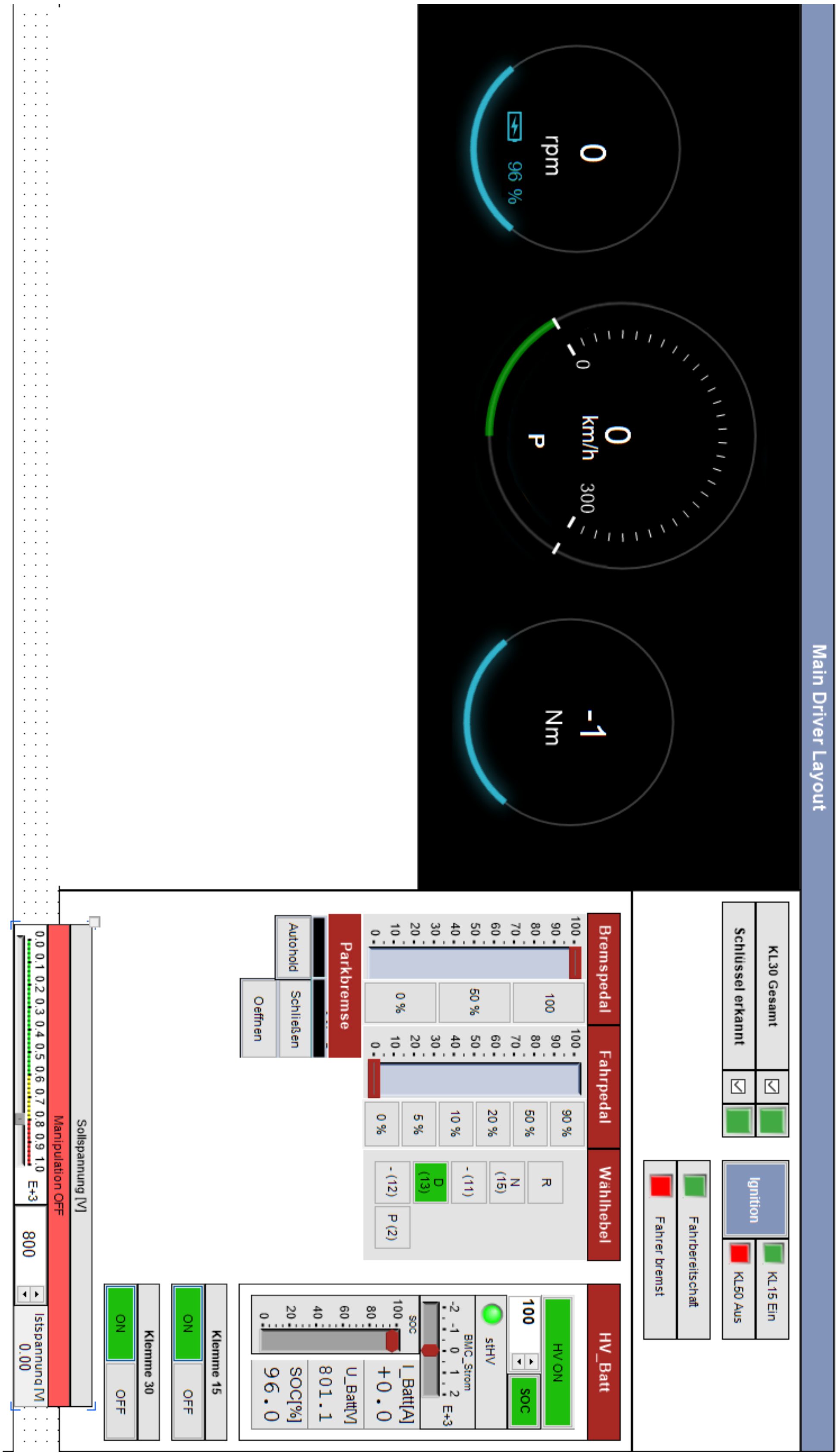
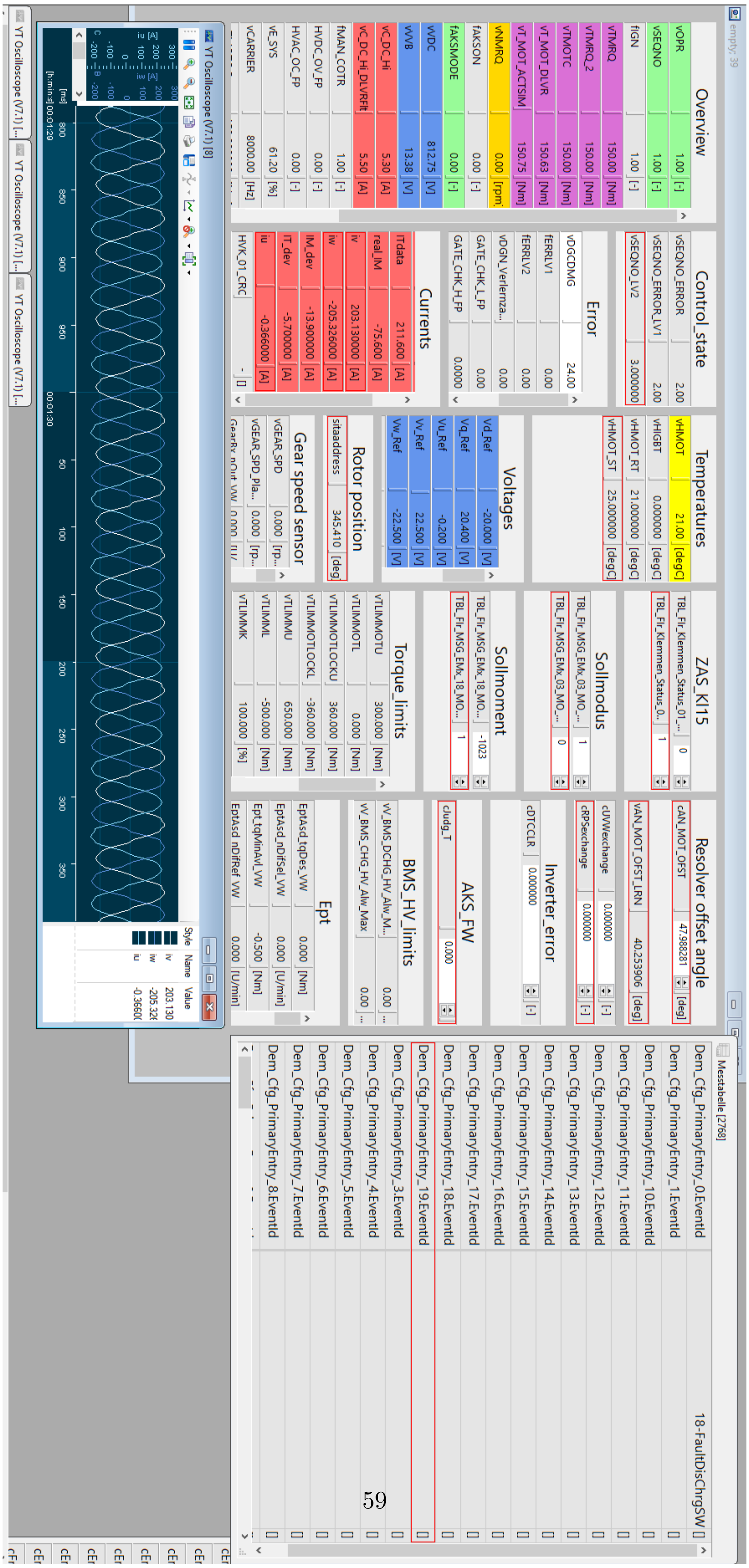


Figure 5.4: Example of ControlDesk layout.



Overview

vOPR	1.00	[-]
vSEQNO	1.00	[-]
fIGN	1.00	[-]
vTMRQ	150.00	[Nm]
vTMRQ_2	150.00	[Nm]
vTMOTC	150.00	[Nm]
vT_MOT_DLVR	150.63	[Nm]
vT_MOT_ACTSIM	150.75	[Nm]
vNMRQ	0.00	[rpm]
fAKSON	0.00	[-]
fAKSMODE	0.00	[-]
vVDC	812.75	[V]
vVVB	13.38	[V]
vC_DC_Hi	5.30	[A]
vC_DC_Hi_DLVRFilt	5.50	[A]
fMAN_COTR	1.00	[-]
HVDC_OV_FP	0.00	[-]
HVAC_OC_FP	0.00	[-]
vE_SYS	61.20	[%]
vCARRIER	8000.00	[Hz]

Control_state

vSEQNO_ERROR	2.00
vSEQNO_ERROR_LV1	2.00
vSEQNO_LV2	3.000000

Error

vDGCDMG	24.00
fERRLV1	0.00
fERRLV2	0.00
vDGN_Verlernza...	0.00
GATE_CHK_L_FP	0.00
GATE_CHK_H_FP	0.0000

Currents

iTdata	211.600	[A]
real_IM	-75.600	[A]
iv	203.130000	[A]
iw	-205.326000	[A]
IM_dev	-13.900000	[A]
IT_dev	-5.700000	[A]
iu	-0.366000	[A]
HVK_01_CRC	-	[]

Temperatures

vHMOT	21.00	[degC]
vHIGBT	0.000000	[degC]
vHMOT_RT	21.000000	[degC]
vHMOT_ST	25.000000	[degC]

Voltages

Vd_Ref	-20.000	[V]
Vq_Ref	20.400	[V]
Vu_Ref	-0.200	[V]
Vv_Ref	22.500	[V]
Vw_Ref	-22.500	[V]

Rotor position

sitaaddress	345.410	[deg]

Gear speed sensor

vGEAR_SPD	0.000	[rp...
vGEAR_SPD_Pla...	0.000	[rp...
GearRx_nOut_VW	0.000	[U/...

ZAS_Kl15

TBL_Flt_Klemmen_Status_01_...	0
TBL_Flt_Klemmen_Status_0...	1

Sollmodus

TBL_Flt_MSG_EMx_03_MO...	1
TBL_Flt_MSG_EMx_03_MO...	0

Sollmoment

TBL_Flt_MSG_EMx_18_MO...	-1023
TBL_Flt_MSG_EMx_18_MO...	1

Torque_limits

vTLIMMOTU	300.000	[Nm]
vTLIMMOTL	0.000	[Nm]
vTLIMMOTLOCKU	360.000	[Nm]
vTLIMMOTLOCKL	-360.000	[Nm]
vTLIMMU	650.000	[Nm]
vTLIMML	-500.000	[Nm]
vTLIMMK	100.000	[%]

Resolver offset angle

cAN_MOT_OFST	47.988281	[deg]
vAN_MOT_OFST_LRN	40.253906	[deg]

Inverter_error

cUVWexchange	0.000000	[-]
cRPSexchange	0.000000	[-]
cDTCCLR	0.000000	[-]

AKS_FW

cIudg_T	0.000

BMS_HV_limits

vV_BMS_DCHG_HV_Alw_M...	0.00
vV_BMS_CHG_HV_Alw_Max	0.00

Ept

EptAsd_tqDes_VW	0.000	[Nm]
EptAsd_nDifSel_VW	0.000	[U/min]
Ept_tqMinAvl_VW	-0.500	[Nm]
EptAsd_nDifRef_VW	0.000	[U/min]

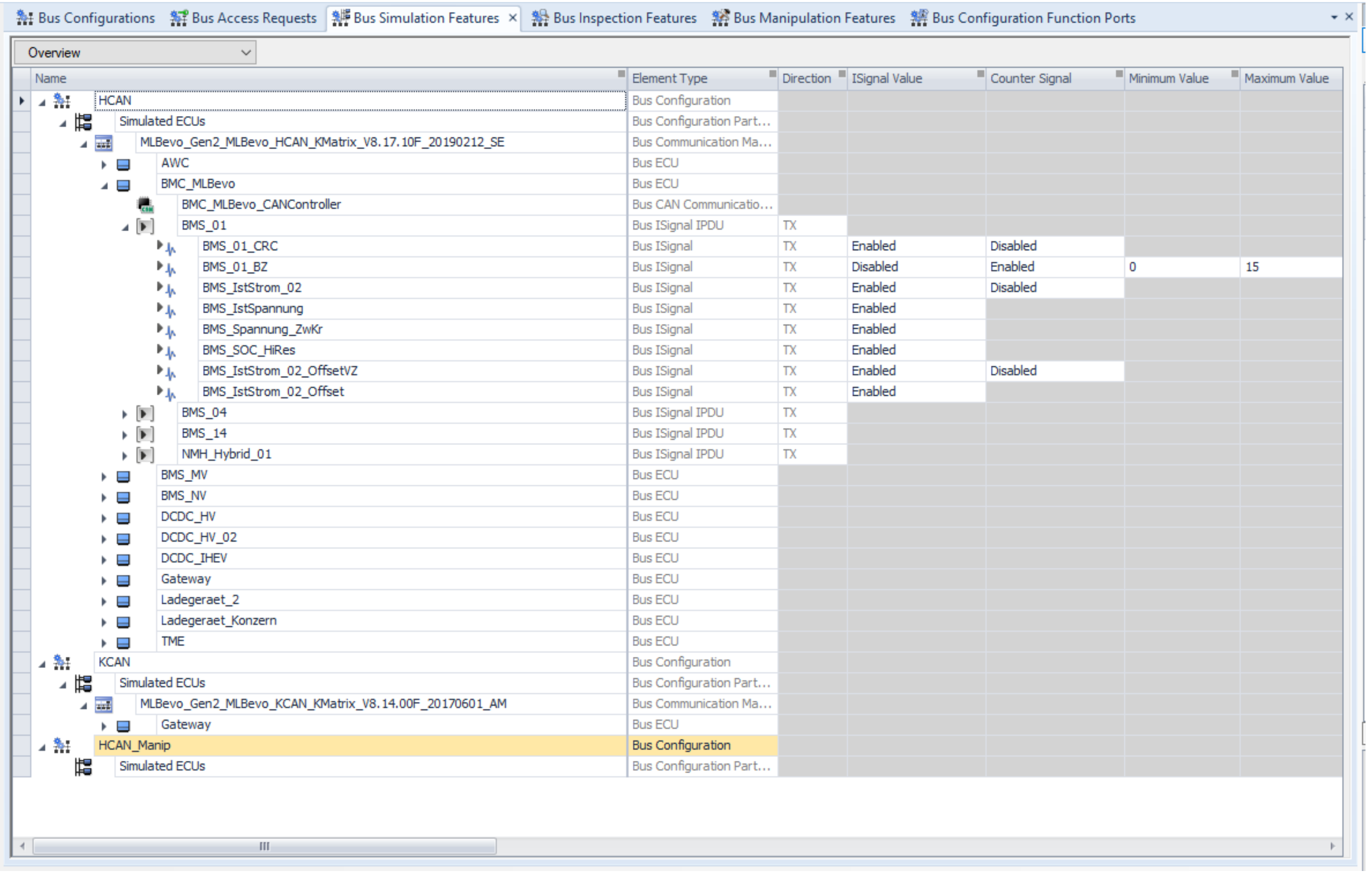
Messtabelle [2768]

Dem_Cfg_PrimaryEntry_0.EventId	18-FaultDisChrgSW
Dem_Cfg_PrimaryEntry_1.EventId	
Dem_Cfg_PrimaryEntry_10.EventId	
Dem_Cfg_PrimaryEntry_11.EventId	
Dem_Cfg_PrimaryEntry_12.EventId	
Dem_Cfg_PrimaryEntry_13.EventId	
Dem_Cfg_PrimaryEntry_14.EventId	
Dem_Cfg_PrimaryEntry_15.EventId	
Dem_Cfg_PrimaryEntry_16.EventId	
Dem_Cfg_PrimaryEntry_17.EventId	
Dem_Cfg_PrimaryEntry_18.EventId	
Dem_Cfg_PrimaryEntry_19.EventId	
Dem_Cfg_PrimaryEntry_3.EventId	
Dem_Cfg_PrimaryEntry_4.EventId	
Dem_Cfg_PrimaryEntry_5.EventId	
Dem_Cfg_PrimaryEntry_6.EventId	
Dem_Cfg_PrimaryEntry_7.EventId	
Dem_Cfg_PrimaryEntry_8.EventId	

YT Oscilloscope (V7.1) [8]

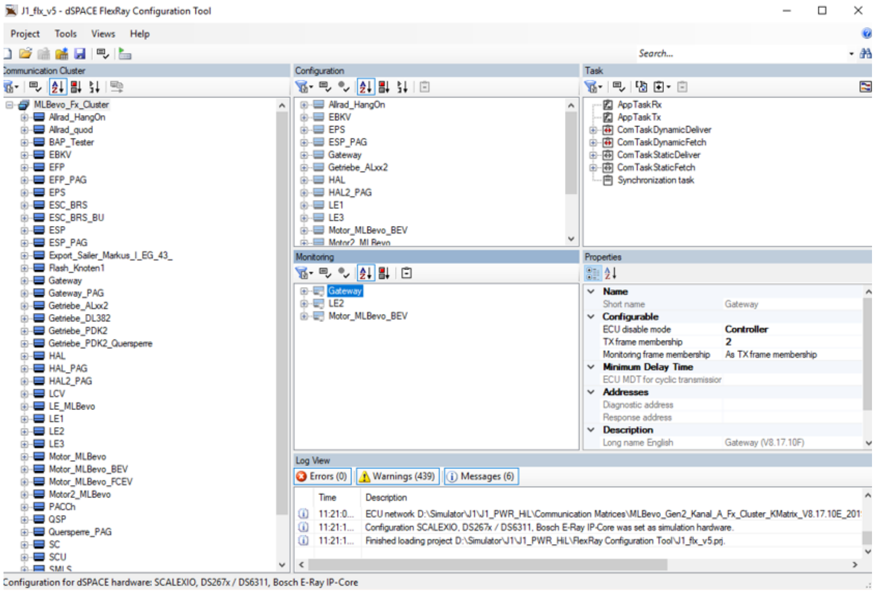
Style	Name	Value
	iv	203.130
	iw	-205.32
	iu	-0.3660

Figure 5.5: Example of INCA interface for the ECU.


Bus Configurations
Bus Access Requests
Bus Simulation Features ×
Bus Inspection Features
Bus Manipulation Features
Bus Configuration Function Ports
Overview

Name	Element Type	Direction	ISignal Value	Counter Signal	Minimum Value	Maximum Value
HCAN	Bus Configuration					
Simulated ECUs	Bus Configuration Part...					
MLBevo_Gen2_MLBevo_HCAN_KMatrix_V8.17.10F_20190212_SE	Bus Communication Ma...					
AWC	Bus ECU					
BMC_MLBevo	Bus ECU					
BMC_MLBevo_CANController	Bus CAN Communicatio...					
BMS_01	Bus ISignal IPDU	TX				
BMS_01_CRC	Bus ISignal	TX	Enabled	Disabled		
BMS_01_BZ	Bus ISignal	TX	Disabled	Enabled	0	15
BMS_IstStrom_02	Bus ISignal	TX	Enabled	Disabled		
BMS_IstSpannung	Bus ISignal	TX	Enabled			
BMS_Spannung_ZwKr	Bus ISignal	TX	Enabled			
BMS_SOC_HiRes	Bus ISignal	TX	Enabled			
BMS_IstStrom_02_OffsetVZ	Bus ISignal	TX	Enabled	Disabled		
BMS_IstStrom_02_Offset	Bus ISignal	TX	Enabled			
BMS_04	Bus ISignal IPDU	TX				
BMS_14	Bus ISignal IPDU	TX				
NMH_Hybrid_01	Bus ISignal IPDU	TX				
BMS_MV	Bus ECU					
BMS_NV	Bus ECU					
DCDC_HV	Bus ECU					
DCDC_HV_02	Bus ECU					
DCDC_IHEV	Bus ECU					
Gateway	Bus ECU					
Ladegeraet_2	Bus ECU					
Ladegeraet_Konzern	Bus ECU					
TME	Bus ECU					
KCAN	Bus Configuration					
Simulated ECUs	Bus Configuration Part...					
MLBevo_Gen2_MLBevo_KCAN_KMatrix_V8.14.00F_20170601_AM	Bus Communication Ma...					
Gateway	Bus ECU					
HCAN_Manip	Bus Configuration					
Simulated ECUs	Bus Configuration Part...					

Figure 5.6: Example of CAN bus implementation in ConfigurationDesk.



```
MLBevo_Fx_Cluster
  Allrad_HangOn
  Allrad_quod
  BAP_Tester
  EBKV
  EFP
  EFP_PAG
  EPS
  ESC_BRS
  ESC_BRS_BU
  ESP
  ESP_PAG
  Export_Sailer_Markus_I_EG_43_
  Flash_Knoten1
  Gateway
  Gateway_PAG
  Getriebe_ALxx2
  Getriebe_DL382
  Getriebe_PDK2
  Getriebe_PDK2_Quersperre
  HAL
  HAL_PAG
  HAL2_PAG
  LCV
  LE_MLBevo
  LE1
  LE2
  LE3
  Motor_MLBevo
  Motor_MLBevo_BEV
  Motor_MLBevo_FCEV
  Motor2_MLBevo
  PACCh
  QSP
  Quersperre_PAG
  SC
  SCU
  SMLS
```

```
Allrad_HangOn
EBKV
EPS
ESP_PAG
Gateway
Getriebe_ALxx2
HAL
HAL2_PAG
LE1
LE3
Motor_MLBevo_BEV
Motor2_MLBevo
```

```
AppTaskRx
AppTaskTx
ComTaskDynamicDeliver
ComTaskDynamicFetch
ComTaskStaticDeliver
ComTaskStaticFetch
Synchronization task
```

```
Gateway
LE2
Motor_MLBevo_BEV
```

```
Name
  Short name                      Gateway
Configurable
  ECU disable mode                Controller
  TX frame membership             2
  Monitoring frame membership     As TX frame membership
Minimum Delay Time
  ECU MDT for cyclic transmissior
Addresses
  Diagnostic address
  Response address
Description
  Long name English               Gateway (V8.17.10F)
```

```
   Time       Description
   11:21:0... ECU network D:\Simulator\J1\J1_PWR_HiL\Communication Matrices\MLBevo_Gen2_Kanal_A_Fx_Cluster_KMatrix_V8.17.10E_201
   11:21:1... Configuration SCALEXIO, DS267x / DS6311, Bosch E-Ray IP-Core was set as simulation hardware.
   11:21:1... Finished loading project D:\Simulator\J1\J1_PWR_HiL\FlexRay Configuration Tool\J1_flx_v5.prj.
```

Configuration for dSPACE hardware: SCALEXIO, DS267x / DS6311, Bosch E-Ray IP-Core

Figure 5.7: Example of FlexRay bus implementation.

5.1.2 Mathematical models

On figure 5.8 can be seen fitted model for the load torque model described in section 3.14. However recalculated model was not tested on the HiL due to the lack of the second PMSM machine and second ECU. Further verification of the load torque model and battery model can be found in [12].

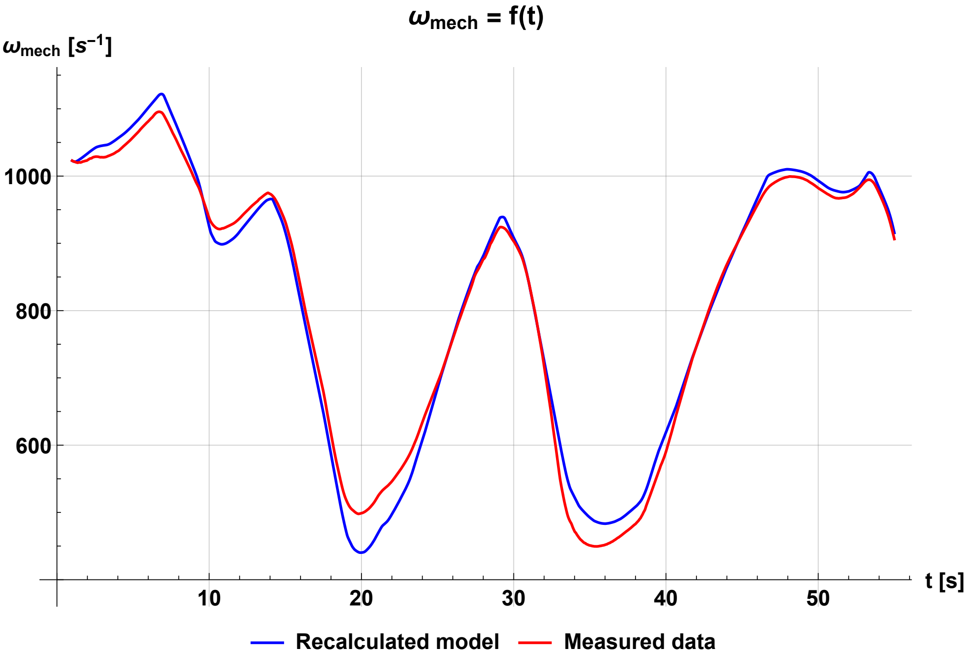


Figure 5.8: Final fit for the load torque model.

On figure 5.9 measured real-time test of intermediate circuit can be seen. As mentioned before, active discharging of the intermediate circuit is carried out by PWM. This can be seen from 800V to cca. 450 V. From 450 V the discharge is no longer carried out by PWM.

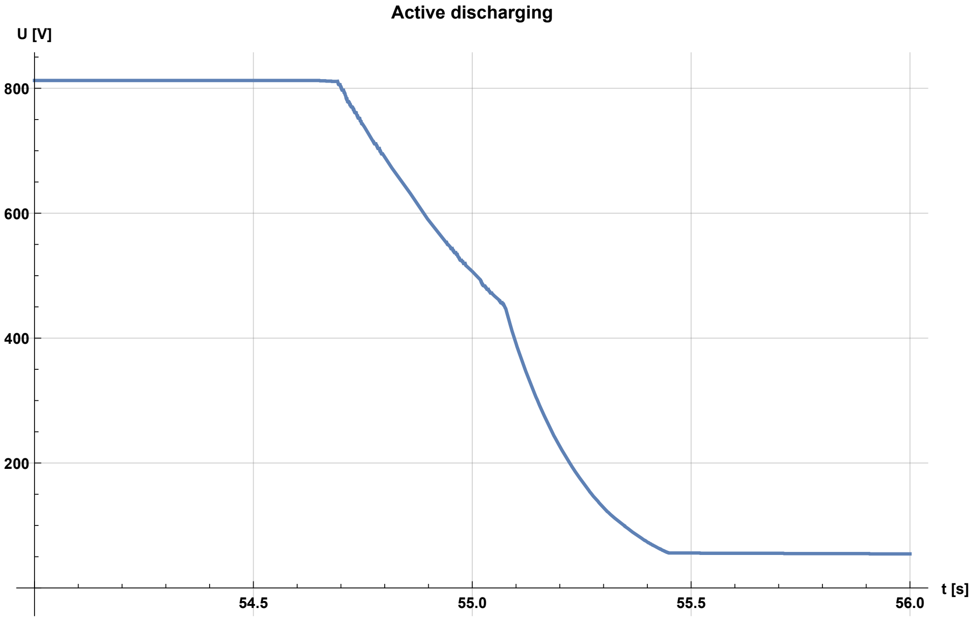


Figure 5.9: Active Discharge of intermediate circuit.

5.1.3 FPGA model

In this subsection, results related to FPGA models are presented.

on figure 5.10, final structure of the model is shown. In this structure, 3D tables are already implemented in the model on the FPGA card.

On figure 5.12, measured phase currents during the acceleration of the machine are shown. There is a clearly visible linkage with the speed and the frequency of the currents.

On figure 5.13 requested and measured current in d/q system is shown. Torque request for this measurement was 280 Nm.

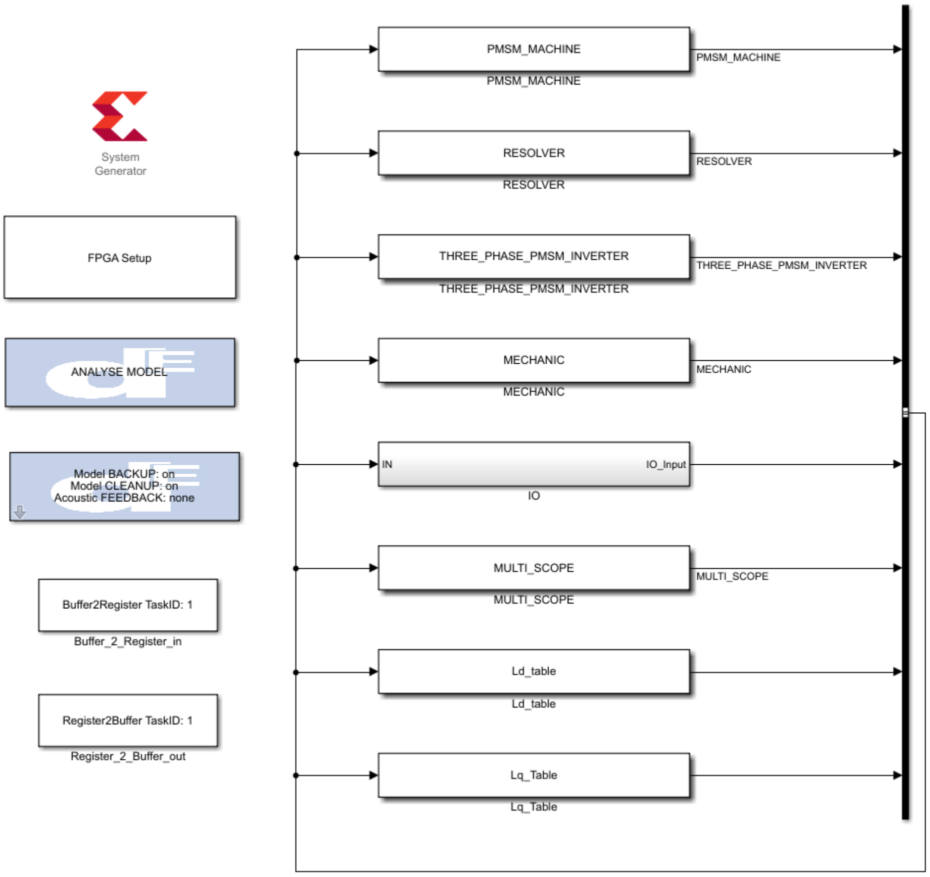


Figure 5.10: Model structure after implementation of 3D tables.

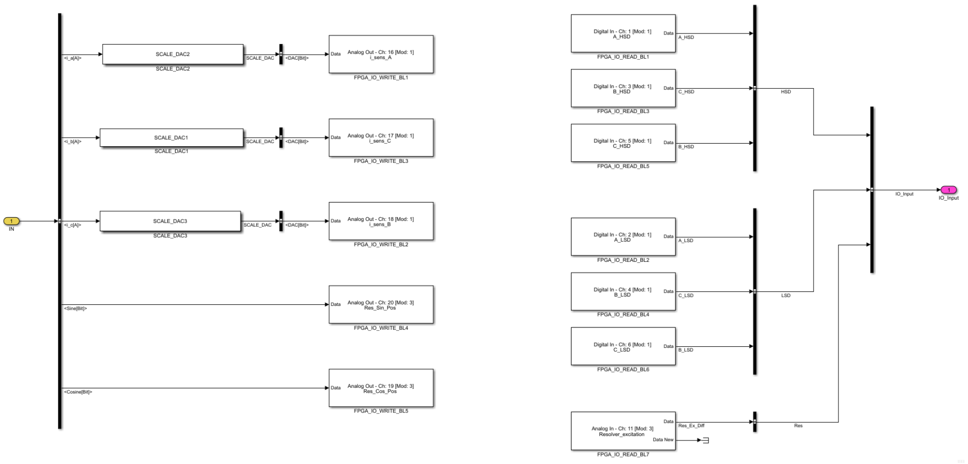


Figure 5.11: I/O submodel implementation.

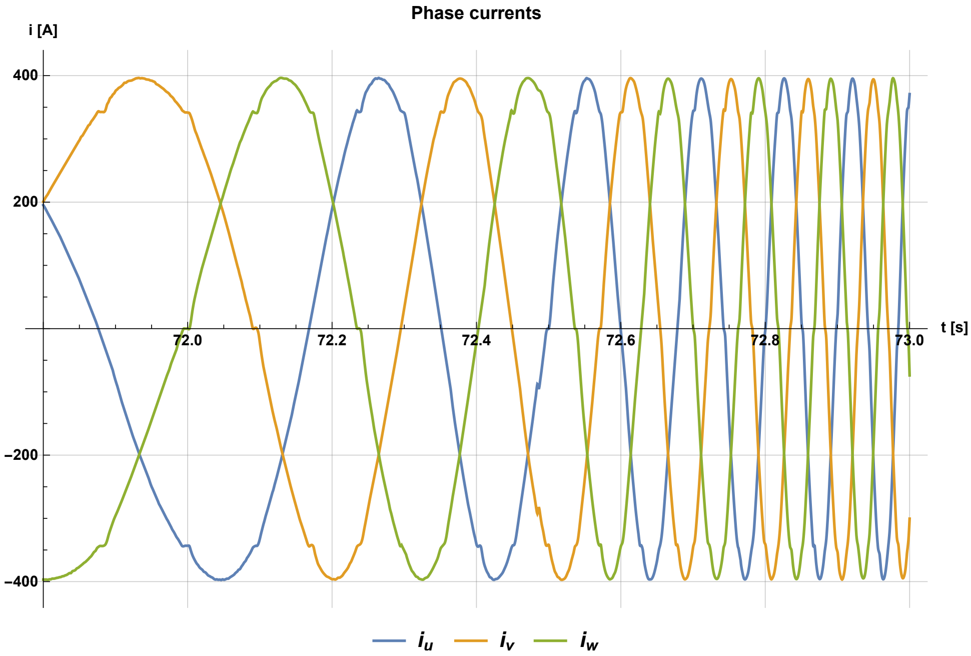


Figure 5.12: Phase currents during acceleration.

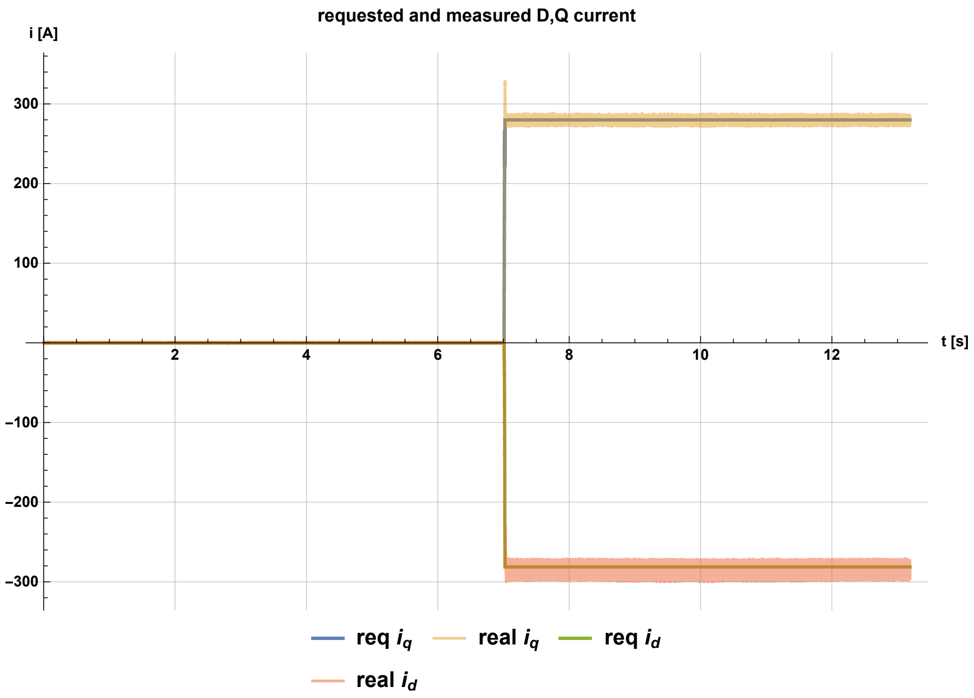


Figure 5.13: i_d and i_q with torque request 280 Nm.

FPGA model improvements

To verify those improvements, simulated values are compared to the reality. Machine was turning with speed 500 rpm and torque request with $280Nm$. Those values were also used in the reality on the testbench.

On figure 5.14, torque measured on the real test-bench, torque measured with original model with tables $L = f(i_d, iq)$ and improved model with $L = f(i_d, i_q, \theta)$ is shown. This figure shows that the harmonic structure of the torque is approaching to the harmonic structure of the measured torque. However mostly 18th harmonic is still missing and this can be seen on figure 5.15. This will be the subject of further study, however reason for this might be the distribution between L_d and L_q.

Furthermore, measured and simulated phase currents are compared on figure 5.16. Currents perfectly corresponds in frequency and amplitude. The simulated current is perfectly sinusoidal probably due to the sampling rate of the measurement.

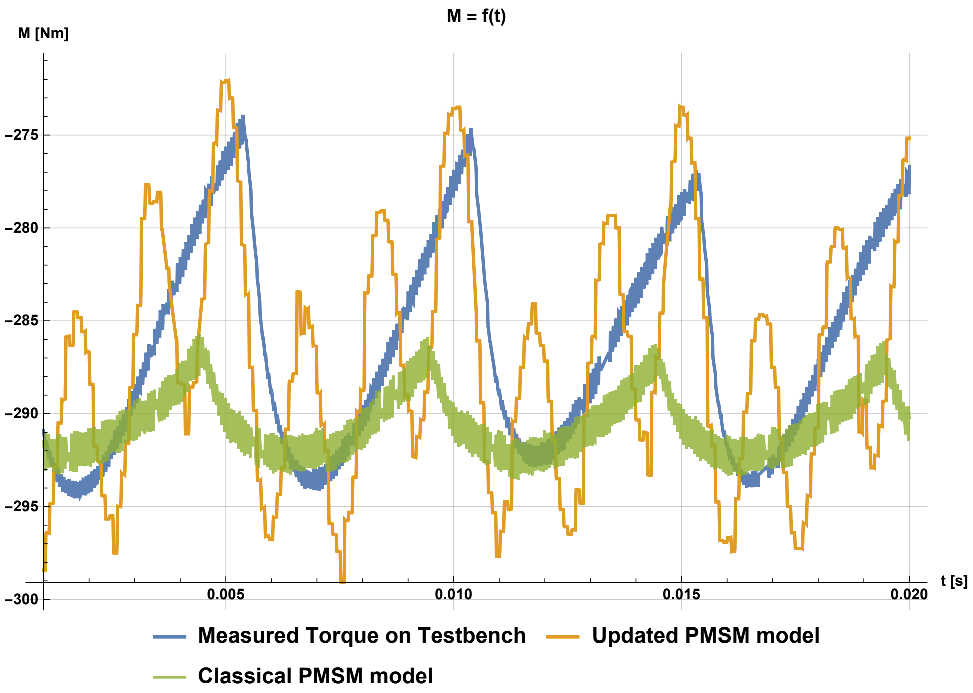


Figure 5.14: Torque comparison.

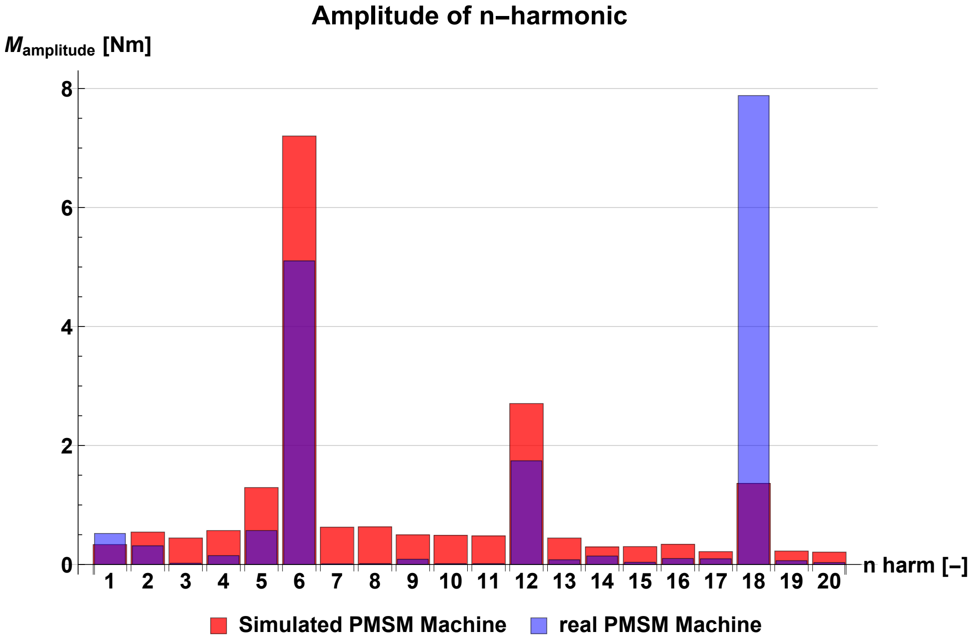


Figure 5.15: Amplitude comparison.

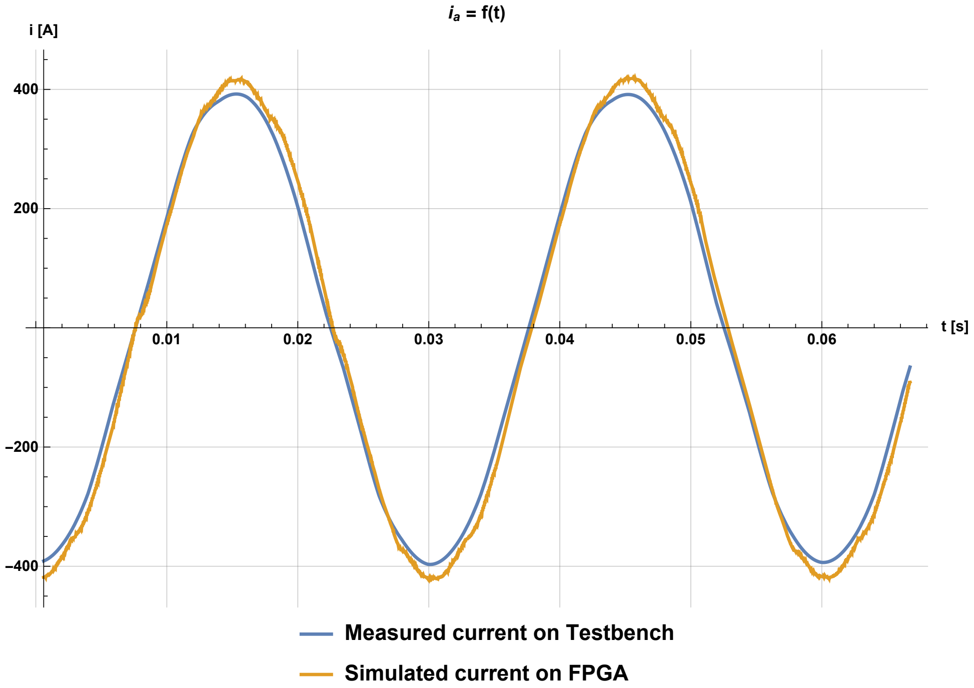
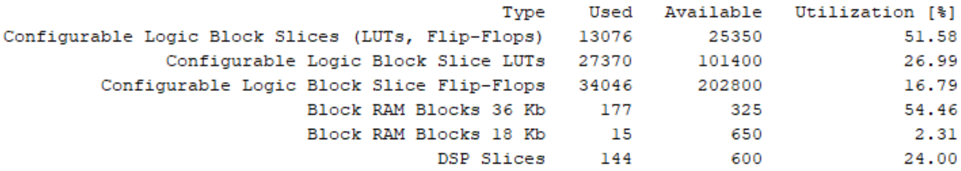


Figure 5.16: Amplitude comparison.

On figure 5.17 is shown built report of the FPGA model.



```
                                            Type   Used   Available   Utilization [%]
Configurable Logic Block Slices (LUTs, Flip-Flops)  13076       25350             51.58
                 Configurable Logic Block Slice LUTs  27370      101400             26.99
           Configurable Logic Block Slice Flip-Flops  34046      202800             16.79
                              Block RAM Blocks 36 Kb    177         325             54.46
                              Block RAM Blocks 18 Kb     15         650              2.31
                                          DSP Slices    144         600             24.00
```

Figure 5.17: FPGA build report.

Chapter 6

Conclusion

6.1 Overview of the results

Chapter 2 provides overall look at HiL systems with accent on HiL systems for Power Inverter ECU. Different solutions currently available on the market are presented.

Chapter 3 mainly focuses on the development of the HiL system and has two main parts

- The first section consists of platform description, cable harness description, FlexRay and CAN implementation.
- The second section deals with all mathematical model needed for simulation.

Chapter 4 mainly focuses on FPGA model

- The first section consists of FPGA model analysis without any improvements. Default FPGA model configuration is described altogether with submodels and their mathematical descriptions.
- The second section deals with possible improvements of PMSM simulation.

6.2 Fulfilment of the objectives defined in goals

I consider all objectives of this thesis to be satisfied.

6.3 Suggestions for the future work

The idea of the HiL system is to provide environment for testing, development and debugging of the ECU with new software versions. It means that the HiL system has to be continuously upgraded and evolved in accordance to test specifications.

Almost every model can be upgraded on HiL system to better describe reality. Load torque model with battery model can be upgraded to fit the reality better. Restbus simulation is model, on which various groups are working.

Major task for the future is to test the model with maps from FEM tools. Further more compare the results with dSPACE table based model.

Another task is to develop a new model of gearbox, that would be running in connection with mechanic model in FPGA card. This topic was requested by additional engineers.

6.4 Data confidentiality

All models mentioned in the master thesis were directly developed for Porsche Engineering Services and therefore, all models and measured data aren't enclosed due to the data confidentiality.

Appendix A

Bibliography

[1] dSPACE, “Scalexio.” "https://www.dspace.com/shared/data/pdf/2020/dSPACE_SCALEXIO_Product-information_02_2020_English.pdf. "[Online; 6.7.2020]".

[2] N. Instruments, “Flexray automotive communication bus overview.” "https://www.ni.com/cs-cz/innovations/white-papers/06/flexray-automotive-communication-bus-overview.html, 2019. "[Online; 24.7.2020".

[3] Vector, “Vx1000 measurement and calibration hardware.” "https://www.vector.com/int/en/products/products-a-z/hardware/vx1000/#c138619, 2020. "[Online; 2.8.2020]".

[4] ETAS, “Es89x – fetk ecu and bus interface modules for highest demands.” "https://www.etas.com/en/products/es891_es892_fetk_ecu_bus_interface_modules.php, 2020. "[Online; 2.8.2020]".

[5] dSPACE, “High-voltage electronic load.” "https://www.dspace.com/shared/data/pdf/2019/dSPACE_High-Voltage_Electronic_Load_Product_information_01_2019_English.pdf, 2020. "[Online; 24.7.2020".

[6] P. R. Wilson and S. (online služba), *Design recipes for FPGAs: using Verilog and VHDL*. San Francisco;New York;San Diego;Singapore;Amsterdam;Tokyo;London;Heidelberg;Oxford;Paris;Sydney;Boston;: Newnes is an imprint of Elsevier, second ed., 2016.

[7] Xilinx, “Understanding fpga architecture.” "https://www.xilinx.com/html_docs/xilinx2017_4/sdaccel_doc/odz1504034293215.html, 2018. "[Online; 24.7.2020"].

[8] dSPACE, “dspace flexray configuration package.” "https://www.dspace.com/shared/data/pdf/2020/dSPACE-FlexRay-Configuration-Package_Product-Information_2020-05_EN.pdf, 2017. "[Online; 24.7.2020".

[9] I. Advanced Micro Controls, “What is a resolver?.” "https://www.amci.com/industrial-automation-resources/plc-automation-tutorials/what-resolver/. "[Online; 6.7.2020]".

[10] OPAL-RT, “Op4510 simulator.” "https://www.opal-rt.com/simulator-platform-op4510/. "[Online; 6.7.2020]".

[11] J. Wirsig, “Automatisierte absicherung einer antriebslestungselektronik fur hochleistungsfahrzeuge und deren steuerung mit hilfe eines hardware-in-the-loop-systems,” 2017.

[12] V. Štěrba, “Modelling of electric drive and vehicle dynamics for hardware-in-the-loop testing system,” 2018.

[13] Xilinx, “What is an fpga?.” "WhatisanFPGA, 2018. "[Online; 24.7.2020"].

[14] R. B. GmBH., “Can specification.” "http://esd.cs.ucr.edu/webres/can20.pdf, 1991. "[Online; 24.7.2020".

[15] dSPACE, “Rti fpga programming blockset.” "https://www.dspace.com/en/pub/home/products/sw/impsw/rti_fpga_programming_blockset.cfm#143_25259, 2020. "[Online; 24.7.2020".

[16] dSPACE, “Xsg electric component models.” "https://www.dspace.com/en/pub/home/products/sw/automotive_simulation_models/produkte_asm/electric_system_models/electric_componen.cfm, 2020. "[Online; 24.7.2020".

[17] ETAS, “Etas - es5340 – pci express simulation boards.” https://www.etas.com/en/products/es5340.php, 2018. "[Online; 31.3.2018]".

[18] OPAL-RT, “Fpga electric machine library.” "https://www.opal-rt.com/software-software-overview/fpga-electric-machine-library/. "[Online; 6.7.2020]".

[19] Xilinx, “Vivado design suite user guide: Model-based dsp design using system generator.” "https://www.xilinx.com/support/documentation/sw_manuals/xilinx2020_1/ug897-vivado-sysgen-user.pdf, 2020. "[Online; 24.7.2020".

[20] Autosar, “E2e protocol specification.” "https://www.autosar.org/fileadmin/user_upload/standards/foundation/1-3/AUTOSAR_PRS_E2EProtocol.pdf, 2017. "[Online; 24.7.2020".

[21] W. M. J.-P. R. Martin Stigge, Henryk Plötz, “Reversing crc Theory and Practice, journal = HU Berlin Public Report , year = 2006, optkey = SAR-PR-2006-05„”

[22] T. Haubert, *Optimal Control Strategy of Hybrid Electric Vehicle*. PhD thesis, CTU, FEE, 2018.

[23] F. Vlk, *Dynamika motorových vozidel: jízdní odpory : hnací charakteristika : brzdění : odpružení : řiditelnost : ovladatelnost : stabilita.* Brno: Nakladatelství a vydavatelství Vlk, 1. vyd. ed., 2000.

[24] P. Porteš, "Podélná dynamika vozidel," Master's thesis, Vysoké učení v Brně, Fakulta strojního inženýrství, 2015.

[25] P. Kobrle, J. Pavelka, and České vysoké učení technické v Praze. Elektrotechnická fakulta, *Elektrické pohony a jejich řízení.* V Praze: České vysoké učení technické, 3. přepracované vydání ed., 2016.

[26] NIST, "Engineering statistics handbook, linear least squares regression." "`https://www.itl.nist.gov/div898/handbook/pmd/section1/pmd141.htm`, 2012. "[Online; 24.7.2020".

[27] K. Sarrafan, D. Sutanto, and K. M. Muttaqi, "An electric circuit based ev battery model for runtime prediction and state of charge tracking," in *2017 IEEE Transportation Electrification Conference (ITEC-India)*, pp. 1–6, Dec 2017.

[28] J. Kacetl, "Mathematical model and validation of powertrain," Master's thesis, Czech Technical University in Prague, 2016.

[29] J. Pavelka, Z. Čeřovský, J. Lettl, and České vysoké učení technické v Praze. Elektrotechnická fakulta, *Výkonová elektronika.* Praha: Nakladatelství ČVUT, vyd. 3. přeprac. ed., 2007.

[30] N. P. Quang and J.-A. Dittrich, *Vector control of three-phase AC machines: system development in the practice.* 2008.

[31] B. Bose, *Modern Power Electronics and AC Drives.* Eastern Economy Edition, Prentice Hall PTR, 2002.

[32] S. Zarate, G. Almandoz, G. Ugalde, J. Poza, and A. J. Escalada, "Extended dq model of a permanent magnet synchronous machine by including magnetic saturation and torque ripple effects," in *2017 IEEE International Workshop of Electronics, Control, Measurement, Signals and their Application to Mechatronics (ECMSM)*, pp. 1–6, 2017.

[33] C. Dutta and S. M. Tripathi, "Comparison between conventional and loss d-q model of pmsm," in *2016 International Conference on Emerging Trends in Electrical Electronics Sustainable Energy Systems (ICETEESES)*, pp. 256–260, 2016.

[34] C. Dufour, S. Cense, T. Yamada, R. Imamura, and J. Bélanger, "Fpga permanent magnet synchronous motor floating-point models with variable-dq and spatial harmonic finite-element analysis solvers," in *2012 15th International Power Electronics and Motion Control Conference (EPE/PEMC)*, pp. LS6b.2–1–LS6b.2–10, 2012.