Czech Technical University in Prague

Faculty of Electrical Engineering

MASTER THESIS

2019 **Lukáš Hostačný**

Czech Technical University in Prague

Faculty of Electrical Engineering

Department of Measurement

MASTER THESIS

Vehicle Measurement System

Study programme: Cybernetics and Robotics

Branch of study: Sensors and Instrumentation

Prague

January 2019

Author: **Lukáš Hostačný**

Supervisor: **Ing. Tomáš Haniš, Ph.D.**

CTU
CZECH TECHNICAL
UNIVERSITY
IN PRAGUE

MASTER‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Hostačný Lukáš** Personal ID number: **425061**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Measurement**

Study program: **Cybernetics and Robotics**

Branch of study: **Sensors and Instrumentation**

II. Master’s thesis details

Master’s thesis title in English:

Vehicle Measurement System

Master’s thesis title in Czech:

Měřicí systém pro pozemní vozidla

Guidelines:

The goal of the thesis is to investigate possibility of land vehicle measurement systems, providing necessary input for advance control architecture providing artificial stability for next generation vehicle platform. Such system shall be verified based on instrumentation of scaled down rapid prototyping platform.
1. Investigate measurement strategies (odometer based, e-engines torque measurement, relative and/or absolute inertial measurements) suited for control system design.
2. Develop suitable scale down platform for purposes of vehicle data acquisition and control strategy demonstration.
3. Develop and implement HW and basic SW for such platform in order to demonstrate measurement system capabilities.
4. Design and perform test drive experiments in order to demonstrate and verify system capabilities.

Bibliography / sources:

[1] Edward M. Kasprzak, L. Daniel Metz, William F. Milliken, Douglas L. Milliken: Race Car Vehicle Dynamics - Problems, Answers and Experiments, Premiere Series Books, 2015, ISBN-10: 0768011272.

Name and workplace of master’s thesis supervisor:

Ing. Tomáš Haniš, Ph.D., Department of Control Engineering, FEE

Name and workplace of second master’s thesis supervisor or consultant:

Date of master’s thesis assignment: **30.08.2018** Deadline for master's thesis submission: **08.01.2019**

Assignment valid until: **29.02.2020**

Ing. Tomáš Haniš, Ph.D.
Supervisor’s signature

Head of department’s signature

prof. Ing. Pavel Ripka, CSc.
Dean’s signature

III. Assignment receipt

The student acknowledges that the master’s thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master’s thesis, the author must state the names of consultants and include a list of references.

.
Date of assignment receipt

Student’s signature

Acknowledgement

I would like to thank my supervisor Ing. Tomáš Haniš, Ph.D. for his constructive advices and useful critics of the thesis. My grateful thanks are also extended to Marek Laszlo for my introduction to this project and for his guidance in the mechanical areas. I would like to express my special thanks to my family. The journey of my study would not have been possible without their continual support.

Declaration

"I hereby declare that I wrote this thesis (Vehicle Measurement System) based on my own theoretical and practical knowledge, consultations and study of professional literature, and that I cited all the information sources used in compliance with the *Methodical Instructions of the ethical principles for writing an academic thesis*."

Prague, _______________ __________________

signature of the author

Anotácia

Diplomová práca sa zaoberá návrhom a zhotovením zmenšenej platformy vhodnej pre testovanie a validáciu pokročilých riadiacich algoritmov vozidiel. Ciel'om práce je poskytnút' l'ahko dostupnú a užívatel'sky priaznivú platformu pre návrhárov riadiacich algoritmov. Návrhár by mal byt' schopný túto platformu programovat' a obsluhovat' sám.

Prvá čast' práce je zameraná na výber vhodného podvozku a jeho úpravy, ako aj rozbor minimálnych vhodných požiadavok pre meracie vybavenie platformy.

Druhá čast' práce sa venuje samotnému výberu elektronického a meracieho vybavenia vozidla a návrhu programového vzoru platformy.

Na záver sú zhodnotené namerané výsledky. Platforma sa otestovala vo vonkajšom prostredí na bežnom asfalte pri manévroch ako slalom, zrýchlenie a iných.

Klúčové slová: Zmenšená platforma pozemného vozidla, RC platform, Simulink embedded coder, STM32 MAT/TARGET, meranie momentu BLDC elektro motoru, meranie rýchlosti, GPS, IMU, CAN zbernica, RC BLDC motory.

Annotation

The Master thesis is focused on the development and the realization of a scaled-down vehicle platform suitable for advanced control algorithms verification. The goal of the thesis is to offer easily accessible and user-friendly vehicle platform for developers of control algorithms. The requirement is to provide such solution , that the developer will be able to operate and program the platform on its own.

The first part of the thesis describes the selection of chassis and its adjustments and the minimum requirements for the measuring equipment of the scaled-down platform.

The second part is dedicated to a selection of the electronics components and design of the software template for the platform.

The last part evaluates the measured data. The platform was tested in outside conditions at certain maneuvers as Slalom, Acceleration, etc..

Keywords: Scaled-down vehicle platform, RC platform, Simulink embedded coder, STM32 MAT/TARGET, BLDC electric motor torque measurement, velocity measurement, GPS, IMU, CAN bus, RC BLDC motors.

Contents

Acronyms

SDP	Scaled-Down Platform
VESC	Vedder Electric Speed Controller (the Benjamin's Vedder motor controller)
EMF	ElectroMotive Force
ESC	Electric Speed Controller
RTK	Real-Time Kinematic
GNSS	Global Navigation Satellite System
FIR	Finite Impulse Response
INS	Inertial Navigation System
IMU	Inertial Measurement Unit
CAN	Controller Area Network
UART	Universal Asynchronous Receiver-Transmitter
SPI	Serial Peripheral Interface
PWM	Pulse Width Modulation
GPS	Global Positioning System
RPM	Rotations Per Minute
ERPM	Electric Rotations Per Minute
FOC	Field Oriented Control
PPM	Pulse Position Modulation
BLDC	BrushLess Direct Current

1 Introduction

Advanced traction algorithms can significantly improve the safety and performance of any vehicle. A necessary precondition is that a platform should support some requirements. For example torque vectoring requires possibility to distribute traction vehicle torque to the wheels independently. Drive-by-wire needs proper servo mechanism, etc..

The Formula student team (called eForce [14]) has a perfect platform for testing such algorithms, but the formula car was not available for a necessary time period that we could develop traction algorithms. On top of that the formula platform requires at least four operators for going testing or riding. Despite those problems, the idea has been still supported.

Lack of easily accessible vehicle platform pushed us to seek more suitable solution, which would offer possibility to test advanced traction controls algorithms. Hence, the idea of Marek Laszlo (who has a great experience with RC car and RC racing) proposed the idea of a scaled platform to the Doc. Ing. Martin Hromčík, Ph.D. who has supported and led this project of Scaled-down vehicle platform for testing advanced control. Later Marek formed the team. I was a member of the team and have also participated in Formula student team.

The scaled platform should reduce number of needed operators to only one person. When the eForce platform formula would not be available, the scaled-down platform can be used. We expect that this idea would help us to develop ans test the traction control algorithms. This would reduce needed time of the eFroce formula platform. Also, we can easily introduce members and newbies in the team, who would be interested in problematics of traction control and, torque vectoring. On top of that, the platform represents the possibility to discover not enough explored field: "let's try to make a car unstable and try to stabilize it electronically. What would happen? Let's try something else". This scaled-down platform (*SDP*) represents a step closer to our goals. However, it is not the final solution.

1.1 Objective

Goals of the diploma thesis are to create an SDP, which allows to independently control at least two driven wheels, controlling of steering servo mechanism and to measure a velocity, g-forces, angular rates and torque of the driven wheels. To make a user-friendly program template that will provide a reliable control interaction with car's hardware components. This platform would have research, educational and promotional purposes.

2 Development of suitable scaled platform

The SDP brings many advantages as I mentioned, but also some disadvantages. A lot of available sensors are not developed for the SDP. Hence we will not be able to measure some parameters with such high accuracy. Despite the cons, the scaled-down platform is a necessary step to get closer to a bigger and more complicated platform.

In the time, when the project started, I could find only one similar project, see [12]. It is professional RC chassis 1:10 adjusted to support all-wheel-drive independently. Nowadays, several RC platforms which support advanced control algorithms are published. However, they usually are not designed for racing performance.

The RC model has been chosen with scale of 1:10, because this category is well supported by manufacturers and it is the popular category by professional drivers too. Finally, the type of RC buggy car has been selected. A Buggy car is designed to pass more kind of terrains in compare to its road car opponent. It increases the potential of usage. Also buggy does not need any particular track, therefore designed algorithms can be verified on a general surface. For example in free parking place.

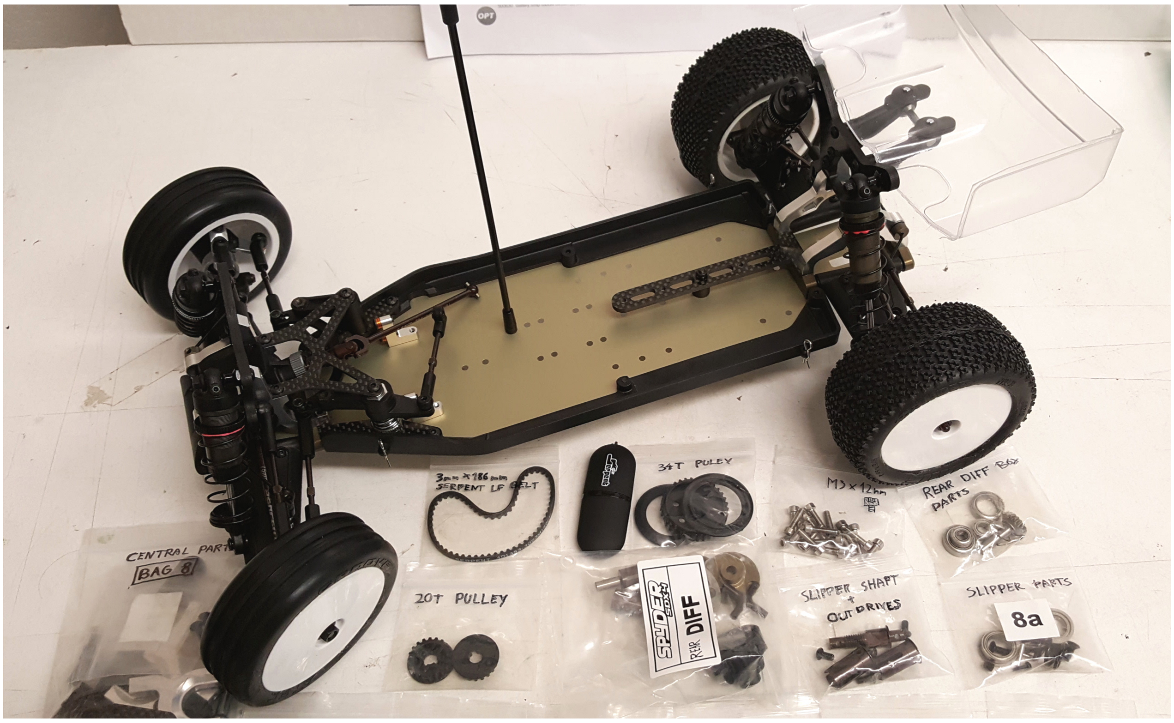

Figure 2.1: The chosen platform from SERPENT manufacturer.

Selection of a proper platform was time-consuming and difficult. The chassis needed to offer the necessary space and possibilities for adjustments. I selected platform from Serpent branch SDX4 Spyder, see figure 2.1.

This platform is all-wheel drive. In comparison to model which is only rear wheel drive, SDX4 (see [21]) has chassis which provides more space. Also, its front differential can be used for speed measuring. Hence the buggy has front wheels from 2WD model SDX2. See picture 2.1.

2.1 Separated control of driven wheels

A lot of RC cars in the category of 1:10 offer all-wheel drive. Even more, some of them use two motors. It seems to be a perfect solution for the SDP, but it is not. Almost all all-wheel-drive RC platforms use an open type of differentials [5]. This type of differential is not usable for SDP as it does not allow to control and distribute a torque on each wheel separately. Models with two motors have the motors connected by one camshaft which leads to the mentioned differentials. Therefore the platform with special rear side assembly was chosen. See figure 2.2.

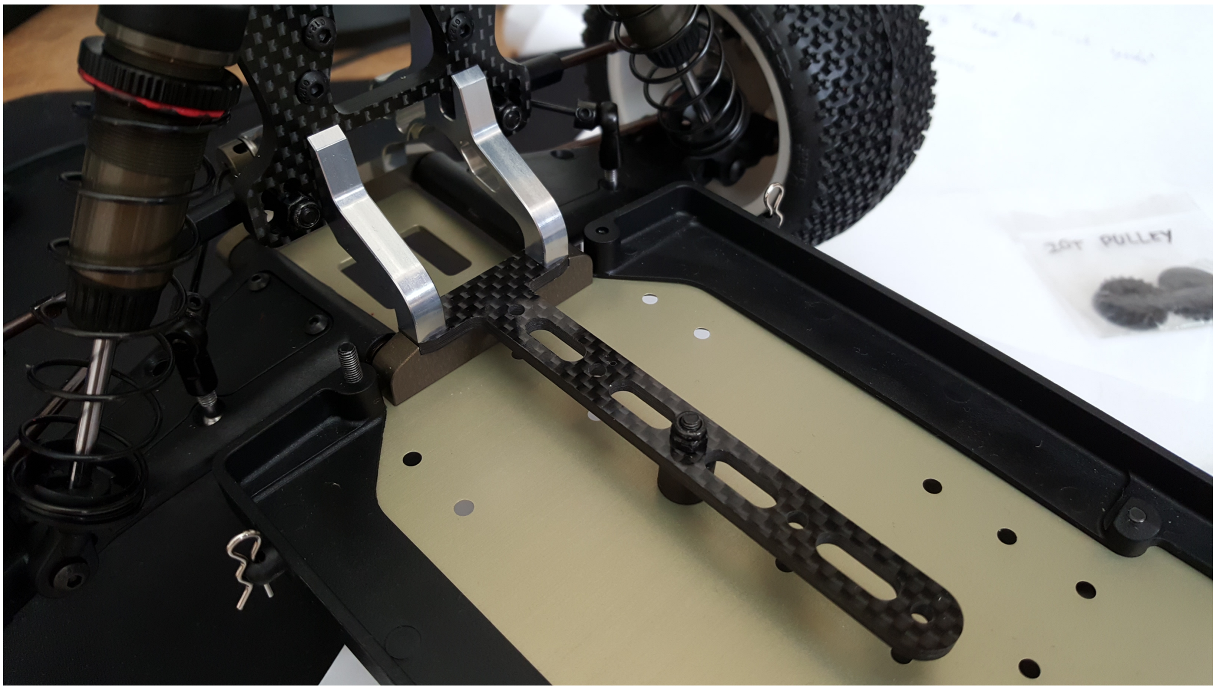

Figure 2.2: Serpent SDX4 rear side. Original construction.

Two metallic pylons hold differential in between. This solution of the rear part of chassis has a potential for custom adjustment. Removing the differential has saved a lot of space. Originally designed pylons provide a basement for attachment of separated custom pulleys. The custom solution is shown in the figure 2.3. New metallic pylons have been made more extended, in order to hold the additional engines. This technical solution was designed in cooperation with Marek Laszlo. The mechanic columns adjustment has been made by external mechanics. The critical point was to select existing chassis providing enough potential for custom solution.

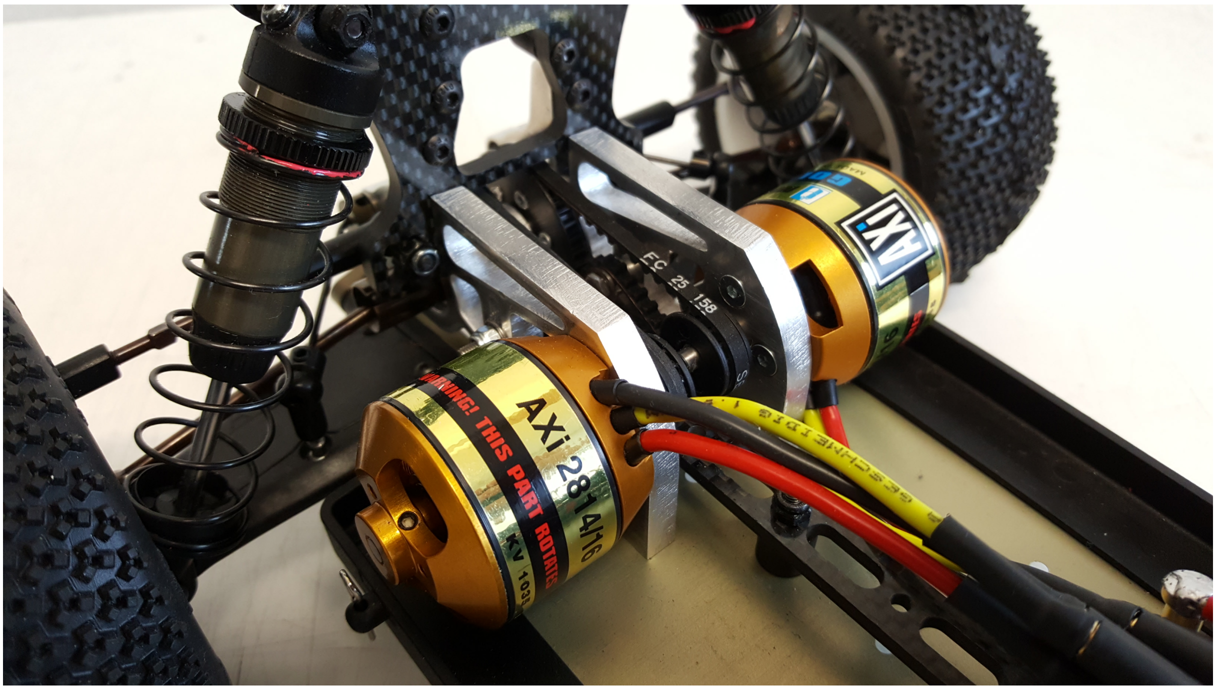

Figure 2.3: Serpent SDX4 rear side. After customization.

The result is simple and reliable. Small pulleys are placed directly on the motor shaft. Smaller gears are linked by a belt with bigger gears. The bigger gears are connected to the wheel by semiaxes. Each rear wheel is driven by its motor independently.

2.2 Power train

The entire power-train consists of batteries, electric motors, motor controllers, belt assembly pulleys and wheels. The picture 2.4 is a block diagram of the power-train.

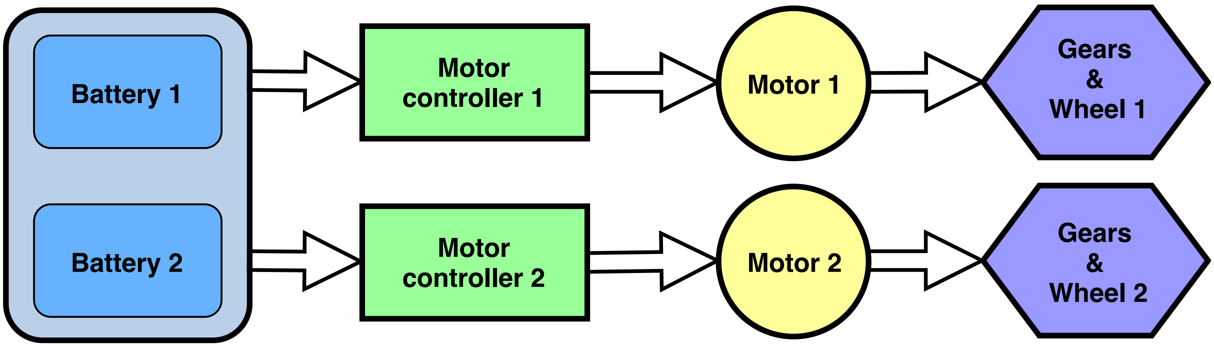

Figure 2.4: Block diagram of the power-train.

The following subsections consider constrains of each power-train components. After the mechanical solution has been determined, motors could be chosen.

2.2.1 Motors

The motor's main constraint was size and shape. Secondary requirement was equipping the RC motors with position sensors. Typical sensor used is such application is hall sensor. Hall sensors measure electro-magnetic field position of the rotor and help the motor controller control the phase switching in a low RPM of the motor. Based of measurement of the mechanical position of the rotor's poles. Then the controller can switch the phase in the right moment. Solution without hall sensor is still usable, however for low speed regime are not as smooth as with hall sensor. The motor controller makes a rotational magnetic field with a certain electro-magnetic field RPM, while it does not have information about the rotor's position. The lack of the rotor's position occurs in low RPMs. This case is presented as "kicking" of torque, while the rotor does not have the same RPM as an magnetic field generated by a motor controller. For more information see. [37]. The information about the rotor's position is important for torque control in low RPM.

Outrunner type of motor is used in aircraft RC models. I could not find any manufacturer who would equip the outrunner motors with hall sensors. In that time I did not know about a solution based on advanced control and estimation of rotor position, which does not need hall sensors for measuring a rotor position (see section 4.2).

The Czech manufacturer AXi model motors was selected. The motors from series 2814/xx and 2808/xx fit by size to the existing mechanical solution. Power of the series 2808/xx was not sufficient enough. Hence I focused on 2814/xx series. Motors with different back EMF constant ($K_e\ V \cdot rad^{-1}$) are offered. For more information about back EMF constant see **[6]**. RC motors are described by K_v constant declared in $RPM \cdot V^{-1}$. The series 2814/xx offers K_v from range (620, 840, 1035, 1390, 1640, 2850 K_v)[28]. The back EMF constant is essential for calculating the maximal speed of the SDP.

2.2.2 Maximal speed

When the electrical and mechanical parameters of the potential motor's series are fixed, the maximal speed needs to be considered. For calculation of maximal speed knowledge of circumference of the tire profile (267mm), gear ratio, the minimum voltage of the battery pack and back EMF constant is necessary. A requirement for maximal speed was consulted with a racer of 1:10 scale-down category. Based on thus, I decided that the maximal speed should be at least 50 $km \cdot h^{-1}$, which is approximately 14 $m \cdot s^{-1}$. Minimum rotations per second of the wheel at the desired speed are $52.4\ rot \cdot s^{-1}$. Next, the minimum voltage needs to be set.

2.2.3 Bateries

The batteries were selected according to the SDP (Serpent SDX spyder) spatial construction. A choice using two shorter batteries was a better solution than using one longer battery. Batteries: 2x Shorty Saddle P5 with configuration 2s1p, nominal voltage $7.4\ V$ and capacity 3900 mAh were selected. Values were adopted from [29]. Mentioned batteries fit well into the bought model. According to a manufacturer recommendation, the capacity of the batteries should sufficiently manage most of driving profiles. Constrains for minimum voltage is based on maximal speed being available for. Usually, nominal voltage of LiPo topology is $3.7\ V$, maximal voltage is $4.2\ V$ and cut-off voltage $2.75\ V$. The values were adopted from [27]. If the configuration of the accumulator pack is 2s1p, then the cut-off voltage is $5.5\ V$. The minimum voltage is $6.5\ V$ in order to protecting the battery pack.

2.2.4 Pulleys

The gear ratio is important part influencing maximal speed. Some of the suitably available pulleys at the market were (15T, 16T, 18T, 20T, 34T). I mentioned only the sizes that fit to the existing mechanical solution. Where T means number of teeth per a pulley. I took into account that the biggest 34T pulley was linked with the wheels directly. Only smaller pulleys (15T, 16T, 18T) could be on motor's side, because of mechanical restrictions. Then potential ratios of pulley system are (2.27, 2.125, 1.89). Those ratios substituting in equation 2.1 together with others parameters as (minimum voltage of the batteries $U_{min} = 6.5\ V$ and the circumference of the driven wheel $O = 0.267\ m$), then it leads to two feasible solutions. First is $v_{max} = 14.09\ m \cdot s^{-1}$ with $K_v = 1035\ RPM \cdot V^{-1}$ and gear ratio 2.125. Second result is 17.71 $m \cdot s^{-1}$ with with $K_v = 1390\ RPM \cdot V^{-1}$ and gear ratio 2.27.

2.2.5 Summarizing

The maximal speed of traction system was designed so that the SDP will be able to reach the speed at the minimum voltage, for sure. The maximal speed will be higher, due to fact that at such high speed as 50 $km \cdot h^{-1}$ a circumference of the tire profile become bigger. It is caused by centrifugal forces. Also, higher speed can be reached when batteries are fully charged. Therefore in the result, the maximal speed can be higher than the designed maximal speed. See equation 2.1.

$$v_{max} = \frac{K_v \cdot U_{min} \cdot O}{60 \cdot G_{ratio}} \tag{2.1}$$

Equation 2.1 shows how fast the buggy will be able to drive. Where $O = 0.267\ [m]$ is the circumference of the driven wheel, $U_{min} = 6.5\ [V]$ is the voltage of the batteries, $Kv\ [rot \cdot min^{-1} \cdot V^{-1}]$ is the constant of the motors and $G_{ratio}\ [-]$ is the gear ratio.

Following list summarizes important parameters of the power-train.

- **Motor:** AXI 2814/16 V1
  - Kv = $1035\ RPM \cdot s^{-1}$
  - Poles = 14
  - Max. Current = $35 - 40\ A$
- **Gears:**
  - Motor pulley = $16\ T$
  - Wheel pulley = $34\ T$
  - Ratio = 2.125
- **Bateries:** 2x Shorty Saddle P5 (2s1p)
  - Max. Voltage = $8.4\ V$
  - Min. Voltage = $5.5\ V$, ($6.5\ V$)
  - Capacity = 2x $3.9\ Ah$
  - Max. discharge I = $55\ C$, ($110\ C$)
- **Platform (SDP):** Serpent SDX4 Spyder
  - Max speed = $14\ m \cdot s^{-1}$, ($50\ km \cdot h^{-1}$)
  - Max. rot wheel = $52.4\ rev \cdot s^{-1}$

2.3 Odometer measurement

The Serpent SDX4 Spyder is a natural all-wheel-drive platform. The designed mechanical solution supports using rear wheel drive, only. Although, the original front drive system can be used for velocity measuring. The front system contains an open differential. The speed of the differential's ring is always equal to average speed of the driven axle wheels [5]. This solution brings a small advantage in compare to measuring each front wheel separately. It is mechanical averaging. In the case of the SDP, the front wheels are not able to break or drive. Initially, the wheels were designed for steering purposes, only.

A contact measuring of velocity is quite sensitive to drive condition. Buggy cars are more-less designed for driving at the off-road condition, as you can see in figure 2.5. Lot of time the front axle does not have contact with the ground. Nevertheless, the platform's front axle is quite stable on an asphalt track. Due to the fact of the off-road condition, the SDP needs additional non-contact velocity sensor. More about velocity measurement in section 3.2.

Figure 2.5: Buggy in an action. Adopted from [10]

2.4 Steering servomechanism

The chosen servomechanism was Servo HBS 880BB. For more information see [35]. The first criterium for servomechanism was its dimension. Not every servo can fit into the existing platform. The second criterium was, how fast and how much load the servo can handle during steering maneuvers. Mentioned servo speed 60 $degree$ at 0.06 s at 6 V and force 0.932 $N \cdot m$. Most of the servos have an internal regulator with a position detection. It means that the input of the servo is a position which the servo is commanded to reach. Hence the servo does not need any additional sensor of the position. Because the position of the steering wheel can be considered as servos' input signal. When the controlling signal is sent to the servo, then the servo will reach the position at the declared time (60 $degree \cdot 0.06$ s).

3 Vehicle measurement system requirements analysis

Torque vectoring is one of the requirements that the SDP should be able to support. Torque vectoring is kind of controlling algorithm based on distributing torque to vehicle wheels to generate the required yaw moment of the car, which can be utilized to stabilize a vehicle. For better understanding see picture 3.1. For more information see [31].

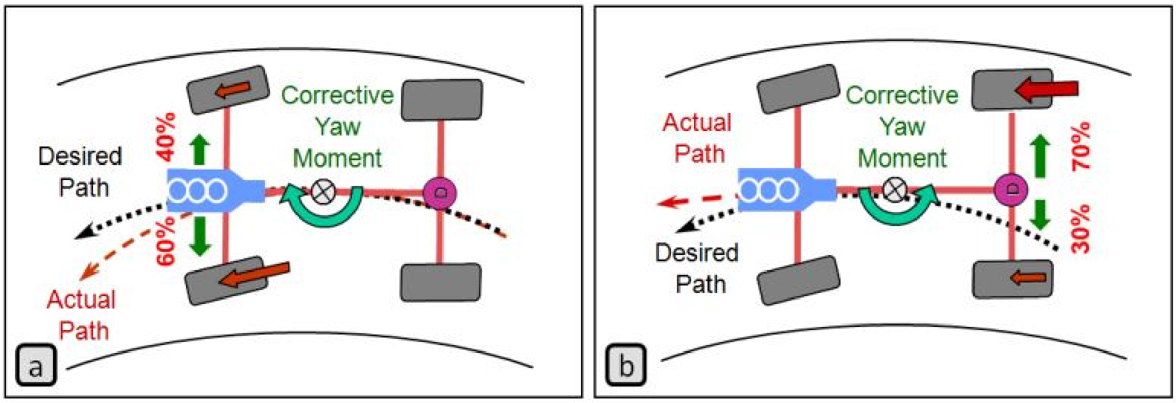

Figure 3.1: Base principle of the torque vectoring. Adopted from [17]

Another requirement is Steer-by-wire. The main idea is that a steering wheel is not connected to steered axle by mechanical parts directly. It means that a driver does not have direct mechanical link to steering axle position. But instead, mechanical part a steering wheel contains a sensor which senses its position. Then the signal is sent to the control unit, and the servo mechanism drive the steer axle to get the requested position by a steering wheel. Such technology allows applying advanced algorithms to help controlling the whole vehicle. For example, the angle of the steering wheel is evaluated by a control unit as a direction which a driver wants to go (reference signal). Control unit calculates necessary data and using the steering axle control keeps the required direction of the car. In practice can happen that steering wheel is in the stable position during the drift of the vehicle, but steering axle travels from one to another position many times to ensure the demanded direction coming from a driver. Adopted from [33]. For more information see [33]. Finally, the SDP should offer a variety of measurements from which a programmer would be able to calculate or read data of the velocity longitudinal vector v_x, velocity lateral vector v_y, the angular speed of the driven wheels and torque applied to the driven wheels. The parameters are commonly used in advanced vehicle control algorithms. Also, the operators of the SDP requires those parameters. Particular ways of measurements will be described in this chapter.

3.1 E-motor torque measurement

A torque measurement of a rotating shaft is a specific measurement, especially when a measuring element needs to be integrated into an existing mechanical shaft system. Not every application allows inserting of a measuring element between source and load. The SDP needs a measurement which will not modify an existing mechanical system. Following list shows some types of measuring which could be used.

List of measurement methods:

- On shaft measuring
  - Contact
    - strain gauge with slip ring
  - Non-contact
    - based on the magnetostrictive effect
    - strain gauge (using telemetry or transformer)
    - phase shift measurement
    - SAW devices
- Indirect measuring (input values of the engine)

Unfortunately, I could not find any available sensor for the SDP purposes, because of its scaled-down size. Hence I was focused on indirect measurement which is based on knowledge of electric motor constants and its input current. For more information about torque measurement see [56], [16], [11], [34], [19] and [6].

3.1.1 Non-direct torque measurements

The cheapest and the best way based on my objectives for the SDP is an estimation of torque by measuring a phase current of the motor and afterward calculation of mechanical torque. Even more, this method does not require an additional sensor which requires mechanical changes.

A torque can be calculated from a phase current of a brushless DC motor, according to the article [6]. Even more, electromagnetic torque is a linear function of phase current, based on [6]. The article also mentions others non-direct methods, which uses different values for estimating or calculating torque of BLDC motors. Consequently, accuracy of the torque depends on current measurement. The motor controller should be able to control a torque of the driven wheels. For this purposes, the measurement of at least phase current is needed. The drawback of RC model's motor controllers is that the most of them do not provide such data.

3.2 Velocity measurement

In the automotive industry a variety of sensors for measuring velocity are used. Type of sensor depends on drive system of a vehicle and operational conditions. For a classic non-all-wheel-drive car can be used following measuring tactics. A speed sensor is mounted on the non-driven axle. It ensures that most of the time the sensor provides relevant data. Except hard braking without ABS.

A more difficult situation occurs when all-wheel-drive vehicle velocity is measured. The situation may happen during the acceleration when all wheels are slipping. Hence they lose relevant information about true vehicle velocity. If the velocity is used for improving acceleration, then it would be better considering non-contact speed measurement. For example, optical measurement, doppler-effect based measurement, or measurement based on GPS. Some of the sensors use accelerometer and gyroscope for increasing update rate and reliability of the sensor. Optical sensor and Doppler sensors have much accuracy but are very expensive. Mind that an active surface of the sensors has to be always clean. Also, minimum distance needs to be kept between the sensor and sensed ground. Adopted from [13]. These constrain make the sensors unusable at many conditions. For example, for the SDP case, it is not usable because the minimum required distance between the ground and the sensors cannot be ensured. Also, a majority of those complex sensors are not manufactured for the SDP dimensions, and they are expensive in comparison to entire costs of the SDP.

In the case of the SDP, the most reliable non-contact sensor is a GPS. It does not need any minimum distance between the ground and vehicle, because it is not focused on the ground but it stares at the sky. However, a weak point will be when GPS would lose an open sky. Despite the limitation the most of the time the GPS based systems could provide reliable data which can be used for calculating the velocity longitudinal vector v_x, velocity lateral vector v_y and slip angle. See figure 3.2 where v_x velocity vector is in the direction True Heading and and v_y is perpendicular to v_x.

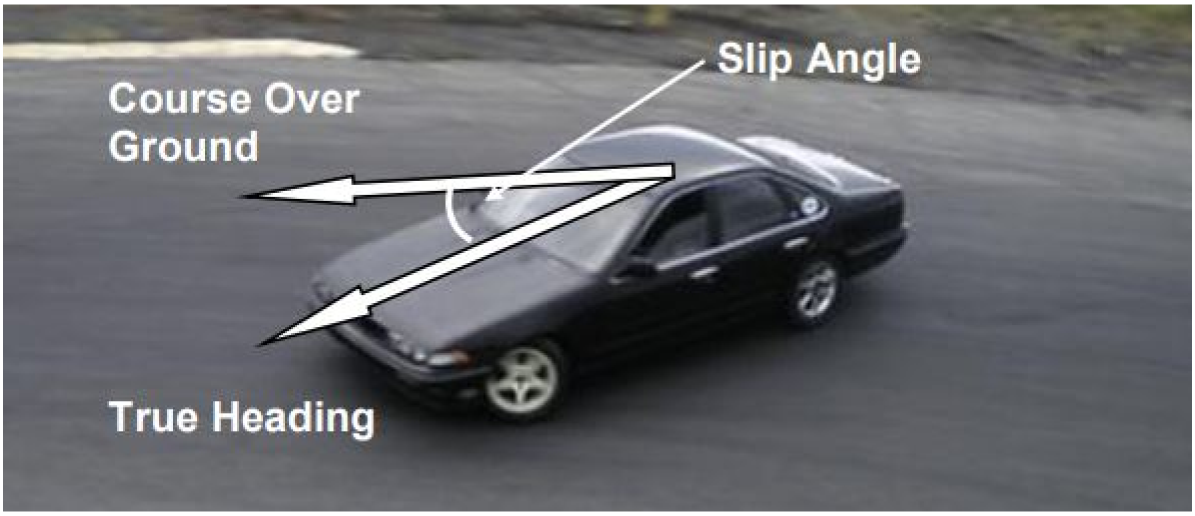

Figure 3.2: Illustration of Slip Angle and True Heading as the direction of v_x vector. Adopted from [3]

3.2.1 Contact based (Odometry)

As I mentioned in the section 2.3 contact measurement of the velocity is very reliable, if driving conditions do not lift the front axle. I assume that this measurement will be enough for most of the verification tests. The front differential could be used for velocity measurement (section 2.3).

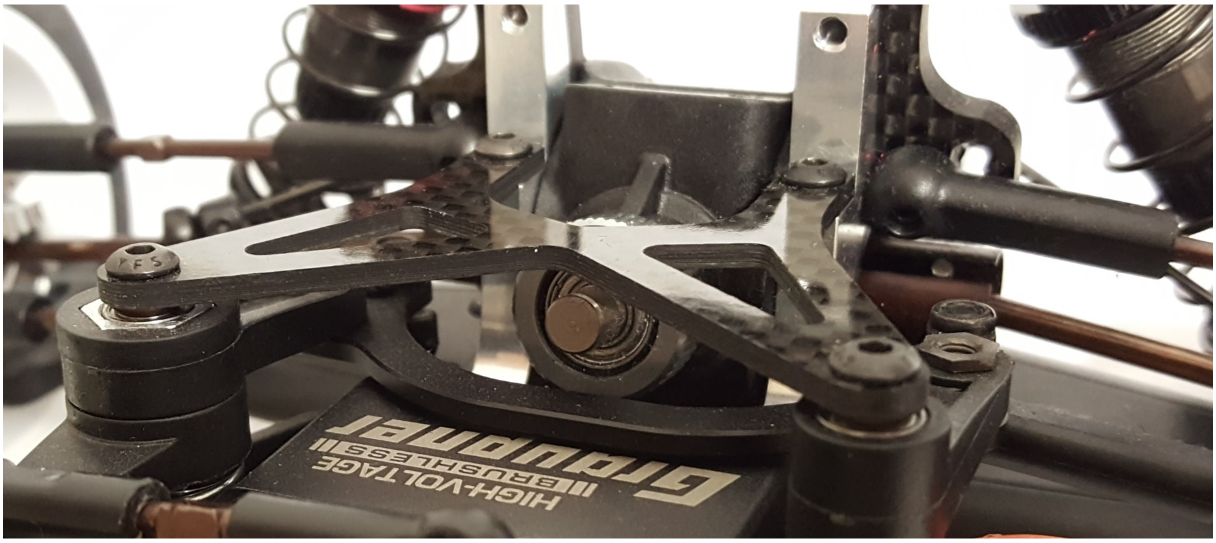

Figure 3.3: Detail of the front differential.

To evaluate limiting factors of the sensor, following parameters are required. Circumference of the front wheels $O_{wheel} = 0.267\ m$, gear ratio of the differential $G_{ratio_{diff}} = 2.6$, considered maximal speed $v_{max} = 14\ m \cdot s^{-1}$. The parameters have been measured on the SDP and will be used to calculate maximal revolutions per minutes Rev_{max} of the differential shaft. Then minimal required construction RPM can be calculated as

$$Rev_{max} = \frac{v_{max} \cdot G_{ratio_{diff}}}{O_{wheel}} = 8400\ [rev. \cdot s^{-1}]. \tag{3.1}$$

Where $Rev_{max}\ [rev. \cdot s^{-1}]$ means maximal revolutions per minutes at maximal speed $v_{max}\ [m \cdot s^{-1}]$. Circumference of the front wheel is $O_{wheel}\ [m]$.

A minimum resolution per rotation of the sensor depends on how accurate the traction algorithm needs to be. A lot of vehicles use sensors which primarily provide data for the ABS system. In the article [4] motorbike used a sensor with 100 pick-ups per revolution. I assume that for motorcycles the ABS requirements are higher than the requirements for cars, due to the motorbike braking issue, while cornering. The article [36] mentioned that Peugeot 306 Rallye used 24 pulses per revolution settings for traction control. In the article [30] was mentioned that conventional sensors for ABS usually offers 48 pulses per revolution. I consider that 24 pulses per revolution of the front wheel as a minimum value.

3.2.2 Non-contact based

Available non-contact sensors on the market with suitable price and accuracy offers velocity update rate around 200 Hz. They are used in motor-sport and also in Formula Student competition. This kind of sensors could not be found in a suitable scale-down form for SDP. The article [39] shows that getting relevant data as slip angle and longitudinal velocity is possible by fusing GPS, accelerometer and gyroscope. The GPS provides quite accurate data. However, $1 - 10\ Hz$ which GPS-es usually provide, is too slow for automotive. Hence it needs to be combined with a much faster IMU, adopted from [39]. The lowest level of quality of the GPS I did not estimate. Quality of the results data depends on fusing IMU and GPS. Therefore I was choosing the best GPS which I could find respectively budget, availability, size, and quality. In the article [39] GPS with update rate 10 Hz has been used with velocity error $0.108\ km \cdot h^{-1}$.

Sensors have been tested on Mercedes Class E car. Using proper algorithms which achieve good results. Therefore I choose the separated implementation of GPS, accelerometer and gyroscope. This solution would require additional signal/data fusions and filtration.

3.3 Inertial measurement

Measuring of the inertial forces is necessary for improving a velocity or slip angle measurements. In the section 3.2.2 was elaborated the advantage of fusing a GPS and an IMU data. In the article [39], the used IMU noise level $0.06\ m \cdot s^-1$ (1σ) for the accelerometer and $0.2\ deg \cdot s^{-1}$ (1σ) for gyroscope and sampled at rate of $100\ Hz$. Similar IMU with mentioned parameters is too expensive in comparison to a price of the SDP project. The SDP has higher dynamic caused by smaller weight and scaled-down size than a regular car. Even the distance between GPS dual antennas is lower than it is expected in a regular car. It increases the GPS course error. Hence I assume that the IMU for SDP should be better than mentioned one in the article.

According to the article [22] and simulations which have been done in eForce formula student team, I assume that FIR filter of order 30 should be able to extract an effective signal. Also, require as minimum $4\ g$ of the scale of the accelerometer and 200 dps for the gyroscope. The scales I would consider twice bigger, due to SDP higher dynamics as $8\ g$ and $400\ dps$. Assuming the output data has to be at frequency of $100\ Hz$ and a delay caused by FIR filter of 30th order should not be more than 2 samples ($0.02\ s$), then minimum update rate of the IMU should be at least $1.5\ kHz$.

$$UpdateRate = \frac{Order \cdot f_{Data_{out}}}{2} \tag{3.2}$$

Where $f_{Data_{out}}$ $[Hz]$ is a required output update rate after filtration, $Order$ $[-]$ is an order of applied FIR filter. I determined mentioned parameters as an orientating minimal reference.

4 Electronic equipment of the vehicle

The goal is to ensure that RC buggy will be able to carry out advanced system algorithms as mentioned in the chapter 1. Hence existing electronic assembly had to be changed, see figure 4.1.

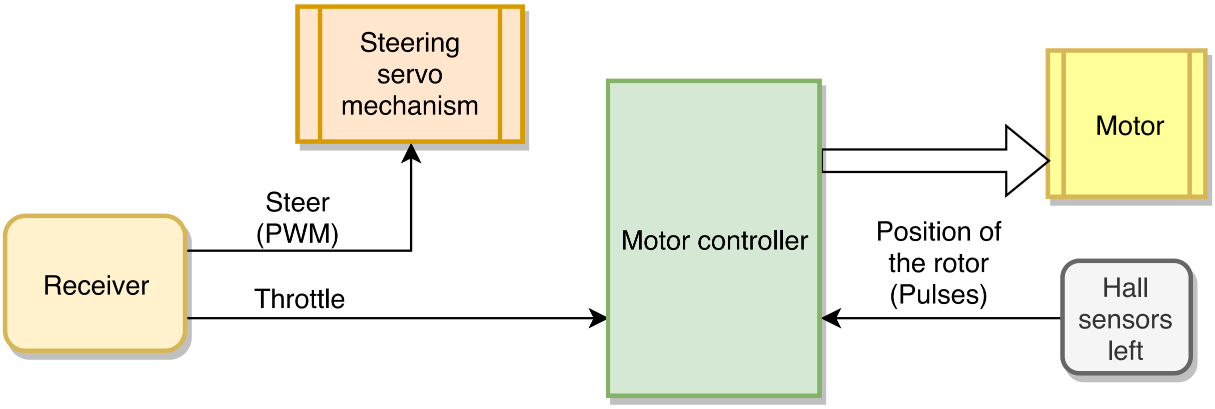

Figure 4.1: Block diagram of the electric connection of a commonly available RC car.

All the controlling signals are distributed directly to the actuators (servo mechanism or motor controller). It is a standard configuration of the RC cars, where an open differential distributes a torque of the motor to the wheels. A driver fully controls steering and throttle, and any additional system does not support him.

In the figure 4.2 you can see significant changes. The figure shows the first solution (Mamba solution) of the configuration of the SDP. This solution has been realized without hall sensors and velocity encoder, as a first working prototype. All of the original devices are also used in the Mamba solution as a Receiver, Servomechanism, and Motor Controllers. The Mamba solution was extended by the Central board (main processor), one more motor, one more motor controller, IMU and is planned to be extended by encoder and custom hall sensors.

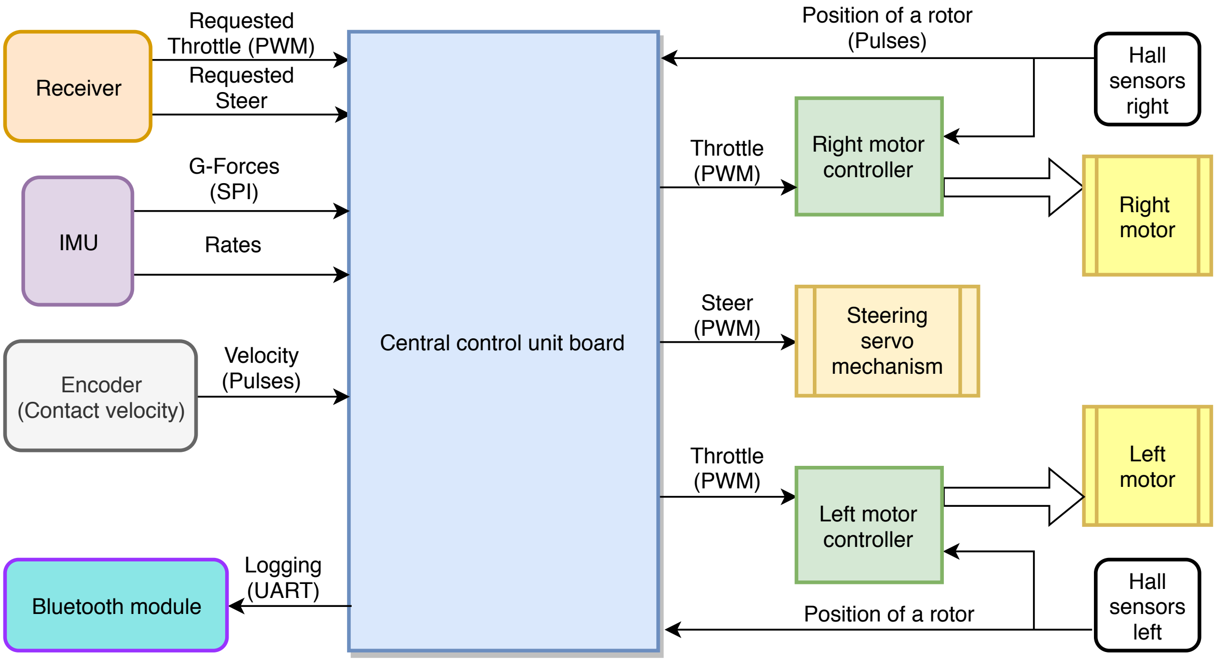

Figure 4.2: Block diagram of the Mamba solution.

In the figure 4.3 is shown the current state of the SDP. The second solution (VESC solution) of the SDP is extended by GPS and VESC motor controllers. The encoder was replaced by custom-made hall sensor solution. Hall sensors which measure a position of the rotor have been replaced by VESC motor controller which can sense the position without additional sensors. Each component is described in this chapter in more detail.

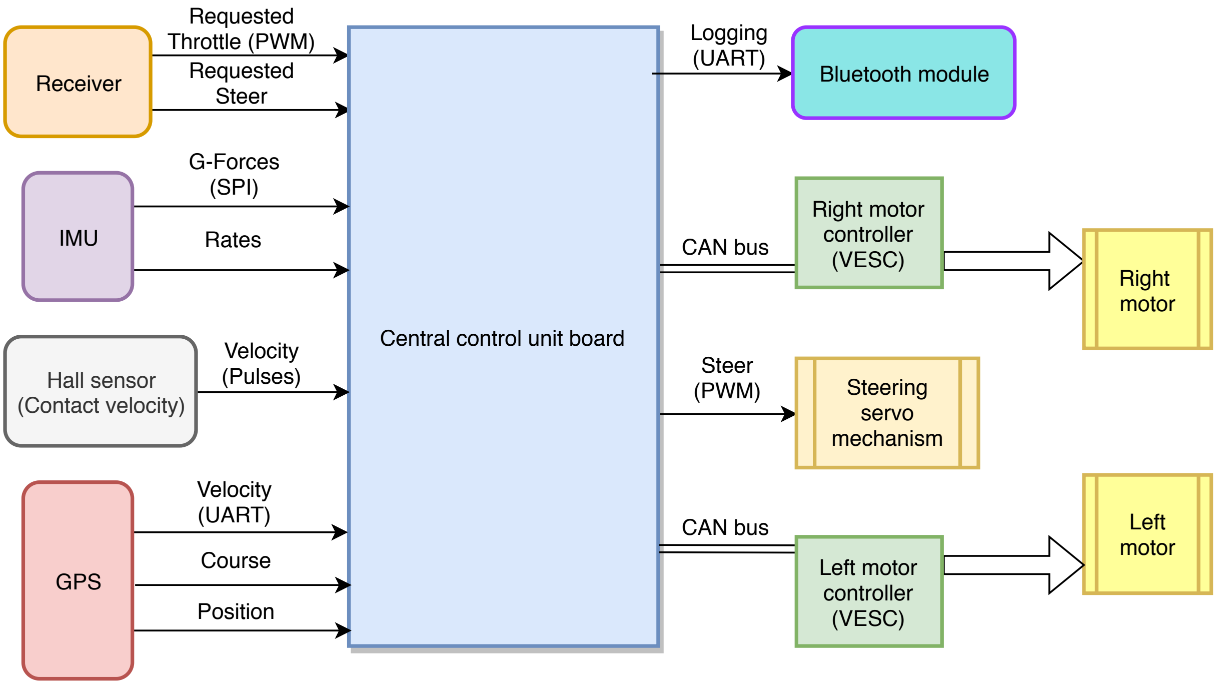

Figure 4.3: Block diagram of the VESC solution. Actual state.

The crucial step was to find a convenient Central board with suitable Input/Output ports with proper alternative functions, which would be available on existing connectors. The proper board already existed on the market. It was a big issue, but still less time consuming than to develop a custom-made board. More about the boards you can read in following statements. Critical limitation of the SDP is a space. The Central board has a perfect ratio of size in comparison to its functionality, but it still contains a lot of unused components, which could be removed. At the end of the development process of the SDP, all bought boards could be replaced by a custom-made board which would contain only necessary parts. It will save space and allows a body case to be installed. Consequently, the body case will protect all components against dirt and dust. A body cover can significantly prolong a lifetime of the SDP. Then the RC buggy can be verified with any similar competitors at real race conditions.

In the figures 4.2 and 4.3 is not shown an Interconnecting board. It is the custom device, which is powered by the batteries and giving stable 5 VDC for electronics. All wirings lead to the Interconnecting board and then continue to their destinations. The Interconnecting board is described in section 4.3.

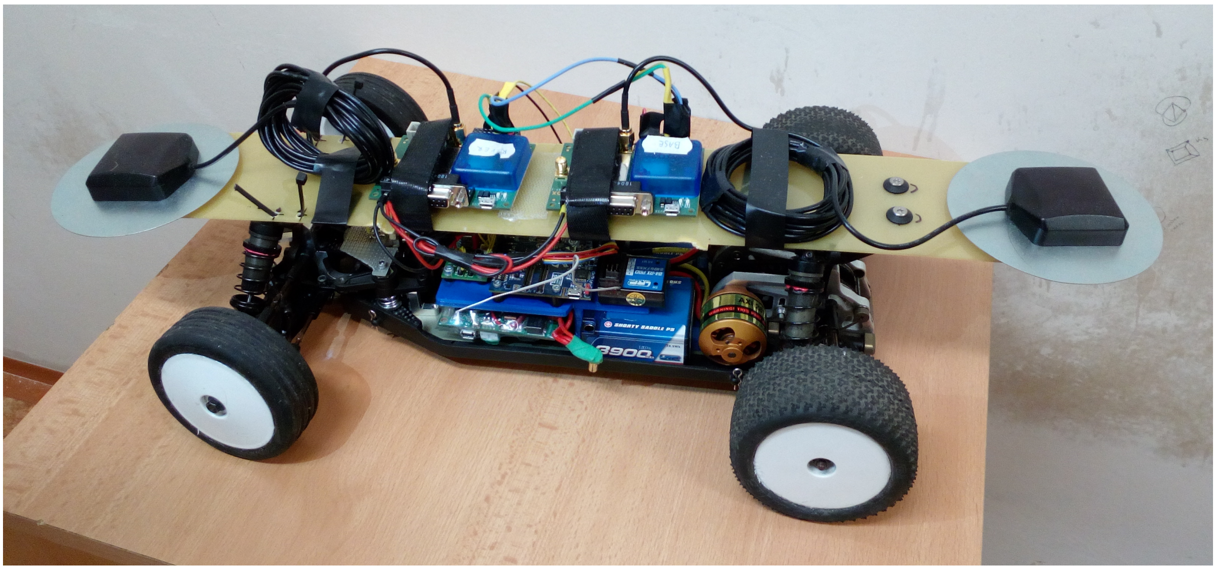

Figure 4.4: Actual physical state of the SDP.

4.1 Central board

The central board is the central "brain" of the SDP. It contains a programmable processor which can fully control received and sent signals. Requirements for that board were as follow

- 9x Input capture timer *(at least* $1\ MHz$ *and 16-bits)*
- 3x Output PWM timer *(at least* $1\ MHz$ *and 16-bits)*
- 2x UART
- CAN bus
- SPI

Every signal is available on the accessible existing connectors.

Let's investigate a timer specification. At maximum designed speed 50 $km \cdot s^{-1}$, see section 2.2, the rotation of the wheel is 52.4 $rotations \cdot s^{-1}$. Regarding section 3.2.1, 24 samples per revolution of the wheel is enough. I consider a resolution of 50 ticks of counter register as a minimum, while occur new event caused by a velocity incremental sensor. The swapping of states is considered at maximum speed. By multiplying the values. Then the timer should be able to work at minimum frequency $24 \cdot 50 \cdot 52.4\ \ rot \cdot s^{-1} = 62880\ Hz$. Therefore, I consider a minimum frequency $0.1\ MHz$.

Let's calculate the minimum resolution of the timer. The minimum measurable speed, which I set to $0.1\ km \cdot h^{-1}$ is approximately $0.1\ rev \cdot s^{-1}$ of the wheel. It is 2.4 timer event per second. With $0.1\ MHz$ timer frequency the counter register will rise on the level of 41667 ticks per the slowest event. I consider the minimum resolution should be 16-bits. Then resolution of the timer could be set to $10\ \mu s$, which is $0.1\ MHz$ of the clock frequency.

Receiver provide PWM signal with period $0.02\ s$ ($50\ Hz$). Width of the PWM pulse is minimum $1\ ms$ and maximum $2\ ms$ see figure 4.10. Effective signal is $2\ ms - 1\ ms = 1\ ms \rightarrow 1000\ \mu s$. I consider the $1\ \mu s$ of the timer is a minimum resolution which the buggy car would need for throttle and steer input resolution of 0 to 100 %. It means 0.1 % of one step. The period of the new timer event is $20000\ \mu s$ (caused by $50\ Hz$ of the received signal) which is 20000 ticks at timer resolution $1\ \mu s$. Hence, minimum requirements for the timer which will handle steer and throttle signal is at minimum $1\ MHz$ clock timer frequency and at least 16-bits resolution of a counter register.

Related to the figure 4.3 UART is necessary for GPS and telemetry. SPI is required for IMU. Output PWM timers for steer signal to the servo mechanism and 2x throttle signal to the old concept motor controllers. Input capture timer for receiving steer and throttle signal from the receiver and 6x signals for hall sensors, which would have measured an position of the motor's rotors according the Mamba solution. Availability of the CAN bus was an advantage. In the time when I was choosing the Central board, I did not know that common RC motor controllers could be replaced by VESC 4.12. Thankfully, the VESC supports CAN bus communication, and it was a good decision because the new motor controller VESC 4.12 can provide a lot of additional information, including ERPM data via CAN bus. It completely removed the need to use complementary rotor position Hall sensors.

The 96Boards Mezzanine board with STM32F446 MCU and sensors (B-F446E-96B01A) satisfied all mentioned requirements. For more information about the board see [43].

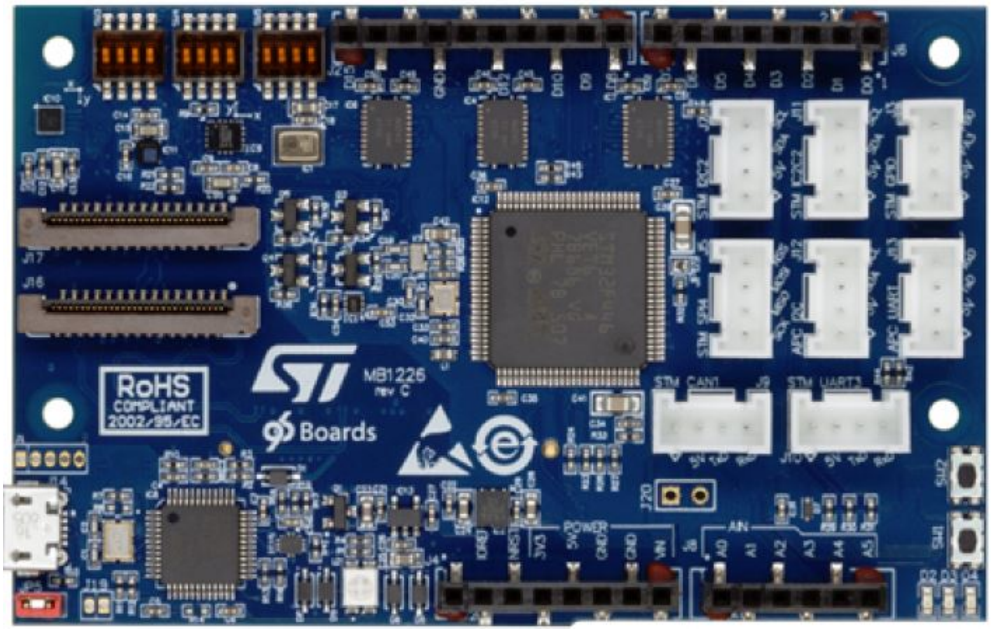

Figure 4.5: View of the central board Mezzanine B-F446E-96B01A. Adopted form [43].

Mezzanine board is based on the STM32F446 processor family, which fulfill all requirements. For more information about the processor see [44]. Following list of the features adopted from [44].

- 12x 16-bit timers, 2x 32-bit timers, each with up to 4 IC/OC/PWM or pulse counter at 90 MHz or 180 MHz
- up to 112 5V tolerant pins
- 180 MHz clock frequency
- all necessary peripherals

I also choose it, because I already had a good experience with this processor and also it has library support of the Simulink. For more information see [25].

Central board does not have implemented CAN transceiver. External board (*TJA1050 CAN controller interface*) is used. Original CAN transceiver (*TJA1050*) has been changed to *MCP2551* for input pins voltage level compatibility.

4.1.1 IMU

IMU is part of the Central board. It is LSM6DS3 from STMicroelectronics manufacturer. For more information see datasheet [42]. This device provide maximal update rates for accelerometer of 6.66 kHz with range of scale ($\pm 2,\ \pm 4,\ \pm 8,\ \pm 16\ g$) full scale and for gyroscope 1.66 kHz with scales ($\pm 125,\ \pm 250,\ \pm 500,\ \pm 1000,\ \pm 2000\ dps$) full scale. The accelerometer offers sufficient update rates and scales as was mentioned in section 3.3. The higher update rate of accelerometer allows applying filters with higher order. The accelerometer should be calibrated to reach better accuracy. Noise density parameter for chosen IMU LSM6DS3 at range 8 g is 110 $ug \cdot \sqrt{Hz}^{-1}$ and for gyroscope it is 7 $dps \cdot \sqrt{Hz}^{-1}$. The parameter of a RMS noise density depends on chosen measured bandwidth. According the article [20] it depends on chosen filter and its cut-off frequency. I consider the solution using separated IMU based on MEMS technology which are more price and weight available than existing complex solutions described in chapter 3.2. SBG manufacturer is using combination of MEMS with GPS, too. The company offers complex INS solution (Ellipse2-D Dual Antenna RTK INS, see [48]) with friendly diameters and weight. Despite extreme high price, similar model can be landed from Formula Student team eForce [14]. I suggest to try to implement this INS in SDP in the future.

4.2 Motor controller

Motor controller and BLDC motor cannot operate without each other. I focused on some parameters as maximal current, maximal operating voltage and maximal supported electro-magnetic field rotations per minutes. All considered motor controllers were designed for RC motors. Hence, I did not investigate any other parameter. I considered that it is widely optimized. In the section 2.2 has been investigated motors and all power-train. The important fact was that used motor has 14 poles. According to the section 2.2 the mechanical RPM can be substituted in equation 4.1. Equation 4.1 was adopted from [15]. Then the maximal electrical RPM was determined as

$$RPM_{electrical} = \frac{p}{2} \cdot RPM_{mechanical} \tag{4.1}$$

$$RPM_{electrical} = \frac{Kv \cdot V_{max} \cdot p}{2} = \frac{1035 \cdot 7.4 \cdot 14}{2} = 53613 \tag{4.2}$$

Where p $[-]$ is a number of poles of the motor. Kv $[rev. \cdot min^{-1} \cdot V^{-1}]$ is back EMF constant, see section 2.2. V_{max} $[V]$ is maximal battery voltage and $RPM_{electrical}$ $[rev. \cdot min^{-1}]$ is revolutions per minute of the electro-magnetic field and $RPM_{mechanical}$ $[rev. \cdot min^{-1}]$ is mechanical revolution per minute of the motor's rotor.

The motors AXI 2814/16 recommend maximal current at $35 - 45$ A.

4.2.1 Mamba Micro X Extreme

Motor controller mamba micro X extreme from Castle manufacturer was first adept for prototyping. Its parameter fulfill all conditions. Data adopted from [7].

- Input voltage from 5.5 V to 12.6 V
- Output current from 40 A to 100 A
- Electrical RPM at least 65000 $rev \cdot min^{-1}$

Some motor controllers measure and provide data as a phase current on the bus. This motor controller did not provide any data which could be used afterwards for monitoring or control. Measurement of motor internal signals turn out to be critical, therefore a motor controller providing such functionality was necessary. Mamba motor controller has an excellent ratio between power and size. It is an good backup alternative to the following controller VESC 4.12.

4.2.2 VESC 4.12

Motor controller VESC 4.12 was fully designed by Benjamin Vedder [53]. It is an open source ESC.

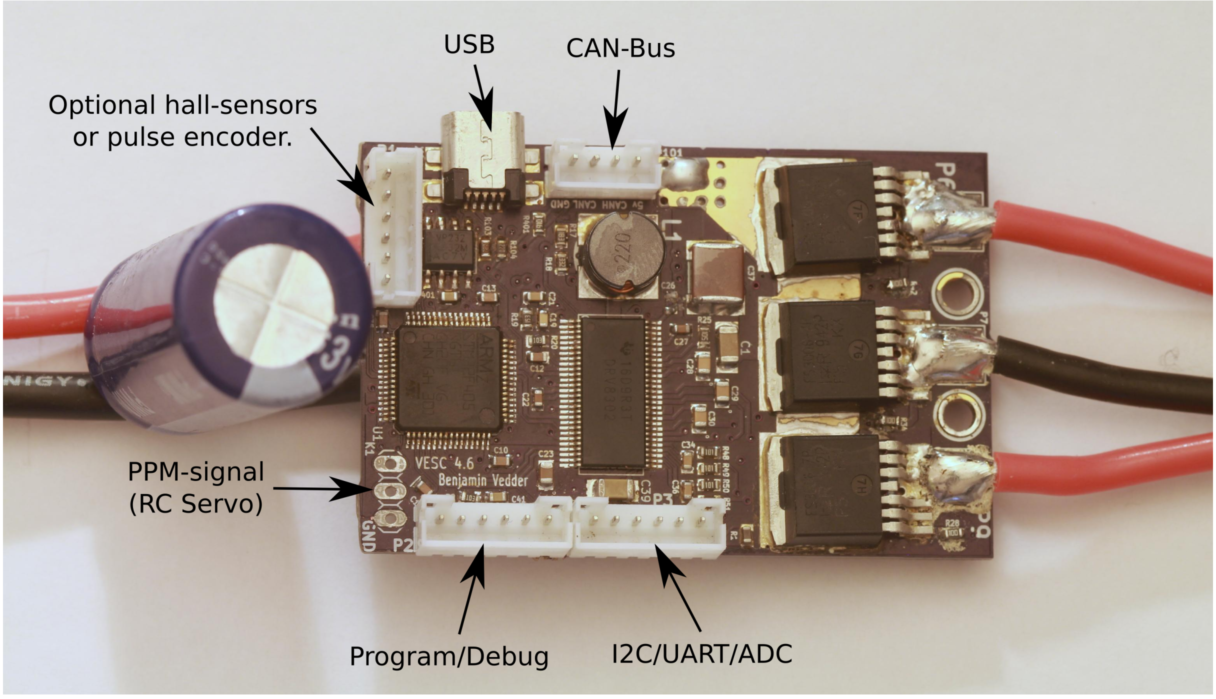

Figure 4.6: Motor controller VESC 4.12 designed by Benjamin Vedder. Picture was adopted from [53].

Specifications of this ESC are not commonly available as products from renowned manufacturer. A lot of information were derived from designer answers at a forum or from demo code. Here are some electrical parameters of VESC 4.12. Adopted from [53].

- Input voltage from $6\ V$ to $60\ V$
- Output current up to $240\ A$ for few seconds or $50\ A$ depending on the conditions
- Electrical RPM at least 60000 $rev \cdot min^{-1}$

Also, the VESC satisfied the requirements. Even more, it offers additional useful features. The VESC is used for measuring some important values.

- Battery voltage and current
- Motor voltage and current
- Electrical RPM

For more information about VESC see [53]. This ESC supports a mode in which it does not need any hall sensors for measuring a position of the rotor. It can operate with sensored or sensor-less motor. It is based on measuring the back EMF of a motor.

BLDC motor controller controls a BLDC motor where one phase is positive energized, the second is negative energized and third is non-energized. When the winding is off, it is the right time for measuring a level of the back EMF produced by rotor's magnets. Regarding the section 3.1.1 back EMF depends on RPM of the rotor. Consequently, this controlling strategy is not able to be used when RPM is low. This is the reason, why some motors are equipped by hall sensors. A sensor-less control may cause non-effective operation when mechanical RPM is low. The VESC solved the problem by applying a certain sequence at start time. Kicking can be felt, if the rotor is stopped. It is caused by applying a predefined sequence at a certain frequency, which not respect the rotor position. The rotor is standing, while the electro-magnetic field is moving, which has a pulling effect on the rotor. If the rotor is not so much loaded, then it starts accelerating and following the rotating electro-magnetic field. If the rotor has enough RPM for measuring the position of the rotor, then the ESC will switch the phases exactly according to the rotor's position requires. Adopted from [18]. For more information see [18].

VESC declares that it does not need any position sensor, as it can estimate position from almost 0 RPM. It is called sensor-less field oriented control (FOC). Where the angular speed and position are observed from measured phase voltages and currents. Sensorless FOC has some limitations, adopted from [15]. The designer of the VESC avoids that using sensorless FOC with motors with high Kv constant which may damage a motor driver chip. The used motors by the SDP are convenient for this case. Even more alternative upgrade of the tractive system would offer a motor with even lower Kv constant.

Communication (controlling and readding data) with the VESC ESC can be realized by UART, CAN, I2C peripherals. In a case when two or more nodes are used, it is better to use a bus. I had an experience with CAN bus. Hence I choose for communication on the CAN bus. As I mentioned, VESC is an open source project, and sometimes lack of the proper documentation may cause a C source code of the device needs to be investigated. Via the GUI tool, the motor constant, limitations, regulators constants and so on, could have been tuned up. For more information see [53].

4.3 Interconnecting board

The interconnecting board is part of the system, which serves as a point, where all wires are physically connected. Its purpose was to supply necessary missing components and features. It provides stable power supply, a physical connection between devices using incommutable connectors, indicating signals occurrence and an ability to switch between original direct control and advanced control for demonstration of benefits. This board has already had two releases. The board is custom-designed in Altium designer program.

4.3.1 Mamba solution

Central board requires stable power supplying of $5\ V$. I choose to use a linear regulator type because I assumed that the Central board does not consume high current. Also, it needs a more stable voltage level for operation which is more important than efficiency. Linear regulator in comparison to a switching regulator is better choice for this purpose almost in all ways. For the first time the problem seemed to be trivial. However power source (battery), which should supply the linear step-down regulator, had the voltage range from $8.4\ V$ to $5.5\ V$ cut-off. The minimal input voltage for linear regulator should be a $6\ V$. I estimated that the Central board could have a current consumption at a maximum $150\ mA$. Output current rating I choose to $1\ A$ at a minimum as a reserve for cases if a new component should be added to the system. The part AZ2940D-5.0 satisfied all the requirements. In the scheme it is component called *LINREG* see 4.7, 4.8.

For saving time during the developing process, I design some signal occurrence LED indicators. In figure 4.7 you can see a LEDs (*LED1, LED2, LED3, LED4*). That LED shine only if there occurs a signal, otherwise they are turned off. *LED1* is for power supply indication. *LED2* to *LED4* are for signal indications. It helps debugging if a malfunction occurs.

The switch called *TV_(On/Off)* in figures 4.7, 4.8, switches between direct control state and advanced control. The direct control means that signals from Transmitter are directly distributed to the actuators (Servomechanism and Motor controllers) as you can see on figure 4.1. The advanced control option switchs to the configuration as you can see on figure 4.1. Purpose of this switch was to show differences between advanced control and direct control. When the old motor controllers was replaced by the VESC controllers the switch lost its purpose. Because it handles the servo mechanism only.

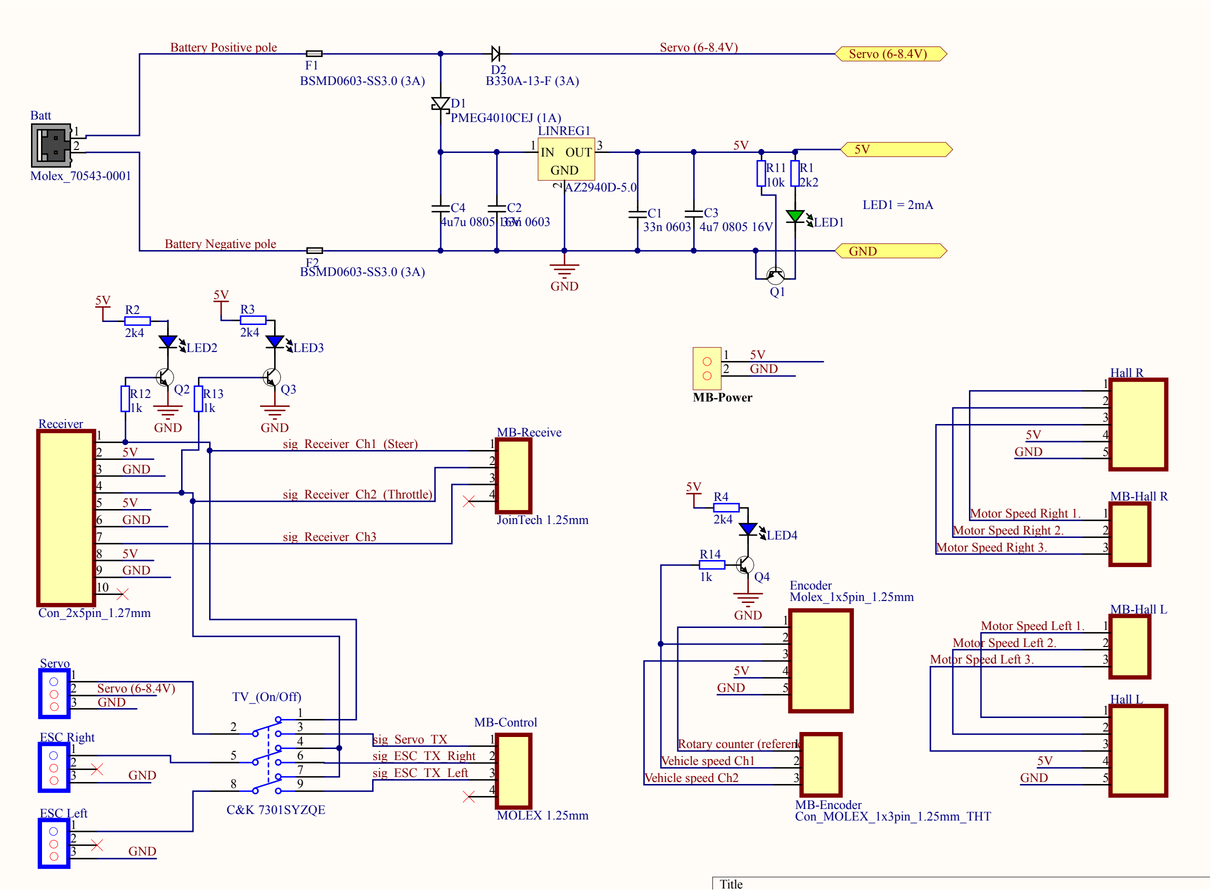

Figure 4.7: Wiring diagram of the Interconnection board ver. 1

4.3.2 Updated board - Mamba solution

The updated board brought some updates as under-voltage protection, signals indications, change of connectors and update some labels. The previous (first) version uses bipolar transistors for switching the signalization LEDs. Transistor switches LEDs that signal entry to its base. It could have influenced a signal condition. Hence I choose Schmitt-triggers for switching LEDs.

Under-voltage protection of the batteries was realized by two red LEDs with high luminosity. The purpose is to shine as much as visible that a driver would see the red light through body case. Then the driver should stop a session. The VESC motor controller has implemented under-voltage protection. It will switch off the output. Then the protection integrated into the updated Interconnecting board lost the purpose.

The previous version of the board used connectors Molex from Picoblade series with pitch 1.25 mm. They were too small that maintenance was difficult. I changed it to connectors which they have pitch 2.54 mm. Bigger connectors required more space. Dimensions of the board stayed the same. All components which were small enough migrated to the bottom side of the PCB.

The labels are essential for correct operation and reliability. An update of the names was required according to experience with the previous version.

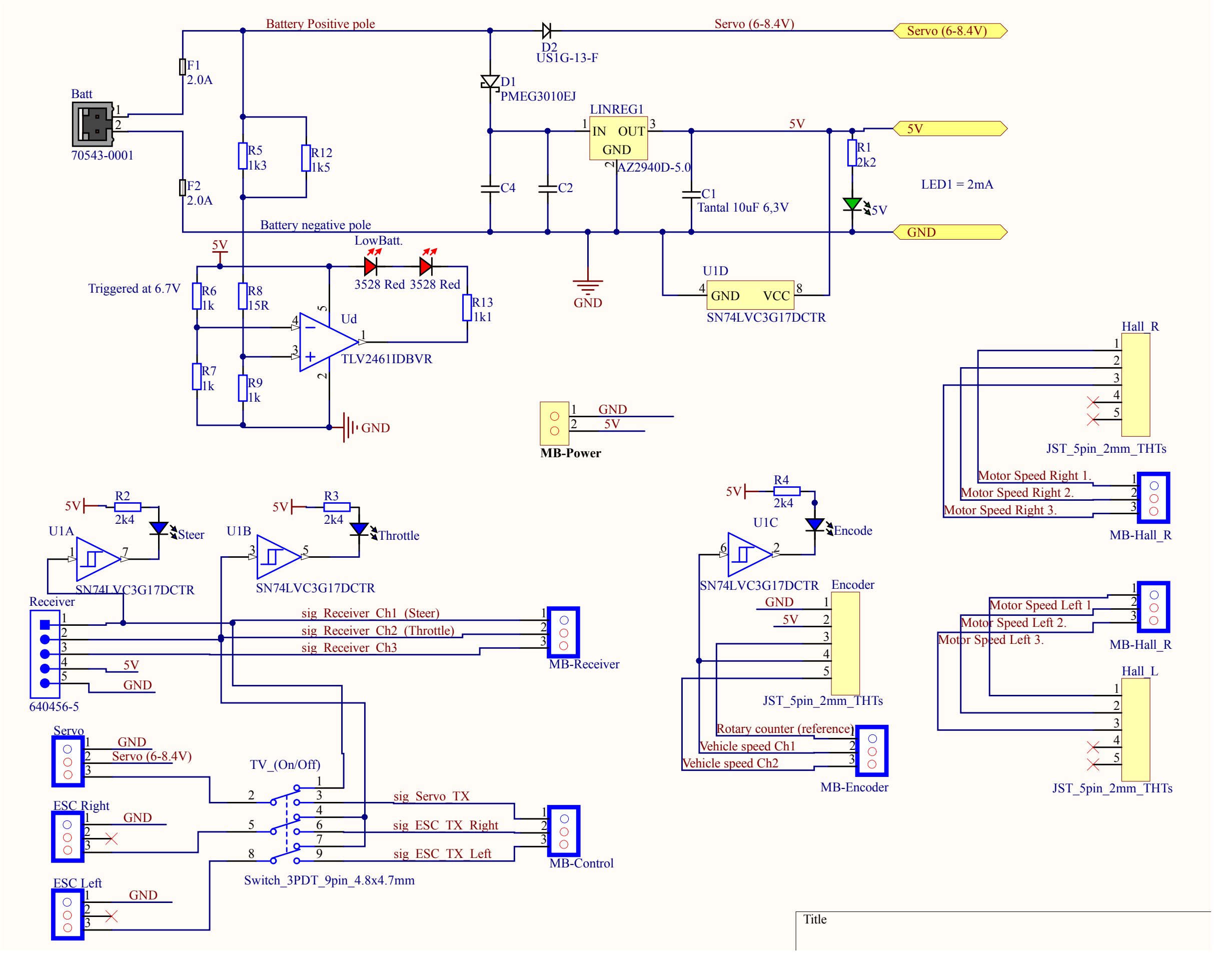

Figure 4.8: Wiring diagram of the updated Interconnection board for the Mamba solution

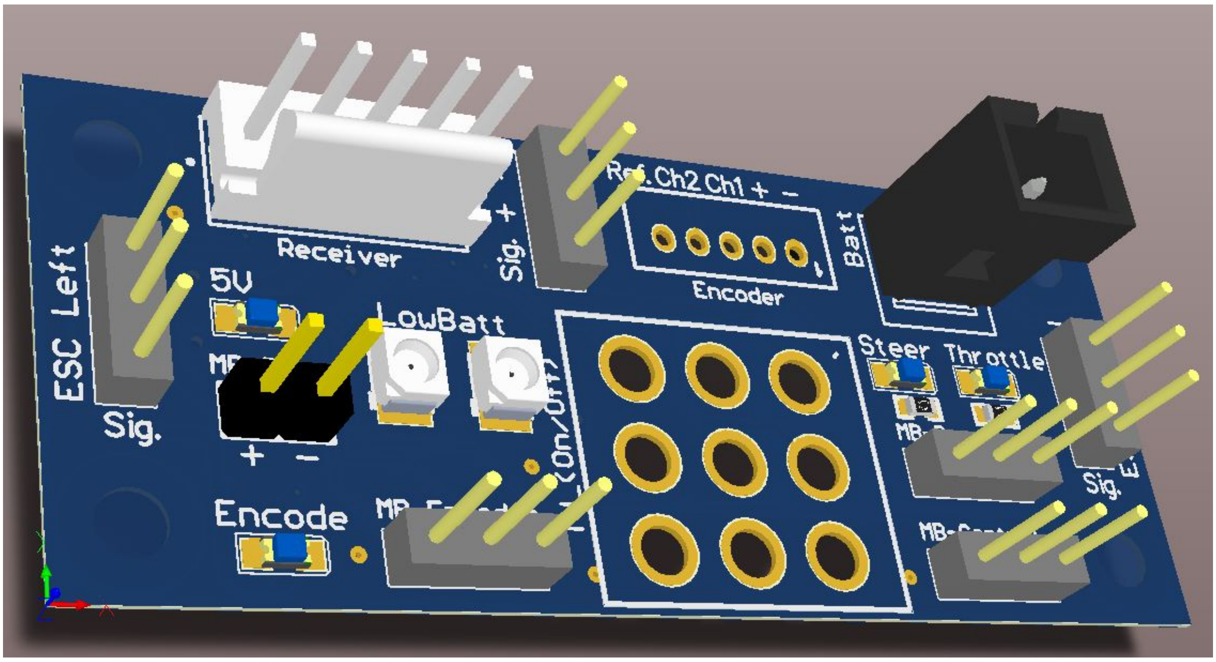

Figure 4.9: 3D view of the updated Interconnection board for the Mamba solution

4.3.3 VESC solution

The VESC solution already works properly with first version of the Mamba solution board. However, it does not use all features of the board. The interconnecting board of the VESC solution should be equipped with the CAN transceiver (mentioned in section 4.1), which is necessary for communication between Central board and VESC 4.12 ESCs. Also, I suggest to remove the switch and under voltage indication.

I would suggest to wait until the design of the electronics part of the SDP to be done. Afterward the interconnecting board can additionally fuse the Central board, the GPS and the odometry measurement board (see 4.11), as I mentioned in chapter 4. It would save a lot of necessary space.

4.4 Receiver and transmitter

A driver using a transmitter for handling a steer and throttle. The signals are passed wireless to the receiver. The receiver converts the signal and sends it as a PPM one channel signal or PWM signal split into many channels. The actual configuration operates with a PWM signal on each channel, see the figure 4.10. This type of the signal is standardized. Frequency 50 Hz and active pulse is from range 1 ms to 2 ms, where 1.5 ms means a middle position of the steer or null throttle. Consequently, 1 ms represents full back or full left turn and 2 ms otherwise.

Buying a good set is essential, as currently used set has a quantization problem. It is not precise enough, the step is clearly visible. Central board can receive more precise step as the set offers. Hence mind the quality of the set.

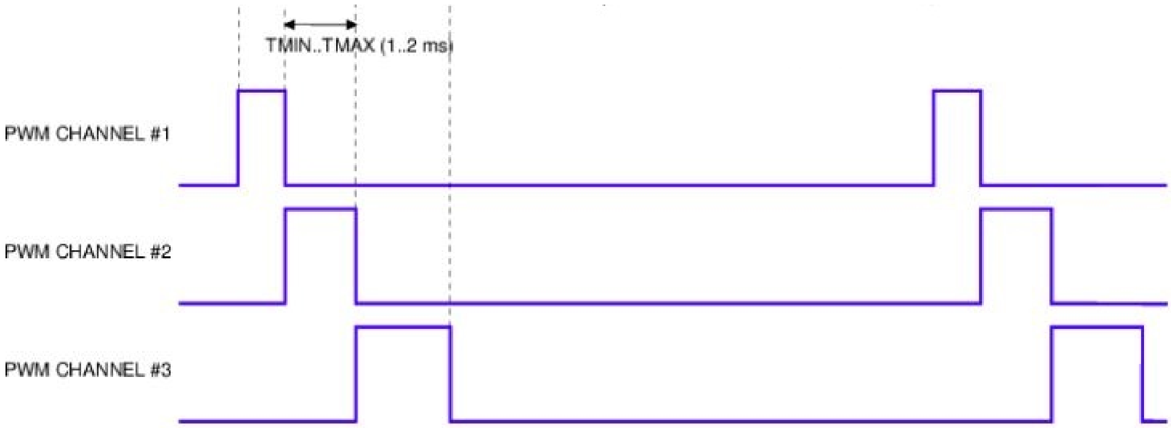

Figure 4.10: An obtained signal from the receiver. Adopted from [23].

4.5 GPS

My survey of the market shows similar results as the article [39]. Available GPS is with update rate from 1 to 10 Hz. Also, I should mention that solutions with higher update rate exist, but it is not so commonly available. For example Ublox SAM-M8Q [52]. I choose the solution C94-M8P kit [51], which offers two kits which can be configured to work together as base and rover to provide vehicle attitude outputs, such as the angle between north and heading, see chapter 29.3.3 of [49]. Then slip-angle of the vehicle according [3] can be calculated. For more information see a web page with datasheets [51].

The parameters are not so good as was mentioned in section 3.2.2, but it was the best what I could find for the respect to the SDP size, availability and accuracy. Dynamic heading accuracy is around $0.3\ ^\circ$ depends on conditions. Velocity accuracy is $0.18\ km \cdot h^{-1}$. Also, the dimensions of GPS-es are not well suited for the SDP.

4.6 Oddometry velocity sensor

First idea of contact velocity sensor was to buy an existing encoder which would be connected with a differential shaft, directly. First issue was the electrical and mechanical maximal RPM that sensor should have stand up. Second issue was its size. The smallest sensor which I found, was OMRON E6A2-CW3C, see [32]. Despite its small size, it was not small enough for SDP dimension. Its applying would leads to complicated mechanical solution.

Mechanically simpler solution was to use hall sensor with custom made magnetic gear or optical sensor with custom made optical gear. I decided for system with hall sensor. See figure 4.11.

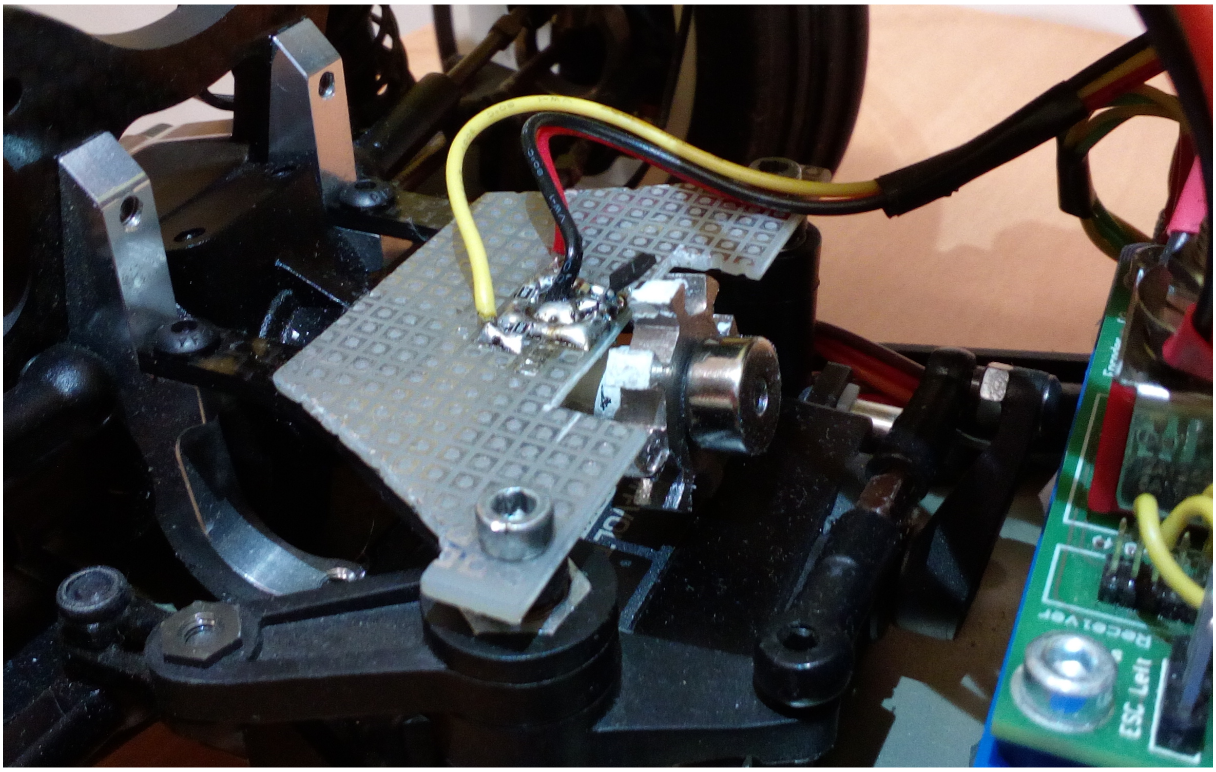

Figure 4.11: Detail of the contact velocity measurement system.

The system consists of magnetic gear which is linked with differential shaft. The magnetic field is generated by magnet which is situated on the end of the magnetic gear. Magnet is axial magnetized. Also the system has electronic part with unipolar hall probe (TLE4905L). Electric interconnection is realized according to datasheet [1]. Where $R_l = 1k5\ \Omega$ and $C_l = 100\ nF$. Power supply is drained from Interconnection board $5\ V$. Output Q is connected to the central board using resistor $1k5\ \Omega$.

Actual solution has 8 tooths on the gear. It is 16 pulses per revolution of magnetic gear. Multiplying the differential ratio (2.6) it is 41 pulses per front wheel revolution.

The magnetic gear has been made by external mechanical engineers. This custom solution still needs some adjustments. Tooths are not high enough. The solution has small differences between state 0 and state 1. The differential has some end plays between differential shaft and its bearing. Hall probe has to be very close to the magnetic ring. The probe is in the range of the travel of end play error. The hall probe can be damaged in higher speed.

To make bigger difference between logical states of the hall probe, the tooths have to be made higher. It has mechanical problem. I suggest that new design of magnetic gear should have less tooths but deeper. It will increase a reliability at the expense of resolution. Regarding the section 3.2.1 the lower level is 5 tooths per gear.

4.7 Bluetooth module

Two ideas of a data logging were considered. Using the SD-card and log all data onto the card. Then the data could be evaluated after the test session (offline). The SDP uses Bluetooth module HC-06. The idea of the using the Bluetooth was to receive the data which could be plotted on-line. Used module declared range is $10\ m$. The maximal supported baud rate of UART communication is $1382400\ bps$.

5 Software

Compatibility with Simulink keeps the goal to design a user-friendly platform. Majority of the control algorithms are programmed in Simulink environment, and programmers are accustomed to use it. When the Simulink was included in the system, some problems occurred. The ST Simulink libraries (ST MAT/Target) was not completely compatible with new HAL libraries. ST MAT/TARGET can be updated by the time by the manufacturer, but the situation required to program my own C drivers for the processor. All files implemented in the SDP are available at appendix A.

5.1 Overview

Following items are necessary before starting programing of the SDP algorithms, adopted from [40]:

1. Cube MX program from ST microelectronics [45]
    (a) HAL_libraries necessary for CUBE MX proper working [46]
2. Matlab from MathWorks [24]
    (a) Simulink extensions [25]
        i. Embeded coder extension [26]
        ii. ST MAT/TARGET libraries from ST Microelectronics [47]
3. IDE Toolchain one of following
    (a) EWARM or
    (b) MDK-ARM or
    (c) TrueSTUDIO [2] or
    (d) SW4STM32

How to set up the mentioned programs working together is described in the PDF which is included in the ST MAT/TARGET downloaded package, which is available at [47].

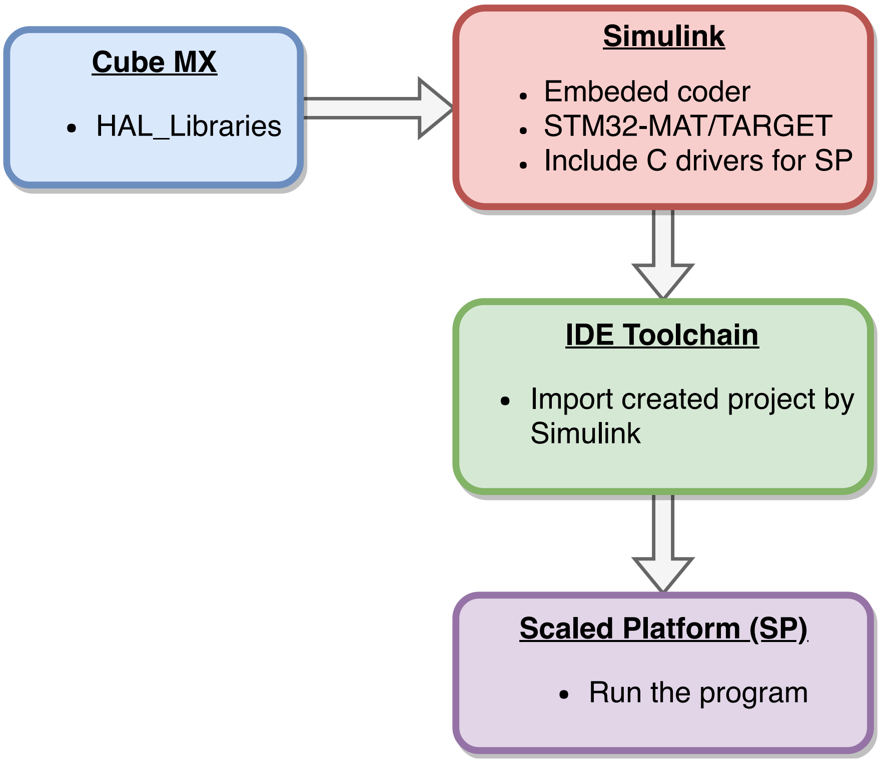

Figure 5.1: Software process overview block diagram.

Figure 5.1 shows needed steps to ensure uploading of the code to the target. Uploading of the code is realized in two steps. First, C code from Simulink is to generated. For the first time a generated project has to be imported into IDE, later this step is not necessary anymore. When the project with C files is imported, then the code can be compiled and uploaded to the target by IDE Toolchain. Necessary HAL libraries and other important steps are done by CUBE MX and Simulink cooperation. Those two simple steps are required for proper upload of the Simulink code on the STM CORTEX family processors.

For proper operation of the platform, the self-designed drivers had to be written. Simulink offers many ways how to include C files into Simulink code. I used calling of a Matlab function block. Afterward, this block calls the C code. I have been avoid using the calling style which I used. It is not a standard way how to call C codes which included others libraries. For purposes of calling the codes exists particular block in Simulink. I tried it, but I was not able to make running the Simulink code using mentioned special blocks. I suggest that in the future it will be necessary to migrate the C codes using mentioned special Simulink blocks. Actual solution is working, but it added one more step into an uploading process, for more information see section 5.3.

5.2 Simulink Template

All blocks reading data from the SDP are situated on the left side of the model. It is called Reading blocks in figure 5.2. They are data which the SDP provide. Data transmitting blocks, ensuring control of the SDP are located on the right side, called as the Writing blocks in figure 5.2. It helps keep the code clean and readable. Middle block called Main cycle is designated for Simulink operator or programmer. Initializing blocks are the blocks which initialize the CPU and its peripherals. Initializing sections is not fully implemented in Simulink, yet. Some C files also contain initializing statements.

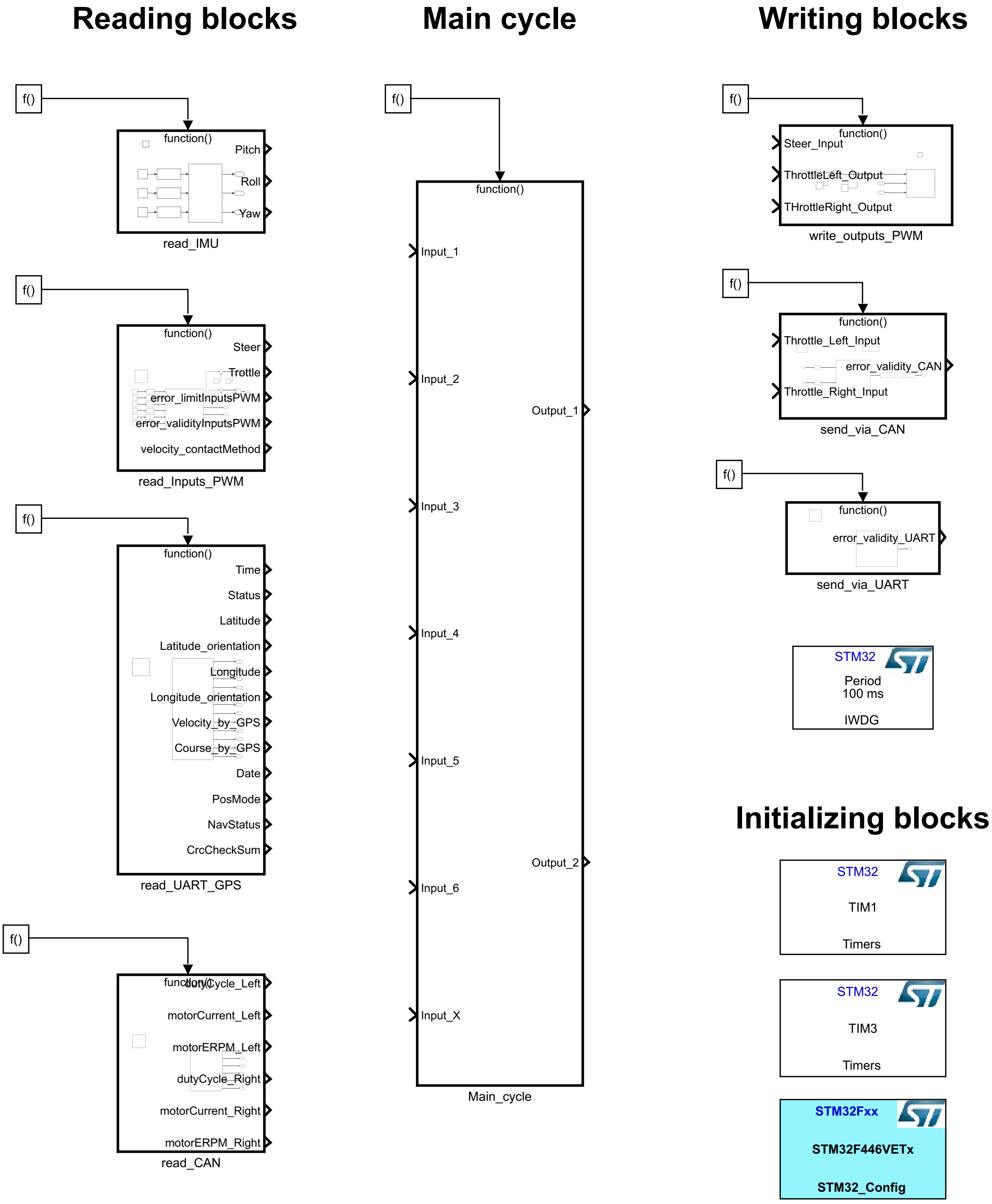

Figure 5.2: The Simulink template.

The structure is divided into several blocks. The operator/programmer can set up the execution period of each function by changing the value of the *function-call generator* block. The execution period determines how often the function (block) should be called. Also, each block represents access to a different peripheral of the processor. It is given by the structure of the HAL libraries. The callback functions which are used for reading data are common for one peripheral.

However some issues related to the HAL structure still exists. For example, those callbacks functions are triggered in every event which occurs on the periphery. The timer periphery measure values on two timers or channels. Recognizing the channels or timers has to be programmed into the mentioned callback. HAL libraries do not offer separated callbacks for those purposes. Hence I did not create a new Simulink block for handling another channel of the same periphery, separately. The block *read_inputs_PWM* belongs to the mentioned case. The timer periphery offers data from the receiver and hall sensor at the same time. It is a error in the structure. However, this case occurs only in one block. The error causes that a programmer loses control above the executing period of the one sensor. The operator can set up the executing frequency only for the certain block. Consequently, all sensors controlled by the block will have the same frequency. Receivers have update rate of 50 Hz. If the programmer would want to have a higher update rate of contact based velocity, he can raise the value according to the highest required update rate. However, the data from the receiver will have the same value in every two successive samples. It will increase computing time, but the processor is fast enough to handle even worse cases.

The speed of the Simulink model is limited to minimal step time of 1 μs (1 kHz). This limits maximal sampling rate of measurement. In case of the SDP, the accelerometer can provide data at 6.66 kHz. This problem can be partly fixed by programming a filter with a decimator, as C function code. I have rather kept all necessary filtering on the side of Matlab or Simulink. I think, that the structure should be kept in the level that C functions are only for hardware accessing and every signal processing should be done on Matlab or Simulink side. I have still used some filters for post-processed verification of the data which were programmed in C.

Conversion from raw data to proper unit or percentage are implemented into some blocks. Also, there might be settings of some constants which can be changed during the time or just some safety features. For example, I choose the block *read_inputsPWM* as a demonstration. It contains safety feature of bounds of the steer or throttle. See the figure 5.3.

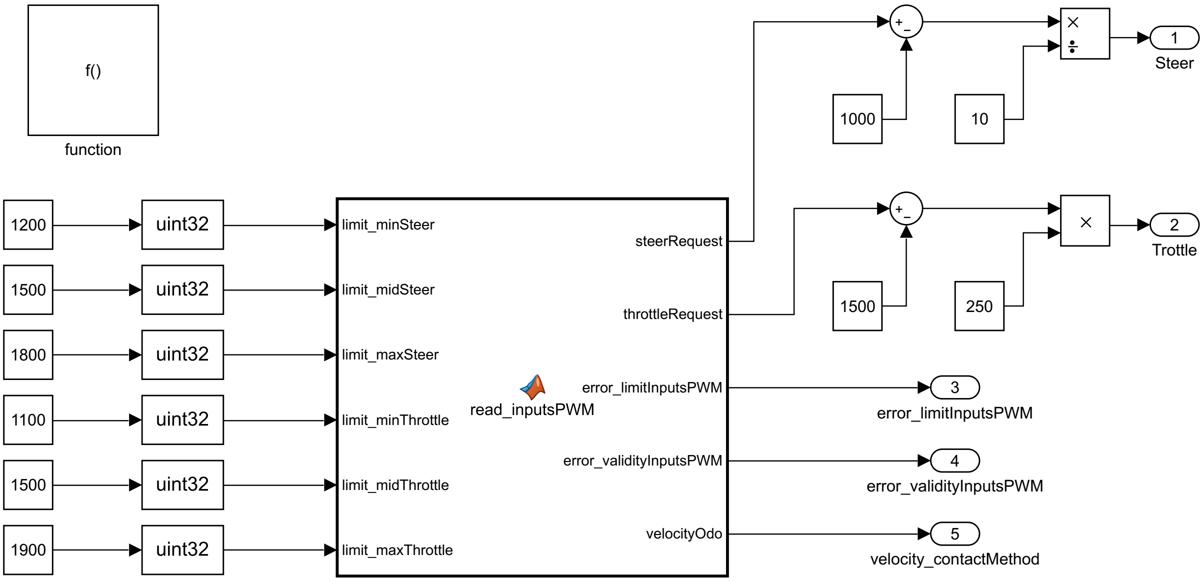

Figure 5.3: Settings of constants.

I decided that some code is better to program in Simulink and some code in Matlab. The logic of mentioned boundary safety feature has been programmed in Matlab. Code is available in appendix A

(Simulink template -> read_inputs_PWM -> read_inputsPWM).

Simple logic of the mentioned safety feature is implemented in the code. For me it was easier to program the safety logic in Matlab than in Simulink. I utilize the fact that I called the C files from Matlab function. On one hand it can raise a confusion, but on another hand, it can be proper utilization of the potential of the cooperation of (Matlab - Simulink - C programming).

By this section, I wanted to demonstrate how robust the template can be and how some issues can be solved using multi programming languages as (Simulink - Matlab - C).

5.3 Embedded C functions

The C functions need to be included in a folder of Simulink project. Then Simulink will create a C project structure and will copy the C files into proper folders. Finally, the whole project has to be imported by one of the mentioned IDE Toolchains. I created 6 C files. Every file corresponds to calling of one Matlab function, consequently the one Simulink block.

• Mezz_read_UART.c,	Mezz_read_UART.h
• Mezz_sent_via_UART.c,	Mezz_sent_via_UART.h
• Mezz_read_PWM.c,	Mezz_read_PWM.h
• Mezz_read_SPI.c,	Mezz_read_SPI.h
• Mezz_read_CAN.c,	Mezz_read_CAN.h
• Mezz_sent_via_CAN.c,	Mezz_sent_via_CAN.h

All those files could be found in the appendix.

5.3.1 UART - GPS

Data from the GPS are read via UART. Corresponding C file is *Mezz_read_UART.c*. GPS provides data as NMEA strings. Several formats of NMEA messages exist, but only $..RMC.. contains position, velocity and course values in the one message. The corresponding C code finds $GNRMC messages and parsing the values.

The code is designed in such a way, that the receiving callback function is triggered by every received letter. The Callback is just saving data to a ring buffer, while it does not receive a specific character which ends a line. It is called *endOfLine* character. Then it increments a value of the writing flag of the ring buffer and continues collecting data. The writing flag represents an actual index of the ring buffer where actually, the program writes. Parser accesses the ring buffer by a value of reading flag. Purpose of the reading flag is analogous as the writing flag. Protecting condition against reading while writing is implemented as a condition of reading flag value cannot be equal to the writing flag in any time.

The parser is accessing and looking for first 6 letters after character endOfLine. There is always character $̈änd remaining characters are a name of the message. Every message should end by specific characters *endOfLine*. If it finds a sequence of $GNRMC on first 6 places then starts parsing data and storing to variables. Then it increments reading flag and continues. If it does not find the sequence, it just increments reading flag and continues without parsing.

The parser may fail when UART bus transmits the various types of message protocols. The character $̈m̈ust be not always followed by the newline character. Therefore, I will suggest making it more robust for better reliability when some additional messages will occur on the bus. The actual configuration of the GPS is set up that the $GNRMC message is sent alone.

5.3.2 UART - Bluetooth

Bluetooth's data are sent by the same way as GPS via UART. The corresponding C file is *Mezz_sent_via_UART.c*. Bluetooth module HC-06 can send data to the board, but in the case of the SDP, it is not necessary. Data flows only from Central board to the Bluetooth module and afterward to any device which are able to read a Bluetooth. The Bluetooth module specification is available at [38].

The corresponding code sends data via UART at the baud rate of 921600. The protocol is fixed with starting and different ending delimiter. Data are transmitted according to the following protocol.

1. add_int16(delimiterStart);
2. add_int16(dutyCycle_Left);
3. add_int16(current_Left);
4. add_int32(erpm_Left);
5. add_int16(dutyCycle_Right);
6. add_int16(current_Right);
7. add_int32(erpm_Right);
8. add_int32(requestSteer);
9. add_int32(requestThrottle);
10. add_int16(roll);
11. add_int16(pitch);
12. add_int16(yaw);
13. add_int16(longitudal);
14. add_int16(latteral);
15. add_int16(vertical);
16. add_double(latitude_GPS);
17. add_double(longitude_GPS);
18. add_float(velocity_GPS);
19. add_float(course_GPS);
20. add_int32(throttleSent_Left);
21. add_int32(throttleSent_Right);
22. add_float(velocityOdometer);
23. add_int16(timeStamp);
24. add_int16(delimiterEnd);

Currently, the UART in the SDP is configured to transmits 8 bits of data. Hence, it was needed to program functions which would ensure deterministic sending of larger variables. Protocol sends variables with length of 2, 4 ,8 bytes. Entire sequence has 24 different variables in the ends and it is transmitted by 78 bytes together. I used non-blocking mode of sending a data. It means that processor is not pending while the entire word is sent. A time which the processor needs for transmitting a whole sequence, was 5 to 6 μs. Calculated time which would be necessary for transmitting entire sequence at the mentioned baud-rate will be $846\ \mu s$. When transmitting 10 bits per word. Sign is not important for the side of the sender. It is important when receiver wants to interpret values correctly. Data are sent via UART as Little-Endian.

I adjusted and used my program which was programmed in Matlab for receiving the data. The original program was created for purposes of my bachelor thesis [13]. It can plot data in online mode. It helps me to detect if the data are corrupted or how did they behave at the time, when car was driving. Data are transmitted via wireless Bluetooth connection. Consequently, the data could be saved according the protocol. In side of the collector program, a receiving is done as reading a Bluetooth port in the loop as fixed number of the bytes.

Such solution has following disadvantages. The Matlab loads the data as a matrix, hence it is very shift-prone. Sending time is significantly smaller than sending period. Hence, there exist a small likelihood that the reading sequence will not start in the time when data are sent. If this case occurs, then restart of the receiving stream routine is required. Even thought the protocol is robust against this case, because it contain starting and ending delimiter. Delimiters can be used for future parsing of the shifted data steams which was saved into a CSV, incorrectly.

5.3.3 TIMER - PWM

As was mentioned in the section 4.4 the receiver uses PWM for passing information. Information is captured by TIMER's input capture mode. Also, the hall sensor uses changing voltage levels as high and low, which is also captured by processors' timer. Corresponding C file is *Mezz_read_PWM.c*. Every timer can be configured independently. Timer for receiving steer request and throttle request has the smallest resolution of 1 μs. It will overflow every 0.05 s. This provides resolution of 1000 quantization steps per entire steering and throttle range. Timer for hall sensor has the smallest resolution of 100 μs, and it will overflow every 4 s during the data acquisition for first evaluating. Also, magnetic gear was not built precise enough. I choose to catch every 4th impulse to neglect non-homogeneity of the design. Consider a situation that the magnetic gear is manufactured, ideally. Then I suggest to use a timer with a resolution 10 μs and counter register a big as possible (65535).

$$\Delta wheel_{cur} = \frac{wheel_{cur} \cdot diffRatio}{numOfTooths} = \frac{0.267 \cdot 2.6}{16} = 0.0434 \tag{5.1}$$

Where $wheel_{cur}$ $[m]$ is circumference of the front wheel 0.267 m, $diffRatio$ $[-]$ is differential ratio 2.6, $numOfTooths$ $[-]$ is number of tooths at measuring magnetic ring which use hall sensor.

$$\frac{\Delta wheel_{cur}}{countRegValue \cdot minResOfTim} = \frac{0.0434}{65535 \cdot 10 \cdot 10^{-6}} = 0.066 \tag{5.2}$$

Where $\Delta wheel_{cur}$ $[m]$ is a length section which car would travel when one tooth will be detected by hall sensor 0.0434 m. $countRegValue$ $[-]$ is the value when counter register will be restarted 65535. $minResOfTim$ $[-]$ is the time when counter is incremented by 10.

The timer which senses the receiver's signal is noise sensitive. The real test shows that the Central board has been receiving a noisy signal from the receiver and then the central unit interprets the noise as a valid signal. The reaction of the Steer servo-mechanism and motor controllers were shown as visible short strikes of motors and steering. In the future dead-zones should be programmed dead-zones for received signals. Also, the transmitter has a quantization problem. The problem is described in section 4.4. This problem is not possible to solve by software correction without negative side effects, for example additional delay.

5.3.4 SPI - IMU

Data from the IMU are available via SPI. Corresponding C file is *Mezz_read_SPI.c*. Acquisition data is realized by checking the status register. The register shows, if the data are complete and ready for transmitting. If so, the data are read and stored as a variable. Reading is executed every predefined period from Simulink environment. I reused the reading section of the belonging code my bachelor thesis [13], with proper adjustments.

5.3.5 CAN - VESC motor controller

Corresponding C file is *Mezz_read_CAN.c*. The VESC measures many parameters where the important one have been described in section 4.2. Data from the VESC motor controllers are available via CAN bus. The VESC uses extended ID for identification of the CAN messages and commands. First LSB byte identifies ID of the device and rest is reserved for commands. For example: Identifier:

0xB07, has ID: 0x07 and Command: 0xB. Firmware version 3.40 has following command table of CAN_PACKED_ID.

- SET_DUTY = 0,
- SET_CURRENT = 1,
- SET_CURRENT_BRAKE = 2,
- SET_RPM = 3,
- SET_POS = 4,
- FILL_RX_BUFFER = 5,
- FILL_RX_BUFFER_LONG = 6,
- PROCESS_RX_BUFFER = 7,
- PROCESS_SHORT_BUFFER = 8,
- STATUS = 9,
- SET_CURRENT_REL = 10,
- SET_CURRENT_BRAKE_REL = 11,
- SET_CURRENT_HANDBRAKE = 12,
- SET_CURRENT_HANDBRK_REL= 13

The command table was adopted from [54] code "datatypes.h". The device ID is assigned by the VESC tool from Benjamin Vedder. For more information see [53]. How to handle particular commands is not described here. Due to lack of documentation the code reversing was necessary. However, it was not possible to convince the VESC sending the messages with measured data due to limited time. But those additional data are not crucial for advance control. The corresponding code is comm_can.c which can be find also at [55]. Data measured by the VESC can be accessed in two ways. First, using command CAN_PACKET_PROCESS_RX_BUFFER. The VESC will send the message containing requested data on demand. Also, the VESC can provide data periodically in the CAN_PACKET_STATUS message. This message is sent according to the predefined adjustable period of 100 Hz. CAN baud-rate is 500 $kbit \cdot s^{-1}$. The message contains actual values of (int16) phase motor current, (int32) electrical RPM and (int16)duty cycle. The current and the ERPM data are crucial for advanced control. For evaluation data, I applied the bi-quad filter with cut-off frequency of 20 Hz for the current, because of noise. The filter is only applied for evaluation purposes of the diploma thesis. For calculating its coefficients I used [9].

VESC allows being controlled by CAN bus. Using massage with ID 0x0.. for duty-cycle or 0x1.. for current or 0x3.. for ERPM. Sending and reading values via CAN bus are executed the same way as others functions of the Simulink template.

5.4 Safety features

Safety is a critical factor. The car is heavy and powerful enough that it can hurt someone's leg even break a bone. For the SDP's, safety is also important. Especially, when the SDP is under developing. Hence following safety features have been implemented.

Watch dog

Watchdog system is implemented in the processor which the SDP uses. The purpose of the independent watchdog is simply described as, if the code will be stocked in some point and it will not reach the state where it will release the watchdog flag until a predefined time, then the watchdog restarted the processor, adopted from [41]. I used it. The period of the watchdog I set at 100 ms.

Input limits

Input limits correction, mention in section 5.3.3, are implemented in C file is *Mezz_read_PWM.c*. If received signal from the receiver will excite the defined boundaries, then the error flag will raise. Even more, the received signal will be limited and saved as the maximal allowed value. It significantly helps not to overloads steer-servo mechanism when the servo tries to push the steer axle over its physical limits or invokes an unexpected behavior.

Lost of the receiver signal

In case that driver does not estimate transmitter's range correctly and the SDP would lose a signal, the *elapseTimeCalback* is used, which is triggered when the timer's counter register overflowed. In the case of the SDP, it means, when it will not receive a signal from the receiver for longer than $50\ ms$. Afterward, the value of the steer will automatically be set in the middle (cars is led straight), and values of the throttle are also set to the middle value (no torque). I suggest to improve it as following. If this situation happens then, the motor controller will break the wheels. The VESC supports this mode, too. Corresponding C file is *Mezz_read_PWM.c*

VESC limits

The VESC motor controller supports a safety feature. If it does not receive a control signal until custom defined period, it will apply a braking sequence. It may happen when the program stucks or wires of CAN bus will disconnect or cut-off. This feature is accessible from VESC tool application [53].

6 Tests and measurement

First series of the data were measured in indoor conditions. Acquired data show that GPS signal was not sufficient for the GPS to function properly. The GPS was not able to keep RTK state despite the data seems to be relevant. The data from the GPS were not usable. Second testing was realized outdoor at parking place at the Strahov, Prague. The GPS was able to stay in RTK 3D/DGNN mode after 10 minutes from starting the SDP. The mentioned mode is 3th most precise mode. For more information see [50]. Presented data are from second, outdoor, attempt of the measurement.

6.1 Position

The graphs with same title belongs to each other. It means that graphs which have the same title are created from the same set of data sharing the same relative plotting time. I decided to verify designed sensors by following maneuvers:

- Static position
- Acceleration without and with over-throttle
- Constant speed
- Running constant velocity in the circles
- Slalom

GPS data provide longitude and latitude position. For determination distance between two points in meters it is necessary a formula requires attitude, however it has not been logged. Hence I used the calculator [8] which does not need attitude. For more information about calculator see [8]. From the position in degrees has been subtracted digits which did not change their value. The SDP was tested in the range of 20 m. It meant that the degree of the latitude of the Prague ($50°$) did not change. It has been subtracted. Afterward, the data have been converted to meters. Finally, the graphs were plotted with this relative position. The axis show relative position in meters. For evaluation of the results, it is sufficient.

In the figure 6.1 it is shown that static position, during 30 seconds, can drift approximately 25 cm. It needs to be mentioned, that the GPS did not reach precision and accuracy as it could. "In moving baseline mode the delay in the Rover receiving valid message groups must be less than 700 mS. If this delay exceeds 3 s the GNSS will fall back into 3D fix mode preventing RTK operation." adopted from [50]. The moving baseline mode means that the Base module and the Rover module are placed on the platform. The GPS is running in RTK mode using UART communication at the highest update rate 4 Hz. It increases bus load, but otherwise it may decreases reliability. I had troubles with communications between the Base module and the Rover module via UART at the beginning, due to presence of wire-less/radio communication, which I fixed by removing radio antennas and switching off the radio transmitter as alternative connection. Despite this issue, the GPS module provides accurate enough data for advanced control algorithms.

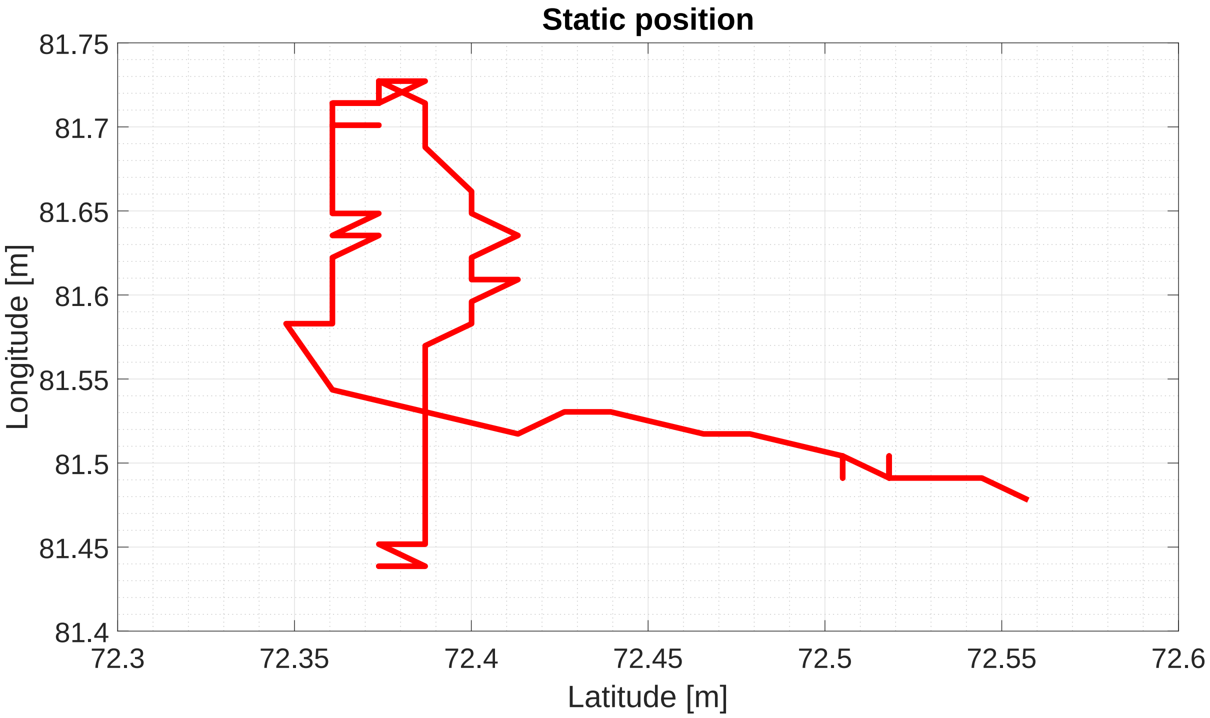

Figure 6.1: Graph of the SDP position during Static position

In the figures 6.2, 6.3, 6.4, 6.5 trajectory of the measured sets of data are plotted.

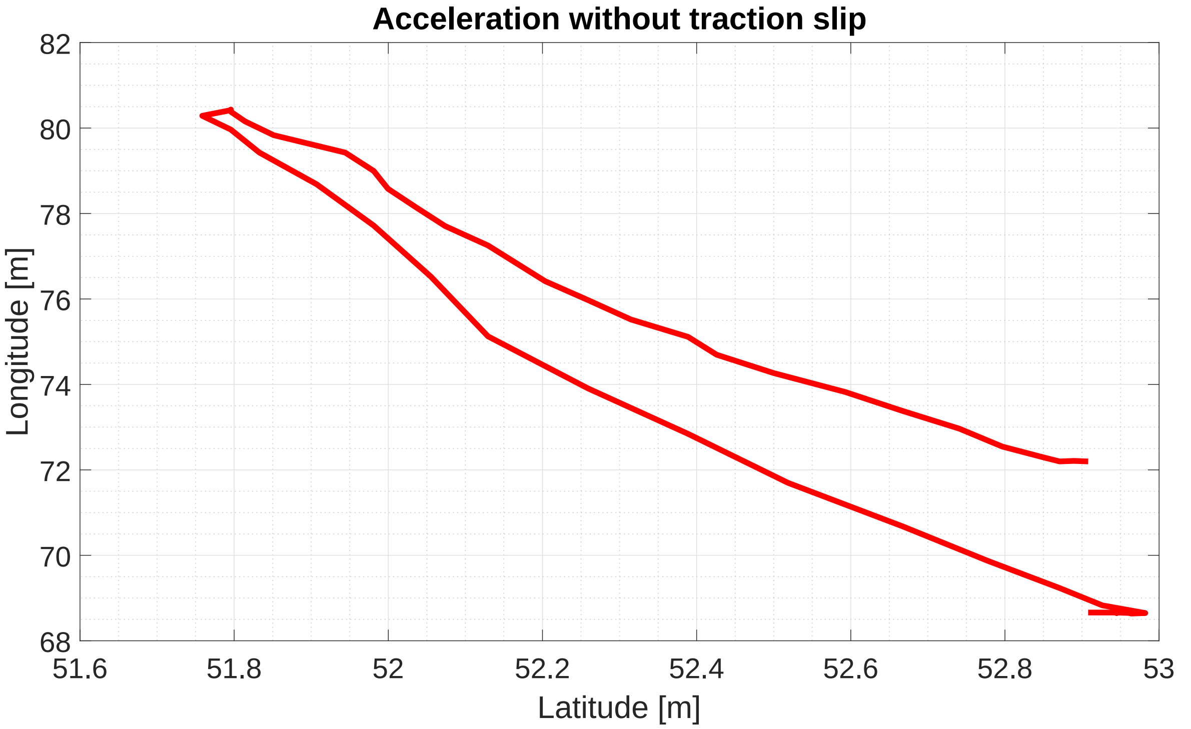

Figure 6.2: Graph of the SDP trajectory during acceleration without slip

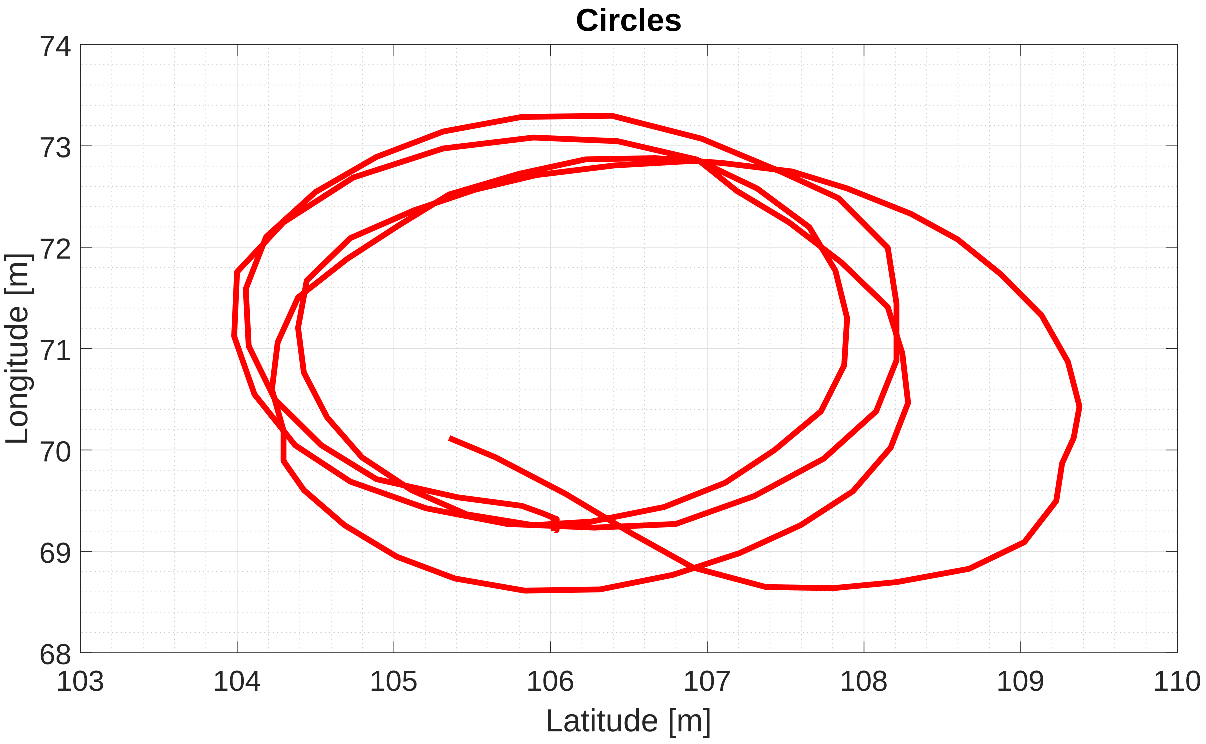

Figure 6.3: Graph of the SDP trajectory during circles

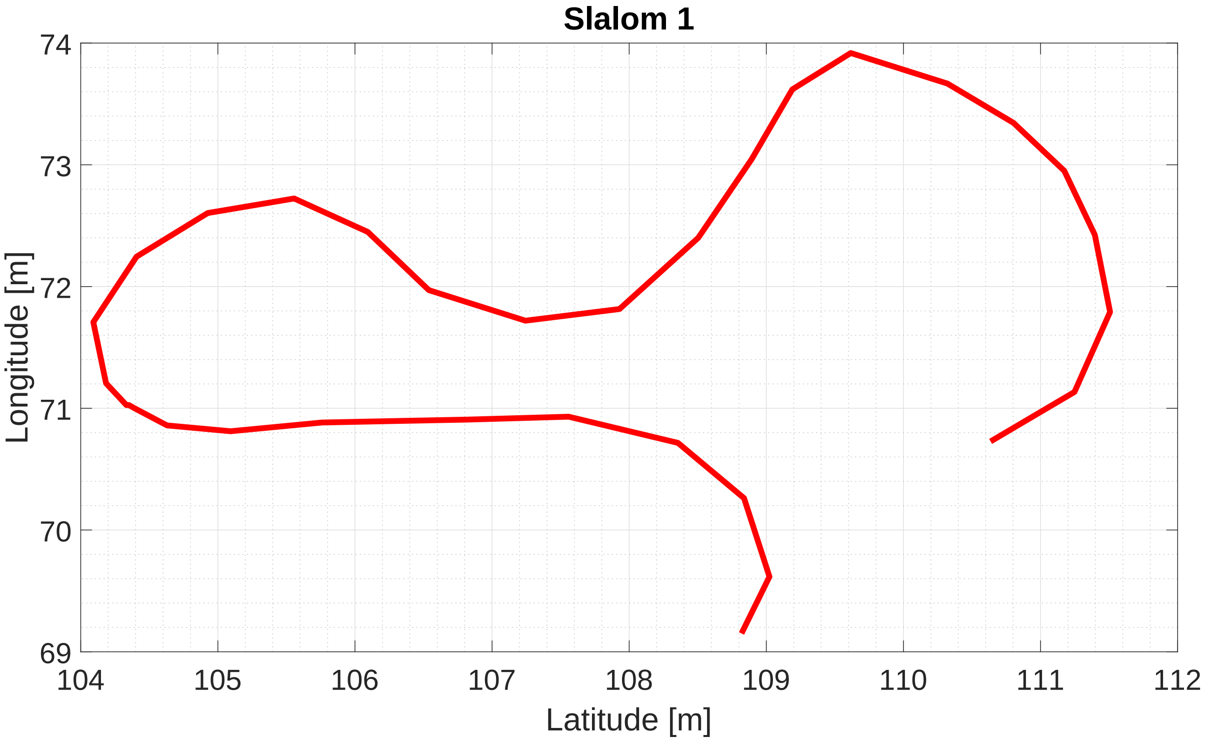

Figure 6.4: Graph of the SDP trajectory during slalom 1

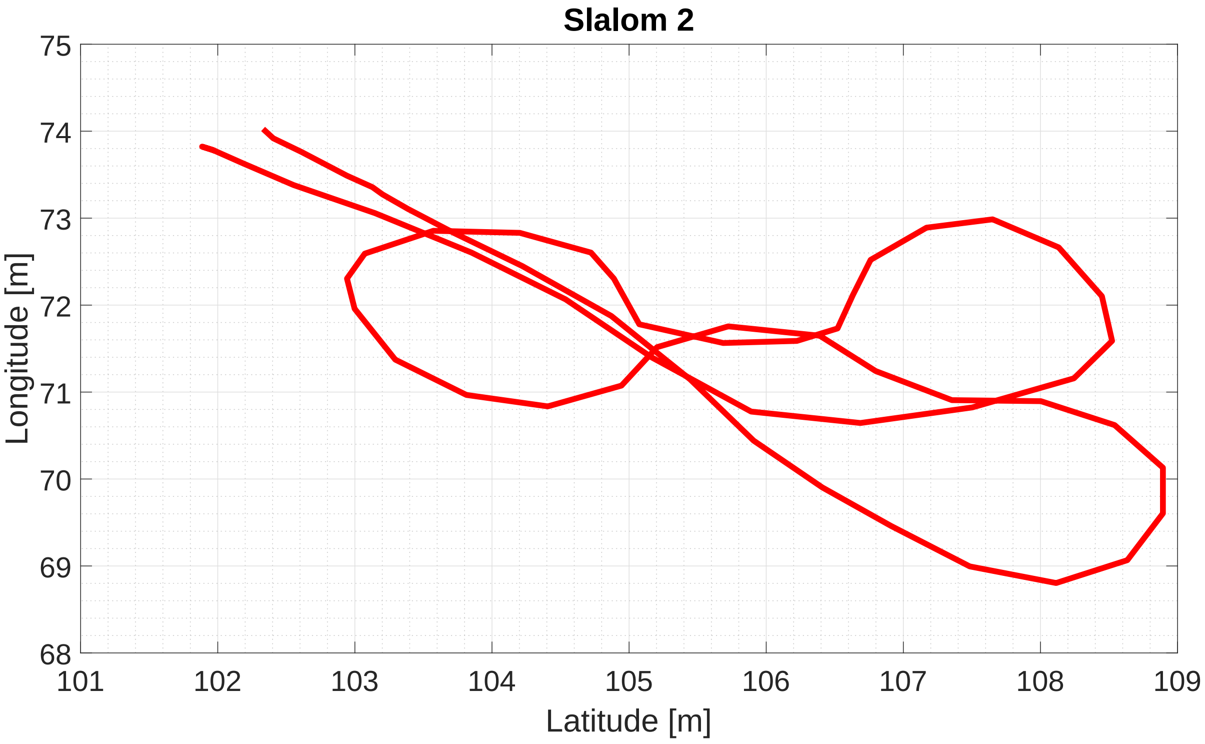

Figure 6.5: Graph of the SDP trajectory during slalom 2

6.2 Velocities

The SDP offer 3 measurement of the vehicle speed. The GPS, the contact method and the indirect method derived from the motor controller, see 4.2. In the figure 6.6 are plotted all mentioned methods.

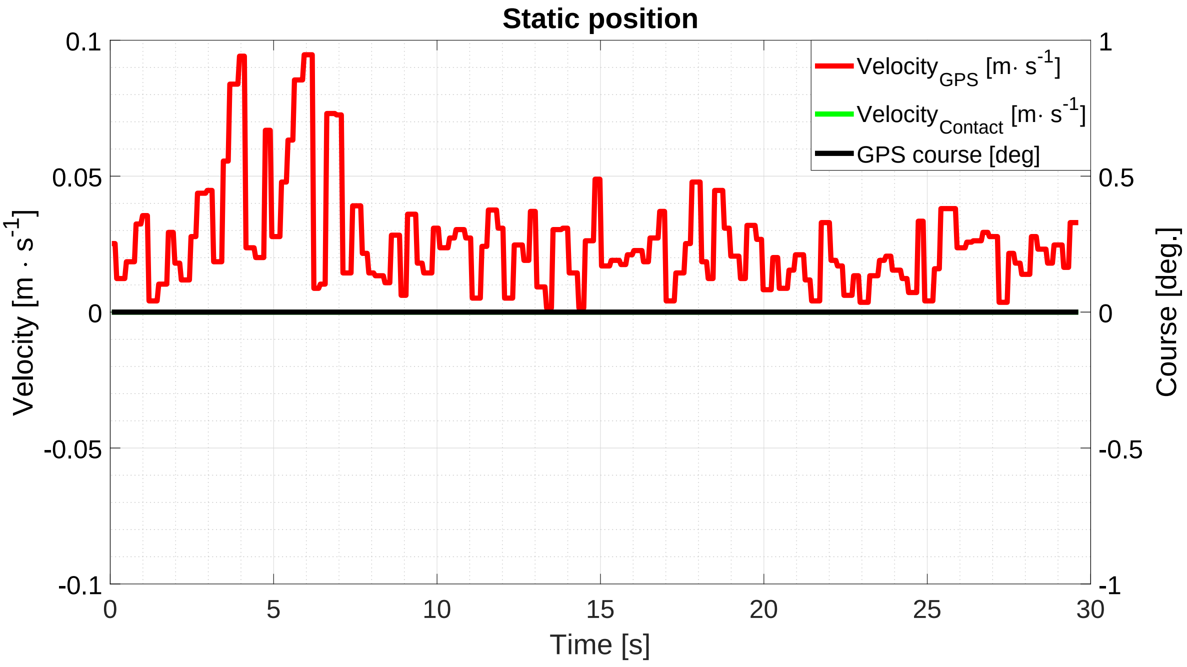

Figure 6.6: Graph of the GPS velocity, contact velocity and GPS course

When the SDP was stopped, all sensors show 0, except GPS. The GPS velocity signal is always corrupted by a noise. Even, when the GPS is in a static position relative to the ground. The figure shows that the noise level at 30 s data record is around $0.1\ m \cdot s^{-1}$. The figure 6.7 shows acceleration from 1^{st} second to 6^{th} second and then constant velocity for 5 seconds. It is visible that the GPS velocity is delayed approximately 500 ms. The ERPM based velocity is not so accurate as contact velocity. Also, this phenomena can be seen in figure 6.11. The ERPM based speed shows that the SDP went slower. I think that it is caused by different diameters of the front and rear wheels. The algorithm is considering the front wheels diameter only. Also in the same figure in 2.3 s, the ERPM velocity drops to the zero. This action I consider as the motor controller stops following the rotor field. It can be usual behavior of the VESC in BLDC mode. Incorrect or lack of a driven wheel's velocity value may cause some problems in controlling system reliability. I suggest focussing on this problem. I think this problem can be solved by FOC mode which was mentioned in section 4.2.

The measured data I have not calibrated before. See figure 6.7 section of constant velocity. All of the velocities values are around $1.5\ m \cdot s^{-1}$ in steady-state. It leads that the theoretical conversions considered in chapter 3 were correct.

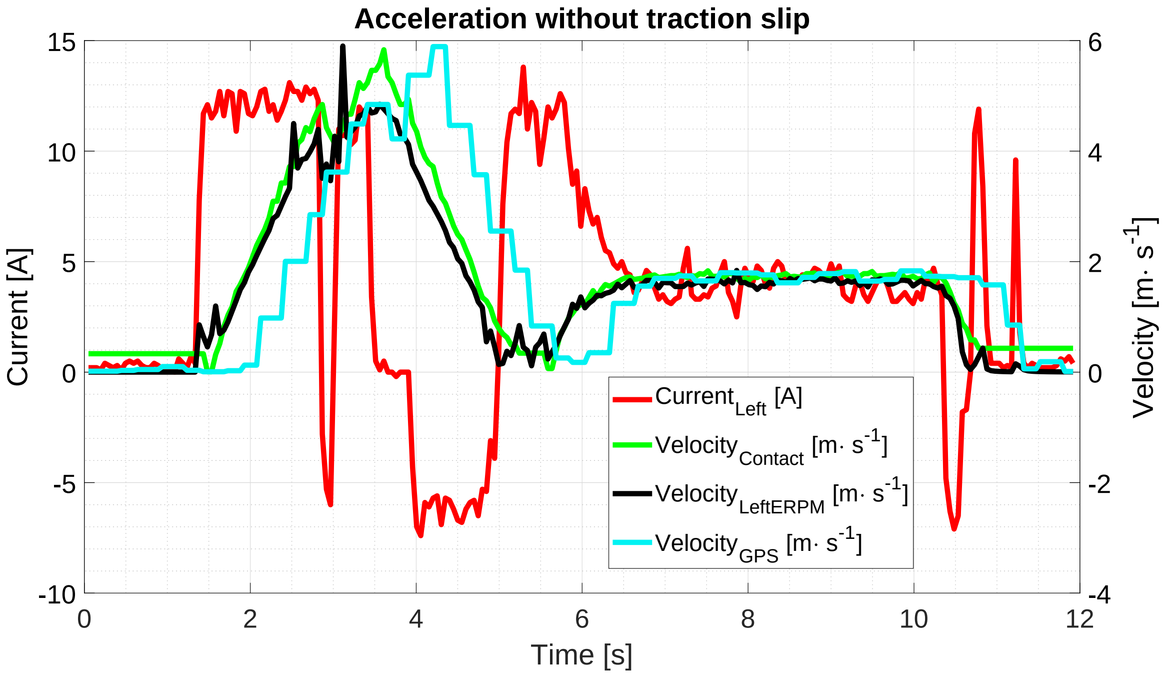

Figure 6.7: Graph of the current of the left motor, contact velocity, electrical RPM based velocity of the left motor and GPS velocity

6.3 Rates, G-forces and course

The figure 6.8 shows the steering angle, yaw rate, and lateral acceleration. The lateral acceleration is filtered by the FIR filter of 20^{th} order and then consequently shifted back by 20 samples. Sample frequency of the data is 20 Hz. The cut-off frequency is 1 Hz. Shifting the data is possible only when the data are post-processed. Related to the section 4.1 I consider that IMU has update rate high enough, that the filter would not shift the phase significantly. Remaining values are plotted as raw data. The figure shows the classical behavior of riding in circles. Lateral acceleration and yaw oscillation. This could be caused by bumpy roads or by tires elasticity.

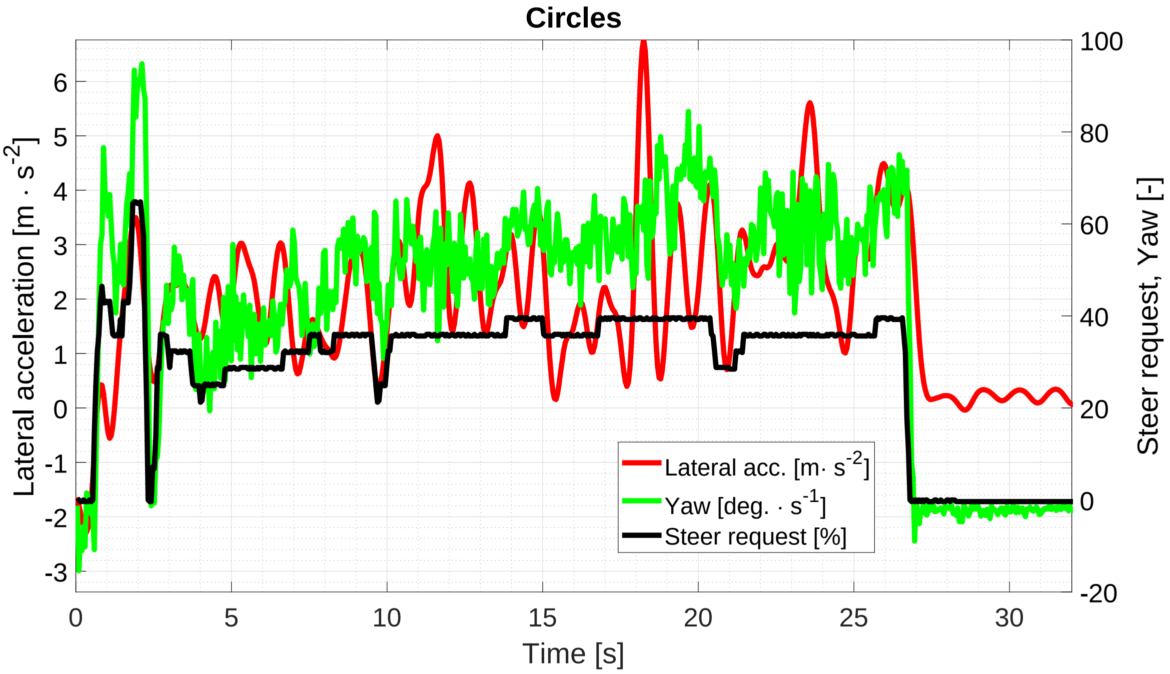

Figure 6.8: Graph of the lateral acceleration, Yaw rate and steer request

The phenomena of oscillating yaw and lateral acceleration is also visible in figure 6.9.

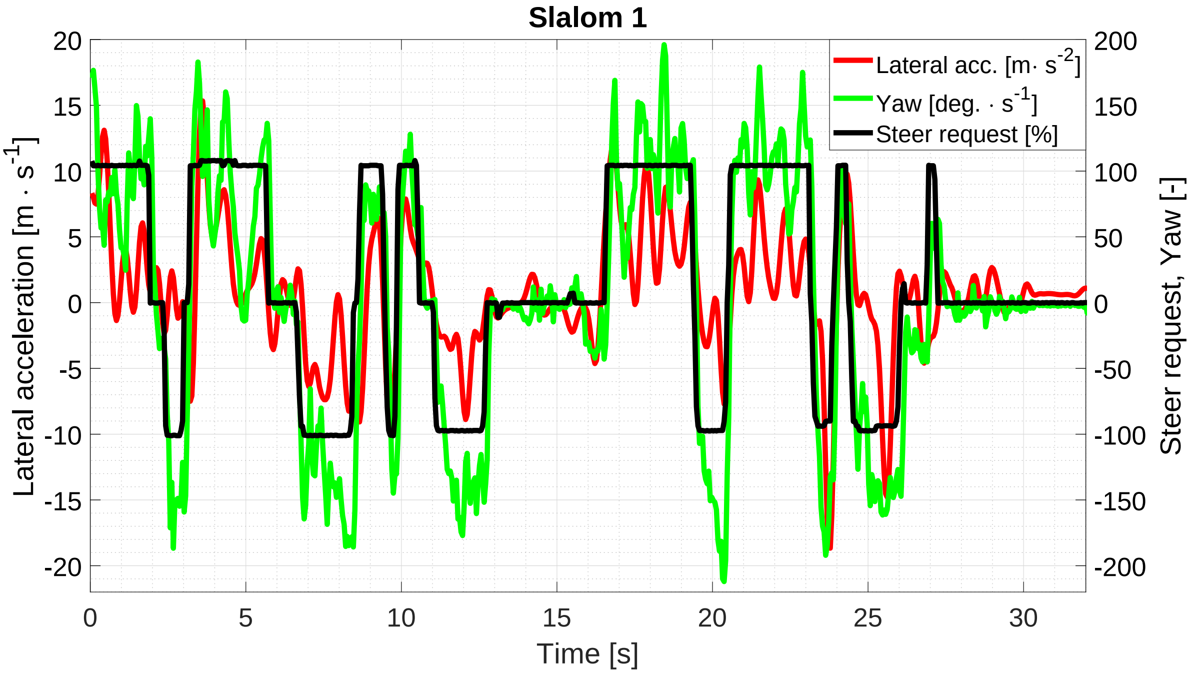

Figure 6.9: Graph of the lateral acceleration, Yaw rate and steer request

The figure 6.10 shows the course of the SDP related to the geographical North. The GPS had provided course data suitably when the GPS was able to see the necessary amount of satellites and to have enough time for initializing itself. The course response is delayed approximately $500 - 600\ ms$.

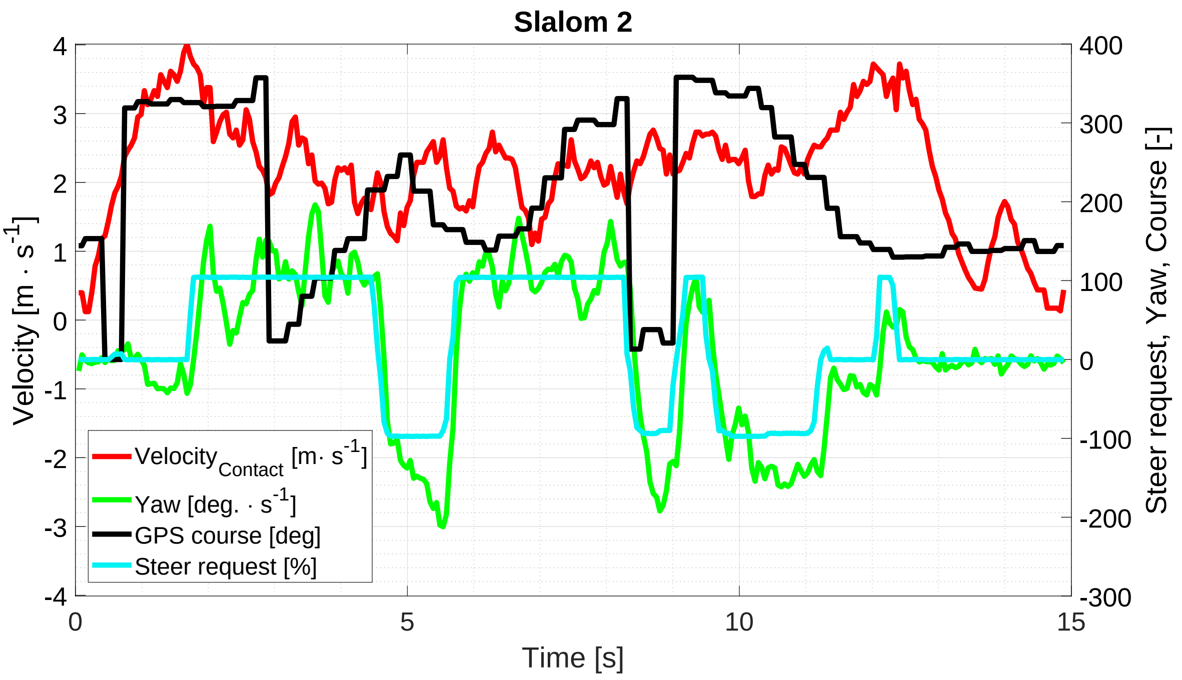

Figure 6.10: Graph of the contact velocity, yaw rate, GPS course and steer request

6.4 Torque

In the figure 6.11 is shown the relation between current and torque. From time $0.1\ s$ to $0.4\ s$ the driven rear left wheel slipped while front wheels not. In the same time, the current reached the highest values from all of the measured data set. The current after $0.4\ s$ did not reach torque amount required for slipping the wheel. Consequently, the wheel did not slip again.

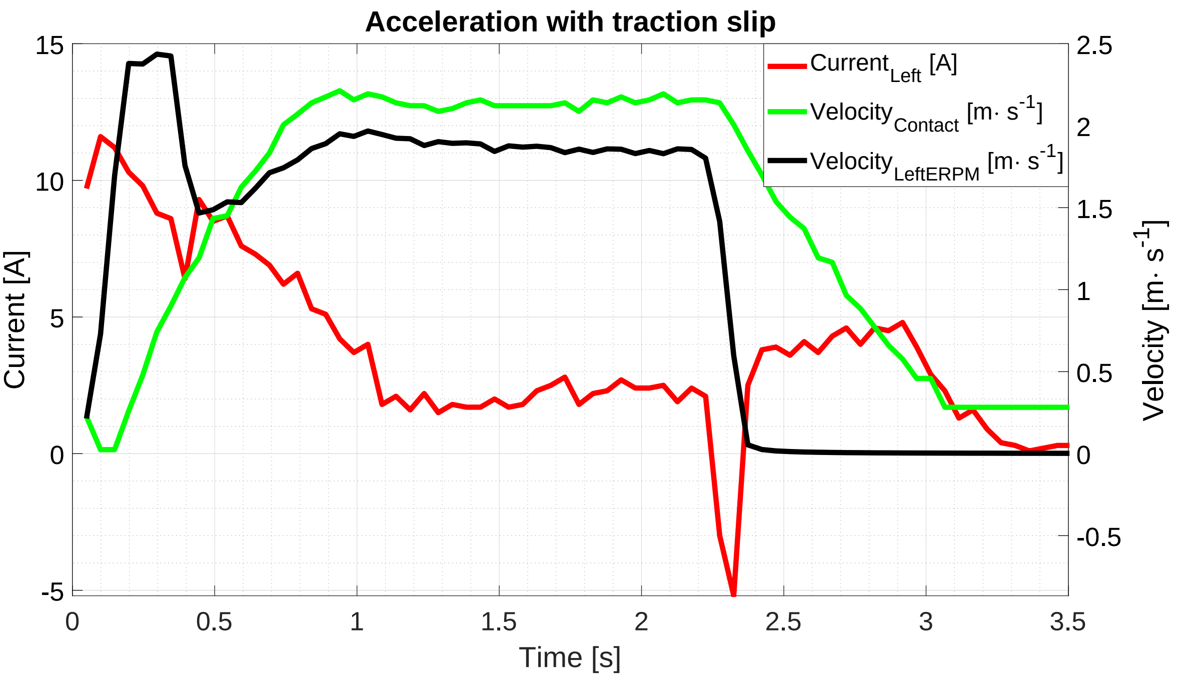

Figure 6.11: Graph of the Acceleration with slip

7 Future work

The SDP is still under development. Following are suggestions what can be improved in the future.

- To adjust and install a body castle for protecting the electronics against dust and dirt.
- To call C drivers by commonly uses Simulink blocks. See section 5.2.
- To fix the problem with Simulink program speed. See section 5.2.
- To design PCB which would merge functionalities already used devices. See chapter 4.
- To buy a more precise receiver and transmitter. See section 4.4.
- To test and improve contact velocity measurement. See section 4.6.
- Try to implement SGB dual antenna sensor. See section 4.5.
- To improve the robustness of the GPS parser. See section 5.3.1.
- To enhance wiring quality of the GPS. See section 6.1.
- To add the wireless real-time camera
- To program dead-zones. See section 5.3.

8 Conclusion

The goal of the thesis was to develop a scaled-down land vehicle platform which would be able to verify advanced control algorithms.

The Scaled-down platform was based on RC chassis (1:10). To reach the goal the bought chasis had to be mechanically adjusted. Also, it requires additionally implemented equipment as GPS, velocity contact sensor, advanced motor controllers, online telemetry module and inertial measurement unit. For these components software drivers in C coding language and Matlab has been developed. The platform was designed to support Simulink. For better usability, the Simulink template environment has been created.

At the beginning the investigation of an existing scaled vehicles chassis commonly available on the market was done. Based on the investigation, the RC buggy Serpent SDX4 Spyder has been bought. Then hardware and software requirements analysis of the sensors and actuators was done. based on the analysis, the hardware part of the platform has been built. Consequently, the software part has been programmed, as well. The functionality of the scaled-down platform has been verified by experiments and data which were logged by on-line telemetry.

The results indicates that the platform and test data are useful for advanced control algorithms validation. The scaled-down platform passed initial tests which approved chapter 6.

Chapter 7 summarized outcomes and possibilities for future work how to improve the SDP. Even though the platform is still under development, it offers a suitable base for testing and verifying the advanced algorithms.

Bibliography

[1] Infineon Technologies AG. Uni- and bipolar hall ic switches for magnetic field applications, 2007. URL http://www.produktinfo.conrad.com/datenblaetter/150000-174999/153775-da-01-en-HALL_SENSOR_TLE_4935L.pdf.

[2] Attolic. Truestudio, 2018. URL https://atollic.com/truestudio/.

[3] VBOX automotive. Slip angle explained, 2015. URL http://www.racelogic.co.uk/_downloads/vbox/Application_Notes/Slip%20Angle%20Explained.pdf.

[4] Pascal Cardinale, Camillo D'Angelo, and Massimo Conti. Traction control system for motorcycles. 2009. doi: 10.1155/2009/161373. URL https://www.researchgate.net/publication/26578103_Traction_Control_System_for_Motorcycles.

[5] CARmag. Automotive differential: Why is it important?, 2015. URL https://www.carmag.co.za/technical/technical/automotive-differential-why-is-it-important/.

[6] Chin-Long Cham and Zahurin Bin Samad. Brushless dc motor electromagnetic torque estimation with single-phase current sensing. 9(3), 2014. URL https://pdfs.semanticscholar.org/5055/9e60567769e1cf2f364a00c01cc75442d1c3.pdf.

[7] Castle Creations. Mamba micro x extreme 1:18 scale esc, 2016. URL http://www.castlecreations.com/en/mamba-micro-x-esc-010-0147-00.

[8] Inc. CSG, Computer Support Group and CSGNetwork.Com. Length of a degree of latitude and longitude calculator, 2011. URL http://www.csgnetwork.com/degreelenllavcalc.html.

[9] EarLevel Engineering. Biquad calculator v2, 2013. URL http://www.earlevel.com/main/2013/10/13/biquad-calculator-v2/.

[10] LLC. Firelands Group. Impact 12b, 2018. URL http://helion-rc.com/impakt12B/.

[11] Joseph Gierut and Ray Lohr. Automotive powertrain & chassis torque sensor technology, 2009. URL https://www.element14.com/community/servlet/JiveServlet/downloadBody/16410-102-1-50374/White%20Paper_SAW%20Torque%20Sensor%20Technology.pdf.

[12] Hackaday. Independent wheel drive r/c car, 2014. URL https://hackaday.com/2014/07/12/independent-wheel-drive-rc-car/.

[13] Lukas Hostacny. Nepriame meranie rýchlosti závodných vozidiel s pohonom všetkých kolies, 2016. URL https://dspace.cvut.cz/bitstream/handle/10467/64682/F3-BP-2016-Hostacny-Lukas-BP_HostacnyLukas_2016.pdf.

[14] CTU in Prague Faculty of Electrical Engineering. eforce fee prague formula, 2018. URL https://eforce.cvut.cz/en/.

[15] Microchip Technology Incorporated. Sensorless field oriented control (foc) for permanent magnet synchronous motors (pmsm), 2007. URL https://www.microchip.com/stellent/groups/SiteComm_sg/documents/Training_Tutorials/en532365.pdf.

[16] Honeywell International. Ways to measure the force acting on a rotating shaft, 2004. URL https://measurementsensors.honeywell.com/techresources/appnotes/Pages/Ways_to_Measure_the_Force_Acting_on_a_Rotating_Shaft.aspx.

[17] Kiumars Jalali, Thomas Uchida, Steve Lambert, and John McPhee. Development of an advanced torque vectoring control system for an electric vehicle with in-wheel motors using soft computing techniques. 2013. URL https://www.researchgate.net/publication/256422220_Development_of_an_Advanced_Torque_Vectoring_Control_System_for_an_Electric_Vehicle_with_In-Wheel_Motors_using_Soft_Computing_Techniques.

[18] Steven Keeping. Controlling sensorless, bldc motors via back emf, 2013. URL https://www.digikey.com/en/articles/techzone/2013/jun/controlling-sensorless-bldc-motors-via-back-emf.

[19] D. Lin, P. Zhou, and Z. J. Cendes. In-depth study of the torque constant for permanent magnet machines. 45:5383 – 5387, 2009. URL https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5326445.

[20] Mark Looney. Anticipating and managing critical noise sources in mems gyroscopes, 2015. URL https://www.analog.com/media/en/technical-documentation/tech-articles/Anticipating-and-Managing-Critical-Noise-Sources-in-MEMS-Gyros.pdf.

[21] Serpent LTD. Spyder sdx4 buggy 1/10 4wd (500020), 2016. URL http://oldmain.serpent.com/product/500020/.

[22] Madison E. Martin. Discrete digital filter design for microelectromechanical systems (mems) accelerometers and gyroscopes, 2010. URL https://fau.digital.flvc.org/islandora/object/fau%3A1408/datastream/OBJ/view/Discrete_digital_filter_design_for_microelectromechanical_systems__MEMS__accelerometers_and_gyroscopes.pdf.

[23] Matevz. Rc simulator interface – ppm signal decoding, 2016. URL https://blog.poscope.com/rc-simulator-interface-ppm-decoding/.

[24] MathWorks. Matlab, 2018. URL https://www.mathworks.com/help/matlab/index.html?s_tid=srchtitle.

[25] MathWorks. Simulink, 2018. URL https://uk.mathworks.com/products/simulink.html.

[26] MathWorks. Embedded coder, 2018. URL https://www.mathworks.com/products/embedded-coder.html.

[27] Microchip. Rn-220-xp lipoly battery, 2018. URL http://ww1.microchip.com/downloads/en/devicedoc/acc-rn-220xp-battery-ds.pdf.

[28] AXi models motors. Axi 28, 2018. URL https://www.modelmotors.cz/cs/product/detail/213/.

[29] MojeRC.cz. Lrp 3900 - shorty saddle p5 - 110c/55c - 7.4v lipo 1/10 competition car line, 2018. URL https://www.mojerc.cz/lrp-3900-shorty-saddle-p5-110c-55c-7-4v-lipo-1-10-competition\-car-line-hardcase_p127928.

[30] Kentaro NISHIKAWA, Toru TAKAHASHI, and Christophe DURET. Hub bearing with an integrated high-resolution rotation sensor. (81), 2013. URL https://www.ntnglobal.com/en/products/review/pdf/NTN_TR81_en_052_057p.pdf.

[31] Leonardo De Novellis, Aldo Sorniotti, Patrick Gruber, Leo Shead, Valentin Ivanov, and Kristian Hoepping. Torque vectoring for electric vehicles with individually controlled motors: State-of-the-art and future developments. 2012. URL https://www.researchgate.net/publication/260639058_Torque_Vectoring_for_Electric_Vehicles_with_Individually_Controlled_Motors_State-of-the-Art_and_Future_Developments.

[32] Omron. E6a2-c, 2016. URL https://assets.omron.eu/downloads/datasheet/en/q018_e6a2_incremental_rotary_encoder_25_mm_datasheet_en.pdf.

[33] Pratik Parsania and Ketan Saradava. Drive-by-wire systems in automobiles. 6, 2012. URL https://www.researchgate.net/publication/287993938_Drive-By-Wire_Systems_In_Automobiles.

[34] Oskar Persson and Gustav Persson. Torque sensor for automotive applications, 2015. URL https://www.iea.lth.se/publications/MS-Theses/Full%20document/5352_full_document.pdf.

[35] Domansky Profimodel.cz. Servo hbs 880bb, 2017. URL https://profimodel.cz/cs/digitalni/41431-servo-hbs-880bb-mg-hi-volt-brushless-tloustka\-20mm-4013389473859.html.

[36] RACELOGIC. *RACELOGIC TRACTION CONTROL*, 2018. URL https://www.racelogic.co.uk/_downloads/traction/Traction-Control-Manual-v3.2.pdf.

[37] K. S. Rama Rao, Nagadeven, and Soib Taib. Sensorless control of a bldc motor with back emf detection method using dspic. In *2008 IEEE 2nd International Power and Energy Conference*, 2008.

[38] Resi-Design. Bezdrátové moduly - bluetooth hc-06, 2017. URL http://www.sakul.cz/bluetooth-hc-06/n.

[39] J. RyuE.and J. RossetterJ. and C. Gerdes. Vehicle sideslip and roll parameter estimation using gps. 2002. URL https://pdfs.semanticscholar.org/4d85/3a868d7306f36dfa02e0bac59f5e69218fb2.pdf.

[40] STMicroelectronics. Code generation for stm32 mcus using matlab® and simulink®, 2014. URL https://www.st.com/content/ccc/resource/sales_and_marketing/presentation/product_presentation/a9/9b/32/0f/30/e7/4a/aa/stm32-matlab.pdf/files/stm32-matlab.pdf/jcr:content/translations/en.stm32-matlab.pdf.

[41] STMicroelectronics. Stm32 - independent watchdog (iwdg), 2016. URL https://www.st.com/content/ccc/resource/training/technical/product_training/d5/97/97/97/ef/b9/48/26/STM32L4_WDG_TIMERS_IWDG.pdf/files/STM32L4_WDG_TIMERS_IWDG.pdf/jcr:content/translations/en.STM32L4_WDG_TIMERS_IWDG.pdf.

[42] STMicroelectronics. Lsm6ds3, 2017. URL https://www.st.com/resource/en/datasheet/lsm6ds3.pdf.

[43] STMicroelectronics. B-f446e-96b01a, 2017. URL https://www.st.com/en/evaluation-tools/b-f446e-96b01a.html.

[44] STMicroelectronics. Stm32f446ve, 2017. URL https://www.st.com/en/microcontrollers/stm32f446ve.html.

[45] STMicroelectronics. Stm32cubemx, 2018. URL https://www.st.com/content/st_com/en/products/development-tools/software-development-tools/stm32-software-development-tools/stm32-configurators-and-code-generators/stm32cubemx.html.

[46] STMicroelectronics. Stm32cubef4, 2018. URL https://www.st.com/content/st_com/en/products/embedded-software/mcus-embedded-software/stm32-embedded-software/stm32cube-mcu-packages/stm32cubef4.html.

[47] STMicroelectronics. Stm32-mat/target, 2018. URL https://www.st.com/en/development-tools/stm32-mat-target.html.

[48] SBG Systems. Ellipse2-d, 2018. URL https://www.sbg-systems.com/products/ellipse-2-series/#ellipse2-d_rtk_gnss_ins.

[49] u Blox. u-blox 8 / u-blox m8 receiver description, 2018. URL https://www.u-blox.com/sites/default/files/products/documents/u-blox8-M8_ReceiverDescrProtSpec_%28UBX-13003221%29_Public.pdf.

[50] ublox. C94-m8p, 2017. URL https://www.u-blox.com/sites/default/files/C94-M8P-AppBoard_UserGuide_%28UBX-15031066%29.pdf.

[51] ublox. C94-m8p, 2017. URL https://www.u-blox.com/en/product/c94-m8p#tab-documentation-resources.

[52] ublox. Sam-m8p, 2017. URL https://www.u-blox.com/sites/default/files/SAM-M8Q_DataSheet_%28UBX-16012619%29.pdf.

[53] Benjamin Vedder. Vesc – open source esc, 2015. URL http://vedder.se/2015/01/vesc-open-source-esc/.

[54] Benjamin Vedder. Firmware vedderb fw 3.39, 2016. URL https://github.com/vedderb/bldc/blob/master/datatypes.h.

[55] Benjamin Vedder. The code for my custom bldc controller., 2018. URL https://github.com/vedderb/bldc.

[56] REINOUD F. WOLFFENBUTTEL and JENS A. FOERSTER. Noncontact capacitive torque sensor for use on a rotating axle. (6):1008–1013, 1990. doi: 10.1109/19.65816. URL https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=65816.

A Appendix

A.1 CD content

```
./
└─C files
│ └─Include
│ │ └─Mezz_read_CAN.h
│ │ └─Mezz_read_PWM.h
│ │ └─Mezz_read_SPI.h
│ │ └─Mezz_read_UART.h
│ │ └─Mezz_send_via_CAN.h
│ │ └─Mezz_send_via_UART.h
│ └─Source
│   └─Mezz_read_CAN.c
│   └─Mezz_read_PWM.c
│   └─Mezz_read_SPI.c
│   └─Mezz_read_UART.c
│   └─Mezz_send_via_CAN.c
│   └─Mezz_send_via_UART.c
└─CubeMX files
│ └─Mezzanine_Config.ioc
└─Simulink files
│ └─Mezzanine_SW_temp.slx
└─DP_Hostačný.pdf
```