

Master's Project work

**Design, implementation, and Evaluation
of an Automotive Experimental Platform
for Autonomous Cooperative Adaptive
Cruise Control (CACC)**

Ahmed Shaheen

Submission Deadline

Design, implementation, and Evaluation of an Automotive Experimental Platform for
Autonomous Cooperative Adaptive Cruise Control (CACC)

Ahmed Shaheen
Matriculation number 54926
Mechatronics M.Sc.

Hamburg University of Technology
Institute of Communication Networks
First examiner: Prof. Dr.-Ing. Timm-Giel
Second examiner: ?
Supervisor: Daniel plöger

Hamburg, Submission Deadline

Declaration of Originality

I hereby declare that the work in this thesis was composed and originated by myself and has not been submitted for another degree or diploma at any university or other institute of tertiary education.

I certify that all information sources and literature used are indicated in the text and a list of references is given in the bibliography.

Hamburg, Submission Deadline

Ahmed Shaheen

Abstract

Autonomous vehicle platooning is a safety critical application in which the hardware and software parts play major role, therefore, experimental effort and trials are required in order to prove which method suits than the other, which topology and communication schemes is more efficient and deterministic, which software algorithms are real time , which control parameters should be considered and which method to finally chose and verify!

Given all the above challenges, having an experimental platform to apply, and combine different methods is a must, all of this leads to identify the requirements of this platform, and it's features.

Contents

1. Introduction	2
1.1. Motivation	2
1.2. Goals	2
1.3. Research questions	3
1.4. Project plan	3
1.4.1. Work load	3
1.4.2. Timeline	3
2. Background	5
2.1. Autonomous platooning	5
2.2. Adaptive Cruise Control	6
2.3. Cooperative Adaptive Cruise Control	6
2.4. Control methods	6
2.5. Wireless Communication	7
2.5.1. Simple comparison	8
2.5.2. Frame structure	9
2.6. Platooning goals summary	10
3. Building the platform	11
3.1. Defining model requirements	11
3.2. Component selection	12
3.3. Development environment	25
3.4. Software Implementation	25
3.4.1. Communication protocol	25
3.4.2. Ultrasonic	26
3.4.3. Laser Sensor	27
3.4.4. Motor Driver	27
3.5. Software Architecture	28
3.5.1. Architecture	28
3.5.2. Task scheduling	29
3.6. Hardware schematic	29
3.7. Hardware implementation	30
3.8. Additional features	30
3.8.1. Remote control	30
3.8.2. System configuration ability	30
4. Platform summary	31

5. Future work and improvements	32
A. Appendix	33

List of ACRONYMS	
ACC	Adaptive Cruise Control
CACC	Cooperative Adaptive Cruise Control
MAC	Media Access Control
L2CAP	Logical link control & adaptation control
SMP	Security manager
ATT	Attribute protocol
GATT	Generic attribute protocol
SoC	System on Chip
CPI	Count Per Inch
IPS	Inch Per Second
PWM	Pulse Width Modulation

1. Introduction

1.1. Motivation

Every research topic serves in a specific area whether to solve a problem, or provide an optimized solution, the provided solution then passes through a set of verification and validation steps, as we can not just rely on assumptions and theoretical methods.

When talking about Autonomous platooning[2.1], the situation is rather complex, dealing with safety critical application, any solution needs to be proved based on real world experiments on a real platoon cars, however, to simplify the situation a little-bit, here we are working on a sub-validation step, which is building an experimental model.

To understand the challenge well, we need to understand the interdisciplinary nature of this model, starting from understanding mechanical concepts about vehicle movement, and vehicle dynamics, then exploring the sensors field, moving to wireless connectivity, and IoT, and finally the Embedded software part which serves as the point of integration to all these fields.

1.2. Goals

Before talking about goals, there is a mindset that I tried to preserve when I was progressing in building this platform, which is ***Generality***. I wanted to build a general purpose platform which allows the researcher to work and try different ideas without worrying about the capability of the underlying Hardware, with minimal cost and effort by trying to keep things simple without downgrading the efficiency of the design.

The goals are as follows:

1. Ability to communicate and pass the CAM message using any message passing method[2.1].
2. Have a deterministic communication behaviour.
3. Preserve CAM messages interval to be ≤ 100 ms.
4. Preserve Controller frequency to be ≤ 20 ms.
5. Have 3 different communication protocols (ZigBee & Wi-Fi, and BLE).
6. Real-time software implementation using FreeRTOS.

7. Fine tuning of the selected sensors, and decreasing sensors errors.
8. Data logging.
9. Remote control and monitoring.

1.3. Research questions

- What track and vehicles should be used as a Model?
- Which communication scheme should be implemented?
- What sensors should be used? how to verify them?
- How to log data and provide remote control over the platoon?
- Which controller is better in terms of cost and efficiency?
- What verification steps should be applied?

1.4. Project plan

1.4.1. Work load

Master's project work load for Mechatronics program is 12 CP which is equivalent to 360 working hours.

1.4.2. Timeline

The first version of the project timeline was as follows:

Please note that this timeline was proposed as full-time work load, however, it was agreed that the project is going to be considered as a part-time work load.

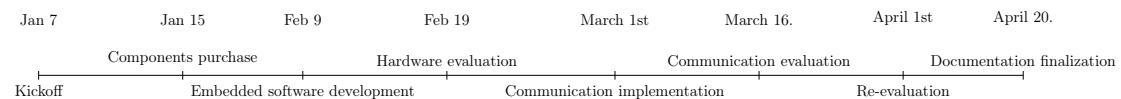


Figure 1.1.: Time Line

However, after discussing it during the Kick-off presentation and understanding the complexity of the system it was agreed to divide the timeline into major and optional parts, and relaxing the implementation a little-bit.

Actual timeline:

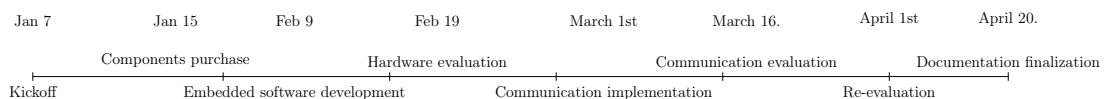


Figure 1.2.: Actual timeline

2. Background

As the project is mainly focused on implementation, here I am giving a brief introduction about each scientific term that is going to be used, or has been previously used. A specific description is going to be provided in the Implementation chapter[3] .

2.1. Autonomous platooning

The emergence of autonomous vehicles (AVs) has engendered innovative solutions for traffic congestion mitigation as well as the improvement of the passenger riding experience. The traveling public can expect level 5 full automation in more than 50% of vehicles by 2030[3]. Herein, AVs can be readily operated as platoons on the streets with minimum gaps between individual AVs, thereby resulting in a significant increase of road capacity and improving fuel economy[4]. On the other hand, by eliminating the driving tasks, vehicle occupants (drivers and passengers) can utilize on-board traveling time for activities such as reading, chatting or even working [??,??], which is expected to increase the productivity and enable other activities to be executed within a day[??].

In transportation, platooning or flocking is a method for driving a group of vehicles together. It is meant to increase the capacity of roads via an automated highway system. Platoons decrease the distances between cars or trucks using electronic, and possibly mechanical, coupling. This capability would allow many cars or trucks to accelerate or brake simultaneously. This system also allows for a closer headway between vehicles by eliminating reacting distance needed for human reaction[1].

Advanced driver-assistance systems and self-driving vehicle control are focus of ongoing development in transportation, for cars, for aircraft, or for mobile robots. Autonomous operating means of transport are developed with the intention of increasing safety, throughput, and comfort, with one part of the automation being autonomous platooning[8].

In autonomous platooning a leader is appointed, driving in front and deciding on the speed and acceleration. The other vehicles follow in short range, trying to maintain the distance. Exemplary fields of application are highway car and truck driving, warehouse automation, and aviation. This particular part of autonomous movement promises improvements in fuel saving, and air and road utilization[??].

2.2. Adaptive Cruise Control

Adaptive cruise control is a cruise control strategy for vehicles, where the vehicles maintain a constant distance from the preceding vehicles using the information received through the sensors, i.e ultrasonic distance sensor, radar or lidar, etc. However, the shortcoming is that the vehicles maintain relatively large inter-vehicle distance and can receive information only from the direct preceding neighbor using forward-looking sensors[16].

2.3. Cooperative Adaptive Cruise Control

In CACC, as depicted in 2.1, in addition to the information received through forward-looking sensors, the vehicle also receives information through the wireless communication link between the vehicles. The wireless link between the vehicles offers the possibility of information sharing between all vehicle of a platooning system and is known to achieve string stability [6]. This research work uses a variant of CACC designed by D.Ploeg, where the platoon members receive information from the preceding vehicles only, using the wireless communication link [6]. In Figure 2.2, d_{i+1} is the distance of vehicle $i+1$ with respect to the vehicle i also, v_i is the velocity of vehicle[16].

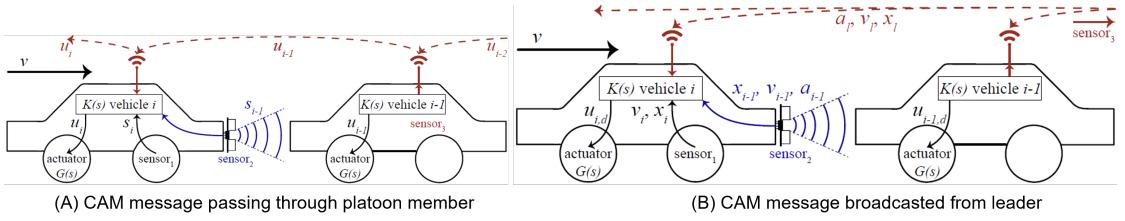


Figure 2.1.: CAM messages passing in CACC could be either A, B, or Combined

2.4. Control methods

When talking about control of robotic vehicle platoons, string stability of nonlinear interconnected systems is the main objective of longitudinal controller design. It makes statements about spacing error amplification of a vehicle platoon[17].

String stability of a vehicle platoon is the primary performance parameter. Intuitively, string stability of a vehicle platoon ensures that the inter-vehicular spacing errors of all the vehicles are bounded uniformly in time provided the initial spacing errors of all the vehicles are bounded. In this dissertation, we design various decentralized control algorithms and characterize their performance in terms of the minimum attenuation of the maximum spacing errors that can be guaranteed from vehicle to vehicle in the platoon. Parametric uncertainties degrade the platoon performance. In order to improve

the robustness of a string stable control algorithm. a direct adaptive control algorithm that guarantees improved performance is designed. The concept of string stability is extended to general nonlinear dynamical systems. we derive sufficient conditions for ensuring stability for a countably infinite interconnection of exponentially stable nonlinear systems. We also show that under the same conditions, string stability is preserved for structural and singular perturbations. Then. we present a decentralized adaptive controller to improve the robustness in the presence of parametric uncertainties for the same class of systems[10].

Theorem 2.1. Definition: String stability of interconnected vehicles is given when the spacing errors

$$\varepsilon_i(t)$$

of vehicles $2 \leq i \leq m$ attenuate along the vehicle platoon:

$$\|\varepsilon_2(t)\|_\infty > \|\varepsilon_3(t)\|_\infty > \dots > \|\varepsilon_m(t)\|_\infty$$

2.5. Wireless Communication

To achieve a stable cooperation between platoon vehicles, wireless communication protocol must be carefully chosen in order to have deterministic communication behavior, minimal packet loss, suitable data rate, and minimal communication delay.

Nowadays, there are several options such as Wi-Fi, Bluetooth, BLE, and ZigBee. Each protocol of them has different parameters and characteristics, to get some insights about the differences between them, here I am presenting a quick comparison, however, in the platform, it has been designed to acquire the 3 of them, to enable further research and investigations in the future, in communication in general not only for CACC application.

2.5.1. Simple comparison

Parameter	WiFi	ZigBee	BLE
Frequency band(GHz)	2.4 & 5	2.4	2.4
IEEE standard	802.11 a b g	802.15.4	None
Max data rate	High	Low	Moderate
Energy consumption	High	Low	Moderate
Communication layers	Physical Data Link Network Transport Session Presentation Application	physical (PHY) MAC "IEEE 802.15.4 radio standard" network (NWK) application support (APS) application framework (AF). ZigBee Device Object (ZDO)	physical (PHY) Link layer L2CAP SMP GAP GATT ATT Application Layer

Layers illustration

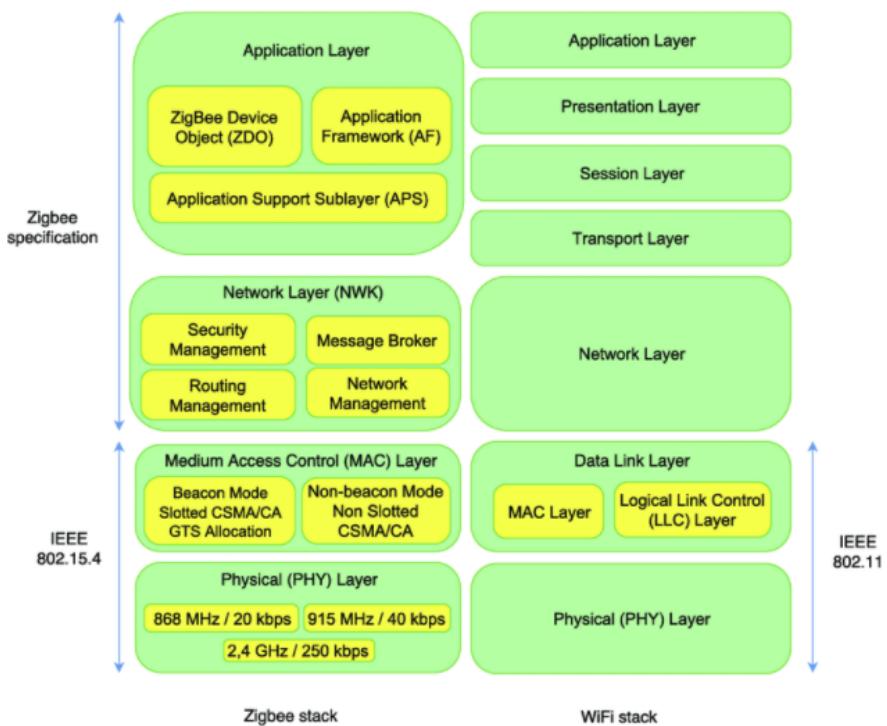


Figure 2.2.: ZigBee and WiFi stack [11]

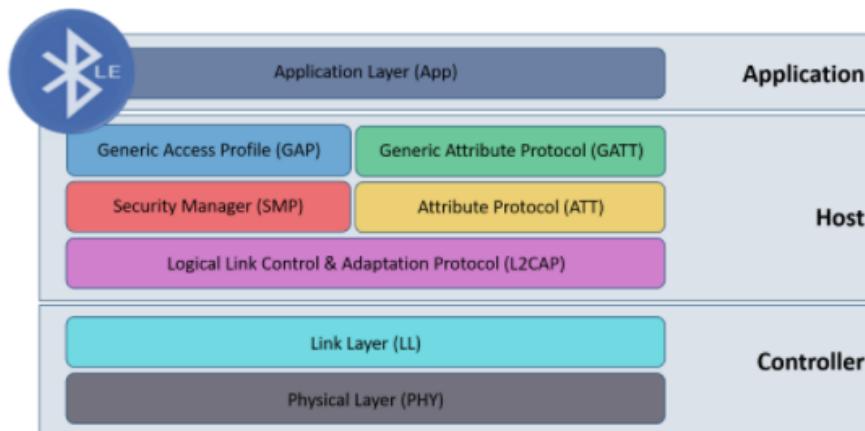


Figure 2.3.: BLE stack[12]

2.5.2. Frame structure

1.Wi-Fi

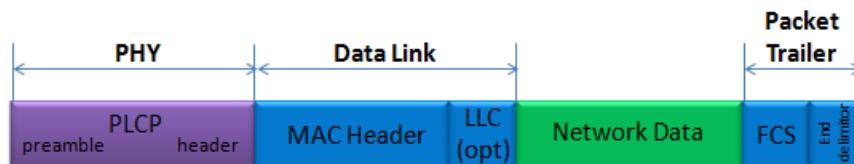


Figure 2.4.: WiFi Frame structure[13]

2.ZigBee

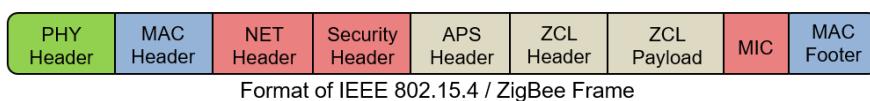


Figure 2.5.: ZigBee frame structure[14]

3.BLE

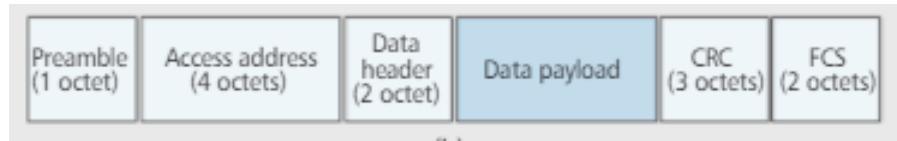


Figure 2.6.: BLE frame structure [15]

2.6. Platooning goals summary

- The ability of trucks to safely drive autonomously in a platoon incorporating Impressive gap management
- The ability of truck platoons to operate safely and accurately even in city environments due to the incorporation of Cohda's unique V2X-Locate technology which enables accurate positioning even in environments where GNSS doesn't work well.
- The provision of an extended perception horizon for each truck in the platoon due to the trucks being connected to each other through Cohda's connected vehicle technology and not just 'aware' of each other's physical presence through sensor technology.
- The ability for trucks to join and leave an active platoon without disrupting the system.
- The ability of the platooning system to detect and accommodate other vehicles without compromising the system.
- The ability for trucks to operate efficiently (smooth acceleration and moderate braking) when connected in the platoon.
- The ability for trucks to brake in unison (cooperative braking) as opposed to a daisy-chain system due to all vehicles being connected to each other. For example, if a truck that is leading a platoon brakes, the other trucks in the platoon receive instant notification and will also brake almost instantaneously, as opposed to reacting to the vehicle in front of it [10].

3. Building the platform

In the upcoming sections, I will be describing the journey of building this platform in accordance with the timeline presented in [].

3.1. Defining model requirements

Until the 7th of January which is the kick-off presentation date, I was collecting data and trying to understand the challenge, so that I can define the needs and requirements for such model, they were defined as follows:

1. A suitable Micro-controller should be selected.
2. Car model and track should have a suitable length, and size.
3. The model should have wireless connectivity for WiFi, BLE, and ZigBee.
4. A set of sensors need to be carefully selected, and they should have a suitable refresh rate, precision, and accuracy.
5. Software design in cooperation with the selected MCU and sensors should allow a 20 ms interval for dynamics update and 100 ms for CAM messages update.
6. Motor driver should be used in order to have control over the motors.
7. Suitable Power supply.
8. Voltage level shifter
9. Create remote control procedure for both ends (Model and user end)
10. Embedded software code should be designed to allow easy test and debugging.
11. Code execution protection techniques shall be used to prevent sensors errors.
12. Code development environment should be flexible and modularized to allow future development and usage.

3.2. Component selection

In this section I will be describing the selection criteria for each and every component that has been used.

Car model and track

Car model and track are the core Hardware for such model, the track need to have a suitable length, and shape, which the car should have a compact size yet suitable for containing all components, the track should also allow easy car movement and minimize the overhead coming from controller the car to follow certain trajectory.

Based on the knowledge provided in [2, the previous model track was just a simple printed circle and vehicles was just following it, although the implementation for such track is easy, however, huge software overhead to design a line following strategy, so the decision was to use a slotted track and slot car.

Valid options were the following:

1. Scalextric



Figure 3.1.: Scalextric (1:32) Car



Figure 3.2.: Scalextric standard tracks

2. Carrera



Figure 3.3.: Carrera (1:24) Car

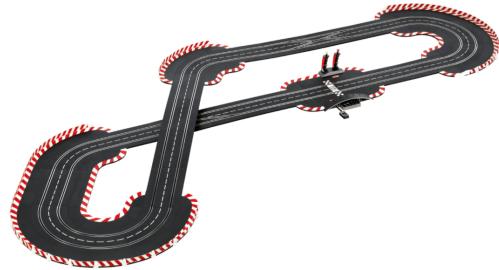


Figure 3.4.: Carrera 1:24 standard tracks

Carrera had better size options and less delivery time so the decision was to use it, and the following Car model and track were ordered at the end, customized track design was made using Carrera's track designer software "Autorennbahnplaner".



Figure 3.5.: BMW M1 Procar "No.5", Hockenheim 1979

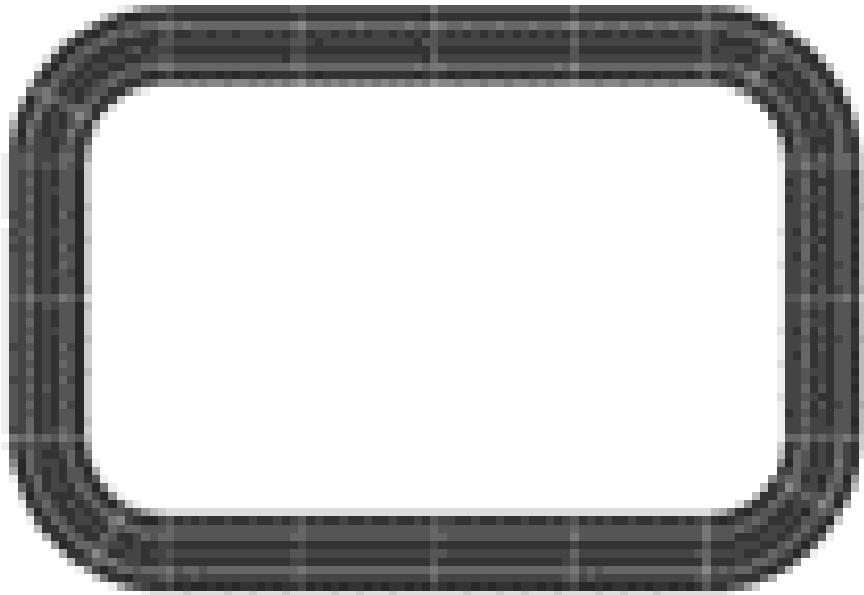


Figure 3.6.: Max length 6.3, with curvature length of 54 cm, according to Carrera track planner software, all gaps and clearances are within the safe limits

Microcontroller

For microcontroller selection, I was trying to achieve the following requirements:

1. High clock frequency.
2. Have several wireless connectivity peripherals as SoC.
3. Has suitable memory capacity.
4. API supports FreeRTOS and multi-thread execution.
5. Has several serial communication peripherals (SPI, I2C, and UART).
6. Available on Arduino Framework as well as having a standalone Framework
7. Cost and energy efficient.
8. small in size.

There were a plenty of options to choose from, popular ones are STM, NXP, Espressif, and Jetson nano kit was proposed. The decision came to choose ESP32, and a small comparison was made between ESP32 and Jetson nano kit.

Controllers	ESP32	Jetson Nano Kit
CPU	low-power Xtensa® 32-bit LX6 microprocessor Dual core	Quad-core ARM A57
CPU Clock	240 MHz	1.43 GHz
Flash memory size	External: Up to 16 MB of external flash can be mapped into CPU instruction Internal: 448 KB of ROM, 520 KB of on-chip SRAM for data and instructions	4 GB 64-bit LPDDR4 25.6 G
Peripherals	SD card, UART, SPI SDIO, I2C, LED PWM Motor PWM, I2S, IR pulse counter, GPIO capacitive touch sensor, ADC, DAC	GPIO, I2C, I2S SPI, UART USB: 4x USB 3.0, USB 2.0 Micro Display: HDMI 2.0 and eDP
GPIO	32	40
Price	51 €	94
Wireless connectivity	BLE & Wi-Fi	None

The selected MCU:

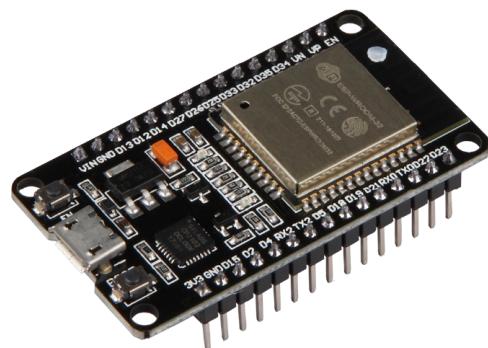


Figure 3.7.: NodeMCU ESP32 based

Selecting ZigBee Module

To complete the full wireless communication set, given that the ESP32 has Wi-Fi and BLE, the only remaining issue was to select a ZigBee module.

The selected model was the Core2530 provided from waveshare and here are some of the module features which were the reason for selecting it.

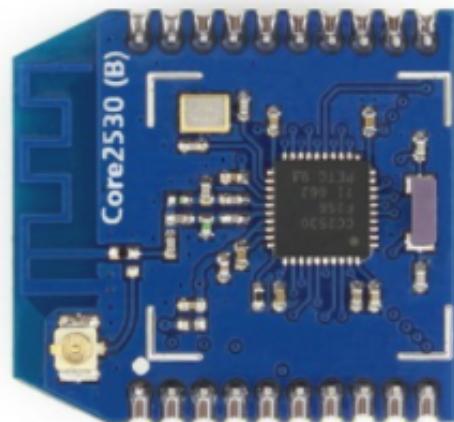


Figure 3.8.: Core2530 SoC ZigBee module

1. The module is considered SoC, which makes it very efficient and compact in size.
 2. The module is XBee usb extension compatible which allows easy configuration.
 3. The module uses UART serial communication for data sharing.
 4. the running firmware can be easily uploaded using the XBee USB adapter.

5. Can be used as Coordinator, Router, End Device.
6. The module uses CC2530F256RHAR chip provided from TeXas Instruments, works on 2.4 GHz, and supports up to 250kbps.
7. The module has 21 GPIOs which makes it able to work as a minor MCU in the system in case of GPIOs shortage in the main MCU, or in case of direct control action need to be taken.
8. Wide supply-voltage range (2 V–3.6 V).

Sensors selection: Speed measurement

1. Measurements within the same vehicle

There are a lot of techniques used to measure vehicle's velocity, the proposed approaches are:

- Method(1): Shaft encoders

There are 2 types of shaft encoders; Optical, and magnetic. Both have almost the same characteristics, however, they perform differently depending on the environment.

- Method(2): Linear distance measurement

This technique was proposed as an alternative to shaft encoders, measuring the linear displacement, and travel time, then calculating the velocity accordingly.

Proposed sensors were: Laser sensors and Grove IR reflective sensor.

At this point, one disadvantage of using slot car appeared, they are very compact, and shaft encoders require specially designed tires so they can be placed on motor shaft, so the plan was to have an optical shaft encoder as low priority alternative in case the second method was not performing well.

Second method implementation and testing:

I proposed using the ADNS-9800 sensor, however, due to some difficulties in shipping and purchase, the plan was changed to use IR sensor, and design a special slots and place them on the track.

IR-slots adaptation plan:

- The sensor was analog, so it requires using an ADC, sample, wait for some time, take new sample. So sampling time was huge "Up to 10 ms per sample, in addition to sampling time and software overhead" compared to the IR sensor which has sampling time of 10 with lower software overhead due to the "digital nature".
- IR sensor was more compact, and weights less.

It was proposed to test and use the QTR-8A Reflectance Sensor Array and check its performance since it was used in the previous model, and was immediately provided. However, the module has failed directly due to the following reasons:

In order to verify the precision and accuracy of such method, at the beginning a test paper has been designed with different test cases.

Calculations: 1. Calculate displacement: whenever a slot is sensed by the IR sensor, a counter is incremented (Using the pulse counter peripheral in the ESP32).

$$Displacement = \text{No.of slots} * \text{Slotwidth}$$

2. Estimate velocity: when a pre-determined time has passed, total count is used to estimate velocity afterwards. Software implementation and drivers are described in section[].

$$Velocity = Displacement / \text{timeperiod} \quad (3.1)$$

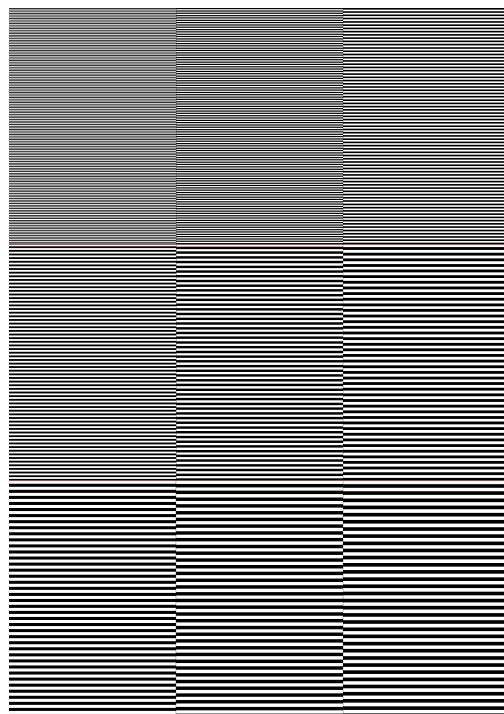


Figure 3.9.: The paper contains 9 sections, each section represents different resolution starting with 0.3 mm until 1.5 mm, so that the optimum resolution can be selected at the end

Results: Unfortunately, as usual the sensor was not performing as expected, error in counts was up to 50%, with very bad precision. New workaround: Another approach was roughly tested, and it was performing surprisingly well. Instead of depending on color difference (White/black slots), a plastic rail has been designed,

an the sensor was used as an artificial touch sensor, test samples were designed using Solid-Works software and manufactured.

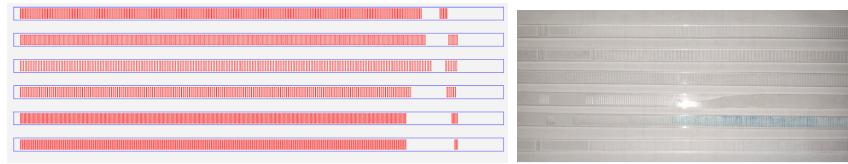


Figure 3.10.: 5 samples was manufactured with the following tuples (slot width "mm", spacing "mm")

(0.3,0.3) (0.5,0.5) (0.8,0.8) (0.8,1) (1,1)

It can be seen that there are 5 extra slots in each sample to allow visual inspection when passing the sensor above them

Results:

Test was done on the samples, and the output signal was captured using logic analyzer, this approach was performing well, and the sensor was able to capture with a very low resolution "Signal analysis was done on the output signal and the latency was minimal in terms of 10th of micro-seconds", however, one main issue appeared, the plastic samples was supposed to be placed underneath the vehicle, in order to do that, the thickness of the samples should not exceed 1.7 mm.

Final decision:

After discussing the issue, it was decided to use the PMW3360 Gaming mouse module and here are some of it's noticeable features:

- The sensor is embedded with system to analyze it's data, therefore, it's using SPI serial communication, which makes it has high and suitable response.
- Frame rate up to 12000 fps.
- Due to high fps, delay was also minimal, approximately (1 – 2) ms.
- Resolution was also very high, up to 12000 CPI, however, the sensor can be easily configured to work with the desired resolution.
- Acceptable acceleration up to 250 IPS/50g.
- 3.3V or 5V compatible.

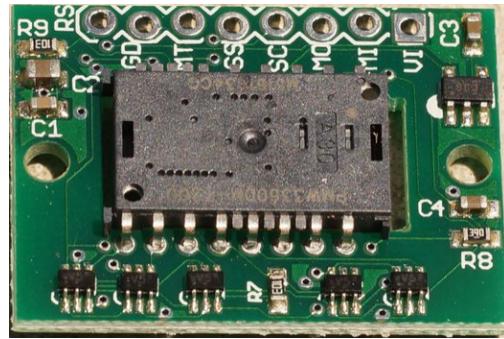


Figure 3.11.: PMW3360 Laser sensor module

2. Measuring preceding vehicle speed

2 methods we suggested to be used in order to calculate preceding vehicle dynamics:

- Method(1): Using Doppler(Radar) speed sensor

After investing sometime searching about suitable doppler effect sensor to measure the velocity of an object, 2 sensors were found which are; HB100 Doppler rada, and K-LD 2-EVAL sensors, however, both were not suitable for use due to the following reasons:

1. They are meant to measure high speed, and high rate of change, therefore, they will not be accurate.
2. Their output readings are noisy and require usage of Hardware and Software filters.
3. Their size is large, and wont fit in our model.

- Method(2): Using distance sensor and elapsed time

Second method is using the distance sensor, measure the rate of change of the spacing distance between each the vehicles and divide it by the elapsed time.

Sensors selection: Distance measurement

Distance measurements is a popular topic and there are plenty of sensors and techniques have been developed for it, each method has certain specs and suits different applications. In this model, 2 sensors have been proposed to be used.

1. ToF ranging sensor (VL53L0X):

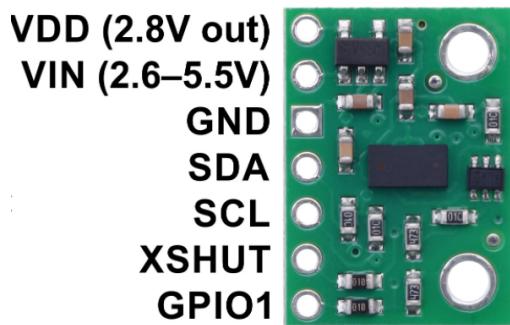


Figure 3.12.: ToF VL53L0X ranging sensor

2. Ulatrasonic sensor (HY-SRF05):

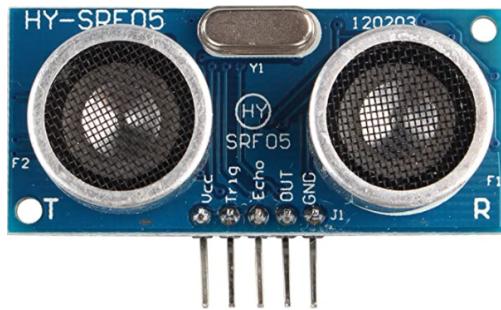


Figure 3.13.: HY-SRF05 Ultrasonic sensor

Comparison between both sensors:

	Ultrasonic	ToF
Output type	Digital	Serial communication (I2C)
Resolution	3 mm	1 mm
Accuracy heightSize	45 × 21	4 × 3 × 1
Wide angle	15	25

Results:

Tests were performed on a test bench, output readings were recorded and the following results were plotted, and analyzed using the Logic analyzer:

- Recorded data:

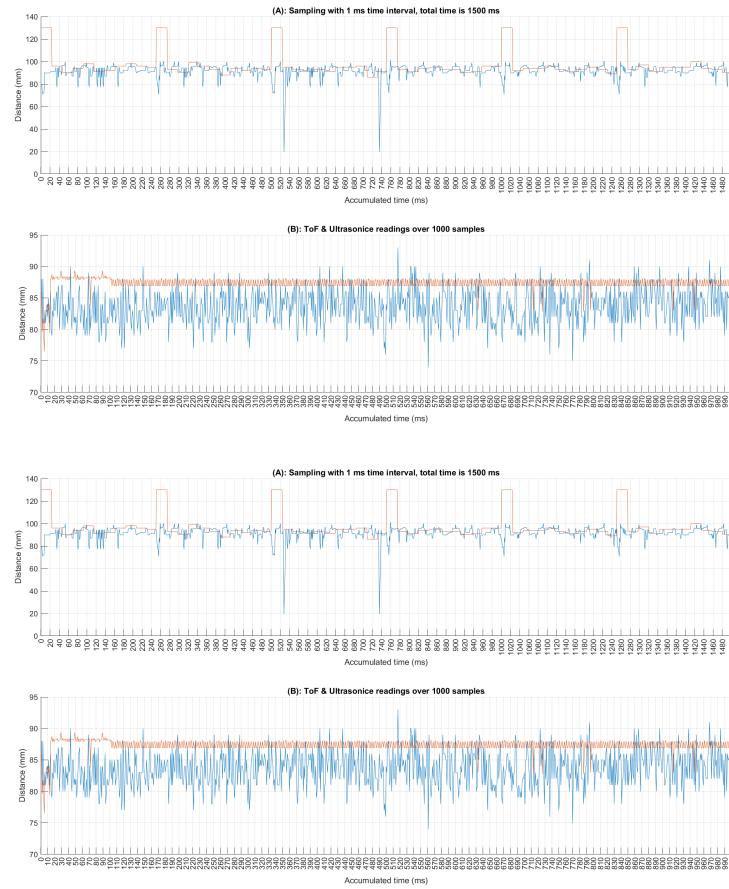


Figure 3.14.: Data recording and analysis using MATLAB software

- Signal analysis: To verify the sampling interval of the Ultrasonic sensor, the output signal was analyzed and recorded as follows:

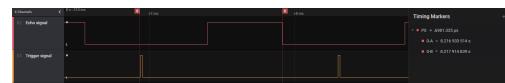


Figure 3.15.: Ultrasonic physical signal snapshot

Conclusions:

Based on the previous observations, it was decided to use Ultrasonic at the end instead of the VL53L0X ToF sensor.

Since the HY-SRF05 Ultrasonic was used at the end, here is some of its specifications:

1. Detection distance: 2cm-450cm.

2. High precision: Up to 0.2cm.
3. Input trigger signal: 10us TTL impulse
4. Input voltage: 5V.
5. Sensor angle: Not more than 15 degrees it is narrower than the ToF, which is an advantage.

Power supply

The power supply should provide the correct voltage level for all components, and also must provide all the platform with the required current. LEDMO adaptor was used to provide this functionality, which has 12 V, and 5 A, to verify the conditions, a simple calculations have been made according to the following table:

Component	Voltage level (V)	Current rating (mA)	Quantity
ESP32	3.3	1200 MAX	3
Ultrasonic	5	2	3
PMW3360 Laser sensor	3.3	23	3
DEV8838 Motor driver	11	200	3
Core2530	3.3	29 MAX	3
Summary		Total current: Voltage levels:	4360 5,3.3,11



Figure 3.16.: LEDMO adaptor

Motor driver

To achieve stable and controllable motion, motor driver should be used to provide control using PWM signals, additional, protection components should be added as well to protect the MCU from reverse current. After some investigation about the used motor in Carrera digital 124 cars, it turned out that it has brushed DC motor and requires 18 V, and consumes about 100 mA[], to limit the speed the selected motor driver supplies up to 11 V, and has 1.7 maximum current for safety.

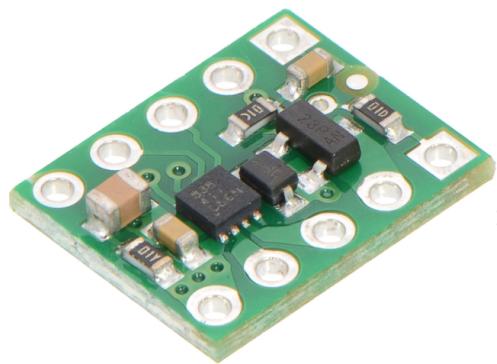


Figure 3.17.: DRV8838 Brushed DC motor driver, output voltage(0 - 11), current rating up to 1.7

Logic analyzer

Logic analyzers are vital in order to debug any serial communication issue that might appear specially when dealing with serial communication sensors, additionally, visual inspection of output waves and the ability to inspect them in Digital and Analog forms allows any research to reach conclusions easily. Therefore, Saleae Logic 8 has been used for this task.



Figure 3.18.: Saleae Logic 8

3.3. Development environment

The ESP32 is a popular MCU in the IoT, therefore, it has more than one framework used for development, available options are microPython, PlatformIO, Arduino, and ESP-IDF, each platform has advantages and disadvantages, and requires different skills and knowledge, additionally, it can be programmed using Python, C, C++,..etc, I have tested PlatformIO, Arduino, and ESP-IDF, and finally, I decided to prepare 2 different frameworks, to allow easy development in the future, which are Arduino, and ESP-IDF. In Git-lab Wiki, I have provided a description for how to setup both frameworks, and additional important information.

3.4. Software Implementation

Although the code is uploaded to the following repository, and its documented, in this section I am going to give a brief explanation about each library/driver, and reason about it.

3.4.1. Communication protocol

With the selected Hardware capabilities, I had the chance to use either the ZigBee protocol using the CC2530 module, Wi-Fi protocol, and BLE protocol which are both embedded on the ESP32 SoC. At the beginning I wanted to make a comparison between the performance of each one of them and the suitability with such application.

However, due to the limitation in time, I decided to start and use the WiFi protocol. At this point, also several options existed, either using the usual Wi-Fi protocol and treat each platoon member as station, and log their data directly, however this scheme will

also require to use each platoon member as Access Point (SoftAP), and to implement that each ESP32 will work as Client and Server, it's not a straight forward plan!

The better alternative is to use the MESH concept, the MESH library does the overhead coming for the previous configuration, the developer just need to perform some configurations regarding the mesh itself. For such plan there were 2 libraries existed, either to use ESP-MDF (It's an API provided from Espressif, contains additional application layers running on top of the ESP-IDF API), or to use the Painlessmesh library.

I decided to use the Painlessmesh library because it was performing some time synchronization mechanism between mesh members, however, the performance was not suitable, and packet were lost with a very high percentage up to 80%, and the behavior was not deterministic at all (packet latency was changing from 10 ms up to 200 ms), no receive acknowledgment (the user should implement it on application level which is not efficient).

The other alternative which I implemented at the end was the use of the ESP-NOW protocol[], the behavior was deterministic, latency was in terms of micro-seconds (approximately 1 – 1.5 ms), automatic receive acknowledgment, broadcasting ability, but MESH formation done manually not like the MDF, or Painlessmesh.

Communication Finite State Machine

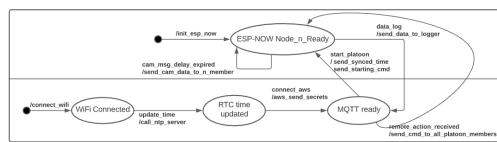


Figure 3.19.: Communication FSM for ESPs Mesh

3.4.2. Ultrasonic

In order to calculate the distance between the sensor and the object, the sensor measures the time it takes between the emission of the sound by the transmitter to its contact with the receiver. The formula for this calculation is:

Theorem 3.1. $D = \frac{1}{2} T \times C$

where D is the distance, T is the time, and C is the speed of sound – 343 meters/second

Ultrasonic FSM

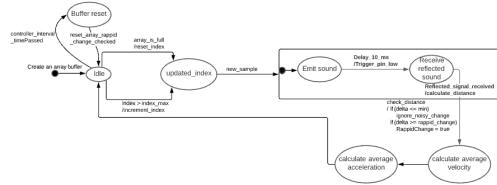


Figure 3.20.: Ultrasonic distance measurement FSM

3.4.3. Laser Sensor

The sensor is using CMOS technology to work as a small camera that captures a number of pixels with a certain speed defined by FPS, in order to that additional hardware peripherals shall be used as DSPs, filters ...etc. Accordingly, a running firmware should exist, and a small MCU as well to manage this data, as well sending it to the main MCU, all of this is embedded in the PMW3360 module.

After capturing the image the firmware uses an algorithm to analyze the differences between each frame and therefore calculate the change in pixels, and the change in distance consequently.

Laser Sensor FSM

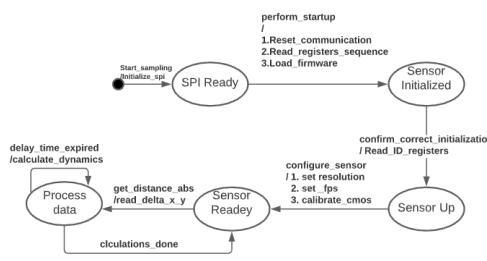


Figure 3.21.: Laser sensor own vehicle dynamics measurements

3.4.4. Motor Driver

Any motor driver is based on a Dido and some additional electronics. Such driver requires a modulated signal.

Signal modulation is done using a technique defined as PWM, done by continuously switching between 0 and 1 with a predefined rate and interval called "Duty".

Motor driver FSM

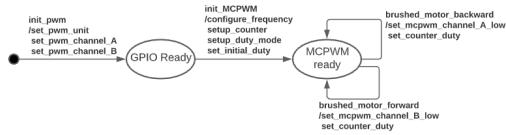


Figure 3.22.: Motor driver velocity control FSM

3.5. Software Architecture

To insure real-timeliness between different tasks, given that the ESP32 has 2 cores, FreeRTOS was selected to run on top of them.

Tasks, peripherals ... etc are illustrated for each member in the platoon; the platoon contains 4 main members, 3 vehicles(Leader 2 followers), and a fourth node responsible for remote operation, control, and data logging.

3.5.1. Architecture

1. Platoon members:

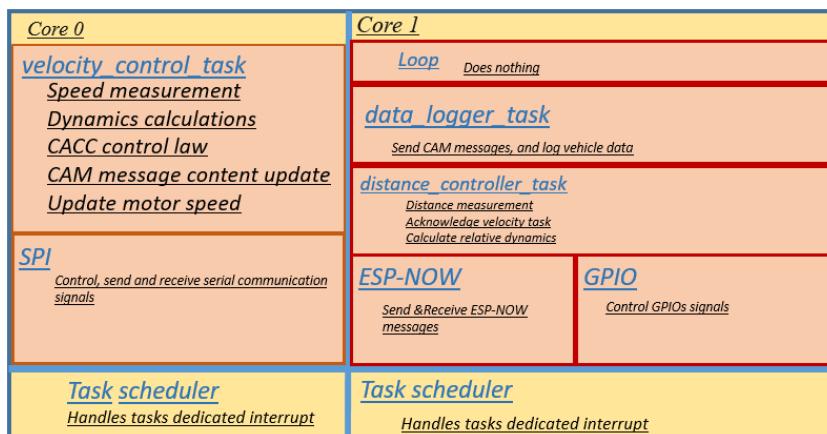


Figure 3.23.: Software Architecture for Platoon members, priority increases from High to low level tasks, each task uses different peripherals, and has different interrupts.

2. Control node:

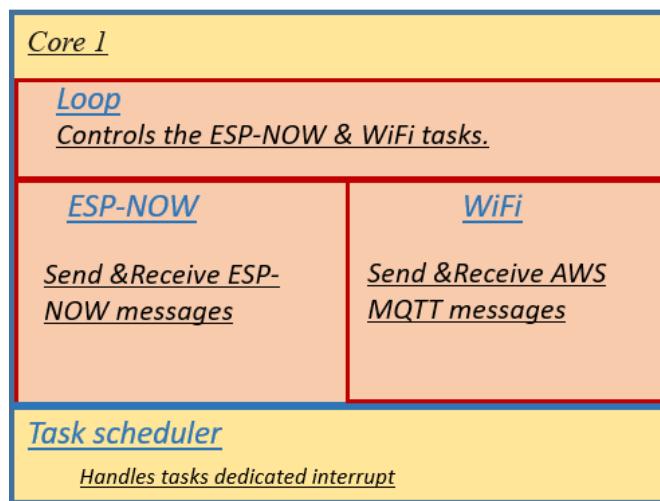


Figure 3.24.: Software Architecture for root node(Control node), priority increases from High to low level tasks, each task uses different peripherals, and has different interrupts.

3.5.2. Task scheduling

3.6. Hardware schematic

In this section the wiring schematic is presented, including all the currently connected components, voltage levels is illustrated as well as other sensor and communication signals.

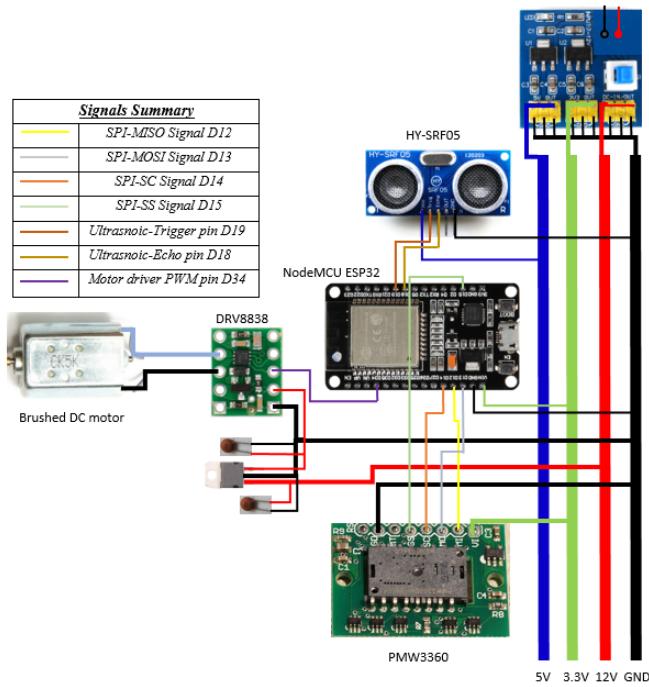


Figure 3.25.: Wiring schematic and signals summary

3.7. Hardware implementation

In this section I am going to describe what how the platform is set, what hardware changes were done, and the assembling steps.

3.8. Additional features

In addition to the previous goals, I have added 2 additional features which will make working on the platform, testing, and developing easier, and more efficient.

3.8.1. Remote control

3.8.2. System configuration ability

4. Platform summary

5. Future work and improvements

A. Appendix

List of Figures

1.1. Time Line	3
1.2. Actual timeline	4
2.1. CAM messages passing in CACC could be either A, B, or Combined	6
2.2. ZigBee and WiFi stack [11]	8
2.3. BLE stack[12]	9
2.4. WiFi Frame structure[13]	9
2.5. ZigBee frame structure[14]	9
2.6. BLE frame structure[15]	10
12figure.caption.17	
12figure.caption.18	
13figure.caption.19	
13figure.caption.5BMW M1 Procar "No.5", Hockenheim 1979	14
3.6. Max length 6.3, with curvature length of 54 cm, according to Carrera track planner software, all gaps and clearances are within the safe limits	14
3.7. NodeMCU ESP32 based	16
3.8. Core2530 SoC ZigBee module	16
3.9. The paper contains 9 sections, each section represents different resolution starting with 0.3 mm until 1.5 mm, so that the optimum resolution can be selected at the end	18
3.10. 5 samples was manufactured with the following tuples (slot width "mm", spacing "mm") (0.3,0.3) (0.5,0.5) (0.8,0.8) (0.8,1) (1,1) (1,0.8) It can be seen that there are 5 extra slots in each sample to allow visual inspection when passing the sensor above them	19
3.11. PMW3360 Laser sensor module	20
3.12. ToF VL53L0X ranging sensor	21
3.13. HY-SRF05 Ultrasonic sensor	21
3.14. Data recording and analysis using MATLAB software	22
3.15. Ultrasonic physical signal snapshot	22
23figure.caption.17onDRV8838 Brushed DC motor driver, output voltage(0 - 11), current rating up to 1.7	24
25figure.caption.18onCommunication FSM for ESPs Mesh	26
3.20. Ultrasonic distance measurement FSM	27
3.21. Laser sensor own vehicle dynamics measurements	27
3.22. Motor driver velocity control FSM	28

3.23. Software Architecture for Platoon members, priority increases from High to low level tasks, each task uses different peripherals, and has different interrupts.	28
3.24. Software Architecture for root node(Control node), priority increases from High to low level tasks, each task uses different peripherals, and has different interrupts.	29
3.25. Wiring schematic and signals summary	30

List of Tables

Bibliography

- [1] Wikipedia. *Zabat, Stabile, Frascaroll, Browand, The Aerodynamic Performance of Platoons, ISSN 1055-1425.*
- [2] Daniel Plöger. *Robust Communication For Control of Robotic Vehicle.* July 2017.
- [3] Kyriakidis, M.; Happee, R.; De Winter, J.C.F. *Public opinion on automated driving: Results of an international questionnaire among 5000 respondents.* *Transp. Res. Part F Traffic Psychol. Behav.* 2015, 32, 127–140 .
- [4] Miao, C.; Liu, H.; Zhu, G.G.; Chen, H. *Connectivity-based optimization of vehicle route and speed for improved fuel economy.* *Transp. Res. Part C Emerg. Technol.* 2018, 91, 353–368..
- [5] Pudane, B.; Rataj, M.; Molin, E.J.E.; Mouter, N.; van Cranenburgh, S.; Chorus, C.G. *How will automated vehicles shape users' daily activities? Insights from focus groups with commuters in the Netherlands.* *Transp. Res. Part D Transp. Environ.* 2019, 71, 222–235..
- [6] Bansal, P.; Kockelman, K.M.; Singh, A. *Assessing public opinions of and interest in new vehicle technologies: An Austin perspective.* *Transp. Res. Part C Emerg. Technol.* 2016, 67, 1–14.
- [7] Pudane, B.; Molin, E.J.E.; Arentze, T.A.; Maknoon, Y.; Chorus, C.G. *A time-use model for the automated vehicle-era.* *Transp. Res. Part C Emerg. Technol.* 2018, 93, 102–114.
- [8] Maider Larburu, Javier Sanchez, and Domingo José Rodriguez. “Safe road trains for environment: Human factors aspects in dual mode transport systems.” In: *ITS World Congress, Busan, Korea (2010)*, pp. 1–12.
- [9] Cohda Wireless. *Platooning article*
- [10] Swaroop, D.v.a.h.g. *String Stability Of Interconnected Systems: An Application To Platooning In Automated Highway Systems,* April 1997, ISSN 1055-1425
- [11] Iván Froiz-Míguez, Tiago M. Fernández-Caramés * , Paula Fraga-Lamas * and Luis Castedo *Design, Implementation and Practical Evaluation of an IoT Home Automation System for Fog Computing Applications Based on MQTT and ZigBee-WiFi Sensor Nodes*
- [12] Olivia *BLE Protocol Stack — Host Controller Interface (HCI)*

- [13] Share Technote *WLAN - Frame Structure*
- [14] Mathworks article *IEEE 802.15.4 - MAC Frame Generation and Decoding*
- [15] Jiun-Ren Lin, Timothy Talty, and Ozan K. Tonguz *On the Potential of Bluetooth Low Energy Technology for Vehicular Applications*, *IEEE Communications Magazine* • January 2015
- [16] JMuhammad Uzair *Evaluation of Different Medium Access Schemes for Vehicle-to-Vehicle Communication*, August 20, 2019
- [17] D. Swaroop *String stability of interconnected systems: An application to platooning in automated highway systems*. Tech. rep. 1. Berkeley: University of California, 1997, p. 65. doi: 10.1016/S0965-8564(97)88297-3.