# Distributed systems

# Minor Project

## DEVELOPMENT OF FAULT-TOLERANT REVERSE ASSIST FOR MULTI-ARTICULATED VEHICLES

**Arnhem, version 2021**

**Master Engineering Systems**

# FRONTMATTER

**Project title: Development of Fault-Tolerant Reverse Assist for Multi-Articulated Vehicles**

Name/number of the group: 2

Name of students:

| | |
|---|---|
| Ilja Minasian | 1669554 |
| Joery Pippel | 1679448 |
| Prakruthi Jayaprakash | 2110970 |
| Haojia Xu | 1545304 |
| Rohan Deshmukh | 1666358 |

Company: HAN University of Applied Sciences

HAN Supervisor: Nikhil Muthakana, Dixon Devasia

Date: 13th June 2023.

## SUMMARY

The aim of the project titled "Development of Fault-Tolerant Reverse Assist for Multi-Articulated Vehicles" was to create and deploy a fault-tolerant reverse assist system for multi-articulated vehicles. The primary objective involved improving the maneuverability of these vehicles during reverse maneuvers through the utilization of sensors and advanced control algorithms.

Multi-Articulated vehicles can pose significant challenges due to their complex structures and poor driver visibility. By offering in-the-moment assistance during reverse maneuvers, this project aimed to address these challenges and reduce potential risks. Therefore, the main objective of this project is to design a system architecture that enables articulation angle sensing, membership service, and steering actuation while ensuring fault tolerance and redundancy for automotive applications. The report presents several research questions, including identifying suitable communication protocols for real-time systems with long lengths, techniques for automatic detection of system trailers and their order, hardware configurations for articulation angle sensing, communication service, and steering actuation, and the incorporation of fault tolerance and redundancy techniques in the system.

To answer these questions the report includes a literature survey, hardware design, software design, an elaborate software description, and a description of the used inverse kinematic model.

The developed system's performance and efficacy were validated through extensive testing. To assess its capacity to deal with various scenarios were simulated. The tests evaluated the system's overall reliability, accuracy, and response time.

## CONTENTS

# 1 INTRODUCTION

## 1.1 Background

In the modern world of transportation and vehicles with heavy loads, the maneuverability question arises often, due to complicated infrastructure and the capacity of the roads. The same question is also common for articulated vehicles, especially with more than one trailer attached.

These multi-articulated vehicles can be of great use in the future, which is good because they provide a wide range of benefits over a single heavy vehicle: The main advantage is that it contributes to economic savings and reduces CO2 emissions [1]. It also improves transportation efficiency and minimizes traffic congestion [2]. Along with the main benefits, there are a few disadvantages, including the Jackknife trailer deviation from the towing unit’s path [3]. Keeping in mind all those narrow lanes, curves, and slopes, especially difficult in surveying the rear part of the vehicle during the maneuvering, increases the complexity and endangers pedestrians and others on the road. An example of such a multi-articulated vehicle is the TRENS Solar Trains -Electric powered vehicle for urban areas [4]. These solar trains can change their composition for every trip, allowing them to transport all kinds of goods or passengers.

Keeping track of all difficulties is complicated for a driver of these solar trains and multi-articulated vehicles in general. To overcome these problems different versions of the parking assist were invented to assist humans in choosing the trajectory or avoiding obstacles. There are two major types of reverse assist systems: 1) The Audio warning and 2) The video monitoring system for providing an image of a blind spot. Reversing for an articulated vehicle would be cumbersome without any assistance during the trajectory as there is frequent change in the number of trailers attached. Due to these changes in the trailers, along with the vehicle's instability and propensity for jackknifing, it makes it difficult to reverse an articulated vehicle without any assistance or trajectory guidance.

## 1.2 Problem definition

The multi-articulated vehicles are unpredictable while driving in the reverse direction, especially considering that more trailers combined could cause a jackknife or inter-unit collision. Moreover, when such multi-articulated vehicles are used in the city environment it requires maneuvering in tight spaces. Because of all these difficulties a reverse assistant would be significantly helpful. However, besides the implementation of an assistance system to improve the usability of the TRENS solar train in an urban area, many design principles are also crucial on the system level. Since the TRENS solar train has a modular design, every added trailer must communicate with each other while being controlled by the driver, this is why a membership service is required. Along with the membership service, the control system also needs to be fault-tolerant and redundant for its automotive application as that will increase its availability, reducing downtime.

## 1.3 Project Objectives

To develop a fault-tolerant reverse assist for multi-articulated vehicles and demonstrate the proof of concept.

- Design a system architecture that enables articulation angle sensing, membership service, and steering actuation.
- Implement a fault-tolerant and redundant real-time environment for the application.

## 1.4 Research question

**Main question:**

- What system design can be implemented in a reverse assist system for multi-articulated vehicles?

**Sub questions:**

- What protocols are suited for communication between modules of a real-time system with lengths of over a meter?
- What techniques can be used to automatically detect system trailer and their order?
- What hardware configuration enables articulation angle sensing, a communication service, and steering actuation?
- How and where to incorporate fault tolerance and redundancy techniques in the system?

## 1.5 Outline of the minor project report

The Structure of the project report is as follows:

1. Introduction
2. Literature survey
3. Methods
4. Results
5. Discussion
6. Conclusions

# 2 LITERATURE SURVEY

This chapter presents a general overview of the specialized literature review, that supports the definition of the process and the general workflow. The literature survey is split into sections representing parts of the system to be built.

The reason for developing a fault-tolerant, real-time system with membership function aspects is to have a robust and reliable communication channel. For this, we need to choose the most appropriate one from the communication protocols, like Ethernet, UART, FlexRay, etc. [5] proposed using Time-Triggered Protocol (TTP) for fault-tolerant real-time systems. In such a type of architecture, the rapid periodic message exchange is initiated to achieve synchronization and guaranteed response. On the other hand, regular Controlled Area Networks (CAN) could be used, but they are less compatible with Real-Time applications due to jitter and latency. To combine these two paradigms [6] proposed TTCAN: time-triggered layer using CAN protocol to communicate in a time-triggered manner. It is shown that it is easy to implement a synchronized network of reasonable redundancy levels.

As pointed out by [7] it is impossible to achieve fault tolerance and redundant data transmission channel without combining two or more TTCAN buses. The paper explains diverse types of networks of this kind such as a Classical redundant network with two buses and a mixed redundant network with 2 or more TTCAN buses. On the other hand, the main problem of such a combination is the synchronization of buses: phase synchronization of cycle time, global time, and rate synchronization.

After fulfillment of hardware fault redundancy, it is needed to understand the way of handling errors, especially when having the intention to build a fail-operational system. For example, as proposed in [8], FDIRO reconfiguration approach could be used for tackling the errors. Such a method is based on FDIR concept [9]. The idea is to handle failures in a stepwise manner by executing the following procedures: 1. Detect the failure, 2. isolate the failure, 3. recover, 4. optimize. Optimization in this case can be initiated by other events apart from failures, for example, synchronization of a global time which supports general workflow and data exchange. To detect failures, the monitoring period must be short and not complex, as well as other steps. This point complements the simplicity principles proposed by [10], which propagates the usage of simple principles to build clean, simple, robust, and scalable systems. To detect errors, we must know the reasons, roots, and consequences. To build a list of this kind, it needs to have, Test plan. The test plan will give us more understanding of the errors when faults are injected manually.

# 3 DESIGN AND METHODOLOGY

## 3.1 Hardware Design

Fig.1 illustrates the hardware design that represents the general arrangement and the connections of the E407 Microcontroller. The communication between the tractor and trailer is achieved using 6 CAN buses. The H-bridge along with the DC motor is also connected to each E407. The 3 POTs are connected to the back end of the tractor and 2 trailers.

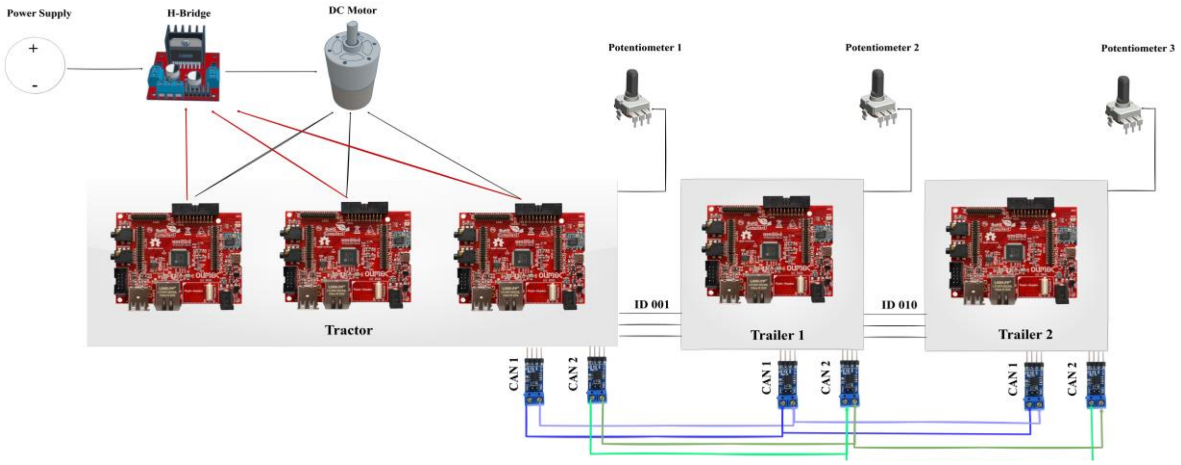


*Figure 1:Hardware design*

## 3.2 Software Design

The development of software design is crucial to implement understandable and simplistic software. The software design functions as a blueprint during software development. In the software design, all functions that have to be developed are divided and placed in subparts to improve simplicity.

The four main subparts of the system are separated in the same manner as [11], this divides the system into the inputs, the controller, the outputs, and the TTCAN system. The inputs subfunction consists of all the inputs regarding the Position control, such as the potentiometer and the encoder. The controller consists of the position controller for the steering angle. The output subpart consists of all outputs regarding the position controller, such as the PWM signal to apply to the motor and the direction of the motor. And the final sub-part, the TTCAN system, contains all the functionalities of the TTCAN system as described in section 3.3.

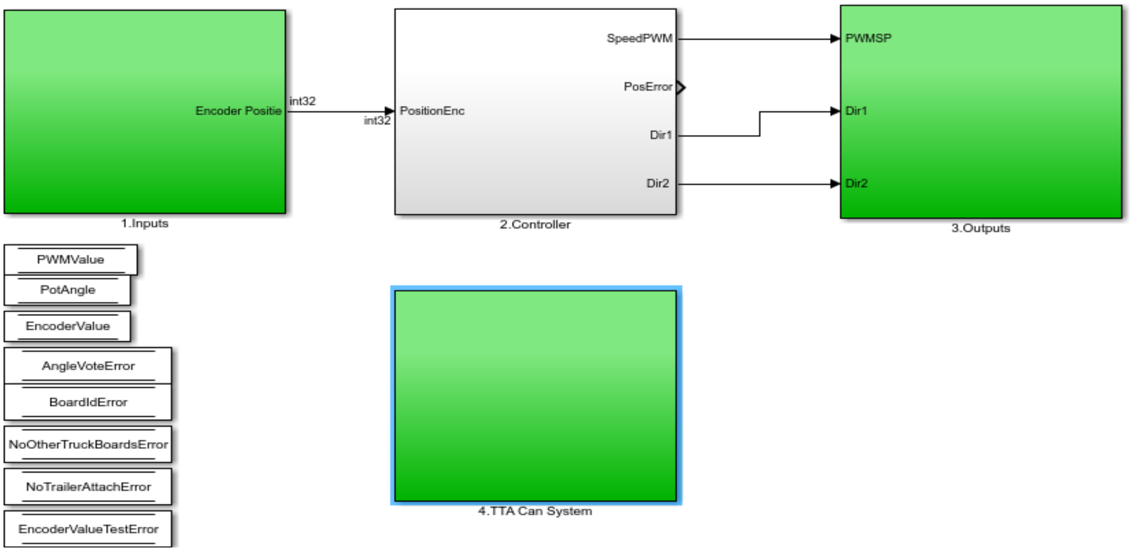


*Figure 2:Software design*

## 3.3 Time Triggered-Controller Area Network (TT-CAN) System

To ensure that a real-time system is predictable and has deterministic behavior a time-triggered protocol (TTP) has been implemented. For this project, a TTCAN system [11] is implemented. This choice ensures that tasks and events are scheduled in a controlled manner, allowing modules to send messages on an available CAN bus while other modules remain ready to receive messages. By utilizing the TTCAN system, critical tasks can be executed on time without conflicts, thereby enhancing system reliability.

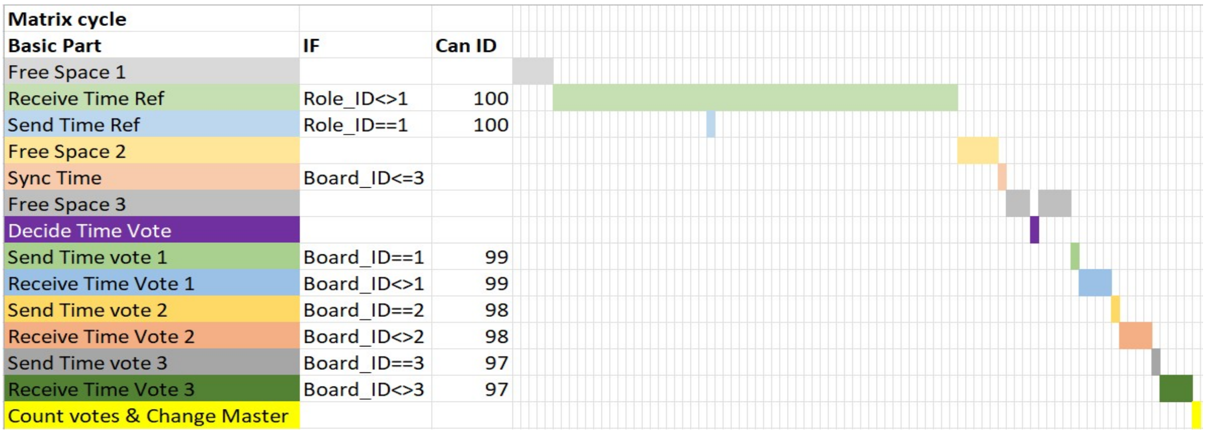
### 3.3.1 Scheduling

A TT-CAN system works with the scheduling of communication and computation, done based on a Matrix Cycle. A Matrix Cycle is a sequence of allocated time marks and windows. At each time mark or during each window a computational or communication task can be executed.

To make sure that each communication or computation task has its designated time slot on the CAN bus a *Cycle Schedule* is developed. This cycle schedule can be divided into a Basic Cycle (BC) and a Main Cycle, in which the BC consists of all the tasks related to *Global Time Synchronization*. The Main Cycle consists of all the actions required for the calculation of the steering angle.

### 3.3.2 Basic Cycle

To schedule all tasks and events, the boards need to be synchronized to a common *Timeline*. To construct this common timeline, reference messages are sent during Initialization and at a designated time in the Basic Cycle. This BC consists of all the parts required to implement board synchronization with a redundant time master.



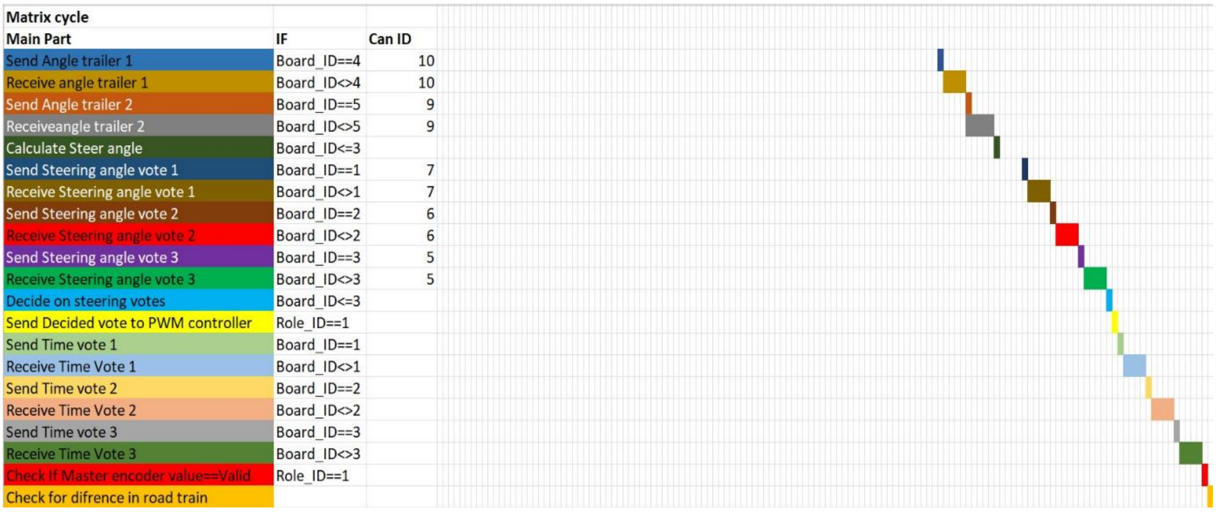
| Matrix cycle | | |
|---|---|---|
| Basic Part | IF | Can ID |
| Free Space 1 | | |
| Receive Time Ref | Role_ID<>1 | 100 |
| Send Time Ref | Role_ID==1 | 100 |
| Free Space 2 | | |
| Sync Time | Board_ID<=3 | |
| Free Space 3 | | |
| Decide Time Vote | | |
| Send Time vote 1 | Board_ID==1 | 99 |
| Receive Time Vote 1 | Board_ID<>1 | 99 |
| Send Time vote 2 | Board_ID==2 | 98 |
| Receive Time Vote 2 | Board_ID<>2 | 98 |
| Send Time vote 3 | Board_ID==3 | 97 |
| Receive Time Vote 3 | Board_ID<>3 | 97 |
| Count votes & Change Master | | |

*Figure 3:Timeline of basic cycle*

Fig.3 shows the timeline of this Basic Cycle, the steps for this timeline are discussed under sub-section Synchronization.

### 3.3.3 Main Cycle: Determination of the steering angle

Tasks like cyclic calculation, and communication for actuating the steering angle are performed in this Main Cycle. This main cycle makes sure the CAN bus is ready to send the necessary messages and makes sure that the calculations and readings are up to date at the time of sending.



| Matrix cycle | | |
|---|---|---|
| Main Part | IF | Can ID |
| Send Angle trailer 1 | Board_ID==4 | 10 |
| Receive angle trailer 1 | Board_ID<>4 | 10 |
| Send Angle trailer 2 | Board_ID==5 | 9 |
| Receiveangle trailer 2 | Board_ID<>5 | 9 |
| Calculate Steer angle | Board_ID<=3 | |
| Send Steering angle vote 1 | Board_ID==1 | 7 |
| Receive Steering angle vote 1 | Board_ID<>1 | 7 |
| Send Steering angle vote 2 | Board_ID==2 | 6 |
| Receive Steering angle vote 2 | Board_ID<>2 | 6 |
| Send Steering angle vote 3 | Board_ID==3 | 5 |
| Receive Steering angle vote 3 | Board_ID<>3 | 5 |
| Decide on steering votes | Board_ID<=3 | |
| Send Decided vote to PWM controller | Role_ID==1 | |
| Send Time vote 1 | Board_ID==1 | |
| Receive Time Vote 1 | Board_ID<>1 | |
| Send Time vote 2 | Board_ID==2 | |
| Receive Time Vote 2 | Board_ID<>2 | |
| Send Time vote 3 | Board_ID==3 | |
| Receive Time Vote 3 | Board_ID<>3 | |
| Check If Master encoder value==Valid | Role_ID==1 | |
| Check for difrence in road train | | |

*Figure 4:Timeline of mean cycle*

The above Figure 4 shows the timeline of the Main Cycle. The tasks and functions of this timeline are discussed in the coming sections.

### 3.3.4 Synchronization

Board synchronization stands as a critical element within a Time-Triggered Architecture (TTA), and its significance is elucidated in Fig. 3, highlighting the essential tasks for time synchronization.

The process begins with the transmission and reception of a time reference at the start of each cycle. The other boards are prepared to receive a time reference message sent by one of the "Truck Boards", which is currently acting as the time master, over the Controller Area Network (CAN) protocol. The time reference message has a distinct CAN ID and is sent within a predetermined time window. With this strategy, clock drift during the cycle can be effectively compensated for by handling out-of-sync reference messages.

Upon receipt of a reference message, the instant of the incoming message is compared to the time the reference message was expected to arrive. The time the reference message was expected to arrive is calculated by forehand based on a calculated time delay of the message. This makes it possible to figure out how many ticks the local clock is off from the global time. When the de-synchronization ticks are determined, the local time is adapted with this value.

Next up is the time master voting system, all three “Truck Boards” sequentially send, while the other boards receive their vote. This vote exchange's goal is to choose the next time master for the following cycle based on the number of CAN errors that each board experienced in comparison to the other two "Truck Boards." The number of errors serves as a measure of the board's communication performance, and this voting process aids in identifying the board with the most reliable communication link.

The last task of the Basic Cycle is to choose the time master for the next cycle based on the votes cast during the prior time windows. The next time the master will be chosen using this selection procedure to ensure that the majority of votes are considered. The current time master stays the same when all boards have an equal number of votes, ensuring consistency and stability in the synchronization procedure. The TTCAN--system achieves a robust and redundant synchronization function through these meticulous tasks, maintaining accurate time coordination between the boards.

For a more thorough explanation of the de-synchronization tick calculations and other tasks, refer to **APPENDIX B**.

### 3.3.5 Fault tolerance methods

The implementation of the TTP and dual channel CAN buses can already be seen as fault tolerance methods. However, there are also additional methods intertwined in the TTCAN system to upgrade the system's fault tolerance. These fault tolerance methods are based on the Triple Modular Redundancy (TMR) of the “Truck boards” and a voting mechanism. The fault tolerance mechanisms in the synchronization method of the TTCAN system are already explained in the previous chapter. The following two chapters contain the other fault tolerance methods implemented.

#### 3.3.5.1 Redundant inverse kinematic calculations

The inverse Kinematic model is elaborated under section 3.4 and is implemented on the three “Truck Boards” and independently calculating the desired steering angle during their specified time window on the main cycle shown in Fig.4 After calculation each “Truck Board” sequentially sends and receives the steering angle votes on the CAN buses. These votes are compared to one another to see if the values are equal within a small margin, if that is the case the validated value is used as a setpoint for the position control. In the occasion that none of the three votes are within range of each other, the error flag of the votes is sent and as a result, the steering function is set to a ground state.

#### 3.3.5.2 Redundant encoder values

Even though the controlling of the actuation is done outside of the TTCAN system, the encoder values are checked to decide if the master board is using a valid encoder value. To achieve that, each “Truck board” sequentially sends and receives the encoder value in the same way as done for the inversed kinematic calculations. However, the validation of the encoder value is different from the validation on the inversed kinematic calculations, because the encoder value changes outside of the TTCAN system the encoder value might already have been updated. Therefore, instead of finding one valid encoder value, the encoder value of the current master is compared to the other 2 in order to see if it is correct. When the encoder value of the current master isn’t correct, it indicates an error flag and sends the steering function to a ground state.

## 3.4 Inverse Kinematic (IK) Model

To demonstrate how fault-tolerant and membership functions are implemented into reverse assist systems for articulation vehicles as shown in Fig. 5, an inverse kinematic (IK) model is used to simulate the scenario when the end steering angle delta_c is known as the input but tractor steering angle delta_12K as the output.

The model is simplified to a single track provided that only one tire located at the Centre of the axle is considered in the analysis. This simplification allows for a more straightforward representation of the vehicle's kinematics, as the complexities associated with multiple tires and their interaction with the ground are ignored [12].

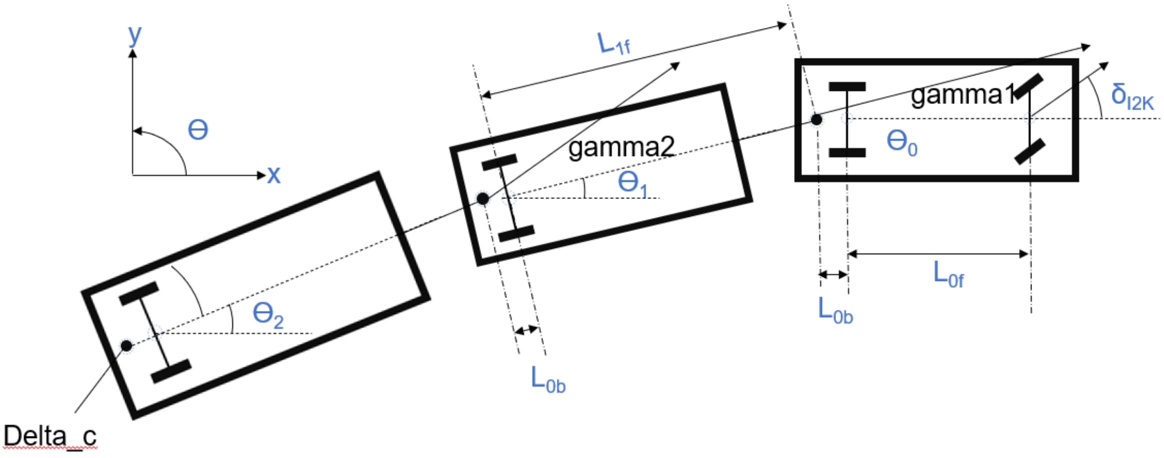


*Figure 5:Articulation with 2 trailers*

Further simplification is taken into account to ease the effort required on the IK model. The maximum number of trailers is limited to 2, meaning that only single-articulation and double-articulation scenarios are considered. In addition, the introduction of fixed yaw angles makes the IK model static, subsequently, the entire IK model acts as a gain. Table 1. shows the parameters involved in the IK model and their values.

*Table 1: IK model variables*

| Symbol | Description | Value |
|---|---|---|
| $\theta_0$ | Yaw angle of the tractor | Constant |
| $\theta_1$ | Yaw angle of the 1st trailer | Constant |
| $\theta_2$ | Yaw angle of the 2nd trailer | Constant |
| L1f | Wheelbase of the trailing units | 3 [m] |
| L0f | Wheelbase of the tractor | 2[m] |
| L0b | Distance between the rear axle and the hitch (kingpin) | 0.3[m] |
| Δ_12k | Steering angles at the tractor | Output |
| Δ_c | desired steering at the end trailer | Input |
| V2 | Speed at the end trailer | -1 [m/s] |

The IK model's response is tested by a step input of 1 [deg], fig.6 below shows their step response when yaw angles are set to 0 [deg], again, in static conditions, they act as a constant gain.

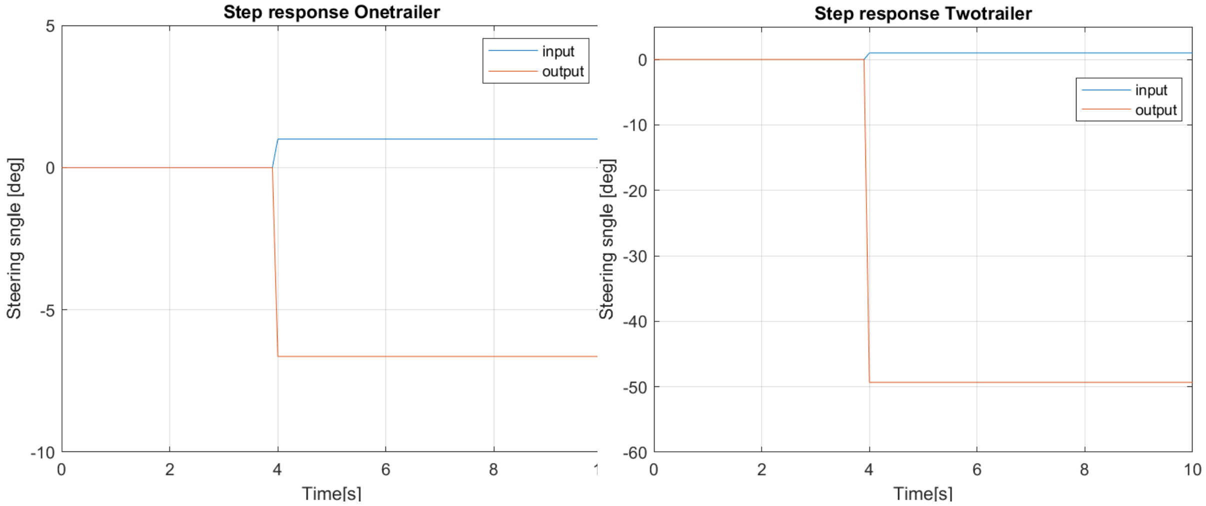


*Figure 6:Step response IK model*

## 3.5 Position controller

The PID position controller as shown in fig.7 below is outside the TTCAN and free running, but its inputs and outputs are continuously checked and monitored in TTCAN to provide fault-tolerance and fail-safe protection.

In the controller, besides the triple redundancy on signals for fault-tolerance, it also continuously monitors the error message from TTCAN cycles, the controller will not generate any output when an error message is received. In addition, the controller also monitors the Role_ID so that only the master (1) role can activate it.

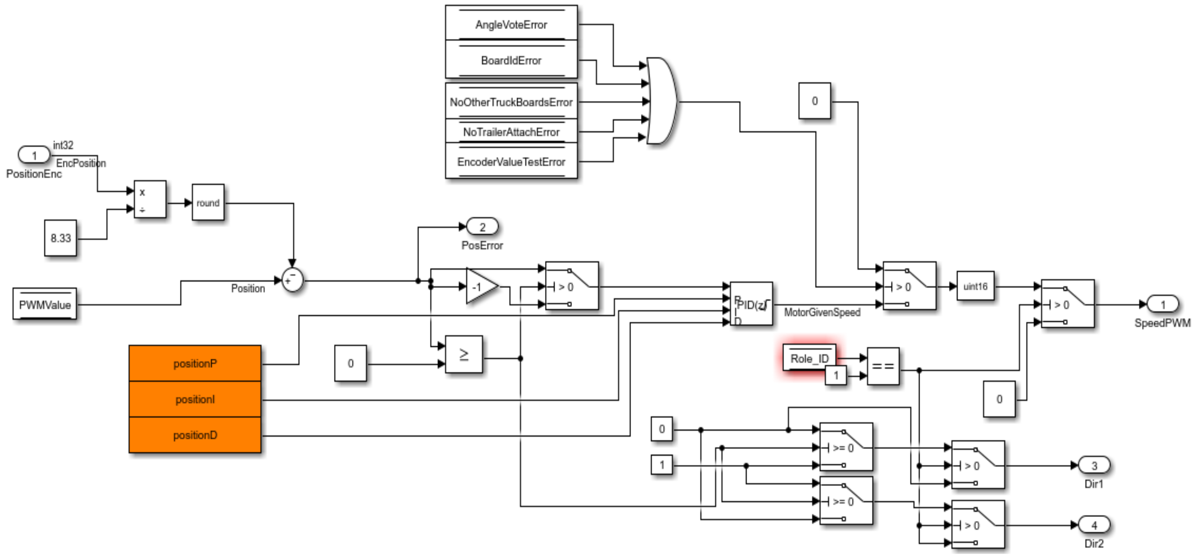


*Figure 7:Position controller*

## 3.6 BOARD ID’s MANAGEMENT

To assign IDs systematically among multiple boards, this methodology is based on a 3-bit coding system where unique identifiers are assigned sequentially through the series of boards. Initially, hardwired IDs are used for one board while subsequent ones can dynamically assign themselves an ID by receiving signals through their 3 digital input (Din) and output (Dout) ports.

If no signal or wire break condition exists, "000" acts as the standard identifier value for all boards involved. Commencing at the first board's assigned ID, each successive board awaits its turn to receive an updated code from its predecessor before designating itself with an appropriate sequential identifier, when necessary, signals are received.

Even if there occurs any such scenario where no signal might be present, or wire breaks occur during this process; employing self-assignment power boards facilitate an uninterrupted course of action by assigning themselves an "000" identifier indicating that there exists no proper signal in place currently.

This feature also increases the system's robustness, as it mitigates the need for a system restart. Overall, this ID assignment methodology leverages these steps to ensure seamless progression between successive power boards while handling such circumstances effectively.

# 4 TEST PLAN

To confirm if the developed system functions as it’s supposed to, a test plan is developed. In this test plan as shown in Table 2. all functionalities regarding fault tolerance and redundancy are transformed into tests. The outcome of these tests indicate how well the system functions according to its requirements. For additional information on the test plan check Appendix C.

*Table 2:Test plan*

| Test Number | Action | Follow up action | Tested subject | Expected result |
|---|---|---|---|---|
| 1 | Unplug power (All) | Plug power | Auto recover from power loss, time master re-establishment | System is operational |
| 2 | Unplug power (One of the first three truck boards) | Plug power | Auto recover from power loss | System is operational |
| 3 | Unplug power (intermediate) | Plug power | Auto recover from power loss | System is operational |
| 4 | Unplug power (Last) | Plug power | Auto recover from power loss | System is operational |
| 5 | Unplug power (One of the first three truck boards) | - | Board failure | System is operational |
| 6 | Unplug power (intermediate) | - | Board failure | System is in safety mode |
| 7 | Unplug power (Last) | - | Board failure | System is in safety mode |
| 8 | Unplug communication wires | Plug comm back | Auto recover from communication failure | System in an operational mode |
| 9 | Unplug communication wires | - | Auto recover from communication failure | System in a safety mode |
| 10 | Unplug id cables | Plug id cables | Auto recover from id failure | System in an operational mode |
| 11 | Unplug id cables (every) | - | Board failure | System in a safety mode |
| 12 | Unplug trailer potentiometer | Plug potentiometer | Potentiometer Failure | System is operational |
| 13 | Unplug trailer potentiometer | - | Potentiometer Failure | System is in safety mode |
| 14 | Disconnect encoder from one of the truck boards | - | Input failure | System is operational |
| 15 | Disconnect encoder from two of the truck boards | - | Input failure | System is in safety mode |
| 16 | Unplug potentiometer from one of the truck boards | - | Analog input failure/Calculation failure | System is operational |
| 17 | Unplug potentiometer from two of the truck boards | - | Analog input failure/ calculation failure | System is in safety mode |

# 5 RESULTS

By implementing all the functionalities as discussed in Chapter 3, the system functions and can alternate the steering angle of the motor. To indicate how well the system functions, it is tested with the use of the test plan. The results of the tests are tabulated below.

*Table 3:Test results*

| Test Nr | Results |
|---|---|
| 1 | System is operational after a sequential restart cycle, at a restart at the same time some boards are out of sync |
| 2 | System remains operational if all other truck boards function normally |
| 3 | System goes to the ground state until the board is back online and then returns to normal |
| 4 | System thinks the last trailer is disconnected, goes to one trailer mode, and at reconnection, it goes back to two trailer mode |
| 5 | System remains operational if all other truck boards function normally |
| 6 | System goes to ground state |
| 7 | The system thinks the last trailer is disconnected, and goes to one trailer mode. |
| 8 | Makes the board disconnected malfunction, based on which board is disconnected either the system goes to the ground state or functions without it. |
| 9 | Makes the board disconnected malfunction, based on which board is disconnected either the system goes to the ground state or functions without it. |
| 10 | System goes to the ground state, depending on which board it is. And functions once it is reconnected |
| 11 | System goes to the ground state, depending on which board it is |
| 12 | Not detected, no double sensor redundancy |
| 13 | Not detected, no double sensor redundancy |
| 14 | System remains operational if all other truck boards function normally |
| 15 | System goes to ground state |
| 16 | System remains operational if all other truck boards function normally |
| 17 | System goes to ground state |

# 6 DISCUSSION

The implemented system plays a crucial role in ensuring the robustness and reliability of the overall system. However, several key limitations have been identified that require careful consideration and potential improvements. In this discussion,  the challenges associated with the CAN channels, encoder errors, redundancy implementation, cable length testing, and ID assignation module are mentioned.

The first point of discussion is the redundancy of the double CAN channels. The two CAN channels in HAN coder do not allow them to be independent, so redundancy using these two channels does not improve the *robustness* of the system.

Another issue arises from the fact that the encoder error does not influence the selection of the time master among the truck boards. As a result, the system may go to the ground state while receiving two valid encoder values of the other boards. To fully use the capabilities of the triple module redundancy of the truck, the master would need to be changed depending on this error as well, instead of just changing depending on the communication errors.

The third issue is the Lack of Redundancy on Trailer Boards. While the system implements redundancy within the truck boards, the same level of redundancy is not extended to the trailer boards. To have a fully redundant real-time system the encoder values of the trailer and trailer board itself should also be made redundant

Another aspect that requires attention is the limited testing of the system with only short cable lengths (less than 30cm). While the system operates optimally under these conditions, it remains unclear how it would perform with longer cable lengths or when transmitting digital signals without any cable load. Conducting comprehensive testing across a range of cable lengths and load conditions is crucial to validate the system's performance and ensure its reliability in various operational scenarios.

The last issue arises from the reliability challenges of the ID assignation module. The ID's assignation module, which is responsible for assigning id to boards, is not fully reliable, because it relies on the correctness of a single channel I/O and cables. In case one of the wire breaks the system malfunctions, and when it does it is hard to determine if there is a board failure or connection problem. The first iteration of the ID functionality implemented passing the id forward and backward to reestablish the order in case of board failure, but it is consuming twice as many cables compared to the current iteration. In our case the id of the first boards is hardwired which plays as a double edge razor: if these are not plugged (or board failure) – id's will not be assigned, but from another perspective, it works well from the safety side by turning off all the activities involved in the angle control.

# 7 CONCLUSION AND RECOMMENDATION

The development of a fault-tolerant reverse assist system for multi-articulated vehicles has been outlined. The project objectives are to design a fault-tolerant real-time system that enables articulation angle sensing, membership service, and steering actuation, as well as implement a fault-tolerant and redundant real-time environment for the application.

The literature survey highlighted the importance of choosing the appropriate communication protocol for a real-time system, such as Time Triggered Protocol (TTP) or TTCAN, to ensure synchronization and guaranteed response. It also emphasized the need for fault tolerance and redundancy techniques.

The design and methodology section presented the hardware and software design aspects of the system. The hardware design included the arrangement and connections of the E407 Microcontroller, as well as the expected use of dual CAN buses for communication between the tractor and trailers. The software design focused on dividing the system into inputs, controller, outputs, and the TTCAN system, providing a blueprint for software development.

The implementation of a TTCAN is proposed to ensure deterministic behavior and enhance system reliability. The system's scheduling, synchronization, fault tolerance methods, and redundancy techniques are described. The use of redundant inverse kinematic calculations and redundant encoder values added an extra layer of fault tolerance to the system.

The inverse kinematic (IK) model is discussed as a means to simulate the scenario where the end steering angle delta_c is known as the input and the tractor steering angle delta_12K is the output. The IK model simplifications are explained, focusing on single-articulation and double-articulation scenarios.

The recommendations for future work focus on the implementation of dual CAN bus to provide redundancy in communication since the TTCAN architecture provided by Diego Lopez doesn’t support it. Additionally, more real-world testing and refining of the Man-Machine Interface (MMI) can be investigated for a more intuitive user experience.

## REFERENCES

[1] Karam, A., & Reinau, K. H. (2021). Evaluating the effects of the a-double vehicle combinations if introduced to a line-haul freight transport network. *Sustainability*, *13*(15), 8622.

[2] Nyman, P., & Uhlén, K. A. R. I. N. (2014). Coordination of actuators for long heavy vehicle combinations using control allocation. *Master's Thesis in Systems, Control and Mechatronics*.

[3] Åkerman, I., & Jonsson, R. (2007). European Modular System for road freight transport: experiences and possibilities. TFK-TransportForsK AB.

[4] https://www.trens.eu/en

[5] H. Kopetz and G. Griinsteidl, Institut fur Technische Informatik, Technische Universit at Wien Treitlstr. 3/182/1, A-1040 Vienna, Austria,*TTP - A Time-Triggered Protocol for Fault-Tolerant Real-Time Systems.*

[6] B. Müller, T. Führer, F. Hartwich, R. Hugel, H. Weiler, Robert Bosch GmbH, *Fault tolerant TTCAN network*

[7] Tobias Kain, Volkswagen Group Innovation, Volkswagen AG, Germany. Hans Tompits, Institute of Logic and Computation, Technische Universitat Wien, Austria, Julian-Steffen Muller, Volkswagen Group Innovation, AG, Germany, Philipp Mundhenk, Autonomous Intelligent Driving GmbH, Germany, Maximilian Wesche, Volkswagen Group Innovation, Volkswagen AG, Germany, Hendrik Decke, Volkswagen Group Innovation, Volkswagen AG, Germany, *FDIRO: A General Approach for a Fail-Operational System Design.*

[8] Tobias Kain, Volkswagen Group Innovation, Volkswagen AG, Germany, *FDIRO: A General Approach for a Fail- Operational System Design.*

[9] Ali Zolghadri, Universite´ de Bordeaux, CNRS, IMS-lab, 351 cours de la Libe´ration, 33405 Talence cedex, France *Advanced model-based FDIR techniques for aerospace systems: Today challenges and opportunities.*

[10] Hermann Kopetz Vienna University of Technology Department of Computer Engineering, *Real-Time systems, Design Principles for Distributed Embedded Applications*, Second edition.

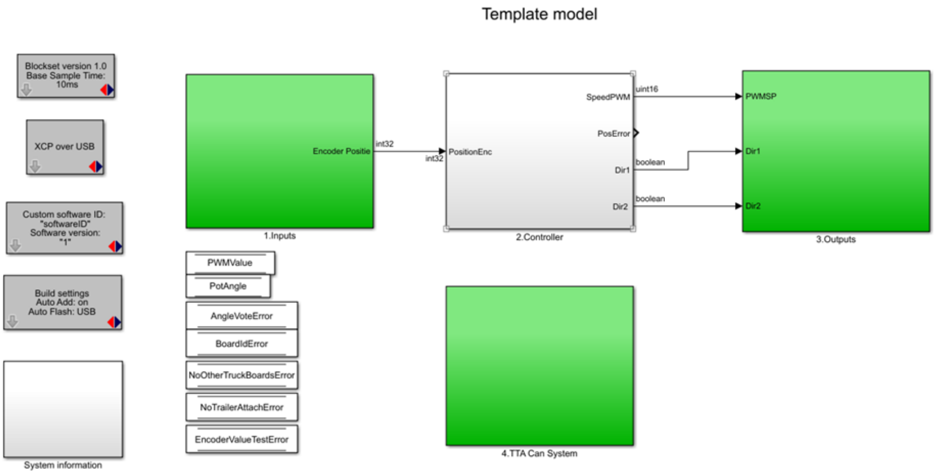
[11] Diego Martın Lopez (2022) , Time-Triggered Architecture, fully redundant, Real-Time embedded distributed system with HANcoder, Master Thesis, Hogeschool Arnhem and Nijmegen.

[12] Rakshith KUSUMAKAR (2017). Autonomous Parking for Articulated Vehicles. *Master's Thesis in Automotive System*, Hogeschool Arnhem and Nijmegen.

# APPENDIX A: NOMENCLATURE

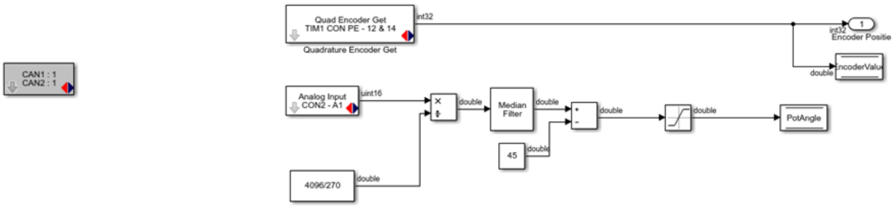
| | |
|---|---|
| TT-CAN | Time-Triggered Controller Area Network |
| TTP | Time-Triggered Protocol |
| TTA | Time-Triggered Architecture |
| MC | Matrix Cycle |
| BC | Basic Cycle |
| CAN | Controller Area Network |
| IK | Inverse Kinematic |
| PWM | Pulse Width Modulation |
| PID | Proportional +Integral+ Derivative |

# APPENDIX B: IMPLEMENTATION OVERVIEW



The system is divided into 4 subsystems,1. The Inputs, 2. The Controller, 3. The Outputs and 4.the TTCAN system.

### THE INPUTS:

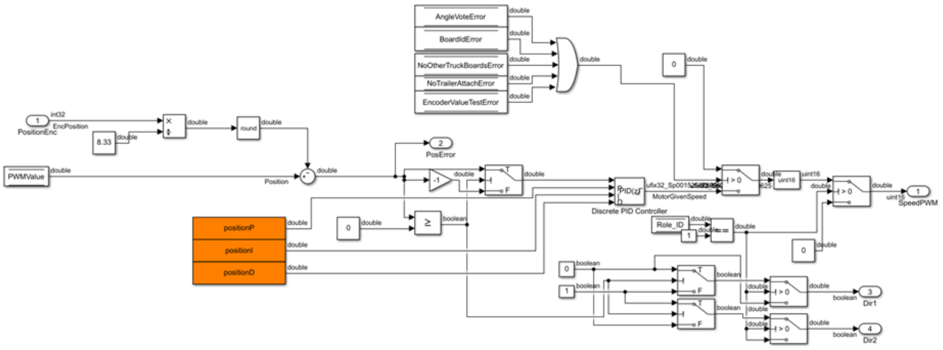


The above figure shows all the inputs used, but only contains the inputs which have nothing to do with the TTCAN implementation. The potentiometer (POT) angle and encoder value are used by the TTCAN system for validation, but are needed for the application, and not for the general functioning of the TTCAN.

In this part the encoder value is passed on to a variable to be validated by the TTCAN system and is sent to an output port to be fed through to the controller system.

The POT value is written to a value to be validated and used by the TTCAN system.

### THE CONTROLLER



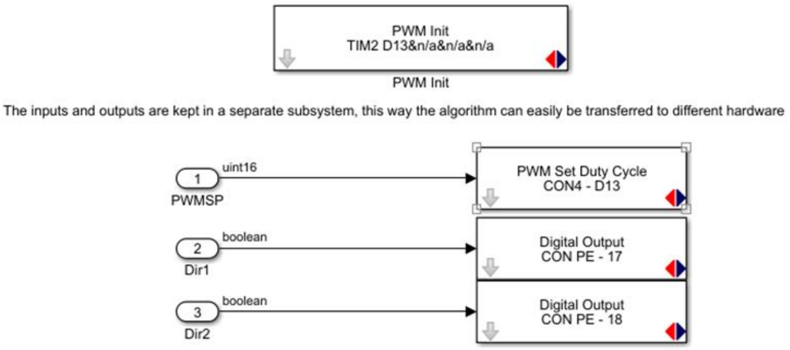
The controller consists of a simple PID block, with PID values initialized with the Ziegler Nichols method. To get the encode values to match the Setpoint of 0 to 180 degrees. The PWM value variable is the validated setpoint originated from the TTCAN system.

There is a ground state added to the controller when there are errors on the system, the errors are:

- *Angle vote error*: this error is high when there are no 2 equal angles in the CAN vote system. So, each "Truck boards" give a different steer angle.
- *Board ID error*: board id is changed during runtime, probably a lose connection
- *No other truck board error*: no truck board connected to validate the SP and Encoder
- *No trailer attached*
- *Encoder Value Test Error*: indicates that the used encoder value isn't the same as on the other truck board.

The system is only controlled when it's the Time Master, indicated by the role ID.

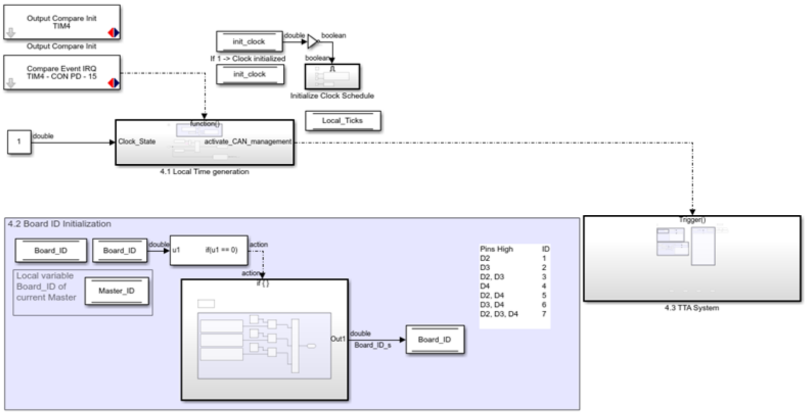
### THE OUTPUTS



The inputs and outputs are kept in a separate subsystem, this way the algorithm can easily be transferred to different hardware

The Outputs block only consists of the outputs that do not have functionality regarding the TTCAN system. In this case it outputs the PWM value and the direction pins.

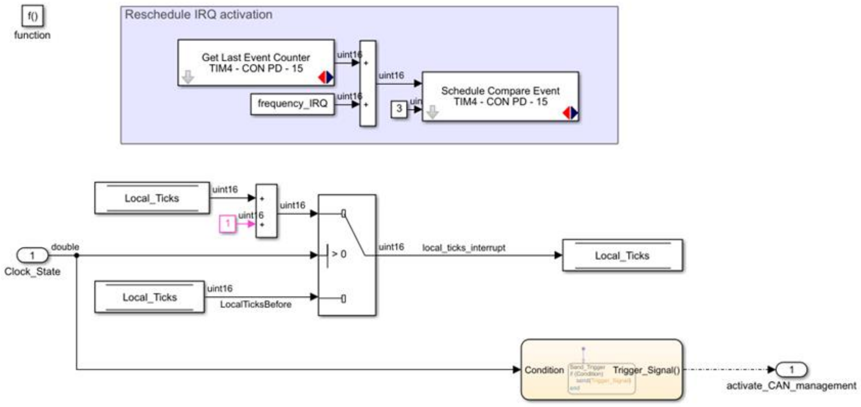
### THE TTCAN SYSTEM



| Pins High | ID |
|---|---|
| D2 | 1 |
| D3 | 2 |
| D2, D3 | 3 |
| D4 | 4 |
| D2, D4 | 5 |
| D3, D4 | 6 |
| D2, D3, D4 | 7 |

This is the main overview of the CAN system consisting of three subparts, *Local time generation, Board ID initialization and the TTA system.*

## 4.1 LOCAL TIME GENEARATION

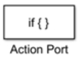


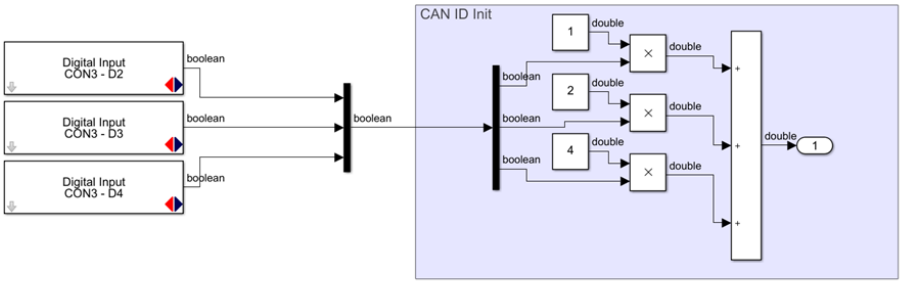
This part adds one to a Local tick each specified amount of hardware ticks (500) which means a local tick each 0.5 millisecond. At each local tick a specified subsystem is enabled.

## 4.2 BOARD INITILIZATION

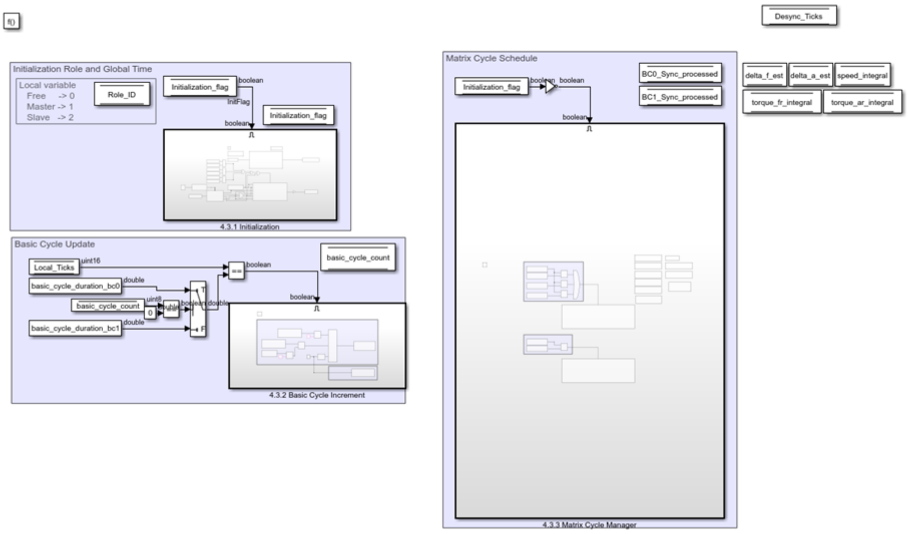
Based on the Input Pins and a binary calculation the Board ID is initialized, the pass through to the next board is done in a different subsystem.

This is only run on startup when the system doesn’t have a Board ID yet.



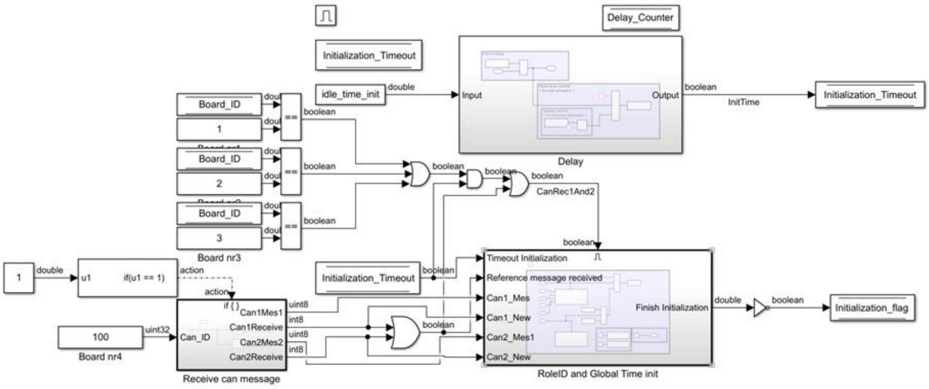


## 4.3 TTA SYSTEM



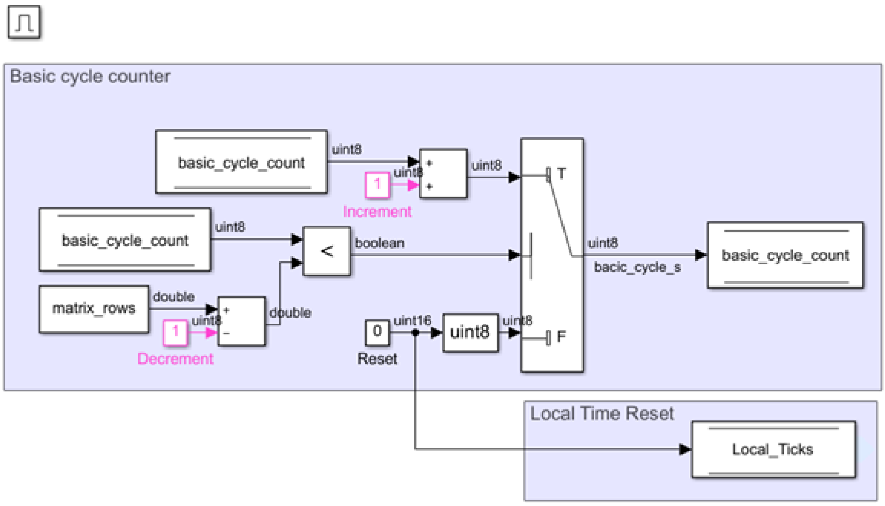
The TTA System consists of 3 sub-parts.

### 4.3.1 Role Initialization, Sets the time master and Local ticks for the time slaves.



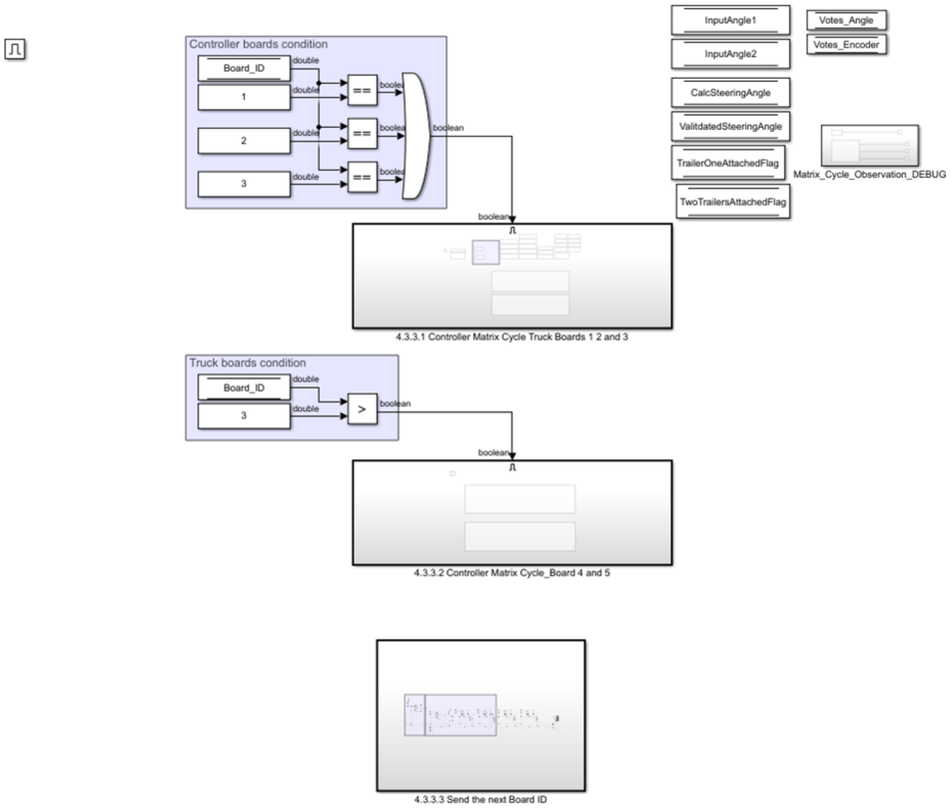
The first block initiates a delay of 2 matrix cycles, if that delay is ended and the current E407 is indicated as a “Truck board” (Id 1 2 3) it starts as a Time Master. When a time reference message has arrived on the system during the delay the delay it starts as a “Time Slave”.

### 4.3.2 Basic cycle update, sets the duration and resets the matrix cycle



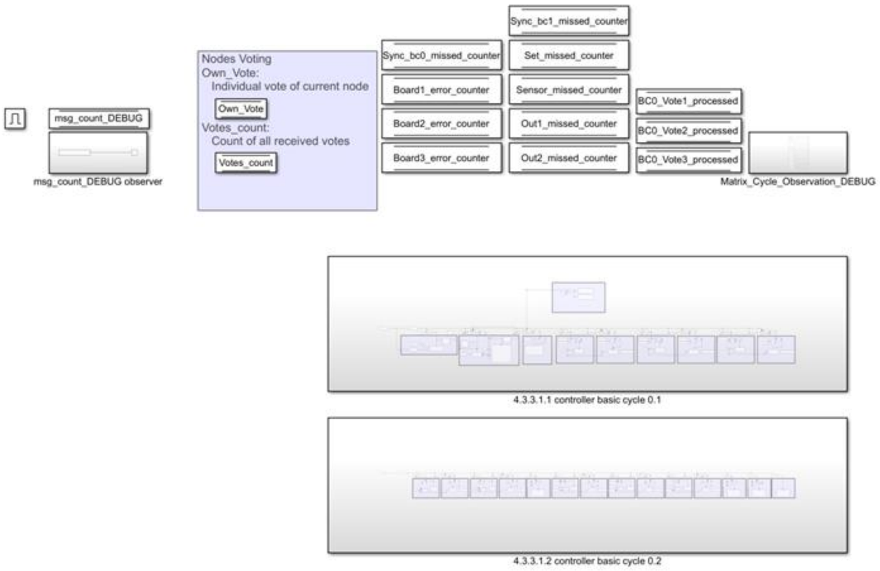
This function is a copy of the provided GitHub, it is able to have different cycle lengths depending on the cycle count. That part isn’t used because the cycle we used runs up until the end.

### 4.3.3 Matrix cycle schedule, after role initialization the cycle is starting



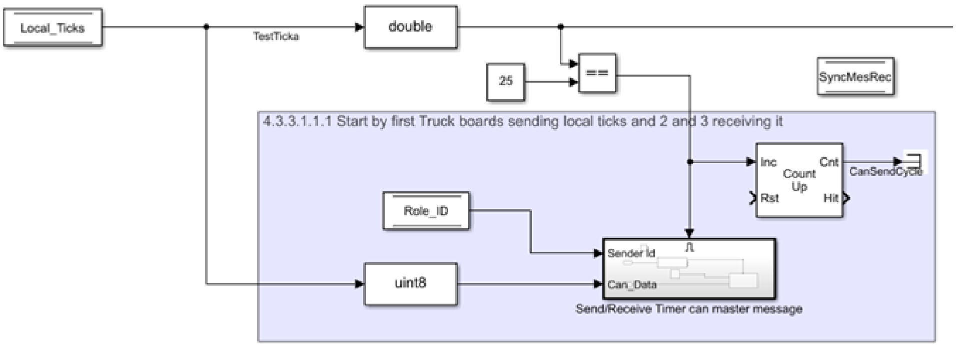
The matrix cycle is separated by board ID the first 3(truck) and last 2(Trailer), also the Board ID shown in report chapther ?? is added here to give the next board its ID. Its placed here so it doesn't give an ID until its initialized.

### 4.3.3.1 Controller matrix cycle for the truck boards, consists of

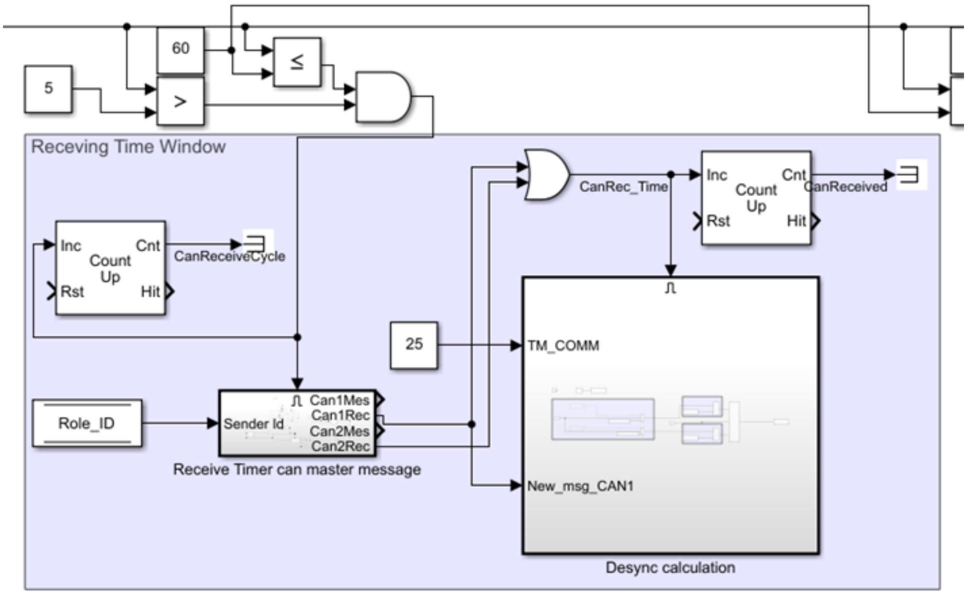


### 4.3.3.1.1 Basic cycle 0.1 (synchronizing and deciding the time master)

4.3.3.1.1.1Sends the local_ticks if the board is the time master (if local ticks == 25) Role Id ==1 when it's the time master.

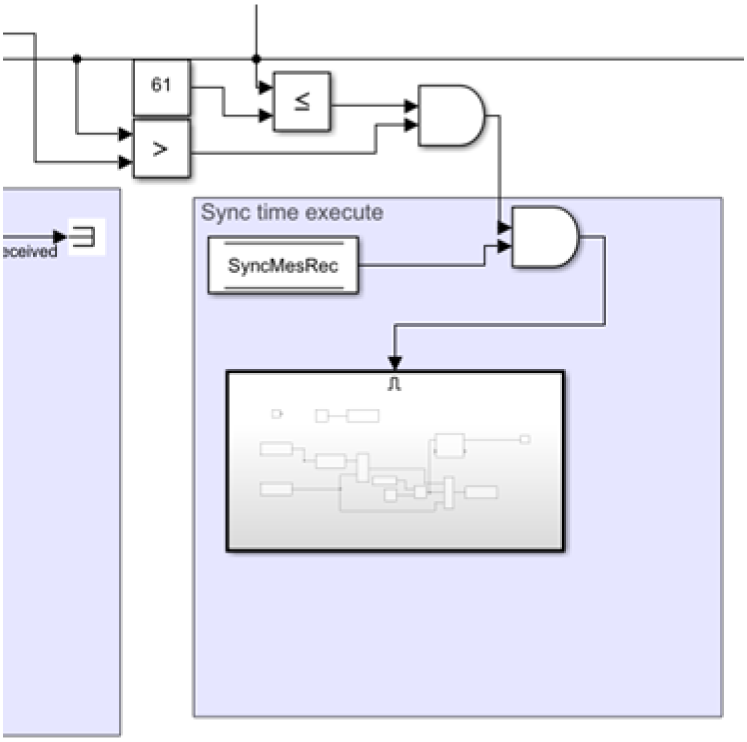

Local_Ticks
TestTicka
double
25
==
SyncMesRec
4.3.3.1.1.1 Start by first Truck boards sending local ticks and 2 and 3 receiving it
Inc
Count Up
Cnt
Rst
Hit
CanSendCycle
Role_ID
uint8
Sender Id
Can_Data
Send/Receive Timer can master message

4.3.3.1.2 Receives the local_ticks if the board isn't the time master(5<LocalTicks<=60)



The “De-Sync” calculation subsystem checks the difference between the time the reference message arrives and the Local Time of the time slave. So, it basically checks the current time of the board and the time which was set for message arrival.

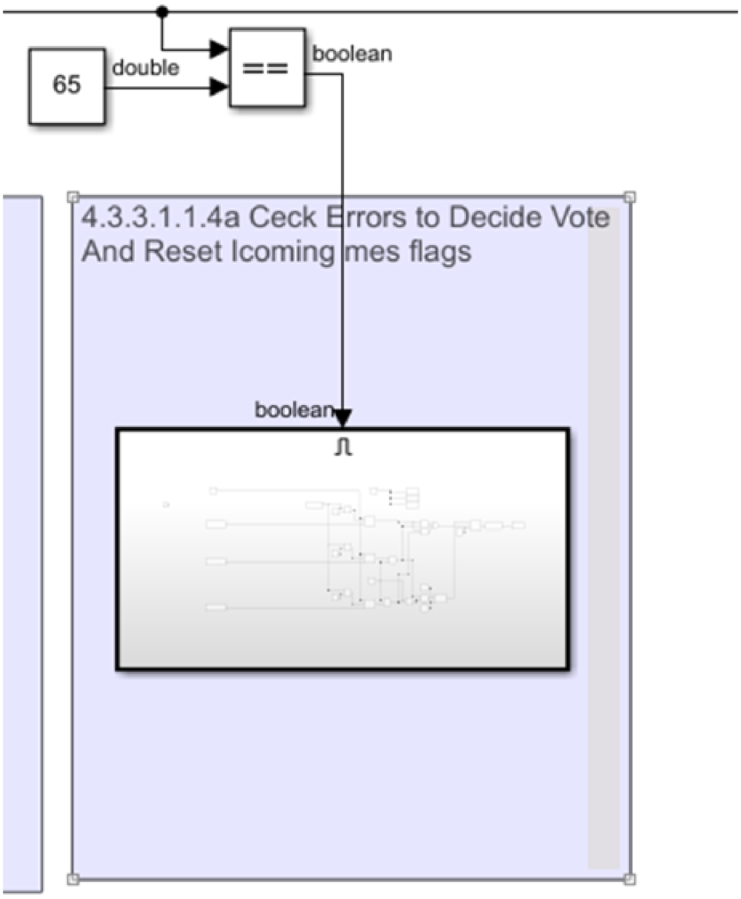
**4.3.3.1.1.3 Syncs the local time to the time of the reference message (if local ticks==61)**



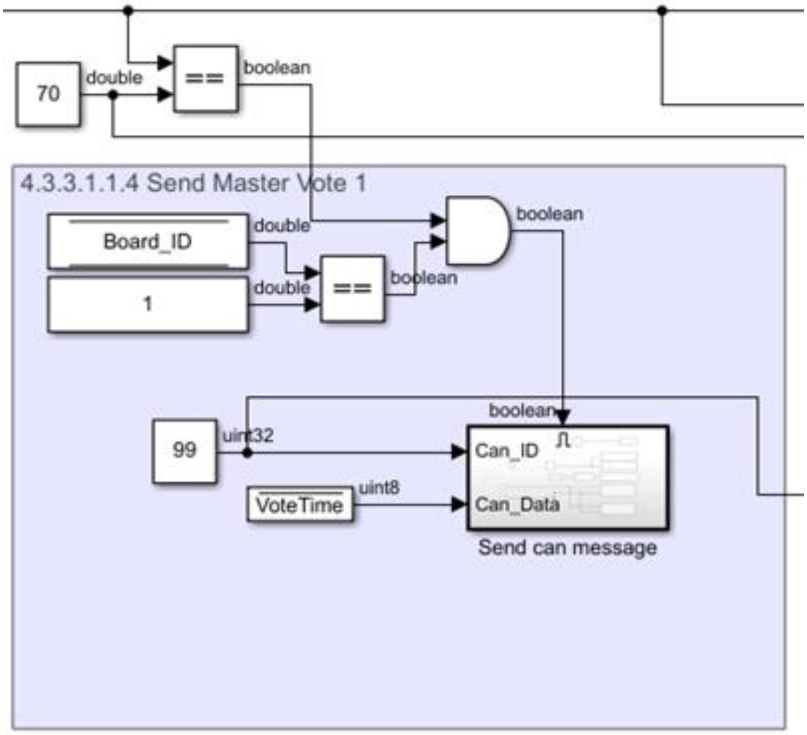
This checks the “De-Sync” variable calculated by the previous block and adds that to the local time.

**4.3.3.1.1.4 A checks the amount of not received messages of truck boards on the can bus, and decides his vote on who should be the time master.**

It also initializes the no message flags; these flags set a flag for each board which is set to zero upon reception of the message.


65
double
==
boolean
4.3.3.1.1.4a Ceck Errors to Decide Vote
And Reset Icoming mes flags
boolean

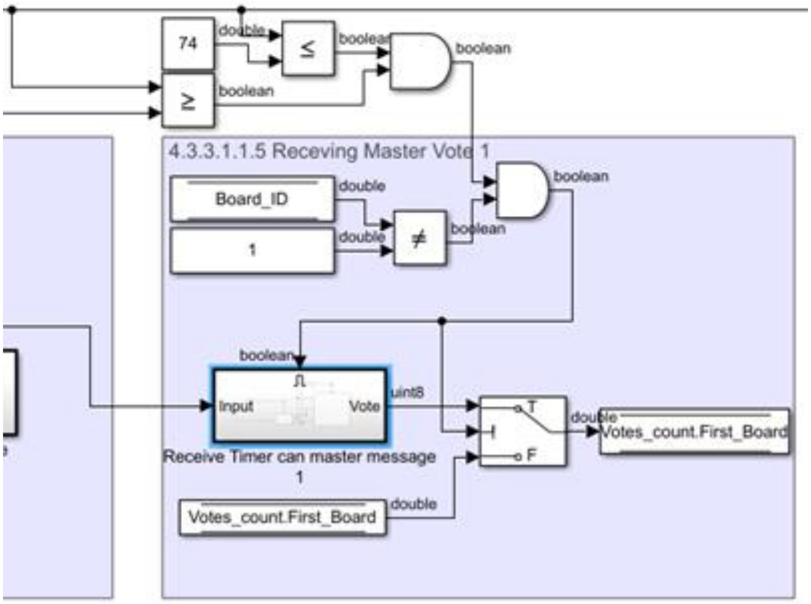
**4.3.3.1.1.4 Sends Truck board 1’s preferred time master (if local ticks==70)**



The blocks 4.3.3.1.1.6 and .8 are the same blocks but used by the other truck boards.

Besides sending these blocks also set the Trailer One ConnectedFlag and Two Trailers Connected Flag to zero, this is done to make sure that when the last trailer is't connected the the flag is zero later in the cycle it can be set to 1.

**4.3.3.1.1.5 Receives Truck board 1's preferred time master (if 70<local ticks<74)**

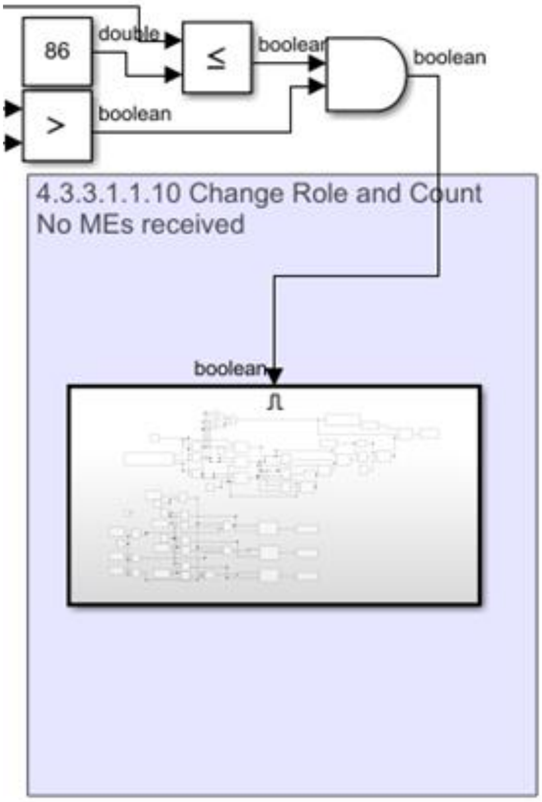


It returns the message of this Vote, additionally if a message on the can bus is received in this window it resets the no message of can as well as the fault counters.

The blocks 4.3.3.1.1.7 and .9 are the same blocks but used by the other truck boards.

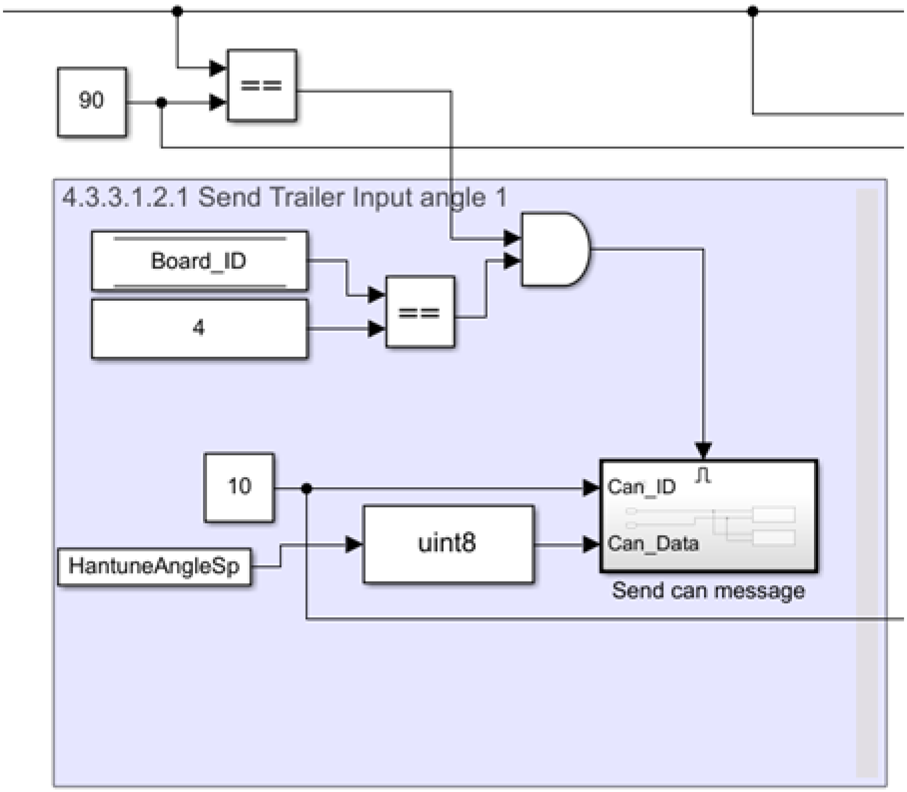
**4.3.3.1.1.10 Changes the role and count if a message flag is still high**

It decides if the same board is going to be time master the next cycle, it changes it based on the incoming votes all three truck boards.



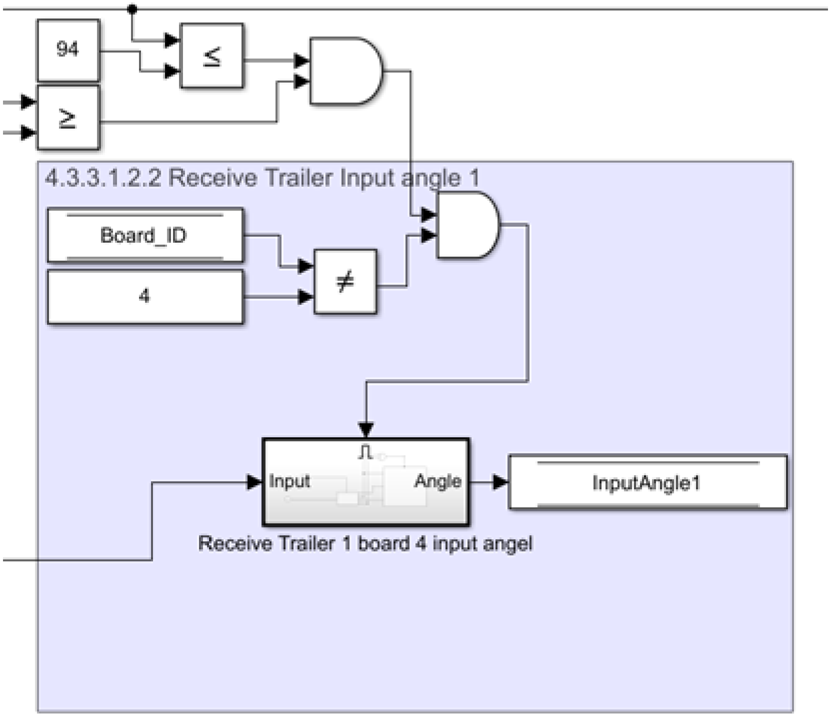
**4.3.3.1.2 Basic cycle 0.2 (Sending/Receiving angles and calculate and send steering angles)**

4.3.3.1.2.1 Sends the angle from the POT on the can bus when it’s the first trailer board



The same blocks are used for 4.3.3.1.2.3

**4.3.3.1.2.2 Receives the angle value from trailer one.**

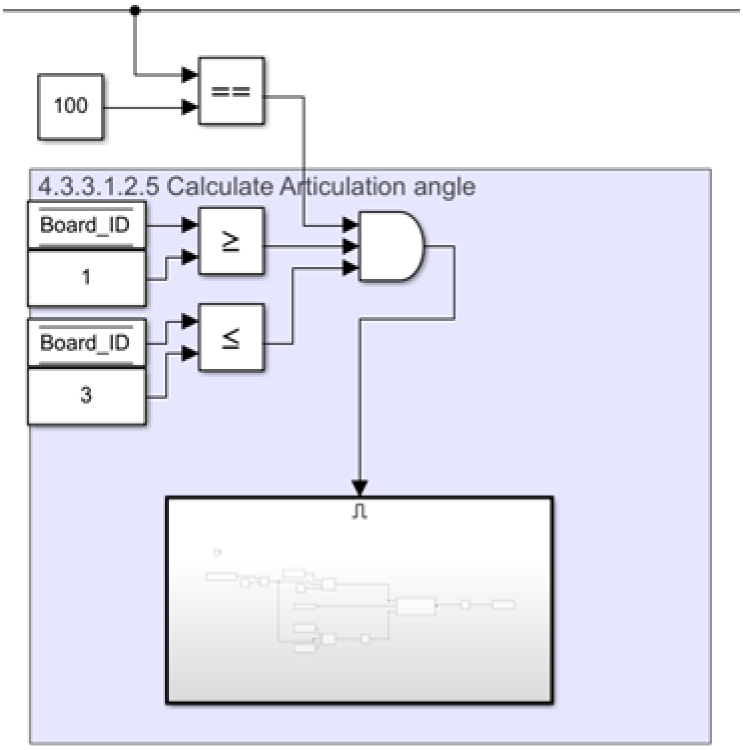

94
≤
≥
4.3.3.1.2.2 Receive Trailer Input angle 1
Board_ID
4
≠
Input
Angle
InputAngle1
Receive Trailer 1 board 4 input angel

Upon Receival it also sets the trailer one Attached flag, indicating its receiving messages from the trailer on the CAN bus.

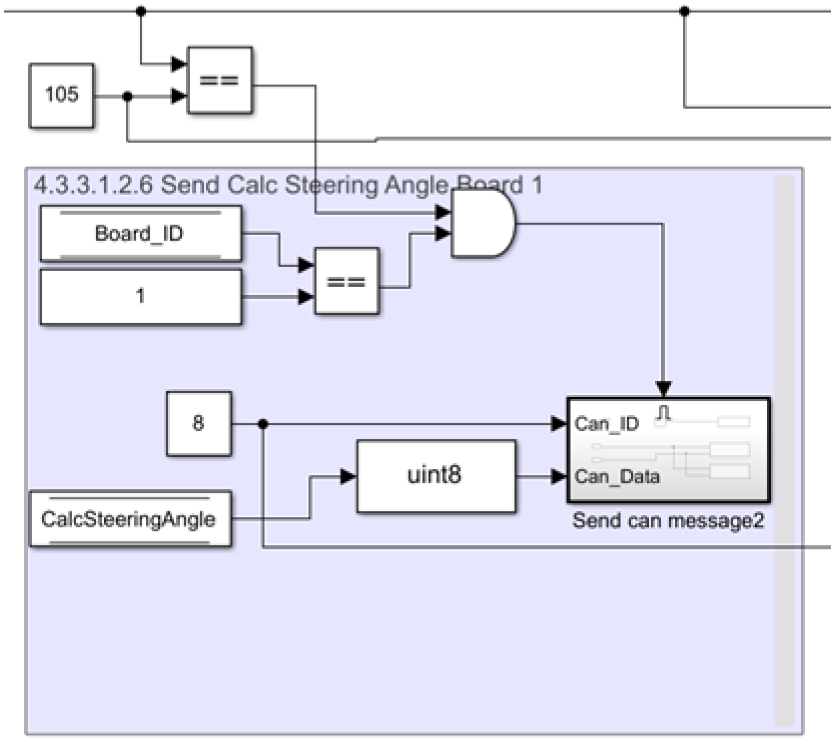
**4.3.3.1.2.3 Reference- 4.3.3.1.2.1**

**4.3.3.1.2.4 The same as 4.3.3.1.2.2** with an adaptation of setting the Two Trailers Connected Flag instead of the trailer one Attached flag if there is an incoming can message from the can id of the second trailer.

**4.3.3.1.2.5 Calculates the steering angle based on the incoming angle**s and the Two Trailers Connected Flag, the IK model is inside this part.



4.3.3.1.2.6 **Send the calculated steering angle on the can buses.**



4.3.3.1.2.8 and .10 use the same blocks for the different boards

**Receive the calculated steering angle from trailer board one**

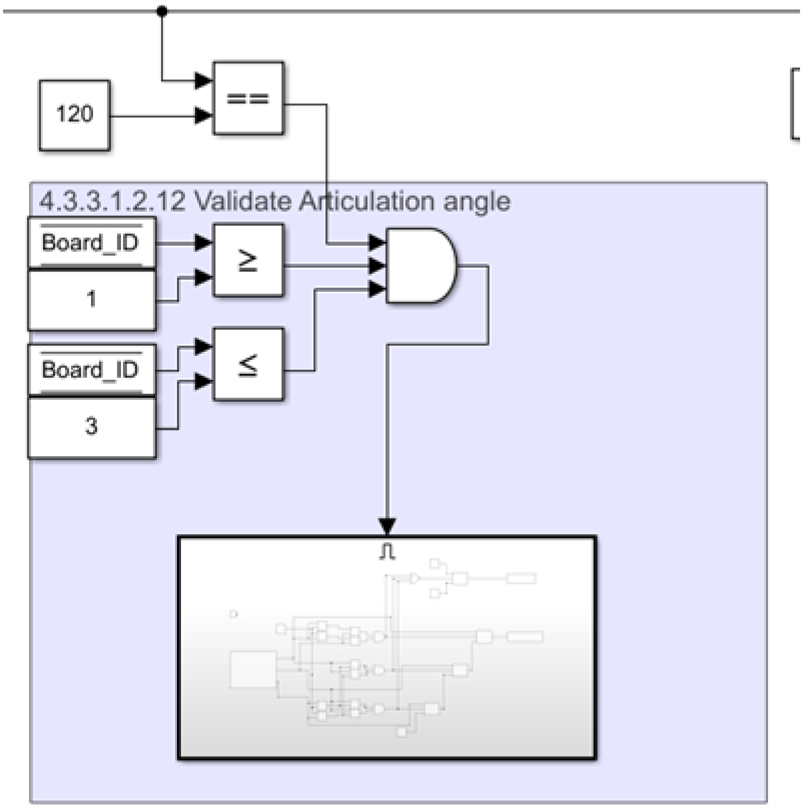
4.3.3.1.2.8 Check .6

4.3.3.1.2.9 Check .7

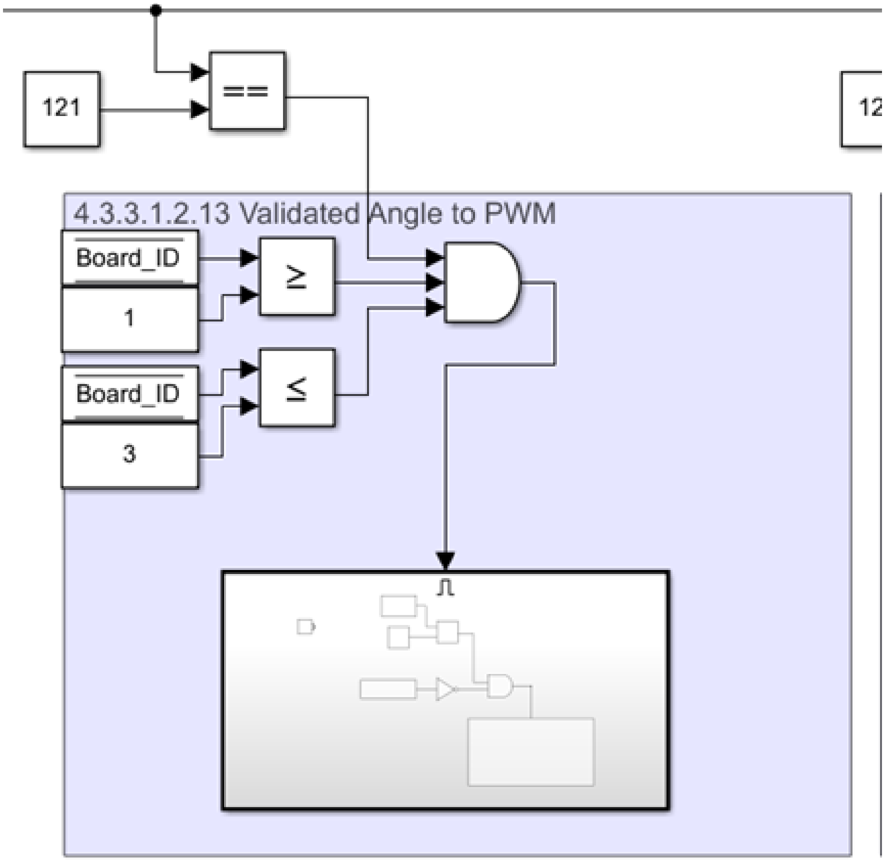
4.3.3.1.2.10 Check .6

4.3.3.1.2.11 Check .7

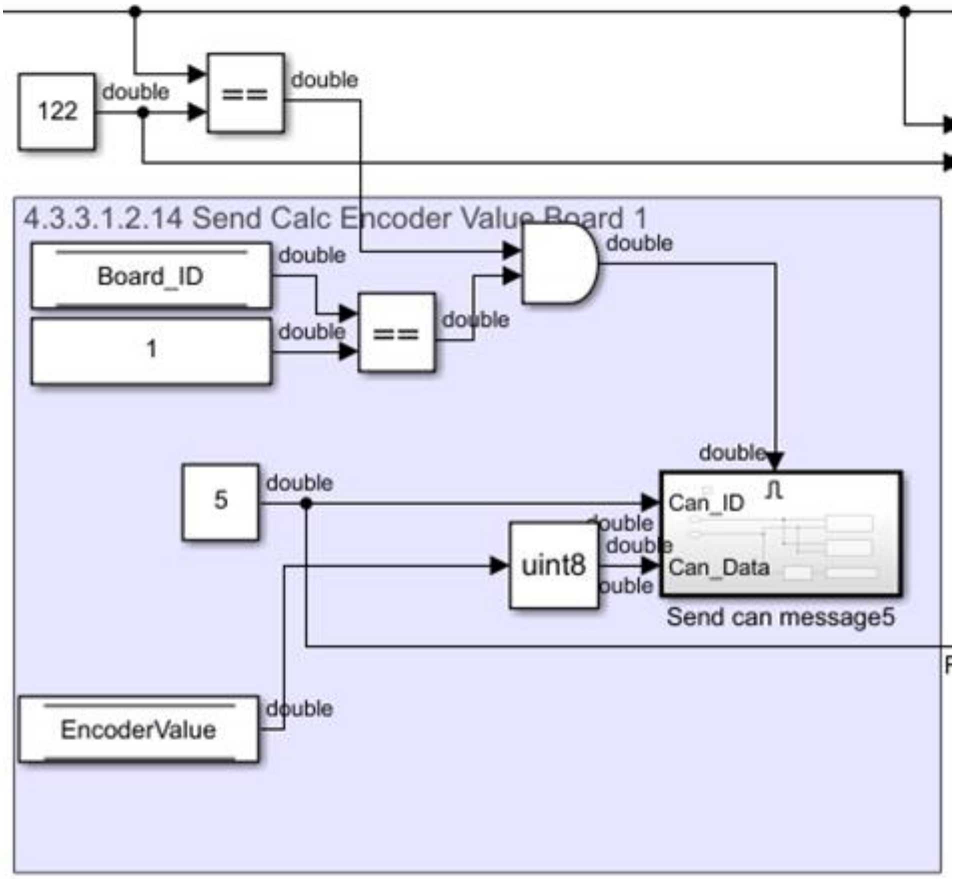
**4.3.3.12.12 Checks the 3 calculated articulation angles** and decide if there close enough to each other and decide which value will be send to the position control.



**4.3.3.1.2.13 Sends the value to the position controller if it's the time master,** and there isn't a vote error (three totally different steering angles calculated)

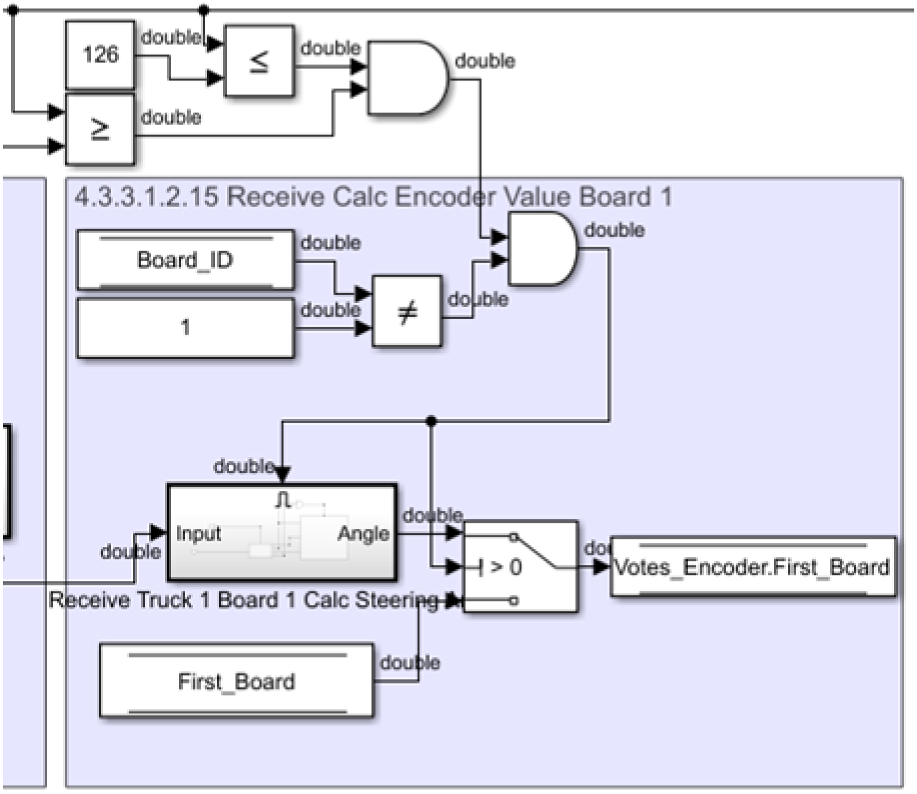


**4.3.3.1.2.14 Sends the Encoder Value Counted by Truck Board 1 On the Can buses**



4.3.3.1.2.16 and .18 use the same blocks for the different boards

**4.3.3.1.2.15 Receives the encoder value of Board 1**



4.3.3.1.2.17 and .19 use the same blocks for the different boards

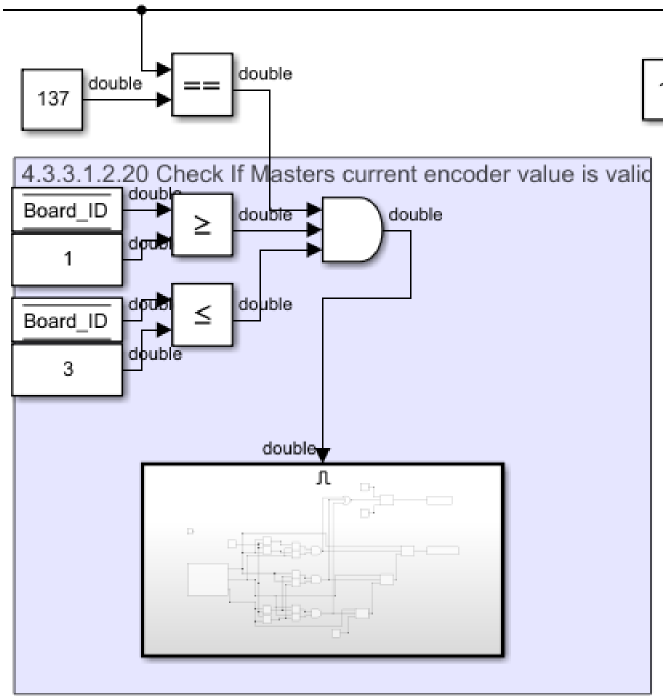
4.3.3.1.2.16 Check .14

4.3.3.1.2.17 Check .15

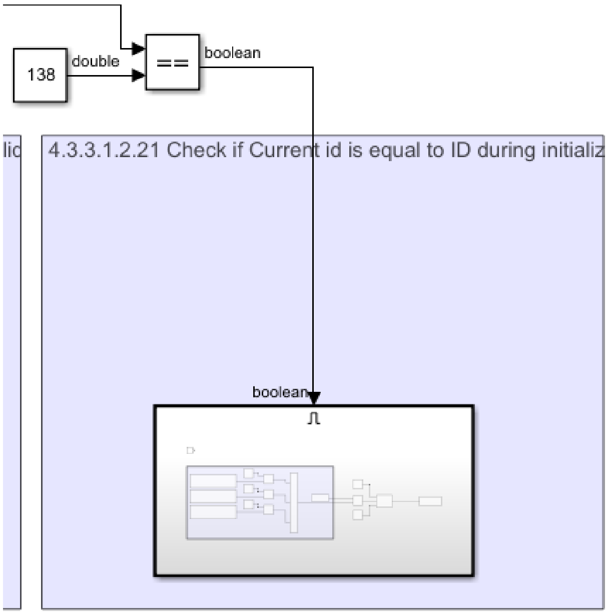
4.3.3.1.2.18 Check .14

4.3.3.1.2.19 Check .15

**4.3.3.1.2.20 Checks if the current encoder value is valid** based on the other boards messages, when its invalid (not equal to one of the others) and the board is the master it indicates an encoder error to the system.



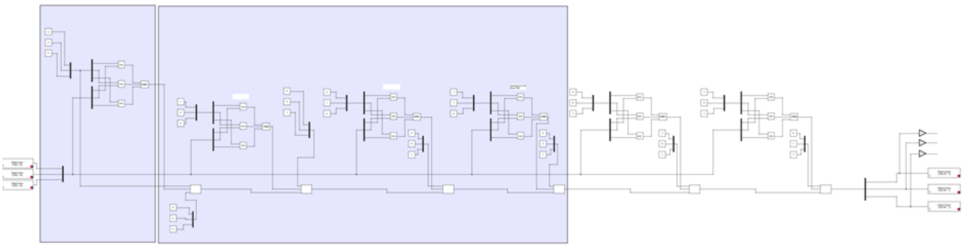
**4.3.3.1.2.21 Checks if the ID has changed during runtime,** if it has changed it indicates an error because the previous board is down. This error is purely informatic since this can only happen on a truck board, The other boards should notice it with use of the other error indicators.


138
double
==
boolean
4.3.3.1.2.21 Check if Current id is equal to ID during initializ
boolean

#### 4.3.3.2 Matrix cycle for the Trailer boards.

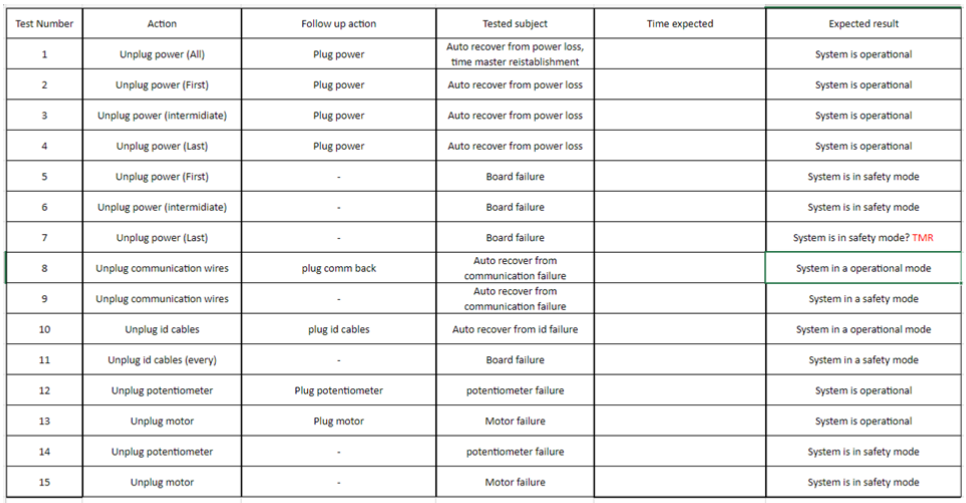
The Matrix cycle for the trailer board is basically the same as the trucks, minus all the voting capabilities. It’s separated because, at first the idea was that the “trailers” could do calculations while the “Trucks” were sending or receiving on the CAN. However, in the current implementation all calculations are done by the “truck” boards, therefore in this case a separate truck cycle was not necessary. However, for future purposes it can be convenient and timesaving.

#### 4.3.3.3 The ID Passthrough



This part determines the output states of the board based on the values of its id inputs. It basically adds 1 to the binary count of the input and places that on the output with use of switches. Its described in chapther ?? of the report

# APPENDIX C: TEST PLAN WITH EXPLANATION



| Test Number | Action | Follow up action | Tested subject | Time expected | Expected result |
|---|---|---|---|---|---|
| 1 | Unplug power (All) | Plug power | Auto recover from power loss, time master reistablishment | | System is operational |
| 2 | Unplug power (First) | Plug power | Auto recover from power loss | | System is operational |
| 3 | Unplug power (intermidiate) | Plug power | Auto recover from power loss | | System is operational |
| 4 | Unplug power (Last) | Plug power | Auto recover from power loss | | System is operational |
| 5 | Unplug power (First) | - | Board failure | | System is in safety mode |
| 6 | Unplug power (intermidiate) | - | Board failure | | System is in safety mode |
| 7 | Unplug power (Last) | - | Board failure | | System is in safety mode? TMR |
| 8 | Unplug communication wires | plug comm back | Auto recover from communication failure | | System in a operational mode |
| 9 | Unplug communication wires | - | Auto recover from communication failure | | System in a safety mode |
| 10 | Unplug id cables | plug id cables | Auto recover from id failure | | System in a operational mode |
| 11 | Unplug id cables (every) | - | Board failure | | System in a safety mode |
| 12 | Unplug potentiometer | Plug potentiometer | potentiometer failure | | System is operational |
| 13 | Unplug motor | Plug motor | Motor failure | | System is operational |
| 14 | Unplug potentiometer | - | potentiometer failure | | System is in safety mode |
| 15 | Unplug motor | - | Motor failure | | System is in safety mode |

Test Number: 1

Action: Unplug power (ALL)

Follow-up action: Plug power

Tested subject: Time master re-establishment and auto recover after power loss

Time expected:

Expected result:

This test measures the system's resilience to a total power loss situation. The system experiences a power loss event when all of the components' power plugs are unplugged. The system should instantly recover and resume operation if power is restored. It should also synchronize time once more with the master clock. It is anticipated that this test will successfully restore time synchronization and make the system functioning once more.

Test Number: 2

Action: Unplug power (First)

Follow-up action: Plug power

Tested subject: Auto recover from power loss

Expected result:

The aim of this test is to evaluate the system's resilience to a power outage incident. The system encounters a power loss event by unplugging the power from the first component. The system should recognize a power loss and start the recovery process as soon as the power is restored. As a result, it is anticipated that the system will automatically recover and resume normal operation.

Test Number: 3

Action: Unplug power (Intermediate)

Follow-up action: Plug power

Tested subject: Auto recover from power loss

Expected result:

The purpose of this test is to confirm how the system reacts when an intermediary component experiences a power loss. The system experiences a power loss condition when one of the intermediate components' power plugs is unplugged. When the power comes back on, the system must notice the outage and start the recovery procedure. The anticipated outcome is that the system will automatically recover and resume functioning without the need for human intervention.

Test Number: 4

Action: Unplug power (Last)

Follow-up action: Plug power

Tested subject: Auto recover from power loss

Expected result:

The system's ability to recover from a power loss event that affects the final component is tested in this test. The system loses power when the final component is unplugged from the power source. When the power comes back on, the system must notice the outage and start the recovery procedure. The anticipated outcome is that the system will automatically recover and resume functioning without human input.

Test Number: 5

Action: Unplug power (one of the First three truck boards)

Follow-up action: -

Tested subject: Board failure

Expected result:

The goal of this test is to locate and address a probable board failure. The system is operational when the power is disconnected from one of the first 3 boards.

Test Number: 6

Action: Unplug power (Intermediate)

Follow-up action: -

Tested subject: Board failure

Expected result:

This test looks at how the system reacts in the event of a suspected board failure. The system enters a safety mode as a preventative measure by unplugging the power from an intermediary board.

Test Number: 7

Action: Unplug power (Last)

Follow-up action: -

Tested subject: Board failure

Expected result: This test measures how the system reacts in the event of a possible board failure. The system is entered by unplugging the power from the previous board.

Test Number: 8

Action: Unplug communication wires

Follow-up action: Plug communication wires back

Tested subject: Auto recover from communication failure

Expected result: System in an operational mode

The system's capacity to automatically recover from a communication breakdown is evaluated by this test. The objective is to assess how the system reacts when communication is established again.

The system needs to immediately acknowledge the recovery and continue regular operating after re-establishing the connection channels. The system should be able to successfully reestablish contact with the impacted components and restart normal operation as a result. The system's faultless recovery from communication failures is ensured by this test.

Test Number: 9

Action: Unplug communication wires

Follow-up action: -

Tested subject: Auto recover from communication failure

Expected result: System in a safety mode

This test measures how the system reacts to an extended communication breakdown

The system should identify the extended communication breakdown and trigger a safety mode as the predicted outcome. In this mode, the system takes precautions to guarantee the security of its parts and its users. To avoid potential risks or damage, it might limit some functionality.

Test Number: 10

Action: Unplug ID cables

Follow-up action: Plug ID cables back

Tested subject: Auto recovery from ID failure

Expected result: System returns to operational mode

This test measures how well the system recovers from an ID failure.

The system should identify and recover from the ID failure automatically after the ID wires are reconnected. It will locate the elements, make the required connections, and restart regular operations effectively. After recovery, the system's ability to manage ID failures and maintain functioning is demonstrated by the fact that it is in operating mode.

This test verifies the system's ability to persevere through ID-related problems and continue continuous functioning.

Test Number: 11

Action: Unplug all ID cables

Follow-up action: -

Tested subject: Board failure

Expected result: System enters safety mode

All ID wires are unplugged during this test to simulate a total failure of the system's component identification mechanism. As a result of the lack of identifying signals, it is anticipated that the system will detect the board failure. The system switches to a safety mode as a precaution to reduce potential risks and stop any additional damage. While the device is in the safety mode, several features might be restricted or disabled.

Test Number: 12

Action: Unplug potentiometer

Follow-up action: Plug potentiometer back

Tested subject: Potentiometer failure

Expected result: System recovers and remains operational

To simulate the failure of this system component, the potentiometer is deliberately disconnected during this test. The system should recognize the potentiometer's repair and continue to function normally when it is connected again. The system's ability to manage component failures and continue to function is shown by the potentiometer failure and subsequent successful recovery.

Test Number: 13

Action: Unplug motor

Follow-up action: Plug motor back

Tested subject: Motor failure

Expected result: System recovers and remains operational

In Test 13, the motor is unplugged to act as a failure of the motor.

The anticipated result is that the system will quickly recognize the motor's repair and continue regular operation after it has been reconnected. The system's capacity to manage motor-related issues and keep its operational functionality is demonstrated by its ability to recover successfully from the motor failure.

Test Number: 14

Action: Disconnect encoder from one of the trucks boards

Follow-up action:

Tested subject: Input failure

Expected result: System is operational.

The anticipated result is that the system is still operational and continues to be operational.

Test Number: 15

Action: Disconnect encoder from two of the truck's boards

Follow-up action:

Tested subject: Input failure

Expected result: System is in safety mode

The anticipated result is that the system switches to safety mode.

Test Number: 16

Action: Unplug POT from one of the truck’s boards

Follow-up action:

Tested subject: Analog Input failure/calculation failure

Expected result: System is operational.

The anticipated result system is operational and continues to be operational.

Test Number: 17

Action: Unplug POT from two of the truck’s boards

Follow-up action:

Tested subject: Analog Input failure/calculation failure

Expected result: System is in safety mode

The expected outcome is the system switches to safety mode.