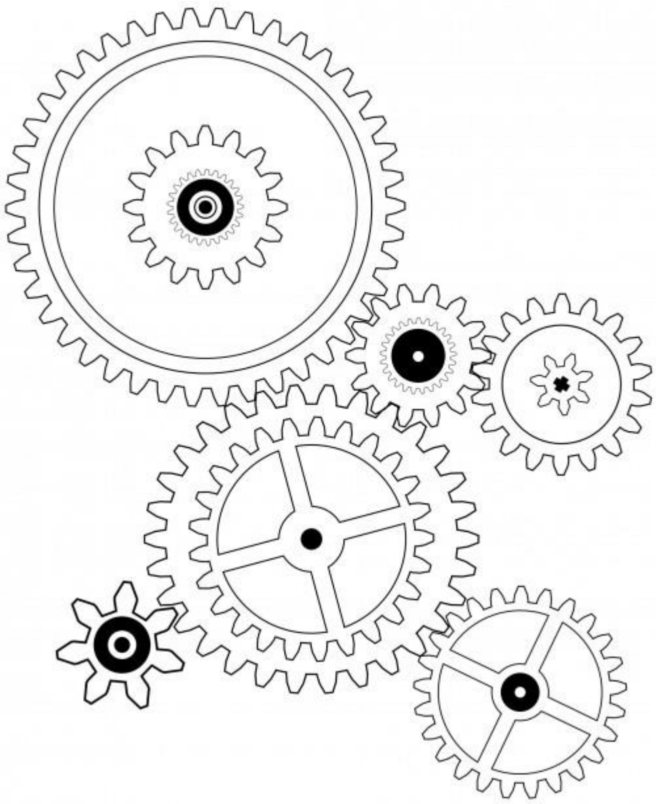
13 October 2017

System Design Template

Smart Quarter Vehicle

Status: Draft



Name	Student nr
William Marcelissen	507428
Chris van der Spek	561633
Gerran Jansma	521675
Teun van den Burg	530770

Table of Contents

1 Intro....3

2 General Overview and Design....4

2.1 General Overview....4

2.2 System Context....4

2.3 Use Cases....4

2.3.1 System Components....5

2.4 Assumptions....5

2.5 Constraints....5

2.6 Risks....6

3 Design Considerations....6

3.1 Problem Statement....6

3.2 Goals....6

4 System Architecture and Design....7

4.1 Logical View....7

4.2 Hardware Architecture....8

4.3 Software Architecture....9

4.4 Internal communication Architecture....9

5 System Detailed Design....10

5.1 Hardware Detailed Design....10

5.1.1 Chassis....10

5.1.2 Module....11

5.2 Software Detailed Design....12

6 External Interfaces....14

6.1 Interface Architecture....14

6.2 Interface Detailed Design....15

Appendix A – References....16

Appendix B – Diagram Conventions....17

1 Intro

During the project a Smart Modular Vehicle (SMV) will be designed, developed and built. The Smart Modular Vehicle consists of a chassis and multiple identical and interchangeable steering and driving modules. The chassis and wheel dimensions will be scaled 1:14 with respect to a real truck. This SMV will be able to follow a predefined path by communicating with a master computer which has an overview of the predefined path via a camera hanging or hoovering above the path.

The SMV will be designed, developed and built for HAN Automotive Research. The SMV will provide them a modular platform which can simulate multiple truck chassis outlay. HAN AR can use this SMV for future and ongoing projects like the “Drone assisted truck navigation project”.

This system design document will first provide a general overview of the system and the goals of the project. It also contains a more detailed architecture and design of the system and all of its subsystems. These, more detailed, designs will both discuss hardware and software. The document will conclude with the design and architecture of the interface which will allow communication between the different subsystems. In this document the system will be visualised in diagrams to give the reader a better understanding of how all subsystems are connected.

2 General Overview and Design

2.1 General Overview

The system exists of the platform and at least four modules. The microcontroller of the system is placed on the platform and controls the steering, accelerating and braking of the four modules. The microcontroller communicates with the master computer which sends a signal that contains the predefined path.

2.2 System Context

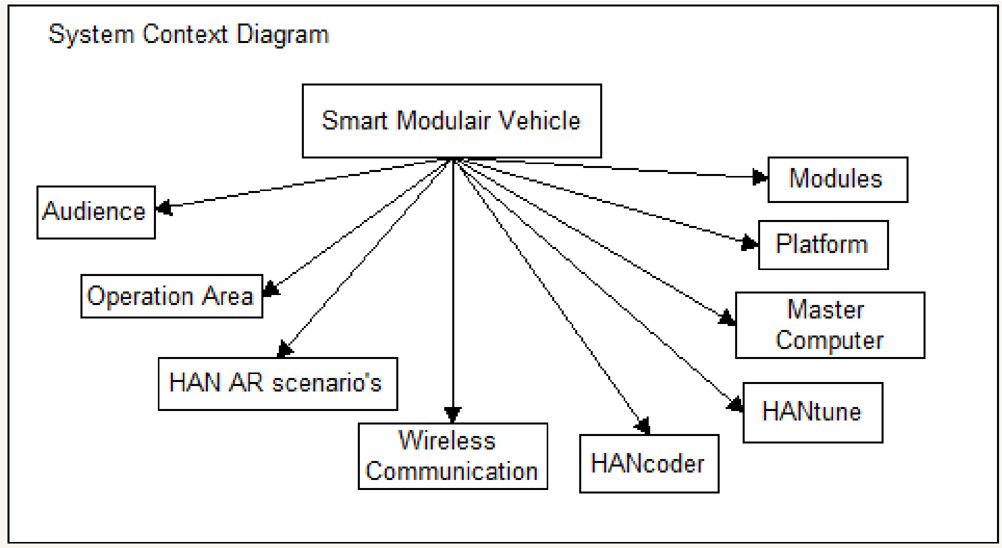


Figure 1 Contextual Diagram

2.3 Use Cases

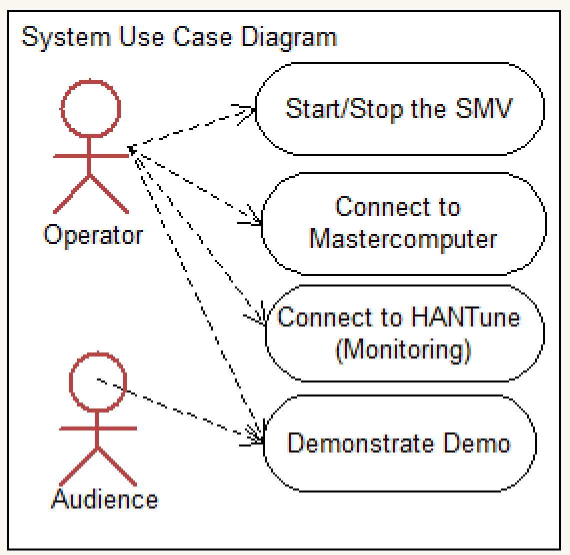


Figure 2 Use Case Diagram

2.3.1 System Components

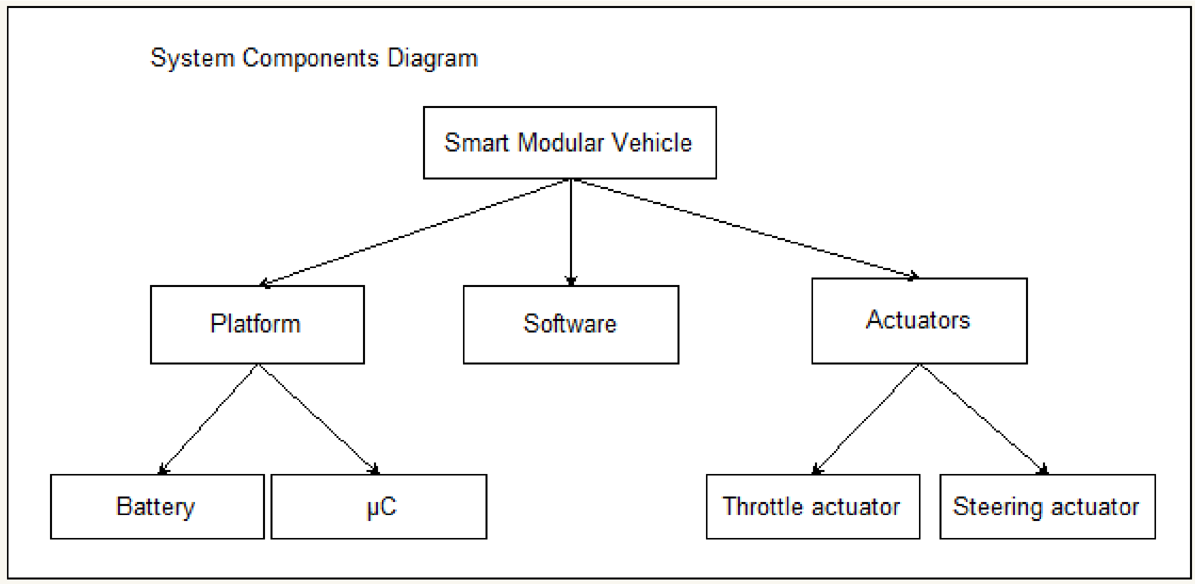


Figure 3 Component Diagram

2.4 Assumptions

- For creating and sending the predefined path to the SMV, the existing software developed by previous students will be used.
- The main software used for programming and debugging will be Matlab, Simulink, HANCoder and HANTune
- The end user can expect a Smart Modular Vehicle that is capable of driving the predefined path.
- The Smart modular Vehicle exist at least out of 4 modules (2 driving and 2 steering) but a different setup with more modules is possible.

2.5 Constraints

- The algorithms of the previous project group has to be investigated before changing and implanting is possible.
- Before the algorithms can be implemented, the platform with 4 modulus must be in prototype phase.
- Because the SMV is a prototype and there is a limited time frame, the end-user environment is not a high priority.

2.6 Risks

Because the project takes only roughly 3 months. The main focus will be the electrical and programming part of the system. Therefor the design of the SMV will be simple but sturdy, so no fancy boltless plug and play system for the modules will be developed. In this way there is more time that will be spent on programming and improving the system. What is more in the scope of this project.

3 Design Considerations

3.1 Problem Statement

The current vehicles are limited in their specifications and dimensions. Fixed truck setups and imprecise steering & driving makes these RC trucks undesired to work with.

As stated above, the current RC trucks are not precise enough in steering and driving. This concludes in a lot of steering actions that can be avoided with a better system. Also is HAN-AR limited in setup options, because of the fact that the truck setups are depending on the purchased vehicles.

3.2 Goals

The goal is to create a platform which makes use of wheel modules. This platform must drive a path that is predefined by the master computer.

- Within this main goal, a sub-goal is to create modular wheel units which receives input from the master PC.
- Another goal is to create a chassis that can accommodate at least 4 modules. The chassis needs standardised connections to connect with the modules.
- Also the modules will include (HANcoder) software to work properly.

4 System Architecture and Design

This chapter describes the different system aspects that could be used for the SMV. First thing that will be described is a logic view of the different system aspects that are going to be used on the SMV. The hardware architecture of the SMV will be described in different diagrams, after the logic view. The next step is also about architecture but then about the software that needs to be made. The last subject in this chapter is the internal communication of the SMV. All of this will be shown in diagrams and an explanation of the diagrams will be given.

4.1 Logical View

A logic view of a product is what it can do after it is turned on and off. The logical view of the SMV is presented here below.

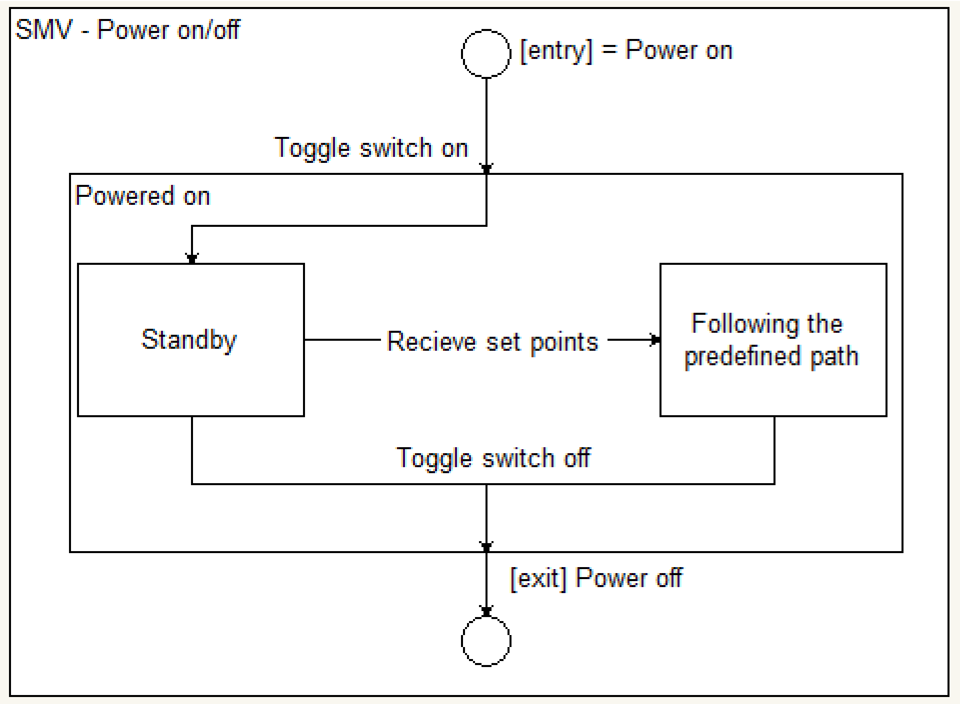


Figure 4 Logical view

4.2 Hardware Architecture

The hardware Architecture will be divide by the chassis and the module. The diagrams of the hardware architecture are shown below.

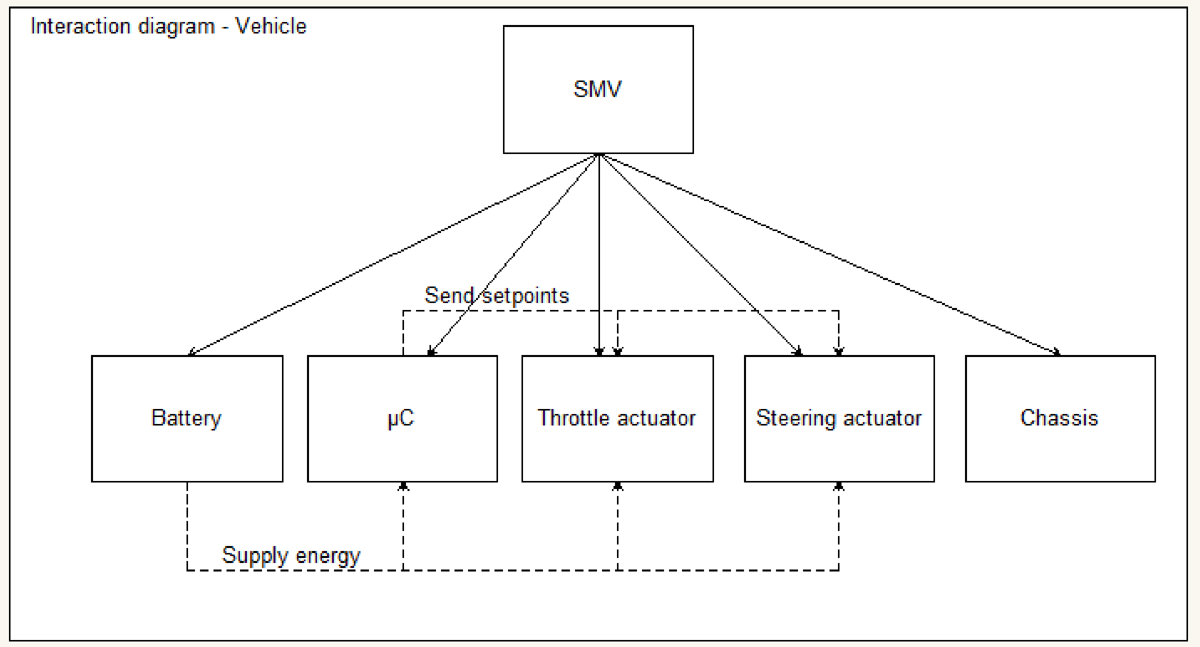


Figure 5 Interaction diagram of the vehicle

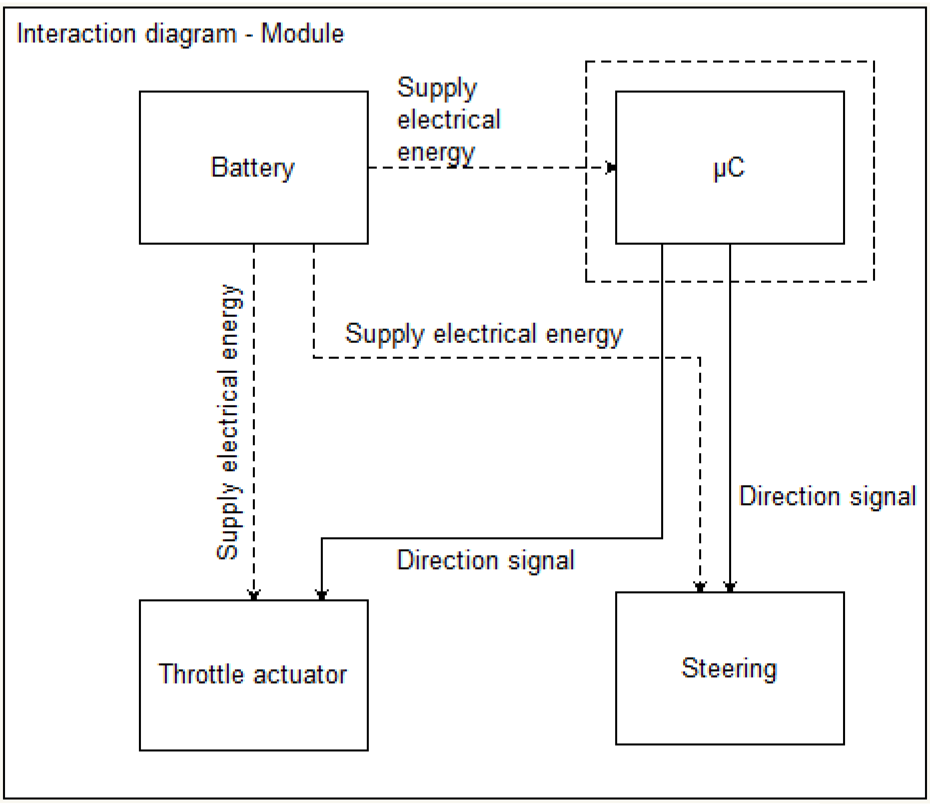


Figure 6 Interaction diagram of the module

4.3 Software Architecture

There needs to be a wireless connection between the master computer and the microcontroller. The microcontroller receives steering and throttle setpoints from the master computer. These setpoints have to be converted to a signal for the throttle and steering actuators of each individual module. Converting a single steering setpoint to multiple independent steering angles according to Ackermann is also done by the microcontroller.

4.4 Internal communication Architecture

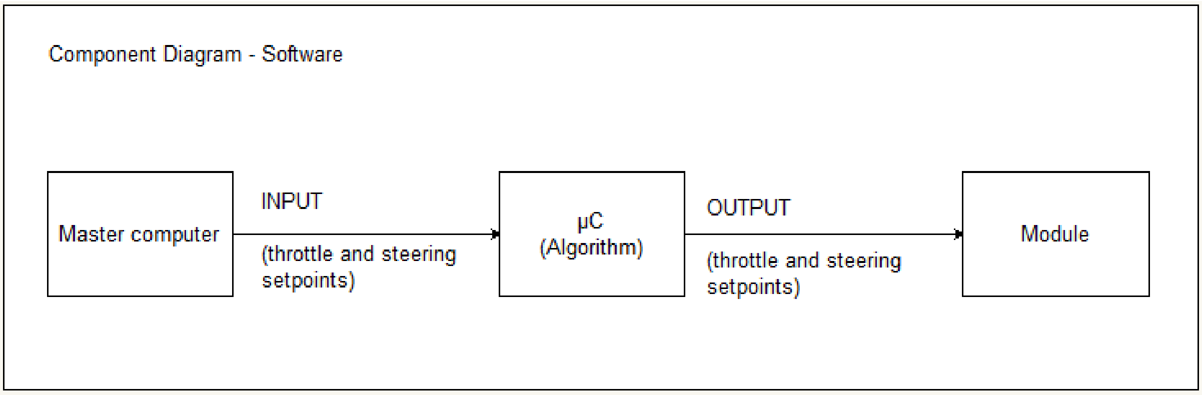


Figure 7 Wireless Interaction Diagram

The communication of the SMV will be wireless from master computer to mainboard. The diagram of the wireless communication is shown in figure 7.

5 System Detailed Design

5.1 Hardware Detailed Design

The SMV will contain both purchased components as components designed and built by the project group. To have a good overview of the hardware designed some components will be elaborated more detailed.

5.1.1 Chassis

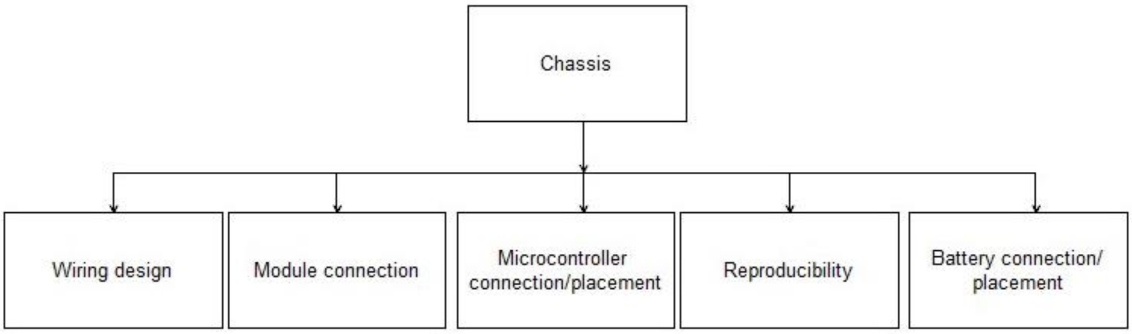


Figure 8 Chassis functions

The chassis is a large hardware component of the system. This component will carry all other subsystems, therefore it is important to have a more detailed look at the chassis.

Module connection
In the set of requirements is stated that de chassis must be able to carry at least four modules. These modules must be interchangeable. Therefore the connection between chassis and module must be modular. Another requirement states that the connection must be able to withstand the driving forces caused by driving the predefined path.

Mainboard connection/placement
The mainboard will be controlling all the modules and receive and process the signals coming from the master computer. The placement of this mainboard will be crucial for the design of the wiring.

Reproducibility
Because HAN AR is planning on using the SMV in different future projects, it is important for them to make more SMV's. Therefore the design of the chassis should be accessible and adjustable to simulate a different vehicle. Also the production of the chassis should be easy to repeat.

Battery connection/placement
Being a heavy component the placement and connection of the battery is important for the design of the chassis.

5.1.2 Module

The modules are the most complex hardware components to be designed and built. A lot of functionalities of the SMV are dependent on the module design and development. Because of its importance the module design will be further specified.

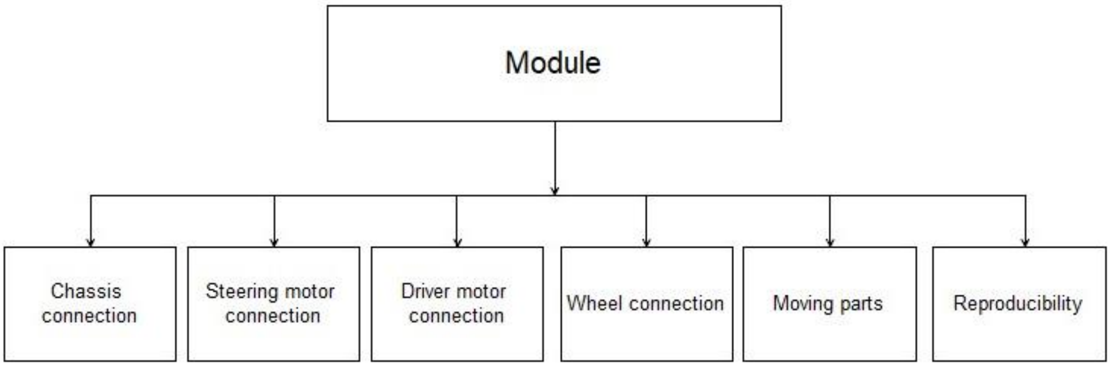


Figure 9 Module functions

Chassis connection

This connection must be able to withstand the driving forces while driving the predefined path. See 5.1.1 Module connection.

Steering motor connection

The steering motor will be connected to the module and the rest of the drivetrain. The connection to the module has to be rigid to set and hold a precise steering angle.

Driver motor connection

The driver motor needs a rigid connection to be able to deliver the driving force.

Wheel connection

HAN AR requires the use of existing 1:14 RC truck wheels.

Moving parts

Because of the steering capability of the modules there will be moving parts inside the module. This puts extra pressure on the tolerances of the hardware. While designing the modules this is an important fact to keep in mind.

Reproducibility

The module must be easy to reproduce to extend the SMV in future projects. This means accessible design and easy production methods.

5.2 Software Detailed Design

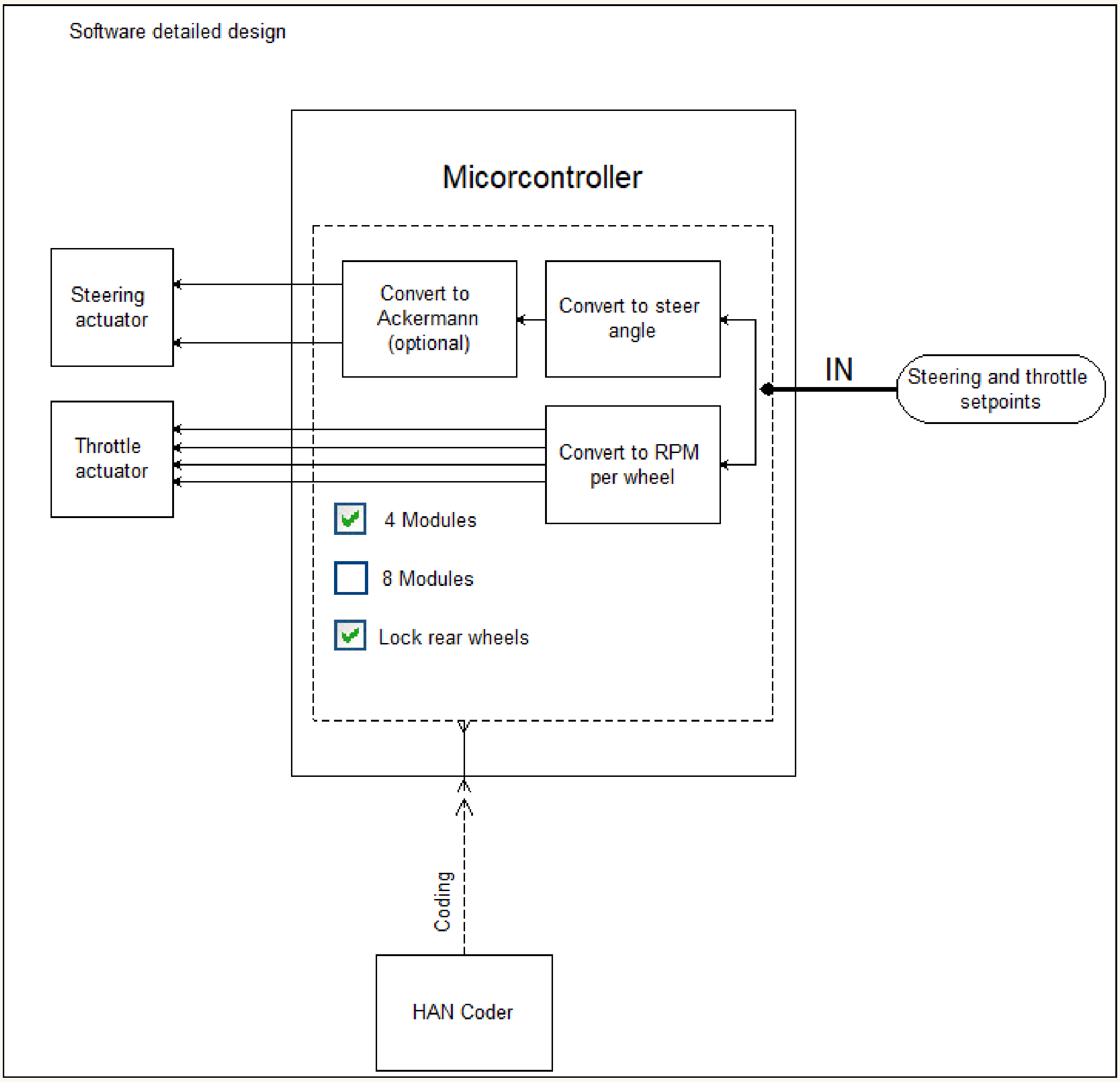


Figure 10 Software detailed design

The SMV will be communicating with the existing master computer, this computer will send out a signal to the SMV containing PWM values. The software programmed in de microcontroller will translate this to driving and steering the vehicle.

HAN Coder

The programming will be programmed using the model based HAN Coder software. Simulink blocks represent different functionalities. By placing these blocks a logical block diagram can be created. The C code will automatically be created by HAN Coder and can be flashed to the microcontroller.

Converting signals

The mainboard will receive its input from the master computer. The microcontroller has to convert this signal into an actual driving signal to the driver and steering motors. The driving signal the microcontroller has to transmit depends on the type of motors that will be used. The SMV might also get Ackermann steering behaviour, this will result in extra calculations of the microcontroller and different steering angles for each wheel.

Set up

HAN AR wants to use the SMV with a varying amount of modules connected. With the 4 wheel setup it should also be possible to lock the rear wheel which will prevent them from steering. This way a two wheel steering truck can be simulated. When changing the set up the driving signal also has to change. Therefore it is important that the set up changes in the software are straightforward and easy to make.

6 External Interfaces

Looking at the external interfaces used within this project, the main external interface is the Master computer. This project works with a predefined path that the Master computer sends in PWM signals to the SMV. The path can be defined in the master computer, or the path can be defined by the drone. Either way the master computer has coordinates of the path and sends it to the SMV.

6.1 Interface Architecture

Referring to the SCD, the master computer is a part of the whole system. The Master computer sends the PWM signals to the chassis. The Master computer receives the X, Y and angle of the vehicle via the drone. The logarithm calculates the PWM values to get the desired position of the vehicle.

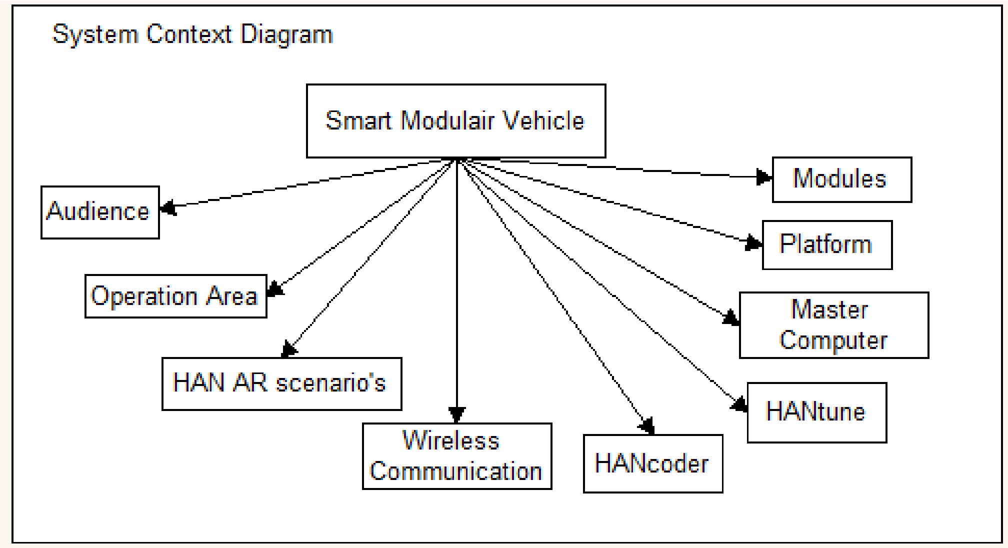


Figure 11 System Context Diagram

6.2 Interface Detailed Design

Looking at the external interfaces, the master computer is the only and most important external interface.

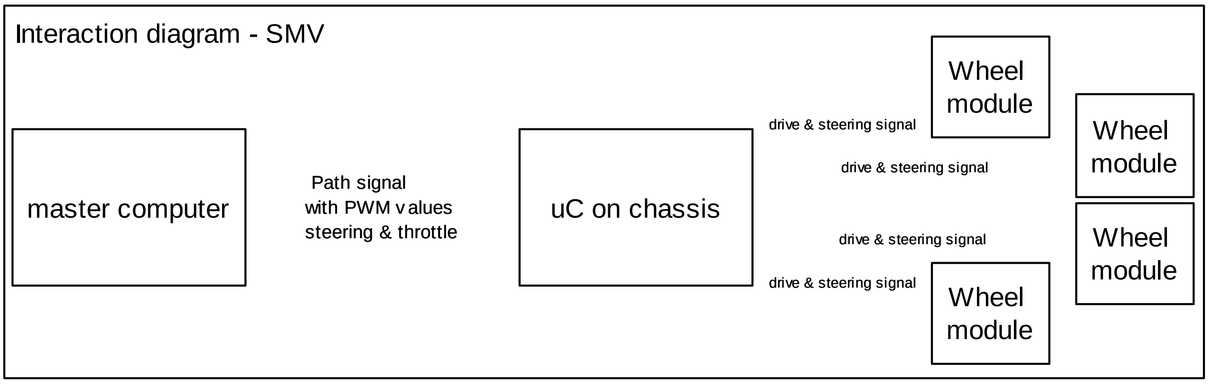


Figure 12 Interaction diagram

Data Direction (Systems viewpoint)	What is interacted?	Details
Send	XCP messages over Zigbee	4 bytes: Start byte, Steering byte with PWM value, throttle byte with PWM value, stop byte

It is possible that the Drone can send the predefined path to the master computer, in that case the Master computer receives the following data:

Data Direction (Systems viewpoint)	What is interacted?	Details
receive	XCP messages over Zigbee	4 bytes: Start byte, Byte Marker 1 (X, Y, angle marker), Byte Marker 2(X, Y, angle marker), stop byte

Appendix A – References

Diagrams:

Figure 1: Contextual Diagram ....**Fout! Bladwijzer niet gedefinieerd.**
Figure 2: Use Case Diagram ....**Fout! Bladwijzer niet gedefinieerd.**
Figure 3: Component Hierarchy...**Fout! Bladwijzer niet gedefinieerd.**
Figure 4: State Machine: Balancing the Suitcase ....**Fout! Bladwijzer niet gedefinieerd.**
Figure 5: State Machine: Connection to HANtune ....**Fout! Bladwijzer niet gedefinieerd.**
Figure 6: Interaction Diagram - Power Supply................................**Fout! Bladwijzer niet gedefinieerd.**
Figure 7: Interaction Diagram - HANtune Connection via WiFi ....**Fout! Bladwijzer niet gedefinieerd.**

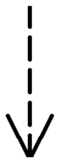
Pictures:

Picture 1: Balancing Suitcase Demo..**Fout! Bladwijzer niet gedefinieerd.**

Appendix B – Diagram Conventions

These relations are a suggestion on how to use the diagrams. Feel free to add new features to your diagrams. However, if new features need to be added, please look at existing conventions in order to avoid confusion. At HAN-AR some knowledge and experience is present regarding formal modelling languages.

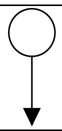
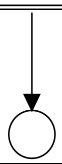
Component Diagram:

Balancing Suitcase Demo	Component (Also used in Interaction Diagrams)
	Component / Subcomponent relation (Also called: Parent/Child, or Super/Sub component)

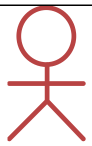
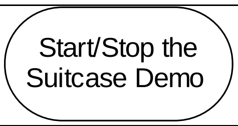
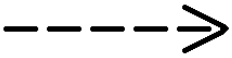
Interaction Diagram:

DC-Motor	Component (from component diagram)
←Apply Voltage [V]—	Item flow (Anything that flows, e.g. Data, Energy, Fluid, Air)

State Machine Diagram:

	Entry point
Standby	State
	Exit point
←Toggle Switch "Sw1"→	State transition
[entry] = Power on	Guard, entry or exit criterium

Use Case Diagram:

	(Usually) Human interacting entity. Interaction may be both ways. Usually entails e.g. Operator, End-User, Maintenance staff, Audience, etc.
	A use case. Could also have descendants in order to group use cases.
	Interaction relation
	Descending relation