***Before you start working in this template document***

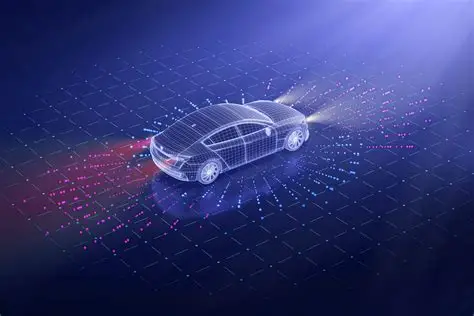
*Several features of MS Word are used in this document. These features are briefly explained here.*

* ***Sections*** *are used to divide and format documents. In this document, each chapter is a section. The transition from one section to another can be made visible by turning on the Show All option ( of CTRL+\*)*
* *The text in the* ***footer*** *(at the bottom of the pages) can be modified by double-clicking it.*
* *The chapter* ***Contents*** *is automatically generated based on the chapter layout. Select the table of contents and press F9 to generate the table of contents again.*
* *For the* ***References*** *chapter, the option to specify sources within MS Word have been used. These sources can be found under References > Manage sources. Adding or removing new sources to the current list automatically generates an updated References chapter.*
* *To align text nicely, tables with non-visible borders have been used here and there. To make non-visible edges of tables visible, select Table Layout > Show Gridlines (Afbeelding met tekst, Rechthoek, schermopname, ontwerp

  Automatisch gegenereerde beschrijving).*

**Embedded Systems Engineering**

**Product report**



*<robot car ese 1>*

**Embedded Systems Engineering  
Academy Engineering and Automotive  
HAN University of Applied Sciences**

Authors

|  |  |
| --- | --- |
| ***<2129345.>*** ***<2173593.>*** ***<2170792.>*** ***<2169996.>*** | ***<Nasib Aljabban>*** ***<Bauke Bergsma>*** ***<Ruben Balduk>*** ***<Mario Filipov >*** |

Date

|  |
| --- |
| ***09-2025*** |

Version

|  |
| --- |
| **0.1** |

# Revisions

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **When** | **Who** | **What** |
| 0.1 | 18/09/2025 | Bauke and Nasib | Introduction and functional design |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Preface

**NA**

*Looking back on the course of the project, how did the project group experience the project, what did the group learn from it, what will the group do better next time; in short, the preface can contain all kinds of personal reflections on the project and its course.*

# Summary

NA

*Description of the starting points, goals to be achieved, what was and was not achieved,* ***results achieved****; the summary should give an overall impression of the whole assignment and is maximum 1 A4.*

# Contents

[Revisions 3](#_Toc168479456)

[Preface 4](#_Toc168479457)

[Summary 5](#_Toc168479458)

[Contents 6](#_Toc168479459)

[1 Introduction 7](#_Toc168479460)

[1.1 Reason 7](#_Toc168479461)

[1.2 Objective 7](#_Toc168479462)

[1.3 Report structure 7](#_Toc168479463)

[2 Functional design 8](#_Toc168479464)

[2.1 Functional specifications 8](#_Toc168479465)

[2.2 Technical specifications 9](#_Toc168479466)

[2.3 User interface 10](#_Toc168479467)

[3 Technical Design 11](#_Toc168479468)

[3.1 Architecture 11](#_Toc168479469)

[3.2 Interfaces 12](#_Toc168479470)

[3.2.1 Power supply 12](#_Toc168479471)

[3.2.2 Microcontroller – Sensor 12](#_Toc168479472)

[3.2.3 Microcontroller – Actuator 12](#_Toc168479473)

[3.2.4 Microcontroller – Communication – PC driver – App 13](#_Toc168479474)

[3.3 Software 13](#_Toc168479475)

[4 Realisation 14](#_Toc168479476)

[4.1 Hardware 14](#_Toc168479477)

[4.2 Software 14](#_Toc168479478)

[5 Testing 15](#_Toc168479479)

[6 Conclusions and recommendations 16](#_Toc168479480)

[7 References 17](#_Toc168479481)

[Appendix A 18](#_Toc168479482)

[Appendix B 19](#_Toc168479483)

[Appendix n 20](#_Toc168479484)

# Introduction

## Reason

This product was requested to the QTC company to help in development for self-driving vehicles. we aim to revolutionise road travel and prevent traffic jams by creating cars that do not need human drivers. For this goal our car needs to be capable of driving on its own without crashing into obstacles.

## Objective

The objective of this project is to create a small prove of concept of a self-driving car, capable of steering by a human, following a set line and avoiding obstacles without human intervention. For these goals the car first needs to be able to drive with variable speed, smooth turns, detecting distant obstacles and following lines on the floor.

## Report structure

###### Chapter 2 Functional Design :

provides a complete functional design for the Robot-car project, created in consultation with the client. It defines what the product will do, not how it will do it. The specifications are detailed using the SMART criteria and prioritized with the MoSCoW method.

# 

# Functional design

## Functional specifications

|  |  |  |
| --- | --- | --- |
| # | MoSCoW | Description |
| **F1** | **M** | The Robot-car must be able to drive. |
| F1.1 | M | The Robot-car can drive forward and backward. |
| F1.2 | M | The Robot-car can turn left and right. |
| F1.3 | M | The Robot-car can drive with a variable speed. |
| F1.4 | M | When turning, an LED will indicate the direction. |
| **F2** | **M** | The Robot-car can drive autonomously. |
| F2.1 | M | The Robot-car can change direction or move backward upon colliding with a wall or another object. |
| F2.2 | M | The Robot-car can avoid bumping into a wall or object by changing direction or moving backward. |
| **F3** | **M** | The Robot-car can drive in slave mode. |
| F3.1 | M | The Robot-car can follow a line. |
| F3.2 | M | The Robot-car can follow a predecessor when following a line. |
| F3.3 | M | The Robot-car can stop when a predecessor stops. |
| **F4** | **S** | The Robot-car can be controlled by a remote control. |
| F4.1 | S | The remote control can make the Robot-car drive forward and backward. |
| F4.2 | S | The remote control can make the Robot-car turn left and right. |
| F4.3 | S | The remote control can make the Robot-car drive with a variable speed. |
| **F5** | **S** | The most important information is displayed on an LCD. |
| F5.1 | S | The display shows the driving direction. |
| F5.2 | S | The display shows the current speed. |
| F5.3 | S | The display shows the total time the system has been in use. |
| **F6** | **S** | The Robot-car's operating mode can be selected via a menu. |
| F6.1 | S | The menu allows for selecting autonomous driving mode. |
| F6.2 | S | The menu allows for selecting slave mode. |
| F6.3 | S | The menu allows for selecting remote control mode. |
| **F7** | **S** | The total system time is persistent. |
|  | S | If the Robot-car is switched off, the total system time must be saved so that it continues counting upon restart. |
|  |  |  |
|  | C | Optional LEDs can indicate that the serial communication is in use. |
|  | C | A Bluetooth remote control can be used. |
|  | C | A buzzer can be added. |
|  | C | A battery indicator can be added. |

## Technical specifications

|  |  |  |
| --- | --- | --- |
| # | MoSCoW | Description |
| T1 | M | The ATmega328(p) microcontroller must be used. |
| T2 | M | The project must be programmed in the C language. |
| T3 | M | The DC motors must be driven via H-bridges, and their speed must be regulated using PWM (Pulse Width Modulation). |
| T4 | M | A self-designed/realized PCB (Printed Circuit Board) must be used for the H-bridge. |
| T5 | M | The total cost of additional components may not exceed €50. |

## User interface

The user interacts with the Robot-car through a variety of inputs and outputs. The system's main inputs are the **remote control** for basic driving, and the **sensors** for autonomous and slave modes.

The user can also use the display **menu** to switch between these modes.

Outputs include the Robot-car's physical movement, **LEDs** that indicate turning directions, and an **LCD display** that provides real-time feedback on the robot's status, speed, and driving direction. This interface acts as a communication tool, allowing the user to understand the robot's state and behaviour at a glance.

Following is a sketch of the user interface in all 3 modes.

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

# 

# Technical Design

*In this phase, thought is given to the way of realizing the various functions. This is about the* ***how*** *of the product to be developed. It is not the intention to describe new functional or technical specifications. That is the* ***what*** *and is described in the previous chapter. The purpose of the technical design is to translate the functional design into a technical implementation. You are now looking with the eyes of the designer.*

## Architecture

*The system is first divided into subsystems. Each subsystem has strong internal coherence and relatively little interaction with the other subsystems. The coherence between the subsystems is represented in an architecture diagram. A reasoned choice is made for the interface between the subsystems based on the functional and/or technical specifications.*

*Figure 5 shows an architecture that is generally applicable to embedded systems. The heart is the microcontroller that communicates through various interfaces with the subsystems. These interfaces must be unambiguously defined at this stage of the project.*

*Sensors have an arrow towards the microcontroller, e.g. for analog measurement, but can also have a double arrow, e.g. for a serial bus like I2C. The latter also applies to actuators, but actuators can also be realized with a single arrow towards the actuator. Think for example of a PWM signal.*

*A commonly used subsystem is communication with another device, such as a laptop, smartphone, etc. That interface usually also communicates in two directions, but it does not have to.*

*It is common to take into account that the microcontroller needs to be able to provide software updates. Therefore, there is often a programming and/or debugging subsystem. There are several interfaces that are common for debugging, such as SWD and JTAG.*

*Finally, the diagram shows how the power supply is controlled. From a voltage source, a voltage converter is used to realize the supply voltage (Vdd) for the embedded system. To keep the schematic clear, Vdd is not drawn to all subsystems. It may be necessary to have multiple voltage levels available in the embedded system, for example to control motors.*

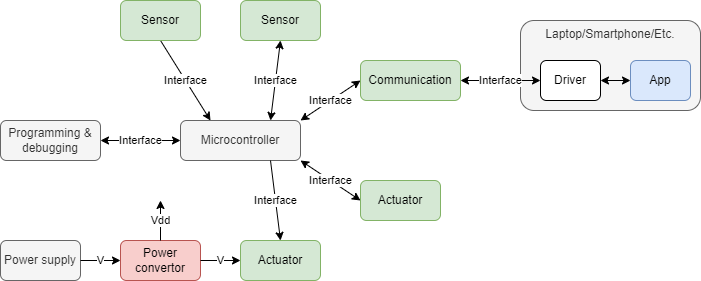


Figure 5. General architectural diagram for an embedded system.

## Interfaces

*For each interface, the electrical and/or data communication properties are described. Sometimes these choices are dictated by the (technical) requirements, but often you have to make choices here as a designer. Also for each interface between the microcontroller and other modules a design is made for the software driver by means of a UML sequence diagram.*

### Power supply

*The power supply specification specifies what voltages are needed in the system, what voltage sources there are, how they are converted, and what maximum current can be expected. A specification is formulated in a clearly recognizable manner:*

|  |  |
| --- | --- |
| **Specification** | *Here comes the text of the specification.* |

### Microcontroller – Sensor

*For sensors, a description of what the sensor measures will be given. Where possible, the choice is linked to a specification. It is important to be as complete as possible, thinking of quantities, units, range, precision, sample frequency, etc. A specification will be clearly recognizable:*

|  |  |
| --- | --- |
| **Specification** | *Here comes the text of the specification.* |

*It is also described that a software driver is realized. A driver for a sensor has at least one function to initialize the driver and one or more functions to read values from the sensor. Optionally, a function for writing to the sensor can be described, for example, to write configuration parameters. As a prefix for the names of the functions, choose the names that were also used in the architectural diagram, or an abbreviation of them.*

|  |  |
| --- | --- |
| **Specification** |  |

### Microcontroller – Actuator

*For actuators, the output signal is described in as much detail as possible. Here, where possible, the choice is linked to a specification. Again, think of quantities, units, range, precision, frequency, etc. A specification is clearly recognizable:*

|  |  |
| --- | --- |
| **Specification** | *Here comes the text of the specification.* |

*It is also described that a software driver will be realized. A driver for an actuator has at least a function to initialize the driver and one or more functions to write values to the actuator. Optionally, a function to read the actuator can be specified, for example to read the state of an actuator. As a prefix for the names of the functions, choose the names that were also used in the architectural diagram, or an abbreviation of them.*

|  |  |
| --- | --- |
| **Specification** |  |

### Microcontroller – Communication – PC driver – App

*The specification of communication with other devices has two parts: the interface(s) and the data format.*

*With respect to the interface, the following must be specified:*

* *electrical – voltage, current, etc.*
* *protocol – RS232, I2C, parallel, etc.*
* *protocol settings – such as bitrate, etc.*

*In addition, it must be unambiguously defined how data will be exchanged between the microcontroller main and app, or in other words the data format. Will an existing data format (such as JSON, XML, CSV, etc.) be used, or will a self-devised data format be implemented? In the case of the latter, then that data format must be unambiguously specified in this section.*

*It is also described that a software driver will be realized. A driver for communication is realized for a microcontroller, but not for the PC. The latter is usually available. A communication driver for a microcontroller has an initialization function, a write function and a read function. The function parameters depend on the selected data format.*

|  |  |
| --- | --- |
| **Specification** |  |

## Software

*For the main program, one or more software designs are shown and described. There are several methods to describe such a design, such as a flowchart, state diagram, sequence diagram, class diagram, etc. From the descriptions it should be clear what architecture has been chosen, e.g. event driven, cyclic executive with interrupts, an RTOS, a state machine, or something similar.*

# 

# Realisation

*Details of the realized hardware and software with accompanying explanations and calculations (such as power consumption, values of components, etc.). Complete and detailed diagrams of the hardware and listings of the software are included in the appendices.*

## Hardware

*The realized hardware is explained by means of wiring diagrams. It is also clarifying to include a picture of, for example, a realized PCB. Preferably use images of a part of the wiring diagram. Not everything needs to be explained. Choose two or three of the most relevant subsystems. The complete wiring diagram must be included in the appendices.*

## Software

*The realized software is explained by means of code snippets. Make sure the code is easy to read by using syntax highlighting. Use code snippets that are no longer than 20 lines and that each line of code fits on one line in the report. Not all of the realized code needs to be explained. Choose two or three of the most relevant subsystems. The full code is included as an appendix. Also pay attention to the software development environment. In doing so, ask yourself what is important information for a fellow engineer who will be using the same development environment for the first time.*

# 

# Testing

*Unambiguous representation of how the system, hardware and/or software was tested. What hardware and or software modules were tested, how were the functional specifications tested during the acceptance test? What test setup was used and what were the final results. Did the tests meet the functional and technical specifications? The results are accompanied by a clear description of any remaining problems and how they might be explained. Were any 'work arounds' performed during testing? The tests must be described in such a way that each test can be reproduced by others.*

# Conclusions and recommendations

*Reflection on the goals of the project. What are the results? What has been achieved and what has not been achieved? What can be added to, expanded on, improved?*

# References

Adrián Sánchez Cano. (2013, 3 5). *Using RTC module on FRDM-KL25Z.* Retrieved from https://community.nxp.com/docs/DOC-94734

ARM. (2022, 04 26). *µVision® IDE*. Retrieved from https://www2.keil.com/mdk5/uvision/

ARM Developer. (2022, 04 26). *KAN232 - MDK V5 Lab for Freescale Freedom KL25Z Board*. Retrieved from https://developer.arm.com/documentation/kan232/latest/

Berckel, M. v.-v. (2017). *Schrijven voor technici.* Noordhoff Uitgevers B.V.

contributors, W. (2022, 07 06). *MoSCoW method*. (Wikipedia, The Free Encyclopedia) Retrieved 07 06, 2022, from https://en.wikipedia.org/w/index.php?title=MoSCoW\_method&oldid=1091822315

contributors, W. (2022, 05 25). *SMART criteria*. (Wikipedia, The Free Encyclopedia) Retrieved 07 07, 2022, from https://en.wikipedia.org/w/index.php?title=SMART\_criteria&oldid=1089766780

ELECFREAKS. (2022, 04 19). Retrieved from Ultrasonic Ranging Module HC - SR04: https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf

Freescale Semiconductor, I. (n.d.). *FRDM-KL25Z Pinouts (Rev 1.0).* Retrieved 3 31, 2023, from https://www.nxp.com/document/guide/get-started-with-the-frdm-kl25z:NGS-FRDM-KL25Z

Freescale Semiconductor, Inc. (2012, 9). *KL25 Sub-Family Reference Manual, Rev. 3.*

Freescale Semiconductor, Inc. (2013, 10 24). *FRDM-KL25Z User's Manual, Rev. 2.0.* Retrieved from https://www.nxp.com/document/guide/get-started-with-the-frdm-kl25z:NGS-FRDM-KL25Z

Freescale Semiconductor, Inc. (2014, 08). *Kinetis KL25 Sub-Family, 48 MHz Cortex-M0+ Based Microcontroller with USB, Rev 5.*

Hmneverl. (2015, 11 18). *De beslismatrix, het maken van keuzes*. (Info.NU.nl) Retrieved 07 06, 2022, from https://mens-en-samenleving.infonu.nl/diversen/164525-de-beslismatrix-het-maken-van-keuzes.html

NXP. (2022, 04 19). *Kinetis® KL2x-72/96 MHz, USB Ultra-Low-Power Microcontrollers (MCUs) based on Arm® Cortex®-M0+ Core*. Retrieved from https://www.nxp.com/products/processors-and-microcontrollers/arm-microcontrollers/general-purpose-mcus/kl-series-cortex-m0-plus/kinetis-kl2x-72-96-mhz-usb-ultra-low-power-microcontrollers-mcus-based-on-arm-cortex-m0-plus-core:KL2x?tab=Buy\_Parametric\_Tab#/

NXP. (2022). *OpenSDA Serial and Debug Adapter*. Retrieved from https://www.nxp.com/design/software/development-software/sensor-toolbox-sensor-development-ecosystem/opensda-serial-and-debug-adapter:OPENSDA?&tid=vanOpenSDA

Solomon Systech Limited. (2008, 4). *SSD1306: Advanced Information.* Retrieved from https://cdn-shop.adafruit.com/datasheets/SSD1306.pdf

Vishay Semiconductors. (2017, 8 9). *TCRT5000(L), Reflective Optical Sensor with Transistor Output, Rev. 1.7*.

# Appendix A

# Appendix B

# Appendix n