Institutionen för systemteknik
Department of Electrical Engineering

Examensarbete

Development of a Collision Avoidance Truck System from a Functional Safety Perspective

Examensarbete utfört i Fordonssystem
vid Tekniska högskolan vid Linköpings universitet
av

Petter Gradin, Victor Ortman

LiTH-ISY-EX--11/4490--SE

Linköping 2011

Linköpings universitet
TEKNISKA HÖGSKOLAN

Department of Electrical Engineering
Linköpings universitet
SE-581 83 Linköping, Sweden

Linköpings tekniska högskola
Linköpings universitet
581 83 Linköping

Development of a Collision Avoidance Truck System from a Functional Safety Perspective

Examensarbete utfört i Fordonssystem
vid Tekniska högskolan i Linköping
av

Petter Gradin, Victor Ortman

LiTH-ISY-EX--11/4490--SE

Handledare: **Emil Larsson**
ISY, Linköpings universitet
Mattias Nyberg
Scania CV AB

Examinator: **Erik Frisk**
ISY, Linköpings universitet

Linköping, 10 October, 2011

Presentationsdatum	Institution och avdelning Institutionen för systemteknik	
20111010 **Publiceringsdatum (elektronisk version)**	**Department of Electrical Engineering**	Linköpings universitet

Språk	Typ av publikation	
___ Svenska _x_ Annat (ange nedan) Engelska **Antal sidor** 162	____Licentiatavhandling __x_Examensarbete ____C-uppsats ____D-uppsats ____Rapport ____Annat (ange nedan)	**ISBN (licentiatavhandling)** **ISRN LiTH-ISY-EX--11/4490--SE** **Serietitel (licentiatavhandling)** **Serienummer/ISSN (licentiatavhandling)**

URL för elektronisk version
http://www.ep.liu.se

Publikationens titel

Development of a Collision Avoidance Truck System
from a Functional Safety Perspective

Författare
Petter Gradin, Victor Ortman

Sammanfattning
ISO 26262 is a functional safety standard under development at the time of this thesis. It is an adaptation of the functional safety standard IEC 61508, aimed at development of automotive electrical/electronic systems. The version of ISO-26262 that was used and discussed in this thesis is the final draft released in January 2011.
In this thesis, a subset of ISO-26262 is applied in the development of a safety critical driver assistance system for a Scania vehicle. The parts of ISO-26262 that are treated are Part 3: Concept phase, Part 4: Product development at the system level and Part 5: Product development at the hardware level. Throughout the thesis we evaluate ISO-26262 and report our experience of working with it. The driver assistance system under development, which ISO-26262 is applied to, is Collision Avoidance by Steering, a system that aims to avoid or mitigate rear-end collisions with vehicles in front by automatic steering of the vehicle.

Nyckelord
ISO-26262, functional safety, collision avoidance

Abstract

ISO 26262 is a functional safety standard under development at the time of this thesis. It is an adaptation of the functional safety standard IEC 61508, aimed at development of automotive electrical/electronic systems. The version of ISO-26262 that was used and discussed in this thesis is the final draft released in January 2011.

In this thesis, a subset of ISO-26262 is applied in the development of a safety critical driver assistance system for a Scania vehicle. The parts of ISO-26262 that are treated are Part 3: Concept phase, Part 4: Product development at the system level and Part 5: Product development at the hardware level. Throughout the thesis we evaluate ISO-26262 and report our experience of working with it. The driver assistance system under development, which ISO-26262 is applied to, is Collision Avoidance by Steering, a system that aims to avoid or mitigate rear-end collisions with vehicles in front by automatic steering of the vehicle.

Sammanfattning

ISO 26262 är en funktionell säkerhetsstandard som vid tidpunkten för detta examensarbete är under utveckling. Det är en anpassning av den funktionella säkerhetsstandarden IEC 61508, som syftar till utveckling av elektriska / elektroniska system inom personbilsindustrin. Den version av ISO-26262 som behandlas i detta examensarbete är det slutgiltiga utkastet som släpptes i januari 2011.

I detta examensarbete tillämpas vissa delar av ISO-26262 i utvecklingen av ett säkerhetskritiskt förarassistanssystem till en Scania lastbil. De delar som tillämpas är Part 3: Concept phase, Part 4: Product development at the system level samt Part 5: Product development at the hardware level. Under examensarbetets gång utvärderas ISO-26262 och den erfarenhet vi fått från att arbeta enligt standarden rapporteras. Förarassistanssystemet som utvecklades, och ISO-26262 tillämpades på, kallas Collision Avoidance by Steering, ett system som syftar till att undvika eller mildra påkörningar av framförvarande fordon med hjälp av automatisk undanstyrning av lastbilen.

Acknowledgements

This master thesis has been carried out at the Department of Electrical Engineering, Linköpings University, in cooperation with Scania AB.

We would like to give thanks to a number of people who have helped us throughout this master thesis: Mattias Nyberg, Technical Manager - REPA Scania, for guidance, advice and support in the field of functional safety and any other concerns we might have had. Erik Frisk, Associate Professor, and Emil Larsson, Ph.D. student, of the Department of Electrical Engineering - Linköpings University for support in writing this master thesis. Anders Johansson, Assad Alam, Henric Pettersson, Håkan Gustavsson, Jan Dellrud, Jon Andersson, Joseph Ah-King, Pär Degerman, Rickard Lyberger and Tony Sandberg of REP - Predevelopment Scania for aid in countless questions and making us feel welcome.

We would also like to give thanks to the numerous persons spread across Scania who have aided us in various ways.

Glossary

For the purpose of reading this document, the following terms and definitions apply.

- **Allocation**

  Assignment of a requirement to an element.

- **Architecture**

  Representation of the structure of the item.

- **ASIL (Automotive Safety Integrity Levels)**

  A level to specify the necessary requirements of ISO 26262 and safety measures to apply to an item or element in order to avoid an unreasonable residual risk. D represent the most stringent and A the least stringent level.

- **ASIL decomposition**

  Apportioning of safety requirements redundantly to independent elements with the objective of reducing the ASIL.

- **Controllability**

  The ability to avoid a specific harm or damage through the timely reactions of the persons involved, possibly with support from external measures.

- **COO**

  See Coordinator.

- **Coordinator**

  Electronic control unit mounted in Scania trucks.

- **Degradation**

  Strategy for providing safety by design after the occurrence of failures.

- **Diagnostic coverage**

  Proportion of the hardware element failure rate that is detected or controlled by the implemented safety mechanisms.

- **E/E system**

  System that consists of electrical and/or electronic elements.

- **EBS**

  Electronic control unit mounted in Scania trucks.

- **ECU**

  Electronic control unit

- **Element**

  System or part of a system.

- **EMS**

  Electronic control unit mounted in Scania trucks.

- **External measure**

  Measure separate and distinct from the item, that reduces or mitigates the risks resulting from the item.

- **Fault**

  Abnormal condition that can cause an element or an item to fail.

- **Fault tolerant time interval**

  Time-span in which a fault can be present before a hazardous event occurs.

- **Functional concept**

  Specification of intended functions and their interactions necessary to achieve the desired behaviour.

- **Functional safety**

  Absence of unreasonable risk due to hazards caused by malfunctioning behavior of E/E systems.

- **Functional safety concept**

  Specification of the functional safety requirements, their allocation to architectural elements and their interactions necessary to achieve the safety goals.

- **Functional safety requirement**

  Specification of a functional behavior or measure necessary to achieve the safety goals.

- **GMS**

  Electronic control unit mounted in Scania trucks.

- **Hardware architectural metrics**

  Metrics for the assessment of the effectiveness of the hardware architecture with respect to safety.

- **Harm**

  Physical injury or damage to the health of persons.

- **Hazard**

  Potential source of harm caused by a malfunction of the item.

- **Hazard analysis and risk assessment**

  Method to categorize hazardous events of items and to specify safety goals and ASILs related to the prevention of the associated hazards.

- **Hazardous event**

  Combination of a hazard and an operational situation.

- **HSI**

  Work product called Hardware Software Interface Specification.

- **Independence**

  Absence of dependent failures two or more elements that could lead to the violation of a safety requirement.

- **Item**

  System of several systems to implement a function at the vehicle level.

- **Latent fault**

  Multiple point fault whose presence is not detected by a safety mechanism nor perceived by the driver within the multiple point fault detection interval.

- **Multiple point failure**

  Failure, resulting from the combination of several independent faults, which leads directly to the violation of a safety goal.

- **Multiple point fault**

  Individual fault that, in combination with other faults, leads to a multiple point failure.

- **Multiple fault detection interval**

  Time-span to detect a multiple point fault before it may contribute to a multiple point failure.

- **Operating mode**

  Perceivable functional state of an item.

- **Operational situation**

  Scenario that may occur during the vehicle's life.

- **Redundancy**

  Existence of means in addition to the means that would be sufficient for an element to perform a required function.

- **Residual risk**

  Risk remaining after the deployment of safety measures.

- **Risk**

  Combination of the probability of occurrence of harm and the severity of that harm.

- **Safe state**

  Operation mode of an item without an unreasonable level of risk.

- **Safety goal**

  Top-level safety requirement as a result of the hazard analysis and risk assessment.

- **Safety mechanism**

  Technical solution to detect faults or control failures in order to achieve or maintain a safe state.

- **Severity**

  Estimate of the extent of harm that may occur in a hazardous situation.

- **Single point failure**

  Failure that results from a single point fault and leads directly to the violation of a safety goal.

- **Single point fault**

  Fault in an element that is not covered by a safety mechanism and that leads directly to the violation of a safety goal.

- **Systematic failure**

  Fault produced by human error during system development and operation.

- **Technical safety concept**

  Specification of the technical safety requirements and their allocation to architectural elements.

- **Technical safety requirement**

  Requirement derived for implementation of associated functional safety requirements.

- **Transient fault**

  Fault that occur once and subsequently disappears.

- **Unreasonable risk**

  Risk judged to be unacceptable.

- **VRU**

  Vulnerable road user

- **Warning and degredation concept**

  Specification for how to alert the driver of potentially reduced functionality and specification of how to provide this reduced functionality to reach a safe state.

- **Work product**

  Result of one or more associated requirements of ISO 26262.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter is an introductory chapter. It explains the background to the thesis, the objectives and questions to be answered. The chapter also includes the method that will be used for completing the objectives and what the expected results are.

1.1 Background

According to the World health organization an estimation of people killed in road accidents every year is 1.2 million [1]. Therefore one of the most important features in future vehicle development is safety. New functionality such as safety systems in the vehicle are directly related to product development where the focus is on safety. Meanwhile there is a larger amount of software, and more sensors and actuators in a vehicle than ever before. This means that the risk of having software bugs or hardware failures in the vehicle increases. It is important that the automotive industry takes this risk seriously and adapt their working methods and products in order to avoid it. It exists strong need for a process that clearly lead to the development of secure systems, and at the same time provide proofs that all security objectives of the system are met.

ISO 26262 is a functional safety standard currently in development. It is an adaptation of the Functional Safety standard IEC 61508, aimed at Automotive Electric/Electronic (E/E) Systems. It is predicted that a similar standard, based on ISO-26262, will be developed for heavy vehicles in a few years.

The version of ISO-26262 managed in this thesis is the draft released in January 2011.

1.2 Objectives

Scania needs to know how development of safety critical systems according to the principles of ISO 26262 are done in order to prepare for the upcoming standard.

The problem is that Scania have no experience of working with ISO 26262 within the company.

Questions that need to be answered are the following:

- What are the advantages and disadvantages of complying with ISO-26262?
- What would an architecture developed according to the principles of ISO 26262 look like? For example, what software functions will be needed? What hardware will this software be executed on? What type of hardware components are needed? What interfaces will be needed that connects all these parts?
- Which parts of ISO 26262 are sensible and which parts will just cause overhead?
- Will a system developed according to ISO-26262 really be safe?

The objectives of this master thesis are to follow parts of ISO-26262 to develop an architecture for a safety critical E/E system which will be placed in a Scania truck. The version of ISO-26262 that will be used is the one released in January 2011. During the development, ISO-26262 will be evaluated in order to answer the questions posed.

The system will analyse the environment around a heavy vehicle and calculate the time to collision with a forward vehicle. If the time to collision is low enough and the vehicle is driving on a highway with a velocity greater than 30 km/h, the system will respond with an action. Table 1.1 describes the desired functionality of the system in relation to the time to collision and the operational situation of the vehicle.

The warning threshold is the time before a collision that is considered enough for the driver to be able to avoid a collision but still low enough to avoid unnecessary warnings. The critical threshold is the time before a collision where the driver is unable, but the system is able, to avoid or mitigate a collision.

Table 1.1: desired functionality of the system in relation to the time to collision and the operational situation of the vehicle.

Operational situation	**Time to collision**	**Action from system**
Highway velocity > 30 km/h	Above warning threshold	None
Highway velocity > 30 km/h	Below warning threshold but over critical threshold	Warn the driver by optical and acoustic measures (warning lights and sound).
Highway velocity > 30 km/h	Below critical threshold	If possible, avoid or mitigate the collision by steering the vehicle into a different trajectory.
Highway velocity < 30 km/h	N/A	None
Not on highway velocity > 30 km/h	N/A	None
Not on highway velocity < 30 km/h	N/A	None

1.3 Related Research

There have been a few reports and articles relating to ISO 26262. For example a master thesis concerning the implementation of part 3 of ISO-26262 [2] and an article showing a practical example how a great portion of the ISO-26262 safety case can be developed, documented, evaluated and managed without loosing the overall picture [3].

1.4 Method

The solution path will follow the parts of the ISO-26262 core process that have a direct impact on the developing of the architecture. Parts such as planning of safety activities, testing, production and verification of work products will not be treated in this thesis. Figure 1.1 Shows the core process of ISO-26262 with all the phases of the process enumerated.

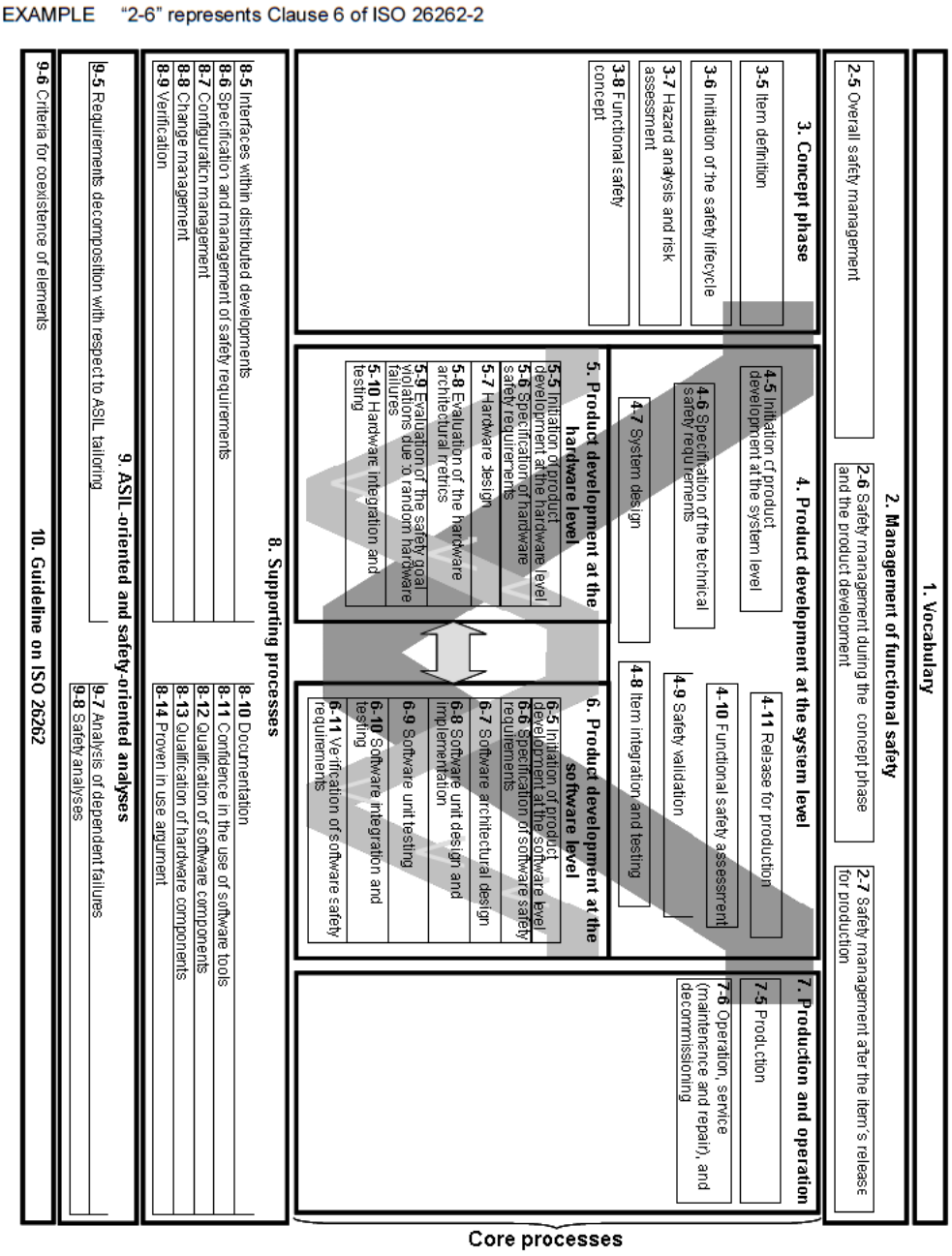

Figure 1.1: The ISO 26262 core process

The following phases of the ISO-26262 core process will be treated in this thesis. Figure 1.2 outline the treated phases of the ISO-26262 core process.

- 3-5 Item definition

The objectives of this sub phase is to define the item and describe it and its dependencies and interaction with its environment.

- 3-7 Hazard analysis and risk assessment

  The objectives of this sub phase is to identify and categorize all the hazards associated to the item and then formulate the safety goals related to the prevention of the hazardous event.

- 3-8 Functional safety concept

  The objectives of this sub phase is to derive the functional safety requirements from the safety goals and allocate them to elements of the preliminary architecture.

- 4-6 Specification of technical requirements

  The objectives of this sub phase is to first specify the technical safety requirements and then verify that they comply with the functional safety requirements.

- 4-7 System design

  The first objective is to develop the system design specification and the technical safety concept so that they comply with the functional and technical safety requirements. The second objective is to verify that the system design and the technical safety concept comply with the technical safety requirements specification. A further objective is to initiate the hardware-software interface specification.

- 5-6 Specification of hardware safety requirements

  The objective of this sub phase is to specify the hardware safety requirements. The requirements are derived from the technical safety concept and the system design specification. It must also be verified that the hardware safety requirements are consistent with the technical safety concept and the system design specification. The hardware-software interface initiated in sub phase 4-7 shall also be detailed.

- 5-7 Hardware design

  The objective of this sub phase is to design the hardware in accordance with the system design specification and the hardware safety requirements. The hardware design must then be verified against the system design specification and the hardware safety requirements.

- 5-9 Evaluation of safety goal violations due to random hardware failure

  The objective of this sub phase is to make available criteria that can be used in a rationale which prove that the risk of violating a safety goal due to random hardware failure is sufficiently low.

In the beginning of the development process (3-5, 3-7, 3-8) the whole defined item will be processed. But after these parts the scope will be reduced and Part 4 and Part 5 will only be applied to an element of the item.

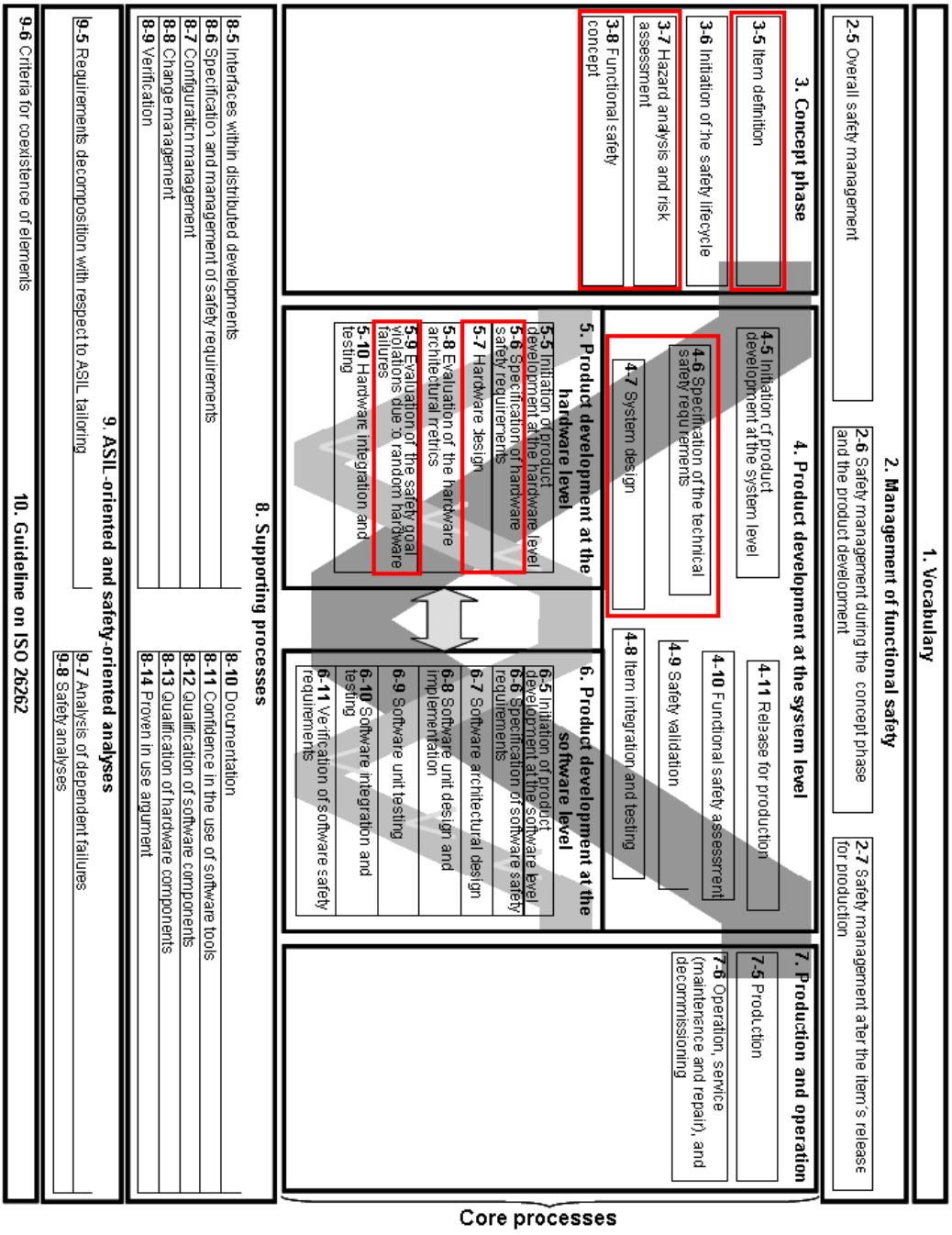

Figure 1.2: The parts of the ISO-26262 process this thesis will treat

1.5 Outline of the Report

The report is structured in the following manner. First comes the introduction chapter, which explains the problems to be solved and the questions to be answered. The chapter also includes the method that will be used for solving the problem and what the expected results are. After the introduction chapter there is a chapter concerning ISO-26262, this chapter attempts to explain what ISO-26262 is and its usage. The report continues with three chapters concerning the implementation of part 3, 4 and 5 of ISO-26262. Each part has its own chapter. Every chapter concerning a part of ISO-26262 has the following outline:

- Objectives of Part X According to ISO-26262

  This section gives a brief explanation of the objectives of part X. Every work product of part X that will be managed will be made clear and their purpose explained.

- Work products and Reflections

  Work products are documents or specifications produced when working according to ISO-26262. A work product is in that manner the result of one or more associated requirements of ISO-26262.

  This section contains the actual work products produced in part X with a following section containing the deviations from and reflections on ISO-26262 related to that work product. Every work product is marked with an identifier of the format "x-y:WPz ID" which means that the work product is from ISO 26262 Part x, clause y and numbered as work product number z in that clause. ID is the name of the work product.

The report ends with a chapter where the results are presented and discussed.

Chapter 2

ISO-26262

This chapter aims to give the reader a better understanding of ISO 26262. It will define ISO 26262 and explain what the purpose of this standard is. The chapter also aims to describe the general working methodology of ISO 26262. The version of ISO 26262 managed in this thesis is the draft standard released in January 2011.

2.1 What is ISO-26262?

ISO-26262 is a functional safety standard currently in development. It is an adaptation of the Functional Safety standard IEC 61508, aimed at Automotive Electric/ Electronic Systems.

2.2 Scope of ISO-26262

ISO-26262 is intended to be applied to safety-related system that includes one or more electrical/electronic (E/E) systems that are installed in a car. The standard addresses the possible hazards that are caused by an E/E system in a car malfunctioning. It does not address hazards that are not directly related to the E/E system. ISO-26262 does not address the nominal performance of E/E systems, for example ISO-26262 does not state how powerful the breaks should be or how fast the airbag should deploy. Instead the standard describes how these systems should be developed in order to avoid a hazard.

2.3 Purpose

The purpose of complying with ISO-26262 is that the vehicle manufactures can develop systems with increased security. ISO-26262 also provide proofs that all reasonable security objectives are met so that the customers and developers can

feel confident that a system developed according to ISO-26262 is assumed to be safe.

2.4 Concept of ISO-26262

ISO-26262 uses the following concept, illustrated in figure 2.1, of safety goals and safety concept in order to eliminate or reduce risks and hazards of the item.

- A hazard analysis and risk assessment identifies hazards that need risk reduction.
- A safety goal is formulated for each hazardous event.
- An Automotive Safety Integrity Level (ASIL) is associated with each safety goal.
- The functional safety concept describes the functionality required to achieve the safety goal(s).
- The technical safety concept describes how this functionality is implemented in hardware and software.
- Software safety requirements and hardware safety requirements state the specific safety requirements which will be implemented as a part of the software and hardware design.

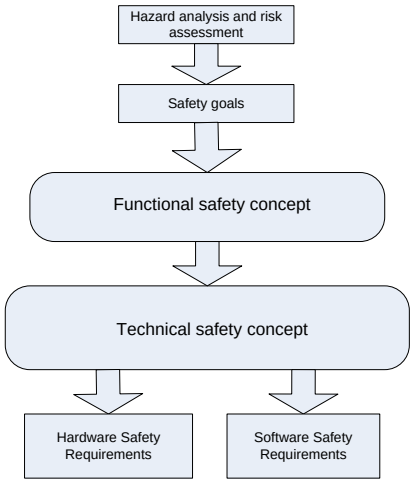

Figure 2.1: Overview of ISO-26262 work flow.

Chapter 3

Part 3 - Concept Phase

This chapter concerns the work that has been done according to part 3 of ISO 26262. The chapter begins with a section explaining objectives of part 3 and then continues with the work products that have been produced during part 3. After each work product reflections on, and deviations from, ISO 26262 are presented.

3.1 Regarding Section Titles

In Section 3.3, some section titles have a number in paranthesis. The number denotes a specific requirement, from [6, Part 3], that the section fulfills.

3.2 Objectives of Part 3 According to ISO 26262

This section attempts to explain the objectives and purpose of part 3 - concept phase.

3.2.1 Item Definition Explanation

The first objective of Part 3 - concept phase is to write the item definition. The purpose of writing the item definition is to define the item by specifying what functionality you desire from the item along with its dependencies and interactions with the environment. This serves to provide sufficient information about the item so that subsequent sub phases can be conducted.

3.2.2 Hazard Analysis and Risk Assessment Explanation

The purpose of the hazard analysis and risk assessment is to analyze the item in order to categorize the hazards that can be triggered by a malfunction in the item.

While analyzing the hazards you assume that all systems external to the item is functioning correctly. Each hazard is combined with an operational situation of the vehicle and this creates a hazardous event.

Each hazardous event is analyzed and then given the following three parameters:

- Severity (S0 - S3) This parameter is a measurement of how severe the potential harm is for each hazardous event. The parameter ranges from S0 to S3 where S0 means no injuries and S3 means life threatening injuries.
- Controllability (C0 - C3) This parameter is a measurement of how probable it is for the driver and other persons potentially at risk to gain control of the hazardous event, such that they are able to avoid the specific harm. The parameter ranges from C0 to C3 where C0 means controllable in general and C3 means that less than 90% of all drivers or other traffic participants are usually able, or barely able, to avoid harm.
- Exposure (E0 - E4) This parameter is the probability that the vehicle and the driver is in such an operational situation that is described in the hazardous event. The parameter ranges from E0 to E4 where E0 means that the operational situation occurs less than once a year for the great majority of drivers and E4 means that the operational situation occurs almost every drive, for most drivers, on average.

An explanation of the different values on these parameters can be seen in the ISO-26262 standard, see [6, Part3, Annex B].

An ASIL(A - D) for the hazardous event is then determined by looking up the given parameters in a table specified in the ISO-26262 [6, Part3, Table 4]. If the controllability, severity and exposure parameters are low enough then the hazardous event is assigned the class QM, this class detonates no requirement to comply with ISO 26262. If an hazardous event is assigned E0, S0 or C0 then no class at all is assigned to the hazardous event. The ASIL is later used to determine what requirements that will be necessary for the item or an element in order to be reasonable confident that the specific hazardous event does not occur.

3.2.3 Safety Goals Explanation

All hazardous event shall have a safety goal associated with them, however, one safety goal can cover multiple hazardous events. A safety goal is a top-level safety requirement for the item. They are not expressed in terms of technological solutions but of terms of functional objectives. Thus, if all safety goals are met and all external systems function correctly, then none of the hazardous events specified in the hazard analysis and risk assessment can occur.

3.2.4 Functional Safety Requirements Explanation

The last step of Part 3 - concept phase is to write the functional safety concept. The aim is to analyze each safety goal and from the obtained information allocate functional requirements, called functional safety requirements, to elements of the item. These functional requirements shall imply that the safety goals are fulfilled.

3.3 3-5:WP1 Item definition

This section is a work product resulting from Part 3, Clause 5 of the ISO-26262 standard. The aim is to describe the item, its dependencies and interaction with the environment and other items.

3.3.1 Functional Concept (5.4.1 a)

This section explains the purpose of the item and the functionality needed to fulfill this purpose.

Purpose

The purpose of the item is, through warnings and in worst case automated steering of the vehicle, to mitigate or preferably avoid collisions with other road-users or objects in front of the heavy vehicle.

Functionality

The item shall, by using data from sensors, calculate the time to collision with a forward vehicle. When the time to collision is less than a certain threshold the system will warn the driver by optical and acoustic measures. If the time to collision decreases even further, so it is too late for the driver to react, the item will maneuver the vehicle in order to mitigate or avoid the collision in a safe way. The automated maneuvering is done by applying torque to the steering shaft through an electrical motor. This will, if not counteracted by the driver, change the trajectory of the vehicle. Figure 3.1 illustrates the functionality of the item in different situations.

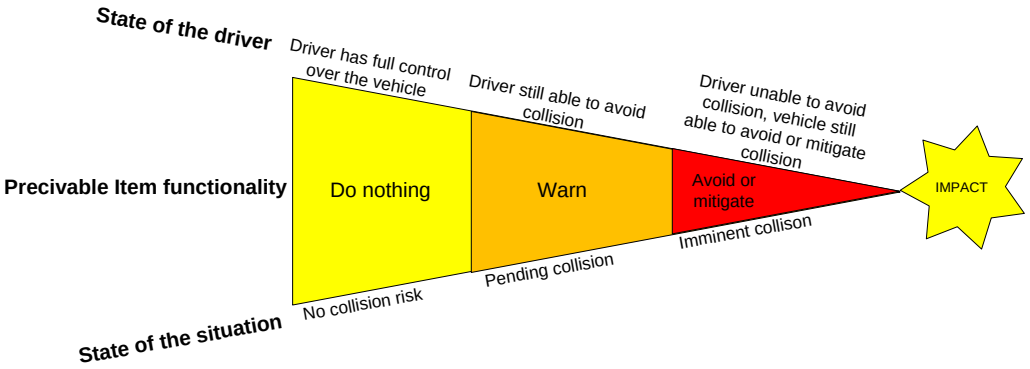

Figure 3.1: Visual description of functionality in different situations.

The function can be in three different operating modes, ON/ACTIVE, ON/PASSIVE and OFF. ON and OFF is controlled by the driver by pushing a designated button in the cab, while ACTIVE and PASSIVE is controlled by the speed of the vehicle and whether or not the vehicle is on a highway. If the vehicle is on a highway and driving with a velocity greater than 30 km/h the item is ACTIVE, otherwise the item is PASSIVE. The default mode when starting the vehicle is ON. When the system is OFF, a warning light will be displayed on the dashboard.

Table 3.1: Operation states and functionality of the item

Item operating mode	Item operating state	Item functionality
OFF	N/A	None
ON/PASSIVE	N/A	Detect if vehicle travels faster than 30 km/h on a highway. In that case change to ACTIVE mode.
ON/ACTIVE	Time to collision is larger than threshold	None
ON/ACTIVE	Time to collision is smaller than threshold but situation still manageable by driver	Warn the driver
ON/ACTIVE	Time to collision is so small that it is too late for the driver to react	Try to mitigate or avoid collision by steering the vehicle
ON/ACTIVE	Fail safe, a state where a fault in the item has been detected	Warn the driver and disable all other functionality of the item

3.3.2 Constraints (5.4.1 b)

Vehicles such as mining trucks or trucks transporting in other environments than roads on a regular basis should not be equipped with this item.

3.3.3 Legal Requirements (5.4.1 c)

There exists legal requirements for Emergency Braking System (AEBS) [4] and Forward Collision Mitigation System (FCMS) [5]. However no specific legal requirements exists for this type of system. Although, since the AEBS and FCMS share many similar characteristics with this item, it can be assumed that the same, or similar, legal requirements will be applicable for this type of system. This item will conform with the following requirements inspired by the legal documents for AEBS and FCMS:

- A collision warning must be given when the item has detected the possibility of a collision. The warning referred shall be provided by at least 2 modes from acoustic, haptic or optical.

- The item shall provide the means for the driver to interrupt the emergency steering phase.

3.3.4 Behavior Achieved by Similar Functions, Items or Elements (5.4.1 d)

A similar function is the Emergency braking system (AEBS) designed to automatically brake before a collision.

3.3.5 Consequences of Behavioral Shortfalls (5.4.1 f)

This section explains the known failure modes and the potential hazards of the item.

Known Failure Modes

- Item sends a CAN message containing false information.
- Item applies torque to steering shaft when not intended.
- Item fails to apply torque to steering shaft when it should have.
- Item applies to much torque to the steering shaft.
- Item applies to little torque to the steering shaft.

Potential Hazards

- Vehicle tries to turn when it should not.
- Vehicle does not try to turn when it should.
- Vehicle turns more than intended.
- Vehicle turns less than intended.
- The driver is given a false warning of a possible collision.
- The driver is not given a warning even though the time to collision is less than the threshold.

3.3.6 Elements of the Item (5.4.2 a)

Figure 3.2 displays the elements of the item in relationship to each other and in relation to external systems.

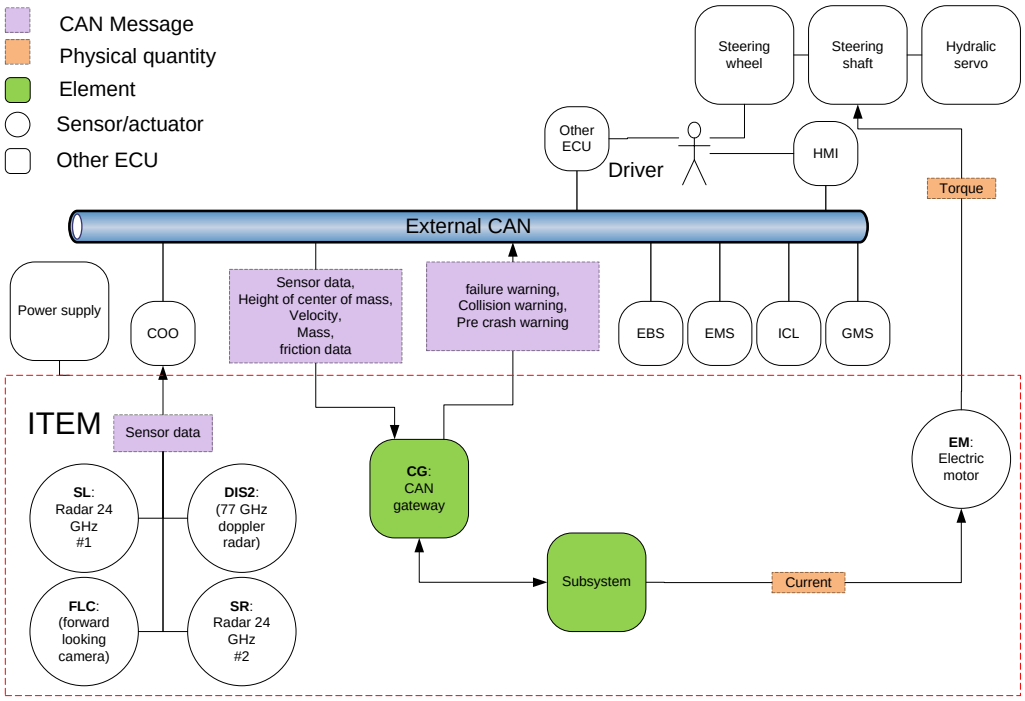

Figure 3.2: The elements of the item.

3.3.7 Allocation of Functionality (5.4.2 f)

This section explains the functionality allocated to the elements of the item.

Sensors

The sensors are responsible for gathering data about the surroundings of the vehicle. The two 24 GHz radars (SL and SR) are mounted on each side of the vehicle. The DIS2 and the FLC are mounted in the front of the vehicle. Figure 3.3 shows a heavy vehicle together with the sensors and their field of vision.

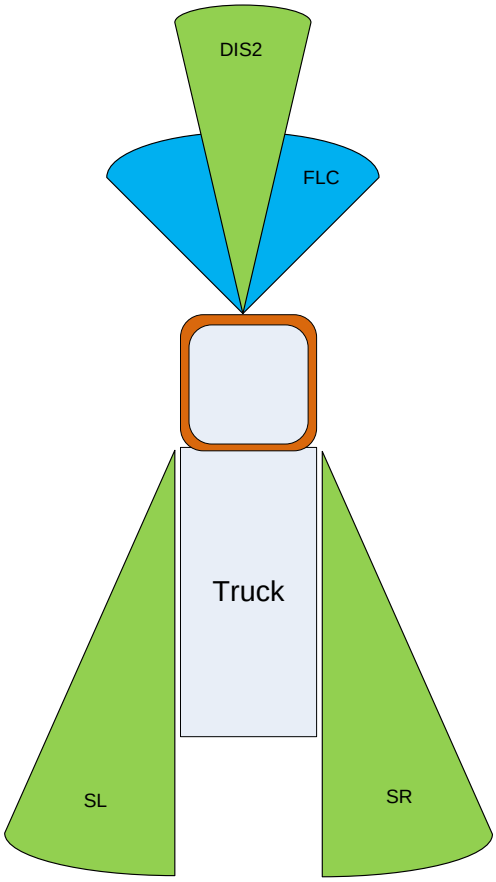

Figure 3.3: The sensor placement and their field of vision.

CAN Gateway

The CAN gateway decides what CAN messages will be passed between the external CAN bus and the subsystem.

Subsystem

The subsystem fuses the data gathered from the sensors in order to create a data structure containing information about surrounding objects of the vehicle and their relative speed to the vehicle. This data structure is then processed in order to make a steering and a warning decision. If the decision to make an evasive maneuver is made, the subsystem uses the information about the vehicle surroundings combined with information about velocity, mass of the vehicle, height of the center of

mass of the vehicle and friction towards the road in order to calculate a safe trajectory. When a safe trajectory has been calculated, a current is sent to the electric motor in order to make the vehicle follow the trajectory.

Electric Motor

The electric motor is responsible for applying the right amount of torque to the steering shaft when making an evasive maneuver.

3.3.8 Effect on Other Items (5.4.2 b)

Since the item has the ability to predict a possible crash situation the item will send out a pre-crash warning on the CAN bus so other ECUs can take proper actions.

3.3.9 Interaction with Other Items (5.4.2 c)

The item need to communicate with other ECU:s through a CAN interface. The item also requires power from an external power supply.

3.3.10 Functionality Required by Other Items (5.4.2 d)

No functionality is required by other items.

3.3.11 Functionality Required from Other Items (5.4.2 e)

In order to predict the time to collision and safe trajectories with high accuracy we need other ECU:s to send CAN messages containing information such as velocity of vehicle, mass of vehicle, height of the center of mass of the vehicle and information about the friction to the road. The battery is also needed to power the hardware in this item.

The following external requirements on other items are defined:

- The GMS[1] is required to periodically send correct information about the evaluated mass of the vehicle on the external CAN bus.
- The EMS[1] is required to periodically send correct information about the velocity of the vehicle on the external CAN bus.
- The GMS[1] is required to periodically send correct information about the height of the center of mass of the vehicle on the external CAN bus.

[1]ECUs can vary between different models of trucks. ECUs given in the text are examples.

- The external CAN bus must forward messages without corrupting them.
- The EBS[1] is required to periodically send correct information about the friction to the road on the external CAN bus.
- The power supply must supply the item with 24 voltage.

3.3.12 Different Operating Scenarios (5.4.2 g)

The item is intended to operate while driving on a road. The following operating scenarios are defined for the item.

- Traveling forward on a highway with a velocity greater than 30 km/h
- Traveling on a highway with a velocity less than 30 km/h
- Traveling on a none-highway road

3.4 Reflections and Deviations from ISO-26262 in Item Definition

This section presents reflections on and deviations from ISO-26262 in Item Definition.

Questionable Parts of Item Definition

During our work with ISO 26262 we have found use of most parts of the item definition. There are however some parts of the item definition to which we have found no use of during our continued work with ISO 26262. For example [6, Part3, 5.4.1 d], which tells us that behavior achieved by similar functions, items or elements should be specified. We wonder what the purpose of this is, could it be that by looking at items with similar functionality one gets a better understanding of the item at an early stage of the development process? If so, is this reason enough to add extra mandatory documentation to the work flow?

Other parts of the item definition raises other questions. Such as [6, Part3, 5.4.1f] which requires specification of behavioral shortfalls of the item, including failure modes and hazards. What is the reason for specifying hazards in the item definition when there is a whole other work product, Hazard analysis and risk assessment, dedicated to this?

"Other Items"

ISO 26262 mentions the term "other items" a few times in the item definition requirements. We have considered this to simply mean E/E systems but a different interpretation is that the term only applies to other systems that also have been designed using ISO 26262.

Sensor Placement Architecture

As can be seen in figure 3.2 the different sensors are connected to COO. This is an assumption we made and does not necessarily reflect reality.

Skipped Documentation

The standard demand that assumptions on behavior expected from the item shall be made clear [6, Part3, 5.4.1e]. We have chosen not to write any specific text in regard to this clause because we don't understand the meaning of it. Our interpretation of the clause is that the expected behavior of the item is the same as the desired functionality of the item and is thus made clear in the functional concept section of the item definition.

3.5 3-7:WP1 Hazard Analysis and Risk Assessment

This section is a work product resulting from requirements 7.4.1.1 to 7.4.4.2 in Part 3 of the ISO-26262 standard. The aim is to identify and categorize the hazards that a malfunction in the item can trigger.

3.5.1 Situation Analysis (7.4.2.1)

In order to create all relevant operational situations, five different scenarios are considered. These scenarios were chosen since the item is only supposed to act when the vehicle is on a highway and has a velocity greater than or equal to 30 km/h. Also, whether or not the vehicle is on a highway can have impact on the ASIL classification. These are the five different scenarios considered:

- The vehicle is traveling on a highway with a velocity greater than or equal to 30 km/h, a collision is **not imminent**.
- The vehicle is traveling on a highway with a velocity greater than or equal to 30 km/h, a collision is **imminent**.

- The vehicle is traveling on a highway with a velocity less than 30 km/h.
- The vehicle is **not** traveling on a highway with a velocity greater than or equal to 30 km/h.
- The vehicle is **not** traveling on a highway with a velocity less than to 30 km/h.

Two different weather conditions will also be taken into consideration due to their impact on the ASIL classification:

- Slippery road.
- Impaired vision.

The above weather conditions can be combined in every possible way and in combination with one of the five scenarios, an operational situation is created.

3.5.2 Hazards (7.4.2.2.1)

In Table 3.2 the possible hazards associated with the item are listed. To find the relevant hazards, brainstorming was used.

Table 3.2: The different hazards of the item.

Identifier	**Hazard**
H1	Vehicle is about to crash into a forward vehicle but fails to do an evasive maneuver by steering.
H2	Vehicle does an unintended evasive maneuver by steering with limited torque (torque applied **less** than 10 Nm).
H3	Vehicle does an unintended evasive maneuver by steering with unlimited torque (torque applied **more** than 10 Nm).
H4	Vehicle gives an unintended warning to the driver of an imminent collision.
H5	Vehicle is about to crash into a forward vehicle but fails to warn the driver.
H6	Vehicle is in a collision situation and does an evasive maneuver by steering into an unsafe trajectory.

3.5.3 Hazardous Event Identification (7.4.2.2.3)

A hazard combined with an operational situation creates a hazardous event [6, Part 1, 1.59]. In Table 3.3 through Table 3.8, each hazard has been combined with all possible and relevant operational situations. Each row in the tables below represents a hazardous event. Each hazardous event is given a severity class S [6, Part3, 7.4.3.2], a probability of exposure class E [6, Part3, 7.4.3.4] and a controllability class C [6, Part3, 7.4.3.9]. These classifications are seen in the S,C and E columns of the tables. Based on those three classes, each hazardous event is assigned an ASIL classification [6, Part3, 7.4.4.1].

Hazard H1. Vehicle is about to crash into a forward vehicle but fails to do an evasive maneuver by steering.

In Table 3.3 each hazardous event associated with hazard H1 is listed.

Table 3.3: Hazardous events associated with hazard H1.

	Operational situation (7.4.2.1.1)						
Identifier	Scenario	Weather conditions	Consequence (7.4.2.2.4)	C	S	E	ASIL
HE1	Highway,speed > 30 km/h, collision imminent	Good conditions	Crash into forward vehicle	C3	S3	E1	A
HE2	– | | –	Slippery road	– | | –	C3	S3	E1	A
HE3	– | | –	Impaired vision	– | | –	C3	S3	E1	A
HE4	– | | –	Slippery road, Impaired vision	– | | –	C3	S3	E1	A

Hazard H2. Vehicle does an unintended evasive maneuver by steering with limited torque (torque applied less than 10 Nm)

In Table 3.4 each hazardous event associated with hazard H2 is listed.

Table 3.4: Hazardous events associated with hazard H2.

	Operational situation (7.4.2.1.1)						
Identifier	Scenario	Weather conditions	Consequence (7.4.2.2.4)	C	S	E	ASIL
HE5	Highway,speed >30 km/h, collision not imminent	Good conditions	Crash into vehicle in adjacent lane or drive off the road.	C2	S3	E4	C
HE6	– | | –	Slippery road	– | | –	C3	S3	E3	C
HE7	– | | –	Impaired vision	– | | –	C2	S3	E2	A
HE8	– | | –	Slippery road, Impaired vision	– | | –	C3	S3	E2	B
HE9	Highway,speed <30 km/h	Good conditions	Crash into vehicle in adjacent lane or drive off the road.	C1	S3	E2	QM
HE10	– | | –	Slippery road	– | | –	C2	S3	E2	A
HE11	– | | –	Impaired vision	– | | –	C1	S3	E2	QM
HE12	– | | –	Slippery road, Impaired vision	– | | –	C2	S3	E2	A
HE13	None Highway, speed< 30 km/h	Good conditions	Crash into vehicle in adjacent lane or hit a VRU.	C1	S3	E4	B
HE14	– | | –	Slippery road	– | | –	C2	S3	E3	B
HE15	– | | –	Impaired vision	– | | –	C1	S3	E2	QM
HE16	– | | –	Slippery road, Impaired vision	– | | –	C2	S3	E2	A
HE17	None Highway, speed> 30 km/h	Good conditions	Crash into vehicle in adjacent lane or hit a VRU.	C2	S3	E4	C

Table continued on next page

	Operational situation (7.4.2.1.1)						
Identifier	Scenario	Weather conditions	Consequence (7.4.2.2.4)	C	S	E	ASIL
HE18	– | | –	Slippery road	– | | –	C3	S3	E3	C
HE19	– | | –	Impaired vision	– | | –	C2	S3	E2	A
HE20	– | | –	Slippery road, Impaired vision	– | | –	C3	S3	E2	B

Hazard H3. Vehicle does an unintended evasive maneuver by steering with unlimited torque (torque applied more than 10 Nm).

In Table 3.5 each hazardous event associated with hazard H3 is listed.

Table 3.5: Hazardous events associated with hazard H3.

	Operational situation (7.4.2.1.1)						
Identifier	Scenario	Weather conditions	Consequence (7.4.2.2.4)	C	S	E	ASIL
HE21	Highway,speed >30 km/h, collision not imminent	Good conditions	Crash into vehicle in adjacent lane or drive off the road.	C3	S3	E4	D
HE22	– | | –	Slippery road	– | | –	C3	S3	E3	C
HE23	– | | –	Impaired vision	– | | –	C3	S3	E2	B
HE24	– | | –	Slippery road, Impaired vision	– | | –	C3	S3	E2	B
HE25	Highway,speed <30 km/h	Good conditions	Crash into vehicle in adjacent lane or drive off the road.	C2	S3	E2	A
HE26	– | | –	Slippery road	– | | –	C3	S3	E2	B
HE27	– | | –	Impaired vision	– | | –	C2	S3	E2	A
HE28	– | | –	Slippery road, Impaired vision	– | | –	C3	S3	E2	B

Table continued on next page

	Operational situation (7.4.2.1.1)						
Identifier	Scenario	Weather conditions	Consequence (7.4.2.2.4)	C	S	E	ASIL
HE29	None Highway, speed< 30 km/h	Good conditions	Crash into vehicle in adjacent lane or hit a VRU.	C2	S3	E4	C
HE30	– | | –	Slippery road	– | | –	C3	S3	E3	C
HE31	– | | –	Impaired vision	– | | –	C2	S3	E2	A
HE32	– | | –	Slippery road, Impaired vision	– | | –	C3	S3	E2	B
HE33	None Highway, speed> 30 km/h	Good conditions	Crash into vehicle in adjacent lane or hit a VRU.	C3	S3	E4	D
HE34	– | | –	Slippery road	– | | –	C3	S3	E3	C
HE35	– | | –	Impaired vision	– | | –	C3	S3	E2	B
HE36	– | | –	Slippery road, Impaired vision	– | | –	C3	S3	E2	B

Hazard H4. Vehicle gives an unintended warning to the driver of an imminent collision.

In Table 3.6 each hazardous event associated with hazard H4 is listed.

Table 3.6: Hazardous events associated with hazard H4.

	Operational situation (7.4.2.1.1)						
Identifier	Scenario	Weather conditions	Consequence (7.4.2.2.4)	C	S	E	ASIL
HE37	Highway,speed >30 km/h, collision not imminent	Good conditions	Driver brakes and gets hit from behind	C0	S3	E4	N/A
Table continued on next page							

	Operational situation (7.4.2.1.1)						
Identifier	Scenario	Weather conditions	Consequence (7.4.2.2.4)	C	S	E	ASIL
HE38	– | | –	Slippery road	– | | –	C0	S3	E3	N/A
HE39	– | | –	Impaired vision	– | | –	C1	S3	E1	QM
HE40	– | | –	Slippery road, Impaired vision	– | | –	C1	S3	E1	QM
HE41	Highway,speed <30 km/h	Good conditions	Driver brakes and gets hit from behind	C0	S1	E2	N/A
HE42	– | | –	Slippery road	– | | –	C0	S1	E2	N/A
HE43	– | | –	Impaired vision	– | | –	C0	S1	E2	N/A
HE44	– | | –	Slippery road, Impaired vision	– | | –	C0	S1	E2	N/A
HE45	None Highway, speed< 30 km/h	Good conditions	Driver brakes and gets hit from behind.	C0	S1	E4	N/A
HE46	– | | –	Slippery road	– | | –	C0	S1	E3	N/A
HE47	– | | –	Impaired vision	– | | –	C0	S1	E2	N/A
HE48	– | | –	Slippery road, Impaired vision	– | | –	C0	S1	E2	N/A
HE49	None Highway, speed> 30 km/h	Good conditions	Driver brakes and gets hit from behind.	C0	S3	E4	N/A
HE50	– | | –	Slippery road	– | | –	C0	S3	E3	N/A
HE51	– | | –	Impaired vision	– | | –	C1	S3	E1	QM
HE52	– | | –	Slippery road, Impaired vision	– | | –	C1	S3	E1	QM

Hazard H5. Vehicle is about to crash into a forward vehicle but fails to warn the driver.

In Table 3.7 each hazardous event associated with hazard H5 is listed.

Table 3.7: Hazardous events associated with hazard H5.

	Operational situation (7.4.2.1.1)						
Identifier	Scenario	Weather conditions	Consequence (7.4.2.2.4)	C	S	E	ASIL
HE53	Highway,speed >30 km/h, collision imminent	Good conditions	Crash into forward vehicle	C0	S3	E3	N/A
HE54	– | | –	Slippery road	– | | –	C0	S3	E2	N/A
HE55	– | | –	Impaired vision	– | | –	C0	S3	E2	N/A
HE56	– | | –	Slippery road, Impaired vision	– | | –	C0	S3	E1	N/A

Hazard H6. Vehicle makes an evasive maneuver into an unsafe trajectory

In Table 3.8 each hazardous event associated with hazard H6 is listed.

Table 3.8: Hazardous events associated with hazard H6.

	Operational situation (7.4.2.1.1)						
Identifier	Scenario	Weather conditions	Consequence (7.4.2.2.4)	C	S	E	ASIL
HE57	Highway,speed >30 km/h, collision imminent	Good conditions	Crash into a forward vehicle, vehicle in adjacent lane or drive of the road.	C2	S3	E1	QM

Table continued on next page

	Operational situation (7.4.2.1.1)						
Identifier	Scenario	Weather conditions	Consequence (7.4.2.2.4)	C	S	E	ASIL
HE58	– | | –	Slippery road	– | | –	C3	S3	E1	A
HE59	– | | –	Impaired vision	– | | –	C3	S3	E1	A
HE60	– | | –	Slippery road, Impaired vision	– | | –	C3	S3	E1	A

3.6 Reflections and Deviations in Hazard Analysis and Risk Assessment

This section presents reflections on and deviations from ISO-26262 in Hazard Analysis and Risk Assessment.

Controllability, Exposure and Severity

When deciding controllability, exposure and severity we used the tables given in ISO-26262 [6, Part 3, Annex B] and had a discussion about each hazardous event. This is not sufficient when strictly following ISO-26262. If one were to follow ISO-26262 correctly, when deciding controllability, exposure or severity, each decision need to be based on a defined rationale. For example by performing tests or analyzing statistics.

This is an example of how we reasoned when we decided ASIL for the hazardous event HE5. The scenario in this case is that the vehicle is driving on a highway with a velocity greater than 30 km/h and a collision is not imminent. We combine this scenario together with the weather conditions, which in this case is good weather in order to get an operational situation.

This operational situation is used to determine the exposure. That is the probability that the vehicle and the driver is in this particular operational situation. The criteria for the exposure rate E4 is that the operational situation should occur almost every drive and it seems quite reasonable to assume that a heavy vehicle is being driven on a highway under good weather condition almost every drive. Thus, the hazardous event was given the exposure rate of E4.

The consequence of hazardous event HE5 is that the heavy vehicle crashes into another vehicle in an adjacent lane or drive off the road. This is used to determine the severity. The criteria for a severity of S3 is that in more than 10% of the cases where this consequense happens someone receives fatal injuries. We determined the chance for a fatality to be greater than 10% when a heavy vehicle crashes into

another vehicle and thus we gave the hazardous event a severity rating of S3.

The last parameter to determine is the controllability. This is a measurement of how probable it is for the driver and other persons potentially at risk to gain control of the hazardous event, such that they are able to avoid the specific harm. In our case the vehicle is making an unintended evasive maneuver but with a torque that should be possible for most drivers to counter by steering into the other direction. The criteria for a controllability of C2 is that 90% or more of all drivers or other trafic participants are usually able to avoid harm. To be honest we both believe that maybe 99% or more than all drivers or participants usually would be able to control the situation, and this is the criteria for C1 but since none of us have any experience with driving a heavy vehicle we decided that we would rather be safe than sorry so we went for a controllability rate of C2.

Regarding Controllability of Unavailable Item

[6, Part3, 7.4.3.8] tells us that "Class C0 may be used for hazard addressing the unavailability of the item if it does not affect the safe operation of the vehicle."[2] This paragraph in ISO-26262 is the reason why the hazardous events associated with H5 did not get an ASIL. One can debate if the hazardous events associated with H1 could get C0 due to this clause as well. However since these hazardous events implies that the vehicle is in a situation which is considered impossible for the driver to gain control over, we chose to assign C3 to those particular events.

Operating Modes

According to [6, Part3, 7.4.2.1.1] the operating modes of the item should be taken into consideration when performing the situation analysis. We do not see any reason, since being in the wrong mode is one of the potential causes to the different hazards, to doing so in this case and has therefore left that part out.

Operational Situations

According to ISO-26262 [6, Part3, 7.4.4.2] it shall be made sure that the chosen operational situations are not too detailed and not too many since this can lead to a lower ASIL classification due to the fact that each operational situation get a very low exposure classification. As can be seen in this chapter, we have quite many operational situations. However, we feel that they are needed due to their impact on especially controllability. To compensate for this we always chose the higher exposure classification when in slightest doubt.

[2]The text is reproduced with the permission of the SIS Förlag AB, www.sis.se, 08-555523 10

There are two reasons to why we chose 30 km/h as a velocity limit between different situations. Firstly it is because the functionality of the item should only be active while driving above 30 km/h and on a highway. The second reason is that a collision with a heavy vehicle with a velocity below 30 km/h does not necessarily mean certain death, while a collision with a heavy vehicle with a velocity greater than 30 km/h most likely does.

ASIL Levels

We have two reflections when it comes to ASIL levels. Firstly, if a hazardous event has the parameters S3, E4 and C1 assigned, the resulting ASIL classification is ASIL B. However, if C1 is changed into C0 the hazardous event does not even get an assigned ASIL. Having no ASIL assigned to a hazardous event with S3 and E4 does not seem reasonable to us, even if the hazardous event can be controlled in general. The reason behind this is that a hazardous event with the parameter C0 does not get an ASIL classification. The mentioned phenomena does not follow the "normal" procedure of ISO-26262 where as in any other case the ASIL will get lowered one step if one of the three parameters are lowered one step, which is logical. Therefore we think it would be a good idea to include C0 in the table used to calculate an ASIL classification given the three parameters, see [6, Part3, Table 4]. This would allow the ASIL classification to follow a logical pattern in all cases.

Secondly, if an adaption of ISO-26262, aimed at heavy vehicles, were to be made it might be a good idea to introduce a higher severity scale such as S4 and maybe even S5. The reason for this is that accidents involving heavy vehicles have the potential to be much more severe than accidents with passenger cars. For example if a bus loaded with people drives off the road the consequences are much worse than if the same thing would happen with a passenger car. As a result of the higher severity scale a new ASIL classification could be introduced, which could for example force the given functionality to be implemented with redundancy.

Torque Component

Hazard H2 and Hazard H3 contains a measurement of torque (10 Nm). This is an estimate and has no scientific background.

3.7 3-7:WP2 Safety Goals

This section is a work product resulting from requirements 7.4.4.3 to 7.4.4.6 in Part 3 of the ISO-26262 standard. The aim is to formulate safety goals that avoids

or mitigates the hazardous events.

3.7.1 Safe state and Fault Tolerant Time Interval

A safe state is an operational mode of the item without an unreasonable level of risk. A safe state must not necessarily implement the intended functionality of the item. Therefore modes like switched off or modes with degraded functionality can be considered as safe states.

A safe state for our item is called "Fail Safe". Fail Safe is a degraded state which the item will enter when a fault is detected, in order to prevent a hazardous event from occurring. In Fail Safe the item stops all normal execution, and thus an evasive maneuver will never be performed. When in Fail Safe, a warning light on the dashboard will be lit to inform the driver that the item is not functional.

The fault tolerant time states the duration a fault can be present in the system before a hazardous event occurs. Hence when a fault occurs, the item need to transition into Fail Safe state within the fault tolerant time interval. The fault tolerant time is thus later used to derive how fast a safety mechanism must react, in order to prevent a hazardous event from occurring.

3.7.2 List of Safety Goals

In Table 3.9 all safety goals are listed. Each hazardous event listed in Table 3.3 through Table 3.8 with an assigned ASIL (ASIL A to ASIL D) is given a safety goal. However, due to their similarity, many safety goals are combined into a single safety goal. The column "From Hazardous Event" states which hazardous event(s) the particular safety goal was derived from. The safety goals are given the same ASIL as the hazardous event from which they were derived. If several safety goals are combined, the combined safety goal is given the highest ASIL of the original safety goals [6, Part3, 7.4.4.4].

Table 3.9: Safety Goals of the Item

Identifier	**Safety Goal**	**From Hazardous Event**	**Safe state**	**Fault tolerant time**	**ASIL**
SG1.	For a velocity input > 30 km/h and sensor data that supports the fact that the vehicle is traveling on a highway: The item must always apply torque to the steering shaft when sensor data indicates that an evasive maneuver should be done.	HE1 to HE4	Fail safe	100 ms	A
SG2.	For a velocity input > 30 km/h and sensor data that supports the fact that the vehicle is traveling on a highway: The item must never apply limited torque (torque applied less than 10 Nm) to the steering shaft when sensor data indicates that an evasive maneuver should not be done.	HE5 to HE8	Fail Safe	100ms	C
SG3.	For a velocity input < 30 km/h and sensor data that supports the fact that the vehicle is traveling on a highway: The item must never apply limited torque (torque applied less than 10 Nm) to the steering shaft.	HE9 to HE12	Fail Safe	100ms	A
SG4.	For sensor data that does not support the fact that the vehicle is traveling on a highway: The item must never apply limited torque (torque applied less than 10 Nm) to the steering shaft.	HE13 to HE20	Fail Safe	100ms	C
Table continued on next page					

Identifier	Safety Goal	From Hazardous Event	Safe state	Fault tolerant time	ASIL
SG5.	For a velocity input > 30 km/h and sensor data that supports the fact that the vehicle is traveling on a highway: The item must never apply unlimited torque (torque applied more than 10 Nm) to the steering shaft when sensor data indicates that an evasive maneuver should not be done.	HE21 to HE24	Fail Safe	20ms	D
SG6.	For a velocity input < 30 km/h and sensor data that supports the fact that the vehicle is traveling on a highway: The item must never apply unlimited torque (torque applied more than 10 Nm) to the steering shaft.	HE25 to HE28	Fail Safe	20ms	B
SG7.	For sensor data that does not support the fact that the vehicle is traveling on a highway: The item must never apply unlimited torque (torque applied more than 10 Nm) to the steering shaft.	HE29 to HE36	Fail Safe	20ms	D
SG8.	When applying torque to the steering shaft: The item must never apply torque in such a way that the planned trajectory is unsafe according to the sensor data.	HE56 to HE60	N/A	N/A	A

3.8 Reflections and Deviations in Safety Goals

This section presents reflections on and deviations from ISO-26262 in Safety Goals.

Combining Safety Goals

As seen in Table 3.9 many of the safety goals are similar, even after combining several of the original ones. It can be argued that we should have combined even

more safety goals. The problem is that when combining safety goals, some architectural solutions to avoiding violation of safety goals might be lost in the process.

For example, if SG2 (ASIL C) and SG5 (ASIL D) are combined, the resulting combined safety goal could look like this: "For a velocity input > 30 km/h and sensor data that supports the fact that the vehicle is traveling on a highway: The item must never apply torque to the steering shaft when sensor data indicates that an evasive maneuver should not be done." This safety goal would get ASIL D, since SG5 was ASIL D. Lets call this safety goal SGex for future reference. Now this might not at a first glance seem like a problem, but what has happened is that a possible architectural solution has been lost. Before combining the safety goals, a simple mechanism (with ASIL D), such as a current limiter, could have been implemented that prevent the electric motor from applying more than 10 Nm to the steering shaft. This would prevent SG5 from being violated. The functionality that make sure that torque is only applied when intended, which is much more complex, would then be implemented according to ASIL C to prevent SG2 from being violated.

Now to the combined safety goal. To prevent SGex from being violated, the functionality that make sure that torque is only applied when intended has to be implemented according to ASIL D. So instead of having a small part of the item implemented according to ASIL D and a big part implemented according to ASIL C, you now have a big part that has to be implemented according to ASIL D.

On the other hand, combining safety goals will most likely lead to less requirements in subsequent phases, which means less workload given that the above example does not happen. A good rule of thumb could be that combining safety goals with the same ASIL is no problem, but if they have different ASIL one should be very cautious.

Formulating the Safety Goals

We want a safety goal that will cover HE1 - HE4. These hazardous events tells us that the vehicle is driving on a highway with a velocity greater than 30 km/h and is about to crash into a forward vehicle. However, the vehicle fails to do an evasive maneuver. This safety goal can be described in two different ways depending on if we want to formulate it on item level or on vehicle level. If we formulated the safety goal on vehicle level it would look something like this: "For speeds > 30 km/h and on highway: The item should always make an evasive maneuver by steering when the situation requires."

For this safety goal to be fulfilled we require three things:

- We require that the item receives correct inputs that reflect the actual behavior of the vehicle. For example if the systems external to the item calculates

a faulty velocity and provides the item with this information it might cause the safety goal to be violated.

- We require the item itself to function correctly.
- We require that the outputs from our item leads to a correct behavior at the vehicle level. For example, when the electric motor applies torque to the steering shaft we require that this leads to a change of vehicle direction.

This can simply be translated to that if our items works as intended and all external requirements are fulfilled then the safety goal will be fulfilled.

Figure 3.4 illustrates how the requirements ensures that the hazardous events are avoided when the safety goals are written on vehicle level.

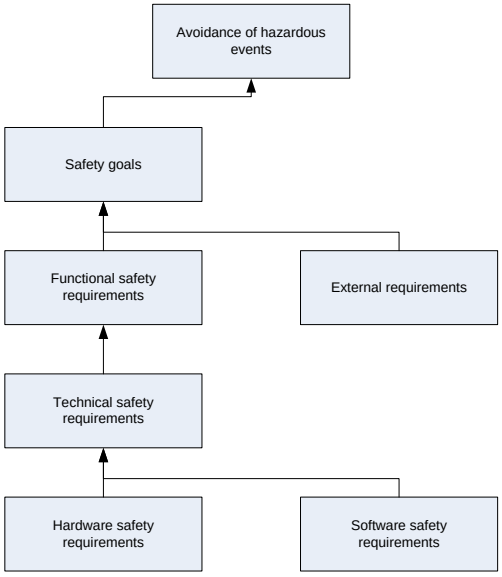

Figure 3.4: Avoidance of hazardous events with vehicle level safety goals

The other way of formulating the safety goal is to formulate it on item level. It would then look like our safety goal SG1, namely: "For a velocity input > 30 km/h and sensor data that supports the fact that the vehicle is traveling on a highway: The item should always apply torque to the steering shaft when sensor data indicates that an evasive maneuver should be done."

A formulation like this only requires the item to function correctly in order to ensure the fulfillment of the safety goal. However this does not mean that the avoidance of hazardous events is ensured. In this case, in order to guarantee the avoidance of hazardous events not only the safety goals must be fulfilled but also the external requirements. Since a faulty input or a faulty handling of an output still could lead to a hazardous event.

Figure 3.5 illustrates how the requirements ensures that the hazardous events are avoided when the safety goals are written on item level.

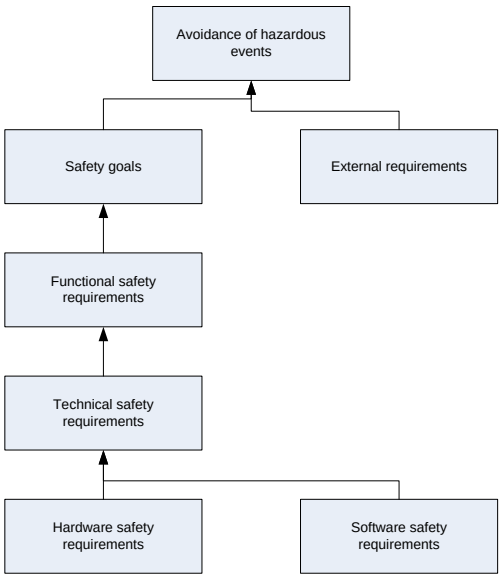

Figure 3.5: Avoidance of hazardous events with item level safety goals

We have chosen to formulate our safety goals at the item level. The reason for this is that it will be easier for our item to comply with the safety goals and require less work because we don't need any safety mechanisms in order to detect external failures. However, even though we believe that writing the safety goals at item level complies with ISO-26262, the vehicle will not be as safe as an item designed with safety goals at the vehicle level. As said, we believe that this approach is compliant with ISO-26262 as we have not found anything in ISO-26262 that states otherwise. Though it shall be noted that [6, Part4, 6.4.2.2 b] becomes obsolete when formulating the safety goals this way, which indicates that it is not wrong to formulate the safety goals on vehicle level.

3.9 Preliminary Architecture

This is not a work product required by Part 3 of ISO-26262, but an external document supporting the work required by part 3. It contains information about the preliminary architectural assumptions made so far about the design of the system. Figure 3.6 illustrates this design. It shows every element of the item and the interactions between elements.

The preliminary architectural assumptions has evolved a lot during the course of this thesis. See Appendix B for an overview of the development.

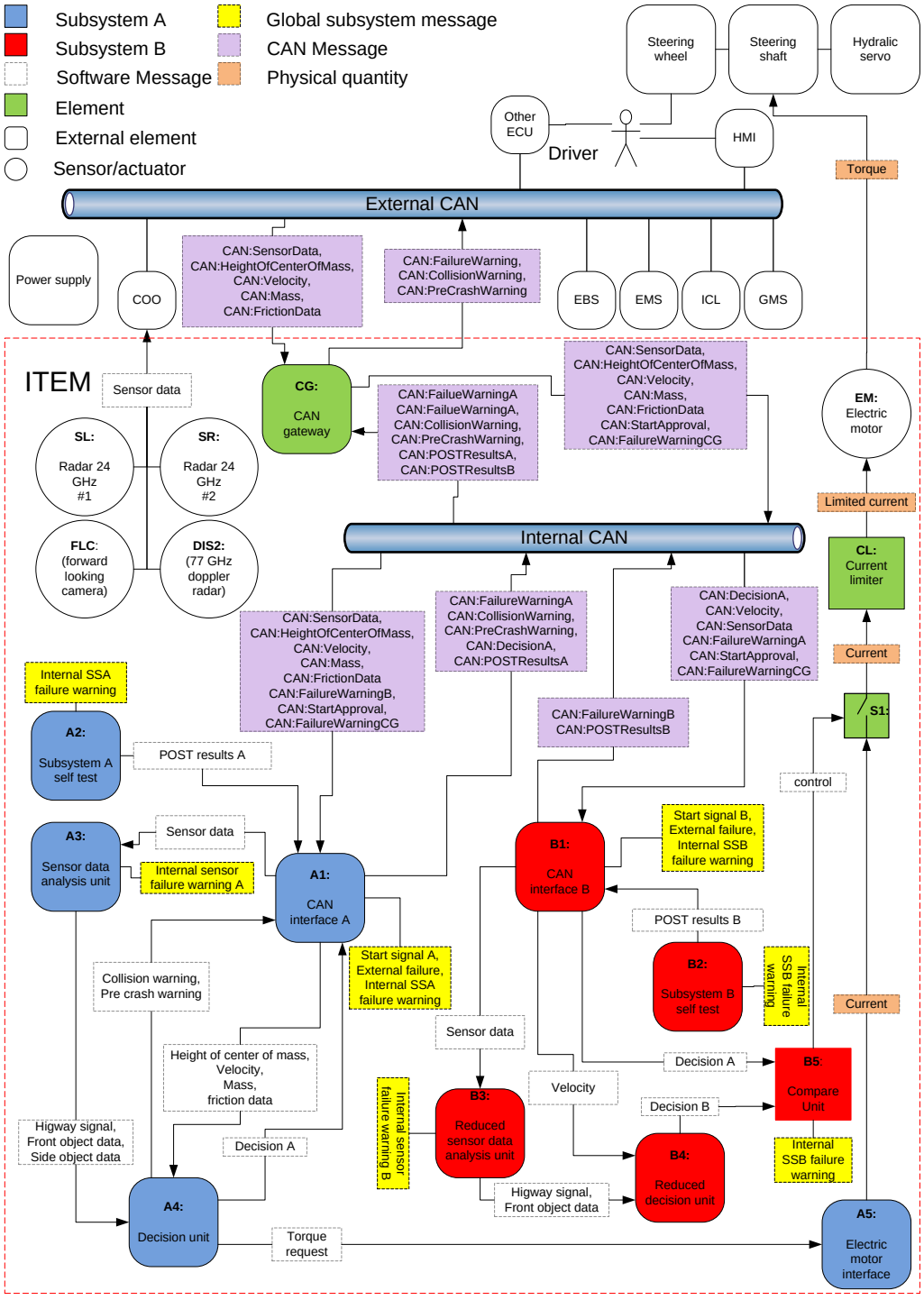

Figure 3.6: The preliminary architectural assumptions.

3.10 3-8:WP1 Functional Safety Concept

This section is a work product resulting from requirements 8.4.1 to 8.4.4 in Part 3 of the ISO-26262 standard. The aim is to derive the functional safety requirements from the safety goals and to allocate them to the preliminary architectural elements.

The compliance with the functional safety requirements implies the fulfillment of the safety goals. Thus if all functional safety requirements derived from a safety goal are fulfilled then the safety goal will not be violated.

The work done in this section is a result of several iterations. To see the first iteration that was made, the basis for this work product, see Appendix A.

The functional safety requirements are listed in Table 3.10 and Table 3.11. When a functional safety requirement has an ASIL in parenthesis it is a decomposed requirement. The ASIL inside the parenthesis states the original ASIL assigned, as demanded by ISO-26262 [6, Part9, 5.4.9c].

Expressions like "front object data = CollisionImminent" should be interpreted as a description of the meaning of the message and does not mean that the message will look exactly this.

3.10.1 Fail Safe for Elements

As mentioned in 3.7.1 Fail Safe is a state in our item where the item shall not be able to steer the vehicle and the driver should be warned about the item not functioning. This means that the elements in the item must take different actions when in Fail Safe. These are the actions taken by the elements:

- Compare Unit

  When in Fail Safe the compare unit must send the control signal which opens the switch, thus preventing any current from reaching the electric motor and disabling steering capabilities of the item.

- CAN gateway

  The CAN gateway must send a failure message on the external CAN bus in order to warn the driver that the item is malfunctioning.

- Other elements

  All other elements must stop execution. The reason for this is to disable steering capabilities for the item in case of a malfunction in the compare unit or the switch.

3.10.2 Functional Safety Requirements

In Table 3.10 and Table 3.11 all the functional safety requirements are listed.

If N/A is written in the Fault tolerant time column, the requirement belongs to a safety mechanism. A reaction time is instead written explicitly in the requirement it self. This reaction time has been derived from the fault reaction time belonging to the requirement(s) which the safety mechanism protects.

Table 3.10: Final functional safety requirements derived from all safety goals

Identifier	**Functional Safety Requirement**	**From Safety Goal**	**Fault tolerant time**	**Allocated to element**	**ASIL**
FSR1.	Faulty messages received from External CAN must be detected.	All	100ms	CG	C
FSR2.	Decision unit must always calculate a decision to perform an evasive maneuver if highway signal = OnHighway AND front object data = CollisionImminent AND velocity signal > 30 km/h.	SG1	100ms	A4	A
FSR3.	Reduced decision unit must always calculate a decision to perform an evasive maneuver if if highway signal = OnHighway AND front object data = CollisionImminent AND velocity signal > 30 km/h.	SG1	100ms	B4	A
Table continued on next page					

Identifier	Functional Safety Requirement	From Safety Goal	Fault tolerant time	Allocated to element	ASIL
FSR4.	The messages forwarded from External CAN to Internal CAN must not be corrupted in the process.	All	100ms	CG	C
FSR5.	Decision unit must never calculate a decision to perform an evasive maneuver by steering if front object data = CollisionNotImminent.	SG2	100ms	A4	A (C)
FSR6.	Decision unit must never calculate a decision to perform an evasive maneuver by steering if highway signal = NotOnHighway.	SG4	100ms	A4	A (C)
FSR7.	Reduced decision unit must never calculate a decision to perform an evasive maneuver by steering if front object data = CollisionNotImminent.	SG2	100ms	B4	B (C)
FSR8.	Reduced decision unit must never calculate a decision to perform an evasive maneuver by steering if highway signal = NotOnHighway.	SG4	100ms	B4	B(C)
FSR9.	The DIS2 must send correct data to the COO.	SG1, SG2, SG4, SG8	100ms	DIS2	B (C)
FSR10.	The FLC must send correct data to the COO.	SG1, SG2, SG4, SG8	100ms	FLC	A (C)

Table continued on next page

Identifier	Functional Safety Requirement	From Safety Goal	Fault tolerant time	Allocated to element	ASIL
FSR11.	Sensor data analysis unit must perform a correct fusion of sensor data given by DIS2 and FLC.	SG1, SG2, SG4	100ms	A3	A (C)
FSR12.	Reduced sensor data analysis unit must perform a correct fusion of sensor data given by DIS2 and FLC.	SG1, SG2, SG4	100ms	B3	B (C)
FSR13.	The switch must never be closed if control signal = OpenSwitch.	SG2, SG3, SG4, SG6	100ms	S1	B (C)
FSR14.	Electric motor interface must never apply current to the electric motor unless given a torque request.	SG2, SG3, SG4, SG6	100ms	A5	A (C)
FSR15.	Decision unit must never calculate a decision to perform an evasive maneuver by steering if velocity signal <= 30 km/h.	SG3, SG6	100ms	A4	A (B)
FSR16.	Reduced decision unit must never calculate a decision to perform an evasive maneuver by steering if velocity signal <= 30 km/h.	SG3, SG6	100ms	B4	A (B)
FSR17.	No more than 20 A current may be applied to the electric motor by Electric motor interface (to prevent uncontrollable torque).	SG5, SG7	20ms	A5	A (D)
FSR18.	Current limiter should limit the current so that no more than 20 A is applied to the electric motor.	SG5, SG7	20ms	CL	C (D)

Table continued on next page

Identifier	Functional Safety Requirement	From Safety Goal	Fault tolerant time	Allocated to element	ASIL
FSR19.	Decision unit must always calculate a safe trajectory when sending a torque request.	SG8	100ms	A4	A
FSR20.	The sensor must send correct sensor data to the COO.	SG8	100ms	SR	A
FSR21.	The sensor must send correct sensor data to the COO.	SG8	100ms	SL	A
FSR22.	Sensor data analysis unit must perform a correct fusion of all sensor data.	SG8	100ms	A3	A
FSR23.	The electric motor must apply torque in proportion to the current received.	SG1, SG8	100ms	EM	A
FSR24.	Electric motor interface must apply current to the electric motor that is proportional to the torque requested.	SG1, SG8	100ms	A5	A
FSR25.	Compare unit must send control signal = CloseSwitch so that current may flow to Current limiter when both Decision unit and Reduced decision unit signals for an evasive maneuver.	SG1, SG8	100ms	CU	A
FSR26.	The switch must always close when control signal = CloseSwitch.	SG1, SG8	100ms	S1	A
FSR27.	Current below 20 A must not be limited.	SG1, SG8	100ms	CL	A
FSR28.	Reduced sensor data analysis unit and Subsystem A must be independently implemented.	SG2, SG3, SG4, SG6	N/A	B3	C

Table continued on next page

Identifier	Functional Safety Requirement	From Safety Goal	Fault tolerant time	Allocated to element	ASIL
FSR29.	Reduced decision unit and Subsystem A must be independently implemented.	SG2, SG3, SG4, SG6	N/A	B4	C
FSR30.	Compare unit and Subsystem A must be independently implemented.	SG2, SG3, SG4, SG6	N/A	CU	C
FSR31.	CAN interface B and Subsystem A must be independently implemented.	SG2, SG3, SG4, SG6	N/A	B1	C
FSR32.	Subsystem B self test and Subsystem A must be independently implemented.	SG2, SG3, SG4, SG6	N/A	B2	C
FSR33.	Sensor data analysis unit and Subsystem B must be independently implemented.	SG2, SG3, SG4, SG6	N/A	A3	C
FSR34.	Decision unit and Subsystem B must be independently implemented.	SG2, SG3, SG4, SG6	N/A	A4	C
FSR35.	CAN interface A and Subsystem B must be independently implemented.	SG2, SG3, SG4, SG6	N/A	A1	C
FSR36.	Subsystem A self test and Subsystem B must be independently implemented.	SG2, SG3, SG4, SG6	N/A	A2	C
FSR37.	Electric motor interface and Subsystem B must be independently implemented.	SG2, SG3, SG4, SG6	N/A	A5	C

Table continued on next page

Identifier	Functional Safety Requirement	From Safety Goal	Fault tolerant time	Allocated to element	ASIL
FSR38.	Messages forwarded to Internal CAN from Subsystem A must be correct.	All	100ms	A1	A (C)
FSR39.	Messages forwarded to Subsystem A from Internal CAN must be correct.	All	100ms	A1	A (C)
FSR40.	Messages forwarded to Internal CAN from Subsystem B must be correct.	All	100ms	B1	B (C)
FSR41.	Messages forwarded to Subsystem B from Internal CAN must be correct.	All	100ms	B1	B (C)
FSR68.	CAN gateway must forward SensorData messages from External CAN to Internal CAN	SG1, SG8	100ms	CG	A
FSR69.	CAN gateway must forward HightOfCenterOfMass messages from External CAN to Internal CAN	SG1, SG8	100ms	CG	A
FSR70.	CAN gateway must forward Velocity messages from External CAN to Internal CAN	SG1, SG8	100ms	CG	A
FSR71.	CAN gateway must forward Mass messages from External CAN to Internal CAN	SG1, SG8	100ms	CG	A
FSR72.	CAN gateway must forward FrictionData messages from External CAN to Internal CAN	SG1, SG8	100ms	CG	A
FSR73.	CAN interface A must forward SensorData messages to Sensor data analysis unit.	SG1, SG8	100ms	A1	A
FSR74.	CAN interface A must forward HeightOfCenterOfMass messages to Decision unit.	SG1, SG8	100ms	A1	A

Table continued on next page

Identifier	Functional Safety Requirement	From Safety Goal	Fault tolerant time	Allocated to element	ASIL
FSR75.	CAN interface A must forward Velocity messages to Decision unit.	SG1, SG8	100ms	A1	A
FSR76.	CAN interface A must forward Mass messages to Decision unit.	SG1, SG8	100ms	A1	A
FSR77.	CAN interface A must forward FrictionData messages to Decision unit.	SG1, SG8	100ms	A1	A
FSR78.	CAN interface B must forward SensorData messages to Reduced sensor data analysis unit.	SG1, SG8	100ms	B1	B (C)
FSR79.	CAN interface B must forward DecisionA messages to Reduced decision unit.	SG1, SG8	100ms	B1	B (C)
FSR80.	CAN interface B must forward Velocity messages to Reduced decision unit.	SG1, SG8	100ms	B1	B (C)
FSR81.	Decision unit must send DecisionA to CAN Interface A when a decision has been made.	SG1, SG8	100ms	A4	A
FSR82.	CAN interface A must forward DecisionA to Internal CAN.	SG1, SG8	100ms	A1	A

Warning and Degradation Concept

In Table 3.11 the warning and degradation concept is described as functional safety requirements, as required by ISO-26262 [6, Part3, 8.4.2.5].

Table 3.11: Warning and degradation concept described as functional safety requirements

Identifier	Functional Safety Requirement	From Safety Goal	Fault tolerant time	Allocated to element	ASIL
FSR42.	Compare unit must send control signal = OpenSwitch so that no current can flow to Current limiter if Decision unit and Reduced decision unit has sent different decisions within 100ms.	SG2, SG3, SG4, SG6	N/A	CU	B (C)
FSR43.	If Decision unit is notified of a failure, it must transition into Fail Safe state within 100ms.	All	N/A	A4	A (C)
FSR44.	If CAN interface A is notified of a failure, it must transition into Fail Safe state within 100ms.	All	N/A	A1	A (C)
FSR45.	If Sensor data analysis unit is notified of a failure, it must transition into Fail Safe state within 100ms.	All	N/A	A3	A (C)
FSR46.	If electric motor interface is notified of a failure, it must transition into Fail Safe state within 100ms.	All	N/A	A5	A (C)
FSR47.	If CAN interface B is notified of a failure, it must transition into Fail Safe state within 100ms.	All	N/A	B1	B (C)
FSR48.	If Reduced decision unit is notified of a failure, it must transition into Fail Safe state within 100ms.	All	N/A	B4	B (C)

Table continued on next page

Identifier	Functional Safety Requirement	From Safety Goal	Fault tolerant time	Allocated to element	ASIL
FSR49.	If Reduced sensor data analysis unit is notified of a failure, it must transition into Fail Safe state within 100ms.	All	N/A	B3	B (C)
FSR50.	If Compare unit is notified of a failure, it must transition into Fail Safe state within 100ms.	All	N/A	CU	B (C)
FSR51.	If Compare unit receives different decisions from Decision unit and Reduced decision unit, it must notify the other elements of the item that a failure has occurred within 100ms.	SG1, SG2, SG3, SG4, SG6	N/A	CU	B(C)
FSR52.	If Subsystem A self test finds a failure, it must notify the other elements of the item that a failure has occurred within 100ms.	All	N/A	A2	A (C)
FSR53.	If Subsystem B self test finds a failure, it must notify the other elements of the item that a failure has occurred within 100ms.	All	N/A	B2	B (C)
FSR54.	If CAN gateway finds a failure, it must notify the other elements of the item that a failure has occurred within 100ms.	All	N/A	CG	C
FSR56.	To recover from Fail Safe state, the vehicle need to be restarted and pass a self test.	All	N/A	CG	C

Table continued on next page

Identifier	Functional Safety Requirement	From Safety Goal	Fault tolerant time	Allocated to element	ASIL
FSR57.	To recover from Fail Safe state, the vehicle need to be restarted and pass a self test.	All	N/A	Subsystem A	A(C)
FSR58.	To recover from Fail Safe state, the vehicle need to be restarted and pass a self test.	All	N/A	Subsystem B	B(C)
FSR59.	When in Fail Safe, Compare unit must never send control signal = CloseSwitch.	SG1, SG2, SG3, SG4, SG6	N/A	CU	B(C)
FSR60.	When in Fail Safe, the CAN gateway must send a message to HMI so that a warning light is turned on (to reduce the risk exposure time) within 100ms.	SG1	N/A	CG	A
FSR61.	When in Fail Safe, CAN interface A must notify the other elements of the item that a failure has occurred within 100ms.	All	N/A	A1	A (C)
FSR62.	When in Fail Safe, CAN interface B must notify the other elements of the item that a failure has occurred within 100ms.	All	N/A	B1	B (C)
FSR63.	If data from FLC and DIS2 is not plausible the sensor data analysis unit must notify the other elements that a failure has occurred within 100ms.	SG1, SG2, SG3, SG4, SG6, SG8	N/A	A3	A(C)

Table continued on next page

Identifier	Functional Safety Requirement	From Safety Goal	Fault tolerant time	Allocated to element	ASIL
FSR64.	If data from FLC and DIS2 is not plausible the reduced sensor data analysis unit must notify the other elements that a failure has occurred within 100ms.	SG1, SG2, SG3, SG4, SG6, SG8	N/A	B3	B(C)
FSR65.	Subsystem A self test must detect random hardware failures in Subsystem A.	All	N/A	A2	A (C)
FSR66.	Subsystem B self test must detect random hardware failures in Subsystem B.	All	N/A	B2	B (C)
FSR67.	If CAN gateway is notified of a failure, it must transition into failsafe within 100ms.	All	N/A	CG	C

Preliminary Architecture

In Figure 3.7 the Preliminary Architecture is displayed. Each element has been assigned an ASIL, in respect to the functional safety requirements they implement. If an element implements several functional safety requirements, the element is assigned the highest ASIL of the functional safety requirement they implement according to ISO-26262 [6, Part3, 8.4.3.1b].

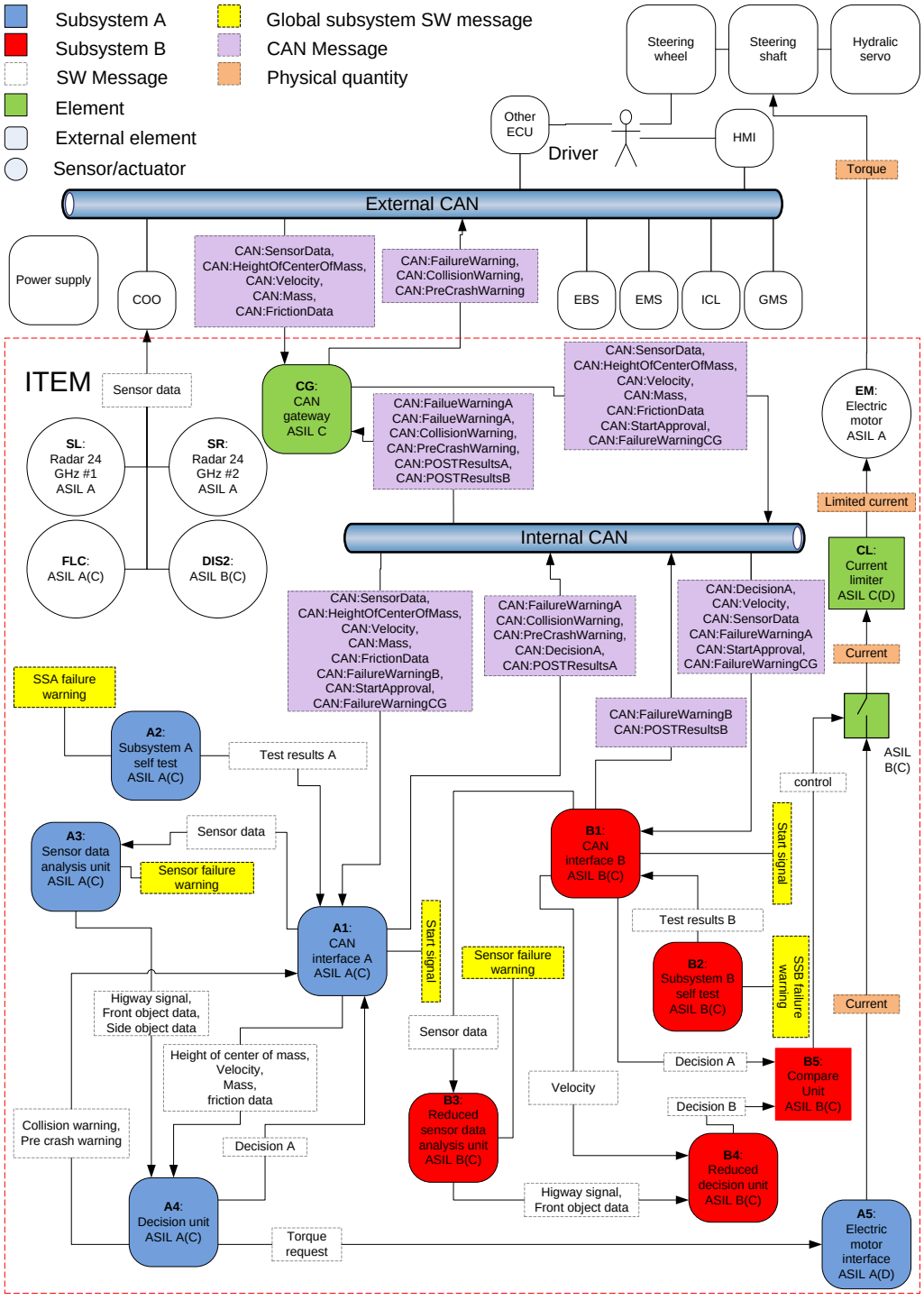

Figure 3.7: Preliminary Architecture, with an ASIL assigned to each element.

3.11 Reflections and Deviations from ISO-26262 in Functional Safety Concept

This section presents reflections on and deviations from ISO-26262 in Functional Safety Concept.

Level of Detail

A still existing question is at what level of detail the preliminary architecture, see Figure 3.7, should be described at. Our preliminary architecture is pretty detailed with many different elements, which in turn results in many functional safety requirements. Another approach could have been to simply combine all elements in Subsystem A into one element, and all elements in Subsystem B into one element. The functional safety requirements would then have been assigned to the different subsystems, rather than specific elements in the subsystems as it is now. As an effect of this, the number of functional safety requirements would be smaller, but the number of technical safety requirements greater. So what is best? If we would start from the beginning again, we would most certainly use the latter approach. The reason behind this is that the gap between safety goals and functional safety requirements would be smaller, thus providing a better overview and it becomes easier to perform between safety goals and functional safety requirements. Another benefit of using the latter approach is that the difference between functional and technical safety requirements is more distinguishable. Also, with the subsystem approach, adding new elements in the upcoming phases is not a problem since each subsystem would be a "black box" here in Part 3. As it is now, if we want to add a new element in Part 4 we have to iterate back to Part 3 (which we also have done) to keep the preliminary architecture consistent.

Another question is, when using our approach, is it still allowed to allocate functional safety requirements to an entire subsystem? I.e. allocate some functional safety requirements to specific elements inside a subsystem and allocate other functional safety requirements to an entire subsystem. As can be seen in Table 3.11 we have done exactly that. We cannot answer this question with certainty, however we have found nothing that implies that it is not allowed, nor have we found anything that implies that it is allowed.

Safety Analyses

When specifying the functional safety requirements, ISO-26262 states that one can use safety analysis techniques such as FMEA, FTA or HAZOP in order to develop a complete set of functional safety requirements. As we have no experience

with the mentioned analyses and at the same time felt confident that we could produce a complete set of functional safety requirements without it, we chose not to use any of them. However, if knowledge of any of the methods exists, it is highly recommended to use one of them as it will become easier to prove that the functional safety requirements specified are complete and correct.

Fault Tolerant Time Interval

Another remark is the fault tolerant time interval for each functional safety requirement and safety goal. In [6, Part4, 6.4.2.3] it states "In-vehicle testing and experimentation can be used to determine the fault tolerant time interval."[3] As there is no guidance to how to determine fault tolerant time interval in part 3 of ISO-26262, we are assuming the same means could be used in Part 3. However, as we lack the means of doing these tests, the fault tolerant time intervals stated in this work product, and the work product Safety Goals, are pure estimates.

Emergency Operation

The ISO-26262 standard, [6, Part3, 8.4.2.4], states that "If a safe state cannot be reached by a transition within an acceptable time interval, an emergency operation shall be specified."[3]. How should this be interpreted though? Does it mean that an emergency operation shall be specified for those requirements which do not have a safe state? Or specified for those requirements that does have a safe state, but it is not possible to reach it within an acceptable time interval? Or maybe even specified for all requirements just in case? Even if these questions were answered, what is an emergency operation? We have not found an answer to the last question in the ISO-26262 standard, nor anywhere else. Due to all these unanswered questions, we have chosen not to specify any emergency operations, since it would be based on guesses.

ASIL Decomposition

When performing ASIL decomposition, ISO-26262 states that the elements used for decomposition need to be independently implemented. But to what extent? For example, if you implement two redundant elements on two different ECUs, does these ECUs have to be bought from two different manufacturers? And if they are, what if these ECU manufacturers buy their components from the same supplier? And so on. If somewhere along the line, they actually do contain a part that is bought from the same supplier, this part could contain some fault that will

[3]The text is reproduced with the permission of the SIS Förlag AB, www.sis.se, 08-555523 10

cause both ECUs to fail due to the same manufacture error. Making sure that this does not happen all the way down to the last supplier is not feasible, so where does the standard draw the line? This is not, as far as we can see, made clear.

The reason we are asking us this question is that in the future, when and if ISO-26262 becomes required by law, vehicle manufacturers will want to put redundant elements on the same ECU for different reasons such as cost and area consumption. This ECU would obviously need separate CPUs, memories etc. But will it also need separate power supplies? Does the CPUs need to be of different brands and type? Questions like these need to be clearly answered in ISO-26262.

Current Limitation

Some of the functional safety requirements contain a value of current. This value is an estimate and has no scientific background.

Chapter 4

Part 4 - Product Development at the System Level

This chapter concerns the work that has been done in Part 4 of ISO 26262. The chapter begins with a section explaining objectives of Part 4 and then continues with the work products that have been produced during Part 4. After each work product reflections on, and deviations from, ISO 26262 are presented.

Only the element CAN gateway will be treated in this chapter.

4.1 Regarding Notation

In this chapter there are some notations that needs explaining. The notation used to describe a CAN message is CAN:ID.DATA. The notation used to assign a variable is Var:Name := VALUE and finally the notation for comparing a variable against a value is Var:Name == VALUE.

4.2 Objectives of Part 4 According to ISO 26262

This section attempts to explain the objectives and purpose of part 4 - Product development at the system level.

4.2.1 Technical Safety Requirement Specification Explanation

The first objective of Part 4 is to write the technical safety requirements specification. The purpose of writing this work product is to refine the functional safety requirements specified in part 3 into more detailed technical requirements. The idea is that if all technical safety requirements that are derived from a functional safety requirement are fulfilled then the functional safety requirement is also fulfilled.

4.2.2 Hardware Software Interface Specification (HSI) Explanation

The objective of this sub phase is to specify the hardware and software interaction. The HSI includes the component's hardware devices that are controlled by software, and the hardware resources that support the execution of software.

4.2.3 System Design Specification Explanation

The objective of this sub phase is to develop a system design that complies with the technical safety requirements. The technical safety requirements given in Technical safety requirements specification shall be allocated to system design elements.

4.3 4-7:WP1 Technical Safety Requirements Specification

This section is a work product resulting from Part 4, Clause 6, requirements 6.4.1 to 6.4.5 of the ISO-26262 standard. The aim is to derive the technical safety requirements from the functional safety requirements.

4.3.1 Technical Safety Requirements

The technical safety requirements listed in Table 4.1 are derived from the functional safety requirements listed in Table 3.10. The technical safety requirements are allocated to system design elements, see Figure 4.1. This allocation has its own work product called Technical Safety Concept. However that work product is the result of the same ISO-26262 requirements as the work product System Design Specification, see 4.5, except for the allocation of technical safety requirements to system design elements. For this reason, we chose to incorporate the Technical Safety Concept in Technical Safety Requirements Specification and System Design Specification.

The name of the allocation element is followed by (SW) or (HW), which denotes whether the requirement is allocated to software or hardware.

As stated earlier, only technical safety requirements that applies to CAN gateway will be presented. There exists technical safety requirements for the rest of the system as well, see Appendix C. However, those requirements are not complete and is only a first draft.

Table 4.1: Technical safety requirements derived from Functional safety requirements

Identifier	**Technical Safety Requirement**	**From FSR**	**Fault tolerant time**	**Allocated to element**	**ASIL**
TSR1.	Var:RAMTestResults must be instantiated to NotSet.	FSR56	100 ms	Gateway (SW)	C
TSR2.	Var:ROMTestResults must be instantiated to NotSet.	FSR56	100 ms	Gateway (SW)	C
TSR3.	Var:PostResultsCG must be instantiated to NotSet.	FSR56	100 ms	Gateway (SW)	C
TSR4.	Var:PostResultsA must be instantiated to NotSet.	FSR56	100 ms	Gateway (SW)	C
TSR5.	Var:PostResultsB must be instantiated to NotSet.	FSR56	100 ms	Gateway (SW)	C
TSR6.	Var:CANTestResults must be instantiated to NotSet.	FSR56	100 ms	Gateway (SW)	C
TSR7.	CAN chip External must be configured to put each incoming message from External CAN in Register InExternal.	FSR56, FSR67, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Micro-controller (HW)	C
TSR8.	When a new message has been put in Register InExternal the CAN chip must call the interrupt IRQ1.	FSR56, FSR67, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Micro-controller (HW)	C

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR9.	Upon the interrupt IRQ1, the message in Register InExternal must be put in ExternalInQueue.	FSR56, FSR67, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Micro-controller (HW)	C
TSR10.	CAN chip Internal must be configured to put each incoming message from Internal CAN in Register InInternal.	FSR4, FSR67, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Micro-controller (HW)	C
TSR11.	When a new message has been put in register InInternal the CAN chip must call the interrupt IRQ2.	FSR4, FSR67, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Micro-controller (HW)	C
TSR12.	Upon the interrupt IRQ2, the message in Register InInternal must be put in InternalInQueue.	FSR4, FSR67, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Micro-controller (HW)	C
TSR13.	There must be a periodic interrupt, IRQ3, that is triggered with a 1000 Hz frequency.	FSR54, FSR56, FSR60, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Micro-controller (HW)	C

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR14.	Upon the interrupt IRQ3, if ExternalOutQueue is not empty, the first message in the queue must be put in Register OutExternal.	FSR60	100 ms	Gateway (SW)	A
TSR15.	When a message has been sent, the CAN chip must call the interupt IRQ4.	FSR60	100 ms	Micro-controller (HW)	A
TSR16.	Upon the interrupt IRQ4, if ExternalOutQueue is not empty, the first message in the queue must be put in Register OutExternal.	FSR60	100 ms	Gateway (SW)	A
TSR17.	Upon the interrupt IRQ3, if InternalOutQueue is not empty, the first message in the queue must be put in Register OutInternal.	FSR54, FSR56, FSR60, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	C
TSR18.	When a message has been sent, the CAN chip must call the interrupt IRQ6.	FSR54, FSR56, FSR60, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Micro-controller (HW)	C

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR19.	Upon the interrupt IRQ6, if ExternalOutQueue is not empty, the first message in the queue must be put in Register OutInternal.	FSR54, FSR56, FSR60, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	C
TSR20.	Not during POST or Fail safe: Gateway must check for messages in InternalInQueue and ExternalInQueue every 10 ms.	FSR4, FSR67, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	C
TSR21.	If the InternalInQueue and ExternalInQueue are not empty when checking, every message in the queues must be processed within 5 ms.	FSR4, FSR67, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	C
TSR22.	When every message has been processed, Var:Checkpoint must be set to 0x55.	FSR54, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	C
TSR23.	While a message is processed after start approval has been sent: If the message read from ExternalInQueue has the same ID as CAN:SensorData then the message must be put in the InternalOutQueue.	FSR68	100 ms	Gateway (SW)	A

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR24.	While a message is processed: If the message read from ExternalInQueue has the same ID as CAN:HightOfCenterOfMass then the message must be put in the InternalOutQueue.	FSR69	100 ms	Gateway (SW)	A
TSR25.	While a message is processed after start approval has been sent: If the message read from InternalInQueue is CAN:FailureWarningA.FAIL the CAN gateway must transition into fail safe within 100 ms.	FSR1, FSR4, FSR54, FSR55, FSR56, FSR60, FSR66, FSR68, FSR69, FSR70, FSR71, FSR72	N/A	Gateway (SW)	C
TSR26.	While a message is processed after start approval has been sent: If the message read from InternalInQueue is CAN:FailureWarningB.FAIL the CAN gateway must transition into fail safe within 100 ms.	FSR1, FSR4, FSR54, FSR55, FSR56, FSR60, FSR66, FSR68, FSR69, FSR70, FSR71, FSR72	N/A	Gateway (SW)	C

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR27.	While a message is processed after start approval has been sent: If the message read from ExternalInQueue has the same ID as CAN:Velocity then the message must be put in the InternalOutQueue.	FSR70	100 ms	Gateway (SW)	A
TSR28.	While a message is processed after start approval has been sent: If the message read from ExternalInQueue has the same ID as CAN:Mass then the message must be put in the InternalOutQueue.	FSR71	100 ms	Gateway (SW)	A
TSR29.	While a message is processed after start approval has been sent: If the message read from ExternalInQueue has the same ID as the FrictionData message then the message must be put in the InternalOutQueue.	FSR72	100 ms	Gateway (SW)	A
TSR30.	When NOT in fail safe: the message CAN:FailureWarning.OK must be put in ExternalOutQueue every 10 ms.	FSR54	100 ms	Gateway (SW)	C
TSR31.	When NOT in fail safe: the message CAN:FailureWarningCG.OK must be put in InternalOutQueue every 10 ms.	FSR54	100 ms	Gateway (SW)	C

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR32.	While a message is processed before start approval has been sent: If the message read from InternalInQueue is CAN:POSTResultsA.SUCCESS then Var:PostResultsA = SUCCESS.	FSR56	100 ms	Gateway (SW)	C
TSR33.	While a message is processed before start approval has been sent: If the message read from InternalInQueue is CAN:POSTResultsB.SUCCESS then Var:PostResultsB = SUCCESS.	FSR56	100 ms	Gateway (SW)	C
TSR34.	While a message is processed before start approval has been sent: If the message read from InternalInQueue is CAN:POSTResultsA.FAIL then Var:PostResultsA = FAIL.	FSR56	100 ms	Gateway (SW)	C
TSR35.	While a message is processed before start approval has been sent: If the message read from InternalInQueue is CAN:POSTResultsB.FAIL then Var:PostResultsB = FAIL.	FSR56	100 ms	Gateway (SW)	C
TSR36.	On power up: The CAN memory shall be filled with '1's and the data shall then be read back and compared against expected data. Then the CAN memory shall be filled with '0's and the data shall then be read back and compared against expected results.	FSR1, FSR4, FSR54, FSR55, FSR56, FSR60, FSR66, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	C

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR37.	If the CAN test is successful then Var:CANTestResults := SUCCESS.	FSR1, FSR4, FSR54, FSR55, FSR56, FSR60, FSR66, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	C
TSR38.	If the CAN test fails then Var:CANTestResults := FAIL.	FSR1, FSR4, FSR54, FSR55, FSR56, FSR60, FSR66, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	C
TSR39.	On power up: External RAM shall be filled with '1's and the data shall then be read back and compared against expected data. Then the RAM shall be filled with '0's and the data shall then be read back and compared against expected results.	FSR1, FSR4, FSR54, FSR55, FSR56, FSR60, FSR66, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	C

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR40.	If the External RAM test is successful then Var:RAMTestResults := SUCCESS.	FSR1, FSR4, FSR54, FSR55, FSR56, FSR60, FSR66, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	C
TSR41.	If the External RAM test fails then Var:RAMTestResults := FAIL.	FSR1, FSR4, FSR54, FSR55, FSR56, FSR60, FSR66, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	C
TSR42.	On power up: The CRC32 in the program memory must be verified.	FSR1, FSR4, FSR54, FSR55, FSR56, FSR60, FSR66, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	C

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR43.	If the CRC32 test is successful then Var: PROGMTestResults := SUCCESS.	FSR1, FSR4, FSR54, FSR55, FSR56, FSR60, FSR66, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	C
TSR44.	If the CRC32 test fails then Var:PROGMTestResults := FAIL.	FSR1, FSR4, FSR54, FSR55, FSR56, FSR60, FSR66, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	C
TSR45.	If Var:RAMTestResults == SUCCESS and Var:PROGMTestResults == SUCCESS and Var:CANTestResults == SUCCESS then Var:PostResultsCG := SUCCESS.	FSR1, FSR4, FSR54, FSR55, FSR56, FSR60, FSR66, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	C

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR46.	If Var:RAMTestResults == FAIL then Var:PostResultsCG := FAIL.	FSR1, FSR4, FSR54, FSR55, FSR56, FSR60, FSR66, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	C
TSR47.	If Var:CANTestResults == FAIL then Var:PostResultsCG := FAIL.	FSR1, FSR4, FSR54, FSR55, FSR56, FSR60, FSR66, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	C
TSR48.	If Var:PROGMTestResults == FAIL then Var:PostResultsCG := FAIL.	FSR1, FSR4, FSR54, FSR55, FSR56, FSR60, FSR66, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	C

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR49.	If Watchdog Timer Reset Indication Flag is set at startup, CAN gateway must transition into fail safe within 100 ms.	FSR54	N/A	Gateway (SW)	C
TSR50.	If Var:PostResultsCG == SUCCESS and Var:PostResultsA == SUCCESS and Var:PostResultsB == SUCCESS then CAN:StartApproval.GO message must be put in the InternalOutQueue.	FSR 56	100 ms	Gateway (SW)	C
TSR51.	If Var:PostResultsCG == FAIL then CAN gateway must transition into Fail Safe within 100 ms.	FSR67	N/A	Gateway (SW)	C
TSR52.	If Var:PostResultsA == FAIL then CAN gateway must transition into Fail Safe within 100 ms.	FSR67	N/A	Gateway (SW)	C
TSR53.	If Var:PostResultsB == FAIL then CAN gateway must transition into Fail Safe within 100 ms.	FSR67	N/A	Gateway (SW)	C
TSR54.	The CAN gateway must have 2 CAN chips, CAN chip Internal and CAN chip External.	FSR4, FSR56, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Micro-controller (HW)	C

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR55.	CAN chip External must be connected through a tristate-buffer to External CAN.	FSR4, FSR54	100 ms	Tristate buffer External (HW)	C
TSR56.	The External RAM memory must be an ECC-RAM	FSR4, FSR54, FSR67, FSR60, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	External RAM (HW)	C
TSR57.	CAN chip Internal must be connected through a tristate-buffer to Internal CAN.	FSR4, FSR54	100 ms	Tristate buffer Internal (HW)	C
TSR58.	The tristate-buffer output must assume a high impedance state when the signal HW_WD_FAIL goes low within 100 ms.	FSR4, FSR54	100 ms	Tristate buffer External (HW)	C
TSR59.	The tristate-buffer output must forward in signal when the signal HW_WD_FAIL is high.	FSR4, FSR54	100 ms	Tristate buffer Internal (HW)	C
TSR60.	CAN chip External must be configured to run CAN 2.0 B.	FSR1, FSR4, FSR54, FSR55, FSR56, FSR60, FSR66, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Micro-controller (HW)	C

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR61.	CAN chip Internal must be configured to run CAN 2.0 B.	FSR1, FSR4, FSR54, FSR55, FSR56, FSR60, FSR66, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Micro-controller (HW)	C
TSR62.	Each periodic CAN message must have a timeout variable associated with them with the format Var:TimeoutID.	FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	A
TSR63.	Each Var:TimeoutID variable must be instantiated to 100.	FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	A
TSR64.	Every time a periodic CAN message is received, its associated Var:TimeoutID must be set to 100.	FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	A
TSR65.	Each Var:TimeoutID variable must be decreased by 1 every 10 ms.	FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	A

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR66.	If a Var:TimeoutID variable reach 0, CAN gateway must transition into fail safe within 100 ms.	FSR68, FSR69, FSR70, FSR71, FSR72	N/A	Gateway (SW)	A
TSR67.	The microcontroller must have a software watchdog.	FSR69, FSR70, FSR71, FSR72	100 ms	Micro-controller (HW)	A
TSR68.	The software watchdog timer must be set to 15 ms.	FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	A
TSR69.	The watchdog timer must be serviced at an interval shorter than 15 ms.	FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	A
TSR70.	If the watchdog timer overflows the CAN gateway must be reset and a flag indicating a software watchdog reset must be set.	FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	A
TSR71.	During self test, if Var:Checkpoint == 0x55 then Var:Checkpoint := 0xAA.	FSR69, FSR70, FSR71, FSR72	100 ms	Gateway (SW)	A
TSR72.	During self test, if NOT(Var:Checkpoint == 0x55) then CAN gateway must transition into fail safe within 100 ms.	FSR69, FSR70, FSR71, FSR72	N/A	Gateway (SW)	A
TSR73.	CAN gateway must have a hardware watchdog.	FSR4, FSR54	100 ms	HW Watch-dog (HW)	C

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR74.	The hardware watchdog must be fed with a PWM signal from the CPU with the frequency 50 Hz.	FSR4, FSR54	100 ms	Micro-controller (HW)	C
TSR75.	If the PWM signal from the CPU drops below or equal to 25 Hz, the digital signal HW_WD_FAIL must go low within 100 ms.	FSR4, FSR54	N/A	HW Watch-dog (HW)	C
TSR76.	If the PWM signal from the CPU goes above or equal to 75 Hz, the signal HW_WD_FAIL must go low within 100 ms.	FSR4, FSR54	N/A	HW Watch-dog (HW)	C
TSR77.	If the PWM signal from the CPU has a frequency above 25 Hz and below 75 Hz, the signal HW_WD_Fail must be high.	FSR4, FSR54	100 ms	HW Watch-dog (HW)	C
TSR78.	When in fail safe: the message CAN:FailureWarning.FAIL must be put in ExternalOutQueue every 10 ms.	FSR60	N/A	Gateway (SW)	A
TSR79.	When in fail safe: CAN:FailureWarningCG.FAIL must be put in InternalOutQueue every 10 ms.	FSR1, FSR4, FSR54, FSR55, FSR56, FSR60, FSR66, FSR68, FSR69, FSR70, FSR71, FSR72	N/A	Gateway (SW)	C

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR80.	The microcontroller must function correctly if it is fed with a supply voltage of 5 +-0.25 V.	FSR1, FSR4, FSR54, FSR56, FSR60, FSR68, FSR69, FSR70, FSR71, FSR72	100 ms	Power unit (HW)	C
TSR81.	The microcontroller must power down safely within 100 ms if supply voltage drops below 4.95 V.	FSR1, FSR4, FSR54, FSR56, FSR60, FSR68, FSR69, FSR70, FSR71, FSR72	N/A	Power unit (HW)	C
TSR82.	The microcontroller must power down safely within 100 ms if supply voltage rise above 5.05 V.	FSR1, FSR4, FSR54, FSR56, FSR60, FSR68, FSR69, FSR70, FSR71, FSR72	N/A	Power unit (HW)	C

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL

4.4 Reflections and Deviations from ISO-26262 in Technical Safety Requirements Specification

This section presents reflections on and deviations from ISO-26262 in Technical Safety Requirements Specification.

4.4.1 Fault Tolerant Time Interval and Safe State

In [6, Part3, 8.4.2.3], ISO-26262 states that safe states and fault tolerant time interval shall be specified for all functional safety requirements, if applicable. In [6, Part 4, 6.4.2] however, safe states and fault tolerant time interval shall only be specified for technical safety requirements regarding safety mechanisms. Our guess is that when ISO-26262 says "if applicable" in Part 3, they mean the same thing as is written in Part 4, but why not state this clearly? In Part 3 there are room for interpretation, while in Part 4 there is not.

4.4.2 Level of Detail of TSRs and System Design

Once again the question of at what level of detail the requirements should be written at arises. A major problem is that ISO-26262 does not clearly specify at what level of detail each "layer" of requirements should be formulated at. As far as we can see, ISO-26262 only states that technical safety requirements shall detail the functional safety requirements and so forth.

As can be seen in the technical safety requirements, see Table 4.1, our requirements are once again very detailed. They specify things such as what variables should be used, what values these variables should be assigned and what registers certain data should be put in. The system design, see Section 4.5, is then in many ways all these technical safety requirements written in plain text and shown in figures, but with some additional information. The system design answers questions such as what microcontroller should be used and how much, if any, additional RAM should be used. We believe a system design should be written at such level of detail that if two independent engineers would be given the design, there would be no significant difference in their implementations. This lead us to feel confident that our system design is at a desirable level of detail, but we cannot be sure.

What absolutely can be argued however is the level of detail at which the technical safety requirements are written at. As a result of functional safety re-

quirements that probably are too detailed, the technical safety requirements may also be too detailed.

4.4.3 Number of Requirements

As a result of the very detailed technical safety requirements, discussed in Section 4.4.2, there are a big number of technical safety requirements. There is a total of 80 technical safety requirements, and that is only for CAN gateway which in comparison to most other elements in the item, e.g. Decision unit, is a non-complex element. If technical safety requirements were to be specified for the whole item, we would be looking at more than 1000 requirements. This raise the question weather all the requirements that we consider safety related, really are so. As we are progressing through ISO-26262, more and more only concern safety mechanisms and thus, our initial believe that ISO-26262 is about strict requirements decomposition, may not be correct. However, the number of requirements will always be quite large for a fairly complex system.

4.4.4 Verification of Technical Safety Requirements

As mentioned in Section 4.4.3, the resulting number of requirements when applying ISO-26262 on a system is quite large. Not only are there a large number of requirements but most of them are also referring to a large number of requirements in the level above. This implies that the work needed to verify that the requirements are complete and consistent with the functional safety requirements is not only highly time consuming, but also very hard to do correctly. Using an automated tool for the verification process is almost necessary to guarantee that the verification is correct.

4.5 4-7:WP2 System Design Specification

This section is a work product resulting from Part 4, Clause 7, requirements 7.4.1 to 7.4.4 of the ISO-26262 standard. The aim is to produce a design for the system that implements the technical safety requirements.

4.5.1 Allocation Elements

The CAN gateway consists of seven modules, one software module and six hardware modules. These modules are shown in figure 4.1.

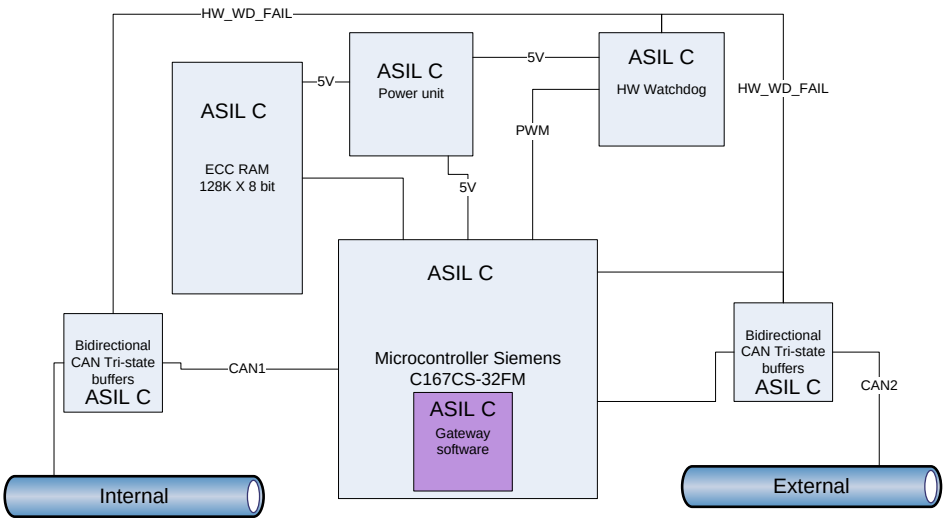

Figure 4.1: Block diagram over the modules used by the CAN gateway.

4.5.2 Power Unit

The voltage from the battery varies during operation. The power unit converts input voltages between 6 and 48 voltages to an output voltage of 5 volt which is fed to the hardware in CAN gateway.

4.5.3 Microcontroller Siemens C167CS-32FM

The Siemens C167CS-32FM is the central part of the CAN gateway.

Important features supported by the microcontroller are:

- Two on-chip CAN interfaces.
- A software watchdog timer.
- Timer units.
- External addressing capability.

- A 256 KByte on-chip Flash memory for program code.
- A 4 KByte 16-bit wide on-chip DataFlash/EEPROM.
- Possibility of automatic shutdown of the CPU in case of power supply deviating more than 0.05 V from 5 V.

4.5.4 External RAM

When buffering several CAN messages it is estimated that more than the C167CS 4kb internal RAM will be needed. Therefore an external RAM has been added in the design. The C167CS has support for addressing external storage and 128kb extra RAM will be enough. The RAM must be of ECC type.

4.5.5 HW Watchdog

The hardware watchdog monitors a PWM signal given by one of the digital outputs on the microcontroller. If the microcontroller is working correctly then the frequency of the PWM signal should be 50 Hz, if so the HW_WD_FAIL signal is high. If the frequency diverts 25 Hz or more from the target value then the watchdog pulls the HW_WD_FAIL signal low.

4.5.6 Bidirectional CAN Tristate Buffers

When the HW_WD_FAIL signal is high then the tristate buffers are transparent and outputs the input logic level. When HW_WD_FAIL is low the output from the tristate buffers assumes a high impedance state effectively removing the CAN gateway from the rest of the CAN network.

4.5.7 Gateway Software

The software can be divided into seven different activities. Figure 4.2 illustrates the flow of these activities. It starts with the system performing a power on self test (POST). When the POST has finished the system waits for the POST results from subsystem A and B. After receiving these results the system periodically performs regular execution and self testing. If the system at any time receives knowledge of a test failing then the system will transition into fail safe state. If the system is deactivated it transitions into an off state where it waits for activation.

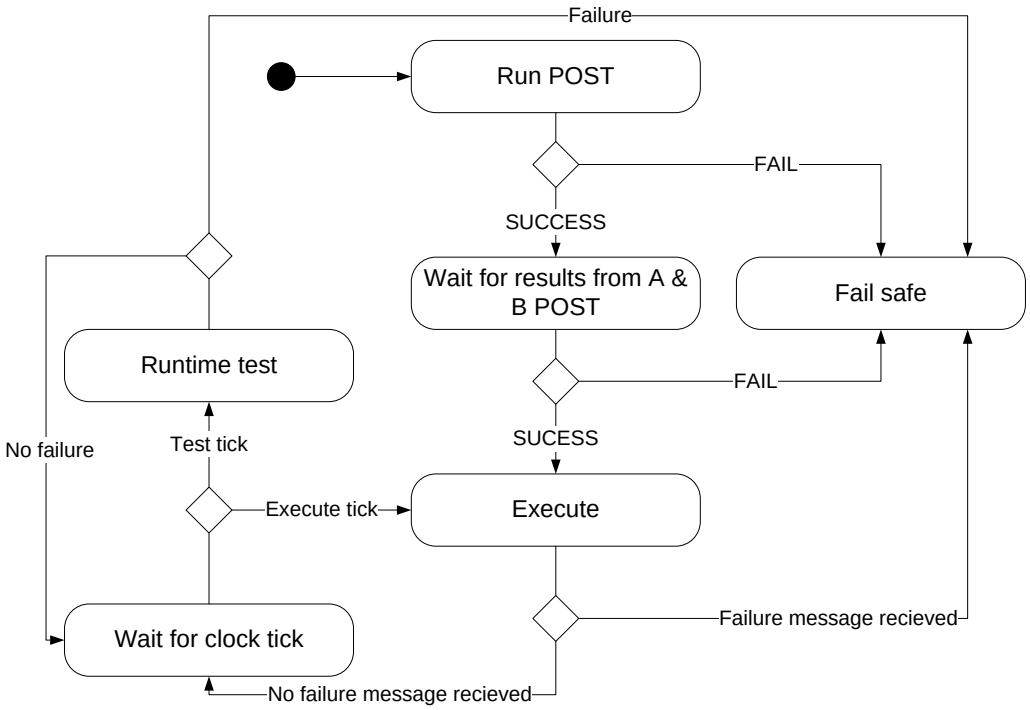

Figure 4.2: Overview of the CAN gateway software activities.

POST Activity

The first step of this activity is to fill External RAM with '1's. When this is completed the contents of External RAM is read back. If External RAM then contains anything but '1's then there has been a failure. The same procedure is the repeated but with '0's instead. If External RAM test is successful then Var:RAMTestResults := SUCCESS else Var:RAMTestResults := FAIL.

Next the CAN memory is filled with '1's. When this is completed the contents of the CAN memory is read back. If the CAN memory then contains anything but '1's then there has been a failure. The same procedure is the repeated but with '0's instead. If the CAN test is successful then Var:CANTestResults := SUCCESS else Var:CANTestResults := FAIL.

The next step is to calculate a CRC32 for the contents of the program memory and compare this to a stored checksum. If there is a mismatch a failure has

occurred. If the CRC32 test is successful then Var:PROGMTestResults := SUCCESS else Var:PROGMTestResults := FAIL.

The last step is to test the hardware watchdog. The hardware watchdog is fed with a PWM signal with a frequency of 25 Hz. HW_WD_FAIL is read and if it is logic low the test has passed. The same procedure is repeated with a PWM frequency of 75 Hz. Again HW_WD_FAIL should be logic low. Finally the hardware watchdog is fed with a PWM frequency of 50 Hz. This time HW_WD_FAIL should be logic high. If these three tests pass then Var:HwWdTestResults := SUCCESS.

If Var:RAMTestResults == SUCCESS and Var:PROGMTestResults == SUCCESS and Var:CANTestResults == SUCCESS and Var:HwWdTestResults == SUCCESS then Var:PostResultsCG := SUCCESS else Var:PostResultsCG := FAIL.

Wait For Results From A & B Activity

The system constantly polls the InternalInQueue looking for the CAN:POSTResultsA and CAN:POSTResultsB messages. Depending on the data of these message the system will respond with different actions. Table 4.2 shows the different messages in relation to the action taken.

Table 4.2: POST results messages and corresponding action

From queue	**Message ID**	**Message data**	**Action**
InternalInQueue	CAN:POSTResultsA	SUCCESS	Var:PostResultsA:= SUCCESS
InternalInQueue	CAN:POSTResultsA	FAIL	Transition into fail safe
InternalInQueue	CAN:POSTResultsB	SUCCESS	Var:PostResultsB := SUCCESS
InternalInQueue	CAN:POSTResultsB	FAIL	Transition into fail safe

If either Var:PostResultsA == FAIL or Var:PostResultsB == FAIL or Var:PostResultsCG := FAIL then the system shall transition into fail safe, else the system will put the message CAN:StartApproval.GO in the InternalOutQueue and transition to the Execute activity.

Execute Activity

The execute activity processes every message in InternalInQueue and ExternalInQueue and takes different action depending on the message. Table 4.3 describes the actions taken in relation to the different messages.

Table 4.3: Incoming messages and corresponding actions in the execution activity.

From queue	**Message ID**	**Message data**	**Action**
ExternalInQueue	CAN:SensorData	Any	Put message in InternalOutQueue
ExternalInQueue	CAN:HeightOfCenterOfMass	Any	Put message in InternalOutQueue
ExternalInQueue	CAN:Velocity	Any	Put message in InternalOutQueue
ExternalInQueue	CAN:Mass	Any	Put message in InternalOutQueue
ExternalInQueue	CAN:FrictionData	Any	Put message in InternalOutQueue
ExternalInQueue	CAN:ON/OFF	Any	Put message in InternalOutQueue
ExternalInQueue	Other	Any	Discard
InternalInQueue	CAN:FailureWarningA	OK	Discard
InternalInQueue	CAN:FailureWarningA	FAIL	Transition into fail safe
InternalInQueue	CAN:FailureWarningB	OK	Discard
InternalInQueue	CAN:FailureWarningB	FAIL	Transition into fail safe
InternalInQueue	CAN:CollisionWarning	Any	Put message in ExternalOutQueue
InternalInQueue	CAN:PreCrashWarning	Any	Put message in ExternalOutQueue
InternalInQueue	Other	Any	Discard

The messages in table 4.4 should be sent every execution activity, in other words every 10 ms.

Table 4.4: CAN messages to be sent every execution activity

Queue	Message ID	Message data
ExternalOutQueue	CAN:FailureWarning	OK
InternalOutQueue	CAN:FailureWarningCG	OK

When all messages are sent the execution activity sets the variable Var:Checkpoint to 0x55 and toggles the digital output producing the PWM signal to the hardware watchdog.

When done the system transitions to Wait for clock tick activity.

Wait For Clock Tick Activity

The wait for clock tick activity keeps track of internal timers and starts the Execution activity and the Runtime test activity periodically every 10 ms.

Runtime Test Activity

Every incoming CAN message has a timeout variable associated with them called Var:TimeoutID where ID is the identifier of the CAN message. These variables are instantiated to 100 and every time a CAN message is received the associated variable is reset to 100. During the Runtime test activity the associated variable is decreased by 1. If an associated variable reaches 0 then a timeout has occurred and CAN gateway will transition into fail safe.

To allow recovery from software or hardware failure, the C167CS provides a Watchdog Timer. This timer is set to overflow over a time period of somewhere in between 12 and 15 ms. The software should be written to reset this timer in even intervals before it overflows. If the software fails to reset the timer before an overflow occurs, this is an indication of a possible hardware failure or a software bug and an internal reset sequence will be initiated. Upon the reset a flag called Watchdog Timer Reset Indication Flag is checked, if it is active this means that a watchdog reset has occurred and CAN gateway will transition into fail safe.

The runtime test activity will reset the watchdog timer each execution.

In the execution activity a variable called Var:Checkpoint is set to 0x55. The runtime test activity evaluates this variable and if it is not set to 0x55 then CAN gateway will transition into fail safe. After evaluation the runtime test activity writes 0xAA to Var:Checkpoint.

Fail Safe Activity

In the fail safe activity the CAN messages in table 4.5 should be sent.

Table 4.5: CAN messages to be sent in the Fail safe activity

Queue	**Message ID**	**Message data**
ExternalOutQueue	CAN:FailureWarning	FAIL
InternalOutQueue	CAN:FailureWarningCG	FAIL

4.5.8 Target Values for Probability of Random Hardware Failure

In Part 5 an analysis of the probability that a random hardware failure will lead to the violation of a safety goal will be evaluated. The target values for this evaluation is given in table 4.6. These values have been derived from a table in ISO 26262 [6, Part5, Table 6]. This part only applies to safety goals with ASIL C or ASIL D.

Table 4.6: Target values for probability of safety goal violation due to random hardware failure

Safety goal	ASIL	**Target value**
SG2	C	$< 10^{-7}h^{-1}$
SG4	C	$< 10^{-7}h^{-1}$

4.6 Reflections and Deviations from ISO-26262 in System Design Specification

This section presents reflections on and deviations from ISO-26262 in System Design Specification.

Measures For Avoiding Systematic Failures

ISO 26262 dictates that you should identify causes of systematic failures [6, Part4,7.4.3.1]. A systematic failure is a consequence of a systematic fault. That is a fault produced by human error, like a bug or misuse of equipment. Specifying the causes of human error was not relevant to our goal so this part was ignored.

Target Values For Single Point Fault Metric And Latent Fault Metric

ISO 26262 requires that target values for single-point fault metric and latent fault metric shall be specified [6, Part4,7.4.4.2]. However we have chosen not to evaluate this metric in part 5 because of reasons explained in Section 6.1, hence this requirement was ignored.

Design Decisions

The reason behind choosing the microcontroller Siemens C167CS-32FM is that it is the microcontroller used in Scanias COO6, which was used for many years, and is thus a well-trusted hardware component, which is favorable. The model is rather old and does not have very high performance. However since CAN gateway is less resource demanding than COO6, we are confident that the performance of Siemens C167CS-32FM is sufficient.

The CAN interfaces within the microcontroller have the ability to filter incoming messages with hardware. Up to 16 unique CAN messages can be filtered and since CAN gateway need to gate less than 10 messages, this feature could have been used instead of filtering the CAN messages with software. We chose not to do so however, since if this system would have been implemented at Scania, CAN gateway would most certainly be a part of COO. Then there is suddenly much more than 16 CAN messages that need to be gated, and using only hardware filtering is no longer possible.

Automatic CPU Shutdown in Case of Power Supply Failure

In Section 4.5.3 it is stated that the microprocessor used in the design has a feature that can safely shut the microcontroller down in case of a power supply failure. This is only half true. The microcontroller does have a certain instruction that can be used together with a power failure signal to shut it down. However it does not exist a mechanism in the microcontroller that can detect this power failure. Although there exists microcontrollers with this feature built in, and therefore we chose to use it in our microcontroller.

4.7 4-7:WP3 Hardware Software Interface Specification

This section is a work product resulting from Part 4, Clause 7, requirement 7.4.6 of the ISO-26262 standard. This specification includes the hardware components of the CAN gateway that are controlled by software.

4.7.1 Microcontroller Siemens C167CS-32FM

Operation modes and configuration parameters

The Siemens C167Cs-32FM has the following three major operational modes:

- **Default mode**
  Every function of the microcontroller is operational.
- **Idle mode**
  In Idle mode the CPU is stopped, while the (enabled) peripherals continue their operation. Idle mode can be terminated by any reset or interrupt request.
- **Sleep mode**
  In Sleep mode both the CPU and the peripherals are stopped. The real time clock and its selected oscillator may optionally be kept running. Sleep mode can be terminated by any reset or interrupt request.
- **Power Down mode**
  In Power Down mode both the CPU and the peripherals are stopped. The real time clock and its selected oscillator may optionally be kept running. Power Down mode can only be terminated by a hardware reset.

There is also a functionality of the Siemens C167Cs-32FM that can be interpreted as a mode. Upon a reset of the device the initialization routine is executed. In this routine the configuration for the microcontroller is set up. The instruction EINIT ends the initialization routine and prevents certain instruction from being executed.

The following configurations is done in the initialization routine:

- CAN interfaces
  The Control / Status Register (CSR) accepts general control settings for the module and provides general status information.

  An X means that the bit should not be set and is only for reading.

  The CSR should be set to :XXXXXXXX00X0111X

- Watchdog timers

  The control register WDTCON for initialization and reset source detection should be set to 000000001XXXX0. This gives the timer a time span of 13.11 ms before it overflows.

- External RAM

  The microcontroller must be configured to use an external RAM instead of the internal RAM. This requires pin 99 to be connected to ground during reset. Also the microcontroller to be set to use the demultiplexed bus mode and to output the upper part of the address on port 4.

Hardware resources

The software implementing the CAN gateway is the only software that will be executed on the Siemens C167Cs-32FM microcontroller, therefore all hardware resources on the microcontroller are exclusive to the CAN gateway. For a detailed specification of the hardware resources see the user manual for the microcontroller [7].

Timing constraints

The Execution activity and the Runtime test activity both have to finish their execution in under 5 ms.

Diagnostics

Three safety mechanisms use software to track the behavior of the hardware. The first one is the watchdog timer. The timer itself is implemented in hardware but relies on software to reset it before an overflow occurs. The second one is a register that is set in one part of the program and cleared in another. If the program reaches the set part without being cleared or vice versa this is a sign of a flow control error. The last safety mechanism that uses software is the timeout detection which monitors the flow of CAN messages.

One safety mechanism is implemented entirely in hardware, this is the hardware watchdog. The hardware watchdog monitors a 50 Hz pulse with modulation signal from the microcontroller. If this signal diverts with more than 25 Hz the watchdog disconnects the CAN gateway from the rest of the network using tristate buffers.

4.8 Reflections and Deviations from ISO-26262 in Hardware Software Interface Specification

This section presents reflections on and deviations from ISO-26262 in Hardware Software Interface Specification.

Regarding Operation Modes And Configuration Parameters

Due to lack of time we have not specified all the configuration and modes available on the Siemens C167CS-32FM. For more in-depth knowledge of the necessary configurations see the Siemens C167CS-32FM user manual [7].

Chapter 5

Part 5 - Product Development at the Hardware Level

This chapter concerns the work that has been done in Part 5 of ISO 26262. The chapter begins with a section explaining objectives of Part 5 and then continues with the work products that have been produced during Part 5. After each work product, reflections on and deviations from ISO 26262 are presented. Only the element CAN gateway will be treated in this chapter.

5.1 Regarding Different Types of Faults

In this chapter, many types of faults are brought up that need explaining. This section is dedicated to explaining the different types of faults.

Single-Point Fault A single-point fault is a fault which is not covered by safety mechanisms, and directly lead to the violation of a safety goal.

Multiple-Point Fault An individual fault that in combination with other independent faults, leads to the violation of a safety goal. Dual-point faults are a subset of multiple-point faults, where an individual fault in combination with another independent fault, lead to the violation of a safety goal.

Latent Fault A latent fault is a multiple-point fault which is not detected nor perceived by the driver, i.e., the fault remains latent until another fault occurs which together with the latent fault violates a safety goal.

Residual Fault A residual fault is a portion of a fault in a hardware component which is not covered by a safety mechanism, that leads to the violation of a safety

goal. That means that in order for a fault on a hardware component to be a residual fault instead of a single-point fault, the hardware component must be protected by a safety mechanism but the safety mechanism does not cover this certain fault.

5.2 Objectives of Part 5 According to ISO 26262

This section attempts to explain the objectives and purpose of Part 5 - Product development at the hardware level.

5.2.1 Hardware Safety Requirement Specification Explanation

The first objective of Part 5 is to write the hardware safety requirements specification. The purpose of writing this work product is to refine technical safety requirements that are allocated to both hardware and software into requirements allocated exclusively to hardware or software. The requirements allocated to hardware are specified in this work product and are called hardware safety requirements.

5.2.2 Hardware Design Specification Explanation

The objective of this work product is to develop a hardware design that is consistent with both the system design from Part 4 and fulfills the hardware safety requirements. The design shall be done in two levels. The first level is called hardware architectural design and it specifies all components in the design and the interaction between them. The second level is called detailed hardware design and consists of the electrical schematics of the components and the interactions between them. Due to the fact that we are using existing components in our design we will not develop the detailed hardware design.

5.2.3 Hardware Safety Analysis Report Explanation

A safety analysis is performed on the hardware design in order to support the hardware design. The analysis can later be used for verification of the hardware design. The first step of the analysis is to identify the faults on hardware parts that might lead to a violation of a safety goal. A fault in this manner is not necessarily at a low level, a fault can be a more general failure mode of a hardware element, such as a value failure. Which means a hardware element outputs a different value than intended.

The identification of faults can be done with techniques such as FMEA [10] or FTA [13]. When the faults are identified they shall be classified as single-point faults, residual-faults, latent faults, or multi-point faults. The next step of

the analysis is to provide evidence of the effectiveness of the safety mechanisms to avoid single-point and latent faults. After this the diagnostic coverage with respect to residual faults and with respect to latent faults is evaluated, that is what proportion of the faults on a hardware part is covered by a safety mechanism.

5.2.4 Analysis of Safety Goal Violations due to Random Hardware Failures Explanation

The objective of this clause is to evaluate the probability that a safety goal is violated due to a random hardware failure. The result can then be used in a rationale that the residual risk of a safety goal violation is sufficiently low. Two alternative methods to conduct this evaluation are proposed. The method used in this thesis is Evaluation of Probabilistic Metric for Random Hardware Failures [6, Part 5, 9.4.2]. A quantitative analysis of the hardware architecture shall provide evidence that the target values given in Section 4.5.8 have been met.

5.2.5 Specification of Dedicated Measures for Hardware Explanation

In the hardware safety analysis report, the diagnostic coverage with respect to residual faults of hardware parts was evaluated. This work product specifies the dedicated measures that must be taken for those hardware parts which had a diagnostic coverage of less than 90%.

5.3 5-6:WP1 Hardware Safety Requirements Specification

This section is a work product resulting from Part 5, Clause 6, requirements 6.4.1 to 6.4.8 of the ISO-26262 standard. The aim is to derive the hardware safety requirements from the technical safety requirements, see Table 4.1, and the system design specification, see Section 4.5.

5.3.1 Hardware Safety Requirements

The technical safety requirements, concerning to CAN gateway, that are allocated to both hardware and software are further partitioned into requirements that are allocated to hardware and software exclusively. The safety requirements allocated to hardware are the hardware safety requirements, which are shown in Table 5.1.

Table 5.1: Hardware safety requirements derived from Technical safety requirements

Identifier	**Hardware Safety Requirement**	**From TSR**	**Fault tolerant time**	**Allocated to element**	**ASIL**
HSR1.	CAN chip External must be configured to put each incoming message from External CAN in Register InExternal.	TSR7	100 ms	Micro-controller	C
HSR2.	When a new message has been put in Register InExternal the CAN chip must call the interrupt IRQ1.	TSR8	100 ms	Micro-controller	C
HSR3.	Upon the interrupt IRQ1, the message in Register InExternal must be put in ExternalInQueue.	TSR9	100 ms	Micro-controller	C
HSR4.	CAN chip Internal must be configured to put each incoming message from Internal CAN in Register InInternal.	TSR10	100 ms	Micro-controller	C
HSR5.	When a new message has been put in register InInternal the CAN chip must call the interrupt IRQ2.	TSR11	100 ms	Micro-controller	C
HSR6.	Upon the interrupt IRQ2, the message in Register InInternal must be put in InternalInQueue.	TSR12	100 ms	Micro-controller	C
HSR7.	There must be a periodic interrupt, IRQ3, that is triggered with a 1000 Hz frequency.	TSR13	100 ms	Micro-controller	C

Table continued on next page

Identifier	Hardware Safety Requirement	From TSR	Fault tolerant time	Allocated to element	ASIL
HSR8.	When a message has been sent, the CAN chip must call the interupt IRQ4.	TSR15	100 ms	Micro-controller	A
HSR9.	When a message has been sent, the CAN chip must call the interrupt IRQ6.	TSR18	100 ms	Micro-controller	C
HSR10.	The CAN gateway must have 2 CAN chips, CAN chip Internal and CAN chip External.	TSR54	100 ms	Micro-controller	C
HSR11.	CAN chip External must be connected through a tristate-buffer to External CAN.	TSR55	100 ms	Tristate buffer External	C
HSR12.	External RAM memory must be an ECC-RAM	TSR56	100 ms	External RAM	C
HSR13.	CAN chip Internal must be connected through a tristate-buffer to Internal CAN.	TSR57	100 ms	Tristate buffer Internal	C
HSR14.	The tristate-buffer output must assume a high impedance state when the signal HW_WD_FAIL goes low within 100 ms.	TSR58	N/A	Tristate buffer External	C
HSR15.	The tristate-buffer output must forward in signal when the signal HW_WD_FAIL is high.	TSR59	100 ms	Tristate buffer Internal	C
HSR16.	CAN chip External must be configured to run CAN 2.0 B.	TSR60	100 ms	Micro-controller	C
HSR17.	CAN chip Internal must be configured to run CAN 2.0 B.	TSR61	100 ms	Micro-controller	C
HSR18.	The microcontroller must have a software watchdog.	TSR67	100 ms	Micro-controller	A
HSR19.	CAN gateway must have a hardware watchdog.	TSR73	100 ms	HW Watch-dog	C

Table continued on next page

Identifier	Hardware Safety Requirement	From TSR	Fault tolerant time	Allocated to element	ASIL
HSR20.	The hardware watchdog must be fed with a PWM signal from the CPU with the frequency 50 Hz.	TSR74	100 ms	Micro-controller	C
HSR21.	If the PWM signal from the CPU drops below or equal to 25 Hz, the digital signal HW_WD_FAIL must go low within 100 ms.	TSR75	N/A	HW Watch-dog	C
HSR22.	If the PWM signal from the CPU goes above or equal to 75 Hz, the signal HW_WD_FAIL must go low within 100 ms.	TSR76	N/A	HW Watch-dog	C
HSR23.	If the PWM signal from the CPU has a frequency above 25 Hz and below 75 Hz, the signal HW_WD_Fail must be high.	TSR77	100 ms	HW Watch-dog	C
HSR24.	The microcontroller must function correctly if it is fed with a supply voltage of 5 +-0.25 V.	TSR80	100 ms	Power unit	C
HSR25.	The microcontroller must power down safely within 100 ms if supply voltage drops below 4.95 V within.	TSR81	N/A	Power unit	C
HSR26.	The microcontroller must power down safely within 100 ms if supply voltage rise above 5.05 V.	TSR82	N/A	Power unit	C
HSR27.	The microcontroller must have a probability of a random hardware failure $<= 10^{-7}h^{-1}$	N/A	N/A	Micro-controller	N/A
HSR28.	The hardware watchdog must have a probability of a random hardware failure $<= 10^{-7}h^{-1}$	N/A	N/A	HW Watch-dog	N/A

Table continued on next page

Identifier	Hardware Safety Requirement	From TSR	Fault tolerant time	Allocated to element	ASIL
HSR29.	The tristate buffers must have a probability of a random hardware failure $<= 10^{-7}h^{-1}$	N/A	N/A	Tristate buffer Internal	N/A
HSR30.	The External RAM must have a probability of a random hardware failure $<= 10^{-7}h^{-1}$	N/A	N/A	External RAM	N/A
HSR31.	The tristate buffers must tolerate temperatures between -40°Cto 90°C.	N/A	N/A	Tristate buffers	N/A
HSR32.	The microcontroller must tolerate temperatures between -40°Cto 90°C.	N/A	N/A	Micro-controller	N/A
HSR33.	The hardware wachdog must tolerate temperatures between -40°Cto 90°C.	N/A	N/A	HW Watch-dog	N/A
HSR34.	The External RAM must tolerate temperatures between-40°Cto 90°C.	N/A	N/A	External RAM	N/A

5.4 Reflections and Deviations from ISO-26262 in Hardware Safety Requirements Specification

This section presents reflections on and deviations from ISO-26262 in Hardware Safety Requirements Specification.

No Refinement of Technical Safety Requirements

Since our technical safety requirements were so detailed, no further refinement of them were necessary. Hence the requirements specified in this work product are those technical safety requirements that were allocated to hardware and the extra hardware safety requirements not derived from and technical safety requirement. A reflection regarding the extra hardware safety requirements can be found in section 5.4 and a reflection regarding the level of detail of the technical safety requirements is found in 4.4.2.

Hardware Safety Requirements not Derived from Technical Safety Requirements

ISO-26262 requires formulation of hardware safety requirements to meet the target values of probability of safety goal violation due to random hardware failures[6, Part5, 6.4.2, Example 5]. This is possible since the target values were specified in section 4.5.8 as part of the system design. However, no technical safety requirements regarding this matter have been written so these requirements cannot be derived from any requirement. Without deriving from a requirement with a higher hierarchical level, no ASIL can be determined. Therefore, these hardware safety requirements are without ASIL.

5.5 5-7:WP1 Hardware Design Specification

This section is a work product resulting from Part 5, Clause 7, requirements 7.4.1 and 7.4.2 of the ISO-26262 standard. The aim is to describe the hardware architectural design and hardware detailed design of CAN gateway. Due to the reasons discussed in section 5.6, this work product will not contain a hardware detailed design. The design complies with the hardware safety requirements from section 5.3.

5.5.1 Hardware Architectural Design

The design consists of eight hardware modules and their interactions with one another. The modules and their interconnections are shown in figure 5.1. All the elements are ASIL C.

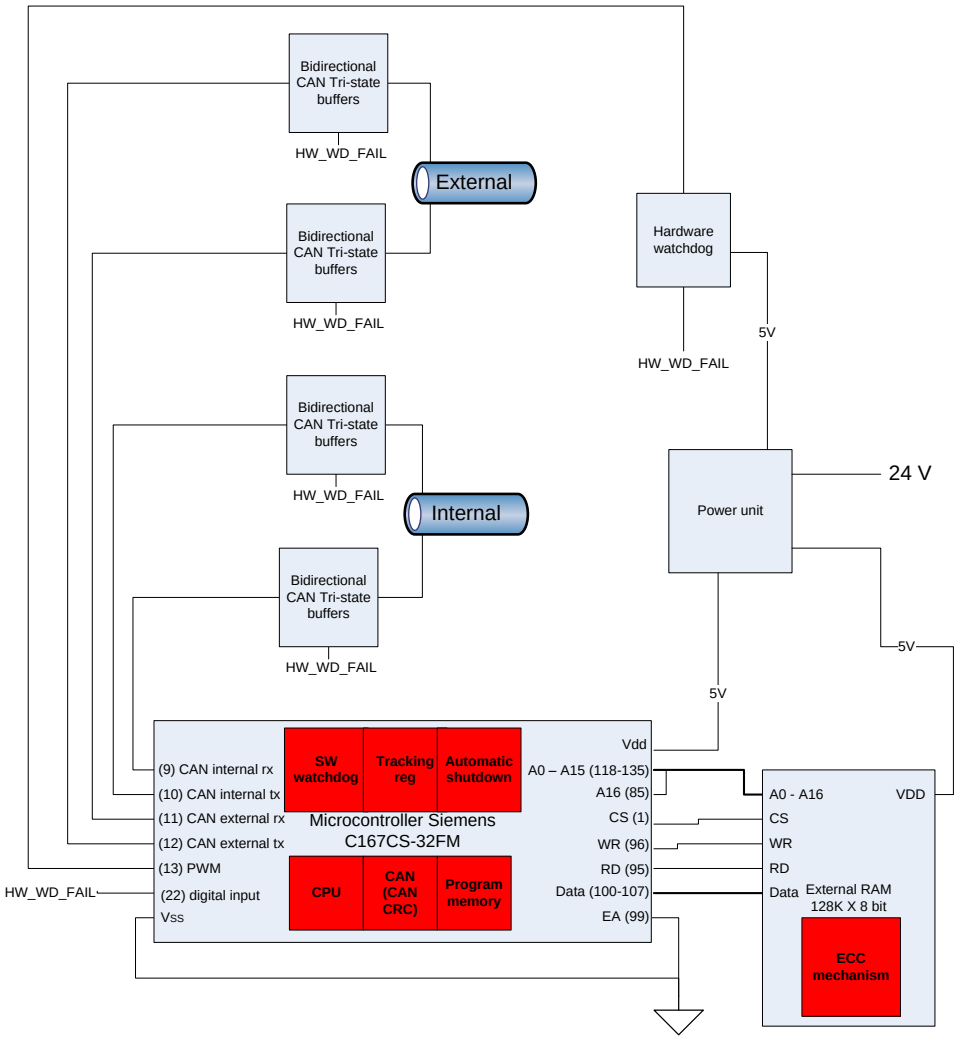

Figure 5.1: Hardware design

5.6 Reflections and Deviations from ISO-26262 in Hardware Design Specification

This section presents reflections on and deviations from ISO-26262 in Hardware Design Specification.

No Detailed Hardware Design

Due to the usage of an already existing microcontroller, we will not be able to present an electric schematic on this. The hardware watchdog and the power unit are existing Scania components used in their trucks. Thus, electric schematics exist and are available to us, but due to privacy reasons they will not be displayed in this thesis. For the other hardware components, such as the tristate buffers, we lack the necessary expertise to be able to produce the electric schematics for them.

No Refined HSI

ISO 26262 dictates that the HSI composed in section 4.7 shall be refined to allow for correct control of the hardware, by the software [6, Part 5, 6.4.10]. As our hardware design specification does not deviate far from the system design specification, we judged the existing HSI to be sufficient. If we would have performed the detailed hardware design, it is likely that we would have needed a refined HSI.

Similarities Between System Design and Hardware Architectural Design

Due to the reasons described in section 4.4.2 the level of detail of the System design specification is fairly high. As a consequence of this the hardware architectural design is very similar to the system design. The major differences are that port numbers and more accurate signal connections are shown in the hardware architectural design.

5.7 5-7:WP2 Hardware Safety Analysis Report

This section is a work product resulting from Part 5, Clause 7, requirement 7.4.3 of the ISO-26262 standard. The work product contains a safety analysis performed on the hardware design. Later bits of the analysis will be used for verification of the hardware design.

5.7.1 Fault Identification and Effects of Faults

Figure 5.2 illustrates an FTA and Figure 5.3 illustrates an FMEA. These analyses are made in order to identify the faults that leads to a possible violation of a safety goal. The FTA is later used in section 5.9, where the probability of violating a safety goal, due to a random hardware failure, is evaluated.

The FMEA deviates a bit from a basic structure of an FMEA [10, Page 8, Basic structure]. The only purpose of this FMEA is to identify the faults that can lead to a violation of a safety goal. Therefore the parts of the FMEA that addresses the detection of the failure modes and the severity of the failure mode will not be displayed since that would be of no use in this analysis.

The FTA has been made using an approach found in [9].

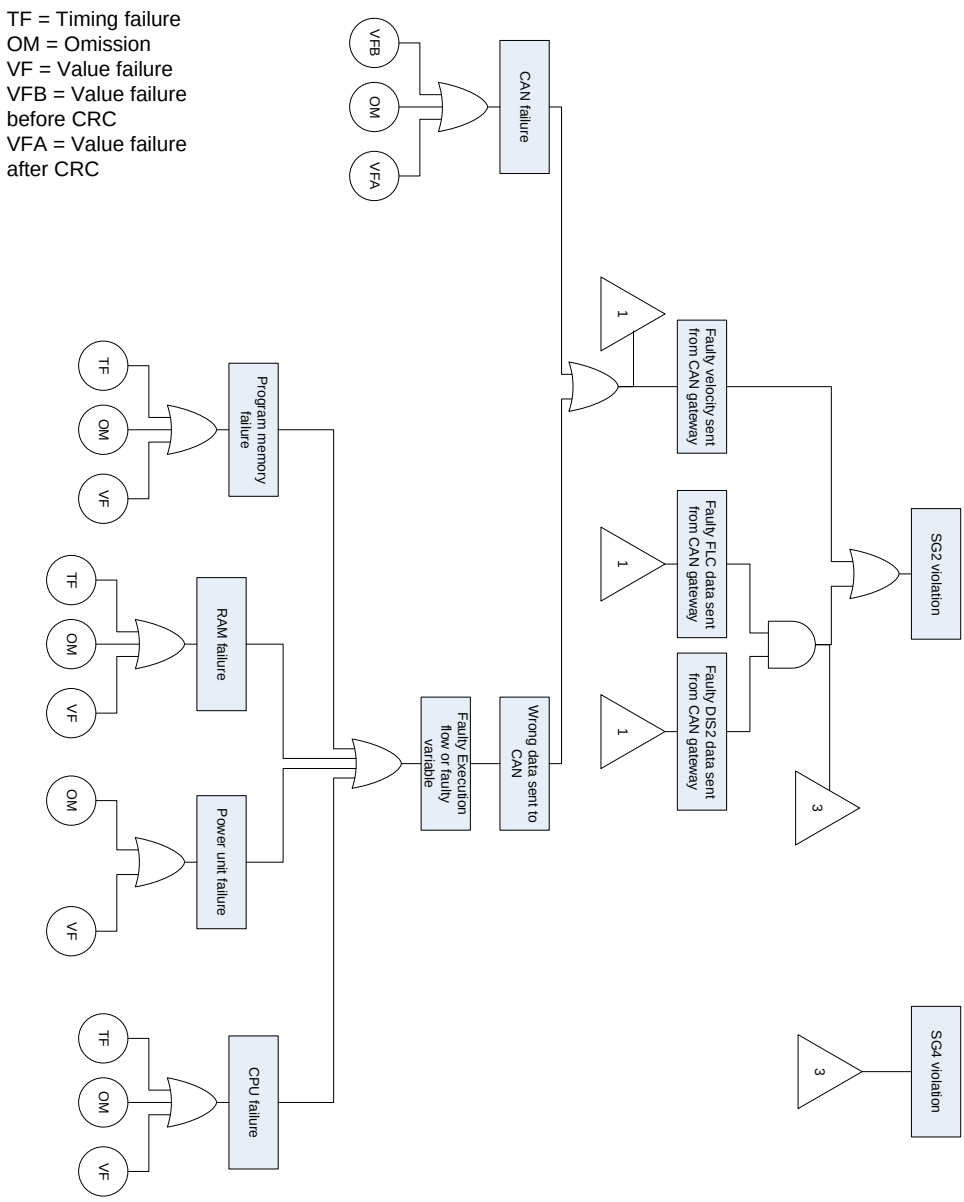

Figure 5.2: Fault tree analysis

Component	Failure Mode	Failure mode explanation	Effects	Causes	Occurence Rating
External RAM	Value failure	The data retrieved from RAM is corrupted.	Faulty execution flow, Wrong value sent to other subsystems.	Bit flip, stuck at faults.	$10^{-7}h^{-1}$
External RAM	Timing failure	Data from RAM not ready when read.	Faulty execution flow, Wrong value sent to other subsystems.	Under voltage.	$10^{-7}h^{-1}$
External RAM	Omission	No data present at output pins.	Faulty execution flow, Wrong value sent to other subsystems.	Fault tristate buffers, power failure.	$10^{-7}h^{-1}$
CAN chip	Value failure after CRC	The data sent to other subsystems is corrupted.	Wrong value sent to other subsystems.	Bit flip, stuck at faults, data retrieved is corrupted.	$10^{-7}h^{-1}$
CAN chip	Value failure before CRC	The data read by the CAN chip is corrupted.	Wrong value sent to other subsystems.	Over/Under voltage, EMC, oversampling.	$10^{-7}h^{-1}$
CAN chip	Omission	CAN module unavailable	Broken communication to other subsystems.	Broken BUS, CAN chip faulty.	$10^{-7}h^{-1}$
Program memory	Value failure	The data retrieved from memory is corrupted.	Faulty execution flow, Wrong value sent to other subsystems.	Bit flip, stuck at faults.	$10^{-7}h^{-1}$
Program memory	Timing failure	Data from memory not ready when read.	Faulty execution flow, Wrong value sent to other subsystems.	Under voltage.	$10^{-7}h^{-1}$
Program memory	Omission	No data present at output pins.	Faulty execution flow, Wrong value sent to other subsystems.	Fault tristate buffers, power failure.	$10^{-7}h^{-1}$
CPU	Value failure	Data is corrupted while processed by the CPU.	Faulty execution flow, Wrong value sent to other subsystems.	Bit flip, stuck at faults.	$10^{-7}h^{-1}$
CPU	Timing failure	CPU operates at the wrong frequency.	Message sent to external system no longer useful.	Over/Under voltage.	$10^{-7}h^{-1}$
CPU	Omission	CPU does not operate at all.	Broken communication to other subsystems.	Power failure, extensive hardware failures.	$10^{-7}h^{-1}$
Power supply	Value failure	Power supply produce an output voltage that deviates from 5 +- 1% V.	All HW components in CAN gateway will produce non-determenistic outputs.	Component failure.	$10^{-7}h^{-1}$
Power supply	Omission	Power supply does not produce an output voltage at all.	All HW components in CAN gateway will cease to function.	Component failure.	$10^{-7}h^{-1}$

Figure 5.3: Failure Mode and Effect Analysis

5.7.2 Fault Classification

Table 5.2 classifies the faults identified in section 5.7 as single point faults, residual faults, latent faults, dual-point faults or multi point faults.

Table 5.2: Classification of hardware faults in CAN gateway

Fault(s)	**Classification**	**Safety mechanism(s)**
RAM value failure	Dual-point	ECC mechanism in RAM
RAM timing failure	Dual-point	HW watchdog, SW watchdog, Tracking register
RAM omission	Dual-point	POST, HW watchdog, SW watchdog, Tracking register
CAN value failure after CRC	Residual	POST
CAN value failure before CRC	Dual-point	CAN CRC
CAN omission	Dual-point	CAN timeout
Program memory value failure	Residual	HW watchdog, SW watchdog, Tracking register
Program memory timing failure	Multi-point	HW watchdog, SW watchdog, Tracking register
Program memory omission	Dual-point	HW watchdog, SW watchdog, Tracking register
CPU value failure	Residual	HW watchdog, SW watchdog, Tracking register
CPU timing failure	Residual	HW watchdog, SW watchdog, Tracking register
CPU omission	Dual-point	HW watchdog, SW watchdog
Power unit value failure	Dual-point	Automatic shutdown mechanism
Power unit omission	Dual-point	Automatic shutdown mechanism
RAM ECC mechanism failure	Latent	None
HW watchdog failure	Dual-Point	POST
SW watchdog failure	Latent	None
CAN CRC failure	Latent	None
Automatic shutdown mechanism failure	Latent	None
Tracking register failure	Latent	None

A fault in any safety mechanism is a dual-point fault, since a failure in those mechanisms will not cause a safety goal violation by them selves. However if a fault occurs in a component which a faulty safety mechanism is monitoring, a safety goal violation may occur. All safety mechanisms are also latent faults, with an exception for HW watchdog failure, since they are not monitored nor tested at power up. An argumentation as to why the other failure modes got their respective classification is made in Section 5.7.4.

5.7.3 Evidence of the Effectiveness of Safety Mechanisms to avoid Single Point Faults

This section explains how single point faults are avoided with the use of safety mechanisms.

Hardware Watchdog and Tristate Buffers

The hardware watchdog monitors a PWM signal from a digital output of the microcontroller. The signal value is toggled every time the software enters the execution activity which gives the signal a frequency of 50 Hz. As long as the frequency does not deviate from 50 Hz the CAN connections are active. If a deviation from this frequency is detected the hardware watchdog will signal the tristate buffers to disconnect the CAN gateway from the rest of the CAN network. Thus if any error causes the software not to toggle the signal every 10 ms this will be detected and the CAN gateway will be cut of from the rest of the system.

ECC Mechanism in RAM

The ECC mechanism in the RAM memory has the ability to correct single bit faults in a byte and detect most dual bit faults in a byte.

Software Watchdog Timer

The watchdog timer is an internal timer that counts up toward a predefined value. The value is set so that the timer will overflow after 13.11 ms. The software resets this timer every 10 ms in the beginning of the runtime test activity. Thus, if any error causes the software to not reset the timer, this will be detected and the CAN gateway will transition into fail safe.

Tracking Register

The tracking register is a register that is set to 0x55 in the execution activity and 0xAA in the runtime test activity. Since these two activities should alternate in

regular program execution flow, it means that if the tracking register is not set to 0x55 when entering the runtime test activity, an error that disturbs normal program execution has occurred and the CAN gateway will transition into fail safe.

CAN CRC

The CAN CRC calculates a checksum of the received message. If this checksum does not add up with the checksum stored in the message then the message is resent. This captures failures on the CAN modules and on external devices.

CAN Timeout

If any specific CAN message is not received at least once every second then a timeout will be declared.

Automatic Shutdown due to Power Failure

Automatic shutdown due to power failure monitors the supply voltage received from Power supply. If the supply voltage deviates more than 0.05 V from 5 V, the safety mechanism immediately shuts down the microcontroller. This prevents any unwanted behavior of the microcontroller in case of under voltage or over voltage.

5.7.4 Diagnostic Coverage with Respect to Residual Faults

Figure 5.4 illustrates where in the fault tree our safety mechanisms prevents/detects faults. This section will evaluate the diagnostic coverage of the safety mechanisms. The diagnostic coverage is a measurement of how large part of the failure modes that will be prevented or detected by the safety mechanisms.

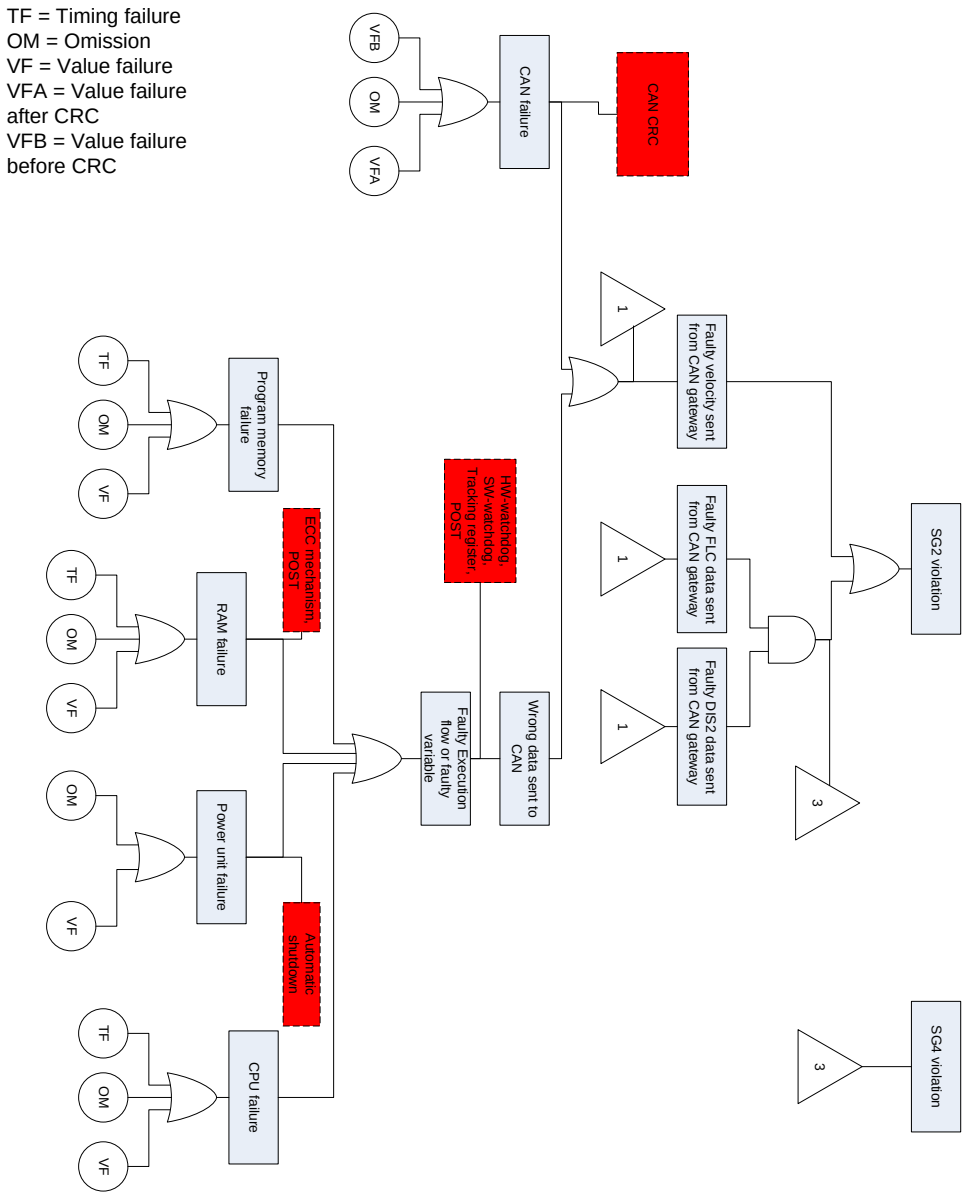

Figure 5.4: Failure modes connected to safety mechanisms in the fault tree

CAN safety mechanisms

The first branch in the tree covered by safety mechanisms is the CAN failure branch. As shown in Figure 5.5 there are three failure modes that are connected to the CAN CRC safety mechanism.

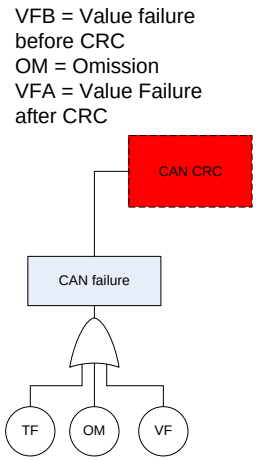

Figure 5.5: Failure modes connected to CAN safety mechanisms in the fault tree

Value failure before CRC of CAN module The first failure mode is Value failure before CRC of the CAN module, this means that the CAN module receives a corrupted message or corrupts the message while reading it. For example the CAN module samples an incoming CAN transmission at the wrong rate. If this is the case then the CRC check in the CAN module will fail and ask for a resend. For a failure to occur it is required that the value failure before CRC occurs when the CRC check function has failed. Thus, the value failure before CRC of the CAN module is a dual-point fault.

Value failure after CRC of CAN module The third failure mode is value failure after CRC of the CAN module. This failure mode can have several causes, for example a stuck at fault or a bit flip in the memory that stores the CAN messages. This would be detected if it affects messages with a known expected content. Also the CAN memory is tested for stuck at faults during the POST. However there is a risk that the failure will not be noticed and propagate upwards in the fault tree. Because of this partial coverage the value failure of CAN module is a residual fault.

Omission of CAN module The second failure mode is omission of the CAN module. This implies that the CAN module is not available to the rest of the CAN gateway. If the omission fault is persistent for more than 1 second then the timeout mechanism will detect this. For a failure to occur it is required that the omission failure occurs when the CAN timeout function has failed. Thus the omission failure of the CAN module is a dual-point fault.

RAM safety mechanisms

The second branch in the tree covered by safety mechanisms is the RAM failure branch. As shown in Figure 5.6 there are three failure modes that are connected to ECC mechanism in the RAM.

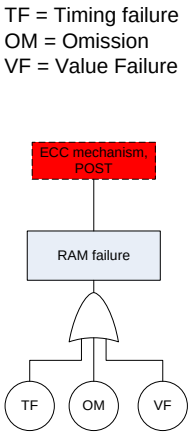

Figure 5.6: Failure modes connected to the RAM safety mechanisms in the fault tree

Timing failure of External RAM The first failure mode is timing failure of the RAM, this means that the RAM operates faster or slower than expected. For example the RAM is given a read request but takes to long to produce a stable output on the data port. This might cause the CPU to read a different byte than what was stored in the RAM. The ECC only corrects corrupted data stored in the memory cells of the RAM thus this failure mode is not covered by the ECC mechanism.

Omission of External RAM The second failure mode is omission of the RAM. This implies that the RAM module is not available to the CPU. Thus any reading from or writing to the RAM will fail. The ECC only corrects corrupted data stored in the memory cells of the RAM thus this failure mode is not covered by the ECC mechanism.

Value failure of External RAM The third failure mode is a value failure of the RAM. This failure mode can have several causes, either the value is corrupted in the memory cell due to a bit flip or a stuck at fault. In this case the fault is detected by the ECC mechanism. But there is also a possibility that a bit flip or a

stuck at fault is present in the output register of the RAM. In this case the fault will not be detected. However a bit flip or a stuck at fault in the millions of memory cells of the RAM is much more likely to occur than a fault in a single register or other parts of the RAM that is not memory cells. Because of the extremely low probability that the value failure would not be discovered we consider this failure mode covered by the ECC mechanism. For a failure to occur it is required that the value failure occurs when the ECC mechanism has failed. Thus the value failure of External RAM is a dual-point fault.

Power failure safety mechanisms

The third branch in the tree covered by safety mechanisms is the power failure branch. As can be seen in figure 5.7 there are two failure modes that are connected to the automatic shutdown mechanism in the CPU.

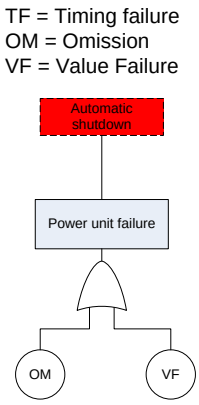

Figure 5.7: Failure modes connected to power failure safety mechanisms in the fault tree

Omission of Power Unit The first failure mode is omission of the power unit. This means that the power is cut off completely from the CPU. The automatic shutdown mechanism detects deviations from the intended power supply that are greater than 1%, and performs a safe shut down of the CPU. For a failure to occur it is required that the omission failure occurs when the Automatic shutdown mechanism has failed. Thus the omission failure of Power unit is a dual-point fault.

Value Failure of Power Unit The second failure mode is value failure of the power unit. This means that the power unit is supplying a voltage that is greater

or smaller than 5 V. For the same reason as 5.7.4 - Omission of Power Unit, the automatic shutdown mechanism will manage a value failure which deviated with more than 1% from 5 V. If the value failure deviated with less than 1% then the CPU will function correctly. For a failure to occur it is required that the value failure occurs when the Automatic shutdown mechanism has failed. Thus the value failure of Power unit is a dual-point fault.

Faulty execution flow safety mechanisms

The fourth branch in the tree covered by safety mechanisms is the faulty execution flow or faulty variable branch. As can be seen in figure 5.8 there are eleven failure modes that are connected to the hardware watchdog, software watchdog and tracking register safety mechanisms. Three of these failure modes are already covered by other safety mechanisms thus, there are eight uncovered failure modes.

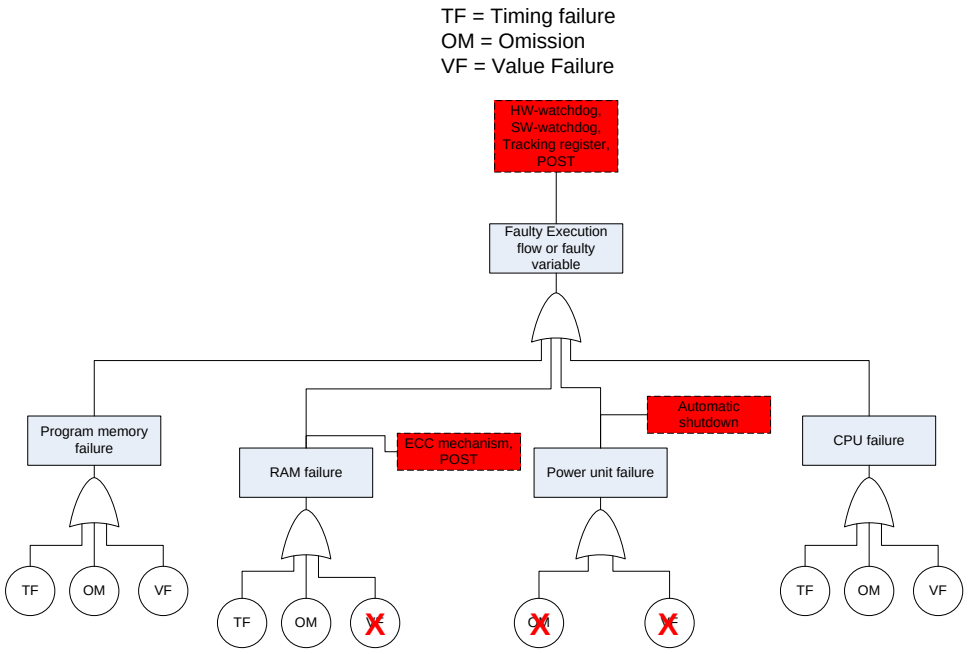

Figure 5.8: Failure modes connected to the faulty execution flow or faulty variable safety mechanisms in the fault tree. Already covered failure modes are marked with an X.

Timing Failure of Program Memory The first failure mode is timing failure of the program memory. This means that the program memory operates faster or

slower than expected. For example the program memory is given a read request but takes to long to produce a stable output on the data port. This will cause the CPU to read a different instruction than what was stored in the program memory. This leads to an undefined behavior and the intended execution flow of the program will not be followed. This will cause the PWM signal to the hardware watchdog to fail, thus this failure mode is covered by the hardware watchdog. Provided that the startup configuration was successful then the software watchdog timer wont be reset and it will overflow, thus this failure mode is also covered by the software watchdog. Even a small deviation from the execution flow which causes an execution cycle to be missed will be discovered by the software watchdog or the tracking register. For a failure to occur it is required that the timing failure occurs when the hardware watchdog, software watchdog and the tracking register mechanism has failed. Thus the timing failure of Program memory is a multi-point fault.

Omission of Program Memory The second failure mode is omission of the program memory. This implies that the program memory is not available and any reading from the program memory will fail. This will cause the CPU to read a different instruction than what was stored in the program memory. This leads to an undefined behavior and for the same reasons as in 5.7.4 - Timing Failure of Program Memory, the omission failure of Program memory is a multi-point fault.

Value Failure of Program Memory The third failure mode is a value failure in the program memory. A value failure in the program memory will cause an instruction or a constant to be faulty. This might cause a great disturbance in the execution flow but there is also the possibility that it just causes a minor fault in the execution flow. For example the condition in an if statement is altered causing the wrong statement to be executed. In the case of a minor execution alteration or a faulty constant its not sure that the fault is detected by the safety mechanisms. Therefore this failure mode is not fully covered even though a part of the possible value failures actually would be detected. Because of the partial coverage the value failure of Program memory is a residual fault.

Timing Failure of External RAM The fourth failure mode is timing failure of the RAM, this means that the RAM operates faster or slower than expected. For example the RAM is given a read request but takes to long to produce a stable output on the data port. This causes the return address from subroutines to be faulty and variables to be corrupted. This leads to an undefined behavior and for the same reasons as in 5.7.4 - Timing Failure of Program Memory, the timing failure of External RAM is a multi-point fault.

Omission of External RAM The fifth failure mode is omission of the External RAM. This implies that the RAM module is not available to the CPU. Thus any reading from or writing to the RAM will fail. This causes the return address from subroutines to be faulty and variables to be corrupted. This leads to an undefined behavior and for the same reasons as in 5.7.4 - Timing Failure of Program Memory, the omission failure of External RAM is a multi-point fault.

Timing Failure of CPU The sixth failure mode is timing failure of the CPU. This means that the CPU is executing in a different frequency than the expected one. If the timing failure is so severe that the hardware watchdog detects it than the fault is detected. This could for example be caused by a faulty clock divider. But in the case of very small deviations from the intended frequency then the fault wont be detected. Because of the partial coverage the timing failure of CPU is a residual fault.

Omission of CPU The seventh failure mode is omission of the CPU. This means that the functionality of the CPU is unavailable and the results of the execution are undefined. This will cause the PWM signal to the hardware watchdog to fail, thus this failure mode is covered by the hardware watchdog. Provided that the startup configuration was successful, the software watchdog timer will not be reset and it will overflow, thus this failure mode is covered by the software watchdog. Even a small deviation from the execution flow which causes an execution cycle to be missed will be discovered by the software watchdog or the tracking register. For a failure to occur it is required that the omission failure occurs when the hardware watchdog, software watchdog and the tracking register mechanism has failed. Thus the omission failure of CPU is a multi-point fault.

Value Failure of CPU The eighth failure mode is a value failure of the CPU. This means that the CPU performs some kind of faulty calculation, perhaps due to an error in the ALU. This could, just like a value failure in the program memory, cause a great disturbance in the execution flow. But, there is also the possibility that it just causes a minor disturbance in the execution flow or produces a faulty variable value. As an example, if a condition in an if statement is miscalculated causing the wrong statement to be executed. In the case of a minor execution alteration it is not sure that the fault is detected by the safety mechanisms. Therefore this failure mode is not fully covered even though a part of the possible value failures actually would be detected. Because of the partial coverage the value failure of CPU is a residual fault.

Results Summary

To calculate diagnostic coverage with respect to residual faults, Equation (5.1) is used. This equation is taken from ISO-26262 [6, Part 5, Annex C, Page 37]. λ_{RF} denotes the total failure rate of all residual faults, and λ_{TOT} denotes the total failure rate of all faults. As can be seen in Table 5.2, there is a total of 20 failure modes, each mode having a failure rate of $10^{-7}h^{-1}$ as given in Figure 5.3. Out of these 20 failure modes, four are classified as residual faults. Hence, $\lambda_{RF} = 4 * 10^{-7}h^{-1}$ and $\lambda_{TOT} = 20 * 10^{-7}h^{-1}$. The diagnostic coverage, in percentage, with respect to residual faults is thus given by Equation (5.2).

$$DC = (1 - \frac{\lambda_{RF}}{\lambda_{TOT}}) * 100 \tag{5.1}$$

$$DC = (1 - \frac{4 * 10^{-7}}{20 * 10^{-7}}) * 100 = 80 \tag{5.2}$$

The total diagnostic coverage with respect to residual faults is 80%

5.7.5 Evidence of the Effectiveness of Safety Mechanisms to avoid Latent Faults

At start up, the POST tests the hardware watchdog for faults by feeding it with three different PWM frequencies. These three frequencies are 25 Hz, 50 Hz and 75 Hz which are the frequencies the hardware watchdog is specified to act upon. This test protects the hardware watchdog from having a latent fault which in turn prevents it from detecting a faulty execution flow.

5.7.6 Diagnostic Coverage with Respect to Latent Faults

To calculate diagnostic coverage with respect to latent faults, Eqation (5.3) is used. The equation is taken from ISO-26262 [6, Part 5, Annex C, Page 37]. λ_{LF} denotes the total failure rate of all latent faults, and λ_{TOT} denotes to total failure rate of all faults. There are 5 failure modes classified as latent faults, and 20 total failure modes, as can be seen in Table 5.2. All failure modes have a failure rate of $10^{-7}h^{-1}$ and therefore $\lambda_{LF} = 5 * 10^{-7}h^{-1}$ and $\lambda_{TOT} = 20 * 10^{-7}h^{-1}$. The diagnostic coverage, in percentage, with respect to latent faults is thus given by Equation (5.4).

$$DC = (1 - \frac{\lambda_{LF}}{\lambda_{TOT}}) * 100 \tag{5.3}$$

$$DC = (1 - \frac{5 * 10^{-7}}{20 * 10^{-7}}) * 100 = 75 \tag{5.4}$$

The total diagnostic coverage with respect to latent faults is 75%.

5.8 Reflections and Deviations from ISO-26262 in Hardware Safety Analysis Report

This section presents reflections on and deviations from ISO-26262 in Hardware Safety Analysis Report.

Diagnostic Coverage with Respect to Latent Faults

The diagnostic coverage with respect to latent faults, presented in 5.7.6 could have been much higher. The reason behind this is that we overlooked the fact that the safety mechanisms need to be covered by safety mechanisms themselves, or at least tested at power up. Testing them at power up would result in a worst case scenario where a particular safety mechanism would not be functional, without the system detecting it, for a full driving session, which is acceptable due to the low failure rates of our components.

However, as far as we can see, there are no requirements in ISO-26262 concerning latent faults that give detailed specifications on, for example, how much diagnostic coverage is required. Though there are a requirement in ISO-26262 that demands safety mechanisms that prevents faults from being latent, if applicable [6, Part4, 6.4.4.1]. This does not clearly state that there may be no possibility of latent faults being present in the system, only that there must be safety mechanisms that prevent some faults from being latent. Another requirement states that evidence of the effectiveness of safety mechanisms to avoid latent faults shall be presented [6, Part5, 7.4.3.4]. Once again, it is not stated that the safety mechanism must prevent all faults from being latent. Considering this, we cannot know whether or not our diagnostic coverage with respect to latent faults fulfills the requirements of ISO-26262.

Evidence of the Effectiveness of Safety Mechanisms

Section 5.7.3 and Section 5.7.5 are not really proofs for the effectiveness of our safety mechanisms but more argumentation for how they work and why they prevent faults from violating safety goals.

Sources of Failure Modes and Failure Rates

We were not able to find the specific failure modes and failure rates of our hardware components at Scania. In order to get access to them we would have had to

visit the supplier. Therefore we used three common failure modes and made up failure rates that seemed plausible. However there are standards that can be used when estimating failure rates [11][12]. We did not, however, have access to them and the accuracy of the failure rates did not need to be very precise in order for us to proceed with our work.

Connections Between the FTA and Safety Requirements

Logically there should be a distinct connection between the fault tree analyses we made in this work product and the safety requirements gathered in this report. Since the FTA we made is a top-down analysis originating from the violation of safety goal 2 and safety goal 4. The safety requirements are also a sort of top down analysis originating from the safety goals. Due to time reasons we did not investigate this matter further.

5.9 5-9:WP1 Analysis of Safety Goal Violations due to Random Hardware Failures

This section is a work product resulting from Part 5, Clause 9, requirement 9.4.2 of the ISO-26262 standard. The aim is to make available criteria that can be used in a rationale that the residual risk of a safety goal violation, due to random hardware failures, is sufficiently low.

5.9.1 Evaluation of Probabilistic Metric for Random Hardware Failures

In Section 4.5.8 target values for the maximum probability of safety goal violations due to random hardware failures were derived from a table in ISO-26262 [6, Part5, Table 6]. The target values are listed in Table 5.3.

Table 5.3: Target values for probability of safety goal violation due to random hardware failure

Safety goal	ASIL	**Target value**
SG2	C	$< 10^{-7}h^{-1}$
SG4	C	$< 10^{-7}h^{-1}$

Analysis of FTA

In this section an analysis of the FTA made in Section 5.2 is conducted. Figure 5.9 and Figure 5.10 shows the individual FTA for Safety Goal 2 and Safety Goal 4 respectively. All failure modes that are covered by safety mechanisms have been removed, thus the failure modes shown in these figures are failures which will not be detected. This analysis is made under the assumption that all safety mechanisms are functional.

First an analysis of probability of a violation of Safety Goal 2 is made. Failure probability per hour of a given failure mode with a constant failure rate of λ is given by (5.5).

$$P = 1 - e^{-\lambda}h^{-1} \approx \lambda h^{-1} \tag{5.5}$$

Where the approximation is true since $\lambda < 0.1$

The failure rates of the components used are not constant, but have a bathtub curve. However, a burn-in test will be conducted on the components, as specified in Section 5.11, the failure rate can be approximated to be constant and thus (5.5) can be used.

As can be seen in Figure 5.9 Node1, Node2 and Node3 all have the same failure modes connected to them. Due to this, any failure mode will lead to a failure at all three nodes and this will propagate all the way up to SG2 violation. Thus, the probability of a failure at Node 1, is equal to the probability of a failure at Node SG2. Failure probability at Node SG2 is given by equation (5.6). In the equation, N_X denotes Node X.

$$\begin{aligned}
P(N_{SG2}) &= P(N_1) = P(N_{CAN} \cup P(N_{CPUTF} \cup N_{CPUVF})) = \\
&= P(N_{CAN}) + P(N_{CPUTF}) + P(N_{CPUVF}) - (P(N_{CAN})P(N_{CPUTF}) + \\
&+ P(N_{CAN})P(N_{CPUVF}) + P(N_{CPUTF})P(N_{CPUVF})) + \\
&+ P(N_{CAN})P(N_{CPUTF})P(N_{CPUVF}) \approx \\
&\approx P(N_{CAN}) + P(N_{CPUTF}) + P(N_{CPUVF}) \approx \\
&\approx (\lambda_{CAN} + \lambda_{CPUTF} + \lambda_{CPUVF})h^{-1}
\end{aligned} \tag{5.6}$$

Where the approximation is true since all products ≈ 0

To meet the target value, the failure rates of the failure modes need to fulfill the inequality given in (5.7).

$$\lambda_{CAN} + \lambda_{CPUTF} + \lambda_{CPUVF} < 10^{-7} \tag{5.7}$$

The failure rates of the failure modes in our design is $\lambda_{CAN} = \lambda_{CPUTF} = \lambda_{CPUVF} = 10^{-7}$, which does not fulfill the inequality given in (5.7).

However, the failure probability per hour given in (5.6) is calculated under the assumption that if one of the failure modes occur, the failure will propagate all the way up to Node SG. In reality, this is not the case.

Lets start with Node 1. If the CAN chip has a value failure due to a bit flip in its internal RAM, it does not imply that a faulty velocity message will be sent. The bit flip could just as well have affected another message, as the velocity message is only one out of thousands of messages being dealt with by the CAN chip. Also it is likely that the bit flip affects a part of the CAN chip internal memory that is not even used in our design, since only 1/16 of the memory is used. Even if the value failure resulted in a faulty velocity message being sent, there are a number of other circumstances that will affect whether or not it will result in a violation of Safety Goal 2. For example, if the correct velocity is 20 km/h but the velocity signal shows anything between 0 and 30 km/h, it will not result in a violation of a safety goal. If the velocity signal shows anything above 30 km/h, it will only result in a violation of a safety goal if the sensors gives object data that together with the faulty velocity signal would cause an evasive maneuver.

Secondly we have the two failure modes resulting in a CPU failure. As explained in Section 5.7.4, a value failure in the CPU is not necessarily detected by our safety mechanisms. That failure mode have the possibility to only cause minor disturbances in the execution flow. However, unless the fault is a transient fault which only affected a variable, it is much more likely that the whole execution flow will fail, and thus be discovered by safety mechanisms, due to bad return and jump addresses.

The same reasoning can be used for Node4 except that here both Node2 and Node3 need to fail, in other words a dual point failure has to occur. Considering all the safety mechanisms explained in Section 5.7.3, this is extremely unlikely to happen. The one mutual fault that would cause both Node2 and Node3 to fail is a stuck at fault in the CAN chip internal memory. As mentioned above, only 1/16 of the CAN chip internal memory is used, thus if a stuck at fault occurs there is only a 1/16 chance that it will affect the messages being sent.

The analysis sabove can also be applied to the probability of a violation of Safety Goal 4.

Given the arguments presented above, a failure rate for each failure mode in the design of 10^{-7} is sufficient.

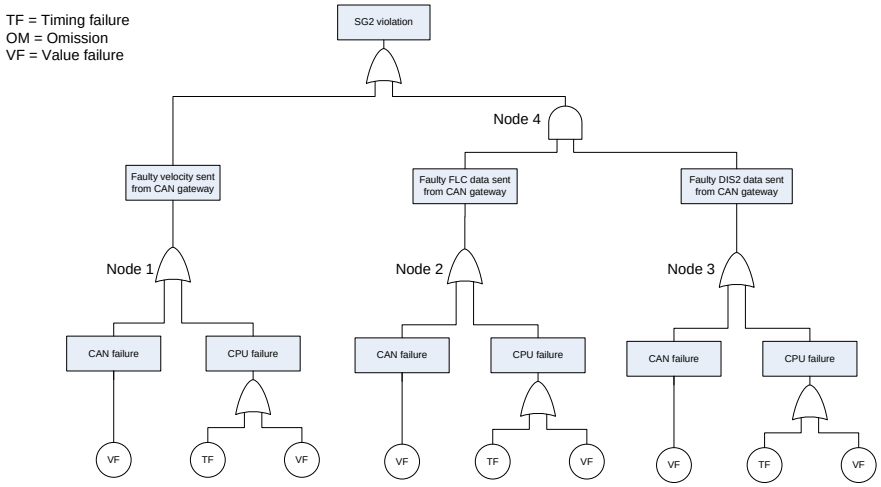

Figure 5.9: FTA of violation due to random hardware failure of Safety Goal 2

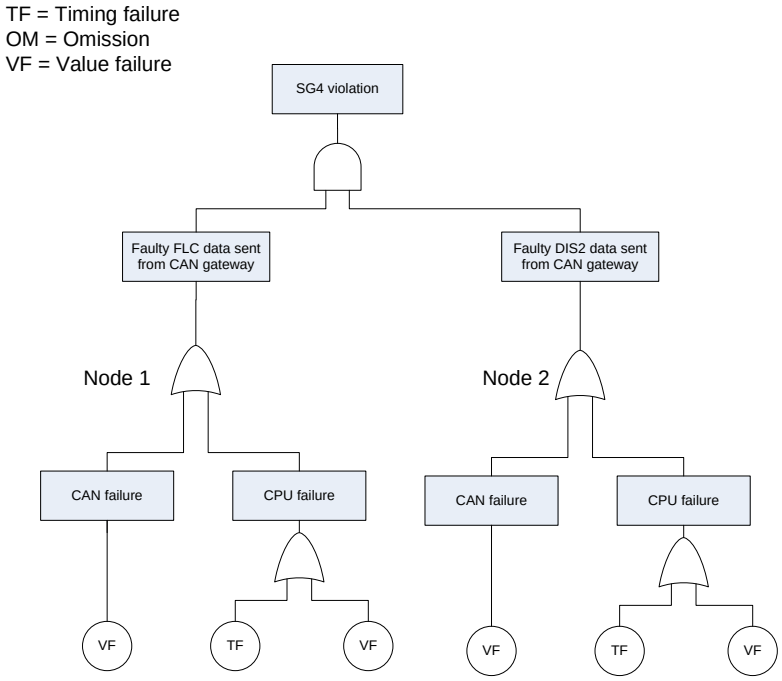

Figure 5.10: FTA of violation due to random hardware failure of Safety Goal 4

Dual Point Faults

In this section the probability of a safety goal violation due to dual-point failure is evaluated.

In our design, the only way a dual-point failure can occur is if a latent fault occurs first, i.e. a fault in a safety mechanism, and then a fault occurs in the component which the faulty safety mechanism was monitoring. Failure probability per hour of a given failure mode with a constant failure rate of λ is given by Equation (5.8), as stated in Section 5.9.1.

$$P = 1 - e^{-\lambda}h^{-1} \approx \lambda h^{-1} \tag{5.8}$$

Where the approximation is true since $\lambda < 0.1$

Due to the manner in which a safety goal violation due to a dual-point failure can occur, the probability of a safety goal violation due to a specific dual-point failure, at a certain hour in the vehicles operating time t, is given in Equation (5.9).

$$P(t) = P(SM)t * (1 - P(A))^t * P(A) \tag{5.9}$$

Where P(SM) denotes the probability per hour of a failure in the specific safety mechanism and P(A) denotes the probability per hour of a failure in the component which the safety mechanism was monitoring.

Given Equation (5.9), the probability of a specific dual-point failure in our design, at a certain hour in the vehicles operating time t, is given in Equation (5.10).

$$P(t) = 10^{-7}t * (1 - 10^{-7})^t * 10^{-7} \tag{5.10}$$

It can easily be seen in Equation (5.10) that the probability increases as t increase. Given $t = 200000 (\approx$ 23 years of operating time), which is far greater than the expected operating time of a vehicle, the probability is given by Equation (5.11)

$$P(2 * 10^5) = 10^{-7} * (2 * 10^5) * (1 - 10^{-7})^{2*10^5} * 10^{-7} = 1.96 * 10^{-9} \tag{5.11}$$

In our design there exists about 10 different possible dual-point failures, each with the worst case probability per hour of failure given in Equation (5.11). Hence the total worst case probability per hour of a safety goal violation due to a dual-point failure is $\approx 2 * 10^{-8}$. This means that our design complies with the target value of 10^{-7}.

5.10 Reflections and Deviations from ISO-26262 in Analysis of Safety Goal Violations due to Random Hardware Failures

This section presents reflections on and deviations from ISO-26262 in Analysis of Safety Goal Violations due to Random Hardware Failures.

5.10.1 Analysis

Instead of performing an analysis as done in 5.9.1, one could make a quantitative FTA as proposed in [8]. A quantitative FTA is as a normal FTA, with the exception that each node in the FTA has a certain probability of propagating the fault to the next node. To be able to do this a great knowledge of all failure modes and the probability that they will propagate up in the fault tree is needed. Since we did not possess that knowledge, we could not perform such a task. Instead an argument was made that the probability of a fault propagating all the way up to safety goal violation is sufficiently low.

5.10.2 Bad Approximation

Since a burn-in test will be conducted on our hardware components we approximated the failure rate of the components to be linear. However, a burn-in test would only cover the nonlinearity at the beginning of a bathtub curve. Therefore, when the components we use becomes old, the approximation will deviate from reality.

Objectives Clause 9

The objectives of this work product, see [6, Part 5, 9.1], is to make available evidence that the residual risk of a safety goal violation is equivalent to the residual risks on items already in use. What good does this make if there are no other items in use, just as in our case?

5.11 Specification of Dedicated Measures for Hardware

This section specifies what dedicated measures a hardware part shall be dealt with in case of a diagnostic coverage of less than 90%.

The CAN gateway has a diagnostic coverage of 80%. Therefore the CAN gateway unit must go through a burn in test before mounted in a truck.

5.12 Reflections and Deviations from ISO-26262 in Specification of Dedicated Measures for Hardware

This section presents reflections on and deviations from ISO-26262 in Specification of Dedicated Measures for Hardware.

Diagnostic Coverage Underestimated

The diagnostic coverage we have evaluated is an analysis over which failure modes our safety mechanisms cover and not cover. However, when analyzing the coverage of the failure modes we considered all failure modes that was partially covered as not covered. This led to an diagnostic coverage of only 80%, when in fact our diagnostic coverage is greater than that.

This problem could probably have been avoided or at least not have been of the same magnitude if we were to have a deeper knowledge of the hardware we are using. If we had analyzed more detailed failure modes than we did then maybe there would have been more failure modes and the distinction between a covered failure mode and a not covered failure mode would have been more clear.

Relevance of Diagnostic Coverage

Since the diagnostic coverage with respect to residual faults are under 90% ISO 26262 demands that the hardware of CAN gateway must be dealt with a dedicated measure [6, Part 5, 9.4.2.5]. Our diagnostic coverage analysis takes into consideration what failure modes are covered or not. But how much weight can you put into the result of such an analysis? A hardware element with no diagnostic coverage at all could have extremely low failure rates on its failure modes and therefore be relativly safe in comparison with a 90% diagnostic coverage hardware element that has uncovered failure modes with high failure rates.

Chapter 6

Reflections

This chapter contain reflections that did not have a direct link to a certain part of ISO-26262.

6.1 General Reflections

Different Formulations in ISO-26262

In Part 3 and Part 4 of ISO-26262 ([6, Part 3], [6, Part 4]) the requirements that only applies to certain ASILs, are written on the form "This requirement applies to ASILs C and D, ... ", while in Part 5 of ISO-26262 ([6, Part 5]) they are written on the form "This requirement applies to ASIL C and D of the safety goal." At the beginning of each part, see e.g. [6, Part 4, 4.3], there is an explanation saying that the requirements and recommendations refer to the ASIL of the safety goal, unless ASIL decomposition has been performed. In that case the ASIL resulting from the decomposition shall be complied with.

Does this mean that the ASIL resulting from decomposition is the ASIL ISO-26262 refers to even when it explicitly says " ... of the safety goal."? If that is the case, why are the requirements not written in the same manners? On the other hand, if it actually does refer to the ASIL of the safety goal, the explanation in the beginning ([6, Part 5, 4.3]) can be misleading.

Part 10, Guidelines

In Part 10 of ISO-26262, [6, Part10], guidelines are given to aid the development of a system according to ISO-26262. This could have been a great help, but the problem is that there is a big gap in it. The guidelines cover the work done up until the point where safety goals are specified. After this there is a gap until Part 5.

Hence, there is no guidelines to be found for more than half of Part 3, and nothing that covers Part 4.

ISO-26262s Use of the Word Consider

There are are number of requirements in ISO-26262 that use the word "consider". For example in Part 3, 5.4.2, ISO-26262 states: "The boundary of the item, its interfaces, and the assumptions concerning its interaction with other items and elements, shall be defined considering: a) the elements of the item; NOTE The elements could be also based on other technology b) the assumptions concerning the effects of the item's behavior on other items or elements, that is the environment of the item; c) ... ". How does one verify that a requirement that requires you to consider a number of aspects is complied with? It is very hard, if not impossible, to prove that something was considered while performing a task.

Requirement Identifiers

A note of caution, it is extremely important that all requirements have unique identifiers that are never changed. If identifiers are changed, it will cause huge problems when iterations are made due to the references between the different levels of requirements. This is standard within the industry, but we felt the need to point it out since we have done a lot of referring between requirements in this thesis.

Part 5 - Clause 8 Skipped

A misunderstanding caused us to believe that Part 5 Clause 8 of ISO-26262 was not a mandatory part of ISO-26262. However when we realized that this was not the case there were no time left to include this part in this thesis.

Chapter 7

Results

In this chapter we answer the questions given in the problem formulation and also present reflections on certain parts of our work that we would have done differently. The reflection sections following each work product are also a part of the result, however they will not be duplicated here.

7.1 Answers to the Questions Posed in the Problem Formulation

In this section we answer the questions posed in the problem formulation.

What are the advantages and disadvantages of complying with ISO 26262?

Advantages

- **Results in safe systems**
  ISO-26262 really encourages you to develop a safe system. The problem is that there are many requirements which are open for interpretation and can thus deliberately be worked around, resulting in a less safe system than ISO-26262 probably intended. Although if safe systems are what you are aiming for, we feel confident to say that ISO-26262 is a great help to achieve it.

- **Promotes thoroughness**
  ISO-26262 always promote, and often force, you to be very thorough in the development. This can lead to new ways of thinking, and make you discover certain safety aspects which might not have been under consideration earlier. ISO-26262 also demands for every step in the process to be documented. As a result, a high quality may be achieved.

- **Requirements decomposition**
  Using requirements decomposition as forced by ISO-26262 gives a good structure and a hierarchical separation of requirements. This can be beneficial in a number of ways. To begin with, the separation of requirement levels can be useful when different part of the development cycle is performed by different organizations or departments. The requirements produced by one organization/department can be used as a deliverable to the next organization/department. This gives a clear distinct interface between development steps and organizations/departments. Another benefit is that it is easier to obtain a complete set of requirements, as you constantly work your way down by detailing the requirements more and more. Also when reaching the last level of detail, you can be confident that fulfillment of those requirements lead to the fulfillment of the top-level requirements.

Disadvantages

- **Lots of room for interpretation**
  As discussed in 7.1 - Advantages, there is room for interpretation in many places in ISO-26262. It is not always fully clear what needs to be done and how. As a result of this, two different companies developing the same system, will most likely end up with two different solutions which provide different levels of functional safety. Even worse is that different departments within the same company may have different views on how certain parts of ISO-26262 should be performed. This may cause problems during the development cycle.

- **Hard to learn**
  As always when working with something new, it takes time to perform the work the first times. The more you work with it, the easier it gets. The same thing happened to us during this thesis as our work progressed. The problem is that due to all the interpretation issues, each step forward take extra time as there are always a number possible ways to do things. Almost each time something new was introduced we had to discuss back and forth what seemed to be the most reasonable approach. This is highly time-consuming.

- **Additional hardware**
  As ASIL decomposition is widely used when developing according to ISO-26262, additional hardware is required to meet the independence requirements between decomposed requirements. As an example our design is using three ECU's, while a design not complying with ISO-26262 would most likely use one ECU or possibly two. It is possible to use already existing hardware, such as ECU's already present in a vehicle. For example,

our CAN gateway could have been implemented in Scanias COO, either by developing the whole COO as ASIL C, or by proving that the functionality already implemented in COO have no possibility of having safety related effects on CAN gateway functionality.

- **Requirements decomposition**
  While this is listed in section 7.1 - Advantages, it can also be a disadvantage. ISO-26262 force you to use predefined hierarchical levels of requirements (safety goal, FSR, TSR, HSR/SSR), and these levels are not always suited and/or necessary for the system being developed or for the organization. A good procedure and tools are also necessary to manage the requirements, see section 7.2. This does not have to be a disadvantage, but if the necessary tools and knowledge is missing it is an additional cost.

What would an architecture developed according to the principles of ISO-26262 look like?

The resulting architecture can be seen in the previous chapters of this thesis report. There are however two things that stands out:

- **Three ECU's**
  In our architecture three ECU's are used. This is because redundancy was needed to reduce the ASIL classification of certain elements and also because we did not want to implement CAN gateway on one of the two redundant ECU's, since this would have resulted in ASIL C on everything implemented in that particular ECU.

- **Extra safety mechanisms**
  While some of our safety mechanisms are already in use in certain Scania systems, we have added extra safety mechanisms such as Current limter and Compare unit, as a result of complying with ISO-26262.

Which parts of ISO 26262 are sensible and which parts will just cause overhead?

There are certain bits of ISO-26262 which we did not understand the purpose of, these can be found in Section 3.4 - Questionable Parts of Item Definition, Section 3.4 - Skipped Documentation, Section 3.6 - Operating Modes and, Section 5.12 - Relevance of Diagnostic Coverage. However that does not mean it is overhead. Overall, we found the content of ISO-26262 to be relevant and not cause any particular overhead.

However there is one concern that is discussed in 7.1 - Disadvantages and that is the fact that you are forced to follow the predefined hierarchical levels of requirements. If you are developing a system of which there exists a good technical knowledge of, the functional safety requirements might not be completely necessary and thus it would have been beneficial to specify the technical safety requirements directly. ISO-26262 does not allow you to do this.

Will a system developed according to ISO-26262 really be safe?

As discussed in 7.1 - Advantages, ISO-26262 encourage you develop safe systems, but does not necessarily force you to do so. The level of functional safety achieved will greatly depend on the interpretations made. However it is impossible to say how much it is possible to get away with, until certification begins. Hopefully companies will want to develop systems according to ISO-26262 because they want safe systems, and not only because they want their system to get a ISO-26262 certification. Another factor that decides how safe the vehicle as a whole will be is how the safety goals are specified. This matter is discussed in Section 3.8.

However, if ISO-26262 is used properly, there will be a maximum probability of $10^{-7}h^{-1}$and$10^{-8}h^{-1}$ for a safety goal violation, due to random hardware failures, with ASIL C and ASIL D respectively. Together with requirements on great thoroughness throughout the whole development cycle, safe systems will in our opinion be achieved.

7.2 Lessons Learned

This section aim to gather our experiences of developing a system according to ISO-26262. The information presented here can also partly be found inside the report, but here the focus is on what we would have done differently if we were to start all over again.

Requirements Management

Throughout our work we noticed more and more how helpful it would have been to use some requirements management tool. For example, ISO-26262 demands that bi-directional references are made between requirements on different hierarchical levels. This is very hard to obtain if a requirements management tool is not used. Another problem is verifying that the requirements at a certain level does indeed fulfill the requirements at the level above. Having a tool to aid in that process would be extremely useful.

Level of Detail in Part 3

As stated earlier in the report, the preliminary architecture we used in Part 3 was very detailed. We had a pretty clear picture of how we wanted the architecture to look and thus we thought, why not describe it in detail when we already know what it will look like? At the time, we could not see any reason not to describe it in detail. However, as our work progressed to subsequent phases, several reasons were revealed:

- The gap between safety goals and functional safety requirements becomes very big and thus it is hard to verify that the functional safety requirements does indeed imply that no safety goal will be violated.
- If an element is added in a subsequent phase, you need to iterate back to Part 3 to keep the preliminary architecture consistent.
- There is no clear distinction between functional and technical safety requirements since the functional safety requirements are described very detailed.
- All subsequent phases tend to get too detailed as well, and in our case it was hard to introduce a higher level of detail in Part 5, since Part 4 was already described partly at hardware level.

Our recommendation is therefore not to use a detailed preliminary architecture in Part 3. Keep it at subsystem level, even if you are fairly sure of what these subsystems will contain.

Understanding of each Part

Parts of ISO-26262 can sometimes seem ambiguous and thus open for interpretation. As our work progressed we realized that it was easier to interpret certain parts correctly, if we had a better understanding of the subsequent parts. For example, when working on Part 3 of ISO-26262 we did certain interpretations that seemed reasonable at the time. When we then started working with Part 4, we realized those interpretations were wrong and thus had to iterate back to Part 3.

It is not easy to understand ISO-26262 without actually applying it, but the more knowledge of subsequent parts that exist, the easier it becomes to perform the work at each part correctly.

Appendix A

Functional Safety Concept

This appendix shows the first iteration of the work product Functional Safety Concept that was made. The purpose of the appendix is to allow the reader to follow the work that has been conducted.

The functional safety requirements listed in this section are not complete. To reach the final safety concept, several iterations were made.

A.1 Functional Safety Requirements

In Table A.1 through Table A.6 the functional safety requirements derived from the safety goals are listed. For understandability reasons, some of the requirements are listed more than once, due to the fact they apply to more than one safety goal. The functional safety requirements can however have different ASILs depending on what Safety Goal they are derived from. It should be noted that this is not allowed according to ISO-26262 [6, Part8, 6.4.3.1e] and requirements is therefore not duplicated in the final work product, see Section 3.10.

Table A.1: Functional safety requirements derived from all safety goals

Identifier	**Functional Safety Requirement**	**Safe state**	**Fault tolerant time**	**Allocated to element**	**ASIL**
FSR1.	The communication between the CAN gateway and the CAN bus must be free of erroneous messages.	Fail Safe	100ms	CAN gateway	C

Table A.2: Functional safety requirements derived from Safety Goal 1 (SG1)

Identifier	**Functional Safety Requirement**	**Safe State**	**Fault Tolerant Time**	**Allocated to element**	**ASIL**
FSR2.	Decision unit must always calculate a decision to perform an evasive maneuver if supported by sensor data and vehicle is on highway with a velocity greater than 30 km/h	N/A	N/A	Decision unit	A
FSR3.	The DIS2 and FLC must send correct sensor data to the COO.	N/A	N/A	DIS2 and FLC	A
FSR4.	The CAN gateway must send correct sensor data to Sensor data analysis unit.	N/A		CAN gateway	A
FSR5.	Sensor data analysis unit must perform a correct calculation of sensor data given by DIS2 and FLC.	N/A		Sensor data analysis unit	A
FSR6.	The CAN gateway must send correct messages to Decision unit.	N/A	N/A	CAN gateway	A
FSR7.	The electric motor must apply the amount of torque requested by Electric motor interface	N/A	N/A	Electric motor	A
FSR8.	Electric motor interface must apply the amount of current that has been requested by Decision unit, to the electric motor.	N/A	N/A	Electric motor interface	A

Table A.3: Functional safety requirements derived from Safety Goal 2 (SG2) and Safety Goal 4 (SG4)

Identifier	**Functional Safety Requirement**	**Safe State**	**Fault tolerant time**	**Allocated to element**	**ASIL**
FSR9.	Decision unit must never calculate a decision to perform an evasive maneuver by steering if it is not supported by sensor data.	Fail Safe	100ms	Decision unit	C
FSR10.	Decision unit must never calculate a decision to perform an evasive maneuver by steering if the vehicle is not on highway.	Fail Safe	100ms	Decision unit	C
FSR3 (Repeated)	The DIS2 and FLC must send correct sensor data to the COO.	Fail Safe	100ms	DIS2 and FLC	C
FSR4 (Repeated)	The CAN gateway must send correct sensor data to Sensor data analysis unit.	Fail Safe	100ms	External CAN gateway	C
FSR5 (Repeated)	Sensor data analysis unit must perform a correct calculation of sensor data given by DIS2 and FLC.	Fail Safe	100ms	Sensor data analysis unit	C
FSR11.	Electric motor interface must never apply current to the electric motor unless told so by Decision unit.	Fail Safe	100ms	Electric motor interface	C

Table A.4: Functional safety requirements derived from Safety Goal 3 (SG3) and Safety Goal 6 (SG6)

Identifier	**Functional Safety Requirement**	**Safe state**	**Fault tolerant time**	**Allocated to element**	**ASIL**
FSR12.	Decision unit must never calculate a decision to perform an evasive maneuver by steering if the vehicle speed is below 30 km/h.	Fail Safe	100ms	Decision unit	B
FSR6 (Repeated)	The CAN gateway must send correct messages to Decision unit.	Fail Safe	100ms	CAN gateway	B
FSR11 (Repeated)	Electric motor interface must never apply current to the electric motor unless told so by Decision unit.	Fail Safe	100ms	Electric motor interface	B

Table A.5: Functional safety requirements derived from Safety Goal 5 (SG5) and Safety Goal 7 (SG7)

Identifier	**Functional Safety Requirement**	**Safe state**	**Fault tolerant time**	**Allocated to element**	**ASIL**
FSR13.	No more than **X A** of current may be applied to the electric motor by Electric motor interface (to prevent uncontrollable torque).	Fail Safe	20ms	Electric motor interface	D

Table A.6: Functional safety requirements derived from Safety Goal 8 (SG8)

Identifier	**Functional Safety Requirement**	**Safe state**	**Fault tolerant time**	**Allocated to element**	**ASIL**
FSR14.	Decision unit must always calculate a safe trajectory to steer the vehicle into when making an evasive maneuver.	N/A	N/A	Decision unit	A
FSR6 (Repeated)	The CAN gateway must send correct messages to Decision unit.	Fail Safe	100ms	CAN gateway	A
FSR3 (Repeated)	The DIS2 and FLC must send correct sensor data to the CAN gateway	Fail Safe	100ms	DIS2 and FLC	A
FSR15.	Both side mounted sensors must send correct sensor data to the COO.	Fail Safe	100ms	Radar 24 GHz	A
FSR16.	Sensor data analysis unit must perform a correct calculation of all sensor data.	Fail Safe	100ms	Sensor data analysis unit	A
FSR4 (Repeated)	The CAN gateway must send correct sensor data to Sensor data analysis unit.	Fail Safe	100ms	CAN gateway	A
FSR7 (Repeated)	The electric motor must apply the amount of torque requested by Electric motor interface.	N/A	100ms	Electric motor	A
FSR8 (Repeated)	Electric motor interface must apply the amount of current that has been requested by Decision unit, to the electric motor.	N/A	100ms	Electric motor interface	A

Preliminary Architecture

Figure A.1 shows the first preliminary architecture that was used. For the updated preliminary architecture, see Figure 3.7. Each element has been assigned an ASIL, in respect to the functional safety requirements they implement. If an element implements several functional safety requirements, the element is assigned the highest ASIL of the functional safety requirement they implement according to

ISO-26262 [6, Part3, 8.4.3.1b].

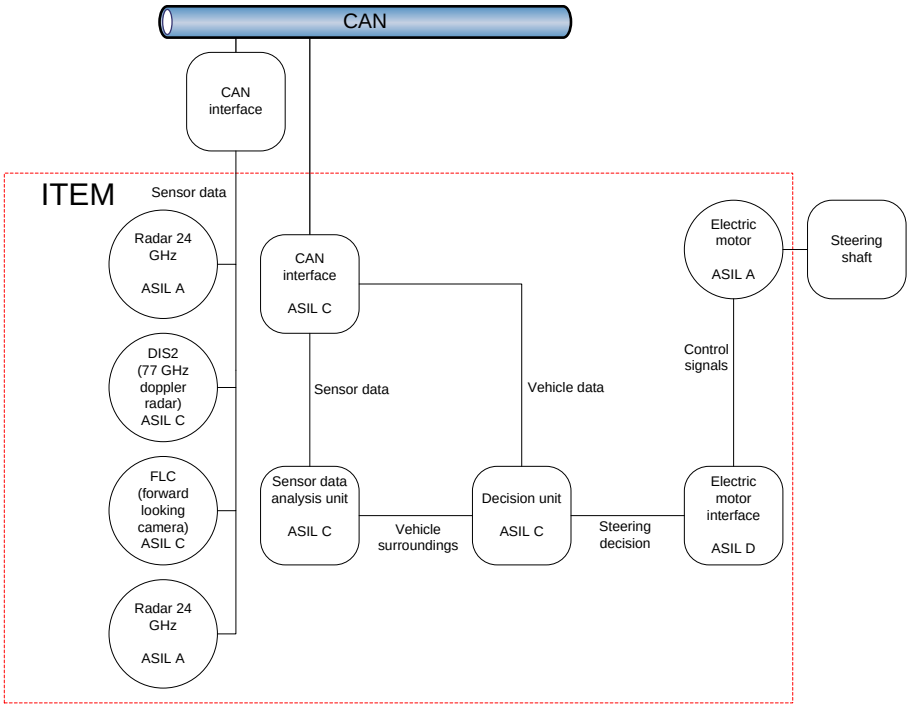

Figure A.1: The preliminary architecture from the first iteration, with ASILs assigned to the various elements.

In order to allow for ASIL decomposition on the functional safety requirements derived from SG2, SG4, SG5 and SG7, the elements Reduced decision unit, Reduced sensor data analysis unit, Compare unit and Current limiter were added to the preliminary architecture. As a result of these new elements, and their need to be independent, the preliminary architecture changed a lot. The refined preliminary architecture, with ASILs assigned to the various elements, is shown in Figure 3.7.

Also, the DIS2 and FLC both cover the area in front of the vehicle (although they have different range). A plausibility check between the values given by these two sensors will be performed by Sensor data analysis unit and Reduced sensor data analysis unit (new element) which means ASIL decomposition can be performed on FSR3.

In Appendix B it is shown how the architecture developed from Figure A.1 to the final one, see Figure 3.7, shown in Section 3.10

Appendix B

Architectural Development

This architectural assumptions have been updated several times during the course of this thesis. This is because while the work progressed the design of the overall system grew because of things like safety mechanisms and redundant elements had to be added. Figure B.1 to Figure B.6 shows how the design of the overall system has evolved during this thesis.

Figure B.1 shows the design we started out with.

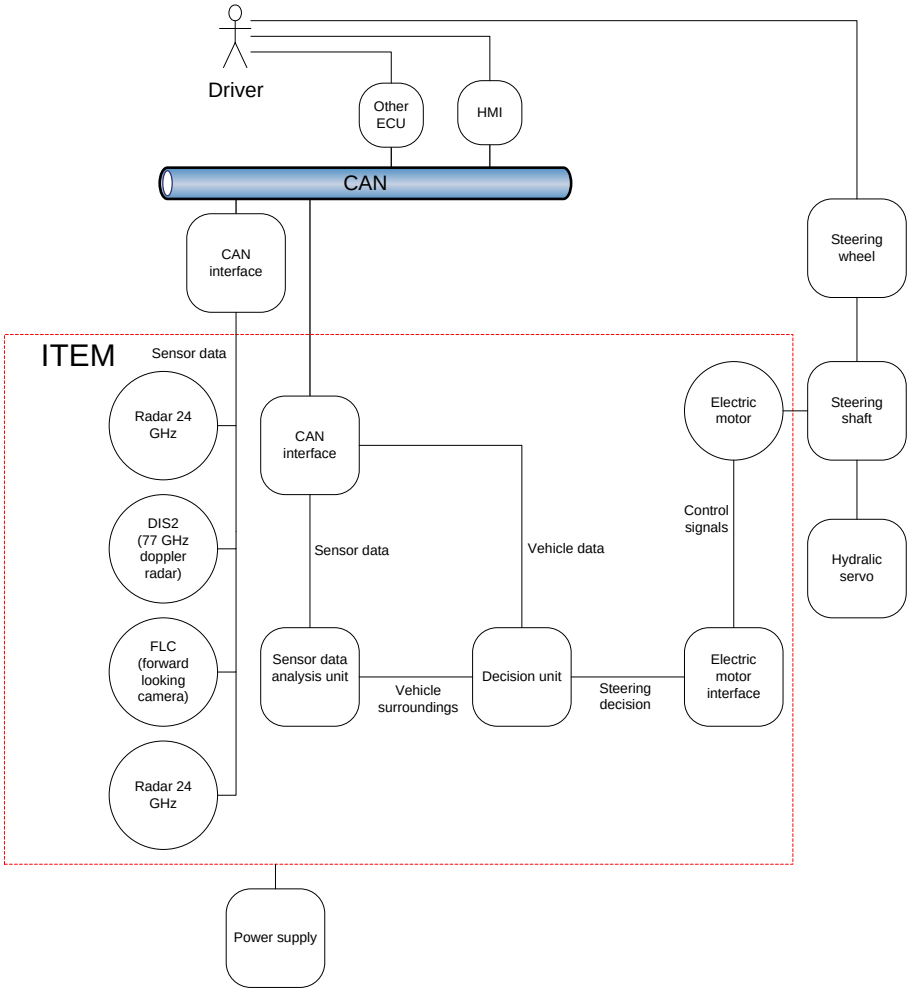

Figure B.1: The first version of the overall system design.

Figure B.2 shows the second design. Some ASIL decomposition has occurred and thus some redundant elements have been added.

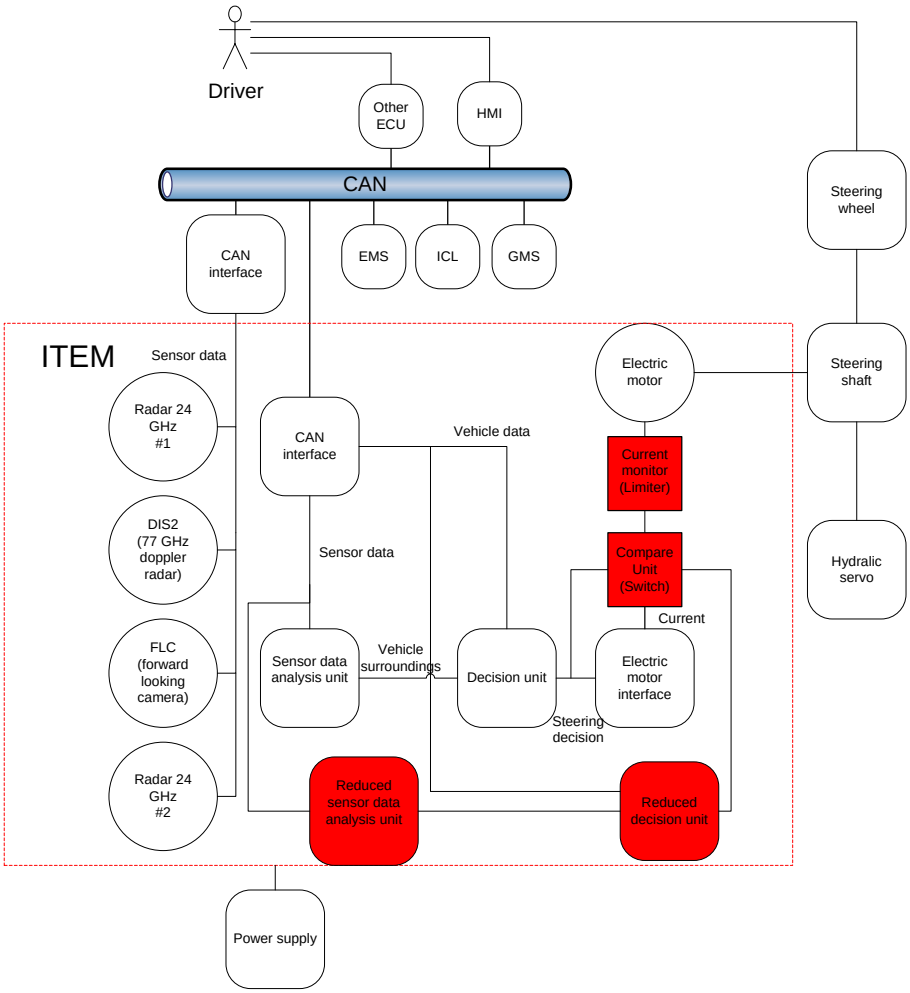

Figure B.2: The second version of the overall system design.

Figure B.3 shows the third design. Some signal descriptions have been added so that more clear requirements could be written. No elements have been added.

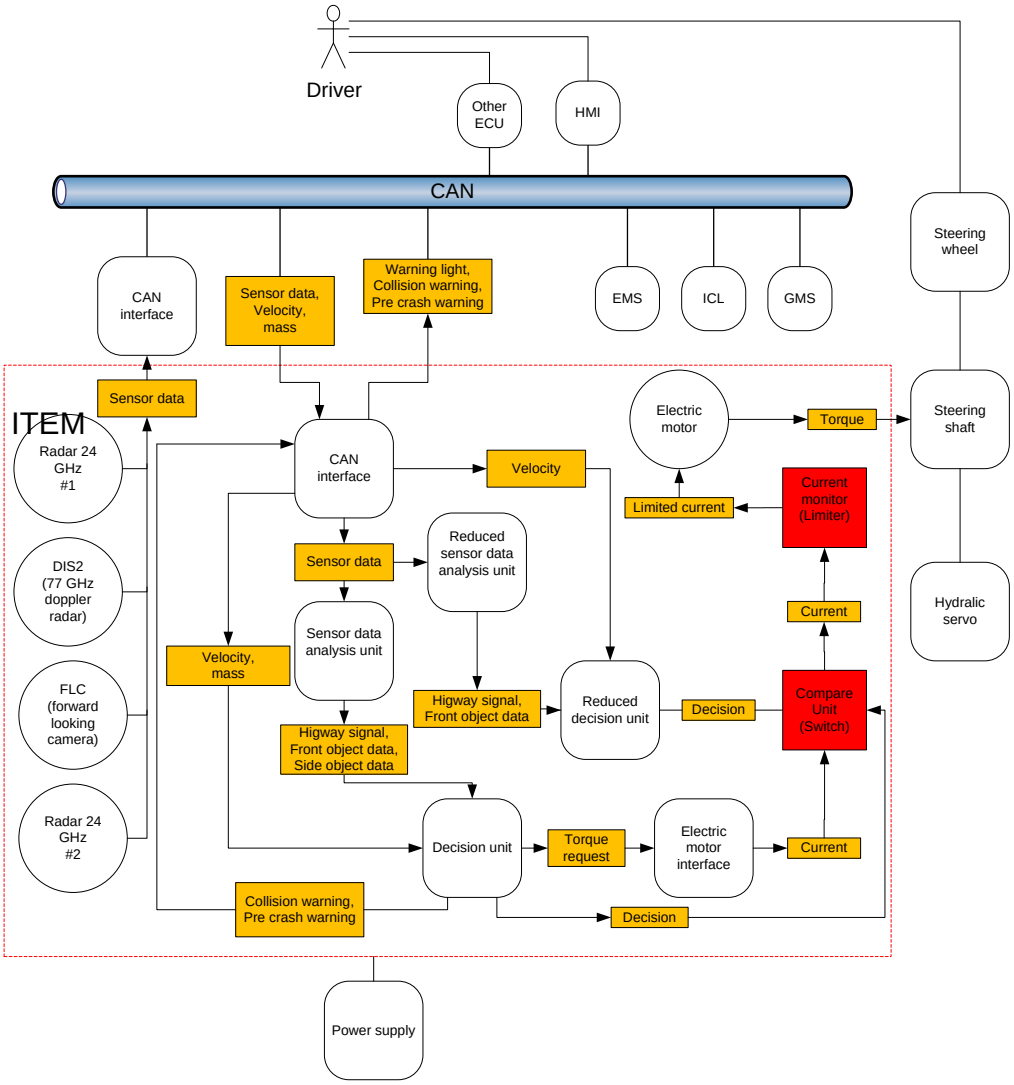

Figure B.3: The third version of the overall system design.

Figure B.4 shows the fourth design. Here some preliminary hardware decisions have been made in order to simplify the process of writing technical safety requirements. Some elements have been divided into subsystems and a CAN interface between the subsystems has been specified.

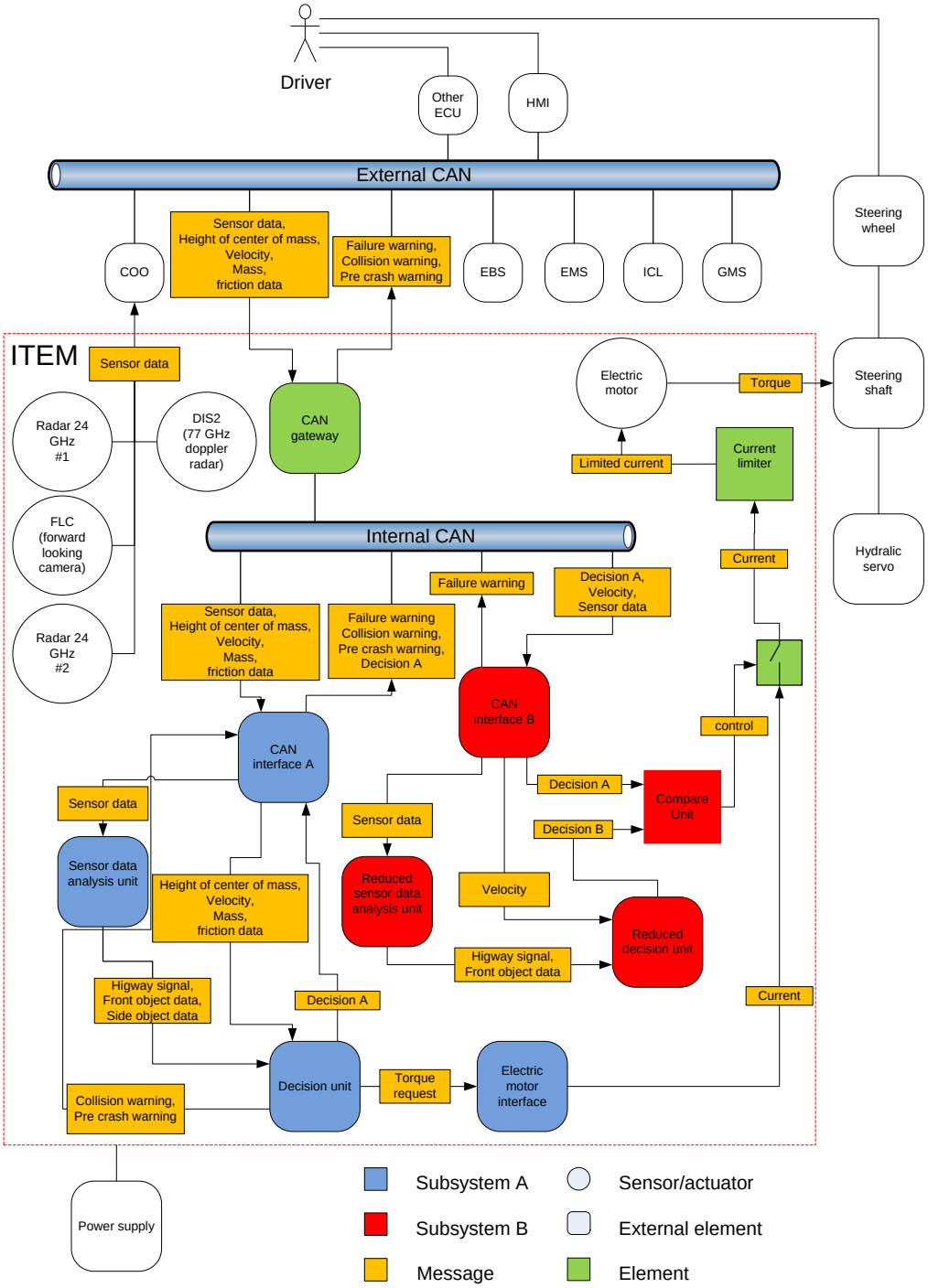

Figure B.4: The fourth version of the overall system design.

Figure B.5 shows the fifth design. Here two new elements, Subsystem A self test and Subsystem B self test, designed to detect random hardware failures have been added.

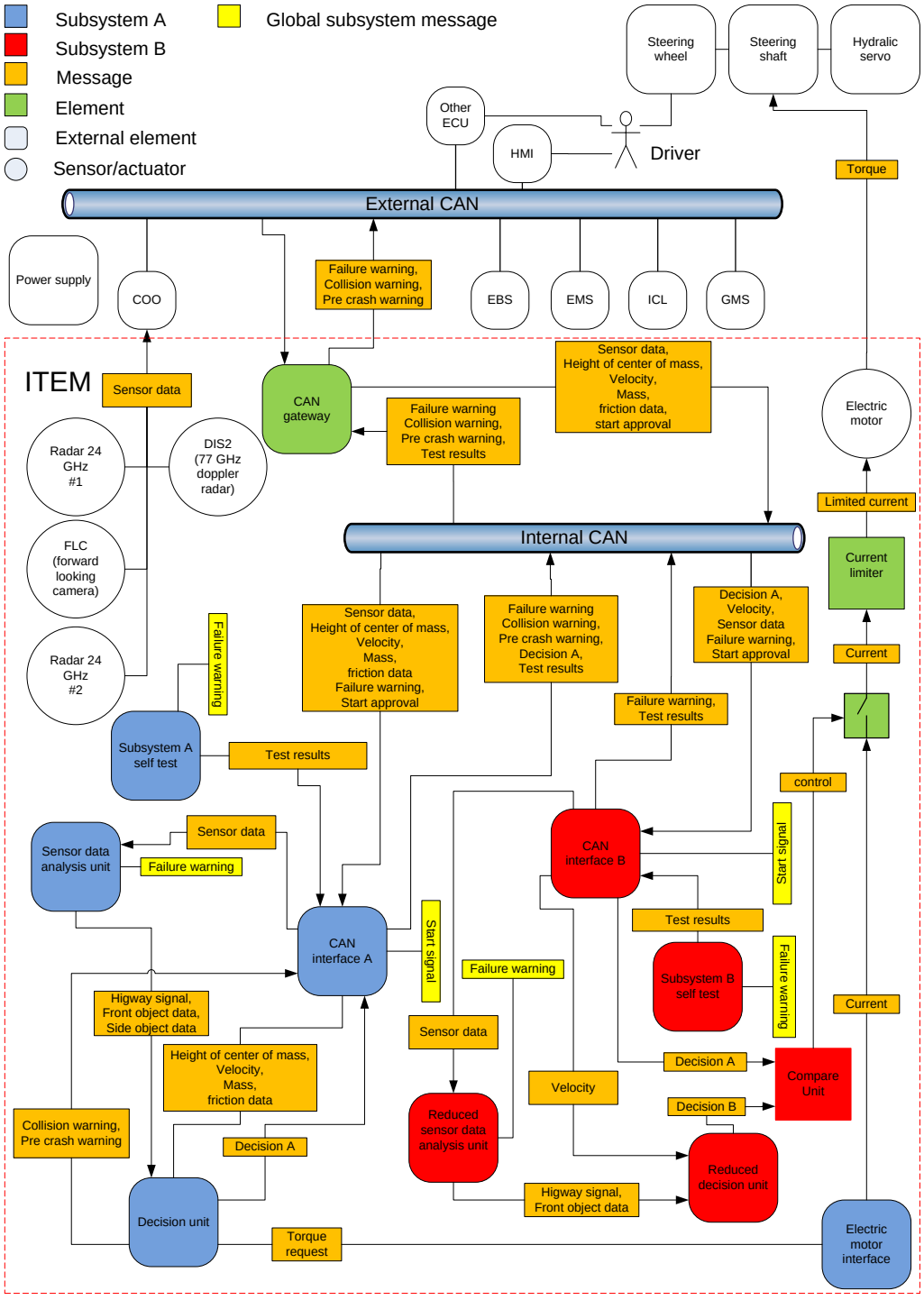

Figure B.5: The fifth version of the overall system design.

Figure B.6 shows the sixth design. The messages have been categorized to increase understandability.

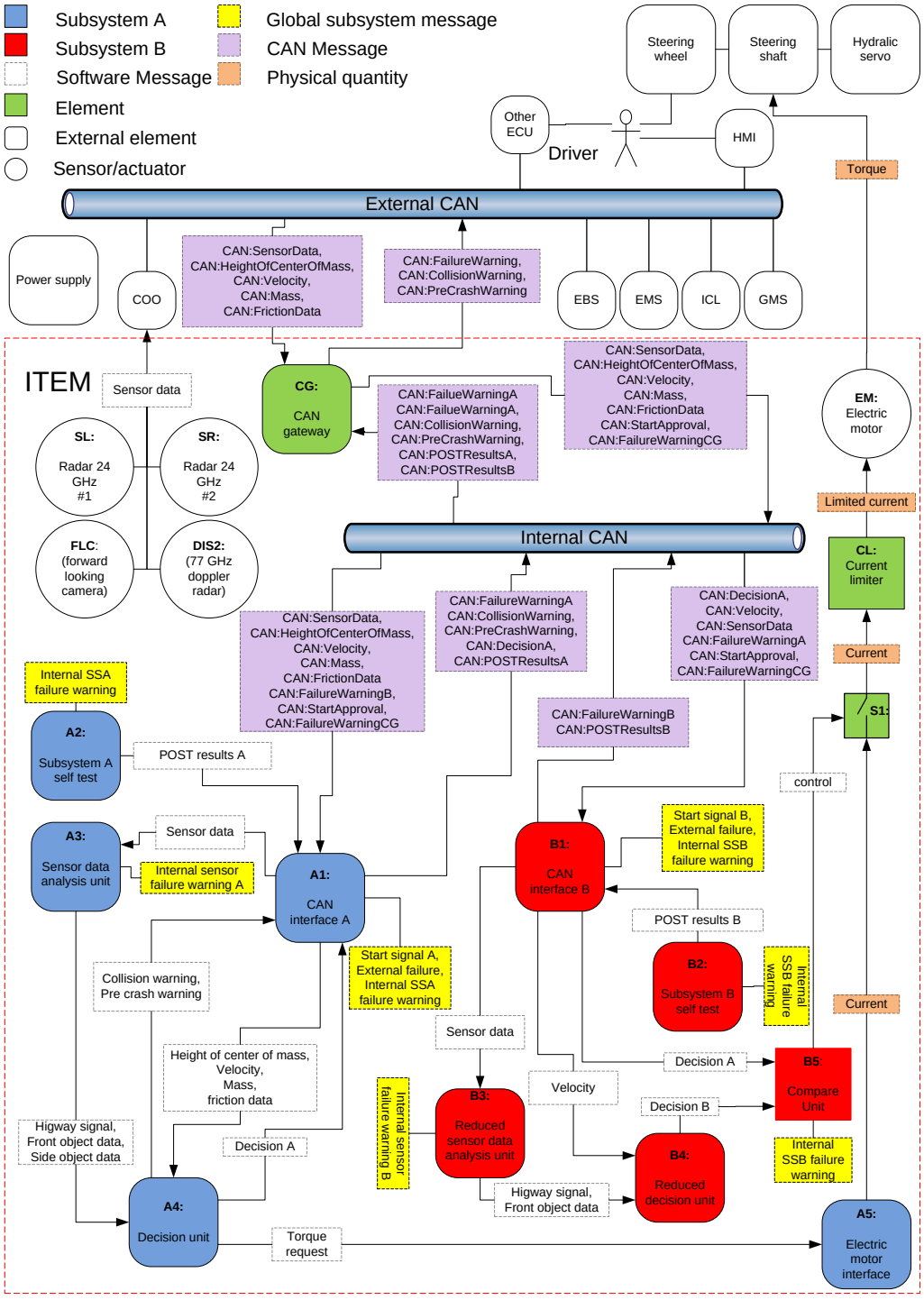

Figure B.6: The sixth version of the overall system design.

Appendix C

All Technical Safety Requirements

In this appendix technical safety requirements for the rest of the item are listed. The technical safety requirements that applies to CAN gateway are not listed in this appendix, see Table 4.1 for a list of those requirements. The requirements listed here are not complete.

Table C.1: Technical safety requirements for the rest of the system

Identifier	**Technical Safety Requirement**	**From FSR**	**Fault tolerant time**	**Allocated to element**	**ASIL**
TSR83.	Subsystem A self test hardware must tolerate temperatures between **[X]** to **[Y]** °C.	FSR57, FSR65		A2	A(C)
TSR84.	If the temperature is out of the range **[X] °Cto [Y] °C** then subsystem A self test must shut it self down.	FSR57, FSR65		A2	A(C)
TSR85.	The software implementing subsystem A self test must be correct.	FSR57, FSR65		A2	A(C)
TSR86.	Subsystem A self test must perform tests of the hardware every 50 ms.	FSR65		A2	A(C)

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR87.	If Subsystem A finds a faliure then internal SSA failure warning signal = FAILURE.	FSR52		A2	A(C)
TSR88.	subsystem A self test must perform a POST.	FSR57, FSR65		A2	A(C)
TSR89.	If the POST is successfull then POST results A = success.	FSR57		A2	A(C)
TSR90.	If the POST is unsuccessfull then POST results A = failure.	FSR57		A2	A(C)
TSR91.	At startup POST results A = testing.	FSR57		A2	A(C)
TSR92.	Subsystem A self test may start regular execution when Start signal A = GO.	FSR57		A2	A(C)
TSR93.	Subsystem A self test and subsystem B must be executed on independent hardware.	FSR36		A2	C
TSR94.	Subsystem A self test and subsystem B software must be developed independent.	FSR36		A2	C
TSR95.	Subsystem A self test and subsystem B software must be compiled using different compilers.	FSR36		A2	C
TSR96.	Subsystem B self test hardware must tolerate temperatures between **[X]** to **[Y]** °C.	FSR58, FSR66		B2	B(C)
TSR97.	If the temperature is out of the range **[X] °Cto [Y] °C** then subsystem B self test must shut it self down.	FSR58, FSR66		B2	B(C)
TSR98.	The software implementing subsystem B self test must be correct.	FSR58, FSR66		B2	B(C)

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR99.	Subsystem B self test must perform tests of the hardware every 50 ms.	FSR66		B2	B(C)
TSR100.	If Subsystem B finds a faliure then internal SSB failure warning signal = FAILURE.	FSR53		B2	B(C)
TSR101.	Subsystem B self test must perform a POST.	FSR58, FSR66		B2	B(C)
TSR102.	If the POST is successfull then POST results B = success.	FSR58		B2	B(C)
TSR103.	If the POST is unsuccessfull then POST results B = failure.	FSR58		B2	B(C)
TSR104.	At startup POST results B = testing.	FSR58		B2	B(C)
TSR105.	Subsystem B self test may start regular execution when Start signal B = GO.	FSR58		B2	B(C)
TSR106.	Subsystem B self test and subsystem A must be executed on independent hardware.	FSR36		B2	C
TSR107.	Subsystem B self test and subsystem A software must be developed independent.	FSR36		B2	C
TSR108.	Subsystem B self test and subsystem A software must be compiled using different compilers.	FSR36		B2	C
TSR109.	Forwarded messages from Internal CAN to an element in subsystem A must contain the same data.	FSR38		A1	A(C)
TSR110.	Forwarded messages from an element in subsystem A to Internal CAN must contain the same data.	FSR39		A1	A(C)

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR111.	If an incoming message from External CAN has the same ID as the SensorData message then the message must be forwarded to Sensor data analysis unit.	FSR73		A1	A
TSR112.	If an incoming message from External CAN has the same ID as the HeightOfCenterOfMass message then the message must be forwarded to Decision unit.	FSR74		A1	A
TSR113.	If an incoming message from External CAN has the same ID as the Velocity message then the message must be forwarded to Decision unit.	FSR75		A1	A
TSR114.	If an incoming message from External CAN has the same ID as the Mass message then the message must be forwarded to Decision unit.	FSR76		A1	A
TSR115.	If an incoming message from External CAN has the same ID as the FrictionData message then the message must be forwarded to Decision unit.	FSR77		A1	A
TSR116.	CAN interface A and subsystem B must be executed on independent hardware.	FSR35		A1	C
TSR117.	CAN interface A and subsystem B software must be developed independent.	FSR35		A1	C
TSR118.	CAN interface A and subsystem B software must be compiled using different compilers.	FSR35		A1	C

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR119.	CAN interface A must be configured to run CAN 2.0 B.	FSR38, FSR39, FSR57, FSR61		A1	A(C)
TSR120.	CAN interface A hardware must tolerate temperatures between **[X]** to **[Y]** °C.	FSR38, FSR39, FSR57, FSR61		A1	A(C)
TSR121.	If the temperature is out of the range **[X]** °**Cto [Y]** °**C** CAN interface A must shut it self down.	FSR38, FSR39, FSR57, FSR61		A1	A(C)
TSR122.	CAN interface A must send FailureWarningSSA message every **någon lämplig tid** ms	FSR44		A1	A(C)
TSR123.	CAN interface A must send PostResultsA message every **någon lämplig tid** ms	FSR57		A1	A(C)
TSR124.	CAN interface A must send DecisionA message every **någon lämplig tid** ms	FSR82		A1	A
TSR125.	If lFailureWarningCG = FAILURE then CAN interface A must transition into fail safe.	FSR44		A1	A(C)
TSR126.	If lFailureWarningSSB = FAILURE then CAN interface A must transition into fail safe.	FSR44		A1	A(C)
TSR127.	If InternalFailureWarningSSA = FAILURE then CAN interface A must transition into fail safe.	FSR44		A1	A(C)
TSR128.	When in fail safe: InternalFailureWarningSSA = FAILURE.	FSR44		A1	A(C)

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR129.	When in fail safe: FailureWarningSSA = FAILURE.	FSR61		A1	A(C)
TSR130.	Forwarded messages from Internal CAN to an element in subsystem B must contain the same data.	FSR41		B1	B(C)
TSR131.	Forwarded messages from an element in subsystem B to Internal CAN must contain the same data.	FSR40		B1	B(C)
TSR132.	If an incoming message from External CAN has the same ID as the SensorData message then the message must be forwarded to Reduced sensor data analysis unit.	FSR78		B1	B(C)
TSR133.	If an incoming message from External CAN has the same ID as the Velocity message then the message must be forwarded to Reduced decision unit.	FSR80		B1	B(C)
TSR134.	If an incoming message from External CAN has the same ID as the DecisionA message then the message must be forwarded to Compare unit.	FSR79		B1	B(C)
TSR135.	The software implementing CAN interface B must be correct.	FSR40, FSR41, FSR58, FSR62		B1	B(C)
TSR136.	CAN interface B and subsystem A must be executed on independent hardware.	FSR31		B1	C

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR137.	CAN interface B and subsystem A software must be developed independent.	FSR31		B1	C
TSR138.	CAN interface B and subsystem A software must be compiled using different compilers.	FSR31		B1	C
TSR139.	CAN interface B must be configured to run CAN 2.0 B.	FSR40, FSR41, FSR58, FSR62		B1	B(C)
TSR140.	CAN interface B hardware must tolerate temperatures between **[X]** to **[Y]** °C.	FSR40, FSR41, FSR58, FSR62		B1	B(C)
TSR141.	If the temperature is out of the range **[X] °Cto [Y] °C** CAN interface B must shut it self down.	FSR40, FSR41, FSR58, FSR62		B1	B(C)
TSR142.	CAN interface B must send FailureWarningSSB message every **någon lämplig tid** ms	FSR47		B1	B(C)
TSR143.	CAN interface B must send PostResultsB message every **någon lämplig tid** ms	FSR58		B1	B(C)
TSR144.	If FailureWarningSSA = FAILURE then CAN interface B must transition into fail safe.	FSR47		B1	B(C)
TSR145.	If FailureWarningCG = FAILURE then CAN interface B must transition into fail safe.	FSR47		B1	B(C)
TSR146.	If InternalFailureWarningSSB = FAILURE then CAN interface B must transition into fail safe.	FSR47		B1	B(C)

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR147.	When in fail safe: InternalFailureWarningSSB = FAILURE.	FSR47		B1	B(C)
TSR148.	When in fail safe: FailureWarningSSB = FAILURE.	FSR62		B1	B(C)
TSR149.	The software implementing the sensor data analysis unit must be correct.	FSR11, FSR22, FSR45, FSR63. FSR 57		A3	A(C)
TSR150.	Sensor data analysis unit hardware must tolerate temperatures between **[X]** to **[Y]** °C.	FSR11, FSR22, FSR45, FSR63. FSR 57		A3	A(C)
TSR151.	If the temperature is out of the range **[X] °Cto [Y] °C** sensor data analysis unit must stop execution.	FSR11, FSR22, FSR45, FSR63. FSR 57		A3	A(C)
TSR152.	Sensor data analysis unit and subsystem B must be executed on independent hardware.	FSR 33		A3	C
TSR153.	Sensor data analysis unit and subsystem B software must be developed independent.	FSR 33		A3	C
TSR154.	Sensor data analysis unit and subsystem B software must be compiled using different compilers.	FSR 33		A3	C
TSR155.	If InternalSensorFailureWarning = FAILURE the sensor data analysis unit must transition into fail safe.	FSR11, FSR22, FSR45, FSR63. FSR 57		A3	A(C)

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR156.	If internal SSA failure warning signal = FAILURE the sensor data analysis unit must transition into fail safe.	FSR11, FSR22, FSR45, FSR63. FSR 57		A3	A(C)
TSR157.	If SSB failure warning signal = FAILURE the sensor data analysis unit must transition into fail safe.	FSR11, FSR22, FSR45, FSR63. FSR 57		A3	A(C)
TSR158.	If CG failure warning signal = FAILURE the sensor data analysis unit must transition into fail safe.	FSR11, FSR22, FSR45, FSR63. FSR 57		A3	A(C)
TSR159.	When in fail safe sensor data analysis unit must halt all execution.	FSR11, FSR22, FSR45, FSR63. FSR 57		A3	A(C)
TSR160.	Sensor data analysis unit must not start execution until start signal A = GO.	FSR57		A3	A(C)
TSR161.	If the plausability check fails then internal sensor failure warning = FAILURE and sensor data analysis unit must transition inte fail safe.	FSR63		A3	A(C)
TSR162.	The software implementing the reduced sensor data analysis unit must be correct.	FSR12, FSR49, FSR64, FSR 58		B3	B(C)
TSR163.	Reduced sensor data analysis unit hardware must tolerate temperatures between **[X]** to **[Y]** °C.	FSR12, FSR49, FSR64, FSR 58		B3	B(C)

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR164.	If the temperature is out of the range **[X] °Cto [Y] °C** reduced sensor data analysis unit must stop execution.	FSR12, FSR49, FSR64, FSR 58		B3	B(C)
TSR165.	Reduced sensor data analysis unit and subsystem A must be executed on independent hardware.	FSR28		B3	C
TSR166.	Reduced sensor data analysis unit and subsystem A software must be developed independent.	FSR28		B3	C
TSR167.	Reduced sensor data analysis unit and subsystem A software must be compiled using different compilers.	FSR28		B3	C
TSR168.	If internal sensor failure warning signal = FAILURE the reduced sensor data analysis unit must transition into fail safe.	FSR12, FSR49, FSR64, FSR 58		B3	B(C)
TSR169.	If internal SSB failure warning signal = FAILURE the reduced sensor data analysis unit must transition into fail safe.	FSR12, FSR49, FSR64, FSR 58		B3	B(C)
TSR170.	If SSA failure warning signal = FAILURE the reduced sensor data analysis unit must transition into fail safe.	FSR12, FSR49, FSR64, FSR 58		B3	B(C)
TSR171.	If CG failure warning signal = FAILURE the reduced sensor data analysis unit must transition into fail safe.	FSR12, FSR49, FSR64, FSR 58		B3	B(C)

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR172.	When in fail safe the reduced sensor data analysis unit must halt all execution.	FSR12, FSR49, FSR64, FSR 58		B3	B(C)
TSR173.	Reduced sensor data analysis unit must not start execution until start signal B = GO.	FSR58		B3	B(C)
TSR174.	If the plausability check fails then internal sensor failure warning = FAILURE and reduced sensor data analysis unit must transition inte fail safe.	FSR64		B3	B(C)
TSR175.	The software implementing the decision unit must be correct.	FSR2, FSR5, FSR6, FSR15, FSR19, FSR43, FSR57		A4	A(C)
TSR176.	Decision unit hardware must tolerate temperatures between **[X]** to **[Y]** °C.	FSR2, FSR5, FSR6, FSR15, FSR19, FSR43, FSR57		A4	A(C)
TSR177.	If the temperature is out of the range **[X]** °**Cto [Y]** °**C** the decision unit must stop execution.	FSR2, FSR5, FSR6, FSR15, FSR19, FSR43, FSR57		A4	A(C)

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR178.	If internal sensor failure warning signal = FAILURE the decision unit must transition into fail safe.	FSR2, FSR5, FSR6, FSR15, FSR19, FSR43, FSR57		A4	A(C)
TSR179.	If internal SSA failure warning signal = FAILURE the decision unit must transition into fail safe.	FSR2, FSR5, FSR6, FSR15, FSR19, FSR43, FSR57		A4	A(C)
TSR180.	If SSB failure warning signal = FAILURE the decision unit must transition into fail safe.	FSR2, FSR5, FSR6, FSR15, FSR19, FSR43, FSR57		A4	A(C)
TSR181.	If CG failure warning signal = FAILURE the decision unit must transition into fail safe.	FSR2, FSR5, FSR6, FSR15, FSR19, FSR43, FSR57		A4	A(C)
TSR182.	When in fail safe the decision unit must not start execution.	FSR2, FSR5, FSR6, FSR15, FSR19, FSR43, FSR57		A4	A(C)

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR183.	Decision unit and subsystem B must be executed on independent hardware.	FSR34		A4	C
TSR184.	Decision unit and subsystem B software must be developed independent.	FSR34		A4	C
TSR185.	Decision unit and subsystem B software must be compiled using different compilers.	FSR34		A4	C
TSR186.	Decision unit must halt all execution until start signal A = GO.	FSR57		A4	A(C)
TSR187.	The software implementing the reduced decision unit must be correct.	FSR3, FSR7, FSR8, FSR16, FSR48, FSR58		B4	B(C)
TSR188.	Reduces decision unit hardware must tolerate temperatures between **[X]** to **[Y]** °C.	FSR3, FSR7, FSR8, FSR16, FSR48, FSR58		B4	B(C)
TSR189.	If the temperature is out of the range **[X] °Cto [Y] °C** the reduced decision unit must stop execution.	FSR3, FSR7, FSR8, FSR16, FSR48, FSR58		B4	B(C)
TSR190.	If internal sensor failure warning signal = FAILURE the reduced decision unit must transition into fail safe.	FSR3, FSR7, FSR8, FSR16, FSR48, FSR58		B4	B(C)

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR191.	If internal SSB failure warning signal = FAILURE the reduced decision unit must transition into fail safe.	FSR3, FSR7, FSR8, FSR16, FSR48, FSR58		B4	B(C)
TSR192.	If SSA failure warning signal = FAILURE the reduced decision unit must transition into fail safe.	FSR3, FSR7, FSR8, FSR16, FSR48, FSR58		B4	B(C)
TSR193.	If CG failure warning signal = FAILURE the reduced decision unit must transition into fail safe.	FSR3, FSR7, FSR8, FSR16, FSR48, FSR58		B4	B(C)
TSR194.	When in fail safe the reduced decision unit must halt all execution.	FSR3, FSR7, FSR8, FSR16, FSR48, FSR58		B4	B(C)
TSR195.	Reduced decision unit and subsystem A must be executed on independent hardware.	FSR29		B4	C
TSR196.	Reduced decision unit and subsystem A software must be developed independent.	FSR29		B4	C
TSR197.	Reduced decision unit and subsystem A software must be compiled using different compilers.	FSR29		B4	C

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR198.	Reduced decision unit must not start execution until start signal B = GO.	FSR58		B4	B(C)
TSR199.	The software implementing the electric motor interface must be correct.	FSR14, FSR17, FSR24, FSR46, FSR57		A5	A(D)
TSR200.	Electric motor interface hardware must tolerate temperatures between **[X]** to **[Y]** °C.	FSR14, FSR17, FSR24, FSR46, FSR57		A5	A(D)
TSR201.	If the temperature is out of the range **[X]** °**Cto [Y]** °**C** the electric motor interface must stop execution.	FSR14, FSR17, FSR24, FSR46, FSR57		A5	A(D)
TSR202.	If internal sensor failure warning signal = FAILURE the decision unit must transition into fail safe.	FSR14, FSR17, FSR24, FSR46, FSR57		A5	A(D)
TSR203.	If internal SSA failure warning signal = FAILURE the electric motor interface must transition into fail safe.	FSR14, FSR17, FSR24, FSR46, FSR57		A5	A(D)
TSR204.	If SSB failure warning signal = FAILURE the electric motor interface must transition into fail safe.	FSR14, FSR17, FSR24, FSR46, FSR57		A5	A(D)

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR205.	If CG failure warning signal = FAILURE the electric motor interface must transition into fail safe.	FSR14, FSR17, FSR24, FSR46, FSR57		A5	A(D)
TSR206.	When in fail safe the electric motor interface must halt all execution.	FSR14, FSR17, FSR24, FSR46, FSR57		A5	A(D)
TSR207.	Electric motor interface and subsystem B must be executed on independent hardware.	FSR37		A5	C
TSR208.	Electric motor interface and subsystem B software must be developed independent.	FSR37		A5	C
TSR209.	Electric motor interface and subsystem B software must be compiled using different compilers.	FSR37		A5	C
TSR210.	Electric motor interface must not start execution until start signal A = GO.	FSR57		A5	A(C)
TSR211.	The software implementing the compare unit must be correct.	FSR25, FSR42, FSR50, FSR51, FSR58, FSR59		CU	B(C)
TSR212.	Compare unit hardware must tolerate temperatures between **[X]** to **[Y]** °C.	FSR25, FSR42, FSR50, FSR51, FSR58, FSR59		CU	B(C)

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR213.	If the temperature is out of the range **[X] °Cto [Y] °C** the compare unit must stop execution.	FSR25, FSR42, FSR50, FSR51, FSR58, FSR59		CU	B(C)
TSR214.	If internal sensor failure warning signal = FAILURE the compare unit must transition into fail safe.	FSR25, FSR42, FSR50, FSR51, FSR58, FSR59		CU	B(C)
TSR215.	If internal SSB failure warning signal = FAILURE the compare unit must transition into fail safe.	FSR25, FSR42, FSR50, FSR51, FSR58, FSR59		CU	B(C)
TSR216.	If SSA failure warning signal = FAILURE the compare unit must transition into fail safe.	FSR25, FSR42, FSR50, FSR51, FSR58, FSR59		CU	B(C)
TSR217.	If CG failure warning signal = FAILURE the compare unit must transition into fail safe.	FSR25, FSR42, FSR50, FSR51, FSR58, FSR59		CU	B(C)
TSR218.	When in fail safe the compare unit must halt all execution.	FSR25, FSR42, FSR50, FSR51, FSR58, FSR59		CU	B(C)

Table continued on next page

Identifier	Technical Safety Requirement	From FSR	Fault tolerant time	Allocated to element	ASIL
TSR219.	Compare unit and subsystem A must be executed on independent hardware.	FSR30		CU	C
TSR220.	Compare unit and subsystem A software must be developed independent.	FSR30		CU	C
TSR221.	Compare unit and subsystem A software must be compiled using different compilers.	FSR30		CU	C
TSR222.	Compare unit must not start execution until start signal B = GO.	FSR58		CU	B(C)
TSR223.	If Decision A != Decision B then SSB failure warning then compare unit must transition into fail safe.	FSR51		CU	B(C)
TSR224.	If Decision A != Decision B then SSB failure warning = Failure.	FSR51		CU	B(C)
TSR225.	The switch must tolerate temperatures between **[X]** to **[Y]** °C.	FSR13, FSR26		S1	B(C)
TSR226.	The switch must be implemented in correct hardware.	FSR13, FSR26		S1	B(C)
TSR227.	The current limiter must tolerate temperatures between **[X]** to **[Y]** °C.	FSR18, FSR27		CL	C(D)

Bibliography

[1] http://www.who.int/violence_injury_prevention/publications/road_traffic/world_report/en/
2011-02-23

[2] Fredrik Walderyd
Hazard identification and safety goals on power electronics in hybrid vehicles
Chalmers University of Technology.
2010

[3] Torsten Dittel and Hans-Jörg Aryus
How to ”Survive” a Safety Case According to ISO 26262
2010

[4] Scania Internal Document AEBS/LDWS-09-01

[5] Scania Internal Document AEBS/LDWS-09-02

[6] ISO-26262
FDIS Jan 2011

[7] C167CS, 16-Bit CMOS Single-Chip Microcontrollers
User's Manual Version 1.0, 1999-05

[8] Quantified Risk Assessment Techniques - Part 3 Fault Tree Analysis - FTA
The Institution of Engineering and Technology Health & Safety Briefing No.
26c October 2010

[9] http://www.dtic.mil/cgi-bin/GetTRDoc?AD=AD739001&Location=U2&doc=GetTRDoc.pdf
2011-05-20

[10] Potential Failure Mode and Effect Analysis
Forth Edition
Chrysler LLC, Ford Motor Company, General Motors Corporation

[11] MIILITARY STANDARD REQUIREMENTS FOR NONDESTRUCTIVE TESTING METHODS
Mil-hdbk-271F
1986

[12] Telcordia document SR-332, Issue 1

[13] Fault Tree Analysis
Clifton A. Ericson II
2000