UNIVERSITÀ DEGLI STUDI DI FERRARA

Dottorato di ricerca in scienze dell'ingegneria

Ciclo XXX

Coordinatore: Prof. Trillo Stefano

Introduction of wireless technology in agricultural and heavy duty vehicles: security and safety issues

Settore Scientifico Disciplinare ING-INF/05

Dottorando

Dott. Selvatici Michele

Tutore

Prof. Ruggeri Massimiliano

Anni 2014-2017

Index

Chapter 1. Introduction 1

1.1 Organization of this thesis 2

Chapter 2. Main safety standards 5

2.1 Standards and definitions 5

2.1.1 Functional safety 5

2.1.2 IEC61508 6

2.1.3 ISO13849 7

2.1.4 ISO15998 9

2.1.5 ISO25119 10

2.2 Software quality 12

2.2.1 Software development cycle 12

2.2.2 Software specifications 14

2.2.3 Software implementation 14

2.2.4 Software testing 18

2.3 Hardware architecture for safety-critical systems 19

2.4 Diagnostic coverage 20

2.5 Safe communication 23

Chapter 3. In-vehicle networks 25

3.1 Overview of wireless networks 25

3.2 Overview of wired networks 27

3.3 Security aspects 28

3.3.1 General considerations 30

3.3.2 Modeling the attacker 31

3.3.3 Challenges 34

3.3.4 CAN: the present standard 35

3.3.5 Ethernet and Time Sensitive Networking: the future network 49
3.4 A gateway for hybrid networks 51
3.4.1 CAN and Ethernet: differences that the gateway should smooth over 51
3.4.2 ISOBUS 52
3.4.3 Comparison between TCP/IP and ISOBUS 53
3.4.4 Implementation challenges 54
3.4.5 Architectural overview 55
3.4.6 Address management 56
3.4.7 Experimental implementation 58
Chapter 4. Safety architecture 61
4.1 Software study for WSN 61
4.1.1 Operative systems 63
4.2 Hardware study for WSN 73
4.2.1 State of the art 73
4.2.2 Hardware category 2: a future proof choice 74
4.2.3 Diagnostic 75
4.3 Experimental prototype 80
4.3.1 Power supply 84
4.3.2 Main microcontroller 85
4.3.3 Safety microcontroller 86
4.3.4 Diagnostic coverage 86
4.4 Test specification and report 89
Chapter 5. Safe WSN for Human to Machine Interface 95
5.1 Gesture recognition 95
5.1.1 State of the art 96
5.1.2 Smart glove 97

Chapter 6. Conclusions 105

Chapter 7. Bibliography 107

Chapter 8. Author's publication list 113

8.1 Journals 113

8.2 Conferences 113

8.3 CNR internal reports and project deliverables 113

Figures

Figure 1. V-model for the overall system design, redrawn from ISO 25119 11
Figure 2. The V-Cycle software development cycle. 13
Figure 3. The Waterfall software development cycle..... 13
Figure 4. Hardware categories according to ISO 25119. 19
Figure 5. Diagnostic coverage and dangerous faults 22
Figure 6. Message protection scheme..... 38
Figure 7. Encryption with CTR mode (left) and decryption with CTR mode (right).40
Figure 8. Normalised average Pre obtained through simulation and from equation 6, for different values of τ_{size}. 44
Figure 9. Normalised worst-case Pre obtained with $E' \sim B(n, k, p)$ through calculation and simulation using a 45
Figure 10. Normalised worst-case Pre obtained with $E' \sim B(n, k, p)$ through calculation and simulation using a DES/SHA1 scheme, with $\mu size + \tau size = 64$, for different values of p. 46
Figure 11. Authentication and Verification procedure with a MAC code. 50
Figure 12. ISO/OSI stack's replacement 53
Figure 13. An example of an hybrid CAN/Ethernet network. 56
Figure 14. Communication diagram for the procedure in case a new CAN device reaches the network. 57
Figure 15. Communication diagram for the procedure in case a new Ethernet device reaches the network. 58
Figure 16. Demonstrator setup. 60
Figure 17. Domain separation for four tasks with different levels of criticality and real-time requirements. Colour indicates the level of criticality: red high, yellow medium, green none. 66
Figure 18. WSN node typical architecture. 73
Figure 19. Power consumption during normal operation compared to consumption during transceiver problems..... 78
Figure 20. WSN module architecture 80

Figure 21. WSN module architecture division according to ISO25119 hardware category 2 definition..... 81
Figure 22. Prototype of wireless sensor network node suitable for safety related applications..... 83
Figure 23. The smart glove..... 98
Figure 24. CTPE Sensor with safety wire. 99
Figure 25. Input acquisition layout..... 101
Figure 26. Setup of the WSN used for sensor reading. 101
Figure 27. Gestures and sensors' readings 102

Tables

Table 1. Efficiency of measures in case of possible transmission errors, redrawn from ISO 15998:2008, Annex D, Table D.1 9
Table 2. Diagnostic coverage estimation for electrical subsystems without micro-controllers, redrawn from ISO 25119:2010, Part 2, Annex C, Table C.1 22
Table 3. Summary of attack and defense strategies..... 33
Table 4. Recommended key length in bits for asymmetric-key and symmetric-key cryptography [28]. Additional references are available at http://www.keylength.com..... 38
Table 5. Summary of message protection schemes. 48
Table 6. Content of the message periodically sent from MMC to SMC. 79
Table 7. Content of the message periodically sent from SMC to MMC. 80
Table 8. Diagnostic coverage estimation for power supply, redrawn from ISO 25119:2010, Part 2, Annex C 86
Table 9. Diagnostic coverage estimation for inputs and outputs, redrawn from ISO 25119:2010, Part 2, Annex C. 87
Table 10. Diagnostic coverage estimation for processing units, redrawn from ISO 25119:2010, Part 2, Annex C, Table C.3..... 88
Table 11. Tests performed in order to verify the correct behaviour in case of fault on input. 89
Table 12. Tests performed in order to verify the correct behaviour in case of fault on SMC. 90
Table 13. Tests performed in order to verify the correct behaviour in case of fault on MMC. 91
Table 14. Tests performed in order to verify the correct behaviour in case of fault on output. 93
Table 15. Gesture match table 103

Chapter 1. Introduction

Automotive and agricultural world are strictly correlated. Automobile sector, in fact, thanks to its remarkable production and sales volumes, is leading for what concerns research and development. For this reason, it is the technological precursor of the sector of agricultural and earthmoving machines. It is possible, as a consequence, to use the experiences of automotive world in order to foresee what will happen for agricultural vehicles and prevent potential problems.

In automotive field the wireless technologies already performed their entrance, in particular for infotainment. But they had begun to be used also for diagnosis, assistance and remote control services. It is sufficient to consider how many car models have their own internet connection through 3G/4G technology generating an on-board Wi-Fi with which it is possible to use this connection. Moreover, just to give a practical example, it is possible to consider the most recent models of BMW™ with their remote services, named "ConnectedDrive Services" that spaces from the easiest and less demanding possibility to see on the smartphone data about consumptions or other usage statistics, arriving to a real remote control of the vehicle, opening and closing doors and windows, switching off and on the lights and, more critical, commanding the vehicle from outside it during the parking phase ("Remote control parking").

Going back to agricultural field, the first wireless services introduced are those of data publication, useful for fleet management or diagnosis. But, as stated before, the automotive world is precursor of agricultural one. For this reason, it is logical to suppose that the use of wireless technologies will grow.

First of all, this implies the creation of an access door on vehicular networks to the outer world. These networks, until now, has always been closed, not accessible from remote and difficult to be accessed in local. This exposition leads to the necessity of protection from potential issues of cyber-security. If adequate measures will not be applied, the risk is to see the repetition of what happened in automotive world (the main example is the case of the hack performed on a Jeep Cherokee by Valasek and Miller in 2015).

The applications realized until now with wireless technology in agricultural field are not relevant for functional safety. However, looking forward to a grow of wireless usage, a good question would be if it is going to be possible to adopt them for safety relevant applications. The answer is a yes, but with many modifications to the present-day architectures. It is necessary to perform a deep study in order to let this technology be compliant to the existing safety standards, trying to give future proof solutions.

1.1 Organization of this thesis

As described in the previous section, in this thesis are going to be analyzed two main topics, both originated by the adoption of wireless technology in agricultural field:

- How the introduction of wireless technology in vehicular networks can impact security aspects, in particular for CAN bus, and how can them be faced.
- How to modify the wireless technology in order to be adopted in safety applications.

Chapter 2 presents the main principles and standards of functional safety, in particular applied to vehicles. This chapter is necessary because it is the basis of the following ones and of the work of this thesis.

Chapter 3 gives an overview of present day vehicular networks, analyzing the possibility of satisfaction of safety and security requirements, as required by the introduction of the outer word accessibility due to wireless technology. Considering that the present day vehicular network is not able to satisfy all the requirements, the future vehicular network is analyzed. In this chapter will be presented the first two contributions of this thesis: the first one is a tradeoff analysis between safety and security in CAN network, the second one is a gateway CAN-Ethernet that will allow a gradual passage from present day-technology to future one.

Chapter 4 presents two contributions of this thesis in order to study the architecture of Wireless Sensor Network modules to let them be suitable for safety critical applications and describes the design of an experimental prototype of a wireless module. The three contributions are: the hardware and software architecture analysis and the presentation of a prototype of node realized according to the analysis performed.

Chapter 5 regards an application of wireless sensor network technology. The application is in the advanced HMI field and will be analyzed the problem of a safe gesture recognition for safety-critical applications and applies those principles to a HMI that uses both safe wireless sensor networks technologies and safe gesture recognition. This application is another contribution of this thesis.

Finally, Chapter 6 concludes the thesis, summarizing the main results reached.

Chapter 2. Main safety standards

This chapter will present the main safety standards applicable to vehicular field. This explanation of the main norms is necessary in order to understand the successive parts of this thesis that will recall these concepts. It will be important to be able to analyze the difficulties of coexistence between safety and security in CAN networks and to analyze how it is possible to make safe the Wireless Sensor Network technology.

2.1 Standards and definitions

2.1.1 Functional safety

The concept of functional safety is very important. It is applicable to Electrical, Electronic, Programmable Electronic (E/E/PE) parts of a safety-relevant system.

A definition is given in IEC 61508-4 [1]:

> *part of the overall safety relating to the EUC and the EUC control system that depends on the correct functioning of the E/E/PE safety-related systems and other risk reduction measures*

Safety is defined as:

> *freedom from unacceptable risk of physical injury or damage*

The dependability of a system is the key of the concepts of functional safety. According to [2] and [3], the objectives of functional safety are:

1. the quantification of every potential risks in terms of severity, probability, controllability for every system functionality;
2. the identification of safety mitigation measures which can reduce those risks to tolerable levels, reducing its negative impact in case of failure;
3. the validation of the functional safety measures implemented by the system.

Therefore, the safety properties change if the system context changes. The safety properties that change include also those mechanisms, checks and attributes designed to mitigate the risks. That is because the risks are associated with the system.

A safe system should be able to reveal hazardous conditions and, in this case, command a safe-state of the machine. A widespread safe state is the de-energized state, in order to prevent further risks or damages.

2.1.2 IEC61508

An A-type standard is a standard that covers only basic concepts, establishing common design principles and a common terminology. It is also very generic and application independent. The IEC61508 [1] is one of this kind of standards and it is the root of every other standard concerning functional safety of E/E/PE systems. In fact, as A-type standard, the principles and definition that it introduces are then recalled and specialized by application-specific standard. A couple of examples of standards that specialize the IEC61508 principles are the ISO 26262 for automotive, ISO 25119 for agricultural and forestry machines and ISO 15998 for heavy-duty machines.

Standards covers the whole life cycle of an E/E/PE system, including the management of design process, the operation and the system maintenance throughout its while life-cycle, from concepts to decommissioning. One of the most important principles defined by the standards are those related to the requirements related to system development.

The purpose of defining every phase of the life of a system is to be able to manage in a planned and methodical way every activity related to function safety. A well-defined set of inputs and outputs describes each phase. This accurate description enables a process of verification in which, at the end of each phase, it is performed a check to confirm that has been produced every required output as planned. This ability to check the respect of the specifications is one of the foundations of functional safety.

Such a structured approach will minimize the number of systematic faults the risks to be introduced in safety related systems.

The standard is divided in 7 parts:

1 General system safety requirements, documentation and safety assessment;
2 Specific requirements for E/E/PE safety-related systems, system design requirements;
3 Additional requirements for E/E/PE safety-related systems, software requirements;
4 Definitions and abbreviations;
5 Guidelines and examples for determining safety integrity levels;
6 Guidelines on the application of parts 2 and 3, calculations, modelling and analysis;
7 Techniques and measures to be used to control and avoid faults

IEC 61508 defines a performance index for safety functions named Safety Integrity Level (SIL). There are four levels, from SIL1 (less demanding) to SIL4 (more demanding), and for each one the standard details the requirements necessary to achieve it. A higher level of safety integrity has more rigorous requirements in order to reduce the residual probability of dangerous failure. Usually an E/E/PE safety-related system implements more safety functions that can require different SILs. In this situation, the entire system shall have to accomplish to the higher resultant SIL required, unless it can be proved a sufficient independence between the implementation of safety functions. Moreover, in case that a single E/E/PE system is capable of providing all the required safety functions, but the required safety integrity is less than that specified for SIL1, then IEC 61508 does not apply.

2.1.3 ISO13849

The ISO 13849 is a type B standard and goes to substitute the EN 954-1. It is the central safety standard for the design of machines systems. The ISO13849-1:2008 has been published as harmonized norm according to the machines directive 2006/42/CE. For this reason it is applicable the presumption of conformity.

At present day this standard is in re-elaboration phase.

The ISO13849-1, differently from EN954-1 that was based on a deterministic approach, is based on a probabilistic one for the evaluation of command systems linked to safety.

The standards regards, in addition to electric, electronic and electronic programmable systems, also other commanding technologies, such as fluid power.

The performance categories defined by EN954-1 have been maintained, but the relevant characteristics from a safety point of view are evaluated also from a quantitative point of view by mean of static calculation procedures. The categories are represented by the Performance Level, described through the following parameters:

- Hardware category;
- Mean time to failure;
- Diagnostic coverage;
- Common cause failure;

For the validation the ISO13849-2:2012 must be used. It is intended, with the word validation, a control verification that includes analysis and testing of safety functions and of the categories of the parts of the system linked to safety.

In order to realize parts of systems that are safety relevant an iterative process is defined, divided in different phases:

- Phase 1: definition of the requirements of safety functions. This is the most important phase. This definition is followed by the determination of the characteristics requested for the safety functions.
- Phase 2: determination of the required Performance Level (PL). The more the risk is high the higher are the requirements of the control system. The levels are divided from “a” to “e”, with “a” as the lowest and “e” as the highest. On the basis of some parameters it is determined the requested performance level PLr. The parameters are: the severity of the injury (S1, S2), the frequency and/or duration of exposition to the risk (F1, F2) and the possibility to avoid the risk (P1, P2);
- Phase 3: design and technical realization of the safety functions.
- Phase 4: determination of the realized performance level and quantitative evaluation. In order to determine the performance level realized the safety function is divided in three part: sensor, logic, actuator. Each part of this system gives its own contribute to the safety function.
- Phase 5: verification. During this phase it verified that the reached PL is greater or equal to the PLr.

- Phase 6: validation. This phase is important in order to avoid systematic failures.

2.1.4 ISO15998

The ISO 15998 [4] is a C-type norm, for earth-moving machinery. It embraces the electronic control units of the machine control systems (MCS) that performs safety functions. It gives guidelines on how to perform a risk assessment and how to identify the required SIL according to IEC 61508. It also defines a set of parameters that has to be considered in the functional safety analysis:

- Climatic conditions (i.e. temperature, humidity)
- Mechanical condition (i.e. vibration, shock)
- Corrosion conditions (i.e. salt spray, gas pollution)
- Electromagnetic compatibility
- Power source voltage fluctuation.

Table 1. Efficiency of measures in case of possible transmission errors, redrawn from ISO 15998:2008, Annex D, Table D.1

	Running Number	Time stamp	Time expiration	Reception acknowledge	Identification for sender and receiver	Data integrity assurance	Redundancy with cross-check	Different data integrity assurance for SR and non-SR messages
Repetition	X	X					X	
Loss	X			X			X	
Insertion	X			X	X		X	
Incorrect Sequence	X	X					X	
Message Falsification				X		X	X	
Retardation		X	X					
Coupling of SR and non-SR information				X	X			X

ISO 15998 also analyze some examples of hazard analysis of some safety functions (for example braking, propelling, steering and operating). If the MCS has a SIL greater or equal to one, the manufacturer must document the mechanisms improved in order to detect and tolerate a system fault. If it is detected a malfunction or a failure on a safety-critical MCS a safe state has to be achieved, providing reduced system performance or an alternate function. A very useful feature of ISO 15998 is that it considers the failures related to communication busses involved in a safety-related function, and it defines a model to transmit safety related messages. This model presents a list of errors that can cause a dangerous loss of a safety function, and, for each of them, it is defined a set of countermeasures that allow to detect such errors and enter the safe state. Table 1 gives a summary of this model.

2.1.5 ISO25119

ISO 25119 [5] is a C-type standard, specific for agricultural and forestry machinery. Because of the fact that it is a C-type standard, it only applies to specific machine categories, and it either refers to general standards like IEC61508 or define different requirements that prevail over A-type standards.

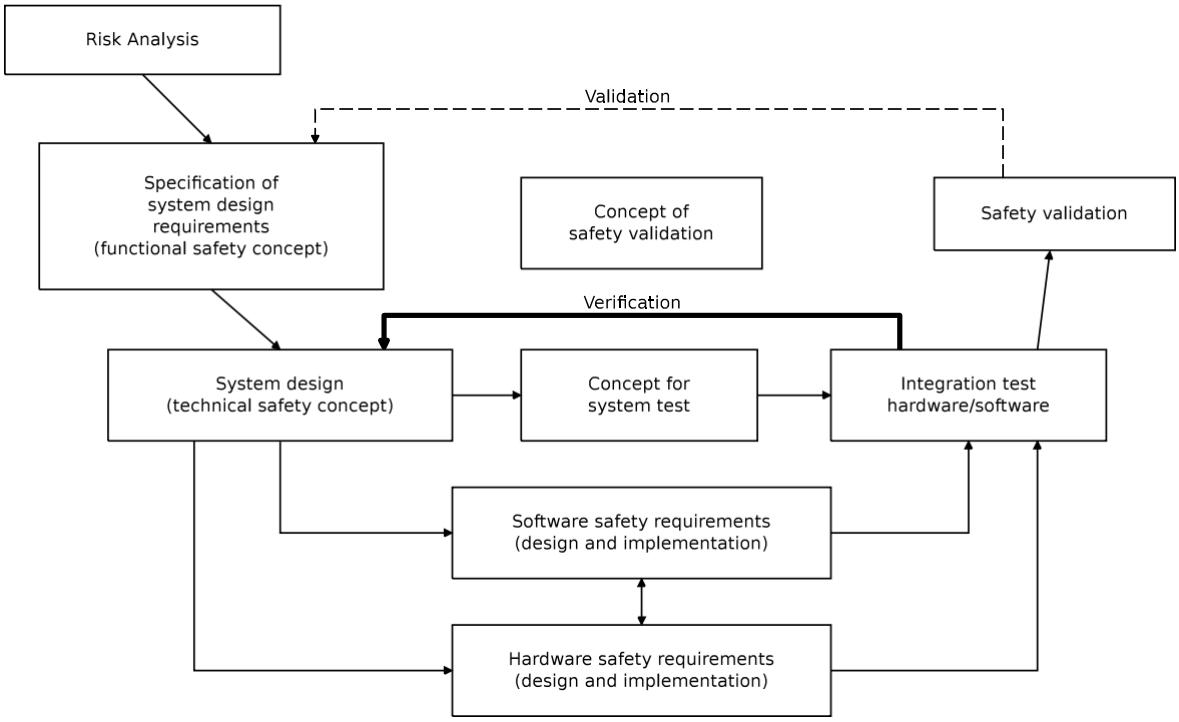

Figure 1. V-model for the overall system design, redrawn from ISO 25119

Four parts composes the standard and it defines the safety requirements for the safety-related parts of a control system. It defines a complete safety life cycle, describing the concept phase, the system design and implementation and the operations after the start of production. A requirement is the adoption of the V-model for the overall system design. Figure 1 shows it. There are different sub-phases going from the definition of specifications at system/module level to implementation and testing at system/module level. In order to measure the safety integrity level of a safety function the ISO 25119 defines the Agricultural Performance Level (AgPL), which can range from A (less demanding) to E (most demanding). Each AgPL is obtainable by combining four parameters:

- Mean time to failure of the hardware components (MTTF);
- Diagnostic coverage of the different blocks (DC);
- Common cause failure (CCF);
- Software required level (SRL);
- Hardware category.

In ISO25119 it is possible to find a thorough description of the requirements for each stage of the V-cycle model in order to obtain a certain SRL. More details are given on section 2.2.

2.2 Software quality

The software quality, from a normative point of view, consists of a set of measures that should reduce the probability of a dangerous systematic error, given by an incorrect implementation of the safety specifications. Must be considered the whole software life cycle, that runs parallel to the product life cycle, going from the specifications to the implementation and testing.

2.2.1 Software development cycle

A list of subsequent phases and a certain temporal execution composes the development cycle of software. Specification, implementation and testing are common phases of the development.

2.2.1.1 V-cycle model

The V-cycle software development model, defines specification, design, implementation and testing phase, allowing for some interaction between them. It defines very clearly how to handle a change. It can be necessary if, for example, to perform a change during the test phase because it is discovered a problem. Referring to Figure 2, a generic V-cycle represents the execution phases referred to the timing as its x-coordinate and with the level of abstraction as y-coordinate, from a high-level functional specification to a low-level implementation detail. It is allowed direct feedback only between phases at the same level of abstraction (same y-coordinate).

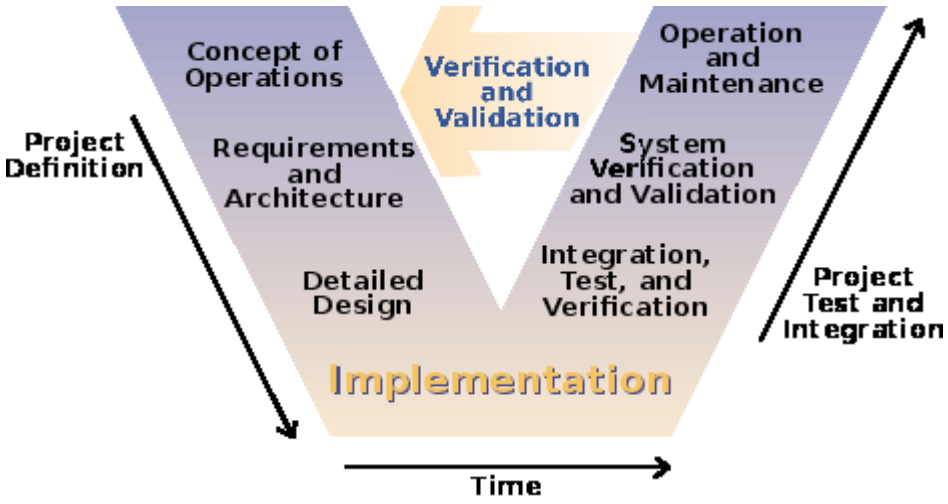

Figure 2. The V-Cycle software development cycle.

2.2.1.2 Waterfall development model

The Waterfall development model considers the requirements of the software as fixed and immutable. There is a strict temporal succession of specification, design, implementation, testing and maintenance phases. This is the oldest software development cycle. One of its greatest disadvantages is that, if a deficiency is discovered at later phases (for example during testing), the modification to the previous ones has a very high cost.

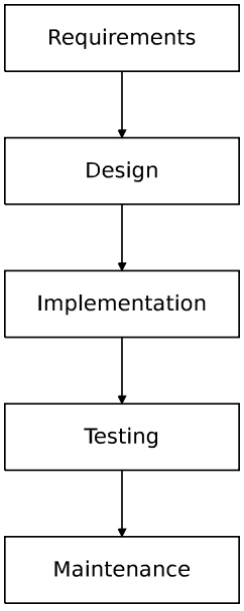

Figure 3. The Waterfall software development cycle.

2.2.2 Software specifications

The functional safety requirements of the software, according to functional safety standards, must be traced through the whole product life cycle. For this purpose exist many commercial tools as, for example, IBM Rational and Polarion. The way in which the software requirements should be expressed depends on the integrity level required. For lower safety integrity levels, a natural language description is considered sufficient; with the increasing of the required integrity level such requirements must be expressed either with semi-formal methods (flow charts, diagrams and figures) and formal methods (finite state automata, mathematical models, analysis and notation).

2.2.3 Software implementation

ISO 25119 contains a detailed description of implementation techniques for safety critical software, depending on the required SRL. These requirements will be analyzed one by one in 2.2.3.1.

From what will emerge, one of the most important is that a strongly typed language (see paragraph 2.2.3.1.2) has to be adopted; this means that the standard C language is not suitable for safety critical systems. A possible solution is to adopt a language subset that enforces a stronger typing than the standard. One well known subset is MISRA C [6], which is widely used in the automotive world. A problem that occurs in the adoption of such subset is that it is not trivial to enforce it, in particular during the reuse of legacy code. In fact, it forbids many commonly adopted constructs of the language, especially if the code was written using older revisions of the C standard like C90. For this reason, it is mandatory to use one or more automatic analysis tools to verify the compliance to MISRA C.

An alternative to the adoption of a language subset would be to directly use a language with native strong type checking such as Ada, but it is usually considered to have higher adoption barriers, and for this reason it is not widely used.

The last solution is to use higher level languages and a model-based approach (e.g. using Mathworks Simulink or Design Verifier), where the application logic is described in a way much more abstracted from the hardware and much closer to a mathematical model. Even

in this case, it is useful to use some automatic tool or language subset to avoid common errors.

2.2.3.1 Language rules defined by the standard

The design and implementation of software modules has to be divided in three phases. In the first one it is necessary to specify in detail the behavior of the safety-related software modules that are prescribed by the software architecture. The second one is to generate a readable, testable and maintainable source (code, model, etc.) suitable to be translated into object code. The third one is to verify that the software architecture has been fully and correctly implemented.

To design and develop software the following prescriptions should be followed:

- The programming language must be suitable and strongly typed. It should also be used a language subset.
- The tools and translators must have increased confidence from use
- Informal design methods have to be used.
- The programming must be structured.
- Library of trusted and/or verified software modules and components should be adopted.
- To verify design and coding an inspection and a walk through of design and/or source code must be done

In the following lines the concepts introduced in the list above are going to be clarified.

2.2.3.1.1 Suitable programming language

The language shall be fully and unambiguously defined. The language shall be user- or problem-oriented rather than processor/platform machine-orientated. Widely used languages or their subsets are preferred to special-purpose languages.

In addition to the already referenced features, the language shall provide for

- block structure,
- translation time checking, and

- run-time type and array bounds checking.

The language shall encourage:

- the use of small and manageable software modules,
- restriction of access to data in specific software modules,
- definition of variable sub-ranges, and
- any other type of error-limiting constructs.
- If safe operation of the system is dependent upon real-time constraints, then the language shall also provide for exception/interrupt handling. It is desirable that the language is supported by a suitable translator, appropriate libraries of pre-existing software modules, a debugger, and tools for both version control and development.

The following features which make verification difficult shall be avoided:

- unconditional jumps, excluding subroutine calls;
- recursion (both direct recursion, where a function calls itself and mutual recursion where a function A calls a function B that calls the function A and so on);
- pointers, heaps, or any type of dynamic variables or objects;
- handling of interrupts at source code level;
- multiple entries or exits of loops, blocks or subprograms;
- implicit variable initialization or declaration;
- variant records (record types that have some fields that are the same for all variables of that type (fixed part) and some fields that may be different (variant part)) and equivalence;
- procedural parameters (a parameter of a procedure that is itself a procedure).

Low-level languages, in particular assembly languages, present problems due to their processor/platform machine-orientated nature. A desirable language property is that the design and use result in programs whose execution is predictable. Given a suitably defined programming language, there is a subset which ensures that program execution is predictable. This subset cannot (in general) be statically determined, although many static constraints can assist in ensuring predictable execution. This would typically require a

demonstration that array indices are within bounds, and that numeric overflow cannot arise, etc.

2.2.3.1.2 Strongly typed programming language

The aim of using a strongly typed language is that the probability of faults shall be reduced. When a strongly typed programming language is compiled or statically analyzed, many checks need to be made on how variable types are used (for example, in procedure calls and external data access). Compilation shall fail and produce an error message for any usage that does not conform to predefined rules.

2.2.3.1.3 Language subset

The aim of using a language subset is to reduce the probability of introducing programming faults and increase the probability of detecting any remaining faults. The programming language shall be examined to determine programming constructs which are either error-prone or difficult to analyze (e.g. using static analysis methods). These programming constructs shall then be excluded and a language subset defined. Also, it shall be documented as to why the constructs used in the language subset are safe.

2.2.3.1.4 Increased confidence from use for tools and translators

In order to avoid any difficulties due to translator failures which can arise during development, verification and maintenance of software shall be applied tools and translators which are proved in use. This means that a translator shall be used where there has been no evidence of improper performance over a substantial number of prior projects. Translators without operating experience or with any serious known faults shall be avoided unless there is some other assurance of correct performance. If the translator has shown small deficiencies, the related language constructs are noted down, and carefully avoided during a safety-related project.

2.2.3.1.5 Structured programming

The aim of using the structured programming is that the program will be practical to analyze without being executed. To minimize structural complexity the following rules should be applied:

- Divide the program into appropriately small software modules, ensuring they are decoupled as far as possible and all interactions are explicit.
- Compose the software module control flow using structured constructs, i.e. sequences, iterations and selection.
- Keep the number of possible paths through a software module small, and the relation between the input and output parameters as simple as possible.
- Avoid complicated branching. In particular, avoid unconditional jumps (go-to) in higher-level languages.
- Where possible, relate loop constraints and branching to input parameters.
- Avoid using complex calculations as the basis of branching and loop decisions. Features of the programming language that encourage the above approach shall be used in preference to other features that are (allegedly) more efficient, except where efficiency takes absolute priority.

2.2.3.1.6 Library of trusted/verified software modules and components

The aim is to avoid the need for extensive revalidation or redesign for each new application. It is allowed the reuse of designs that have not been formally or rigorously validated but for which considerable operational history is available.

In order to be well-designed and structured, electric/electronic/programmable electronic systems shall be composed of hardware components, software components and software modules which are clearly distinct, and which interact with one another in clearly specified ways. E/E/PE electronic systems designed for differing applications can contain a number of software modules or components which are the same or very similar. Building up a library of such generally applicable software modules allows many of the resources necessary for validating the designs to be shared by more than one application. Furthermore, the use of such software modules in multiple applications provides empirical evidence of successful operational use. This empirical evidence justifiably enhances the trust which users are likely to have in the software modules.

2.2.4 Software testing

The aim of software testing, in the field of safety critical systems, is to verify that the implementation corresponds to the software safety specifications. As described in Figure 2 for the V-cycle model, testing has to be performed at the module level, at the integration level and at system level. The testing method to apply depends on how the specifications

are redacted, considering that the purpose of testing is to provide an appropriate guarantee that the implementation corresponds to specifications.

There are several testing methods. The most used is the “test cases” one that consists in defining test cases or unit tests where the program is executed to perform a very specific function. The result is then compared with an expected one. This kind of testing is also used to perform regression tests: after a modification the test cases are repeated in order to verify that no functionality has been lost. Other test methodologies are Model-in-the-Loop (MIL), Hardware-in-the-Loop (HIL), Software-in-the-loop (SIL).

2.3 Hardware architecture for safety-critical systems

This section will present some commonly used hardware architectures for safety critical systems. The main reference adopted is ISO25119, but the concept expressed by other standards is very similar.

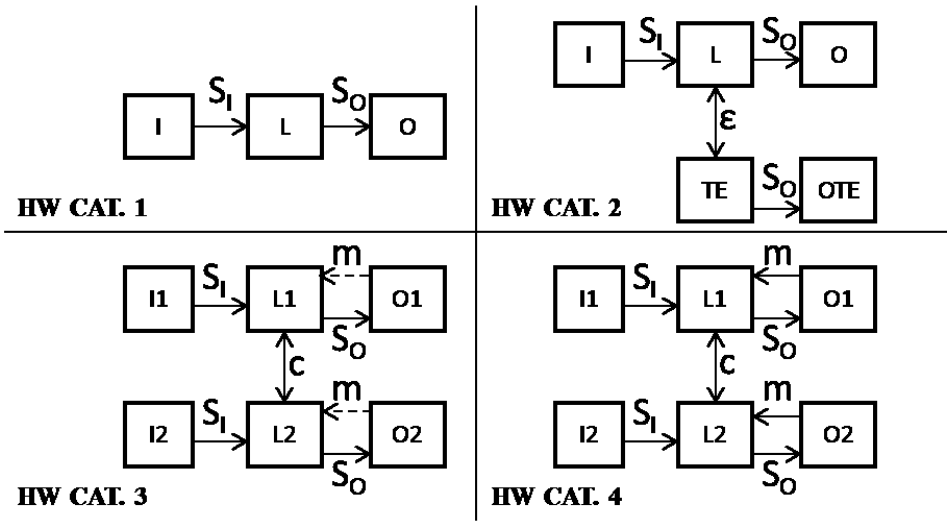

Figure 4. Hardware categories according to ISO 25119.

Figure 4 shows the four possible hardware categories. In each one there are some common blocks and links:

- I input block
- L logic block, e.g. a microcontroller, often called Main Micro Controller (MMC)
- output block

- TE test equipment, e.g. a component that tests the MMC for errors, often called Safety Micro Controller (SMC)
- OTE output of test equipment
- SI interconnection with input signal
- SO interconnection with output signal
- m monitoring
- c cross-monitoring

Category 1 has a single and unidirectional chain from input (I) to output (O) passing through the logic (L), which generally corresponds to a microcontroller. This is referred to as main microcontroller (MMC). It does not require any diagnosis on input or output by the logic. If a similar hardware structure is used in a safety critical application, it is required to use only well tried devices and components.

Category 2 adds a test equipment (TE) that usually corresponds to a second microcontroller, which has the role to monitor the logic L. In case of fault on the logic, the test equipment must be able to override MMC's commands and lead the system to a safe state. The second microcontroller is often referred as the safety microcontroller (SMC). In category 2 the logic must be able to diagnose faults on input and on outputs, increasing the overall Performance Level (PL) from the functional safety point of view.

Category3 has a completely redundant chain, composed by the input blocks I1 and I2, the logic blocks L1 and L2, and the output blocks O1 and O2. The logic blocks cross-monitor each other. The system will not suffer any loss of safety functions in case of faults. The faulty channel (I1-L1-O1 or I2-L2-O2) is turned off by the other one. This kind of system is called fault-tolerant.

Category 4 has the same architecture of the third one but with a higher diagnostic coverage requested on the output (continuous instead of discontinuous tests).

2.4 Diagnostic coverage

One of the most important aspects introduced by safety standards is the diagnostic coverage. Even if, according to the standards, the components adopted must be reliable and

well tried, in order to have an acceptable mean time to failure, that is the time before in one of the components a fault is verified, the electronic system, particularly starting from hardware category 2 has to be able to reveal a fault of a components.

The capability to diagnose a fault is fundamental. In fact, a silent fault, that is a fault not detected, may not lead to a direct consequence, but it can lead to a double fault on the system in the near future that may have serious consequences. An example can be performed to understand in a better way the concept. In an hardware category 4, the most reliable one, the decision chain (input-logic-output) is completely redounded because the system can't tolerate the loss of functionality in case of single fault (e.g. the braking system's control unit). If the system is not able to diagnose itself it is possible that the first microcontroller goes in a faulty condition. The operator does not see any evidence of fault because the control chain is redounded, then the second chain continues to supply the full functionality of the system. After a certain time also the second microcontroller goes in a faulty condition. There will be a complete loss of functionality, that is not tolerable in this kind of systems, leading to a dangerous condition, that can cause serious injuries to the operator or to bystanders (in the example of the braking system the machine is no more able to stop itself). The example of fault on both microcontrollers (in successive moments) is not unrealistic, in fact, as it is possible to see in [7], the microcontrollers are between the components with the lowest MTTF, that mean that they are the components in which is easier to have a fault.

The ISO25119 part 2, annex C proposes two ways of estimating the diagnostic coverage of a system. In the first one, for each component of the control chain that realizes the safety function (or safety goal), it is necessary to determine the diagnostic coverage (DC) according to the tables reported in the same annex. An example of such a table is reported in Table 2, describing the DC reached, on the base of which detection measure has been applied.

Table 2. Diagnostic coverage estimation for electrical subsystems without micro-controllers, redrawn from ISO 25119:2010, Part 2, Annex C, Table C.1

Technique	Diagnostic coverage	Example
Failure detection by on-line monitoring	Medium	Indication lamp for turn-signal
Monitoring of relay contacts	Medium	Indication lamp for mechanical front wheel drive (MFWD)
Comparator	Medium	Indication lamp for charging system
Positive-activated switch	High	Switch with mechanical interlock

The second method performs a calculation, according to diagnostic coverage definition. Diagnostic coverage is defined as the fraction of the probability of detected dangerous failures λ_{dd} and the probability of total dangerous failures λ_d:

$$DC = \frac{\sum \lambda_{dd}}{\sum \lambda_d} \quad (1)$$

An aspect that has to be highlighted is that this definition does not consider any fault that can happen, but considers only the dangerous faults, that are those faults that can lead to a dangerous condition. This concept is clarified by Figure 5.

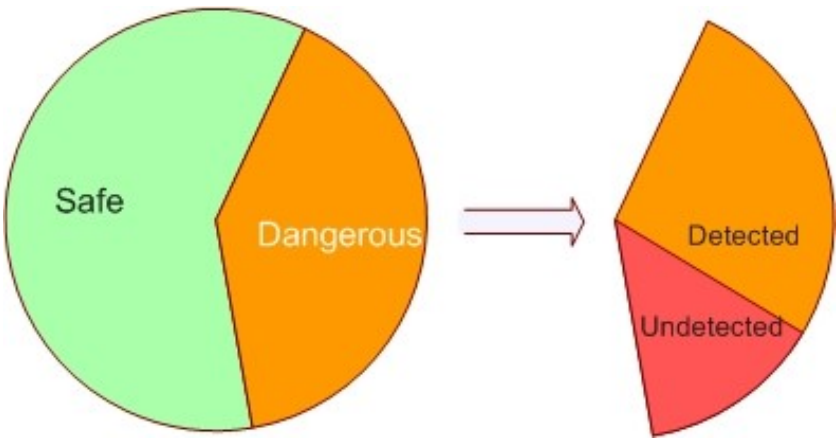

Figure 5. Diagnostic coverage and dangerous faults

2.5 Safe communication

The safe communication can be seen as a consequence of the diagnostic coverage. In fact it is strictly correlated to the ability to diagnose faults on the communication channel.

Both a recent standard, like ISO 25119 in Annex B, and a previous one, like ISO 15998 in Annex D consider the communication errors. They require the use, in communication systems, of some techniques in order to detect communication errors that can lead to a dangerous condition of the system. In ISO 25119, the focus is on the partitioning of different safety functions over a communication network. It does not only consider communication errors, but in Annex B, it considers the requirements for software partitioning in a single ECUs, in order to have software functions of different criticality in the same ECU. To consider guaranteed the separations between different execution domains there must be a certain separation in memory and time domain as well as inter-task communication. In ISO15998, the focus is on the typical transmission errors, as described in section 2.1.4.

Chapter 3. In-vehicle networks

This chapter will describe the evolution of today's vehicular networks that has seen the introduction of wireless networks flanked to wired ones. This coexistence introduced new issues, particularly in wired networks that are mainly used for safety applications. The first contribution of this thesis analyzes the aspects that derives from the necessity of satisfaction both of safety and security requirements, according to what reported in [8]. This analysis leads to the conclusion that CAN capabilities are not enough to satisfy both safety and security requirements. As a consequence, vehicular networks, in order to satisfy these requirements, will have to migrate to a most performing technology.

Ethernet field bus has been designated as CAN successor both in automotive and agricultural applications by many international councils because of its high throughput and of the possibility to reuse standard technologies like the TCP/IP stack for a better integration with Internet of Things paradigm and more recent technologies as Ethernet AVB. This field bus, thanks to its better performances, can be able to provide both safety and security. For this reason, the second contribution of this thesis will be presented, describing how it has been realized a gateway to interface ISOBUS networks working on a CAN bus with TCP/IP networks working on Ethernet [9].

3.1 Overview of wireless networks

The wireless technology appeared as a key component, in the last years, for traditional and new applications in agricultural vehicles. Some of the most common applications are:

- Human-to-machine interaction. It is an emerging application. Some implements have short-range connectivity like Bluetooth and Wi-Fi. Thanks to this connectivity the operator can interact to the machine using a smartphone or another mobile device.
- On-field and on-machine sensor networks. It has many applications in precision agriculture: load/weight sensing, Tire-Pressure Monitoring Systems (TPMS), field

monitoring. It can also be used to cooperate with drones, which, by flying over the field, can collect the data transmitted by short-range low-power sensors.

- Machine-to-machine communication. For M2M (machine-to-machine) communications Wi-Fi (IEEE 802.11) connections are used. An example, which has been often used during media campaign, is the cooperation between a harvester machine and a storage truck in which they are connected through a mesh network, exchanging data, such as position and other operational parameters, in real-time. In this way, the truck is able to follow the harvester machine at the right speed and distance.
- Fleet-management. This application is, usually, based on 3G and 4G technology. It allows download of the field map, upload of operational data and real-time positioning of the machine fleet. Data format is standardized as ISO-XML (also called AGRO-XML), but the protocols used for the remote communication are usually proprietary.

In-Vehicle networks are currently addressing two main challenges: the need of a sustainable architecture evolution and the improvement of security.

Regarding the first challenge, it derives from the fact that new applications often require a drastic change in the network architecture. This is a very expensive process, so it is necessary to find an architecture that can survive to the evolution required, in part or completely.

The security characteristics of the network are becoming always more important in vehicular networks. In the past they were closed networks, so they were intrinsically safe. At present day, with the growing adoption of wireless connections they are becoming more vulnerable to a malicious operation.

The following sections will threat in a more detailed way these two aspects because the wireless is an enabling technology for the interesting applications currently developed, though it is not the only one.

3.2 Overview of wired networks

The Controller Area Network (CAN) has been the main communication technology adopted in vehicular network, since their introduction, in order to connect two or more Electronic Control Units (ECUs). Its origin dates back to years '80. It has been later standardized in 1992 with version CAN 2.0, becoming an international standard, namely ISO 11898, that has been revision the last time in 2015 [10]. The CAN bus uses a base-band communication with bit-stuffing, allowing a maximum of 8 byte of payload and an identifier of the message which can be 11 or 29 bit long. The channel access is a Carrier Sense Multiple Access with Bit Arbitration (CSMA-BA), performed thanks to the identifier that represents the priority of the CAN message. This characteristic permit to the CAN bus to perform an arbitration between colliding messages using its nature. Standard raw bit rates are 250 kbit/s, 500 kbit/s and 1000 kbit/s.

At present day CAN bus still represents the main and most widely used vehicular networks, thanks to the very low cost and to its reliability. However in small subnetworks have been used many other technologies. One of them is Local Interconnect Network (LIN), used for sensors and actuators. A short term alternative to CAN bus is the Flexray bus, that uses a TDMA access method instead of the classic CSMA-BA and allows packets up to 256-byte at 10Mbit/s. Flexray has been introduced to supply the real-time requirements of many new applications, like ABS and x-by-wire.

The interest is growing for high-speed vehicular networks, which key-enabler are multimedia applications like audio and video streaming. In this field on of the most interesting and prominent technologies is Ethernet, enabled by the IEEE 802.3bw BroadR-Reach automotive-compliant physical layer.

In agricultural field, is used a protocol stack named ISOBUS. This is a standard protocol, adopted by every vehicle producer, described by ISO 11783 [11] and based on the CAN bus.

Previous research work at CNR-IMAMOTER was aimed at developing an ISOBUS 2.0, also called high-speed ISOBUS, based on Ethernet and suitable for control applications. This requires at least soft real-time capabilities, which is now available to some degree on

commercial components and belonging to the Audio-Video Bridging standards group (AVB, e.g. IEEE 802.1AS and 802.1BA), more recently generalized as Time Sensitive Networking (TSN e.g. IEEE 802.1Qat and 802.1Qcc for stream reservation).

Further research has been performed in order to permit the co-existence of Ethernet UDP/IP and CAN ISOBUS, creating a gateway that is able to interface with the addressing systems of both networks and is able to re-format the frames from ISOBUS to UDP/IP and vice versa. Paragraph 3.4 presents this work.

3.3 Security aspects

Agricultural and heavy-duty machines, such as cars are becoming always more connected to the Internet, transforming themselves from isolated entities to devices, just like a smartphone, a computer or a tablet. There are two sides of the coin: this evolution introduces new important functionalities that make the vehicles more attractive, but it also exposes them to cyber security threats. For example, malicious users may remotely find and exploit some vulnerability to remotely access the car and perform malicious actions. A famous attack is the one performed on a Jeep Cherokee, performed by Valasek and Miller [12], published in the summer of 2015, in which the authors has been able to control remotely a Jeep Cherokee (by overriding the commands for braking and steering systems) through the internet connection, exploiting various vulnerability in the infotainment system. As a consequence Fiat Chrysler had to provide a software update [13] to be installed through the USB port of their vehicle's dashboard.

In literature it is, now, possible to find many examples of attacks, with an analysis of their feasibility. A couple of examples are those reported in [14] and in [15]. In the first one the authors uses the local CAN access to gain complete control of the car, while in the second one the same authors analyze the possibility of a remote attack, aiming, also in this case, to obtain the full control of the vehicle.

The examples reported are focused on automotive field, but they are relevant also for heavy-duty vehicles, considering that the increased connectivity is permeating also this field, exposing, in this way, also this field to similar scenarios, although with different consequences due to the different nature of vehicles.

In consideration of the risks just analyzed, the emerging opinion is that past communication protocols and new deployed ones should be should be adapted to cover some security requirements and to avoid, or at least mitigate, the security threats.

Regarding the new protocols they should be designed having in mind the attack surfaces [16] in order to provide the right security solutions by the principles of Security by Design.

This thesis is mainly focused on agricultural and heavy-duty machines, equipped with one or more CAN-based networks, used by all the electronic control units (ECUs) to communicate. In the last years has been considered Ethernet network as a long-term replacement for the vehicle networks in agricultural and heavy-duty machines. This is due to the increasing demand for bandwidth from the applications implemented in the ECUs. Also an increasing need for secure communication is emerging, caused by the growing direct or indirect connection of in-vehicle networks with other networks, in particular with the Internet. This is leading to new technological and business possibilities but, This increase in connectivity opens new technological and business possibilities but, as reported in [17], it creates a situation where a vulnerability can have disastrous consequences, both in terms of security and safety.

In this section, according to what has been exposed in [8], is exposed a solution to turn CAN messages into a Security by Design format by having “on board” authentication, integrity and confidentiality properties. In particular, this solution is built through a Message Authentication Code (MAC) that is then encrypted with an additional key. In this way, the created message guarantees authentication and integrity through the MAC and confidentiality with the additional encryption. This defense strategy is studied and applied against a model of attacker that applies both an Honest-But-Curious (HBC) and Fully Malicious attack strategy. Furthermore, this solution is evaluated from a safety point of view, in particular regarding the residual probability of error. This is necessary since the outcome of the security MAC i.e. accept or reject a particular message is a form of error detection which could reveal also transmission errors (e.g. caused by noise), and the message containing the MAC could bring safety-critical information. For example, this scheme could be applied to SAE J1939 Torque/Speed Control 1 message, which embeds a

4-bit checksum within the 8-byte CAN payload. The residual probability of error is first evaluated using an ideal block cipher model, and then simulation results are presented for a specific implementation choice. It is shown that averaging over the secret keys and over the possible messages, the value of P_{re} depends only on the length of the integrity tag used to decide whether the message is valid or not, and this is independent from how the integrity tag is generated. On the other hand, it is shown how the worst-case combination of key and message reduces massively the ability of this scheme to detect transmission errors, assuming a simplified channel model. The worst-case P_{re} is then simulated, with considerations on the difficulty of finding the worst-case combination of key and message. The main contributions of this section is to analyze a message protection protocol from both a security and safety point of view, and highlight the trade-offs that result from the analysis.

An alternative to CAN bus, that is emerging as its successor is Ethernet network. In this kind of network only the computational resources of the ECUs connected give the limits. As a consequence of this consideration, it is described the implementation of a gateway ISOBUS-UDP/IP in order to allow the transition between those two buses in prevision of a coexistence period, as described in [9].

3.3.1 General considerations

It is necessary to consider the requirements of typical periodic real-time traffic. These requirements generates the following constraints that a security measure or algorithm should fulfill:

1 Minimum overhead on the data packets. The overhead bytes added to the message must be in a reduced number if compared to the network packet constraints, in order to be transmitted without fragmentation. This is necessary because in many real-time messages, fragmentation is often not tolerable.

2 Limited computational complexity. The algorithm must be executed in a time that is negligible compared with the time repetition of the messages. This characteristic is able to satisfy real-time requirements but is also useful to avoid timing attacks.

The expectation on computational power available on a typical ECU is of a growth, but network packet constraints, generally are fixed and depending on the network technology used. The networks, usually, imposes constraints also on the possible topology and on the overall achievable security level.

3.3.2 Modeling the attacker

Nowadays, the evolution of communication technology into heavy-duty machines can introduce new vulnerabilities that affect the proper machine functionality. Section 3.3.3 lists some security threats that attackers could exploit being located far away from the vehicle, or in proximity, using a physical connection with the vehicle, e.g., a USB-port. The choice to consider versatile attackers is enforced by the fact that vehicles are not isolated entities anymore, but they are connected to the Internet through, for instance, 3G/4G or Wi-Fi connectivity. Thus, by exploiting wrong security configuration or security flaws in the protocols, attackers may easily gain access to the machine without particular effort. In fact, if we consider that a generic communication protocol, developed in the past and used for decades, was not thought to be secure by design, for example without an authentication system, once it is moved into a different domain, e.g., the Internet, attackers can access the vehicle's.

As exposed, to model the attackers, it must be taken into account that they can have local or remote access to the vehicle to compromise the CAN bus network by forging or altering messages that are verified as valid by the recipients. For instance, attackers may be able to exploit a bug of the authentication system of the vehicle and remotely access the CAN bus infrastructure using a classic IP connection. Once inside the vehicle, they may be able to forge valid messages or even to alter their contents.

The defense strategy proposed is to avoid that one or more attackers can forge valid messages, keeping enabled confidentiality of proper messages generated. Moreover, part of the solution is to identify messages that were altered, losing their integrity. Practically the defense is based on three properties:

- Authentication: a recipient should be able to verify whether the message is sent by a legitimate sender.

- Integrity: a recipient should be able to verify whether the message has been altered during its transmission.
- Confidentiality: it guarantees that the content of the message is not revealed to an illegitimate entity, as it can happen with the Man-in-the-Middle (MITM) attack.

A first kind of classification can be performed between the possible attacks. They can be:

- Local;
- Remote;

An attack of the first category can be performed only when the attacker is placed near the target, while, for the second category the Internet connection is used to access the vehicle. Usually a remote attack requires considerable effort to be exploited if compared to a local one. The reason of this difference is that a remote attack can be performed from a larger number of users than those which can perform a local one, so, when a remote link is provided, it is used more attention in security measurements to be adopted during the design of the network.

Another aspect to consider in order to define the attacker is the following classification:

- Honest-but-Curious (HBC): also known as passive attack; an attacker may exploit the information legitimately gleaned by capturing messages exchanged over the CAN bus infrastructure, but he/she will not perform any malicious activity to harvest it.
- Fully Malicious (FM): also known as active attack; an attacker is able to forge or alter messages that are considered valid, after a verification step, by the recipient. The attacker strategy is to succeed in at least one of the following attacks:
    - Impersonation attack: the attacker is able to forge or alter messages that are considered valid by the recipient;
    - Replay attack: the attacker is able to re-use valid messages with a malicious or fraudulent aim;
    - Sniffing attack: the attacker is able to read the content of any messages exchanged through the CAN bus infrastructure.

The least aspect to consider, after the division from local to remote and from honest and malicious is performed (as anticipated in the description of the fully malicious attacker) on the kind of attack, depending on its goal. Table 3 shows the defense strategies applied in response to each attack.

Table 3. Summary of attack and defense strategies

Attack	Goal	Defense
Impersonation	Forging or altering messages that are valid by the recipient	Confidentiality Authentication Integrity
Replay	Re-use messages that are considered valid by the recipient	Authentication
Sniffing	Read content of messages	Confidentiality

Some practical examples are going to be analyzed.

The first one is a local sniffing attack, where a malicious user may be able to read the content of all messages exchanged through the CAN bus. Considering that the messages do not contain the confidentiality property, any message is vulnerable to this attack.

An attack that can be performed after the sniffing phase, continuing to consider a local attack, is the replay one. The bound to the sniffing attack is due to the fact that it uses messages captured that are retransmitted on the network for specific purposes. For example, an attacker may store the sniffed communications between the ECU that controls the brakes, and the ECU that controls the brakes' movements, to replay the actions without control of the driver. Once the attacker has captured the command to perform the given action, it can replay the same message in order to generate the same movement of the brakes.

Considering some examples of remote attacks, the first described is the Denial of Service (DoS) attack. Here, the attacker wants to make unavailable one or more services of the vehicle. This kind of attack is common on the traditional communication networks, and in particular it involves attacks on Web Servers and Web Services, to make them not reachable from end-users. Within heavy-duty machines an attacker, who performs the DoS attack, can make specific ECUs unavailable. If the target of the attack is an ECU that

performs a safety functions, as, for example, the one that controls the steering of the wheels, the DoS attack strongly affects the safety of the vehicle.

Another remote attack is the Black hole [18], which can be considered a special case of DoS attacks. In this case, the attacker remotely gets control of one or more ECUs to block and drop messages that transit through them.

3.3.3 Challenges

As described in section 3.3.2 there are security issues that may affect heavy-duty machines, in particular if these vehicles have a remote access, by being connected, for example, to the Internet. Therefore, vehicles cannot be considered anymore intrinsically protected from cyber-security attacks thanks to their isolation.

This considerations lead to the conclusion that in the field of heavy-duty vehicles has born the requirements of protection from the attacks listed. However, it is not always feasible by using solutions “off the shelves”, already adopted in IT sector, due to the current physical infrastructure adopted in these machines. To make an example, to improve confidentiality it can be adopted a cryptographic solution to encrypt messages. Moreover, in order to mitigate a DoS attack a good solution would be to use firewalls on the border of vehicle’s network, which purpose is to block message flooding that may make services unavailable. Another solution to mitigate the DoS attack is to use message validation techniques by sharing a secret used to evaluate the validity of messages. An attacker, in this way, must be able to forge a valid message to be accepted by the counterpart by guessing the secret. In this case, the probabilities are those of a random message attack.

The solutions that has just been proposed would help in improving the level of protection against security attacks, but they require an adequate network infrastructure and enough computational resources. The next paragraphs will analyze the current network infrastructure of heavy duty machines (CAN bus) and will try to propose solutions, applicable to such infrastructure, to reduce the risk of attacks. Then these solutions will be compared to those possible on an Ethernet network.

3.3.4 CAN: the present standard

The CAN bus is one of the most widespread vehicular networks and is used on automotive, as well as heavy-duty, agriculture vehicles. The network architecture of a CAN network is that of a physically shared bus, therefore everyone can listen the messages generated from other ECUs and can transmit messages received by every other ECU. This exposes to sniffing and replay attacks, as well as DoS and Black Hole attacks. A plain CAN network has a maximum message size of 8 bytes. Exists various Transport Protocols, as ISO-TP [19], SAEJ1939 Connection Mode [20] or Extended Transport Protocol [21] that allow to send larger messages. For this purpose they split a single block of data over different messages and use explicit acknowledge packets for reliability. For these reasons they are not typically suitable for real time traffic.

Typically a vehicle has more than one CAN subnetwork, usually divided accordingly to the purpose of the ECUs connected (powertrain, implement, diagnostic, etc.). Moreover a single ECU has more than one CAN interface and can be connected to more subnetworks. To mitigate the attacks exposed above CAN gateways could be employed but they are rarely used.

Different CAN subnetworks have different exposures:

- The powertrain is a private internal network;
- The implement (ISOBUS) is open to the connection of third party implements that often are out of the control of the manufacturer.

It is important to highlight that the use of multiple subnets is not in order to increase security, but it is just for performance reasons. A single network would not be sufficient to grant the throughput necessary to the applications.

3.3.4.1 Security: authentication

On CAN bus, authentication is performed only in order to enable certain functions, usually for maintenance purposes. One of the most adopted protocol for authentication is the Seed and Key, used for example in the CAN Calibration Protocol (CCP) [22] and in the Keyword 2000 protocol (KW2000) [23]. It works in two phases:

1 the client ECU that wants to authenticate on a server ECU requests a seed S, usually a random number. The client ECU then computes the key K, according to a certain function **f**, depending on S and optionally on a fixed key k_s shared between the client and the server;
2 the server ECU receives K from the client ECU, and verifies its correctness in order to decide if the functionality has to be enabled, usually by recalculating it with the same algorithm used by the client ECU.

The algorithm to be used in order to calculate the key from the seed is not specified by any standards and can depend on the application, on the manufacturer and on the functionality. These algorithms usually rely on security through obscurity (for example $K = f(S, ks)$ where f is a secret function, which sometimes is a simple addition or a bit-wise xor). It is worth noting that the main purpose of the Seed and key protocol is to reduce the chance of unauthorized operations. However, it is not difficult enough to discover the secret algorithm, by means of reverse engineering, for example for tuning or cloning of ECUs. This protocol, usually, is vulnerable to the replay attack, since for a given seed S_i there is only one possible key K_i. Although the manufacturer of an ECU can limit the number of Seed and key transactions per second to mitigate the possibility of a Replay attack, the number of S_i, K_i pairs that can be obtained in reasonable time is normally sufficient for reverse engineering the secret algorithm. Therefore, seed and key protocol is not secure enough for the security goals exposed in this thesis. In fact it is necessary that data are continuously authenticated and that the authentication is difficult to be violated. To authenticate every can message it is possible to use a Message Authentication Code (MAC). Considering that, as discussed in 3.3.4.4, the use of multiple messages is not compatible with real-time traffic, the authentication of a CAN message is problematic because its messages are limited to 8 bytes and an additional field is required to be added to the message.

There are three possible ways to insert the MAC:

- Reduce the length of data field, reserving a number of bits for a MAC.
- Allow the use of a DLC (which maximum value is 8) in the range 9-15.

- Use the existing 15 bits normally dedicated to the CRC field to store a 15-bit MAC.

The first solution has the disadvantage to reduce the number of bits reserved to the message, but it uses a standard CAN transceiver.

The last two solutions implies a limited MAC length and non-standard behavior, but allows the full use of the 64 bit data payload. A similar approach, with a modified CAN bus is presented in [24]. Moreover the last solution goes to sacrifice the integrity check in favor of the authentication.

Further details on how to improve MAC mechanism in CAN messages is described in paragraph 3.3.4.6.

In SAE J1939 [20] or ISOBUS [21], that are the most used protocols for heavy-duty machines, the majority of messages do not have room for a MAC, since all the 64 bits of CAN are used. In this kind of messages, proper authentication is considered not possible using the standard CAN, unless additional messages are allowed. Authentication schemes such as Leia [25] provide lightweight authentication and require the use of only one additional CAN message for each message to be authenticated. While this could be acceptable for highly critical messages, its use of the CAN ID is not compatible with ISOBUS. The importance of using the CAN ID here lies in the transmission of a counter related to the authentication protocol and to operation like synchronization between ECUs.

Other authentication mechanisms have been proposed for extensions of the classic CAN bus like Flexray and CAN-FD, see for example [26] for a comparison.

3.3.4.2 Security: integrity

CAN network improve message integrity through a CRC16 checksum, that has a dedicated field in the message. CRC16 is not suited for cryptographic purposes because it is a linear function and, then, it is easily invertible. Thanks to the nature of CAN it not easy to modify a message in transit without being detected, even if it is feasible, as demonstrated in [27]. However, a replay attack is easily applicable when messages are not authenticated since an attacker can inject packets in the network pretending to be anyone, even using an higher priority compared to the legitimate message. It is possible to use additional checksum or

hash algorithms, but, in some standard messages, there is no room for additional fields (e.g. SAEJ1939 TSC1 - Torque Speed Control 1). Then it is possible to increase the integrity protection on for some messages.

3.3.4.3 Security: confidentiality

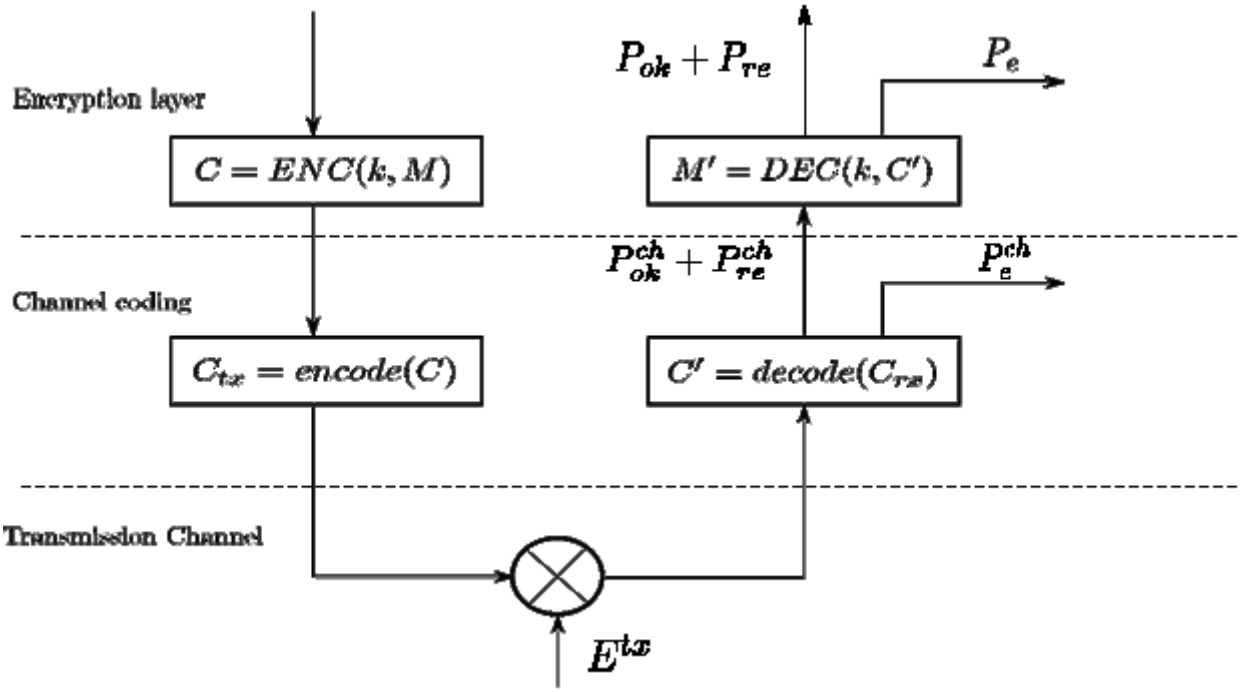

Figure 6. Message protection scheme

Encryption is the most widely used method to allow the achievement of confidentiality on a communication. An encryption scheme is represented in Figure 6. In general, a message μ with length μ_{size} bits is combined with an integrity tag $\tau = H(k_2, \mu)$ of length τ_{size}, where k_2 is the authentication key. The combined $\mu||\tau$ is then encrypted to obtain the cipher text $C = ENC(k_1, \mu||\tau)$, where k_1 is the encryption key; the cipher text is then transmitted on the CAN bus. The receiver receives C', decrypts it and checks if $\tau' = H(k_2, \mu')$, with $\mu'||\tau' = DEC(k_1, C')$, to decide if the message is valid.

Table 4. Recommended key length in bits for asymmetric-key and symmetric-key cryptography [28]. Additional references are available at http://www.keylength.com.

Security Level	Symmet	EC	RS
Very short term, weak attacker	6	12	81
Short-term (Standard level)	9	19	177

Middle-term	1	22	243
Long-term	1	25	324

Regarding the key length, as shown in Table 4, a 96 bit key is enough for short term security, a 112 bit for middle-term and a 128 bit for long term. In CAN, considering that real time traffic is present, it is enough a 64 bit key with a key refresh procedure.

There can be some variations in this scheme; for example the integrity code can be appended to μ to form the plaintext (also known as MAC-then-encrypt approach) with $C = ENC(k_1, \mu||\tau)$, or it can be excluded from encryption (encrypt-then-MAC approach) with $C = ENC(k_1, \mu)||\tau$. In both cases the MAC has to be computed using the original message μ. Sometimes the MAC-then-encrypt approach is considered less secure, for example see [29], but in this section the scheme has no padding and a fixed length of the message, so these considerations do not apply. Considering the CAN bus, the first approach is more practical since there exist encryption algorithms, for example symmetric block cipher, with 64-bit block size, equal to the maximum payload of a CAN message; examples of algorithms that can be adopted are DES and Blowfish. In this case there is no need for additional data to perform the encryption, and the cipher text is computed as $C = ENC(k_1, \mu||\tau)$. On the other hand, if the plaintext is different from the block size (like in the encrypt-then-MAC approach), alternatives must be considered. The first one is the stream cipher that, unfortunately, offers a weak protection against sniffing attacks. This weakness can be mitigated using a time-variant component added to the message (e.g. xor-ing the plaintext with a time-varying data called nonce).

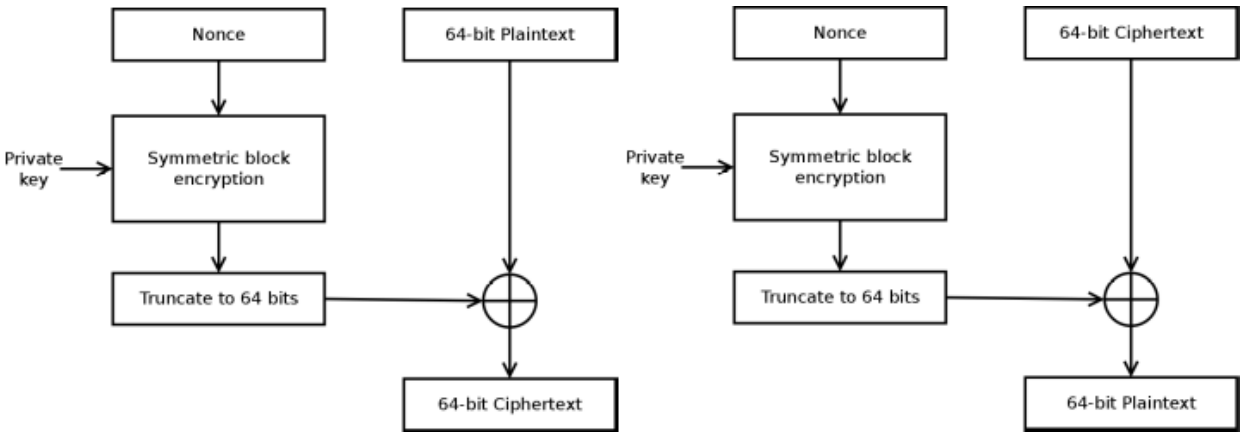

Figure 7. Encryption with CTR mode (left) and decryption with CTR mode (right).

A second alternative can be the symmetric block cypher in counter mode (CTR), shown in Figure 7 that would grant a higher security level through an increased key size. In this case the plaintext data are xor-ed with the result of a nonce encryption, and are not directly encrypted.

Either way, there needs to be additional information shared between the sender and the receiver, to perform the encryption; the cipher text is computed as $C = ENC(k_1, I, \mu)||\tau$ with I being the shared information.

To implement the shared information it is possible to use a distributed running counter as the one of [30], but the key exchange procedure will have to use a transport protocol. For a fast key exchange and computation, it is essential to use short key sizes. The best algorithm to exchange the key is an elliptic curve algorithm with a key size that will be a compromise between security and overhead.

Another variant is to avoid the use of two different shared keys and define the integrity code as $\tau = H(\mu)$, where $H(\cdot)$ is a hash function like SHA1 or an error-detection code of the CRC family.

3.3.4.4 Safety: fragmentation issues

It is well known that a single CAN message is too short for the proper implementation of many security properties. However there are use cases where using a second CAN message

(e.g. for authentication purposes) is not acceptable since it introduce unnecessary latency in the complete reception of a message and it increases the residual error rate in the communication (see for example the appendix D.5.2 of ISO 15998 [4]). In this case the application requirements limit the security level achievable without changing the network communication protocol stack, and is fundamentally due, for a point-to-point communications, to the limited payload length of a single CAN 2.0 bus message.

3.3.4.5 Safety: probability of residual error

The analysis of the probability of residual error P_{re} in a communication protocol is usually a difficult task. However, the use of encryption simplify a lot the calculation, provided that a measure of P_{re} is given for the channel coding, which correspond to P_{re}^{ch} with reference to Figure 7. Here P_{re} is obtained assuming Shannon's ideal cipher model, which has been used in other works like [31], and the results are validated through simulations.

3.3.4.5.1 Theoretical model

According to Shannon's model, an ideal cipher is a random family of permutations, chosen independently for each possible key. More precisely, suppose K is the set of all keys and M is the set of all messages. An ideal block cipher is a map ENC : K × M → M where, for each key k ∈ K, the function $ENC_k(\cdot) = ENC(k, \cdot)$ is a random permutation on the message set M (independent of any other permutation).

In this context, it is more useful to fix a specific pair (k, m) of key and message, and consider the cipher text message set as:

$$C' = \{C' : C' = C + e', e' \in E'\} \tag{2}$$

while the plain text message set is:

$$M' = \{M' : M' = M + e, e \in E\} \tag{3}$$

where the set E ′ is the set of all possible undetected error vectors after channel decoding and E is the set of all error vectors after decryption. Recalling from 3.3.4.4 that M = μ||τ , the probability of residual error can be defined as:

$$P_{re} = P(\mu = \mu', \tau' = H(\mu'))\} \tag{4}$$

with M ′ = μ′||τ ′. The distribution of the values in E depends on the distribution of E ′, in a different way for each different pair (k, m), since the function E(k, ·) will correspond to a different and independent permutation.

Considering all the possible pairs (k, m), each element of C′ can correspond to all the possible values of M′. If the elements of C′ are uniformly distributed (such as when considering a random message attack) the probability that τ ′ = H(μ′), averaged over all the possible (k, m), can be computed by counting as:

$$P^{avg}\big(T = H(\mu)\big) = \frac{2^{\mu_{size}}}{2^{\mu_{size}+\tau_{size}}} = \frac{1}{2^{\tau_{size}}} \tag{5}$$

which is the probability of guessing a valid message. On the other hand, when transmission errors are considered, only one of the elements of C′ correspond to the correct message, while all other elements have cumulative probability P_{re}^{ch} . The residual probability of error in this case can be computed as:

$$P_{re}^{avg} = P_{re}^{ch}\frac{2^{\mu_{size}} - 1}{2^{\mu_{size}+\tau_{size}} - 1} \approx \frac{P_{re}^{ch}}{2^{\tau_{size}}} \tag{6}$$

Both equations (4) and (5) are valid without making any assumption on the actual algorithm H used for computing τ . This is consistent with the usual case where H is a CRC code and there is no encryption, with the value of P_{re} approaching 2−τsize as the probability of error per bit approaches 0.5, i.e. a uniform distribution, see for example [32].

This measure of P_{re} is, however, a measure for the average case, while from the safety point of view it is necessary to consider the worst case, that is:

$$P_{re}^{wc} = \max_{k\in K, m\in M} P_{re} \tag{7}$$

This again can be computed by counting, but this time the actual distribution of E, given by transmission errors, must be considered, specifically the one which maximize equation 7, which is obtained using the worst-case pair (k^{wc}, m^{wc}). Using the ideal cipher model, the distribution in E can be obtained from a permutation of the distribution in E′, so a simpler way to compute P_{re}^{wc} is to take the $2^{\mu size}$–1 error vectors of E′ with higher probability $P_{e\prime}$ and sum their probability. Previous work on Pre for the CAN bus like [33] and [34] only

compute the cumulative value of P_{re}^{ch}, with assumptions also on the structure of the CAN network. This could be considered as an upper bound for the various E , that is

$$P_{re}^{wc} \leq P_{re}^{ch} \tag{8}$$

where equality would mean that the encryption procedure, even with an integrity check, is not effective at all in detecting transmission errors when (k^{wc}, m^{wc}) are used. More specifically, this correspond to the case where the $2^{\mu_{size}} - 1$ most probable error vectors in E cover practically the whole amount of P_{re}^{ch}. To illustrate this problem, we consider a simplified model with a binomial distribution E ~ B(n,k,p) (which would correspond in

Figure 6 to the case with no channel coding and a Binary Symmetric Channel with probability of bit error p). The worst case P_{re}^{wc} can then be obtained by first listing the probability P_l of each error vector el with l bit set; this list is then sorted incrementally and then the first values are taken, one error vector at a time, until exactly $2^{\mu_{size}} - 1$ error vectors are chosen.

3.3.4.5.2 Simulation results

The simulations have been performed applying the encryption scheme to randomly selected (k, m) and using different error vectors to obtain an approximation of P_{re}. For the ENC cipher, DES and AES have been used, while for H we used CRC, a truncated SHA1 hash function and a truncated HMAC scheme based on SHA1. The simulation have been implemented as a C++ program using the Nettle v2.7.1 cryptographic library. The plotted results represent a normalized $\frac{P_{re}}{P_{re}^{ch}}$, where the value of 1 represent equality in equation (7). The value of τ_{size} varies from 0 to 16; greater values, which in theory correspond to a lower P_{re}, have not been simulated since they would have taken too much time to yield a result with reasonable precision; however the results are still meaningful with respect to the theoretical model. The complexity of the simulation has three factors: k, m and e′. For example, using 1000 different keys, 1000 different messages and 10000 error patterns the total number of iterations is 1000 · 1000 · 10000 = 1010. However each value of P_{re} is evaluated using 10000 samples, so lower values close to 10−4 will have a lower accuracy.

This explains the convenience to simulate with low τ_{size}. In Figure 8 the normalized P_{re}^{avg} is plotted, both resulting from equation 5 and from simulations. The correspondence between the theoretical model and the simulation results is very good, and the results are independent either from the H algorithm used to compute τ and the encryption algorithm ENC. Only a small glitch is visible for $\tau_{size} = 14$, presumably due to the relatively small number of iterations. The simulations results are accurate because all the P_{re} are averaged over the pairs (k, m).

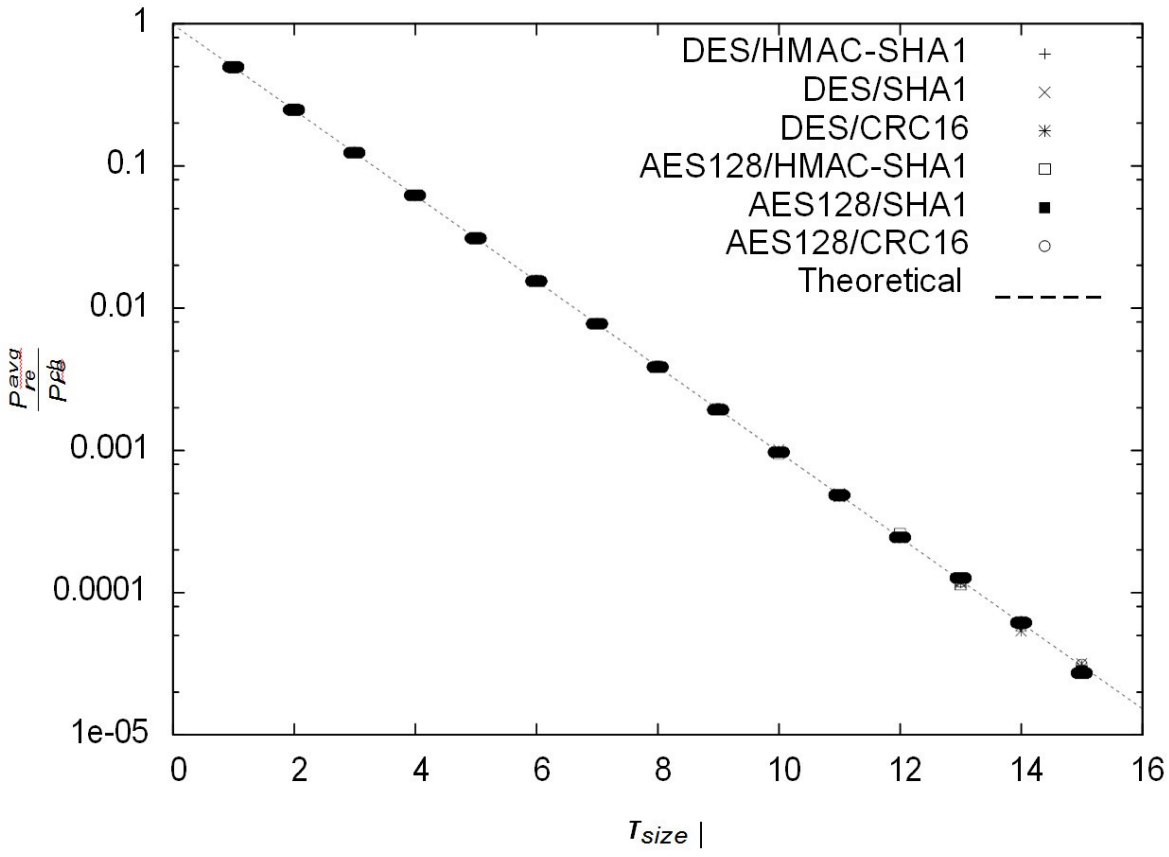

Figure 8. Normalised average P_{re} obtained through simulation and from equation 6, for different values of τ_{size}.

In Figure 9 the normalized P_{re}^{wc} is plotted with varying τ_{size}. The numerical values, displayed with a continuous line, are evaluated with the algorithm described in section 3.3.4.5.1, while the simulation results are taken as the value of P_{re}^{wc} from all the tested (k, m). In this case the simulations do not match the theoretical model; the reason is that while the theoretical model assumes that (k^{wc}, m^{wc}) is known, in practice this is not true, although

for some ciphers it could be feasible to calculate it. In this case a great number of (k, m) combinations are chosen and the worst case is considered. However, being unable to scan all the (k, m) space, it is unlikely to find the worst case but only a ”bad“ pair (k, m) is found, for which P_{re} differ significantly from the average case.

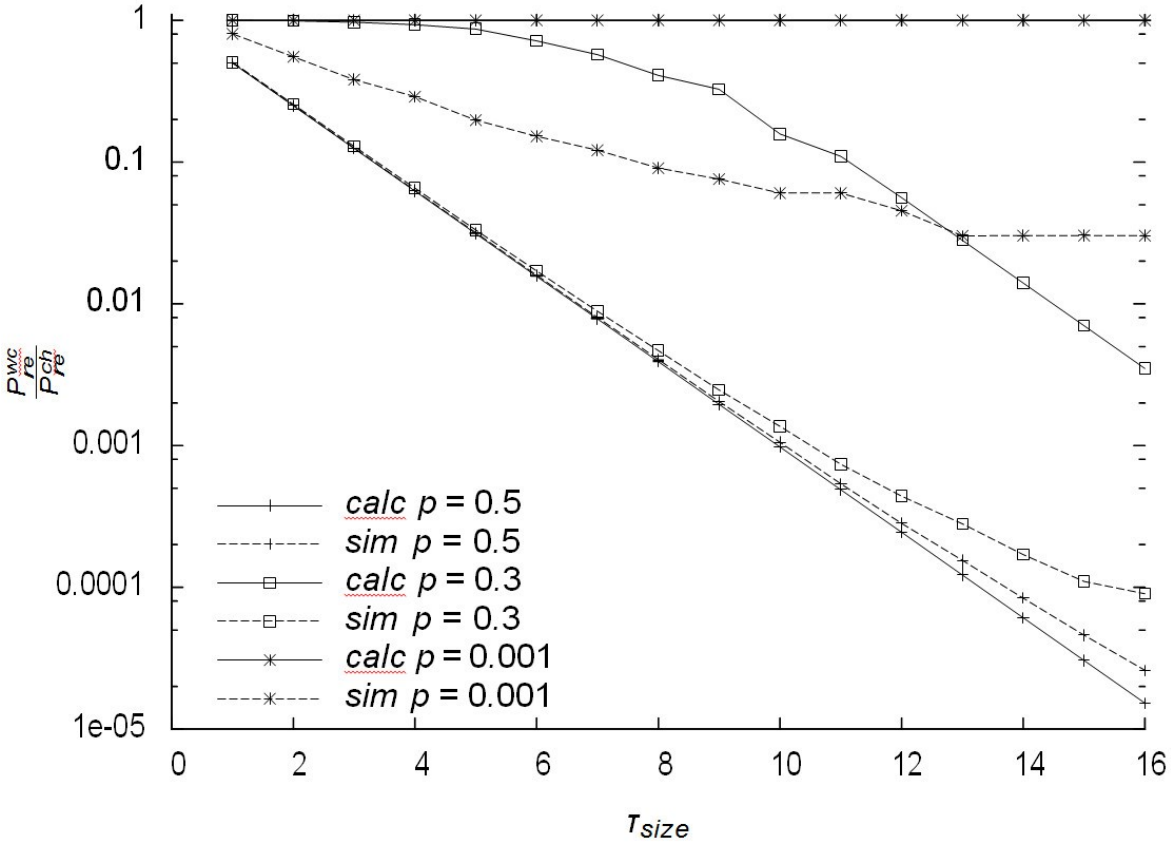

Figure 9. Normalised worst-case Pre obtained with E ′ ∼ B(n, k, p) through calculation and simulation using a

DES/SHA1 scheme, with $\mu_{size} + \tau_{size} = 64$, for different values of τ_{size}.

In Figure 10 the normalized P_{re}^{wc} is plotted with varying p. For low values of p, the value of P_{re}^{wc} does not change a lot, since the most probable error patterns are always the ones with 1 bit error. On the other hand, with higher p the value of P_{re}^{wc} approaches P_{re}^{avg}, which is reached with p = 0.5, corresponding to a uniform distribution. The issue of finding the worst case (k, m) is then different depending on the bit error probability of B(n, k, p). For low p, approximately under 0.1, it is easier to find a pair (k, m) with high P_{re} since the most probable error patterns are the ones with only 1 bit error and are $\mu_{size} + \tau_{size}$. On the other hand, for higher p, the most probable error patterns are a much great number, because

it is easier to find more than one bit error. This explains the difficulty of finding the pair (k^{wc}, m^{wc}) to simulate P_{re}^{wc} accurately.

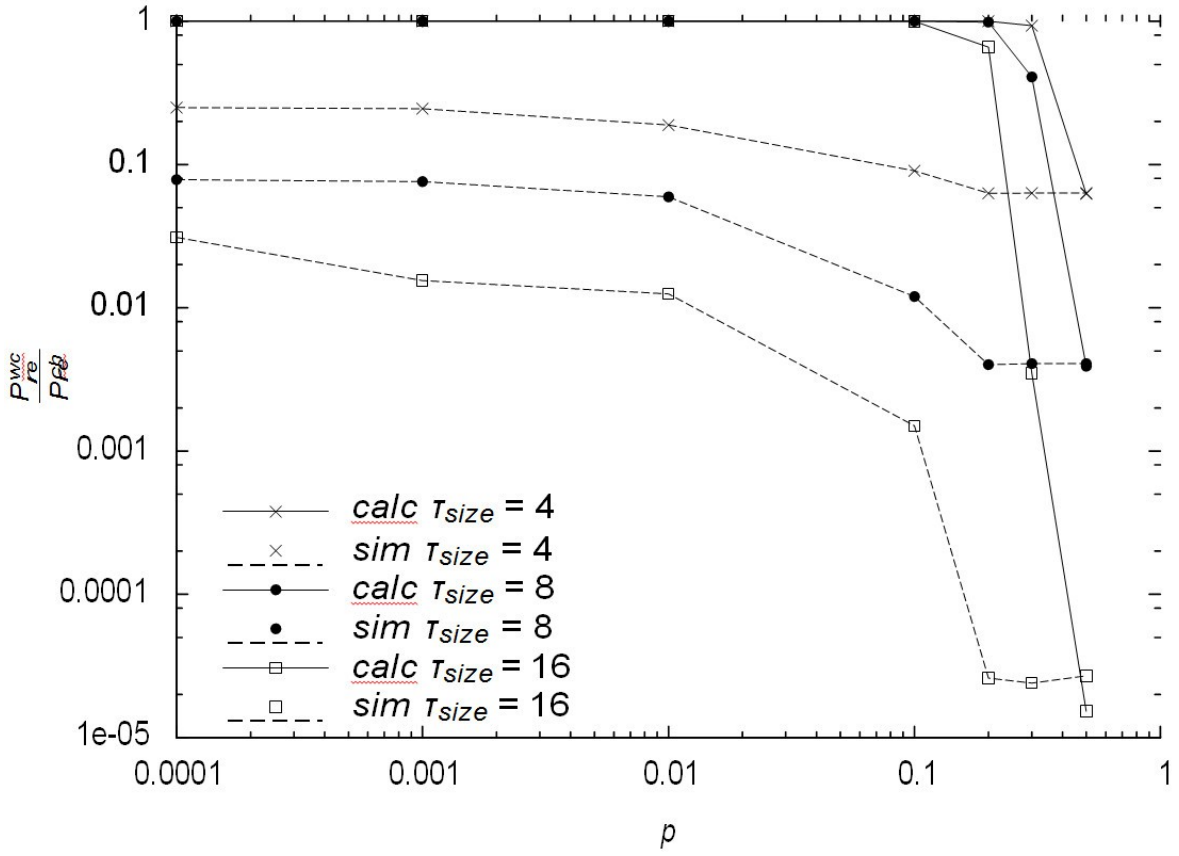

Figure 10. Normalised worst-case Pre obtained with E′ ~ B(n, k, p) through calculation and simulation using a DES/SHA1 scheme, with $\mu_{size} + \tau_{size} = 64$, for different values of p.

3.3.4.6 Safety and security: encrypted MAC and MAC size

While authentication and integrity are reached with the message authentication code (MAC), the confidentiality property is achieved by encrypting $\mu||\tau$. It avoids that an attacker is able to sniff the content of the messages exchanged among ECUs, incrementing the security. Without encryption, an attacker is capable to sniff messages, but, thus it does not know the structure of the message, the forge of a valid message requires an important effort. The encryption, further improves the difficulty, for the attacker, to forge a valid message, and the probability to forge a valid message is tied to the guessing attack (the probability to forge a valid MAC after a number of trials) plus the encryption of $\mu||\tau$. In [14] it is highlighted the importance of having a robust MAC to be resistant against the

guessing attack. In particular, the Dworkin says that a sound MAC is achieved when its size is greater than 64bits, i.e, $\tau_{size} \geq 64\ bits$. Because CAN-bus maximum message size is of 64bits, it is impossible to satisfy the MAC dimension just expressed. It is necessary a workaround to guarantee the security properties. It is possible to apply two different strategies:

- To concatenate a MAC which size is at least 64 bit.
- To consider invalid the key that generates the MAC after a limited number of repeated trials.

The first solution may cause the fragmentation issue detailed in . The second one can lower the τ_{size} requirement. In [14] it is explained how to calculate the right τ_{size} depending on two bounds:

- MaxInvalids: is the limit on the number of trials that an attacker can perform before the key is retired;
- Risk: the highest acceptable probability for an inauthentic message to be accepted as valid;

The τ_{size} should satisfy the following condition:

$$\tau_{size} \geq \log \frac{MaxInvalids}{Risk} \tag{9}$$

From the above equation, if the system can tolerate up to 30 (2^5) messages before considering the key invalid, and the system can accept 2^{-11}, Risk = 2048, chance of inauthentic messages; in this case, the inequality is satisfied for any value of τ_{size} that is greater than or equal to 16. Considering that the size of a CAN-message is generically 48 bits (μ_{size}) and the maximum bandwidth of the communication channel is 64 bit ($bandwidth_{max}$), we obtain that $\mu_{size} + \tau_{size} \leq bandwidth_{max}$. This inequity defines the lowest condition to have a $\mu||\tau$ message for CAN having $MaxInvalids = 25$ and $Risk = 2048$.

3.3.4.7 Safety and security: trade-off

In Table 5 different message protection schemes are compared and ordered with decreasing security properties.

The message protection schemes addressed in this chapter correspond to the ENC+MAC and ENC+CRC schemes, depending on the choice of H(), to resist a fully malicious attacker with chosen plaintext. The main alternative scheme, which does not consider the confidentiality property, is evaluated for reference, based on literature work.

Here the trade-off appears clear comparing the ENC+CRC and plain + CRC scheme; while the first has better security properties, the latter has better safety properties under common channel models, because CRC codes are designed specifically for correcting transmission errors.

The analysis of the scheme presented in this thesis showed that a security property like Encryption directly influences a safety property like the probability of residual error. On the other hand, the length of a MAC code must respect the fragmentation constraints which can be imposed by real-time requirements. With respect to similar schemes without encryption, where a second CRC is used in addition to the one at the physical layer, the scheme ENC+CRC performs worse; this is due to the intrinsic properties of block ciphers, which transform the distribution of errors to uniform on average. Other message protection schemes could have a less drastic impact on the worst-case error detection capability, or even this error detection capability could be embedded into the encryption algorithm itself, but then the risk is to offer the possibility for a side-channel attack. Future works include the design and study of different cryptographic schemes, to offer a better trade-off between safety and security.

Table 5. Summary of message protection schemes.

Scheme	Resists to	Security properties	Leaking out	Safety properties
ENC + MAC	Fully Malicious with chosen plaintext	Confidentiality Authentication Integrity	-	Strongly depends on (k, m)

ENC + CRC	Fully Malicious with chosen plaintext	Confidentiality Integrity	-	Strongly depends on (k, m)
plain + MAC	Fully Malicious	Authentication Integrity	Plaintext to FM and HBC attackers	Depends on H()
plain + CRC	Honest-But-Curious	-	Plaintext to FM and HBC attackers	Good under common channel assumption [10]
plain	-	-	Plaintext to FM and HBC attackers	-

3.3.5 Ethernet and Time Sensitive Networking: the future network

An Ethernet network can be realized with the desired topology. In fact, it is possible to adopt either a segmented network topology or a physical bus topology. The first one is obtained through the use of switches that separate the collision domain, the second one is obtained using hubs to interconnect the ECUs. The packet size can reach up to 1500 bytes when there are no additional tags such as 802.1q or 802.1ac.

Attacks such as replay, sniffing, DoS and black hole are more difficult to be realized thanks to the use of switches that filter the traffic delivering to a node only the traffic directed to it. Furthermore it is possible to create VLANs, according to IEEE 802.1q [28], in order to separate different network, even if they share the same link.

A new set of standards is growing on Ethernet networks: the Time Sensitive Networking (TSN). It was born as Audio/Video Bridging (AVB) task, but has been renamed because of the extension of the working area. Most of the projects are represented by extensions to VLANs, created for those applications with the requirements of a very low transmission latency and high availability. The key components of time sensitive networks are three:

- Common rules to process and forward communication packets;
- Common rules to select communication paths and to reserve bandwidth and time slots.
- Common time reference.

3.3.5.1 Security: authentication

The authentication in Ethernet-based networks can be performed at different levels. The bigger packet size is a great advantage because MAC code can be used for every single message without having the problem of fragmentation. Despite the packet size, a limitation in the choice of the authentication system is represented by the elaboration capabilities of the ECUs. For example, the use of an asymmetric key to authenticate the messages such as a digital signature scheme, is not possible. Also the asymmetric cryptography is not possible due to the elaboration time required, incompatible with a real-time traffic, even if in [35] it has been shown to be feasible on constrained embedded platforms.

The best approach remain the use of a Message Authentication Code (MAC) with a key exchange procedure. An example is HMAC [36]. For key exchange the use of an asymmetric key cryptography can be considered since it is admissible to take hundreds of milliseconds for the key exchange to terminate.

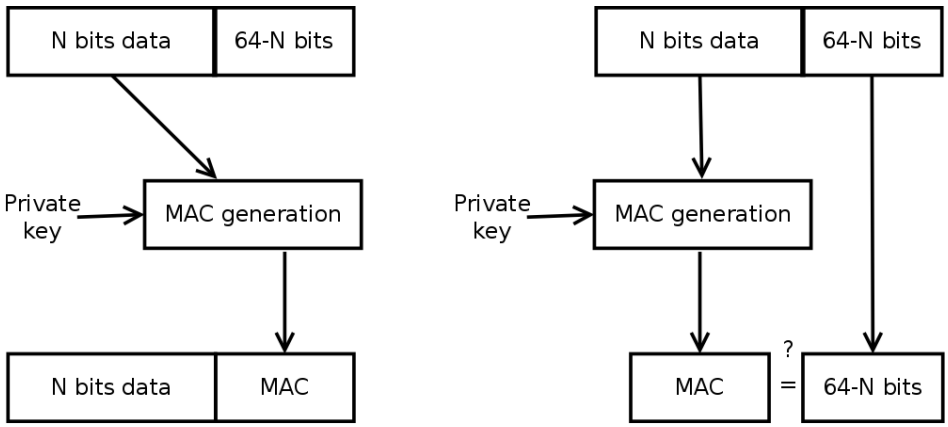

Figure 11. Authentication and Verification procedure with a MAC code.

3.3.5.2 Security: integrity

For the integrity, Ethernet uses a 32 bit CRC at link level. A problem is that the switches can regenerate the CRC of each packet that transit inside it and this can cause an integrity loss. The integrity of the CRC, because of that, is not an integrity end-to-end but just hop-to-hop. The desired end-to-end integrity is provided by higher levels' protocols, such as TCP and UDP [37].

Ethernet provide, anyway, the possibility to reach higher integrity level if compared to CAN bus. For example, many IP based networks adopt has algorithms.

3.3.5.3 Security: confidentiality

The maximum frame size of 1500 bytes allows to use a symmetric cypher like AES and allow to exchange the keys without a transport protocol if sufficiently short keys are adopted. An example can be the elliptic curve algorithm.

3.4 A gateway for hybrid networks

Considered the results of the comparison between CAN and Ethernet for in-vehicle networks and considered that, as reported in [38], [39], [40] and [41] Ethernet has been designed as the successor of CAN, it is necessary to grant a gradual transition between the actual network technology and the future one. Considering that in agricultural and forestry environment vehicles life times are extremely long (20-30 years), it is absolutely grounded to suppose that the technologies based on Ethernet will require tens of years to completely displace the previous ones based on CAN. For this reason a gateway that allows the interfacing between these two kinds of networks is mandatory, supplying retro-compatibility.

3.4.1 CAN and Ethernet: differences that the gateway should smooth over

Both physical and link layers used in CAN and Ethernet networks have several differences. The first one is about access method and traffic prioritization: in CAN networks a CSMA/BA (Carrier Sense Multiple Access/Bitwise Arbitration) method is employed, that embeds a priority scheme to avoid collisions, so the packet with higher priority is not destroyed nor delayed by the packet with lower priority, and it's handled by hardware, using signal levels and bus monitoring, On the other hand Ethernet uses a CSMA/CD (Carrier Sense Multiple Access/Collision Detect) method, and there isn't a native mechanism of priority handling, although several extensions exists, both at Ethernet layer or upper ISO/OSI layers. On the other hand Ethernet's access method is not much important because nowadays' networks are mainly switched Ethernet, in which each device form a single collision domain, with the result that collisions don't exist anymore. Another aspect in which these two networks differ is the network topology. In CAN

networks there's a shared bus topology, whereas Ethernet was born using a bus topology, but can use also a star or a tree topology using switches to put every device in a separate collision domain. The last difference that, in this document, is important to highlight, is in the throughput. In fact the choice to transport modern CAN applications to Ethernet is mainly due to the considerably highest throughput that characterizes Ethernet.

3.4.2 ISOBUS

ISOBUS [11] is a communication protocol for agricultural and heavy duty vehicles defined by ISO11783, based on CAN 2.0B, namely the CAN version that has a 29 bit ID, which is used by ISOBUS as a message header containing the identifier, the source and the destination address. This normative has been defined with the main purpose of allowing the interoperability between different manufacturers, posing as a successor of the SAE J1939 normative [20], adding a main characteristic: the hot plug. This feature has permitted the birth of innovative applications such as precision farming, fleet management and automation farming, for example see [42]. The main result of the gateway presented in this section has been to join, on a single bus, applications that, nowadays, work on different busses. One of the biggest problems to face lies in the very heterogeneous requirements that these applications present:

- There are applications that supply connectivity services as GSM, GPRS and UMTS.
- There are applications that request big data transfers, not necessarily real-time. Some examples are farm management systems and precision farming (e.g. prescription maps), passing by applications that, in addition to demanding the transfer of big amounts of data, need also a certain real-time, like video streams directed to let the worker see critical zones or to recognize animals, humans or objects in working vehicle's action range, in manner to modify the behavior of the autonomous guide vehicle (safety).
- Finally there are applications that, even if they don't need to transfer big amounts of data, they need a very strict real-time and very frequent communications, like drive-by-wire.

3.4.3 Comparison between TCP/IP and ISOBUS

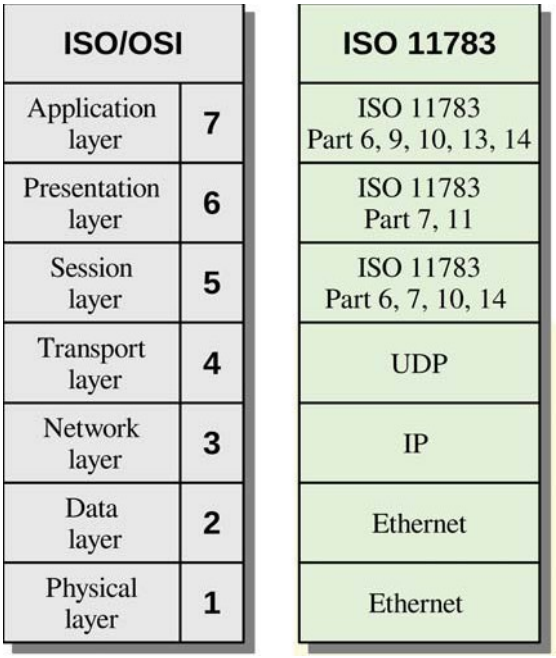

Figure 12. ISO/OSI stack's replacement

This chapter compares Ethernet and CAN networks considering the ISO/OSI layer 3 and 4 protocols. As shown in Figure 12, in order to adapt the ISOBUS stack to the standard ISO/OSI protocol stack, this thesis deals with the process of using Ethernet and the TCP/IP stack as a replacement for the lower four layers. The most important difference to highlight for better understanding of project challenges in this kind of gateway is about addressing systems. In fact these systems differ very much. ISOBUS uses a distributed addressing system based on address claiming to assign to every device a unique 8-bit CAN address, a mechanism defined by ISO-11783 normative and that uses a set of messages that let the device be able to ask to other devices to claim their address through "request for address claimed" messages and let the devices claim their own address by using "address claiming" messages. Using this mechanism the normative implements a system to solve addressing conflicts by giving priority to those devices characterized by a lowest ISOBUS NAME, a 64-bit unique identifier that distinguishes every device (similar to MAC address in Ethernet). On IP networks [43] the de-facto standard is DHCP [44], a centralized protocol where a server assigns IP addresses to every device that asks for it using a particular protocol based on UDP as transport layer. An important difference between the

two addressing systems is the maximum number of hosts that can be present in network, which is limited to 254 for ISOBUS but can be up to about 4 billion for TCP/IP. Both ISOBUS and TCP/IP support connection-oriented and connection-less communications. In the former the available protocols are BAM, TP and ETP, while in the latter the protocols used are respectively UDP and TCP. One important difference is that ISOBUS has a native broadcast protocol, which must be replicated on TCP/IP networks using UDP. Furthermore, while TCP permits exchange of an arbitrary amount of data, TP and ETP have both size limits.

3.4.4 Implementation challenges

The gateway has been realized to be used with a standard Ethernet physical layer working at 100 Mbit/s, but it can be used even with 10 Mbit/s Ethernet or Gigabit Ethernet, modifying the transceiver. Other Ethernet field busses are not compatible with the functionalities implemented because in future implementation it is going to be necessary to have a high priority (real-time) traffic using TCP/IP stack. Most field busses, like Powerlink, Ethercat, Profinet, etc., considers all TCP/IP traffic as low priority , and uses a custom protocol for high-priority traffic. The adoption of these field busses would require a more complicated address management in order to integrate the high-priority protocol, and for this reason they are not considered in this thesis. Furthermore, usually real- time Ethernet field busses don't support hot plug, which is mandatory for ISOBUS.

The gateway has been realized trying to reuse, as much as possible, already existing protocols and standards to have a better integration with external world, particularly with the Internet of Things paradigm. Accordingly to this decision address management on Ethernet side relies on DHCP protocol, with the addition of standard vendor options. The same strategy has been respected in the choice of using IP as network layer and UDP as transport layer. For the network layer it's been chosen to use IPv4 instead of IPv6 for its reduced information overhead. In fact IPv6, even if it supports a native procedure for distributed addressing, has a higher overhead and is designed with a set of features that are unnecessary for this thesis's purposes and that could make difficult the accomplishment of real-time requirements and causes an error probability increase, due to the major packet length. However, it could be possible to use IPv6 instead of IPv4 with some modifications

on the packet format but it's discouraged for the reasons expressed above. The choice of using UDP instead of TCP is motivated by having a better control over latency, since the former employs mechanisms such as retransmissions and sliding window which can add a significant delay. Latency control is very important to be able to realize real-time applications.

The firmware is based on ChibiOS that is an open source real time operative system. This choice is due to its high compatibility with the hardware adopted, and to its support to LwIP TCP/IP stack.

3.4.5 Architectural overview

In this chapter will be described those project's choices that have been made to optimize gateway's functions. The first choice regards addressing procedures. The gateway has the role to hide CAN network to Ethernet devices and vice-versa. In address claiming procedures on CAN network it has to respond to requests for address claim messages in place of Ethernet devices and to forward the address claiming message of the incoming device to Ethernet network. It has to do the same things on Ethernet side, with the only difference that on Ethernet network it is necessary to implement a new set of messages to use address claiming procedures over UDP. The second choice that has been made is about which parameter to use as pivot in address translation. Accordingly to previous publications [41] the choice has been to use ISOBUS name because it's the only parameter that univocally identifies a device and that characterize both ISOBUS over CAN and ISOBUS over Ethernet devices. Moreover, this allows a greater freedom in choosing the topology of the network both at network and link layer and allows a simple address management, as described in the following section. The third choice that has been necessary to make regards IP addressing space division, which in turn creates different network topologies. There were two possible choices:

- To use a single IP subnet and let the gateway simulate CAN devices. This means that the gateway has to intercept all the traffic originated from Ethernet devices that is directed to CAN devices and forward it to CAN network, and to forward the traffic generated from CAN devices to the Ethernet network. In this case the

gateway shall employ IP aliasing in order to intercept the traffic from Ethernet devices. This topology is very simple, as there is no logical separation between Ethernet and CAN network;

- To use two IP subnets and let the gateway appear as a router for devices on the Ethernet network. In this case, because CAN devices are on another IP subnet, Ethernet devices must be configured to have a static route to the gateway, which can be achieved with a DHCP Classless Static Route option. This topology has the Ethernet and the CAN network logically separated, which can be useful for management purposes, as it reflects the physical topology of the network.

An example of an hybrid CAN/Ethernet network is in Figure 13.

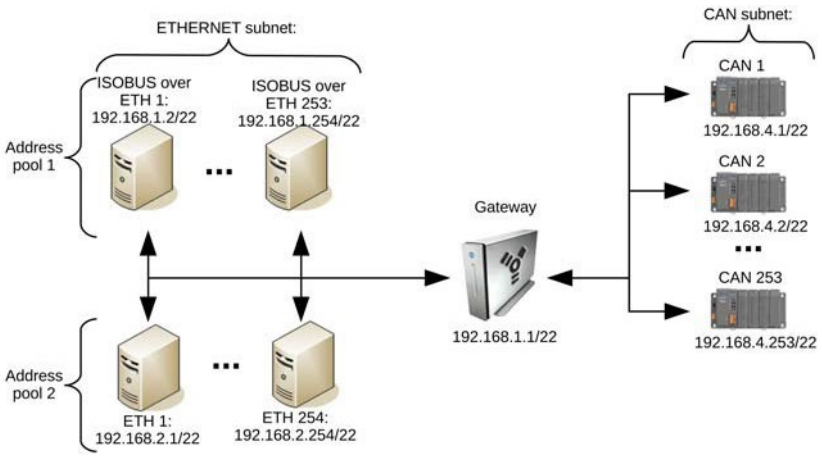

Figure 13. An example of an hybrid CAN/Ethernet network.

3.4.6 Address management

The key characteristic of the gateway is that it has to be transparent for the devices both on CAN network and on Ethernet network. To achieve this, it must handle the different procedures of address assignment on both networks, namely DHCP and address claiming, in such a way that every Ethernet device must know which devices are active on the CAN network, and vice versa. To realize this feature, the gateway employs both the standard DHCP and address claiming procedures, in order to publish a new device on the network where it is not directly connected. In order to maintain the compatibility with existing CAN devices and non-ISOBUS Ethernet devices, it has been chosen to create an UDP message set that duplicates the address claim messages defined by ISO11783 normative.

All messages have a standard UDP header; the payload contains the 64-bit ISOBUS name for the UDP Address Claim message, while for the UDP Request for Address Claim message the 64-bit payload is set to all zeros.

The standard address claiming is a distributed procedure which starts when a new device joins the network, sending a Request for Address Claim message, then all devices reply with an Address Claim message. At this point, the new device sends its own Address Claim message, publishing its ISOBUS address. The gateway here must send the Address Claim messages on behalf of all the Ethernet devices, and finally sends an UDP Address Claim message on the Ethernet network, in order to publish the new device. This procedure is reported in Figure 14.

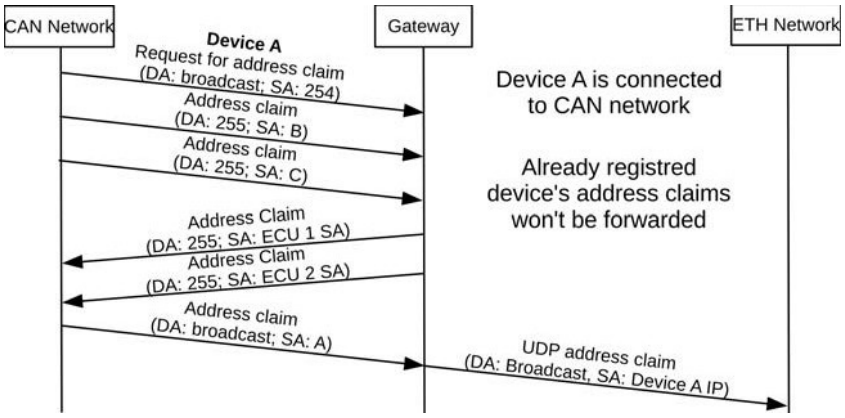

Figure 14. Communication diagram for the procedure in case a new CAN device reaches the network.

On the other hand, the standard DHCP is a centralized procedure which starts when a new device joins the network and sends a DHCP Discovery message, and the DHCP server answers with a DHCP Offer message, containing the address that the client should use. If the client accepts the address, it sends a DHCP Request message, and the server finally answers with a DHCP Ack message. At this point, the Ethernet device broadcasts an UDP Request for Address Claim, and each Ethernet device replies with an UDP Address Claim. Here the gateway must send the UDP Address Claim on behalf of the CAN devices, and sends an Address Claim message on the CAN network to publish the new device. A communication diagram for this case is in Figure 15. These procedures are specifically designed so that only new associations must be forwarded from one network to another, leaving the possibility of a device changing its address on one network without affecting

the other. This is a fundamental feature of address management, since it decouples ISOBUS addresses from IP addresses, allowing the address of a device to change without the need of complicated address resolution procedures. In short, the ISOBUS addresses assigned to CAN devices are completely independent of IP addresses assigned to Ethernet devices. In order to forward packets from one network to another, the gateway must keep an address translation table, where the associations IP address-ISOBUS Name and CAN address -ISOBUS Name are kept. There associations are added when one of the two procedures described above ends, and allows the translation of the addresses from one network to another, pivoting on device's ISOBUS Name.

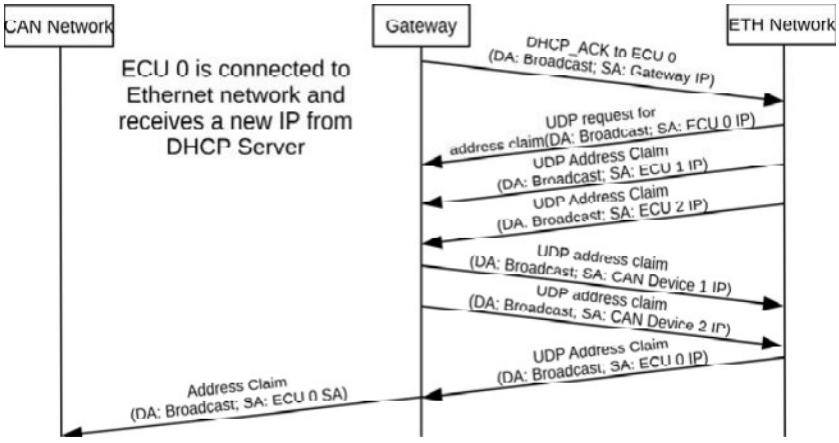

Figure 15. Communication diagram for the procedure in case a new Ethernet device reaches the network.

3.4.7 Experimental implementation

The gateway is based on a Cortex M4 microcontroller, that is a 32 bit ARM device with Harvard architecture [45], and the firmware is based on ChibiOS [46], a real-time operative system, portable and open-source, that natively supports LwIP [47], a TCP/IP stack optimized for embedded environments, therefore lacking some advanced functionalities. This stack has also the feature of being open-source, fundamental in order to fulfill gateway requirements because it has been necessary to customize it in order to adapt to system requirements. In fact LwIP does not natively support IP Aliasing , nor receiving packets which destination’s IP address differs from the host IP address . For this reason the stack has been modified to allow the reception of every packet containing the MAC address of the gateway in Ethernet header and to allow the transmission of packets

with different IPs than the one associated to the host. In this way it has been possible to make the device work as a real gateway should do, “virtualizing” a second IP subnet formed by CAN devices. For this reason it has been chosen to split, through an appropriate subnet mask, the IP addressing space in two separate subnets: one formed by ISOBUS over CAN devices and the other one formed by ISOBUS over Ethernet devices. On the Ethernet network, using the vendor options mechanism offered by DHCP protocol, it's been possible to distinguish ISOBUS over Ethernet devices from simple Ethernet devices, dividing the address space reserved for this network in two different address pools. In this way standard Ethernet devices can work on this hybrid network without disturbing the ISOBUS communications. Gateway tasks can be divided in two big categories: address management and packets forwarding. About the first item, in order to conciliate the requests of both networks, the gateway implements both the centralized IP address management through a DHCP server expressly written, and the distributed ISOBUS address management through the support to address claiming procedures described by ISO11783 normative. One of the most important components of the gateway is the DHCP Server. It is based on LwIP stack that, being designed for embedded systems, does not natively support advanced standard features like DHCP vendor options. Consequently it has been necessary to add this functionality in order to permit ISOBUS over Ethernet devices’ identification . A client that needs to obtain a new IP address will then use DHCP’s vendor option field containing the device’s ISOBUS name. Gateway takes are of publishing on CAN network the ISOBUS addresses assigned to Ethernet devices and vice versa, through the procedures of address management described in the previous section. In order to have a complete test network, a second device, implementing an ethernet ECU, is needed, allowing to realize a demonstrative application. The Ethernet ECU includes a DHCP client, that is, basically, the LwIP one, with standard vendor options added to let it identify itself as an ISOBUS device. It must also support address claiming procedures and must allow to exchange ISOBUS data both in reception and in transmission with CAN network across the gateway. The demonstrative application consists of a couple of Olimex demo boards, a CAN-USB converter, an Ethernet hub, an ISOBUS working set master, a virtual terminal and a PC to monitor traffic. After an address initialization phase, made accordingly to previously exposed ways, ISOBUS over Ethernet ECU sends some

ISOBUS data to gateway, that will forward the packages to CAN network and, in particular, to ISOBUS working set master, that will forward them to virtual terminal where they will be displayed. To test the communication in both directions, the ISOBUS working set master will make a second elaboration on data, sending an answer packet that will reach ISOBUS over Ethernet ECU passing through the gateway.

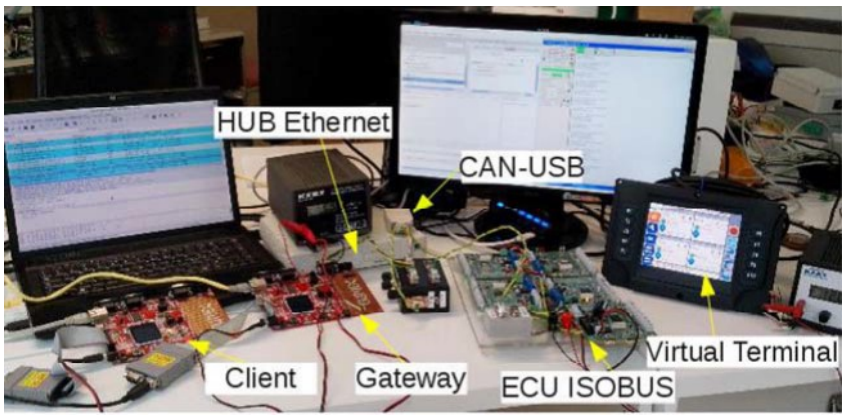

Figure 16. Demonstrator setup.

Chapter 4. Safety architecture

Considering the necessity to adopt wireless technologies for agricultural and heavy-duty machines a deep study is necessary in order to be able to adopt it in safety critical applications.

A field of wireless that has a particular interest is that of Wireless Sensor Networks (WSN), thanks to its low power consumption and to its low cost. A contribution of this thesis, contained in this section, is the analysis of both hardware and software aspects of a wireless sensor node, and the proposal of solutions in order to be able to adopt this technology in safety related applications [48], [49].

Both analysis are going to be performed by keeping in mind the statements of ISO25119, that is the reference standard for the field of agricultural vehicles and that has one of the best set of rules for software development between type C norms in vehicle sector.

After the analysis of hardware and software aspects, it will be presented another contribution of this thesis, that is a prototype of wireless sensor network node suitable for safety critical applications [50].

4.1 Software study for WSN

The subject of software quality has been debated in 2.2, describing what do safety norms establish in order to develop software suitable for safety critical applications. This paragraph will analyze the particular aspects that regards the software quality for WSN applications, with a particular attention to the OS dedicated to this field.

To improve the reliability of WSN software by design it is possible to use an extension of the C language, but it requires a certain effort.

Usually a safety-related application seeks to obtain a certification. In order to reach the certificate, it would be necessary to use tools with their own compliance statement for this kind of applications, but, as far as the author knows, no such tool exist at present day.

A possible solution can be the use of widely adopted tools and architectures that can be considered proven in use, as defined by ISO25119. Some examples are ARM and PowerPC architectures with their commercial compilers.

The best solution would be the use of languages, like ADA, already adopted for safety-related applications, but this requires a considerable effort and the agricultural world is not able to deal with it. The solution adopted is the use of a commonly adopted C subset, named MISRA-C, that allows to reach an higher software safety level. Alternative coding standards for embedded systems are presented in [51].

A method that might be necessary to adopt is the fault tree analysis in order to reach a medium/high software safety level.

One of the greatest problem of wireless sensor networks' software is the architecture adopted. In fact, usually, WSN operating systems have a cooperative scheduler. This mean that it is necessary to analyze the system as a whole in order to determine the reaction time of system functions, such as sensor reading, data elaboration and transmission. An help in solving this task is provided by nesC language. Another critical aspect is that, often, the low level drivers operate polling on some registry flag in memory-mapped peripherals in order to wait for a state transition. This behavior should be avoided since it can be the cause of lockups in case a peripheral has a malfunction or an unexpected condition. In case polling is strictly necessary it should be added a timeout and the error condition generated should be handled.

Both these aspect does not allow the system to be predictable. The solutions proposed leads to a less efficient WSN node in terms of power consumption, but they are necessary to reach a certain safety level.

The respect of ISO25119 guidelines about software quality will also affect the procedural aspects of the development method, causing an extra-cost for the company that will have to adapt its tools, IT systems and team.

4.1.1 Operative systems

4.1.1.1 State of the art

Many operative system dedicated to WSN applications have been developed. The first one has been tinyOS [52], studied to optimize the power consumption of WSN nodes. Another important OS is Contiki [53], that also provides a TCP/IP stack named uIP.

Some available OS are: MANTIS OS [54], SOS [55], MagnetOS [56], LiteOS [57] and AmbientRT [58].

In an OS can be identified three main characteristics that are useful as parameters to be evaluated in order to adopt an operative system in place of another. These characteristics are very important, in particular, in safety related applications. These characteristics are:

- Programming paradigm: it can be synchronous or asynchronous.
- Memory management type: it can be static or dynamic.
- Kind of architecture: it can modular or monolithic.

The first characteristic describes if the OS is an event based one or a polling one. Most operative systems for WSN are event-based. The reason is that an event-based OS allow a better power management, optimizing the power consumption by performing operations only when those are requested by external events. This programming paradigm is not the best one for safety applications, in which a polling (with timeout) strategy is to be preferred. The reasons are mainly two: in an event-based system it is not possible to determine the execution times and this is not compatible with the real time requirements of safety applications and, in case of some error it is easier to get a lockup of the system. The best thing would be to have the support to multi-threading thanks to its bigger control of timeouts of scheduled events and activities. Unfortunately, this kind of support in not natively included in most operative systems. However it is possible to add it through libraries, as, for example, TinyThread [59] or TOSThreads [60].

The second characteristic is determined by the strategy of memory management of the OS. In a dynamic system the memory is allocated when it is necessary to use it, while in static ones the memory is allocated at compilation times. Again, there is a tradeoff between the

optimization in the use of resources and the safety aspects. In fact a dynamic system would optimize the memory requirements, considering that only the necessary one is used, but the ISO 25119 forbids the use of dynamic allocation because it is potentially very dangerous. It is dangerous, in particular, in embedded systems where resources are limited. The reason is that it is not possible to be sure that the memory will be available in the moment in which it is requested and this can lead to critical race conditions. The consequence is that static systems must be used or, in case of usage of a dynamic system, only the static allocation of memory has to be adopted. To make an example with the OSs named before TinyOs is a static system and Contiki is a dynamic one.

The last characteristic continues the tradeoff between performance optimization and safety requirements. From one side a monolithic system has a smaller footprint and can fit in devices with a limited memory size, but on the other hand a modular system permits a better division between kernel space and user space, granting a greater safety in the execution of the OS. Another important characteristic of a modular system is to allow the update of the application using Over The Air updates.

4.1.1.2 Safety requirements

The safety requirements of and operative system, such as any piece of code, are defined by the related norms and are more strict for higher Software Requirement Levels (SRL), as stated in 2.2.

For OSs two approaches are possible:

- To use a certified operative system: this is the best approach for small series of products or for products that requires a very high SRL. The use of a pre-certified operative system permits to not have to follow the long and expensive processes of developing a certifiable system and of certifying it. However, usually, pre-certified operative systems has also a cost per piece.
- To certify the operative system to be used: this approach can be adopted for low SRL or for big volumes of production for the same reasons described in the previous point.

An operative system, from the point of view of the standards, is considered as a software component that is able to create different execution domains and can divide safety critical applications from non-safety ones. This a very important characteristics because, if the OS suites the requirements of norm, then is considered able to separate the domains, it is possible to certify separately the part of software related to safety and not all of it. Moreover, if in the software is implemented a safety function that requires an SRL 1 and another one that requires an SRL 2, those parts can be certified for the respective SRLs. In absence of this separation all the software should have been realized and certified for an SRL 2.

The OS must, of course, withstand a SRL equal or higher than the highest SRL of all the tasks handled by the OS.

The requirements are usually similar to those of a high reliability system. A note can be performed on this aspect. The difference between high reliability and safety is that the first one refers to the percentage of time in which the system operates as expected, while the second one is related to the residual error probability in a system. For safety applications errors are admitted, but the system must be able to detect them and to react accordingly in order to avoid damages or injuries to persons, environments or other entities.

Between the OSs dedicated to WSN, none of them is certified, nor is certifiable without work. All of them require a certain work in order to obtain a certain SRL.

Some parameters are going to be defined in order to assess the overall software quality. After that definition, the parameters are going to be evaluated for some open-source operating systems.

4.1.1.2.1 Domain Separation

As briefly described in the previous section, for an OS to be used in safety critical applications and important characteristic is to be able to completely separate the execution domains of the different tasks. Appendix B of ISO25119 gives an example of how to partition memory, time and communication resources.

The different domains can be executed in different regions of memory, by means of an hardware MPU (Memory Protection Unit) or MMU (Memory Management Unit). The memory separation should include both code and data memory and also I/O memory, e.g. memory-mapped peripherals. If different tasks in different memory partitions need to share the same hardware peripheral, a proxy task must be used, forcing an appropriate policy for accessing the resource.

The temporal independency of tasks is very important. For example, a task, which takes a long time to execute, must not prevent the correct execution of a fast task. Usually it is possible to implement this feature by using a preemptive scheduler with a priority system, where priorities are correctly assigned depending, for example, on the criticality of the task, deadline of periodic tasks and so on. Common priority scheme are the Rate Monotonic (RM) and Earliest Deadline First (EDF).

In the implementation of the domain separation it must have been inserted also the ability to separate the other shared resources, like a communication stack or a communication interface, in order to avoid that a malfunction in a task can affect the other tasks. An example of domain separation is depicted in Figure 17.

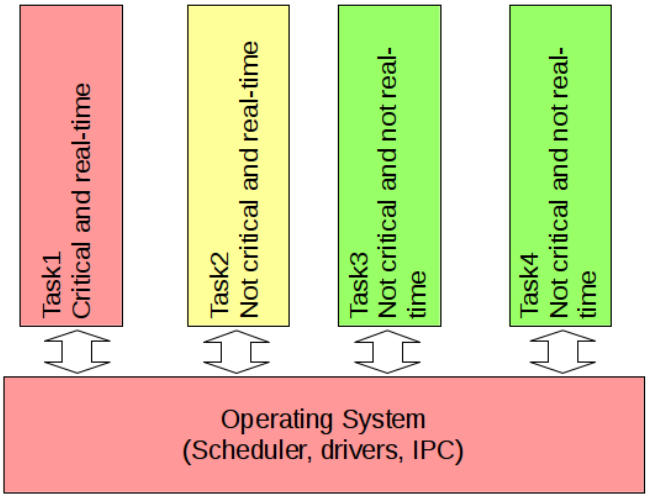

Figure 17. Domain separation for four tasks with different levels of criticality and real-time requirements. Colour indicates the level of criticality: red high, yellow medium, green none.

4.1.1.2.2 Real-Time Performance

A requirement that safety related systems has is that the recovery to a fault must be performed within a certain time and can't be delayed or prevented to execute. An OS can satisfy this requirement, for example, with priority-based preemptive scheduler. Such an OS is referred to as hard real time.

Also different concurrency model, if used in the right way, can satisfy this requirement.

Most Electronic Control Units (ECUs) has no operative system and the tasks are periodically executed in the main function. This behavior is similar to that of a cooperative scheduler. If the execution time of all tasks is upper bounded the worst-case reaction time will be upper bounded too. However such an architecture is difficult to be used for WSN nodes since it would inhibit the use of the low power mode of the module.

4.1.1.2.3 Implementation

Considering that the software for embedded systems is usually implemented in C language and that this code does not fully fit the requirements of ISO25119, some precautions must be taken. First a coding standard must be followed. A coding standard defines many aspects as, for example, the conventions used to formatted the source code, complexity limits for modules, hierarchical organization of the modules and language subsets.

A commonly adopted set of rules adopted for safe and reliable software is the MISRA C, already described in previous paragraphs. The great advantage is that MISRA C 2012 is sometimes considered a strongly typed subset of C. It is possible to check the compliance to MISRA rules thanks to the numerous static analyzers present on the market, that can automatically find violations in the source code. In this chapter the results of static analysis will be adopted as a quantitative measure to assess implementation quality. It must be noted that a full MISRA check is not limited to a static analysis, but should include a manual inspection. However, in order to compare the qualities of the OSs available for WSN applications it is considered sufficient the use of a static analyzer.

4.1.1.2.4 Test Coverage

A characteristic that has assumed a great importance is the possibility to perform automated tests in order to avoid regressions and ensure the correctness of software functionalities. Such tests may be performed after every commit in the codebase. This approach is called test-driven.

In order to evaluate the quality of the tests performed it is used a parameter called test coverage that represents the percentage of code covered by the tests. It can be defined in several ways. The most common one is that of the line of codes executed, but other definitions can be: the number of modules, the number of functionalities defined in the specifications, the number of branches.

4.1.1.3 Evaluation of existing OS

The parameters defined in the previous section are now going to be used in order to evaluate three different operative systems that can be used in a wireless sensor node. The systems that will be analyzed will be ContikyOS, TinyOS and ChibiOS. The first two are specifically designed for WSN applications, while the third one is not but, even if it is not certified against a safety standard, it has many characteristics that are relevant to achieve a medium SRL and that are missing in the other two.

4.1.1.3.1 ChibiOS/RT

The ChibiOS project is composed of three major software components: Chibios/Nil, ChibiOS/RT are two RTOS, the first aimed at extremely low resource usage, but with limited features, while the second is slightly more resource- demanding, but with a more complete feature set. Finally, the ChibiOS/HAL is a set of peripheral drivers for various embedded platforms, which can be used theoretically with any RTOS, like ChibiOS/Nil or ChibiOS/RT, or even without RTOS.

4.1.1.3.1.1 Domain Separation

The separation of the different threads can be done in the time domain, that is with a priority-based preemptive scheduling. Even in this case, care must be taken using critical regions, which can introduce unexpected latency. The different tasks must then be developed in a coherent way, to avoid timing interference between different threads. The

only execution model available is the single process-multi thread execution model, that is all tasks share the same addressing space, and memory separation is not implemented, even if available by means of a MPU or MMU peripheral on some of the supported architectures like ARM Cortex M3, M4, M7 and PowerPC e200x.

Finally, the communication between different tasks can be done with mailboxes with an asynchronous interface. This allows to decouple the communication between different tasks.

4.1.1.3.1.2 Real-Time Performance

The scheduler of ChibiOS/RT is a priority-based preemptive scheduler, suited for hard-real time systems. The time overhead of operations like context-switch, interrupts and synchronization primitives can be obtained using the test suite. There is support for performance statistics, so although not fully integrated it is possible to develop a task supervisor, to check for timing errors.

4.1.1.3.1.3 Implementation

The ChibiOS/RT operating system is implemented in MISRA C 2012, with some justified exceptions, based on design decisions. A static code analysis with the PC-Lint analyzer, on the ChibiOS 16.1.4 release, reported no violation of the checked MISRA rules, using the configuration provided with the source code. Additionally, the coding standard in terms of naming conventions, design patterns are explicit and consistently used on all the software. The documentation for the source code is available and generated with Doxygen, and includes a description of all the modules and their relationship.

4.1.1.3.1.4 Test Coverage

The ChibiOS/RT kernel has a test suite to verify the functional correctness of mechanisms like priority-based scheduling, priority inversion, timers and all the synchronization primitives offered by the operating system. The test suite can be executed both on a simulator and on real hardware, and can be used also for benchmarking a given platform. The test suite is available for both the Nil and RT operating systems, and also for the HAL, so the peripheral drivers can also be tested.

4.1.1.3.2 Contiki-OS

4.1.1.3.2.1 Domain Separation

Memory separation is not supported because of lack of hardware in most supported architectures (e.g. msp430, ARM Cortex M0). Its scheduler supports both cooperative and preemptive tasks, but there are no well-defined mechanisms for inter-task communication; each module is free to expose its API, and concurrency issues must be handled manually.

4.1.1.3.2.2 Real-Time Performance

The scheduler does not support different priorities for the tasks, which can however be cooperative or preemptive. A number of timer modules are available, notably the rtimer module provides real-time timers, which can be used to schedule preemptive tasks.

4.1.1.3.2.3 Implementation

Contiki is an operating system developed mainly in standard C and portable to various platforms. Its coding style is contained in the documentation, and it covers only the formatting style. A MISRA C check has been performed, using the PC-Lint static analyzer, with the default configuration for MISRA checking. A total of 1138 files has been checked, covering the dev/, core/, platform/ and cpu/ directories of the Contiki 3.0 source tree. The apps/ directory has not been analyzed, as it contains application-level modules. In total, 73.681 messages has been generated, most of them related to either styling issues or easily correctable errors. However, there were many reports of more severe errors:

- Uninitialized variables (28 messages)
- Unused return codes from functions (284 messages)
- Null pointer dereferencing (13 messages)
- Multiple side effects in expressions (34 messages)
- Buffer overrun (12 messages)
- Shadowing of variables (44 messages)
- Discarding of const or volatile qualifiers (4 messages)

- Recursive functions (1 message)

Some messages were related to false positives given by the static analyzer, while other has been manually verified, although not all results have been checked.

Finally, a lot of messages were related either to violations of the MISRA data type model, much stricter about mixing variables of different types, or warnings about incompatible pointers or pointer arithmetic.

4.1.1.3.2.4 Test Coverage

Regression tests are performed on every new version of Contiki, although they seem to cover mostly the communication protocols and not the operating system services like scheduling and inter-task communication.

4.1.1.3.3 TinyOS

4.1.1.3.3.1 Domain Separation

Memory domains are not supported, because of lack of support in the main reference architectures (AVR and MSP430). Although support has been added for architectures with Memory Protection Unit like ARM Cortex M, currently the MPU is not used. The separation of tasks in time is statically checked by the NesC compiler for race conditions, and these checks are improved using the Ivy Toolchain. Inter- task communication is handled asynchronously by means of events

4.1.1.3.3.2 Real-Time Performance

TinyOS execution model is composed of events, which must be designed to run to completion. Although initially events were scheduled with a FIFO policy, subsequent works added both priority-based and preemptive scheduling.

4.1.1.3.3.3 Implementation

The TinyOS operating system is implemented using NesC, a language developed ad-hoc for WSN firmware. It supports asynchronous programming end energy saving, as well a component-based organization of the source code. The NesC compiler translates the source

file in a C source file, which then can be compiled with a C compiler. The NesC compiler also executes whole-program tests in order to check the concurrent access to resources.

An extension of the C language has been developed for SafeTinyOS [61], where an additional step has been added to the compilation. In this case, NesC code is translated into annotated C code, which is then translated into classic C code. The annotated C code, to be used with the Deputy compiler as part of the Ivy Toolchain [62] and [63], has been developed with type safety in mind, and it allows to check pointers, arrays, unions, structures, string/array termination, as well dynamic features like memory allocation. Furthermore, the Heapsafe and SharC components of Ivy allow to execute whole-program checks for memory allocation and sharing between concurrent tasks.

Although this certainly contribute to enhance the reliability of the firmware, it is a substantially different approach than MISRA C. The Ivy Toolchain appears more focused on the most common causes of implementation errors, that are pointers and type casts for C software, as well as sharing of resources for concurrent tasks. On the other hand MISRA C have a more general target, that is to remove the inconsistencies and weaknesses of C. For example, the Deputy documentation does not consider undefined behaviors that can be present in the auto generated code, while the MISRA C guidelines cover all the aspects of the C language.

4.1.1.3.3.4 Test Coverage

Unit testing is not part of the main TinyOS repository, but can be found on the tinyos-2.x-contriob repository, with a framework called TUnit. However it seems that tests for the operating system are not comprehensive, but the TUnit module is more focused on supporting the test of the final application.

4.1.1.4 Discussion

From the analysis performed emerges that the systems commonly used for WSN applications do not have the characteristics expected for safety relevant applications. They are developed with the purpose of optimizing the power consumption, but the characteristics that permit this low power consumption are incompatible with the safety requirements. The most interesting result is the amount of work performed on NesC and

TinyOS to increase the reliability. It is not yet consolidated as tool for safety relevant systems but it may become an interesting future option.

4.2 Hardware study for WSN

The present-day hardware architecture of the WSN modules off-the-shelves is very far from being compliant to what the ISO25119 and similar standard describe.

The current state of hardware design will be analyzed, in order to be able to define the safety requirements to satisfy in order to have a wireless module that can be used in safety relevant applications. After the definition of the requirements a possible hardware architecture is proposed.

4.2.1 State of the art

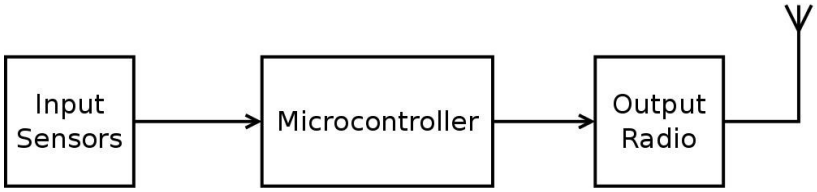

Figure 18. WSN node typical architecture.

Typically, a WSN node has a single control chain composed by input, logic and output. As shown in Figure 18 the input is the sensor to be acquired, the logic is usually a microcontroller and the output is a transceiver with its RF front-end. It is immediate to notice that this kind of architecture is and hardware category 1, as reported in Figure 4.

Going into the detail of every single component of the control chain there are:

- Input: a single sensor is used for each physical quantity to be measured. The conditioning circuit is as simple as possible, in order to save as much power as possible.
- Logic: it is normally composed by a low-power microcontroller, for example based on the ARM architecture, or on the 8051 architecture or on some extremely low-power architecture such as MSP430, which use the analog or digital inputs to read the inputs.

- Output: the output is composed by a transceiver which is used to transmit the data on the WSN, usually to a network coordinator or gateway and by its RF frontend.

The microcontroller and the transceiver can in some cases be embedded in the same chip, for example as in the TI CC2538 System on Chip [64]; in this case any action performed on the microcontroller (e.g. reset or power) can directly affect the transceiver. WSNs are often built using pre-built modules rather than custom hardware; for example the Bluegiga BLE112 [65] or the Zolertia Z1 [66]. An example of a standalone transceiver is the Texas Instruments CC2520 [67].

4.2.2 Hardware category 2: a future proof choice

The hardware category adopted by the architecture is decisive in the determination of the maximum reachable Functional Safety Performance Level, or, as named in ISO25119, Agricultural Performance Level.

In fact the elements that determine the performance level are four:

- Hardware category.
- Software requirement level.
- Mean time to failure.
- Diagnostic coverage.

With the basic architecture adopted by the off-the-shelves nodes it is possible to reach an AgPL equal to b, that is a low safety level for most applications. The proposal of this thesis is the use of a hardware category 2 for the following reasons:

- Such a category allows to reach a performance level equal to “e” that covers the majority of safety related applications.
- It can reduce the time and effort necessary for software development because it allows the reuse of legacy code as a starting point.

Thanks to a hardware category two it is possible to prevent the inhibition of the WSN channel. In fact, with a category one, if a bubbling idiot fault happens, the faulty node will continue to transmit occupying the channel. With a category two, the safety micro

controller will detect the fault and stop the transmission, allowing all the other nodes to continue to work. A good choice, with this architecture, would be to add a timeout in the gateway that receives the data in order to detect when a node is faulty and has stopped its transmission.

4.2.3 Diagnostic

Because of the choice of a hardware category two, it is necessary to provide certain diagnostic capabilities. In particular:

- The logic must be able to diagnose both faults on an input and on an output.
- The test equipment must be able to diagnose faults on the logic.
- The test equipment must be able to override the commands of the logic on the output in order to command the recovery in case of fault on the logic itself.

According to these requirements, the diagnostic can be divided in three parts: diagnostic on the input, diagnostic on the output, diagnostic on the logic.

4.2.3.1 Input diagnosis

The faults on the inputs must be diagnosed by the logic. In order to perform such diagnosis two possible ways are available:

- To use diagnosable sensors;
- To use redundant sensors;

The first approach is based on the use of sensors that can be easily diagnosed. An example of that kind of sensors are the [0,5; 4,5] V analogic ones. These sensors has a characteristic curve that associates the physical entity to be measured to a voltage value, usually with a linear characteristic, but the range of voltage will always vary between the indicated bounds of 0,5 Volts minimum and 4,5 Volts maximum. This mean that it is possible to diagnose the faults of “short to ground” and “short to power supply”. These two faults are not the only ones that can happen. There is also the possibility of a drift in the sensor characteristic curve.

To be able to diagnose this kind of malfunction it is necessary to adopt the second approach indicated. For this approach are used two sensors for each physical entity to be measured. In this way, when a sensor begins to drift the characteristic curve it is possible to notice a difference between the read of the two transducers. The logic is able to reveal that something is going wrong and, depending on the entity of the drift, to perform an adequate recovery action. For example, it is possible to use the mean between the values in case of a small drift or to command a safe state of the system if the sensor is heavily corrupted.

The number of sensors to be used for each physical entity must be decided considering both the costs and the importance of the value to be measured. In fact, if two sensors are used the costs can be reduced but it will not be possible to determine which of them if faulty and, in case of a relevant difference in the read it is necessary to interrupt the safety function. If three sensors are used there are higher costs but it will be possible to continue to provide the safety function in case of a single fault because it will be possible to reveal which is the faulty one by using the major voting strategy. Focusing on the architecture analyzed in this thesis it is necessary to notice that to accomplish an hardware category 2 it is enough to simply reveal faults and non-reliable information. A good compromise to use a double transducer that can grant the coverage of both short circuit and drift errors, stopping the safety function when it is revealed.

A good precaution in case of use of redundant sensors is to apply the diversity principle by using sensors of different kind in order to avoid systematic failures.

4.2.3.2 Output diagnosis

The logic must be able to diagnose also the output. The logic has also to be able to perform actions in order to lead the system to a “safe state”, that is a state from which the electronic system considered does not cause issues or damages to the surrounding environment.

In the case of a wireless sensor network node the safe state consists in stopping the transceiver from transmitting or receiving data when a problem on it has been revealed.

The solution in order to diagnose the transceiver is as simple as efficient. Considering that, usually, a transceiver alternates periods of transmission or reception operation to idle ones, the current consumption of that component has the behavior reported in Figure 19 at the

first row. A faulty condition is characterized by a continuous high or low current consumption. The alternation of periods is no more visible, how shown in Figure 19 in the last two rows. In light of that consideration the diagnose can be performed by using a shunt resistor, which voltage value is acquired by the main micro controller, on the power supply of the transceiver.

If a fault on the transceiver is revealed the main micro controller must be able to shut it off in order to reach the safe state of the system. In this way the wireless channel will be left free for the other nodes and will not be blocked by a faulty node.

The solutions applicable to shut off the transceiver are different and depends on the architecture of the hardware adopted. In particular three different situations are possible:

- The main microcontroller and the transceiver are completely separated.
- The main microcontroller are on the same system on module;
- The main microcontroller embeds the transceiver;

In the first case it is possible to act directly on the power supply of the transceiver. In the second case the solution depends on how is implemented the system on module. In this situation usually there is an enable pin on which it is possible to act in order to stop the radio operations. In the third case it is possible to shut off the transceiver by acting on the internal registers of the MMC itself. The three solutions are ordered in accordance to the effectiveness of the solution applied. The first one is the safest, because it is possible to cut off the power supply to the transceiver. The second one is a middle way because it still is an hardware solution, but an enable pin is less strong than the cut off of the power supply. The third one is the weakest because it is, basically, a software solution.

It is possible to strengthen the solution thanks to the use of the safety micro controller according to what explained in 4.2.3.3.

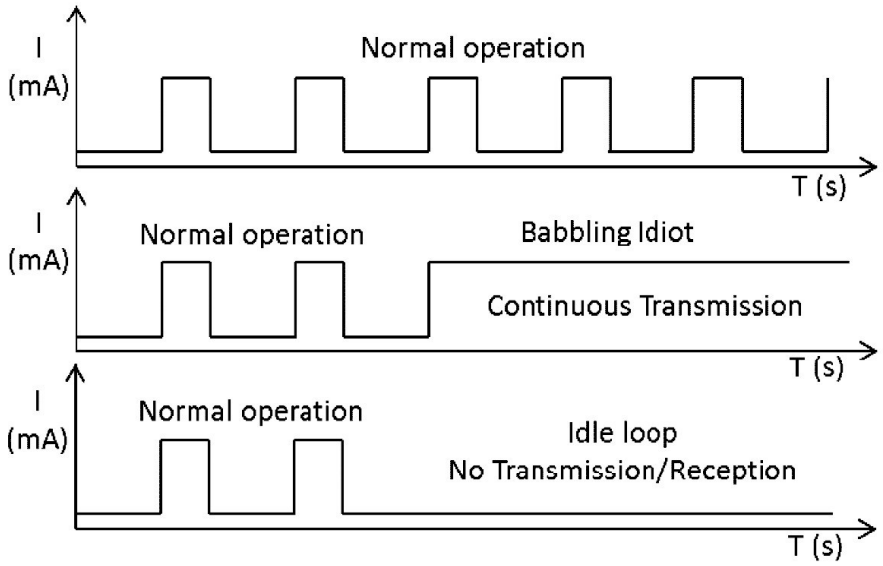

Figure 19. Power consumption during normal operation compared to consumption during transceiver problems

4.2.3.3 Logic diagnosis

The logic has to be diagnosed by the test equipment. Often the test equipment is realized with a second microcontroller named safety micro controller. The reason is that with a microcontroller it is possible to improve advanced strategies of diagnosis and recovery. The alternative would be to use a simpler component such as an external watchdog but this would mean to have limited possibility in control strategies and in the definition of the recovery actions.

The SMC will communicate with the MMC by using the available busses, like e.g. the I2C or the SPI. That communication is necessary for the safety microcontroller to test the correctness of the functions of the main microcontroller. These tests will include every test necessary, as, for example, a smart watchdog testing in which the safety microcontroller (uC) expects to receive a message periodically. It will use an ALU test by giving to the main uC a random number that will be elaborated through a series of fixed operation and the result will be sent back to the safety uC. Even the acquisition system of the main uC can be tested. . For example, if the transducer is acquired through an analog port it can be tested by letting the main uC acquire a shared input or an special input configurable by the supervisor uC, and sharing through the supervisory communication channel the read value. If any test performed by the safety uC will fail an appropriate recovery action will be

performed. In case of fault of the logic the recovery action shall be to switch off the transceiver by cutting off its power supply. In case of a module with main micro controller and transceiver or a microcontroller with embedded transceiver the recovery action will shut down also the MMC.

An example of the content of the message exchanged between the main and the safety microcontroller is reported in Table 6 and Table 7. Each parameter is represented by a bit of the message. If the bit is set it means that the relative error happened. In case that only a bit from one of the microcontrollers is high the transceiver should be shut off.

Table 6. Content of the message periodically sent from MMC to SMC.

Parameter	Detection
MMC ADC Error	Test executed periodically. A fixed value should be acquired and the result must be within a certain range.
MMC FLASH Test	An Hamming algorithm can be used to check the entire FLASH portion. The Hamming algorithm result should be periodically checked by the application. If the result is not what expected the error is set.
MMC RAM Test	An Hamming algorithm can be used to check the entire RAM portion. The Hamming algorithm result should be periodically checked by the application. If the result is not what expected the error is set.
Scheduler supervisor	The MMC can perform a check on its capability to schedule the tasks. Each time that a task is called a dedicated counter is incremented. At the moment of transmission of the periodical message the counter must have the expected value, otherwise the error is set.
MMC answer to SMC question (ALU check)	The MMC performs calculations on a random number received by the SMC to let it understand if some of the main microprocessor fundamental unit (branch, integer, load\store and floating point) are faulty. The calculation performed is the following: $\frac{[(x+2)*9-3]*10}{16} * \frac{14}{7} + \frac{FPU}{4}$ $FPU = uint16[287 * \frac{\left(\frac{1}{x}+\pi\right)}{\frac{\pi+2x}{4x}}]$ The result will be calculated also by the SMC and then compared.

Table 7. Content of the message periodically sent from SMC to MMC.

Parameter	Detection
SMC ADC Error	Test executed periodically. A fixed value should be acquired and the result must be within a certain range.
RAM Error	An Hamming algorithm can be used to check the entire RAM portion. The Hamming algorithm result should be periodically checked by the application. If the result is not what expected the error is set.
ROM Error	An Hamming algorithm can be used to check the entire FLASH portion. The Hamming algorithm result should be periodically checked by the application. If the result is not what expected the error is set.
SMC CPU Error	Three different tests can be performed: • Register test: bits over the register are checked for "stuck at" fault. • Addressing modes: all the addressing mode are tested by loading a constant application key. • Instruction set test: Some instruction has to be executed and compared with a known value

4.3 Experimental prototype

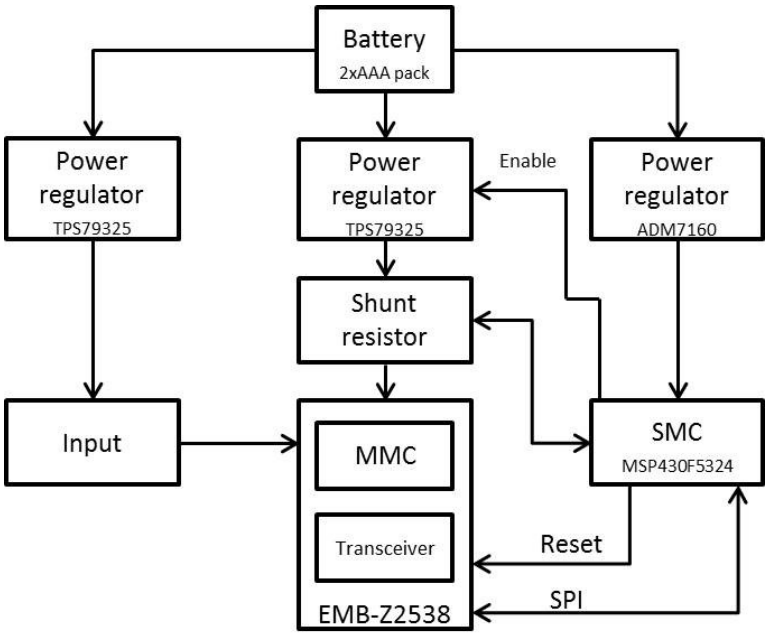

Figure 20. WSN module architecture

In [68] three different architectures have been described in order to realize a Wireless Sensor Node for safety-related applications. The architectures are realized depending on the available commercial module architectures:

- transceiver-only module,
- transceiver module with on-chip microcontroller,
- microcontroller with integrated transceiver.

The node described hereafter belongs to the third category.

In Figure 20 is shown the block diagram of the architecture adopted. It is easily attributable to an ISO25119's hardware category 2, as shown in Figure 21, where the main components are drown and coupled to the blocks of the standard's hardware categories.

The module is battery powered, it uses three LDOs in order to regulate the voltage that supplies the module, the SMC and input's pull-up network.

The module includes both a system-on-chip, ARM Cortex M3 based (Texas Instruments CC2538), with WSN transceiver and a RF frontend. As a consequence the module represents both the logic (MMC) and the output (transceiver + RF) and it's then classifiable as a System On Module (SOM).

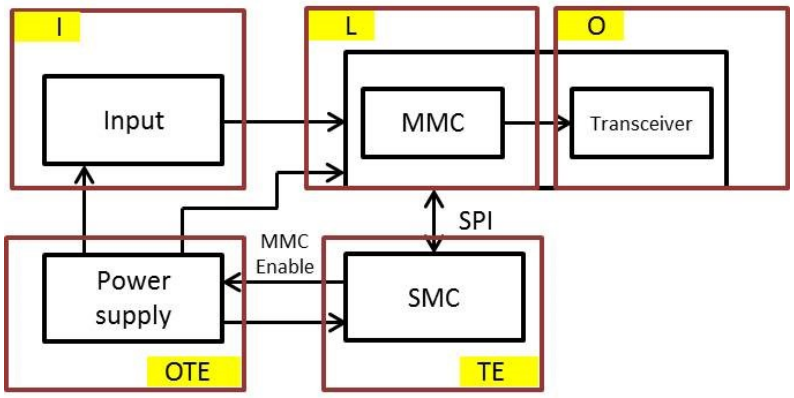

Figure 21. WSN module architecture division according to ISO25119 hardware category 2 definition.

The SOM is connected to a second microcontroller, with the function of safety microcontroller, through an SPI line that will allow a continuous check of the status of the

SOM logic. The checks will be software dependent, but the SPI ensure a great amount of diagnosis and functional controls.

For example, it will be possible to implement:

- a smart watchdog,
- a task monitor,
- ALU tests,
- ADC integrity tests,
- Clock integrity tests,
- memory integrity tests,

and so on, depending on the diagnostic coverage level requested by the application.

A second SMC to MMC diagnosis has been adopted: a shunt resistor has been placed between the module power regulator and the battery, in order to read the SOM current consumption.

Performing a continuous current consumption control, coupled with the SOM module functionality, it is possible to diagnose if the transceiver is in stall or blocked in a uncontrolled and/or dangerous status.

In fact, when the transceiver is transmitting or receiving data, the current consumption of the SOM is significantly increased with respect to other operating states. This allows the SMC to distinguish if the module is in idle state (no transmission nor reception) or in working state from the communication point of view.

If working state is read for too long time the module is considered in error. In fact the transmission status could affect the communication channel blocking the channel and inhibiting the other WSN nodes communication.

To accomplish to hardware category 2 requirements, SMC must be able to act directly on outputs, in case of an error on the logic, commanding the safe state. In this architecture the safe state is reached by switching off the transceiver, to free the communication channel.

To allow the SMC to directly control the output power status, it has the possibility to switch off MMC's power regulator by acting on its enable pin.

SMC is provided with another method to avoid dangerous WSN node conditions; in fact it has an output pin connected to MMC's reset input. The purposes of this connection are two:

- to be able to synchronize the devices at start up. In fact the phase of initialization of the two microcontrollers can be significantly different and it is necessary to have a signal to start both the microcontrollers at the same time.
- to provide a method to reach a safe state in case of errors of the MMC logic. In fact in case integrity tests previously listed fail, SMC shall be able to perform recovery strategies; resetting MMC, the SMC can restart the functionalities of the module in case an error on the logic occurs.

Figure 22 shows the prototype of a wireless sensor network node suitable for safety related applications realized at IMAMOTER-CNR accordingly to what has been described thus far.

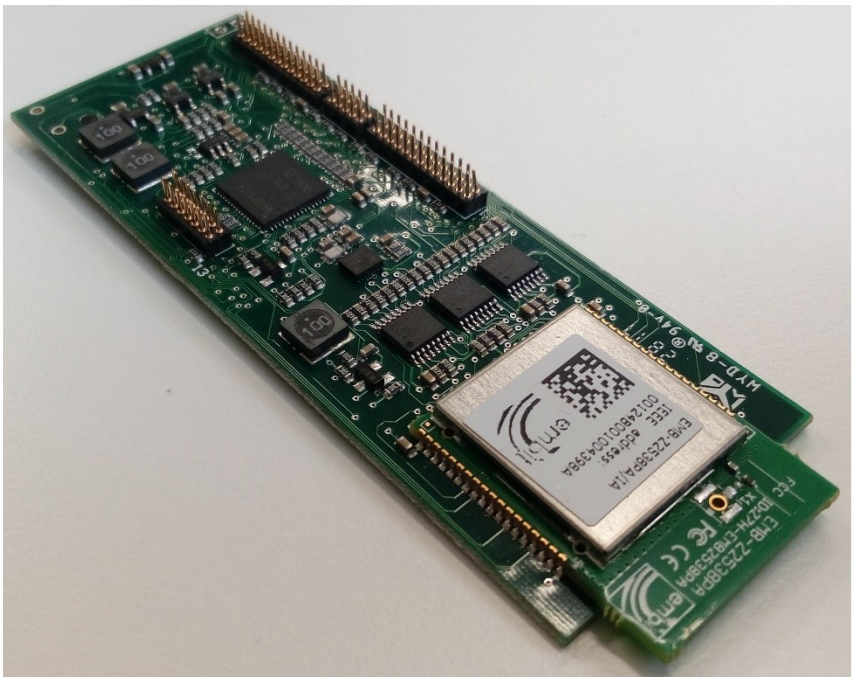

Figure 22. Prototype of wireless sensor network node suitable for safety related applications

4.3.1 Power supply

As described in the previous paragraph, the node is battery powered and it has three power regulators. Three LDOs (Low Drop-Out) regulators are used: one to supply the inputs' pull-up network, the second one to supply the wireless module and the last one to supply the SMC. The reasons to adopt three regulators are the following:

- The SOM module containing the radio transceiver must have a dedicated power regulator in order to allow the SMC to switch it off through the regulator enable pin, in case problems on the logic are revealed. Moreover the chosen LDO's support a maximum current of 200 mA and the peak current absorbed by the SOM module in transmission is 160 mA. So, connecting other devices to the same power regulator would put it near its limits, incrementing the stress of the component.
- The inputs do not have a shared regulator with SMC because they are external components, they can be wrongly connected or a short circuit can occur on the sensor cable, and so on, causing the power regulator to switch off itself, going in protection mode without diagnostic information available at network level.
- Using a separate power regulator for the SMC, in all the previously described error conditions the SMC can't be switched off reducing the Common Cause Failures (CCF) of the WSN Node. In summary there is not a common power regulator to separate the components external to the board from the internal ones.

LDO regulators has been chosen instead of switching regulators for several reasons: the first one is that, due to the low voltage provided by the couple of AAA batteries, it was not possible to have a big drop-out; the second one is that it is not possible to use switching power regulators because of their noisy nature. In fact, in WSN applications the power supply should be as clean as possible, as stated in EMB-Z2538 module's datasheet [69].

It has also been excluded to connect the components directly to the batteries because in this case the voltage would slowly vary while the batteries discharge, changing the analog inputs voltage reference.

Regarding the batteries it has been chosen to use two AAA. In this kind of applications the most relevant factor for the choice of batteries is their size, considering that WSN modules

have to be as small as possible. In this module it has not been possible to use Li-ion coin batteries because they can supply a very small current, while the transceiver can require up to 160 mA peak current during transmission.

The choice to use only two AAA batteries and not a bigger number has been performed for space saving reasons, but caused the power supply to be at a low voltage. In fact there is a 3V battery package regulated by LDOs to supply 2.5 V to the components.

4.3.2 Main microcontroller

The MMC is represented by a System-On-Chip that integrates both a transceiver and a Cortex M3. Its role is to acquire the inputs and to send data periodically through wireless sensor network.

This is the main component of the board and it is the EMB-Z2538. The reasons that led to the choice of this module are mainly three:

- it has an integrated RF front end,
- it has a Cortex M3 which is powerful enough to handle even complex operations,
- it has a crystal oscillator, which is fundamental to have the precise packet timing required for low-latency networks.

In fact, previous experience at CNR-IMAMOTER with Components Off The Shelf (COTS) based boards and an RC oscillator are the motivation for this chosen design option, since in this case the oscillator jitter limited the achievable real-time performance.

The integrated RF front end is very important in order to reduce design effort due to transmission lines. The choice of an architecture that integrates a Cortex M3 has also been performed in order to use a widely adopted architecture that ensure a good level of support with compilers, debuggers and so on.

For example, in addition to a standard C toolchain available from many vendors, even the GNAT Ada toolchain is available for ARM Cortex M, which GPL version can be freely downloaded from either Adacore or upstream GCC, and which includes support for the Ravenscar profile, offering services usually found in C based Real time Operating Systems

(RTOS). This means that even tools for high-integrity systems are already available and may be used in future.

Unfortunately, even if this module is based on Texas Instruments CC2538, the pins that can be connected are only a small number of analog and digital I/O.

In order to be able to connect a greater number of inputs to the module, it has been necessary to introduce some multiplexers with low series resistance.

4.3.3 Safety microcontroller

Another important component is the second microcontroller used for safety purposes. In this case an MSP430F5324 has been chosen. This microcontroller has one of the lowest power consumption available on the market. That aspect is very important in a WSN application.

Another feature that lead to the choice of this microcontroller is the 12 bits ADC, necessary in order to acquire with a good precision the voltage on the shunt resistor. In fact that resistor is very small to avoid an excessive drop out on its ends considering that the working voltage is already low because off the battery power supply.

4.3.4 Diagnostic coverage

In this paragraph will be performed an analysis of the diagnostic coverage level reached by the prototype designed and realized. The analysis will be performed according to ISO25119 norm, in particular to its Annex C in part 2.

The analysis will consider 4 parts: power supply, input, output and logic.

Table 8. Diagnostic coverage estimation for power supply, redrawn from ISO 25119:2010, Part 2, Annex C

Technique/measure	Diagnostic coverage	Example
Overvoltage protection	Low	Load dump protection on alternator
Voltage source control	High	Monitoring voltage and switching to alternative power supply, if required (operating limit is reached)

Technique/measure	Diagnostic coverage	Example
Power-down control	Medium	Monitoring key-switch (ignition-switch) voltage and initiating ECU shut-down, saving data to memory Shut system down when battery voltages drops down

Table 8 describes the measures that can be adopted for the diagnosis of power supply. In the WSN node realized it was not possible to adopt the voltage source control. It would not have been possible to provide an alternative power supply due to constraints in the dimensions of the node itself. In fact WSN applications are very sensitive to size constraints. In consideration of that has been adopted the measure of power down control. The MMC is supplied though a low-drop out regulator and the battery voltage is acquired through a voltage reducer realized using a resistive divider. A characteristic of this resistive divider is that it can be switched off thanks to a series MOSFET. The reason of the use of a MOSFET is that, without that component a route from power supply and ground would have been created, causing an unwanted power consumption decisive in wireless applications.

Table 9. Diagnostic coverage estimation for inputs and outputs, redrawn from ISO 25119:2010, Part 2, Annex C.

Technique/measure	Diagnostic coverage	Example
Failure detection by on-line monitoring	Medium	Monitoring feedback signal (within range)
Majority voter	High	Compare redundant signals
Tests by redundant hardware	Medium (periodic tests) High (continuous tests)	Test at start-up (medium) Watchdog (high)
Reference sensor	High	Measure a specified signal to check the ADC and/or ECU

For input and output must be applied Table 9.

In the experimental prototype the measures adopted for the input are two: failure detection by on-line monitoring and tests by redundant hardware. The inputs are all acquired through an analogic input pin and the sensors must be chosen in order to let their voltage vary within a certain range. Moreover, for each physical entity to be measured, two sensors must be applied in order to have redundant hardware and to be able to diagnose drifts in a sensor. This lead to a level of diagnostic coverage for the input that is high.

The output adopts only the failure detection by on-line monitoring that leads to a medium diagnostic coverage. In facts the output, that is represented by the transceiver and by the RF front-end is diagnosed through the monitoring of its current consumption that must be compliant with an expected range, as described in 4.2.3.2.

Table 10. Diagnostic coverage estimation for processing units, redrawn from ISO 25119:2010, Part 2, Annex C, Table C.3.

Technique	Diagnostic coverage	Example
Comparator	High	Compare output and/or synchronized deterministic intermediate data of redundant processors (see categories 2, 3 and 4)
Self-test by software: limited number of patterns (one channel)	High	Walking bit
Self-test by software (one channel)	Low	Internal watchdog
Self-test supported by hardware (one channel)	Medium	External watchdog

For processing units a second microcontroller has been adopted, with functions of smart watchdog. This leads to a medium diagnostic coverage.

In accordance to what established by ISO25119 the overall diagnostic coverage of the system is determined by the lowest one of the parts that compose the control chain. In this

prototype the lowest diagnostic coverage is a medium one. This means that the overall diagnostic coverage of the system is medium.

4.4 Test specification and report

In this section shall be reported several tables that describe the tests performed in order to verify the satisfaction of the requirements stated in the previous paragraphs of this chapter. In particular these tests aim to verify the correct behavior of the system in case of a fault occurrence in one of its main parts. It shall also be reported the results of the tests performed.

Table 11. Tests performed in order to verify the correct behaviour in case of fault on input.

Requirement	Approval procedure	Result
A fault on the input shall be detected. In case of fault, the transmission of data shall be stopped within 500 ms	The input pin shall be shorted to ground. The test shall be passed if the transmission of data stops within 500 ms from the fault injection.	Transmission stopped in 103 ms
	The input pin shall be shorted to Vcc. The test shall be passed if the transmission of data stops within 500 ms from the fault injection.	Transmission stopped in 107 ms.
	A voltage of 2V shall be applied to the first input pin, while a voltage of 1V shall be applied to the second one. In consequence of the difference, greater than the 10% of the maximum voltage accepted (3,3V), the error shall be detected. The test shall be passed if the transmission of data stops within 500 ms from the fault injection.	Transmission stopped in 235 ms.

Table 12. Tests performed in order to verify the correct behaviour in case of fault on SMC.

Requirement	Approval procedure	Result
A fault on SMC shall be detected by SMC. In case of fault, the transmission of data shall be stopped within 500 ms.	The SMC shall set, thanks to a software fault injection, the bit of SMC ADC Error (Table 7), in the message periodically exchanged with MMC. The test shall be passed if the transmission of data stops within 500 ms from the fault injection.	Transmission stopped in 349 ms.
	The SMC shall set, thanks to a software fault injection, the bit of RAM Error (Table 7), in the message periodically exchanged with MMC. The test shall be passed if the transmission of data stops within 500 ms from the fault injection.	Transmission stopped in 343 ms.
	The SMC shall set, thanks to a software fault injection, the bit of ROM Error (Table 7), in the message periodically exchanged with MMC. The test shall be passed if the transmission of data stops within 500 ms from the fault injection.	Transmission stopped in 343 ms
	The SMC shall set, thanks to a software fault injection, the bit of SMC CPU Error (Table 7), in the message periodically exchanged with MMC. The test shall be passed if the transmission of data stops within 500 ms from the fault injection.	Transmission stopped in 348 ms

Requirement	Approval procedure	Result
	The SMC shall stop sending the periodical message exchanged with MMC. The test shall be passed if the transmission of data stops within 500 ms from the fault injection.	Transmission stopped in 301 ms

Table 13. Tests performed in order to verify the correct behaviour in case of fault on MMC.

Requirement	Approval procedure	Result
A fault on MMC shall be detected by SMC. In case of fault, the transmission of data shall be stopped within 500 ms.	The MMC shall set, thanks to a software fault injection, the bit of MMC ADC Error (Table 7), in the message periodically exchanged with SMC. The test shall be passed if the transmission of data stops within 500 ms from the fault injection.	Transmission stopped in 347 ms.
	The MMC shall set, thanks to a software fault injection, the bit of MMC Flash test (Table 6), in the message periodically exchanged with SMC. The test shall be passed if the transmission of data stops within 500 ms from the fault injection.	Transmission stopped in 348 ms.
	The MMC shall set, thanks to a software fault injection, the bit of RAM Test (Table 6), in the message periodically exchanged with SMC. The test shall be passed if the transmission of data stops within 500 ms from the fault injection.	Transmission stopped in 341 ms

Requirement	Approval procedure	Result
	The MMC shall set, thanks to a software fault injection, the bit of Scheduler Supervisor Error (Table 6), in the message periodically exchanged with SMC. The test shall be passed if the transmission of data stops within 500 ms from the fault injection.	Transmission stopped in 345 ms
	The MMC shall send an incorrect value, thanks to a software fault injection, in the field of MMC answer to MMC question (Table 6), of the message periodically exchanged with SMC. The test shall be passed if the transmission of data stops within 500 ms from the fault injection.	Transmission stopped in 340 ms
	The SMC shall stop sending the periodical message exchanged with MMC. The test shall be passed if the transmission of data stops within 500 ms from the fault injection.	Transmission stopped in 303 ms

Table 14. Tests performed in order to verify the correct behaviour in case of fault on output.

Requirement	Approval procedure	Result
A fault on the transceiver shall be detected. In case of fault, the transmission of data shall be stopped within 500 ms.	A babbling idiot situation (Figure 19) shall be simulated through hardware fault injection, by stubbing to high the voltage on the shunt resistor. The test shall be passed if the transmission of data stops within 500 ms from the fault injection.	Transmission stopped in 430 ms.
	An idle loop situation (Figure 19) shall be simulated through hardware fault injection, by stubbing to high the voltage on the shunt resistor. The test shall be passed if the transmission of data stops within 500 ms from the fault injection.	Transmission stopped in 443 ms.

Chapter 5. Safe WSN for Human to Machine Interface

This last chapter will present an example of application that uses the WSN technology for a safety critical purpose [70]. In particular will be analyzed the aspects of gesture recognition as HMI for safety relevant applications. In order to realize this system, it will be used the prototype of WSN node presented in 4.3.

5.1 Gesture recognition

In the past, human machine interface (HMI) motivated many studies to develop systems that were able to transfer analog commands from the user to machine. However, many design solutions are still affected by user's comfort and performance limitations due to wire communications, long calibration procedures, and unsuitable hardware for safety critical application, low resistance to hand motions, and power supplies requiring cables or batteries. The challenge to deal with is the gesture acquisition for safety critical application, for example in the HMI for industrial field's hazardous area; usually the system does not have a direct acquisition of the gesture through the sensors but they try to reconstruct the movement from indirect measures. It is possible to do it by controlling the contraction of muscles and tendons, or with accelerometer and magnetometers or by video capturing. For example using a Myo armband, the gesture recognition is performed by complex algorithms that works on data captured by myographic sensors, gyroscope magnetometer and accelerometer. An alternative is to use a camera system, but it is possible that in some positions an image cannot be acquired. It is difficulty applicable also in many applications like inside a vehicle or inside a factory, where it is hard to install an efficient camera system to acquire detailed images of the user, typically because of the uncontrolled environment. These kind of systems are more suited to industrial automation systems, in a controlled environment. In this thesis, it is proposed a system to acquire directly the human motion and to have a safe and reliable gesture acquisition. This is possible with sensors installed in the gloves, acquired as an analog signal, which does not

need a complex algorithm of data elaboration. For this purpose the choice of the best kind of sensors and a proper installation are decisive. Sensors have more and more importance in each type of application, improving the precision, the comfort and the security of the user. Many applications, some years ago, were impossible to be realized. Nowadays they are growing thanks to sensors' evolution. One of the limits for the typical strain sensor is the use for milli-strain application. The costs are high and both the setting and the installation of the sensors are difficult. Using conductive thermoplastic elastomer (CTPE) sensors, it is possible to extend the use and the application of the normal strain gauge sensors and greatly cut the costs.

Another important aspect is the possibility to use a wireless sensor network node. For this purpose is useful the prototype described in 4.3 because this would permit to adopt the HMI realized also in safety related applications. The use of wireless technology is very attractive because it would reduce costs, eliminating or reducing the need of wiring, and, as a consequence, becoming a more scalable alternative for data gathering.

5.1.1 State of the art

A "smart glove" can be defined as an electronic system with a series of sensors embedded in a glove worn by the user, which is able to acquire and send data from the sensors to a receiver for the data processing.

In the 1970s and 1980s the first prototypes of glove interface of the history were proposed, but they were only limited to research experiments [71]. In 1982 the data glove [72] was commercialized and in the 1989 the power glove [73] was introduced as a remote control for Nintendo videogame console. Other prototypes of gloves for the human gesture recognition were introduced in recent years: some examples are the super glove [74] and the TUB-Sensor glove [75].

All the above-mentioned gloves are based on the measurement of flexion of user's fingers. In 1990s the trend in the way of measurement changed in favor of the use of accelerometers, magnetometers and/or a series of camera and LED to detect the movements of the user's hand. Examples of such gloves are the lightglove [76], the acceleGlove [75] and the sensing glove for robot control [77].

The most recent glove devices for HMI measure the finger flexion with different type of sensors, such the cyberGlove [78] that uses piezoresistive sensors, the humanglove [79] that uses Hall-effect sensors, myo armband that uses myographic sensors [80] or other devices that use different types of sensors and strategy like [81]–[86].

5.1.2 Smart glove

5.1.2.1 CTPE Sensors

The conductive thermoplastic elastomer used for this prototype is the one presented in [86]. This kind of sensors can be used in the milli-strain field and consists of a mixture of graphite and thermoplastic elastomer (TPE), which can work in a wide temperature range (from -40°C to 110°C). The CTPE sensors are extensible stripes obtained by extrusion that change their resistance with the elongation. In particular, they have a linear behavior in a limit that can vary from the 5% to the 7% of elongation. The resistance of the sensor can be managed by the geometry of the sensors (section and length) and the amount of graphite in the mixture. After several applications tests the CTPE sensors have demonstrated to be really user friendly in order to detect human movements. For this reason, they are particularly suitable for wearable applications.

5.1.2.2 Glove

The prototype here presented is based on a common mechanical glove with five CTPE sensors, one for each finger:

- Sensor 1 on thumb (Cyano on Figure 23),
- Sensor 2 on index finger (Green on Figure 23),
- Sensor 3 on middle finger (Red on Figure 23),
- Sensor 4 on ring finger (Yellow on Figure 23),
- Sensor 5 on pinky (Magenta on Figure 23).

Figure 23. The smart glove

Any sensor can bend freely according to the movement of the hand and detect different angles of the fingers thanks to the resistance variation of the sensors during the elongation. For the purpose of demonstration only five sensors have been used, but it's possible to extend the same methodology to the wrist, or other parts of the hand, in order to increase the number of recognizable gestures and their precision. All the sensors are monitored via a WSN node designed for safety critical applications (see 4.3). Each sensor, in order to limit its elongation, has a parallel and not extendable wire which role is to not permit to reach an elongation greater than 5%. To avoid the limiting of human movement every sensor have a serial rubber band that is stretched when the "safety" wire is in use (see Figure 24), in order to let the movement continue while the sensor remains to its full-scale value. One of the far end of the sensor is sewn over the knuckle, while the other one is sewn over the fingertip.

In order to diagnose the inputs a range check is performed, thanks to the fact that the sensor have a well determined acceptable range of resistance values. Therefore, when the value read is outside the range it is possible to reach a safe state by stopping every command and keeping the system in idle state until the fault is solved. In particular it is able to detect:

- Open load (sensor disconnected);
- Short circuit;
- Short to Vdd;
- Short to Ground;

In the complete project of the glove two sensors per finger are used in order to perform a coherence check and add the ability to diagnose a drift in the resistance value, even if it remains inside the accepted range.

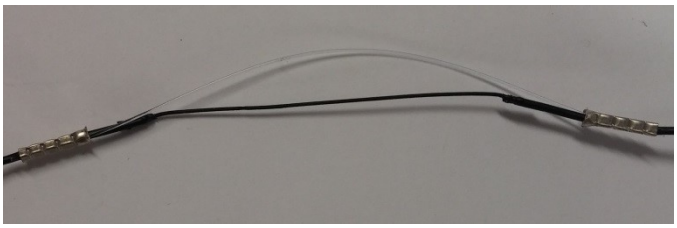

Figure 24. CTPE Sensor with safety wire.

5.1.2.3 Safe wireless communication

The WSN communication system is based on IEEE 802.15.4e LLDN, with a custom transmission protocol, specific for safety-critical data. The standard MAC protocol is not sufficiently protected against transmission errors, since for example timing errors are not detected, and most importantly is handled at MAC layer, while the diagnosis of errors has to be performed at application layer. The MAC layer mechanisms, of course, can aid in the detection of errors such as a corrupted message or an out-of-order message, but it is still possible that the application is in a faulty state and the message are transmitted anyway (e.g. the so-called babbling idiot). For this reason, an additional safety layer is required at application level, adding a set of countermeasures which role is to increase the error detection capability if compared to the standard IEEE 802.15.4. The added measures are:

- Timeout: it is defined a time interval in which it is expected the reception of exactly one packet. In case during this interval are received more messages or no message is received it is considered an error.
- Running number: every packet generated has an identification number. This number changes with a pre-determined function for each given sequence of packets.
- Explicit reception acknowledges: the receiver will send an explicit acknowledge after the reception of each message. If the acknowledge is not received it is considered an error.
- Additional integrity check: an integrity check is added in data payload in order to further reduce the residual error probability of the message.

These measures, as it possible to see in Table 1 are enough to cover every transmission error considered by safety standards. The safe transmission protocol maintains its error-detection capabilities also in case of multi-hop transmission, i.e. if the receiver of the sensor data is not the same device as the PAN coordinator. This is another reason to use an application-level safety protocol instead of a MAC layer one, and is therefore well suited for complex network architectures. The packet transmission rate is of a message every 15 ms. In this way a single packet is used to transmit the raw ADC value of the 5 sensors. This transmission rate is feasible on a LLDN network as long as the number of nodes in the WSN is low; to collect the experimental data only one sensor node is present, and it is foreseeable that a small number of nodes will be present on the Body WSN. If a higher transmission rate is required, it will be possible to aggregate more sensor readings in a single packet.

5.1.2.4 Experimental setup

In order to run the tests, the prototype described in 5.1.2.2 has been wear by a group of persons in order to generate the gestures. The sensors where connected to the safe wireless sensor node described in 4.3, according to the scheme shown in Figure 25. Another safe wireless sensor node has been used to receive the data. This node has been connected through USB to a PC running a graphical plot of the raw voltage data acquired and the gesture recognition algorithm.

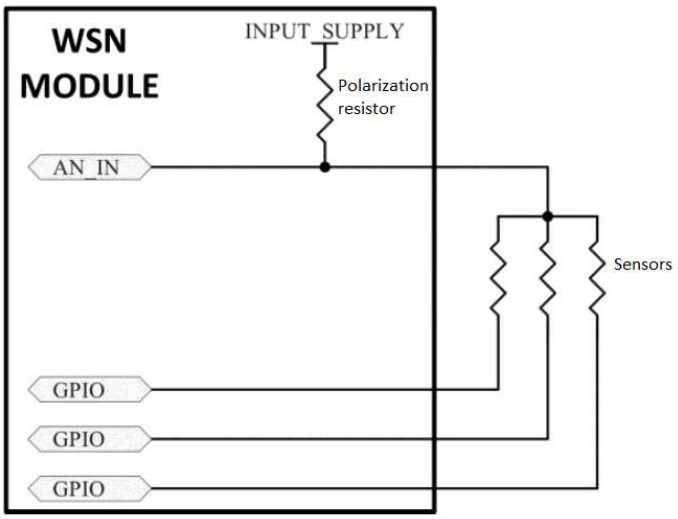

Figure 25. Input acquisition layout

The value acquired from the sensors of the user's hand are sent to the receiving station without performing any elaboration. The choice to leave the elaboration effort in receiver side is due to a couple of reasons. The first is that, in this way, the sender node, that is battery powered (while the receiver is not) can consume less power. The second is that the sender nodes can be used as is for any application, by implementing the required gesture recognition algorithm only in the receiving station.

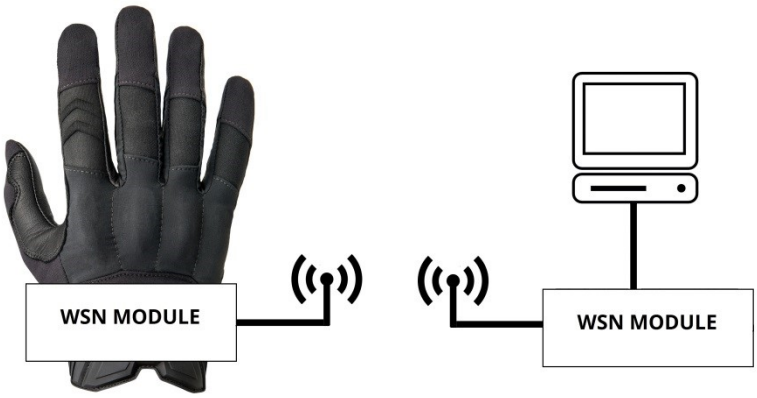

Figure 26. Setup of the WSN used for sensor reading.

5.1.2.5 Gesture recognition

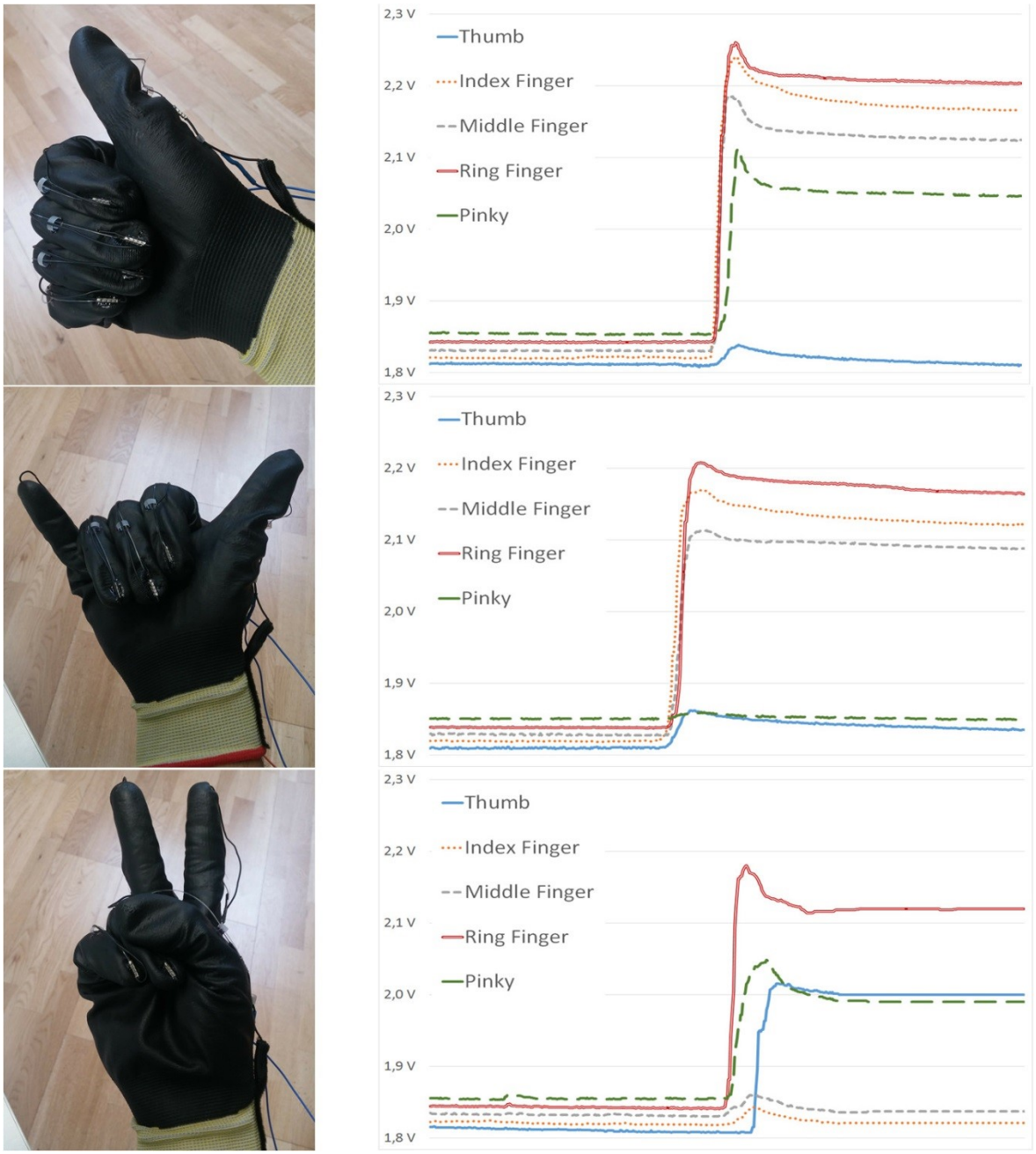

Figure 27. Gestures and sensors' readings

The PC, with a QT-based software, plots and processes the acquired and received data in order to perform the gesture recognition. Thanks to the cleanliness of the signal received, the algorithm used to recognize the gestures is quite simple. The algorithm for the data elaboration and the consequent movement detection are based on a specific threshold for

each finger, empirically determined by studying different gestures on different persons. When the acquired value becomes higher than the threshold, the flag of the relative sensor changes its status, in order to recognize a gesture as a determined combination of finger's flags. The reason why each finger has its own threshold is their different anatomy. As example it is possible to see, in Figure 27, the different behaviors of the sensors for the same movement of the fingers (finger completely closed and finger completely opened).

In Table 15 are reported the matchings between the gesture to be recognized and the status of the different finger's sensors. An 'H' indicates that the sensors is over threshold and 'L' indicates that it is under threshold. The CTPE sensors work on stretch, that means that when a finger is bent the resistive value increases, while, when the fingers gets back to the neutral position its value decreases. Figure 27 shows three simple gestures. As explained in 5.1.2.2, each colored line corresponds to a different finger. The tracks have been shifted through a software offset in order to make them easily readable, following sensors' order. As shown in the figures, the noise with the finger both in neutral and in bent position is very low, nearly absent. Moreover the sensors follow very precisely hand's movement.

Table 15. Gesture match table

	Thumb	Index finger	Middle finger	Ring finger	Pinky
Gesture 1	L	H	H	H	L
Gesture 2	H	L	L	H	H
Gesture 3	L	H	H	H	L

5.1.2.6 Future applications

At the moment of the writing of this thesis the HMI has not yet been used in any real application. It has only been used in laboratory's tests. The real application for which it has been conceived is still in process of realization. At the end of the realization this HMI will be used to command a hydraulic grabber, that is a safety critical agricultural device.

Chapter 6. Conclusions

The cyber-security aspects deriving from the introduction of a wireless connection to vehicular networks for agricultural machines has been analyzed, keeping an eye on the requirements that comes from the compliancy to the safety standards applicable to this field. The conclusion of this analysis has been that CAN bus has not enough capabilities to satisfy the requirements of both safety and security. A technological leap forward must be performed in order to adapt vehicular networks to the requirements of the present day market.

As exposed, Ethernet based networks are considered the future of in-vehicle communication, accordingly to many technical groups created for this purpose. An analysis has been performed on the ability of Ethernet to satisfy both safety and security requirements and the result has been its compliance to the performance necessary. For this reason it has been realized a gateway CAN-Ethernet, that has been described in this thesis, which role is to allow the necessary transition from the present day network to the future one by realizing hybrid networks, requested considering the life-time of agricultural vehicles.

Considering the fact that in automotive the wireless technology has become to be adopted for safety critical applications (e.g. remote parking of BMW), and considered that automotive technological trends are the precursor for agricultural ones, it has been analyzed how it can be possible to adopt an attractive technology, such as WSN, for safety relevant applications in agricultural networks. For this purpose have been analyzed both hardware and software design rules that should be respected and has been presented a prototype of wireless sensor network node realized respecting these rules and, for this reason, suitable for safety relevant applications.

Finally it has been shown an application, based on the prototype of node realized, that is is a human to machine interface to be used in safety relevant implementations. For the development of this HMI have also been analyzed the aspects related to the problem of

gesture recognition for safety applications, considering that the instrument to be used to send the commands is a sensorized glove, named “Smart Glove”. The main feature of this realization is the direct acquisition of the gestures, that is a more reliable solution if compared to those present on the market that use indirect measures. For this reason it is more adapt for safety critical applications.

Chapter 7. Bibliography

[1] IEC, 'IEC 61508:2010: Functional safety of electrical/electronic/ programmable electronic safety- related systems'. International Electotechnical Commission, 2010.

[2] D. J. Smith and K. G. L. Simpson, *Functional safety: a straightforward guide to IEC 61508 and related standards*. Amsterdam; London: Elsevier Butterworth-Heinemann, 2004.

[3] ISO, 'ISO 12100:2010: Safety of machinery – General principles for design – Risk assessment and risk reduction'. International Organization for Standardization, 2010.

[4] ISO, 'ISO 15998:2008: Earth-moving machinery – Machine-control systems (MCS) using electronic components – Performance criteria and tests for functional safety'. International Organization for Standardization, 2008.

[5] ISO, 'ISO 25119:2010: Tractors and machinery for agriculture and forestry — Safety-related parts of control systems (all parts)'. International Organization for Standardization, 2010.

[6] Motor Industry Software Reliability Association, Ed., *MISRA C:2012: guidelines for the use of the C language in critical systems*. Nuneaton: Misra, 2013.

[7] *Reliability data handbook - Universal model for reliability prediction of electronics components, PCBs and equipment*. Geneva: IEC, International Electrotechnical Commission, 2004.

[8] L. Dariz, M. Selvatici, M. Ruggeri, G. Costantino, and F. Martinelli, 'Trade-Off Analysis of Safety and Security in CAN bus communication', in *5th IEEE International Conference on models and technologies for intelligent transportation systems*, Naples (Italy), 2017.

[9] M. Selvatici, M. Ruggeri, and L. Dariz, 'Advanced gateway services for real-time in-vehicle ethernet network', in *Industrial Technology (ICIT), 2015 IEEE International Conference on*, 2015, pp. 1826–1831.

[10] ISO, 'ISO 11898:2015: Road vehicles – Controller area network (CAN)'. International Organization for Standardization, 2015.

[11] ISO, 'ISO 11783:2014: Tractors and machinery for agriculture and forestry – Serial control and communications data network (All Parts)'. International Organization for Standardization, 2014.

[12] C. Miller and C. Valasek, 'Adventures in Automotive Networks and Control Units', in *DEFCON 23*, Las vegas, 2015.

[13] 'Hackers Remotely Kill a Jeep on the Highway—With Me in It | WIRED'. [Online]. Available: https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/. [Accessed: 13-Jun-2017].

[14] K. Koscher *et al.*, 'Experimental Security Analysis of a Modern Automobile', 2010, pp. 447–462.

[15] S. Checkoway *et al.*, 'Comprehensive Experimental Analyses of Automotive Attack Surfaces', in *Proceedings of the 20th USENIX Conference on Security*, Berkeley, CA, USA, 2011, pp. 6–6.

[16] C. Miller and C. Valasek, ‘A Survey of Remote Automotive Attack Surfaces’, Tech., 2014.
[17] ‘After Jeep Hack, Chrysler Recalls 1.4M Vehicles for Bug Fix | WIRED’. [Online]. Available: https://www.wired.com/2015/07/jeep-hack-chrysler-recalls-1-4m-vehicles-bug-fix/. [Accessed: 13-Jun-2017].
[18] M. Al-Shurman, S.-M. Yoo, and S. Park, ‘Black Hole Attack in Mobile Ad Hoc Networks’, in *Proceedings of the 42Nd Annual Southeast Regional Conference*, New York, NY, USA, 2004, pp. 96–97.
[19] ISO, ‘ISO 15765:2011: Road vehicles — Diagnostic communication over Controller Area Network (DoCAN)’. International Organization for Standardization, 2011.
[20] SAE, ‘SAE J1939: Recommended Practice for a Serial Control and Communications Vehicle Network’. Society of Automotive Engineers, 2013.
[21] ISO, ‘ISO 11783:2014: Tractors and machinery for agriculture and forestry – Serial control and communications data network - Part 3: Data Link Layer’. International Organization for Standardization, 2014.
[22] H. Kleinknecht, ‘CAN Calibration Protocol Version 2.1’, Technical, 1999.
[23] ISO, ‘SO 14230:2013: Road Vehicles - Diagnostic systems - Keyword Protocol 2000 - Part 3: Application Layer’. International Organization for Standardization, 2013.
[24] A. Van Herrewege, D. Singelee, and I. Verbauwhede, ‘CANAuth - A Simple, Backward Compatible Broadcast Authentication Protocol for CAN bus’, presented at the ECRYPT Workshop on Lightweight Cryptography, 2011, pp. 229–235.
[25] A.-I. Radu and F. D. Garcia, ‘LeiA: A Lightweight Authentication Protocol for CAN’, in *Computer Security – ESORICS 2016*, vol. 9879, I. Askoxylakis, S. Ioannidis, S. Katsikas, and C. Meadows, Eds. Cham: Springer International Publishing, 2016, pp. 283–300.
[26] P. Vasile, B. Groza, and S. Murvay, ‘Performance analysis of broadcast authentication protocols on CAN-FD and FlexRay’, 2015, pp. 1–8.
[27] S. Horihata, Y. Miyashita, N. Adachi, H. Takada, Y. Matsubara, and R. Kurachi, ‘CaCAN - Centralized Authentication System in CAN’, presented at the Embedded Security in Cars (ESCAR), 2014.
[28] IEEE Computer Society, LAN/MAN Standards Committee, Institute of Electrical and Electronics Engineers, and IEEE-SA Standards Board, *IEEE standard for local and metropolitan area networks-- bridges and bridged networks*. 2014.
[29] S. Vaudenay, ‘Security Flaws Induced by CBC Padding—Applications to SSL, IPSEC, WTLS...’, in *International Conference on the Theory and Applications of Cryptographic Techniques*, 2002, pp. 534–545.
[30] C.-W. Lin, Q. Zhu, and A. Sangiovanni-Vincentelli, ‘Security-Aware Modeling and Efficient Mapping for CAN-Based Real-Time Distributed Automotive Systems’, *IEEE Embedded Systems Letters*, vol. 7, no. 1, pp. 11–14, Mar. 2015.
[31] C. Petit, F.-X. Standaert, O. Pereira, T. G. Malkin, and M. Yung, ‘A block cipher based pseudo random number generator secure against side-channel key recovery’, in *Proceedings of the 2008 ACM symposium on Information, computer and communications security*, 2008, pp. 56–65.
[32] F. Schiller and T. Mattes, ‘Analysis of nested CRC with additional net data by means of stochastic automata for safety-critical communication’, in *Factory Communication Systems, 2008. WFCS 2008. IEEE International Workshop on*, 2008, pp. 295–304.

[33] J. Charzinski, ‘Performance of the error detection mechanisms in CAN’, in *Proceedings of the 1st International CAN Conference*, 1994, vol. 9.
[34] J. Unruh, H.-J. Mathony, and K.-H. Kaiser, ‘Error detection analysis of automotive communication protocols’, SAE Technical Paper, 1990.
[35] M. Dell *et al.*, ‘High-speed Curve25519 on 8-bit, 16-bit, and 32-bit microcontrollers’, *Designs, Codes and Cryptography*, vol. 77, no. 2–3, pp. 493–514, Dec. 2015.
[36] H. Krawczyk, M. Bellare, and R. Canetti, ‘HMAC: Keyed-Hashing for Message Authentication’. .
[37] J. H. Saltzer, D. P. Reed, and D. D. Clark, ‘End-to-end arguments in system design’, *ACM Transactions on Computer Systems*, vol. 2, no. 4, pp. 277–288, Nov. 1984.
[38] C. Ferraresi, M. Dian, G. Malaguti, and M. Ruggeri, ‘Isobus Over Ethernet: A First Implementation’, in *Proceedings of 7Th FPNI Phd Symposium On Fluid Power*, Modena, Italy, 2012, pp. 751–766.
[39] M. Ruggeri, G. Malaguti, and M. Dian, ‘Real Time Ethernet For Heavy- Duty Vehicle Powertrain Control’, presented at the 12th European Regional Conference of the International Society for Terrain-Vehicle Systems, Pretoria, South Africa, 2012.
[40] M. Ruggeri, G. Malaguti, and M. Dian, ‘Sae j 1939 over real time ethernet: The future of heavy duty vehicle networks’, SAE Technical Paper, 2012.
[41] M. Ruggeri, G. Malaguti, and M. Dian, ‘The future of heavy-duty networking’, *Off Highway Engineering SAE International*, pp. 9–17.
[42] P. Mondal and V. Tewari, ‘Present status of precision farming: A review’, *International Journal of Agricultural Research*, vol. 2, no. 1, pp. 1–10, 2007.
[43] H. Krawczyk, M. Bellare, and R. Canetti, ‘RFC 791 – Internet Protocol’. RFC Editor, 1997.
[44] R. Droms, *RFC 2131 - Dynamic Host Configuration Protocol*. Internet Engineering Task Force, 1997.
[45] ‘STM32F407/417 - STMicroelectronics’. [Online]. Available: http://www.st.com/en/microcontrollers/stm32f407-417.html?querycriteria=productId=LN11. [Accessed: 14-Jun-2017].
[46] ‘ChibiOS free embedded RTOS - ChibiOS Homepage’. [Online]. Available: http://www.chibios.org/dokuwiki/doku.php. [Accessed: 12-Jun-2017].
[47] A. Dunkels, ‘Design and Implementation of the lwIP TCP/IP Stack’, *Swedish Institute of Computer Science*, vol. 2, p. 77, 2001.
[48] L. Dariz, M. Selvatici, and M. Ruggeri, ‘Evaluation of operating system requirements for safe Wireless Sensor Networks’, in *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, 2016, pp. 5671–5676.
[49] M. Selvatici, L. Dariz, and M. Ruggeri, ‘A safety approach to wireless sensor network modules’, in *2016 IEEE 25th International Symposium on Industrial Electronics (ISIE)*, 2016, pp. 1184–1189.
[50] M. Selvatici, L. Dariz, and M. Ruggeri, ‘SWSN: A safe wireless sensor network module for fail-silent systems’, in *2017 IEEE International Conference on Industrial Technology (ICIT)*, 2017, pp. 1372–1377.
[51] P. Anderson, ‘Coding standards for high-confidence embedded systems’, in *Military Communications Conference, 2008. MILCOM 2008. IEEE*, 2008, pp. 1–7.

[52] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, 'System architecture directions for networked sensors', *ACM SIGOPS operating systems review*, vol. 34, no. 5, pp. 93–104, 2000.
[53] A. Dunkels, B. Gronvall, and T. Voigt, 'Contiki-a lightweight and flexible operating system for tiny networked sensors', in *Local Computer Networks, 2004. 29th Annual IEEE International Conference on*, 2004, pp. 455–462.
[54] H. Abrach *et al.*, 'MANTIS: System support for multimodal networks of in-situ sensors', in *Proceedings of the 2nd ACM international conference on Wireless sensor networks and applications*, 2003, pp. 50–59.
[55] C.-C. Han, R. Kumar, R. Shea, E. Kohler, and M. Srivastava, 'A dynamic operating system for sensor nodes', in *Proceedings of the 3rd international conference on Mobile systems, applications, and services*, 2005, pp. 163–176.
[56] R. Barr *et al.*, 'On the need for system-level support for ad hoc and sensor networks', *ACM SIGOPS Operating Systems Review*, vol. 36, no. 2, pp. 1–5, 2002.
[57] Q. Cao, T. Abdelzaher, J. Stankovic, and T. He, 'The liteos operating system: Towards unix-like abstractions for wireless sensor networks', in *Information Processing in Sensor Networks, 2008. IPSN'08. International Conference On*, 2008, pp. 233–244.
[58] T. Hofmeijer, S. Dulman, P. Jansen, and P. J. Havinga, 'AmbientRT-real time system software support for data centric sensor networks', in *Intelligent Sensors, Sensor Networks and Information Processing Conference, 2004. Proceedings of the 2004*, 2004, pp. 61–66.
[59] W. P. McCartney and N. Sridhar, 'Abstractions for safe concurrent programming in networked embedded systems', in *Proceedings of the 4th international conference on Embedded networked sensor systems*, 2006, pp. 167–180.
[60] T. Alliance, 'TinyOS 2.1 adding threads and memory protection to TinyOS', in *Proceedings of the 6th ACM conference on Embedded network sensor systems*, 2008, pp. 413–414.
[61] N. Cooprider, W. Archer, E. Eide, D. Gay, and J. Regehr, 'Efficient memory safety for TinyOS', in *Proceedings of the 5th international conference on Embedded networked sensor systems*, 2007, pp. 205–218.
[62] 'IvyWiki : Home Page'. [Online]. Available: http://ivy.cs.berkeley.edu/ivywiki/index.php. [Accessed: 04-Sep-2017].
[63] E. A. Brewer, J. Condit, B. McCloskey, and F. Zhou, 'Thirty Years Is Long Enough: Getting Beyond C.', in *HotOS*, 2005.
[64] 'CC2538 A Powerful System-On-Chip for 2.4-GHz IEEE 802.15.4-2006 and ZigBee Applications | TI.com'. [Online]. Available: http://www.ti.com/product/CC2538. [Accessed: 12-Jun-2017].
[65] 'Bluegiga BLE112 Bluetooth Smart Module | Silicon Labs'. [Online]. Available: http://www.silabs.com/products/wireless/bluetooth/bluetooth-low-energy-modules/ble112-bluetooth-smart-module. [Accessed: 12-Jun-2017].
[66] 'Z1 | Zolertia'. [Online]. Available: http://zolertia.io/z1. [Accessed: 12-Jun-2017].
[67] 'CC2520 Second generation 2.4 GHz ZigBee/IEEE 802.15.4 RF transceiver | TI.com'. [Online]. Available: http://www.ti.com/product/CC2520. [Accessed: 12-Jun-2017].

[68] M. Selvatici, L. Dariz, and M. Ruggeri, 'A safety approach to wireless sensor network modules', in *2016 IEEE 25th International Symposium on Industrial Electronics (ISIE)*, 2016, pp. 1184–1189.
[69] 'EMB-Z2538, Embit datasheet'. [Online]. Available: http://www.embit.eu/wp-content/datasheets/EMB-Z2538PA-datasheet-latest.pdf. [Accessed: 12-Jun-2017].
[70] A. Vecchiattini, M. Selvatici, L. Dariz, and M. Ruggeri, 'A WSN HMI glove for safety critical applications in hazardous areas', in *IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society*, 2017, pp. 8464–8470.
[71] D. J. Sturman and D. Zeltzer, 'A survey of glove-based input', *IEEE Computer graphics and Applications*, vol. 14, no. 1, pp. 30–39, 1994.
[72] D. L. Gardner, 'The power glove', *Des. News*, vol. 45, pp. 63–68, 1989.
[73] H. Eglowstein, 'Reach out and touch your data.', *Byte*, vol. 15, no. 7, pp. 283–289, 1990.
[74] J. LaViola, 'A survey of hand posture and gesture recognition techniques and technology', *Brown University, Providence, RI*, vol. 29, 1999.
[75] J. L. Hernandez-Rebollar, N. Kyriakopoulos, and R. W. Lindeman, 'The AcceleGlove: a whole-hand input device for virtual reality', in *ACM SIGGRAPH 2002 conference abstracts and applications*, 2002, pp. 259–259.
[76] B. Howard and S. Howard, 'Lightglove: Wrist-worn virtual typing and pointing', in *Wearable Computers, 2001. Proceedings. Fifth International Symposium on*, 2001, pp. 172–173.
[77] J. Ryu, Y. Kim, H. O. Wang, and D. H. Kim, 'Wireless control of a board robot using a sensing glove', in *Ubiquitous Robots and Ambient Intelligence (URAI), 2014 11th International Conference on*, 2014, pp. 423–428.
[78] C. Burdea Grigore and P. Coiffet, *Virtual reality technology*. London: Wiley-Interscience, 1994.
[79] L. Dipietro, A. M. Sabatini, and P. Dario, 'Evaluation of an instrumented glove for hand-movement acquisition', *Journal of rehabilitation research and development*, vol. 40, no. 2, p. 179, 2003.
[80] 'Myo Gesture Control Armband | Wearable Technology by Thalmic Labs'. [Online]. Available: https://www.myo.com/. [Accessed: 04-Sep-2017].
[81] T. Kuroda, Y. Tabata, A. Goto, H. Ikuta, M. Murakami, and others, 'Consumer price data-glove for sign language recognition', in *Proc. of 5th Intl Conf. Disability, Virtual Reality Assoc. Tech., Oxford, UK*, 2004, pp. 253–258.
[82] C. Park, J. Bae, and I. Moon, 'Development of wireless data glove for unrestricted upper-extremity rehabilitation system', in *ICCAS-SICE, 2009*, 2009, pp. 790–793.
[83] G. Saggio, S. Bocchetti, C. A. Pinto, and G. Orengo, 'Wireless data glove system developed for HMI', in *Applied Sciences in Biomedical and Communication Technologies (ISABEL), 2010 3rd International Symposium on*, 2010, pp. 1–5.
[84] G. Saggio, B. Gupta, M. Quagliani, M. De Sanctis, E. Cianca, and T. Rossi, 'Power efficient wireless connectivity of a wearable data glove', in *Applied Sciences in Biomedical and Communication Technologies (ISABEL), 2010 3rd International Symposium on*, 2010, pp. 1–5.
[85] G. De Pasquale, S.-G. Kim, and D. De Pasquale, 'GoldFinger: Wireless Human–Machine Interface With Dedicated Software and Biomechanical Energy Harvesting

System', *IEEE/ASME Transactions on Mechatronics*, vol. 21, no. 1, pp. 565–575, 2016.

[86] A. Pecora *et al.*, 'Strain gauge sensors based on thermoplastic nanocomposite for monitoring inflatable structures', in *Metrology for Aerospace (MetroAeroSpace), 2014 IEEE*, 2014, pp. 84–88.

Chapter 8. Author's publication list

8.1 Journals

[J1] G. Malaguti, M. Ruggeri, L. Dariz, and M. Selvatici, 'In-Tractor Cloud: A Vision of Service-Oriented System Design Enabled by High-Speed In-Vehicle Networks for a Safer Task and Machine Management', 2016.

[J2] M. Ruggeri, P. Marani, and M. Selvatici, 'Functional Safety Oriented Design of an Electro-Hydraulic Stationary Braking System', 2016.

8.2 Conferences

[C1] L. Dariz, M. Ruggeri, and M. Selvatici, 'A static microcode analysis tool for programmable load drivers', in 2015 IEEE 15th International Working Conference on Source Code Analysis and Manipulation (SCAM), 2015, pp. 265–270.

[C2] M. Selvatici, M. Ruggeri, and L. Dariz, 'Advanced gateway services for real-time in-vehicle ethernet network', in 2015 IEEE International Conference on Industrial Technology (ICIT), 2015, pp. 1826–1831.

[C3] M. Selvatici, L. Dariz, and M. Ruggeri, 'A safety approach to wireless sensor network modules', in 2016 IEEE 25th International Symposium on Industrial Electronics (ISIE), 2016, pp. 1184–1189.

[C4] L. Dariz, M. Selvatici, and M. Ruggeri, 'Evaluation of operating system requirements for safe Wireless Sensor Networks', in IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society, 2016, pp. 5671–5676.

[C5] L. Dariz, M. Selvatici, M. Ruggeri, and R. Abrishambaf, 'Smart and wearable wireless sensors: Scenario analysis and communication issues', in 2016 IEEE International Conference on Industrial Technology (ICIT), 2016, pp. 1938–1943.

[C6] M. Selvatici, L. Dariz, and M. Ruggeri, 'SWSN: A safe wireless sensor network module for fail-silent systems', in 2017 IEEE International Conference on Industrial Technology (ICIT), 2017, pp. 1372–1377.

[C7] L. Dariz, M. Selvatici, M. Ruggeri, G. Costantino, and F. Martinelli, 'Trade-off analysis of safety and security in CAN bus communication', in 2017 5th IEEE International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS), 2017, pp. 226–231.

8.3 CNR internal reports and project deliverables

[D1] Massimiliano Ruggeri, Michele Selvatici, and Luca Dariz, 'Code Review Guidelines', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367632.

[D2] Luca Dariz, Massimiliano Ruggeri, Michele Selvatici, (in collaboration with the customer), 'ECU Compliance drawing', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367683.
[D3] Luca Dariz, Massimiliano Ruggeri, Michele Selvatici, (in collaboration with the customer), 'ECU Environmental Test Specifications', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367675.
[D4] Luca Dariz, Massimiliano Ruggeri, Michele Selvatici, (in collaboration with the customer), 'ECU Firmware Specifications', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367672.
[D5] Luca Dariz, Massimiliano Ruggeri, Michele Selvatici, (in collaboration with the customer), 'ECU Installation prescriptions', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367684.
[D6] Luca Dariz, Massimiliano Ruggeri, Michele Selvatici, (in collaboration with the customer), 'ECU RTOS Manual', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367677.
[D7] Luca Dariz, Massimiliano Ruggeri, Michele Selvatici, (in collaboration with the customer), 'ECU Software Architecture', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367671.
[D8] Luca Dariz, Massimiliano Ruggeri, Michele Selvatici, (in collaboration with the customer), 'ECU Software Development Process', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367673.
[D9] Luca Dariz, Massimiliano Ruggeri, Michele Selvatici, (in collaboration with the customer), 'Firmware Implementation Manual', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367613.
[D10] Luca Dariz, Massimiliano Ruggeri, Michele Selvatici, (in collaboration with the customer), 'Firmware User Manual', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367611.
[D11] Luca Dariz, Massimiliano Ruggeri, Michele Selvatici, (in collaboration with the customer), 'Hardware Safety Concept', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367605.
[D12] Luca Dariz, Massimiliano Ruggeri, Michele Selvatici, (in collaboration with the customer), 'ISO 25119 Safety plan', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367601.
[D13] Massimiliano Ruggeri, Michele Selvatici, Luca Dariz, 'Resonator MEMS - Schematic, PCB, BOM', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367717.
[D14] Michele Selvatici, Massimiliano Ruggeri, Luca Dariz, (in collaboration with the customer), 'Risk Analysis according to ISO 25119', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367603.
[D15] Massimiliano Ruggeri, Michele Selvatici, Luca Dariz, 'Safe Wireless Sensor Node - Schematics, PCB, BOM', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367715.
[D16] Luca Dariz, Massimiliano Ruggeri, Michele Selvatici, (in collaboration with the customer), 'Safety Assessment for Control Systems', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367682.

[D17] Luca Dariz, Massimiliano Ruggeri, Michele Selvatici, (in collaboration with the customer), 'Safety Levels Functional description', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367678.
[D18] Luca Dariz, Massimiliano Ruggeri, Michele Selvatici, (in collaboration with the customer), 'Safety Manual', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367616.
[D19] Luca Dariz, Massimiliano Ruggeri, Michele Selvatici, (in collaboration with the customer), 'Safety Mechanisms for network protection', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367680.
[D20] Luca Dariz, Massimiliano Ruggeri, Michele Selvatici, (in collaboration with the customer), 'Safety Requirements Specifications', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367604.
[D21] Luca Dariz, Massimiliano Ruggeri, Michele Selvatici, (in collaboration with the customer), 'Software Development Plan', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367608.
[D22] Luca Dariz, Massimiliano Ruggeri, Michele Selvatici, (in collaboration with the customer), 'Software SRL Analysis according to ISO 25119', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367610.
[D23] Luca Dariz, Massimiliano Ruggeri, Michele Selvatici, (in collaboration with the customer), 'Systematic Failure Analysis', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367607.
[D24] Luca Dariz, Massimiliano Ruggeri, Michele Selvatici, (in collaboration with the customer), 'Task Monitor and Resource Analysis', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367681.
[D25] Michele Selvatici, Massimliano Ruggeri, (in collaboration with the customer), 'Unit Of Observation', 2017. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=367602.
[D26] Michele Selvatici, Luca Dariz, Massimiliano Ruggeri, 'Report on MISRA C 2012 status of the firmware', 2016. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=356536.
[D27] Michele Selvatici, Luca Dariz, Massimiliano Ruggeri, 'Safety Manual for a client's ECU', 2016. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=356597.
[D28] Michele Selvatici, Massimiliano Ruggeri, Luca Dariz, 'Technical Safety Concept for a client's ECU', 2016. [Online]. Available: http://www.cnr.it/istituti/ProdottoDellaRicerca.html?cds=049&id=356599.