

UNIVERSITÀ
DEGLI STUDI
DI PADOVA



Analysis of an IEEE 802.11-based protocol for real-time applications in agriculture



Laureando
Michele Luvisotto

Relatore
Prof. Stefano Vitturi

Correlatore
Dott. Federico Tramarin

Corso di Laurea Magistrale in
Ingegneria dell'Automazione

XXIV ciclo
2013-2014

Contents

ACRONYMS	xiii
ABSTRACT	xxi
SOMMARIO	xxiii
FOREWORD	xxv
1 Introduction	1
1.1 Related work	2
1.2 Organization of the thesis	4
2 Description of the prototype system	7
2.1 System components	7
2.1.1 Electronic Control Unit	8
2.1.2 Hand-held device	11
2.1.3 Other components on the machine	11
2.2 Protocol overview	11
2.2.1 Structure of messages	12
2.2.2 Setup phase	14
2.2.3 Polling cycle	15
3 Introduction to the IEEE 802.11 standard	17
3.1 Architecture of a WLAN	18
3.1.1 WLAN components	18
3.1.2 WLAN services	19

3.2	MAC layer overview	20
3.2.1	Access to the medium	21
3.2.2	Fragmentation of frames	27
3.2.3	Multirate support	27
3.2.4	MAC frame format	29
3.3	PHY layer overview	34
3.3.1	Types of physical media	34
3.3.2	Transmission power	35
3.3.3	Clear Channel Assessment	35
3.4	IEEE 802.11b (HR/DSSS)	36
3.4.1	Modulation and Data rate	38
3.4.2	PPDU format	38
3.4.3	Transmission time	39
3.5	IEEE 802.11g (ERP-OFDM)	40
3.5.1	Modulation and Data rate	40
3.5.2	PPDU format	41
3.5.3	Transmission time	42
3.6	Most recent amendments to the standard	43
3.6.1	IEEE 802.11e	44
3.6.2	IEEE 802.11n	45
3.6.3	IEEE 802.11s	47
3.6.4	Amendments to the last standard	49
4	Theoretical performance analysis	51
4.1	Polling time	52
4.1.1	Relation with the cycle period	52
4.1.2	Relation with other metrics	53
4.1.3	Components of polling time	53
4.2	Transmission time on the UART bus	54
4.3	Response time of the ECU	55
4.4	Transmission time on the wireless channel	56
4.4.1	Packets exchange in absence of retransmissions	56
4.4.2	Packets exchange with a single retransmission	59
4.4.3	Packets exchange with multiple retransmissions	60
4.5	Statistical characterization of polling time	61
4.6	Final considerations	65

5	Experimental measurements	67
5.1	On-field measurements	67
5.1.1	Setup description	68
5.1.2	Outcomes	68
5.2	Laboratory measurements	70
5.2.1	Setup description	70
5.2.2	Outcomes with TCP	71
5.2.3	Outcomes with UDP	73
5.2.4	Tests with the iPad	75
5.3	Measurements on the wireless interface	76
5.3.1	Tests with TCP	77
5.3.2	Tests with UDP	78
5.4	Conclusions	79
6	Analysis and solutions of the most relevant issues	81
6.1	Issues related to the adoption of COTS devices	82
6.1.1	Self-interference	82
6.1.2	Power management	83
6.1.3	Tests without network services	85
6.2	Issues related to the wireless module	87
6.2.1	Synchronization in message transmission	87
6.2.2	Interaction with other devices	89
6.3	Modifications to the protocol	90
6.3.1	Choice of transport layer protocol	91
6.3.2	Changes in the structure of messages	91
6.3.3	New structure of polling cycle	94
6.3.4	Results with the new configuration	97
7	Conclusions	103
7.1	Overview of the activities	103
7.2	Review of the results	104
7.3	Future works	105

List of Figures

2.1	Schematic representation of the prototype system.	8
2.2	Close picture of the ECU installed on the agricultural machine.	9
2.3	Structure of application message.	12
2.4	Flow diagram of the setup phase.	14
2.5	Flow diagram of the polling cycle.	15
3.1	The three possible topologies defined in the IEEE 802.11 standard [25]. . .	18
3.2	MAC architecture.	21
3.3	Flow diagram of the channel access procedure with DCF.	23
3.4	Exponential increase of the CW counter.	25
3.5	Relationships between different IFSs.	26
3.6	Transmission of a burst of fragment.	27
3.7	Success probability on a 46 bytes packet versus channel SNR for IEEE 802.11g [25].	28
3.8	Format of a IEEE 802.11 frame at MAC level.	29
3.9	Structure of Frame Control field.	30
3.10	Disposition of WLAN channels in the ISM band.	37
3.11	Format of HR/DSSS PPDU adopting short preamble and header option. . .	39
3.12	Format of ERP-OFDM PPDU.	42
3.13	Data rate increase in IEEE 802.11n [21].	46
3.14	Mesh WLAN components [4].	48
4.1	Transmission of a byte over the UART bus.	54
4.2	Exchange of packets at MAC layer (ideal case).	57

4.3	Exchange of packets at MAC layer with the retransmission of a command frame.	59
4.4	Evolution of polling time versus the number of transmissions for IEEE 802.11g at 54 Mb/s.	63
4.5	Evolution of polling time versus the number of transmissions for IEEE 802.11b at 1 Mb/s.	64
4.6	Relative weights of polling time components for different scenarios.	65
5.1	Interpacket times measured in the on-field session.	69
5.2	Experimental setup employed in the indoor laboratory.	71
5.3	Interframe times measured in the laboratory session with TCP.	72
5.4	Histogram of frames that required more than one transmission attempt at MAC layer.	73
5.5	Interframe times measured in the laboratory session with UDP.	74
5.6	ECDF of interframe times for state frames with both TCP and UDP.	74
5.7	Interframe times measured in the laboratory session with the iPad as client device.	75
5.8	Picture of the evaluation board used to test the wireless module as a stand-alone component.	76
5.9	TCP service time using only the wireless module.	78
5.10	Statistical analysis of service time using only the wireless module.	78
5.11	Interdeparture times with UDP using only the wireless module.	79
6.1	Screenshot of a packet capture which highlights the presence of services not related to the application.	82
6.2	Screenshot of a packet capture which highlights the presence of Null Data frames.	83
6.3	Dissection of the Null Data frame which evidences that the Power Management bit is on.	84
6.4	Interframe times measured without network services using TCP.	85
6.5	Interframe times measured without network services using UDP.	86
6.6	ECDF of interframe times for state frames without network services.	86
6.7	Measures polling time for a series of consecutive state packets.	88
6.8	New message format, with the new SN field highlighted in yellow.	92
6.9	Encoder for byte stuffing technique, implemented in the ECU.	93
6.10	Decoder for byte stuffing technique, implemented in the client device.	94

6.11	New structure of polling cycle, client side.	96
6.12	New structure of polling cycle, ECU side.	97
6.13	Polling time at application layer with the new configuration and network services active.	98
6.14	ECDF of polling time with the new configuration and network services active.	99
6.15	PDF of the number of command message transmissions with the new configuration and network services active.	99
6.16	Results obtained with the new configuration and network services disabled.	100

List of Tables

2.1	Types of application messages.	13
2.2	Application protocol parameters.	16
3.1	Some possible combinations of Type and Subtype fields.	30
3.2	Encoding of the Duration/ID field.	31
3.3	CCA Modes employed by different physical media.	36
3.4	HR/DSSS channels.	37
3.5	HR/DSSS parameters.	40
3.6	Modulation and Coding schemes available with ERP-OFDM.	41
3.7	ERP-OFDM parameters.	43
3.8	EDCA Access Categories	45
4.1	Transmission times on the UART bus	55
4.2	Header sizes for upper layer protocols	58
4.3	Transmission times of command and state frames on the wireless channel.	58
4.4	Maximum and minimum value assumed by polling time for different data rates.	63
4.5	Maximum, minimum and average values assumed by polling time for transmissions number with IEEE 802.11g at 54 Mb/s.	64
6.1	Example of interframe times calculation which explains the role of wireless module synchronization issues.	89
6.2	Empirical probabilities of cycle outcome with and without network services.	101

ACRONYMS

AARF	Adaptive Automatic Rate Fallback
AC	Access Categories
ACK	Acknowledgment
AID	Association IDentifier
AIFS	Arbitration Inter Frame Space
A-MPDU	Aggregated MAC Protocol Data Unit
A-MSDU	Aggregated MAC Service Data Unit
AODV	Ad-hoc On-demand Distance Vector
AP	Access Point
ARF	Automatic Rate Fallback
ARM	Advanced RISC Machine
BPSK	Binary Phase Shift-Keying
BSS	Basic Service Set
BSSID	Basic Service Set IDentification
CAN	Controller Area Network
CCA	Clear Channel Assessment
CCK	Complementary Code Keying
CF	Coordination Function
CFP	Contention Free Period
CGN	Command Group Number

COTS	Commercial Off-the-Shelf
CP	Contention Period
CPS	Cyber-Physical System
CRC	Cyclic Redundancy Code
CSMA	Carrier Sense Multiple Access
CS	Carrier Sense
CSMA/CA	Carrier Sense Multiple Access/Collision Avoidance
CTS	Clear To Send
CW	Contention Window
DA	Destination Address
DBPSK	Differential Binary Phase Shift-Keying
DCF	Distributed Coordination Function
DNS	Domain Name System
DFS	Dynamic Frequency Selection
DIFS	DCF Inter Frame Space
DQPSK	Differential Quadrature Phase Shift-Keying
DS	Distribution System
DSE	Dynamic Station Enablement
DSM	Distribution System Medium
DSS	Distribution System Services
DSSS	Direct-Sequence Spread Spectrum
ECDF	Empirical Cumulative Density Function
ECU	Electronic Control Unit
ED	Energy Detection
EDCA	Enhanced Distributed Channel Access
EIFS	Extended Inter Frame Space

EIRP	Equivalent Isotropically Radiated Power
ERP	Extended Rate PHY
ERP-OFDM	Extended Rate PHY-OFDM
ESS	Extended Service Set
ETSI	European Telecommunication Standards Institute
FARF	Fast rate reduction Automatic Rate Fallback
FCS	Frame Check Sequence
FEC	Forward Error Correction
FHSS	Frequency Hopping Spread Spectrum
FN	Fragment Number
GPRS	General Packet Radio Service
GSM	Global System for Mobile communication
HC	Hybrid Coordinator
HCCA	HCF Controlled Channel Access
HCF	Hybrid Coordination Function
HR	High Rate
HR/DSSS	High Rate/Direct-Sequence Spread Spectrum
HT	High Throughput
HWMP	Hybrid Wireless Mesh Protocol
IBSS	Independent Basic Service Set
ICMP	Internet Control Message Protocol
ICT	Information and Communications Technology
IDE	Integrated Development Environment
IEEE	Institute of Electrical & Electronics Engineers
IFS	Inter Frame Space
IP	Internet Protocol

IR	InfraRed
ISM	Industrial, Scientific and Medical
IWSN	Industrial Wireless Sensor Network
JTAG	Joint Test Action Group
LAN	Local Area Network
LDPC	Low-Density Parity-Check
LLC	Link Layer Control
LLMNR	Link-Local Multicast Name Resolution
LOS	Line of Sight
LTE	Long Term Evolution
MAC	Medium Access Control
MAN	Metropolitan Area Network
MAP	Mesh Access Point
MCCA	MCF Controlled Channel Access
MCF	Mesh Coordination Function
MIMO	Multiple Input–Multiple Output
MLME	MAC Layer Management Entity
MMPDU	MAC Management Protocol Data Unit
MP	Mesh Point
MPDU	MAC Protocol Data Unit
MPP	Mesh Portal
MSDU	MAC Service Data Unit
MSS	Maximum Segment Size
MTU	Maximum Transmission Unit
NAV	Network Allocation Vector
NBNS	NetBIOS Name Service

Nic	Network Interface Card
OFDM	Orthogonal Frequency Division Multiplexing
OS	Operating System
PBCC	Packet Binary Convolutional Coding
PC	Point Coordinator
PCF	Point Coordination Function
PDF	Probability Density Function
PDU	Protocol Data Unit
PHY	Physical Layer
PIFS	PCF Inter Frame Space
PLCP	Physical Layer Convergence Procedure
PLME	Physical Layer Management Entity
PMD	Physical Media Dependent
PN	Pseudo-Noise
PPDU	PLCP Protocol Data Unit
PS	Power Save
PSDU	PLCP Service Data Unit
PSK	Phase Shift-Keying
QAM	Quadrature Amplitude Modulation
QAP	QoS Access Point
QBSS	QoS Basic Service Set
QoS	Quality of Service
QPSK	Quadrature Phase Shift-Keying
QSTA	QoS Station
RA	Rate Adaptation
RBAR	Receiver-Based Auto Rate

RF	Radio Frequency
RIFS	Reduced Inter Frame Space
RISC	Reduced Instruction Set Computer
RTS	Request To Send
RSS	Received Signal Strength
RSSLA	Received Signal Strength Link Adaptation
SA	Source Address
SARF	Static retransmission rate Automatic Rate Fallback
SCADA	Supervisory Control And Data Acquisition
SDM	Spatial Division Multiplexing
SDU	Service Data Unit
SFD	Synchronization Frame Delimiter
SIFS	Short Inter Frame Space
SLRC	STA Long Retry Count
SN	Sequence Number
SNR	Signal to Noise Ratio
SS	Station Services
SSRC	STA Short Retry Count
STA	Station
STBC	Space-Time Block Coding
TA	Transmitter Address
TC	Traffic Class
TCP	Transport Control Protocol
TPC	Transmit Power Control
TS	Traffic Stream
TXOP	Transmit Opportunity

UART	Universal Asynchronous Receiver/Transmitter
UDP	User Datagram Protocol
UMTS	Universal Mobile Telecommunication System
WLAN	Wireless Local Area Network
WM	Wireless Medium
WSN	Wireless Sensor Network

ABSTRACT

IN the last years agriculture has been greatly changed by the introduction of Information and Communications Technology (ICT) systems, which allowed for a significant growth in production amount and effectiveness of agricultural processes. In particular, the adoption of suitable communication architectures is an open problem and several solutions are being presented in literature.


This thesis proposes an original IEEE 802.11-based system which implements a wireless communication architecture for the remote monitoring and control of an agricultural machine. The adoption of this widespread protocol allows to use commonly available mobile devices, like tablet PCs or smartphones, to remotely control the machine.

Unfortunately, these devices are not natively conceived for real-time applications such as monitoring and control of a complex system and several issues can arise from their adoption in such a context. Specifically, the strong connectivity that characterizes these components may represent a problem, since, in order to provide rapid interactions with the surrounding ICT systems, they have to periodically pause any other operation to detect the presence of other devices. Such a behavior is clearly incompatible with a real-time control application, so it must be taken into account and possibly avoided.

The performance of the proposed system have been thoroughly characterized from both the theoretical and the practical point of view. Extensive experimental sessions have been conducted to test the system behavior in various contexts and with different configurations. The issues derived from the adoption of widely available devices are considered and easy to implement solutions are proposed.

Finally, a revised design of the communication protocol that enables the exchange of data between the remote device and the control unit on the machine is presented, capitalizing on the outcomes of the experimental sessions and on theoretical considerations. A final measurement campaign is carried out to validate the design choices, demonstrating a satisfactory behavior.

SOMMARIO

 NEGLI ultimi anni il settore dell'agricoltura è stato rivoluzionato dall'introduzione di sistemi ICT, che hanno portato ad una crescita significativa nell'ammontare della produzione e nell'efficacia dei processi agricoli. In particolare, l'utilizzo di appropriate architetture di comunicazione è un problema ancora aperto e numerose soluzioni sono state presentate a riguardo nella letteratura scientifica.

Questa tesi propone un sistema originale basato sullo standard IEEE 802.11 che realizza un'architettura di comunicazione wireless per il monitoraggio ed il controllo remoto di una macchina agricola. L'impiego di questo protocollo così diffuso consente l'uso di dispositivi mobili commerciali quali smartphones e tablet per il controllo remoto della macchina.

Purtroppo, questi dispositivi non sono originariamente progettati per applicazioni in tempo reale come il monitoraggio e controllo di un sistema complesso e diversi problemi possono nascere dal loro utilizzo in questo contesto. Nello specifico, questi componenti sono caratterizzati da una forte connettività che può rappresentare un problema, dal momento che, per essere in grado di fornire interazioni rapide con i sistemi ICT circostanti, devono periodicamente mettere in pausa qualsiasi altra attività per controllare la presenza di altri dispositivi. Un comportamento di questo tipo è chiaramente incompatibile con un'applicazione in tempo reale, quindi deve essere considerato e possibilmente evitato.

Le prestazioni del sistema proposto sono state attentamente caratterizzate sia dal punto di vista teorico che da quello pratico. Sono state realizzate esaurienti sessioni di misure sperimentali per testare il comportamento del sistema in contesti diversi e variandone la configurazione. Le problematiche derivate dall'utilizzo di dispositivi commerciali di uso comune sono analizzate e soluzioni di facile implementazione sono proposte.

Infine, è presentato un nuovo design del protocollo di comunicazione che consente lo scambio di dati tra il dispositivo mobile e l'unità di controllo sulla macchina agricola, basandosi sui risultati delle misure sperimentali e su considerazioni teoriche. E' stata effettuata una sessione di misura finale per dimostrare la validità delle scelte fatte, evidenziando un comportamento soddisfacente da parte del sistema.

FOREWORD

This thesis is the result of a research collaboration between MC Elettronica S.r.l. and CNR-IEIT.

The author would like to thank MC Elettronica S.r.l., and in particular Mr. Claudio Mantovani, for the possibility to work at this project and also the technical staff, in particular Ing. Raffaele Parrozzani, for the constant support offered throughout the entire realization of this thesis.

Chapter 1

Introduction

In the last decades, the worldwide agricultural production has greatly benefited from the introduction of ICT systems. Nowadays, the need for efficient agricultural systems, that could exploit the advantages offered by always improving technologies and devices, is greater than ever, due to the continue worsening of indices such as global population growth, climate changes and food waste. A 2011 study suggested that almost one-third of the food produced worldwide is lost before being consumed [9], with inefficiencies in agricultural production being one of the most relevant causes.

The need for smarter agricultural systems has led to the development of *precision agriculture*, a new approach which integrates recent technologies such as positioning systems, automatic control, sensor networks, embedded systems and communications, with thousand-year old concepts like crop rotation [30]. Several scientific programs arose from this paradigm and a considerable amount of publications appeared in literature concerned with such a topic[5].

The precision agriculture concept is a perfect example of Cyber-Physical System (CPS), in which distributed and collaborating computational units are used to control physical entities [22]. Indeed, the CPS program of the U.S. National Science Foundation [26] recognizes agriculture as one of the most important drivers in this research field.

The adoption of appropriate and efficient communication architectures is a key factor in the development of smart agricultural systems. The various agents involved in the different production phases (seeding, irrigation, harvesting, ecc.) need to exchange data and commands in a fast and reliable way, allowing for a precise control of agricultural production. Moreover, the communication infrastructure has to be integrated in the existing environment and to cope with the severe constraints imposed by outdoor context.

The possibilities offered by wireless communication systems, in particular, have drawn a lot of interests because of their easy implementation, low installation costs and capacity of integration with commonly used devices. The rural environment is also generally characterized by less presence of obstacles and radio sources with respect to urban contexts, either indoor or outdoor, allowing to reach larger coverage ranges. However, the robustness of wireless communication is a concern and design efforts have to be made in order to ensure a real-time behavior to systems based on such an infrastructure.

This thesis presents an original application for the remote control and monitoring of an agricultural machine adopting an IEEE 802.11-based wireless communication architecture, which allows for remote handling via common mobile devices. The project has been developed in collaboration with a company specialized in the agricultural field and has the purpose of analyzing the software and hardware that constitute the control system in order to assess its performance figures and optimize the communication between the device and the control unit of the machine.

1.1 Related work

Several contributions appeared in literature dealing with the introduction of suitable communication architectures in agricultural applications. Specifically, systems based on the numerous wireless communication technologies available have been proposed. Two different groups of technologies have been mainly discussed, each one with precise strong points that make it valid for specific purposes.

The first group of technologies is represented by cellular networks such as Global System for Mobile communication (GSM), General Packet Radio Service (GPRS), Universal Mobile Telecommunication System (UMTS) and Long Term Evolution (LTE). The advantage of these systems is that they are ubiquitous and are supported by almost every Commercial Off-the-Shelf (COTS) mobile device. On the other hand, they might offer poor performance in terms of delay and throughput if the coverage is not adequate, which is often the case in rural environments. As a consequence, they are mainly adopted for the implementation of remote monitoring systems, even for very long distances, while their use is discouraged for time-critical activities.

The second solution is represented by Industrial Wireless Sensor Networks (IWSNs), typically based on the IEEE 802.15.4 standard [8]. A Wireless Sensor Network (WSN) is constituted by a certain number devices distributed in a certain area, with the possibility to acquire data, exchange them on the wireless medium and possibly taking some actions.

The term *IWSN* is used for a *WSN* in which specific techniques are adopted to overcome the intrinsic unreliability of wireless links in order to cope with real-time requirements. These features allow the use of this solution in time-critical control applications. The drawback is that the coverage range is limited, so only surrounding device can exploit architectures based on *IWSN*. However, wired extensions or gateway to cellular networks can be realized to carry the data at longer distances.

The work in [14] proposes a monitoring framework for agricultural systems based on *WSNs*. Precision agriculture is researched through a constant monitoring on land and crop conditions and exchange of the gathered information. The possibility offered by *WSNs* allow to form a monitoring network composed by low-cost and low-power nodes. The remote access to data is possible also from long distances thanks to specific gateways to the *GSM* network.

A similar solution is proposed in [17], where *TinyOS*-based nodes are adopted to realize a crop monitoring system, with the possibility to easily visualize the displacement of sensors on a map and the data they are collecting.

The work in [28] evidences the possibilities offered by the integration of cellular technologies and sensor networks, presenting a real-time remote environmental monitoring system in which the information collected by solar power supplied small sensors are delivered to mobile phones thanks to *GSM* short messaging service. The system is conceived as a warning system, in which the final user sets some guard parameters with the mobile phone and receives a notification when the specified values have exceeded guard limits.

Hybrid wired/wireless solutions have also been proposed [18, 20], in which separate *IEEE 802.15.4* networks are used for monitoring and management of a very wide area and a wired medium, such as *CAN* or *Ethernet* bus is adopted to connect the different networks.

Very few contributions can be found in literature that propose solutions based on *IEEE 802.11*. An example can be found in [24], where a multi-layer *Supervisory Control And Data Acquisition (SCADA)* system for agricultural applications is proposed and different wireless technologies are considered for the short-distance communication, among which *Bluetooth* and *IEEE 802.11* are mentioned. However, to the best of the author knowledge, there is no contribution that specifically deals with the possibilities offered by this protocol in agricultural field and the difficulties that might be encountered.

Actually, there is an ever growing interest towards the use of *IEEE 802.11* for industrial applications [19, 23]. A great effort is being done by the industrial communication community to adopt appropriate expedients in order to obtain a soft real-time behavior from this protocol. In particular, rate adaptation techniques are studied to ensure robust

transmission even in presence of bad channel conditions [27] and time-division strategies are considered to provide real-time guarantees on packet delivery [29].

As a matter of fact, the adoption of IEEE 802.11 could allow for a different set of operations with respect to the solutions currently adopted, namely cellular networks and WSNs. Specifically, it could be used with available COTS devices as in the case of cellular networks and can provide (even limited) support to time-critical control applications, in addition to monitoring. The application studied in this thesis aims at exploiting these two features to allow for remote control and monitoring of an agricultural machine via a COTS hand-held device.

1.2 Organization of the thesis

In the following, an outline of this thesis is presented and a brief description is given for each Chapter.

Chapter 2 presents the system that will be studied in this work. The hardware elements that compose it are characterized and the effect of each component is carefully taken into account. Moreover, the communication protocol developed for the remote control and monitoring is studied. The content of each packet exchanged by the hand-held device and the control unit on the agricultural machine are described. Finally, the structure of the implemented periodic polling cycle is thoroughly analyzed.

In **Chapter 3** a deep overview of the IEEE 802.11 standard is provided, describing the specifications for Medium Access Control (MAC) layer and different types of physical layer. In particular, the focus is on IEEE 802.11b and IEEE 802.11g physical layers, since they are the most widespread versions of this standard and also the only two possibilities available in the considered system. Nonetheless, some insights on the most recent amendments to the standard are also provided.

In **Chapter 4** the behavior of the prototype system is characterized through a theoretical analysis of its communication performance. In particular, the focus is on the maximum time required to complete a polling operation from the hand-held device. The different source of randomness are taken into account and statistical analysis are performed in order to obtain a range of variability for this metric.

Chapter 5 presents the outcomes of a series of experimental campaigns aimed at characterizing the real performance figures of the prototype system. A first experimental campaign has been carried out in a real application environment, namely an outdoor field. Subsequently, different measurement sessions have been performed in a research labora-

tory to provide a more controlled environment and allow to rapidly change configuration parameters. The measurements highlighted several issues in the wireless communication that caused the behavior of the system to differ sensibly from the expected one, derived through the theoretical analysis.

The identification of the causes of performance degradation evidenced in the measurement sessions is made in **Chapter 6**. Here, appropriate solutions are provided and new measurements are taken to prove that the solutions are effective. Moreover, a new design of the communication protocol is presented to further enhance the performance figures of the system and to fully exploit the communication resources. Final tests are carried out to validate the design choices, showing a satisfactory behavior.

Chapter 7 provides a summary of the work, listing the main activities conducted and the principal tools employed. A review of the most important results obtained through the thesis is presented. Finally, some considerations on possible future developments are made, presenting new ways to exploit the application and possible actions to improve the performance even further.

Chapter 2

Description of the prototype system

The application analyzed in this thesis consists of an original IEEE 802.11-based system which adopts a wireless communication architecture to perform soft-real time operations, in particular remote control and monitoring of agriculture equipment. This is achieved through the installation on the agricultural machine of an ECU equipped with a IEEE 802.11 compliant wireless module, which enables it to communicate with hand-held devices that support this standard.

The considered prototype system is described in detail, providing a list of all its physical components and their role in the control architecture. Moreover, the communication protocol developed for this application is thoroughly analyzed, giving a precise characterization of all the packets exchanged over the wireless link. This overview serves as a starting point for the theoretical and experimental evaluation of the system performance that will be carried out in the following chapters. Subsequently, possible enhancements to the structure presented in this Chapter will be discussed in order to achieve a more controlled behavior for the system.

THE analyzed architecture is the control and monitoring system of a machine employed for typical agricultural operations such as seeding or spraying, mechanically connected to a tractor.

2.1 System components

A schematic representation of the prototype system considered in this thesis is given in Fig. 2.1. The two most important components for the purposes of communication are the hand-held device and the Electronic Control Unit (ECU).

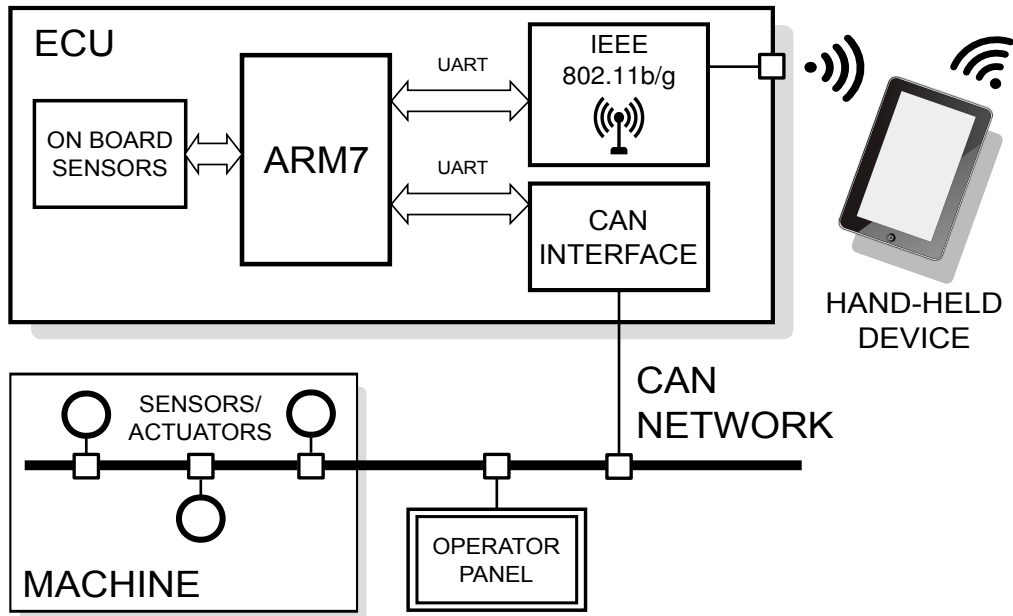


Figure 2.1 – Schematic representation of the prototype system.

2.1.1 Electronic Control Unit

The *ECU* is installed on the agricultural machine whose operations have to be monitored and possibly controlled. The machine is mechanically attached to a tractor which, however, is totally independent and does not interact with the presented system.

In this context, the *ECU* acts as a bridge between the control network inside the machine, represented by a Controller Area Network (*CAN*) bus, and an hand-held device used by an operator on board or nearby the tractor.

The core of the *ECU* is represented by a 32-bit Advanced RISC Machine (*ARM*) v7 controller, which executes all the required operations and regulates the interactions between the other components of the system. The firmware has been written in C with μ vision™ Integrated Development Environment (*IDE*) by Keil™, then flashed with an USB *JTAG* programmer. To collect debug data and insert preregistered commands, a serial port was available.

The *ECU* comprises also some on-board sensors and two communication interfaces. The first one is a *CAN* interface, which enables the *ECU* to collect and distribute data to the other machine components over the *CAN*. The second and most important one is a IEEE 802.11 compliant wireless module. The introduction of wireless communication represents a significant novelty in the agricultural environment and differentiates this *ECU* from all

other control units that may be installed on this type of machines. This interface creates an IEEE 802.11 *WLAN* configured in Infrastructure mode, acting as an *AP* to which the remote hand-held device connects as a *STA*. Since this module is the key element that allows wireless communication between the machine and the operator holding a remote device, it will be later discussed more in detail in this Section.

The controller is linked with the two interfaces (*CAN* and wireless) via a Universal Asynchronous Receiver/Transmitter (*UART*) serial bus. This choice (which represents a project constraint) has a non-negligible impact on the exchange of packets, since the achievable data rates are quite low and therefore the connection between the *ARM* controller and the wireless module can act as a bottleneck on the system.

A picture of the *ECU* is given in Fig. 2.2. The aforementioned components are difficult to distinguish, since they are all mounted on the motherboard. However, the wireless module is clearly recognizable.

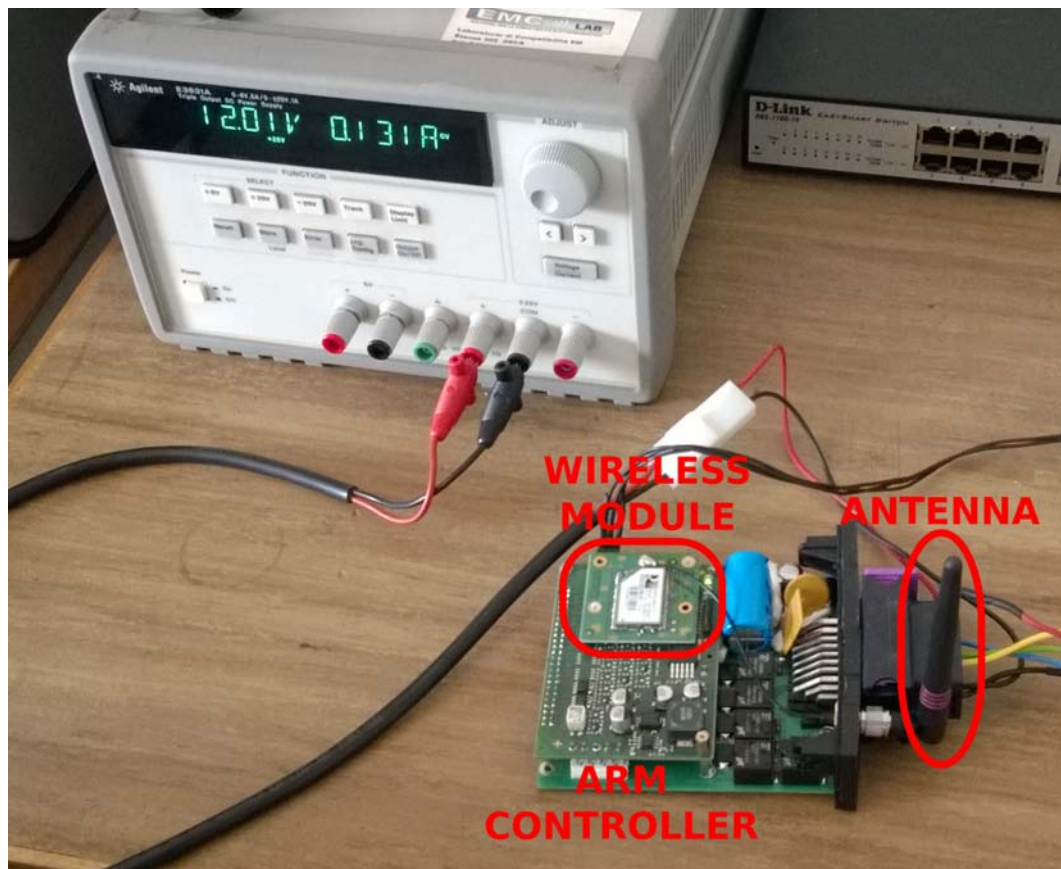


Figure 2.2 – Close picture of the *ECU* installed on the agricultural machine.

During machine operation, the **ARM** controller receives commands and set points for the machine from the hand-held device, via the wireless module and the **UART** bus, and forwards them to the other components on the machine, via the **UART** bus and the **CAN** interface. In the same way, data gathered from the machine via the **CAN** bus can be forwarded by the controller to the remote device via the wireless module. It is worth observing that the described system represents, to a certain extent, a wireless extension of the wired **CAN** fieldbus, implemented through the wireless module installed on the **ECU**.

IEEE 802.11 wireless module

The agricultural field of application forces the employment of devices purposely designed to support operations in harsh outdoor environments, characterized, for example, by wide ranges of rapidly variable temperatures. There are only few manufacturers able to supply IEEE 802.11 compliant interfaces able to cope with these strict environmental requirements. Moreover, the available devices typically grant with the access to only a reduced set of communication parameters. Specifically, in the prototype system considered in this thesis, an IEEE 802.11b/g Microchip™ RN171 wireless module has been adopted. Such an interface has been selected by the company that developed the prototype system. As a consequence, it represents a project constraint that cannot be changed and presents some very impacting issues from the communication perspective.

First of all, this wireless module does not implement any rate adaptation algorithm, a feature typically provided by the **MAC** level of IEEE 802.11 (see Section 3.2.3), which reveals particularly helpful in error prone environments. Moreover, the maximum available transmission power is limited to 12 dBm, which is quite lower than the maximum value allowed by the standard for the 2.4 GHz band (20 dBm, see Section 3.3.2), worsening the Signal to Noise Ratio (**SNR**).

Furthermore, this module allows to use only the **DCF** channel access function, while other functionalities that could ensure more timely performance, such as the **PCF** or the improvements offered by the IEEE 802.11e version of the standard are not available.

Finally, the wireless interface does not allow to directly transmit and receive packets at **MAC** layer level forcing the employment of transport layer protocols such as Transport Control Protocol (**TCP**) or, alternatively, User Datagram Protocol (**UDP**). The use of **TCP** is quite uncommon in real-time applications, given its considerable overhead which may result in performance degradations.

2.1.2 Hand-held device

The hand-held device employed in the soft real-time control and monitoring application described in this thesis is an available COTS device, such as a smartphone or a tablet. The only strict requirement that the device must fulfill in order to implement this application is the presence of an IEEE 802.11b/g compliant wireless interface. This facility, however, is present in the totality of COTS mobile devices.

The tests described in this thesis have been performed using an Apple™ iPad as the remote hand-held device. However, to have more control on the application parameters and system configuration, a desktop PC has also been used as client device in some experimental campaigns.

2.1.3 Other components on the machine

Agricultural machines are natively equipped with a wired control system, which is composed by several distributed sensors and actuators and also by some control units, that coordinate the agricultural operations that have to be performed. All these components are connected through a wired CAN fieldbus, which allows for command and data exchange.

Traditionally, the behavior of this system could be monitored and regulated by the operator aboard the tractor via an operator panel, connected to the CAN network, in which an appropriate SCADA application had been installed. The novelty described in this thesis is the seamless replacement of this panel with an hand-held device in which the same application can be used, thanks to the WLAN created by the ECU. However, the traditional operator panel may still be present in the system (as shown in Fig. 2.1), so that the operator driving the tractor can take over the machine operation at any instant bypassing the remote handling system.

2.2 Protocol overview

A communication protocol has been designed to allow the remote control and monitoring of the agricultural machine via the hand-held device. The two actors of the communication (also referred to as nodes) are the ECU inside the machine and the remote hand-held device used by the operator. The possibility of employing multiple remote devices has also been considered, although in this work all the considerations are relevant to the case of a single device.

The designed protocol is a cyclic polling protocol. The hand-held device acts as the *master* of the polling cycle, periodically transmitting updated set points relevant to the operations to be performed with a *command* message. The ECU, on the contrary, is the polling *slave*, which receives the commands and replies with a *state* message, carrying information originating from the sensors distributed on the machine.

At the beginning of the communication a setup phase is required, where the two nodes exchange a set of parameters and some security information. Then, a steady-state is reached, where the command and state messages are periodically exchanged. All the messages transmitted by the two nodes in these two phases have the same structure, although the values on the fields may vary.

2.2.1 Structure of messages

The data exchanged between the ECU and the hand-held device have a fixed structure, which contains:

- ✓ an *header*, containing fields that identify the type of message and allow to decode the rest of it;
- ✓ a *payload*, containing all the useful content of the message (for example the data read from sensors on the machine in the state message);
- ✓ a *trailer*, inserted for synchronization and security purposes.

While the header and trailer have a fixed length, which results in 10 total Bytes, the length of the payload may change according to the type of message. Fig. 2.3 illustrates the structure of a message, explicitating the length of each field. It can be seen that the minimum length of a message is 14 Bytes, while the maximum one with the common settings is 257 Bytes. It is worth noting that this value is substantially lower than the Maximum Segment Size (MSS) normally considered for TCP, which in turn depends on the Maximum Transmission Unit (MTU) size at MAC layer, so there is the possibility to expand message size up to 1460 Bytes.

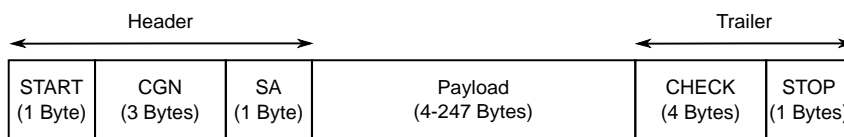


Figure 2.3 – Structure of application message.

The meaning of each message field is briefly discussed in the remainder of this Section.

START

This byte is used to identify the beginning of a message. It has a fixed value, namely 0x3C, which corresponds to the ASCII character '<'.

Command Group Number (CGN)

This 3 Bytes field is used to identify the type of message. Currently, four different types of messages have been defined, as reported in Tab. 2.1. The most important messages are the command and state ones, while the other two types are used only during the setup phase. As it can be seen in the table, a specific payload length corresponds to each message type.

Table 2.1 – Types of application messages.

Message Type	CGN	Payload length	Message length
System_Info_App	0x1	4 Bytes	14 Bytes
Command	0x14	8 Bytes	18 Bytes
State	0x15	247 Bytes	257 Bytes
ECU_Param	0x16	177 Bytes	187 Bytes

Source Address (SA)

This byte uniquely identifies the node which has sent the message. The value 0x0 is set if the message has been sent by the ECU, while it is increased by one unit for every different hand-held device connected to the network. In this work only one device at a time is connected, so this byte can only assume 0x0 and 0x1 as values.

Payload

This is the only variable length field, containing the message payload. Length varies according to the type of message, as reported in Tab. 2.1.

CHECK

This 4 Bytes field contains a 32-bit Cyclic Redundancy Code (CRC) code computed on the previous message bytes, for security purposes.

STOP

This byte is used to identify the end of a message. It has a fixed value, namely 0x3E, which corresponds to the ASCII character '>'.

2.2.2 Setup phase

As soon as the agricultural machine is started, the ECU is switched on and the wireless interface creates a Wireless Local Area Network (WLAN), to which every IEEE 802.11b/g compliant device can associate via a secure (cyphered) access.

Once the hand-held device in which the control application is installed has associated and authenticated to the network, the application can be launched and the setup phase described in Fig. 5.2 begins.

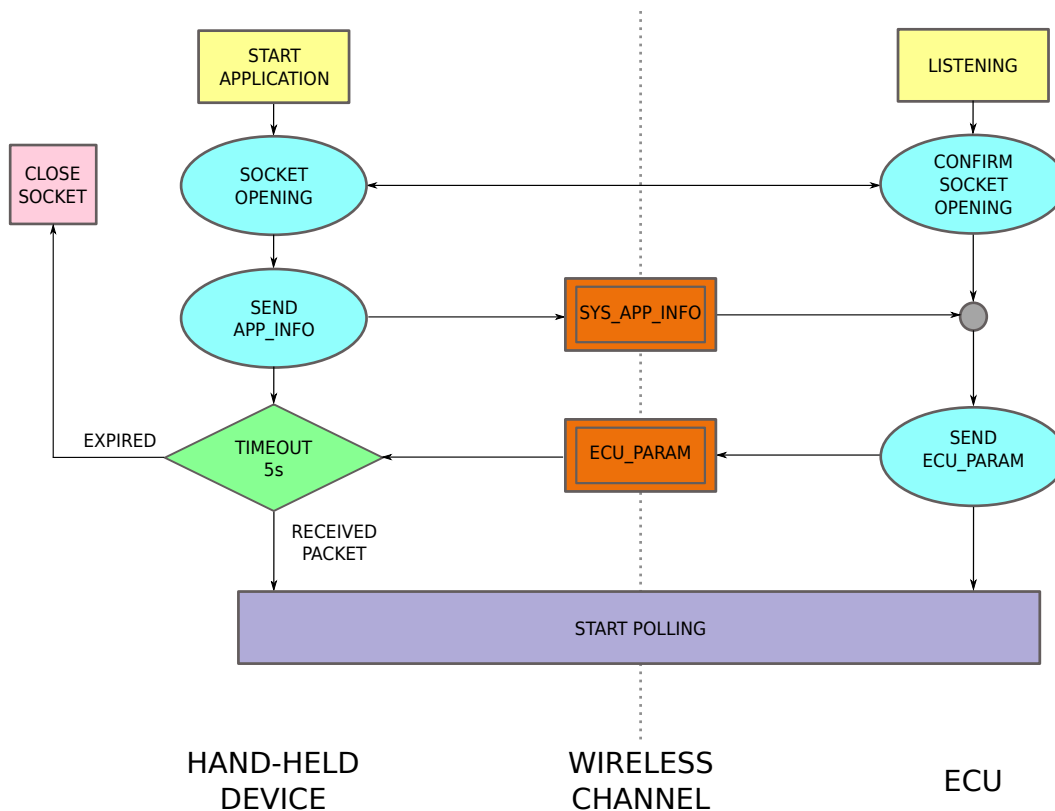


Figure 2.4 – Flow diagram of the setup phase.

The setup is initialized by the control master, *i.e.* the hand-held device, which opens the TCP socket¹. As soon as the socket opening is confirmed with the typical three-way

¹This part is skipped if UDP is used as transport layer protocol.

handshake TCP procedure, the remote device sends a *System_Info_App* message to the ECU, containing a code and some parameters which identify the current version of the application. The ECU checks if these parameters are consistent with those hard-coded in its firmware and, if this is the case, replies with a *ECU_Param* message, containing a set of parameters that describe the operations currently undergoing on the agricultural machine. If the hand-held device receives this message within 5 seconds from the transmission of the *System_Info_App* message, the setup phase is considered as concluded and the polling cycle can finally start. If the timeout expires, the TCP socket is closed and the application has to be manually restarted.

2.2.3 Polling cycle

The flow of packets during the polling cycle is illustrated in Fig. 2.5.

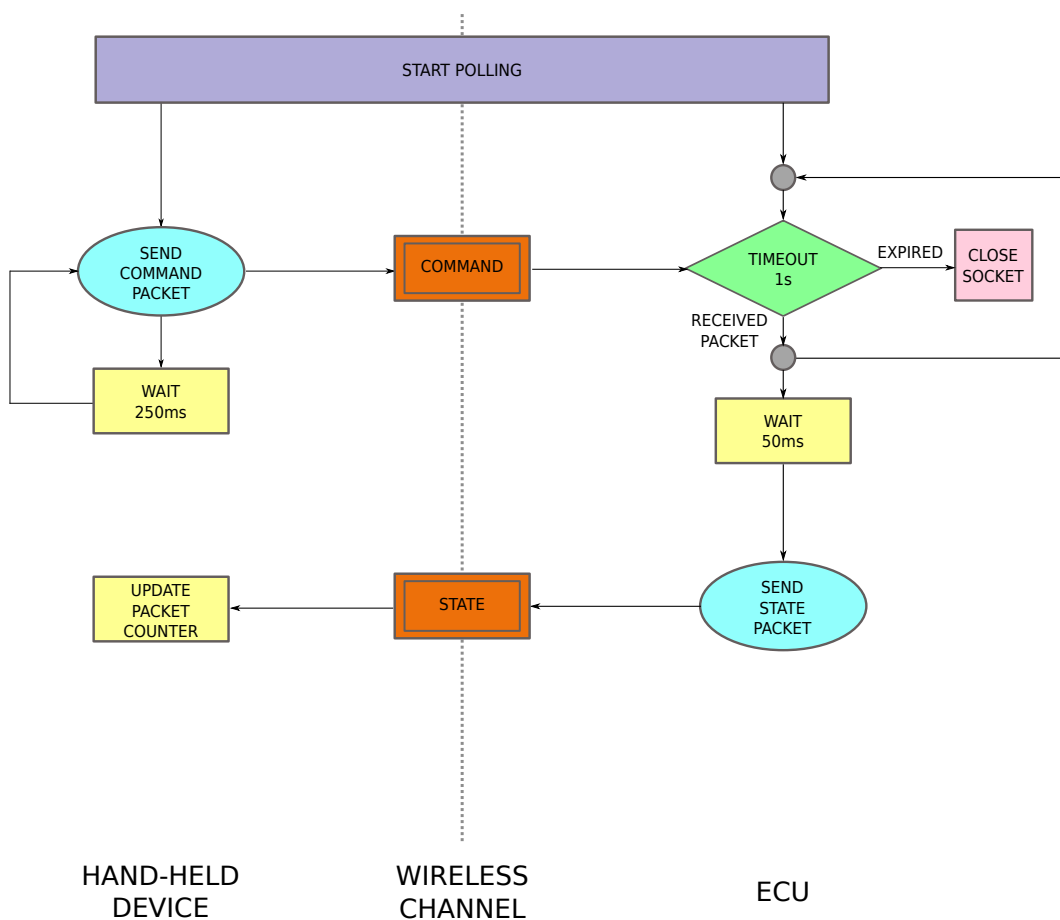


Figure 2.5 – Flow diagram of the polling cycle.

The hand-held device acts as the master of the polling cycle, sending a *command* message, which contains the updated commands and set points for the agricultural machine, with a fixed periodicity. The period duration, called *cycle period* and indicated with T_C , is initially set to 250 ms, which is expected to ensure good performance given the dynamics of the system to control.

Upon successful reception of the command message, the ECU replies with a *state* message, containing the information gathered from a selected set of sensors distributed on the machine. In this version of the protocol, the ARM controller on the ECU inserts an idle time between the reception of the command message and the transmission of the corresponding state message, indicated with T_{d_ARM} and set to 50 ms.

The ECU also runs a timer to ensure the polling procedure is performed successfully. This timer, initialized to the value 1 s, is restarted each time a command message is received and reset if a new command message arrives. In this way, if the ECU does not receive any command message within a 1 second interval, the polling application is stopped and the communication socket is closed. By doing this, the application ensures that the ECU is not using a set of commands that might be too old and potentially dangerous for the system.

As a final summary, Tab. 2.2 contains a list of the relevant protocol parameters and their values, that will be considered in the following Chapter, where a careful performance analysis of the presented protocol is carried out.

Table 2.2 – Application protocol parameters.

Parameter	Description	Value
T_C	Cycle period	250 ms
T_{d_ARM}	Delay inserted by the ARM controller	50 ms
L_{com}	Length of the command message at application layer	18 Bytes
L_{sta}	Length of the state message at application layer	257 Bytes

Introduction to the IEEE 802.11 standard

The 802.11 standard of Institute of Electrical & Electronics Engineers (IEEE) is a part of the IEEE 802 standard family, which deals with Local Area Networks (LANs) and Metropolitan Area Networks (MANs). Specifically, IEEE 802.11 provides MAC layer and Physical Layer (PHY) layer specifications for Wireless Local Area Network radio communication over several frequency bands. In such a network, nodes can be either fixed or mobile and are displaced within a local area. The high throughput guaranteed by this standard allows IEEE 802.11-based devices to use common upper level protocols traditionally adopted in wired networks, such as Internet Protocol (IP), Transport Control Protocol (TCP), User Datagram Protocol (UDP), etc.

The first version of IEEE 802.11 has been released in 1997, then several updates have been published over the years, the last one in 2012 [13]. Between different versions of the standard, several amendments are released by the IEEE 802.11 task group, such as IEEE 802.11b [13, Clause 17] and IEEE 802.11g [13, Clause 19]. When a new version of the standard is released, amendments published in the meanwhile are incorporated.

The main features of IEEE 802.11 are summarized in this Chapter, with a specific focus on the frame transmission process, giving results and models that will be used in the following Chapters. A brief insight on the most updated versions of this standard is also provided, even though the wireless module adopted in the system described in this thesis do not support them.

THE IEEE 802.11 standard deals with the MAC part of the data link layer and the PHY layer. It also provides specification on the topologies available for a WLAN and the types of physical media that can be used.

3.1 Architecture of a WLAN

This Section provides an overview of the different components that may be present in a WLAN network according to the IEEE 802.11 standard. A general outlook of the services that can be implemented in such a network is also provided.

3.1.1 WLAN components

The basic element of a WLAN is the Station (STA), a logical entity that has an address within the network. STAs can be fixed, portable (can be moved but are usable only when fixed) or mobile (can be used while moving).

A first possible network topology is the *infrastructure mode*, where several STAs are connected to an Access Point (AP). Together, AP and STAs form a Basic Service Set (BSS). The AP acts like a normal STA and, in addition, provides access to the Distribution System (DS), which connects a specific BSS with other BSSs or non-IEEE 802.11 LANs. When different APs, pertaining to different BSSs, are linked together by a DS, the term Extended Service Set (ESS) is used. The standard logically distinguishes between the Wireless Medium (WM), which connects AP to STAs, and the Distribution System Medium (DSM), which connects different APs, while the adopted physical media can be either the same or different.

The standard defines also a different topology, called *ad-hoc network*, in which there is no AP and two or more STAs are connected together, with no access to the DS. This type of network is also called Independent Basic Service Set (IBSS).

Fig. 3.1, taken from [25], summarizes possible network topologies in a WLAN.

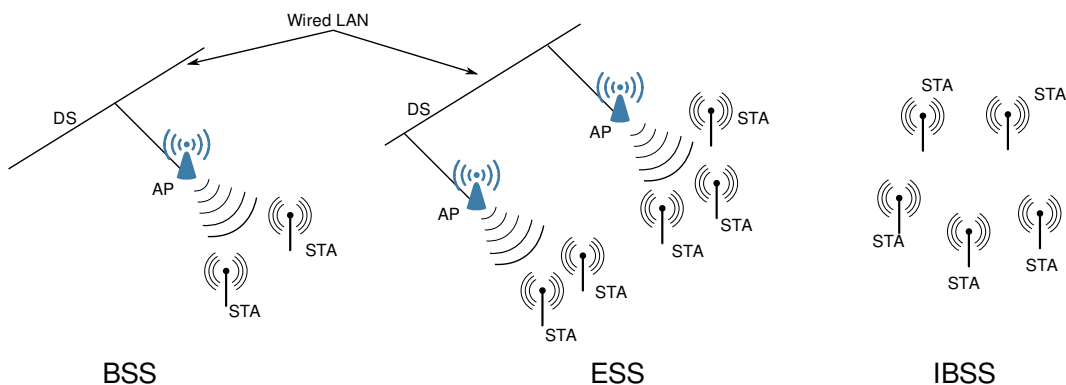


Figure 3.1 – The three possible topologies defined in the IEEE 802.11 standard [25].

The standard uses the word *area* to refer to the portion of the space covered by the

same BSS, although a good definition of coverage area for a wireless PHY does not exist. Indeed, the strength of the transmitted signal, and therefore the area that a STA can reach with communication, changes irregularly over space and time due to various phenomena, such as interference, fading, shadowing and multipath.

3.1.2 WLAN services

The IEEE 802.11 standard specifies a list of services which are provided by different WLAN components. Some of them are optional, while others are mandatory. The set of services is divided into two groups: Station Services (SS), part of every STA, and Distribution System Services (DSS), provided by the DS.

The SS set includes the following services:

- ✓ **Authentication:** used by a STA to establish its identity to the other STAs.
- ✓ **Deauthentication:** invoked when an existing authentication is terminated.
- ✓ **Data confidentiality:** used to protect the contents of messages exchanged over the WM.
- ✓ **Data delivery:** used to perform the actual exchange of messages over the WM.
- ✓ **Dynamic Frequency Selection (DFS):** used to satisfy regulatory requirements on the utilization of frequency bands.
- ✓ **Transmit Power Control (TPC):** used to satisfy regulatory requirements on the transmitter power.
- ✓ **Higher layer timer synchronization:** provides accurate synchronized timers shared among different STAs to applications that demand this feature (e.g. audio or video streaming).
- ✓ **Quality of Service (QoS) traffic scheduling:** used in advanced version of the standard to differentiate traffic classes and handle QoS requirements.
- ✓ **Dynamic Station Enablement (DSE):** used to automate the channel provisioning and regulatory controls needed for unregistered STAs to operate.

The DSS set includes the following services:

- ✓ **Association:** used to map a specific STA to the DS via a certain AP.

- ✓ **Disassociation:** invoked when an existing association is to be terminated.
- ✓ **Reassociation:** invoked when an associated *STA* transit from an *AP* to another within the same *ESS*.
- ✓ **Distribution:** used whenever a frame needs to be transmitted from a *STA* to another *STA* in the same *ESS* through the *DSS*.
- ✓ **Integration:** invoked after *Distribution* if the frame has to be delivered to the member of an integrated IEEE 802.x *LAN*, that can be reached by the *DS* through a specific device called *portal*.
- ✓ **Interworking with the DS:** used in mesh network to allow a non-*AP STA* to access services provided by an external network to which it may be subscribed.
- ✓ **QoS traffic scheduling:** analogous to the corresponding *SS*.
- ✓ **DSE:** analogous to the corresponding *SS*.

3.2 MAC layer overview

IEEE 802.11 provides specification for the *MAC* part of the data link layer, also called *MAC* sublayer. This part includes also a management entity, called *MAC* Layer Management Entity (*MLME*). Together, *MAC* and *MLME* provide a series of functionalities, among which there are:

- ✓ **Access to the medium:** it is the basic service that the *MAC* layer has to provide, allowing different *STAs* to share the same medium.
- ✓ **Fragmentation and defragmentation:** sometimes Protocol Data Unit (*PDU*) at *MAC* level can be fragmented into smaller frames to enhance transmission performance, especially on channels with bad quality.
- ✓ **Multirate support:** if the *PHY* layer supports different transmit rates, an adaptive approach can be used, selecting the best transmit rate given channel conditions.

There are other important functionalities provided by the *MAC* layer, such as the management of nodes mobility, the setup and teardown of connections, etc. However, in this Chapter only the three aforementioned functionalities will be discussed, since they are the most important for the scope of the present work.

3.2.1 Access to the medium

The access to the medium in WLANs is regulated by a Coordination Function (CF), a logical function that determines when a STA is allowed to transmit over the WM. The basic type of CF is the Distributed Coordination Function (DCF), through which each STA decides independently when to transmit data over the wireless channel. Any station that belongs to an IEEE 802.11 network implements DCF, while there are some other CFs that may or may not be implemented, namely the Point Coordination Function (PCF), the Hybrid Coordination Function (HCF) and the Mesh Coordination Function (MCF). Fig. 3.2 depicts the general architecture of the IEEE 802.11 MAC, distinguishing between the different types of CFs.

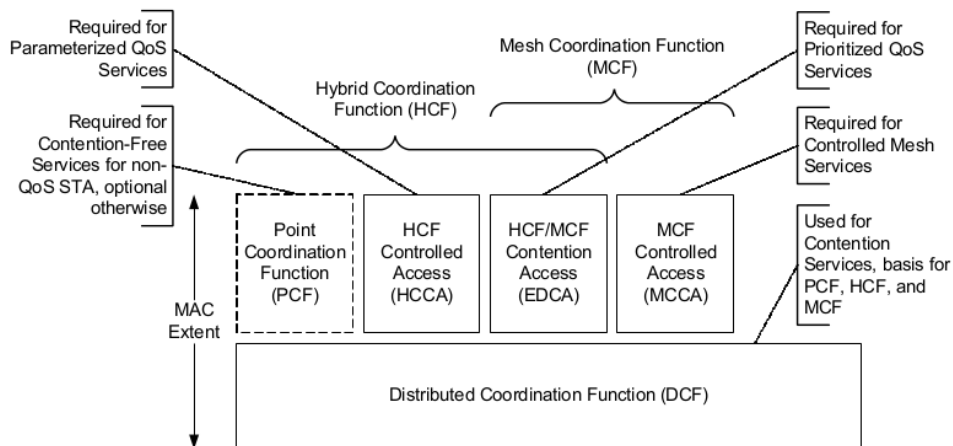


Figure 3.2 – MAC architecture.

Optional Coordination Functions

The PCF can only be implemented in infrastructure mode networks, in which an AP is present. The AP acts as a Point Coordinator (PC), a centralized control unit which polls other STAs in order to determine the node that can be given channel access. Since this feature is optional, each STA can decide whether to accept operating in PCF mode (hence becoming *CF-Pollable*) or to refuse it and continue using DCF. If there is at least a STA which adopts PCF, there is a continuous alternation of Contention Free Periods (CFPs), where channel access is controlled by the PC, and Contention Periods (CPs), where the standard DCF rules are followed.

The HCF was introduced in the IEEE 802.11e version of the standard to allow for QoS

enhancements and it is briefly described in Section 3.6.1.

The MCF was introduced in the IEEE 802.11s version of the standard and it is briefly described in Section 3.6.3.

Distributed Coordination Function

The DCF is the standard channel access mode in IEEE 802.11 and its implementation is mandatory for all devices that are compliant to the standard. The channel access method provided is the common Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA) probabilistic method.

The first important feature of this method is the Carrier Sense (CS): each STA, before transmitting, has to sense the channel to understand if other STAs are transmitting. The sensing operation can be done in a "physical" way as well as in a "virtual" one. The former uses services provided by the underlying PHY layer. The latter is based on the exchange of frames containing information about the traffic and will be discussed later. The information provided by the two sensing modes (physical and virtual) can be combined to have a detailed knowledge of the channel state.

Each STA that has data to transmit, must check that the channel remains idle for a specific amount of time, called DCF Inter Frame Space (DIFS). If this is the case, the transmission can begin. Conversely, if the channel is sensed as busy, transmission is deferred and the channel sensing continues until the end of the transmission that was occupying the channel. After that, the STA has to check again if the channel remains idle for the duration of a DIFS, then the Contention Window (CW) begins, when all STAs that have data to transmit can attempt to access the channel. To avoid collisions, each STA initializes (or resume) a timer, whose duration is a random variable, called *random backoff time*. While this timer decreases, the STA continues channel sensing, stopping the timer if activity is sensed on the channel. If the timer expires, the STA gains access to the channel and can start transmission. Fig. 3.3 summarizes the CSMA/CA algorithm performed by each STA in order to transmit a frame.

Another important feature of DCF procedure is that each transmission of a data frame must be *acknowledged*, which means that a specific frame, called Acknowledgment (ACK) frame, must be sent by the receiver to the transmitter upon the successful reception of the data frame. The period that the receiver waits after the end of reception and the beginning of ACK transmission is called Short Inter Frame Space (SIFS). The acknowledgement mechanism is used by STA to verify the success of transmissions and detect the presence of collisions. Indeed, when a STA has completed the transmission of a frame, a timeout

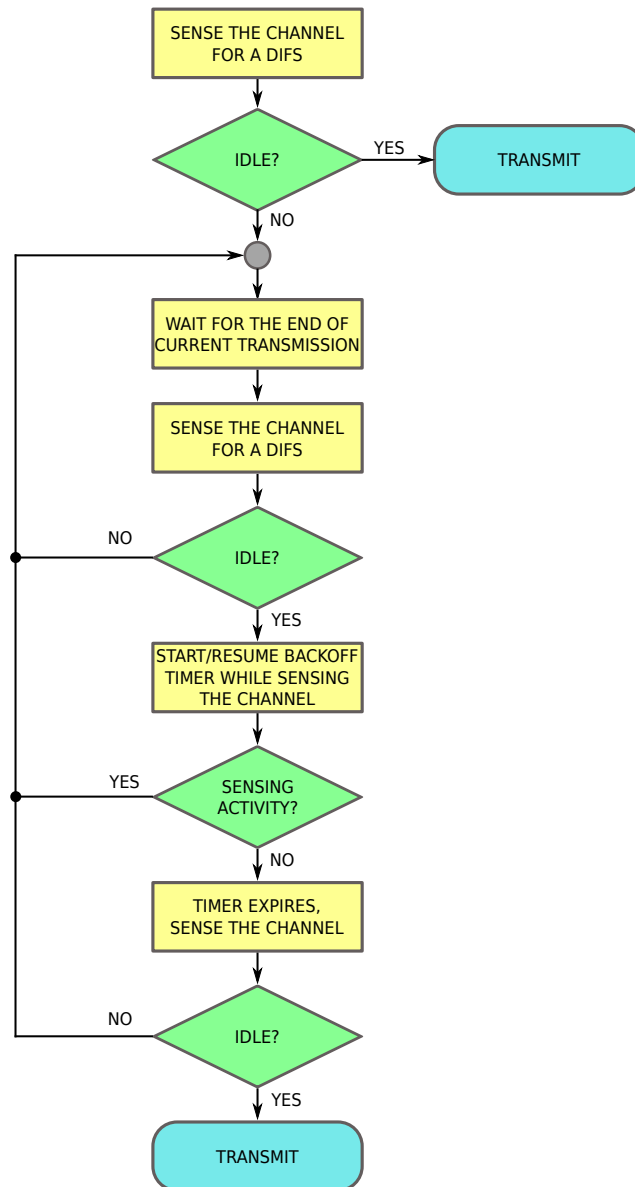


Figure 3.3 – Flow diagram of the channel access procedure with DCF.

procedure is started. If the corresponding **ACK** frame is not received within the expiration of the timeout, a collision is detected ¹. When a collision is detected, the **STA** starts a random backoff timer and, after its expiration, attempts to access the channel to retransmit the data frame that was subjected to a collision. The IEEE 802.11e version of the standard introduced the Block Ack feature, according to which several acknowledgements relative

¹This may be caused both by an error on the data frame reception or by an error on the **ACK** frame reception

to different data frames can be aggregated in a single ACK frame, thus improving channel efficiency.

Random backoff time

When a collision is detected or when a STA senses the channel as idle after the end of a busy period, a random period is waited in order to avoid that different STAs transmit data simultaneously. This period, called random backoff time, is computed as:

$$\text{BackoffTime} = \text{randInt}() \times \text{slotTime} \quad (3.1)$$

The backoff time is, therefore, a multiple of a fixed quantity, namely the *slot time*, whose value depends on the adopted PHY layer. This is multiplied by a pseudo-random integer drawn from a uniform distribution over the interval $[0, CW]$. CW is a parameter, called Contention Window, which is bounded by the two values CW_{min} and CW_{max} , that are related to the PHY layer used.

In particular, the initial value of CW is set to CW_{min} , then, each time a transmission failure is detected, it is updated to $2CW + 1$. In case of several consecutive failures, the update process continues until the CW_{max} value is reached, then CW value remains constant. Fig. 3.4 describes the exponential increase of CW with the number of consecutive transmission attempts, assuming $CW_{min} = 15$ and $CW_{max} = 1023$.

At the first successful transmission, CW is restored to CW_{min} value. The update process relies on an internal counter kept by each STA, namely the STA Short Retry Count (SSRC), which keeps track of consecutive transmission failures. There is also another counter, the STA Long Retry Count (SLRC), which is used with if the Request To Send (RTS)/Clear To Send (CTS) mechanism is implemented. The standard also specifies a maximum value that the SSRC can reach, after which the update process stops and the transmission is considered as permanently failed.

Inter Frame Spaces

Several time intervals are introduced by the standard in the description of the channel access method. Time intervals between two consecutive operations concerning the transmission of frames are called Inter Frame Spaces (IFSs). The different IFSs are related to the adopted PHY layer, although they are independent of the STA bit rate. Their duration is inversely proportional to the priority of the frame that have to be sent.

Six different IFSs are defined in the most updated version of the standard:

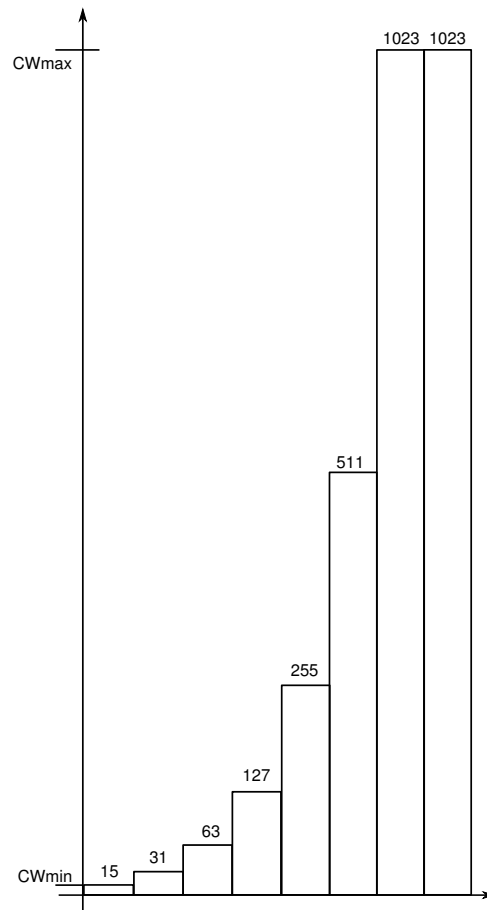


Figure 3.4 – Exponential increase of the CW counter.

- ✓ **Reduced Inter Frame Space (RIFS):** this is the shortest possible IFS, introduced in IEEE 802.11n version of the standard [13, Clause 20]. Can be used in some occasions to separate the transmission of multiple frames from a single transmitter.
- ✓ **Short Inter Frame Space (SIFS):** this IFS is used for ACK frames, CTS frames, a fragment which is not the first in a fragment burst and by a STA responding to a polling request in PCF mode.
- ✓ **PCF Inter Frame Space (PIFS):** this IFS is used by CF-pollable STA to gain access to the medium during the CFP in the PCF procedure.
- ✓ **DCF Inter Frame Space (DIFS):** this IFS is used in the DCF procedure to gain access to the medium.

- ✓ **Arbitration Inter Frame Space (AIFS):** this IFS is used in the Enhanced Distributed Channel Access (EDCA) procedure by QoS STAs, in order to prioritize different categories of traffic.
- ✓ **Extended Inter Frame Space (EIFS):** this is the longest possible IFS. It is used if a STA receives a packet but can not identify its content, so is not able to gather information on the occupation of the medium.

Fig. 3.5 summarizes the different types of IFSs.

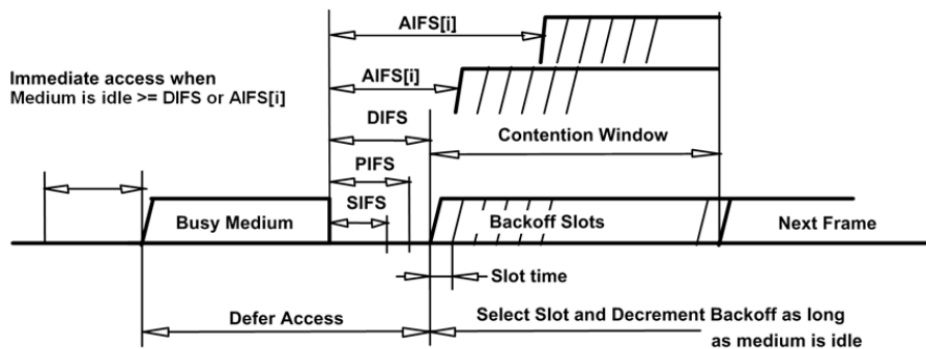


Figure 3.5 – Relationships between different IFSs.

RTS/CTS mechanism

In addition to the "physical" channel sensing performed by devices at PHY layer, a "virtual" investigation of the channel conditions can be achieved through the exchange of RTS and CTS frames. A STA that wishes to transmit sends a RTS frame to the potential receiver, which replies with a CTS frame. If the CTS frame is not received within a certain time interval, the RTS frame is retransmitted using a random backoff procedure. If the RTS/CTS exchange is successful, the transmission of data frame starts after a SIFS.

RTS and CTS frames are received also by other STAs not directly involved in the exchange, which use these information to inspect the state of the channel. In particular, each STA keeps a Network Allocation Vector (NAV), which stores information advertised by the RTS/CTS frames. In fact, these frames include a duration field that specifies the time required for the transmission of data frame and corresponding ACK, so a timer can be set up which indicates a free channel upon its expiration.

The RTS/CTS mechanism adds a significant overhead to the transmission of frames and in many cases can be avoided. Indeed, each STA defines a *RTSthreshold* and adopts this

mechanism only if the size of the frame to transmit exceeds this threshold. If this threshold is set to a value above the maximum PDU size, the mechanism is always disabled, while it is always employed if the threshold is set to 0. In the application described in this thesis, the RTS/CTS procedure is never used.

3.2.2 Fragmentation of frames

An important functionality provided by the IEEE 802.11 MAC is the possibility to split a large information unit in smaller frames that are sent independently over the wireless channel. This may be necessary in the case of poor quality channels, since the transmission of smaller frames is less subject to errors. The receiver has to carry out a de-fragmentation process once all the frames have been captured, in order to reconstruct the initial information unit.

The initial information unit can be a MAC Service Data Unit (MSDU) or a MAC Management Protocol Data Unit (MMPDU), depending on its content (data or management information), while the frame actually sent over the channel is called MAC Protocol Data Unit (MPDU). The fragmentation process can be applied only to *unicast* frames.

If an information unit is fragmented in several MPDUs, the transmitter sends what is called a *fragment burst*. The transmission of two consecutive fragments in the same burst is separated by a SIFS (or a RIFS), so the source does not have to contend with other STA for channel access and can transmit all frames without interruptions. Fig. 3.6 describes the transmission of a fragment burst.

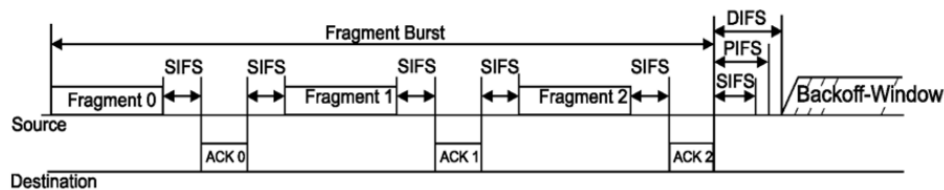


Figure 3.6 – Transmission of a burst of fragment.

3.2.3 Multirate support

Some types of PHY have multiple data transfer rate capabilities, so that dynamic rate switching can be performed with the objective of improving performance. The motivation for this behavior arises from the intrinsic nature of the wireless channel, which is characterized by fading, shadowing, interference ecc. Moreover, the channel conditions are not stationary but may change greatly over time, so an adaptive mechanism is required.

The most common technologies at PHY layer define a wide range of modulation schemes that can be employed, offering different levels of robustness and quickness in data delivery. There is a trade-off, in the sense that the modulations that provide the highest data rates also imply a low communication robustness, and viceversa. Fig. 3.7, based from the analysis in [25], explicits this result, showing the success probability on a 46 Bytes packet against the channel SNR, adopting the various data rates offered by the IEEE 802.11g version of the standard. It is evident that a lower transmit rate guarantees a higher success probability, for a given SNR.

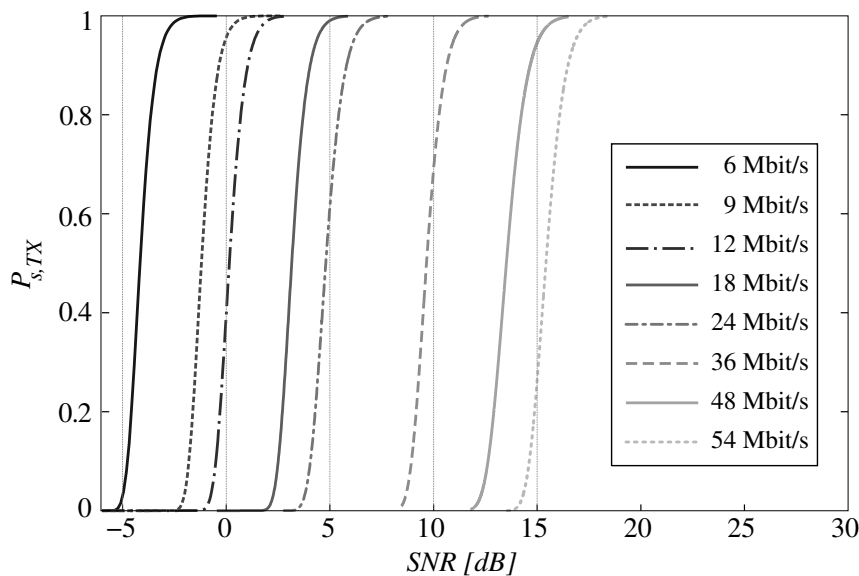


Figure 3.7 – Success probability on a 46 bytes packet versus channel SNR for IEEE 802.11g [25].

The adaptive mechanism used to dynamically select the most appropriate transmit rate is called Rate Adaptation (RA). The IEEE 802.11 standard [13] does not specify any RA technique, leaving to the device manufacturer the choice of a proper implementation. As a consequence, several RA techniques have been proposed in literature and can be divided in two main groups according to their features.

The first category contains RA techniques in which the rate is chosen considering the result of previous communication history: a sequence of consecutive successful transmissions lead to an increase in data rate, while a series of failures causes it to decrease. The leading technique in this category is the Automatic Rate Fallback (ARF), proposed for the first time in [15] for devices compliant with the original version of the IEEE 802.11 standard (WaveLAN-II devices). Several modifications to this technique have been studied, such as the Adaptive Automatic Rate Fallback (AARF) [16], Fast rate reduction Automatic

Rate Fallback (FARF) and Static retransmission rate Automatic Rate Fallback (SARF) [27].

The second category of RA techniques includes strategies based on the dynamic estimation of the channel SNR, in order to choose an optimal transmission rate and possibly modify it as the channel conditions change. The first technique proposed in this category was the Receiver-Based Auto Rate (RBAR), presented in [11], which uses the RTS/CTS mechanism to estimate the channel conditions. An alternative algorithm is the Received Signal Strength Link Adaptation (RSSLA) [6], in which the channel estimation is based on the Received Signal Strength (RSS) indicator.

In an industrial context, characterized by small size frames and low latency requirements, the first group techniques are usually preferred, to avoid any communication overhead with respect to the useful data. In particular, the ARF technique is the most widely accepted rate adaptation between the manufacturers of IEEE 802.11 devices [12].

3.2.4 MAC frame format

A frame at MAC level in IEEE 802.11 is composed by three basic components: a MAC header, a variable-length frame body and a Frame Check Sequence (FCS). The general frame format is depicted in Fig.3.8.

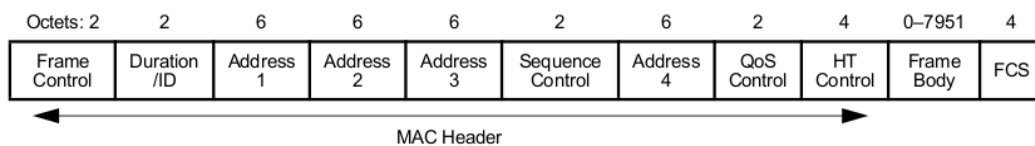


Figure 3.8 – Format of a IEEE 802.11 frame at MAC level.

The first three fields (Frame Control, Duration/ID and Address 1) and the last field (FCS) make up the minimal frame format and are present in all IEEE 802.11 MAC frames, while all the other fields are present only in some types and subtypes of frames. The meaning of each field is briefly discussed in the remainder of this Section.

Frame Control

This field is 2 bytes long and is further subdivided in many subfields, as illustrated in Fig. 3.9.

A brief explanation is given here for each subfield.

Protocol Version Indicates the version of the standard. Currently set to 0, all other values are reserved for future use.

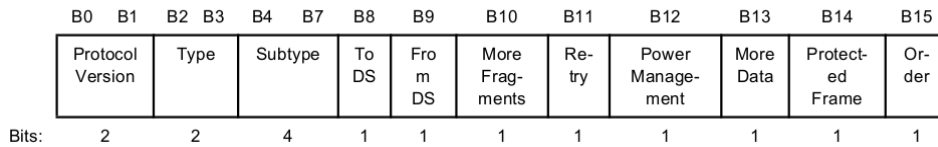


Figure 3.9 – Structure of Frame Control field.

Type and Subtype Together these two subfields point out the function of the MAC frame. The Type subfield (2 bits) can assume the following values: 00 (Management), 01 (Control), 10 (Data). The Subtype value further specifies the function of the frame. Table 3.1 presents some of the possible combinations of these two subfields.

Table 3.1 – Some possible combinations of Type and Subtype fields.

Type value B3 B2	Type description	Subtype value B7 B6 B5 B4	Subtype description
00	Management	0000	Association Request
00	Management	0001	Association Response
00	Management	1000	Beacon
		...	
01	Control	1011	RTS
01	Control	1100	CTS
01	Control	1101	ACK
		...	
10	Data	0000	Data
10	Data	0100	Null (No Data)
10	Data	1000	QoS Data
		...	

The values of these two fields determine which of the optional fields that make up the frame will be used and what will their meaning be.

ToDS and FromDS Indicates the direction of the frame with respect to the Distribution System. Possible combinations are:

- ✓ ToDS=0, FromDS=0: data frame directed from a STA to another one as well as all management and control frames.
- ✓ ToDS=1, FromDS=0: data frame destined to the DS.

- ✓ ToDS=0, FromDS=1: data frame coming from the DS.
- ✓ ToDS=1, FromDS=1: data frame used only in mesh networking.

The values of this fields determine how many Address fields will be present in the frame.

More Fragments Set to 1 if more fragments belonging to the same fragment burst are coming after the current frame. Set to 0 otherwise.

Retry Set to 1 if this frame is actually being retransmitted after previous transmission attempt resulted in a failure. Set to 0 otherwise.

Power Management Indicates the power management mode of the transmitting STA. If its value is 1, it points out that the transmitting STA is entering Power Save (PS) mode. If its value is 0, it indicates that the transmitting STA is active. In many types of frames, this bit is reserved [13].

More Data Set to 1 to notify to a STA in PS mode that more information unit destined to it are being buffered at the AP.

Protected Frame Set to 1 if the content of the frame body has been encrypted with some cryptographic encapsulation algorithm. This bit can be set to 1 only for data frames and management frames dealing with authentication issues.

Order Set to 1 only in specific cases concerned with QoS handling [13]. Set to 0 otherwise.

Table 3.2 – Encoding of the Duration/ID field.

Bit 15	Bit 14	Bits 13–0	Meaning
0		0–32767	Length of the frame
1	0	0	Fixed value within frames transmitted during the CFP
1	0	1–16383	Reserved
1	1	0	Reserved
1	1	1–2007	Association IDentifier (AID) in PS–Poll frames
1	1	2008–16383	Reserved

Duration/ID

This field is 2 bytes long and generally indicates the duration (in microseconds) of the transmission of the frame, to be used to update the NAV of receiving STAs. Can be encoded in different ways to display different information according to the frame type. Encoding possibilities are shown in Table 3.2.

Address Fields

These fields contain 48 bits addresses as defined in [13]. Each address can be of the following types:

- ✓ *Individual address*: the address assigned to a specific STA.
- ✓ *Group address*: a multideestination address, further divided in:
 - Multicast–group address*: an address associated with a group of STAs.
 - Broadcast address*: a distinguished address denoting all STAs in a LAN.

There are four possible slots for addresses in each frame. How many of these will be used and which depend on the Frame Control field and, more specifically, on the Type, Subtype, ToDS and FromDS values. The addresses that can be reported in these slots are the following:

- ✓ **Basic Service Set Identification (BSSID)**: uniquely identifies a BSS.
- ✓ **Destination Address (DA)**: identifies the final recipient of the MSDU or fragment being transmitted.
- ✓ **Receiver Address (RA)**: identifies the immediate recipient of the MPDU being transmitted.
- ✓ **Source Address (SA)**: identifies the initial sender of the MSDU or fragment being transmitted.
- ✓ **Transmitter Address (TA)**: identifies the immediate sender of the MPDU being transmitted.

Sequence Control

This field is 2 bytes long. The first 4 bits are dedicated to the Fragment Number (FN) and the remaining 12 bits to the Sequence Number (SN). The SN is unique for each MSDU or

MMPDU, so it remains the same for different fragments or different transmission attempts of the same frame. The FN is 0 if there is only one fragment, otherwise it indicates the number of this fragment with respect to the original MSDU.

QoS Control

This field is 2 bytes long and identifies the Traffic Class (TC) or Traffic Stream (TS) to which the frame belongs. It also contains various other QoS-related and mesh-related information about the frame, according to the frame type and the type of transmitting STA.

HT Control

This field is 4 bytes long and it is present only in those QoS data, control and management frames whose Order bit in Frame Control fields is set to 1, as reported in [13]. It contains several information and parameters used to enhance the throughput of the network, as specified in the IEEE 802.11n version of the standard.

Frame Body

This field has a variable length and contains information which depend on the specific frame Type and Subtype. The minimum length of the frame body is 0 bytes, while the maximum length is defined by the maximum length of the MSDU (2304 bytes) plus the length of Mesh Control field (up to 18 bytes). This maximum value can be increased if Aggregated MAC Service Data Unit (A-MSDU) is used, a structure containing multiple MSDUs, transported within a single MPDU. There can also be some overhead due to security encapsulation, reaching a maximum size of 7951 bytes.

Frame Check Sequence

This is a 4 bytes field containing a CRC, calculated over all the fields belonging to MAC Header and Frame Body. The generator polynomial is:

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

3.3 PHY layer overview

The IEEE 802.11 standard provides specification for the PHY layer in WLAN networks. Different implementations of this layer are considered according to the underlying physical medium, which can be of various types. However, the structure of this layer is unique and composed by the following items:

1. A Physical Media Dependent (PMD) part, which defines the characteristics of data transmissions over a specific physical wireless medium.
2. A PHY convergence function, indicated with Physical Layer Convergence Procedure (PLCP), which provides the MAC layer with an interface to the specific PMD system. Basically, a mapping of the formed MPDU into a format suitable for the associated physical system is performed.

Each PMD may require the definition and implementation of a unique PLCP. It can also happen that services required to the PHY layer are already provided by the PMD sublayer itself, so that no convergence function is necessary.

Similarly to MAC layer, a management entity is also present, called Physical Layer Management Entity (PLME). Together, PHY and PLME offer a series of functionalities to the overlying MAC layer. The two most important functionalities offered are:

- ✓ **Transmission of frames:** this layer performs the actual transmission over the physical medium of the MPDUs and notifies to the MAC layer the reception of MPDUs from other STAs.
- ✓ **Clear Channel Assessment (CCA):** several methods for sensing the channel and detect the possible presence of activity are provided by this layer, so Carrier Sense Multiple Access (CSMA) techniques can be adopted at MAC layer.

3.3.1 Types of physical media

The IEEE 802.11 standard allows the adoption of several different physical media. The most relevant examples are:

- ✓ **InfraRed (IR):** uses near-visible light in the 850 nm to 950 nm range.
- ✓ **Frequency Hopping Spread Spectrum (FHSS):** uses the 2.4 GHz frequency band (Industrial, Scientific and Medical (ISM) band).

- ✓ **Direct-Sequence Spread Spectrum (DSSS)**: also uses the 2.4 GHz band.
- ✓ **High Rate/Direct-Sequence Spread Spectrum (HR/DSSS)**: extension of DSSS, still operating in the 2.4 GHz band.
- ✓ **Orthogonal Frequency Division Multiplexing (OFDM)**: uses the 5 GHz ISM band.
- ✓ **Extended Rate PHY-OFDM (ERP-OFDM)**: extension of OFDM, uses the 2.4 GHz band.
- ✓ **High Throughput (HT)**: can use both the 2.4 GHz and the 5 GHz bands.

The IR and FHSS media were used in the first edition of the standard, also known as *IEEE 802.11 legacy*, and are no longer maintained, so compatibility with more recent features may be an issue. HR/DSSS, defined in the IEEE 802.11b version of the standard, and ERP-OFDM, defined in the IEEE 802.11g version, are by far the most widespread choices nowadays and will be better explained in Section 3.4 and 3.5. Finally, HT, defined in the IEEE 802.11n version of the standard, represents the most recent upgrade and will be briefly discussed in Section 3.6.

3.3.2 Transmission power

Limitations for the transmitted power of IEEE 802.11 compliant devices vary greatly over different countries. In Europe, regulations are provided by the European Telecommunication Standards Institute (ETSI), which imposes limits on the Equivalent Isotropically Radiated Power (EIRP). The following limitations apply for ISM bands:

- ✓ **2.4 GHz band**: EIRP limited to 100 mW (20 dBm).
- ✓ **5 GHz band**: limitations vary over different countries, ranging from 50 mW to 200 mW.

If the transmitted power exceeds these limitations, appropriate Transmit Power Control techniques must be adopted by the PMD sublayer.

3.3.3 Clear Channel Assessment

The physical layer of IEEE 802.11 WLANs must provide methods for sensing the communication channel in order to establish if other devices are transmitting. The channel is

sensed as *busy* if another transmission is recognized, otherwise it is sensed as *idle*. This information is handed to the MAC layer, which will use it (together with other information on channel occupation) to perform CSMA algorithm.

There are several methods foreseen by the standard to detect the state of the channel. In particular, four different CCA modes are defined:

- ✓ **CCA Mode 1: Energy above threshold** The medium is sensed as busy upon detection of any energy above a certain threshold, called Energy Detection (ED) threshold, dependent on the employed physical medium.
- ✓ **CCA Mode 2: Carrier Sense** The medium is sensed as busy upon detection of a standard compliant signal, which may be above or below the ED threshold.
- ✓ **CCA Mode 3: Carrier Sense and energy above threshold** The medium is sensed as busy if a standard compliant signal with energy above the ED threshold is detected.
- ✓ **CCA Mode 4: Carrier Sense with timer** A timer is initialized and the medium is sensed as idle if no standard compliant signal is detected before the expiration of the timer.

Not all the available modes are employed by every amendment of the IEEE 802.11 standard, but their use greatly vary according to the physical medium employed. Table 3.3 reports the employed modes for every physical technology analyzed.

Table 3.3 – CCA Modes employed by different physical media.

Medium	Mode 1	Mode 2	Mode 3	Mode 4
IR	✓	✓	✓	✓
FHSS	✓	✓	✓	
DSSS	✓	✓	✓	
HR/DSSS	✓		✓	✓
OFDM			✓	
ERP-OFDM			✓	
HT	✓	✓	✓	

3.4 IEEE 802.11b (HR/DSSS)

The IEEE 802.11b version of the standard [13, Clause 17] specifies the High Rate extension of the PHY for the DSSS system in the 2.4 GHz band designated for ISM applications.

Higher rates are provided with respect to the original DSSS PHY while occupying the same channel bandwidth thanks to new modulation schemes and a new organization of the PDU.

The ISM band is precisely defined in the 2.4–2.4835 GHz range, which is subdivided into 14 channels, whose numbers and center frequencies are reported in Table 3.4.

Table 3.4 – HR/DSSS channels.

Channel number	Center frequency [MHz]
1	2412
2	2417
3	2422
4	2427
5	2432
6	2437
7	2442
8	2447
9	2452
10	2457
11	2462
12	2467
13	2472
14	2484

As it can be seen on the table, center frequencies are 5 MHz spaced (except for channel 14). Each channel has a 22 MHz bandwidth, so if multiple WLANs coexist in the same area, there can be only three carriers employed spaced by 25 MHz, to limit reciprocal interference. A typical choice is to use channels 1, 6 and 11, as illustrated in Fig. 3.10. Not all the channels are used worldwide: channel 14 is used only in Japan, while in North America only the first 11 channels are employed.

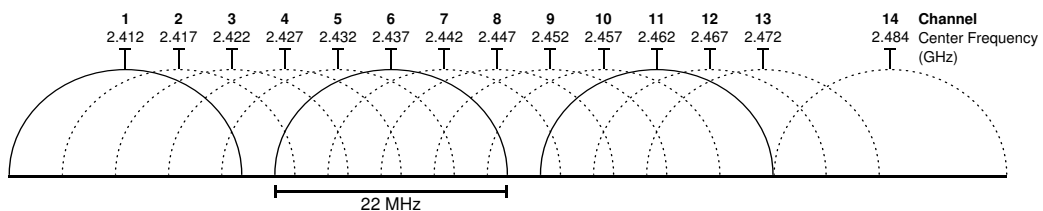


Figure 3.10 – Disposition of WLAN channels in the ISM band.

3.4.1 Modulation and Data rate

The original DSSS system used baseband modulations like Differential Binary Phase Shift-Keying (DBPSK) and Differential Quadrature Phase Shift-Keying (DQPSK) to provide data rates of 1 Mb/s and 2 Mb/s. These modulations are still adopted in the High Rate extension and, in addition, Complementary Code Keying (CCK) is employed to achieve higher data rates of 5.5 Mb/s and 11 Mb/s.

An optional mode replacing the CCK modulation is provided, which makes use of Packet Binary Convolutional Coding (PBCC). However, this option is obsolete and may be removed in a later revision of the standard.

Spread spectrum techniques based on coding are used to enhance SNR, limiting interferences and allowing multiple users to transmit on the same frequency range. In particular, a 11-chip Barker sequence is used as a Pseudo-Noise (PN) sequence for lower data rates (1 Mb/s and 2 Mb/s), while a faster 8-chip spreading code based on complementary codes is employed for higher rates with CCK.

3.4.2 PPDU format

A typical packet at PHY layer, called PLCP Protocol Data Unit (PPDU), is composed by three fields: PLCP preamble, PLCP header and PLCP Service Data Unit (PSDU). The latter one is merely the MPDU generated by the MAC layer and it is actually transmitted at the selected data rate, while the first two fields are transmitted at lower data rates to ensure they are received without errors.

The High Rate extension of DSSS provides an optional mode that allows data throughput at the higher rates (2, 5.5, and 11 Mb/s) to be significantly increased by using shorter PLCP preamble and header. The use of this mode can compromise interoperability with older devices realized for DSSS communication, so its employment is still optional. However, the great majority of the devices in use nowadays support this feature.

PPDU format is illustrated in Fig. 3.11 considering the short preamble option.

The PLCP preamble, transmitted at the lowest possible rate, is made up of a SYNC field, which consists of 56 scrambled zero bits, used by the receiver for synchronization purposes, and a 16 bits Synchronization Frame Delimiter (SFD), which is the reverse sequence of the SFD used within the long preamble.

The PLCP header is transmitted at 2 Mb/s and composed by four fields:

- ✓ SIGNAL (8 bits): indicates the modulation employed for the payload.

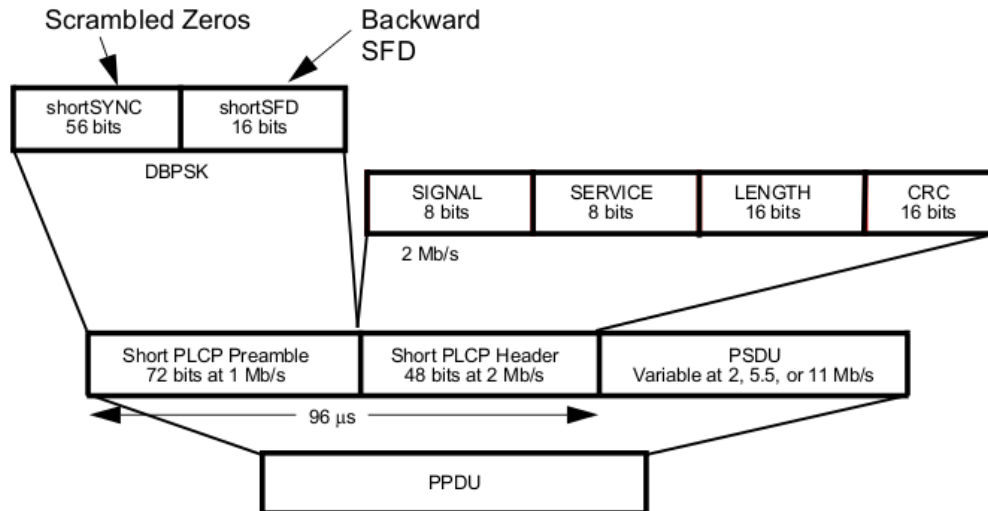


Figure 3.11 – Format of HR/DSSS PPDU adopting short preamble and header option.

- ✓ SERVICE (8 bits): carries some information, such as the optional choice of PBCC instead of CCK.
- ✓ LENGTH (16 bits): indicates the number of microseconds required to transmit the payload.
- ✓ CRC (16 bits): contains a CRC calculated on the three previous fields.

3.4.3 Transmission time

The transmission time of a HR/DSSS PPDU expressed in microseconds can be calculated according to

$$T_{TX} = T_{preamble} + T_{header} + \left[\frac{(l + PBCC) \times 8}{DataRate} \right] \quad (3.2)$$

where l is the length, in Bytes, of the PSDU, while $DataRate$ represents the selected rate for payload transmission (in Mb/s). $PBCC$ is equal to 1 if this option is used or 0 otherwise.

The values of $T_{preamble}$ and T_{header} for the short preamble option are reported in Tab. 3.5, together with the values assumed in HR/DSSS by some typical parameters whose meaning has been discussed in Section 3.2.

Table 3.5 – HR/DSSS parameters.

Parameter	Description	Value
T_{SLOT}	Duration of slot time	20 μ s
T_{SIFS}	Duration of SIFS	10 μ s
T_{DIFS}	Duration of DIFS	50 μ s
CW_{min}	Minimum value of CW	31
CW_{max}	Maximum value of CW	1023
$T_{preamble}$	PLCP preamble duration	72 μ s
T_{signal}	SIGNAL field duration	24 μ s

3.5 IEEE 802.11g (ERP-OFDM)

The Extended Rate PHY specification, defined in Clause 19 of the IEEE 802.11 standard [13], and commonly known as IEEE 802.11g, exploits the potential of the OFDM modulation in order to provide high transmission rates, up to 54 Mbit/s.

There are several similarities with the HR/DSSS physical layer described in Section 3.4 and many expedients have been adopted to ensure interoperability between these two types of physical media. As a result, nowadays almost any portable device has an IEEE 802.11b/g interface, compliant with these technologies.

The frequency band adopted is the same of HR/DSSS, that is the 2.4 GHz ISM band. The subdivision in channels remains unchanged, following that depicted in Fig. 3.10.

3.5.1 Modulation and Data rate

To ensure compatibility with IEEE 802.11b, the DBPSK, DQPSK and CCK modulation schemes can be adopted, obtaining data rates of 1, 2, 5.5 and 11 Mb/s. There is still the possibility of using PBCC modulation, although it is obsolete and might be removed. Moreover, a new set of data rates has been added thanks to the adoption of the OFDM modulation.

OFDM is a multi-carrier modulation method, in which many orthogonal sub-carrier signals are adopted to carry data in parallel, each one modulated with a traditional modulation scheme. Specifically, 52 sub-carriers are employed, 48 of which are actually used for data while 4 are "guard" subcarriers. Different modulation schemes can be used to modulate sub-carriers in order to achieve higher data rates, such as Binary Phase Shift-Keying (BPSK) or Quadrature Phase Shift-Keying (QPSK) for lower rates and Quadrature Amplitude Modulation (QAM) for higher ones. Moreover, a Forward Error Correction (FEC) technique is adopted, exploiting convolutional coding to enhance robustness.

The possible choices for modulation and coding schemes are summarized in Tab. 3.6, together with the resulting data rates and number of bytes carried with a single OFDM symbol. It can be observed that data rates of 6, 9, 12, 18, 24, 36, 48 and 54 Mb/s can be achieved. In particular, a compliant device must implement at least the transmission rates 6, 12 and 24 Mb/s, to ensure interoperability.

Table 3.6 – Modulation and Coding schemes available with ERP-OFDM.

Mode	Data Rate [Mb/s]	Modulation	Code rate	N _{BPS}	Ack Rate [Mb/s]
1	6	BPSK	1/2	3	6
2	9	BPSK	3/4	4.5	6
3	12	QPSK	1/2	6	12
4	18	QPSK	3/4	9	12
5	24	16-QAM	1/2	12	24
6	36	16-QAM	3/4	18	24
7	48	64-QAM	2/3	24	24
8	54	64-QAM	3/4	27	24

It is worth noting that, if a device is able to adopt all these modes, a IEEE 802.11 *data* frame can be transmitted at any of these rates. However, the *control* frames (such as the acknowledgement frames) must be transmitted at one of the rates in the basic rate set, which is the set of rates that each host in the network has to support, as reported in the table.

3.5.2 PPDU format

A device compliant to Extended Rate PHY specification shall support three different PPDU formats, to ensure backward compatibility with IEEE 802.11b. The first two are those described in the HR/DSSS amendment, both the version with the long preamble and header and that with the short ones. The choice of these formats imposes the use of the 1, 2, 5.5 and 11 Mb/s rates.

The third possible format is a new one specifically designed for ERP-OFDM, which allows for the adoption of higher data rates and it is illustrated in Fig. 3.12.

The PPDU starts with a PLCP preamble field, used for synchronization. It consists of 10 short symbols and two long symbols, for a total of 12 OFDM symbols. The preamble is followed by the PLCP header, which includes the following fields:

- ✓ RATE (4 bits): indicates the type of modulation and the coding rate used for the rest of the packet.

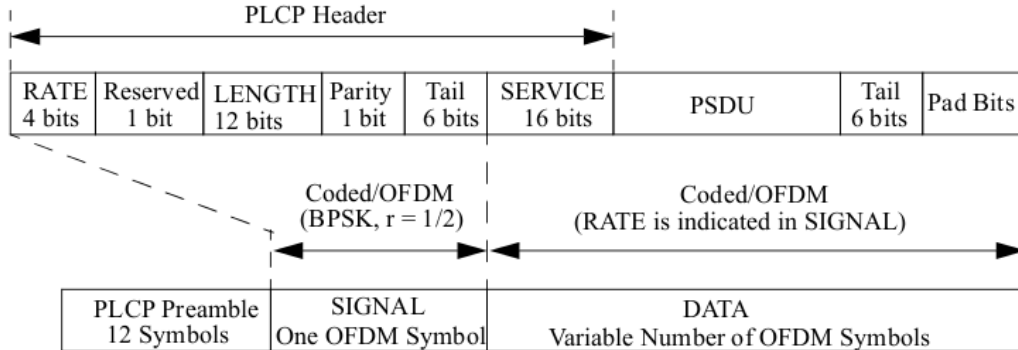


Figure 3.12 – Format of ERP-OFDM PPDU.

- ✓ Reserved (1 bit): a bit reserved for future use, set to 0 on transmit.
- ✓ LENGTH (12 bits): indicates the number of byte of the PSDU to transmit.
- ✓ Parity (1 bit): a parity bit relative to the previous 15 bits.
- ✓ Tail (6 bits): tail bits, all set to zero, which enable immediate decoding of the RATE and LENGTH fields.
- ✓ SERVICE (16 bits): first 7 bits are used to synchronize the descrambling operation, while the remaining bits are reserved for future use.

The header is followed by the PSDU, which is the MPDU delivered to the PHY by the MAC layer, then there are 6 tail bits (set to zero) and optional padding bits.

More specifically, all the fields that form the PLCP header, except the SERVICE field, constitute a separate single OFDM symbol, which is called SIGNAL, transmitted with the most robust scheme, namely mode 1 (BPSK and code rate 1/2). The rest of the packet (SERVICE field, PSDU, tail and padding bits) is called DATA and may constitute multiple OFDM symbols, transmitted with the selected data rate.

3.5.3 Transmission time

The transmission time of a ERP-OFDM PPDU in microseconds can be calculated according to

$$T_{TX} = T_{preamble} + T_{signal} + T_{sym} \cdot \left\lceil \frac{16 + l \times 8 + 6}{N_{BPS} \times 8} \right\rceil + SignalExtension \quad (3.3)$$

where l is the length, in Bytes, of the PSDU, while N_{BPS} is the number of bytes per symbol, as defined in Tab. 3.6.

The parameters $T_{preamble}$ and T_{signal} indicate the duration in microseconds of the transmission of the PLCP preamble and the SIGNAL field and are reported in Tab. 3.7, together with T_{sym} , which indicates the duration of the transmission of each single OFDM symbol in microseconds. The table also reports the values assumed in ERP-OFDM by some typical MAC layer parameters.

Table 3.7 – ERP-OFDM parameters.

Parameter	Description	Value
T_{SLOT}	Duration of slot time	9 μ s
T_{SIFS}	Duration of SIFS	10 μ s
T_{DIFS}	Duration of DIFS	28 μ s
CW_{min}	Minimum value of CW	{15,31}
CW_{max}	Maximum value of CW	1023
$T_{preamble}$	PLCP preamble duration	16 μ s
T_{signal}	SIGNAL field duration	4 μ s
T_{sym}	Duration of a single OFDM symbol	4 μ s

Finally, the *SignalExtension* parameter represents a 6 μ s addition to the transmission time inserted to ensure backward compatibility with IEEE 802.11a version of the standard. Indeed, in this older version the value of SIFS parameter was 16 μ s instead of 10 μ s, because of the high computational effort required by OFDM modulation (particularly the encoding procedure). Therefore, at the end of each transmission, a period of pause called signal extension is added to guarantee interoperability between standards and correct decoding of frames.

3.6 Most recent amendments to the standard

The first draft of IEEE 802.11 standard was published in 1997 and foreseen the use of three different PHY technologies: IR, FHSS and DSSS. Several amendments to this original draft were approved ever since, some of them introducing new technologies to increase throughput. Among these, the most important are:

- ✓ IEEE 802.11a (1999): introduces the OFDM PHY on the 5 GHz band.
- ✓ IEEE 802.11b (1999): enhances the performance of DSSS, introducing HR/DSSS.
- ✓ IEEE 802.11g (2001): extends the OFDM system on the 2.4 GHz band, introducing ERP-OFDM.

- ✓ IEEE 802.11n (2009): introduces HT concepts on both 2.4 GHz and 5 GHz bands.

Other amendments to the standard did not introduce new transmission technologies but presented new structures for better organization of WLAN, such as:

- ✓ IEEE 802.11e (2005): enhancements at MAC layer to introduce the QoS concept.
- ✓ IEEE 802.11s (2012): modifications at network structure to allow for mesh networking.

Some of these amendments have already been discussed in detail. The others, and most recents, will be briefly covered in the rest of this Section, highlighting their potentialities.

3.6.1 IEEE 802.11e

The IEEE 802.11e version of the standard modifies the MAC layer in order to introduce a series of QoS enhancements for WLAN applications, in order to handle different types of traffic with distinct requirements over the same IEEE 802.11 network. The AP and the STA that provide these new kind of services are referred to as QoS Access Point (QAP) and QoS Station (QSTA), while the corresponding BSS is called QoS Basic Service Set (QBSS).

The main novelty introduced in this amendment is the creation of a new CF, namely the HCF, in addition to the DCF and PCF, that did not provide any QoS guarantee or traffic differentiation. This new function defines two additional channel access modes, Enhanced Distributed Channel Access (EDCA) and HCF Controlled Channel Access (HCCA).

The former introduces a prioritization of traffic during CP, allowing some QSTAs to wait less before transmitting than others. This is accomplished by the use of AIFS instead of DIFS, whose duration depend on the type of traffic, and the use of different values for CW. The priority levels are called Access Categories (AC) and are directly mapped to the IEEE 802.1D user priorities [21]. Furthermore, a QSTA can be assigned a Transmit Opportunity (TXOP) period of various length, during which it has unlimited access to the channel. Tab. 3.8 summarizes the different ACs and the default parameters for each level, comparing them to the traditional DCF.

The HCCA mode enhances the PCF, allowing a CFP to be initiated at any time, so that many CFPs can be executed in a row. A more detailed classification on the traffic is foreseen, with the introduction of Traffic Class and Traffic Stream concepts. In this way each node can have a specific priority and also separate traffic sessions pertaining to the same node can be managed differently. There is the need for a central control unit, called Hybrid Coordinator (HC), which corresponds to the AP. The HCCA is the most

Table 3.8 – EDCA Access Categories

AC	AIFS	Maximum TXOP	CW _{min}	CW _{max}
Background (AC_BK)	7	0	31	1023
Best Effort (AC_BE)	3	0	31	1023
Video (AC_VI)	2	3.008 ms	15	31
Voice (AC_VO)	2	1.504 ms	7	15
Legacy DCF	2	0	31	1023

complex channel access method available for WLANs and, although it allows to handle QoS requirements with great precision, it is rarely used by real devices.

To ensure backward compatibility, a QSTA can operate also in a normal BSS and, on the other hand, a normal STA can associate with a QAP in a QBSS, continuing operating as a traditional non-QoS station.

The MAC frame defined in Section 3.2.4 contains some fields which are relevant to the QoS management. The first is the *order* bit inside the Subtype field of the Frame Control, which is set to 1 if the station is a QSTA and 0 otherwise. The other important field is the *QoS control* field, specifically added with this new version of the standard. This field may contain some information on the transmitting STA, such as:

- ✓ The priority of the station, in terms of AC, TC or TS.
- ✓ The station policy for acknowledgements, that can be Normal Ack, Block Ack (single acknowledgement for an entire TXOP) or No Ack (not acknowledged, for highly time-critical applications).
- ✓ Possible limitations to the TXOP.
- ✓ The queue size, *i.e.* the amount of buffered traffic for a given TC or TS at the station, used by the HC to choose the polling order of the nodes.

3.6.2 IEEE 802.11n

The IEEE 802.11n version of the standard, often referred to as High Throughput, offers significantly improved PHY data rates with respect to IEEE 802.11b/g. This is mainly obtained through the use of spatial multiplexing with Multiple Input–Multiple Output (MIMO) multiple antennas systems and double-spaced channels at 40 MHz. Fig. 3.13, taken from [21], shows how significant the increase in data rate is, compared to previous technologies.

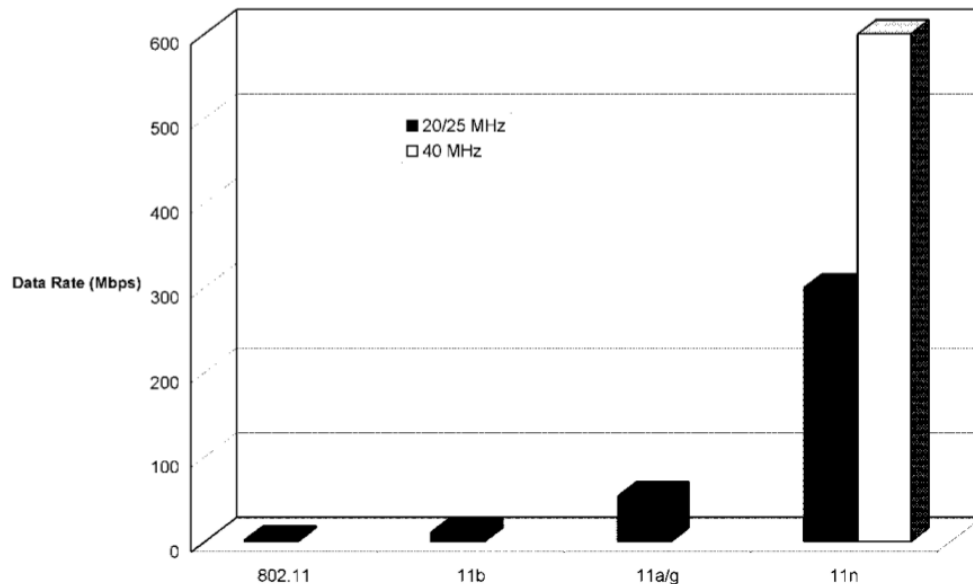


Figure 3.13 – Data rate increase in IEEE 802.11n [21].

The first great improvement introduced in this version of the standard is the Spatial Division Multiplexing (SDM), which allows to spatially multiplex independent data streams using a single frequency channel, thanks to the presence of multiple antennas at both receiver and transmitter. The standard allows the use of up to 4 different spatial streams.

A second substantial improvement is the introduction of 40 MHz bandwidth channels, doubling the typical channel width of previous PHY technologies, thus providing twice the data rate. This feature can be enabled both in the 5 GHz band and in the 2.4 GHz band. A station transmitting on a 40 MHz channel is actually occupying two separate 20 MHz channels, so coexistence issues in a network with 20 MHz and 40 MHz devices must be considered and handled through careful channel selection and possibly alternate operation between the two modes.

Other expedients adopted at PHY layer to improve transmission rates include the use of a reduced guard interval between transmissions (only under certain channel conditions) and the optional adoption of a shorter preamble, namely the Greenfield preamble, which, however is not backward compatible with existing devices. All together, these enhancements allow data rates to increase up to 288.8 Mb/s with 20 MHz channels and 600 Mb/s with 40 MHz channels.

In addition to throughput enhancements, IEEE 802.11n introduces also strategies to improve communication robustness (which is also inherently increased by the use of mul-

multiple antennas). The use of Space-Time Block Coding (STBC) exploits spatial diversity to overcome bad channel conditions, and the robustness is further improved through the use of Low-Density Parity-Check (LDPC) codes instead of convolutional codes. A fast link adaptation mechanism, to rapidly track changing channel conditions and react accordingly is also introduced. Finally, transmit beamforming can also be adopted to achieve spatial selectivity.

In order to take advantage of higher data rates provided by the renewed PHY, MAC layer expedients are also introduced. Specifically, the aggregation of several MSDUs or MPDUs is considered, with the resulting frames called Aggregated MAC Service Data Unit (A-MSDU) or Aggregated MAC Protocol Data Unit (A-MPDU). This allows to reduce the communication overhead, thus increasing the actual data rate perceived by the user. To support these new types of frames, the Block Acknowledgement feature introduced in IEEE 802.11e has been enhanced and made mandatory for all IEEE 802.11n-compliant devices. Finally, a smaller IFS, the RIFS, can be used instead of SIFS to separate the transmission of frames in a burst.

3.6.3 IEEE 802.11s

The IEEE 802.11s version of the standard introduces a new network topology for WLANs, namely *mesh network*, to cope with the demand for an increased devices mobility. In this topology, a STA do not need to be in the same area of its AP, but can gain access to the services through other APs situated nearby. More specifically, the APs are no more connected to a wire infrastructure (the DSM), but can communicate through mesh radio links.

There are several motivations behind the use of a mesh infrastructure. First of all, it enables a faster network development, allowing coverage also in hard-to-wire areas. Then, a mesh topology is inherently self-healing, robust and scalable. Finally, the multi-hop wireless communication allows for a wider coverage range, higher bandwidth and lower power transmission.

Several new network actors are introduced with the new standard. First of all, the Mesh Portal (MPP) is the entity situated at the border of a mesh WLAN, which connects it to an external non-mesh network. The new mesh STA, called Mesh Point (MP), fully participates to the mesh infrastructure, establishing communication links with other MPs. The Mesh Access Point (MAP) has the same functionality of a MP and, in addition, works as an AP, providing BSS services to traditional STAs. Fig. 3.14, taken from [4], summarizes the main actors in new mesh infrastructure.

An important feature that the MP must implement is the topology discovery. This is

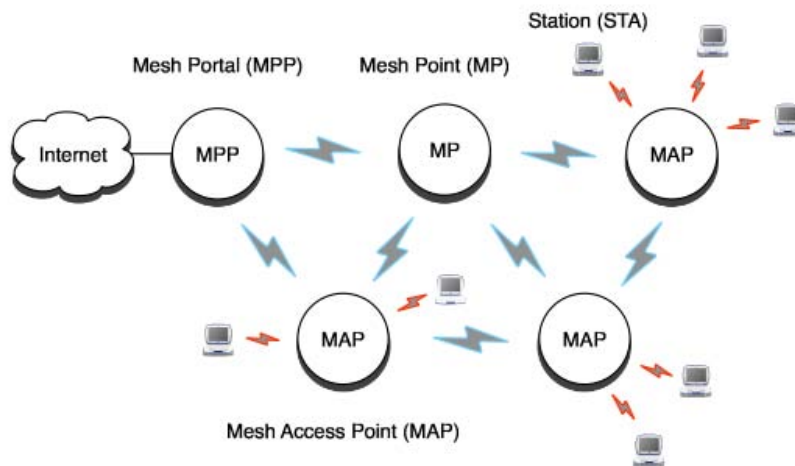


Figure 3.14 – Mesh WLAN components [4].

done through passive scanning (via periodic beacons) or active scanning (via probe requests). The answers, in form of beacons or probe responses, may contain a mesh profile that matches with that of the sending MP, where a profile contains mesh ID and configuration details. If this is the case, a link is established via a secure four-way handshaking.

Another important issue in a mesh topology is routing, which includes the path selection and forwarding of messages. The standard introduces a default routing protocol for interoperability between devices, called Hybrid Wireless Mesh Protocol (HWMP). This protocol combines an on-demand routing algorithm, based on Ad-hoc On-demand Distance Vector (AODV) strategy, for predominantly mobile topologies with a proactive tree-based routing algorithm for predominantly fixed topologies. The implementations of alternative routing protocols is left to the vendors.

Security is also a key concept in wireless mesh networks. Since there are no fixed roles in such networks, two peers exchanging safety-critical information must negotiate security roles and a specific policy before using the communication link between them. Once the roles have been established, the nodes will perform a secure four-way handshaking during which a security key will be generated and used for future communications.

The channel access procedure in mesh networks is handled by the new MCF, which takes care of topology learning, routing and forwarding in such a complex network scheme. The adoption of EDCA as a channel access method is mandatory. In addition, a new method called MCF Controlled Channel Access (MCCA), can be used together with EDCA, allowing to STAs to make reservations for transmissions, in order to avoid collisions.

3.6.4 Amendments to the last standard

Several amendments to the last IEEE 802.11 standard are just been released or currently on development, in order to achieve always increasing performance figures.

The following versions can be cited as the most defined ones:

- ✓ IEEE 802.11ac: published in December 2013, enhances the performance provided by IEEE 802.11n through the adoption of wider channels (80 and 160 MHz), higher number of spatial streams and higher modulations, allowing to reach data rates up to 1300 Mb/s and beyond.
- ✓ IEEE 802.11ad: introduces a new type of PHY which allows for transmission in the 60 GHz millimeter wave spectrum, allowing to reach 7 Gb/s. Devices that implement this standard are indicated with the brand WiGig.
- ✓ IEEE 802.11af: approved in February 2014, this amendment allows operations in the white space spectrum between 54 and 790 MHz (frequency normally used by television transmissions). Appropriate measurements are taken to avoid interference with other signals in this frequency band. High data rates are reachable, thanks to the use of MIMO and OFDM, while offering a larger coverage range, because path loss and attenuation is lower in the adopted band.

Chapter 4

Theoretical performance analysis

In order to analyze the behavior of the system described in Chapter 2 and to correctly configure the communication protocol, a theoretical study of the communication system is carried out. The goal of this analysis is to provide an evaluation of the best performance figures that can be reached by the system given its desired behavior and some application-related constraints. These results will be used in the commissioning phase, when several adjustments will be made to the application in order to get as close as possible to the computed theoretical bound.

In particular, with reference to the cyclic polling protocol explained in Section 2.2, a precise assessment of the time needed to complete the polling procedure is provided. This metric describes the timeliness of the communication system. The possible sources of variability and randomness are discussed and appropriate confidence intervals are given for the considered metric. Specifically, the focus is on the worst-case scenario, with the goal of estimating a maximum time limit in which the polling procedure is definitely completed, therefore ensuring a real-time behavior.

Among the various factors that influence the considered metric, wireless communication is the major source of randomness, due to the intrinsic behavior of the MAC layer. Therefore, an extensive analysis of service time in IEEE 802.11b/g is carried out, based on the description of the standard reported in Chapter 3.

THE presented communication protocol belongs to the class of cyclic polling protocols [10]. A typical meaningful metric for these kind of protocols is the time in which a polling cycle is executed.

4.1 Polling time

The main performance index considered in this Chapter is the *polling time* T_p .

It represents the time needed for a successful completion of the polling procedure, which prescribes the exchange of a command message and a state message between the hand-held device and the ECU. More formally, T_p is the time elapsed from the instant in which the application on the remote device starts the transmission of a command message and the instant in which the corresponding state message sent by the ECU is successfully received at the remote device.

This metric is influenced by the protocol design, the architecture of the system, the intrinsic behavior of the components and the conditions of the wireless channel. All these factors have to be taken into account to provide a proper estimation of the range of values that the polling time can assume.

4.1.1 Relation with the cycle period

When describing the cyclic polling protocol, the cycle period T_C has been introduced as the fixed length of a polling cycle. This means that T_C is the upper limit to the time available to the two devices to successfully complete a polling operation. If this operation can not be concluded within this period, the current cycle is considered lost and a new one is started with the transmission of a new command message.

In the light of the definition of polling time, it can be stated that, to avoid any cycle loss, the following inequality has to hold for any polling operation

$$T_p \leq T_C \quad (4.1)$$

It is worth to notice that the cycle period is a fixed parameter, defined during the protocol design phase. Consequently, it is fundamental to estimate an upper bound for the polling time, given all other protocol parameters and expected behavior as constraints. In this way, the cycle period can be set equal to this maximum value, to achieve the best performance for what concern communication timeliness while ensuring that inequality Eq. (4.1) is always fulfilled and therefore cycle loss never happens. Actually, to avoid the effect of components non-idealities and possible approximations in the theoretical model, the cycle period will be set slightly above the computed maximum value of the polling time.

Another possible way to improve the system performance, that will be taken into account later in this work, is to maintain the cycle period T_C fixed to its current value

(250 ms), considered satisfactory from the control perspective, and to exploit the time remained after the end of a polling cycle ($T_C - T_p$) to perform retransmissions of non-acknowledged messages, in order to improve the robustness of the system.

4.1.2 Relation with other metrics

Another performance index that will be addressed by this work, especially during the experimental measurements, is the *interpacket time*, *i.e.* the time elapsed between two consecutive command messages or state messages. This value can be quite easily measured in a laboratory setup and can be registered at different levels (application, transport, data-link), as it will be explained in Chapter 5.

Regarding the interpacket time of command messages, it is clear that this value should be fixed and equal to the cycle period T_C , since the application installed on the hand-held device sends a message at the beginning of each period, regardless of how the ECU reacts.

The interpacket time of state messages, on the other hand, is related to the polling time T_p , whose value is variable. In particular, if the value of this metric is bounded between T_p^{MIN} and T_p^{MAX} , the interpacket time of state messages will assume values in the range $[T_C - \Delta T_p, T_C + \Delta T_p]$, where $\Delta T_p = T_p^{MAX} - T_p^{MIN}$.

It is clear from these results that an analysis of the values that T_p can assume is sufficient to determine the expected theoretical values for the interpacket time, so only the first metric will be taken into account in the following of this Chapter.

4.1.3 Components of polling time

Considering the prototype system defined in Section 2.1, the main elements that affect the time required for the polling procedure can be identified. In particular, the polling time T_p turns out to be the sum of three specific components, summarized as follows:

- ✓ the time necessary for the transmission on the wireless medium of the two messages (command and state), T_{wifi}
- ✓ the time needed for the transmission of both messages on the UART bus which connects the wireless module of the ECU to the ARM controller, T_{uart}
- ✓ the response time introduced by the ECU between the reception of the command message and the corresponding transmission of the state message, T_{d_ECU}

Analytically, it can be hence written

$$T_p = T_{wifi} + T_{uart} + T_{d_ECU} \quad (4.2)$$

In the rest of this Chapter, the three components that make up the polling time are carefully analyzed and the range of values they can assume is provided for the selected protocol parameters. In particular, the variability introduced by each component will be thoroughly taken into account and quantified. Since the analysis of the wireless channel behavior is the most complex one and introduces the major part of randomness, it is left at the end.

4.2 Transmission time on the UART bus

The communication on the UART bus is an example of *asynchronous serial communication*. The word *serial* means that data are sent over the physical channel sequentially, one bit after the other. *Asynchronous* indicates that a start signal is transmitted at the beginning of each character and a stop signal is sent appended at the end. This process, also called *start–stop transmission*, allows for synchronization between transmitter and receiver.

The UART bus installed on the prototype system adopts a character length of 8 bits (1 Byte) and uses 1 bit as a start signal and 1 bit as a stop signal. Fig. 4.1 provides a schematic representation of the process to transmit 1 Byte.

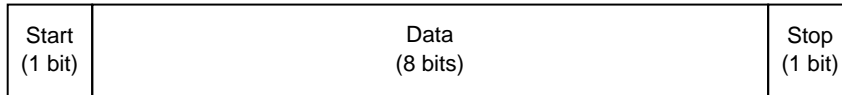


Figure 4.1 – Transmission of a byte over the UART bus.

The time for the transmission of a message over the UART bus is related to the bus baud rate R_{uart} and the message size l , according to

$$T_{mess}(l) = \frac{10}{8} \cdot \frac{l}{R_{uart}} = 1.25 \cdot \frac{l}{R_{uart}} \quad (4.3)$$

where the correction factor 1.25 accounts for the fact that 10 bits are transmitted every 8 useful bits, because of the start–stop mechanism.

It is worth observing that, given the configuration of the system, only the application payload of the command and state messages is transmitted on the UART bus. This differs from what happens on the wireless channel, where the messages contain also the headers

due to lower layers protocols (TCP/UDP, IP, etc.), resulting in an increasing length.

In the system considered, the baud rate is set to $R_{uart}=115200$ bit/s, where the sizes of payloads are defined in Tab. 2.2. The transmission times of the two messages (command and state) on the UART bus can be hence computed, with the results shown in Table 4.1.

Table 4.1 – Transmission times on the UART bus

Message	Size	Transmission time
Command	18 Bytes	1.56 ms
State	257 Bytes	22.31 ms

The overall time for the transmission of data on the UART bus during a polling cycle is therefore $T_{uart}=23.87$ ms. This value is deterministic and not subject to any variability.

4.3 Response time of the ECU

This specific component of the polling time accounts for the delay inserted in the polling procedure by all the components of the ECU, except the UART bus. In particular, it is related to the intrinsic hardware behavior of the ARM controller and the wireless module and the execution time of the program on the controller.

A first source of delay is the behavior of the 32-bit ARMv7 controller mounted on the ECU. The order in which instructions are executed and the execution time of each instruction actually depend on the implementation of the firmware, which is not optimized for real-time applications, so random delays can be added. The two main issues are the interrupt handling and the priority of concurrent task. The controller has two physical interrupt lines, which are multiplexed in several virtual interrupt lines, every one of which can stop the execution of other instructions depending on priority of interrupt channels. Furthermore, the controller can not handle multiple processes simultaneously, so different priorities have to be set in order to ensure that application-related tasks are always executed as soon as possible.

Another issue to consider is the protocol implementation, summarized in Section 2.2. It can be seen that, after receiving the command message from the hand-held device, the ECU remains idle for a fixed amount of time, namely $T_{d_ARM}=50$ ms, before starting the elaboration of the gathered information and the transmission of corresponding state message. This waiting period apparently contributes in a deterministic way to the computation of ECU response time. However, its presence, basically unnecessary for the correct functioning of the polling application, may lead to the emergence of synchronization prob-

lems. Indeed, in this way the response time becomes highly dependent on the way and the precision with which elapsed time is measured by the *ARM* controller, possibly causing malfunctioning and jitters in the communication.

A similar synchronization issue can arise with the intrinsic behavior of the wireless module. In order to trigger the transmission of the state message on the wireless interface, the module has to identify the end of the message on the byte stream coming from the *UART* bus. According to the module configuration, a message is marked as completed if no data were sent over the bus in the last $T_{d_module}=20$ ms. Again, this quantity contributes as a fixed component of the response time but can also lead to the presence of jitter, due to the way time is measured by the wireless module. In the Chapter 6 some measurements which prove this effect will be shown and possible solutions to this issue will be discussed.

In the rest of this Chapter, the variability introduced by synchronization problems and the potential delays due to the non real-time implementation of the firmware are neglected, leaving the discussion on their impact to following chapters. Consequently, the response time is assumed fixed and equal to

$$T_{d_ECU} = T_{d_ARM} + T_{d_module} = 70 \text{ ms} \quad (4.4)$$

4.4 Transmission time on the wireless channel

As discussed in Chapter 3, the time needed for the transmission of messages on the wireless channel is a random variable with a wide range of possible values. Among the factors that influence its behavior, there are the *MAC* protocol operations, the messages sizes, the *SNR* and the external interference.

In this Section, a statistical analysis of the time employed for transmission of messages on the wireless medium for this specific application is performed, on the model of the analysis carried out in [23].

4.4.1 Packets exchange in absence of retransmissions

From the wireless communication perspective, the considered prototype system basically corresponds to an IEEE 802.11 network configured in Infrastructure mode, in which the base station is represented by the *ECU*, while the hand-held device acts as a client *STA*. *DCF* is used as Coordination Function.

A first analysis can be carried out assuming that no other sources are transmitting on the *ISM* frequency band, apart from the *ECU* and the remote device, which exchange

only traffic related to the application. Under this hypothesis, the channel sensing procedure always returns an idle channel and, if the SNR is also assumed sufficiently high, the message transmission is always completed with success at the first attempt, without the intervention of the retransmission mechanism. The exchange of packets at each polling cycle at MAC layer under these assumptions is depicted in Fig. 4.2.

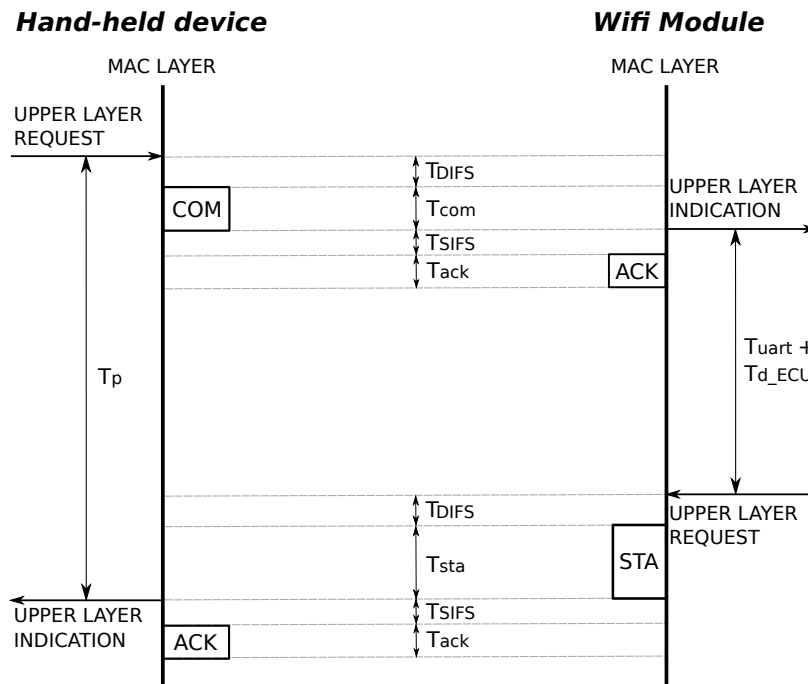


Figure 4.2 – Exchange of packets at MAC layer (ideal case).

Looking at the figure and recalling the definition of polling time Eq. (4.2), the transmission time on the wireless channel in absence of retransmissions can be expressed as

$$T_{wifi} = 2 \cdot T_{DIFS} + T_{com} + T_{sta} \quad (4.5)$$

where:

- ✓ T_{DIFS} is a fixed value which depends on the version of the IEEE 802.11 standard (see Tab. 3.5 and Tab. 3.7);
- ✓ T_{com} and T_{sta} are the transmission times of, respectively, the MAC frames relevant to command and state messages, that can be computed according to either 3.2 (IEEE 802.11b) or 3.3 (IEEE 802.11g).

It is worth observing that the size of the MPDU, essential to compute the message transmission time, is obtained by adding to the length of the application payload (reported in Tab. 2.2), the size of the upper layer headers, summarized in Table 4.2. As an example, if TCP is used as transport layer protocol, the sizes of command and state frames at MAC layer are, respectively, 90 and 329 Bytes.

Table 4.2 – Header sizes for upper layer protocols

Layer	Protocol	Header size
Data link layer	MAC	24 Bytes
Logical link layer	Link Layer Control (LLC)	8 Bytes
Network layer	IP	20 Bytes
Transport layer	TCP	20 Bytes
	UDP	8 Bytes

With these assumptions, the transmission time of the two frames on the wireless channel ($T_{com} + T_{sta}$) is a deterministic quantity, which can be easily computed for different versions of the IEEE 802.11 standard and different choices of the transmit rate, as reported in Tab. 4.3.

Table 4.3 – Transmission times of command and state frames on the wireless channel.

Version	Transmit rate [Mb/s]	Transmission time [ms]
IEEE 802.11b	1	3.544
	2	1.868
	5.5	0.802
	11	0.498
IEEE 802.11g	6	0.62
	9	0.432
	12	0.34
	18	0.244
	24	0.196
	36	0.152
	48	0.124
	54	0.12

It is worth noting that the transmission of additional frames that other protocols could require (such as those relevant to the TCP Acknowledgements) are not represented in Fig. 4.2. Actually, these frames are not taken into account when computing T_{wifi} , since

their transmission occurs while the useful data are being forwarded to the ECU over the UART bus.

4.4.2 Packets exchange with a single retransmission

The exchange of frames illustrated in Fig. 4.2 represents an ideal case, in which both data and acknowledgement frames are successfully delivered at the first attempt. However, as it is well known, wireless transmissions may have non-negligible error rates.

In case a transmission does not succeed, the IEEE 802.11 retransmission mechanism, described in Section 3.2.1, kicks in. Fig. 4.3 illustrates the packet flow at MAC layer in case the transmission of the data frame relative to the command message fails at the first attempt and succeeds at the first retransmission.

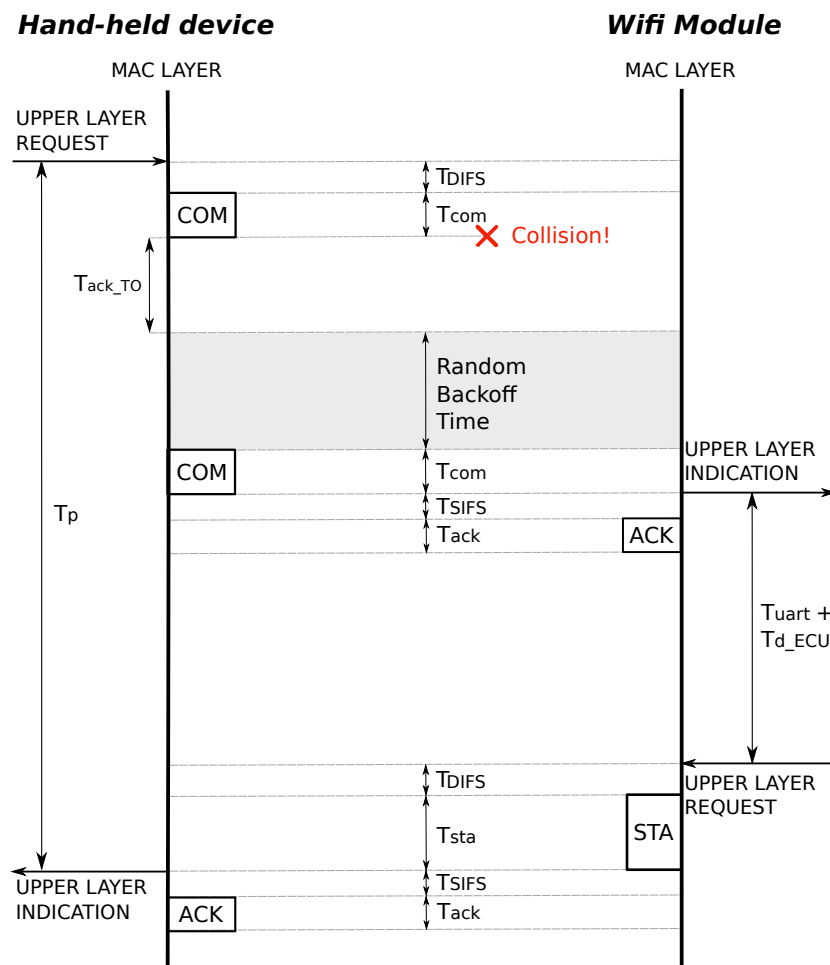


Figure 4.3 – Exchange of packets at MAC layer with the retransmission of a command frame.

With respect to the figure, the transmission time on the wireless channel in this case can be expressed as

$$T_{wifi} = 2 \cdot T_{DIFS} + 2 \cdot T_{com} + T_{ack_TO} + T_{BO} + T_{sta} \quad (4.6)$$

Specifically, T_{ack_TO} is the time that an IEEE 802.11 STA waits for the acknowledgement frame before declaring the loss of the last frame sent and starting the retransmission process. To ensure a proper system behavior, this value is set (by the standard) to

$$T_{ack_TO} = T_{SIFS} + T_{ack} \quad (4.7)$$

where T_{SIFS} is standard dependent (see Tab. 3.5 and Tab. 3.7), while T_{ack} is the time needed for the transmission of an acknowledgement frame, dependent on the data rate employed. As an example, if IEEE 802.11g is used with a 24 Mb/s data rate for acknowledgement packets (the maximum rate allowed by the standard for this type of packets), the transmission requires 34 μ s, therefore $T_{ack_TO}=44 \mu$ s.

Moreover, T_{BO} is the random backoff time that a STA waits before attempting again the transmission. According to Eq. (3.1), for a single retransmission this value is equal to

$$T_{BO} = T_{SLOT} \cdot \mathbf{U} [0, CW_{min}] \quad (4.8)$$

where the letter \mathbf{U} indicates a discrete random variable uniformly distributed in the specified range, while the parameter T_{SLOT} depends on the standard version (see Tab. 3.5 and Tab. 3.7).

It is worth remarking that the introduction of a random variable causes the whole T_{wifi} value (and therefore the value of the polling time) to be no more deterministic. In fact, if retransmissions are taken into account, T_p itself becomes a random variable.

4.4.3 Packets exchange with multiple retransmissions

The more realistic case that can be considered foresees that both the command frame and the state frame could be retransmitted an arbitrary number of times.

First of all, the computation of the wireless transmission time is done in case the command frame is retransmitted n times, for a total of $n + 1$ transmissions, while the state frame is always transmitted successfully at the first attempt. In this case, the total time

required for the transmission of frames on the wireless channel is

$$T_{wifi} = 2 \cdot T_{DIFS} + (n + 1) \cdot T_{com} + n \cdot T_{ack_TO} + T_{BO}(n) + T_{sta} \quad (4.9)$$

where, considering the evolution of the CW, the total backoff time in case of n retransmissions, $T_{BO}(n)$, can be computed as

$$T_{BO}(n) = T_{SLOT} \cdot \sum_{t=1}^n \mathbf{U} [0, 2^{t-1} (CW_{min} + 1) - 1] \quad (4.10)$$

Here it is implicitly assumed that the number of retransmission n is chosen so that, when the maximum CW value is reached, the transmission is considered as permanently failed and no more retransmissions are performed.

An even more general case is considered if also the state messages can be retransmitted. The number of retransmissions for the command frames is indicated again with n , while that of state frames is indicated with m . The total wireless transmission time hence becomes

$$T_{wifi} = 2 \cdot T_{DIFS} + (n + 1) \cdot T_{com} + (n + m) \cdot T_{ack_TO} + T_{BO}(n, m) + (m + 1) \cdot T_{sta} \quad (4.11)$$

The total backoff time here is given by

$$T_{BO}(n, m) = T_{SLOT} \cdot \left[\sum_{t=1}^n \mathbf{U} [0, 2^{t-1} (CW_{min} + 1) - 1] + \sum_{s=1}^m \mathbf{U} [0, 2^{s-1} (CW_{min} + 1) - 1] \right] \quad (4.12)$$

4.5 Statistical characterization of polling time

The detailed analysis of all polling time components carried out in this Section can be exploited to obtain a statistical characterization of the behavior of the random variable polling time T_p . In particular, the interest is on the maximum and minimum value that this variable can assume. For the sake of completeness, the average value (in the stochastic meaning) is also computed.

From the performed analysis, it is evident that the randomness of T_p lies entirely on the transmission time on the wireless medium. More specifically, the only source of randomness is the random backoff time, T_{BO} , whose distribution depends on the number of retransmissions of command and state frames.

To compute the minimum value assumed by T_p , the ideal case is considered, where all

frames are transmitted successfully at the first attempt. In this case, $T_{BO} = 0$ and

$$T_p = T_{uart} + T_{wifi} + T_{d_ECU} = T_{uart} + 2 \cdot T_{DIFS} + T_{com} + T_{sta} + T_{d_ECU} \quad (4.13)$$

Conversely, the maximum value of T_p is obtained when the maximum allowed number of retransmission attempts (n_{max}) is performed for both the command and the state packet and, each time the backoff time is randomly selected, the worst-case value is picked. In this case

$$\begin{aligned} T_{BO}^{max} &= T_{SLOT} \cdot \left[2 \sum_{t=1}^{n_{max}} (2^{t-1} (CW_{min} + 1) - 1) \right] \\ &= 2 \cdot T_{SLOT} \cdot [(CW_{min} + 1) \cdot (2^{n_{max}} - 1) - n_{max}] \end{aligned} \quad (4.14)$$

and

$$T_p = T_{uart} + 2 \cdot T_{DIFS} + (n_{max} + 1) \cdot T_{com} + 2 \cdot n_{max} \cdot T_{ack_TO} + T_{BO}^{max} + (n_{max} + 1) \cdot T_{sta} + T_{d_ECU} \quad (4.15)$$

Finally, the average value of T_{BO} for a given number of retransmissions of command and state packets is

$$\begin{aligned} T_{BO}^{avg}(n, m) &= T_{SLOT} \cdot E \left[\sum_{t=1}^n \mathbf{U} [0, 2^{t-1} (CW_{min} + 1) - 1] + \sum_{s=1}^m \mathbf{U} [0, 2^{s-1} (CW_{min} + 1) - 1] \right] \\ &= \frac{1}{2} T_{SLOT} \cdot \left[\sum_{t=1}^n [2^{t-1} (CW_{min} + 1) - 1] + \sum_{s=1}^m [2^{s-1} (CW_{min} + 1) - 1] \right] \end{aligned} \quad (4.16)$$

As a first result, Tab. 4.4 shows both the minimum and maximum values assumed by T_p in correspondence to the chosen value of data transmission rate on the wireless link.

The most important consequence that can be derived is that, with the considered wireless module, the value of the polling time lies in the range $[T_p^{min}, T_p^{max}]$, where

$$T_p^{min} = 94.05 \text{ ms} \quad (4.17)$$

$$T_p^{max} = 201.76 \text{ ms} \quad (4.18)$$

It is also interesting to analyze how the value of the polling time T_p increases with the number of consecutive retransmissions, for a given version of the standard and choice of data rate. As an example, Fig. 4.4 shows the behavior of polling time for the IEEE 802.11g version of the standard with data rate 54 Mb/s (the highest possible choice). In particular, Fig. 4.4a shows the minimum, maximum and average values assumed by T_p

Table 4.4 – Maximum and minimum value assumed by polling time for different data rates.

Version	Transmit rate [Mb/s]	Minimum T_p [ms]	Maximum T_p [ms]
IEEE 802.11b	1	97.52	201.76
	2	95.84	189.35
	5.5	94.77	181.47
	11	94.47	179.22
IEEE 802.11g	6	94.55	118.88
	9	94.36	117.56
	12	94.27	115.86
	18	94.17	115.19
	24	94.12	114.33
	36	94.08	114.02
	48	94.05	113.82
	54	94.05	113.79

when the command frame is transmitted n times, while the state frame is always transmitted successfully at first attempt. Fig. 4.4b, instead, shows the same values when both the command and the state frames are transmitted the same number of times (again indicated with n). The same results are also reported in Tab. 4.5.

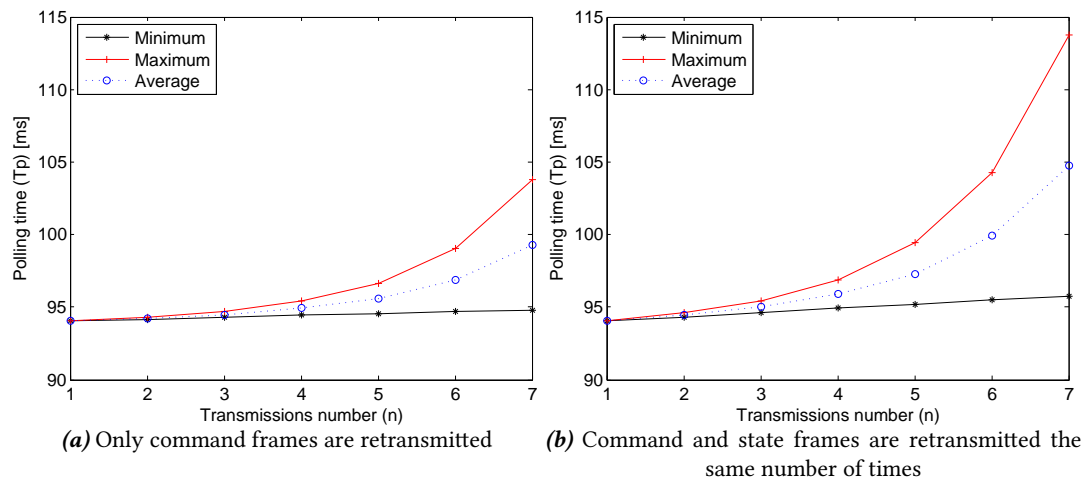
**Figure 4.4** – Evolution of polling time versus the number of transmissions for IEEE 802.11g at 54 Mb/s.

Fig. 4.5 presents the same results of Fig. 4.4 adopting the IEEE 802.11b version of the standard with data rate 1 Mb/s, thus using the lowest possible data rate. It is evident from the figure that the values assumed by polling time are much higher than those showed

Table 4.5 – Maximum, minimum and average values assumed by polling time for transmissions number with IEEE 802.11g at 54 Mb/s.

Transm. number	Only command retransmissions			Command and state retransmissions		
	Minimum	Maximum	Average	Minimum	Maximum	Average
1	97.52 ms	97.52 ms	97.52 ms	97.52 ms	97.52 ms	97.52 ms
2	98.55 ms	99.13 ms	98.84 ms	101.50 ms	102.70 ms	102.10 ms
3	99.58 ms	101.42 ms	100.50 ms	105.48 ms	109.20 ms	107.34 ms
4	100.62 ms	105.00 ms	102.81 ms	109.46 ms	118.26 ms	113.86 ms
5	101.65 ms	111.13 ms	106.39 ms	113.44 ms	132.44 ms	122.94 ms
6	102.69 ms	122.39 ms	112.54 ms	117.42 ms	156.86 ms	137.14 ms
7	103.72 ms	143.88 ms	123.80 ms	121.40 ms	201.76 ms	161.58 ms

in the previous figure, therefore it can be concluded that the choice of data rate has quite an impact. Clearly, using a higher data rate allows to speed up the transmission process but also worsens the robustness of communication, possibly leading to errors which may require more retransmissions, thus causing an increase in polling time.

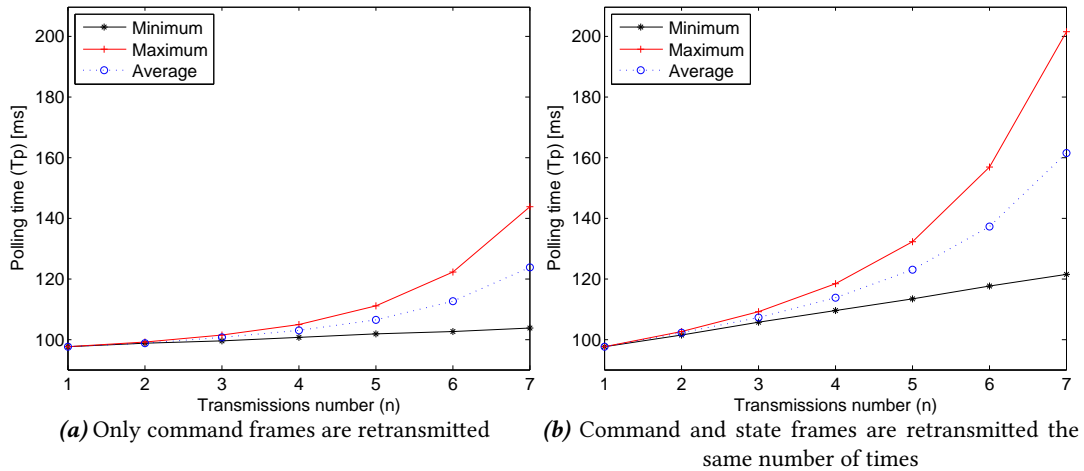


Figure 4.5 – Evolution of polling time versus the number of transmissions for IEEE 802.11b at 1 Mb/s.

As a final result, the relative weights of the polling time components, namely T_{wifi} , T_{uart} and T_{d_ECU} , on the total value of T_p have been considered for different scenarios:

- ✓ **Scenario A:** IEEE 802.11g at 54 Mb/s, no retransmissions.
- ✓ **Scenario B:** IEEE 802.11g at 24 Mb/s, 3 transmissions of both state and command packets, average case.

- ✓ **Scenario C:** IEEE 802.11b at 11 Mb/s, 4 transmissions of both state and command packets, average case.
- ✓ **Scenario D:** IEEE 802.11b at 1 Mb/s, 7 transmissions of both state and command packets, worst case.

The results are reported in Fig. 4.6. It can be seen that, if the highest data rate is used and transmissions are successful at first attempt (an high SNR is required), the time needed for wireless communication is negligible compared to the other components. Conversely, if the data rate is very low and/or the channel conditions are very poor, the weight of this component can rise up over the 50% of the entire polling time.

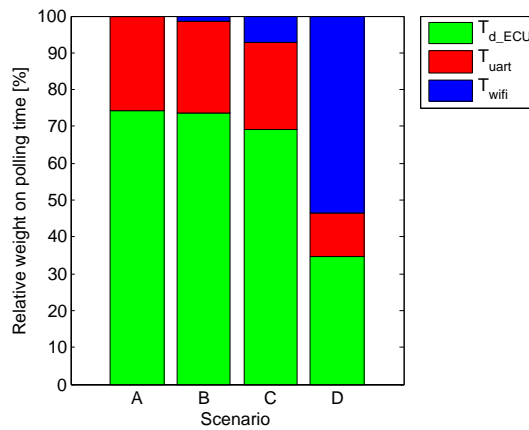


Figure 4.6 – Relative weights of polling time components for different scenarios.

4.6 Final considerations

The theoretical analysis carried out brought to light that, even if the channel conditions are very bad and the lowest possible data rate is adopted, inequality 4.1 is always fulfilled, since $T_C = 250 \text{ ms}$ and $T_p^{max} \approx 200 \text{ ms}$.

Therefore, the value of cycle period can be safely lowered down to about 200 ms, obtaining a more timely monitor and control of the agricultural system. Actually, since the value T_p^{max} is very unlikely to be reached, the value of T_C could be decreased even further. Alternatively, it could be chosen to keep this value fixed to 250 ms and use the amount of time remained after the conclusion of polling operations (which spans roughly from 50 ms to 150 ms) to perform operations to increase the robustness of the system, such as further retransmissions.

Moreover, the interpacket time for state frames, defined in Section 4.1.2, should always fall in the approximate range [150 *ms*, 350 *ms*] and never exceed these values. However, the experimental sessions performed on the prototype system have highlighted that the computed theoretical bounds are often violated, leading to a very strong inconsistency between theory and practice. Chapter 6 will investigate the causes of this discrepancy and provide possible solutions.

Experimental measurements

The prototype system presented in Chapter 2 has been subject to a series of experimental campaigns, in order to assess its performance figures and compare them with the results of the theoretical analysis carried out in Chapter 4. In particular, the interpacket time, whose expected behavior has been analyzed in detail, has been tracked in different scenarios.

Several measurement sessions have been performed, in different environments and conditions, which required specific instrumentation and setups. More specifically, an experimental session has been carried out in a large open field with the ECU aboard an agricultural machine, which represents the ultimate field of application for the described system. Then, to obtain more detailed measurements and a more controlled environment, two further measurement campaigns have been performed in a research laboratory.

The outcomes of these campaigns have highlighted several issues in the communication, which make the system behavior unacceptable for the desired scope. In particular, the theoretical bounds computed with the analysis in Chapter 4 are often violated in practice.

THREE different experimental campaigns have been performed to investigate the behavior of the system and, in particular, of the communication protocol between the ECU and the hand-held device.

5.1 On-field measurements

This first measurement campaign has been carried out in an environment which represents a real application scenario for the considered system, namely a large open field. This allowed to test the behavior of the system when facing the real operating conditions that

it may encounter during its actual use.

5.1.1 Setup description

The measurements have been performed in a large open field where neither obstacles nor buildings were presented. The ECU was installed on an agricultural machine, mechanically attached to a tractor, involved in seeding operation. The hand-held device employed was an Apple™ iPad with a remote control and monitoring application installed. The iPad was held by an operator aboard the tractor, so the distance between the two communicating nodes was about 5–7 meters, thus limiting fading issues. However, a continuous Line of Sight (LOS) communication path was not always available, due to the movement of involved machineries and the presence of metallic structures that increased multi-path effects. With regard to interference from other sources, it has been assessed that no other active IEEE 802.11 devices (or other devices transmitting in the ISM band) were present in the field during the experimental session.

TCP was used as transport layer protocol. The application software on the iPad was designed to collect meaningful debug data, such as the time elapsed between the transmission of packets at transport layer. Unfortunately, more accurate instrumentation could not be used due to the harsh operational environment. More specifically, both the wireless module on the ECU and the iPad device did not grant for any interaction with the MAC layer, so that a detailed analysis of packet flow and a precise establishment of channel conditions could not take place.

Several consecutive sessions have been performed, each one lasting some hours, which is the expected working time for the system. The results have shown a common trend, so the outcomes presented in this Section all refer to the first five hours of operations in a specific session.

5.1.2 Outcomes

The results of the aforementioned experimental session are presented here in terms of interpacket time, measured at transport layer. For the sake of clarity, this metric indicates the time elapsed between each transmission attempt at this layer and the following one, for both devices. The results for command and state messages are reported in Fig. 5.1.

The behavior of interpacket times for command packets, reported in Fig. 5.1a, highlights the high number of TCP retransmissions, that can be observed in the figure as downward peaks. In fact, TCP is a reliable protocol which retransmits the same packet until its successful delivery is confirmed through an acknowledgement. As a consequence,

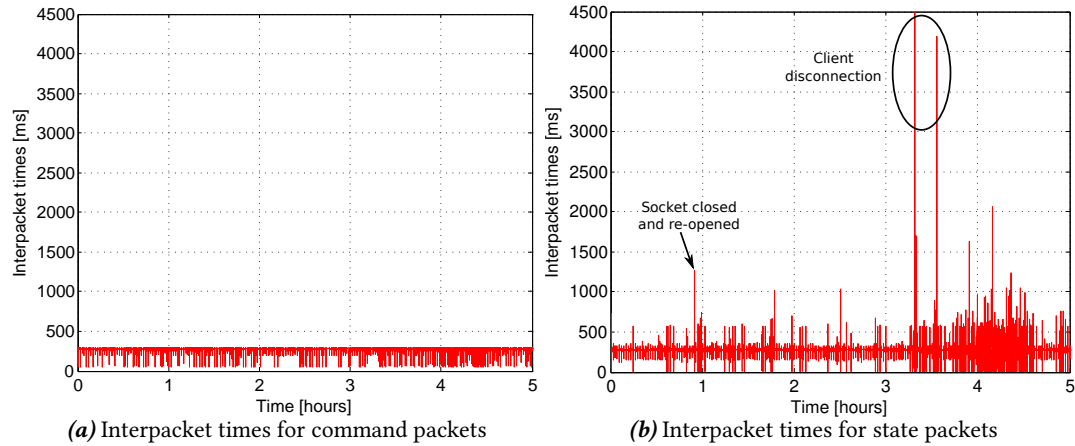


Figure 5.1 – Interpacket times measured in the on-field session.

there is a very short distance (related to the timeout set by TCP) between two consecutive transmissions of the same packet, which causes a downward peak in the interpacket time measured at transport layer. It is worth noting that a TCP retransmission is required only if the underlying data-link layer has declared the transmission as failed, so after a certain number of MAC layer retransmissions have been performed, according to the scheme presented in Section 3.2.1. As a final remark, it can be seen that the interpacket time never exceeds the cycle period $T_C=250$ ms. Indeed, the communication protocol is configured so that every 250 ms a new command message is sent by the hand-held device, regardless of what happened at lower layers (see Section 2.2). In this way, there can never be a delay greater than 250 ms between two packet transmissions at transport layer.

Looking at the interpacket times for state messages, presented in Fig. 5.1b, an even worse behavior is observed. Several packets experienced delays well above the maximum theoretical bounds of 350 ms, as pointed out by numerous upward peaks in the interpacket time. This is again an indication that the communication between the two devices is disturbed, as already witnessed by the high number of TCP retransmissions of command packets, although a more in-depth analysis of the packets exchange is required. As a consequence of these very long delays, two very dangerous events were observed during the test. The first one is represented by a socket closing that took place just after one hour of operation, probably triggered by the 1 s timeout showed in Fig. 2.5. The second event is even more serious, since in this case the client iPad device disconnected from the WLAN created by the ECU, with the consequent abnormal increase of the interpacket time. As a final consideration, the figure highlights that TCP retransmissions are performed also at the wireless module of the ECU, as confirmed by the downward peaks in interpacket

times.

Summarizing, the transmission of both command and state messages highlights possible communication problems occurring at lower layers, that can not be detected and explained with the limited debug configuration available during this campaign. As a consequence, additional tests have been performed in a more controlled environment, where it has been possible to track the packet flow down to the data-link layer, to better establish the issues that compromise the communication.

5.2 Laboratory measurements

The on-field campaign has pointed out the need for more accurate measurements, in order to better establish what is going on at **MAC** layer. To achieve this result, a second campaign was executed in a research laboratory, that ensured a better control of the whole measurement procedure.

5.2.1 Setup description

The first important difference with the previous experimental sessions was the replacement of the iPad with a desktop PC. This device allowed for more configuration possibilities and rapid exchange of system parameters with respect to what was possible with the iPad. The Operating System (OS) running on the PC was Windows™ 7 and a specific application was developed to emulate the behavior of the iPad and to collect meaningful debug data. Moreover, the **ECU** firmware was also modified, to emulate the operation of the **CAN** network to which it was previously connected. The **ECU** and the PC were positioned quite close (0.5–1 meters) and a direct propagation path was this time available, although multi-path effects could not be entirely avoided. With respect to the on-field scenario, a higher number of interference sources was present in this case, because of many co-located wireless **LANs**. An accurate choice of the **WLAN** channel was made to limit the interference effects, even though a total isolation of the **WLAN** created by the **ECU** was not possible.

To improve the precision of measurements, a separate computer was added to the setup, running a packet sniffing software [2]. Thanks to this software, the second computer was able to capture IEEE 802.11 packets from its network interface card, providing an accurate dissection of these packets down to the **MAC** layer. This computer was set in monitor mode, in order to capture every packet that was crossing the radio link, so a detailed picture of the communication on the wireless medium could be drawn. In ad-

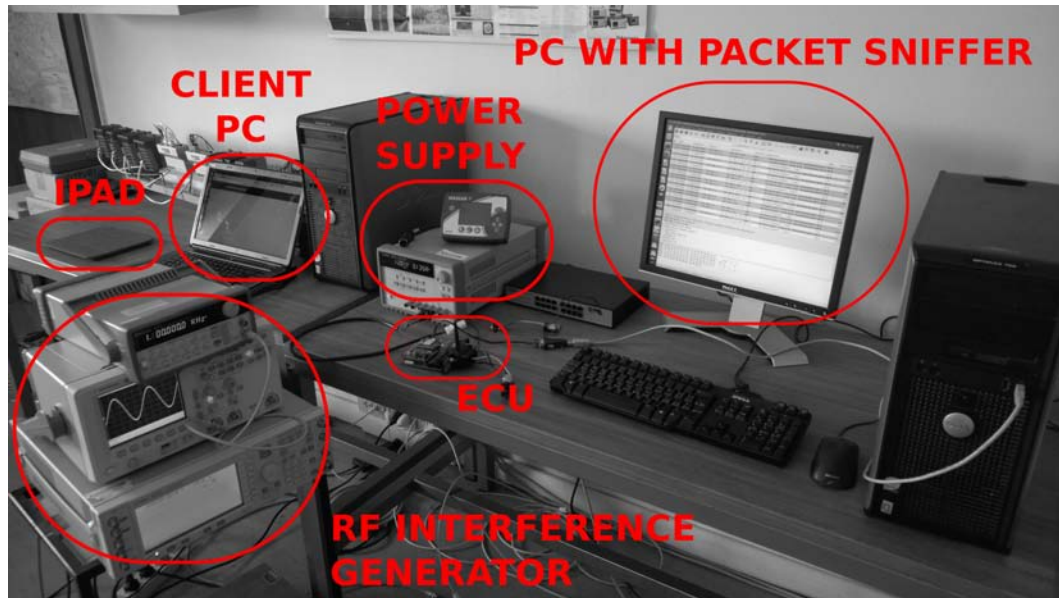


Figure 5.2 – Experimental setup employed in the indoor laboratory.

dition to this new measurement instrument, the application on the PC was still able to record high-layer packets exchange, as the iPad did on the previous campaign. Finally, a spectrum analyzer was occasionally used to monitor the exchange of packets on the physical medium and an appropriate setup to generate Radio Frequency (RF) interference was sometimes adopted to test the system behavior in presence of specifically shaped sources of interference. Fig. 5.2 provides a picture of the complete setup adopted for the tests in the indoor laboratory.

Several experimental sessions have been performed in this indoor laboratory. Since they evidenced a more stable behavior over time with respect to the outdoor measurements, the results reported in this Section will refer only to a 30 minutes window. The application software on the PC allowed to rapidly change the communication parameters, in particular the choice of the transport layer protocol. As a result, different tests have been performed with TCP and UDP, comparing the performance obtained with the two protocols.

5.2.2 Outcomes with TCP

Since the new setup allows to explore packet flow at data-link layer, the results are presented in terms of interframe time, *i.e.* the time elapsed between two consecutive transmissions of MAC layer data frames, for both devices. The interframe times for command

and state packets using TCP as transport layer protocol are reported in Fig. 5.3.

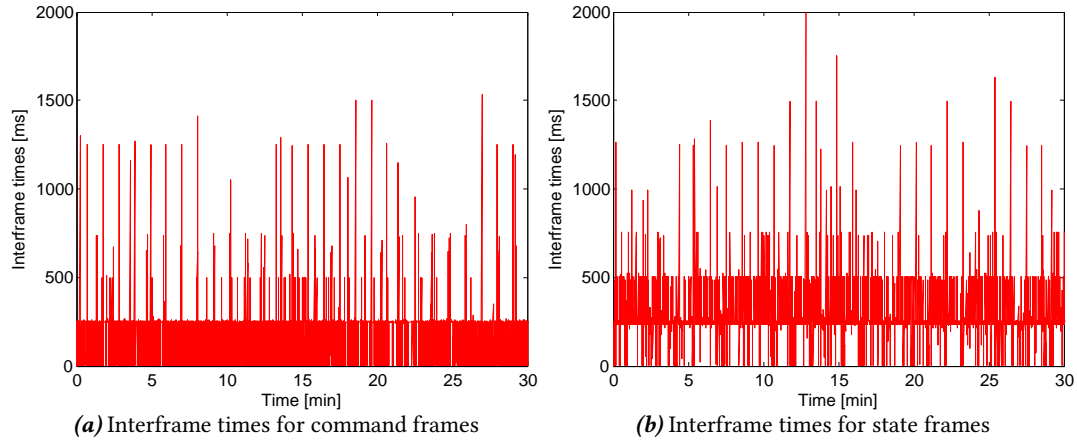


Figure 5.3 – Interframe times measured in the laboratory session with TCP.

Looking at the interframe times for command frames (Fig. 5.3a), it can be seen that there are several downward and upward peaks with respect to the expected behavior, which is a constant value of $T_C=250$ ms. The downward peaks, that were present also during the on-field session (even if at the application layer), are the consequence of MAC layer retransmissions. With respect to Fig. 5.1a, these peaks are more numerous, since the occurrence of this kind of retransmissions is much more frequent than that of TCP retransmissions. In fact, one TCP retransmission is required only after a series of failures in transmission attempts at MAC layer. All these retransmissions highlights that the communication between the two devices is deeply disturbed. Although interference and multi-path might have an impact, the environmental conditions are not sufficient to explain such a high number of transmission failures, so there must be other reasons, that will be tackled in the next Chapter. The upward peaks represent a novelty, since they were not present in Fig. 5.1a. Moreover, these peaks show a periodic behavior, in the sense that they are recorded every 60/70 seconds with similar intensity. A detailed analysis of the traffic on the radio link around these peaks revealed that they depended on the intrinsic behavior of the network interface card of the PC used in place of the iPad for the measurement session. The next Chapter will investigate more deeply into this effect.

The interframe times for state frames, shown in Fig. 5.3b, present an even more non-ideal behavior, because of the ECU reaction to the irregular flow of command packets coming from the PC. Moreover, the synchronization issues present in the ECU, discussed in Section 4.3, could cause a degradation of performance.

To stress the occurrence of MAC layer retransmissions and to highlight their high num-

ber, Fig. 5.4 reports the histogram of the command and state frames that required more than one transmission attempt.

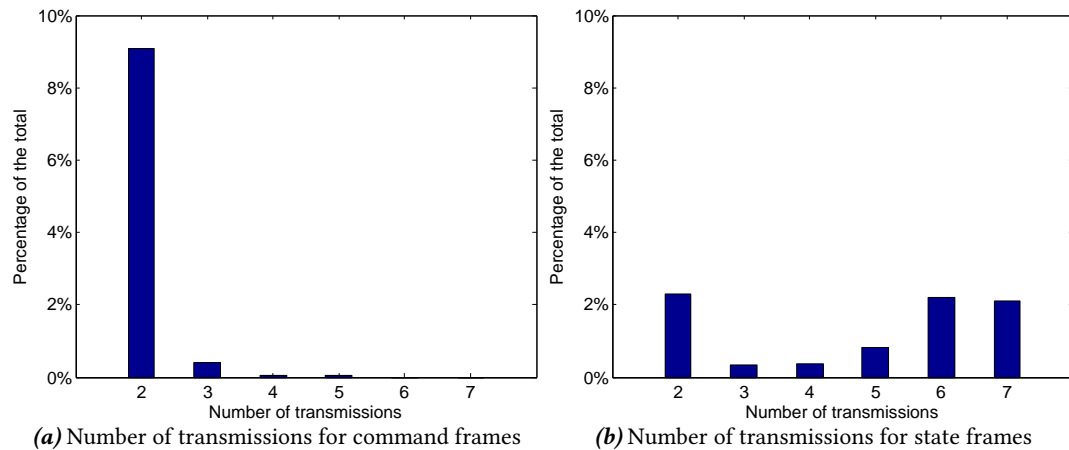


Figure 5.4 – Histogram of frames that required more than one transmission attempt at MAC layer.

The figure shows the number of transmission attempts required by each frame as percentage of the total. It can be seen that the occurrence of retransmissions is higher for command frames (almost 10%). However, for these frames a single retransmission is almost always sufficient, and the same frame never gets to be retransmitted more than 4 consecutive times, meaning that there is no frame loss. Conversely, state frames are more likely to require a high number of retransmissions, with a non-negligible percentage (about 2%) that need 7 transmissions and could be declared as definitely lost.

5.2.3 Outcomes with UDP

The choice of working with a PC instead of an iPad allowed to rapidly test new configuration for the system. Specifically, the most significant change made was the selection of UDP in place of TCP as transport layer protocol. UDP is expected to provide a more stable behavior, since this protocol does not add a secondary retransmission mechanism in addition to that at data-link layer. The drawback is that, if a packet is declared as lost after a certain number of MAC layer retransmission, it is never retransmitted again, and the system waits until the next cycle begins before sending a new message. As a consequence, there might be a certain percentage of lost messages, while TCP ensures that all the scheduled packets are sooner or later delivered.

Fig. 5.5 shows the evolution of interframe times for command and state packets obtained using UDP as transport layer protocol.

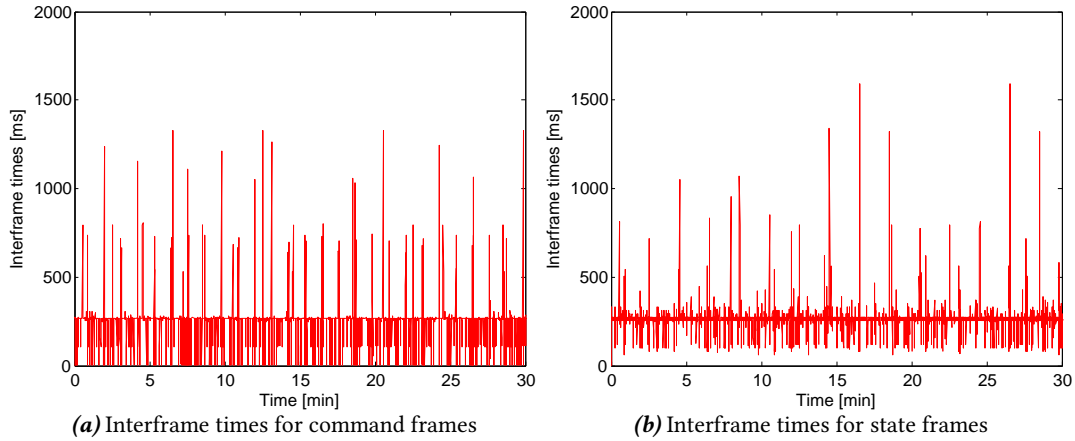


Figure 5.5 – Interframe times measured in the laboratory session with UDP.

A direct comparison with Fig. 5.3 highlights that the system behavior is actually more stable, and the interframe times are closer to the expected values. More specifically, the number of downward peaks is much lower for both command and state packets, since there is no additive retransmission mechanism other than that at data-link layer. However, the periodic upward peaks are still present, with approximately the same periodicity. This result evidences that this issue is not related to the chosen transport layer protocol.

To further establish the improvements derived from the adoption of UDP and to have an immediate assessment of how much the system behavior differs from the expected one, a statistical analysis of the interframe times is carried out. More specifically, the Empirical Cumulative Density Function (ECDF) of this quantity for state frames with both transport layer protocols is reported in Fig. 5.6.

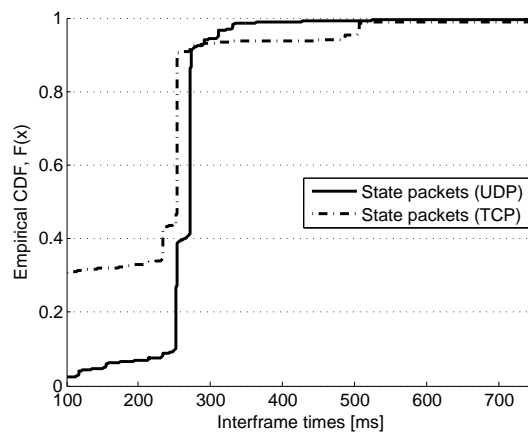


Figure 5.6 – ECDF of interframe times for state frames with both TCP and UDP.

As a general consideration, it can be noticed that the UDP protocol provides a more stable behavior with respect to TCP, in the sense that the ECDF plot is closer to the expected one, which should be a constant vertical line at 250 ms. Moreover, with TCP there is a relevant percentage (about 30 %) of state frames that are transmitted with a separation less than 100 ms, while with UDP this percentage is much lower (less than 5%). The reason lies on the aforementioned presence of an additional retransmission mechanism.

5.2.4 Tests with the iPad

To allow a direct comparison with the experimental session carried out in Section 5.1 and to discuss the behavior of the real system, some measurement campaigns have also been carried out in the laboratory using the iPad instead of the PC. The second computer with the packet sniffing software was still present, allowing a detailed inspection of MAC layer frames, and the environmental conditions were the same of the other tests carried out in the laboratory. TCP was used as a transport layer protocol, since the application on the iPad does not permit to change this parameter.

The outcomes, in terms of interframe times for both command and state frames, are reported in Fig. 5.7.

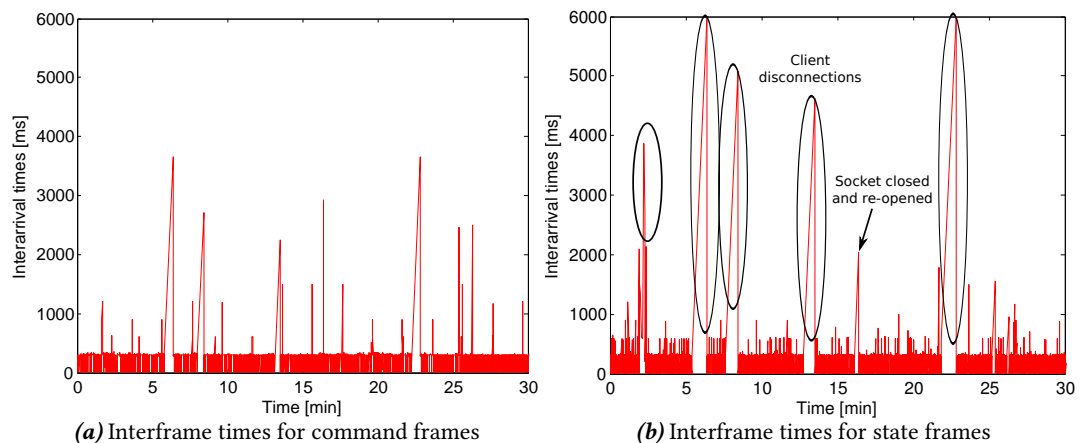


Figure 5.7 – Interframe times measured in the laboratory session with the iPad as client device.

The tests evidenced a very unstable behavior, with respect to both the on-field measurements with the same device and the laboratory measurements with a PC instead of the iPad. More specifically, five different client disassociation events (and a socket closure) were registered during a thirty minutes test, while the same device disconnected just once in five hours in the on-field test. This result proves that the effect of interferences, which are present in the laboratory and almost absent in the field, has a really bad

impact on the application, as already showed by other tests in interfering environments not reported in this work. However, it has to be pointed out that the ultimate application scenario for the application is the on-field one, so extensive actions to limit the effect of interferences are not considered in this thesis.

By comparing Fig. 5.7 and Fig. 5.3, which represent the outcomes of two sessions in which the only difference was the use of an iPad instead of a PC, it may be concluded that the iPad device is much more sensible to the presence of other WLANs in the environment. In fact, client disconnections are often a consequence of the periodical detection by the iPad of a new WLAN, which causes the disconnection from the WLAN created by the ECU and the stop of the remote control application.

5.3 Measurements on the wireless interface

A supplementary set of measurements has been carried out in the laboratory to provide further insights on the occurrence of the periodic upward peaks highlighted in Fig. 5.3 and Fig. 5.5 and also to assess the performance the wireless module mounted on the ECU is able to provide. In these measurements, the ECU has been excluded from the setup and only the communication between the PC and the wireless module has been analyzed. This was possible thanks to an evaluation board, which allowed to program the module through a serial interface and to test its performance, represented in Fig. 5.8.

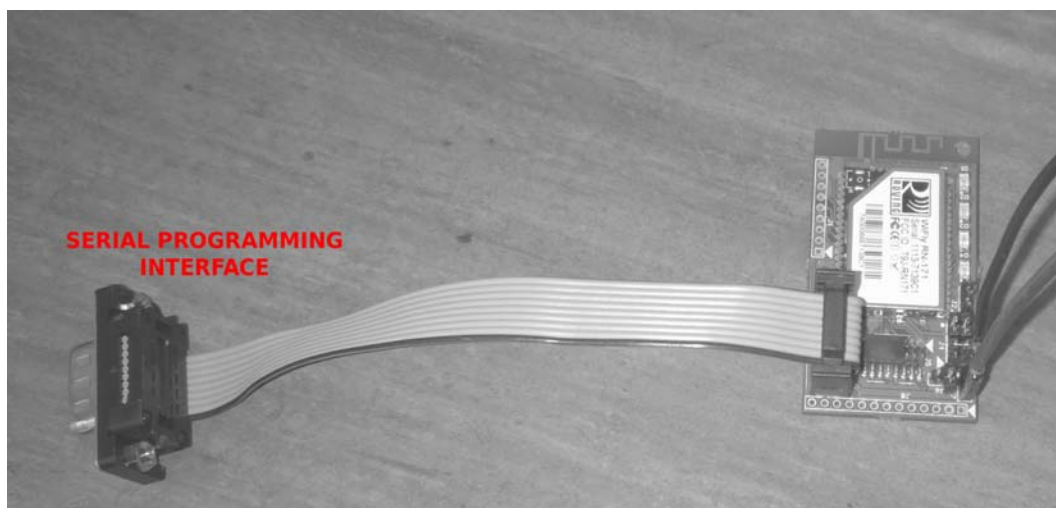


Figure 5.8 – Picture of the evaluation board used to test the wireless module as a stand-alone component.

Clearly, the operations available with the stand-alone wireless module are a restricted

subset of those possible with the use of the ECU. More specifically, if TCP is used as transport layer protocol, the module allows only to accept the opening of a socket from the client and to acknowledge the received TCP segments. An even more limited set of operations is available with UDP. As a consequence, the results presented in this Section will not be directly comparable with those of previous experimental campaigns and also the considered metric will change.

Two different experimental sessions have been performed, one using TCP and the other using UDP. The environmental conditions (interference, distance, LOS) are the same considered in the previous laboratory campaigns.

5.3.1 Tests with TCP

An ad-hoc TCP polling application has been developed, in which the client PC sends, with a fixed periodicity, a packet of length 18 Bytes (the same of command messages) to the module, which replies with a TCP acknowledgement. The periodicity has been set to 50 ms. Even if this value is lower than that used in the previous application, there is still enough time to perform all possible MAC layer retransmissions.

The metric considered to evaluate communication performance is the *service time*, defined as the time elapsed between the transmission of the packet from the client PC and the successful reception of the corresponding acknowledgement. The expected value for this quantity can be easily computed as:

$$S = T_{DIFS} + T_{data} + T_{SIFS} + T_{ack,WIFI} + T_{DIFS} + T_{ack,TCP} \quad (5.1)$$

where the ideal case has been taken into account, in which there are no packet losses and therefore retransmission are not required. IEEE 802.11b has been used and the data rate adopted for the *data* packet and the TCP acknowledgement was 11 Mb/s, while the data rate of the WLAN acknowledgement was 1 Mb/s. With these settings, the expected value for service time was approximately 1 ms.

Fig. 5.9 shows the evolution of service time as registered in the experimental measurements.

The measurements highlighted a significant variability of service time with respect to the expected behavior. Surprisingly, in some cases the service time even exceeds the polling period of 50 ms. This result confirms that the communication between the two devices is deeply disturbed and several data-link layer retransmissions are required.

Fig. 5.10 reports the empirical Probability Density Function (PDF) of the service time,

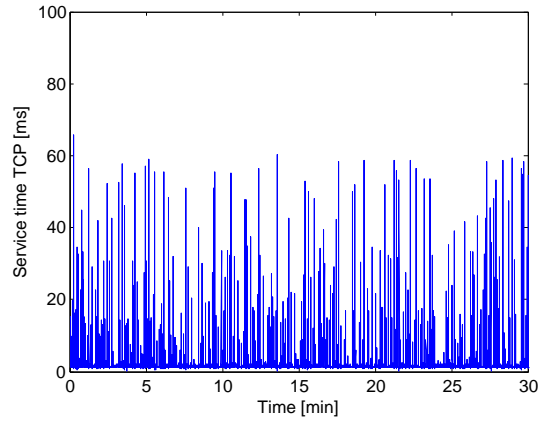


Figure 5.9 – TCP service time using only the wireless module.

showing that in most cases this quantity is concentrated in the range 1–1.5 ms, which corresponds to the expected ideal behavior. However, there is a non-negligible probability that it assumes much higher values.

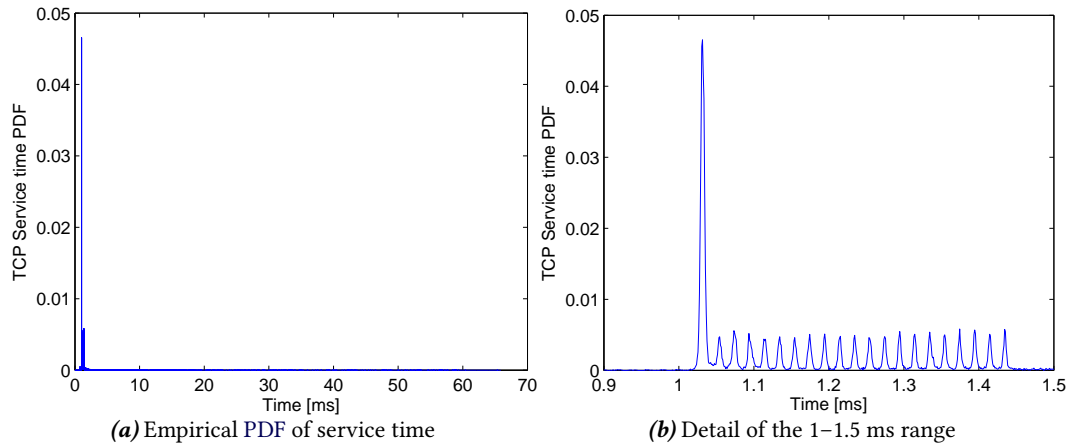


Figure 5.10 – Statistical analysis of service time using only the wireless module.

5.3.2 Tests with UDP

With the available functionalities, the only possible test using UDP as transport layer protocol was the periodic polling (always with a period of 50 ms) of the wireless module from the client PC with packets of length 18 Bytes. Since UDP is an unacknowledged protocol, the module did not reply in any way to the packets sent by the PC, behaving as a sink.

The considered metric here is the *interdeparture time*, which is the time elapsed between two consecutive packets sent by the PC measured at MAC layer. The outcomes are

reported in Fig. 5.11

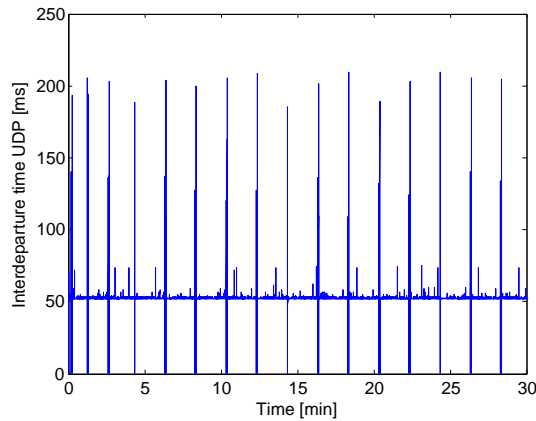


Figure 5.11 – Interdeparture times with UDP using only the wireless module.

It can be seen that the behavior is quite different from the expected one, which is a constant 50 ms value. Specifically, the same periodic upward and downward peaks that were present in Fig. 5.3 and Fig. 5.5 can be noticed in Fig. 5.11. This result confirms that the motivation for these peaks has to be found on the intrinsic behavior of the client device, since in this experiment the wireless module does not contribute in any way to the network traffic.

5.4 Conclusions

The different experimental campaigns described in this Chapter pointed out that the system behavior is very different from the ideal one, derived from the theoretical analysis.

In particular, the communication seems to be extremely disturbed even if the environmental conditions are good, leading to a high number of MAC layer retransmissions, that could cause the polling time to increase dramatically. Moreover, a strange behavior was recorded, namely the periodic presence of delays in the communication, that has proven to be dependent only on the client device.

Finally, it has been confirmed that the choice of TCP as transport layer protocol contributes to worsen the system performance, since it adds a second retransmission mechanism. The results obtained with UDP were slightly better, even if far from being acceptable.

The next Chapter will use the results of the described experimental sessions to identify the main causes of performance degradation and to provide solutions. The system will be tested again after the implementation of these solutions, comparing the performance with those reported in this Chapter.

Chapter 6

Analysis and solutions of the most relevant issues

The experimental sessions described in Chapter 5 evidenced several problems in the communication between the client device and the ECU, that cause the remote monitoring and control application to be unreliable and ineffective. In this Chapter the causes of these issues are analyzed and solutions are proposed, in order to reach a satisfactory behavior.

In particular, the first part of this Chapter analyzes two problems that induce most of the low-level retransmissions highlighted during the measurement campaigns. These issues are strictly related to the use of COTS devices, such as the tablet PC and desktop PC adopted in the experimental sessions, that are not natively conceived for real-time applications. A proper tuning of these devices allows an immediate performance improvement, without any change to the ECU firmware.

In order to reach better overall performance figures, however, a new design of the protocol is suggested, based on the theoretical analysis carried out in Chapter 4. With the proposed changes, a much more stable system behavior is achieved, even in presence of intrinsic problems due to the use of COTS devices. Finally, if the new protocol design is adopted and the client devices are properly tuned, excellent system performance figures are obtained.

MOST of the communication issues raised up by the outcomes of the experimental sessions are related to the use of common-purpose devices as the remote client in the control and monitoring application.

6.1 Issues related to the adoption of COTS devices

COTS devices are conceived to achieve an extreme connectivity with surrounding networks, introducing a series of functionalities for network discovery. These features may be detrimental when a soft real-time behavior is required to the system, which is the case in industrial applications. In this Section, it will be explained how some built-in features of the client device can compromise the timeliness of the communication with the ECU and how disabling these functionalities can improve the system performance.

6.1.1 Self-interference

The experimental sessions highlighted that the communication between the wireless module on the ECU and the client device is strongly disturbed, leading to a high number of MAC layer retransmissions. A detailed analysis of traffic has pointed out that the majority of disturbances were not related to interference from other sources, but to the intrinsic behavior of the client device. In particular, several background network services, not specifically pertaining to the considered application, are active on the device and try to access the channel, possibly obstructing the traffic related to the application. This issue is known as *self-interference*.

Fig. 6.1 represents a screenshot of the packet flow as captured by the packet sniffing software during an experimental session carried out in the laboratory, using the PC as a client device and TCP at transport layer.

1429	0.00021000	192.168.2.1	192.168.2.11	TCP	90 cisco-sccp > 49301 [ACK] Seq=23583 Ack=1658 Win=2048 Len=0
1430	0.000164000		RovingMe 72:23:47 (RA)	802.11	28 Acknowledgement, Flags=.....
1432	0.105171000	192.168.2.1	192.168.2.11	TCP	347 cisco-sccp > 49301 [PSH, ACK] Seq=23583 Ack=1658 Win=1520 Len=257
1433	0.000150000		RovingMe 72:23:47 (RA)	802.11	28 Acknowledgement, Flags=.....
1434	0.022768000	fe80::b9cd:6617:1f	ff02::1:3	LLMNR	119 Standard query 0xe8fe A 560
1435	0.000309000		IntelCor bc:31:54 (RA)	802.11	28 Acknowledgement, Flags=.....
1436	0.000607000	192.168.2.11	224.0.0.252	LLMNR	99 Standard query 0xe8fe A 560
1438	0.098722000	fe80::b9cd:6617:1f	ff02::1:3	LLMNR	119 Standard query 0xe8fe A 560
1439	0.000311000		IntelCor bc:31:54 (RA)	802.11	28 Acknowledgement, Flags=.....
1440	0.000807000	192.168.2.11	224.0.0.252	LLMNR	99 Standard query 0xe8fe A 560
1441	0.000314000		IntelCor bc:31:54 (RA)	802.11	28 Acknowledgement, Flags=.....
1442	0.019486000	192.168.2.11	192.168.2.1	TCP	108 49301 > cisco-sccp [PSH, ACK] Seq=1658 Ack=23840 Win=17066 Len=18
1443	0.000315000		IntelCor bc:31:54 (RA)	802.11	28 Acknowledgement, Flags=.....
1444	0.000803000	192.168.2.1	192.168.2.11	TCP	90 cisco-sccp > 49301 [ACK] Seq=23840 Ack=1676 Win=2048 Len=0
1445	0.000162000		RovingMe 72:23:47 (RA)	802.11	28 Acknowledgement, Flags=.....
1447	0.108802000	192.168.2.1	192.168.2.11	TCP	347 cisco-sccp > 49301 [PSH, ACK] Seq=23840 Ack=1676 Win=1520 Len=257
1448	0.000156000		RovingMe 72:23:47 (RA)	802.11	28 Acknowledgement, Flags=.....
1450	0.070639000	192.168.2.11	192.168.2.255	NBNS	128 Name query NB 560=00>
1451	0.000311000		IntelCor bc:31:54 (RA)	802.11	28 Acknowledgement, Flags=.....
1452	0.000639000	192.168.2.11	192.168.2.255	NBNS	128 Name query NB 560=00>
1454	0.068931000	192.168.2.11	192.168.2.1	TCP	108 49301 > cisco-sccp [PSH, ACK] Seq=1676 Ack=24097 Win=16749 Len=18
1455	0.000314000		IntelCor bc:31:54 (RA)	802.11	28 Acknowledgement, Flags=.....
1456	0.000801000	192.168.2.1	192.168.2.11	TCP	90 cisco-sccp > 49301 [ACK] Seq=24097 Ack=1694 Win=2048 Len=0
1457	0.000161000		RovingMe 72:23:47 (RA)	802.11	28 Acknowledgement, Flags=.....
1459	0.110810000	192.168.2.1	192.168.2.11	TCP	347 cisco-sccp > 49301 [PSH, ACK] Seq=24097 Ack=1694 Win=1520 Len=257
1461	0.067282000	192.168.2.11	192.168.2.1	DNS	121 Standard query 0x6fa9 A teredo.ipv6.microsoft.com
1462	0.000584000	192.168.2.11	192.168.2.1	DNS	121 Standard query 0x6fa9 A teredo.ipv6.microsoft.com
1463	0.000310000		IntelCor bc:31:54 (RA)	802.11	28 Acknowledgement, Flags=.....
1464	0.000251000	192.168.2.1	192.168.2.11	TCP	108 Destination unreachable (network unreachable)
1465	0.000163000		RovingMe 72:23:47 (RA)	802.11	28 Acknowledgement, Flags=.....
1466	0.069360000	192.168.2.11	192.168.2.1	TCP	108 49301 > cisco-sccp [PSH, ACK] Seq=1694 Ack=24354 Win=16492 Len=18

Figure 6.1 – Screenshot of a packet capture which highlights the presence of services not related to the application.

The TCP packets pertaining to the application (command, state and the relative acknowledgements) are highlighted in green. The packets highlighted in blue and yellow are relevant to name resolution services, such as Domain Name System (DNS), Link-Local Multicast Name Resolution (LLMNR) and NetBIOS Name Service (NBNS). Finally, the packet highlighted in black pertains to the Internet Control Message Protocol (ICMP) messaging service. While only the first category of packets is related to the application and therefore accepted by the ECU, it can be seen that also some of the packets in the other categories are addressed to the network created by the ECU, which has network address 192.168.2.0. Clearly, these packets are rejected by the ECU but their presence consumes resources that are destined to the control application.

Indeed, once the client device associates to the WLAN created by the wireless module on the ECU, this latter is elected as default gateway for all communication services. As a result, many of the background services on the PC will try to send requests to the ECU, even though they are not being answered, deferring the transmission of the packets related to the application. Moreover, this additional traffic is totally unsynchronized with the polling process and collisions may occur, requiring frequent retransmissions.

Similar self-interference issues have been encountered also using the iPad as client device. Since the background network services are definitely not needed for the operation of the remote control and monitoring application, they should be disabled, to ensure that the application always gets all the available resources.

6.1.2 Power management

Another issue highlighted by the experimental sessions was the presence of periodic delays in the communication between the client device and the ECU. An accurate analysis of the packet flow allowed to understand that the responsible of this behavior is the power management mechanism implemented in the client device. It has been observed that, with a certain periodicity (60/70 seconds), this device sends a IEEE 802.11 *Null Data* frame to the ECU, as reported in Fig. 6.2, where it is highlighted in pink.

6203	155.507892000	192.168.2.11	192.168.2.1	TCP	108	49301 > cisco-sccp [PSH, ACK] Seq=10550 Ack=150541 Win=16749 Len=18
6204	155.507404000		IntelCor bc:31:54 (RA)	802.11	28	Acknowledgement, Flags=.....
6205	155.508206000	192.168.2.1	192.168.2.11	TCP	90	cisco-sccp > 49301 [ACK] Seq=150541 Ack=10568 Win=2048 Len=0
6206	155.508372000		RovingNe 72:23:47 (RA)	802.11	28	Acknowledgement, Flags=.....
6208	155.607592000	192.168.2.1	192.168.2.11	TCP	347	cisco-sccp > 49301 [PSH, ACK] Seq=150541 Ack=10568 Win=1520 Len=257
6209	155.607746000		RovingNe 72:23:47 (RA)	802.11	28	Acknowledgement, Flags=.....
6210	155.757148000	192.168.2.11	192.168.2.1	TCP	108	49301 > cisco-sccp [PSH, ACK] Seq=10568 Ack=150798 Win=16492 Len=18
6211	155.757438000		IntelCor bc:31:54 (RA)	802.11	28	Acknowledgement, Flags=.....
6212	155.758260000	192.168.2.1	192.168.2.11	TCP	90	cisco-sccp > 49301 [ACK] Seq=150798 Ack=10586 Win=2048 Len=0
6213	155.758422000		RovingNe 72:23:47 (RA)	802.11	28	Acknowledgement, Flags=.....
6216	155.860357000	192.168.2.1	192.168.2.11	TCP	347	cisco-sccp > 49301 [PSH, ACK] Seq=150798 Ack=10586 Win=1520 Len=257
6217	155.860514000		RovingNe 72:23:47 (RA)	802.11	28	Acknowledgement, Flags=.....
6218	155.955229000	IntelCor bc:31:54	RovingNe 72:23:47	802.11	42	Null function (No data), SN=1834, FN=0, Flags=...P...T
6219	155.955420000		IntelCor bc:31:54 (RA)	802.11	28	Acknowledgement, Flags=.....

Figure 6.2 – Screenshot of a packet capture which highlights the presence of Null Data frames.

The Null Data frame is declared in the standard as a data frame with empty frame body [13], but its usage is not explicitly specified. However, Network Interface Card (Nic) vendors make great use of this frame type in real implementations for control applications [7]. An example of application is the implementation of a power management mechanism: a STA sends a Null Data frame with Power Management bit set to 1 to advertise that it is entering PS mode and a Null Data frame with the same bit to 0 to inform that it has exited this mode. In the meanwhile, the AP buffers all the frames destined to the STA. Analogously, the frames scheduled for transmission by the STA itself are buffered too. This is exactly what has been observed in the communication between the client device and the ECU, as witnessed by the dissection of the captured Null Data frame, reported in Fig. 6.3.

```

6218 155.955220800 IntelCor_bc:31:54 RovingNe_72:23:47 802.11 42 Null function (No data), SN=1834, FReq, Flags=...P...T
6219 155.955420900 IntelCor_bc:31:54 (RA) 802.11 28 Acknowledgement, Flags=.....
6220 155.955420900 RovingNe_72:23:47 (RA) 802.11 28 Acknowledgement, Flags=.....
▶ Frame 6218: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0
▶ Radiotap Header v0, Length 18
▼ IEEE 802.11 Null function (No data), Flags: ...P...T
  Type/Subtype: Null function (No data) (0x24)
  ▼ Frame Control: 0x1148 (Normal)
    Version: 0
    Type: Data frame (2)
    Subtype: 4
    Flags: 0x11
      ...01 = DS status: Frame from STA to DS via an AP (To DS: 1 From DS: 0) (0x01)
      ...0.. = More Fragments: This is the last fragment
      ...0... = Retry: Frame is not being retransmitted
      ...1... = PWR MGT: STA will go to sleep
      ..0. .... = More Data: No data buffered
      .0. .... = Protected flag: Data is not protected
      0... .... = Order flag: Not strictly ordered
    Duration: 162
    BSS Id: RovingNe_72:23:47 (00:06:06:72:23:47)
    Source address: IntelCor_bc:31:54 (08:24:d7:bc:31:54)
    Destination address: RovingNe_72:23:47 (00:06:06:72:23:47)
    Fragment number: 0
    Sequence number: 1834
  
```

Figure 6.3 – Dissection of the Null Data frame which evidences that the Power Management bit is on.

Moreover, the same type of frame, with the Power Management bit set to 1, can be used also to perform an active scanning of available WLAN networks on other channels. This is probably the cause of the Null Data frames sent by the client device with a periodic schedule. These COTS devices, as already stated, are strongly connectivity-oriented and need to have an updated overview of the available surrounding networks. Therefore, the aforementioned power management mechanism is adopted to perform an active channel scanning.

The scanning process can last a certain number of polling cycles, during which the corresponding command packets are necessarily buffered at the client device. When the scanning ends and the device exits from PS mode, all these packets are sent in succession. As a result, in the interframe times plot there is an upward peak (the delay between the last packets delivered before entering PS mode and the first one after the exit) followed

by a downward peak (the interval between two consecutively delivered packets after the end of this mode). This behavior explains the nature of the peaks visible, for example, in Fig. 5.3.

From the experimental measurements, it emerges that also the iPad uses these Null Data frames to perform network scanning, even if with a different periodicity.

In conclusion, this feature is very dangerous for the application and, since it is absolutely not necessary, it must be disabled.

6.1.3 Tests without network services

The two described issues are related to network services that are not needed by the considered control and monitoring application. The client device, therefore, can be appropriately tuned in order to block these services. In order to do this in a simpler and faster way, a PC based on Linux, mounting Ubuntu 13.10 as OS, has been used as client device. All the unnecessary network services have been disabled and an application similar to that used in the Windows PC has been developed. The environmental conditions in terms of interference, fading, LOS, ecc. are the same experienced in previous laboratory experiments.

The interframe times obtained without network services and using TCP as transport layer protocol are reported in Fig. 6.4, while Fig. 6.5 shows the same results with UDP as transport layer protocol.

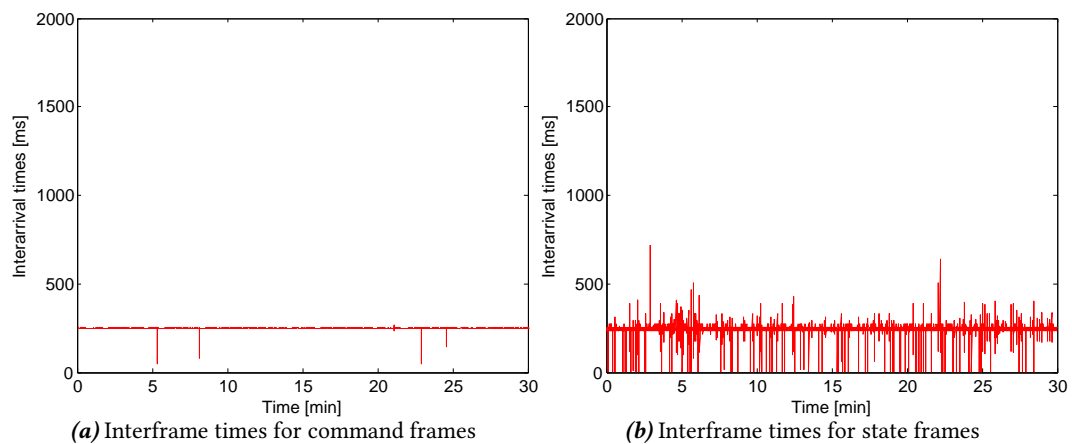


Figure 6.4 – Interframe times measured without network services using TCP.

The outcomes evidence a substantial regularity with respect to Fig. 5.3 and Fig. 5.5, which provided the same values with network services enabled.

In particular, the variability of interframe times for command frames is strongly limited with both TCP and UDP, with values very close to the ideal one (250 ms). The behavior

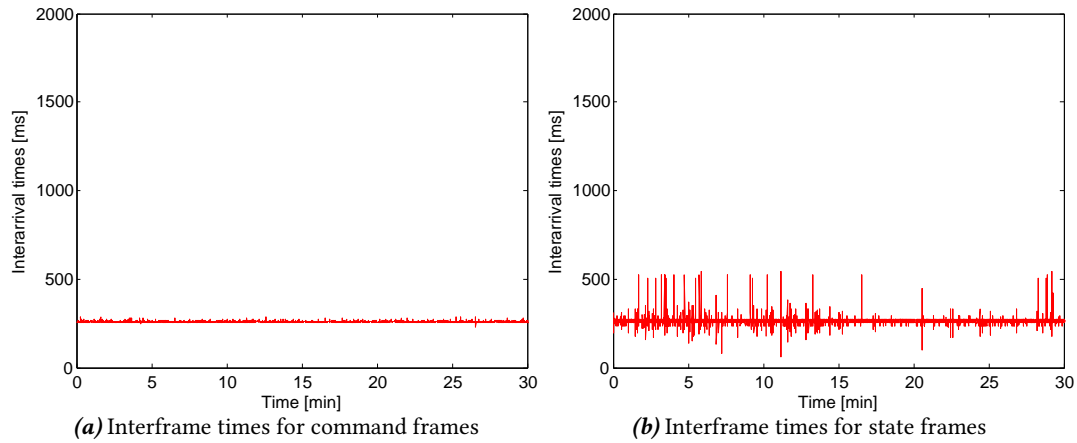


Figure 6.5 – Interframe times measured without network services using UDP.

of state frames is more unstable and a certain number of retransmissions is still required. However, the maximum value measured for the interframe time is around 500 ms with UDP and 700 ms with TCP, whereas in the previous sessions it raised up to more than 2 seconds.

As a final comparison, Fig. 6.6 presents the ECDF of interframe times for state frames without network services, for both TCP and UDP.

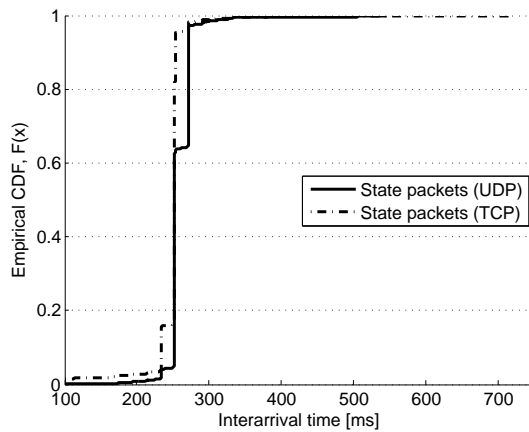


Figure 6.6 – ECDF of interframe times for state frames without network services.

By comparing this figure to Fig. 5.6, it is evident that the behavior without network services is much more stable and the quantity of packets with very low interframe times is much less relevant. It can be also noticed that there is a considerable amount of state frames that experience an interframe time of 270 ms (with UDP) or 230 ms (with TCP). The reason behind the presence of these values, shifted by 20 ms from the ideal value, relies in

the intrinsic behavior of the wireless module mounted on the ECU and will be discussed in the next Section.

6.2 Issues related to the wireless module

In addition to the problems caused by the adoption of COTS devices on the client side of the application, also the server side may reveal uncertainties due to some intrinsic and non-documented behaviors of the adopted wireless module.

6.2.1 Synchronization in message transmission

As briefly discussed in Section 4.3, the wireless module adopts a specific policy in order to trigger the transmission on the wireless interface of the data arrived from the ARM controller through the UART bus. The solution currently adopted specifies that a message is marked as completed when 20 ms are passed and no more data are received on the bus. Once this timeout is expired, the module encapsulates the received message with the lower-layer headers and send it over the wireless channel.

In the theoretical analysis conducted in Chapter 4, these 20 ms were considered as a fixed delay and added to the computation of total polling time. Actually, the measurements revealed that the time taken by the module to trigger the transmission varies at each cycle.

Fig. 6.7 displays the measured polling time for a series of consecutive state packets, that did not experience any retransmission, referenced by their TCP sequence number¹, taken from a capture during an experimental session in the laboratory. The polling time has been measured by the packet sniffer software as the time elapsed between the captures of the command packet and the corresponding state packet.

It can be seen that there is a certain variability and the difference between the maximum and the minimum value is exactly 20 ms. The experiment has been repeated by configuring the wireless module to wait 10 ms before sending a packet and the gap between the minimum and maximum value of the estimated polling time was this time 10 ms, so the motivation of this variability clearly relies in this transmission mechanism.

It is likely that, instead of continuously checking the presence of data on the UART bus and starting a 20 ms timeout once no data are detected, the module adopts a different mechanism. That is, the module checks the presence of data on the bus only every 20 ms and triggers the transmission if the last control returns a negative result while the previous one was positive. In this way, an arbitrary delay is added to the time at which the flow

¹In TCP the SN of each segment is equal to the SN of the last segment plus its total number of bytes.

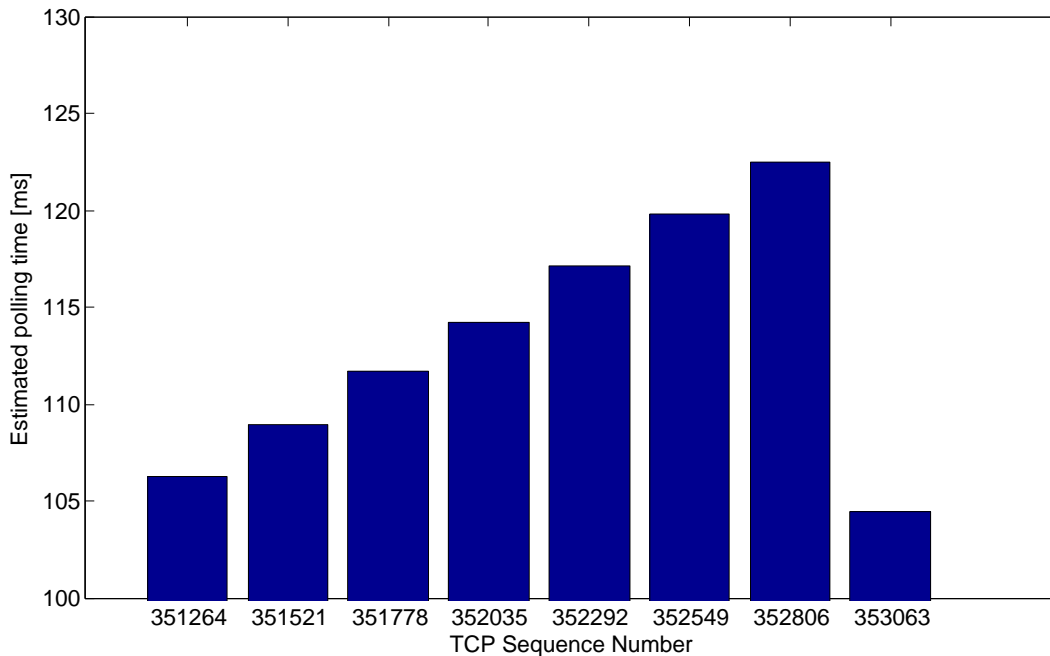


Figure 6.7 – Measures polling time for a series of consecutive state packets.

of data on the bus actually stopped. This delay, however, cannot exceed 20 ms. As a consequence, the 20 ms value has not to be considered as a fixed delay when computing polling time, but as an upper bound to a random delay.

The behavior of polling time shown in Fig. 6.7, due to the intrinsic behavior of the wireless module, explains also the fact, clearly visible in Fig. 6.6, that a considerable amount of TCP state frames registered an interframe time 20 ms lower than the other frames. To understand that, Tab. 6.1 provides a rough calculation of interframe time for the state packets shown in Fig. 6.7. The first packet (SN 351264) is considered relative to the n -th cycle, started at time t . All the quantities in the table are in milliseconds.

It emerges that, while the interframe time for most of the frames is approximately equal to the nominal value 250 ms, every once in a while there is a frame for which the interframe time is about 20 ms less than this value.

In order to eliminate this randomness from the communication, another modality of triggering data transmission in the wireless module should be considered. This module offers three other possible modes:

- ✓ **Length-based:** the module sends a packet on the wireless channel every time a certain number of bytes has been accumulated on the data buffer. The adoption of this mode is unfeasible, since the message to be transmitted from the module have

Table 6.1 – Example of interframe times calculation which explains the role of wireless module synchronization issues.

Cycle number	Start time	Estimated polling time	Arrival time of state frame	Interframe time for state frame
n	t	106	$t + 106$	–
$n + 1$	$t + 250$	109	$t + 250 + 109$	253
$n + 2$	$t + 500$	112	$t + 500 + 112$	253
$n + 3$	$t + 750$	114	$t + 750 + 114$	252
$n + 4$	$t + 1000$	117	$t + 1000 + 117$	253
$n + 5$	$t + 1250$	120	$t + 1250 + 120$	253
$n + 6$	$t + 1500$	122	$t + 1500 + 122$	252
$n + 7$	$t + 1750$	104	$t + 1750 + 104$	232

different length.

- ✓ **Character-based:** the module sends a packet containing all the data in the buffer every time a specific character is sent over the [UART](#) bus. The adoption of this mode is possible, there is already a specific character declared in the protocol as the end character for a message (0x3E).
- ✓ **Hardware-based:** there is the possibility to connect a [CTS](#) pin in the module to an analog output of the [ARM](#) controller, so that the transmission could be commanded directly by the controller via an electric impulse. This possibility would be the most precise, but it would require modifications on the hardware of the [ECU](#).

The character-based mode seems the most feasible one and apparently no modification to the protocol is required, since all application messages already terminate with the same character, which could be used as the triggering character for data transmission. However, it has to be taken into account that the same character could appear also inside the message payload, whose content is random. To avoid that, appropriate *byte stuffing* techniques should be used, that will be discussed in the next Section, when an updated design of the protocol will be considered.

6.2.2 Interaction with other devices

The wireless module mounted on the [ECU](#) creates a [WLAN](#) in Infrastructure mode, in which it acts as an [AP](#). With the default configuration, this network is public and any IEEE 802.11 device can associate with it. To advertise the presence of the network, the module sends a *Beacon frame* with a certain periodicity (102.4 ms). This frame is a specific subtype

of IEEE 802.11 management frame, used by an AP to announce the presence of a WLAN. It contains all the information about the network, such as identifier, capability, supported data rates, traffic information and so on. It also serves for synchronization purposes inside the network, since a Timestamp field is present to regulate the clocks of all STAs.

The beacon frames do not interfere with the traffic generated by the polling application. However, their presence may be detrimental for the application because they are received by all IEEE 802.11-compliant devices that are close enough to the ECU, not only the client device on which the application is running. These devices, after receiving the beacon, reply with an IEEE 802.11 *Probe Request* frame, with which they require to the wireless module more information on the network. The module automatically answers with a *Probe Response* frame, containing the information required. At this point, the device can choose whether to associate to the network or not. The experimental sessions carried out in the research laboratory showed that probe requests and responses were received and sent by the module with a certain regularity from a wide range of devices (PCs, tablets, smartphones, ecc.). Moreover, in most of the cases these frames were retransmitted for a certain number of consecutive times, so the amount of resources subtracted to the polling application was even higher.

Clearly, for the conceived application, there is no need to have an open WLAN to which any device could connect, since only the devices that have the monitoring and control application installed could make an efficient use of the network. Therefore, the transmission of beacon frames from the wireless module has been disabled to hide the network from unauthorized devices. As a result, a device that has to be used as client for the application needs to manually associate to the hidden network.

6.3 Modifications to the protocol

The results obtained in Section 6.1.3 are quite good, but still better performance can be achieved. Moreover, there are some synchronization issues as well as other sources of randomness that need to be addressed, such as the problem described in Section 6.2.1. In this Section several modifications to the protocol described in Chapter 2 will be discussed and the performance figures obtained with the new configuration will be analyzed and compared to those previously achieved.

6.3.1 Choice of transport layer protocol

The first choice to be made was the selection of the transport layer protocol, between TCP and UDP. The results provided in Chapter 5 evidenced that UDP offered a more stable behavior. This result is expected, since TCP is known for offering a very poor real-time behavior [3]. Indeed, most of the times when a soft real-time behavior is required, neither UDP nor TCP are used, but packets are sent directly at MAC layer. However, since the wireless module adopted in this solution forces the use of a transport layer protocol, UDP is the best possible choice.

The advantages of using UDP over TCP are the reduced overhead in terms of header length (see Tab. 4.2) and the best-effort strategy in message delivery strategy, in the sense that retransmissions are not adopted and acknowledgement packets are not required. The drawback is the absence of any additional reliability other than that offered by data-link layer services. As a consequence, a retransmission mechanism can be considered at the application layer, to enhance the robustness of the system. However, a total reliability cannot be guaranteed and some cycle loss will be experienced. This is not a great problem for the system, since the control dynamics are not critical and older setpoint values can be conserved by the ECU for a cycle or two without compromising performance.

6.3.2 Changes in the structure of messages

In order to implement a retransmission mechanism at application layer and to recognize if a cycle has been lost, an appropriate numbering of messages should be introduced, by adding a new field. Moreover, if the character-based mode is selected in the wireless module to trigger the transmission of messages on the radio link (see Section 6.2.1), an appropriate byte-stuffing mechanism should be considered. These two expedients cause a modification in the length of messages, which is no more fixed to the values reported in Chapter 2, but can vary depending on the payload content. However, this can be done safely since the maximum length allowed (1460 Bytes) is much higher than current messages sizes.

Introduction of Sequence Numbers

To allow for message numbering, a *Sequence Number* field has been inserted in the message structure. The new message format is reported in Fig. 6.8.

The new SN field has length 1 Byte, so it can contain integers from 0 to 255, and it is present in all application messages exchanged between the client device and the ECU. Its

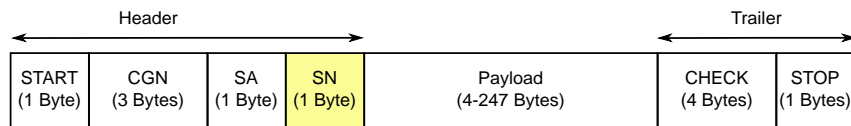


Figure 6.8 – New message format, with the new SN field highlighted in yellow.

value is fixed to 0 for *System_Info_App* and *ECU_Param* messages. For command messages, its value is set by a counter kept in the client device. This counter is initialized to 1 at the first polling cycle, then increased by 1 at each cycle. When the maximum value (255) is reached, the counter is reinitialized to 1. For state messages, this field contains the value of the SN field in the last received command message. In this way, the client side knows how many cycles have been lost.

Byte stuffing

Section 6.2.1 evidenced that the strategy adopted by the wireless module to trigger the transmission of messages on the wireless link was ineffective and led to random delays in the communication. As a consequence, a character-based strategy has been employed, which triggers the transmission of a message when a specific character is received on the UART bus. This character is clearly the STOP character inserted at the end of each application message (0x3E).

To ensure that this character appears only at the end of the message, and not in its payload, a byte stuffing technique is introduced. This technique consists in a preliminary coding of the message payload implemented by the ARM controller: each time this character appears in the payload, it is replaced by the *escape character* 0x3D. To allow for appropriate decoding, each time the escape character appears on the payload it is repeated. As an example, the following part of a payload

0x2A, 0x01, 0x3E, 0x3B, 0x3D, 0x3D, 0x3E, 0x1F

is encoded in this way

0x2A, 0x01, 0x3D, 0x3B, 0x3D, 0x3D, 0x3D, 0x3D, 0x3D, 0x1F

where the characters in blue represent a replacement (0x3D instead of 0x3E), while the characters in red represent a repetition of 0x3D. Clearly, each time a repetition is made the size of the encoded payload increases with respect to the original payload. Fig. 6.9 summarizes the coding algorithm in a schematic way.

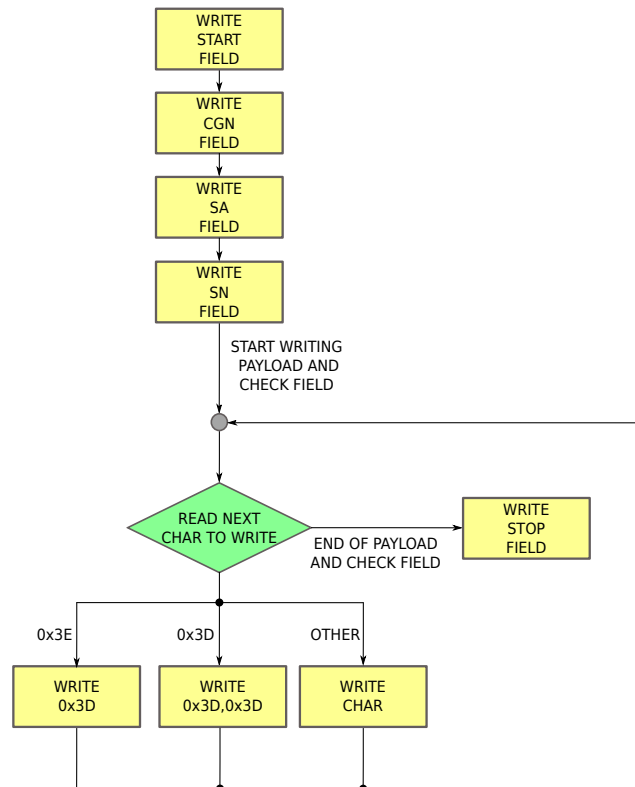


Figure 6.9 – Encoder for byte stuffing technique, implemented in the ECU.

In order to retrieve the original payload, an appropriate decoding strategy must be implemented at the client device, building on the coding algorithm. Fig. 6.10 reports the scheme of a possible decoder.

The decoder, in addition to retrieving the original message, makes also a series of controls on its content. In particular, the four following controls are performed:

1. The START character must be equal to 0x3A.
2. The CGN must be consistent with that expected (the CGN of *ECU_Param* in the setup phase, that of state message in the polling phase).
3. The SA must be that of the ECU (0x0).
4. The CRC code contained in the CHECK field must be correct.

If any of these controls fails, the message is discarded. Conversely, if all the controls have a positive result, the message is considered successfully received.

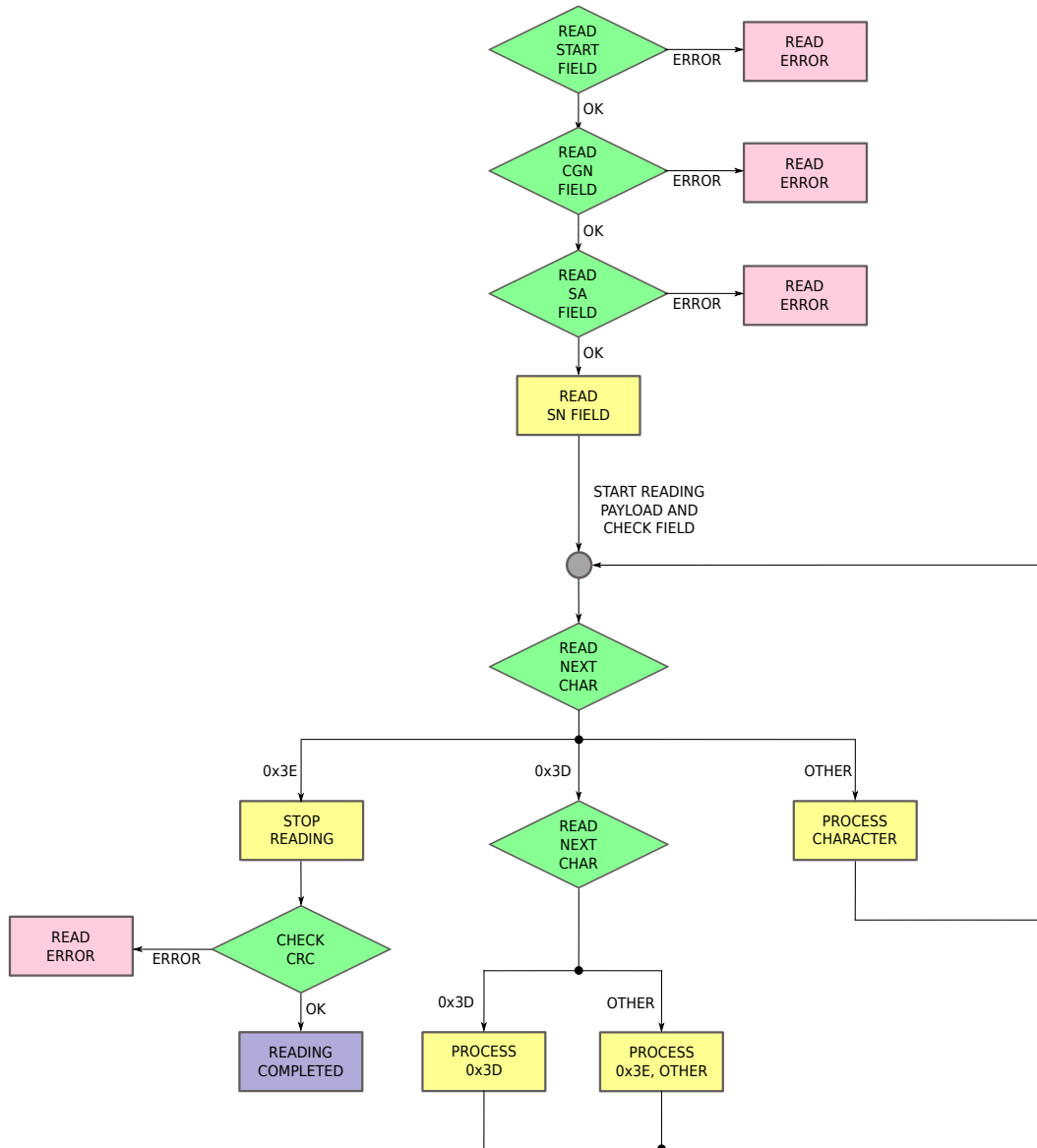


Figure 6.10 – Decoder for byte stuffing technique, implemented in the client device.

6.3.3 New structure of polling cycle

The introduction of the *SN* field allows for the implementation of a retransmission mechanism at application layer. In this way, the whole cycle period T_C is exploited to try to complete a polling cycle. If this does not happen by the end of the period, the cycle is considered as lost and a new one is started.

In order to compute an appropriate timeout value to declare the command message

as lost and perform a retransmission, an estimation of the polling time is required. The theoretical analysis performed in Chapter 4 may be helpful, even if some system parameters have changed. First of all, the waiting time at the ARM controller, T_{d_ARM} , has been removed, since it was not necessary for the correct behavior of the application and could cause synchronization problems, besides increasing significantly the polling time value. Then, the adoption of the character-based mode as transmission trigger in the wireless module allowed to remove the random delay (upper bounded by 20 ms) caused by the previously adopted mode. As a result, the whole T_{d_ECU} component, which accounted for 70 ms, is removed from the computation of polling time. Finally, the adoption of UDP instead of TCP reduces the transport layer header and, therefore, the transmission time of packets on the wireless interface. However, this reduction might be roughly compensated by the increase in the size of payload, due to the introduction of the SN field and (possible) byte stuffing operation.

As a final result, the maximum value for polling time computed in Chapter 4 should be decreased by 70 ms. If IEEE 802.11g is used, the maximum value for polling time is about 50 ms (even considering the lowest data rate of 6 Mb/s). However, the retransmission timeout at application layer has been set equal to 70 ms, to have a certain security margin. In this way, also the two higher rates adopted by IEEE 802.11b (5.5 Mb/s and 11 Mb/s) could be used.

If the retransmission timeout is set to such a value, in a cycle period, of duration $T_C=250$ ms, at most two application-layer retransmissions can be performed. This means that the same command message is transmitted for a maximum of three times.

Fig. 6.11 shows the updated protocol design for the client side. The setup phase is omitted, since no changes were made, and only the polling cycle is shown.

It can be seen that the polling process on the client device is characterized by two loops. An outer loop describes the sequence of polling cycles, sending a new command message, with updated SN, every 250 ms. An inner loop regulates the retransmission process inside each polling cycle. After the command message has been set, a 70 ms timeout is started. If a state message is received and decoded correctly before the expiration, the cycle is considered as correctly concluded and the device waits until the beginning of the next one. Conversely, if the timeout expires or there are errors in the decoding process, the same command message is retransmitted, without increasing the SN, and an internal variable n , which keeps count of the number of transmissions, is incremented. When this variable reaches the value 4, it means that the message has not been successfully delivered after three transmissions, so the cycle is declared lost and the device waits until the next

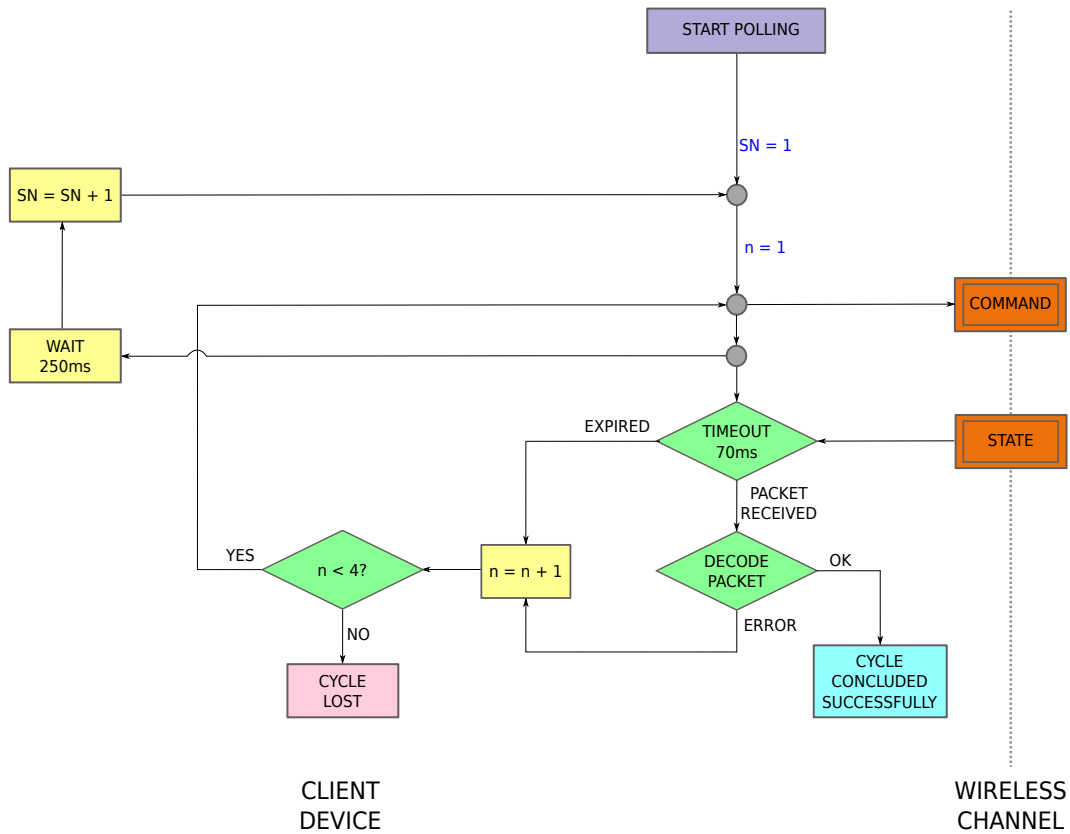


Figure 6.11 – New structure of polling cycle, client side.

one.

Fig. 6.12 shows the updated protocol design for the ECU side, for what concerns the polling cycle.

The 1 second timeout that was present in the original version of the protocol is maintained, to ensure that the socket is closed if the command messages are not received for a while. After the reception and successful decoding of a command message, the ECU checks its SN and compares it to the SN of the last received message, denoted with LN. If the new SN is the old value increased by 1, it means that the last cycle ended up correctly and this message is relative to the new cycle, so a new state message is loaded. Conversely, if $SN=LN$, it means that the last state message sent has not been correctly received and the received command message is the result of an application layer retransmission, therefore the last state message is reloaded. Finally, if SN is greater than $LN + 1$, it means that one or more cycles have been lost, and a new state message is loaded. After the end of this comparison, the ECU updates the LN variable and sends the loaded state message.

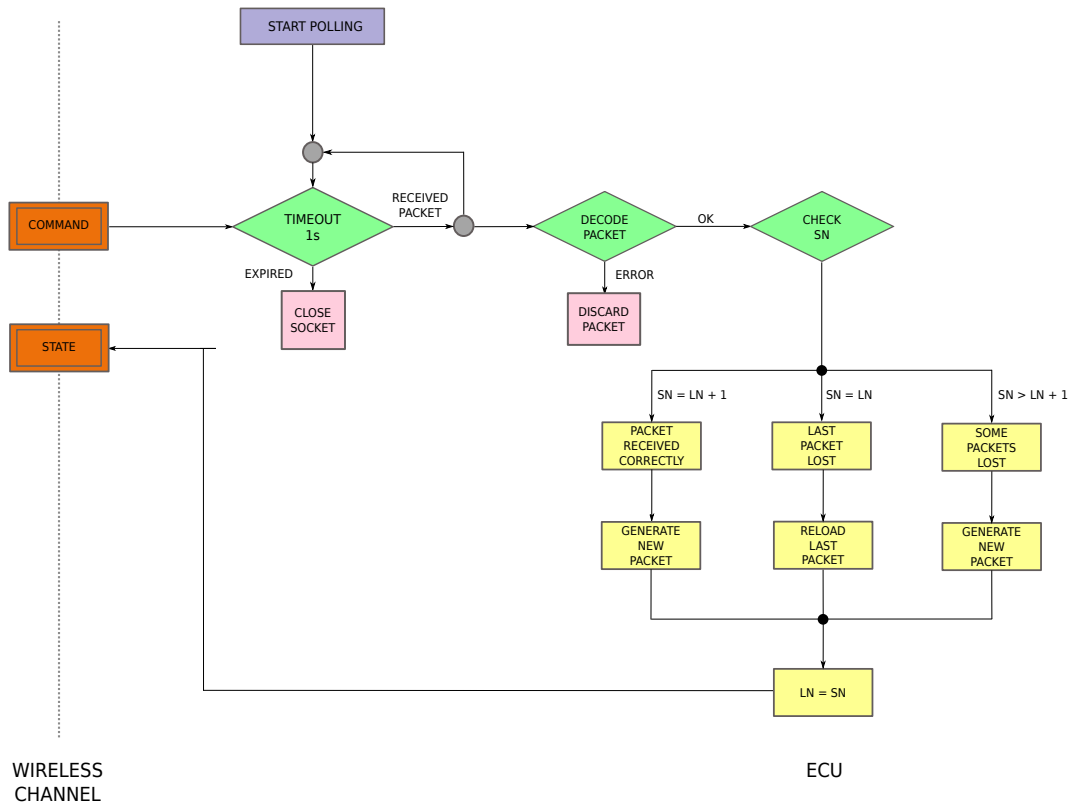


Figure 6.12 – New structure of polling cycle, ECU side.

The new structure of the polling cycle allows to fully exploit the cycle period, retransmitting unacknowledged messages in case of bad channel conditions. At the same time, a limit to the retransmission number is set, to avoid the problems that were observed with TCP, when the transport layer retransmission mechanism interfered with the MAC layer one. The results obtained with this new configuration, reported in the next Section, are very satisfactory.

6.3.4 Results with the new configuration

In order to implement the new configuration, both the firmware of the ARM controller on the ECU and the application software on the client device were modified in agreement with the considerations made in the previous paragraphs. The settings of the wireless module were also modified, in particular the transmission trigger that was set to character-based. The client device used was still a Linux based PC, in which the application software collected meaningful debug data, from which most of the results presented in this Section

have been derived. The second computer running a packet-sniffing software was still present. The environmental conditions were the same of the other experiments in the laboratory.

At the beginning, the network services that have been discussed in Section 6.1 have been kept active, to assess whether the new structure of the protocol is sufficiently robust to cope with the issues derived from the adoption of COTS devices. Subsequently, these services have been disabled, to assess whether a further performance improvement can be achieved.

With the new configuration, it is no longer interesting to study the interarrival times of command and state packets. Indeed, while in the original configuration the proximity of this metric to the cycle period provided an assessment of the system stability, with the modifications made this is no longer true, since the retransmission mechanism introduced at the application layer could cause consecutive state or command messages to be separated by sensibly different amounts of time. Therefore, a new metric was selected, namely the polling time measured at application layer by the client device. It is worth reminding that this quantity represents the time elapsed from the first transmission of a command message in a cycle to the successful reception of the corresponding state message.

Fig. 6.13 shows the polling time of each cycle, measured at application layer during a 30 minutes experimental session, when the network services are active.

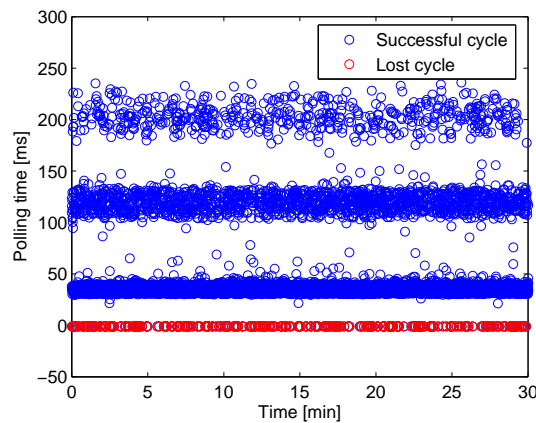


Figure 6.13 – Polling time at application layer with the new configuration and network services active.

It can be seen that the values of polling time are mainly distributed in three bands, spaced by approximately 70 ms. It is clear that these three different values are relevant to cycles that required a different number of transmissions in order to be completed. It is worth noting that the polling time never exceeds the cycle period $T_C=250$ ms. However,

there are some lost cycles, whose occurrence is indicated with a red circle positioned at 0 ms.

The distribution of polling times in three different bands is evident also in Fig. 6.14, which shows the ECDF of polling time measured at application layer with the network services enabled.

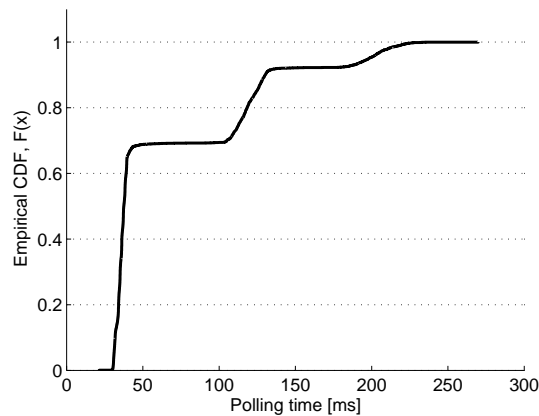


Figure 6.14 – ECDF of polling time with the new configuration and network services active.

From the figure, it emerges that, in most of the occasions (almost 70%), the polling time value falls in the first band (30–40 ms), which corresponds to the case when only one transmission of both the command and state messages is required. The other two bands are characterized by a wider range of values and the probability that a measured polling time value falls in these bands is sensibly lower: more than 20% for the 100–120 ms band (2 transmissions) and almost 10% for the 170–220 ms one (3 transmissions).

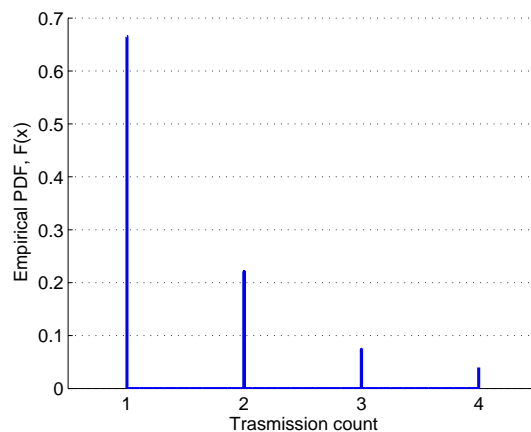


Figure 6.15 – PDF of the number of command message transmissions with the new configuration and network services active.

To better understand how often a cycle is lost, Fig. 6.15 shows the empirical PDF of the number of transmission of command message needed for each cycle. It can be seen that the percentage of lost cycles, indicated with a value of 4 for transmission number, is quite low (about 4%). The results already discussed about the probability that a successful cycle requires 1, 2 or 3 transmissions are also visible in this figure.

Another experimental session has been conducted with network and power management services disabled. Results are reported in Fig. 6.16.

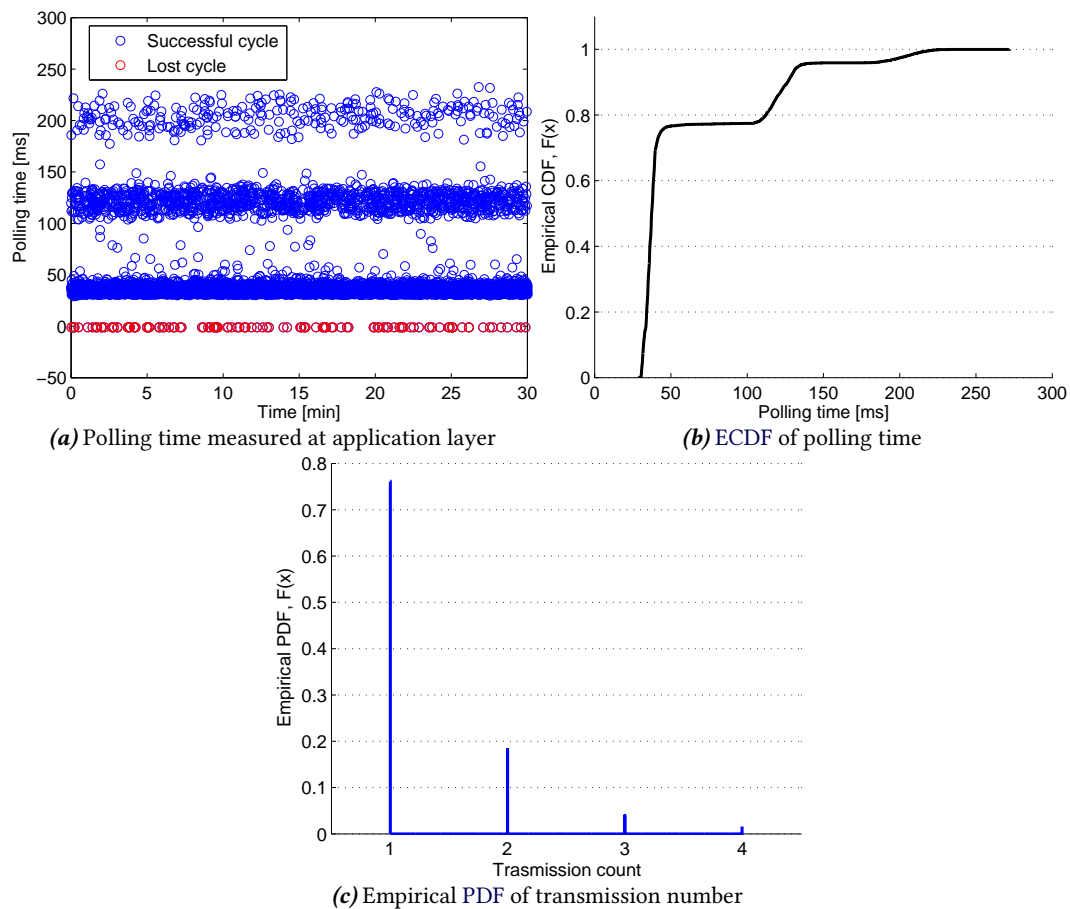


Figure 6.16 – Results obtained with the new configuration and network services disabled.

The trend is the same observed in the previous session with services enabled, with the values of polling time distributed in three bands. With a careful look, it can be seen that there has been a performance improvement, since the probability that the polling takes place without retransmissions is higher and the number of lost cycles is lower.

The performance improvement obtained through the disabling of network services is

evidenced in Tab. 6.2, which summarizes the empirical probabilities that each cycle requires 1, 2 or 3 transmissions (or is lost), comparing the session with services enabled and that with services disabled.

Table 6.2 – Empirical probabilities of cycle outcome with and without network services.

Scenario	Successful cycle			Lost cycle
	1 Transmission	2 Transmissions	3 Transmissions	
With services	66.6%	22.1%	7.4%	4.0%
Without services	76.2%	18.2%	4.0%	1.6%

Conclusions

This thesis presented an original application for monitoring and control in agricultural environments, based on a IEEE 802.11 communication architecture.

7.1 Overview of the activities

As a first relevant aspect, the proposed system for the real-time monitoring and control of an agricultural machine has been thoroughly analyzed, both from the hardware perspective, with a detailed description of all its components, and from the software perspective, with an overview of the adopted communication protocol.

The performance figures of the described system have been subsequently characterized from both the theoretical and the practical point of view. The theoretical analysis has taken into account all the components of the communication system that enable the exchange of control and monitoring data between the hand-held device and the control unit on the ECU. The time required for the transmission and elaboration of data has been carefully computed, with a particular attention to the components that introduced randomness in the communication. This task required a stochastic analysis of the transmission time over the wireless link, based on common IEEE 802.11 equations and models taken from the standard [13].

Several experimental sessions have been performed on the proposed system, in order to characterize its real performance figures and compare them to those obtained through the theoretical analysis. The system has been tested in several operational conditions, which offered different degrees of measurement accuracy and configuration possibility. The results of the experimental sessions highlighted a certain number of issues which

caused a performance degradation with respect to the expected behavior. Additional sessions have been carried out to collect further information about these issues.

The main causes of performance degradation have been identified and a description of their effects on the system performance has been given. Subsequently, viable solutions to these issues have been proposed and new measurement campaigns have been performed, showing strong improvements of the system performance.

Finally, a new design of the communication protocol has been presented, as a result of the combination between the theoretical study of the system and the observation of its real behavior, in order to improve robustness and timeliness of the system. The updated protocol has been implemented on the system and other measurement sessions have been carried out, demonstrating a much better system behavior.

During the course of this work, several software and hardware tools have been used to achieve the aforementioned results. The greatest part of work was dedicated to the measurement campaigns and to the elaboration of the collected data, mainly the capture provided by the packet-sniffing software Wireshark™. The data taken from the measurements have been subsequently elaborated with Matlab™ to produce the plots visible in Chapter 5 and Chapter 6. Matlab has also been used to perform the theoretical analysis in Chapter 4. To write the code for the application several languages and IDEs have been used, mainly C, C++ and Python. A network simulator [1] has been occasionally used to simulate the considered communication system and test its performance.

7.2 Review of the results

This thesis had a clear goal: to analyze the performance of the proposed monitoring and control system, identify the main issues in the communication, propose and test valuable solutions to obtain a more satisfactory behavior. However, the work done to reach this goal allowed to achieve a series of relevant results which may reveal interesting also for soft real-time wireless control systems similar to the one proposed in this thesis and, in general, for industrial communication systems based on the IEEE 802.11 standard.

To this concern, the main result achieved was definitely the analysis of the behavior of available COTS devices when adopted as nodes in a soft real-time system of this kind. While these devices are commonly adopted because of their availability and low cost, it emerged that their intrinsic features can introduce considerable performance degradation, to the point that a true real-time behavior is not achievable by the overall system. Indeed, these devices are not natively conceived to be employed in such a system, whereas

they are designed to strongly interact with other ICT systems and to rapidly adapt to the surrounding communication environment. To these purposes, several network services are natively implemented and enabled in these devices and run in background, possibly interfering with the real-time requirements of the wireless control application.

Another issue highlighted by this work was that the wireless module adopted to act as IEEE 802.11 AP could have some intrinsic and non-documented behaviors that might introduce randomness in the system, often leading to strong discordances between theoretical results and experimental measurements. Moreover, in the agricultural field of applications the constraints on the module characteristics, particularly in terms of temperature and humidity range, greatly restrict the range of available wireless modules. This could result, as it did in this work, in the selection of a module which offers a very limited possibility of configuration and excludes the adoption of well-known techniques to enhance performance of wireless networks, forcing to find new and creative ways to achieve the desired results.

An example of the limitations imposed by the module was that it did not allow to directly send and receive data at MAC layer, imposing the adoption of a transport layer protocol. To this concern, a known result has been confirmed, namely that TCP is not a good choice as transport layer protocol if a real-time behavior is required, since it introduces delays and retransmissions in order to achieve a total reliability, which was less important than timeliness for the considered system. UDP performs much better since it offers a best-effort delivery service, at the cost of being unreliable in the transmission of packets. Appropriate measurements to ensure the robustness of the system, in addition to those already provided at data link-layer, have been considered at application layer.

7.3 Future works

The communication protocol, with the new structure proposed for the polling cycle, could be easily extended with the addition of new types of messages. For example, it has already been foreseen the exchange between the remote device and the ECU of a set of system parameters which are normally fixed but that can be modified on-demand, with a specific option on the client application. This could be done with the creation of new *parameter request* and *parameter response* messages to be exchanged between the two devices whenever the client requires it. However, this exchange would be asynchronous with respect to the running cyclic polling operation and might interfere with it. A better option would be the creation of a new command message, containing both updated set points and com-

mands and the parameters to modify, that can replace the traditional command message in a given polling cycle upon client request, thus not breaking the polling synchronism.

The updated protocol should be again tested on the real application environment, to see if the satisfactory results obtained in the research laboratory still hold. Moreover, the client side of the application should be implemented for different types of fixed and mobile devices and tested, in order to establish their different behaviors and study how to properly tune them in order to reach the desired results.

The protocol could be modified to allow the connection of multiple client devices to the ECU simultaneously. The Infrastructure mode of the WLAN already allows for this possibility and the protocol already foresees it (see SA field in the header). However, the theoretical analysis must be repeated considering the case of multiple devices and a dimensioning of the system is required (for example what is the maximum number of client devices that can be connected to ensure good performance figures). Moreover, from the control perspective it does not make sense (and could be dangerous) that different devices could control the same variable simultaneously. The system should be implemented so that one device at a time has control privileges, while the others can only monitor the system state.

If a different wireless communication module can be mounted on the ECU, with additional features, several solutions can be considered to improve the performance of the overall system. For example, the adoption of IEEE 802.11n and MIMO could limit the effect of multi-path and interference, therefore improving the system robustness. Appropriate rate adaptation techniques, opportunely designed for the specific system and environment, could be considered to reduce the effect of fading, possibly allowing to monitor and control the machine with the remote device from greater distances. Finally, the possibilities offered by IEEE 802.11e in terms of traffic prioritization and channel access might allow to obtain a more real-time behavior.

Bibliography

- [1] Omnet++. [Online]. Available: <http://www.omnetpp.org>
- [2] Wireshark. [Online]. Available: <http://www.wireshark.org>
- [3] H. Balakrishnan, V. Padmanabhan, S. Seshan, and R. Katz, "A comparison of mechanisms for improving tcp performance over wireless links," *Networking, IEEE/ACM Transactions*, vol. 5, no. 6, pp. 756–769, 1997.
- [4] J. Camp and E. W. Knightly, "The IEEE 802.11 s extended service set mesh networking standard," *Communications Magazine, IEEE*, vol. 46, no. 8, pp. 120–126, 2008.
- [5] S. Cox, "Information technology: the global key to precision agriculture and sustainability," *Computers and Electronics in Agriculture*, vol. 36, no. 2–3, pp. 93 – 111, 2002.
- [6] J. del Prado Pavon and S. Choi, "Link adaptation strategy for IEEE 802.11 WLAN via received signal strength measurement," in *Communications, 2003. ICC '03. IEEE International Conference on*, vol. 2, May 2003, pp. 1108–1113 vol.2.
- [7] W. Gu, Z. Yang, D. Xuan, W. Jia, and C. Que, "Null data frame: A double-edged sword in IEEE 802.11 WLANs," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 21, no. 7, pp. 897–910, July 2010.
- [8] V. C. Gungor and G. P. Hancke, "Industrial Wireless Sensor Networks: Challenges, Design Principles, and Technical Aspects," *IEEE Trans. on Industrial Electronics*, vol. 56, no. 10, pp. 4258–4265, October 2009.
- [9] J. Gustavsson, C. Cederberg, U. Sonesson, R. Van Otterdijk, and A. Meybeck, *Global food losses and food waste: extent, causes and prevention*. FAO Rome, 2011.
- [10] B. Havenkort, *Performance of Computer Communication Systems—A Model Based Approach*. Chichester, UK: Wiley, 1998.
- [11] G. Holland, N. H. Vaidya, and P. Bahl, "A Rate-Adaptive MAC Protocol for Multi-Hop Wireless Networks," in *Proceedings of the ACM SIGMOBILE 7/01*, 2001, pp. 236–251.

- [12] K. Hong, S. Lee, and N. Golmie, "Throughput Study for Admission Control in IEEE 802.11 DCF with ARF," *Communications Letters, IEEE*, vol. 13, no. 6, pp. 432–434, June 2009.
- [13] *IEEE Standard for Information technology–Telecommunications and information exchange between systems–Local and metropolitan area networks–Specific requirements. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Std., June 2012.
- [14] Y. Jiber, H. Harroud, and A. Karmouch, "Precision agriculture monitoring framework based on WSN," in *Wireless Communications and Mobile Computing Conference (IWCMC), 2011 7th International*, July 2011, pp. 2015–2020.
- [15] A. Kamerman and L. Monteban, "WaveLAN-II: A High-Performance Wireless LAN for the Unlicensed Band," *Bell Labs Technical Journal*, pp. 118–133, August 1997.
- [16] M. Lacage, M. H. Manshaei, and T. Turletti, "IEEE 802.11 Rate Adaptation: a Practical Approach," in *Proceedings of the 7th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*, ser. MSWiM '04. New York, NY, USA: ACM, 2004, pp. 126–134.
- [17] Z. Liao, S. Dai, and C. Shen, "Precision agriculture monitoring system based on wireless sensor networks," in *Wireless Communications and Applications (ICWCA 2012), IET International Conference on*, Oct 2012, pp. 1–5.
- [18] O. Mirabella and M. Brischetto, "A Hybrid Wired/Wireless Networking Infrastructure for Greenhouse Management," *Instrumentation and Measurement, IEEE Transactions on*, vol. 60, no. 2, pp. 398–407, Feb 2011.
- [19] R. Moraes, F. Vasques, P. Portugal, and J. Fonseca, "VTP-CSMA: A Virtual Token Passing Approach for Real-Time Communication in IEEE 802.11 Wireless Networks," *Industrial Informatics, IEEE Transactions on*, vol. 3, no. 3, pp. 215–224, 2007.
- [20] G. Patti, S. Denaro, G. Alderisi, and L. Lo Bello, "A three-tiered architecture based on IEEE 802.15.4 and ethernet for precision farming applications," in *Industrial Electronics Society, IECON 2013 - 39th Annual Conference of the IEEE*, Nov 2013, pp. 4444–4450.
- [21] E. Perahia and R. Stacey, *Next Generation Wireless LANs: Throughput, Robustness, and Reliability in 802.11n*, ser. IT Pro.
- [22] R. Poovendran, K. Sampigethaya, S. K. S. Gupta, I. Lee, K. V. Prasad, D. Corman, and J. Paunicka, "Special issue on cyber - physical systems [scanning the issue]," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 6–12, Jan 2012.
- [23] L. Seno, F. Tramarin, and S. Vitturi, "Tuning of IEEE 802.11 MAC for Improving Real-Time in Industrial Wireless Networks," in *Proc. of IEEE Conf. Emerging Technologies & Factory Automation ETFA 2012*, Warsaw, Poland, September 2012.

-
- [24] P. Silva, C. Serodio, and J. Monteiro, "Ubiquitous scada systems on agricultural applications," in *Industrial Electronics, 2006 IEEE International Symposium on*, vol. 4, July 2006, pp. 2978–2983.
- [25] F. Tramarin, "Industrial wireless sensor networks - simulation and measurement in an interfering environment," Ph.D. dissertation, Univ. of Padua, Padova, 2011.
- [26] US National Science Foundation. Founding – Cyber-Physical Systems. [Online]. Available: https://www.nsf.gov/funding/pgm_summ.jsp?pims_id=503286
- [27] S. Vitturi, L. Seno, F. Tramarin, and M. Bertocco, "On the rate adaptation techniques of ieee 802.11 networks for industrial applications," *Industrial Informatics, IEEE Transactions on*, vol. 9, no. 1, pp. 198–208, Feb 2013.
- [28] Z. Wang, C. Zhao, H. Zhang, and H. Fan, "Real-Time Remote Monitoring and Warning System in General Agriculture Environment," in *Information Technology, Computer Engineering and Management Sciences (ICM), 2011 International Conference on*, vol. 3, Sept 2011, pp. 160–163.
- [29] Y.-H. Wei, Q. Leng, S. Han, A. Mok, W. Zhang, and M. Tomizuka, "Rt-wifi: Real-time high-speed communication protocol for wireless cyber-physical control applications," in *Real-Time Systems Symposium (RTSS), 2013 IEEE 34th*, Dec 2013, pp. 140–149.
- [30] N. Zhang, M. Wang, and N. Wang, "Precision agriculture – a worldwide overview," *Computers and Electronics in Agriculture*, vol. 36, no. 2–3, pp. 113 – 132, 2002.