Universidad de Oviedo
Universidá d'Uviéu
University of Oviedo

Ph.D. Thesis

Design and implementation of a high-performance hybrid network for industrial applications with real-time requirements

Óscar Seijo Gómez

Supervisors: Iñaki Val Beitia
Jesús Alberto López Fernández

Programa de Doctorado en Tecnologías de la Información y Comunicaciones en Redes Móviles

2020

Justificación

El trabajo realizado por el autor de la tesis tiene gran calidad técnica y es altamente innovador, lo que está refrendado por diferentes publicaciones científicas tanto en revistas como en conferencias de ámbito internacional.

Publicaciones en revistas internacionales:

[1] Z. Fernandez, **Ó. Seijo**, M. Mendicute, and I. Val, "Analysis and evaluation of a wired/wireless hybrid architecture for distributed control systems with mobility requirements," *IEEE Access*, vol. 7, pp. 95915–95931, 2019.

[2] **Ó. Seijo**, J. A. López-fernández, H.-P. Bernhard, and I. Val, "Enhanced Timestamping Method for Sub-Nanosecond Time Synchronization in IEEE 802.11 over WLAN Standard Conditions," *IEEE Transactions on Industrial Informatics*, January 2020.

[3] **Ó. Seijo**, J. A. López-fernández, and I. Val, "w-SHARP: Implementation of a High-Performance Wireless Time-Sensitive Network for Low Latency and Ultra-Low Cycle Time Industrial Applications," *IEEE Transactions on Industrial Informatics*, June 2020.

[4] **Ó. Seijo**, J. A. López-fernández, and I. Val. "Portable Full Channel Sounder for industrial wireless applications with mobility by Using Wireless Sub-Nanosecond Time Synchronization," *IEEE Access,* September 2020.

[5] J. Montalbán, E. Iradier, P. Angueira, **Ó. Seijo**, and I. Val "NOMA-based 802.11 for Industrial Automation," *IEEE Access*, September 2020.

Publicaciones en conferencias internacionales:

[1] **Ó. Seijo**, C. Cruces, R. Torrego, I. Val, and J. A. López-fernández, "IEEE 1588 Hardware-Based Timestamp Implementation over 802.11a/g for IWSAN with High Precision Synchronization Requirements," *IEEE Int. Symp. Precis. Clock Synchronization Meas. Control. Commun. ISPCS*, 2017.

[2] **Ó. Seijo**, Z. Fernández, I. Val, and J. A. López-Fernández, "SHARP : A Novel Hybrid Architecture for Industrial Wireless Sensor and Actuator Networks," in *IEEE International Workshop on Factory Communication Systems (WFCS)*, 2018.

[3] Z. Fernández, **Ó. Seijo**, I. Val, and M. Mendicute, "Soft-Handover Algorithm for Hybrid Industrial Wireless Sensor and Actuator Networks," *IEEE Int. Conf. Emerg. Technol. Fact. Autom. ETFA*, 2018.

[4] **Ó. Seijo**, I. Val, J. A. López-fernández, and M. Vélez, "IEEE 1588 Clock Synchronization Performance over Time-Varying Wireless Channels," *IEEE Int. Symp. Precis. Clock Synchronization Meas. Control. Commun. ISPCS*, 2018.

[5] **Ó. Seijo**, Z. Fernández, I. Val, and J. A. Lopez-Fernandez, "SHARP: Towards the Integration of Time-Sensitive Communications in Legacy LAN/WLAN," *2018 IEEE Globecom Work. GC Wkshps 2018 - Proc.*, 2018.

[6] **Ó. Seijo**, I. Val, and J. A. López, "On the use of White Rabbit for Precise Time Transfer in 5G URLLC Networks for Factory Automation Applications," *IEEE Int. Conf. Ind. Cyber-Physical Syst.*, 2019.

[7] E. Iradier, J. Montalban, L. Fanari, P. Angueira, **Ó. Seijo**, and I. Val, "NOMA-based 802.11n for Broadcasting Multimedia Content in Factory Automation Environments," *IEEE Int. Conf. Emerg. Technol. Fact. Autom. ETFA*, 2019.

[8] **Ó. Seijo**, I. Val, and J. A. López, "Portable Full Channel Sounder for Mobile Robotics by using Sub-Nanosecond Time Synchronization over Wireless," *IEEE International Workshop on Factory Communication Systems (WFCS)*, 2020.

[9] L. Fanari, E. Iradier, J. Montalban, P. Angueira, **Ó. Seijo**, and I. Val, "PEG-LDPC Coding for Critical Communications in Factory Automation," *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2020.

Universidad de Oviedo
Universidá d'Uviéu
University of Oviedo

RESUMEN DEL CONTENIDO DE TESIS DOCTORAL

1.- Título de la Tesis	
Español/Otro Idioma: Diseño e implementación de una red híbrida de alto rendimiento para aplicaciones industriales con requisitos de tiempo real.	Inglés: Design and implementation of a high-performance hybrid network for industrial applications with real-time requirements.

2.- Autor	
Nombre: Óscar Seijo Gómez	DNI/Pasaporte/NIE:
Programa de Doctorado: TICRM	
Órgano responsable: Centro Internacional de Postgrado	

RESUMEN (en español)

Las aplicaciones con requisitos de tiempo real en la era de la industria 4.0. necesitan sistemas de comunicaciones con características muy específicas y complejas como, por ejemplo, muy baja latencia, ultra-alta fiabilidad, determinismo, flexibilidad de operación, y configurabilidad. Estas demandas están prácticamente resueltas en redes cableadas, y, por tanto, las aplicaciones industriales son comúnmente desarrolladas sobre tecnologías cableadas, como Ethernet. Al igual que en el mercado de consumo, en el que las redes de comunicaciones inalámbricas han reemplazado gradualmente las comunicaciones cableadas (especialmente en la última milla), las comunicaciones inalámbricas están siendo consideradas por la industria como la llave para conseguir alcanzar los niveles de automatización esperados en la industria 4.0. Las razones principales del interés en el uso de comunicaciones inalámbricas son: su bajo coste de montaje, su alta escalabilidad, su flexibilidad, y la posibilidad de que los nodos comunicados inalámbricamente se muevan libremente. Sin embargo, el diseño de una solución inalámbrica que cumpla los requisitos industriales es un desafío considerable, y los sistemas de comunicación actuales no alcanzan el rendimiento necesario. Las causas son diversas, pero la mayoría de los desafíos en comunicaciones industriales inalámbricas son derivados, en primer lugar, del impredecible comportamiento de los canales inalámbricos industriales y, en segundo lugar, de los requisitos tan complejos de las aplicaciones industriales.

En esta tesis se proponen diversas soluciones para reducir las diferencias de rendimiento existentes entre comunicaciones inalámbricas y comunicaciones cableadas en aplicaciones industriales. En primer lugar, esta tesis estudia los desafíos en sincronización de reloj inalámbrica bajo condiciones realistas. A partir de este estudio, se proponen dos esquemas de sincronización, que son evaluados mediante simulaciones y mediante una implementación experimental en una plataforma SDR. En segundo lugar, se ha diseñado e implementado un medidor de canal basado en uno de los esquemas de sincronización de alto rendimiento. Comparado con un medidor de canal tradicional, el medidor desarrollado en esta tesis es más sencillo de usar, más flexible, y más barato. Por último, se ha diseñado, implementado, y validado un sistema de comunicaciones inalámbrico de alto rendimiento denominado w-SHARP. El sistema ha sido implementado en una plataforma SDR y se ha validado en un entorno industrial realista. Los resultados obtenidos mediante los experimentos en hardware demuestran que w-SHARP mejora los resultados de los estándares 5G y 802.11ax para aplicaciones industriales con requisitos de tiempo real.

Universidad de Oviedo
Universidá d'Uviéu
University of Oviedo

RESUMEN (en Inglés)

Real-Time applications in the scope of the industry 4.0. require communication systems with very challenging features, such as low latency, ultra-reliability, determinism, flexibility, and reconfigurability. These challenges are mostly solved in wired networks, and, consequently, industrial applications are usually built with wired technologies (Ethernet, fiber links). As in the consumer market, where wireless communications have been progressively replacing wired communications at the edge of the network, wireless is now being considered in industrial applications as the key driver to achieve the industry 4.0. vision. The main reasons behind the interest in industrial wireless are their lower commissioning costs, higher scalability, flexibility, and the possibility of free movement of the wireless nodes. Nonetheless, the design of a wireless solution that can be seamlessly used in industrial environments is a serious challenge, and nowadays wireless systems struggle to fulfill the industry 4.0. requirements. The causes are diverse, though most of the challenges in industrial wireless are derived, first, from the unpredictable behavior of the wireless transmission medium, which is especially harsh in industrial environments, and second, from the stringent requirements of industrial applications.
This thesis proposes several solutions to reduce the gap between the performance of industrial wired and wireless communications. In the first place, the challenges of wireless synchronization are studied over realistic conditions, and two wireless synchronization schemes are proposed and evaluated through simulation and experimental means. In the second place, a portable channel sounder is designed and implemented based on one of the wireless synchronization schemes. Compared to a traditional channel sounder, the one developed during this thesis is simpler to use, more flexible, and more cost-effective. Finally, a high-performance industrial wireless system named w-SHARP is designed, implemented in a hardware-based Software Defined Radio platform, and validated under a realistic industrial environment. The performance evaluation through the hardware testbed demonstrates that w-SHARP outperforms 5G and 802.11ax for Real-Time industrial applications.

SR. PRESIDENTE DE LA COMISIÓN ACADÉMICA DEL PROGRAMA DE DOCTORADO EN Tecnologías de la Información y Comunicaciones en Redes Móviles

Para todos aquellos que crean que esta tesis podría
resultarles útil para su trabajo actual
y futuro
Y a los que han sufrido más de la cuenta
en este 2020.

For those who believe that this thesis
may be useful for their current
and future work
and for those who have suffered too much
during 2020.

Óscar

Abstract

Real-Time applications in the scope of the industry 4.0. require communication systems with very challenging features, such as low latency, ultra-reliability, determinism, flexibility, and reconfigurability. These challenges are mostly solved in wired networks, and, consequently, industrial applications are usually built with wired technologies (Ethernet, fiber links). As in the consumer market, where wireless communications have been progressively replacing wired communications at the edge of the network, wireless is now being considered in industrial applications as the key driver to achieve the industry 4.0. vision. The main reasons behind the interest in industrial wireless are their lower commissioning costs, higher scalability, flexibility, and the possibility of free movement of the wireless nodes. Nonetheless, the design of a wireless solution that can be seamlessly used in industrial environments is a serious challenge, and nowadays wireless systems struggle to fulfill the industry 4.0. requirements. The causes are diverse, though most of the challenges in industrial wireless are derived, first, from the unpredictable behavior of the wireless transmission medium, which is especially harsh in industrial environments, and second, from the stringent requirements of industrial applications.

This thesis proposes several solutions to reduce the gap between the performance of industrial wired and wireless communications. In the first place, the challenges of wireless synchronization are studied over realistic conditions, and two wireless synchronization schemes are proposed and evaluated through simulation and experimental means. In the second place, a portable channel sounder is designed and implemented based on one of the wireless synchronization schemes. Compared to a traditional channel sounder, the one developed during this thesis is simpler to use, more flexible, and more cost-effective. Finally, a high-performance industrial wireless system named w-SHARP is designed, implemented in a hardware-based Software Defined Radio platform, and validated under a realistic industrial environment. The performance evaluation through the hardware testbed demonstrates that w-SHARP outperforms 5G and 802.11ax for Real-Time industrial applications.

Resumen

Las aplicaciones con requisitos de tiempo real en la era de la industria 4.0. necesitan sistemas de comunicaciones con características muy específicas y complejas como, por ejemplo, muy baja latencia, ultra-alta fiabilidad, determinismo, flexibilidad de operación, y configurabilidad. Estas demandas están prácticamente resueltas en redes cableadas, y, por tanto, las aplicaciones industriales son comúnmente desarrolladas sobre tecnologías cableadas, como Ethernet. Al igual que en el mercado de consumo, en el que las redes de comunicaciones inalámbricas han reemplazado gradualmente las comunicaciones cableadas (especialmente en la última milla), las comunicaciones inalámbricas están siendo consideradas por la industria como la llave para conseguir alcanzar los niveles de automatización esperados en la industria 4.0. Las razones principales del interés en el uso de comunicaciones inalámbricas son: su bajo coste de montaje, su alta escalabilidad, su flexibilidad, y la posibilidad de que los nodos comunicados inalámbricamente se muevan libremente. Sin embargo, el diseño de una solución inalámbrica que cumpla los requisitos industriales es un desafío considerable, y los sistemas de comunicación actuales no alcanzan el rendimiento necesario. Las causas son diversas, pero la mayoría de los desafíos en comunicaciones industriales inalámbricas son derivados, en primer lugar, del impredecible comportamiento de los canales inalámbricos industriales y, en segundo lugar, de los requisitos tan complejos de las aplicaciones industriales.

En esta tesis se proponen diversas soluciones para reducir las diferencias de rendimiento existentes entre comunicaciones inalámbricas y comunicaciones cableadas en aplicaciones industriales. En primer lugar, esta tesis estudia los desafíos en sincronización de reloj inalámbrica bajo condiciones realistas. A partir de este estudio, se proponen dos esquemas de sincronización, que son evaluados mediante simulaciones y mediante una implementación experimental en una plataforma SDR. En segundo lugar, se ha diseñado e implementado un medidor de canal basado en uno de los esquemas de sincronización de alto rendimiento. Comparado con un medidor de canal tradicional, el medidor desarrollado en esta tesis es más sencillo de usar, más flexible, y más barato. Por último, se ha

diseñado, implementado, y validado un sistema de comunicaciones inalámbrico de alto rendimiento denominado w-SHARP. El sistema ha sido implementado en una plataforma SDR y se ha validado en un entorno industrial realista. Los resultados obtenidos mediante los experimentos en hardware demuestran que w-SHARP mejora los resultados de los estándares 5G y 802.11ax para aplicaciones industriales con requisitos de tiempo real.

Conclusiones

Esta tesis aborda diferentes desafíos de las redes de comunicaciones inalámbricas para aplicaciones industriales utilizadas como buses de campo, especialmente los desafíos relacionados con las capas más inferiores de la pila de protocolos de comunicaciones. En base a estos desafíos, tres temas de investigación han sido explorados, que están descritos a lo largo de los capítulos 3, 4, y 5.

El capítulo 3 discute desafíos de la sincronización de reloj sobre comunicaciones inalámbricas en entornos realistas (que incluyan variación del canal y propagación con multicamino). En este capítulo se analizan dichos desafíos y se proponen dos posibles esquemas de sincronización, que están basados en dos métodos de timestamping diferentes, denominados convencionales y mejorados. El método de timestamping convencional es el utilizado típicamente en redes cableadas (como Ethernet). Por ello, está pensado para canales casi ideales, con gran ancho de banda, alta SNR, y baja dispersión. Por otro lado, los timestamps mejorados están diseñados para combatir la propagación multicamino de los escenarios industriales. Ambos métodos de timestamping han sido diseñados para ser independientes del canal inalámbrico y sistema de comunicación y, por lo tanto, se pueden utilizar en casi cualquier escenario. Los esquemas de sincronización propuestos han sido evaluados mediante análisis numérico y mediante un demostrador hardware. Los resultados mediante análisis numérico demuestran que ambos esquemas son apropiados para implementar sincronización de reloj en redes de comunicaciones inalámbricas, si bien es cierto que los timestamps mejorados dan una precisión sensiblemente mayor. Esta conclusión es reforzada por los experimentos en el demostrador HW utilizando un modem WiFi implementado sobre una plataforma SDR basada en SoC-FPGA.

El capítulo 4 describe el diseño, implementación y validación de un medidor de canal portátil que puede ser utilizado de forma sencilla en aplicaciones industriales que requieran de movilidad. El medidor de canal portátil está basado en un esquema de sincronización inalámbrico que hace uso de los timestamps

mejorados propuestos en el capítulo 3. El uso de sincronización inalámbrica ofrece ventajas significativas si se compara con medidores de canal convencionales, que suelen requerir de señales de sincronización externas. Por ejemplo, el medidor de canal es más sencillo de usar, es mucho más compacto, más barato, y, debido a que su diseño está basado en SDR, puede ser fácilmente adaptado para la banda o escenario deseado. El medidor de canal ha sido desarrollado en dos piezas. La primera pieza ha sido desarrollada sobre un modem WiFi implementado en una plataforma SDR, que realiza la captura de los datos del canal, y la segunda pieza ha sido desarrollada sobre MATLAB. El medidor de canal ha sido validado utilizando un emulador de canal programado con canales con variación temporal y multicamino, y en condiciones industriales en el taller de Ikerlan. Estos experimentos han demostrado que el medidor de canal puede ser utilizado de forma efectiva para medir canales complejos en escenarios móviles.

Por último, el capítulo 5 muestra el diseño, implementación, y validación de la tecnología w-SHARP. w-SHARP es una tecnología inalámbrica que mejora WiFi para conseguir menor latencia, determinismo y mayor eficiencia en aplicaciones industriales. Sus aspectos fundamentales son: una capa física de bajo overhead, una MAC TDMA persistente, y el uso de sincronización de reloj de alta precisión. El diseño de w-SHARP, de acuerdo con el análisis teórico realizado, mejora las capacidades de los sistemas más avanzados encontrados en el estado del arte (5G y WiFi) para comunicaciones inalámbricas industriales. Dichos resultados han sido confirmados mediante los experimentos realizados utilizando un prototipo de w-SHARP implementado en una plataforma SDR basada en SoC-FPGA.

Publications

Publications in international journals

[1] Z. Fernandez, **Ó. Seijo**, M. Mendicute, and I. Val, "Analysis and evaluation of a wired/wireless hybrid architecture for distributed control systems with mobility requirements," *IEEE Access*, vol. 7, pp. 95915–95931, 2019.

[2] **Ó. Seijo**, J. A. López-fernández, H.-P. Bernhard, and I. Val, "Enhanced Timestamping Method for Sub-Nanosecond Time Synchronization in IEEE 802.11 over WLAN Standard Conditions," *IEEE Transactions on Industrial Informatics*, January 2020.

[3] **Ó. Seijo**, J. A. López-fernández, and I. Val, "w-SHARP: Implementation of a High-Performance Wireless Time-Sensitive Network for Low Latency and Ultra-Low Cycle Time Industrial Applications," *IEEE Transactions on Industrial Informatics*, June 2020.

[4] **Ó. Seijo**, J. A. López-fernández, and I. Val. "Portable Full Channel Sounder for industrial wireless applications with mobility by Using Wireless Sub-Nanosecond Time Synchronization," *IEEE Access,* September 2020.

[5] J. Montalbán, E. Iradier, P. Angueira, **Ó. Seijo**, and I. Val "NOMA-based 802.11 for Industrial Automation," *IEEE Access*, September 2020.

Publications in international conferences

[1] **Ó. Seijo**, C. Cruces, R. Torrego, I. Val, and J. A. López-fernández, "IEEE 1588 Hardware-Based Timestamp Implementation over 802.11a/g for IWSAN with High Precision Synchronization Requirements," *IEEE Int. Symp. Precis. Clock Synchronization Meas. Control. Commun. ISPCS*, 2017.

[2] **Ó. Seijo**, Z. Fernández, I. Val, and J. A. López-Fernández, "SHARP : A Novel Hybrid Architecture for Industrial Wireless Sensor and Actuator Networks," in *IEEE International Workshop on Factory Communication Systems (WFCS)*, 2018.

[3] Z. Fernández, **Ó. Seijo**, I. Val, and M. Mendicute, "Soft-Handover Algorithm for Hybrid Industrial Wireless Sensor and Actuator Networks," *IEEE Int. Conf. Emerg. Technol. Fact. Autom. ETFA*, 2018.

[4] **Ó. Seijo**, I. Val, J. A. López-fernández, and M. Vélez, "IEEE 1588 Clock Synchronization Performance over Time-Varying Wireless Channels," *IEEE Int. Symp. Precis. Clock Synchronization Meas. Control. Commun. ISPCS*, 2018.

[5] **Ó. Seijo**, Z. Fernández, I. Val, and J. A. Lopez-Fernandez, "SHARP: Towards the Integration of Time-Sensitive Communications in Legacy LAN/WLAN," *2018 IEEE Globecom Work. GC Wkshps 2018 - Proc.*, 2018.

[6] **Ó. Seijo**, I. Val, and J. A. López, "On the use of White Rabbit for Precise Time Transfer in 5G URLLC Networks for Factory Automation Applications," *IEEE Int. Conf. Ind. Cyber-Physical Syst.*, 2019.

[7] E. Iradier, J. Montalban, L. Fanari, P. Angueira, **Ó. Seijo**, and I. Val, "NOMA-based 802.11n for Broadcasting Multimedia Content in Factory Automation Environments," *IEEE Int. Conf. Emerg. Technol. Fact. Autom. ETFA*, 2019.

[8] **Ó. Seijo**, I. Val, and J. A. López, “Portable Full Channel Sounder for Mobile Robotics by using Sub-Nanosecond Time Synchronization over Wireless,” *IEEE International Workshop on Factory Communication Systems (WFCS)*, 2020.

[9] L. Fanari, E. Iradier, J. Montalban, P. Angueira, **Ó. Seijo**, and I. Val, "PEG-LDPC Coding for Critical Communications in Factory Automation," *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2020.

Declaration of Originality

I hereby declare that the research recorded in this thesis and the thesis itself were developed entirely by myself at Communication Systems Team, HW and Communication Systems Area, Ikerlan Technology Research Centre.

Óscar Seijo Gómez
Communication Systems Team
Ikerlan Technology Research Centre
May 2020

Contents

List of Tables

List of Figures

List of Algorithms

Acronyms

3GPP	*3rd Generation Partnership Project*
5G-NR	*Fifth Generation New Radio*
ACK	*Acknowledgment*
ADC	*Analog-to-Digital Converter*
AGC	*Automatic Gain Control*
AP	*Access Point*
AXI	*Advanced eXtensible Interface*
AWGN	*Additive White Gaussian Noise*
BE	*Best-Effort / Best-Effort Period*
BECP	*Best-Effort Controlled Phase*
BPSK	*Binary Phase Shift Keying*
CCA	*Clear Channel Assessment*
CFO	*Carrier Frequency Offset*
COTS	*Commercial Off-The-Shelf*
CP	*Cyclic Prefix*
CSMA/CA	*Carrier Sense Multiple Access with Collision Avoidance*
CSS	*Chirp Spread Spectrum*
CTS	*Clear-To-Send*
DL	*Downlink*
EMBB	*Enhanced Mobile BroadBand*
FA	*Factory Automation*
FCS	*Frame Check Sequence*
FDD	*Frequency Division Duplex*
FFT	*Fast Fourier Transform*
FIFO	*First In, First Out*

FIR	*Finite Impulse Response*
FPGA	*Field Programmable Gate Array*
GPS	*Global Positioning System*
GNSS	*Global Navigation Satellite System*
HDL	*Hardware Description Language*
HW	*Hardware*
IFFT	*Inverse Fourier Transform*
IQ	*In-phase and Quadrature*
IT	*Information Technology*
LoS	*Line-of-Sight*
MAC	*Medium Access Layer*
MATLAB	*MATrix LABoratory*
MCS	*Modulation and Coding Scheme*
MIMO	*Multiple-Input Multiple-Output*
mMTC	*Massive Machine-Type Communications*
NAV	*Network Allocation Vector*
NLoS	*Non-Line-of-Sight*
OFDM	*Orthogonal Frequency Division Multiplexing*
OFDMA	*Orthogonal Frequency Division Multiple Access*
OMNeT++	*Objective Modular Network Testbed in C++*
OT	*Operational technology*
PAPR	*Peak to Average Power Ratio*
PDP	*Power Delay Profile*
PER	*Packet Error Rate*
PHY	*Physical*
PLC	*Programmable Logic Controller*
PN	*Pseudo-Noise*
PPS	*Pulse Per Second*

QPSK	*Quadrature Phase Shift Keying*
QAM	*Quadrature Amplitude Modulation*
RF	*Radio Frequency*
RMS	*Root Mean Square*
RT	*Real-Time*
RTx	*Retransmission*
Rx	*Receiver*
SHARP	*Synchronous and Hybrid Architecture for Real-time Performance*
SIFS	*Short Interframe Space*
SNR	*Signal-to-Noise Ratio*
STR	*Symbol Timing Recovery*
SW	*Software*
TDD	*Time Division Duplexing*
TDMA	*Time Division Multiple Access*
TCXO	*Temperature Controlled Crystal Oscillator*
TSN	*Time-Sensitive Networking*
Tx	*Transmitter*
UAV	*Unmanned Aerial Vehicle*
UL	*Uplink*
USRP	*Universal Software Radio Peripheral*
VHDL	*Very High-Speed Integrated Circuit Hardware Description Language*
w-SHARP	*wireless SHARP*
WirelessHP	*Wireless High-Performance*
WPA2	*Wi-Fi Protected Access 2*

Notation

$\cdot$	*Scalar product*
$*$	*Convolution*
$\star$	*cross-correlation*
$\lvert \cdot \rvert$	*Absolute value*
$\lfloor \cdot \rfloor$	*Floor*
$\lceil \cdot \rceil$	*Ceil*
f	*Frequency*
f_c	*Carrier frequency*
t	*time*
T	*Sampling period*
δ	*Dirac delta*
$\delta_{m,n}$	*Kronecker Delta*
$\mathcal{N}$	*Gaussian distribution*
$R_{r,s}$	*Cross-correlation of signals r and s*
μ	*Mean*
σ	*Standard deviation*
$\bar{\tau}$	*Mean delay spread*
τ_{rms}	*RMS delay spread*

1 Introduction

1.1. Motivation

The tasks automatization in industrial processes driven by the industry 3.0. has brought a significant cost reduction in product manufacturing [1]. Now, the processes digitalization in the industry 4.0. era will produce an automatization level never seen before. Specifically, the industry 4.0. envisions industry facilities crowded with a huge number of heterogeneous interconnected devices, which can be automatically controlled, upgraded, and configured through a ubiquitous network [2]. This network revolution would drastically cut the maintenance, installation, and operation costs of industrial facilities. To achieve this vision, it is necessary to partially or totally replace the wired connectivity with wireless. The main advantages of wireless networks compared to wired ones are lower installation costs, higher scalability and flexibility, and free movement of the nodes.

Nonetheless, some significant challenges arise when building such an industrial network. The challenges of the field level, where the communication between sensors, actuators, and the controller takes place, are the most stringent ones. Among others, networks at the field level require predictable behavior, high reliability, bounded low latency, tight time synchronization, the possibility of handling numerous nodes, and highly-secured data transfer. Most of these challenges are solved in the wired domain thanks to the Time-Sensitive Networking (TSN) technology [3], a family of standards defined by the IEEE 802.1 task group that enables deterministic network operation over Ethernet.

However, from the wireless side, several challenges are still blocking the massive deployment of wireless connectivity on the factory floor [4]. These challenges are derived from two important aspects in wireless communications, the unreliable and unpredictable behavior of the wireless propagation, and the historical design of wireless systems.

First, the prone-error nature of the wireless propagation, which is especially noticeable in industrial scenarios, is confronted with the reliability required by most industrial applications. In addition, the shared medium can produce significant interferences between different wireless systems that further compromise the reliability. Second, wireless systems have been widely designed for the consumer market, which is characterized by long packets with little requirements in terms of low latency, and per-packet reliability. On the contrary, industrial applications are characterized by short (few bytes), and with strict latency and per-packet reliability constraints. Consequently, most consumer market wireless systems are not prepared to be deployed in industrial scenarios.

The main objective of this thesis is the design of a high-performance wireless technology that fulfills requirements found in nowadays and future industrial applications. In this sense, main three topics are covered in this thesis: the characterization of the industrial propagation phenomena, the issues of high-performance time synchronization over wireless as the main enabler of industrial wireless networks, and the design, Hardware (HW) implementation, and validation in an industrial scenario of an original wireless technology which aims to partially or totally fulfill the requirements of industrial communications at the field level.

The work related to channel characterization in industrial scenarios continues with previous research carried out by the Communications Systems team of Ikerlan in the context of the Roll2Rail EU project [5] [6]. The particular objective, in this case, was to simplify the industrial channel characterization to enable measurements in complex scenarios.

In the topic of wireless time synchronization, the previous research at Ikerlan was focused on narrowband radios combined with Software (SW) timestamping

[7] and [8]. In this case, the goal was to improve the performance of the time synchronization schemes by moving from a narrowband to a wideband wireless system and by enhancing the timestamping quality.

In the context of the design and implementation of a high-performance wireless technology, the starting point is the work presented by Zaloa Fernández [9], Pedro M. Rodríguez [10], and Raúl Torrego [11] in their respective theses, the wideband Orthogonal Frequency Division Multiplexing (OFDM) modem (802.11-like) developed by the industrial communications team over an SDR platform [12], and the demonstrator of the wired/wireless TSN technology presented in [7]. Two objectives are pursued in this topic: enhance the performance of the designs proposed in [9] and [10] by using a novel high-performance PHY layer, and the development and validation of the new technology based on Ikerlan's wideband OFDM modem.

1.2. Background and State of the art

The introduction of industrial automation in the 1920s-1930s brought an enormous revolution to the manufacturing processes [13] and a dramatic reduction in the production of material goods. In the very first steps of the industrial automation, the control systems were implemented using simple analog circuitry that comprises the connection to a few sensors and actuators. As a consequence, their operation was rather limited, and they typically relied on humans that continuously monitor the behavior of the control system [13].

With the advent of microcontrollers and, in general, programmable HW, the connectivity of the sensors and actuators of the control system (named as edge devices in this thesis) was gradually replaced from purely analog control to mixed digital-analog. The digitalization of the controllers triggered a huge revolution in industrial automatization: the controllers could be replaced from analog costly elements that can only perform simple tasks, to elements that were able to perform complex tasks and with great flexibility. The digital-physical system paradigm was the precursor of the nowadays so-called Cyber-Physical Systems (CPS).

The next step was to extend the digital domain towards the edge devices, replacing, as far as possible, the analog elements with digital elements. Thus, the data gathered by the edge devices could be directly digitized in the edges, removing the issues of analog connections, such as noise and cable attenuation. Consequently, the analog wired connections were replaced with communication systems that transported the discrete version of the analog signal from the edges to the core of the control system. Initially, dedicated wired connections between the edge devices and the core of the control system were still used. However, dedicated connections entail large deployment costs due to the length of the wires. Then, the concept of Networked Control System (NCS) emerged and took advantage over direct connections [14]. In NCS, the control system is built over a packet-switched network that can transmit several data streams with diverse traffic profiles in a single wire. However, packet-switched networks designed for Information Technology (IT) (e.g. Ethernet) did not cope with the requirements of industrial applications: high reliability, deterministic communication, latency, and synchronization among the nodes. Widely used technologies emerged in this step to fulfill these requirements, such as IEEE 1588 [15], widely known as Precision Time Protocol (PTP) to provide a global time base enabling Real-Time (RT) operation, and RT networks, such as EtherCat [16] to provide known bounded latency. Finally, TSN [3] is rapidly emerging in the last years as the technology that will integrate into a single network the industrial applications and IT applications.

Even though the number of required wired connections in industrial networks has tremendously decreased due to data multiplexing, the connection between the edge devices and the network, the so-called last-mile connection still entails significant deployment costs and complexity [4]. In essence, industrial control systems, from its analog version, to its fully digital version based on RT wired networks, are still constrained due to the wired connectivity from the edges to the core.

In the last decade, with the emergence of high-performance wireless systems (e.g. Wi-Fi, LTE), and, in the imminent future, 5G [2], the last mile of the IT communications has been progressively replaced with wireless, because of its flexibility, simplicity, scalability, lower costs, and free mobility of the wireless

interconnected nodes. As in IT applications, wirelessly interconnected edge devices in industrial applications have several advantages, such as flexibility, scalability, and lower commissioning costs, and opens a new range of use cases that may require mobility, such as collaborative robotics [17]. Thus, wireless technologies are being considered nowadays as the next step to reach new automatization dimensions as never seen before [2].

However standard wireless systems designed for IT are not, in general, trustworthy/ do not offer enough performance to fulfill the challenges of nowadays and future industrial applications [18]. The reasons are diverse. First, wireless links have some inherent limitations, such as limited radio resources, possible collisions due to the coexistence of several wireless networks in the same band, and reduced reliability due to the wireless channel propagation phenomena [19]. The latter is particularly critical since industrial environments are characterized by harsh propagation conditions, due to the presence of metallic elements that produce strong multipath propagation and high-power machines that generate interferences [20]. Second, historically, wireless systems have been broadly designed for IT because there was not a clear interest in pushing wireless communications in industrial environments, and, as a consequence, they do not include the set of technologies required to support industrial applications and they are not efficient for industrial applications.

In summary, the use of wireless connectivity in the last mile connection of industrial networks is a critical step in the forthcoming industrial revolution. However, several implementations and research gaps must be solved before the massive adoption of wireless connectivity in industrial applications. The next three subsections overview the state-of-the-art in different topics, namely the current state of industrial wireless technologies, the approaches of time synchronization in industrial networks, and the propagation environment in industrial applications.

1.2.1. Industrial wireless technologies

As in the wired systems, where some IT technologies (e.g. Ethernet) were adapted to create industrial technologies (e.g. EtherCat [21], Profinet [22],

SERCOS III [23]), standard wireless systems have been also customized to implement industrial wireless networks. For instance, some technologies like WSAN [24], or wirelessHART [25] combined narrowband standard PHYs (802.15.1 [26] or 802.15.4 [27]) with custom Medium Access Control (MAC) layers to provide similar behavior as the industrial wired solutions, though with a lower data rate. Other industrial wireless solutions, such as WIA-Factory Automation (FA) [28] went one step further and used similar MAC designs but over wideband PHYs, such as 802.11n [29], providing higher data rate, lower latency, and improved reliability. Even so, the 802.11 PHYs are designed for IT applications and they are not efficient in industrial applications [30], which are typically characterized by short packets. Consequently, their performance is still far from the provided by the advanced industrial wired network technologies [18].

In the last few years, the research trend around wireless TSN and the industry 4.0. has boosted the standardization of wireless systems for industrial applications. wireless TSN refers to a wireless system that offers similar features if compared to TSN over wires. For instance, the 3rd Generation Partnership Project (3GPP) [31] has included in the fifth Generation New Radio (5G-NR) the Ultra-Reliable and Low Latency Communications (URLLC) profile [32], which is designed to provide sub-ms latencies and Packet Error Rate (PER) below 10^{-5}. However, despite that the 5G-NR performance is closer to wired industrial networks than other industrial wireless solutions [33], it is still out of the requirements of several use cases [34].

The last 802.11 standard, 802.11ax [35], is also positioning as a suitable candidate for wireless TSN. 802.11ax includes a new PHY that is more efficient for industrial applications [36] and new MAC operation modes that enables non-persistent scheduling of [4]. Also, the next 802.11 standard, 802.11be will provide a higher data rate than 802.11ax [37] and an operation mode to guarantee worst-case latencies [4].

In the case that Commercial-Off-The-Shelf (COTS) solutions are not suitable for the application, hybrid custom-standard or totally custom proprietary solutions can be considered. On the one hand, hybrid custom-standard solutions are based on COTS PHY layers and they implement custom upper layers (mainly

MAC) with the required features, such as determinism, network synchronization, retransmissions, etc. For instance, in [8] some researchers from Ikerlan presented a solution based on a COTS radio chip with a custom Time Division Multiple Access (TDMA) MAC layer designed for a specific railway application. On the other hand, totally custom solutions use a full custom communication stack from PHY to the application [38]. Despite that they present large engineering efforts due to the complexity of the design, implementation, and validation of a full communication stack, these solutions are being considered in recent years because they can be perfectly tailored to the specific industrial application providing much higher performance than general solutions [18].

1.2.2. Challenges in time synchronization

One of the main requirements to enable the correct operation of industrial networks and applications is to synchronize the local time of all the nodes in the network to a common time base. The required synchronization performance is derived from two sources: the communications system design (either wired/wireless) and the applications built on top of the industrial network. In the first place, industrial networks require high-performance time synchronization to minimize the scheduling latency between different elements of the network and to reduce the gaps between frames to improve the overall system efficiency. In the second place, most applications in the industrial domains and some applications in other domains such as economics and scientific experiments have strict time synchronization requirements, because every element of the system must synchronously perform specific tasks, such as reading a sensor or changing the state of an actuator [19]. One single failure in the time synchronization can lead to the failure of the entire system, which, depending on its severity, can cause from loss of profits to loss of human lives. At this moment, two main approaches are considered to perform time synchronization, Global Navigation System (GNSS)-based synchronization, and protocol-based synchronization.

In GNSS-based time synchronization, the nodes have a GNSS receiver that is used to synchronize their local time and the overall network and applications operation. GNSS-based time synchronization can provide a precision in the tens

of nanoseconds range or even below which well suits the requirements of industrial applications [39]. Consequently, this solution is widely used nowadays in different industrial domains, such as synchronization among electrical substations. Still, GNSS-based synchronizations have some drawbacks [39]. Among others: the network relies on an external network that could be potentially insecure, the network should be placed outdoor to ensure good reception quality, or at least their antennas, and its performance depends on the climate conditions in the deployment scenario.

In protocol-based time synchronization, the nodes exchange through the network several time synchronization messages that carry timing information (timestamps) [15]. The messages and their timestamps are used to evaluate the error between the node that serves the time (master), and the nodes that synchronize their time (slaves). The protocol-based time synchronization is currently dominated by two protocols: The Network Time Protocol (NTP) [40], and the Precision Time Protocol (PTP) [15]. NTP is widely used in IT applications and its precision target is in the millisecond range. Consequently, it does not suit industrial applications. On the contrary, PTP is designed to cover the lower precision range, up to a few ns. However, the performance of PTP greatly depends on the implementation of the protocol, mainly on the layer of the nodes that the timestamps of the timing messages are taken [41]. SW timestamps provide precision in the µs range [42], whereas HW timestamps reach the tens of nanoseconds [43].

The state of protocol-based time synchronization differs in wired and wireless networks. On the wired domain, industrial communication systems seamlessly support HW timestamps and PTP and hence the precision required by industrial applications is totally guaranteed [22]. On the wireless side, most standard wireless systems do not require/specify high-performance time synchronization support and, as a result, most implementations do not include HW timestamps either PTP support. Consequently, at this moment, the only realistic way to reach the nanosecond range using standard wireless systems is the use of GNSS-based synchronization. The main reason is again historical: the industry has not been interested in the use of wireless for industrial applications, which is the main niche that requires this level of performance.

In the last few years, the research in protocol-based time synchronization over wireless has produced significant outcomes over a large variety of wireless technologies [41] using different messaging protocols and timestamping techniques. For instance, A. Mahmood et al. describe the use of PTP combined with SW timestamping over 802.11 [44], whereas R. Exel uses a similar setup in [45] but using HW timestamps instead of SW timestamps. Other authors have considered other wireless standards and different timestamping techniques, such as [46], using an ultra-wideband radio and HW timestamps, or some researches from Ikerlan that, in [8], used a narrowband radio and SW timestamps. These implementations resulted in different precisions/implementation complexity upon the timestamping technique, protocol, and wireless system. Finally, the 802.11 standard has included a specific function for synchronization over the 802.11 MAC named fine timing measurement [47], which can be adopted by 802.1as (PTP). However, its synchronization performance is yet to be determined.

1.2.3. Wireless propagation

Typically, wireless systems are affected by the common propagation phenomena in Radio Frequency (RF) transmissions. These phenomena include attenuation, distortion, reflections, and scattering of the signals transmitted by the wireless nodes. As a result, a signal that traverses through a wireless channel is typically highly distorted and the properties of the wireless channel influence, to some extent, the overall wireless system design and limitations. Consequently, the modeling of the wireless propagation is a critical step in the design and implementation of any wireless system. In the case of industrial wireless communications, the channel modeling is even more critical, since industrial scenarios are characterized by their harshness and unpredictability due to the huge number of metal elements that reflect the radio signals [48].

Wireless channel modeling aims at the precise characterization of the wireless propagation phenomena in a specific scenario [49]. Two main approaches are commonly used: analytical models or stochastic models. The analytical models study the propagation phenomena using a geometrical approach or solving the Maxwell equations through approximations. These methods may provide a very

accurate characterization, but they require an extremely detailed physical model of the scenario, including materials, the precise shape of the objects up to cm, or even below depending on the operating frequency, etc. On the other hand, stochastic approaches provide a general model for a specific scenario, but they do not contain the information of the scenario itself. Generally, these models are less accurate but more practical, as they are easier to develop, they can be used to evaluate similar scenarios and they do not require intensive simulations [50].

The stochastic models represent the wireless propagation phenomena for a specific frequency band and generic scenarios [51], such as small office, large office, open space, small industrial, large industrial, etc. A stochastic model is developed from one or several measurement campaigns in scenarios with similar shapes. Unfortunately, industrial wireless channels present huge differences from one scenario to another, and thus the development of a general channel model is not straightforward. In addition, channel measurement campaigns in industrial scenarios are quite complex [52] and, as a result, their availability in the literature is limited. Note that industrial scenarios include small workshops, big refineries, automotive factories, Unmanned Aerial Vehicles (UAV) communications, railway, etc. As evidenced, the variability of these scenarios is enormous.

Some researchers from the Communications Systems team of Ikerlan in collaboration with the Universidad Politécnica de Madrid presented in [6] a measurement campaign for railway communications in Metro Madrid. Another example of a channel measurement campaign in industrial scenarios is the one performed by the National Institute of Standards (NIST) [53]. Finally, R. Croonenbroeck et al. presented in [54] their results of a measurement campaign performed in a mechanical workshop. Given that the availability of industrial wireless channel models in the literature is limited, a suitable option to characterize a specific industrial scenario is to directly perform a channel measurement campaign [55]. However, and as noted before, carrying out a channel measurement campaign in an industrial scenario is typically quite complex.

In addition, and as a complement to wireless channel measurement campaigns, the use of wireless channel emulation is very convenient when

targeting the design of wireless systems for industrial applications [56]. A wireless channel emulator is a device that can be programmed with a statistical channel model and that will emulate the behavior of the wireless channel. Hence, the wireless channel model can be programmed to emulate the behavior of the wireless channel measured in the industrial scenario, enabling the validation of a wireless system over a realistic wireless channel without leaving the lab. The advantage of a wireless channel emulator is the significant reduction of the commissioning costs as the wireless device is tested against the real wireless channel prior to its deployment.

1.3. Contributions

In this section, the main contributions of the developed research work are described along with the corresponding publications of each contribution.

In the topic of wireless time synchronization, it is worth noting the design of two timestamping methods, one conventional method based on the assumption of perfect wireless conditions [57], and an evolution of the former, the enhanced timestamping method, that provides much higher performance over non-ideal wireless conditions [58]. The conventional timestamps have been implemented in an 802.11 modem built over the ADI RF SOM platform and the experiments demonstrate that the conventional timestamps are able to fulfill the requirements of most industrial networks. On the other hand, The enhanced timestamping method has been extensively evaluated through simulations [58] showing very promising results for any kind of wireless conditions. Additionally, the enhanced timestamping method has been also indirectly tested in the HW platform showing similar and promising results compared with the simulations.

Regarding the channel characterization in industrial environments, the main contribution is the design and implementation of a portable channel sounder that can be seamlessly used in industrial and other environments [55], [59]. Compared to a traditional channel sounder, the design presented in this thesis does not require any wired connection to synchronize the devices that conform the channel sounder devices, which provides improved flexibility and simpler operation. The

channel sounder core is a modification of PTP combined with the enhanced timestamping method [58]. The channel sounder has been validated over a wireless channel emulator and over real factory conditions showing that the proposed design is able to measure the wireless channel without the need of wired synchronization.

Finally, in the context of high-performance wireless networks for industrial applications, the main contribution is the design of a novel wireless protocol named wireless-SHARP (w-SHARP) specifically designed for industrial communications at the field level [19] [60]. A prototype of the w-SHARP protocol has been implemented in the ADI RF SOM platform and tested in the laboratory and in a real industrial environment [61]. The prototype has shown very promising results in terms of reliability, synchronization performance, and latencies, and even outperforming for industrial applications the most advanced nowadays wireless systems, such as 5G-NR or 802.11ax.

1.4. Technologies, tools, and hardware platforms used in this thesis

This thesis is mainly focused on communications for industrial applications and the research expected from an industrial thesis should be generally accompanied by the implementation of prototypes and, if possible, validation in real environments. Consequently, the state-of-the-art of different technologies, tools, and platforms that can be used to develop the proposed designs are very relevant to understand the scope and the research output of this thesis. The implementation efforts during the development of this thesis have been mostly focused on digital HW over SDR platforms and, thus, this section is divided into two subsections: Subsection 1.4.1 describes the nowadays existing platforms to implement wireless systems, and Subsection 1.4.2 overviews the existing approaches for HW development.

1.4.1. SDR platforms

From the implementation perspective, a wireless communication system is typically divided into roughly three main blocks, which are depicted in Figure 1.1: the RF HW, the baseband processor, and the SW. The RF HW includes the antennas, analog processing (mixers, filters, amplifiers, etc.), and Analog-to-Digital Converter (ADC)/Digital-To-Analog Converter(DACs). The baseband processor includes the PHY layer (coding, modulation, etc.), and the MAC. Finally, the SW implements two elements: the driver, which abstracts the Apps from the specific operation of the baseband processor, and the applications, that use the NIC to communicate with other devices that support the same protocol stack.

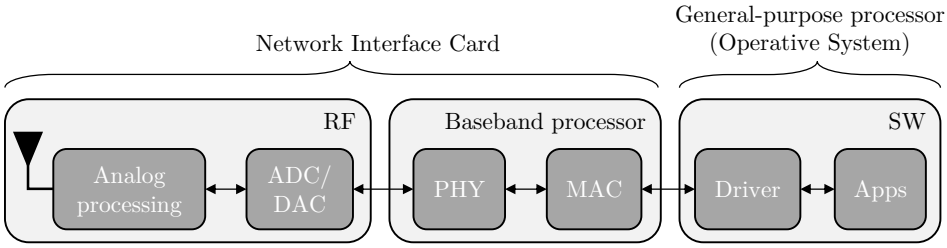


Figure 1.1. Three main elements of a communication system.

Depending on the objective of the implementation, the RF, baseband processor, and SW can be either found integrated into a single chip or separately. For instance, in the case of computers, the RF and baseband processor are typically integrated into a Network Interface Card (NIC), which is connected through a standard electrical interface to the general-purpose processor. As another example, smartphones commonly integrate the general-purpose processor, baseband processor, and most of the RF circuitry in a single chip. In both cases, the HW of the RF and baseband processor is implemented in an Integrated Circuit and it has little programmability capabilities. Consequently, they cannot be used as a research platform in industrial communications focused on new PHY and MAC designs.

With the advent of Software Defined Radio (SDR) platforms in the last decade [62], the design, implementation, and testing of custom PHY/MAC layers

became a reality. SDR platforms allow the complete programmability of the PHY/MAC layers. According to their purpose, two kinds of SDR platforms can be distinguished, SW-based SDR platforms and System on Chip (SoC)-Field Programmable Gate Array (FPGA)-based SDR platforms. On the one hand, SW-based SDR platforms include an RF module whose ADC/DAC can be directly connected to a computer [63]. The RF module can be programmed with different BW, filters, and carrier frequency. The computer receives the discretized RF signal from the SDR platform and runs the baseband processor. Unfortunately, even with the processing speed of modern computers, the baseband processing in a general-purpose processor is rather slow and hence these platforms typically suffer from low throughput and latencies in the order of several milliseconds [20]. Consequently, the implementation of an industrial wireless system in SW-based SDR in most cases serves as a proof of concept rather than a full implementation.

On the other hand, SoC-FPGA-based SDR platforms contain a reprogrammable RF module (as SW-based SDR), reprogrammable HW (FPGA), and one or several general-purpose processors [64] [4]. The reprogrammable HW allows the researcher to implement a dedicated baseband processor that will directly run on the HW, as in ICs. In addition, the general-purpose processor is also embedded in the SDR platform that can run specific SW, such as an RT OS, or Linux for embedded systems. SoC-FPGA-based development provides closer performance to common wireless NICS though, unfortunately, it entails great implementation complexity and thus suffers from high engineering costs. In fact, there exist very few FPGA-based SDR implementations of wireless systems in the state-of-the-art. For instance, WARP [65] (802.11b/g/n), Tick [66], or an 802.11ax modem developed over an Intel Arria [4].

The real experiments performed in this thesis has been carried out over the ADI RF SOM SoC-FPGA-based SDR platform, also named ADRV9361-Z7035 SDR [67]. This platform includes a reprogrammable radio chip and a Zynq SoC that integrates programmable logic (FPGA) and programmable SW (a dual-core ARM microcontroller) in a single chip. This platform has been used to implement a fully compliant 802.11 modem (see Appendix A) and to implement the w-

SHARP modem (see Chapter 5). The experiments of chapters 3 and 4 were carried out using a modified version of the 802.11 modem.

1.4.2. Digital hardware development

The journey of the digital HW development started with the creation of the first in the Bell laboratories in 1947 [68]. Only around ten years later, the first Integrated Circuit (IC) with several transistors in a single chip was created [68]. Soon, the number of transistors within an IC has tremendously grown from a few transistors to hundreds or thousands of them. Naturally, the engineering efforts and the probability of errors in the design grown with the number of transistors of the IC, and given the long manufacturing time of an IC, its debugging could take a long time. Consequently, the need for automated tools to ease the implementation of ICs became an urgent reality [69]. Several decades have passed since the creation of the first IC until today, and nowadays there exist several abstraction layers that greatly simplify the IC development. Figure 1.2 depicts these abstraction levels accompanied by their emerging year, from the lowest level (transistor) to the High-Level Synthesis (HLS) tools [69]. Naturally, the same abstraction philosophy applies to HW design over programmable HW (FPGAs), which are the main development platform used during this thesis. In the context of 2020 and from the viewpoint of HW design over FPGAs, the Register Transfer Level (RTL), and the High-Level Synthesis (HLS) tools are probably the most relevant abstraction layers.

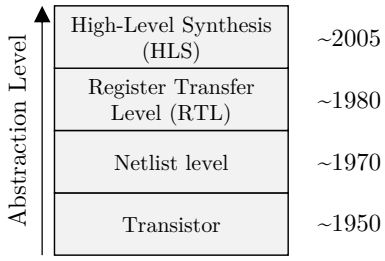


Figure 1.2. Abstraction levels in IC development.

The RTL layer is characterized by the use of Hardware Description Languages (HDL), such as Very High Speed Integrated Circuit HDL (VHDL) or

Verilog. An RTL design describes the interactions between combinatory logic and registers, and it is independent of the subjacent technology used to implement the circuit. In FPGA development, automated tools such as Vivado [70] are typically used to move from an HDL design to a final HW file used to program the FPGA. Typically, two steps are performed: the synthesis and the place and route. The synthesis translates the RTL into a netlist design, which describes the logic interconnection of the HW elements (technology dependent). Then, the elements of the netlist design are placed and routed into a specific FPGA and the final HW file is generated. HDLs significantly reduce the development costs and improves the portability of a HW design from one HW platform to another platform. Even so, the implementation effort of HW using HDLs is significant and thus it is unfeasible to build complex HW mainly for research purposes.

High-Level Synthesis (HLS) tools were born in the mid-2000 as an additional abstraction layer to HDLs. Nonetheless, they were rather limited in terms of features and they provide significantly less area efficiency than pure HDL programming [69]. Consequently, their adoption was rather timid compared to the adoption of HDL in the mid-1980s. Now, Newest HLS tools feature a user-friendly interface, behavioral-equivalent models to reduce the simulation time, integrated automated verification tools, and seamless modification of the clock frequency, signals data types including floating-point operations, and propagation of the data types through the design. Then, the HLS design can be directly compiled and translated to HDL, which in turn is used to generate the final HW design.

Given the advantages of HLS tools compared to RTL programming, the digital HW development in the context of this thesis has been carried out using HLS tools. Specifically, System generator by Xilinx [71] has been used to implement the w-SHARP protocol (Chapter 5), and the 802.11 modem used in chapters 3 and 4 (see Appendix A).

1.5. Structure of the thesis

This thesis is divided into six chapters and one appendix. This chapter includes the motivation of the research, with a review of the scope and state-of-the-art, and a conceptual overview of the technologies used during the thesis. Additionally, the chapter also includes the main contributions of the thesis. The appendix describes the implementation of the 802.11 modem.

Chapter 2 describes the basic concepts of industrial control systems and their current implementation approach using industrial networks, including the most relevant requirements of industrial applications from the communication perspective. The chapter ends with a brief review of different use cases and application requirements.

In Chapter 3, two wireless time synchronization schemes are described and evaluated through simulation and experimental means. The chapter starts with a detailed review of the current state of research in wireless time synchronization. Then, the proposed schemes are detailed, and some guidelines to integrate the schemes into different wireless standards are stated, including a detailed view of how to modify a Wi-Fi modem to support the schemes. Finally, the schemes are evaluated over simulation and experimental means over different wireless propagation conditions.

Chapter 4 presents the design and implementation of a channel sounder for industrial channel characterization. The chapter starts with the description of the design philosophies of channel sounders and the discussion of their limitations in terms of performance, flexibility, and usability. Later, the design of the channel sounder and its implementation over 802.11 using an SDR platform is presented. The chapter ends with two representative experiments to validate the proposed design.

Chapter 5 is focused on the SHARP architecture, and more specifically, in the design and development of the w-SHARP protocol. The chapter starts with an overview of the SHARP architecture, which includes a brief description of TSN and the w-SHARP protocol. Then, the w-SHARP specification is comprehensively detailed, and, based on the specification, a possible implementation architecture of the protocol is described. Finally, the chapter

includes some meaningful experiments that highlight the capabilities of the proposed protocol.

Lastly, Chapter 6 summarizes the conclusions of this thesis and outlines the possible future research lines to continue the research developed during the thesis.

2
Communications Networks in Industrial Applications

Nowadays, the communications networks are a fundamental element in the implementation of control systems and the performance of the network limits, to some extent, the overall control system performance. This chapter overviews the use of communications networks in the industry, from a general view of the communications within a factory to a detailed description of the challenges and issues of communications at the field level.

2.1. Hierarchy of communications in industrial automation

The communications in a factory are classified upon their position in the so-called automation pyramid, which is depicted in Figure 2.1. The initial design of the pyramid presented 5 levels: management, planning, supervisory, control, and field level. As in the evolution from the 7-layer ISO/OSI stack to the actual internet stack, the pyramid was simplified to three levels: the enterprise level, characterized by the Enterprise Resource Planning (ERP) SW, the cell and control level, and the field level [72]. Each level of the pyramid has different purposes within an enterprise, and, in consequence, they have different communication requirements.

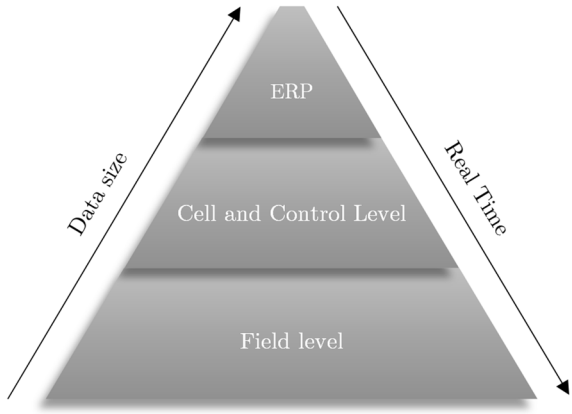


Figure 2.1. The three-level automation pyramid, adapted from [72].

The *enterprise level* is the highest level of the pyramid and it is responsible for the general management of the company by using the ERP. The ERP includes all the general information about the whole company chain, including human resources, logistics, etc. The enterprise level handles gigabytes of data and has latency constraints that can range from days to months. Based on its requirements, standard Information Technology (IT) communications (Ethernet, Wi-Fi) typically suits the requirements of the enterprise layer.

The *cell and control level* is in charge of the control, supervisory, and management of the physical processes involved in the factory. The supervisory and management monitors the data through Human to Machine Interfaces (HMI). The control is performed by the Programmable Logic Controllers (PLC) which are the interface between this level and the field level. The cell and control levels are closer to the physical processes, and thus this level requires faster reaction times, which is translated into lower communication latency.

Finally, the *field level* is directly involved with the physical processes of the factory. It comprises the sensors that measure the variables of the physical process and the actuators used to modify the behavior of the physical process. The data generated at this level is exchanged between the PLC and the sensors/actuators. Being purely a machine to machine communication, the communications at the field level present the most stringent time requirements

of the pyramid. The traffic is characterized by small amounts of data (few bytes) that is periodically generated and that must be transported by the network with as low latency as possible. The field and the cell and control levels are built with the so-called Operational Technology (OT), which includes all the SW/HW that enables the correct operation of the physical process of the factory. The topics tackled in this thesis are specifically related to the communications at this, which presents the tightest communication requirements.

2.2. Communications at the field level: Networked Control Systems

This section briefly describes the basic principles of the design of control systems, introduces the concept of control systems based on a communications network, or Networked Control Systems (NCS), and highlights the particularities of industrial communications networks as opposed to typical IT networks.

2.2.1. Principles of control systems

A control system is a conjunction of elements whose objective is to provide a specific response. It comprises four main elements [14]: a variable to be controlled,

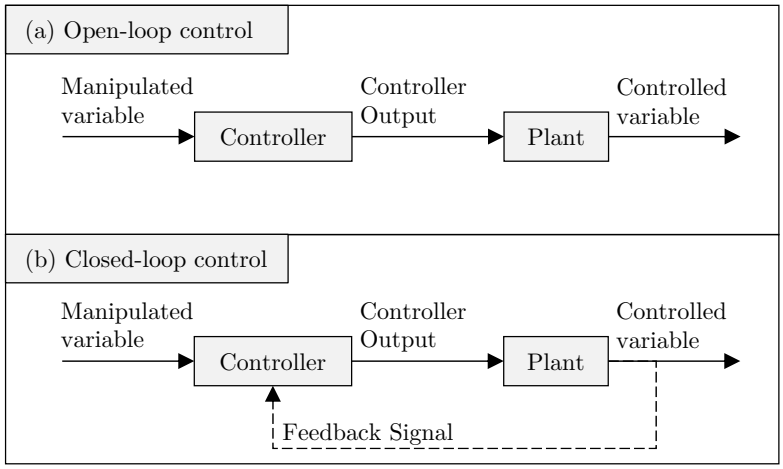


Figure 2.2. Basic representation of an open-loop System (a) and a closed-loop system (b) [14].

named controlled variable, the desired value of the controlled variable, named manipulated variable, A physical process with determined properties named plant, whose output is the controlled variable, and the controller, that acts over the plant to match its value to the value of the manipulated variable.

Two main types of control systems are typically distinguished according to the way the controller is implemented: open-loop control and closed-loop control (Figure 2.2). In open-loop control, the controller does not know the state of the controlled variable and thus its control is applied only based on the value of the manipulated variable and its past values. In closed-loop control, the controlled variable or other variables related to the plant process are also an input of the controller. Consequently, the output of the controller is not only determined by known inputs, but also by the actual value of the controlled variable, usually named feedback signal. From the signal's theory point of view [73] an open-loop control system is a time-invariant system (i.e. its output is deterministically determined by the input of the control system), and a closed-loop control system is a time-variant system since its behavior can vary over time upon external uncontrollable effects, such as variations in the feedback signal. In general, closed-loop control can compensate for the unpredictable variations of the controlled variable due to external effects and hence they offer superior performance compared to open-loop control.

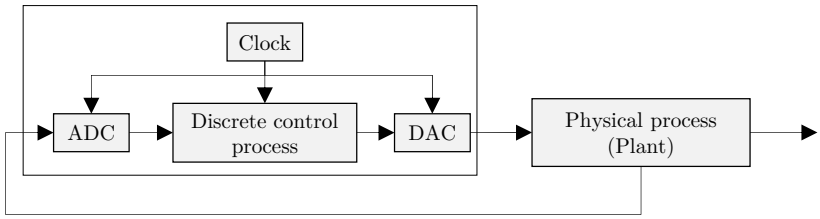


Figure 2.3. Controller implemented over the discrete domain [14].

Both open and closed-loop control systems were initially designed and developed in the continuous domain using analog circuitry. However, this approach was inflexible from a design viewpoint and now most control systems are implemented in the discrete domain using digital processors build over programmable multipurpose digital circuitry (Figure 2.3). In discrete control, the

continuous-time signals used to sense or act over the physical process are now discretized using ADC and DACs. The use of discrete processing reveals new challenges in the control system design, such as synchronous operation between the different devices involved in the control system (DAC, DAC, and the discrete control process that implements the controller algorithm).

2.2.2. Local vs Networked control systems

From the communications' point of view, the implementation of control systems can be classified into two classes: local control systems, and Networked Control Systems (NCS). Local control systems have all the digital logic integrated into one element (the controller) and the sensors and actuators are directly connected by point-to-point dedicated analog wires to the controller. This approach requires little engineering efforts from the communications perspective since point-to-point wires have a negligible impact on the control system performance. However, this approach suffers from low flexibility and interoperability. For instance, the full commissioning of the control system or a partial reconfiguration may require a complex wiring/re-wiring of the connections between sensors/actuators to the controller [74].

In Networked Control Systems (NCS), the sensors, actuators, and controllers are distributed and interconnected through the same network (Figure 2.4). In NCS, each element of the control system includes microcontrollers, a network interface, etc. In essential, the sensors and actuators within an NCS are complex modular devices rather than simple elements that measure or receive an analog signal. The distribution of the processing logic of the control systems introduces several challenges in their design, such as the need for precise synchronization, network latency control, and optimization, ensure enough reliability, etc. Despite the higher engineering complexity derived by these challenges, NCS has several advantages that result in more cost-effective solutions if compared to traditional local control systems [74]: the devices involved in the control system are designed to be interoperable, to support different network architectures, they can be deployed over longer distances, and they can be seamlessly reconfigured and supervised through the network.

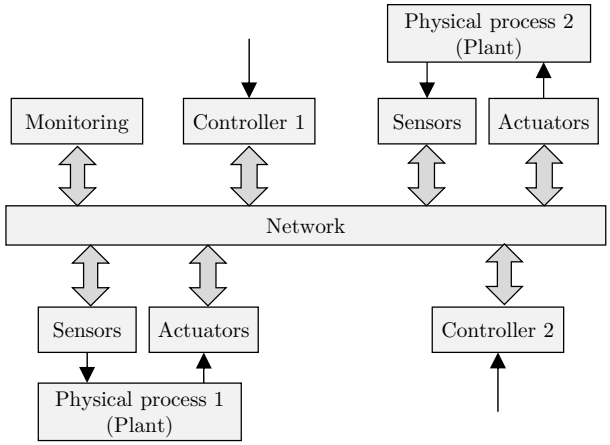


Figure 2.4. Networked Control System.

The introduction of a network between the devices of the control system has a significant impact on the latency of the control system and, naturally, in the control system performance, especially if compared to traditional local control systems [14]. Consequently, the network itself is considered a fundamental element of the control system design and implementation, i.e. the applications running over the control system are developed based on the network properties, and networks operate taking into account the requirements of the applications running on it.

2.2.3. Networks Particularities in Networked Control systems

Industrial networks present three significant particularities if compared to the IT networks, which are mainly driven by the differences in the requirements between their target applications. These three particularities are the communication stack, the traffic model, and the cooperation between the layers of the stack.

IT vs industrial communications stack

Most IT applications require as much throughput as possible with little efforts in other aspects such as latency, whereas industrial applications are characterized by smaller throughput, but with strict latency and reliability. Consequently, the

traditional "weighty" TCP/IP (see Figure 2.5 (a)), which is designed for IT applications, does not suit industrial applications. A simpler one, based only on PHY, link, and application layer (Figure 2.5 (b)), is commonly adopted to reduce the processing/network overhead thus reducing the overall network latency. In addition, the PHY and link technologies adopted in industrial networks typically differ from traditional ones, though the RT variants of Ethernet are widely used nowadays [2]. In recent years, and with the advent of the TSN paradigm, the adoption of a mixed stack that supports both IT and industrial applications is gaining popularity as a way to simplify the network design of a factory. TSN works between the IT PHY/MAC technologies (e.g. 802.11, 802.3) and upper layers, and allows the coexistence of real-time and best-effort traffic in the same network. As in the adoption of NCS, the objective of TSN is cost reduction: the same wiring and network equipment can be used for different purposes, enhancing the flexibility and interoperability of the different devices of the factory.

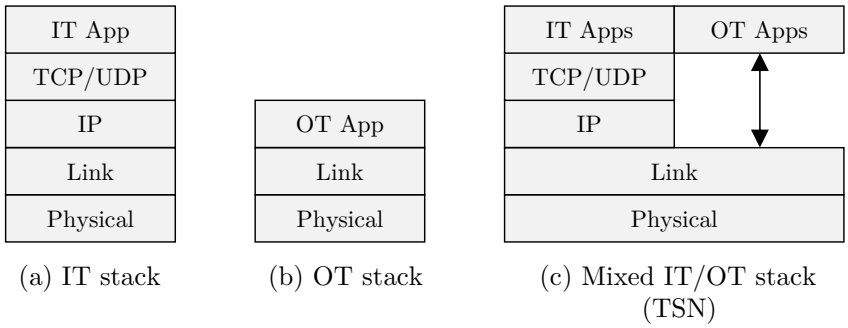


Figure 2.5. Protocol stack adopted in (a) IT, (b) Industrial, Operational Technology (OT), and (c) mixed IT/OT (TSN).

Traffic profile

The second relevant particularity of industrial networks compared to traditional IT networks resides in the traffic generated by the applications (Table 2.1). IT applications (multimedia/web browsing, etc.) are typically characterized by the exchange of large packets ($>$ 250 bytes) of variable size and that are generated by the asynchronous events of the IT application (e.g. a click on a link in a web browser). IT applications require high end-to-end reliability, which is typically provided by the TCP layer (layer 4). Consequently, IT applications do not have

specific constraints of latency/reliability at the link/PHY layer. Moreover, two main traffic profiles can be distinguished in industrial applications: control messages, and alarm messages [75]. On the one hand, control messages are characterized by small data sizes (sensing/actuation values of few bytes), that are periodically and synchronously generated, and that must be processed and received before a specific deadline. On the other hand, alarm messages are generated upon an asynchronous specific event and they are very infrequent. Alarm messages also require high reliability and low latency. The traffic generated by industrial applications presents more challenges than IT, but it is also far more predictable due to the synchronous operation of industrial applications.

Table 2.1. Traffic profile of different applications.

	IT apps	**Ind. apps, control**	**Ind. apps, alarm**
Payload size	> 250 bytes	10 - 100 bytes	10 - 100 bytes
Periodicity	Aperiodic	Periodic	Aperiodic, very rare
Generation	Event-based	Time-based	Event-based
PHY/MAC Latency constraints	No specific constraints	Bounded latency (0.25 - 10 ms)	Bounded latency (1 ms - 10 ms)
PHY/MAC Reliability constraints	No specific constraints	PER<10^{-7}	PER<10^{-7}

Cross-layer design

Finally, the last meaningful particularity of the industrial networks is the cross-layer design of the network. In general, IT networks follow an uncoupled or event-based network model. In this model, the operation of the layers of the communication stack is triggered based on events. These events are asynchronously generated (i.e. each layer does not have prior information about the events). The event-based network-application model provides simple interoperability among different layers and implementations. Nonetheless, control systems are characterized by periodically and synchronously generated events. To some extent, these events are triggered upon a specific time instant. Consequently, a time-triggered network model is more convenient for industrial

networks. This model is characterized by two relevant properties. In the first place, the applications and the network interfaces share a very accurate common notion of time, which enables synchronous operation. In the second place, the network and applications share their timing and other relevant information about the traffic profile (periodicity, payload size, No. of packets, etc.) and network properties (network scheduling, network saturation, throughput, latency capabilities, etc.). The operation of this model is very predictable and enables the joint design of the applications and network as a whole.

2.2.4. Communication requirements in Networked Control Systems

The ultimate goal of the industrial networks is to deliver the data generated by the applications with low and bounded latency. From this goal, four requirements are typically derived. Note that from them, other requisites, such as throughput, can be estimated. Nonetheless, they are a consequence of the NCS goal, and thus they are not described as separate requirements here.

Reliability

An industrial control system is designed to periodically execute specific tasks based on the exchanged information among the sensors/actuators and controller. Hence, reliability is a fundamental requirement in industrial networks and more significantly, in industrial wireless networks due to wireless links are more prone to errors than wired links. The metric used to evaluate the reliability is the PER. The PER is the ratio between the erroneous delivered packets (N_e) and the number of transmitted packets (N_T)

$$\text{PER} = \frac{N_e}{N_T}. \tag{2.1}$$

Fortunately, control systems are typically designed to be robust to packet losses, and hence one lost packet in a long time is not usually critical in the general performance of a control system. However, several consecutive packet losses can lead to a reduced performance of the control system, or, in more severe situations, the shutdown of the application. Thus, the number of consecutive erroneous packets must be minimized to ensure an appropriate performance [76]. The

maximum number of consecutive erroneous packets is a significant challenge in wireless systems since the wireless links are typically characterized by a time-varying error probability, which leads to error bursts, where several packets are lost in a row. The Packet Error Burst [76] (PEB) can be convenient to evaluate the wireless system reliability. The PEB characterize the probability of experience several packet losses in a row

$$\mathrm{PEB}_i = \frac{N_{e,i}}{N_T}, \tag{2.2}$$

being i the number of consecutive erroneous packets, $N_{e,i}$ the number of erroneous bursts with i packets. Note that PEB_1 does not equal the PER, since the former only considers the erroneous packets without any contiguous error.

The PEB_i can be also presented through its cumulative version (CPEB_n) that is expressed as follows

$$\mathrm{CPEB}_n = \sum_{i=n}^{\infty} \mathrm{PEB}_i \cdot i. \tag{2.3}$$

CPEB_n represents the probability of experience an error burst of n or more packets. Finally, the PER can be also computed from the PEB as follows

$$\mathrm{PER} = \sum_{i=1}^{\infty} \mathrm{PEB}_i \cdot i. \tag{2.4}$$

Global notion of time

Every individual element within a control system must synchronously and periodically perform specific tasks, such as reading the value of a sensor. Thus, the synchronization of the control system elements to a global notion of time is a baseline for its adequate performance. The synchronization performance is typically evaluated through three parameters, which are, according to the ISO 5725-11994 [77] definition, named precision, trueness, and accuracy.

The *trueness* (μ) is the systematic deviation of the error, or mean error between the reference time ($\mathrm{t}_{m,i}$) and the time of one node ($\mathrm{t}_{s,i}$).

$$\mu = \frac{1}{N}\sum_{i=1}^{N} \mathrm{t}_{m,i} - t_{s,i}. \tag{2.5}$$

The *precision* (σ) in synchronization is defined as the standard deviation of the error between the reference time and the time of one node. The precision is also commonly named as synchronization jitter.

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\mathrm{t}_{m,i} - t_{s,i} - \mu)^2}. \tag{2.6}$$

Finally, *accuracy* is a combination of trueness and precision. An accurate synchronization is characterized by both low trueness and precision. The details about the requirement of a global notion of time are further detailed in Section 3.1.

Update rate

The physical process to be controlled by the control system requires a minimum sensing/actuation rate to guarantee an adequate control operation. Hence, the sensors and actuators data generation rate, or update rate, is a relevant requirement in the design of the control system. The update rate is proportional to the variation speed of the measured variables. For instance, a slow varying process like a temperature requires a lower update rate than the control of a robot. The update period is the inverse of the update rate, and it indicates the periodicity of the sensing/actuation of the control system. Typically, it is considered that the update rate is the inverse of the control cycle of the control system.

Bounded end to end latency with low jitter

Typically, though it depends on the control system design, the overall latency of the control system should equal the update period of the sensors and actuators. The latency of the control system includes sensing, transmission of the data from the sensor to the controller, controller processing, transmission of the data from the controller to the actuator, and actuation. As a result, the network end to end latency (T_{E-E}) should be bounded by a latency deadline T_d.

$$T_{E-E} \leq T_d. \tag{2.7}$$

T_d typically takes a value equal to half of the update rate period. For instance, for an update period of 1 ms, T_d equals 0.5 ms.

T_{E-E} can be expressed as the sum of the latency of several processes. These latencies are represented in Figure 2.6 and described below. Note that some of the latencies, such as the data generation and consumption are included in the description, though they are not included in T_{E-E} since they are not related to the network itself.

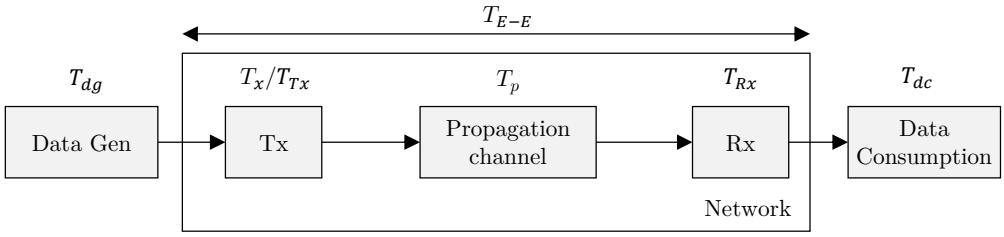


Figure 2.6. Source of latencies in a communication network.

- Data generation latency (T_{dg}). It is the time elapsed between the data is ordered and the data is available.
- Channel access latency (T_x). It is the time elapsed between the data is available at the network interface and the frame that carries the data is transmitted to the physical layer of the network interface.
- Coding and modulation latency (T_{Tx}). It is the time taken by the physical layer to generate a frame from the data received from the link layer.
- Frame duration (T_f). It is the time elapsed between the start of the frame (first symbol of the preamble), and the end of the frame.
- Propagation latency (T_p). It is the latency of the communication link. Its value is in the range of few nanoseconds for industrial wireless communications at the field level and it is considered negligible compared to the overall network length.

- Demodulation latency (T_{Rx}). It is the time taken by the receiver to demodulate and decode the PHY frame and send the decoded data to the upper layers.
- Data consumption latency (T_{dc}). It is the time taken by the receiver application to use the data received from the network.

Considering that the network latency does not consider T_{dg} and T_{dc}, T_{E-E} can be expressed as

$$T_{E-E} = T_x + T_{Tx} + T_f + T_{Rx}. \tag{2.8}$$

T_{Tx} greatly depends on the MAC and network architecture design, and on the PHY implementation, T_{Rx} depends on the implementation of the PHY and MAC layer, T_f depends on the design of the PHY layer and its throughput, and T_x mostly depends on the network design and the relation between network and application. Generally, T_f and T_x are the limiting factors of the achievable T_{E-E}.

Additionally, as the elements of the NCS, the network is expected to behave as predictable as possible. Consequently, the data generated and transmitted through the network should present minimum jitter (i.e. T_{E-E} should be constant for the periodic packets generated by a specific application.

2.3. Examples of industrial use cases and their requirements

Industrial applications at the field level present, to large extent, latency, update rate, synchronization, and reliability constraints. Nonetheless, the numbers can significantly vary across different applications, from, for instance, modest latency constraints in the order of hundreds of milliseconds to the microsecond level.

Table 2.2. Communication requirements of industrial control applications [38], [78].

Scenario	Update period	No. of nodes	Goodput [bps]	Reliability
Building automation	10 s	$10^2 - 10^3$	$10^2 - 10^3$	Medium
Process automation	100 ms	$10^2 - 10^3$	$10^5 - 10^6$	Medium
Factory automation	1 ms	$10^2 - 10^3$	$10^7 - 10^8$	High
Power systems automation	100 µs	$10^1 - 10^2$	$10^7 - 10^8$	High
Power electronics control	10 µs	$10^2 - 10^3$	$10^9 - 10^{10}$	Very high

Several authors and institutions have carried out different classifications of the industrial applications and some of their corresponding requirements. For instance, M. Luvisotto, Z. Pang, and D. Dzung presented in [38] and [78] 5 applications ordered by their required update period, goodput, and reliability: Building Automation, Process Automation, Factory Automation, Power Systems Automation, and Power Electronics Control. The goodput (throughput at the application layer) is derived from the No. of nodes within the industrial system, its update rate, and the amount of data periodically generated by each node (12 bytes). The latency requirement is not stated, but it may be derived from the update period considering that network latency should not exceed the update period. For instance, in Factory automation the latency should not exceed half of the update period i.e. 500 µs. Power electronics control has an update period of 10 µs and hence its latency should be in the range of few microseconds, which is a real challenge in nowadays communication systems.

K. Montgomery et al. from the National Institute of Standards and Technology (NIST) recently published a detailed report [75] of the requirements of wireless communication systems for industrial applications. This report includes the requirements stated by several authors and institutions, such as [38], [78], and also their perspective. The requisites stated by NIST, (summarized in Table 2.3), are classified into five classes upon the purpose of the traffic generated by the applications. Class 0 corresponds to alert/safety messages that are generated to prevent damages. Class 1, 2, and 3 are related to control applications. Class 1 refers to traffic generated by a control loop used to regulate a physical process. Class 2 is related to applications that send commands to perform different tasks within a large process. In class 3, the control is performed

by a human and hence it has a lower latency constraint. Finally, class 4 refers to applications of monitoring purposes.

Table 2.3. Communication requirements of the factory workcell by NIST [75].

Class	Name	Update period [ms]	Latency [ms]	No. of nodes (per workcell)	Payload [Bytes]	Reliability
0	**Safety**	$1-8$	$0.5-4$	$8-16$	$6-24$	10^{-7}
1	**Closed Loop Reg. Control**	$0.5-8$	$0.25-4$	$10-30$	$8-1024$	10^{-7}
2	**Closed Loop Superv. Control**	$8-40$	$4-20$	$10-30$	$8-1024$	10^{-7}
3	**Open Loop Reg. Control**	$1-8$	$0.5-4$	$1-4$	$8-1024$	10^{-7}
4	**Alerting Monitoring**	$8-100$	$4-50$	$100-300$	12-33.000	10^{-7}

The requisites stated by NIST are in-line with the ones of [38], [78] for factory automation: the control update period is in the range of the millisecond or even below, the network latency should be as much as half of the update period, and the system requires high reliability (note that high reliability typically corresponds with a PER in the order of 10^{-7}). The requirements of NIST accepts a wider range of payload values, which is derived from the possibility of having multiple sensors/actuators connected to a single network interface, whose data can be aggregated in a large, single packet.

3
High-Performance Wireless Time Synchronization

One of the primary challenges in industrial networks is to accurately synchronize the local time of the nodes within the network to a common global notion of time. This requirement is already solved in the wired networks by several technologies and techniques, as PTP combined with HW timestamping [15]. However, from the wireless side, there are several research challenges and implementation gaps that must be solved to successfully deliver the performance required by the industrial applications.

In this chapter, protocol-based time synchronization mechanisms and their limitations for successful high-performance wireless time synchronization are analyzed in detail. Based on this analysis, two timestamping methods are proposed: the conventional timestamps and the enhanced timestamps. The former is based on the timestamping method commonly used in wired communications and it is not optimized for wireless communications. The latter introduces a novel correction in the conventional timestamps that compensates the wireless propagation impairments and the limited resolution of the wireless system. The correction factor dramatically increases the timestamping precision compared to the conventional one. Both conventional and enhanced timestamps have been evaluated by simulation and experimental means showing promising results for industrial applications reaching performances in the tens of nanoseconds (conventional) and below one nanosecond (enhanced).

This chapter is organized as follows. Section 3.1 describes the principles of wireless time synchronization, including basic definitions, time synchronization protocols, timestamping methods, and algorithms to perform the time correction. The section ends with the description of the issues that limit high-performance wireless time synchronization. Some relevant works about wireless time synchronization are summarized in Section 3.2. In Section 3.3, two HW timestamping methods are proposed: the conventional timestamps and their evolution, the enhanced timestamps. The integration of a time synchronization protocol into a wireless system is discussed in Section 3.4. Section 3.5 presents the modifications to the 802.11 modem implemented over the ADI RF SOM platform to support HW timestamping and high-performance time synchronization. Finally, Section 3.6 presents the attainable results through simulation and experimental means.

3.1. Protocol-based time synchronization

This section describes the basic principles of protocol-based time synchronization. The first subsection defines the basic terminology used in time synchronization. Subsection 3.1.2 shows the common clock model and states the challenges derived from the imprecisions of real clocks. Afterward, the different technologies that influence the time synchronization precision are shown in Subsections 3.1.3, 3.1.4, and 3.1.5. Finally, the specific challenges of wireless time synchronization due to the unpredictable behavior of the wireless channels are highlighted in 3.1.6.

3.1.1. Basic definitions

PTP Hardware Clock (PHC)

The PHC is an element that measures the time elapsed from 00:00:00 on January 1st, 1970 (Unix time). The PHC output has a total of 80 bits: 48 bits are used to represent the seconds elapsed from the starting date, and the 32 remaining bits represent the elapsed fraction of nanoseconds from 0 to 10^9 ns in the actual second. The PHC has reconfiguration capabilities and thus its output can be

synchronized to an external clock. Additionally, the PHC may provide a Pulse Per Second (PPS) output.

Pulse Per Second

A PPS is a periodic squared-shape binary signal with a period of one second. It has a rising edge every time that the nanoseconds of the PHC reach 0 ns. A PPS signal has two main purposes: it can be used to compare the synchronization of two or more nodes by comparing the difference in time of the rising edge of their PPS, or it can be also used as a synchronization signal for an external device.

Messaging protocol

A messaging protocol defines a frame exchange used to exchange messages with timing information (timestamps). These messages are used to evaluate and correct the time synchronization error between two or more nodes.

Timestamp

A timestamp is a record of the time due to a specific event, such as the transmission or reception of a frame of a messaging protocol. The timestamps quality can vary depending on the way that they are implemented, resulting in largely different levels of synchronization.

3.1.2. Clock model

A clock is a device that is used to measure time and time intervals. It comprises an oscillator and a timer. The oscillator is an element that generates a very stable square-like signal with a period approximately equal to T. The timer is a discrete element that increases its output by T every time that it detects a rising edge in its input oscillator signal. In an ideal case, the oscillator signal period exactly matches T, and thus the timer perfectly follows the time advancement. Additionally, a clock can be thought of as an element that samples the time in the continuous domain and generates a discrete representation of the time

$$C(t) = t + t_o, \tag{3.1}$$

$$C[n] = nT + t_o, \tag{3.2}$$

being $C(t)$ the time measured by the clock, $C[n]$ a discrete version of $C(t)$, which is the output of the clock, and being t_o the clock initialization time, usually named time offset, since it is the difference between the system time reference and the actual time represented by the clock.

Despite being very stable, real oscillators present jitter (fast deviation) and an oscillator offset (slow deviation) from T. The jitter t_j is the difference between the expected rising edge instant and the true rising edge instant. In order to simplify the notation through the next sections, the jitter will be noted as the uncertainty in the phase of the oscillator rising edge: $t_j = \phi_j T$, being ϕ_j the uncertainty of the phase caused by the jitter. ϕ_j is usually modeled as a random variable that follows a zero-mean Gaussian distribution $\phi_j \sim \mathcal{N}(0, \sigma^2_{\phi_j})$, being σ_{ϕ_j} the phase std. deviation. A common oscillator for communications applications may be characterized by t_j in the ten picoseconds order, which is noticeably small, and it does not have a major impact on the time synchronization performance.

The oscillator offset is defined as the difference between the true oscillator period and the expected period (T) in the long term. The oscillator offset can vary from one oscillator to another due to small imprecisions in the manufacturing processes, the aging of the oscillators, different input voltages, or due to each oscillator is affected by different temperature conditions. In addition, a temperature variation may yield to a time-variant oscillator offset. The oscillator offset produces a drift in the output of the clock, usually named time drift (t_d). t_d is represents the time deviation of the output of the oscillator for a given period and it is a non-dimensional variable. It is characterized as the maximum possible deviation of the oscillator $t_{d,M}$ and, for a typical oscillator in a communications system, it may be in the order of $t_{d,M} = 10^{-5}$, or 10 parts per million (ppm). Thus, the output of a clock, rather than being a perfect representation of the time, as modeled in (3.2), is an approximated version of the time

$$C[n] = nT(1 + t_d) + \phi_j T \ + t_o. \tag{3.3}$$

The objective of the time synchronization is to adjust the output of a clock, so its value approximately matches the time of an external clock. The external clock is usually named the master clock, and the local clock is usually named the slave clock. As evidenced by the clock model description and from (3.3), the problem of time synchronization is reduced to estimate the time offset t_o and the continuous estimation of the time drift t_d. The next subsections describe four key aspects in wireless time synchronization: messaging protocol, timestamping approaches, time correction, and the time synchronization particularities over wireless channels.

3.1.3. Messaging protocols

Precision Time Protocol (PTP)

IEEE 1588 standard [15], commonly known as PTP, is the de-facto standard to perform time synchronization in industrial networks over fiber or Ethernet because of its simplicity and precision in the tens of nanoseconds range. The protocol follows a master-slave structure, where the master shares its local time with the slaves connected to it. PTP uses the timestamps taken in a four-frame exchange to evaluate the time synchronization error. The principles of the protocol are summarized in Figure 3.1 and further explained below.

The frame exchange starts with a Sync frame transmitted by the master, which takes the t_1 timestamp. The frame reaches the slave after traversing through the channel with delay t_{ms} and the slave takes the timestamp t_2'. Then, a Follow_up frame is sent from the master to the slave to transmit the timestamp t_1 to the slave. The Follow_up frame is optional since t_1 can be also delivered through the Sync frame. Afterward, the slave transmits a Delay Request frame (Delay_Req) to obtain two more timestamps (t_3', t_4). The delay of the channel between the slave and the master is t_{sm}. Finally, the Delay Response frame (Delay_Resp) delivers the t_4 timestamp to the slave. The frame exchange is periodically and indefinitely repeated with a period T_{Sync} to continuously estimate the time synchronization error.

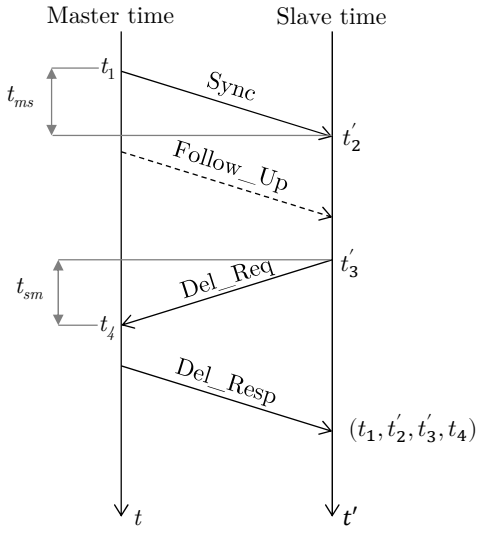


Figure 3.1. PTP frame exchange, being t the time of the master clock and t' the time of the slave clock.

Once the four timestamps are on the slave side, it performs the next calculations to synchronize its time with the master time

$$\tilde{t}_{ms} = \left(\frac{t_{ms} + t_{sm}}{2}\right) = \frac{t_2' - t_1 + t_4 - t_3'}{2}, \tag{3.4}$$

$$\tilde{t}_o = t_2' - t_1 - \tilde{t}_{ms}, \tag{3.5}$$

where $\tilde{t}_{ms}$ represents the estimated channel delay between the master and the slave and $\tilde{t}_o$ is the estimated difference between the slave time and the master time. $\tilde{t}_o$ is used by a time correction algorithm to synchronize the slave clock to the master clock (see Subsection 3.1.5.).

Simplified PTP scheme

PTP requires a full-frame exchange between the master and the slave nodes along with multicast / unicast operation [58]. This fact is translated into a considerable amount of traffic that linearly increases with the No. nodes of the network [41]. The traffic generated by PTP is not an issue in wired networks since the network capacity typically exceeds the requirements of industrial applications. However,

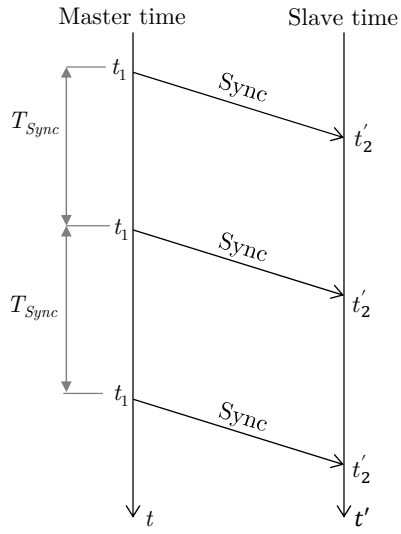


Figure 3.2. Simplified PTP exchange based on the periodic broadcasting of sync frames with period T_{Sync}.

wireless networks have limited radio resources, and consequently, other schemes with lower resources consumption may be more suitable depending on the network load and the synchronization requirements.

An alternative time synchronization scheme with minimum radio resources consumption is based on the broadcasting of beacon frames [8] and without feedback information from the slaves to compute the channel delay (see Figure 3.2). In this scheme, the master node periodically generates a sync frame that contains an egress timestamp (t_1). The frame reaches the slave t_{ms} later and the slave takes the timestamp t_2'. t_{ms} cannot be computed in this scheme and thus it is typically considered negligible. In some cases, it can be pre-calibrated with prior-known information about the distance between the wireless nodes. The time offset is then computed as

$$\tilde{t}_o = t_2 - t_1 - \tilde{t}_{ms}, \tag{3.6}$$

being $\tilde{t}_{ms}$ equal to 0 if the channel delay is considered negligible, or equal to the pre-calibrated channel delay.

IEEE 802.11 Fine Timing Measurements (FTM) scheme

The Wi-Fi FTM messaging was defined for the first time in the IEEE 802.11-2016 standard [47]. The main purpose of this messaging protocol was to perform ranging measurements based on radio frequency localization between Wi-Fi devices. This protocol can be also used to perform clock synchronization and was adopted by 802.1as over Wi-Fi [4]. The protocol follows a messaging scheme similar to the PTP messaging, though exploiting the intrinsic Wi-Fi frame exchange based on data + Acknowledgment (ACK) frames (see Figure 3.3). The FTM messaging replaces the Sync, Delay_Req, and Delay_Resp frames with FTM frames, which combines the Sync frames and Delay_Resp frames in a single frame, and Wi-Fi ACK frames, which are equivalent to the Delay_Req frames. Additionally, the FTM messaging includes the concept of FTM bursts. The FTM bursts are used to send several FTM messages in a row to reduce the error of the timing calculation.

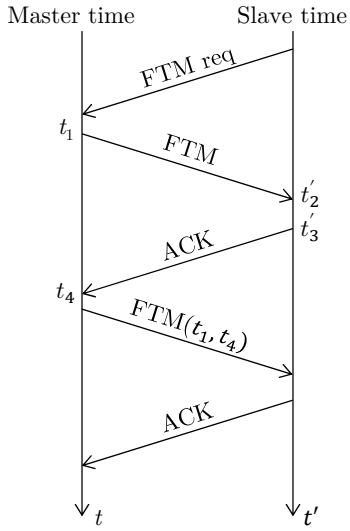


Figure 3.3. FTM frame exchange as defined in the IEEE 802.11 standard.

3.1.4. Timestamping approaches over wireless

In general, the key to successfully provide high-performance time synchronization is tightly related to the timestamping quality, which is typically defined as the precision of the timestamps, i.e. the std deviation of the timestamping error. According to the layer used to take the timestamps, the timestamps can be classified into 3 classes that provide different timestamping qualities. These classes are summarized in Table 3.1 and detailed below.

Table 3.1. Classes of timestamping techniques.

Class	Layer	Pros	Cons	Precision	Target networks
A	APP	COTS HW, simple	Low performance	ms order	Information technology networks
B	MAC	COTS HW	Affected by the processor and network loads	0.5 - 5 µs	Low-performance industrial networks
C	PHY	Stable, high performance	Custom HW	20 – 100 ns	High-performance Industrial networks

Class A: Software in Application Layer

In Class A, the timestamps are taken when a time synchronization packet leaves or arrives at the application layer. Class A timestamps suffer the unpredictable delays of the communication stack and provide a precision in the order of milliseconds. These timestamps are easy to implement in wireless COTS devices without requiring knowledge of the subjacent HW, but their performance is out of the pursued requirements for industrial networks [79].

Class B: Software in MAC Layer

Class B moves the timestamps from the application layer to the interruption routine of the network interface driver, as close to the HW as possible. These timestamps are only affected by the unknown delay of the MAC layer and by the interruption routine delay. To provide an adequate performance, Class B timestamps require deep HW/SW knowledge about the wireless system, the processor architecture, the operating system, and the driver. In addition, its performance depends on the microcontroller and network loads, and on the

priority of the interrupt lines. Class B provides a synchronization performance in the order of $0.5 - 10$ µs [41], two to three orders of magnitude better than class A.

Class C: Hardware in Physical Layer-Baseband

In Class C, the timestamps are directly taken at the exact moment of departure and detection of the frames from/to the baseband processor of the network interface. They are commonly named PHY timestamps, or HW timestamps since PHY is typically implemented in HW. The HW timestamps avoid the uncertainty of the communication stacks providing much higher precision than classes A and B reaching a performance in the nanoseconds range. Class C timestamps performance is bounded by the wireless system properties (BW, frame design) and the wireless propagation of the specific scenario (Signal-to-Noise Ratio (SNR), variation speed, and multipath). Most wireless COTS devices do not include HW timestamping and thus class C timestamps are usually implemented using custom solutions over FPGA [45].

3.1.5. Time correction

After taking the timestamps through the frame exchange and performing computation of $\tilde{t}_o$, the slave node has a time correction algorithm to synchronize its local time to the master time. A common implementation of a time correction algorithm follows two steps, which are summarized in Figure 3.4 and detailed below.

Step 1. Time offset correction

The slave clock is initialized in an unsynchronized stage. After the first frame exchange, the slave computes the time offset error $\tilde{t}_o$. If $|\tilde{t}_o|$ is higher than a threshold t_ϕ, the slave will consider that its local clock is totally desynchronized from the master time. t_ϕ may have a value in the range of 1 ms. In this step, the slave abruptly adjusts its local clock by directly subtracting $\tilde{t}_o$ from its clock output (see Figure 3.4 step 1),

$$C_{S,c} = C_S - \tilde{t}_o, \tag{3.7}$$

being $C_{S,c}$ the output of the timer after the correction.

Step 2. Time drift correction

$\tilde{t}_o$ is refreshed after each frame exchange. In the case that the new value of $|\tilde{t}_o|$ is smaller than t_ϕ, the slave node will estimate and correct its time drift t_d. There are several algorithms to estimate t_d. One common algorithm is based on a Proportional Integral (PI) filter with constants K_p and K_i

$$t_{e,I} = t_{e,I} + \tilde{t}_o, \tag{3.8}$$

$$\tilde{t}_d = \frac{1}{T_{Sync}} \left(K_p t_e + K_i t_{e,I}\right), \tag{3.9}$$

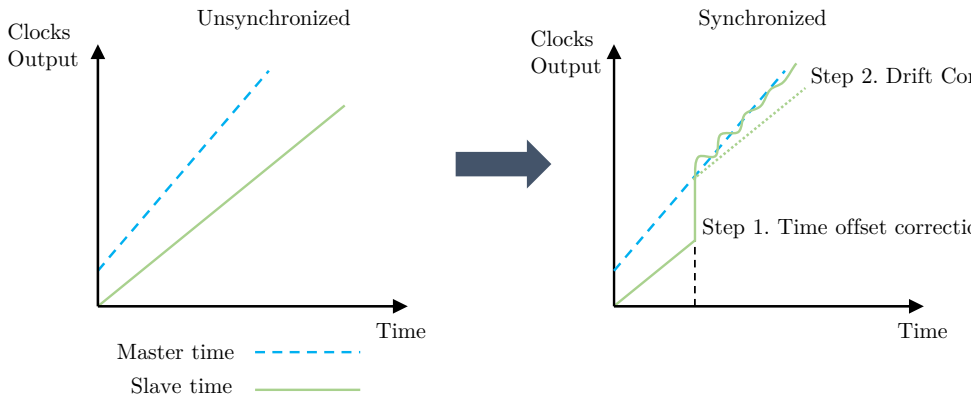


Figure 3.4. Steps to synchronize a slave clock to a master clock.

being $t_{e,I}$ the integral error component of the loop. In general, lower K_p and K_i will produce better time synchronization results, but at the cost of larger convergence time. Then, the slave clock is corrected by

$$C_{S,c} = \mathrm{C}_S(1-\tilde{t}_d) - \tilde{t}_o. \tag{3.10}$$

As shown in step 2 of Figure 3.4, Eq. (3.10) is used to correct the slope difference between the slave clock and the master clock. $\tilde{t}_d$ is periodically reevaluated in each frame exchange to compensate for the variations of the time drift and continuously resynchronize the local time to the master time.

3.1.6. Time synchronization particularities over wireless channels

In general, the key to successfully provide high-performance time synchronization is mostly limited by the timestamping quality. SW timestamps are limited by the jitter of the communication stack and hence they provide the same performance over wired and wireless links. On the other side, HW timestamps are only affected by the physical properties of the channel and the wireless system. Consequently, and given the superiority of HW timestamps, this subsection is specifically related to the particularities of HW timestamps over wireless.

Ingress/egress HW timestamps are differently affected by wireless links. On the one hand, egress HW timestamps are deterministic because the transmitter knows the exact egress time of the transmitted frame. Its error is bounded to the jitter of the communication chain (tens of picoseconds) or systematic errors due to incorrect calibration and can be usually considered negligible. On the other hand, ingress HW timestamps are affected by all the impairments derived from the wireless system and channel. Consequently, they are the main bottleneck in high-performance wireless time synchronization. The relation between the egress time of a frame and its estimated ingress time may be expressed as follows

$$t_A = \tau_h + t_D + t_e, \tag{3.11}$$

being τ_h the channel delay, t_e the timestamping error, t_D the egress time or time of departure, and t_A the frame estimated ingress time. τ_h is assumed to be equal from master to slave or from slave to master ($t_{ms} = t_{sm}$).

In the case of wired links (e.g. Ethernet), τ_h is fixed as the length of the wires does not typically vary. The timestamping error t_e basically depends on the frame detector, which has a resolution of 8 ns. Thus, the time synchronization performance is mainly limited by the ingress timestamping resolution. On the contrary, wireless systems exhibit significant limitations that challenge successful high-precision ingress timestamping, and as a result, high-performance time synchronization is a real challenge over wireless. These limitations are stated below.

- Wireless system BW. The timestamping resolution is typically the inverse of the BW for conventional timestamping techniques. For instance, a wireless system with 100 MHz BW will have a timestamping resolution of 10 ns, but a wireless system with 10 MHz BW will only have a timestamping resolution of 100 ns, which is a noticeable limitation for high-performance time synchronization.
- Multipath propagation. A wireless channel usually produces signal time-dispersion, which means that several signal replicas are received at different time instants at the receiver. Hence, the channel will have, rather than a unique channel delay τ_h, several τ_h, one for each signal replica.
- Wireless channel variation over time. The changes in the environment over time yield to a time-variant CIR and, thus, a variant signal time dispersion. Time-variant channels are present in networks with mobility which might be produced by moving communication nodes and/or moving environment (people, machinery, etc.).
- Lack of Line-of-Sight (LoS). The lack of LoS or Non-LoS (NLoS) produces strong variations of the wireless channel delay τ_h, which in turn introduces unpredictable results in the timestamping precision.

The combined effect of these phenomena significantly increases the error on the timestamps and, thus, these issues are challenging hurdles for high-

performance time synchronization over real-world wireless channels. For more details about wireless propagation, Section 4.1. further develops the wireless propagation phenomena and presents a commonly used wireless channel modeling based on the stochastic analysis.

3.2. Related work

Despite the apparent simplicity of the wireless time synchronization schemes and time correction algorithms, the implementation of wireless systems with sub-µs time synchronization is not straightforward. Through this subsection, the literature about wireless time synchronization is reviewed, from low-performance based on class A and B timestamps to high-performance solutions. Table 3.2 summarizes some works that illustrate the relation between the timestamping method, the layer of the communication stack, and the timestamping quality. In general, the timestamping quality is increased as the timestamps are taken closer to the PHY layer.

Table 3.2. Relation between the timestamping method and the time sync accuracy.

Ref.	Year	Validation	Wireless system	Messaging protocol	Timestamp class	Accuracy
[79]	2009	Experiment	802.11	NTP	A	1 ms
[8], [7]	2015	Experiment	Custom	Simplified PTP	B	1 – 2 µs
[80]	2018	Experiment	Custom	Custom	B	1 – 2 µs
[44]	2014	Experiment	802.11	PTP	B	0.5 – 3 µs
[45], [81], [82].	2012, 2012, 2014	Experiment	802.11b	PTP	C	Few ns over specific channels
[83], [84], [85]	2013, 2014, 2014	Experiment	802.15.4	PTP	C	Few ns over specific channels

Class A timestamps are simple to implement but their performance is not enough for the industrial networks and applications. As an example, [79] shows the time synchronization performance of application layer timestamps using Network Time Protocol (NTP) over the 802.11 standard (Wi-Fi). The

experiment resulted in a time synchronization precision in the millisecond order. This technique cannot fulfill the time synchronization requirements of industrial networks and applications.

Class B timestamps provide significantly better precision because the timestamps are taken at the driver and the unpredictable delays introduced by the SW layers of the communication stack are avoided. In this research line, some researchers from Ikerlan presented in [8] a wireless sensor network specifically tailored for train structure health monitoring with time synchronization capabilities. The wireless sensor network uses a narrowband radio with a custom TDMA MAC and the simplified messaging protocol. The system uses a clock with a resolution of 1 MHz to take SW timestamps in the interrupt service routine. The end-to-end time synchronization error is bounded to 2 µs, fulfilling the requirements of some industrial networks and applications. This synchronization scheme with the same radio chip was exploited in [7] to build a hybrid TSN (Ethernet TSN and wireless TSN) network. The synchronization is propagated from a Grand Master clock connected to the Ethernet TSN network to the whole network and providing 2 µs end-to-end synchronization. A similar approach was presented in [80] using a narrowband 802.15.4 PHY transceiver combined with a custom TDMA MAC. The results show a time synchronization precision in the order of 2-5 µs depending on the size of the wireless sensor network. In [44], Mahmood et al. combined Class B timestamps and PTP over Wi-Fi to achieve a time synchronization performance in the order of 0.5 µs. However, the experiments show a great deterioration of the time synchronization performance under elevated processor or network loads, which halts its trustworthiness for the implementation of industrial wireless networks.

When the timestamps are getting closer to the PHY layer, the physical properties of the wireless systems and the wireless propagation phenomena start to dominate the time synchronization achievable accuracy. HW timestamping (class C) techniques proposed in the literature are usually related to a specific wireless system and cannot be generalized to other wireless standards. Since the research of high-performance time synchronization is tightly related to wireless localization, some results are stated in terms of ranging performance. For instance, a timestamping technique designed for 802.11b is proposed and

evaluated in [45], [81], and [82]. In [45], it is described the implementation in FPGA of the 802.11b modem with HW timestamps and subsample timestamping precision using the specific chip structure of the 802.11b modulation. This design resulted in a synchronization accuracy below 1 ns for static conditions. However, according to the experiments documented in [45], the performance of the system is highly deteriorated for time-variant channels, showing an synchronization error larger than 20 ns. The receiver presented in [45] is improved in [81] and in [82] to combat the multipath propagation. The improvements proposed in [81] are mainly based on the use of frequency hopping to take timestamps in several independent channels. This is used to average the timestamping error introduced by the multipath propagation. Nonetheless, such a system needs a very specific implementation and the authors state in the paper that the ranging error is significantly larger when the target is moving. The solution proposed in [82] uses the interpolation of the cross-correlation peak to take timestamps with subsample precision and an equalizer to combat the multipath components and reduce the ranging error. However, the system is still designed for LoS conditions, and it is vulnerable to strong multipath components.

Another wireless synchronization scheme based on the interpolation of the cross-correlation peak but over IEEE 802.15.4 Chirp Spread Spectrum (CSS) is shown in [83], [84], and [85]. In [83] it is developed a theoretical analysis of the feasibility of the CSS modulation to perform precise timestamping and an early experimental validation is presented. The system performance is reported through simulations under direct LoS and little multipath propagation. Furthermore, the authors note that the system is not designed for NLoS conditions and the lack of LoS would strongly affect the timestamping precision and the overall system performance. A real testbed is developed, and its performance is evaluated in [84]. It is shown that the system can obtain sub-nanosecond synchronization accuracy over LoS conditions, but its performance for NLoS conditions is not reported. Finally, the timestamping validation strategy presented in [85] does neither support NLoS conditions.

3.3. Proposed hardware timestamping methods

As evidenced by the related work, the achievable wireless time synchronization is mainly limited by the technique used to take the timestamps. This section proposes two class C (PHY layer) timestamping methods. The first one, named conventional timestamping, is the timestamping method commonly used in PTP over wired networks, which considers nearly ideal channel conditions. The second one, named enhanced timestamping, has been designed considering the wireless propagation phenomena and wireless systems limitations, and it significantly outperforms the precision provided by the conventional timestamps.

This section starts with a description of the signal model used to design and evaluate the timestamps. In Subsection 3.3.2, the expressions that model the conventional timestamps and their limitations based on the developed signal model are stated. Subsection 3.3.3 develops the enhanced timestamps, which introduces a novel correction factor that compensates the limitations of the conventional timestamps. Finally, Subsection 3.3.4 presents an algorithm to implement the enhanced timestamps.

3.3.1. Signal model

Let be $s[l]$ a pseudorandom sequence with length and energy L and known by the receiver and transmitter. $s[l]$ represents the preamble of a time synchronization frame. The sequence has the next property

$$R_{s,s}[n] \approx L\ \delta[n], \tag{3.12}$$

where $R_{s,s}[n]$ is the autocorrelation of $s[l]$ and $\delta[n]$ is the Kronecker delta. These properties maximize the sequence detection probability and hence are usually found in the preambles of wireless frames.

The sequence $s[l]$ is sent using a pulse-shaping filter with impulse response $g(t)$, thus, the transmitted signal may be expressed as follows

$$s_1(t) = \sum_{l=0}^{L-1} s[l]\, g(t - Tl + \phi_{Tx}T), \tag{3.13}$$

with T as symbol duration and $\phi_{Tx}T$ as the uncertainty in the sampling instant (i.e. the jitter of the transmitter) that is modeled as a zero-mean Gaussian distribution $\phi_{Tx} \sim \mathcal{N}(0, \sigma^2_{\phi_{Tx}})$. Without loss of generality, $g(t)$ is assumed as a real, band-limited signal, with BW $B = \frac{1}{2T}$ and unit energy. $s_1(t)$ is the complex baseband representation of the passband signal with carrier frequency f_c. Although $g(t)$ is not time-limited, it can be assumed that most of the energy $g(t)$ is confined in the interval $[0, T_g]$, thus it is approximated by a truncated version of $g(t)$. Therefore, $g(t)$ is considered a finite impulse response. The signal $s_1(t)$ is transmitted through a communication channel whose impulse response is noted as $h(t)$. The CIR is defined in the interval $[T_d, T_h + T_d]$, where T_d is the CIR start and T_h is the CIR duration. $h(t)$ is the complex baseband representation of the passband CIR with a carrier frequency of f_c. Hence, the signal at the input of the matched filter is

$$s_2(t) = \sum_{l=0}^{L-1} s[l]\, g_h(t - lT + \phi_{Tx}T) + n_2(t), \tag{3.14}$$

where $g_h(t)$ is the convolution of $g(t)$ and $h(t)$, i.e., $g_h(t) = (g * h)(t)$. The noise component $n_2(t)$ is modeled as Additive White Gaussian Noise (AWGN). The matched filter with a response equal to $g(-t + T_g)$ is applied to $s_2(t)$ and it results

$$r(t) = \sum_{l=0}^{L-1} s[l]\, p(t - Tl + \phi_{Tx}T) + n(t), \tag{3.15}$$

being $p(t)$ the convolution of $g_h(t)$ and $g(-t + T_g)$, and $n(t)$ the filtered noise resulted from the convolution of $n_2(t)$ and $g(-t + T_g)$. The modeling of $h(t)$ and, more specifically, of $p(t)$ is described in Section 4.1. Finally, $r(t)$ is sampled at the receiver with a sampling period T

$$r[k] = r(t)|_{t=kT+\phi_{Rx}T} = \sum_{l=0}^{L-1} s[l]p((k-l)T + \phi T) + n(kT + \phi_{Rx}T), \tag{3.16}$$

where ϕ_{Rx} represents the timing jitter resulted from the sampling of $r(t)$ and $\phi = \phi_{Rx} + \phi_{Tx}$, the whole jitter. ϕ_{Rx} is modeled as $\mathcal{N}(\mu_{\phi_{Rx}}, \sigma_{\phi_{Rx}}^2)$, being $\mu_{\phi_{Rx}}$ the unknown phase difference between the master and slave clocks. $\mu_{\phi_{Rx}}$ is considered time-invariant because of the variation caused by the time drift t_d is negligible taking into account the short duration of $r(t)$.

3.3.2. Conventional timestamps

A common approach to take timestamps in wired systems is to use the frame start detector included in the PHY layer of the communication system. A widely used frame detector is based on the detection of a known sequence in the received samples. In order to find the start of a frame, $r[k]$ is cross-correlated with $s[k]$

$$R_{r,s}[n] = (r \star s)[n]. \tag{3.17}$$

$R_{r,s}[\mathrm{n}]$ will have a peak when the training sequence is cross-correlated with the received training sequence. If the peak exceeds a threshold, it will be considered that a frame has been detected. Hence, the estimated frame start (n_{ini}) is

$$n_{ini} = \min \{n \in \mathbb{N} / |R_{r,s}[n]|^2 > \theta\}. \tag{3.18}$$

The timestamping methods based on threshold detection rely on detecting the first component of $R_{r,s}[n]$, or the first signal replica. The value of θ is usually set based on minimizing the number of undetected frames without exceeding a false alarm probability. This simple timestamping solution has two main drawbacks in wireless systems. First, the timestamps have a resolution bound equal to the sampling period T, due they are based on the detection of a peak in a discrete sequence sampled at period T. Thus, they have an unknown error that follows a uniform distribution between -$T/2$ and $T/2$. Second, the timestamps may have a large variation under small variations of $R_{r,s}[n]$. A simple example of this drawback is depicted in Figure 3.5, where similar $R_{r,s}[n]$ sequences result in totally different timestamping values. This situation may be very common in wireless channels without a strong path, such as in NLoS conditions.

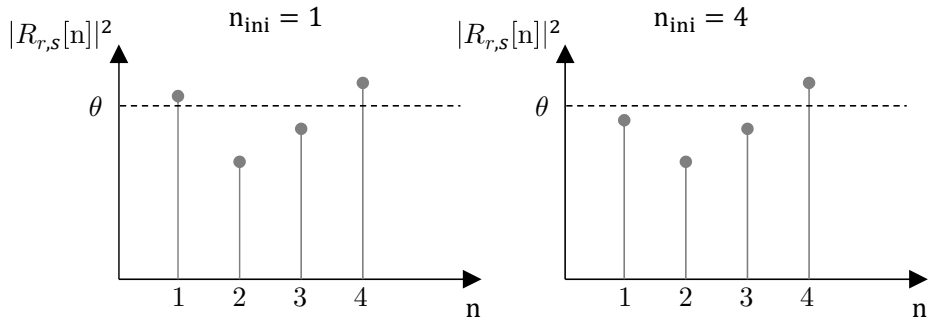


Figure 3.5. Issue of conventional timestamps: small CIR variations can cause a large frame start detection error.

3.3.3. Enhanced timestamps

The enhanced timestamps are designed to overcome the resolution bound of the conventional timestamps and improve their robustness over multipath propagation. For the sake of clarity, (3.16) is repeated here

$$r[k] = r(t)|_{t=kT+\phi_{Rx}} = \sum_{l=0}^{L-1} s[l]\ p(kT - lT + \phi T) + n(kT + \phi_{Rx}T). \tag{3.19}$$

Eq. (3.19) leads to a similar expression of the classical Symbol Timing Recovery (STR) problem [86]. From here two cases can be distinguished

$$\begin{aligned} B_c &> B, \\ B_c &\leq B, \end{aligned} \tag{3.20}$$

where B_c is the coherence BW of the channel and B the wireless system BW. When B_c is higher than B, the CIR can be approximated by a Dirac delta with weight h_0 [87]. The magnitude of h_0 is equal to the square root of the channel power gain and its phase is equal to the phase of the CIR. Thus, the received signal can be expressed as

$$r[k] = h_0 \sum_{l=0}^{L-1} s[l]\ g_1(kT - lT + \phi T) + n(kT + \phi_{Rx}T), \tag{3.21}$$

being g_1 the convolution of $g(t)$ and $g(-t+T_g)$. Therefore, the ingress timestamp of the frame can be precisely estimated by using any STR algorithm, which is already included in most communication systems. Very similar solutions are described in other works [18] [21] as detailed in Section 3.2, where the receiver estimates the phase of the master clock and takes the timestamps with sub-sample precision. Nonetheless, this solution can only be effectively used over channels with a strong direct component and static channels because the performance of STR algorithms deteriorates under time-dispersive and time-variant channels. Furthermore, STR algorithms rely on the detection of only one component, which would lead to an unstable timestamping precision in NLoS. Hence, the aim is to design an algorithm to precisely take the timestamps in time-dispersive channels (i.e. $B_c \leq B$).

The wireless multipath propagation generates several signal replicas that may arrive at different times at the receiver. Hence, the channel delay τ_h, rather than being a unique instant, is a distribution of delays, as the signal is replicated and received in multiple instants. The mean delay spread operator [50] applied to $p(t)$ is a convenient definition of τ_h, as the mean delay spread operator integrates and weights the delay of each time instant.

$$\tau_h = \overline{\tau} = \frac{\int_{-\infty}^{+\infty} |p(t)|^2 \ t \ dt}{\int_{-\infty}^{+\infty} |p(t)|^2 \ dt}. \tag{3.22}$$

It must be noted that, under time-variant channel propagation, the CIR can change over time, so τ_h may also change. Nonetheless, from (3.22), it is clear that two similar $p(t)$ will produce similar τ_h. Furthermore, the mean delay spread property

$$\overline{\tau} = \frac{\int_{-\infty}^{\infty} |p(t-t_0)|^2 \ t \ dt}{\int_{-\infty}^{\infty} |p(t-t_0)|^2 \ dt} = \frac{\int_{-\infty}^{\infty} |p(t)|^2 \ t \ dt}{\int_{-\infty}^{\infty} |p(t)|^2 \ dt} + t_0, \tag{3.23}$$

shows that it is linearly affected by the reference instant. This property offers a great advantage in the implementation of the enhanced timestamps, as they can be implemented as an evolution of the conventional timestamps. Furthermore, it

can be proven by using Lemma 1 that $\overline{\tau}$ can be calculated from the discrete mean delay operator applied to the discrete version of $p(t)$ sampled at the Nyquist rate

$$\overline{\tau} = \frac{\int_{-\infty}^{+\infty} |p(t)|^2 t \; dt}{\int_{-\infty}^{+\infty} |p(t)|^2 \; dt} = T \frac{\sum_{n=-\infty}^{+\infty} |p[n]|^2 n}{\sum_{n=-\infty}^{+\infty} |p[n]|^2}. \tag{3.24}$$

Lemma 1. Let $p(t)$ be a causal signal of duration T_g, with unit energy, and approximately band-limited to BW B; and let $T = 1/2B$ be the sampling period. Then, it follows the next identity:

$$\int_{-\infty}^{\infty} |p(t)|^2 \cdot t \; dt = T^2 \cdot \sum_{n=-\infty}^{+\infty} |p[n]|^2 n, \tag{3.25}$$

being

$$p[n] = p(t)|_{t=nT} = p(nT). \tag{3.26}$$

Proof.

Let be $f(t)$

$$f(t) = \sqrt{t} \; p(t). \tag{3.27}$$

The energy of $f(t)$ is

$$\|f(t)\|^2 = \int_{-\infty}^{+\infty} |f(t)|^2 \, dt = \int_{-\infty}^{+\infty} |p(t)|^2 t \; dt, \tag{3.28}$$

taking into account that $p(t)$ is causal. Now let consider

$$f[n] = f(t)|_{t=nT} = \sqrt{nT} \; p(nT), \tag{3.29}$$

$$p[n] = p(t)|_{t=nT} = p(nT). \tag{3.30}$$

$f(t)$ can be expressed as a function of $f[n]$

$$f(t) = \sqrt{T} \sum_{n=-\infty}^{\infty} f[n] \; h(t - nT), \tag{3.31}$$

being $h(t)$ an ideal interpolator filter of unit energy

$$h(t) = \frac{1}{\sqrt{T}} \operatorname{sinc}\left(\frac{t}{T}\right), \tag{3.32}$$

and being

$$\operatorname{sinc}(t) = \frac{\sin(\pi t)}{\pi t}. \tag{3.33}$$

The energy of $f(t)$ can be expressed in terms of the energy of the sampled version

$$\begin{aligned} \|f(t)\|^2 &= \int_{-\infty}^{\infty} |f(t)|^2 dt \\ &= T \int_{-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f[n]\ h(t-nT) \cdot \sum_{m=-\infty}^{\infty} f^*[m] h^*(t-mT) \\ &= T \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f[n]\, f^*[m] \int_{-\infty}^{\infty} h(t-nT) \cdot h^*(t-mT). \end{aligned} \tag{3.34}$$

Due to

$$\int_{-\infty}^{\infty} h(t-nT) \cdot h^*(t-mT) = sinc(m-n) = \delta_{m,n}, \tag{3.35}$$

being

$$\delta_{m,n} = \begin{cases} 1, & if\ m = n \\ 0, & if\ m \neq n. \end{cases} \tag{3.36}$$

Then

$$\|f(t)\|^2 = T \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f[n]\, f^*[m]\, \delta_{m,n} = T \sum_{n=-\infty}^{\infty} |f[n]|^2 = T^2 \cdot \sum_{n=-\infty}^{+\infty} |p[n]|^2 n. \tag{3.37}$$

Finally,

$$\int_{-\infty}^{\infty} |p(t)|^2 \cdot t\ dt = T^2 \cdot \sum_{n=-\infty}^{+\infty} |p[n]|^2 n. \tag{3.38}$$

Hence, the discrete mean delay spread also has the shift property (3.23) which means that it is unaffected by sub-sample delay shifts in p[n], such as the unknown sampling phase.

The CIR is not known in most scenarios and thus it has to be estimated. A suitable option is to estimate the CIR at the receiver based on the received sequence $r[k]$. A convenient and general method to estimate the CIR at the

receiver could be the cross-correlation of $r[k]$ and $s[k]$, because the estimation is obtained in the time domain.

$$\tilde{p}[n] \propto R_{r,s}[n] = (r \star s)[n]. \tag{3.39}$$

$R_{r,s}[n]$ is proportional to $\tilde{p}[n]$ in the absence of noise considering that the autocorrelation of $s[k]$ is approximately equal to a Kronecker delta with amplitude L. $R_{r,s}[n]$ is not limited in energy due to the presence of noise, thus the sum of the discrete mean delay spread operator must be limited. To establish the sum limits, it is assumed that the autocorrelation of $s[k]$ is approximately equal to a Kronecker delta. Then, the sum start is

$$n_s = \left\lceil \frac{T_d}{T} - \phi \right\rceil, \tag{3.40}$$

and its length is

$$N = \left\lceil \frac{T_h + 2T_g}{T} \right\rceil. \tag{3.41}$$

Thus, the enhanced timestamps operator reduces to

$$\overline{\tau} = T_s \frac{\sum_{n=n_s}^{n_s+N-1} |R_{r,s}[n]|^2 n}{\sum_{n=n_s}^{n_s+N-1} |R_{r,s}[n]|^2}. \tag{3.42}$$

Eq. (3.42) results in a very simple and robust expression to estimate the ingress time of the received frames, because: it is not vulnerable to start errors, as it integrates the whole CIR and it does not have a resolution bound. Therefore, the problem inherent to conventional timestamps is overcome by the definition of the enhanced timestamps. Nonetheless, n_s and N are not known in advance, and they must be estimated to ensure that $\overline{\tau}$ is obtained with minimum jitter. To do so, a simple and robust algorithm based on a dual $\overline{\tau}$ calculator is proposed. The algorithm is described in the next section.

3.3.4. Algorithm to implement the enhanced timestamps

The enhanced timestamping method relies on the knowledge of n_s and N to establish the correlation window limits, but this information is not known in

Algorithm 3.1. Enhanced timestamps algorithm.

Input: $r[k]$, $s[k]$, I_{it}
Output: $\overline{\tau}$
Initialize $i_{it} = 0$
1. Compute $R_{r,s}[n] = (r \star s)[n]$;
2. Find $n_{ini} = \min\{n \in \mathbb{N} / \left|R_{r,s}[n]\right|^2 > \theta\}$;
3. **repeat**
   3.1. $n_s = n_{ini} - N/2$;
   3.2. $\overline{\tau} = T \frac{\sum_{n=n_s}^{n_s+N-1} |R_{r,s}[n]|^2 \cdot n}{\sum_{n=n_s}^{n_s+N-1} |R_{r,s}[n]|^2}$;
   3.3. $\mathrm{n}_{ini} = round(\frac{\tilde{\tau}}{T})$;
   3.4. $i_{it} = i_{it} + 1$;
   3.5. **until** $i_{it} == I_{it}$;

advance. To obtain adequate performance, the estimation of n_s and N must be very robust because errors in the integration window position will deteriorate the precision of the timestamp.

First, the correlation window length, N, is pre-configured according to the communication systems properties. For example, in an OFDM-based system T_h can be set to the duration of the cyclic prefix, due to it is the maximum CIR duration to avoid inter-symbol interferences. On the other hand, T_g should be chosen according to the implemented pulse-shaping filter. Therefore, the problem is reduced to find n_s.

To estimate n_s, an iterative algorithm based on the combination of the conventional timestamps and the enhanced timestamps is proposed (Algorithm 3.1). The received signal is first introduced to the cross-correlator and the frame start index is detected using a threshold. n_{ini} is set as the mid position of the CIR, so $n_s = n_{ini} - N/2$. The captured samples are sent to the discrete delay spread operator. The result of the operation will be the enhanced timestamp ($\tilde{\tau}$). However, the threshold detector is prone to errors, thus n_{ini} estimation may not be exact. Therefore, n_{ini} is recalculated using $\tilde{\tau}$ to improve its precision. This process is iteratively done until a fixed number of iterations I_{it}.

This algorithm performs a rounding in step 3.3, which adds an error to the correlation window position. To eliminate this error, a fractional delay filter based on sinc interpolation [89] can be used instead of the rounding operation. This operation allows a perfect alignment of the correlation window and eliminates the error bound of Algorithm 3.1. However, the computational complexity of this operation is considerably higher than the complexity of the rounding operation and its implementation should only be considered for specific cases when the error caused by the rounding operation limits the synchronization performance.

3.4. Guidelines to deliver high-performance wireless time synchronization using the enhanced timestamps

The enhanced timestamping method significantly improves the timestamping precision compared with conventional timestamping methods. However, the enhanced timestamps apply some constraints to the wireless system and to the time synchronization scheme that must be followed to maximize the time synchronization performance. Through this section, some critical aspects to achieve the objective of high-performance time synchronization are described.

3.4.1. Wireless system constraints

The delay of the wireless channel depends on the distance between the nodes, which can range from few nanoseconds to several tens of nanoseconds or hundreds upon the distance between the wirelessly interconnected nodes. Consequently, to maximize the performance, the use of a messaging protocol with channel delay compensation is mandatory. Additionally, the channel delay from master to slave t_{ms} and from slave to master t_{ms} must be equal ($\tau_h = t_{ms} = t_{sm}$) to ensure that the channel delay is correctly estimated. Otherwise, the time synchronization will present an unresolvable deviation from the true channel delay.

τ_h definition depends on the timestamping method. The enhanced timestamps define τ_h as the mean delay spread of the wireless channel impulse response $p(t)$

(3.22). Hence, $p(t)$ must be reciprocal to guarantee equal channel delay from master to slave and from slave to master.

$p(t)$ reciprocity imposes three constraints to the wireless system. First, the wireless nodes must operate in Time Division Duplexing (TDD). The fulfillment of this condition depends on the chosen wireless standard or solution. For instance, 802.11 operates in TDD mode, whereas LTE and 5G can indistinctly operate in Frequency Division Duplexing (FDD) or TDD. Second, the node must use the same antenna to transmit and receive the frames. Third, the wireless channel must not vary during the time elapsed to take the timestamps, i.e. the elapsed time between the transmission of the PTP sync and the detection of the PTP Delay_Req (T_{S-D_r}).

The third constraint is the most challenging one since the wireless channel may continuously vary due to the movement of the nodes or the environment. Upon the fulfillment of the two first conditions, $p(t)$ must be invariant during T_{S-D_r} and consequently, T_{S-D_r} much be smaller than the channel coherence time. This condition entails a fast frame exchange whose requirements have been found through simulations (see Subsection 3.6.2). For instance, for low mobile conditions (up to 3 km/h), $T_{S-Dr} = 1$ ms is enough to maximize the time synchronization performance, meanwhile for higher speeds (more than 80 km/h) $T_{S-Dr} = 0.1$ ms is necessary. Through the next subsection, some guidelines to minimize T_{S-Dr} using different standard wireless systems are described.

3.4.2. Guidelines to maximize the performance under time-varying channels

Sub-millisecond T_{S-Dr} is a real challenge over wireless systems depending on their throughput and their MAC layer design. This subsection highlights how to minimize T_{S-Dr} over the most widely used wireless standards (802.11 and LTE/5G) and over w-SHARP, the custom wireless solution described in this thesis.

For instance, $T_{S-Dr} = 1$ ms is unfeasible in legacy 802.11 because 802.11 medium access is based on Carrier Sense Multiple Access with Collision

Avoidance (CSMA/CA) [29], which is not deterministic. Nonetheless, the latest versions of the standard include the FTM scheme, which replaces the PTP exchange with FTM / ACK frames. Since the ACK frames are always deterministically transmitted right after the FTM frames, the gap between them can be as small as tens of microseconds, which would be enough to ensure low channel variation.

Additionally, frame collisions may challenge PTP deployment over 802.11. For instance, if a master transmits a multicast/broadcast PTP sync frame, the frame could be received at the same time by several slaves. Then, the slaves would try to answer with a PTP Delay_Req at the same time, causing frame collisions. To avoid collisions, 802.11-based PTP should use a unicast PTP implementation. Although common PTP implementations over wires are multicast, the use of unicast PTP over 802.11 is recommended to reduce the collisions.

Other wireless systems, such as 5G-NR and LTE, use TDMA-based access, which allows an easier implementation of the synchronization scheme. In LTE or 5G-NR, the master (eNodeB or gNodeB) could pre-allocate radio resources to transmit the PTP sync, the PTP delay request, and (if needed) the PTP delay response. In this case, $T_{S-Dr} < 1$ ms can be fulfilled if a proper scheduler is designed. Nonetheless, it is worth noting that only the TDD versions of 5G-NR and LTE can be used.

3.5. Modified 802.11 modem with HW timestamps and PTP support

To enable wireless time synchronization with HW timestamping, a wireless network interface requires three additional modules compared to a common interface: a one-step HW Transmitter (Tx) timestamping module, an Receiver (Rx) HW timestamping module, and a PHC. These modules can be integrated without major changes in the baseband processor of a wireless system. The one-step Tx timestamping module is used to insert the egress timestamp in the timestamp field of a PTP frame. The Rx timestamping module uses the frame detector to detect the start of the frame and record the ingress time. Finally, the

PHC is used to take the timestamps and it represents the global notion of time of the node. Through the next four subsections, the implementation and integration of these modules into the 802.11 modem built over the ADI RF SOM platform are described. The Rx timestamping module is divided into two sections since two Rx timestamping methods are proposed.

3.5.1. One-step Tx timestamping module

The one-step Tx timestamping module is used to deterministically take and insert the egress timestamp t_{Tx} in a PTP Sync frame without SW intervention. An overview of the 802.11 modem Tx PHY with the one-step timestamping module is depicted in Figure 3.6. First, a PTP packet is transmitted by the 802.11 link layer to the HW. The packet is received at the timestamp insertion module. The timestamp insertion module captures the value of the PHC plus a calibration delay d_{Tx}. A calibration delay d_{Tx} is required because the timestamp is taken before the effective transmission of the frame. Then, the timestamp insertion module inserts t_{Tx} into the Tx timestamp field of the packet and the 802.11 Field Check Sequence (FCS) is calculated and appended at the end of the packet. The data bits are coded and modulated to generate an 802.11 frame, which is sent to the radio chip (AD9361). The radio chip performs the digital-to-analog conversion, filtering, amplifying, and up-conversion. Finally, the signal is transmitted through the antenna.

The calibration delay d_{Tx} can be divided into the delay introduced by the radio delay ($d_{Tx\,\mathrm{radio}}$), performed in the AD9361 chip, and the Tx baseband processing $d_{Tx\,\mathrm{PHY}}$, carried out in the FPGA. On the one hand, the delay of the radio chip $d_{Tx\,\mathrm{radio}}$ depends on its configuration, but it is fixed for a given configuration [90]. On the other hand, $d_{Tx\,\mathrm{PHY}}$ depends on the implementation of the coding and modulation processes. In this case, the PHY of the 802.11 modem has been specifically implemented with a fixed $d_{Tx\,\mathrm{PHY}}$, and thus the calibration delay d_{Tx} is fixed and equal to $d_{Tx} = d_{Tx\,\mathrm{PHY}} + d_{Tx\,\mathrm{radio}}$. Upon a perfect delay calibration, the egress timestamps precisely represent the egress time of the frames.

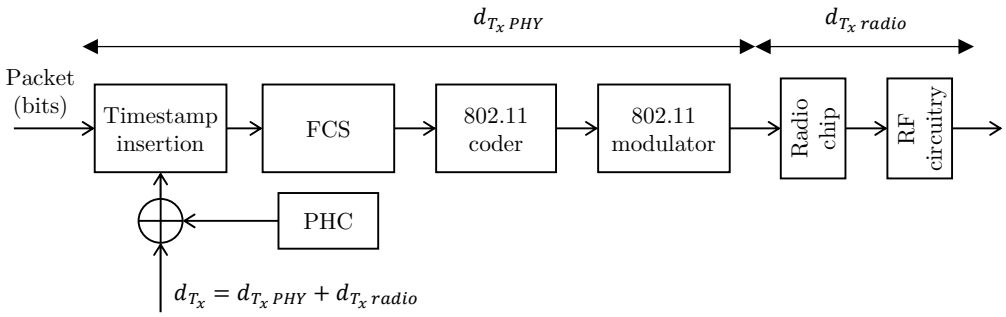


Figure 3.6. One-step Tx timestamping module.

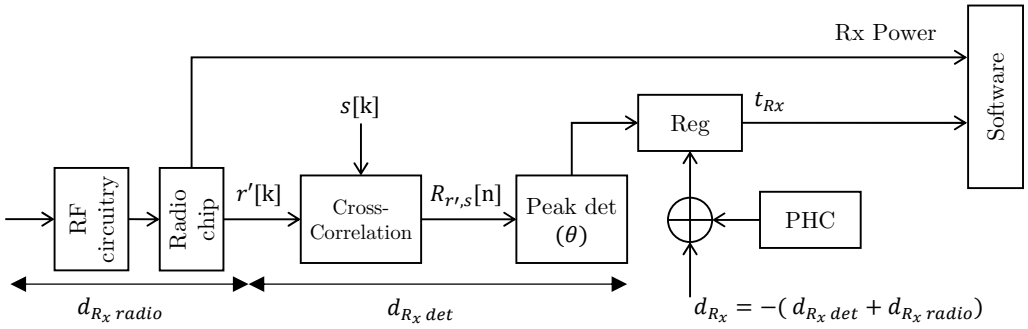


Figure 3.7. Conventional Rx timestamping module.

3.5.2. Rx Conventional timestamping module

The modifications to the Rx chain to allow HW timestamping are described in this section and depicted in Figure 3.7. The HW timestamping is defined in the frame detector and hence most modules of the 802.11 baseband processor (Fast Fourier Transform (FFT), channel eq, decoding, etc.) are unaffected by these modifications. For the sake of clarity, the coarse frequency estimation, and the detection of the first part of the 802.11 preamble are not depicted because they do not influence the Rx timestamping module implementation.

First, the HW receives from the antenna an RF signal. The signal goes through the RF circuitry and is received by the radio chip, which performs the down-conversion, Automatic Gain Control (AGC) correction, and analog-to-digital conversion at a sampling period of T. The output of the ADC is a

normalized version of $r[k]$ with unit energy $r'[k]$. $r'[k]$ is cross-correlated with $s[k]$ using a Finite Impulse Response (FIR) filter. If the cross-correlation block generates a peak that exceeds a threshold θ it means (with high probability) that a frame has been detected. In this case, the peak detector generates a trigger that captures the t_{Rx} timestamp, computed as the PHC output plus a calibration delay d_{Rx}. Then, t_{Rx} is sent to the SW that will store the timestamp for further processing. As in the transmitter side, the calibration delay d_{Rx} equals the radio chip delay $d_{Rx\,\mathrm{radio}}$ plus the delay of the frame detector $d_{Rx\,\mathrm{det}}$, which are constant. Finally, if the frame is correctly demodulated, the timestamp is appended to the frame metadata information.

3.5.3. Rx Enhanced timestamping module

The enhanced timestamps have been designed as an evolution of the conventional timestamps and, thus, it is natural to implement them reusing the HW of the conventional timestamps. Two approaches are proposed as an extension to the HW of the conventional timestamps, which are depicted in Figure 3.7 (a) and (b). Figure 3.7 (a) represents a pure HW approach, where the enhanced timestamps are directly computed and taken in the HW, and Figure 3.7 (b) depicts a hybrid HW/SW approach, where the raw information is obtained in the HW and the enhanced timestamps are computed in SW.

In approach (a), the cross-correlation $R_{r',s}[n]$ is directly introduced to a module that computes a correction factor $\overline{\tau}$ of t_{Rx} as the mean delay spread of $R_{r',s}[n]$. The mid position of the window of the mean delay spread is estimated by the frame start detector. This block may perform several iterations as described in Algorithm 3.1. Then, $\overline{\tau}$ is directly added to the PHC output to generate the enhanced timestamps. The main advantage of this approach is zero SW overhead since the timestamp is fully taken in the HW of the baseband processor.

In approach (b), $p'[n]$ is estimated in the frequency domain ($P'[l]$) through the in-built channel estimator of the 802.11 OFDM receiver. $P'[l]$ is transmitted to the SW along with the conventional timestamp. Then, the SW computes the enhanced timestamps by obtaining $p'[n]$ from $P'[l]$ using the Inverse FFT (IFFT) algorithm, obtaining the correction factor $\overline{\tau}$ by iteratively applying the mean delay spread operator to $p'[n]$, and finally, using $\overline{\tau}$ to correct t_{Rx}. The main

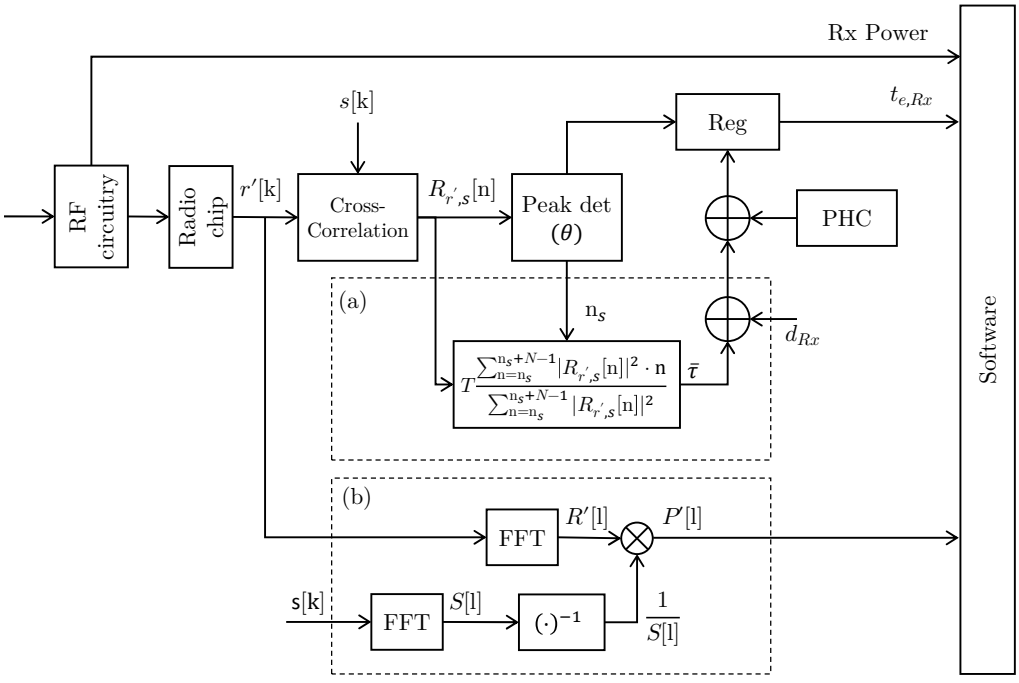


Figure 3.8. Implementation of the enhanced timestamps in the 802.11 modem. Pure HW (a) and hybrid HW/SW (b).

advantage of this approach is that it does not require extra HW to compute the timestamps, as the channel estimation is required in OFDM modems. Given that HW implementation is more complex than SW implementation, approach (b) seems to be more appropriate for wireless systems with fewer timing messages, and approach (a) better suits systems that generate more timing messages.

3.5.4. PTP Hardware Clock (PHC)

The PHC is a clock that can be adjusted in time offset $\tilde{t}_o$ and time drift $\tilde{t}_d$ to match its time count to an external timer. The PHC may have two synchronization interfaces: a SW interface (mandatory) and a PPS signal input (optional). On the one hand, the SW interface has two inputs: the estimation of the time offset ($\tilde{t}_o$) and the estimation of the time drift. On the other hand, the PPS signal input is fed with a PPS signal from an external HW that is used to estimate $\tilde{t}_o$ and $\tilde{t}_d$. The PHC HW architecture depends on the way that the time drift correction $\tilde{t}_d$ is implemented, and the way that the PHC output is separated in seconds/nanoseconds.

Two main approaches are distinguished to correct $\tilde{t}_d$: fully digital or mixed analog-digital. In the fully digital approach, $\tilde{t}_d$ is corrected in the digital domain by periodically adjusting the timer output. In the mixed digital-analog, the PHC time drift is adjusted through a Phase-Locked Loop (PLL) that controls the oscillator frequency. In general, a fully digital implementation is more convenient because it can be implemented in general-purpose HW platforms. Hence, the PHC of the 802.11 modem has been implemented using the fully digital approach.

Regarding the implementation of the output of the PHC, general-purpose timers only have one output, whereas the PHC separates its output into seconds/nanoseconds. There are several approaches to implement a timer with output in seconds/nanoseconds that differ in engineering efforts and HW complexity. For instance, one simple approach is to run the internal logic of the PHC in nanoseconds and then convert the output from nanoseconds to seconds/nanoseconds using arithmetic operations. This approach is simple from the implementation perspective if using HLS tools but consumes significant HW resources because it requires large division/multiplication operations. On the

other hand, the PHC itself can be built with the output divided into nanoseconds/seconds. This choice has fewer HW resources though has a significant logic complexity. In this case, the PHC has been implemented using the second option.

For the sake of clarity and without the loss of generality, the implemented PHC is described assuming only one output that represents the time in nanoseconds. Figure 3.9 depicts the block diagram of the digital implementation of the PHC in the SoC-FPGA 802.11 modem. Its behavior is as follows. First, the SW must choose if the PHC will be synchronized to the external PPS signal (PPS-Sync mode), or through the SW using a frame exchange (SW-Sync mode). The decision is generated through the SW/PPS bit, which multiplexes the outputs of the SW or the PPS time correction algorithm.

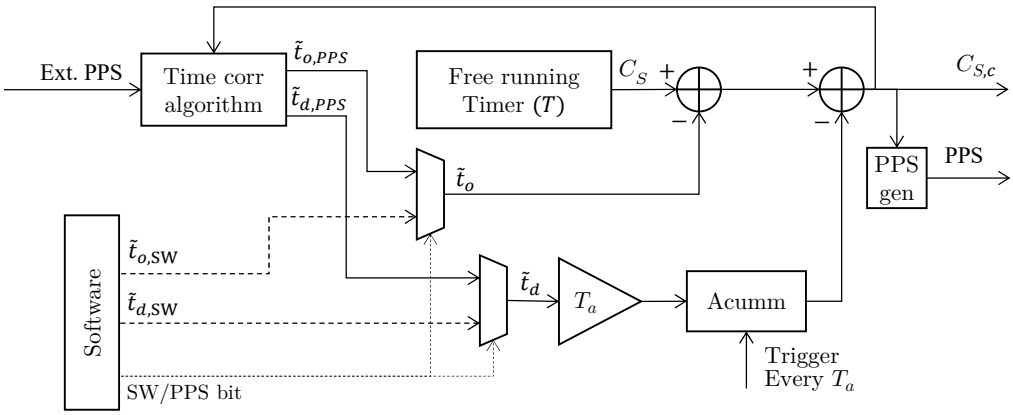


Figure 3.9. Implementation of the PHC in the 802.11 modem.

In SW-Sync mode, the input of $\tilde{t}_o$ and $\tilde{t}_d$ is directly fed by the SW, which has previously estimated $\tilde{t}_o$ and $\tilde{t}_d$ through a frame exchange. In the PPS-Sync mode, the operation is as follows. The time corr. algorithm block waits for a rising edge in the Ext. PPS signal. When the time correction algorithm detects the rising edge, it captures the output of the PHC $C_{S,c}$. $\tilde{t}_o = C_{S,c}$ because the rising edge of the PPS occurs when the external PHC output equals 0 ns. Finally, $\tilde{t}_o$ is used to correct the PHC offset and to estimate $\tilde{t}_d$ using the two-step algorithm described in Section 3.1.5.

Regarding the correction of the output of the PHC, $\tilde{t}_o$ is directly added to the output of the PHC, whereas the implementation of the time drift $\tilde{t}_d$ correction using the fully digital approach is as follows. First, $\tilde{t}_d$ is multiplied by a constant T_a. Then, the output of the PHC is adjusted by $\tilde{t}_d \cdot T_a$ every T_a, that gives a total drift adjustment of $\tilde{t}_d$ after one second. This operation is implemented as an accumulator with input $\tilde{t}_d \cdot T_a$ triggered every T_a.

Two constraints are applied to T_a. First, T_a should be multiple of 2 to avoid running complex division/multiplication algorithms. Second, T_a must be small enough to avoid abrupt adjustments in the output of the PHC. Adjustments larger than T are considered abrupt. Then, T_a minimum value is given by the maximum time drift ($t_{d,M}$) (provided in the oscillator datasheet) and the oscillator period T

$$T_a < \frac{T}{2t_{d,M}}. \tag{3.43}$$

$t_{d,M}$ is multiplied by 2 considering that the master and the slave use an oscillator with the same $t_{d,M}$. For instance, an appropriate value of T_a for $T = 50$ ns and $t_{d,M} = 20$ ppm is 2^{-11} seconds. Finally, the output of the PHC $C_{S,c}$ is introduced to the PPS gen block that generates a rising edge every time that the PHC output reaches 0 ns.

3.6. Performance analysis

This section presents the achievable performance by the proposed timestamps by both simulation and experimental means. First, Subsection 3.6.1 describes the simulation and experimental setups. The attainable time synchronization with the conventional and enhanced timestamps over simulation means are presented in Subsection 3.6.2. shows the results obtained with the enhanced timestamps by simulation means. Subsection 3.6.3 presents the performance of the time synchronization using the conventional timestamps by experimental means. Finally, Subsection 3.6.4 introduces the early experimental results obtained using the enhanced timestamps.

3.6.1. Simulation and experimental setups

The evaluation of both timestamping techniques has been evaluated by both simulation and experimental means. However, due to the engineering efforts of the implementation of the enhanced timestamps, they have not been implemented in the HW platform. Three setups are considered, the simulation setup, the emulation setup, and the early validation setup. On the one hand, the simulation setup is used to evaluate the synchronization performance of conventional and enhanced timestamps over simulations, and the emulation setup is used to test the conventional timestamps over a HW testbed. In order to obtain similar results, the simulation and the emulation setup have very similar conditions. On the other hand, the early validation setup is used to evaluate the enhanced timestamps over a HW testbed, but it does not fully address the whole time synchronization in the HW platform. The full HW evaluation of the enhanced timestamps will be subject to future work.

3.6.1.1. Simulation setup

The simulation tests have been carried in MATLAB® using MATLAB® WLAN Toolbox™ and MATLAB® Communications Toolbox™. The simulation setup is depicted in Figure 3.11. This setup has been used to evaluate the timestamps over a high variety of wireless conditions, from a small office and low movement speed, to open space and high movement speed. The WLAN Toolbox™ has been used to generate and decode the 802.11g frames. The MATLAB® Communications Toolbox™ has been used to generate the wireless channel models. Besides, some additional elements, namely the PHC, the timestamping techniques, and a PTP application have been implemented over MATLAB® to perform the simulation.

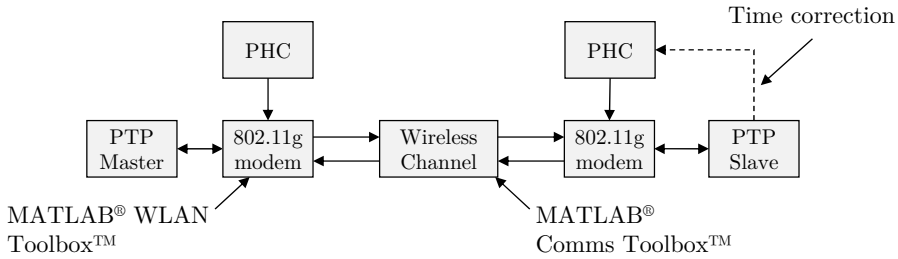


Figure 3.10. Simulation testbed to evaluate the time synchronization performance of the proposed timestamping techniques.

The simulation has been carried out as follows. Firstly, the channel models are generated using the Power Delay Profile (PDP) obtained from [51] and the nodes speed configuration. Afterward, a fixed number of PTP frame exchanges are performed and the slave clock is corrected in each frame exchange. Finally, the time synchronization error is calculated in each frame exchange as the difference between the master time and the slave corrected time. A total of 10^4 PTP frame exchanges have been carried out for each speed and SNR.

Regarding the PTP configuration, the PTP frame exchange period has been set to 1 s, and it is considered that $T_{S-Dr} = 1$ ms. Furthermore, a Proportional-Integral Loop filter has been used to reduce the noise of the $\tilde{t}_0$ calculation. The filter was configured with K_i and K_p constants equal to $2.6 \cdot 10^{-3}$ and $5.5 \cdot 10^{-2}$ respectively.

Besides, the clock model has been configured as follows. The maximum time drift $t_{d,M}$ of both master and slave clocks has been set to 10 ppm and the clock jitter std deviation has been set to 8 ps. The clock drift t_d and the time offset t_o have been set to a random value for each simulation.

It has been found in preliminary simulations that the algorithm to estimate the start window shows little improvements for more than 2 iterations. Therefore, the number of iterations of the algorithm for the simulation has been set to $I_{it} = 2$. Besides, the length of the channel has been set to $T_h = 16T$ (800 ns), (equal

to the cyclic prefix length in an 802.11 frame), and $T_g = 7T$ (350 ns), thus $N = 30$.

Regarding the wireless channel models, industrial wireless solutions are commonly designed to operate in the unlicensed 2.4 GHz band (e.g. WIA-FA, or the one developed in this thesis w-SHARP) and hence, it is natural to study the time synchronization performance over this band. IEEE 802.11 is probably the most widely used wireless system in the 2.4 GHz band and its specification contains a detailed characterization of the wireless propagation phenomena for different wireless scenarios [51], from a small office with low multipath propagation to large open environments. [51] includes 5 wireless channel models, classified from A to E in ascending richness of multipath propagation. The models are summarized in Table 3.3. The A and B models represent office conditions with relatively low multipath, whereas C, D, and E models represent large open spaces with higher multipath. All of them represent NLoS propagation (Rayleigh fading) and mobile nodes (Jakes Doppler spectrum), which may be considered as worst-case conditions for each of the represented scenarios. The WLAN D channel model has been excluded from the simulation/emulation setup because its propagation conditions are very similar to the WLAN C model. The simulation and experiments have been run over a different speed of the nodes from < 2km/h to 300 km/h.

Table 3.3. Wireless channel models used to evaluate the time synchronization [51].

Name	Scenario	LOS / NLOS	rms Delay Spread [ns]	Fading distribution	Doppler spectrum
WLAN A	Small Office	NLOS	50	Rayleigh	Jakes
WLAN B	Large Office	NLOS	100	Rayleigh	Jakes
WLAN C	Large open Space	NLOS	150	Rayleigh	Jakes
WLAN D	Large open Space	NLOS	150	Rayleigh	Jakes
WLAN E	Very large open Space	NLOS	250	Rayleigh	Jakes

3.6.1.2. Emulation setup

The objective of the experiments is to test the time synchronization error of the real wireless system with a setup as similar as possible to the simulation setup. Consequently, the experiments have been performed over 802.11g, using the PTP-like frame exchange, and over the same wireless channel models (Table 3.3).

The use of the same wireless conditions in the experiments could be a real challenge, since the wireless channel in a specific scenario (e.g. small office), may not behave as the WLAN A model. Probably, the only suitable way to ensure similar and reproducible results is to use a channel emulator programmed with the wireless channel models used in the simulation. The experimental setup is depicted in Figure 3.11 and further detailed below.

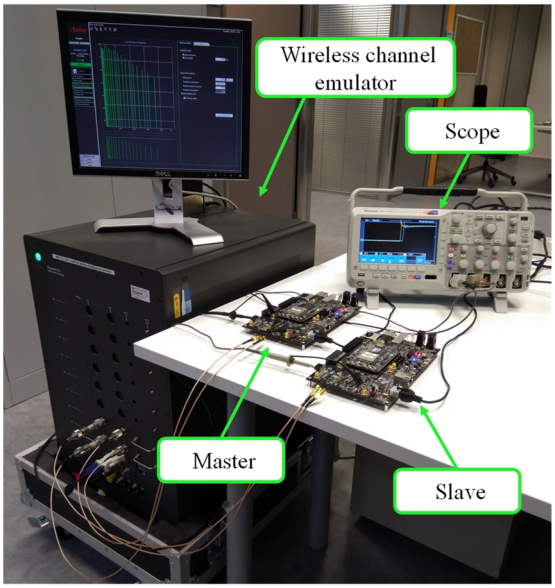


Figure 3.11. Experimental testbed to evaluate the time synchronization accuracy attainable with the conventional timestamps.

The experimental testbed comprises 4 main elements: two ADI RF SOM platforms that are programmed with the 802.11g modem, the Anite Propsim F8 Radio channel emulator [91], and the Tektronix MSO 2040B oscilloscope [92]. The main features of the channel emulator are summarized in Table 3.4. One of the boards was configured to be the master, while the second board was

configured to be a PTP slave. The RF input/output port of each board is connected to the wireless channel emulator. The PPS output of each board is connected to the oscilloscope, which has been used to evaluate the time synchronization error. The PPS output has a resolution of 50 ns, which means that the measured time synchronization error will be affected by a uniform random error of 50 ns.

Table 3.4. Characteristics of the *Propsim F8 Radio channel emulator*.

Characteristic	**Value**
BW [MHz]	40
f_c [MHz]	from 350 to 3000
Mobile speed (v) [km/h]	from 0.06 to 11000
Output Gain [dB]	From -100 to 0
Mean channel delay t_{ms} [µs]	3.1
Fading model	Rayleigh, Rice, Nakagami, log-normal, Gaussian, flat, constant
Doppler spectrum model	Jakes, Bell

Regarding the PTP configuration, the PTP frame exchange period has been set to 100 ms. Since this experiment uses the 802.11 MAC, T_{S-Dr} has not been configured. Still, as the nodes are connected to the emulator, the external interferences are minimal and thus the PTP answers are very fast. Finally, the PI filter of the PTP has been configured with the PTP standard K_i and K_p, which equals 0.1 and 0.3 respectively.

Since performing the experiments over the broad range of conditions used in the simulations would take significant efforts, only a small subset of conditions compared to the simulations have been used. Specifically, the SNR was set to 20 dB, the speed of the nodes have been set to 2, 10, 30, 100, and 300 km/h, and the channel emulator was programmed with all the channel models used in the simulation.

3.6.1.3. Early experimental validation setup

Since the enhanced timestamps have not been implemented in the 802.11 modem built over the ADI RF SOM platform, the validation of the time synchronization

performance cannot be fully addressed. Nonetheless, an early experimental validation setup has been used to evaluate the enhanced timestamps over the real HW, which is described through this subsection.

The validation setup of the enhanced timestamps uses two ADI RF SOM boards. The first board is configured as master, and it periodically transmits PTP sync frames every 500 µs. The second board was configured to be a slave, and its task was to receive the frames transmitted from the master to take the enhanced timestamps. The slave detects the frames, takes the conventional Rx timestamp, and calculates $R_{r,s}[n]$ and n_{ini}. Afterward, the timestamp, $R_{r,s}[n]$, and n_{ini} are transmitted to a computer, which calculates the enhanced timestamps correction factor. Since the correlator of the 802.11 modem comprises only 48 coefficients the CIR estimation quality is limited compared to the one used in the simulation, which has 128 coefficients.

Figure 3.12 depicts the measurement setup used to evaluate the enhanced timestamps. Both boards have been placed at a distance of approximately 1 meter and with a metal plate between them create an NLoS channel. During the experiment, the wireless channel did not vary, and hence the result of the enhanced timestamps should only be affected by the relative phase between the transmitter and receiver clock.

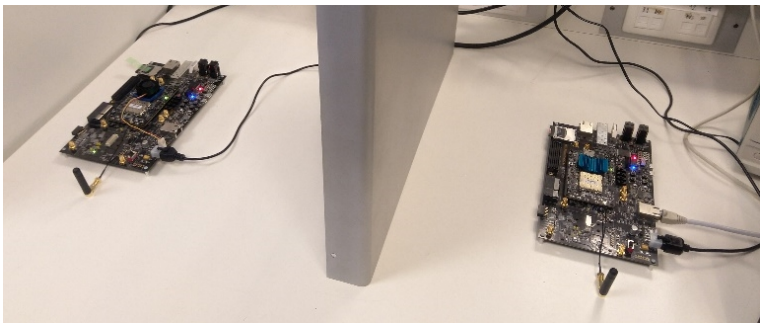


Figure 3.12. Measurement setup of the enhanced timestamps.

3.6.2. Evaluation of the time synchronization performance by simulation means

The results of the time synchronization accuracy are depicted in Figure 3.13 and Figure 3.14. The enhanced timestamps are labeled as (E. TS.) and the conventional timestamps are labeled as (C. TS.). The time synchronization accuracy is the combination of the precision and trueness of the measurement. Nonetheless, the trueness of the measurement is negligible compared to the precisions, and, as a result, it does not affect the accuracy. Thus, accuracy and precision match for the simulations shown in this section.

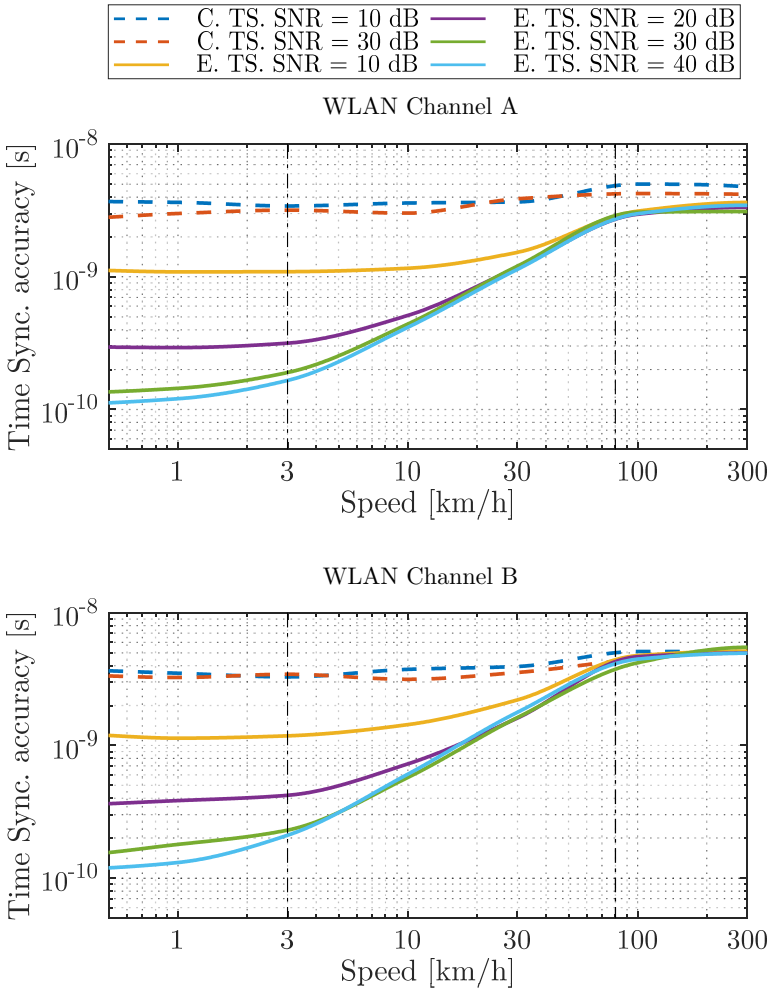


Figure 3.13. Time synchronization accuracy over WLAN channels A and B as a function of the speed of the nodes for different SNR and $T_{S\text{-}Dr} = 1$ ms.

Regarding the conventional timestamps results, three remarkable characteristics can be highlighted. First, their performance for SNR = 10 dB and 30 dB shows a very slight difference, thus their performance is not affected for low SNR conditions. Second, they show slight differences between each channel model because they have different multipath conditions. Finally, there is a noticeable difference when the nodes are moving at high speed. Still, the conventional timestamps show quite stable performance in every condition of the simulations. The main limitation of the conventional timestamps over 802.11g seems to be their poor resolution (50 ns) limited by the BW of the wireless system. Even with 50 ns resolution, the obtained results are far better than the required synchronization in most industrial applications, which shows that a simple timestamping method can be well enough for the implementation of industrial wireless networks.

Regarding the enhanced timestamps results, the first remarkable difference compared to the conventional timestamps is their performance: the enhanced timestamps provide from 1 to 2 orders of magnitude better accuracy than the conventional timestamps when running over the expected conditions. Three performance regions regarding the channel variation rate can be distinguished: slowly time-variant ($v < 3$ km/h), mid-time-variant (3 km/h $< v <$ 80 km/h), and fast time-variant ($v > 80$ km/h). The three regions are indicated by the vertical black dashed lines in the plots.

In the first region, the slowly time-variant region ($v < 3$ km/h), the channel coherence time T_{ch} is 30 times higher than T_{S-Dr}, thus the wireless channel can be considered symmetric during the PTP frame exchange. This is clearly represented in the results as the proposed synchronization scheme obtains a time synchronization accuracy better than 220 ps at an SNR of 30 dB over channels A and B, which is 25 times smaller than the achieved by the conventional timestamps under the same conditions. On the other hand, and compared to the results over channel A and B, the performance of the enhanced timestamps is slightly deteriorated in the simulation over channel C and severely deteriorated in the simulation over channel E. This is caused by the error of the correlation window. The number of CIR components increases and so it does the probability of misalignment in the integration window and the probability of missing some

CIR components in the integration. This causes a small error in the frame ingress time estimation, which is added to the time synchronization error. Furthermore, the performance bound over channels A and B, which is found at an SNR of about 30 dB, is caused by the error of the rounding operation in step 3.3 of the algorithm described in Subsection 3.3.4. Nonetheless, the bound is found for unrealistic SNR values unrealistic in wireless systems, and thus the use of more complex algorithms to implement the enhanced timestamps is not justified.

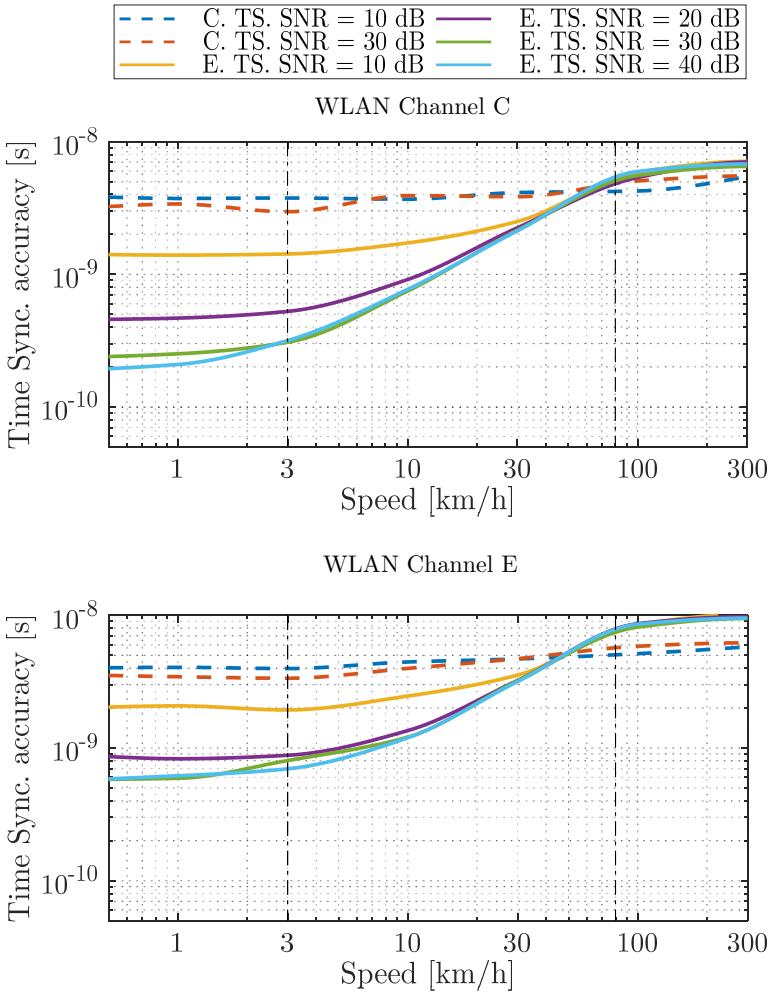


Figure 3.14. Time synchronization accuracy over WLAN channels C and E as a function of the speed of the nodes for different SNR and $T_{S\text{-}Dr} = 1$ ms.

In the second region (3 km/h $< v <$ 80 km/h), the synchronization performance linearly deteriorates as a function of the speed of the nodes. This is the expected behavior, as the channel coherence time is still very high, but it is not enough to consider a perfectly symmetric channel. Regarding the synchronization accuracy with conventional timestamps, it is not very affected by the changes in the environment, but their performance is still very far from the performance of the enhanced timestamps.

Finally, in the fast time-variant region ($v >$ 80 km/h) the synchronization performance of the conventional timestamps and the enhanced timestamps converges. This situation is reached at 80 km/h because the channel coherence time at such speed is approximately equal to $T_{S-Dr} = 1$ ms.

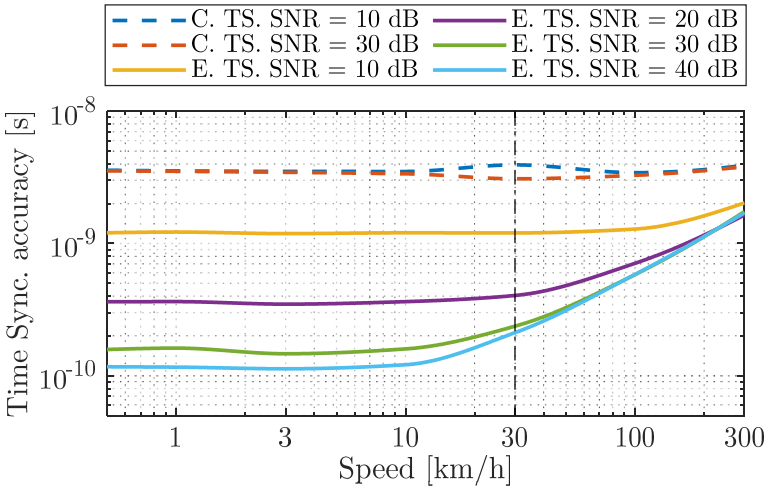


Figure 3.15. Time synchronization over WLAN channel B as a function of the speed of the nodes and the SNR. $T_{S\text{-}Dr} = 0.1$ ms.

To gain more insights about the relation between the channel variation rate and the value of T_{S-Dr}, one more simulation over the channel model B but using $T_{S-Dr} = 0.1$ ms has been carried out. The results of this simulation are depicted in Figure 3.15. The results show that there is a strong relationship between time synchronization performance and T_{S-Dr}. In fact, the threshold between the first and the second region is shifted to the right from 3 km/h to 30 km/h. Therefore, the requirement of channel symmetry during the PTP frame

exchange has been verified numerically, and it has been shown that T_{S-Dr} has a great impact in the synchronization performance. Thus, to ensure almost perfect channel symmetry and obtain appropriate time synchronization performance T_{S-Dr} should be 20 times lower than the channel coherence time.

3.6.3. Evaluation of the time synchronization performance by experimental means using the conventional timestamps

This subsection presents the time synchronization results obtained by means of the HW testbed described in Subsection 3.6.1.2. Table 3.5 summarizes the results in terms of precision σ and trueness μ of the time synchronization measured with the PPS of the master and slave nodes. In this case, the trueness is also reported since it is not negligible compared to the precision.

Note that K_i and K_p are significantly larger in the experimental setup than in the simulation setup and that the measurement resolution is 50 ns, whereas the simulation does not have a resolution bound. Consequently, the simulation results are necessarily better than the experimental results using the conventional timestamps. Even so, similar conclusions can be derived from the simulations and the experiments, as shown below.

Regarding the trueness (μ) results, μ is very close to 0 ns for each experiment. Since the delay of the channel emulator is in the range of 3 ms, the results demonstrate the capability of the PTP implementation to correctly compensate for the channel delay. The worst-case μ is found in WLAN C over the highest variation speed (300 km/h), where $\mu = -18.2$ ns.

Focusing on the precision results (σ), the experiments and the simulation show similar behavior for the conventional timestamps, but the experiments are around one order of magnitude worse than the simulations. These differences are caused by the resolution of the PPS, and by the use of a PI filter with a higher K_p and K_i constants. The first remarkable consideration that can be extracted from Table 3.5 is that the results are quite similar for every condition. For instance, the results over WLAN A, B, and C do not show any significant performance difference for any speed, with σ ranging between 30 ns and 40 ns.

Table 3.5. Clock synchronization trueness (μ) and precision (σ) over several wireless channels.

Speed [km/h]		**WLAN A**	**WLAN B**	**WLAN C**	**WLAN E**	**Gaussian**
0	μ **[ns]**	-	-	-	-	3.7
	σ **[ns]**	-	-	-	-	25.5
2	μ **[ns]**	3.70	8.82	8.96	-9.1	-
	σ **[ns]**	31.7	35.1	35.2	66.0	-
10	μ **[ns]**	11.2	-0.7	8.09	-10.7	-
	σ **[ns]**	27.8	30.9	30.6	44.3	-
30	μ **[ns]**	9.22	-1.20	-4.39	-5.8	-
	σ **[ns]**	28.4	29.9	28.4	43.7	-
100	μ **[ns]**	11.3	2.48	-5.10	-3.89	-
	σ **[ns]**	27.3	30.8	32.1	49.1	-
300	μ **[ns]**	-3.97	-8.9	-18.2	5.53	-
	σ **[ns]**	31.5	34.2	37.4	70.2	-

Besides, these results further validate the results over simulation means, that showed a very stable synchronization performance for almost any tested condition. The results over WLAN E are slightly worse compared to the other channel models. This difference is caused by the large number of packets lost during the experiments. The 802.11g modem is not prepared to operate in such high delay spread conditions, and, as a consequence, the PER is rather high, ranging from 10^{-2} to 10^{-1}. Even so, the results are quite similar to the results obtained in less challenging channels, having a worst-case precision of 70.2 ns over the highest speed of the nodes, and around 44 ns for medium node speed in the order of 10 to 30 km/h. Finally, it is also remarkable that the precision is better for mid speeds than for the lowest speed (2 km/h). In channels with very low variation speed, the channel impulse response may not vary during several hundreds of milliseconds. Consequently, if several frames in a row are affected by the same bad channel conditions, the overall time synchronization is deteriorated.

3.6.4. Early laboratory validation of the enhanced timestamps performance

The setup described in Subsection 3.6.1.3 has been used to perform the validation of the enhanced timestamps. A total of 20.000 sync frames were transmitted from the master to the slave during the experiment. The slave receives the sync frames, takes the timestamps, and calculates $R_{r,s}[n]$, which are sent to the computer to evaluate the enhanced timestamps.

Since the experiment was run over a static channel, the unique factor that influences the enhanced timestamps value is the variation of the phase between the master and slave clocks. The phase variation is caused by the slight difference in the frequency of the local oscillator of each board. If the frequency offset is considered constant, which can be assumed for a small period, then the frequency offset will generate a linear variation of the relative phase between the master and slave clocks. As a consequence, the enhanced timestamps should present a linear variation.

The mean delay spread ($\bar{\tau}$) for a subset of sync frames is depicted in Figure 3.16. According to the expectations, $\bar{\tau}$ has a very regular linear variation with some small imprecisions, such as, for instance, around the sample 50, where it can be seen some peaks in the variation. Besides, $\bar{\tau}$ also periodically wraps with a maximum range of 50 ns (equal to the clock period. The double glitch around sample n=100 is caused by the inherent imprecision of the threshold-based frame detector. Nonetheless, the enhanced timestamps algorithm is able to detect and then correct the error caused by the glitch.

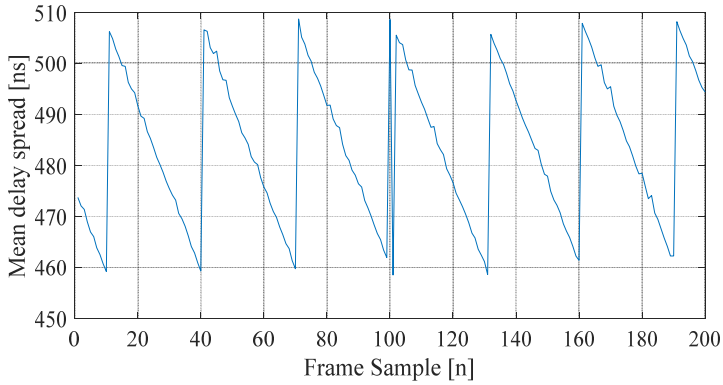


Figure 3.16. CIR mean delay spread as function of the frame ID (500 µs period).

Finally, the error between the true ToA and the estimated ToA has been estimated from the results of $\bar{\tau}$. To do so, the channel delay and the time drift t_d of the oscillators have been considered as constant. Hence, the time drift can be estimated from the enhanced timestamps and further corrected. In a real setup running PTP, the frequency offset would be corrected using the full PTP frame exchange.

The timestamping error is depicted in Figure 3.17. The absolute maximum error of the enhanced timestamps is below 2 ns, and the standard deviation equals 0.65. which makes the presented technique a promising approach for high-performance time synchronization over real-world wireless networks.

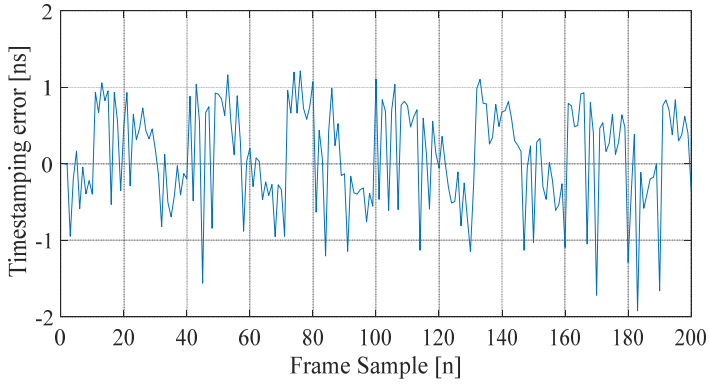


Figure 3.17. Enhanced timestamps error.

The implementation of the HW timestamps in the 802.11g platform is subject to future works. Even so, the results presented in this section highlight the potential of the enhanced timestamps over wireless channels. Besides, and as a promising application, the enhanced timestamps have been used in the next chapter to implement a full portable channel sounder.

4

Precise Channel Characterization in Industrial Environments

One of the firsts and critical steps to develop a wireless solution for an industrial application is the wireless channel characterization in the industrial scenario. The wireless channels present several impairments such as attenuation, fading, time dispersion, and time variation. These impairments can be analyzed through different methods, though it is quite common to use channel sounders to measure the wireless channel in the specific environment and develop a channel model from the measured data. However, channel sounders with synchronous operation (named through this thesis as full channel sounders) require perfect time synchronization to appropriately measure the wireless channel. Thus, their use in some industrial applications, such as mobile applications (UAV, terrestrial robots), is not straightforward.

In this chapter, the design, HW implementation, and validation of a full portable channel sounder that can be seamlessly used to precisely characterize the wireless propagation in industrial scenarios is presented. The channel sounder exploits the sub-nanosecond wireless time synchronization scheme described in Chapter 3. Thanks to the wireless time synchronization, the channel sounder does not require any wired connection, which provides significant advantages if compared to a conventional channel sounder, such as lower cost, simpler operation, and portability.

The chapter is organized as follows. The first section of this chapter describes the statistical tools commonly used to model the wireless propagation phenomena. The different approaches used to implement channel sounders based on their synchronization technique are overviewed in Section 4.2. Section 4.3 describes the design and implementation of the portable full channel sounder. The design comprises the channel sounder operation, and a post-processing algorithm to perform the time synchronization. The implementation has been carried out using the 802.11 modem implemented over a SoC-FPGA platform. Finally, Section 4.4 presents the experimental results and verification of the channel sounder through the use of a wireless channel emulator.

4.1. Principles of wireless channel modeling

This section describes the most common models used to statistically characterize wireless channels in SISO antenna configurations [50]. Statistical Channel modeling in Multiple-Input, Multiple-Output (MIMO) configurations is based on the same principles, though including space-correlation modeling.

4.1.1. Path loss

The path loss is the attenuation of a signal traversing through the space without any obstacle and dispersive/reflecting element. It is also commonly defined as the channel gain, the inverse of the path loss. The channel gain in an ideal free space scenario is characterized by the next expression

$$g_h = \frac{\mathrm{P}_{Rx}}{P_{Tx}} = \left(\frac{c}{4\pi df}\right)^2, \tag{4.1}$$

being f the wave frequency, d the distance between the transmitter and receiver, c the speed of light, and g_h the channel gain. g_h decreases for higher frequencies and larger distances.

4.1.2. Multipath propagation

Multipath propagation is a common phenomenon in wireless propagation that consist of several signal replicas echoed by different obstacles of the environment.

These replicas may arrive at different times to the receiver causing a time-dispersion of the signal. Time-dispersion refers to the fact that the power of the signal is dispersed over time.

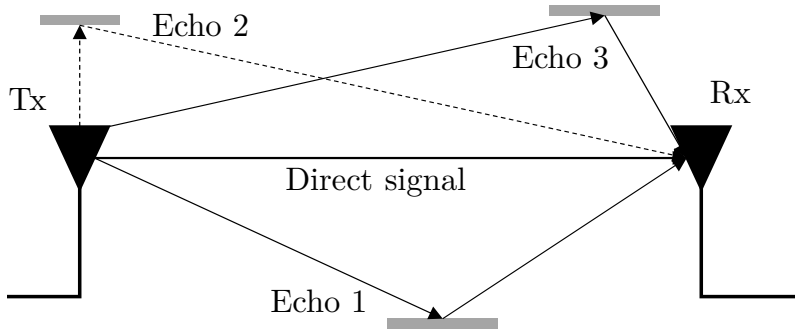


Figure 4.1. Multipath propagation due to reflective elements.

This phenomenon is represented by the CIR. The Tapped Delay Line (TDL) model [49] is one of the most commonly used due to its simplicity and precision in the representation of the CIR. The TDL model assumes several independent signal echoes with different gain and phases that are represented as a tap in the TDL model

$$h(t,\tau) = \sum_{i=0}^{I-1} c_i(t)\delta(\tau - \tau_i(t)), \tag{4.2}$$

being $h(t,\tau)$ the complex baseband equivalent CIR, τ the time in the delay domain, c_i the In-phase and Quadrature (IQ) amplitude of the ith tap, τ_i the delay of the ith tap, and I the number of taps. c_i and τ_i may change due to the changes in the environment and thus they are time-dependent. For the sake of simplicity in the model, τ_i is usually assumed constant, thus the CIR expression reduces to

$$h(t,\tau) = \sum_{i=0}^{I-1} c_i(t)\delta(\tau - \tau_i). \tag{4.3}$$

A generic $h(t,\tau)$ over the delay τ and time t domains is depicted in Figure 4.2. The taps of $h(t,\tau)$ are represented by vertical lines with different amplitudes $|c_i(t)|$ that present a variation over time and that are situated at different delays.

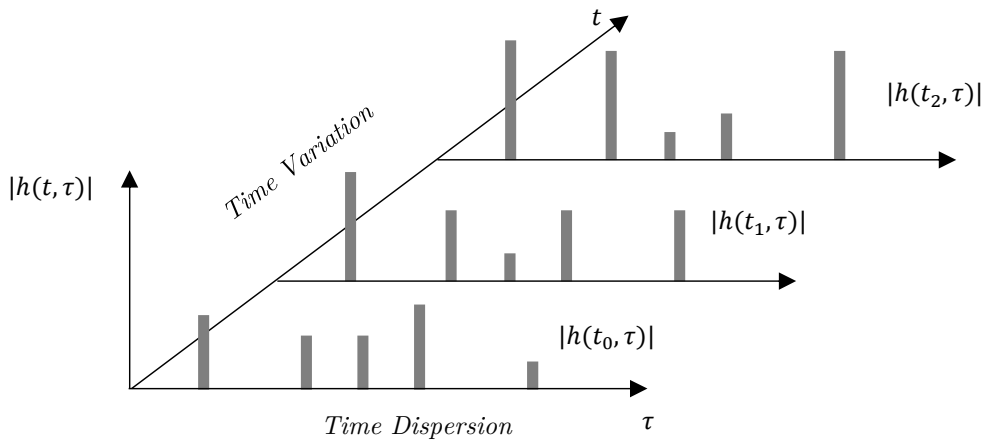


Figure 4.2. Representation of a generic CIR affected by time dispersion and time variation.

Due to the BW limitations of channel sounding devices, the CIR is not analyzed for the whole spectrum, but for a specific frequency band, and hence the measured CIR is a filtered version of $h(t,\tau)$. Ideally, the filter has a rectangular shape in the frequency domain where all the frequencies out of the desired band are removed and the desired frequency components are unaltered

$$p(t,\tau) = h(t,\tau) * \mathrm{g}(\tau), \tag{4.4}$$

being $\mathrm{g}(\tau)$ an ideal rectangular filter centered at the desired frequency and with a specific BW. Additionally, the CIR is typically characterized by a discrete version of $p(t,\tau)$ sampled over τ and t because channel sounders processing is usually implemented in the digital domain

$$p_i[n] = p(t,\tau)|_{t=iT_{cs}+nT,\tau=nT}, \tag{4.5}$$

being T_{cs} the time elapsed between the measurement of two instantaneous CIR, i the CIR index, and T the sampling period in the delay domain. The inverse of T typically equals $\mathrm{g}(\tau)$ BW. Note that $p(t,\tau)$ and $p_i[n]$ definitions are identical to the expressions use to model the channel in Chapter 3, but over time-variant channels. T_{cs} and T are two relevant performance indicators of a wireless channel sounder. A channel sounder with a small T_{cs} is able to measure channels with

high variation speeds, whereas a small T gives better resolution for the identification of different taps.

Through the next subsections, the most common statistical mechanisms to characterize $h(t,\tau)$ are described. These mechanisms are the ones used to evaluate the measurements of the wireless channel sounder presented in this chapter and verify its behavior. Note that the expressions described through the next subsections are identical for $h(t,\tau)$ and $p(t,\tau)$ and, in the case of its discrete version $p_i[n]$, the integrals are replaced with the sum operator.

4.1.3. PDP and RMS delay spread

The PDP represents the average power of the $h(t,\tau)$ as a function of the channel delay τ over a specific period

$$\mathrm{PDP}(\tau) = \int_{t=t_s}^{t_s+T_{\mathrm{PDP}}} \frac{1}{T_{\mathrm{PDP}}} |h(t,\tau)|^2 dt, \tag{4.6}$$

being t_s the start of the observation time, and T_{PDP} the observation time. The PDP is a convenient way to estimate the number of taps of the channel and the average power of each tap. From the PDP, the Root Mean Square (RMS) delay spread can be computed

$$\overline{\tau} = \frac{\int_{-\infty}^{+\infty} |\mathrm{PDP}(\tau)|^2 \ \tau \ d\tau}{\int_{-\infty}^{+\infty} |\mathrm{PDP}(\tau)|^2 \ d\tau}, \tag{4.7}$$

$$\tau_{rms} = \sqrt{\frac{\int_{-\infty}^{+\infty} (\tau - \overline{\tau})^2 |\mathrm{PDP}(\tau)|^2 \ t \ d\tau}{\int_{-\infty}^{+\infty} |\mathrm{PDP}(\tau)|^2 \ d\tau}}, \tag{4.8}$$

being $\overline{\tau}$ the mean delay spread as defined in Chapter 3 and τ_{rms} the rms delay spread. The rms delay spread represents the power spreading of the PDP. Low RMS delay spread means that most taps of the wireless channel are confined in a specific instant, i.e. it has low time dispersion, whereas large RMS delay spread indicates that the power of the taps is spread over the PDP duration.

4.1.4. CIR variation over time

The changes in the environment caused by the movement of the nodes or surrounding objects may introduce a variation of the phase and gain of the signal replicas. The variation of the signal replicas produces constructive and destructive interferences, which in turn produces a variation in the complex gain of each tap of the CIR. For simplicity, the taps are usually assumed as statistically independent, thus the variation of each tap is uncorrelated from the variation of other taps.

The tap variation is usually modeled from two complementary perspectives: fading modeling, and time-correlation modeling. The former represents the probability of the tap to have a specific gain, whereas the latter models the variation speed of the channel. The fading is modeled using statistical distributions derived from the physical properties of the channel, whereas the variation rate of the channel is modeled through the Doppler frequency caused by the relative speed of the channel.

4.1.4.1. Fading modeling

The fading distributions model the gain fluctuations of $c_i(t)$. To do so, it is assumed that each $c_i(t)$ is statistically independent because they are physically produced by different sets of signal replicas. Several distributions are used to model the channel fading, though the Rayleigh and Rice distributions are two of the most common distributions. The use of these distributions is derived from the properties of multipath propagation, which is described as follows.

Let us assume that, in a specific instant, several signal replicas with unitary gain and uniform random carrier phases between [0 and 2π], are detected at the antenna receiver. The sum of the signal replicas produces a random constructive/destructive interference depending on the phases of the replicas, which in turn produces a random gain and phase of $c_i(t)$. According to the Central Limit Theorem [49], and upon the condition of enough signal replicas with random phases, $c_i(t)$ can be modeled as a complex Gaussian distribution. The module of a random variable that follows a complex Gaussian distribution is the Rayleigh distribution (see Figure 4.3)

$$f_{Ray}(x) = \frac{x}{\sigma^2} \cdot e^{-x^2/2\sigma^2}, x > 0. \tag{4.9}$$

Specifically, this distribution is commonly found in NLoS channels or in LoS time-dispersive channels.

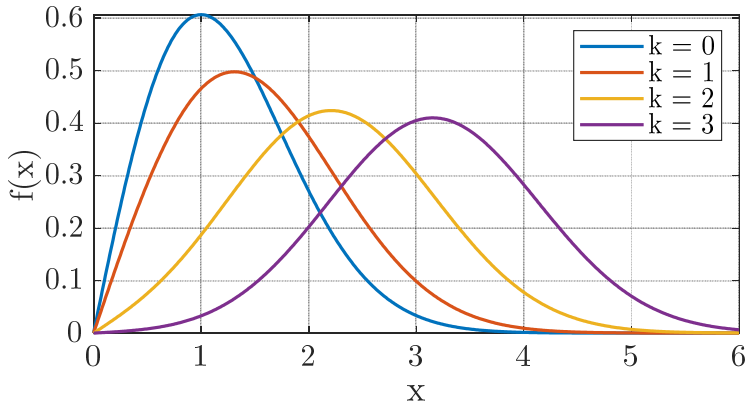


Figure 4.3. Rayleigh (k=0) and Rice probability density functions (pdf) as function of the tap amplitude module (x).

In the case that the channel presents a predominant LoS component, the $c_i(t)$ is modeled with a complex Gaussian distribution plus a constant value k that represents the LoS component. The module of the random variable is a Rician distribution (see Figure 4.3). k establishes the strength of the LoS that depends on the scenario. For instance, if the LoS is partially obstructed, k will be lower. Finally, the Rayleigh distribution is a particular case of Rice with $k = 0$, whereas Rice with $k \to \infty$ represents a constant $c_i(t)$ with no fading and no phase variation.

It is worth noting that, in LoS channels with several taps, the first tap $c_0(t)$ will be usually modeled with a Rice distribution, as the first tap represents the first channel component, whereas the rest of the taps will be usually modeled with a Rayleigh distribution.

4.1.4.2. Time-correlation modeling

The wireless channel variation is caused by changes in the physical properties of the wireless channel between the nodes, which can be caused by the movement

of the nodes themselves or by changes in the environment. These movements are modeled by a frequency shift of the fundamental carrier frequency f_c, usually called Doppler frequency shift f_d. The Doppler shift modifies the phase of the signal replicas causing changes in the complex gain of the channel. f_d depends on the carrier frequency f_c and on the variation speed of the wirelessly interconnected nodes and surrounding objects, and their relationship may be expressed as

$$f_d(t) = \frac{|\vec{v}(t)|f_c}{c}, \tag{4.10}$$

being f_d the Doppler shift, $\vec{v}(t)$ the vector that represents the nodes / surrounding element instantaneous relative speed, and c the speed of light in the transmission medium.

Since $\vec{v}(t)$ is typically modeled as a random variable, and consequently $f_d(t)$ also has a random nature. $f_d(t)$ is commonly characterized by the Doppler Power spectrum or simply named Doppler spectrum. The Doppler spectrum represents the spreadness of $f_d(t)$, and it is obtained from the computation of the Fourier transform (FT$\{\cdot\}$) of the CIR variation over time for a specific delay τ_0 [49].

$$H(\tau_0, f) = \mathrm{FT}\{h(\tau_0, t)\}, \tag{4.11}$$

being H the representation of h in the frequency domain, and f the frequency.

The Doppler spectrum presents different shapes depending on the movement of the scenario. Two canonical Doppler spectrum models are the Jakes model and the Bell model. The jakes model represents a static scenario with moving nodes, whereas the Bell model represents a scenario with static nodes and elements moving around the static nodes. Their shapes are depicted in Figure 4.4. As can be seen, the jakes Doppler power is condensed around the maximum doppler shift, which is equivalent to the maximum relative speed between the nodes. The Jakes doppler shape represents that, most of the time, the nodes are moving at their maximum relative speed. On the other hand, the Bell model has a gaussian-like rounded shape, because the movement of the surrounding elements follows a random pattern that follows a uniform distribution. In general,

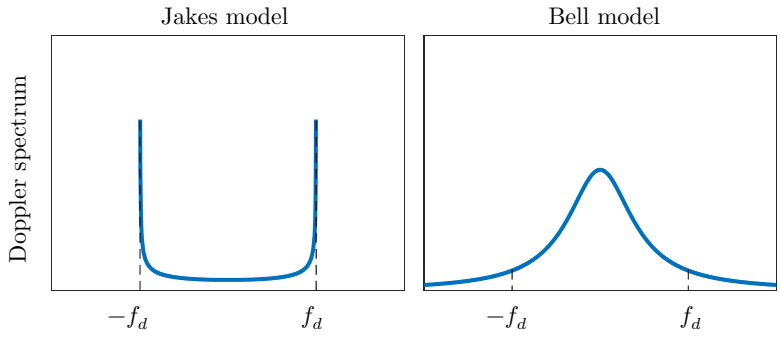


Figure 4.4. Shapes of Jakes and Bell Doppler spectrum models.

the Jakes model is present in scenarios with high variation speed, whereas the Bell model is present in scenarios with low variation speed.

4.2. Overview of the implementation of state-of-the-art channel sounders

Stochastic channel models are derived from the measurement of the CIR $h(t, \tau)$ (or its discrete/filtered version $p_i[n]$) in a specific environment. Channel measurement campaigns are performed using a wireless channel sounder. A Wireless channel sounder comprises two main elements: a master and a slave. The master task is to periodically generate and transmit a signal known by the slave. The signal traverses through the communication channel and reaches the slave, that detects and stores some parameters of the signal or the signal itself. From the prior-known information about the transmitted signal, the slave can estimate some parameters of the channel or the CIR itself.

The implementation of a channel sounder is quite complex and, depending on their design philosophy, they might not: i) offer a trustworthy-enough view of the wireless channel behavior or ii) be flexible enough to be used in an industrial scenario or other scenarios. Two kinds of channel sounders may be distinguished based on the information extracted from the channel: limited channel sounders, or full channel sounders:

- Limited channel sounders. Limited channel sounders [93] are designed to measure parameters such as the received power, the PER, or the system throughput. The information provided by limited channel sounders is not raw, i.e. they cannot directly measure $p_{\mathrm{i}}[n]$, and hence the knowledge that can be extracted from a measurement campaign is rather limited. They are commonly built with COTS devices, such as Universal Software Radio Peripheral (USRP) [94]. These channel sounders are preferred for mobile robotics applications, despite their lower performance, because of their size, weight, and cost.

- Full channel sounders. Full channel sounders are designed to measure the complex baseband CIR $p(t, \tau)$ or $p_{\mathrm{i}}[n]$ [95]. From $p_{\mathrm{i}}[n]$, relevant channel statistics, such as the PDP or the Doppler spectrum, can be obtained. Full channel sounders provide deeper information about the wireless channel though, compared to limited channel sounders, their implementation is more complex and more expensive, and their operation is less flexible because they require synchronization between the master and slave.

Through the next two subsections, the state-of-the-art techniques to build channel sounders and some examples of channel measurement campaigns are described.

4.2.1. Limited channel sounders

Limited channel sounders are designed to measure some indirect parameters of the wireless channel. Some of these parameters are the attainable PER of the specific wireless system, the wireless system throughput, and in some cases, some direct parameters of the wireless channel, such as attenuation of the channel or the PDP. Due to their lack of synchronization, they are not able to measure raw synchronized CIR samples. Limited channel sounders are usually built with low-cost COTS equipment, such as smartphones, small laptops with a wireless NIC (LTE / WiFi), or SDR equipment, such as USRP [94].

The literature about channel measurements using limited channel sounders in industrial scenarios is mainly focused on mobile robotics, such as UAV

communications. For instance, an extensive survey on channel measurements and modeling for UAV communications is presented in [52], where more than 20 measurement campaigns are analyzed using different channel sounders. Most of them are done with compact limited channel sounders. For instance, in [96] a measurement campaign using COTS 802.11 HW running in the 5 GHz band is presented. The measurements were carried out from air to ground in an open field with different flying altitudes. The measured variables were the received power (i.e. the channel gain) and the 802.11 throughput. [97] is another example of a measurement campaign using COTS HW. In this case, the authors used LTE running at 800 MHz, and the UAV was deployed in a rural area. The measured parameters where the PER and the fading probability.

As can be seen, these measurements, rather than provide a CIR measurement $p_i[n]$, they provide some secondary parameters that can be interesting to understand the behavior of the channel, though they do not exactly represent the channel itself.

4.2.2. Full channel sounders

Full channel sounders are designed to directly measure the CIR. There are several implementations of full channel sounders in the literature based on different techniques and with different features in terms of BW, maximum Doppler frequency shift, dynamic fading range, etc. Based on the technique used to measure the CIR, three main full channel sounders can be distinguished: Vectorial Network Analyzer (VNA)-based, time domain-based, or frequency domain-based. The common point of all of them is the requirement of time synchronization between the master and slave.

VNA-based sounders use a 2-port VNA and two antennas connected at each port. The VNA is configured to measure the S21 parameter, which is the impulse response from port 1 to port 2 (i.e. the CIR). The measurement is performed by transmitting small BW pulses at different frequencies to cover a large BW. These channel sounders are very appropriate for the measurement of very large BW (>10 GHz), but their operation is very slow, and they cannot be used to evaluate time-variant channels. As an example, in [98] P. Pajusco et al. presented an

extensive 3-dimension measurement campaign of the residential indoor channel using a VNA-based channel sounder. The channel sounder used in the measurement campaign has an effective BW of 10 GHz and an array antenna electronically controlled to provide beamforming capabilities. The measurement campaign took 130 hours for 32 static measurement locations, which represents the slow capture speed of VNA-based channel sounders.

Time domain and frequency domain channel sounders periodically transmit short signals and they are suitable to measure time-variant channels [99]. Their operation is as follows. The master and slave are perfectly synchronized, and their operation is synchronously triggered. When the master is triggered, it transmits a signal $s(t)$ with a specific waveform. $s(t)$ traverses the communication channel and it is received in the antenna of the slave $r(t)$. The slave samples the signal $r(t)$ and either stores the $r(t)$ for further post-processing or extracts $p_i[n]$ from $r(t)$ and $s(t)$.

An appropriate waveform for these channel sounders has the next properties: low autocorrelation outside of 0 (i.e. its autocorrelation must be similar to a Delta), low Peak to Average Power Ratio (PAPR), flat frequency response, and a duration smaller than the channel coherence time. These desirable properties are shared with the preambles of frames in most wireless systems (see Section 3.3.1). In the time domain, Pseudo Noise (PN) sequences are commonly used [100], whereas Zhadoff-Chu sequences [101] are chosen for frequency-based channel sounders because they provide minimum PAPR with flat frequency response.

The operation of the slave of time domain channel sounders is based on the cross-correlation of the received signal $r(t)$ with the transmitted signal $s(t)$. Upon the condition of autocorrelation approximately equal to a delta, the CIR can be extracted through the cross-correlation of $r(t)$ and $s(t)$. This operation is identical to the operation performed by a frame detector (see Section 3.3.2 eq. (3.17)).

$$\tilde{p}(t) = (r \star s)(t). \tag{4.12}$$

On the contrary, the operation of frequency domain sounders is as follows. The received signal $r(t)$ is converted from the time domain to the frequency domain

$$R(f) = \mathrm{FT}\{r(t)\}, \tag{4.13}$$

being $R(f)$ the representation of $r(t)$ in the frequency domain. Then, $p(t)$ is estimated in the frequency domain $\tilde{P}(f)$ by multiplying $R(f)$ with the conjugate of $s(t)$ in the frequency domain $S(f)^*$, being $(\cdot)^*$ the conjugate.

$$\tilde{P}(f) = R(f) \cdot S(f)^*. \tag{4.14}$$

Both approaches perform similar operations. The former estimates the CIR through a cross-correlation, whereas the latter uses a Fourier transform to estimate the CIR in the frequency domain. In general, frequency domain-based channel sounders are more simple to implement because the cross-correlation operation (4.12) has a high computation complexity, especially for long sequences. Still, their performance is similar since they basically perform the same operations but in different domains.

As an example of a time domain-based channel sounder, a PN sequence-based channel sounder is shown in [102]. This channel sounder is built with instrumentation equipment and offers a chip rate of 12.5 GHz and a CIR sampling period T_{cs} of 0.625 ms, which enables the measurement of fast time-varying channels. As an example of a frequency domain-based channel sounder, B. T. Maharaj et al. presented in [99] a frequency domain channel sounder with 100 MHz BW, f_c from 2 to 8 GHz, $T_{cs} = 3.2$ ms and MIMO support. The second channel sounder is built with low-cost components and thus it offers lower capabilities than the former.

4.2.3. Synchronization in full channel sounders

In order to measure $p_i[n]$, full channel sounders require synchronization between the master and slave. In state-of-the-art full channel sounders, the synchronization is typically delivered from the master to the slave through one or more wired connections. The time synchronization has two purposes: eliminate

Table 4.1. Different time synchronization approaches in conventional full channel sounders.

Class	Time Synchronization method	Oscillators	Advantages	Disadvantages
A	Permanent wired connection	At least one high-end oscillator	Perfect synchronization	Low distance and inflexible
B	wired connection during synchronization	Two high-end oscillators	Longer distances	Limited phase/frequency coherency
C	Totally non-wired connection	Two high-end oscillators	Simple operation, longer distances	Cannot measure true channel delay, limited phase/frequency coherency

the frequency offset of the local oscillators, and simultaneously trigger the master and slave operations.

Regarding the necessity of a wired connection between master and slave nodes, there are different implementations of full channel sounders that have been classified into 3 classes: A. permanent wired connection, B. wired connection during synchronization, and C. totally non-wired connection. Table 4.1 describes the characteristics of each time synchronization approach, whereas Table 4.2 classifies some relevant channel sounders in the literature based on their time synchronization technique.

As an example of the first type (permanent wired connection), a frequency domain-based full channel sounder with a BW of 100 MHz and MIMO support is presented in [99]. This channel sounder uses two rubidium clocks (one on each side), and a synchronization unit that simultaneously triggers the master and slave. The synchronization unit is connected by fiber to both the master and slave. The channel sounder presented in [102] has two connections between the master and slave: one connection to share the clock and the other one to synchronously trigger the transmitter and receiver. Other full channel sounders follow similar procedures sharing the clock through other signals. For instance, in [103] the master shares its local oscillator with the slave through a wired connection. Additionally, the master sends a trigger to the slave every time that a sequence is transmitted.

Table 4.2. Summary of state-of-the-art channel sounders based on their time synchronization.

Ref.	Year	Time Sync Class	Time Synchronization method	Clock type	Channel sounder type
[99]	2008	A	Permanent trigger connection	Two rubidium one at each side	Frequency domain
[102]	2015	A	One wire for the clock, one wire for triggering	One Rubidium	Time domain
[103]	2016	A	One wire to share OL, one wire for triggering	One Rubidium	Time domain
[95]	2017	B	One synchronization phase and one free-running phase	Two rubidium one at each side	Time domain
[104]	2015	C	Wired connection not required	Two rubidium one at each side	Time domain

In some applications, the channel sounder cannot have a permanent wired connection between the master and slave (e.g. mobility applications or long distances). In this case, a two-step procedure with a wired synchronization phase and a free-running phase is commonly used to eliminate the wired connection during the measurement campaign. In [95], another time domain-based channel sounder based on PN sequences is presented. The channel sounder provides a BW of 1 GHz and a fading range of 40 dB, enough for the measurement of most wireless channels. The channel sounder uses two rubidium clocks, one for the master and one for the slave. In the synchronization phase, the rubidium clock of the slave is connected to the rubidium clock of the master. This operation is used to synchronize the rubidium clock of the slave. Once they are stable and synchronized, the synchronization wire can be eliminated, and both clocks will slowly drift apart. The small drift of Rubidium clocks ensures that the system will be synchronized for a few hours.

Finally, some channel sounders do not require a wired connection at all. For instance, The channel sounder described in [104] uses two rubidium clocks (one in the master and one in the slave) that provide high-frequency stability, and it relies on the oscillator characteristics to maintain an adequate relative synchronization. Nonetheless, the channel sounder cannot measure the true channel delay since the master and slave are not synchronized to a common time,

the measured CIR will be coherent for less time, and it still uses expensive and weighty clocks.

Summarizing, limited channel sounders can be seamlessly used in industrial facilities and specifically in mobile applications because they are small, low cost, portable, and easy to use. On the downside, the knowledge extracted from a channel measurement campaign carried out with a limited channel sounder may not be enough to create a trustworthy channel model. On the other hand, full channel sounders provide an accurate characterization, though their operation is complex, and they are inflexible because they require synchronization.

4.3. Portable Full Channel Sounder based on Wireless Synchronization

The classification of channel sounders presented in the previous section shows the significant performance differences and capabilities among the two design philosophies: limited channel sounders are flexible, simple to operate, and portable, though they cannot provide raw measurements, whereas the opposite features are found in full channel sounders: they provide high-precision raw measurements, but they are inflexible, complex to operate, large, and weighty.

Given that the main difference between the design philosophies of channel sounders is related to synchronization, the use of a high-performance wireless time synchronization algorithm may stand as a reasonable trade-off between a limited channel sounder and a full channel sounder, upon the pre-condition of sufficient wireless time synchronization performance. High-Performance wireless time synchronization gathers the advantages of both designs: high-precision measurements from full channel sounders, and portability, operability, and flexibility from limited channel sounders.

In order to consider the measured CIRs as synchronized, the maximum acceptable error in the CIRs synchronization should be 20 - 50 lower than the CIR resolution T. As a result, for $T = 50$ ns (20 MHz BW), the time synchronization must be in the range of 1-2.5 ns, and for $T = 1$ ns (1 GHz BW),

a maximum error of 50 ps is required. The performance delivered by the time synchronization algorithm presented in Chapter 3 could be enough to fulfill the synchronization requirement of the channel sounder.

Through this section, the design and implementation of a full channel sounder based on the high-performance wireless time synchronization scheme proposed in Chapter 3 are described. The proposed design is detailed in Subsection 4.3.1 and the implementation of the channel sounder using the custom 802.11 modem built over the ADI RF SOM platform is described in Subsection 4.3.2.

4.3.1. Channel sounder design

Given the complexity from the implementation perspective of the time synchronization scheme, the design of the channel sounder has been divided into two parts: channel sounder HW operation and post-processing algorithm. On the one hand, the channel sounder HW is designed to operate without synchronization and its main task is to perform a PTP-like frame exchange to periodically collect several timestamps and CIRs. On the other hand, the post-processing algorithm receives the data from the frame exchange, estimates the time synchronization error, and synchronizes the estimated CIR to absolute time. The post-processing algorithm is run offline to reduce the HW complexity of the channel sounder.

4.3.1.1. HW operation

Full channel sounders commonly use a unidirectional communication, where the master periodically transmits a sequence and the slave detects the sequence. However, the time synchronization algorithm uses a bidirectional frame exchange, and thus a bidirectional communication is required. Due to the channel sounder has been implemented over an 802.11 modem, its design is described based on 802.11 terminology and features. Nonetheless, the design can be generalized to other wireless systems or custom solutions upon the condition of appropriate CIR estimation and timestamping capabilities.

The channel sounder performs the time synchronization and the channel sounding with a frame exchange similar to the IEEE 802.11 FTM exchange, which exploits the 802.11 ACKs to minimize the airtime consumption. The frame exchange is depicted in Figure 4.5, being i the index of the frame exchange, and its operation is as follows. The master transmits every T_{cs} a modified sync frame to the slave and takes the Tx timestamp ($t_{1,i}$). The modified sync frames are received by the slave, which takes the Rx timestamp ($t'_{2,i}$) and estimates the CIR from the modified Sync. Then, the slave answers to the modified Sync frame with a standard 802.11 ACK and takes the Tx timestamp ($t'_{3,i}$). Afterward, the master receives the ACK, takes the Rx timestamp ($t_{4,i}$), and estimates the CIR from the ACK. The frame exchange is indefinitely executed until the end of the measurement.

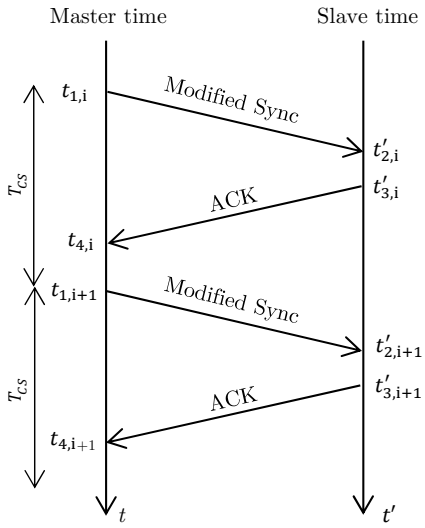


Figure 4.5. Modified PTP frame exchange with a reduced number of frames.

The modified sync frame joints the functionalities of the PTP sync and PTP delay response frames, whereas the ACK transmitted from the slave to the master acts as the PTP delay request. The structure of the sync frame is depicted in Figure 4.6. The frame comprises 254 bytes and has 5 data fields. The 802.11 header is used to be compliant with the 802.11 standard. The *Frame seq ID* field is used to identify the current frame, and it contains the channel sample index i.

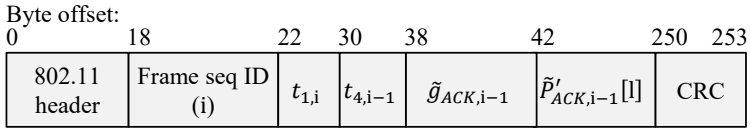


Figure 4.6. Modified Sync frame.

The Most Significant Bit (MSB) of the *Frame seq ID* field indicates if the ACK of the previous exchange was correctly received. The field $t_{1,i}$ contains the transmission timestamp of the sync frame. $t_{4,i-1}$, $\tilde{g}_{\mathrm{ACK},i-1}$, and $\tilde{P}'_{ACK,i-1}[l]$ are related to the previous ACK received by the master. $t_{4,i-1}$ contains the ACK timestamp, $\tilde{g}_{ACK,i-1}$ is the estimation of the channel gain in dB, which is computed as the estimated Rx power minus the Tx power in dBm, and $\tilde{P}'_{ACK,i-1}[l]$ is the estimated CIR of the ACK in the frequency domain normalized to unit energy. The ACK fields are set to 0 if the MSB bit of the *Frame seq ID* field is set to 0. $\tilde{P}'_{ACK,i-1}[l]$ is represented by its IQ components in complex format. It uses 4 bytes for each subcarrier and has a length $L = 52$. Finally, the standard 4-byte Cyclic Redundancy Check (CRC) is used to check if there is an error in the frame.

At the end of each frame exchange, the slave computes the estimated CIRs in the time domain as

$$\tilde{p}_{\mathrm{ACK},i}[n] = \mathrm{IFT}\left\{10^{\frac{\tilde{g}_{\mathrm{ACK},i}}{20}}\tilde{P}'_{\mathrm{ACK},i}[l]\right\}, \tag{4.15}$$

$$\tilde{p}_{\mathrm{Sync},i}[n] = \mathrm{IFT}\left\{10^{\frac{\tilde{g}_{Sync,i}}{20}}\tilde{P}'_{\mathrm{Sync},i}[l]\right\}, \tag{4.16}$$

being IFT$\{\cdot\}$ the Inverse Fourier Transform, $\tilde{P}'_{\mathrm{Sync},i}[l]$ the estimated CIR from the Sync frame normalized to unit energy, and $\tilde{g}_{\mathrm{Sync},i}$ the estimation of the channel gain. Then, the slave stores:

- The channel sample index, *Frame seq ID*: i.
- The four timestamps: $t_{1,i}, t'_{2,i}, t'_{3,i}, t_{4,i}$.
- The estimated CIR from the Sync: $\tilde{p}_{\mathrm{Sync},i}[n]$.
- The estimated CIR from the ACK: $\tilde{p}_{\mathrm{ACK},i}[n]$.

- A Boolean to indicate if the frame exchange was correctly performed: $\dot{x}_i$.
- A Boolean to indicate if the sync frame was correctly received: $\dot{s}_i$.
- Additionally, the slave also stores the number of frame exchanges (I), the carrier frequency (f_c), and T_{CS}.

4.3.1.2. Post-processing algorithm

The offline time synchronization algorithm is used to align $\tilde{p}_{\text{Sync}}$ to an absolute time reference and to compensate the local oscillator drift. The post-processing is divided into several steps, which may be classified into three main tasks: compute the enhanced timestamps, apply the PTP correction algorithm, and correct $\tilde{p}_{\text{Sync}}$. The first two tasks are summarized in Algorithm 4.1 and further explained through the next paragraphs. This algorithm is based on the one described in Subsection 3.1.5.

The algorithm output is the time synchronization error ($\tilde{t}_{err,\text{i}}$) and the corrected conventional timestamps ($t'_{c2,i}$). The algorithm uses the enhanced timestamps combined with a PI filter with constants (k_p, k_{int}) to minimize the time synchronization error. $t_1, t'_2, t'_3, t_4, \dot{x}$ are vectors with length I, and $\tilde{p}_{\text{ACK}}$ and $\tilde{p}_{\text{Sync}}$ are matrices with I rows and $N = 52$ columns. The algorithm performs one iteration for each frame exchange (i).

Algorithm 4.1. Offline time synchronization algorithm.

Input: $\tilde{p}_{\text{ACK}}, \tilde{p}_{\text{Sync}}, t_1, t'_2, t'_3, t_4, \dot{x}_i, k_p, k_{Int}, I$
Output: $\tilde{t}_{err,i}, t'_{c,2}$
Initialization:
$t_{\text{Int}} = 0$.
$\tilde{t}_{acum\ error} = 0$.

1. for i = 0 to I-1 do:
   2. Correct slave clock timestamps:
      2.1. $t'_{c,2,i} = t'_{2,i} - \tilde{t}_{acum\ err}$.
      2.2. $t'_{c,3,i} = t'_{3,i} - \tilde{t}_{acum\ err}$.

   3. if $\dot{x}_i$ = false do:
      3.1. $\tilde{t}_{err,i} = k_{Int} t_{\text{Int}}$.
      3.2. $\tilde{t}_{acum\ err} = \tilde{t}_{acum\ err} + \tilde{t}_{err,i}$.
      3.3. Go to the next iteration.

   4. Take the enhanced timestamps:
      4.1 $t'_{e,2,i} = t'_{c,2,i} + \text{DS}\{\tilde{p}_{\text{Sync},i}[n]\}$.
      4.2 $t_{e,4,i} = t_{4,i} + \text{DS}\{\tilde{p}_{\text{ACK},i}[n]\}$.

   5. Compute Time Sync error:
      5.1. $\tilde{t}_{ms,i} = (t'_{e,2,i} - t_{1,i} + t_{e,4,i} - t'_{c,3,i})/2$.
      5.2. $\tilde{t}_{o,i} = t'_{e,2,i} - t_{1,i} - \tilde{t}_{ms,i}$.
      5.3. $t_{Int} = t_{Int} + \tilde{t}_{o,i}$.
      5.4. $\tilde{t}_{err,i} = k_p \tilde{t}_{o,i} + k_{Int} t_{Int}$.
      5.5. $\tilde{t}_{acum\ err} = \tilde{t}_{acum\ err} + \tilde{t}_{err,i}$.
6. end for
7. return

In step 2., the slave timestamps $t'_{2,i}$ and $t'_{3,i}$ are corrected based on the accumulated time synchronization error ($\tilde{t}_{accum\ err}$). The result is the corrected conventional timestamp ($t'_{c,2,i}$) and the corrected transmission timestamp ($t'_{c,3,i}$) after applying the time synchronization algorithm.

In step 3., the algorithm checks if the i frame exchange was successfully performed. In case that $\dot{x}_i = false$, it will mean that one of the frames in the i frame exchange failed. In this case, the time synchronization error ($\tilde{t}_{o,i}$) is

assumed to be 0, and $\tilde{t}_{accum\ err}$ is updated only with the integral part of the PI filter. Then, the algorithm skips the iteration.

In step 4., The enhanced timestamps $(t'_{e,2,i}, t_{e,4,i})$ are computed. To do so, the conventional timestamps $t'_{c,2,i}$ and $t_{4,i}$ are corrected by the mean delay spread (DS$\{\cdot\}$) of $\tilde{p}_{\mathrm{Sync},i}[n]$ and $\tilde{p}_{\mathrm{ACK},i}[n]$ respectively. DS$\{\cdot\}$ corresponds with the next expression

$$\mathrm{DS}\{p[n]\} = T\frac{\sum_{n=n_s}^{n_s+N-1}|p[n]|^2 \cdot n}{\sum_{n=n_s}^{n_s+N-1}|p[n]|^2}. \tag{4.17}$$

Note that n_s and N are reconfigurable based on the measured channel, and that more than one iteration can be performed for improved performance as detailed in Subsection 3.3.4.

In step 5. the PTP synchronization algorithm is used to compute the time synchronization error. First, $\tilde{t}_{ms,i}$ and $\tilde{t}_{o,i}$ are obtained. Then, $\tilde{t}_{o,i}$ is introduced to a PI filter with constants k_p, k_{Int} and the error of this frame exchange $(\tilde{t}_{err})$ is computed. $\tilde{t}_{err}$ is accumulated in $\tilde{t}_{acum\ err}$.

After the computation of the time synchronization error, the algorithm aligns the estimated CIRs obtained from the Sync frames and corrects the carrier frequency drift. To align $\tilde{p}_{\mathrm{Sync,i}}[\mathrm{n}]$, the CIRs must be shifted by a delay d_{i} using a fractional delay filter. d_{i} can be obtained as the time difference between the detection of the sync frames $(t'_{c,2,i})$ and its transmission $(t_{1,i})$

$$d_i = (t'_{c,2,i} - t_{1,i}). \tag{4.18}$$

In addition, the carrier frequency offset in each frame exchange $(\tilde{f}_{o,i})$ can be estimated from $\tilde{t}_{err,i}$

$$\tilde{f}_{o,i} = \frac{\tilde{t}_{err,i}}{T_{CS}} f_c. \tag{4.19}$$

Then, the relative carrier frequency phase between the master and slave $(\tilde{\phi}_i)$ is

$$\tilde{\phi}_i = \tilde{f}_{o,i} \cdot T_{CS} = \tilde{t}_{err,i} \cdot f_c. \tag{4.20}$$

Given $\tilde{\phi}_i$ and d_I, $\tilde{p}_{\text{Sync aligned},i}$[k] is computed using a fractional delay filter as

$$\tilde{p}_{\text{Sync aligned},i}[k] = e^{-j\tilde{\phi}_i} \sum_{n=-\infty}^{+\infty} \tilde{p}_{\text{Sync},i}[n] \,\text{sinc}\left(k - n - \frac{1}{T_s} d_i\right). \tag{4.21}$$

being

$$\text{sinc}(y) = \frac{\sin(\pi y)}{y}. \tag{4.22}$$

Eq. (4.21) performs two operations. It shifts the trigger from the frame detection instant ($t'_{c,2,i}$), which depends on the CIR and the frame detector, to the frame transmission instant ($t_{1,i}$), and it eliminates the carrier frequency offset. Hence, upon enough time synchronization performance, the channel samples of this channel sounder are equivalent to the operation of a full channel sounder, where the master periodically transmits a signal and triggers the slave at the same time to measure the received signal and extract the CIR.

Finally, if a sync frame in a specific frame exchange (i_0) was not correctly received ($\dot{s}_i = false$), the CIR from that sample is lost. Still, $\tilde{p}_{\text{Sync aligned},i_0}[k]$ may be interpolated from the adjacent $\tilde{p}_{\text{Sync aligned},i}[k]$ using, for instance, a linear interpolation

$$\tilde{p}_{\text{Sync aligned},i_0}[k] = \frac{\tilde{p}_{\text{Sync aligned},i_0-1}[k] + \tilde{p}_{\text{Sync aligned},i_0+1}[k]}{2}. \tag{4.23}$$

4.3.2. Channel Sounder implementation

The channel sounder comprises three main elements a master, a slave, and a computer. The master periodically transmits the modified sync frames, the slave receives the modified sync frames from the master and answers with ACKs, and the computer receives the frame exchange data from the slave. In a channel measurement campaign with mobile devices, the frame exchange data could be stored in the non-volatile memory of the slave. The channel sounder nodes have been built using the 802.11 modem implemented over an SoC FPGA-based SDR

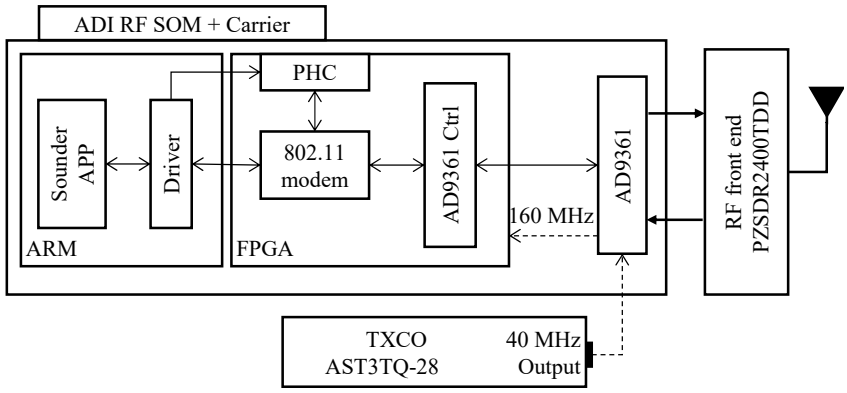


Figure 4.7. Block Diagram of a node of the channel sounder.

platform. This 802.11 modem has been also used to evaluate the time synchronization performance in Chapter 3. The engineering efforts of the implementation are mainly found in the implementation of the channel sounder node (FPGA HW and control SW), which are described through the next two subsections.

HW platform

The HW of the master and slave is identical, and its behavior is configured through SW. The block diagram of one node is shown in Figure 4.7. The HW implemented in the FPGA comprises the PHY layer of the 802.11 modem, the triggering and timestamping clock, and the interfaces with the AD9361 radio and with the SW. The SW performs the initial HW configuration, collects the data from the HW (CIRs, timestamps, etc.), and sends the data to the computer. The timestamping units used for the HW operation of the channel sounder and the PHC are the ones described in Section 3.5. The main modification here compared to the 802.11 modem of Chapter 3 is that the transmission of the sync frames is deterministically triggered by the PHC to ensure that the frames are transmitted every T_{cs}.

Apart from the ADI RF SOM platform, some external extra HW has been added to the testbed. First, the internal oscillator of the ADI RF SOM platform has a frequency stability of 25 ppm. Based on previous measurements (not reported in the document), the stability of this clock is not enough to measure

the phase of the CIR. This oscillator has been replaced with the Temperature Controller Crystal Oscillator (TCXO) AST3TQ-28 from Abracon [90]. This oscillator has a frequency stability of 0.28 ppm and a jitter of 0.4 ps, significantly better than the internal oscillator of the ADI RFM SOM platform. The AD-PZSDR2400TDD-EB RF front end has been also used. The RF front end is designed to operate in the 2.4 GHz band and contains a high-gain power amplifier, a low-noise amplifier, an RF switch, and Tx and Rx filters centered in the 2.4 GHz band. The output power P_{Tx} of the channel sounder is 10 dBm, which is enough to perform channel measurements.

Control SW

The control SW is implemented over a Linux operating system with kernel version 4.14. The Linux sources are provided by Analog Devices for the ADI RF SOM platform. The SW comprises three elements, the master driver, the slave driver, and the slave application. The master driver operation performs the master initialization and manages the data from the ACK. The slave driver collects the data of the frame exchange and transmits the data to the slave application. Finally, the slave application receives the data from the slave driver and transmits it through Ethernet.

The master driver initializes the HW to send a sync frame every T_{cs}. The frame transmission is commanded in HW and the frames are deterministically sent without the intervention of the driver. After one frame is transmitted, the 802.11 modem generates an interruption. In the interruption, the driver erases the data of the buffer of the previous frame and writes the timestamp $t_{1,i}$ and the *Frame seq ID i* for the next sync frame. If an ACK frame is received, another interruption is generated. Then, the driver extracts from the HW $\tilde{P}'_{\mathrm{ACK},i}[l]$, the timestamp ($t_{4,i}$), and computes $\tilde{g}_{\mathrm{ACK},i}$. Finally, it writes the information to the transmission buffer of the next Sync frame.

The slave driver is in idle state until an interruption is received. The interruption is generated every time that a frame is correctly received. The ACK is automatically generated by the HW without any intervention of the driver. The slave driver collects the information generated during the frame exchange: the *Frame seq ID i*, the timestamps $(t_{1,i}, t_{2,i}, t_{3,i}, t_{4,i})$, $\tilde{p}_{\mathrm{ACK},i}[n]$, and $\tilde{p}_{\mathrm{Sync},i}[n]$.

Besides, and based on the frames that have been successfully delivered, the slave calculates the $\dot{x}_i$ and $\dot{s}_i$ Booleans. With this information, it creates a data structure that is sent to the application.

Finally, the slave application waits in idle state until the driver sends a data structure with the data of the frame exchange Then, it generates an Ethernet frame, which contains the data structure, which is transmitted to the computer. In a real channel measurement campaign, the application may also store the data in the SD card of the Picozed.

4.4. Experimental results

The wireless channel sounder design and HW architecture presented in the previous sections have been implemented in the HW platform. The wireless channel sounder has been used in two scenarios: in the lab using a wireless channel emulator programmed with a wireless channel with mobility and in a mechanical workshop over static conditions.

4.4.1. Results over a wireless channel emulator

The channel sounder has been in-lab verified using a custom wireless channel emulator built by the communication systems team of Ikerlan. A previous version of the channel emulator is described in [56]. The channel emulator operates in the 2.4 GHz band and has a BW of 100 MHz. It has four bidirectional ports with a delay from port to port of 910 ns, and a minimum fixed attenuation of 44 dB. The RF circuitry of the channel emulator is designed to operate in the 2.4 GHz band, but it can be replaced to adopt other wireless bands. The emulator can hold a wireless channel model up to 10 taps that can be programmed with different gains and fading profiles. It supports four fading distributions: Rayleigh, Rice, Nakagami and constant; and three Doppler spectrum shapes: Jakes, Bell, and Gaussian. Note that this channel emulator is not the one used to evaluate the time synchronization performance of the conventional timestamps (Section 3.6).

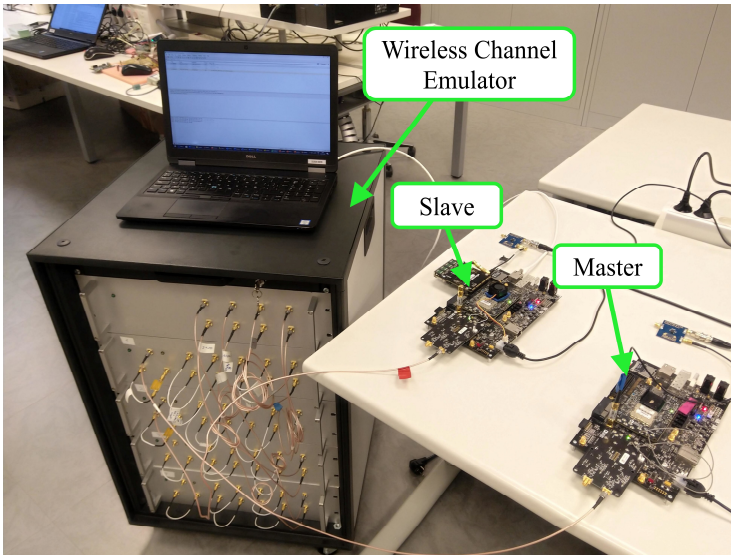


Figure 4.8. Channel sounder verification testbed.

The verification testbed is shown in Figure 4.8. The master RF port is connected to the first port of the channel emulator, and the slave RF port is connected to the second port. The Ethernet port of the slave board is connected to a laptop, which is used to record the data from the frame exchanges. f_c has been set to 2.48 GHz, T_{CS} to 500 µs, and P_{Tx} to 10 dBm.

The channel emulator was programmed with the wireless channel model described in Table 4.3. The channel model emulates an NLoS scenario with mobile nodes. It comprises three taps with different gains and evenly spaced 195.3 ns. The fading has been set to Rayleigh (NLoS), and the Doppler spectrum has been set to Jakes with a Doppler frequency of 62 Hz. The Doppler configuration represents the movement of the nodes at a maximum relative speed of 7.5 m/s given that $f_c = 2.48$ GHz. The channel model described in Table 4.3 includes the fixed attenuation of the channel emulator (44 dB), and its delay (910 ns).

Table 4.3. Emulated wireless channel model.

	Tap 1	**Tap 2**	**Tap 3**
Gain [dB]	-44	-54	-47
Delay [ns]	910	910 + 195.3	910 + 390.6
Fading distribution	Rayleigh	Rayleigh	Rayleigh
Doppler spectrum	Jakes with a maximum Doppler shift of 62 Hz.		

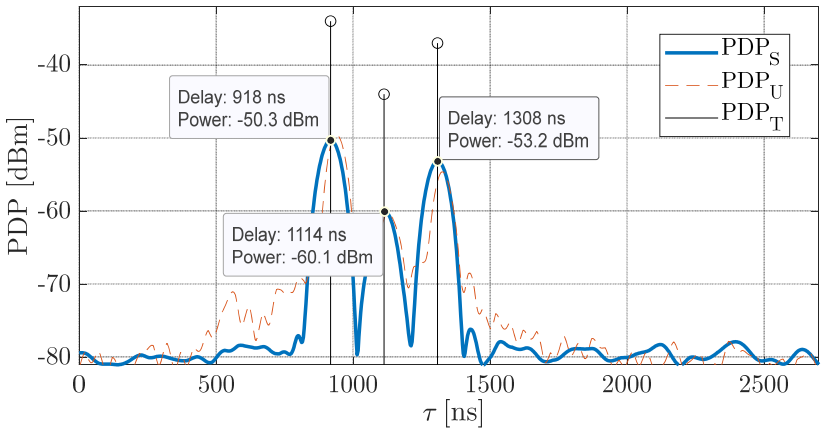


Figure 4.9. PDP vs. channel delay (τ) for synchronized PDP (PDP_S), unsynchronized PDP (PDP_U), and theoretical PDP (PDP_T).

The experiment was run for a total of 20 seconds, equivalent to $I = 40.000$ CIR. The record length is enough to measure the tap position, gain, and fading properties. During the record, the wireless channel randomly varied according to the programmed configuration.

The data obtained from the measurement were introduced to the post-processing time synchronization algorithm, which output was $\tilde{p}_{\text{Sync aligned}}$. The PDPs obtained from the $\tilde{p}_{\text{Sync aligned}}$ and from $\tilde{p}_{\text{Sync}}$ along with the theoretical PDP are depicted in Figure 4.9. The PDPs have been filtered with a Hanning window [50] to reduce their secondary lobes. $\tilde{p}_{\text{Sync}}$ PDP has been manually aligned to $\tilde{p}_{\text{Sync aligned}}$ PDP.

There are significant differences between the PDP with and without alignment. Regarding the $\tilde{p}_{\text{Sync}}$ PDP, the position of the taps randomly varies

around their real position in $\tilde{p}_{\text{Sync}}$. Due to this, the PDP has a significant tail around the taps and a considerable error in the relative gain between taps (2 dB). On the contrary, the $\tilde{p}_{\text{Sync aligned}}$ PDP is smooth without tails, the relative gain between taps have an error below 0.5 dB, and their relative delay is very accurate. In addition, the difference between the peak and the noise floor is around 30 dB, which gives a reasonable PDP dynamic range.

From the $\tilde{p}_{Sync}$ PDP representation, it is clear that the wireless channel presents 3 taps, which are situated at 918 ns, 1114 ns, and 1308 ns. The gain and phase of the taps randomly vary over time following the model programmed in the channel emulator (jakes with 62 Hz and Rayleigh fading).

The Doppler spectrum shows the variation speed of the magnitude of each tap, and it is represented by the spectrum of the gain and phase of each of the taps [49]. The Doppler spectrum of each of the taps is depicted in Figure 4.10. It is clear from the figure that, the Doppler spectrum of the three taps follows a Jakes Doppler distribution with a maximum Doppler shift of 65.7 Hz. The Jakes Doppler spectrum is not perfectly centered because the local oscillator drift was not completely compensated. Still, the frequency error is quite low, in the range of 3-4 Hz. If the remaining frequency offset is manually compensated, the maximum Doppler shift is 62.1 Hz, very close to the Doppler shift programmed in the channel emulator.

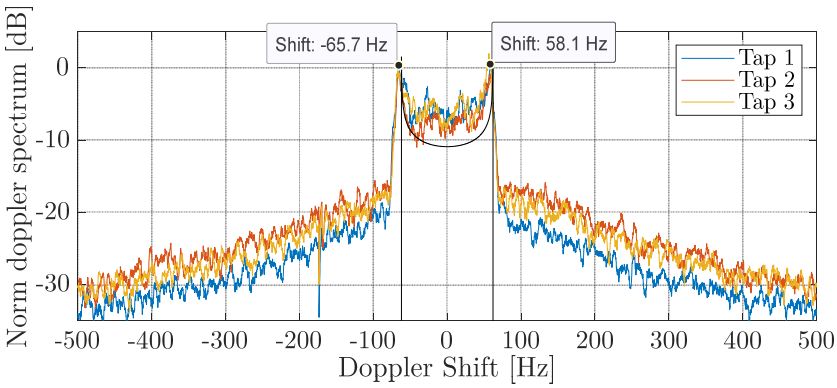


Figure 4.10. Measured Doppler spectrum of Taps 1, 2, and 3.

Regarding the fading results, Figure 4.11. depicts the fading distribution of each tap along with the theoretical Rayleigh distribution. It is noticeable from the plot that the amplitude of each tap approximately fits a Rayleigh distribution at their respective power. These results confirm the feasibility of the channel sounder to measure raw channel samples in static and mobile conditions.

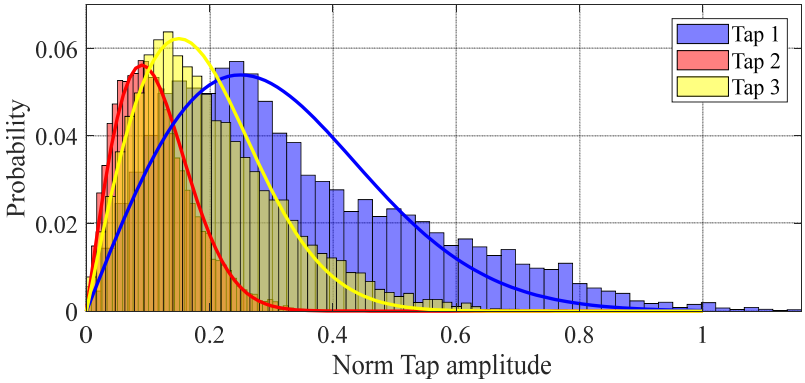


Figure 4.11. Measured fading distributions of Taps 1, 2, and 3.

4.4.2. Results in a workshop

After the verification of the channel sounder using the wireless channel emulator, the channel sounder has been used to measure a wireless channel of a mechanical workshop (Figure 4.12). The mechanical workshop has been also used in the experiments to test the performance of w-SHARP (see Section 5.5). The nodes were placed as shown in Figure 4.12 in a static position, at a distance around 13 m, and with several elements blocking the LoS. The channel sounder was programmed with $f_c = 2.58$ GHz, $T_{CS} = 500$ µs, and P_{Tx} was set to 10 dBm. f_c has been set outside the 2.4 GHz band to avoid interferences from nearby Wi-Fi networks. The workshop operators were working during the measurements, but there was no movement of large machines.

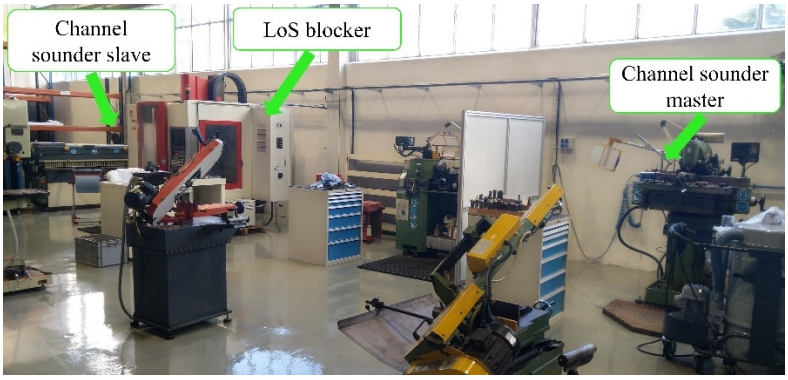


Figure 4.12. Measurement setup in the workshop at Ikerlan.

This experiment represents a nearly static industrial channel since the nodes were placed in a fixed position and only the operators of the workshop were moving around the machines. Consequently, a wireless channel with low variations is expected. The experiment was run for a total of 5 mins and a total of $I = 600.000$ CIRs were taken. The record length is enough to measure the tap position, gain, and fading properties and capture the small variations of the wireless channel.

As in the experiments with the channel emulator, the raw data obtained from the experiment was post-processed by the offline time synchronization algorithm. The PDPs obtained from the $\tilde{p}_{\text{Sync aligned}}$ and from $\tilde{p}_{\text{Sync}}$ are depicted in Figure 4.13 with a time resolution of 3.25 ns. Based on the PDP results, it can be concluded that the measured wireless channel presents two taps, one tap situated at a delay of 61 ns and with a peak power of -70 dBm (main tap), and a second tap situated at 233 ns with a power of -82.5 dBm. The main tap probably corresponds with the reflection of the ceil of the workshop, whereas the second tap could be the combination of reflections from several directions. The RMS delay spread of the PDP is 72 ns, which is a quite low spreadness for industrial scenarios, which are characterized by significant multipath. The reason for the low spreadness is that the main tap, which is caused by the ceiling, has a significant power compared to the next tap.

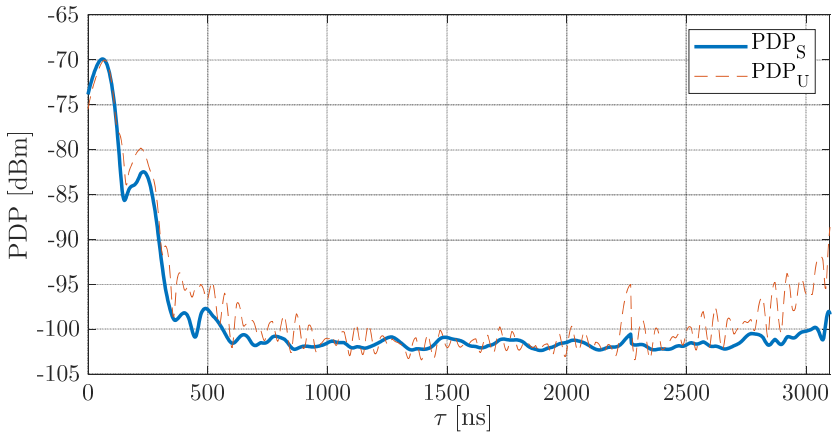


Figure 4.13. PDP vs. channel delay (τ) for synchronized PDP (PDP_S) and unsynchronized PDP (PDP_U) measured at the workshop.

The Doppler spectrum distributions for both taps are depicted in Figure 4.14. Note that Tap 1 is the tap at 58 ns and Tap 2 is the tap at 233 ns. The Doppler spectrum distributions present a very narrow shape, with a maximum deviation of 6 Hz from the peak with a small deviation from the center of 5 Hz due to a small imprecision of the time synchronization algorithm. The Doppler spectrum shape clearly represents a channel with a very low variability. If compared to the Doppler spectrum found in the experiments of the previous subsection, the differences are notorious. The former was measured over a channel with mobility and clearly follows a Jakes shape with a considerable maximum deviation whereas the one of this experiment has a very narrow Bell shape because the channel variation was very small.

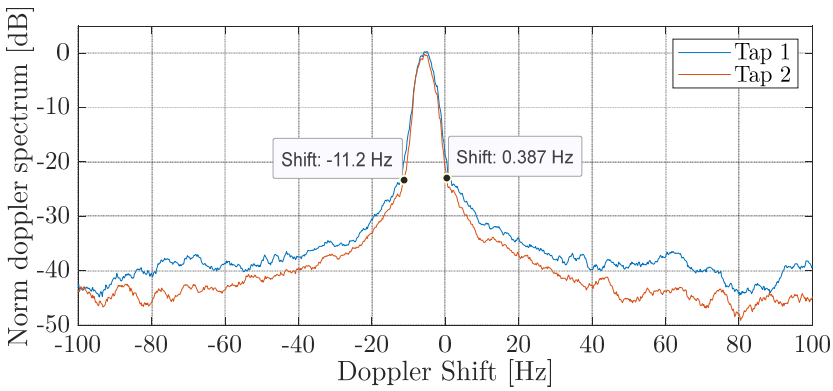


Figure 4.14. Measured Doppler spectrum at the workshop.

Finally, the fading distributions for both taps are represented in Figure 4.15. As in the Doppler spectrum, the taps do not present a significant power variation over time. On the one hand, tap 1 has a maximum power deviation of 3 dB for the whole measurement, which is a reasonable result because it is the strongest tap and it is caused by a large reflection in the ceiling. On the other hand, the second tap presents a larger variation of around 5 dB of maximum deviation because the tap comprises several signal reflections. Still, it is very stable due to the low channel variability. The taps do not follow any specific distribution since the channel variation is very low.

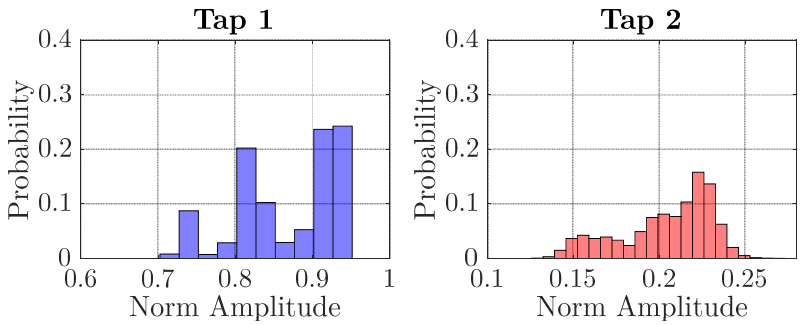


Figure 4.15. Measured fading distributions for tap 1 and 2 at the workshop.

5

Synchronous and Hybrid Architecture for Real-time Performance (SHARP)

This chapter presents the Synchronous and Hybrid Architecture for Real-Time Performance (SHARP), a high-performance hybrid communications system specifically designed to tame the strict requirements of industrial applications. SHARP comprises a wired segment built with the TSN technology and a wireless segment based on a wireless technology named w-SHARP, which has been developed in the context of this thesis. w-SHARP is a high-performance wireless system based on IEEE 802.11 and specifically designed to tame the strict requirements of industrial applications. The w-SHARP protocol follows the principles of TSN and this makes w-SHARP a suitable candidate for wireless TSN. The w-SHARP protocol has been implemented in a HW platform and the experiments performed in a real industrial scenario confirms its suitability to fulfill the requirements of the targeted applications.

The chapter is organized as follows. First, Section 5.1 reviews different high-performance wireless solutions considered for industrial applications and in general wireless TSN. The general SHARP architecture is described through Section 5.2, focusing on its wireless segment, w-SHARP. The implementation of a w-SHARP node in a SoC-FPGA-based SDR platform is described in Section 5.3. Section 5.4 describes the evaluation of w-SHARP through simulation means. Finally, Section 5.5 presents the validation and performance results of w-SHARP by experimental means through a HW testbed in a real industrial scenario.

5.1. High-Performance wireless for industrial communications

This section briefly reviews different technologies that are currently being considered as a suitable candidate for fulfilling the requirements of industrial applications and, in general, wireless TSN.

5.1.1. 5G-NR

The 5G-NR standard is designed to support most applications in both industrial and IT domains through the support of three differentiated traffic profiles, the Enhanced Mobile BroadBand (EMBB), the massive Machine-type (Mmtc), and the Ultra-Reliable and Low Latency Communications (URLLC) [105]. EMBB targets IT applications and it is an evolution of the services provided by LTE. massive Machine-Type Communications (mMTC) targets low-rate, low-power IoT devices, and URLLC targets industrial applications by providing low bounded latency, very high reliability, and deterministic behavior. Some real measurements and simulations have demonstrated that 5G-NR can provide a worst-case latency below 1 ms [32] and a cycle time of 2-3 ms for specific configurations [33]. In addition, the integration of 5G and TSN is a currently ongoing discussion in the 5G standardization process [106]. Still, several industrial applications are out of the performance provided by URLLC, especially in terms of latency and cycle time. For instance, some motion control [34] or power electronics [107] applications require sub-millisecond cycle time.

5.1.2. 802.11-based solutions

A second solution to implement a high-performance wireless network towards the wireless TSN paradigm could be based on 802.11, as 802.11 is a natural extension to Ethernet networks. However, legacy 802.11 is not deterministic and hence it cannot provide time-aware operation. Meaningful research has been done around how to modify 802.11 towards a wireless TSN version of 802.11. The efforts have been mainly focused on modifying the Medium Access Control (MAC) layer to provide time-aware scheduling [108] and the use of Commercial-Off-The-Shelf (COTS) devices to validate the MAC design in HW testbeds [109]. For instance,

IsoMAC [110] implements time synchronization to enable a persistent TDMA structure using two periods: a scheduled period to transmit RT traffic and a contention period to transmit 802.11 frames. This scheme can provide a cycle time of 4 ms but with a reduced No. of nodes. Similar solutions have been also proposed by some other researchers from Ikerlan [111] for time-critical applications. Compared to the former solution, [111] includes a novel handover process with no deadline losses, which is convenient for applications that involve mobile robots, and integration with TSN. Other custom TDMA examples over 802.11 are RT-WiFi [112] or Priority MAC [113]. In any case, these solutions share the same downside: they are based on the legacy 802.11 PHY that is not very efficient for short packets, and thus their cycle time and latencies are in the order of milliseconds [30].

5.1.3. 802.11ax

In the last few years, significant standardization efforts have been done over 802.11 to improve its efficiency and provide latency control. These efforts have resulted in the 802.11ax standard [36], a considerable evolution in the 802.11 family. 802.11ax has three main differential characteristics compared to older 802.11 standards that significantly increase its predictability and its efficiency for communication of short packets: the redesign of the OFDM numerology, the adoption of Orthogonal Frequency Division Multiple Access (OFDMA), and the introduction of the trigger frame, which allows an efficient contention-based mode.

OFDM numerology

802.11ax has changed the OFDM numerology from a ratio of 64 subcarriers for each 20 MHz BW in previous 802.11 standards to 256 subcarriers for each 20 MHz BW, which increases the OFDM symbol duration. Compared to previous 802.11 standards, 802.11ax has an OFDM symbol duration of 12.8 µs and a variable Cyclic Prefix (CP) duration of 0.8, 1.6, and 3.2 µs, which results in a maximum OFDM efficiency of 94%. On the contrary, 802.11 standards before ax have an OFDM symbol duration of 3.2 µs and a typical CP duration of 0.8 µs that provides an efficiency of 80%. On the downside, a larger No. of subcarriers may increase the sensitivity to carrier frequency offset, and more complex

algorithms may be necessary to ensure good reception quality. Additionally, the use of large OFDM symbols may provide reduced efficiency for very short frames, because most of the subcarriers of the last OFDM symbol may remain unused [19].

OFDMA

OFDMA splits the available BW of the network into multiple sub-bands, which can be arbitrarily assigned to different users enabling frequency-based multi-user transmissions. Specifically, 802.11ax OFDMA design is intra-frame-based, i.e. the OFDMA partitioning and frame assignment is sent within each frame and thus the OFDMA configuration only persists for the duration of the frame. The adoption of OFDMA significantly reduces the overhead of the transmission of short packets, since multiple packets transmitted to different users can be aggregated into a single PHY frame sharing the same preamble. On the downside, the use of a smaller BW for each user may lead to reduced reliability, since it will necessarily produce higher fading probability, especially in channels with NLoS conditions. OFDMA in 802.11ax is defined for both Downlink (DL) and Uplink (UL) streams. In the DL streams, the Access Point (AP) directly transmits the OFDMA frames, whereas the UL streams require the previous synchronization of the STAs involved in the OFDMA transmission. The synchronization is achieved with the trigger frame.

Trigger frame

The 802.11 trigger frame was defined for its first time in the 802.11ax standard. It can only be used by an AP and it has two purposes: it allows the synchronization of multiple STA and it carries the resource allocation of an UL multi-user OFDMA transmission. The resource allocation configuration comprises general information about the UL multi-user frame, such as BW, CP duration, frame length, and expected Rx power, and user-dependent information that contains the configuration of the BW allocated for each user, the Modulation and Coding Scheme (MCS), etc. The STAs that have correctly demodulated the trigger frame will be able to transmit in the UL multi-user frame. Figure 5.1 depicts the trigger frame-based UL transmission and the legacy single user UL transmissions from N arbitrary users. As evidenced, the trigger frame-based

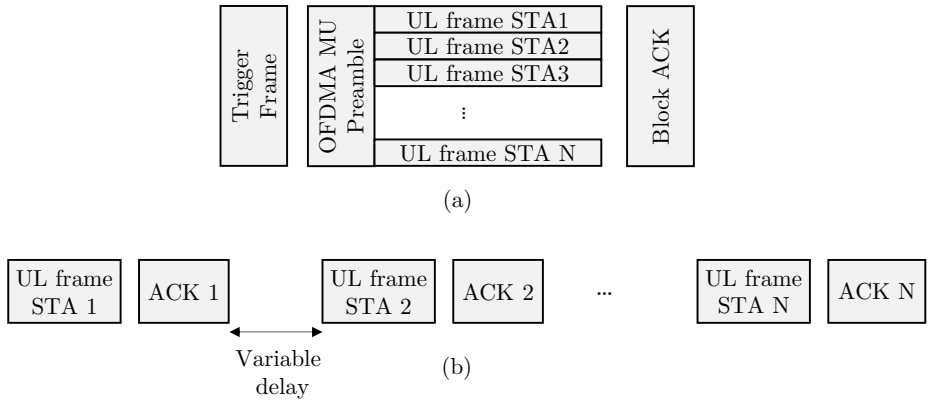


Figure 5.1. Trigger frame-based UL transmissions (a) vs. single user UL transmissions (b).

transmission is much faster than a single user transmission thanks to the grouping of frames into a single PHY frame.

These enhancements significantly improve the efficiency of 802.11ax in scenarios with a large number of nodes [36]. In addition, the use of contention-based modes combined with the trigger frame gives the opportunity to provide time-aware scheduling and to build soft RT applications [4]. Still, 802.11ax lacks some other aspects to enable stringent industrial use cases deterministic performance [30]. In addition, despite that 802.11ax includes a specific procedure for time synchronization [47], its precision may not be enough to fulfill the required performance. Finally, the next 802.11 standard, 802.11be [37], will contain some enhancements to cut down the 802.11 medium access latency. However, 802.11be specification has just started and its exact features are still unclear.

5.1.4. WirelessHP

Outside of the official 802.11 standardization process, but also based on the 802.11 PHY design, the wireless High-Performance (WirelessHP) project [38] aims at the design and implementation of a high-performance PHY layer for industrial applications. The targeted applications of WirelessHP are similar to the targeted applications in this thesis: industrial applications with RT

requirements that generate a very specific traffic profile: short packets with low bounded latency and sub-millisecond cycle times.

The main contribution of the WirelessHP design for industrial communications is the reduction of the transmission overhead to the minimum expression by using an optimized OFDM design for the transmission of short packets [114] and the use of a shortened preamble by exploiting the traffic periodicity and low-channel variability [115]. The WirelessHP design enables the transmission of ultra-short frames in the order of a few µs or even below [18].

WirelessHP research has been especially focused on the PHY layer, but there are some other critical aspects in industrial communications that have not been fully addressed, such as high-performance time synchronization, efficient MAC design, scheduling, etc. In addition, the published experiments are mostly performed using SW-based SDR platforms [20], which presents several limitations in terms of throughput, PER, and achievable BW.

5.2. SHARP technology

SHARP is a hybrid high-performance communications system specifically designed to implement NCS. It follows a network architecture with a tree-shaped topology. A possible example of a SHARP network is depicted in Figure 5.2. The objective of the network is to transmit the frames generated by the controller and by the wireless/wired end devices with minimum latency. The wired segment is used as the backhaul of the network and communicates the SHARP APs with the controller. The SHARP APs are the domain translators and perform the conversion of frames from the wired domain to the wireless. The wireless segment is used as a wireless extension to the wired network; hence the wireless devices are not designed to further schedule the frames to other nodes. The edge devices can be either nodes that run an RT application (RT nodes), nodes that do not run an RT application (non-RT nodes), or mixed nodes.

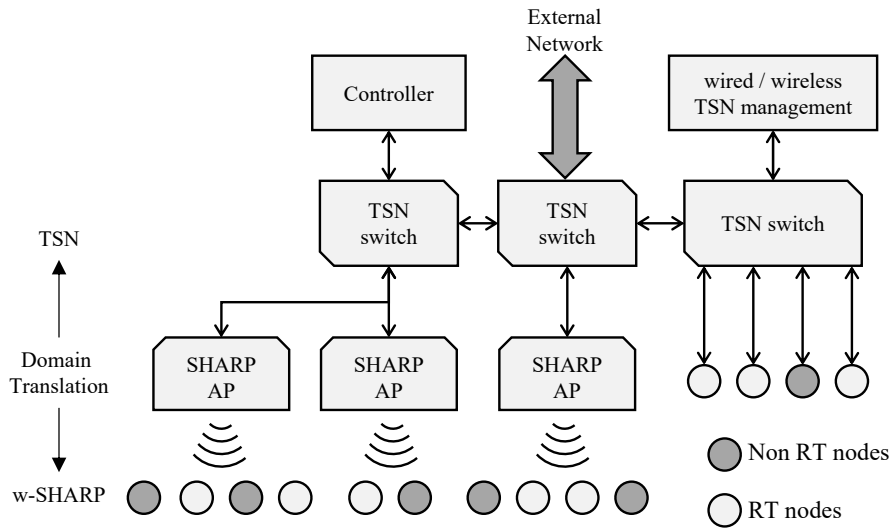


Figure 5.2. SHARP network topology.

The industrial requirements of most industrial applications are already fulfilled in the wired domain by some proprietary industrial standards, such as EtherCat [21], and by TSN [3], the evolution of ethernet towards the time-sensitive paradigm. TSN is a very convenient option for the SHARP architecture because it can mix critical and non-critical traffic and it is a non-proprietary standard defined by the IEEE. Additionally, TSN provides centralized TSN management, which could be also used to synchronize the operation of the wireless domain. On the other hand, nowadays wireless technologies are not able to provide similar (not even close) performance to TSN. Since the aim of the SHARP architecture is to seamlessly provide a similar performance and features in both wired and wireless domains, the design of a new wireless technology, named w-SHARP, is mandatory to build the wireless segment of the SHARP architecture.

The next subsection provides a brief description of the main features of TSN to provide low latency and deterministic performance over Ethernet. Afterward, Subsection 5.2.2 overviews the w-SHARP technology. After the w-SHARP technology overview, the next subsections comprehensively specify several aspects of w-SHARP, namely its PHY layer (Subsection 5.2.3), MAC layer (Subsection 5.2.4), control frames (Subsection 5.2.5), network discovery, time

synchronization, and network attaching (Subsection 5.2.6), and the integration of 802.11 devices in w-SHARP (Subsection 5.2.7).

5.2.1. Time-Sensitive Networking overview

TSN is a set of standards developed by the 802.1 task group [3] designed to provide deterministic, low latency, and low jitter communications over switched Ethernet networks with multiple hops. The TSN specification is divided into several sub-standards that solve specific challenges of deterministic communications over Ethernet [116]. The sub-standards can be classified into four groups upon the addressed challenges: reliability, time synchronization, QoS provisioning, and network management [106]. From the whole family, only a subset of standards is relevant to ensure deterministic, low-latency operation. These are 802.1AS [15] (synchronization), 802.1Qbv [117] (QoS provisioning), and 802.1Qcc [118] (network management). 802.1Qcc and 802.1Qbv are described in the next paragraphs. 802.1AS is based on PTP, which is already described in Section 3.1, and it is not repeated here.

802.1Qcc

The 802.1Qcc sub-standard defines the management of the TSN resources in a network to meet the target requirements of the applications. Hence, it does not include any enhancement itself rather than providing a standardized interface to configure the network. 802.1Qcc defines three network configuration structures: a fully centralized model, a fully distributed, and a centralized network/distributed user model. The fully centralized model is represented in Figure 5.3, whereas the fully distributed and centralized network/distributed user model is not depicted since they are a simplification of the former. Also note that the management paths and data paths depicted in Figure 5.3 are logical connections driven by the Ethernet networks rather than physical Ethernet connections.

The fully centralized model defines two entities named Centralized User Configuration (CUC) and Centralized Network Configuration (CNC) are defined. These entities centralize the decisions about the network and the configuration

of the nodes. Hence, the end devices of the TSN network can request a slot assignment to the CUC based on their traffic requirements. Then, the CUC and CNC will generate an end device/network configuration based on the network topology, state, and user requirements. The fully distributed mode considers a network without CUC and CNC, where the TSN configuration of the elements of the network are managed locally. Finally, the centralized network/distributed user model only defines the use of the CNC to perform the network configuration and allows the end devices to manage their resources. The fully centralized model necessarily offers higher performance than the distributed one due to the CUC/CNC centralizes the information of the whole network but at the cost of higher configuration complexity.

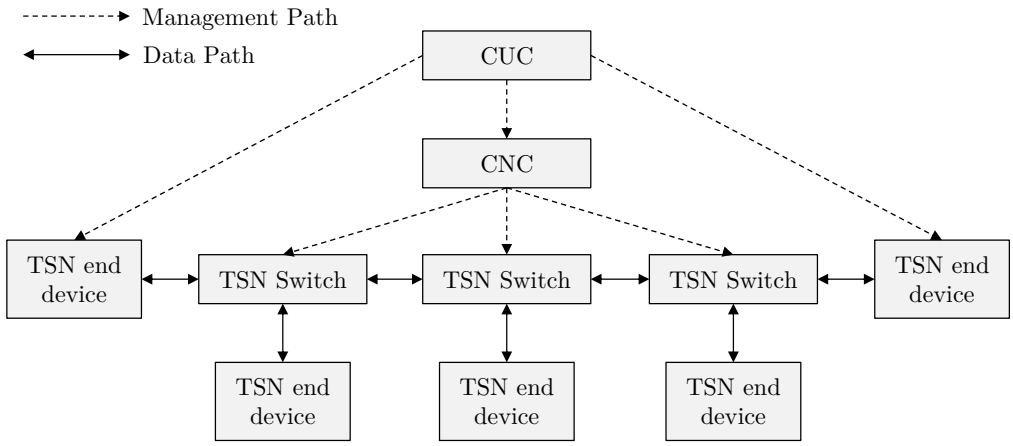


Figure 5.3. 802.1Qcc Fully centralized network model.

802.1Qbv

802.1Qbv introduces the concept of synchronized gates/queues to a global notion of time and per-gate packet classification, which enables TDMA-like operation over Ethernet. The gates/queues of a TSN switch is depicted in Figure 5.4, whereas an example of a TDMA configuration is depicted in Figure 5.5.

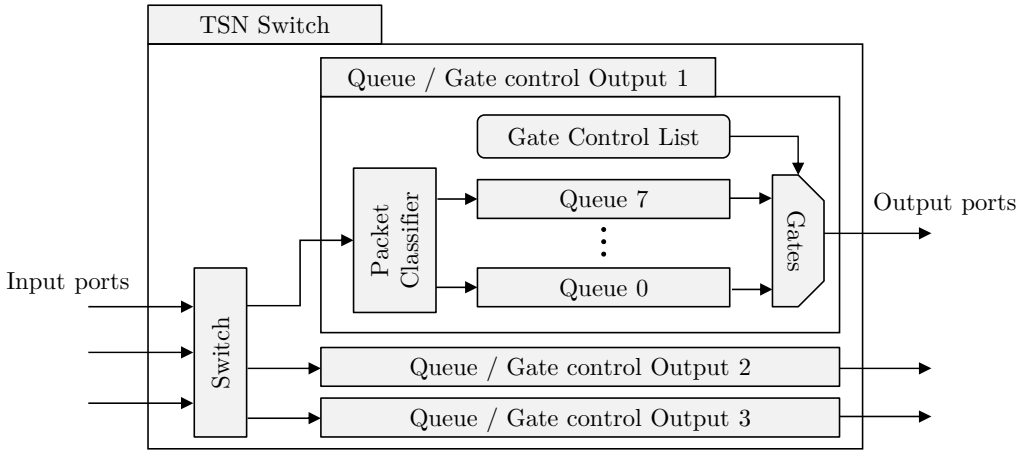


Figure 5.4. TSN switch.

The Ethernet packets are classified by their priority into 8 classes from the lowest priority (class 0) to the highest priority (class 7). Anytime that an Ethernet packet is received by a switch, it is introduced to an output HW queue based on its class (a total of 8 queues per Ethernet interface). A queue is connected to an output gate that can be either opened (allowed to transmit) or closed (cannot transmit). When a gate is opened, its queue can transmit its stored frames. The gates from several TSN switches within a network can be opened at the same allowing a very fast transmission and forwarding of high priority packets through the network.

The operation of the gates is defined in the gate control list. The gate control list contains the state of the gates for an operation cycle of duration T_s, which is periodically and indefinitely executed. The operation cycle structure is divided into slots of duration T_{sd}, being T_s multiple of T_{sd}. Thus, an operation cycle comprises a specific number of slots (N_s) of duration T_{sd}. The position of the gates in each slot is represented by a Boolean vector (opened/closed). Hence, the TSN operation cycle is represented by a Boolean matrix of 8 x N_s elements. Based on the matrix configuration, different gates will be opened at different times allowing the deterministic transmission of high-priority frames. In the case that multiple gates are opened in the same slot, the traffic is forwarded by its priority, from the highest priority class to the lowest priority class. A

configuration example is depicted in Figure 5.5, where one time slot is reserved for RT traffic and the rest of the slots are used for Best-Effort (BE) traffic.

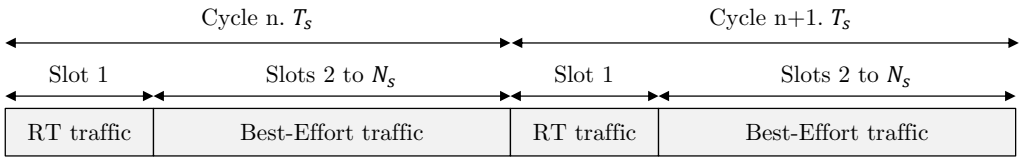


Figure 5.5. TSN operation cycle configured to serve RT traffic and BE traffic.

5.2.2. w-SHARP overview

w-SHARP is a wireless communications system specifically designed for industrial applications at the field level. w-SHARP inherits from 802.11 the basic (de)coding operations, and redefines: i) its MAC layer, and ii) the (de)modulation processes, iii) the preamble format. These modifications significantly increase its performance in terms of predictable behavior, bounded latency, update period, and efficiency for the transmission of short packets. Note that w-SHARP follows the TSN principles and provides similar behavior, but it is not compliant with the TSN family of standards (802.1as, 802.1Qbv, etc.).

The major changes of the PHY layer include the addition of PHY frame aggregation in the time domain and in the frequency domain (OFDMA), and the use of two different waveforms in UL and DL, which exploits the predictability of the industrial traffic. From the MAC layer, the next modifications may be highlighted: the use of a TDMA-based scheme that supports both time-critical and BE traffic in a similar way as TSN, the minimization of the InterFrame Spacing (IFS) to increase the radio resources effective usage, and the use of a low-overhead synchronization scheme to synchronize the nodes of the network. The next paragraphs summarize the content of the subsequent subsections, which includes a comprehensive description of the w-SHARP technology.

Subsection 5.2.3 specifies the w-SHARP PHY operation. It describes the OFDM symbols numerology, the UL and DL waveforms, and the proposed aggregation in time and frequency.

The w-SHARP MAC is specified in Subsection 5.2.4. It includes the definition of the general TDMA structure (superframe), the definition of the periods of the superframe used to schedule RT traffic and BE traffic, and the design of the IFS.

w-SHARP defines some basic control frames used to enable basic control operations, such as beacons (synchronization), scheduling requests (slot petition), etc. The structure of the control frames is defined in Subsection 5.2.5.

Subsection 5.2.6 describes the proposed network discovery, synchronization, and attaching mechanisms from the UE perspective. It includes two procedures. The first procedure is based on network pre-design where every node has prior information about the network, whereas the second procedure is totally blind, and it is based on the connection procedure of 802.11.

Finally, Subsection 5.2.7 describes a set of mechanisms that can be used to maintain the compatibility with older 802.11 devices by applying some limitations to the w-SHARP and to the 802.11 nodes.

5.2.3. w-SHARP PHY

The w-SHARP PHY has been co-designed with its MAC layer based on the typical industrial traffic profile (short packets periodically generated that require bounded low latency). It uses two main waveforms (DL and UL) based on the legacy 802.11 OFDM PHY combined with frame aggregation. The DL waveform is used to transmit frames from the AP to the STAs and the UL waveform is used to transmit frames from the STAs to the AP. Both waveforms use the same basic OFDM symbol structure, further detailed below. In addition, and as a further consideration, OFDMA has been also introduced in the design to boost the throughput of a w-SHARP network. The (de)coder is based on the one of our 802.11g implementation, which is described in Appendix A.

5.2.3.1. Basic OFDM symbol structure

The OFDM symbol structure has a significant impact on the performance of a wireless system [18]. A large separation between subcarriers reduces the

intercarrier interferences under high mobility channels, but reduces the number of subcarriers per OFDM symbol and its duration and, as a result, the efficiency [18]. In addition, when targeting industrial applications, which are typically characterized by short packets, the use of more subcarriers per OFDM symbol may not increase the efficiency of the wireless system. The reason is that, if the frames occupy few OFDM symbols, a significant percentage of the subcarriers may remain unused [30]. In fact, this may be a significant efficient issue of 802.11ax, which has increased its OFDM symbol size from 64 to 256 for a BW of 20 MHz, though the use of OFDMA reduces the effective number of subcarriers per frame.

In [114], M. Luvisotto et al. presented an algorithm to optimize OFDM-based PHY for industrial applications that use short packets. The results depend on the scenario, but the number of optimum subcarriers is between 32 and 128, which is very close to the one originally used in IEEE 802.11. Therefore, w-SHARP reuses the legacy OFDM symbol structure of 802.11, which comprises 64 subcarriers and a CP duration of 800 ns for a 20 MHz BW. The total duration of a single 802.11 OFDM symbol is 4 µs. Besides, the use of the legacy 802.11 OFDM numerology eases the compatibility of w-SHARP with 802.11 devices before 802.11ax.

5.2.3.2. DL Waveform

The DL waveform has two key components: a long preamble structure similar to the 802.11 legacy preamble, and the use of PHY frame aggregation that reduces the preamble overhead. The preamble duration is 16 µs and it has two fields: the Short Training Field (STF), and the Long Training Field (LTF). The STF ensures appropriate Carrier Frequency Offset (CFO) and AGC estimation. The LTF is used to perform the fine CFO estimation, and precisely detect the start of the frame. This preamble structure is quite large, which is one of the main well-known inefficiencies of 802.11 in the communication of short packets [30].

Given the periodic character of industrial traffic, a suitable option to improve the 802.11 PHY efficiency is to perform frame aggregation. Frame aggregation can be defined at different stack levels. For instance, the frames can be joined in a single MAC data unit and transmitted to all users through a broadcast frame.

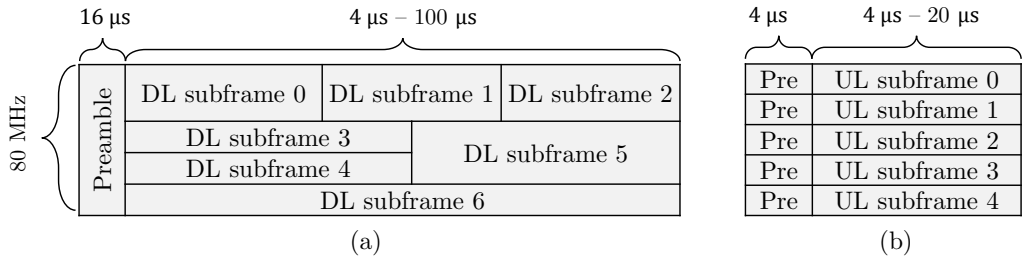


Figure 5.6. DL (a) and UL (b) w-SHARP frame with BW = 80 MHz.

However, the users with the worst channel quality will limit the MCS of the frame, which will in turn reduce the efficiency. The second option is to perform PHY frame aggregation. In this scheme. several subframes with different lengths and MCS are prepended with only one preamble.

The MCS and length of each subframe inside a DL frame are decided by upper layers and thus the frame does not need to carry any extra parameter to be decodable by each of the users. This is translated in very low control overhead compared to 802.11, which always includes several PHY parameters through the SIGNAL field [29]. The MCS and length of each subframe can be modified at running time through the procedure specified in subsection 5.2.6.

The PHY frame aggregation can be performed in the frequency domain, the time domain, and the spatial domain. In general, the spatial aggregation provides lower orthogonality than frequency or time aggregation, which is usually translated in lower reliability, and hence it is not considered to perform frame aggregation. Thus, time and frequency aggregation are the chosen schemes.

On the one hand, The BW and OFDMA configurations defined for w-SHARP are summarized in Table 5.1., where OFDMA is only used with a BW of 40 MHz and above. As can be seen, the fundamental OFDM symbol structure is maintained from 802.11, which provides a near-optimum performance in the transmission of short frames [114]. On the other hand, the frames can be freely aggregated in time by modulating each independent OFDM symbol and joining them into a single frame. A frame with 80 MHz BW using both time and frequency aggregation is depicted in Figure 5.6 (a).

The proposed DL waveform has several advantages for industrial applications compared to the 802.11 standard. First, it vastly improves the efficiency when transmitting short packets as several packets are aggregated using the same preamble. Furthermore, PHY time-frequency domain aggregation provides more flexibility than other forms of aggregation (e.g. MAC aggregation), as the OFDM symbols of each frame may use different MCS.

Table 5.1. w-SHARP OFDMA Design.

BW [MHz]	FFT Size	Data Subcarriers per channel	Pilots per channel	No. OFDMA channels
20	64	48	4	1
40	128	56	4	2
80	256	48	4	5
160	512	54	4	9

5.2.3.3. UL Waveform

The UL waveform is used by the w-SHARP STAs to transmit frames to the w-SHARP AP. In the case of UL transmissions, each w-SHARP STA transmits few RT frames in each superframe to the w-SHARP AP. As a result, the preamble used in DL is quite long and not efficient for UL frames. To increase the efficiency, the UL frames may use a shortened preamble with fewer OFDM symbols. The impact of a shortened preamble in industrial applications has been studied in [115], where it has been found that a short preamble may be feasible under some specific constraints, such as high Rx power. However, in [115], both the AP and the STAs use the same shortened preamble, requiring a more complex AGC and CFO estimation.

The w-SHARP STAs estimate the channel state information from the DL frames. Thanks to this, the w-SHARP STAs can compensate for the channel response to ease the w-SHARP AP detection and demodulation of UL frames. Specifically, w-SHARP moves the AGC and CFO correction from the Rx PHY of the w-SHARP AP to the Tx PHY of the w-SHARP STA. As a result, the preamble of the UL frames has a length of one OFDM symbol, as the fields of the preamble used to perform the AGC and CFO estimation are not required.

Finally, OFDMA aggregation is similarly followed in UL frames, using the same OFDMA configuration (Table 5.1). An UL frame with 80 MHz BW using frequency aggregation is depicted in Figure 5.6 (b).

5.2.4. w-SHARP MAC

The design of an efficient and flexible MAC several challenges that should be addressed taking into account the PHY features to maximize the MAC performance (i.e. provided latency and throughput). The main desired MAC feature from the industrial communication perspective is stable and predictable operation. The MAC that has been mainly adopted to obtain RT operation is the TDMA scheme [18] [112] [108] [119]. In this scheme, the air is divided into slices of a fixed duration, which are periodically repeated every T_S. The air slices are usually named "TDMA superframe". The superframe is in turn divided into timeslots that are periodically repeated in each T_S. Each timeslot is assigned to different nodes, which can use the timeslots to transmit frames. This procedure offers high-performance and simple RT operation. However, simple TDMA-based schemes are inflexible and can only support the transmission of RT frames. Apart from RT operation, other appealing features in industrial communications are:

- Support different traffic profiles to enable different types of applications. Typically, three main traffic profiles are considered: RT periodic frames, RT aperiodic frames used for alarms, and BE frames, used mainly for configuration purposes.
- Provide adaptability to the wireless medium. The MAC should be able to detect a bad/good quality wireless channel and change the timeslot configuration to maintain the reliability below a specific bound.
- Support different data sampling rates in the RT data generation. The nodes may generate data with different periodicity, and then the MAC should be able to efficiently allocate timeslots to support different transmission rates,

w-SHARP defines a flexible superframe structure with 5 different periods that allow the efficient transmission and mixing of RT and BE traffic. Through the subsections, the superframe structure and each of the w-SHARP periods are

detailed. Additionally, w-SHARP defines several control frames, which are described in Subsection 5.2.5.

5.2.4.1. Superframe structure

w-SHARP follows a flexible TDMA structure based on the periodic transmission of superframes with a configurable duration T_s. The superframes are conformed by 5 fundamental periods that provide different functionalities: the RT Downlink period (RT DL), RT Downlink Retransmission period (RT DL RTx), the RT Uplink period (RT UL), the RT Uplink Retransmission Period (RT UL RTx), and the BE period (BE). The first four periods are used to transmit RT frames, whereas the BE period is used to transmit BE frames. A w-SHARP superframe can comprise an indefinite number of RT periods with arbitrary order, and it must start with a RT DL period with a beacon frame used for synchronization purposes. The BE period should be the last period of a superframe. An example of a typical w-SHARP superframe with one of each period is depicted in Figure 5.7.

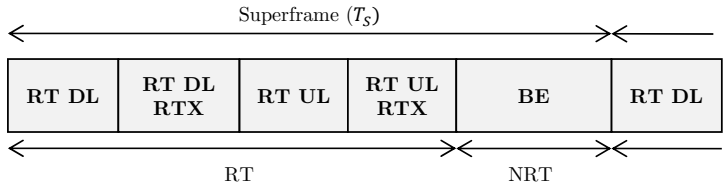


Figure 5.7. w-SHARP superframe example with one of each period.

The depicted w-SHARP superframe contains one of each period. It starts with a RT DL period, which contains a DL frame starting with a beacon. The next period is the RT DL RTx, which is used to perform retransmission. Afterward, it has two more periods to transmit UL information and UL RTx in case that retransmissions are needed. Finally, the superframe ends with a BE period used to transmit non-RT frames.

5.2.4.2. Real-Time Downlink period (RT DL)

RT frames transmitted from the SHARP AP to the slave nodes are transmitted during this period. The frames of this period follow the waveform described in Subsection 5.2.3.2, which is based on PHY frame aggregation in time/frequency.

Figure 5.8 presents an example of the frames exchanged during a RT DL period. The first frame transmitted in a RT DL period is a w-SHARP beacon. The beacon is transmitted through every OFDMA channel to ensure that a w-SHARP STA can retrieve the basic information about the w-SHARP network. After the beacon, the DL frame contains data subframes for specific users, which are independently modulated to different MCS and different lengths.

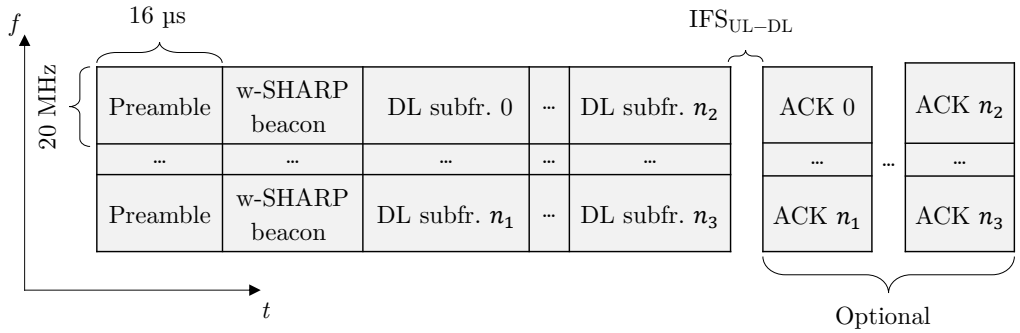


Figure 5.8. RT DL period.

Once the DL frame is transmitted and based on the network configuration, the w-SHARP STAs can return dedicated ACKs or piggybacked ACKs. In case that dedicated ACK frames are necessary, they are transmitted right after the RT DL period using the UL waveform. In the case that they use the piggybacked ACKs, the ACKs are transmitted in the header of the UL frames in the UL period.

5.2.4.3. Real-Time Downlink Retransmissions period (RT DL RTx)

The RT DL RTx period is used to retransmit specific RT DL subframes if some subframes in the actual superframe or past superframes were not correctly received. Figure 5.9 represents the structure of the RT DL RTx period. At the start of the RT DL RTx period, the AP must decide the scheduling of the frames based on higher layer parameters (priority, length, probability of failure /success, length of the DL RTx period, etc.). Then, it lists a group of frames that will be retransmitted. The RTx is done through a DL RTx frame (structure defined in Subsection 5.2.5). After the transmission of a DL RTx frame, the w-SHARP will transmit the DL frame that carries the subframes for each user. Finally, the STAs

must transmit an ACK in the corresponding slot if they correctly received the DL subframe. Additionally, once the RTx task is over, the w-SHARP AP may use this period to perform other tasks, such as frame rescheduling (see Subsection 5.2.6).

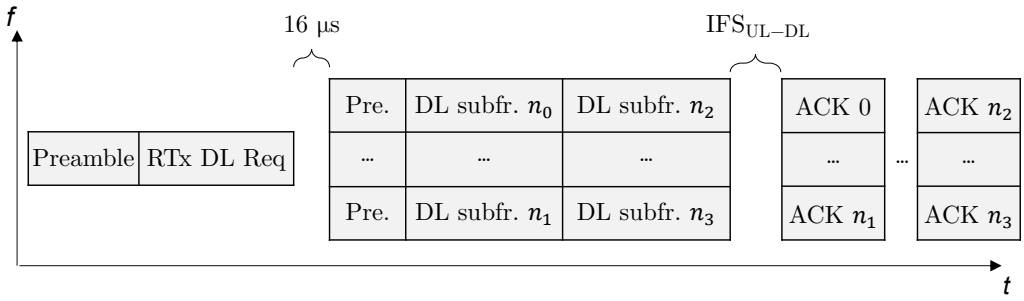


Figure 5.9. RT DL RTx period.

5.2.4.4. Real-Time Uplink (RT UL)

The RT UL period is used to transmit RT frames from the w-SHARP STAs to the w-SHARP period. The frames use the UL waveform as defined in Subsection 5.2.3.3. Figure 5.10 represents the structure of the RT UL period. As in DL, the RT UL frames are tightly scheduled in time and frequency (OFDMA). The frames Tx instant, length, and MCS of each frame are pre-scheduled and known by the SHARP AP. The frame spacing between RT UL frames (IFS$_{UL-UL}$) must be small as possible to ensure high efficiency (see Subsection 5.2.4.7 InterFrame Space (IFS) design).

Data carried by RT UL frames have a small lifetime (the data expires once that new data from the same sensor is available). Thus, the RT UL frames are not acknowledged by the w-SHARP AP.

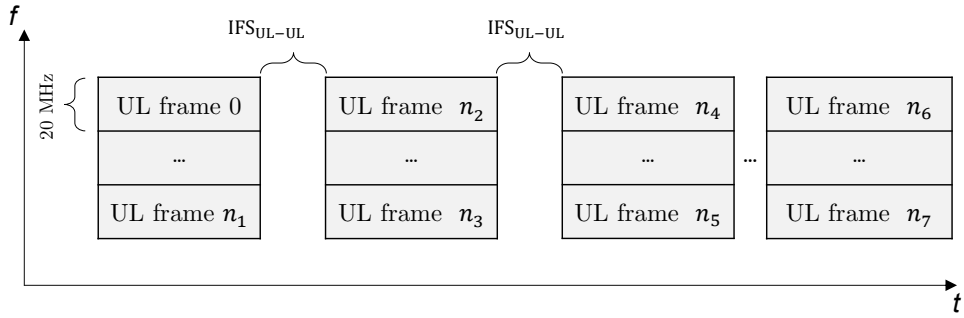


Figure 5.10. RT UL period.

5.2.4.5. Real-Time Uplink Retransmission period (RT UL RTx)

The RT UL RTx period is used to retransmit RT UL frames if some frames in the actual Superframe or past Superframes were not correctly received by the w-SHARP AP. Figure 5.11 represents the structure of the RT UL RTx period. The definition of this period is similar to the definition of the RT DL RTx period.

At the start of the RT UL RTx period, the AP must decide the scheduling of the frames based on higher layer parameters (priority, length, probability of failure /success, length of the DL RTx period, etc.). Then, it lists a group of UL frames that must be retransmitted. The RTx is scheduled through a RT RTx req frame (structure defined in Subsection 5.2.5). After the transmission of a RT RTx req frame, the w-SHARP STAs must transmit UL frames in the corresponding slot if they were asked to perform an RTx.

Additionally, once the RTx task is over, this period may be used by the w-SHARP AP or other nodes to perform other tasks, such as in the RT DL RTx period.

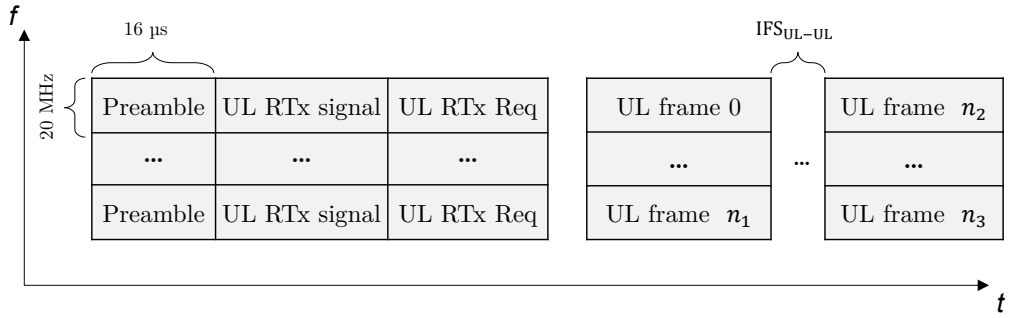


Figure 5.11. RT UL RTx period.

5.2.4.6. Best-Effort Period

Finally, the BE period is used by every node connected to the network to freely access the channel using the CSMA/CA scheme used in 802.11. It implements all the mechanisms of the basic 802.11 including fragmentation, ACK transmission, etc. It also has three modifications compared to the 802.11 standard, which allows a simple integration between the BE period and the RT periods.

First, the Network Allocation Vector (NAV) has been modified to avoid interframe interferences. The modification of the NAV uses the information about the superframe structure to know in advance if a frame can be safely transmitted during the 802.11 period or not

$$T_{rSTD} < \begin{cases} T_{STD} + T_{ACK} + 2 \cdot SIFS + T_G; & if\ ACK\ required \\ T_{STD} + SIFS + T_G; & if\ ACK\ not\ required \end{cases}, \tag{5.1}$$

being T_{rSTD} the remaining time of the 802.11 period, T_{ACK} the duration of the ACK, T_{STD} the duration of the 802.11 frame, the Short InterFrame Space (SIFS) as defined in the 802.11 standard, and T_G a guard period used to guarantee the switching between the MACs. With this modification, the 802.11 frames transmitted by w-SHARP nodes can be safely delivered without causing any collisions to the RT frames.

Second, the AP can skip the CSMA/CA at the start of the BE period if needed to operate with more priority than the transmission of BE frames. For

instance, a frame rescheduling or the transmission of an RT aperiodic frame (alarm).

Finally, the w-SHARP network can use a subperiod named BE Controlled Phase (BECP) within the BE period to ensure coexistence with older 802.11 devices in the same frequency band. The compatibility is further detailed in Subsection 5.2.7.

5.2.4.7. InterFrame Space (IFS) design

The InterFrame Space (IFS) in 802.11 is defined as is the guard time used between the transmission of adjacent frames transmitted by different users. The IFS is totally necessary to avoid inter-frame interferences, though it must be minimized to maximize the available radio resource. For instance, the 802.11 IFS last 16 µs, which is quite large and, as a result, 802.11 does not well suit industrial applications that transmit short frames. Obviously, the IFS cannot be arbitrarily small, and its minimum value depends on some RF and PHY parameters, the synchronization between the nodes, and the wireless channel properties.

Two IFS have been considered in w-SHARP: the IFS between UL frames ($\text{IFS}_{\text{UL-UL}}$) and the IFS between DL-UL frames or UL-DL frames ($\text{IFS}_{\text{UL-DL}}$). The IFS is the sum of a few contributions. First, the w-SHARP nodes must have a common notion of time to transmit the frames in their specific timeslots, which is typically obtained through a synchronization algorithm. The time synchronization error T_e could be in the range of 1-100 ns, depending on the wireless channel and the synchronization algorithm, as shown in Chapter 3. Second, the variable channel delay T_h, which can range from a few ns to 100 ns in indoor scenarios, also introduces jitter. Finally, the RF radio Tx/Rx switching delay T_{sRF} must be considered for the $\text{IFS}_{\text{UL-DL}}$. T_{sRF} for a common radio chip can be in the order of 2-3 µs [90]. Then, the minimum IFS can be expressed as

$$\text{IFS}_{\text{UL-DL}} = T_{sRF} + 2 \cdot T_h + T_e, \tag{5.2}$$

$$\text{IFS}_{\text{UL-UL}} = 2 \cdot T_h + T_e. \tag{5.3}$$

Common values for these parameters could be $T_{sRF} = 3$ µs, $T_e = 200$ ns, and $\tau_h = 100$ ns, which results in $\text{IFS}_{\text{UL-DL}} = 3.5$ µs and $\text{IFS}_{\text{UL-UL}} = 0.5$ µs. The given $\text{IFS}_{\text{UL-DL}}$ and $\text{IFS}_{\text{UL-UL}}$ are quite conservative, and they could be even lower depending on the time synchronization algorithm, or if a specific radio is selected.

5.2.4.8. Synchronization of the w-SHARP and TSN RT slots

The periods defined in w-SHARP has been designed in consequence to be integrated into the TSN operation as described in the IEEE 802.1Qbv standard [117]. This integration differs whether the TSN flow occurs from the TSN to the wireless TSN domain or from the wireless TSN to the TSN domain. Both cases are depicted in Figure 5.12. As can be seen, in the former case, the TSN slot ends when the wireless TSN slot starts, ensuring that every TSN frame has been correctly transmitted to the domain translator before the wireless TSN slot starts to transmit the frame. Note that TSN does not ensure that a RT frame will be transmitted in a specific time, the specification only ensures that a RT frame will be deterministically forwarded during a RT slot. In the wireless TSN to TSN case, the wireless TSN slot and the TSN slot are synchronized, but the TSN slot must have some guard gaps to accommodate the coding and decoding latency of the wireless TSN segment and the latency of the translation between domains.

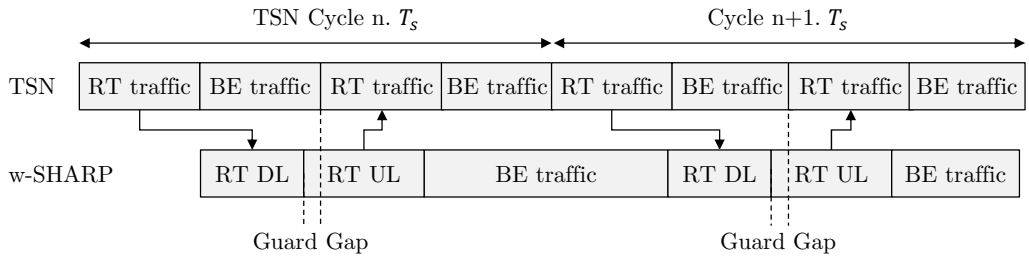


Figure 5.12. Integration of w-SHARP and TSN RT slots.

5.2.5. Specification of w-SHARP control frames

5.2.5.1. Beacon

The w-SHARP beacon frame is periodically transmitted by the w-SHARP AP at the start of the DL period and contains the fundamental information about the w-SHARP node. It is modulated with Binary Phase Shift Keying (BPSK) modulation with ½ CC and its length is 16 bytes. The beacon structure is depicted in Figure 5.13. It has the next fields: the version ID, which is always set to 0, a w-SHARP AP ID that represents a unique identifier in the network, a timestamp that contains the actual time in seconds/nanoseconds, the BW operation mode (20, 40, 80 or 160 MHz), the OFDMA channel index used to transmit the w-SHARP beacon (0 to 7), the superframe duration (T_S) in µs, and the duration of the RT periods (T_{RT}) in µs. The beacon is transmitted through all OFDMA channels. The rest of the DL frame comprises subframes independently transmitted to each user.

Version ID	w-SHARP AP ID	Timestamp	BW // OFDMA CH ID	T_S	T_{RT}	CRC

Figure 5.13. w-SHARP beacon frame format.

5.2.5.2. RT RTx req frame

The RT RTx req frame is transmitted in the DL/UL RTx period and it is used to schedule the subsequent retransmitted frames an ACK in case. Its structure is depicted in Figure 5.14.

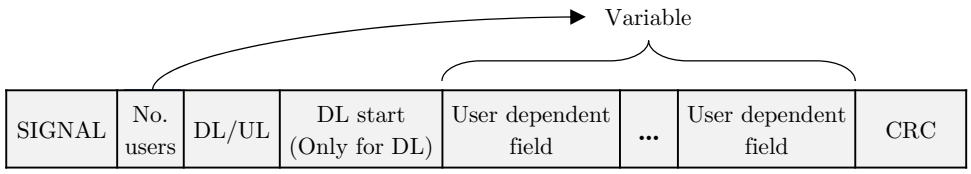
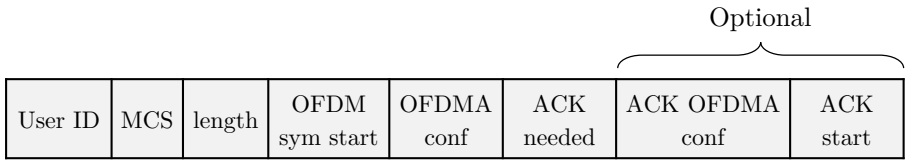


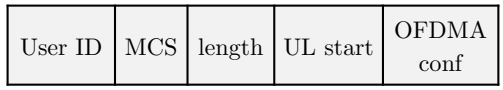
Figure 5.14. RT RTx frame format.

The RT req RTx frame is transmitted using the basic 20 MHz bandwidth. The frame is divided into three parts, the SIGNAL, the common field data, and the user-dependent field data. As in 802.11, the SIGNAL indicates the length of

the RT RTx req data and the modulation of the payload (typically BPSK with CC 1/2 to ensure a reliable transmission). The common field indicates the No. of allocated slots and either if the frame carries DL RTx or UL RTx information. In the case of DL, the frame also carries the start of the DL frame in ns from the start of the superframe. The user-dependent field contains the user-specific frame configuration and depends on whether the RT RTx frame is used for DL or UL (Figure 5.15).



a) DL user dependent configuration.



b) UL user dependent configuration.

Figure 5.15. RT RTx user-dependent fields.

The DL user-dependent field includes the user ID, MCS, length, OFDM symbol start of the user subframe within the DL frame, and OFDMA configuration. In addition, in case that a dedicated ACK is needed, the DL user-dependent field may include the ACK information: OFDMA configuration and the start of the ACK frame in ns from the start of the superframe. The MCS of the ACK is set to the MCS of the DL subframe. The UL user-dependent field includes the user ID, MCS, payload, OFDMA configuration, and the time slot of the UL frame expressed in ns from the start of the superframe.

5.2.5.3. ACK

The ACK frame is used to acknowledge DL frames received by the STAs and hence it can only be used for UL transmissions. The ACK carries the user ID, an ACK/NACK bit, and the CRC field (Figure 5.16).

User ID	ACK/NACK	CRC

Figure 5.16. w-SHARP ACK frame format.

5.2.6. Network discovery, synchronization, attaching, and resources assignment

This subsection outlines the processes of network discovery, time synchronization, and network attaching. Network discovery deals with the initial step of the w-SHARP STAs when the STA does not have any information about the w-SHARP networks. Time synchronization refers to the correction of the time of the w-SHARP STAs to be able to transmit in a w-SHARP network without interferences. Finally, network attaching refers to the process of logical connection to a w-SHARP network (authentication, slot petition, etc.) to be able to transmit data frames.

Depending on the network design philosophy, two network configurations can be distinguished: pre-engineered network configuration, and automatic network configuration. In a pre-engineered network configuration, the superframe format, No. of nodes, payloads, transmission frequency, nodes ID, time-slot configurations, etc., are preconfigured in each of the nodes before the network initialization. In an automatic network configuration, the w-SHARP nodes do not have explicit information about the scheduling, superframe structure, etc. Thus, the w-SHARP STAs must perform the network discovery, synchronization, and attaching to the w-SHARP network to transmit data frames.

In general, pre-engineered networks offer higher predictability and performance, since the network is explicitly designed to fulfill the application requirements, though, on the downside, the scheduler design for dense networks may suffer from high engineering costs. Besides, it also suffers from low flexibility since the network configuration cannot be automatically adapted in runtime. On the other hand, automatic network configurations are more flexible, and their operation can be adapted to allow more/fewer nodes at running time, change the

payload sizes, etc. On the downside, the control information necessary occupies radio resources, which reduces the amount of radio resources used for data.

The implementation of a node with automatic network configuration capabilities requires the specification of the link layer and complex scheduling algorithmic, which is out of this thesis, focused on PHY and MAC layers. Thus, the HW implementation of the w-SHARP prototype uses a pre-engineered configuration, where each of the w-SHARP nodes has total prior knowledge about the network configuration. Still, some hints to enable automatic network configurations are provided in this subsection, such as the network discovery and attaching processes, and a simple scheduling and rescheduling process. The implementation of these processes will be carried out in future works.

5.2.6.1. w-SHARP AP discovery and time synchronization in automatic network configuration

In an automatic network configuration, the w-SHARP STAs do not have any information about the w-SHARP APs configuration and localization and thus they must perform a discovery process to find the w-SHARP networks and establish a wireless link. The w-SHARP networks are discovered through the beacons transmitted at the start of the superframes by the w-SHARP AP. As in 802.11, the w-SHARP STAs listens for beacons transmitted by the w-SHARP APs at different frequencies until they demodulate one of them. Then, the w-SHARP STA will store the w-SHARP network information in a network table, which contains the information of all the network discovered by the STA.

After the network discovery, the w-SHARP STAs must synchronize their internal time to the w-SHARP AP. As described in Chapter 3, three messaging protocols are widely used to perform time synchronization in wireless networks: the full PTP messaging, the simplified PTP messaging, and the IEEE 802.11 FTM messaging. The full PTP and FTM messaging provides better performance, though they require a bidirectional frame exchange, whereas the simplified PTP messaging is based on frame broadcasting, which is simpler to implement, as it does not require the response of the STA. Besides, it can provide enough performance for wireless networks deployed at the edge and with low/medium distances. Consequently, w-SHARP adopts the simplified PTP. In the case of

large distance networks, a dual solution could be used: the simplified PTP messaging is used during the first steps of the attaching process, and the full / FTM messaging or other similar procedures can be used.

The network discovery and time synchronization are seamlessly integrated and the w-SHARP STAs perform both actions at the same time through the demodulation of beacons. Once that a w-SHARP STA has demodulated one beacon, it can continuously perform the beacon demodulation until the time synchronization error is below a threshold (typically in the range of 100 ns). Once the STA is synchronized, the STA changes its operation mode from "unsynchronized mode" to "synchronized mode" and proceeds with the network attaching. In the case of pre-engineered networks, the network attaching is not necessary, and the w-SHARP nodes will directly start the transmission and reception of data frames.

5.2.6.2. Network Attaching

After the network discovery and synchronization, the w-SHARP node must establish the connection to the network to transmit and receive data. A new network attaching procedure could be proposed for w-SHARP but, since it has an 802.11-like period that can be used to transmit 802.11 frames, a natural option could be to use the network attaching process used in common 802.11 networks [29]. The main difference compared to common 802.11 networks is that the attaching process may present a higher latency since only a portion of the radio resources is reserved for the transmission of 802.11 frames. The attaching process is depicted in Figure 5.17 and briefly described below. The attaching process is entirely performed through the BE period.

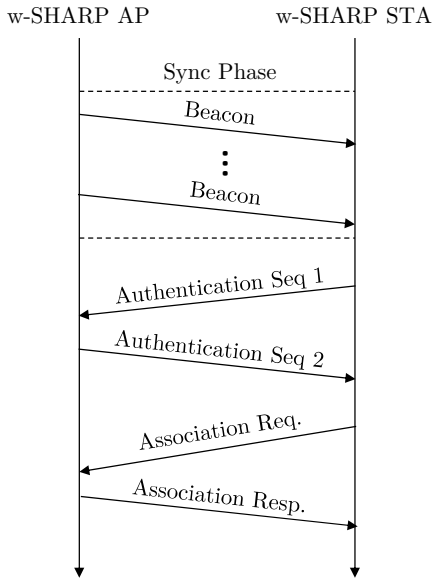


Figure 5.17. w-SHARP attaching process based on 802.11.

After the time synchronization process, the w-SHARP node starts the attaching process. The w-SHARP AP must be configured in 802.11 with Wi-Fi Protected Access 2 (WPA2). The w-SHARP node starts the attaching process with the transmission of an authentication frame with Seq = 1. Afterward, the w-SHARP AP will answer with an authentication frame with Seq = 2. This frame indicates to the w-SHARP STA that the node is already authenticated. Then, the w-SHARP STA and the w-SHARP AP exchange two more frames to perform the association (Association Request and Association response). After this frame exchange, the w-SHARP STA is already connected to the w-SHARP AP. Finally, the nodes have to perform the WPA2 frame exchange to end the attaching process. Since WPA2 is out of the context of PHY/MAC layers, the WPA2 data exchange is not described here.

5.2.6.3. Resources assignment

Once the w-SHARP network grants access to the w-SHARP node, the w-SHARP nodes perform a scheduling process to decide the assignment/reassignment of the radio resources to each user. The scheduling process can be used whether to add

new time slots or to modify the already radio resources. Specifically, w-SHARP uses the scheduling for:

- Create a new subframe in a DL/UL frame for the communication of periodic data between the w-SHARP AP and STA.
- Change the configuration of an already established subframe (MCS/payload length) based on:
  - A worsening/improvement of the wireless channel quality. The wireless channel conditions may vary along the time due to the movement of the STAs/AP or changes in the environment. Hence, it may be necessary to adjust the MCS of a specific subframe to match the desired PER. The MCS can either increase or decreased depending on the variations of the channel quality.
  - A rescheduling request of an end-device due to changes in the packet periodicity, length, PER target, or position within the superframe. A specific application may change its operation conditions and then it may require a change in the scheduler. This process is similar to the creation of a new subframe.
- Remove a subframe in a DL / UL frame when it is not going to be used anymore and upon the request of an end device or the TSN scheduler. In the case that an end-device does not require a subframe anymore, the w-SHARP AP can erase the subframe from the subsequent superframes to minimize the traffic consumption.

The scheduling process is summarized in Figure 5.18 and described below.

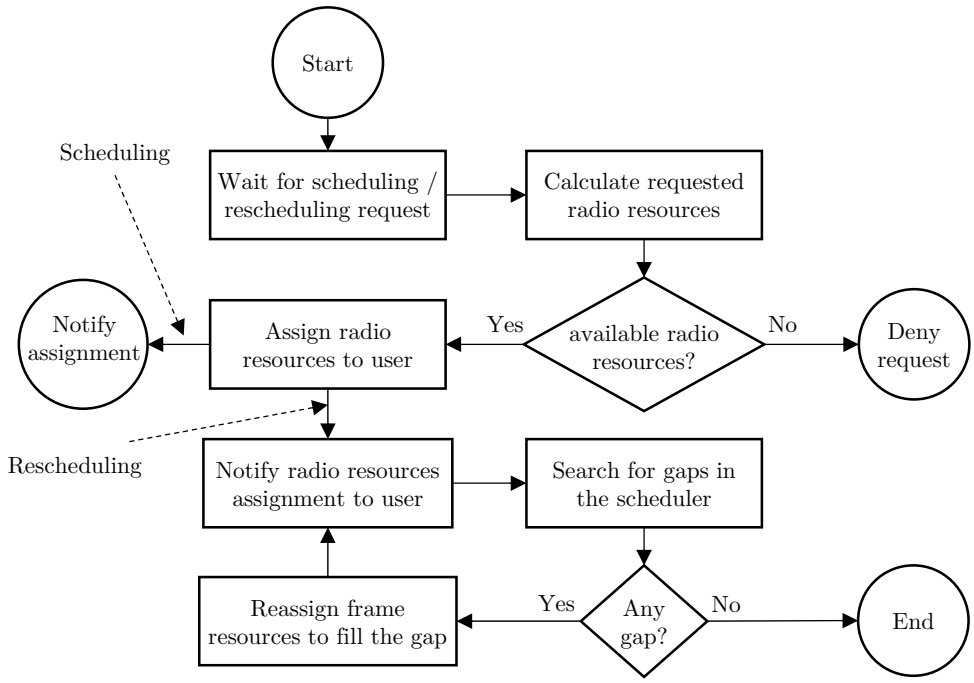


Figure 5.18. Scheduling / Rescheduling flowchart in the w-SHARP AP.

First, the w-SHARP AP waits for a scheduling/rescheduling request from the TSN network, a w-SHARP STA, or by the AP itself if it detects changes in the quality of the channel. Once the w-SHARP AP receives the scheduling request and, based on the scheduling information of the frame (periodicity, payload, time window) and the channel quality between the w-SHARP AP and the STA end device, the w-SHARP AP will calculate the amount of required radio resources and range of possible positions to allocate the subframe. If there are not enough radio resources available, it will deny the scheduling request. If there are enough radio resources, it will assign in its scheduler the radio resources, and it will notify the STA of the creation or reassignment of radio resources. Once the STA has been notified, it will start the transmission/reception in the assigned radio resources. In the case that a rescheduling is taking place, the w-SHARP AP will maintain the previous slot until the w-SHARP AP detects that the w-SHARP STA has started to use the new time slot.

5.2.7. Integration of 802.11 devices in a w-SHARP network

Despite that w-SHARP PHY can be configured to behave as the 802.11 PHY, the MAC differences of w-SHARP and 802.11 hinder the use of legacy 802.11 devices in a w-SHARP network. Nonetheless, the integration of both technologies has several advantages. For instance, the interoperability of different devices in the same network and an easier integration in already built networks. To do such integration, w-SHARP must support the 802.11 standard (MAC and PHY) and guarantee a collision-free coexistence between the 802.11 nodes and w-SHARP nodes. w-SHARP specification already supports legacy 802.11 in its basic version (20 MHz), through the BE period. However, the coexistence between 802.11 nodes and w-SHARP nodes may produce frame collisions that must be explicitly avoided to achieve the required deterministic performance.

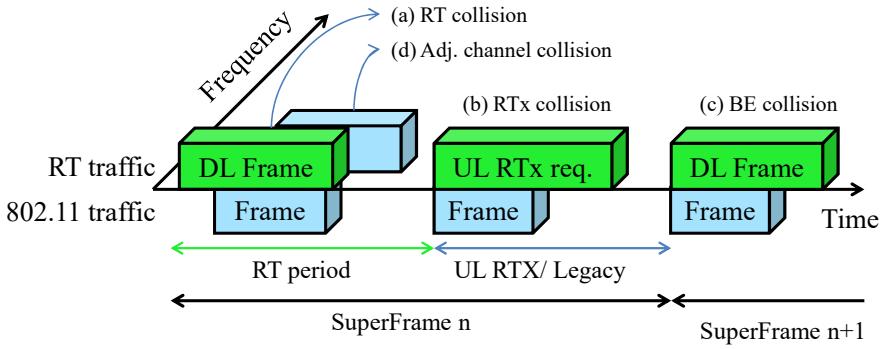


Figure 5.19. Possible frame collisions between 802.11 frames and w-SHARP frames.

There exist four main types of collisions that can occur between w-SHARP nodes and 802.11 nodes. These collisions are summarized in Figure 5.19 and described in the next four statements. The four statements must be accomplished to guarantee collision-free communication and the compatibility of w-SHARP nodes and 802.11 nodes.

a) 802.11 nodes must not gain access to the channel during the RT period. Figure 5.19 collision type (a).

b) There must not be any 802.11 frame transmitted during the RTx UL / BE period if retransmissions are needed in the actual Superframe. Figure 5.19 collision type (b).
c) 802.11 frames transmitted by 802.11 nodes during the BE period must not occupy the RT period. Figure 5.19 collision type (c).
d) There must not be any 802.11 transmission in an overlapping adjacent channel. Figure 5.19 collision type (d).

These four statements can be accomplished through eight mechanisms that are deeply described in the next paragraphs and summarized in Table 5.2. However, the compatibility is obtained at the cost of lower throughput in the RT and BE periods. The mechanisms are described for w-SHARP networks with 20 MHz BW, but it can be generalized to w-SHARP networks with greater BW and more OFDMA channels. It should be noted that, since this compatibility is obtained at the cost of lower throughput, it has not been implemented in the HW prototype.

Table 5.2. Mechanisms to avoid collisions between w-SHARP and 802.11.

ID	**Name**	**Collision type**	**Description**
1	CTS-to-self	(a), (b)	Refreshes the NAV of 802.11 nodes to the duration of the RT period to avoid transmissions during the RT period.
2	SIGNAL symbol in DL frame	(a), (b)	Blocks 802.11 nodes into the receiving state.
3	PHY carrier sense during the RT period	(a), (b)	802.11 nodes detect energy in the channel (busy channel).
4	BECP before RT period	(c)	Controls the transmission of 802.11 frames during the end of the BE period.
5	802.11 frames max duration	(c)	Shortens the BECP and improve traffic predictability.
6	SIFS in UL RTx request	(b)	Prioritizes UL RTx requests against BE frames.
7	Deactivate Fragmentation	(c)	Improves predictability as fragment burst are deactivated.
8	Smart Frequency Planning	(d)	Eliminates adjacent channel interferences.

1) Virtual Carrier Sense blocking through Clear-To-Send (CTS)-to-self

Before the start of every Superframe, the w-SHARP AP must send at least one Clear-To-Send (CTS) frame at a low 802.11g rate. The content of the CTS frame must be:

- The duration field must be set to the duration of the RT period of the next Superframe plus the remaining time to the start of the Superframe.
- The destination MAC Address field must be set to the AP MAC Address.

The duration field of the CTS frame indicates to every 802.11 node that has received the frame that they must set their NAV [29] to the value of the duration field. This length is specified in mechanism No. 5) BECP before the RT period.

This mechanism is used to avoid type (a) collisions. However, the frame may not be correctly demodulated by some 802.11 nodes due to bad channel conditions, thus two additional mechanisms are included.

2) SIGNAL symbol in DL frame

Wireless SHARP long preamble Waveform, which is used in DL frames, is compatible with 802.11g. The third field of the preamble, the SIGNAL symbol [29], is used by 802.11g nodes to know, among the frame decoding information, the transmitted frame airtime duration. The frame airtime duration can be used to block 802.11g nodes in receiving mode and to deny their access to the wireless channel during the RT period. To do so, the MCS field of the SIGNAL symbol is set to 0 (lowest PHY rate, 6 Mbps), while the length field value is such that 802.11 nodes will detect that the first DL frame airtime duration is equal to the RT period. This erroneous frame will be discarded thanks to the FCS field [29]. This mechanism is used in a similar way in 802.11n [29] to block 802.11g stations during the transmission of 802.11n frames. Eq. (5.4) is used to calculate the length field

$$length = \left\lfloor \frac{\frac{T_{RT}}{T_{OFDMS}} \cdot NBPS - N_{SB} - N_{PB}}{8} \right\rfloor, \tag{5.4}$$

where T_{RT} is the duration of the entire RT period, T_{OFDMS} is the duration of each OFDM symbol (4 µs), *NBPS* is the number of bits in each OFDM symbol (24 bits with MCS = 0), and being N_{SB} and N_{PB} service and padding zeros, which values are 16 and 6 respectively. The operator $\lfloor \cdot \rfloor$ is the operator floor, which rounds down the value to the biggest integer value lower than or equal to its argument value.

3) PHY Carrier sense in RT period

PHY Carrier sense is a mechanism included in 802.11 to decide whether the channel is busy or idle by continuously reading the energy in the RF channel. As the RT period is fully occupied by wireless transmissions, the channel should appear to be busy during the duration of the RT period for 802.11 nodes. The 802.11 standard specifies that the minimum power detected to decide if the channel is busy is –62 dBm, which is a relatively high Rx power.

4) Best-Effort Controlled Phase (BECP) before RT period

The BECP is a small period between the end of the BE period and the start of the first RT period which is used to avoid the type (c) collision and to ensure the transmission of at least one CTS frame. To do so, the w-SHARP AP must gain access when the BECP starts. To do so, if the channel is idle at the start of the BECP, the AP must start sending a burst of CTS frames. On the other hand, if there is a current transmission, the AP must wait for the end of the current transmission (including ACK). The BECP duration must be at least equal to the time taken to transmit a frame, its ACK, and a CTS frame plus 3 SIFS

$$t_{CP} = t_{frame} + t_{ACK} + 3 \cdot t_{SIFS,sta} + t_{CTS}. \tag{5.5}$$

5) 802.11 frames maximum duration

The maximum 802.11 frame size must be small to ensure that the frame exchanges do not exceed the duration of the BE period. Two mechanisms can be used to force 802.11 nodes to send small frames:

- *802.11 basic rate configuration:* The w-SHARP AP can force the nodes to use a minimum MCS (e.g. MCS = 2) during the authentication process to improve the network mean bitrate.

- *Maximum Transmission Unit (MTU) size at the link layer:* it defines the maximum packet length that can be encapsulated into a MAC frame. The MTU size configuration must be done manually in every 802.11 node. However, an industrial network should be pre-engineered, thus 802.11 nodes connected are always pre-configured.

The frame duration should be correctly chosen to let 802.11 nodes gain access to the channel. In any case, the size of the Legacy period must be higher than the consumed airtime to perform a frame exchange.

6) Use of SIFS in UL RTx request frames

Once 802.11 nodes have ended the decoding of the main DL frame, they are free to gain access to the medium using the CSMA/CA mechanism, waiting an Extended Interframe Space (EIFS) period (88 µs), due to the frame they have demodulated is erroneous, plus a backoff period [29]. Hence, if an RTx request must be performed, it must be sent in less than an EIFS period. A standard SIFS period (16 µs) is a suitable option, as it is enough time to demodulate the last UL frame and generate the RTx request, ensuring no 802.11 node can transmit. The SIGNAL symbol must be filled identically as stated in mechanism 2), but T_{RT} must be equal to the time that will be consumed by the RTx request plus retransmissions.

7) Deactivate Fragmentation

The dot11FragmentationThreshold parameter must be set to its maximum value to avoid possible interferences caused by fragments bursts at the end of the BE period.

8) Smart frequency planning

Finally, in order to avoid co-channel interferences, the SHARP APs in the same channel or in overlapping adjacent channels must be spatially separated to avoid interferences between wireless networks.

5.3. Implementation architecture of the SHARP technology

The SHARP technology integrates TSN and w-SHARP within a single network with end to end time-sensitive capabilities. Figure 5.20 depicts an example of a SHARP network with four w-SHARP STAs and two SHARP APs, which are further communicated with an Ethernet network with TSN capabilities. The w-SHARP STAs comprise three elements, the w-SHARP NIC, the control SW, and the group of sensors and actuators connected to the w-SHARP STA. The w-SHARP APs are a bridge between w-SHARP (wireless TSN) and wired TSN. Consequently, the w-SHARP APs include a w-SHARP NIC, a TSN NIC, and a bridge to schedule the RT frames between the TSN domain and the w-SHARP domain. Additionally, it also includes a configuration SW which is responsible for the coordination of the NICs and the bridge. The elements of the implementation architecture are briefly described below.

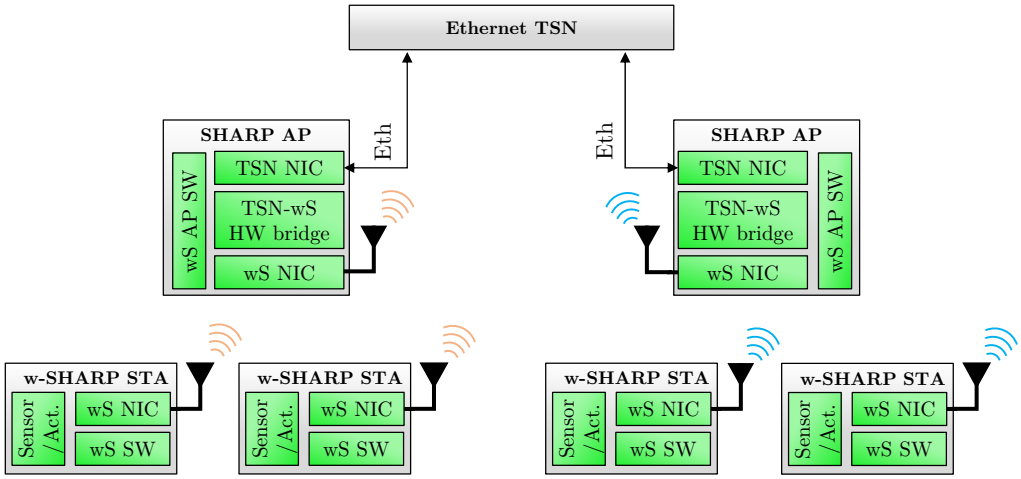


Figure 5.20. Elements of the w-SHARP (wS) architecture.

- The w-SHARP NIC performs the wireless communication according to the specification of the w-SHARP protocol, which includes transmission/reception of RT and BE frames (PHY), superframe and periods execution and scheduling of RT frames (MAC), and wireless time synchronization. It is half-implemented in HW and half-implemented in SW. The HW contains the specification of the PHY/MAC and the SW is a RT application that controls the correct operation of the HW.

- The TSN NIC is the wired counterpart of the w-SHARP NIC and it performs the wired communication according to the TSN standard. It is fully implemented in HW.
- The TSN – w-SHARP HW bridge performs the translation of the frames from the TSN domain to the w-SHARP (wireless TSN domain). It includes the classification of the data packets according to their MAC headers, the MAC header replacement between the Rx and the Tx domains, and the transmission of the packets to the destination NIC. It is implemented in HW.
- The w-SHARP STAs SW includes the configuration of the sensor/actuator, the initial configuration of the w-SHARP NIC, and the high-layer functionalities, such as the automated network attaching, frame rescheduling, etc.
- The w-SHARP AP SW implements the initialization of the w-SHARP NIC, the TSN NIC, and the TSN – w-SHARP HW bridge, and the overall coordination of the elements to ensure the scheduling of frames through the AP with minimum latency.
- Finally, the sensor/actuator in the w-SHARP STAs measure a variable of the physical process or the actuation over the physical process.

At this moment, several COTS devices implement the TSN specification, for instance, switches, or IP modules for FPGA [120]. Nonetheless, the implementation of w-SHARP (wireless TSN) does not exist and, naturally, nor the integration of wireless TSN and TSN. Consequently, the implementation challenges in the SHARP technology are specifically related to the implementation of the w-SHARP protocol and to the integration of w-SHARP and TSN.

The design and implementation efforts during this thesis have been mainly focused on the wireless domain. Consequently, only the w-SHARP NIC has been implemented, whereas the rest of the modules are subject to future work. Even so, some of the modules are currently being implemented or integrated. For instance, the integration of a commercial TSN IP core [120] into the w-SHARP prototype and the implementation of the TSN – wS HW bridge.

The structure of this subsection is as follows. Subsection 5.3.1 overviews the w-SHARP NIC implementation. Subsection 5.3.2 describes the implementation of the w-SHARP PHC. The MAC scheduler is described in Subsection 5.3.3. The MAC comprises several modules, one for each period, which are detailed in Subsection 5.3.4. The implementation of the PHY layer is described in Subsection 5.3.5. Finally, the control SW of the w-SHARP NIC is reported in Subsection 5.3.6.

5.3.1. Implementation Overview of the w-SHARP NIC prototype

The implementation architecture of a w-SHARP NIC is depicted in Figure 5.21. It is divided into two main elements: HW and SW. The HW contains the w-SHARP PHY, MAC, and timing control (PHC). The PHY includes the interface with the RF chip, which converts the IQ samples into an RF signal. The SW is a RT application developed over FreeRTOS. The RT application performs the HW configuration and the data I/O. To do so, the RT application initially receives a high-level configuration structure that is used to initialize the HW. In a more advanced implementation of the w-SHARP STA/AP, the configuration structure shall be generated by a higher SW layer, such as an application running over a general-purpose operative system (e.g. Linux).

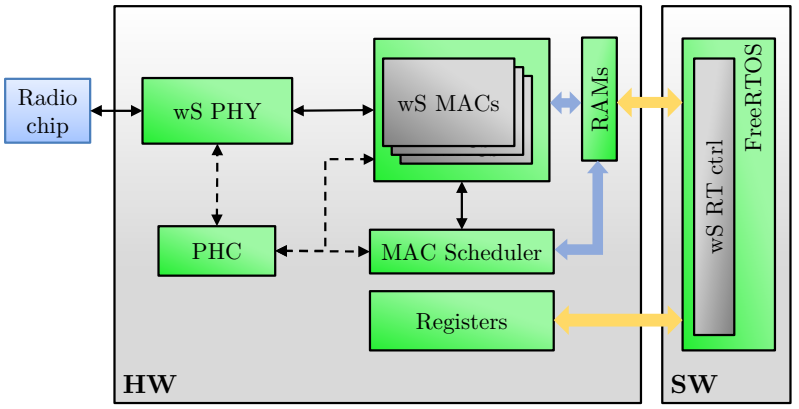


Figure 5.21. Block diagram of a w-SHARP node.

Two main interfaces between the HW and the SW can be distinguished in Figure 5.21 (yellow arrows): the RAM interface (HW-SW) and the register interface (HW-SW). The RAM interface is used to perform the data exchange between the SW/HW, and to perform the low-level configuration of the w-SHARP HW, such as the configuration of the scheduling and time-slots. The register interface carries the most basic configuration, e.g. the position within the RAM used to write the scheduler configuration structure. Both interfaces are implemented through an Advanced eXtensible Interface (AXI) bus.

Since the purpose of the implementation is the validation of the low-latency and deterministic behavior of the w-SHARP technology, the w-SHARP NIC prototype has some limitations. For instance, the prototype is limited to the w-SHARP PHY version with 20 MHz and some of the MAC periods (RT DL, RT UL, and BE). The implementation of further features of the w-SHARP specification is subject for future work. For instance, larger BW and OFDMA are being considered as the next main features of the w-SHARP NIC.

The next subsections describe the modules implemented in the prototype. Subsection 5.3.2 details the implementation of the PHC. Subsection 5.3.3 describes the scheduler operation, which is responsible of the switching between periods. The w-SHARP MAC modules are described in Subsection 5.3.4. Subsection 5.3.5 presents the w-SHARP PHY with 20 MHz BW. Finally, the RT app running over FreeRTOS is detailed in 5.3.6.

5.3.2. w-SHARP PTP Hardware Clock

As the PHC described in Chapter 3, the w-SHARP PHC is a free-running clock that can be synchronized to an external time source. The clock has two time outputs and a PPS output: the first time output goes from 0 to 10^9 ns, and it is called the global timer. The second time output is called the superframe timer and has a configurable period multiple of 100 µs. The superframe timer is used by the MAC scheduler to switch between different periods and by the PHY to deterministically transmit the w-SHARP frames. The implementation of the PHC in w-SHARP is similar to the PHC described in Subsection 3.5.4. The only

significant difference is that this PHC includes a second output which is used to synchronize the operation of the w-SHARP HW.

5.3.3. MAC Scheduler

The MAC Scheduler is the element responsible for the switching between the w-SHARP MAC blocks and provides the time-aware operation. It receives a configuration structure from the SW layer named the scheduler configuration structure. The configuration structure contains the superframe duration, the periods inside the superframe (DL/UL, etc.), and their start time and duration and it is used to switch between different MAC blocks at a specific time. The MAC scheduler operation is similar to the gate control list of TSN, as it decides which periods are allowed to transmit/receive.

The configuration structure of the MAC scheduler contains two substructures: the *general scheduler configuration*, and the *basic period configuration* (Table 5.3). The *general scheduler configuration* contains the number of subperiods of the superframe and the subframe periodicity T_s. The *basic period configuration* structure defines the basic parameters of each period: period type, starting time in nanoseconds, its duration, and a pointer to the initialization information of the period. The periods are implemented by different MAC modules, which are described through the next subsection.

Table 5.3. Scheduler configuration structure.

General scheduler configuration			
Field	**Size [bytes]**	**Data type**	**Description**
superframe period (T_s)	1	Unsigned int	This value multiplied by 100 µs represents the PHC superframe periodicity and thus the superframe duration.
No. periods	1	Unsigned int	No. periods inside the superframe.
Basic Period configuration			
Field	**Size [bytes]**	**Data type**	**Description**
Period Type	1	Unsigned int	Type of MAC block used in the actual period (Tx, Rx, BE, or Sync).

Start time	4	Unsigned int	Start time in ns from the start of the superframe.
Duration	2	Unsigned int	Duration in ns of the period. If equal to 0 duration is infinite.
Ini ADDR	4	MEM ADDR	It contains the memory ADDR that includes the initialization information for the period. Unused in BE period.

5.3.4. Medium Access Control layer modules

The w-SHARP MAC blocks serve as a scheduling interface between the PHY and the SW. The w-SHARP MAC contains 4 MAC blocks: The Tx MAC, the Rx MAC, the BE MAC, and the Sync MAC. Their operation is described in detail through subsections 5.3.4.1 to 5.3.4.4. Only one MAC block can be enabled at the same time and the MAC scheduler is responsible for the coordination between the MAC blocks.

The interface between the SW, MAC blocks, and PHY uses four messages named descriptors: *Tx Desc, Tx End Desc, Rx Desc, and Rx End Desc.* The descriptors and their functionality are summarized in Table 5.4 and further specified in Subsection 5.3.4.5.

Table 5.4. Summary of w-SHARP descriptors.

Class	Orig	Dest	Specified in	Description
Tx Desc	SW	HW	Table 5.5	Defines the configuration of a frame and subsequent subframes that are going to be transmitted.
Rx Desc	SW	HW	Table 5.6	Defines the configuration of a frame and subsequent subframes that are going to be received.
Tx End Desc	HW	SW	Table 5.7	Contains the metadata information of a transmitted frame
Rx End Desc	HW	SW	Table 5.8	Contains the metadata information of a received frame.

5.3.4.1. Tx MAC

The Tx MAC implements the functionalities of a DL period in the case of a w-SHARP AP or a UL period in the case of a w-SHARP STA. It performs the next tasks in chronological order:

1) Read the *Tx Desc* from the pointer written in the basic period configuration structure, which has been previously configured by the SW application,
2) Read the payloads of the frames that are going to be transmitted based on the data pointers written in *Tx Desc*,
3) Write the *Tx Desc* and the data into the PHY,
4) Wait until the PHY finishes the transmission and receive the transmission confirmation from the PHY,
5) Generate and write the *Tx End Desc* to the RAM that will be read by the SW, and
6) Generate an interruption to notify the SW at the end of the period.

5.3.4.2. Rx MAC

The Rx MAC implements the functionalities of a UL period in the case of a w-SHARP AP or a DL period in the case of a w-SHARP STA. It performs the next tasks in chronological order:

1) gather the *Rx Desc* of the Rx subframes from the RAM interface (previously written by the SW application),
2) configure the Rx PHY based on the *Rx Desc* received from the SW,
3) wait until the Rx PHY receives the frames from the specific slots,
4) generate the *Rx End Desc*,
5) write the received data and the *Rx End Desc* to the RAM that will be read by the SW, and
6) generate an interruption to notify the SW at the end of the period.

5.3.4.3. BE MAC

The BE MAC implements a compliant 802.11 MAC with the modifications to the NAV stated in Subsection 5.2.4.6 in order to avoid interferences between the frames transmitted in different periods. The BECP subperiod has not been

included in the BE MAC implementation because the coexistence with 802.11 nodes is rather limited.

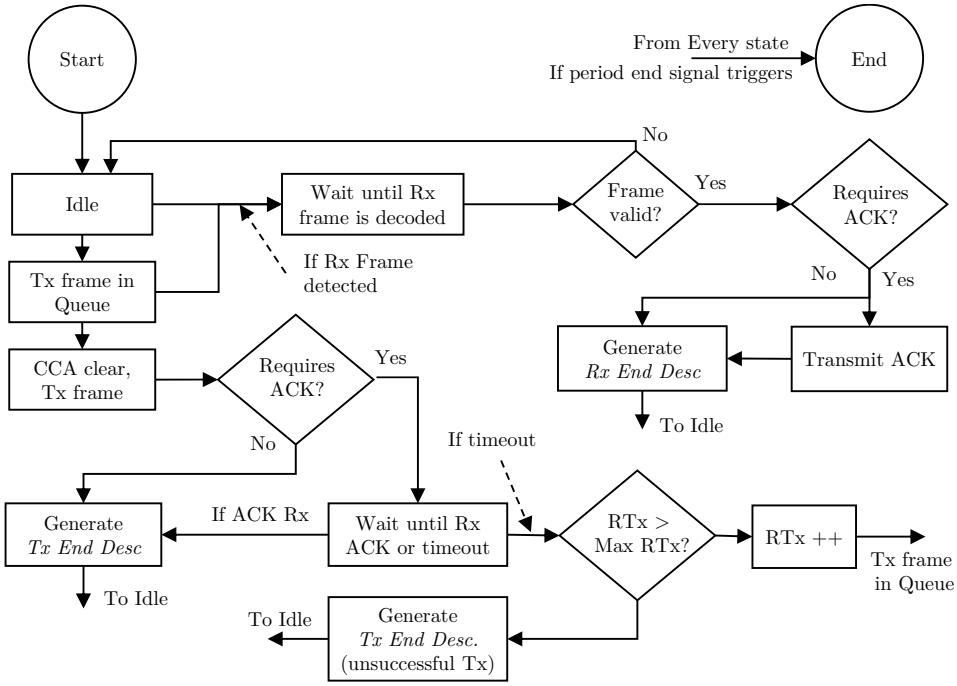


Figure 5.22. State flow implemented in the BE MAC.

The BE MAC implements the state machine depicted in Figure 5.22. The BE MAC functions are asynchronous, so it waits for specific events to perform actions. The main events are a Tx petition, the detection of an incoming frame, or the end of the BE period. The BE MAC comprises two tasks, the Tx task, and the Rx task.

Tx task

The BE MAC is in Idle state until a BE Tx petition is sent to the HW. At that moment, the BE MAC gathers the *Tx Desc* that describes the frame structure and waits until the Clear Channel Assessment (CCA) is idle, which means that the channel is free and the node can transmit. Once the CCA is triggered, the frame is transmitted. In the case that the frame does not require an ACK, and once the frame has completely transmitted, the BE MAC generates a *Tx End*

Desc to indicate the end of the transmission and it goes back to the Idle state. In the case that the frame requires an ACK, the BE MAC will wait until an ACK is received, or for an ACK timeout. If the ACK is received, the BE MAC will generate a *Tx End Desc* and it will go to the Idle state. If the ACK is not received, the BE MAC will generate a timeout signal to indicate the unsuccessful transmission of the frame. Then, the BE MAC will evaluate the number of attempted retransmissions. On the one hand, if the frame has not reached the maximum number of retransmissions, the BE MAC will try to transmit the frame again. On the other hand, if the frame has already reached the maximum number of allowed transmissions, the BE MAC will generate a Tx End Desc to indicate that the frame was unsuccessfully transmitted, and it will go to the Idle state.

Rx task

The Rx task starts when the PHY layer detects an incoming frame and it can occur during the idle or Tx frame in the queue state. When the PHY notifies the BE MAC that a frame is being demodulated, it must wait until the frame demodulation finishes. If the incoming frame is not correctly demodulated, or its destination ADDR does not match the MAC ADDR of the node, the frame is discarded and considered as invalid. If the frame is considered valid, the BE MAC will check if the frame requires an ACK based on its format. If it does not require an ACK, it will generate a Rx End Desc to indicate to the SW that a frame has been received along with the data of the frame. If it requires an ACK, the BE MAC will generate a trigger to the PHY to transmit an ACK. Once the ACK is transmitted, a Rx End Desc is generated and the BE MAC goes back to Idle state.

Finally, the BE MAC operation can be interrupted at any moment when the BE period finishes. Thus, the BE MAC will automatically go to the End state if the period end signal is triggered by every state.

5.3.4.4. Sync MAC

The Sync MAC implements the AP discovery and time synchronization. It is used by the w-SHARP STA to detect and demodulate beacons, and to be able to synchronize its local clock to the external one. It performs the next tasks:

1) gather the *Rx desc* that describes the w-SHARP beacon structure,
2) wait for a predefined time until a beacon is either detected 3) or the 4) timeout triggers,
3) send a *Rx End Desc* to the SW and the beacon data to indicate that the beacon was correctly detected.
4) send a *Rx End Desc* to the SW to indicate that no beacon was detected.

In the case that a beacon is detected, the SW will use the timestamp information to correct its local clock and it will run the sync MAC operation again until the w-SHARP STA is synchronized to the AP.

5.3.4.5. Specification of descriptors

1) Tx Desc

The *Tx Desc* is written by the SW to the HW and specifies the structure of a Tx frame, including all the details to correctly generate the subsequent subframes. Given that a frame can hold an arbitrary number of subframes, the *Tx Desc* is divided into two sub-descriptors, the *Main Tx Desc,* and the *Sub Tx Desc.* The *Main Tx Desc* contains the general information about the frame common to all subframes (Tx instant, Tx power, number of subframes, preamble type, etc.), whereas the *Sub Tx Desc* contains the specific information about each subframe. Table 5.5 gathers the parameters of the *Tx Desc* and describes their functionalities.

Table 5.5. *Tx Desc* Structure.

Main Tx Desc

Field	Size [bytes]	Data type	Description
Tx time	4	Unsigned int	Tx time of the frame relative to the start of the current superframe. It is expressed in ns.
Tx type	1	Unsigned int	Type of frame transmitted. '0' for DL frames, '1' for UL frames, and '2' for BE frames.
Frame ID	1	Unsigned int	Frame identifier used only for control purposes. It is returned in the *Tx End Desc.*
N subframes	1	Unsigned int	No. subframes within the actual DL frame. Set to '1' for UL frames and '2' for BE frames.
Frequency offset	2	Signed int	Frequency offset in Hz with respect to the carrier frequency. Used by the STA to perform the Tx equalization in UL frames.
Tx power	1	Signed int	Transmission power of the frame in dBm. Used by the STA to perform the Tx equalization in UL frames.
No. OFDM symbols	2	Unsigned int	No. of OFDM symbols of the frame. Sum of all the OFDM symbols of the subframes.

Sub Tx Desc

Field	Size [bytes]	Data type	Description
ID subframe	1	Unsigned int	Frame identifier used only for control purposes. It is returned in the *Tx End Desc.*
Subframe length	2	Unsigned int	Length in bytes of the subframe. Its value ranges from 4 bytes to 2500 bytes.
MCS	1	Unsigned int	MCS used for the subframe. Its value ranges from 0 (BPSK 1/2) to 7 (64-QAM 3/4) as in 802.11g [29].
Data ADDR	4	RAM ADDR	Position in the RAM used to store the data corresponding to the actual subframe.

2) Rx Desc

The *Rx Desc* is written by the SW to the HW and specifies the structure of a Rx frame, including all the details to correctly receive the subsequent subframes of the frame. Since a frame can hold an arbitrary number of subframes, the *Rx Desc* is divided into two sub-descriptors, the *Main Rx Desc,* and the *Sub Rx Desc.* The *Main Rx Desc* contains the general information about the frame common to all subframes (Tx instant, Tx power, number of subframes, etc.),

whereas the *Sub Rx Desc* contains the specific information about a subframe. Table 5.6 includes the *Rx Desc* parameters and describes their functionalities.

Table 5.6. *Rx Desc* Structure.

Main Rx Desc

Field	Size [bytes]	Data type	Description
Rx time	4	Unsigned int	Tx time of the frame in ns relative to the start of the current superframe.
Time Sync tolerance	1	Signed int	Maximum allowed desynchronization between the w-SHARP AP and w-SHARP STA. Expressed as a multiple of the sampling period *T*.
Rx type	1	Unsigned int	Type of frame transmitted. '0' for DL frames, '1' for UL frames, and '2' for BE frames.
Frame ID	1	Unsigned int	Frame identifier used only for control purposes. It is returned in the *Tx End Desc.*
N subframes	1	Unsigned int	No. of subframes from the frame that must be demodulated by the w-SHARP node.
Skip frequency offset corr.	1	Boolean	Set to '1' in the UL period at the w-SHARP AP to skip the frequency offset correction.
No. OFDM symbols	2	Unsigned int	No. of symbols of the frame.

Sub Rx Desc

Field	Size [bytes]	Data type	Description
ID subframe	1	Unsigned int	Frame identifier used only for control purposes. It is returned in the *Tx End Desc.*
Subframe length	2	Unsigned int	Length in bytes of the subframe. Can present values from 4 bytes to 2500 bytes.
MCS	1	Unsigned int	MCS used for the subframe. Can present values from 0 (BPSK 1/2) to 7 (64-QAM 3/4).
No. OFDM symbols subfr.	2	Unsigned int	No. OFDM symbols of the subframe.
Subframe position	2	Unsigned int	Position of the subframe within the frame. Expressed in No. of OFDM symbols from the start of the frame.
Data ADDR	4	RAM ADDR	Position in the RAM used to store the data corresponding to the actual subframe.

3) Tx End Desc

The *Tx End Desc* is returned by the MAC to the SW to indicate the correct transmission of a frame along with the metadata information of the transmission. The structure is defined in Table 5.7.

Table 5.7. *Tx End Desc* structure.

Field	Size [bytes]	Data type	Description
Frame ID	1	Unsigned int	Frame identifier used only for control purposes. It has the value introduced in the *Tx Desc.*
Timestamp	4	Signed int	Instant of the transmission of the frame in ns. In the case of a deterministic frame, it matches the value of the Tx time of the *Tx Desc.*

4) Rx End Desc

The *Rx End Desc* is returned by the MAC to the SW to indicate the reception of a subframe and includes the configuration information of the subframe. Its fields are described in Table 5.8.

Table 5.8. *Rx End Desc* structure.

Field	Size [bytes]	Data type	Description
Status	1	Unsigned int	For a deterministic transmission, identifies if the frame has been detected without errors '0', detected but with errors '1', or non-detected '2'.
Frame ID	1	Unsigned int	Frame identifier used only for control purposes. It has the value introduced in the *Rx Desc.*
Subframe ID	1	Unsigned int	Frame identifier used only for control purposes. It has the value introduced in the *Rx Desc.*
Frame type	1	Signed int	Type of frame transmitted. '0' for DL frames, '1' for UL frames, and '2' for BE frames.
Timestamp	4	Unsigned int	Instant of the detection of the frame in ns.
Rx Power	1	Signed int	Estimated Rx power in dBm.
Est. frequency offset	2	Signed int	Estimated Frequency offset in Hz with respect to the carrier frequency.
MCS	1	Unsigned int	MCS used for the subframe. Can present values from 0 (BPSK 1/2) to 7 (64-QAM 3/4).
Subframe length	2	Unsigned int	Length in bytes of the subframe. Can present values from 4 bytes to 2500 bytes.
Data ADDR	4	RAM ADDR	Position in the RAM used to store the data of the subframe. obtained from the corresponding *Rx End Desc.*
Est. CIR	64	Signed 16-bit integers	IQ representation of the CIR in the frequency domain. Each 4 bytes represents the complex gain of one subcarrier.

5.3.5. Physical layer

The w-SHARP PHY is depicted in Figure 5.23. The coder and modulator is the one used in 802.11g [29] and includes all the processing steps of the standard: (de)scrambling, convolutional (de)coder, (de)interleaving, Quadrature Amplitude Modulation (QAM) (de)modulation, carrier (de)mapping, and FFT/IFFT. The (de)coder supports the next modulations: BPSK, Quadrature Phase Shift Keying (QPSK), 16 QAM, and 64 QAM, combined with a convolutional code with redundancy of 1/2, 2/3, or 3/4. The payload length can

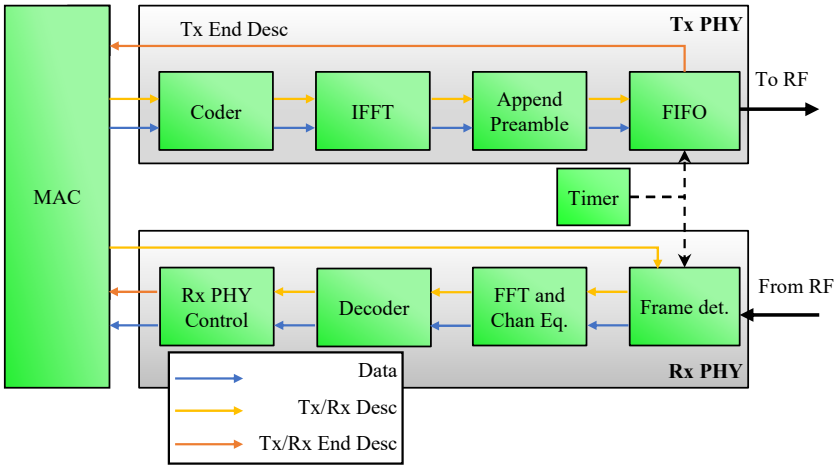


Figure 5.23. Block diagram of the w-SHARP PHY.

be configured from 4 bytes to 2500 bytes. The frames include a 4 bytes Cyclic Redundancy Check (CRC) to check the frame integrity after its demodulation. The FFT size is set to 64 carriers, 48 of them are used for data, 4 are used for pilots, and the rest are used as band guards. The DC subcarrier is set to null. The preamble depends on which type of frame is transmitted.

The Tx PHY operation is as follows. The Tx PHY receives a *Tx Desc* with the frame and subframes configuration along with the payload of each subframe. The data is sent to the coder, which is configured according to the *Sub Tx Desc* of each subframe and performs the scrambling, channel coding, interleaving, and QAM modulation. The data resulted from the coder is sent to the IFFT module and the output of the IFFT module is append with the preamble, which is specified in the *Main Tx Desc*. Finally, the resulted frame is stored in a First In, First Out (FIFO) memory until the PHC superframe output matches the *Main Tx desc* Tx time. Finally, the Tx PHY generates a *Tx End Desc* to indicate that the frame was transmitted.

The Rx PHY design is more complex since it involves the frame detection and channel compensation. First, the Rx PHY receives from the MAC an *Rx Desc* that includes the frame and subframes format and the expected Rx time. The frame is detected through the frame detector block which includes an energy

detector, a frequency offset corrector, and a correlator to detect the start of the frame. Once the frame is detected, it is sent to the FFT and Channel equalizer block. The channel equalization is performed in the frequency domain and implements the gain/phase estimation and equalization and pilot tracking. Its output is connected to the decoder that is configured by the *Sub Rx Desc* of each subframe and performs the QAM demodulation, deinterleaving, channel decoding, and descrambling. Finally, the Rx PHY sends the data to the MAC along with an *Rx End Desc*. If the frame is not detected, the Rx PHY will generate an *Rx End Desc* to notify the MAC that the frame was not detected.

5.3.6. RT application over Free RT Operative System (FreeRTOS)

The control of the HW is performed using an RT application running over FreeRTOS. The RT application carries out the initial HW configuration and the data I/O from the SW to the HW. The RT app has two operation states: the unsync state (STA only), and the execution state (STA and APs). The Sync state is used by the STA to synchronize its local time to the local time of the AP, whereas the execution state performs the exchange of data through the network. The STAs are initialized in the unsync state and they move to the execution state after they are time synchronized to the AP. The w-SHARP AP skips the unsync state and is directly initialized in the execution state.

Unsync state

The w-SHARP STAs are initialized in the unsync state. The unsync state generates a configuration structure that includes the beacon frame format (MCS and length), and the carrier frequency f_c. The configuration structure is used to configure the Sync MAC, which will perform the search of the beacon frame. If the beacon is detected, the Sync MAC generates a Rx end desc with the beacon information including timestamps. The timestamps are used to adjust the local time to the w-SHARP AP time. This process is repeated several times until the w-SHARP STA is synchronized to the w-SHARP AP. Finally, the w-SHARP STA switches to the execution state once the time synchronization has converged.

Execution state

The execution state is used by the STAs and AP to perform the exchange of data. This state is initialized by a high-level structure that contains the superframe format, including length, time slots, etc. The high-level structure is translated by the SW to a low-level structure, which is used to configure the w-SHARP IP through the registers and RAM interfaces. Afterward, the w-SHARP node starts the data exchange.

Figure 5.24 depicts the data flow from the data generation tasks to the HW and from the HW to the data consumption (end) tasks. The data gen tasks are periodically activated with a period equal to the data generation period. When a data gen task is activated, it reads the variable associated with their corresponding sensor and they write the result in a specific position in the DDR memory. The position and data length within the DDR memory is preconfigured in the *Tx Desc* associated with the data. The memory positions are reserved for the specific task and no other task can write in the same positions. The Data I/O task is activated some microseconds before the effective transmission of a frame. This task reads the values of the intermediate memory and writes the data into the RAM SW-HW interface, which is connected to the w-SHARP IP. This intermediate step is used to protect the RAM interface from any unwanted access. Finally, the w-SHARP IP downloads the data from the RAM SW-HW interface according to the configuration of the scheduler and descriptors.

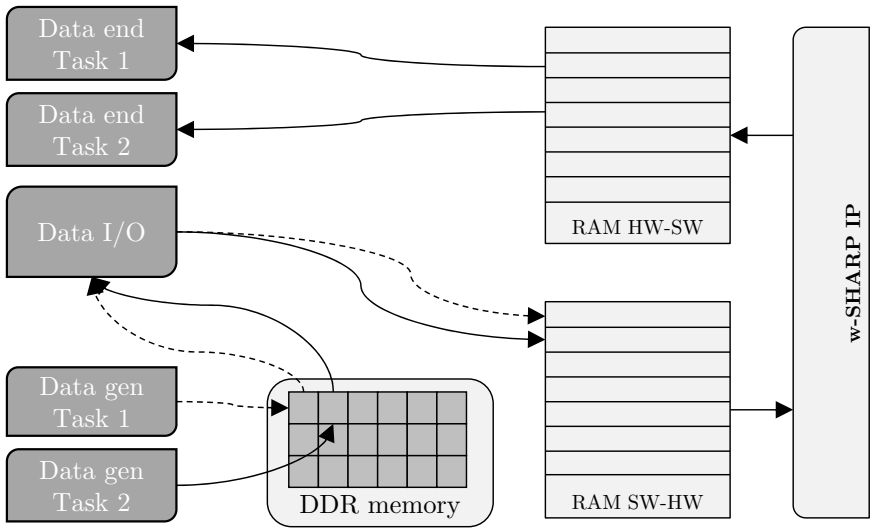


Figure 5.24. Data flow in the w-SHARP NIC during the execution state.

The receiving process is as follows. Whenever a frame is correctly demodulated, the data is stored in the RAM HW-SW memory according to the configuration of the descriptors and scheduler. Then, the w-SHARP IP notifies the SW through the interruption line that a frame has been received. Based on the *Rx end desc* associated with the interruption line, it launches the specific data end task associated with the received data. Finally, the data end task reads the Rx position of the RAM HW-SW interface and consumes the data. In this case, since the w-SHARP prototype does not include any sensor/actuator, the data is simply written into an array used to check the integrity of the data exchanged in the network.

5.4. Evaluation through simulations

This section contains the simulations carried out to evaluate the performance of w-SHARP. The simulations have been divided into achievable cycle time (Subsection 5.4.1), PER evaluation (Subsection 5.4.2), and BE period throughput (Subsection 5.4.3).

5.4.1. Cycle time evaluation and comparison with 802.11ax

This section shows the comparison between the achievable cycle time of existing wireless standards and w-SHARP. Few wireless systems are indeed able to fulfill the targeted cycle time of w-SHARP. For instance, 802.11g/n/ac has low efficiency in the transmission of short packets due to its long preamble [30], and then they cannot support low cycle times. 5G-NR minimum achievable cycle time is in the range of 2 ms even with BW $>$ 100 MHz [33], which is above the pursued sub-millisecond cycle time. 802.11be [37] may be promising, but its standardization process has just started. Then, w-SHARP and 802.11ax are the only suitable candidates for this comparison. To perform a fair comparison between both systems for the targeted applications, we have replaced the 802.11ax MAC with a TDMA similar to the one used in w-SHARP. The 802.11ax MAC could be seen as an RT-WiFi version that uses the 802.11ax PHY [112]. This combination exploits the novel high-efficiency features of 802.11ax, such as the trigger frame and multi-user OFDMA, whereas the TDMA MAC provides a larger degree of predictability. For the sake of clarity, we have considered that the cycle time matches T_s.

The minimum achievable cycle time has been evaluated through simulation analysis using the w-SHARP and 802.11ax specification implemented in MATLAB®. 802.11ax PHY-TDMA has been evaluated through the implementation of the WLAN System Toolbox™ included in MATLAB®. A w-SHARP Superframe and an 802.11ax PHY-TDMA Superframe are plotted in Figure 5.25 (a) and (b). The configuration comprises 1 AP and 9 STAs. The BW has been set to 20 MHz. The MCS is QPSK with channel coding of 1/2 and the data frames have a payload of 13 bytes. At the start of each cycle, a beacon frame with a payload of 8 bytes is transmitted. w-SHARP uses a MAC overhead of 5 bytes. According to this configuration, the w-SHARP Superframe has a duration of $T_S = 359$ µs, which is rather short and quite efficient thanks to its low overhead. On the contrary, 802.11ax PHY-TDMA provides $T_S = 1304$ µs, which is 3.6 times the w-SHARP Superframe duration.

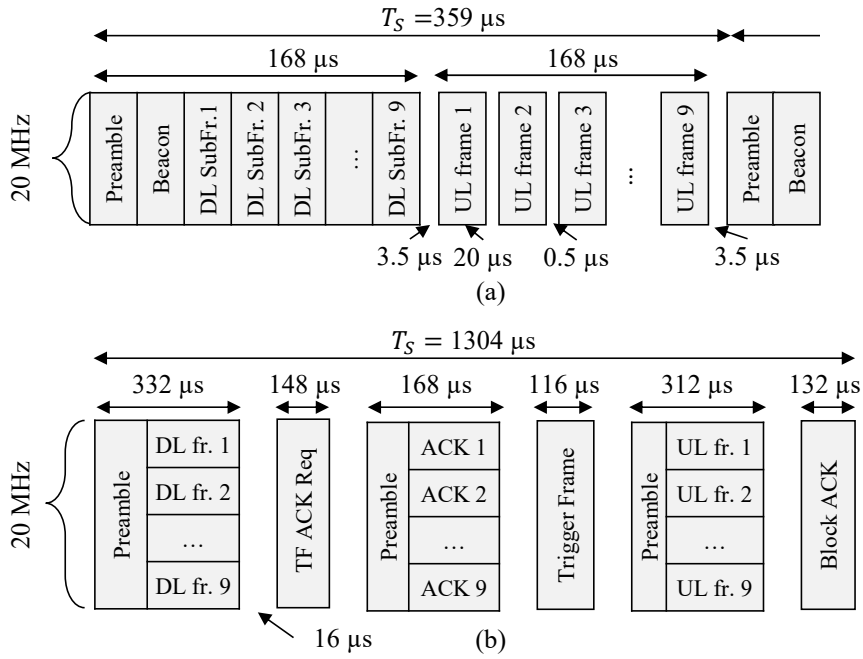


Figure 5.25. w-SHARP Superframe structure (a) and 802.11ax PHY-TDMA Superframe structure (b).

To gain more insights into the performance of w-SHARP, plots the minimum attainable cycle time, both for w-SHARP and 802.11ax PHY-TDMA, in scenarios with an increasing number of nodes using different MCS and two different BW (20 MHz and 160 MHz), and for payload sizes of 13 bytes and 100 bytes respectively. Note that, for any number of simulated nodes and size of the payload, w-SHARP significantly exceeds 802.11ax PHY-TDMA for the same BW. Furthermore, the efficiency gap between both systems increases as the size of the payload decreases. For instance, for small payloads (13 bytes), 20 MHz w-SHARP reaches a performance similar to 160 MHz 802.11ax. Finally, it is worth mentioning that for the highest BW, w-SHARP clearly satisfies the sub-millisecond cycle time for any MSC and No. nodes except for more than 27 STAs, 100 bytes, and BPSK, while 802.11ax may not meet this requirement depending on No. nodes and/or the MSC.

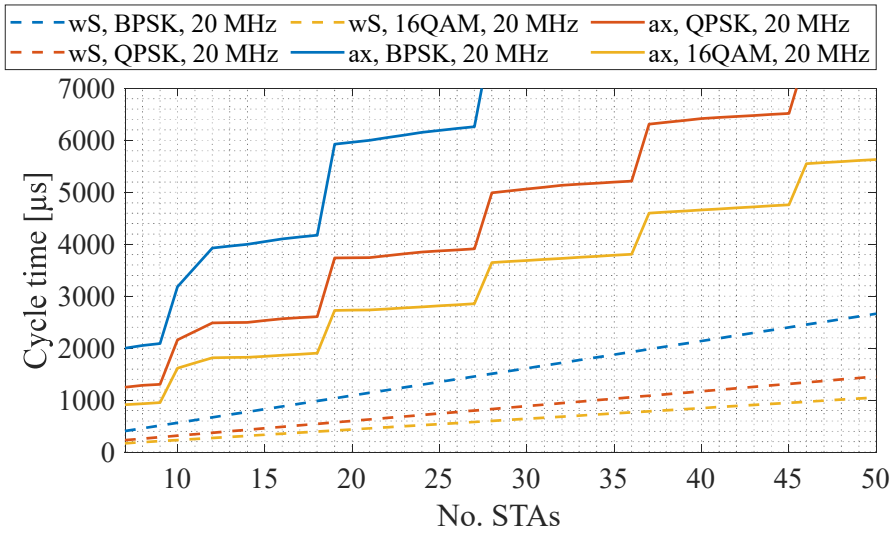


Figure 5.26. Minimum achievable Superframe duration (T_S) vs. the No. nodes of 802.11ax (ax) and w-SHARP (wS) for 20 MHz BW, different modulations, and a payload size of 13 bytes (a) and 100 bytes (b).

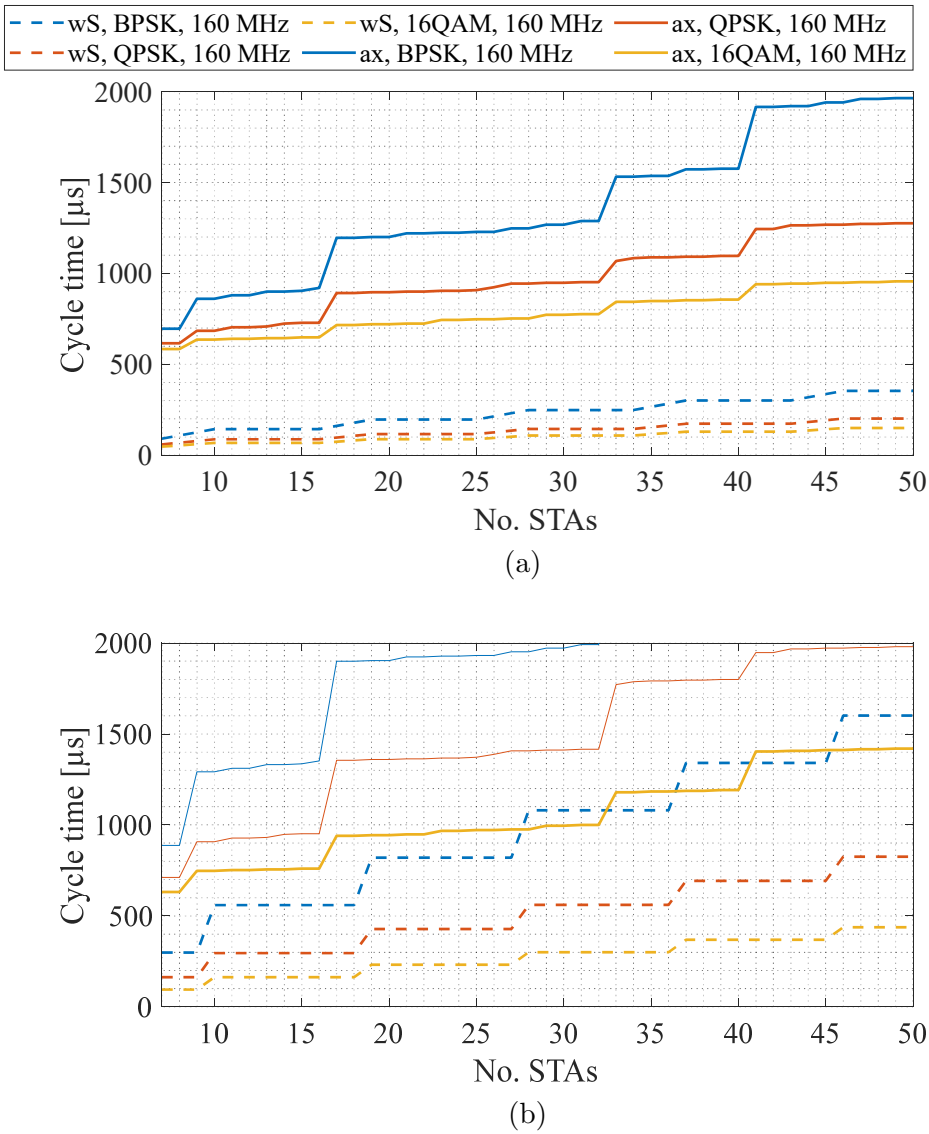


Figure 5.27. Minimum achievable Superframe duration (T_s) vs. the No. nodes of 802.11ax (ax) and w-SHARP (wS) for 160 MHz BW, different modulations, and a payload size of 13 bytes (a) and 100 bytes (b).

5.4.2. PER evaluation

The performance of w-SHARP in terms of PER, BE throughput, and the coexistence of 802.11 nodes in a w-SHARP network has been jointly evaluated using Objective Modular Network Testbed in C++ (OMNeT++) 5.0 and MATLAB®. The simulation has been carried out in two steps. In the first step, the PER curves of the w-SHARP PHY has been obtained in MATLAB® using two different industrial channel models. The PER has been evaluated as a function of the instantaneous P_{Rx} and MCS and includes the frequency selectivity effects due to multipath dispersion, but it does not include the channel fading. The algorithm used to estimate the PER is summarized in Algorithm 5.1. Also, the simulation generates the channel gain vector $g_h(t)$ that includes the channel gain as function of the time. In the second step, the $g_h(t)$ and the PER curves are introduced in OMNET++ and are combined to evaluate the PER as a function of the $P_{Rx,\mu}$ with retransmissions in a real network setup.

Algorithm 5.1. Algorithm to compute the PER as a function of P_{Rx}.

Input: *Channel model*, $P_{Rx,\mu\ min}$, $P_{Rx,\mu\ max}$, N
Output: $PER[P_{Rx}]$
1. for $P_{Rx,\mu} = P_{Rx,\mu\ min}$ to $P_{Rx,\mu\ max}$
2. for k = 0 to $N-1$
   2.1. Obtain a random CIR based on the channel model
   2.2. Transmit a frame through the communication channel
   2.3. Estimate the frame received power P_{Rx}
   2.4. Demodulate the frame and check if it is correctly demodulated or not
   2.5. Classify the result based on P_{Rx}.

The simulations have been carried out over two industrial channel models with Non-Line-Of-Sight (NLOS) conditions and Rayleigh fading (CM8 and Scenario 7 [54], noted here as channel 1 and channel 2 respectively). Channel 1 has an RMS delay spread of 89 ns and channel 2 has an RMS delay spread of 29 ns. The channel models have been generated using the MATLAB® Communications System Toolbox™.

The simulation parameters introduced in OMNeT++ to test the PER performance are summarized in Table 5.9. These parameters represent a common

industrial scenario with millisecond T_S and a slow channel variation. The noise threshold has been chosen according to the 802.11 standard typical threshold [29].

Table 5.9. Simulation parameters to test w-SHARP PER.

Parameter	**Value**	**Unit**
BW	20	MHz
T_S	1	ms
Rx noise threshold	-90	dBm
Channel coherence time	30	ms
P_{Tx}	10	dBm
Fading model	Rayleigh NLOS (Both channels)	

The performance of the system for DL and UL periods has been assessed in terms of raw PER (without retransmissions), and PER with 1 retransmission for both channel 1 and channel 2. The retransmissions are performed with a lower MCS. In the case of 64-QAM ¾, the retransmission uses 16-QAM ½, in the case of 16-QAM ½, the retransmission uses QPSK ½, and in the case of QPSK ½, the retransmission uses BPSK ½. PER results under channel 1 are represented in Figure 5.28, while Figure 5.29 depicts the PER results obtained under channel 2. DL and UL results are represented by the same curve, as there were no significant differences among their PER results.

The results obtained under channel 1 (Figure 5.28) shows that high reliability (PER $< 10^{-7}$) is feasible without and with retransmissions under the condition of high Rx power. PER $< 10^{-7}$ is especially feasible with low-order modulations (BPSK and QPSK) since these modulations offer higher protection against noise. Regarding the PER results with retransmission, the retransmission scheme seems to obtain little improvements with BPSK 1/2 and QPSK 1/2 (1-2 dB), unlike 16 QAM with RTx, which has a gain of 5 - 6 dB compared to 16 QAM without RTx.

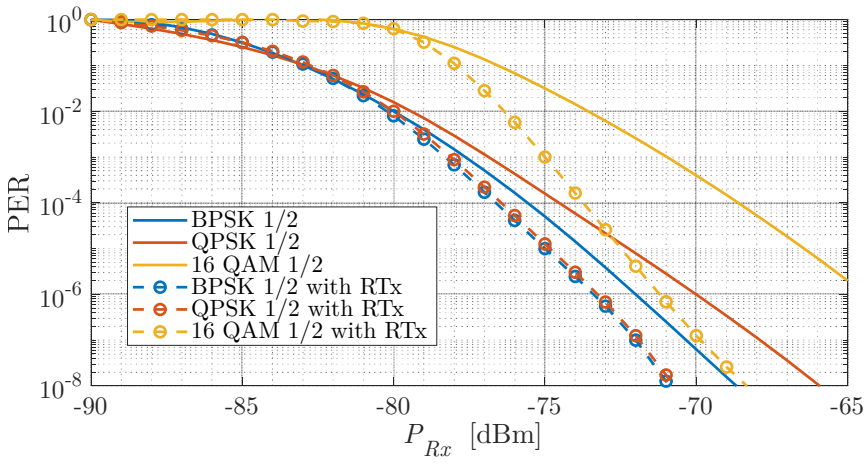


Figure 5.28. PER and PER with retransmissions vs. received power (P_{Rx}) under channel 1.

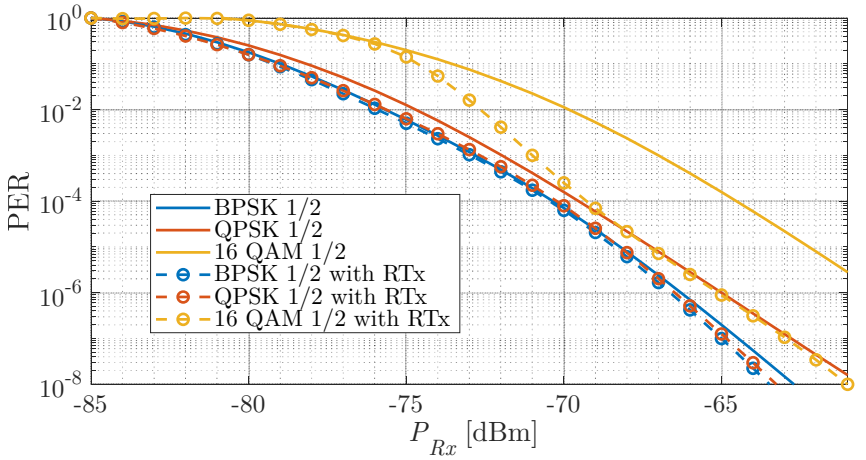


Figure 5.29. PER and PER with retransmissions vs. received power (P_{Rx}) under channel 2.

On the other hand, PER curves obtained under channel 2 (Figure 5.29) have a similar behavior but with a 7 dB shift to the right with respect to channel 1 curves. Channel 2 presents a lower delay spread that, combined with Rayleigh fading, results in a larger fading probability. A large fade strongly influences the

overall PER result, and thus the attainable PER results over channel 2 are worse than the PER results over channel 1.

Two conclusions may be highlighted from these simulations. First, the wireless channel behavior greatly influences the attainable PER and must be taken into account when designing the network, and second, a retransmission scheme in the same frequency band provides little gain because the retransmitted frame is affected by the same wireless channel conditions. The use of different frequency channels to perform retransmissions by using OFDMA and the use of MIMO will be studied in future works in order to increase the system reliability.

5.4.3. Coexistence with 802.11 nodes and BE throughput

The evaluation of the BE throughput and coexistence with 802.11 devices have been done through 3 scenarios, which are described in Table 5.10. Scenario 1 is an IoT scenario, where several nodes transmit small frames, and scenarios 2 and 3 have fewer nodes, which transmit higher amounts of data. Scenarios 2 and 3 only differ in their superframe duration.

Table 5.10. Simulation parameters to test the BE throughput and coexistence with 802.11 nodes.

	Scenario 1	**Scenario 2**	**Scenario 3**
Type of scenario	IoT	Internet	Internet
Number of 802.11 nodes	8	4	3
802.11 payload length	57 B	213 B	500 B
BW	20 MHz	20 MHz	20 MHz
MCS	QPSK ½	64-QAM ¾	16-QAM ½
802.11 frames duration	60 µs	52 µs	188 µs
CTS duration	32 µs	32 µs	32 µs
ACK duration	44 µs	44 µs	44 µs
Min BECP duration	184 µs	176 µs	312 µs
$\boldsymbol{T_S}$	1 ms	1 ms	2 ms

Two sets of simulations to separately study the BE throughput and the coexistence between nodes have been performed. The first set of simulations is

used to study the coexistence between the nodes. In this scenario, T_S has been set to 0.5 ms and the minimum BECP duration has been varied. The second set of simulations have been used to evaluate the BE period throughput. To do so, the BECP duration has been set to its minimum theoretical value to avoid collisions and the T_{RT} has been varied.

The results of the coexistence between the w-SHARP network and the 802.11 nodes are shown in Figure 5.30. As can be seen, when the BECP value is equal or higher than its limit value (represented as squares), the number of collisions is 0 %, which demonstrates that the coexistence between w-SHARP and 802.11 nodes is achievable under specific circumstances.

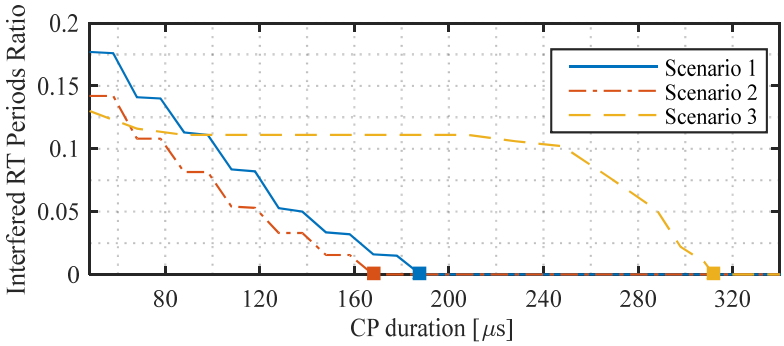


Figure 5.30. Number of collisions as a function of the BECP period duration.

Figure 5.31 depicts the attainable traffic rate for each scenario as a function of the duration of the BE period. As can be derived from the plot, the traffic rate is considerably lower than the achievable rate in 802.11g networks. In addition, it can be also seen in the plot that the traffic rate curves follow a stair shape. The low traffic rate and the shape of the curves are caused by the small amount of 802.11 frames that can be transmitted in each superframe, which depends on the length of the BE period and on the frames maximum size. Although the BE traffic rate is not very high, it could be enough for each scenario when the BE period is large enough: 500 µs for scenario 1 and 2, and 800 µs for scenario 3. Finally, it should be remarked that the traffic can be easily scaled using a w-SHARP AP with larger BW or using more advanced 802.11 coding schemes at the cost of no compatibility with 802.11 devices.

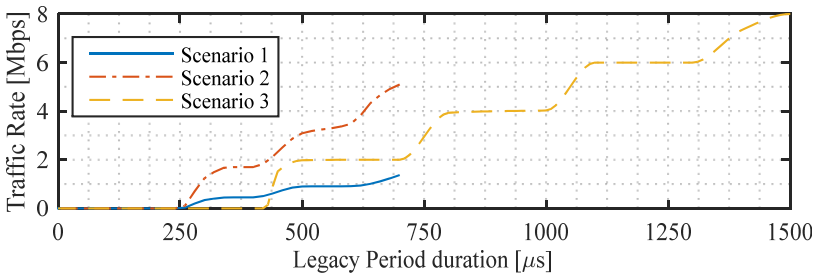


Figure 5.31. Achievable BE throughput with a w-SHARP network with BW = 20 MHz.

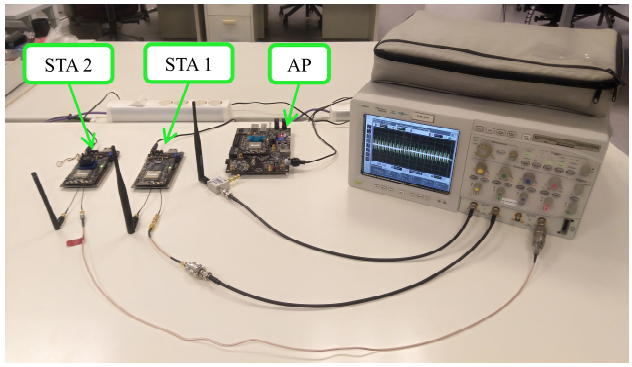


Figure 5.32. Validation setup of the ultra-low cycle time of w-SHARP.

5.5. Evaluation through a HW testbed

The proposed HW architecture described in Section 5.3 has been successfully applied to implement a w-SHARP prototype over the ADI RF SOM platform and using the implementation methodology based on System Generator by Xilinx. The next subsections detail the experiments to validate the node and to test its performance. Note that the w-SHARP prototype is limited to OFDM and 20 MHz BW. Nonetheless, the results with this prototype could be extrapolated to w-SHARP networks with larger BW and more OFDMA channels.

5.5.1. Achievable Cycle time and latency

In order to evaluate the achievable cycle time and latencies, two w-SHARP networks with different No. of STAs, cycle time, and superframe structures have been configured in the HW. The first network was used to test the minimum achievable cycle time and comprised three nodes: one node configured as an AP and two nodes configured as STAs. The second network comprises one AP and five STAs and was used to validate the system behavior under more relaxed cycle time conditions using more STAs and the transmission of non-RT frames.

A photo of the setup for the first network configuration is shown in Figure 5.32. The setup comprises a four-port oscilloscope with 4 GHz BW used to measure the exchanged frames, one AP, and two STAs. The RF output port of the AP is connected to a splitter whose outputs are connected to an antenna and the scope. In the first network, each node was connected to one input of the oscilloscope. In the second network, the AP was connected to the first input of the scope, two STAs were connected to the second input of the scope through splitters, and the two remaining STAs were connected to the third input of the scope.

Table 5.11. Superframe Structure with $T_S = T_{RT} = 100$ µs.

RT DL frames			**RT UL frames**		
Subframe type	**Payload length [B]**	**MCS**	**Subframe type**	**Payload length [B]**	**MCS**
Beacon	14	BPSK ½	-	-	-
DL STA 1	11	16-QAM ½	UL STA1	50	64-QAM ¾
DL STA 2	11	QPSK ½	UL STA2	9	QPSK ½

The superframe structure for the first network is summarized in Table 5.11. For this configuration, the duration of the DL frame is 48 µs, including the beacon and the two data frames, whereas the duration of each of the UL frames is 12 µs. Then, $T_s = 100$ µs can be achieved if $\text{IFS}_{\text{UL-DL}} = 13$ µs and $\text{IFS}_{\text{UL-UL}} = 2$ µs. This setup has been successfully validated in the HW platform. A capture of the oscilloscope during one Superframe is depicted in Figure 5.33. The first frame is the DL frame transmitted by the AP to the STAs which carries the

beacon and the two data subframes. The second frame is transmitted by the STA1 to the AP. Finally, the third frame is transmitted by the STA 2 to the AP. This setup demonstrates the capabilities of w-SHARP to support ultra-low cycle time.

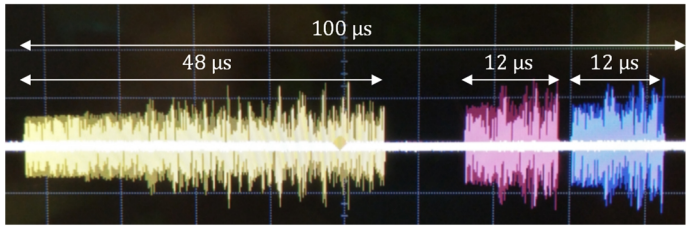


Figure 5.33. w-SHARP Superframe with $T_S = 100$ µs.

The configuration of the RT periods for the second setup is summarized in Table 5.12. With $\text{IFS}_{\text{UL-UL}} = 2$ µs and $\text{IFS}_{\text{UL-DL}} = 12$ µs, $T_{RT} = 200$ µs. T_S has been set to 500 µs to have enough room to create a BE period to transmit 802.11 frames. This setup has been also successfully validated. The data captured by the oscilloscope during 500 µs is depicted in Figure 5.34. In this setup, the network was able to accommodate more nodes, thanks to larger T_S, and some non-RT transmissions during the BE period. In this specific capture, the AP sends a unicast 802.11 frame to the STA 1 during the BE period and the STA 1 answers the 802.11 frame with an ACK. This second experiment demonstrates the flexibility of w-SHARP to support ultra-low cycle times and BE traffic at the same time.

Table 5.12. Superframe Structure with $T_{RT} = 200$ µs, $T_S = 500$ µs.

RT Downlink Subframes			**RT Uplink Frames**		
Subframe type	**Payload length [B]**	**MCS**	**Subframe type**	**Payload length [B]**	**MCS**
Beacon	14	BPSK ½	-	-	-
DL STA 1	10	QPSK ½	UL STA 1	17	QPSK ½
DL STA 2	16	QPSK ½	UL STA 2	23	QPSK ½
DL STA 3	9	BPSK ½	UL STA 3	11	BPSK ½
DL STA 4	35	16-QAM ½	UL STA 4	10	16-QAM ½
DL STA 5	50	64-QAM ¾	UL STA 5	24	64-QAM ¾

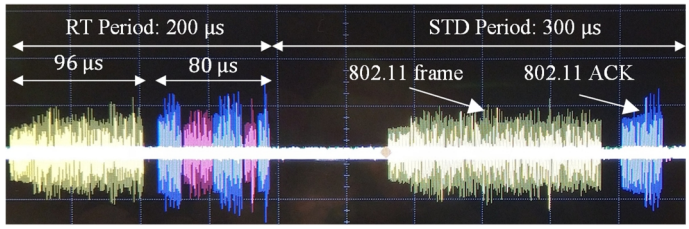


Figure 5.34. w-SHARP Superframe with RT and BE periods, $T_S = 500$ µs.

Additionally, these experiments have also been used to evaluate the modulation and demodulation latencies (T_{Tx} and T_{Rx}), as a function of the MCS, which are reported in Table 5.13. T_{Tx} is below 7 µs for every MCS, whereas the T_{Rx} is below 11 µs.

Table 5.13. w-SHARP PHY Latency.

	BPSK ½	**QPSK ½**	**16-QAM ½**	**64-QAM ¾**
T_{Tx} **[µs]**	3.5	3.7	4.9	6.8
T_{Rx} **[µs]**	9.1	9.8	10.2	10.7

For the first scenario (Table 5.11), and considering that the data of each frame is available at the start of the superframe, T_{E-E} is: 50.2 µs and 57.8 µs for the DL data to STA 1 and DL data to STA 2 frames respectively, and 83.7 µs and 96.8 µs for the UL data from STA 1 and UL data for STA 2 frames

respectively. The reported latencies demonstrate the feasibility of w-SHARP to provide very low latency operation. Still, it must be highlighted that the latency strongly depends on the scheduler design. For instance, the UL frames have a significantly larger latency compared to the DL frames because they are transmitted after the DL frames.

The reported latencies confirm the feasibility of the w-SHARP implementation to provide ultra-low latency operation and ultra-low cycle times, yet being flexible in its configuration, supporting different superframe structures for different application needs.

5.5.2. PER and synchronization results

The PER and synchronization have been evaluated with a network that comprises one AP and one STA. a The AP transmits every 500 µs a DL frame that contains 4 subframes with 20 bytes each and with MCS BPSK ½, QPSK ½, 16-QAM ½, and 64-QAM ¾. The STA transmits four UL frames in a row with also 20 bytes each and the same modulation schemes. The w-SHARP AP P_{Tx} was set to 10 dBm and f_c was set to 2.6 GHz. The w-SHARP STA maximum P_{Tx} was also set to 10 dBm, but it was dynamically adjusted to compensate for the channel attenuation. A total of $5 \cdot 10^7$ packets for each modulation and each direction were transmitted in each experiment.

The experiments were run in a mechanical workshop (see Figure 5.35 and Figure 5.36). Four possible scenarios have been considered, which represents different possibilities in an industrial facility. The STA position was fixed for all the experiments. In scenario 1, shown in Figure 5.36 (a), the AP was put on top of one machine at 5.5 meters from the STA with LoS. In scenario 2, a metal plate was introduced between the AP and STA to block the LoS emulating the case that an operator or a machine moves between the nodes. In scenario 3, depicted in Figure 5.36 (b), the AP was moved from the initial position to 12 meters apart from the STA with large machines blocking the LoS. Finally, in scenario 4, the AP was placed in the opposite corner of the workshop at approximately 23 m and with a wall between the AP and STA.

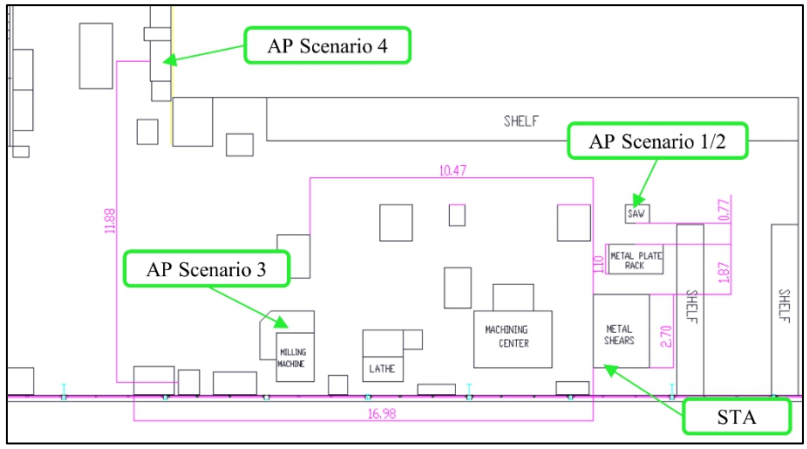
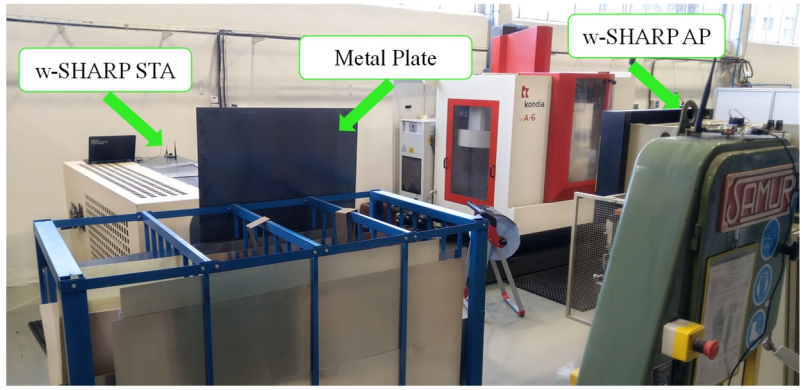
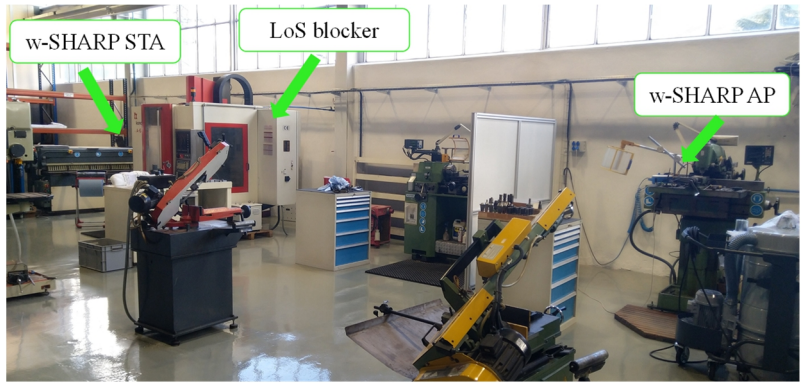


Figure 5.35. Map of the mechanical workshop.



(a)



(b)

Figure 5.36. Photos of the mechanical workshop used to test the PER and jitter. (a) Scenarios 1 and 2 (without and with the metal plate), (b) scenario 3.

Table 5.14. Packet Error Rate, Jitter and mean Rx power results in the Mechanical workshop.

	Scenario 1	**Scenario 2**	**Scenario 3**	**Scenario 4**
Mean Rx power [dBm]	-47	-53	-59	-73
Jitter [ns]	30	35	42	59
PER BPSK ½	$< 10^{-7}$	$< 10^{-7}$	$4 \cdot 10^{-7}$	$2.1 \cdot 10^{-6}$
PER QPSK ½	$< 10^{-7}$	$< 10^{-7}$	$8 \cdot 10^{-7}$	$4.4 \cdot 10^{-6}$
PER 16-QAM ½	$< 10^{-7}$	$< 10^{-7}$	$1.6 \cdot 10^{-5}$	$2.6 \cdot 10^{-4}$
PER 64-QAM ¾	$< 10^{-7}$	$6 \cdot 10^{-7}$	$3.3 \cdot 10^{-4}$	0.67

Table 5.14 summarizes the measured Mean Rx power, jitter, and PER for the four scenarios. The PER results of DL and UL are reported together since there were no significant differences between them. The distance in scenario 1 was relatively small and thus the received power was -47 dBm. In these nearly ideal conditions, there were no erroneous packets for any MCS. In scenario 2, the LoS was blocked reducing the Rx power by 6 dB, to -53 dBm. Note that the workshop ceiling and walls are covered with metal that produces strong multipath components that maintain a high Rx power. 0 errors were found for BPSK ½, QPSK ½, 16-QAM ½. In 64-QAM ½ the measured PER was below $6 \cdot 10^{-7}$, which is still enough for some industrial applications. In scenario 3, the distance between the nodes was 12 m and some machines were blocking the LoS. The PER results for this scenario are still very acceptable. The PER was in the order of 10^{-7} for BPSK and QPSK, which is enough for most industrial applications. However, the PER results are degraded two and three orders of magnitude for 16-QAM and 64-QAM, respectively, caused by wireless channel time-dispersion. Note that the CIR of scenario 2 was measured in Subsection 4.4.2, where it was found that the channel had two taps with a separation of around 150 ns between them. Finally, the Rx power is considerably lower in scenario 4 (-73 dBm), because the wall blocked the LoS and thus multiple reflections were needed to communicate the AP and the STA. In this scenario, the SNR was around 18 dB, which is quite acceptable for low-order modulations, but not for high-order modulations. For instance, the PER with 64-QAM was 0.67, which is impractical even for the transmission of BE frames. The PER with 16-QAM was $2.6 \cdot 10^{-4}$, which is acceptable for non-RT

transmissions, but it is not enough for industrial applications. Finally, the PER with BPSK and QPSK in scenario 4 was in the order of 10^{-6}, which may be acceptable for some industrial applications, but it is not acceptable for most applications.

Finally, the jitter resulted from the three experiments was very similar and smaller than the jitter required by the w-SHARP network and possible industrial applications running over w-SHARP. Specifically, the measured jitter was 30 ns for scenario 1, 35 ns for scenario 2, 40 ns for scenario 3, and 60 ns for scenario 4. These results are also in line with the results reported in Chapter 3 over the channel emulator, where the Wi-Fi modem using the conventional timestamps and PTP obtained a synchronization error in the order of the tens of nanoseconds.

6
Conclusions and Future Research

6.1. Conclusions

This thesis deals with several challenges found in wireless communications when targeting industrial applications at the field level, especially those challenges related to the lower layers of the communication stack. Based on these challenges, three main research topics have been explored, which have been described in Chapters 3, 4, and 5.

Chapter 3 discusses the challenges of time synchronization over realistic wireless conditions with multipath propagation and proposes two synchronization schemes, which are based on two timestamping methods. The conventional timestamping method is based on the timestamps commonly found in wired networks (e.g. Ethernet), and, therefore, it considers a nearly ideal channel, and the enhanced timestamps are designed taken into account the common wireless propagation phenomena found in industrial communications. Both timestamping techniques are designed to be independent of any specific wireless channel and wireless system and, consequently, they can be seamlessly in virtually any wireless standard and scenario. Regarding the synchronization results, the numeric simulations performed over MATLAB® showed that both of them can fulfill the requirements found in most industrial wireless networks and applications. This conclusion is further confirmed by the HW experiments carried out in the laboratory using a custom 802.11 modem implemented over the SoC-FPGA-based SDR platform.

A portable channel sounder that can be seamlessly used in industrial applications with mobility is proposed, implemented, and validated in Chapter 4. The portable channel sounder is based on a clock synchronization scheme combined with the enhanced timestamps proposed in Chapter 3. The use of wireless synchronization provides significant advantages compared to conventional channel sounders that rely on wired synchronization. For instance, the channel sounder operation is more flexible, it is very compact and cost-effective, and its implementation based on SDR can be modified to target other wireless bands or scenarios. Also, the channel sounder is totally self-contained, as the synchronization is delivered by the frame exchange performed by the channel sounder and, therefore, no other external devices, such as Global Positioning System (GPS) receivers, are required. The frame exchange required by the channel sounder has been implemented over our custom 802.11 modem, whereas the synchronization processing has been implemented offline in MATLAB®. The channel sounder has been validated using a channel emulator programmed with an NLoS channel with mobility and over static conditions in a real factory environment. The experiments confirm that the channel sounder is able to correctly measure the CIR of the wireless channel and represents its fundamental statistics, such as the PDP, the fading distributions, and the Doppler spectrum. The proposed design solves significant issues of conventional channel sounders and simplifies the realization of channel measurement campaigns in industrial scenarios, especially in those that include mobile robots.

Finally, Chapter 5 discusses the design, implementation, and validation of a wireless technology named w-SHARP specifically designed to tame the stringent requirements of industrial applications at the field level. w-SHARP is an enhancement to the latest 802.11 standards and provides lower communication latency, deterministic transmissions, and significantly higher efficiency for industrial applications characterized by short packets. The key aspects of the technology are the low overhead of its PHY layer, its persistent TDMA-based MAC layer, and the use of wireless time synchronization to provide a global notion of time to the wireless nodes. The theoretical analysis of w-SHARP shows promising results, reaching sub-millisecond latencies, sub-millisecond cycle times, and very high reliability characterized by a PER in the order of 10^{-7} upon the

condition of enough received power. Given these promising results, a prototype of w-SHARP has been implemented over a SoC-FPGA-based SDR platform and validated under realistic industrial conditions. The experiments confirm the results obtained by simulation means, showing that the prototype reaches the performance derived from the simulations, and that outperforms current state-of-the-art wireless standards, such as 5G and 802.11ax.

6.2. Future Research

The three research directions explored during this thesis may open up new interesting research possibilities within the topic of industrial wireless communications. Most of them are related to the implementation of more powerful versions of the presented designs or the integration with other technologies.

Regarding the research about high-performance wireless synchronization, the enhanced timestamps have been successfully validated through simulation means and in a post-processing algorithm in the channel sounder, but they have not been fully implemented in a hardware platform. As an immediate future line, the enhanced timestamps may be implemented in a HW testbed to fully validate their capabilities to offer sub-nanosecond time synchronization over wireless. In the long term, the achieved synchronization may open interesting research lines in complementary topics. This thesis has discussed the use of wireless synchronization applied to channel sounding, but other applications in industrial applications, such as indoor localization or physical layer security based on time of arrival are still to be explored.

In the context of precise channel characterization, the presented channel sounder has been validated over a wireless channel emulator and has been used to perform a channel measurement campaign in an industrial scenario over static conditions. However, the channel sounder has not been used in a real mobile application. Therefore, an immediate future step could be the use of the channel sounder to perform a channel measurement campaign in an application that involves mobile robots. Additionally, the implementation of the channel sounder

has some limitations due to the performance limits of the ADI RF SOM platform and the 802.11 modem. For instance, the channel sounder has little BW and only one antenna. As a result, it mainly serves as a proof of concept the proposed wireless synchronization scheme can be successfully applied to channel sounding. The implementation of a dedicated HW based on FPGA and optimized for channel sounding is currently being implemented by the Communication Systems team of Ikerlan. The use of a HW platform with larger BW (>200 MHz) and more antennas will provide a comprehensive characterization of virtually any industrial wireless channel in bands below mm-Wave. As the last future research line in this context, the proposed post-processing synchronization algorithm of the channel sounder is based on a PI loop, which well suits an application running in real-time, but it is not optimum for post-processing. Hence, the algorithm could be redesigned to improve the performance of the synchronization, reaching better characterization of the channel, especially in scenarios with nodes moving at high-speed.

Regarding the implementation of SHARP, the presented prototype is still at an early stage. As a future research line in the short term, the integration of w-SHARP and Ethernet TSN in the SoC-FPGA-based SDR platform is currently being developed. In the integrated design, the data frames will be exchanged through a HW interface, allowing an extremely low scheduling latency between the wireless and wired domains. Additionally, the HW will be accompanied by an orchestrator that will synchronize the schedulers of both domains ensuring that the frames are delivered with minimum end-to-end latency. Additionally, the current implementation of the wireless segment is limited to 20 MHz and only OFDM. Therefore, an interesting line of research could be the migration of the w-SHARP HW design to a new platform with a high-end FPGA and radio chip to boost the performance of the prototype. Finally, the w-SHARP design is focused on time-critical industrial applications and, as a result, it has very stable performance, at the cost of limited flexibility to support high data rate in BE applications. Thus, the integration of the strong features of w-SHARP into state-of-the-art wireless standards, such as 5G-NR or IEEE 802.11be could be a highly interesting future line of work to introduce the w-SHARP advantages to a broader range of applications and scenarios.

A

Appendix

This appendix describes the design and implementation of the 802.11 modem IP over the ADI RF SOM platform. It is divided into three subsections that include: A.1. a summary of the 802.11 modem features, A.2. the PHY implementation, A.3. the MAC implementation, and A.4. the interface with the SW.

A.1. Main features of the 802.11 modem IP

The 802.11 IP described in this Appendix is a complete 802.11 transceiver including a fully compliant Wi-Fi PHY (802.11a/g) and its corresponding MAC layer. The 802.11 modem IP is designed to be capable of act as an AP or as an STA. The block diagram of the 802.11 modem IP connected to an ARM microcontroller (e.g. on the Zynq of the ADI RF SOM platform) is shown in Figure A.1. The 802.11 modem IP is divided into three main functional blocks: the MAC, the PHY, and the timing block, which implements their corresponding functionalities. The timing functionalities are not described in this appendix since they are part of Chapter 3.

The 802.11 modem IP is prepared to be controlled by an ARM microcontroller using its two interfaces: the register interface (configuration), and the memory interface (data transfer from/to the MAC). For the sake of illustration, Figure A.1. also includes an example of the interconnection of the 802.11 modem IP with an ARM microcontroller. In this case, the interface is performed through two AXI buses. The first one is connected to the register

interface and the second one is connected to the memory interface. Additionally, a Direct Memory Access (DMA) is also included to reduce the ARM microcontroller load during the copying operations. A DMA is a module that performs copy operations from one memory to another memory.

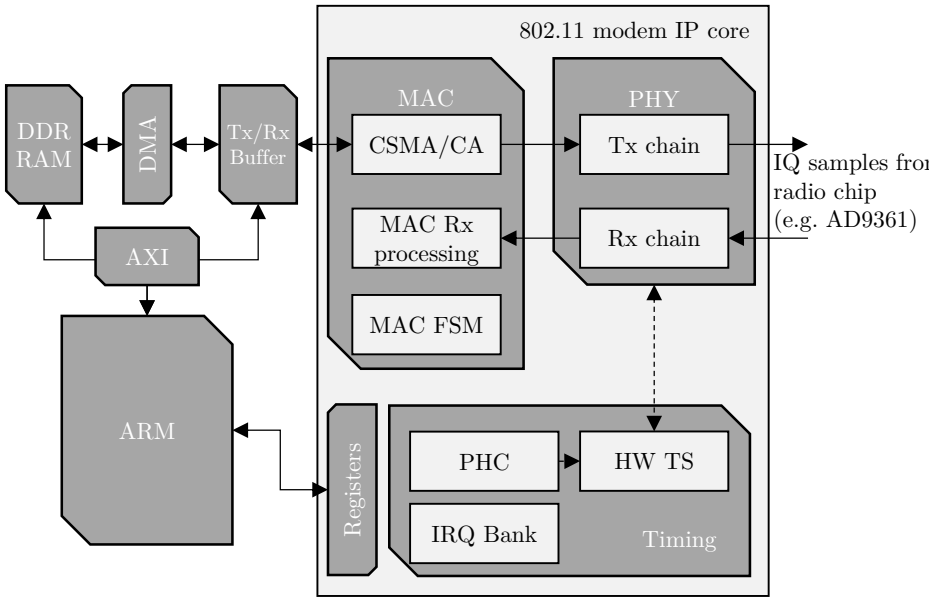


Figure A.1. Block diagram of the 802.11 modem IP core connected to an ARM microcontroller in a Zynq.

The main functionalities of the 802.11 modem IP are summarized as follows:

- PHY:
  - 20 MHz BW (802.11g).
  - All data rates considered in 802.11g from 6 Mbps (BPSK 1/2) to 54 Mbps (64 QAM 3/4) supported.
  - Direct Access to the PHY (skip MAC) if required.
- 802.11 MAC:
  - Automatic ACK transmission.

- Frame fragmentation with a reconfigurable threshold.
- CTS/RTS procedure.
- Proprietary Priority scheduling based on three priorities.
- Up to 64 KB of Tx Queue.
- Up to 64 KB of Rx Queue.

- Timing:
  - High-precision HW timestamping.
  - PHC with sub-nanosecond internal precision.
  - Standard PHC interface support.
  - Programmable interruptions based on the PHC output.
  - PPS signal generation with 6.25 ns of resolution.

A.2. PHY

The PHY layer is divided into two main blocks: the Tx PHY and the Rx PHY. The Tx PHY forms the 802.11 frames based on the data bytes and the configuration information received from the MAC. The Rx PHY listens to the wireless medium, demodulates the received frames, and recovers the data bytes transmitted by other 802.11 nodes.

A.2.1 Tx PHY

The block diagram of the Tx PHY is shown in Figure A.2. It comprises several steps that are divided into 802.11 coder that receives the bytes and generates IQ symbols, and the OFDM modulator, which performs the actual generation of the frame. Regarding the coder, it has four blocks: the scrambler, the convolutional coder, the interleaver, and the QAM modulator, which are described as follows.

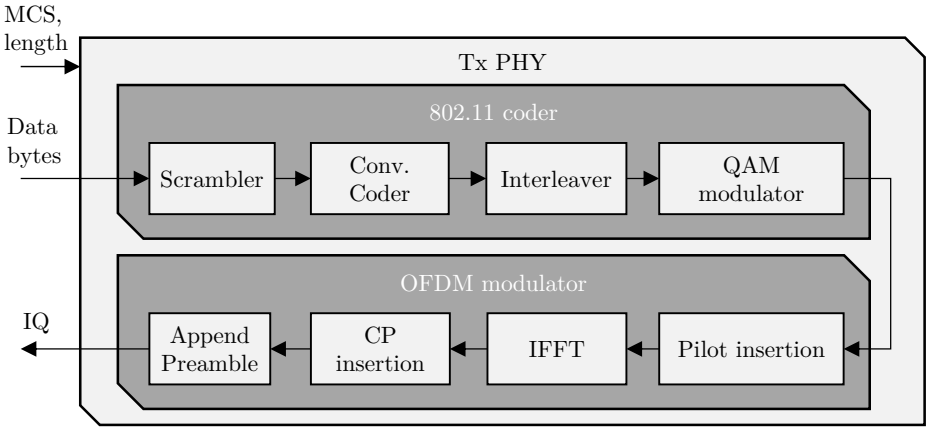


Figure A.2. 802.11 modem IP Tx PHY.

The scrambler randomizes the input data bits with a pseudorandom sequence. It is used to ensure a uniform distribution of bits with zero and one values. The uniform distribution reduces the PAPR of the OFDM signal.

The bits generated by the scrambler are introduced into the convolutional encoder. The convolutional encoder provides error protection under bad wireless channel conditions and enhances the error rate probability. It can be configured with a redundancy rate of 1/2, 2/3, and 3/4 according to the standard. 1/2 refers to 1 bit of redundancy per 1 bit of data, 2/3 is 1 bit of redundancy per 2 bits of data, and 3/4 is 1 bit of redundancy per 3 bits of data. The rate of 2/3 can only be used with the highest modulation scheme (64-QAM).

After the convolutional encoding, the bits are sent to the interleaver. The interleaver task is to separate the data bits thus two or more consecutive bits are placed apart. The interleaver enhances the convolutional encoder bit protection for wireless channels that produce multiple erroneous bits in a row. The interleaver process depends on the number of bits of each OFDM symbol, which is set based on the MCS of the frame.

The last element of the 802.11 coder is the QAM modulator. The QAM modulator transforms the data bits into IQ symbols. The QAM modulator supports the 4 modulations considered in the 802.11g standard: BPSK, QPSK,

16-QAM, and 64-QAM. The IQ symbols of the QAM modulator are grouped in packets of 48 symbols, which is the number of IQ data symbols in each OFDM symbol.

The QAM modulator output is sent to the OFDM modulator, which has four blocks: the pilot insertion, IFFT, CP prefix insertion, and the append preamble block. The pilot insertion introduces 4 extra symbols in each OFDM symbol and outputs a total of 52 IQ symbols per OFDM symbol. The pilots are used to correct the possible rotation of the constellation due to the frequency errors of the RF chain. Afterward, the IQ symbols are sent to the IFFT. The IFFT transforms the data from the frequency domain to the time domain. The IFFT has a size of 64 bins. 52 bins are filled with IQ symbols, 11 extreme bins are filled with zeros and they are used for guard bands, and the dc bin is also set to 0. Finally, the preamble is appended to the frame and it is transmitted to the RF chip.

It must be noted that the Tx PHY performs the coding and modulation of two "subframes". The first "subframe" corresponds with the SIGNAL symbol, which is modulated with MCS 0 (BPSK 1/2) and it has a length of 24 bits. Afterward, the data itself is modulated. Therefore, the frame is the combination of the preamble, the SIGNAL symbol, and the OFDM symbols with data.

A.2.1 Rx PHY

The block diagram of the Rx PHY is shown in Figure A.3. The Rx PHY recovers the data bytes of the frames received in the wireless medium. The 802.11 decoder performs the exact inverse steps of the 802.11 coder, whereas the main differences of the Rx PHY and Tx PHY reside in the OFDM demodulator. Therefore, this subsection only describes the OFDM demodulator.

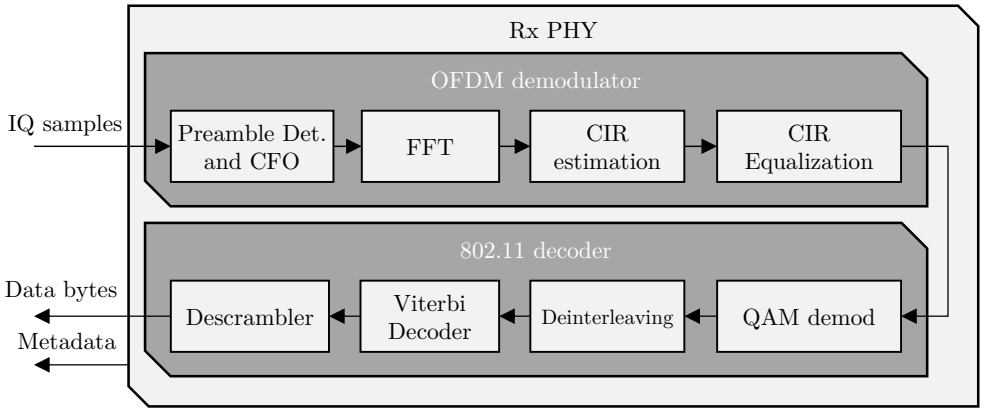


Figure A.3. 802.11 modem IP Rx PHY.

The first step of the OFDM demodulator is the preamble detector. The preamble detector is permanently listening to the wireless medium to find possible incoming frames. The preamble detector performs three steps to know if a frame is being received. The first step is to see if the wireless medium is busy by estimated the receiving power. Afterward, it performs a cross-correlation of the input signal to detect the STF of the 802.11 preamble. Finally, it performs the cross-correlation of the input signal to precisely detect the LTF of the 802.11 preamble and delimit the start of the frame. In this process, the preamble detector also estimates the frequency offset between the transmitting node and the receiving node.

After the detection of the frame, the OFDM demodulator performs the FFT of the incoming OFDM symbols to retrieve the data symbols. The frame is affected by the wireless channel impairments, which are corrected by the CIR estimation and the CIR equalization. The CIR estimation is performed using the last LTF symbol of the preamble. In particular, the CIR estimation compares the expected LTF symbol (transmitted one) with the received one to estimate the wireless channel. The channel estimation is applied to the data symbols in the CIR equalization to compensate for the phase and amplitude impairments introduced by the wireless channel. Additionally, these blocks also use the pilots of the OFDM symbols to evaluate the marginal frequency offset and compensate

for the constellation rotation. The equalized IQ symbols are sent to the 802.11 decoder in groups of 48 symbols, which recovers the transmitted data bits.

It must be noted that the demodulation process is done in two steps. In the first step, the OFDM demodulator and 802.11 decoder performs the decoding of the SIGNAL symbol, which has a known structure. After the decoding of the SIGNAL, the rest of the frame is demodulated and decoded based on the information contained in the SIGNAL.

A.3. 802.11 MAC

The 802.11 MAC implements the CSMA/CA procedure of the 802.11 modem. It comprises an FSM and some basic processing HW. On the one hand, the FTM determines the current state of the modem (Tx ongoing, Rx ongoing, idle) and the specific step of the Tx/Rx chain that is being performed. On the other hand, the processing HW is used as an interface between the FSM and the SW/PHY. The processing HW for the Rx side comprises the CRC check to know if the frame contains errors or not and the 802.11 header parsing to evaluate the destination of the 802.11 frame and if it requires an acknowledgment. The processing HW for the Tx side comprises the whole CSMA/CA procedure that is used to trigger the frame transmission. It includes the NAV, back-off, the RTx management module, and a signal from the PHY that indicates if the channel is busy or not.

The FSM uses the processing HW to determine the state of the 802.11 MAC and operate over the SW and PHY interfaces. The FSM is depicted in Figure A.4 and described as follows. The FSM starts in the idle state, when the 802.11 modem does not have any pending transmission nor pending reception. When the SW writes a frame and descriptor to the modem, the FSM goes to the Tx state. In the Tx state, the first step is to extract the frame from the Queue and store it in its local memory. Afterward, the 802.11 MAC waits for the trigger generated by the CSMA/CA module. In the case that a frame is detected by the PHY, the FSM goes automatically to the Rx state. When the CSMA/CA finishes the FSM orders the transmission of the frame. After the transmission, the 802.11 waits for an ACK if needed (in some cases, e.g. broadcast frames, the ack is not

required). If the ACK is not received, the FSM retries the transmission until a configurable No. of retries. In the case that an ACK is received, the FSM checks if the frame was fragmented. In this case, the FSM remains in the Tx state and extracts the next frame from the Queue.

The Rx states start with a signal from the PHY indicating that a frame is being received. The first state indicates that a preamble has been detected and the 802.11 PHY is decoding the SIGNAL. If the SIGNAL is incorrect, it goes to the idle state. If the signal is correct, it goes to the frame decoding state and waits for the frame data. In this state, the 802.11 MAC parses the 802.11 header and checks the CRC. If the CRC check result is incorrect or if the frame is for another node, it discards the frame. If the CRC is correct and if the frame is for the local node, the local node replies with an ACK and sends the frame to the upper layers. Finally, if the CRC is correct and the frame is a broadcast frame, it does not transmit an ACK and sends the frame to upper layers.

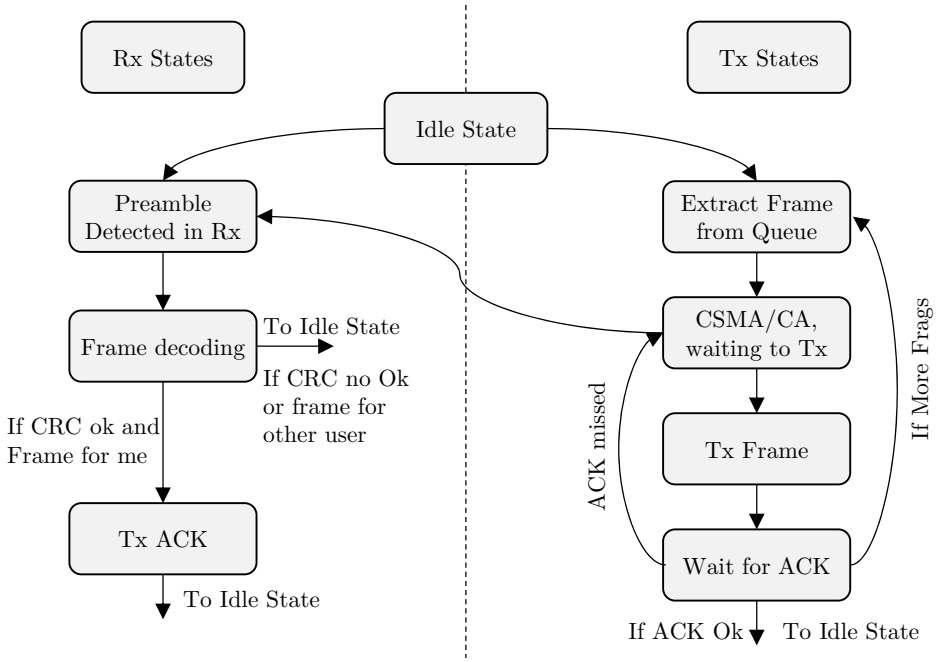


Figure A.4. 802.11 modem IP MAC FSM.

A.4. 802.11 modem IP SW interface

The 802.11 modem IP control interface comprises two sub-interfaces: the register interface and the memory interface. On the one hand, the register interface is used to configure the IP and to perform the control operations, such as triggering the transmission of a frame. On the other hand, the memory interface is used to transmit the metadata/data of the packets that must be transmitted to the wireless link, and to receive the metadata/data of the packets that are being received through the wireless link.

A.4.1 Register interface

The register interface is used to configure the 802.11 modem and to perform the control operations. The next table summarizes the most relevant registers present in the 802.11 modem IP.

Table A.1. Registers of the 802.11 modem IP.

Register	**In/Out**	**Functionality**
reset	In	20
Tx Request	In	When this flag is activated, the user indicates to the 802.11 IP that a frame is currently in the queue waiting for transmission.
MAC Address	In	MAC Address of the 802.11 interface
IRQ Mask	In	Indicates if the Tx/Rx IRQ signals are activated or deactivated.
IRQ flags	In/Out	Indicates which IRQ is currently ongoing. It is also used to free the IRQ line.
No. of RTx	In	Indicates the maximum No. of RTx according to the 802.11 CSMA/CA procedure.
Skip 802.11 MAC	In	If this flag is activated, the 802.11 modem will skip the listening before talk (CSMA/CA) process.
Rx ADDR	Out	Contains the memory position used to store the last 802.11 frame received.

Tx ADDR	In	It is registered when the Tx Request flag is set to '1'. Indicates the position of the 802.11 frame in the Queue.
Timing Regs	-	-
PHC Time	Out	Displays the actual time of the PHC in nanoseconds.
Tx timestamp	In	Returns the egress time of the transmitted frame.
Clock drift	In	Adjusts the clock drift of the PHC. Used for synchronization purposes.
Clock offset	In	Adjusts the clock offset of the PHC. Used for synchronization purposes.
PHC IRQ time	In	Sets the PHC IRQ signal time.

A.4.2 Memory interface

The memory interface has two data structures used to share data with the SW. The first structure is used by the SW to indicate the frame structure and data of a frame that must be transmitted and the second one is generated by the 802.11 modem IP when an 802.11 frame has been received.

Tx structure

When the user wants to transmit a packet through the 802.11 interface, it must write the next structure into the memory interface of the 802.11 modem IP.

Tx Descriptor						Data
Data length	MCS	duration	Req ack	More frags	Next Frag ADDR	Tx Data bytes

Figure A.5. Data structure used to transmit 802.11 frames.

The Tx structure comprises 6 fields of fixed size (Tx Descriptor) and the data field, which has a variable size. Related to the transmission, the structure includes the length of the Tx Data byte fields, the MCS used to modulate the frame according to the MCS supported by 802.11g, the duration of the 802.11

transmission opportunity as defined in 802.11. In addition, the structure also contains metadata information about the 802.11 fragments and the Req ACK flag that indicates if the FSM must wait for the reception of an ACK or not. In the case that the Req ACK flag is activated, the FSM also performs retransmissions in case that the ACK is not received.

Rx structure

Every time that a frame is received by the 802.11 modem, it generates a structure that is sent to the SW through the memory interface. The structure is depicted in Figure A.6.

Rx Descriptor						Data
Data length	MCS	CRC status	Rx Timestamp	RSSI	CIR estimation	Rx Data bytes

Figure A.6. Data structure generated by the 802.11 after the reception of an 802.11 frame.

The structure comprises some basic 802.11 information, such as the No. of Data bytes received, the MCS, the RSSI, and the CRC status, which indicates if the frame was correctly received or not. In addition, the Rx structure also includes a high-precision Rx timestamp taken with the PHC, and the estimation of the CIR in the frequency domain, which is performed by the CIR estimation and equalizer block of the PHY. Finally, the structure has a Data field which includes the actual data demodulated from the 802.11 frame.

References

[1] T. Sauter, "The Continuing Evolution of Integration in Manufacturing Automation," *IEEE Ind. Electron. Mag.*, vol. 5, no. 1, pp. 68–68, 2009, doi: 10.1109/tii.2009.2017083.

[2] M. Wollschlaeger, T. Sauter, and J. Jasperneite, "The future of industrial Communication," *IEEE Ind. Electron. Mag.*, vol. 12, no. 4, pp. 370–376, 2017, doi: 10.1021/ie50124a022.

[3] "Time-Sensitive Networking Task Group." http://www.ieee802.org/1/pages/tsn.html.

[4] D. Cavalcanti, J. Perez-Ramirez, M. M. Rashid, J. Fang, M. Galeev, and K. B. Stanton, "Extending Accurate Time Distribution and Timeliness Capabilities over the Air to Enable Future Wireless Industrial Automation Systems," *Proc. IEEE*, vol. 107, no. 6, pp. 1132–1152, 2019, doi: 10.1109/JPROC.2019.2903414.

[5] A. Arriola, I. Val, J. M. García-Loygorri, and C. Briso, "Narrowband characterization of a train-to-train wireless link at 2.6 GHz in metro environments," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 10796 LNCS, pp. 100–109, 2018, doi: 10.1007/978-3-319-90371-2_11.

[6] I. Val *et al.*, "Wireless channel measurements and modeling for TCMS communications in metro environments," *2017 11th Eur. Conf. Antennas Propagation, EUCAP 2017*, pp. 108–112, 2017, doi: 10.23919/EuCAP.2017.7928370.

[7] C. Cruces, R. Torrego, A. Arriola, and I. Val, "Deterministic Hybrid Architecture with Time Sensitive Network and Wireless Capabilities," *IEEE Int. Conf. Emerg. Technol. Fact. Autom. ETFA*, vol. 2018-Septe, pp. 1119–1122, 2018, doi: 10.1109/ETFA.2018.8502524.

[8] I. Val, A. Arriola, C. Cruces, R. Torrego, and E. Gomez, "Structural Health Monitoring Applications in Railway Environments," *IEEE World Conf. Fact. Commun. Syst.*, pp. 1–9, 2015.

[9] Z. Fernández Ganzabal, "Analysis of the Impact of Wireless Mobile Devices in Critical Industrial Applications," 2019.

[10] P. M. Rodriguez, "Spectrum Handoff Strategy for Cognitive Radio-based MAC for Time-Critical and Mission-Critical Industrial Wireless Sensor and Actuator Networks," 2016.

[11] R. Torrego, "Design Methodology Addressing Static/Reconfigurable Partitioning for Optimizing Software Defined Radio (SDR) Implementation Through FPGA Dynamic Partial Reconfiguration and Rapid Prototyping Tools," Ph.D. Thesis, Mondragón Unibersitatea, 2013.

[12] R. Torrego, A. Etxabe, P. M. Rodriguez, and I. Val, "Deterministic and cognitive wireless communication system with jamming-resistant capabilities for tactical or industrial communications," *Commun. Technol. Softw. Defin. Radio WinnComm Eur.*, 2017.

[13] S. Bennet, *A History of Control Engineering 1930-1955*. Institution of Engineering and Technology (IET), 1993.

[14] P. Martí, R. Villá, and J. M. Fuertes, *Networked Control Systems Overview*. CRC press, 2005.

[15] *IEEE Standard for a precision clock synchronization protocol for networked measurement and control systems*. IEEE Standard 1588, 2009.

[16] "EtherCAT Technology Group." https://www.ethercat.org/default.htm (accessed Apr. 24, 2020).

[17] T. Zheng, M. Gidlund, and J. Åkerberg, "WirArb: A New MAC Protocol for Time Critical Industrial Wireless Sensor Network Applications," *IEEE Sens. J.*, vol. 16, no. 7, pp. 2127–2139, 2016, doi: 10.1109/JSEN.2015.2504948.

[18] M. Luvisotto, Z. Pang, and D. Dzung, "High-Performance Wireless Networks for Industrial Control Applications: New Targets and Feasibility," *Proc. IEEE*, vol. 107, no. 6, pp. 1074–1093, 2019, doi: 10.1109/JPROC.2019.2898993.

[19] Ó. Seijo, Z. Fernández, I. Val, and J. A. López-Fernández, "SHARP : A Novel Hybrid Architecture for Industrial Wireless Sensor and Actuator Networks," 2018.

[20] M. Zhan, Z. Pang, D. Dzung, M. Luvisotto, K. Yu, and M. Xiao, "Towards High-performance Wireless Control: 10^-7 Packet Error Rate in Real Factory Environments," *IEEE Trans. Ind. Informatics*, vol. PP, no. X, pp. 1–1, 2019, doi: 10.1109/tii.2019.2919653.

[21] K. Langlois *et al.*, "EtherCAT tutorial: An introduction for real-time hardware communication on windows [tutorial]," *IEEE Robot. Autom. Mag.*, vol. 25, no. 1, pp. 22–26, 2018, doi: 10.1109/MRA.2017.2787224.

[22] P. Ferrari *et al.*, "Clock synchronization of PTP-based devices through PROFINET IO networks," *IEEE Int. Conf. Emerg. Technol. Fact. Autom. ETFA*, pp. 496–499, 2008, doi: 10.1109/ETFA.2008.4638445.

[23] "SERCOS website." https://www.sercos.org/ (accessed Jun. 01, 2020).

[24] *Industrial wireless networks WISA specification, part i: WIA system architecture and communication specification for process automation (WIA-PA).* 2008.

[25] Fieldcommgroup, "WirelessHART main webpage." https://www.fieldcommgroup.org/technologies/hart (accessed Jun. 01, 2020).

[26] IEEE, "802.15.1- IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 15.1a: Wireless Medium Access Control (MAC) and Physical Layer (PHY) specifications for Wireless Personal Area Networks (WPAN)," 2005.

[27] IEEE, *802.15.4: IEEE Standard for Low-Rate Wireless Networks.* 2011.

[28] W. Liang *et al.*, "WIA-FA and Its Applications to Digital Factory: A Wireless Network Solution for Factory Automation," *Proc. IEEE*, vol. 107, no. 6, pp. 1053–1073, 2019, doi: 10.1109/JPROC.2019.2897627.

[29] *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, 2012.

[30] F. Tramarin, S. Vitturi, M. Luvisotto, and A. Zanella, "On the Use of IEEE 802.11n for Industrial Communications," *IEEE Trans. Ind. Informatics*, vol. 12, no. 5, pp. 1877–1886, 2016, doi: 10.1109/TII.2015.2504872.

[31] 3GPP, "Third Generation Partnership Project main webpage." https://www.3gpp.org/ (accessed Jun. 01, 2020).

[32] J. Sachs, G. Wikstrom, T. Dudda, R. Baldemair, and K. Kittichokechai, "5G Radio Network Design for Ultra-Reliable Low-Latency Communication," *IEEE Netw.*, vol. 32, no. 2, pp. 24–31, 2018, doi: 10.1109/MNET.2018.1700232.

[33] X. Jiang, M. Luvisotto, Z. Pang, and C. Fischione, "Latency Performance of 5G New Radio for Critical Industrial Control Systems," *IEEE Int. Conf. Emerg. Technol. Fact. Autom. ETFA*, vol. 2019-Septe, pp. 1135–1142, 2019, doi: 10.1109/ETFA.2019.8869241.

[34] S. Vitturi, L. Peretti, L. Seno, M. Zigliotto, and C. Zunino, "Real-time Ethernet networks for motion control," *Comput. Stand. Interfaces*, vol. 33, no. 5, pp. 465–476, 2011, doi: 10.1016/j.csi.2011.01.005.

[35] *802.11ax. Draft 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) Specif.* 2019.

[36] D. Deng *et al.*, "IEEE 802.11ax : Highly Efficient WLANs for Intelligent Information Infrastructure," *IEEE Commun. Mag.*, vol. 55, no. 12, pp. 52–59, 2017, doi: 10.1109/MCOM.2017.1700285.

[37] D. Lopez-Perez, A. Garcia-Rodriguez, L. Galati-Giordano, M. Kasslin, and K. Doppler, "IEEE 802.11be Extremely High Throughput: The Next Generation of Wi-Fi Technology Beyond 802.11ax," *IEEE Commun. Mag.*, vol. 57, no. 9, pp. 113–119, 2019, doi: 10.1109/mcom.001.1900338.

[38] M. Luvisotto, Z. Pang, and D. Dzung, "Ultra High Performance Wireless Control for Critical Applications: Challenges and Directions," *IEEE Trans. Ind. Informatics*, vol. 13, no. 3, pp. 1448–1459, 2017, doi: 10.1109/TII.2016.2617459.

[39] M. Weiss, "Getting accurate time from GNSS receivers: Considerations to approach nanosecond time," *IEEE Int. Symp. Precis. Clock Synchronization Meas. Control. Commun. ISPCS*, 2017, doi: 10.1109/ISPCS.2017.8056746.

[40] "Network Time Protocol Main website." http://ntp.org/ (accessed Feb. 05, 2020).

[41] A. Mahmood, R. Exel, H. Trsek, and T. Sauter, "Clock synchronization over IEEE 802.11 - A survey of methodologies and protocols," *IEEE Trans. Ind. Informatics*, vol. 13, no. 2, pp. 907–922, 2017, doi: 10.1109/TII.2016.2629669.

[42] P. Ferrari, A. Flammini, S. Rinaldi, A. Bondavalli, and F. Brancati, "Evaluation of timestamping uncertainty in a software-based IEEE 1588 implementation," in *Conference Record - IEEE Instrumentation and Measurement Technology Conference*, 2011, pp. 604–609, doi: 10.1109/IMTC.2011.5944319.

[43] P. Loschmidt, R. Exel, A. Nagy, and G. Gaderer, "Limits of synchronization accuracy using hardware support in IEEE 1588," *2008 IEEE Int. Symp. Precis. Clock Synchronization Meas. Control Commun. ISPCS 2008, Proc.*, pp. 12–16, 2008, doi: 10.1109/ISPCS.2008.4659205.

[44] A. Mahmood, R. Exel, and T. Sauter, "Delay and jitter characterization for software-based clock synchronization over WLAN using PTP," *IEEE Trans. Ind. Informatics*, vol. 10, no. 2, pp. 1198–1206, 2014, doi: 10.1109/TII.2014.2304413.

[45] R. Exel, "Clock Synchronization in IEEE 802.11 Wireless LANs using Physical Layer Timestamps," *IEEE Int. Symp. Precis. Clock Synchronization Meas. Control Commun. Proc.*, 2012.

[46] F. M. Anwar and M. B. Srivastava, "Precision time protocol over LR-WPAN and 6LoWPAN," *IEEE Int. Symp. Precis. Clock Synchronization Meas. Control. Commun. ISPCS*, 2017, doi: 10.1109/ISPCS.2017.8056739.

[47] *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 Computer Society, 2016.

[48] J. Moreno Garcia-Loygorri, I. Val, A. Arriola, and C. Briso, "Channel Model and Interference Evaluation for a Wireless Train Backbone," *IEEE Access*, vol. 7, pp. 115518–115527, 2019, doi: 10.1109/access.2019.2934759.

[49] Y. Xuefeng and C. Xiang, *Propagation Channel Characterization, Parameter Estimation, and Modeling for Wireless Communications*, 1st ed. Wiley, 2016.

[50] F. Pérez Fontán and P. Mariño Espiñeira, *Modeling the Wireless Propagation Channel: A Simulation Approach with MATLAB®*. Wiley, 2008.

[51] A. Vinko Erceg, Laurent Schumacher, Persefoni Kyritsi, Molisch, D. S. Baum, A. Y. Gorokhov, and C. Oestges, "TGn Channel Models," 2004.

[52] A. A. Khuwaja, Y. Chen, N. Zhao, M. S. Alouini, and P. Dobbins, "A survey of channel modeling for uav communications," *IEEE Commun. Surv. Tutorials*, vol. 20, no. 4, pp. 2804–2821, 2018, doi: 10.1109/COMST.2018.2856587.

[53] J. T. Quimby *et al.*, "NIST channel sounder overview and channel measurements in manufacturing facilities," 2017, doi: 10.6028/NIST.TN.1979.

[54] R. Croonenbroeck, L. Underberg, A. Wulf, and R. Kays, "Measurements for the Development of an Enhanced Model for Wireless Channels in Industrial Environments," *IEEE Int. Conf. Wirel. Mob. Comput. Netw. Commun.*, 2017.

[55] Ó. Seijo, I. Val, and J. A. López-Fernández, "Portable Full Channel Sounder for Mobile Robotics by Using Sub-Nanosecond Time Synchronization over Wireless," *IEEE Int. Work. Fact. Commun. Syst. - Proceedings, WFCS*, 2020.

[56] I. Val, F. Casado, P. M. Rodriguez, and A. Arriola, "FPGA-based wideband channel emulator for evaluation of Wireless Sensor Networks in industrial environments," *19th IEEE Int. Conf. Emerg. Technol. Fact. Autom. ETFA 2014*, 2014, doi: 10.1109/ETFA.2014.7005361.

[57] Ó. Seijo, C. Cruces, R. Torrego, I. Val, and J. A. López-fernández, "IEEE 1588 Hardware-Based Timestamp Implementation over 802 . 11a/g for IWSAN with High Precision Synchronization Requirements," *IEEE Int. Symp. Precis. Clock Synchronization Meas. Control. Commun. ISPCS*, 2017.

[58] Ó. Seijo, J. A. López-fernández, H.-P. Bernhard, and I. Val, "Enhanced Timestamping Method for Sub-Nanosecond Time Synchronization in IEEE 802.11 over WLAN Standard Conditions," *IEEE Trans. Ind. Informatics*, 2020.

[59] Ó. Seijo, I. Val, and J. A. Lopez-Fernandez, "Portable Full Channel Sounder for Industrial Wireless Applications With Mobility by Using Sub-Nanosecond Wireless Time Synchronization," *IEEE Access*, vol. 8, pp. 175576–175588, 2020, doi: 10.1109/access.2020.3025896.

[60] Ó. Seijo, Z. Fernandez, I. Val, and J. A. Lopez-Fernandez, "SHARP: Towards the Integration of Time-Sensitive Communications in Legacy LAN/WLAN," *2018 IEEE Globecom Work. GC Wkshps 2018 - Proc.*, 2019, doi: 10.1109/GLOCOMW.2018.8644124.

[61] Ó. Seijo, J. A. López-fernández, I. Val, and S. Member, "w-SHARP : Implementation of a High-Performance Wireless Time-Sensitive Network for Low Latency and Ultra-Low Cycle Time Industrial Applications," *IEEE Trans. Ind. Informatics*, 2020.

[62] H. Hellstrom, M. Luvisotto, R. Jansson, and Z. Pang, "Software-Defined Wireless Communication for Industrial Control: A Realistic Approach," *IEEE Ind. Electron. Mag.*, vol. 13, no. 4, pp. 31–37, 2019, doi: 10.1109/MIE.2019.2942454.

[63] X. Wei *et al.*, "Software Defined Radio Implementation of a Non-Orthogonal Multiple Access System Towards 5G," *IEEE Access*, vol. 4, pp. 9604–9613, 2016, doi: 10.1109/ACCESS.2016.2634038.

[64] Ó. Seijo, I. Val, J. A. López-fernández, and M. Vélez, "IEEE 1588 Clock Synchronization Performance over Time-Varying Wireless Channels," *IEEE Int. Symp. Precis. Clock Synchronization Meas. Control. Commun. ISPCS*, 2018.

[65] H. Wu *et al.*, "The tick programmable low-latency SDR system," *Proc. Annu. Int. Conf. Mob. Comput. Networking, MOBICOM*, pp. 101–113, 2017, doi: 10.1145/3117811.3117834.

[66] "WARP." https://mangocomm.com/products/kits/warp-v3-kit/ (accessed Jun. 04, 2020).

[67] "Online documentation of the ADI RF SOM platform." https://wiki.analog.com/resources/eval/user-guides/adrv936x_rfsom (accessed May 23, 2020).

[68] W. F. Brinkman, D. E. Haggan, and W. W. Troutman, "A history of the invention of the transistor and where it will lead us," *IEEE J. Solid-State Circuits*, vol. 32, no. 12, pp. 1858–1864, 1997, doi: 10.1109/4.643644.

[69] W. Meeus, K. Van Beeck, T. Goedemé, J. Meel, and D. Stroobandt, "An overview of today's high-level synthesis tools," *Des. Autom. Embed. Syst.*, vol. 16, no. 3, pp. 31–51, 2012, doi: 10.1007/s10617-012-9096-8.

[70] "Xilinx main website." https://www.xilinx.com/ (accessed Apr. 13, 2020).

[71] Xilinx, "Vivado Design Suite User Guide: Model-Based DSP Design Using System Generator," *UG897 (v2018.1)*, 2018.

[72] H. Trsek, *Isochronous Wireless Network for Real-time Communication in Industrial Automation*. 2016.

[73] A. V. Oppenheim and A. S. Willsky, *Signals and Systems*, 2nd ed. Prentice Hall, 1997.

[74] T. Sauter, "The Three Generations of Field-Level Networks—Evolution and Compatibility Issues," *IEEE Trans. Ind. Electron.*, vol. 57, pp. 3585–3595, 2010, doi: 10.1109/TIE.2010.2062473.

[75] K. Montgomery, R. Candell, Y. Liu, and M. Hany, *Wireless User Requirements for the Factory Workcell*. 2019.

[76] V. Díez, A. Arriola, I. Val, and M. Velez, "Reliability evaluation of point-to-point links based on IEEE 802.15.4 physical layer for IWSAN applications," *AEU - Int. J. Electron. Commun.*, vol. 113, 2020, doi: 10.1016/j.aeue.2019.152967.

[77] "ISO Accuracy (trueness and precision) of measurement methods and results – Part 1: General principles and definitions," *ISO Stand. 5725-11994*, 1994.

[78] Z. Pang, M. Luvisotto, and D. Dzung, "Wireless High-Performance Communications: The Challenges and Opportunities of a New Target," *IEEE Ind. Electron. Mag.*, vol. 11, no. 3, pp. 20–25, 2017, doi: 10.1109/MIE.2017.2703603.

[79] D. M. Anand, D. Sharma, Y. Li-Baboud, and J. Moyne, "EDA performance and clock synchronization over a wireless network: Analysis, experimentation and application to semiconductor manufacturing," *IEEE Int. Symp. Precis. Clock Synchronization Meas. Control Commun. ISPCS '09 - Proc.*, pp. 81–86, 2009, doi: 10.1109/ISPCS.2009.5340200.

[80] T. Lennvall, J. Åkerberg, E. Hansen, and K. Yu, "A new wireless sensor network TDMA timing synchronization protocol," *2016 IEEE 14th Int. Conf. Ind. Informatics*, pp. 606–611, 2016, doi: 10.1109/INDIN.2016.7819233.

[81] R. Exel, "Receiver Design for Time-Based Ranging with IEEE 802.11b Signals," *Int. J. Navig. Obs.*, vol. 2012, 2012, doi: 10.1155/2012/743625.

[82] R. Exel and T. Bigler, "ToA Ranging using Subsample Peak Estimation and Equalizer-based Multipath Reduction," *IEEE Wirel. Commun. Netw. Conf.*, 2014.

[83] C. M. De Dominicis, P. Pivato, P. Ferrari, D. MacIi, E. Sisinni, and A. Flammini, "Timestamping of IEEE 802.15.4a CSS signals for wireless ranging and time synchronization," *IEEE Trans. Instrum. Meas.*, vol. 62, no. 8, pp. 2286–2296, 2013, doi: 10.1109/TIM.2013.2255988.

[84] P. Ferrari *et al.*, "Timestamping and Ranging Performance for IEEE 802.15.4 CSS Systems," *IEEE Trans. Instrum. Meas.*, vol. 63, no. 5, pp. 1244–1252, 2014.

[85] P. Ferrari, G. Giorgi, C. Narduzzi, S. Rinaldi, and M. Rizzi, "Timestamp Validation Strategy for Wireless Sensor Networks Based on IEEE 802.15.4 CSS," *IEEE Trans. Instrum. Meas.*, vol. 63, no. 11, pp. 2512–2521, 2014.

[86] U. Mengali, *Synchronization Techniques for Digital Receivers*. Springer, 1997.

[87] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*, 2nd ed. Prentice Hall.

[88] Ó. Seijo, I. Val, and J. A. López, "On the use of White Rabbit for Precise Time Transfer in 5G URLLC Networks for Factory Automation Applications," *IEEE Int. Conf. Ind. Cyber-Physical Syst.*, 2019.

[89] H. Zhao and J. Yu, "A Simple and Efficient Design of Variable Fractional Delay FIR Filters," *IEEE Trans. Circuits Syst. II Express Briefs*, vol. 53, no. 2, pp. 157–160, 2006, doi: 10.1109/TCSII.2005.856673.

[90] "AD9361 Reference Manual." http://www.farnell.com/datasheets/2007082.pdf (accessed Apr. 19, 2020).

[91] "Propsim Channel Emulation Solutions." https://www.keysight.com/en/pc-2697334/propsim-channel-emulation-solutions (accessed Jan. 08, 2020).

[92] Tektronix, "DPO2000 and MSO2000 Series Oscilloscopes User Manual," 2004.

[93] Y. Mostofi, M. Malmirchegini, and A. Ghaffarkhah, "Estimation of communication signal strength in robotic networks," *Proc. - IEEE Int. Conf. Robot. Autom.*, pp. 1946–1951, 2010, doi: 10.1109/ROBOT.2010.5509677.

[94] D. Maas, M. H. Firooz, J. Zhang, N. Patwari, and S. K. Kasera, "Channel sounding for the masses: Low complexity GNU 802.11b channel impulse response estimation," *IEEE Trans. Wirel. Commun.*, vol. 11, no. 1, pp. 1–8, 2012, doi: 10.1109/TWC.2011.111611.091774.

[95] G. R. MacCartney and T. S. Rappaport, "A flexible millimeter-wave channel sounder with absolute timing," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 6, pp. 1402–1418, 2017, doi: 10.1109/JSAC.2017.2687838.

[96] E. Yanmaz, R. Kuschnig, and C. Bettstetter, "Channel measurements over 802.11a-based UAV-to-ground links," *2011 IEEE GLOBECOM Work. GC Wkshps 2011*, pp. 1280–1284, 2011, doi: 10.1109/GLOCOMW.2011.6162389.

[97] D. Hague, H. T. Kung, and B. Suter, "Field Experimentation of COTS-Based UAV Networking," *IEEE Mil. Commun. Conf.*, 2006.

[98] P. Pajusco, N. Malhouroux-Gaffet, and G. El Zein, "Comprehensive characterization of the double directional UWB residential indoor channel," *IEEE Trans. Antennas Propag.*, vol. 63, no. 3, pp. 1129–1139, 2015, doi: 10.1109/TAP.2014.2387418.

[99] B. T. Maharaj, J. W. Wallace, M. A. Jensen, and L. P. Linde, "A low-cost open-hardware wideband multiple-input-multiple-output (MIMO) wireless channel sounder," *IEEE Trans. Instrum. Meas.*, vol. 57, no. 10, pp. 2283–2289, 2008, doi: 10.1109/TIM.2008.919943.

[100] T. B. Welch, W. C. H. G., and M. G. Morrow, *Real-Time Digital Signal Processing from MATLAB to C with the TMS320C6x DSPs*, 3rd ed. CRC Press, 2007.

[101] E. Kassem, R. Marsalek, and J. Blumenstein, "Frequency Domain Zadoff-Chu Sounding Technique for USRPs," *2018 25th Int. Conf. Telecommun. ICT 2018*, pp. 302–306, 2018, doi: 10.1109/ICT.2018.8464940.

[102] J. Vychodil, A. Chandra, T. Mikulasek, A. Prokes, and V. Derbek, "UWB time domain channel sounder," *Proc. 25th Int. Conf. Radioelektronika, RADIOELEKTRONIKA 2015*, pp. 268–271, 2015, doi: 10.1109/RADIOELEK.2015.7129028.

[103] F. Talebi and T. Pratt, "Channel sounding and parameter estimation for a wideband correlation-based MIMO model," *IEEE Trans. Veh. Technol.*, vol. 65, no. 2, pp. 499–508, 2016, doi: 10.1109/TVT.2015.2404571.

[104] R. J. Weiler, M. Peter, T. Kühne, M. Wisotzki, and W. Keusgen, "Simultaneous millimeter-wave multi-band channel sounding in an urban access scenario," *2015 9th Eur. Conf. Antennas Propagation, EuCAP 2015*, 2015.

[105] G. Durisi, T. Koch, and P. Popovski, "Toward Massive, Ultrareliable, and Low-Latency Wireless Communication with Short Packets," *Proc. IEEE*, vol. 104, no. 9, pp. 1711–1726, 2016, doi: 10.1109/JPROC.2016.2537298.

[106] J. Farkas, B. Varga, G. Miklós, and J. Sachs, "5G-TSN Integration For Industrial Automation," 2019.

[107] L. Mathe, P. D. Burlacu, and R. Teodorescu, "Control of a Modular Multilevel Converter with Reduced Internal Data Exchange," *IEEE Trans. Ind. Informatics*, vol. 13, no. 1, pp. 248–257, 2017, doi: 10.1109/TII.2016.2598494.

[108] F. Tramarin, A. K. Mok, and S. Han, "Real-Time and Reliable Industrial Control over Wireless LANs: Algorithms, Protocols, and Future Directions," *Proc. IEEE*, vol. 107, no. 6, pp. 1027–1052, 2019, doi: 10.1109/JPROC.2019.2913450.

[109] L. Seno, G. Cena, S. Scanzio, A. Valenzano, and C. Zunino, "Enhancing communication determinism in Wi-Fi networks for soft real-time industrial applications," *IEEE Trans. Ind. Informatics*, vol. 13, no. 2, pp. 866–876, 2017, doi: 10.1109/TII.2016.2641468.

[110] A. Mahmood, T. Sauter, H. Trsek, and R. Exel, "Methods and performance aspects for wireless clock synchronization in IEEE 802.11 for the IoT," *IEEE Int. Work. Fact. Commun. Syst. - Proceedings, WFCS*, vol. 2016-June, 2016, doi: 10.1109/WFCS.2016.7496508.

[111] Z. Fernandez, Ó. Seijo, M. Mendicute, and I. Val, "Analysis and evaluation of a wired/wireless hybrid architecture for distributed control systems with mobility requirements," *IEEE Access*, vol. 7, pp. 95915–95931, 2019, doi: 10.1109/ACCESS.2019.2927298.

[112] R. Costa, J. Lau, P. Portugal, F. Vasques, and R. Moraes, "Handling real-time communication in infrastructured IEEE 802.11 wireless networks: The RT-WiFi approach," *J. Commun. Networks*, vol. 21, no. 3, pp. 319–334, 2019, doi: 10.1109/jcn.2019.000013.

[113] W. Shen, T. Zhang, F. Barac, and M. Gidlund, "PriorityMAC: A priority-enhanced MAC protocol for critical traffic in industrial wireless sensor and actuator networks," *IEEE Trans. Ind. Informatics*, vol. 10, no. 1, pp. 824–835, 2014, doi: 10.1109/TII.2013.2280081.

[114] M. Luvisotto, Z. Pang, D. Dzung, M. Zhan, and X. Jiang, "Physical Layer Design

of High Performance Wireless Transmission for Critical Control Applications," *IEEE Trans. Ind. Informatics*, vol. 3203, no. c, pp. 1551–3203, 2017, doi: 10.1109/TII.2017.2703116.

[115] X. Jiang, Z. Pang, M. Zhan, D. Dzung, M. Luvisotto, and C. Fischione, "Packet Detection by a Single OFDM Symbol in URLLC for Critical Industrial Control: A Realistic Study," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 4, pp. 933–946, 2019, doi: 10.1109/JSAC.2019.2898761.

[116] "Time-Sensitive Networking : A Technical Introduction," *CISCO White paper*, 2017. https://www.cisco.com/c/dam/en/us/solutions/collateral/industry-solutions/white-paper-c11-738950.pdf.

[117] *802.1Qbv - Enhancements for Scheduled Traffic*. IEEE standard 802.11Qbv, 2016.

[118] *802.1Qbv - Enhancements for Scheduled Traffic*. IEEE standard 802.1Qcc, 2018.

[119] M. Anwar, Y. Xia, and Y. Zhan, "TDMA-Based IEEE 802.15.4 for Low-Latency Deterministic Control Applications," *IEEE Trans. Ind. Informatics*, vol. 12, no. 1, pp. 338–347, 2016, doi: 10.1109/TII.2015.2508719.

[120] "SoC-e main webpage." https://soc-e.com (accessed Sep. 01, 2020).