IEEE 1588 Clock Synchronization Performance over Time-Varying Wireless Channels

Óscar Seijo, Iñaki Val
Communication Systems Group,
IK4-Ikerlan Technology Research Centre,
Pº J.M. Arizmendiarrieta 2, 20500,
Mondragón Spain
oseijo@ikerlan.es, ival@ikerlan.es

Jesús A. López-Fernández
Department of Electrical Engineering
Group of Signal Theory and Communications,
University of Oviedo, Gijón 33203, Spain
jelofer@uniovi.es

Manuel Vélez
Department of Communications Engineering
University of the Basque Country, UPV/EHU
Bilbao, Spain
manuel.velez@ehu.eus

***Abstract*—One of the major requirements of Ultra-Reliable Low-Latency Communications (URLLC) is a high clock synchronization performance between nodes. This is especially critical in Industrial Wireless Sensor and Actuator Networks (IWSAN), where every sensor and actuator of the network must take samples with very precise timing. In this paper, we detail the challenges of distributing a global base time in wireless communications and address some insights to achieve a very high clock synchronization performance. Besides this, Orthogonal Frequency-Division Multiplexing (OFDM) has been proposed to be used in the next generation of industrial communication standards, such as SHARP, due to their robustness in high dispersive channels. Based on these considerations, we have built an IEEE 802.11a/g modem with hardware timestamping and we have evaluated the clock synchronization over several wireless channels. The wireless channels have been run in a channel emulator and they represent four different scenarios: small office, big office, open space, and small factory. The results show that clock synchronization is quite affected by channel dispersion and channel Doppler speed, but the performance of our system is good enough to its use in industrial scenarios and comparable to Ethernet-based synchronization at channels with low and medium dispersion.**

Keywords—SHARP, clock synchronization, IEEE 1588, PTP, industrial communications, IEEE 802.11, WIFI, channel emulator, Industrial wireless sensor networks, IWSN

I. INTRODUCTION

The use of wireless communications in industrial applications, such as sensor and actuator networks, is gaining interest due to the advantages of wireless communications over wired communications. These main advantages are: cost reduction, more flexibility, free movement and scalability. However, there are still some issues that must be solved to replace wired solutions with wireless solutions. One of these issues is the clock synchronization between the network elements. The sensors and actuators of the network, which measure physical variables or act on them, must perform their operations with very precise timing. Furthermore, high clock synchronization performance is also a requirement in Distributed Control Systems (DCS), where several controllers, distributed in a factory floor, need to share the same global base time. Finally, high clock synchronization performance is not only a requirement for control system operation, but also in Ultra-Reliable Low-Latency Communications (URLLC), in order to avoid inter-frame interferences and obtain a high channel usage. As an example, wireless SHARP URLLC solution [1] provides high packet rate and high reliability under the assumption that base time can be distributed with an accuracy of few tens of nanoseconds to every element of the network. Clock synchronization between elements in a network is generally done through two approaches that are described in the next paragraphs.

The first approach is based on retrieve clock time from Global Positioning System (GPS). Solutions based on this approach can achieve a precision as low as 20 ns [2], but only over good climate conditions and Line-Of-Sight (outdoor).

The second option is to distribute the clock through the communication network. To do so, the network has a Grand Master Clock (GMC) that owns the network global time and distributes it by performing a frame exchange with the network nodes to let them calculate the clock offset and the path delay between the node and the GMC. The precision of this approach can vary from some microseconds to hundred of picoseconds and mainly depends on three factors: the protocol, the implementation of the protocol, and the channel characteristics.

Precision Time protocol (PTP), which is described in IEEE 1588 standard [3], is the most used protocol to distribute clock time through an Ethernet-based wired network with high precision. The implementation of the protocol is very relevant to achieve a high precision. Using software timestamps leads to a poor synchronization in the range of some microseconds. On the other hand, using hardware timestamps greatly improves clock precision to tens of nanoseconds or less.

The nearly ideal channel in wired communications and its high bandwidth allows the clock synchronization protocols to achieve a very high precision. However, wireless channels response is typically quite far from ideal. Multipath propagation is the main cause of the channel non-ideality. The transmitted signal is scattered by the elements of the scenario and may arrive at the receiver at different times. Besides, the channel is continuously varying due to the movement of the scatterers. These two effects deteriorate the communication quality and hence the clock synchronization performance.

There are several wireless synchronization proposals in the literature which are based on PTP. Solutions with software

timestamps are simple and easy to implement. However, the jitter introduced by the software stack leads to a modest clock precision (1-5 μs) [4]. On the other hand, the proposed system in [5] achieves a precision in the order of nanoseconds using hardware timestamps, but it needs a 500 MHz ultra wide band radio. Finally, the system proposed in [6] reaches sub-nanosecond accuracy over 802.11b standard using an optimized hardware timestamping for 802.11b and not varying channel conditions. This paper also exhibits that the clock precision is greatly deteriorated under a varying channel.

To the best of our knowledge, neither of the articles in the literature includes a study of the effects of the wireless medium in clock synchronization, which can affect the maximum achievable accuracy when the clock precision is in the nanosecond range. Therefore, in this paper we briefly review the clock synchronization theory in wired links, and we further extend this review to the wireless domain, detailing the effects of the wireless links in the timestamp quality and hence in the clock synchronization performance.

In order to evaluate through measurements the effects of the wireless medium in the clock synchronization performance, we have built an 802.11g modem with hardware timestamping on a picozed SDR. The clock synchronization performance of the system has been tested over several wireless channels generated by a wireless channel emulator and different Doppler speeds. The rest of the article is organized as follows. First, in section II we suggest some proposals to achieve a high clock synchronization performance with PTP over wireless systems. The node architecture and its implementation are shown in section III. In section IV, the measurement setup is described. Clock synchronization results are shown in section V. Finally, section VI summarizes some conclusions of the article.

II. Clock synchronization with PTP

PTP clock synchronization technique is well known and described in several works and in IEEE 1588 standard [3]. It is based on the correction of the clock offset between clocks (t_{offset}), and the calculation of the channel delay (usually named path delay, and denoted as t_{ms} and t_{sm}). $t_{ms}$is the path delay from master to slave, while t_{sm} is the path delay from slave to master. PTP is based on a four frame exchange, which is shown in Fig. 1.

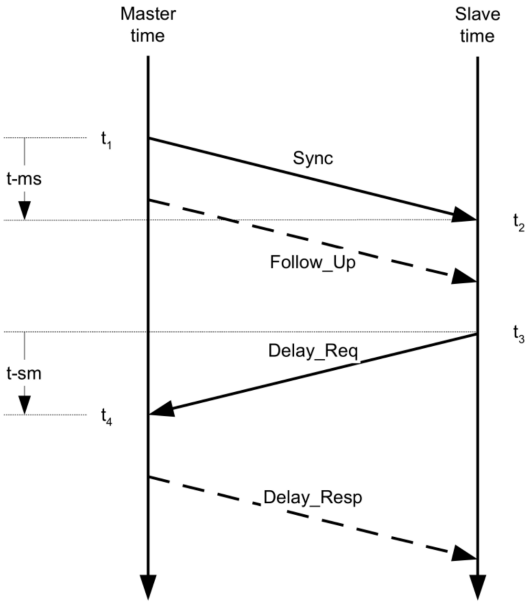

Fig. 1. PTP frame exchange [3].

Clock offset calculation is used to adjust the clock drift and the clock time, while the path delay is calculated to correct the time difference between t_1 and t_2. These calculations are stated in (1) and (2):

$$\tilde{t}_d = \frac{t_2 - t_1 + t_4 - t_3}{2}, \tag{1}$$

$$\tilde{t}_{offset} = t_2 - t_1 - t_{ms}, \tag{2}$$

where $\tilde{t}_d$ is the estimated path delay and $\tilde{t}_{offset}$ the estimated time difference between the slave clock and the master clock. $\tilde{t}_{offset}$ and $\tilde{t}_d$ may be filtered with a Proportional Integral loop (PI Loop) to reduce their variance and the clock synchronization error.

As stated in section I, software timestamps lead to poor clock synchronization results, so this article is focused on hardware timestamps. The difference between t_{ms} and t_{sm} are the main limits of clock synchronization in wired communications when hardware timestamps are used. When the channel is reciprocal t_{ms} is equal to t_{sm}, as it is the case of wireless channels.

Hardware timestamping is based on the detection of the exact moment of reception and transmission of the frame preamble. This operation is done through two steps. Firstly, the system performs the correlation of the received signal with the theoretical preamble, and then the output of the correlator is sent to a comparator that searches the first value of the correlation that passes a threshold (θ):

$$P_{rx}(t) = P_{tx}(t) * h(t) + n_i(t). \tag{3}$$

$P_{rx}(t)$ is sampled at an adequate sampling rate (T) at the receiver and the frame received is cross-correlated with the discrete equivalent transmitted preamble:

$$t = nT, n \in \mathbb{Z}. \tag{4}$$

$$\tilde{h}[n] = P_{tx}[-n]^* * (P_{tx}[n] * h[n] + n_i[n]), \tag{5}$$

$$n_{TS} = \min\{n/|\tilde{h}[n]| > \theta\}, \tag{6}$$

where $*$ denotes the convolution operator, $[\cdot]^*$ denotes the conjugate operator, and being $P_{tx}[n]$ the transmitted preamble, $h[n]$ the discrete equivalent channel impulse response and $n_i[n]$ the Gaussian communication noise component. It should be mentioned that the estimation of the impulse response ($\tilde{h}[n]$) depends on the autocorrelation properties of $P_{tx}[n]$, and it may not be equal (or even similar) to the impulse response $h[n]$. Furthermore, the value of the threshold (θ) must be carefully chosen. A high threshold would be more prone to errors in the timestamps instants, as the system will follow the highest peak. On the other hand, a low threshold would resolve the timestamps instants more accurately, but it has a higher false alarm probability, especially at low SNR conditions. It should be also mentioned that the operation stated in (6) is equivalent to perform a simple estimation of $h(t)$. When $h(t)$ is ideal (a delta) and the bandwidth is very high (such as in

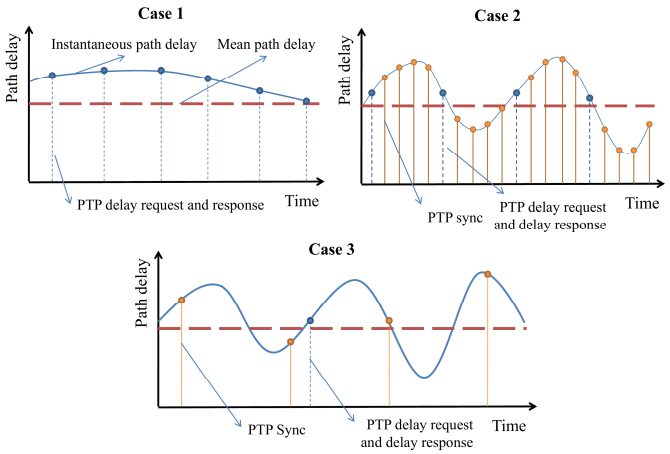

Fig. 2. Relation between PTP periods and channel variation speed. In case 1 there are multiple PTP syncs between each PTP delay request and response.

wired communications), this approximation is enough to achieve a high clock synchronization. Indeed, this operation results in a uniform random error in the timestamps of $\pm T_s/2$, being T_s the signal sampling period (inverse of the bandwidth). This imprecision is caused due to the unknown relative clock phase between the transmitter clock and receiver clock.

A. *Clock synchronization in wireless communications*

There are multiple issues in wireless communications that can deteriorate clock synchronization performance, and they are mainly related on channel characteristics. Firstly, the multipath propagation causes the transmitted signals to be received replicated and delayed at different instants, which leads to multiple local maximums in the preamble detector. Furthermore, the channel varies over time, thus the position of each of the maximums also varies, which introduces inaccuracies in the timestamps. Errors in the timestamps caused by these effects depend on the channel characteristic (delay spread and variation speed), and they could be from 50 ns to more than 500 ns. Finally, most of the wireless systems have low bandwidth, which effectively limits the time precision of the output of the correlator (5). As an example, an error in a timestamp of an 802.11g modem (20 MHz bandwidth) produces a minimum error of ± 50 ns.

Errors in the clock synchronization may also depend on the relation between the channel variation, defined by the coherence time (t_{ch}), and PTP periods, that are the PTP sync period (t_{sync}) and the PTP path delay exchange period (t_{fe}). Channel variation speed can be also defined by a maximum Doppler speed. Eq. (7) is commonly used to relation Doppler speed and t_{ch}.

$$t_{ch} = \sqrt{\frac{9}{16\pi}} \cdot \frac{c}{v_d \cdot f_0}, \tag{7}$$

where c is the speed of light, v_d is the Doppler speed, and f_0 is the communication frequency.

Three cases can be distinguished depending on the relation between t_{sync}, t_{fe} and t_{ch} (Fig. 2):

- *Case 1:* $t_{fe} << t_{ch}$. In this case, PTP can correctly follow the channel delay variations, as the path delay refresh rate is higher than the channel variation. Achieving this case is usually not possible, as the amount of traffic generated by the protocol would be higher than the system capacity.
- *Case 2:* $t_{fe} < t_{ch}$ *and* $t_{sync} << t_{ch}$. When t_{fe} is smaller but comparable to t_{ch} and t_{sync} is much smaller than t_{ch}, PTP syncs will follow the path delay variation. This effect will introduce some inaccuracies due to the path delay is refreshed with a higher period.
- *Case 3:* $t_{sync} < t_{ch}$ *or/and* $t_{sync} > t_{ch}$. PTP syncs are not able to follow channel variation in this case. If the timestamps are passed through a filter, the clock synchronization should remain near the mean delay value, thus the clock synchronization error should be lower than *case 2*.

The borders between each case depend on the PTP filter, so the relations between periods may slightly vary. PTP periods and filter should be carefully chosen in order to avoid a poor clock synchronization performance. It should be noted that *Case 2* is easily avoidable and can be converted to *Case 3* at runtime through discarding PTP sync frames, although this approach uses more traffic rate than optimizing t_{sync} and t_{fe} period.

III. System Architecture and Implementation

The system has been implemented over the Picozed SDR platform, which contains a Xilinx Zynq System on Chip (SoC) and an AD9361 radio [7] with a frequency range between 70 MHz and 6 GHz and a bandwidth of 56 MHz. The Zynq SoC comprises an FPGA and an ARM double core microcontroller in a single integrated circuit.

There are three elements that have been built in the FPGA: the 802.11a/g PHY core, the 802.11 CSMA/CA MAC Access Scheme core, and the PTP core that comprises the PTP clock and the Tx/Rx hardware timestamping modules. The Rx timestamping module uses the timestamps stated in (6).

Concerning about software, the ARM core is running FreeRTOS, a hard-real-time operating system, which is needed for high-precision applications with real-time requirements. The software over FreeRTOS includes the frame processing, and the PTP offset (2) and path delay computation (1) (PTP calculation core and PTP MAC layer). The results of the PTP computation are sent to the hardware which performs the clock correction. The node architecture is shown in Fig. 3.

The PTP clock also generates a Pulse Per Second (PPS) signal based on its time. Due to the system clock rate is 20 MHz, the PPS signal has a base resolution of 50 ns. This limits to 25 ns the resolution of the clock synchronization measurements of our system.

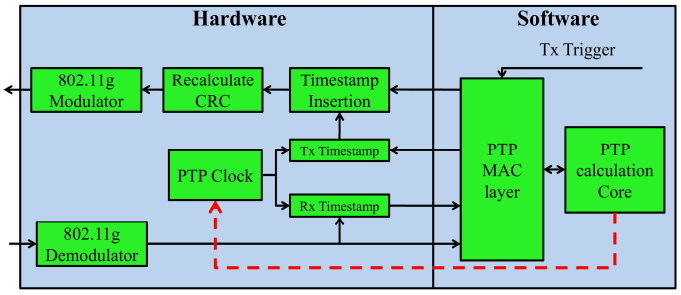

Fig. 3. Block diagram of the implemented architecture.

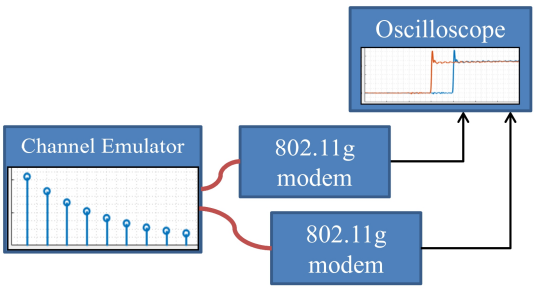

Fig. 4. Representation of the setup.

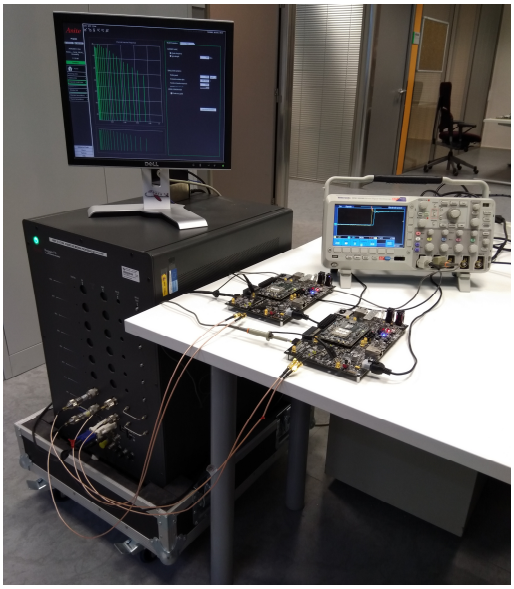

Fig. 5. Real image of the measurement setup.

IV. Measurement Setup

A. Measurement setup

The following setup was used to carry out the measurements: two boards running the system stated in section III connected through a channel emulator. One of the boards was configured to be the network GMC, while the second board was configured to be a PTP slave. The rising edges of the PPS generated by each board are introduced in a *Tektronix MSO 2040B* oscilloscope [8]. Clock synchronization error has been measured as the time difference between rising edges. Fig. 4 and Fig. 5 illustrate the setup.

The channel emulator is a *Propsim F8 Radio channel emulator* provided by Anite (Keysight) [9]. The model offers a wide range of configurable characteristic. This particular emulator has:

- 40 MHz bandwidth.
- Frequency range from 350 MHz to 3000 MHz.
- Mobile speed from 0.06 km/h to 11000 km/h.
- Configurable output gain from -100 dB to 0 dB.
- It includes several default channels for WLAN scenarios and the possibility of creating custom channels.
- Several fading models (Rayleigh, Rice, Nakagami, log-normal, Gaussian, flat, constant, etc.)
- A t_{ms} of approximately 3.1 µs when emulating a channel with only 1 tap with constant fading.

TABLE I. CHANNELS CHARACTERISTICS

Name	Scenario	LOS / NLOS	rms Delay Spread [ns]	Fading Model
WLAN Ind	Industrial	LOS	29	Classical
WLAN A	Small Office	NLOS	50	Classical
WLAN B	Large Office	NLOS	100	Classical
WLAN C	Large open Space	NLOS	150	Classical
WLAN E	Very large open Space	NLOS	250	Classical
WLAN Control	None	-	0	Constant

*Line-Of-Sight (LOS), Non-Line-Of-Sight (NLOS)
*Classical fading model: Rayleigh Fading and Jakes Doppler

The path delay of the emulator is much higher than a typical path delay in a WLAN scenario, which is from 10 ns to 300 ns. Thus we configure our PTP algorithm to compensate 3 µs. With this compensation, the PTP behavior through the channel emulator is similar to the behavior of a wireless link. It must be taken into account that the channel emulator path delay may not be reciprocal due to it is a wired system, thus it may introduce additional errors in the clock synchronization.

We have taken 500 clock error samples for each scenario, which should be enough to calculate with high precision the mean (µ) and accuracy (σ) of the clock synchronization.

1) Wireless channel models

We have carried out measurements over five wireless channels and a control channel (*WLAN Control*). The control channel is an ideal wireless channel with no time variation and an impulse response of a delta. Four of the wireless channels are included in the channel emulator by default and they are deeply described in [10]. The remaining wireless channel is the industrial wireless channel included in [11] (named scenario 7), referred in this article as *WLAN Ind*. Channels main characteristics are shown in Table I. Although *WLAN Ind* was measured with Line-Of-Sight conditions, the huge multipath in the industrial environment resulted in a Rayleigh-like behavior that is a typical distribution in Non-Line-Of-Sight conditions.

We have evaluated the synchronization performance over 5 speeds: 2 km/h, 10 km/h, 30 km/h, 100 km/h, and 300 km/h. The three first speeds could correspond to a factory/process automation scenario [12], while the last two speeds could be an automotive and high-speed railway scenario respectively.

2) Modem and PTP Configuration

The PTP configuration used in the measurement setup was:

- Sync packet period: 10 ms.
- PTP frame exchange period: 100 ms.
- One-Step PTP.
- Time elapsed between PTP sync and PTP delay request: 200 µs.

The hardware configuration was:

- f_0: 2.6 GHz.
- Bandwidth: 20 MHz.
- 802.11g Modulation and Encoding Scheme: BPSK with channel coding of ½ (MCS = 0).
- PPS resolution: 50 ns.

TABLE II. CLOCK SYNCHRONIZATION MEAN (μ) AND ACCURACY (σ) OVER SEVERAL WIRELESS CHANNELS

Doppler Speed [km/h] / Channel	0		2		10		30		100		300	
	μ [ns]	σ [ns]	μ [ns]	σ [ns]	μ [ns]	σ [ns]	μ [ns]	σ [ns]	μ [ns]	σ [ns]	μ [ns]	σ [ns]
WLAN Ind	-	-	6.3	27.7	-1.38	28.0	-1.47	29.2	0.85	28.8	0.54	29.4
WLAN A	-	-	3.70	31.7	11.2	27.8	9.22	28.4	11.3	27.3	-3.97	31.5
WLAN B	-	-	8.82	35.1	-0.7	30.9	-1.20	29.9	2.48	30.8	-8.9	34.2
WLAN C	-	-	8.96	35.2	8.09	30.6	-4.39	28.4	-5.10	32.1	-18.2	37.4
WLAN E	-	-	-9.1	66.0	-10.7	44.3	-5.8	43.7	-3.89	49.1	5.53	70.2
Control	3.7	25.5	-	-	-	-	-	-	-	-	-	-

It should be noted that the PPS resolution limits the measurement capacities of our system. Due to this, the measurements have a precision bound of ±25 ns. Furthermore, the measurements have been carried out at a center frequency of 2.6 GHz, instead of at a channel of the crowded 2.4 GHz band, in order to avoid external interferences from other communications systems.

V. RESULTS

Clock synchronization mean and accuracy results are indicated in Table II. The most representative cases are represented through histograms in Fig. 6.

Focusing on clock synchronization accuracy (σ), we have found that the clock synchronization performance is deteriorated over channels with high delay spread: a high delay spread leads to multiple peaks in the preamble detector (6), causing inaccuracies in the timestamping instants. As shown in the measurements, this effect is especially critical in channels with a high delay spread, such as WLAN E channel. On the other hand, if we study the results as a function of channel speed we found that the system seems to have a better synchronization when the Doppler speed is higher. In fact, this effect is perfectly described by *Cases 2* and *3* stated in section II.*A*. The speed of 2 km/h corresponds to *Case 2*, while the rest of the speeds correspond to *Case 3*. When the speed is very high (300 km/h) the performance of our modem is very low (PER > 10^{-2}), so the synchronization is also deteriorated compared with the cases with lower speeds.

The measurements show that the clock synchronization accuracy at speeds of 10, 30 and 100 km/h is very similar in

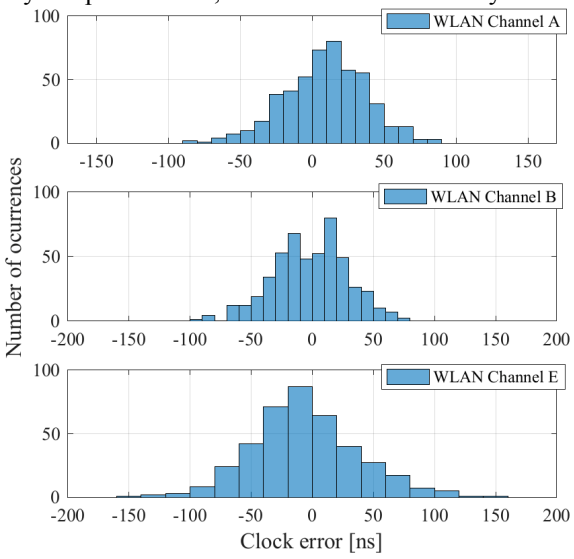

Fig. 6. Clock error in channels A, B and E with a Doppler speed of 30 km/h.

TABLE III. CLOCK SYNCHRONIZATION MAXIMUM ERROR

Doppler Speed [km/h] / Channel	0	2	10	30	100	300
WLAN Ind	-	84	83	105	101	97
WLAN A	-	99	105	88	100	123
WLAN B	-	109	124	97	109	132
WLAN C	-	123	118	71	102	182
WLAN E	-	188	149	153	189	194
Control	65	-	-	-	-	-

*Values are in ns

WLAN Ind, WLAN A, WLAN B, WLAN C and *WLAN E* channels. This can be also described by *Case 3* conditions: the channel varies so fast that every PTP calculation is made under different channel realizations. If the PTP filter has low bandwidth, it is able to filter channel fast variations, thus the clock synchronization remains more stable.

Focusing on the mean (μ) results, they are really close to 0 ns, the ideal value, although they appear to be erratic between the different cases. This is probably due to the PTP calculations have a small bias in the estimation of the emulator path delay.

Finally, maximum clock error is shown in Table III. As expected, the maximum error seems to increase at channels with higher delay spreads. However, the small number of samples taken for each scenario limits the representativeness of the clock synchronization maximum error.

A. Towards subnanosecond clock synchronization over time-varying and high-dispersive wireless channels

As stated in section II, the classic timestamping method based on the detection of the peak between the correlation of the received frame and the preamble is very precise in wired communications, as the bandwidth is high and the impulse response is nearly ideal. Thus, the largest source of clock synchronization error is due to the difference between t_{ms} and t_{sm}. In wireless scenarios, our measurements show that there are three main error sources in clock synchronization over wireless systems: inaccuracies in the timestamp instants caused by multipath propagation, path delay variation, and low bandwidth. The measurements also show that the deterioration of the clock synchronization is especially visible in channels with a high delay spread.

To address wireless channels issues, we are currently working on the design of an improved timestamping method that should be immune to dispersive channels and may be very robust against time-varying channels. The main idea is to take advantage of the channel delay diversity to accurately set the timestamp instants and obtain a great improvement in its precision. This timestamping method would soften and disperse the errors over every timestamp. If the error in the

timestamps is low enough, it may be also possible to achieve timestamps as precise as the timestamps proposed in [6], but over dispersive channels and OFDM-based communications. If these assumptions are correct, it may be possible to achieve sub-nanosecond clock synchronization performance over high-dispersive and time-varying channels.

VI. Conclusions

In this paper, we have addressed the challenges of the clock synchronization over wireless channels. The time-varying wireless channels character, the non idealities of the channel impulse response and the low communication bandwidth are the main challenges of clock synchronization over real wireless channels. Our measurements show that the wireless medium deteriorates the timestamping quality, leading to a worse clock synchronization performance, especially in channels with high delay spread. However, it continues to meet the clock synchronization requirements of most industrial networks under real wireless conditions and its performance is very similar to that of Ethernet-based systems.

However, wireless channel variability is still a challenge to clock synchronization. Although our implementation has robust clock synchronization over time-varying wireless channel, maximum clock error bounds cannot be established due to channel variation unpredictability in real conditions.

Finally, it may be still possible to further improve the clock synchronization performance in wireless links. In future works we will deeply study in both theoretical and experimental domains how to take advantage of the channel delay diversity to provide sub-nanosecond clock synchronization in real wireless scenarios.

Acknowledgment

This work has been partly supported by DETRED (ELKARTEK 2017) project of the Basque Government (Spain) and by ARTEINE (TEC2017-86619-R) project of the Spanish Ministry of Economy and Competitiveness.

References

[1] 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.

[2] F. Gao, Z. Huang, S. Wang, and Z. Wang, "A Hybrid Clock Synchronization Architecture for Many-core Cluster System Based on GPS and IEEE 1588," pp. 2645–2649, 2016.

[3] *IEEE Standard for a precision clock synchronization protocol for networked measurement and control systems*. IEEE Standard 1588, 2009.

[4] 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.

[5] 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.

[6] R. Exel, "Clock Synchronization in IEEE 802.11 Wireless LANs using Physical Layer Timestamps," 2012.

[7] "AD9361 Reference Manual UG-570." [Online]. Available: http://www.farnell.com/datasheets/2007082.pdf.

[8] Tektronix, "DPO2000 and MSO2000 Series Oscilloscopes User Manual," 2004.

[9] "Propsim Channel Emulation Solutions." [Online]. Available: https://www.keysight.com/en/pc-2697334/propsim-channel-emulation-solutions.

[10] Anite, "Propsim Standard Channel Models," 2013.

[11] 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.

[12] P. Schulz *et al.*, "Latency Critical IoT Applications in 5G: Perspective on the Design of Radio Interface and Network Architecture," *IEEE Commun. Mag.*, vol. 55, no. 2, pp. 70–78, 2017.