Università degli Studi di Padova

Padua Research Archive - Institutional Repository

A Calibrated Test-set for Measurement of Access-Point Time Specifications in Hybrid Wired/Wireless Industrial Communication

(Article begins on next page)

Article

A Calibrated Test-set for Measurement of Access-Point Time Specifications in Hybrid Wired/Wireless Industrial Communication‡

Federico Tramarin [1,†], Claudio Narduzzi [2], Stefano Vitturi [1], Matteo Bertocco [2]

[1]   National Research Council of Italy, CNR–IEIIT, Padova, Italy
[2]   Dept. Information Engineering, University of Padova, Padova, Italy
*   Correspondence: federico.tramarin@ieiit.cnr.it; Tel.: +39-049-827-7639
†   Current address: Via Gradenigo 6/B, I–35131 Padova, Italy
‡   This paper is an extended and improved version of our paper published at the 2012 IEEE International Instrumentation and Measurement Technology Conference, Graz, Austria, 13–16 May 2012

Abstract: In factory automation and process control systems, hybrid wired/wireless networks are often deployed to connect devices of difficult reachability such as those mounted on mobile equipment. A widespread implementation of these networks makes use of Access Points (APs) to implement wireless extensions of Real–Time Ethernet (RTE) networks via the IEEE 802.11 Wireless LAN (WLAN). Unfortunately, APs may introduce random delays in frame forwarding, mainly related to the their internal behavior (e.g. queue management, processing times), that clearly impact on the overall worst case execution time of real–time tasks involved in industrial process control systems. As a consequence, the knowledge of such delays becomes a crucial design parameter, and their estimation is definitely of utterly importance. In this scenario, the paper presents an original and effective method to measure the aforementioned delays introduced by APs, exploiting a hybrid loop-back link and a simple yet accurate set–up with moderate instrumentation requirements. The proposed method, which requires an initial calibration phase by means of a reference AP, has been successfully tested on some commercial APs to prove its effectiveness. The proposed measurement procedure is proven to be general and, as such, can be profitably adopted in even different scenarios.

Keywords: Real-time systems; Industrial networks; calibration; measurements; access point; IEEE 802.11; WLAN

1. Introduction

At the lowest level (or *field level*) of factory automation systems, the communication infrastructure typically supports fast cyclic exchange of small payload data packets between sensors/actuators and controllers [1]. A controller cyclically polls its attached nodes under tight timing requirements which, in turn, imply that packet delivery is subject to strict determinism where jitter has to be kept as low as possible and, in any case, below a specified threshold [2]. Time guarantees may also have to be provided for aperiodic traffic, since event notification messages (the most common type of aperiodic traffic in industrial applications) require real–time operations with quite tight delivery deadlines.

The demanding requirements of this communication environment, briefly summarized in Table 1, have been largely addressed by wired networks such as field-busses (e.g. Profibus DP [3], CANopen [4] and DeviceNet [5] to mention some) and Real-Time Ethernet (RTE), that is an effective family of communication systems for factory automation developed in the last decade [6], [7].

Moving to the wireless domain, although dedicated industry standard solutions are currently available for process automation scenarios, such as WirelessHART and ISA 100.11a [8,9], so far stand-alone wireless networks have not been widely considered at the field level in factory automation applications. Wireless links can nevertheless be found wherever network nodes can be used to effectively solve mobility or cabling-related issues in factory automation and process control systems. In this context, the IEEE 802.11 wireless LANs family of standards provides an effective answer to

Table 1. Typical requirements of industrial control applications

Scenario	No. of nodes	Cycle time/Deadline
Building Automation	1000	10 s
Process Automation	10000	100 ms
Factory Automation	100	1 ms
Power System Automation	100	100 μs
Power Electronic Control	100	10 μs

the conflicting needs of cabling simplification, coexistence, real-time behavior and reliability [10–12], since it is able to provide performance theoretically comparable to those of RTE networks up to factory automation contexts.
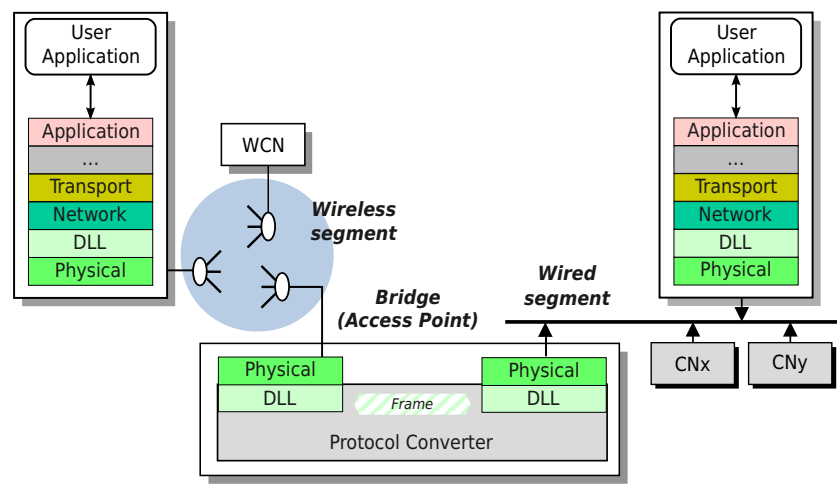

Figure 1. An AP serving as a bridge between a wireless extension of a wired network.

As a matter of fact, a common employment of the IEEE 802.11 WLAN is the development of hybrid solutions that include wireless extensions of wired networks [13]. A hybrid wired/wireless network is obtained by the interconnection of two different communication subsystems by means of an *Intermediate System* (IS), which is inserted at a specific layer of the ISO/OSI stack. In a IEEE 802.11–based wireless extension of a wired RTE link, interconnection takes place at the Data Link-Layer (DLL), just above the physical layer, by means of an *Access Point* (AP) acting as a bridge between the two sides (Fig. 1) and providing connectivity in a so–called "infrastructured" network. Such hybrid solutions are deployed, often, to solve reachability issues and to serve a distant or separate production cell in a factory automation system. In this perspective, the wireless segment is interested by traffic profiles mostly relevant to automation purposes, hence with real–time and determinism requirements.

In such configurations, particular care has to be paid to the possibly non–negligible random delays the AP may introduce while forwarding data from the wired part of the network to the wireless segment ad vice versa. Indeed, considering the aforementioned stringent requirements about factory automation networks, such delays may significantly impact on the timeliness of the network and consequently on the the performance of the whole factory automation system [14]. In facts, in a number of scenarios the AP–related latencies together with other protocol latencies may constitute, in the case of real–time communications, a lower bound to the task completion time eventually limiting the real–time performance of the network itself.

Being a potentially critical device, the performance figures of an AP should be known and well understood in advance. Unfortunately, AP specifications are seldom documented with the necessary degree of detail and, particularly, the frame forwarding delays and queue management are usually not provided, even for devices specifically conceived for industrial applications [15]. In real–time factory

communication systems, when an AP is deployed to extend a wired RTE segment, such a device is required to provide suitable time guarantees. Therefore, the characterization of an AP represents a crucial task that has to be pursued via appropriate measurements.

The goal of this paper is to address the aforementioned issue, providing a general, effective and simple–to–use measurement method for characterizing the AP forwarding latency, namely D_{AP}. To this aim the paper presents a calibrated test-set enabling the measurement and discusses the necessary calibration steps.[1] In particular, a dedicated hybrid wired/wireless loop-back link is employed where the device under test (DUT) is being placed within the loop. The test–set is at first calibrated by means of a characterized reference AP, so that the contributions to total transmission delay that are known to be unrelated to the DUT can be compensated for. Finally, the collected DUT data are post–processed by a deconvolution algorithm to provide the required corrected information.

To prove the effectiveness of the proposed method, we present a set of experimental results aiming to show that measurements can accurately characterize AP devices, and could also help to verify and validate analytic models. The final aim of this work is hence to define a general, simple and inexpensive measurement procedure, rather than discussing the actual latencies verified in our experiments. In this light, the proposed approach, which has been described in the context of factory automation systems, is demonstrated to be not related to a specific field of application and hence it can be adopted in several different scenarios.

2. Measurement Test-set

Since one of the aims of an industrial network is the timely delivery of packets, in a hybrid network there should be ideally no difference, in this respect, between the wired and the wireless segment. Knowledge of the degree of determinism provided for this basic service is a key aspect for the effective implementation of a wireless extension to a RTE network. Unfortunately, AP designers do not usually have such system requirements in mind targeting at different applications.

Indeed, an AP is typically realized as an embedded system, running a lightweight operating system (OS) that manages the supported protocol stacks, implements bridging and schedules the execution of traffic and queue management algorithms. The architecture is conceived with general purpose communication, Internet browsing, file sharing, *etc.* as target applications, so that delays an AP may introduce are not guaranteed nor specified, even for devices declared to be specifically designed for industrial usage [17]. Clearly, this does not represent an issue in general purpose, home or office communications, where the aforementioned delays are much lower with respect to other delay sources and are anyway well managed by more complex protocol mechanisms. Nevertheless, the diverse scenarios provided by the case of industrial communication systems for factory automation share different requirements, where delay sources should be limited to the minimum. In fact, looking the problem from a real–time system perspective, what is needed is to have network components whose delays are accurately known and characterized, in order to take them into account in the planning of network cycle, or at a scheduling phase.

This shortcoming provided the motivation for developing the test-set presented in this paper. The loop-back scheme shown in Fig. 2 is the simplest AP measurement test-set that can reproduce the hybrid network effects of a factory automation system. It can be simply implemented through the use of a general purpose personal computer (PC) equipped with two communication interfaces, namely an Ethernet card and a wireless LAN adapter.

The device under test is connected to the two interfaces, being stimulated by the traffic patterns generated by the PC and forwarding packets within the communication loop. Packet timestamps, taken by the hardware time–stamp counter of the PC microprocessor, allow the characterization of AP time specifications.

---

[1]   The work takes origin from the research reported in [16].
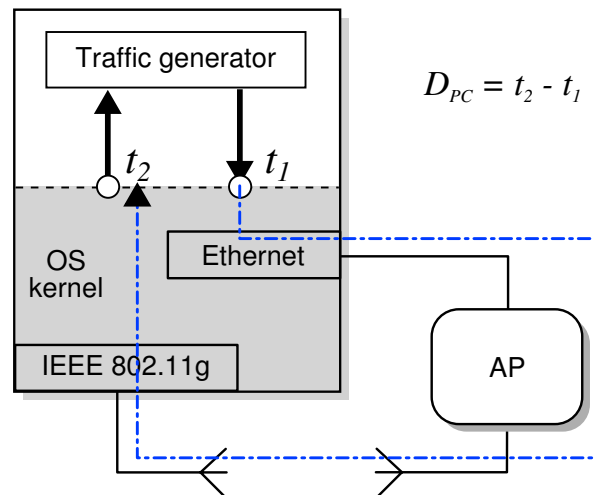
Figure 2. The setup adopted for emulating a simple hybrid industrial network.

In principle, a simple measurement of the one-way delay experienced by a packet through the hybrid link, e.g., outgoing from the controller (Ethernet interface) to its attached wireless slave (IEEE 802.11 interface), is obtained by comparing the send and receive timestamps taken at the respective interfaces. To the aim of reducing measurement uncertainty a significant option is to perform an hardware timestamping. To accomplish this goal, we rely on the Time Stamp Counter (TSC), which has been long recognised as a reliable timing source. Moreover, since all the generated timestamps are then employed within the same PC, the use of TSC is possibly the simplest solution and, in our opinion, a very practical way to measure latencies with adequate time resolution. As a further consideration it is worth observing that the adoption of more structured solutions, such as the IEEE 1588 standard, even though they may provide intrinsic traceability to a calibrated time source[2], appears to be rather overkill for the purpose of designing a simple test set, given also the intrinsic synchronization of the proposed measurement setup. On the basis of these consideration, providing accurate hardware timestamping is by no means trivial and a more detailed discussion of its implementation in this work is necessary.

2.1. Timestamping

Packet timestamping is a well-known problem in network measurement, where an acceptable tradeoff between accuracy and implementation complexity has to be defined. In fact, the time when a packet physically leaves a network interface can be determined precisely, but this typically requires dedicated measurement hardware. On the other hand, monitoring a communication interface at a higher-level protocol layer is often easier but timestamp uncertainty may be larger, since timing jitter with respect to a low–level hardware timestamp tends to be introduced as the packet progresses through the lower protocol layers.

Measuring network interface boards providing hardware timestamping have been developed for a number of research and commercial projects, such as the well–known Endace DAG Cards (Data Acquisition and Generation packet capture cards [18]). In general, these boards provide high-resolution counters and a stable internal clock, which can be synchronized to an external reference such as a GPS signal for distributed network measurements or for synchronization purposes.

For the application considered in this work, the level of cost and complexity associated with such hardware appears to be unjustified. However, as the test-set is aimed at the characterization of a single device, hardware timestamps can be obtained from a common time reference within the PC that

---

2   In the proposed setup the same goal can be equally obtained through basic laboratory instrumentation, e.g., a counter

manages the test. This results in a simpler measurement set-up, where any possible synchronization issue between the clocks of different hardware measuring interfaces is avoided. Measurements can prove equally reliable, provided suitable calibration steps are introduced so that measurement uncertainty can be mostly attributed to residual short-term clock instabilities.

It is worth observing that different possible sources for a time reference within a computing system actually exist, and have been used within time related functions in operating systems, such as a Real-Time Clocks (RTC) or, more recently, the High Precision Event Timer (HPET). These are basically provided by an uncompensated quartz crystal oscillator, typically providing a frequency of about 14.318 MHz, and while they may result adequate for interrupts generation and for triggering scheduling events, they are far from acceptable in code profiling and in time measurement applications both for the comparatively low resolution they provide and for their intrinsic instability.

Conversely, in this paper our choice has been to adopt a superior high–resolution and low–overhead source for gathering CPU time measurements, that is, the internal microprocessor timestamp counter (TSC) commonly found in many recent CPUs [19]. This register counts CPU clock cycles, that is it gets incremented internally by the CPU at every single clock cycle from the last CPU power up or reset. A significant observation is that it provides a deterministic[3] way of measuring the flow of processors' operations, rather than time, which can be theoretically obtained considering the measured CPU frequency.

The specific microprocessor instruction to read the TSC microprocessor register is RDTSC, which returns the 64-bit value stored in the TSC counter as two 32-bit values stored within the two internal registers EDX:EAX. A basic snippet of code that allows to read the TSC value within a program written in C language is given in Listing 1. Intuitively, the overhead which is incurred in reading the TSC is basically due to the execution time of this instruction and to the need for transferring the content of these two register in the main process stack memory.

```
static __inline__ unsigned long long rdtsc(void) {
    unsigned hi, lo;
    asm volatile("rdtsc" : "=a"(lo), "=d"(hi));
    return ( (unsigned long long)lo)|( ((unsigned long long)hi)<<32 );
}
```

Listing 1: A C snippet to read the TSC register

However, it is of prominent importance considering that the introduction of several features to increase modern processors performance, such as power saving, frequency scaling, out–of–order and speculative execution, requires that several cares are considered in the adoption of the TSC for direct time measurement. Nonetheless, if used cleverly, for current architectures the TSC can provide very high-resolution timestamps (resolutions around 10 ns) and for this reason it has already been considered in network monitoring projects (e.g., [20–22]).

The first and most trivial feature that affects the accuracy of time stamps taken through the TSC is the frequency scaling which characterizes almost all modern CPUs, to allow for better power consumption and thermal dissipation. This feature may cause an inaccurate interpretation of the value T_{TSC} read from the TSC, which indeed accounted for CPU cycles, and not directly time. The relationship with current time t_{ct} is through the CPU's actual frequency f_c, that is, $t_{ct} = T_{TSC}/f_c$. Additionally, we should consider that modern architectures are typically multi–core. Each CPU core has its own TSC register and one should take into account that there is no guarantees in general that all TSC registers in the system are kept synchronized, nor that their tick rate is equal throughout time.

---

[3]　Indeed, TSC values are incremented deterministically, and there is no way to interfere with the counting process. Clearly, reading the TSC value may instead be subjected to non–deterministic latencies.

In our work, we avoided these issues by disabling both power saving features and frequency scaling at the operating system level, imposing for all the CPU cores in our system to work at the nominal maximum frequency. Moreover, we programmed our traffic generator so that the corresponding process would be executed always on the same CPU core, by specifying a suitable CPU affinity mask to the system scheduler.

A further issue is due to the out–of–order execution feature typically found in modern processors, where instructions are not necessarily performed in the same order they are written in the code. Hence, the RDTSC instruction can be shifted before or after its intended location, further decreasing the accuracy of its value. To avoid such an effect the most effective action is to insert a "serializing" instruction just before the RDTSC one, i.e. an instruction which forces all the other operations to finish before its execution. A typical example of such an operation is the CPUID instruction.

All the aforementioned precautions have been introduced within the traffic generator software purposely designed for this work, to the aim of reducing as much as possible the uncertainty associated to the timestamping procedure. Indeed, as a final observation, according to the literature [23] it is possible to state that the a carefully programmed TSC is able to provide a short-term stability, which is the main concern for the purposes of this work, that can be considered good enough and satisfactory for the intended goals. Also, given that all timestamps are generated and employed within the same PC, hence sharing the same time source, the use of TSC is possibly the simplest solution and, in our opinion, a very practical way to measure latencies with adequate time resolution, without resorting to complex synchronization systems, such as IEEE 1588.

2.2. System and Traffic model

In factory automation several traffic profiles are multiplexed on the same network, which can be mainly categorized in real–time isochronous (i.e. periodic) exchange of small frames, real–time asynchronous frames (i.e. alarms, set points, etc.) and classic general purpose traffic not subjected to real–time constraints.

In this framework, the different profiles are also characterized by different characteristic length of the involved frames. As an example, the use of longer packets (several hundreds to thousands of Bytes) is typically relevant to background non real-time traffic, such as Internet traffic, management, etc, for which the forwarding latency of the AP are typically not an issue, as may be also inferred by the fact that in several RTE protocols this traffic class is possibly delayed or queued to prioritize real–time traffic [24–27].

The most important pattern of traffic within a typical automation network is, instead, relevant to the transfer of sensors/actuators' readings within a periodic cycle, under timeliness constraints and typically based on a simple TDMA scheme (Fig. 3). New packets, typically of small sizes, are sent within cycle slots, and hence the inter–departure times may be either fixed (the most common situation) or variable on a rather discrete basis.

For the sake of this work, the system is designed taking into account this most critical type of traffic, i.e. small, cyclic data transmission, hence subjected to real-time and deterministic constraints. In this traffic model, a new packet is sent on a regular basis every t_{cycle} seconds. Also, the inter–departure time between two consecutive packets can be easily varied dynamically in order to emulate different design solutions of a factory automation system.

To implement such a traffic profile, a suitable traffic generator software has been designed, running within the PC at the application level (refer to Fig. 2). Specifically, it delivers a packet to the specified communication interface (typically the Ethernet port) and waits for the same packet arriving back from the other interface. This software allows the tuning of different communication parameters, such as the length of packets, the inter–departure time, etc.

This has been purposely developed with the goal of reducing the impact of latencies due to OS interrupts, by adopting low–level system calls towards the OS Linux kernel and the processor registers. Also, carefully programmed "raw" sockets have been used, in order to avoid any protocol overhead
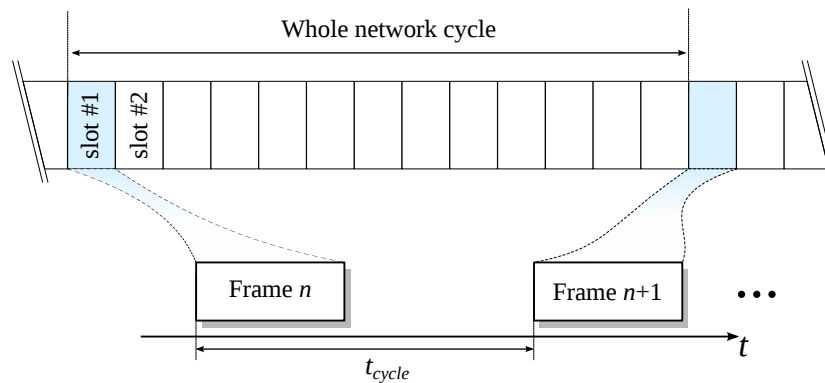
Figure 3. The generated traffic profile.

both at the networking and the processing levels. Relying on low–level calls helps reducing packet time stamping uncertainty, since the sequence of computations and call is accurately designed and optimized for just the required operations, hence minimizing the intervention of "generic" libraries and foreign uncontrolled pieces of code.

Timestamps are taken by the traffic generator itself, following the guidelines provided in Sec. 2.1, at the interfaces with the relevant protocol layers. With reference to Fig. 2, a packet is delivered by the packet generator to the kernel at the timestamped instant t_1. The time stamp t_2 indicates the instant when the process receives the same packet back through the loop. This allows, adopting the precautions about TSC readings discussed above, and taking into account the microprocessor frequency, to obtain the delay measured at the OS level as the difference $D'_{PC} = t_2 - t_1$. In the rest of the paper we will refer to this delay as D'_{PC}, where the meaning of the prime sign is that this is the whole delay measured by the traffic generator, and includes both deterministic and random terms, as will be detailed afterwards. It is worth observing, however, that this procedure measures a number of different contributions, where the delay D_{AP} introduced by the AP is somehow hidden within.

The differences between the above two quantities are now discussed, with an extended view on removing them by means of test-set calibration.

2.3. Considerations about the wireless link

In the research literature there is not a single setup that has been proposed for the use of IEEE 802.11 in factory automation systems, as well as a standard "industrial" implementation is absent. The main reason of this long debate is to find a shared alternative to the Distributed Channel Function (DCF) channel access function, that represents a significant source of randomness in the timings related to frame delivery [17,28], and that, if possible, should be avoided in real-time communication setups.

However, the alternatives provided by the same IEEE 802.11 specifications (i.e. PCF and the most advanced HCCA) that could be profitably exploited in the intended scenarios are seldom (or almost never) implemented in real components, even if their theoretical performance figures would solve several issues in these applications. Particularly, with reference to the IEEE 802.11e amendment, it introduced the new HCF function, which enabled two more channel access functions, namely EDCA and HCCA. The latter, as already said, is almost never implemented in real APs. EDCA provides support for traffic classes through the use of different AIFS and for TXOP to provide a time-limited contention free channel access to stations. From a commercial point of view, this has been called WiFi multimedia (WMM), and almost any recent device is able to support at least a part of these functionalities.

There have been also efforts to design new MAC–layer protocols to support real-time operations in industrial WLAN communications, such as RT-WiFi [26] and iWLAN [29] by Siemens. The first one is a proposal for a protocol design based on the IEEE 802.11 PHY to enable "real" real-time communications, but at the moment there are no public implementations, despite its high effectiveness.

The proprietary system proposed by Siemens in its APs for industrial traffic, namely iWLAN, is based on a variation of the standard PCF not disclosed publicly: scientific works actually analyzed its performance, but there is no effective way to model this function and hence to compare its performance with baseline 802.11 systems. Actually, in one of our tests we have included a Siemens APs supporting the iWLAN function in order to understand, at an initial stage, if the component presents a much different behavior with respect to a simple low-cost commercial one.

In this paper, hence, considering also the traffic models we considered, all devices included in our tests supports at least WMM. The adoption of such prioritization functions impacts on two different sides: the transmission time and, eventually, the AP delay. The former reflects in possibly different interframe spaces (IFS) to be accounted for in Eq. (1), but these terms are deterministic, known and defined by the standard. The latter are more relevant to the use of possibly different queues within the AP, but if we limit to industrial traffic, then it is common to use a single traffic class.

A further consideration about the wireless link is also necessary. This paper focuses at first on the calibration phase of the measurement system, as discussed in the next Section, and the need to provide a method with a high degree of effectiveness, repeatability and reproducibility of the different involved steps. Hence, in this initial procedure we have to face an accurate characterization of the different delay terms (see next Eq. (1) for reference): the presence of retransmissions in this phase, due for instance to collisions/interference, would introduce further contribution due to backoff times and frame transmission times possibly at different rates (whose value may be unknown since very few APs declare what is the rate selection policy they implement). Actually, even hypothesizing to know precisely the number of involved retransmissions and the respective rates, the distribution of backoff times would add a further random term that would make impracticable the characterization of the whole measurement chain.

Hence, the next calibration phase is carried out under controlled and reproducible conditions. This is a common procedure with measurement instrumentation, where the initial calibration is carried out in a controlled and well-known configuration.

3. Test–set calibration

An illustrative sketch about the relationship between D_{AP} and D'_{PC} is represented in Fig. 4. Here, we highlighted the different terms involved in the frame delivery process. In particular, we indicated generically with D_{OS} the overall latency relevant to the operating system, which include delay contributions due to the execution of the involved system calls, to the processing time required by device drivers operations, to latencies within communication interfaces (i.e., the Ethernet and wireless card drivers, *etc.*). A description of the different terms represented in the figure follows.
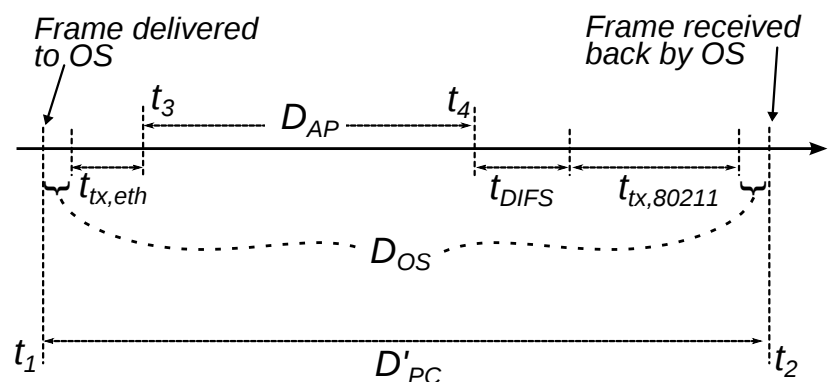
Figure 4. A schematic representation of the different terms of delay that are measured by the traffic generator, and are included in D'_{PC}.

A data packet is delivered by the traffic generator to the OS kernel at the timestamped instant t_1. Then, it takes some random time (that is part of D_{OS}) before this packet traverse the system, is

processed by the Ethernet Network Interface Card (NIC) device driver, and finally is actually delivered through the Ethernet transceiver.

After a transmission delay $t_{tx,eth}$ (defined by the standard), the whole frame is decoded at the physical Ethernet transceiver of the AP at time t_3. The frame is then passed up to the Data Link Layer (DLL), where protocol conversion from the IEEE 802.3 to the IEEE 802.11 format is performed (see Fig. 1). At time instant t_4 the resulting frame is delivered at the input of the wireless transceiver, where it is used to generate the modulated physical signal to be transmitted. The time interval $t_4 - t_3$ is then the delay introduced by the AP and, apart from protocol conversion, it is composed by various contributions such as queue management, coding, logging, *etc*.

For the sake of mathematical tractability, wireless transmission follows the rules given by the DCF channel access scheme of the IEEE 802.11 standard, found in [30]. Under the assumptions of an interference free channel and correct transmission at first attempt, that hold for the proposed test-set, the radio chip shall sense the medium for a fixed amount of time (t_{DIFS}) and, since it is sensed idle during this period, the transmission may starts, lasting for a specific and easily obtainable transmission time [31]. Therefore, the time D'_{PC} necessary to receive a frame back on the PC can be decomposed into the following sum of terms:

$$D'_{PC} = t_{tx,eth} + D_{AP} + t_{tx,802.11} + t_{DIFS} + D_{OS} \tag{1}$$

where the one-way delay is $D'_{PC} = t_2 - t_1$, while delay due to the AP is $D_{AP} = t_4 - t_3$. Hence, calibration of the measurement set-up is required to infer D_{AP} from the timestamp difference $t_2 - t_1$.

In Eq. (1), signal propagation times can be neglected on both the wired and the wireless segment, since distances among components were very short in the test-set. The terms $t_{tx,eth}$, $t_{tx,802.11}$ and t_{DIFS} represent, with very good accuracy, deterministic delay contributions defined in the relevant standard documents [30,32], whose parameters and values are reported in Table 2.

Table 2. Communication parameters

Description		Value
Application sending periods	t_{cycle}	$3, 5, 10, 15, 30$ ms
Application payload size	l	46 Bytes
IEEE 802.11g transmission rates		54 Mb/s
IEEE 802.11g transmission channel		6 (2.437 GHz)
Short Inter-Frame Space	SIFS	$10\ \mu$s
Distributed Inter-Frame Space	t_{DIFS}	$28\ \mu$s
Slot Time		$9\ \mu$s
Max number of retransmissions (N_{max})		7
CTS protection		Switched off
Encryption		Open-system
Transmission time for Ethernet frames	$t_{tx,eth}$	$5.76\ \mu$s
Transmission time for IEEE 802.11 frames @54 Mb/s	$t_{tx,802.11}$	$42\ \mu$s
Transmission time for IEEE 802.11 ACKs @24 Mb/s	$t_{tx,ack}$	$36\ \mu$s

Terms represented by capital letters in Eq. (1) denote random variables. Indeed, the behavior of the delays introduced by the AP and the PC is not known *a priori* and, in general, is not deterministic. Since deterministic terms in the general model of Eq. (1) can be either calculated or measured, as shown in [31], then Eq. (1) can be rewritten to account only for the delays introduced by the AP and OS, considering the random component of D'_{PC}:

$$D_{PC} = D_{AP} + D_{OS} \tag{2}$$

with $D_{PC} = D'_{PC} - t_{tx,eth} - t_{tx,802.11} - t_{DIFS}$.

Delay D_{OS} introduced by the OS kernel is unknown and, as noted, includes contributions associated both to OS latencies and to NICs. Moreover, it also depends on update of the system

software, kernel version, active services, *etc*. This is in fact the only term that can not be measured, neither directly nor indirectly. Its estimation is indeed the goal of the calibration procedure discussed in this Section.

Roughly speaking, to determine D_{OS} the measured AP delay D_{AP} should be subtracted from the corresponding total delay D_{PC}, measured at the application level. In practice, however, it is difficult to exactly associate a given measurement to the specific value of D_{PC} from which it should be subtracted. Consequently, these quantities are more conveniently described in statistical terms by means of their probability mass function (*pmf*).

Since the *pmf* of the sum of two random variables can be calculated through the convolution of their respective *pmf*s, the distribution of the random delay (2) through the PC is:

$$p_{D_{PC}}(z) = p_{D_{AP}} \otimes p_{D_{OS}}(z) = \sum_{k=0}^{z} p_{D_{AP}}(k) p_{D_{OS}}(z-k) \qquad (3)$$

where $\otimes$ denotes the convolution operator, and $p_{D_{PC}}$, $p_{D_{OS}}$ and $p_{D_{AP}}$ are the *pmf*s of the three discrete random variables.

The test setup described in this Section allows actual measurements of D_{PC} and D_{AP} to be taken. A measurement session can provide sample values for both D_{PC} and D_{AP}, that can be conveniently described through normalized histograms, representing the empirical *pmf*s $\overline{p_{D_{PC}}}$ and $\overline{p_{D_{AP}}}$ (denoted by an overline) of the two discrete random variables. The estimate of the *pmf* $p_{D_{OS}}$ (denoted by the wide caret symbol ˆ) can be obtained by deconvolution, that is equivalent to computing the cross-correlation of two discrete time signals:

$$\widehat{p_{D_{OS}}}(z) = \sum_{k=0}^{z} \overline{p_{D_{PC}}}(k) \overline{p_{D_{AP}}}(z+k). \qquad (4)$$

3.1. Direct measurement of D_{AP}

While the timestamping operation by the TSC provides D_{PC}, to accurately characterize the AP delay D_{AP} a direct measurement should be preferred. In this perspective, , as indicated in Fig. 5, we located on the AP internal printed circuit two test points providing hardware-level signaling of the arrival of a frame on the Ethernet port and of the start of radio transmission, respectively. Therefore, using the same system architecture and traffic profile, we was able to measure the delay between the two positive edges of these control signals, occurring at times t_3 and t_4, through the use of a high resolution digital sampling oscilloscope.

The whole measurement setup is sketched in Fig. 6. Each measurement session provides delay readings collected both through the PC and through the oscilloscope, delay samples being acquired directly by means of a LabView virtual instrument.
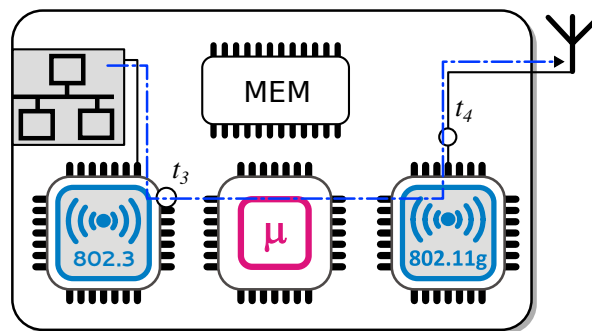
Figure 5. An illustration of the measurement points within the reference AP, used for measurements with the oscilloscope.
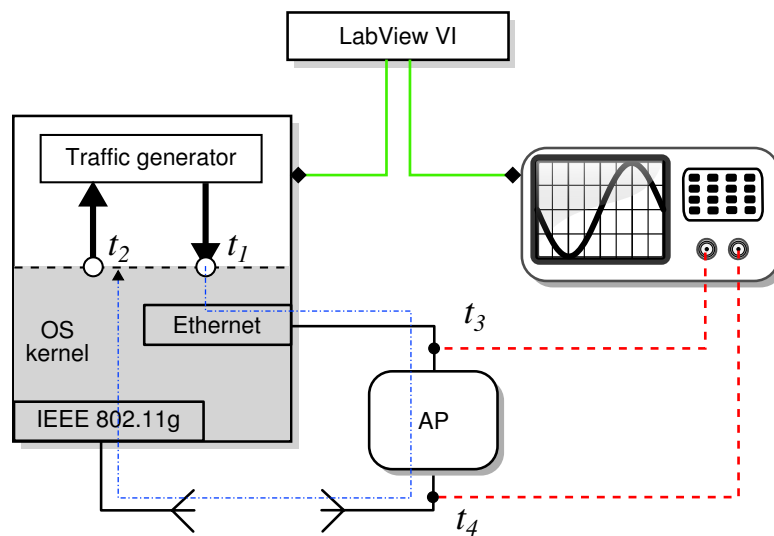
Figure 6. Test-set calibration setup.

4. Access point characterization

The experimental procedure involving oscilloscope measurements presents, however, some drawbacks that limit its application. As suggested by Fig. 5, direct measurement of AP delay requires "invasive" and device–dependent operations, which are little suitable for routine AP characterization. For example, it may not always be practical to look for suitable test points within the AP device under examination, nor are the needed pins available in every device, hence this is not an effective general route to estimate D_{AP}.

Instead, knowledge of $\widehat{p_{D_{OS}}}$ obtained through the use of an already characterized "reference" AP can allow to infer delay statistics of any new AP under test. It has been observed that (4) provides the same results with different APs, as long as the OS level is not changed. Experimentally, this has been seen to hold until low–level software layers (kernel, drivers, *etc*.) are not updated.

Therefore, once the test-set has been purposely characterized itself, by exploiting this assumption a simple yet effective way for estimating the delay introduced by any AP under test can be devised:

1. estimation of the *pmf* of the delay D_{OS} introduced by the operating system and the PC communication boards. This test-set calibration step can be effectively carried out through the use of a well–known reference AP with metrologically characterized delay.
2. a new set of delay values $D_{PC_{test}}$ is collected with the new AP under examination, whose delay is called $D_{AP_{test}}$ in the following, according to the scheme of Fig. 2. This step is relatively quick, simple to perform and non–invasive.
3. From the empirical *pmf* $\overline{p_{D_{PC_{test}}}}$ of these delays, and following the same passages the brought to Eq. (4), one has:

$$\widehat{p_{D_{AP_{test}}}}(z) = \sum_{k=0}^{z} \overline{p_{D_{PC_{test}}}}(k)\widehat{p_{D_{OS}}}(z + k) \tag{5}$$

which estimates the required probabilistic description of the forwarding latency time for the AP under investigation.

5. Experimental results

The current section presents the outcomes obtained through the adoption of the calibrated test–set described above, by describing a thorough measurement session on real devices.

The reference AP we adopted in our work is a general purpose off–the–shelf available one, namely a 3Com OfficeConnect 3CRWE454G75, which supports a 10/100 BASE-TX Ethernet connection and an IEEE 802.11g/e compliant infrastructured network, in which we enabled the Wireless Multimedia Extensions (WMM) option, that is a subset of the IEEE 802.11e specifications to support traffic profiles. In the presented experimental campaign also another AP is taken into account as a term of validation of the current procedure. Namely, it is a Siemens Scalance W784-1, which is an industry–grade device particularly designed for use in industrial WLANs, for which it may be expected that a lower forwarding delay is introduced, hence allowing to verify the validity of the methodology here proposed.

The test beds presented in Sec. 2 have been adopted to obtain samples of the delays introduced by this reference AP. Particularly, the PC exploited during tests was a Dell Optiplex 960 running an Ubuntu 16.04 LTS operating system, with an Atheros AR9287 IEEE 80211n compliant wireless NIC, and an Intel 82567LM-3 Ethernet NIC. Also, the high resolution digital sampling oscilloscope was an Agilent DSO 6032A. It is worth observing that in our experimental setup, we have not resorted to the use of a real–time OS. Actually, this choice has been carried out with a precise intent, i.e. to assess that the method is general and not related with the use of a particular or specific environment. Indeed, the use of a RT OS would have provided a better performance in terms of variance of the D_{OS} term (not necessarily in its magnitude), but in this work that was not the focus. Moreover, it has to be observed that in typical RT OS, the system is generally able to behave under real–time constraints if all the involved components, included the device drivers, support that feature (which means that they have been programmed having in mind what are the requirements of a real-time system, hence avoiding unbounded latencies, spin locks, etc.). Unfortunately, most of the device drivers of commercial WiFi cards are seldom able to respect real-time constraints. Hence, in this case, even if the OS in a RT one, the wireless communication section would have been carried out outside the real-time path of execution, leading to unbounded latencies to possibly arise.

Fig. 7 shows the empirical *pmfs* for the delays D_{PC} and D_{AP} measured with the setups of Fig. 2 and Fig. 5, respectively. The figure is relevant to tests with inter–departure time of packets set to the lowest value of t_{cycle} =3 ms. Given the high resolution of both TSC and oscilloscope raw data, in the post–processing phase we have chosen a histogram bin width of 1 μs.

We have actually carried out different experiments with other inter–departure times, as indicated by Table 2, and also with dynamically variable ones in the same range. The obtained outcomes are substantially analogous to the ones provided in Fig. 7, and hence the respective figures are not reported here to avoid clutter.

It is also worth highlighting that the choice of the values reported in Table 2 comes from the consideration that this work focuses on the measurement of the AP latency under non-saturated conditions, i.e., when the AP is not overwhelmed by packets coming from the Ethernet connections, so that the queue is not saturated and/or packets have not to be dropped by the AP. This simplifying assumption, which allows to avoid to take into account the model for queue management within the AP in Eq. (1), is actually appropriate for the field of industrial automation, since although the traffic profiles typical of that scenario are characterized by very stringent constraints on the timeliness, they are also characterized by a very low throughput, often allowing the networking devices (and APs in particular) to work under rather under-loaded conditions.

The rightmost curve is the estimated *pmf* of the delays measured through the software $\overline{p_{D_{PC}}}$, while the leftmost one, $\overline{p_{D_{AP}}}$, is that collected through oscilloscope measurements. Please note that we have already subtracted the deterministic contributions from the measured value D'_{PC}, actually showing D_{PC}. Looking at Fig. 7, a shift between the two *pmfs* of roughly 50μs can be observed, so that it is expected that the latency of the OS will result, from the application of the deconvolution method, in a *pmf* centered around that value.
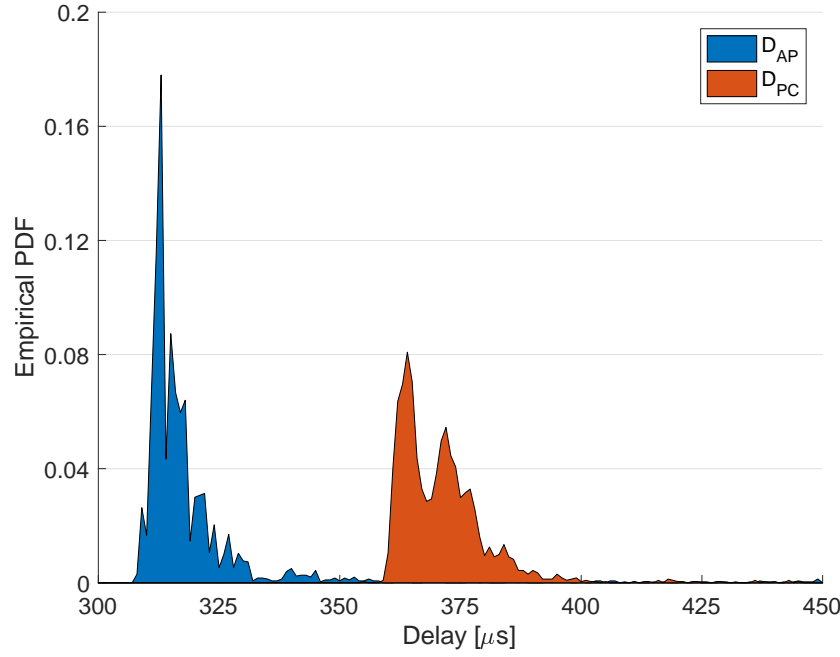
Figure 7. PDFs of the measured delays for the 3Com OfficeConnect AP (reference).

Indeed, we applied the proposed calibration scheme to our data, and the result of Eqs. (4)-(5) is shown in Fig. 8. Particularly, the leftmost PDF is the result of the application of Eq. (4), showing the estimation of $\widehat{p_{D_{OS}}}$. The expected behavior of D_{OS} is therefore confirmed by this curve, whose peak is find at 52 μs and which highlights that the operating system latencies together with those due to the network interfaces on the PC (Ethernet and Wireless boards) constitutes in fact a non negligible fraction of the overall delay measurement obtained at traffic generator level D_{PC}. Moreover, as it is trivially expected, D_{PC} changes from machine to machine, or if any of the adopted network cards is substituted, as it has been confirmed by other tests we performed, not shown here for the sake of brevity. Nonetheless, the setup here proposed showed to be able to provide accurate results independently of these changes, because of the use of a reference AP with metrologically characterized delay.

To provide an initial assessment that the procedure produced a meaningful estimate of *pfm*, in Fig. 8 we have also provided a comparison between the estimated and the measured *pmf* of the 3Com AP. The measured AP *pmf* is represented with the dashed thick blue line, while for the estimated *pmf* we used a solid gray area.

Nonetheless, it is worth to observe that the shape of the *pmf* $\widehat{p_{D_{OS}}}$ (solid orange area) is rather more smooth than that directly measured through the oscilloscope $\overline{p_{D_{AP}}}$ and TSC readings $\overline{p_{D_{PC}}}$, as can be noticed both in Fig. 7 and in the rightmost pmf in Fig. 8. Indeed, the different probability mass functions are represented through normalized histograms, where a specific and meaningful bin width has to be selected. This hence produces an implicit filtering operation on the acquired samples, so that we have:

$$\overline{p_{D_X}}(z) = p_{D_X} \otimes F(z) = \sum_{k=0}^{z} p_{D_X}(k)F(z-k) \tag{6}$$

where p_{D_X} is the theoretical *pmf* of the original acquired data (without filtering) and the filter $F(k)$ is defined as
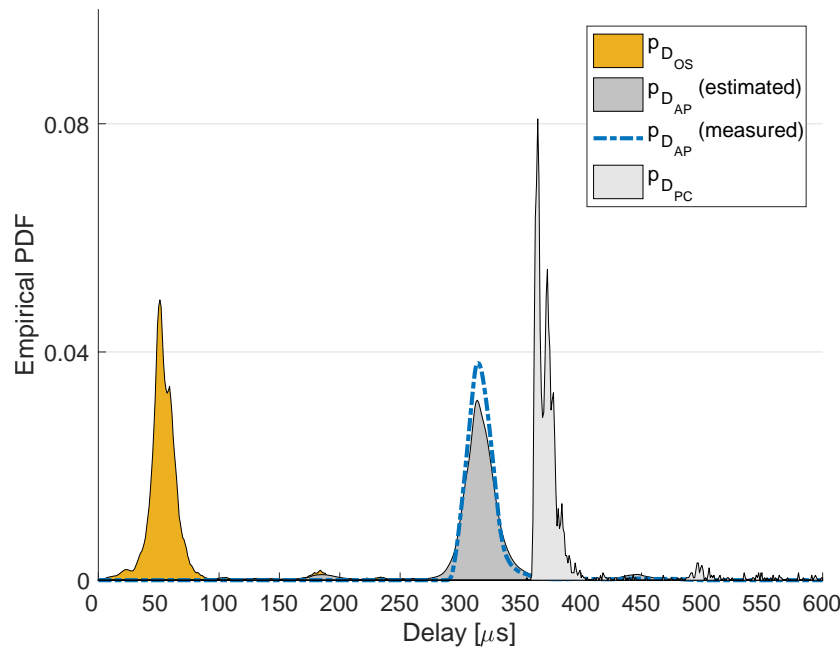
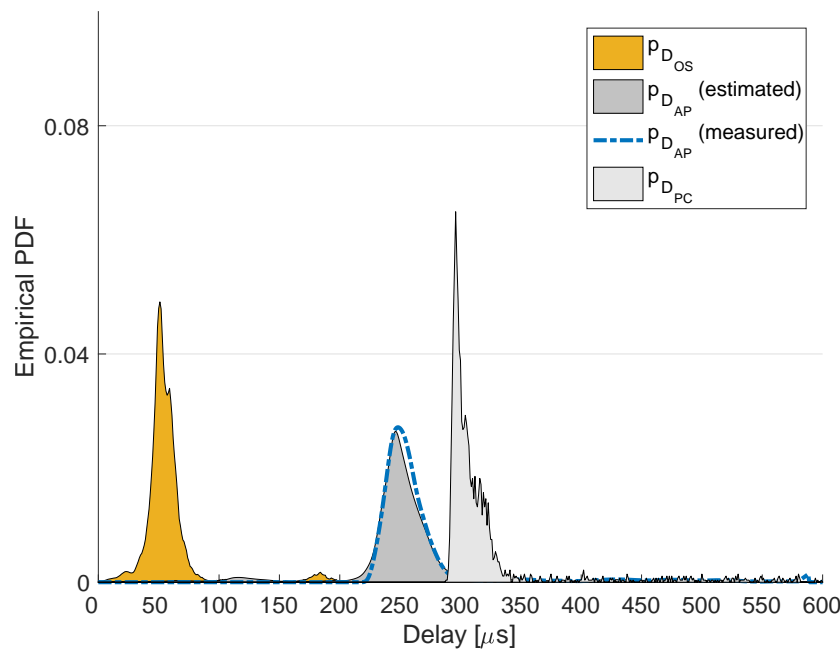**Figure 8.** The result of the estimation, and a comparison between the measured and estimated AP behaviour.

$$F(k) = \begin{cases} \dfrac{1}{n} & 0 \le k \le n-1 \\[2ex] 0 & \text{elsewhere} \end{cases} \tag{7}$$

with n being the length of the filter. Moreover, the subsequent application of (de)convolution or, similarly, cross–correlation operations, as in Eq. (4) and Eq. (5), to obtain the needed estimates proposed in this work further exacerbate this filtering operation.

Therefore, returning to look at Fig. 8, we have to consider that both measurement procedures (oscilloscope and TSC) returns data with a granularity in the order of tens of nanoseconds and that we built the *pfms* using bins that are 1 μs large. Hence, to effectively compare the measured $\overline{p_{D_{AP}}}$ with the estimated one, we have to apply to the former the same number of filtering operations involved in the estimation of the latter. We actually already carried out this post–processing in Fig. 8, and as such we may definitely state the there is a very strong agreement between the estimated AP forwarding latency and the measured one. Furthermore, it is possible to give a rough estimation of the goodness of fit, adopting the normalized root mean square (RMS) difference between the *pmfs* as a performance indicator of the method. To this regard, a value well below 1% is obtained, providing a raw figure of merit of the whole measurement procedure accuracy.

To further assess the performance of the proposed approach, we have applied the method described to another AP that were previously characterized both according to the proposed black-box estimation method (through the set–up of Fig. 2), and according to the same direct measurement method used to characterize the reference AP, *i.e.* through direct measurements with a high–resolution digital sampling oscilloscope. Namely, the new access point is a Siemens Scalance W784-1, which is an industry–grade device particularly designed within industrial WLANs, for which it may be expected that a lower forwarding delay is introduced.

The results of the application of the same characterization procedure described in this work to this new AP are summarized in Fig. 9. In the figure, the leftmost solid orange area is the same estimated $\widehat{p_{D_{OS}}}$, as determined before as the result of the calibration of the measurement test–set. The rightmost

Figure 9. The result of the estimation, and a comparison between the measured and estimated AP behaviour.

solid gray area is the *pmf* relevant to the delay D_{PC_S}, *i.e.* measured at the OS level with the loopback setup of Fig. 2 and with the new AP as the device under test. The central solid dark gray area is the final result of the estimation of the delay D_{AP_S} introduced by the new AP, and the comparison with the measured delay (dashed thick blue line) confirms again the goodness of the estimation procedure.

As a significant observation, we may state that, as expected, this industry–grade AP introduces a nearly 60 μs lower forwarding latency than the one introduced by the reference AP, which is a commercial off–the–shelf office device. Nonetheless, we may also observe that, despite the Scalance W784-1 AP is realized for the management of time–critical industrial traffic, its forwarding latency is nevertheless considerable (about 250 μs) and associated to a non–negligible randomness, which has to be taken into account for the deployment of real–time communication tasks in the system.

Nonetheless, it is worth highlighting that, given that the focus of this paper is on the measurement system rather than on the actual delay results, it is out of our intentions to provide a performance comparison between one AP and another one, also given that the two considered APs belong to very different device classes. Conversely, these final figures and experiments are intended only to demonstrate the validity and accuracy of the measurement method in the given application context.

Finally, it is worth drawing some considerations about the measurement uncertainty associated to the proposed estimation procedure. Several factors contribute to the overall expanded uncertainty, the most significant being the measurement procedure for the metrological characterization of the reference AP through the oscilloscope, the software time stamping operations, the post–processing of data to obtain the empirical *pmfs*. As far as the time stamping is concerned, the TSC has a resolution equal to the internal CPU frequency multiplier, which is in the range 10-20. Since the clock frequency on the machine used in these experiments is 2.66 GHz, the resolution may be considered roughly 10 ns. However, considering the different sources of uncertainty cited in Sec. 2.1, and some test carried out to assess the overhead involved in TSC reading operation, it may result safe to overestimate the accuracy related to the software time stamp and set it to one order of magnitude greater, *i.e.* around 100 ns.

The resolution of the oscilloscope is in the order of the 2 ns, but for the kind of measurement (a time difference) and looking at the instrument data sheet, the accuracy result to be about 50 ns.

The histogram bin width in the Figg. 7, 8 and 9 has been set to 1 μs. This means that, considering the aforementioned uncertainties, there is quite a little chance that samples at the boundary between two bins are misplaced. Hence, it may be safe to consider that the accuracy involved in the post–processing of data is not significantly degraded during these operations, and may be set to 2 μs at the worst case. Filtering effects due to consecutive applications of Eqs. (4) and (5) tend to further decrease this accuracy. However, since the final estimation of the AP forwarding delay $D_{AP_{test}}$ requires actually two steps of (de)convolution, we may assume that the overall uncertainty associated to the proposed estimation procedure is in the order of 5 μs.

6. Conclusions

This paper proposed a calibrated test–set for the characterization of the forwarding delay introduced by an Access Point, used to forwards packets from/to a wired real–time industrial Ethernet segment to/from a wireless IEEE 802.11 one. The proposed approach exploits the availability of a reference AP, which is at first metrologically characterized through the use of a high–resolution digital sampling oscilloscope, and the application of a deconvolution algorithm to obtain the characterization of the measurement system.

Provided that a reference AP has been made available, once for all, for the initial calibration step, the method reveals to be easy to perform, fast, accurate and does not require expensive dedicated instrumentation, allowing to characterize the forwarding delay introduced by any other AP under test in a non–invasive procedure.

The knowledge of the delay introduced by these kind of devices is a very significant topic in the design of wireless communication networks for factory automation systems, where deterministic timings are often required. In the case of an AP used to extend a wired network with a wireless segment, the randomness it introduces in delivering packets must be included in the calculation of the inter–polling time between two consecutive nodes, but such relevant parameter is usually neither predictable nor specified by manufacturers. In a deterministic, soft real–time context, such as that of industrial automation, this forwarding delay has to be carefully considered in order to maximize the chance that the system is able to respect deadlines.

As a future development of this work, further experiments have to be carried out in order to further characterize the method with different APs, potentially supporting more IEEE 802.11 recent features. Moreover, tests in different channel and environment conditions have to be carried out, in order to assess the methodology and also obtain a latency characterization when the device is deployed in real industrial and automation environments.

Author Contributions: F. Tramarin conceived and performed the experiments, and analyzed the data; C. Narduzzi and S. Vitturi contributed to the analysis of data and to the refinement of the methodology; these authors equally contributed to write the paper. M. Bertocco was involved in the first version of this work, and revised the final version of the paper.

Conflicts of Interest: The authors declare no conflict of interest. Moreover, the founding sponsors had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, and in the decision to publish the results.

References

1.   Zurawski, R., Ed. *Industrial Communication Technology Handbook, 2nd Edition*; CRC Press, 2014.
2.   Wollschlaeger, M.; Sauter, T.; Jasperneite, J. The Future of Industrial Communication: Automation Networks in the Era of the Internet of Things and Industry 4.0. *IEEE Industrial Electronics Magazine* **2017**, *11*, 17–27.

3.   International Electrotechnical Commission. *IEC 61158-5: Digital data communications for measurement and control - Fieldbus for use in industrial control systems - parts 5 and 6: Application Layer service definition and protocol specification, communication model type 3*, 2000.

4.   CAN In Automation, International Users and Manufacturers Group e.V. *CANopen Application Layer and Communication Profile, CiA/DS 301, Version 4.01*, 2000.

5.   European Committee for Electrotechnical Standardization,. *EN 50325-2: Industrial communication subsystem based on ISO 11898 (CAN) for controller-device interface -Part 2: DeviceNet*, 2000.

6.   International Electrotechnical Commission. *IEC 61784: Digital data communications for measurement and control – Part 2: Additional profiles for ISO/IEC 8802–3 based communication networks in real–time applications*, 2007.

7.   Decotignie, J.D. The Many Faces of Industrial Ethernet [Past and Present]. *IEEE Industrial Electronic Magazine* **2009**, *3*, 8–19.

8.   HART Communication Foundation. http://www.hartcomm.org/.

9.   International Society of Automation. *ISA-100.11a-2011 Wireless systems for industrial automation: Process control and related applications*, 2011.

10.   Willig, A. Recent and Emerging Topics in Wireless Industrial Communications: A Selection. *IEEE Trans. on Industrial Informatics* **2008**, *4*, 102–124.

11.   Ferrari, P.; Flammini, A.; Marioli, D.; Taroni, A. IEEE802.11 sensor networking. *Instrumentation and Measurement, IEEE Transactions on* **2006**, *55*, 615 – 619.

12.   Bertocco, M.; Gamba, G.; Sona, A.; Tramarin, F. Investigating wireless networks coexistence issues through an interference aware simulator. Proc. IEEE Int. Conf. Emerging Technologies and Factory Automation ETFA 2008, 2008, pp. 1153–1156.

13.   Cena, G.; Valenzano, A.; Vitturi, S. Hybrid Wired/Wireless Networks for Real-Time Industrial Communications. *IEEE Industrial Electronic Magazine* **2008**, *2*, 8–20.

14.   Vitturi, S.; Tramarin, F.; Seno, L. Industrial Wireless Networks: The Significance of Timeliness in Communication Systems. *Industrial Electronics Magazine, IEEE* **2013**, *7*, 40–51.

15.   Seno, L.; Tramarin, F.; Vitturi, S. Influence of Real Components Behavior on the Performance of Wireless Industrial Communication Systems. Proc. of the 20th IEEE ISIE; , 2011.

16.   Bertocco, M.; Narduzzi, C.; Tramarin, F. Estimation of the delay of network devices in hybrid wired/wireless real-time industrial communication systems. 2012 IEEE International Instrumentation and Measurement Technology Conference Proceedings, 2012, pp. 2016–2021.

17.   Seno, L.; Tramarin, F.; Vitturi, S. Experimental Evaluation of the Service Time for Industrial Hybrid (Wired/Wireless) Networks under Non-Ideal Environmental Conditions. Proc. of IEEE Conf. Emerging Technologies & Factory Automation ETFA 2011; , 2011.

18.   Endace DAG Packet Capture Cards. https://www.endace.com.

19.   Intel®. *Intel® 64 and IA-32 Architectures Software Developer Manuals*, 2016.

20.   Tian, G.S.; Tian, Y.C.; Fidge, C. High-Precision Relative Clock Synchronization Using Time Stamp Counters. 13th IEEE International Conference on Engineering of Complex Computer Systems (iceccs 2008), 2008, pp. 69–78.

21.   Pásztor, A.; Veitch, D. PC Based Precision Timing Without GPS. *SIGMETRICS Perform. Eval. Rev.* **2002**, *30*, 1–10.

22.   Veitch, D.; Ridoux, J.; Korada, S.B. Robust Synchronization of Absolute and Difference Clocks Over Networks. *IEEE/ACM Transactions on Networking* **2009**, *17*, 417–430.

23.   Ridoux, J.; Veitch, D. Ten Microseconds Over LAN, for Free (Extended). *IEEE Transactions on Instrumentation and Measurement* **2009**, *58*, 1841–1848.

24.   Ethernet POWERLINK Standardization Group. *Ethernet POWERLINK Communication Profile Specification V. 2.0*, 2003.

25.   L., L.B. Novel Trends in Automotive Networks: A perspective on Ethernet and the IEEE Audio Video Bridging. Emerging Technologies Factory Automation (ETFA), 2014 IEEE 19th Conference on, 2014, pp. 1–8.

26.   Wei, Y.H.; Leng, Q.; Han, S.; Mok, A.; Zhang, W.; Tomizuka, M. RT-WiFi: Real-Time High-Speed Communication Protocol for Wireless Cyber-Physical Control Applications. Real-Time Systems Symposium (RTSS), 2013 IEEE 34th, 2013, pp. 140–149.

27. Reimann, F.; Graf, S.; Streit, F.; Glas, M.; Teich, J. Timing Analysis of Ethernet AVB-based Automotive E/E Architectures. Emerging Technologies Factory Automation (ETFA), 2013 IEEE 18th Conference on, 2013, pp. 1–8.
28. Bertocco, M.; Gamba, G.; Sona, A. Is CSMA/CA Really Efficient Against Interference in a Wireless Control System? An Experimental Answer. Proc. of IEEE ETFA; , 2008.
29. Siemens AG. *Industrial Wireless LAN: Industrial Features and Current Standards, White paper* 2009.
30. *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*, 2007.
31. Bertocco, M.; Tramarin, F. A cross-layer simulator for industrial wireless communication systems. Proc. IEEE International Workshop on Measurement&Networking (M&N2011), 2011.
32. IEEE. *IEEE 802.3 standard: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications*, 2000.