

w-SHARP: Implementation of a High-Performance Wireless Time-Sensitive Network for Low Latency and Ultra-Low Cycle Time Industrial Applications

Abstract— Real-time industrial applications in the scope of the industry 4.0. present significant challenges from the communication perspective: low latency, ultra-reliability, and determinism. Given that wireless networks provide a significant cost reduction, lower deployment time, and free movement of the wireless nodes, wireless solutions have attracted the industry attention. However, industrial networks are mostly built by wired means because state-of-the-art wireless networks cannot cope with the industrial applications requirements. In this paper, we present the hardware implementation of wireless SHARP (w-SHARP), a promising wireless technology for real-time industrial applications. w-SHARP follows the principles of Time-Sensitive Networking and provides time synchronization, time-aware scheduling with bounded latency, and high reliability. The implementation has been carried out on a Field Programmable Gate Array-based Software Defined Radio platform. We demonstrate, through a hardware testbed, that w-SHARP is able to provide ultra-low control cycles, low latency, and high reliability. This implementation may open new perspectives in the implementation of high-performance industrial wireless networks, as both PHY and MAC layers are now subject to be optimized for specific industrial applications.

Key Terms—w-SHARP, industry 4.0., factory automation, wireless communication, wireless TSN, 5G, URLLC, IEEE 802.11.

I. INTRODUCTION

The industry 4.0. envisions industry facilities crowded with a huge number of heterogeneous wireless interconnected devices, which can be automatically controlled, upgraded and configured through a ubiquitous network. This network revolution would drastically cut the maintenance, installation, and operation costs of industry facilities thanks to the replacement of wired connections with wireless connectivity. Nonetheless, a major challenge arises when building such an industrial network, the wireless/wired technologies used to build the network must support the heterogeneous and stringent requirements of industrial applications: high reliability (Packet Error Rate [PER] $< 10^{-7}$) [1], sub-millisecond control cycles, time-aware operation, handling from hundreds to thousands of nodes, and highly-secured information exchange [2].

Most of these challenges are solved in the wired domain thanks to the Time-Sensitive Networking (TSN) technology defined by the IEEE 802.1 TSN working group [3]. However, the harshness of industrial propagation environments (channel variation with strong multipath, interference), and the issues of wireless systems, such as lack of determinism and their inefficiencies for the transmission of short packets difficults the deployment of wireless industrial networks that comply TSN requirements [4].

Some technologies are aiming at providing TSN-like capabilities over wireless (wireless TSN). For instance, 5G-ACIA is promoting 5G New Radio (NR) for factory automation, as it is expected that 5G-NR will fulfill the requirements of most industrial use cases with the Ultra-Reliable and Low Latency

Communications (URLLC) profile. Moreover, the Avnu Alliance is promoting 5G and also 802.11 (a natural extension to Ethernet) as suitable candidates for wireless TSN [5].

Regarding the use of 5G-NR in industrial applications, some real measurements and simulations have demonstrated that 5G-NR is able to provide a worst-case latency below 1 ms [6] and a cycle time of 2-3 ms for specific configurations [7]. In addition, significant efforts have been doing in the 5G standardization process to integrate 5G and TSN (3GPP Release 16) [8]. Still, several industrial applications are out of the performance provided by URLLC in terms of latency and cycle time. For instance, some motion control [9] or power electronics [10] applications require sub-millisecond cycle time.

With respect to legacy 802.11 (any version before 802.11ax), it is worth mentioning that its lack of determinism makes difficult to comply with the time-critical requirements of industrial applications. In that sense, meaningful research has been done around how to modify legacy 802.11 towards a TSN 802.11. The efforts have been mainly focused on modifying the MAC layer to provide time-aware scheduling [11] and use Commercial-Off-The-Shelf (COTS) devices to validate the MAC design in hardware testbeds [12]. For instance, RT-WiFi [13], IsoMAC [14], or Priority MAC [15] use the 802.11 PHY along with a Time Division Multiple Access (TDMA) MAC layer. However, these solutions use the 802.11 PHY that is not very efficient for short packets, thus their cycle times and latencies are in the order of milliseconds [16].

In the last few years, significant standardization efforts have been done over 802.11 to improve its efficiency. For instance, 802.11ax has a significant efficiency improvement in scenarios with a large number of nodes [17]. In addition, the introduction of the trigger frame and OFDMA combined with a contention-based MAC scheme give the opportunity to perform non-persistent scheduling of frames that allows to build soft Real-Time (RT) applications [4]. Still, 802.11ax is not suitable for time-critical applications. In addition, the next 802.11 standard, 802.11be [18], will contain some enhancements to the reduce the worst case latency and jitter and will provide integration with TSN.

Outside of the official 802.11 standardization process, the wireless High-Performance (WirelessHP) project [19] aims at the design and implementation of a high-performance PHY layer for industrial applications. In theory, WirelessHP is able to provide extremely low cycle time ($< 100 \mu\text{s}$) [20].

The Synchronous and Hybrid Architecture for RT Performance (SHARP) [21] is a hybrid network composed of Ethernet TSN and a wireless counterpart named wireless SHARP (w-SHARP). w-SHARP is a custom wireless system that enhances 802.11 PHY and MAC layers to provide high efficiency, low latency and high reliability, which are some of the required features of TSN. Simulation results [21] show that w-SHARP guarantees sub-millisecond cycle times and high-

reliability ($PER < 10^{-7}$). As WirelessHP [20], w-SHARP specifically targets the strict traffic profile of time industrial applications [1]. This traffic is characterized by small portions of data that are periodically and synchronously generated (sensors), and that must be processed and arrive to its destination (actuators) before a delay bound. Finally, as its main drawback, we may highlight that w-SHARP has little flexibility at runtime, because w-SHARP is designed to have a very stable performance in order to tame the strict industrial traffic profile.

In this paper, we improve the w-SHARP specification to effectively boost its throughput and efficiency using Orthogonal Frequency Division Multiple Access (OFDMA) and the optimization of the InterFrame Spacing (IFS). We compare the performance of 802.11ax and w-SHARP using OFDMA by simulation means, showing that w-SHARP greatly outperforms 802.11ax for sub-millisecond control cycles. Based on these promising results and using an advanced implementation methodology, we have implemented the first prototype of the w-SHARP technology over a Field Programmable Gate Array (FPGA) based Software Defined Radio (SDR) platform. The prototype is constrained to OFDM and 20 MHz BW due to the limited hardware resources of the FPGA-based SDR platform. With the prototype, we have built a w-SHARP network and we have performed several experiments in the laboratory and in a real industrial environment, where we have demonstrated that w-SHARP is able to provide low latency, ultra-low cycle time and high reliability.

The implementation methodology is the key factor to drastically cut down the time taken to implement the system. This methodology may open a new perspective in the implementation of wireless systems for industrial applications, as the whole communication stack including PHY is now subject to be optimized for specific requirements. For instance, Open Air Interface (OAI) 5G-NR is implemented on software-based SDR, its latency is in the order of milliseconds and hence it does not support the URLLC profile. Meanwhile, w-SHARP could be integrated into OAI to provide low latency operation.

The rest of the paper is organized as follows. First, an overview of the w-SHARP design is detailed in Section II. In Section III, 802.11 and w-SHARP performances are compared. In Section IV, the implementation methodology is described. The proposed implementation of a w-SHARP node over an FPGA is shown in Section V. The validation and performance results of the system in a hardware testbed are presented in Section VI. Finally, Section VII summarizes some conclusions of the work.

II. w-SHARP PHY AND MAC DESIGN

In this section, we present a brief review of the w-SHARP PHY and MAC design [21], and we introduce two significant upgrades to improve w-SHARP efficiency and throughput: the IFS optimization and OFDMA. w-SHARP does not strictly comply with the TSN specifications, though it follows the TSN principles. For instance, w-SHARP incorporates an efficient PHY layer, a MAC layer with time-aware scheduling support (similar to 802.1Qbv [22]), and delivers an absolute notion of time through a synchronization protocol (similar to 802.1as [23]).

A. Low Latency and efficient PHY design

The w-SHARP PHY has been co-designed with its MAC layer based on the typical industrial traffic profile (short packets with bounded low latency). w-SHARP follows a star topology with one Access Point (AP) and several Stations (STAs) connected to the AP. In addition, it uses two waveforms based on the 802.11 Orthogonal Frequency Division Multiplexing (OFDM) PHY: the DownLink (DL) waveform (frames from the AP to the STAs) and the UpLink (UL) waveform (frames from the STAs to the AP).

The DL waveform uses the 802.11 legacy preamble. Its duration is $16 \mu\text{s}$ and it has two fields: The Short Training Field (STF), and the Long Training Field (LTF). This preamble structure is quite large and hence it is not very efficient for communications of short packets. To provide high efficiency, the RT data frames transmitted by the AP to different STAs are joined into a single frame using frame aggregation. In w-SHARP, it is defined the use of frame aggregation at the PHY layer [21]. Several subframes with different lengths and different Modulation and Coding Schemes (MCS) are prepended with only one preamble in frame aggregation (see Fig. 1 (a)). The minimum and maximum lengths allowed for a UL PHY frame are 4 bytes and 2500 bytes, respectively. To provide a latency according to the requirements of each subframe, the aggregation in DL frames can be performed by subframe criticality level: the most critical subframes are placed before the least critical ones to ensure a lower latency of the former ones.

In the UL, the STAs send dedicated frames to the AP and hence frame aggregation is not feasible. To provide high efficiency, the UL frames use a small preamble with only one OFDM symbol of $4 \mu\text{s}$ (Fig. 1 (b)). The use of a small preamble makes impractical the use of automatic gain control and carrier frequency correction in the AP side, and then, these operations are moved from the Reception (Rx) chain of the AP to the Transmission (Tx) chain of the STA. Hence, the AP just keeps a fixed Rx gain and carrier frequency, whereas the STAs adjust their Tx power and carrier frequency based on the channel information retrieved from the DL frames transmitted by the AP. As in the DL PHY frames, the minimum and maximum lengths allowed for a UL PHY frame are 4 bytes and 2500 bytes.

w-SHARP was initially designed with a 20 MHz BW [21]. However, this BW may not be enough to support sub-millisecond cycle times and tens of nodes. Greater BW can significantly increase the system throughput, as the number of carriers inside each OFDM symbol typically grows

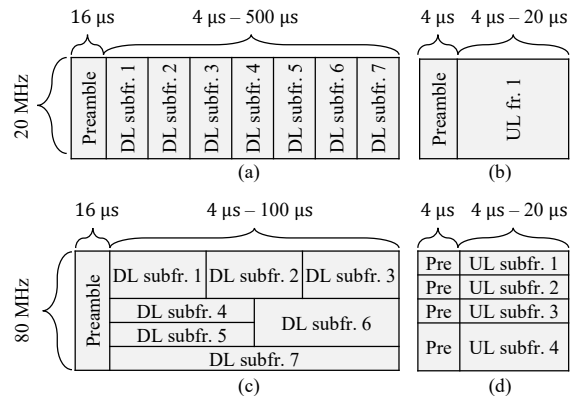


Fig. 1. DL (a) and UL (b) w-SHARP Frame with BW = 20 MHz, and DL (c) and UL (d) w-SHARP OFDMA Frame with BW = 80 MHz.

TABLE I. 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

proportionally with the BW. However, using a higher BW may not reduce the frame duration of short packets, because the extra OFDM data carriers may remain unused [16].

As a forward consideration to increase the performance of w-SHARP, OFDMA has been introduced in the w-SHARP specification. With OFDMA, the number of effective data carriers for each frame can be controlled and then the efficiency for higher BWs is maintained. We have defined four possible BW and OFDMA configurations (see Table I). OFDMA is only used with 40 MHz and above. A w-SHARP AP with OFDMA can split the STAs into multiple OFDMA channels increasing the overall w-SHARP network throughput. As can be seen, the OFDMA design is equivalent to have several w-SHARP AP operating at adjacent bands. An example of a DL frame and a UL frame using OFDMA is depicted in Fig. 1 (c) and Fig. 1 (d). In order to ease the STAs hardware requirements and its implementation, the STAs are only allowed to transmit one subframe within an OFDMA frame. In the case that an STA requires the transmission of two or more packets, the packets may be aggregated at the link layer. Thus, the UL PHY frame will carry the data from both packets. The performance of w-SHARP using OFDMA has been evaluated through simulation means (see Section III), but its implementation in the prototype is a further challenge that will be addressed in future works.

B. Time-aware Flexible MAC design

w-SHARP uses a Time Division Multiple Access (TDMA) scheme combined with a persistent scheduler, which is divided in periodic superframes of duration (T_S). The superframe always starts with the transmission of one beacon frame that carries the superframe design and its transmission timestamp. The beacons are used to synchronize the STAs to a global notion of time. The w-SHARP MAC scheme provides high-efficiency and time-aware scheduling enabling 802.1Qbv [22] operation, despite not being compliant with 802.1Qbv at the moment. The MAC provides two traffic priorities (RT and non-RT), adaptability to the wireless medium, and fast retransmissions. To do so, the w-SHARP MAC has different periods inside each superframe: DL Period, DL Retransmission (DL RTx) period, UL Period, UpLink Retransmission (UL RTx) period and 802.11-like (STD). These periods can be placed in arbitrary order and number within the superframe.

The DL period is used to transmit DL frames from the AP to the STAs and the DL RTx is used to perform fast retransmissions, in case that the DL frame was not correctly received. The UL and UL RTx follow an operation similar to the DL periods, though they are defined for UL. The 802.11-like period defines the use of a random-access scheme based on the 802.11 standard, and it is used to transmit non-RT frames. w-SHARP uses the 802.11 Channel Clear Assessment in the STD period, whereas it directly transmits in the RT periods assuming the channel is idle. This assumption is valid in industrial environments over managed conditions. The AP can modify the time-aware scheduler at running time using the RTx or the STD period. For instance, the w-SHARP AP can add or

remove slots, change the duration of the periods, or adapt the MCS of an RT frame to improve its robustness. On the downside, the actual specification of w-SHARP does not allow the modification of the superframe duration at running time. Thus, a w-SHARP network with a specific superframe periodicity cannot support nodes with other data periodicities.

Regarding the acknowledgment frames (ACK), dedicated frames consume a considerable portion of time in applications with short packets, because their length is comparable to the data packets. Hence, a suitable option for these applications is to piggyback the ACKs on other data frames. This is the scheme followed in w-SHARP.

The w-SHARP MAC has been demonstrated to be flexible, efficient, and to provide low-latency [21], though the optimization of some parameters, such as the IFS, has not been fully studied. The IFS is the time gap between two adjacent frames. It is used to avoid collisions and must be minimized when targeting ultra-low cycle times in the sub-millisecond range. Unfortunately, it cannot be arbitrarily small, and it depends on some RF and PHY parameters and on the wireless channel properties. We have separately studied how to minimize the IFS between UL frames (IFS_{UL-UL}) and the IFS between DL-UL frames or UL-DL frames (IFS_{UL-DL}). The IFS between DL frames is not considered because the DL frames are always sent by the AP. The IFS is the sum of a few contributions. First, the 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 may be in the range of 50-150 ns, depending on the wireless channel and the synchronization algorithm [24]. 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 IFS_{UL-DL} . T_{SRF} for a common radio chip can be in the order of 2-3 μ s [25]. Then, the minimum IFS can be expressed as

$$\begin{aligned} IFS_{UL-DL} &= T_{SRF} + 2 \cdot T_h + T_e. \\ IFS_{UL-UL} &= 2 \cdot T_h + T_e. \end{aligned} \quad (1)$$

Considering a common industrial network with a maximum distance between the AP and STAs of 30 m, and the high-performance time synchronization algorithm of [24], appropriate values for these parameters could be $T_{SRF} = 3 \mu$ s, $T_e = 200$ ns, and $\tau_h = 100$ ns, which results in $IFS_{UL-DL} = 3.5 \mu$ s and $IFS_{UL-UL} = 0.5 \mu$ s. IFS_{UL-DL} and IFS_{UL-UL} have been calculated for this specific scenario, but they could be adapted to match the requirements of larger scenarios. In addition, it should be noted that the proposed IFS take its minimum possible values, but larger values of the IFS may be used upon the network traffic.

C. Time Synchronization

w-SHARP uses a time synchronization mechanism based on the transmission of beacons. At the start of the superframes, the w-SHARP AP transmits a beacon frame that contains basic scheduling information (superframe size, periods, etc.), and the beacon Tx timestamp t_1 . The STAs use two phases to synchronize their operation with the AP. In the first phase, the STA searches the beacons. The STA takes the Rx timestamp t_2 when it detects the beacon frame. Then, the STA estimates the time offset by

$$\tilde{t}_o = t_2 - t_1. \quad (2)$$

\tilde{t}_o is introduced to a Proportional Integral (PI) filter to reduce its variance, which outputs \tilde{t}_e . Then, the STA corrects its time with \tilde{t}_e . Once that the STA is synchronized, the STA operation changes to TDMA. In TDMA mode, the STA is still listening to beacons and performing the time synchronization, but it can also access the w-SHARP network.

w-SHARP time synchronization is based on the scheme proposed in [24], though w-SHARP does not consider channel delay compensation and, thus, adds a systematic error equal to AP-STAs delay, preserving the precision of [24].

III. CYCLE TIME COMPARISON BETWEEN EXISTING WIRELESS STANDARDS AND w-SHARP

In this section, we compare the attainable cycle time of existing wireless standards and w-SHARP for RT industrial applications. There are few wireless systems that are indeed able to fulfill the targeted sub-millisecond cycle time. For instance, 802.11g/n/ac have low efficiency in the transmission of short packets due to its long preamble [16], 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 [7], which is above the sub-millisecond cycle time pursued. 802.11be [18] is quite promising, but its standardization process has just

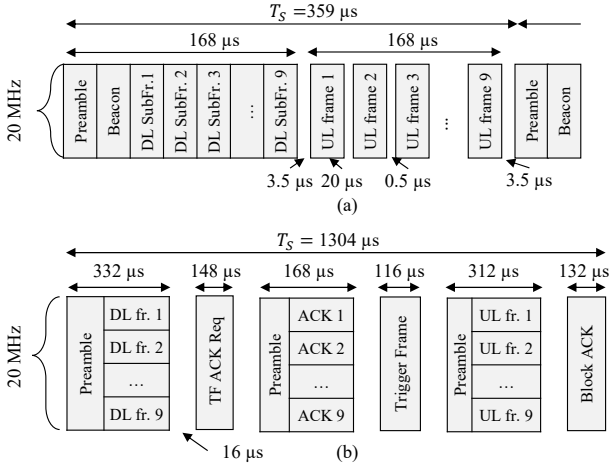


Fig. 2. w-SHARP superframe structure (a), and 802.11ax PHY + TDMA MAC superframe structure (b).

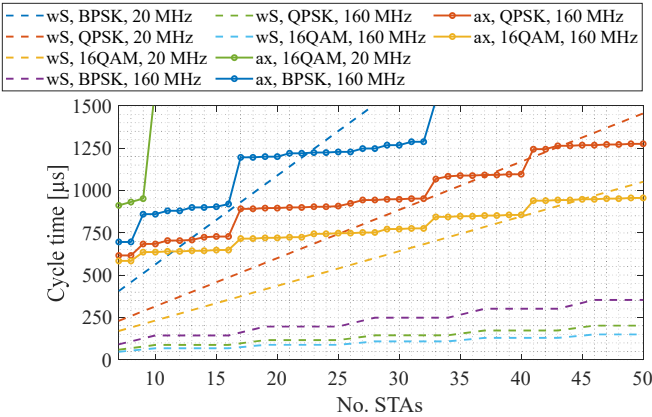


Fig. 3. Minimum achievable cycle time vs. the No. STAs of 802.11ax PHY + TDMA (ax) and w-SHARP (wS) for different modulations and BWs and for a payload size of 13 bytes.

started. Then, we have considered for this comparison w-SHARP and 802.11ax. 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 [13]. 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 .

A w-SHARP superframe and an 802.11ax PHY + TDMA MAC superframe are plotted in Fig. 2. The configuration comprises 1 AP and 9 STAs. The BW has been set to 20 MHz and The MCS to QPSK with channel coding of 1/2. The payload of the frames has been set to 13 bytes, which is in the lower part of the payload range of common industrial applications that involve closed control loops [1]. At the start of each cycle, a beacon frame with a payload of 8 bytes is transmitted. We have used a MAC overhead of 5 bytes for w-SHARP. According to this configuration, the w-SHARP superframe has a duration of $T_s = 359 \mu s$, which is rather short and quite efficient thanks to its low overhead. On the contrary, 802.11ax provides $T_s = 1304 \mu s$, which is 3.6 times the w-SHARP superframe duration.

To gain more insights of the performance of w-SHARP, Fig. 3 and Fig. 5 plots the minimum attainable cycle time, both for w-SHARP and 802.11ax, 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 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.

IV. IMPLEMENTATION METHODOLOGY

The adoption of a high-performance wireless solution for industrial applications requires experimental validation in an

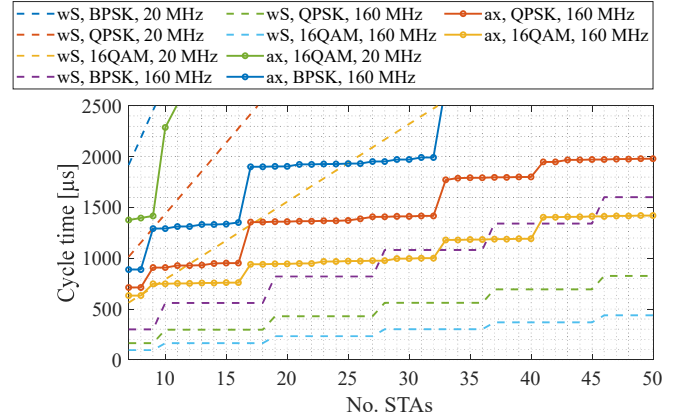


Fig. 4. Minimum achievable cycle time vs. the No. STAs of 802.11ax PHY + TDMA (ax) and w-SHARP (wS) for different modulations and BWs and for a payload size of 100 bytes.

industrial scenario. Since w-SHARP has a specific PHY/MAC design, its implementation has been carried out using an SDR platform. On the one hand, software-based SDR platforms, such as Universal Software Radio Peripheral (USRP), are an adequate solution for early PHY validation. However, their latency is in the order of several milliseconds [26], thus they only serve as a proof of concept of a low latency system. On the other hand, the performance of FPGA-based SDR platforms is close to that of wireless integrated circuits, though it entails great implementation complexity because the system is entirely programmed in the FPGA. In fact, there exist very few FPGA-based SDR implementations of wireless systems in the state-of-the-art. For instance, WARP [27] (802.11b/g/n), Tick [28], or an 802.11ax modem developed over an Intel Arria [4]. Based on the FPGA-based SDR performance, we decided to build a w-SHARP node in an FPGA-based SDR platform.

The implementation methodology and tools are of utmost importance to successfully build a complex hardware. Hardware designs are commonly programmed using low-level Hardware Description Languages (HDL) (i.e., VHDL or Verilog). However, the implementation effort of HDL is significant and thus it is unfeasible to build complex custom solutions or to do fast prototyping. Now, High-Level Synthesis (HLS) tools are standing as a compelling alternative to HDL programming, as they provide an abstraction layer that eases the modeling, simulation, and verification of complex hardware. In addition, nowadays System-on-Chip (SoC) FPGAs integrate in a single circuit FPGA logic and several ARM microcontrollers. HLS tools combined with SoC FPGAs yields the ideal scope to build complex hardware designs.

HLS tools provide several basic hardware models (register, memories, logic doors, etc.) that can be configured and interconnected to create a complex model. The model can be seamlessly simulated and verified through the utilities of the HLS tool. Once the model is verified, the tool translates the

model into an HDL, which is synthesized and programmed in the programmable logic of the SoC FPGA. A testbench application can be programmed in the microcontroller of the SoC FPGA to verify that the behavior of the hardware matches the behavior of the model. The HLS verification testbench can be manually translated in the SoC FPGA to replicate in the hardware the simulation conditions. If some errors are found in the hardware model, then some signals can be extracted from the hardware at running time to compare the behavior of the hardware and the simulation.

To implement the w-SHARP node, we have used System Generator over Matlab Simulink as HLS tool, Vivado, and the ADRV9361-Z7035 platform. The ADRV9361-Z7035 platform has a Xilinx 7035i Zynq SoC FPGA, which comprises an FPGA and two ARM microcontrollers, and an AD9361 radio chip, which has 56 MHz BW and supports a carrier frequency from 100 MHz to 6 GHz. Similar methodologies are also being used to implement other high-performance wireless systems, such as WirelessHP [29]. The main differences between our approach and WirelessHP are the tools and platforms. WirelessHP uses HDL-coder and the ZC706 platform, whereas we have used System Generator and the ADRV9361-Z7035 platform.

The implementation methodology flowchart (see Fig. 4) starts with the development of a System generator model with two identical w-SHARP models. The first one is a w-SHARP AP, and the second one is a w-SHARP STA. They are connected through a reconfigurable wireless channel model. The model verification is performed through a verification testbench written in a Matlab script. The testbench receives a configuration file, which includes: the node mode (AP / STA), the superframe structure (periods and T_s), and the frame configuration. From this file, the testbench creates the low-level configuration for the model and its expected output. Once the w-SHARP model has been verified in simulation for the desired use cases, an IP can be generated and imported in Vivado.

The ADRV9361-Z7035 reference design for Vivado contains a complete design to test the ADRV9361-Z7035 peripherals. This design is a convenient starting point to create a new hardware project for the ADRV9361-Z7035. From here, the w-SHARP IP can be integrated into the Vivado project by connecting its interfaces to the AD9361 controller and to the software, which will run the verification testbench. An Integrated Logic Analyzer (ILA) core can be included in the design to capture specific signals from the hardware and ease the verification and debugging. Finally, the Vivado project can be synthesized and programmed in the Zynq.

The semi-automated hardware verification can be performed using a software application equivalent to the verification testbench. The application receives a high-level configuration file and it generates the same low-level configuration. Then, the hardware and HLS model can be run over the same conditions and their outputs can be directly compared. For instance, to verify the transmitter, we can generate several configuration files that contain the Tx configuration of different frames (DL/UL, No. subframes, MCS, payload length, data, etc.). The signals connected to the ILA core (e.g., the Tx / Rx frames) are extracted and compared to the expected data from the system generator model. If there are differences between them, then the ILA core can be used to check the intermediate signals of the transmitter and track the error from the hardware to the model.

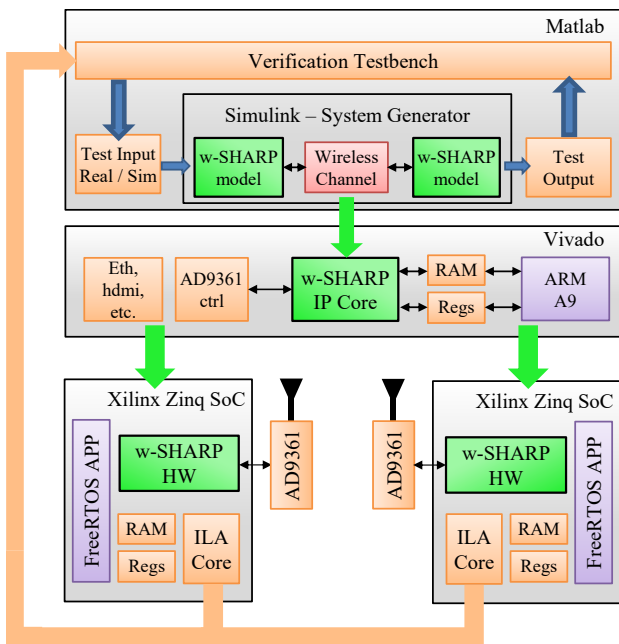


Fig. 5. Flowchart of w-SHARP implementation methodology over an FPGA-based SDR platform.

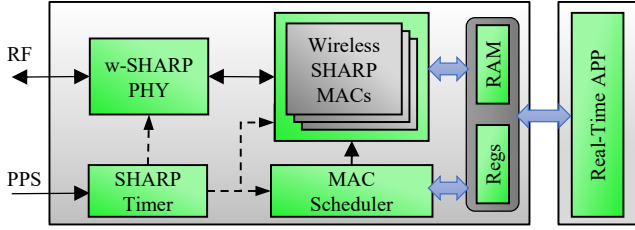


Fig. 6. Block diagram of a w-SHARP node.

Finally, the model is resynthesized until the behavior of the hardware is completely verified.

V. PROPOSED HARDWARE IMPLEMENTATION

The proposed implementation of a w-SHARP node is depicted in Fig. 6. It comprises both hardware and software. The software is an RT application, which performs the initial hardware configuration and reads/writes data from/to the hardware. Along the following paragraphs, we detail the four main elements of the hardware: the PHY layer, the MAC layer, the MAC scheduler, and the SHARP Timer.

A. SHARP Timer

The SHARP timer is a free-running timer that can be synchronized to an external time source. The timer has two outputs: the first output goes from 0 to 10^9 ns, and it is called the global timer. The second output is called the superframe timer and has a configurable period multiple of $100 \mu\text{s}$. The superframe timer is used by the time-aware scheduler and its period equals T_S .

There are two interfaces to externally synchronize the SHARP timer: a Pulse Per Second (PPS) signal and a register interface. The PPS signal can be obtained from an external IP core or hardware, and it can be used to synchronize the w-SHARP hardware with, for instance, a TSN switch embedded in the same node. The register interface can be used to synchronize the w-SHARP node through the synchronization protocol described in Section II.C.

B. MAC Scheduler

The MAC Scheduler is the element responsible for the switching between the w-SHARP MAC blocks and provides the time-aware scheduling operation. It receives a configuration structure from the software layer. This configuration structure consists of the superframe duration, the periods inside the superframe (DL/UL, etc.) and their start and end time. With this information, it switches the MAC blocks to enable traffic with different priorities.

C. W-SHARP MAC

The w-SHARP MAC blocks are used as the interface between the software and the PHY. This interface comprises 4 messages:

- *Tx Descriptor* ($Tx Desc$) which is used to command the transmission of a frame. It includes the frame format and the Tx time.
- *Rx Descriptor* ($Rx Desc$) which is used to prepare the Rx PHY to receive an incoming frame. It includes the frame format and the expected Rx time.
- *Tx End Descriptor* ($Tx End Desc$) which is sent from the PHY to the MAC to indicate the end of a frame transmission.

- *Rx End Descriptor* ($Rx End Desc$) which is sent from the Rx PHY to the MAC and carries the metadata of an Rx frame (CIR estimation, Rx Timestamp, MCS, length, etc.).

The main task of the MAC blocks is to serve as a control interface between the PHY and the software. Their operation comprises: gathering the $Tx Desc$, $Rx Desc$ and Tx data from the RAM interface, configuring the PHY to execute the operation described in the descriptors, receive the $Tx End Desc$, $Rx End Desc$ and Rx Data from the PHY, and finally, write this information in the RAM. The proposed implementation comprises a total of four MAC blocks: RT Tx MAC, RT Rx MAC, 802.11 MAC, and STA Sync MAC. The retransmission periods have not been considered because they are impractical for ultra-low latency applications.

The RT Tx MAC and RT Rx MAC blocks are used to transmit and receive the w-SHARP frames. These blocks enable the RT transmissions. For instance, in the DL periods, the AP uses the RT Tx MAC, whereas the STAs use the RT Rx MAC. The 802.11 MAC block contains an implementation of the 802.11 access scheme, with some slight modifications to avoid collisions between RT and non-RT frames. Finally, The STA Sync MAC implements the first phase of the synchronization algorithm.

The superframe duration T_S and the minimum allowable cycle time are constrained by two factors: the granularity of the w-SHARP timer, and the memory occupied by the scheduler and MAC descriptors. On the one hand, T_S must be a multiple of the time granularity ($100 \mu\text{s}$ for w-SHARP). On the other hand, the maximum allowed T_S is constrained by the size of the RAM memory given that the scheduler and descriptors are stored in the RAM interface between w-SHARP hardware and software (see Fig. 6).

D. w-SHARP PHY

The w-SHARP PHY is depicted in Fig. 7. The coder/decoder supports the next modulations: BPSK, QPSK, 16QAM, and 64QAM, combined with a convolutional coder of 1/2, which can be punctured to 2/3 or 3/4. The payload length can 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 Tx PHY operation is as follows. The Tx PHY receives a $Tx Desc$ with the frame structure along with the payload of each subframe. The data is sent to the coder, which is configured according to the $Tx Desc$ and performs the scrambling, channel coding, interleaving, and QAM modulation. The data resulted from the coder is sent to an IFFT module and the output of the IFFT module is append with the preamble. Finally, the resulted

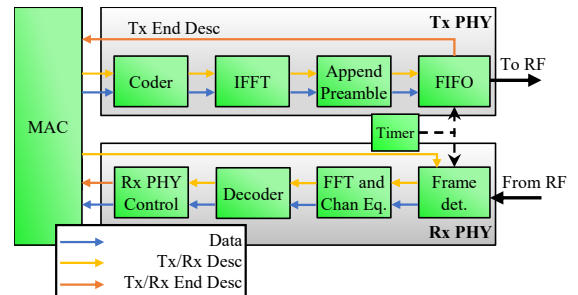


Fig. 7. w-SHARP PHY.

frame is stored in a FIFO memory until the timer output matches the $T_x \text{ desc}$ Tx time. Finally, the Tx PHY generates a $T_x \text{ End Desc}$ to indicate that the frame was transmitted.

The Rx PHY design is more complex. First, the Rx PHY receives from the MAC an $Rx \text{ Desc}$, which includes the frame 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 contains the gain/phase equalization and pilot tracking. Its output is connected to the decoder that performs the QAM demodulation, deinterleaving, channel decoding, and descrambling. Finally, the Rx PHY sends the data to the MAC along with an $Rx \text{ End Desc}$. If the frame is not detected, the Rx PHY will generate an $Rx \text{ End Desc}$ to notify the MAC that the frame was not detected.

VI. W-SHARP EXPERIMENTAL VALIDATION AND PERFORMANCE RESULTS

The methodology and hardware architecture presented in Section IV and Section V have been successfully used to implement a w-SHARP prototype over the hardware platform. We detail in the next subsections the experiments to validate the node and to test its performance. The PHY has been limited to 20 MHz BW and to OFDM (first configuration of Table I), due to the limited amount of hardware resources of the FPGA. This configuration would be equivalent to one OFDMA channel and the results with this prototype could be extrapolated to w-SHARP networks with larger BW and more OFDMA channels.

A. Definition of Key Performance Indicators

We have used the next Key Performance Indicators to evaluate the performance of the implementation: PER, communication jitter, and latency.

1) PER

The PER has been defined as the ratio between the number of erroneous delivered packets (N_e) and the number of transmitted packets (N_T):

$$\text{PER} = \frac{N_e}{N_T}, \quad (3)$$

Given that retransmissions are not considered, the PER equals the frame error rate.

2) Communication Jitter

Taking into account the deterministic behavior of w-SHARP, the communication jitter of real-time packets exactly matches the time synchronization jitter. Thus, the communication jitter has been measured as the standard deviation of the time synchronization error. There are two main methods to measure the time synchronization error: PPS-based and software-based. In PPS-based, the rising edges of the PPS signals of two synchronized nodes are compared using an oscilloscope, and the time difference between them is the time synchronization error. In software-based measurements, the error is directly estimated in the time synchronization algorithm as (2).

In this work, software-based measurements are used to measure the communication jitter because the distance between the nodes exceed the acceptable distance to perform PPS-based measurements.

3) End-to-End Latency

The End-to-End latency in industrial communications is defined as the time elapsed between the data generation ordering and the data being consumed in its final destination. The end-to-end latency can be expressed as the sum of the latency of several processes:

- Data generation latency (T_g) is the time elapsed between data is ordered and data is available.
- Scheduling latency (T_x) is the time elapsed between data is available and the frame with data starts to be transmitted.
- Modulation latency (T_{Tx}) is the time taken by the transmitter to generate and transmit the frame. It can be measured as the time elapsed between the transmission of the first byte to the Tx PHY, and the transmission of the first IQ symbol (excluding preamble).
- Frame duration (T_f) is the time taken to transmit a frame through the radio.
- Propagation latency (T_p) is the latency of the wireless channel, which is in the range of few to tens of nanoseconds for low range wireless networks.
- Demodulation latency (T_{Rx}) is the time taken by the receiver to demodulate the frame and send the data to upper layers. It can be measured as the time elapsed between the reception of the last IQ symbol of the frame and the decoding of the last bit of the frame.
- Data consumption latency (T_c) is the time elapsed between data reception and data usage.

For the sake of clarity, we assume that the logical distance between the origin and destination is only one hop, and that the time taken to perform the data generation T_g and consumption T_c is negligible compared to the network latency. In addition, T_p is in the range of the ns for indoor communications and, thus, it is considered negligible as well. Finally, T_{Tx} , rather than increase the latency, imposes a constraint to the scheduling latency T_x because the scheduler must ensure that the data arrives to the Tx PHY before the modulation process starts

$$T_x \geq T_{Tx}. \quad (4)$$

Thus, the End-to-End latency (T_{E-E}) may be expressed as

$$T_{E-E} = T_x + T_f + T_{Rx}. \quad (5)$$

T_{E-E} mainly depends on the scheduler design, the frame duration, and the modulation and demodulation latencies T_{Rx} and T_{Tx} . The minimum achievable T_{E-E} is the sum of T_{Tx} , T_{Rx} , and T_f . However, it is unlikely that the data generator will generate the data T_{Tx} μ s prior to the transmission. Generally, the data is generated at the start of every superframe or, in some cases, the data for the DL period is generated at the start of the DL period, and the data for the UL period at the start of the UL period. Thus, T_x will basically depend on the application and the relation between the network and the application.

B. Experimental validation

We have built two SHARP networks using our w-SHARP node. The first network was used to test the minimum achievable cycle time and comprised one AP and two STAs. The second network comprises one AP and five STAs and was used to validate the system under more relaxed cycle time conditions using more STAs and the transmission of non-RT

frames. The carrier frequency f_c has been set to 2.6 GHz to avoid the interferences of the 2.4 GHz band. To visualize the frames, we have used a four-port oscilloscope with a BW = 4 GHz. The RF output of the AP is connected to a splitter which outputs are connected to an antenna and to the scope. In the first network, the nodes were directly connected to the scope. 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. A photo of the setup for the first network configuration is shown in Fig. 8.

The superframe structure for the first network is summarized in Table II. For this configuration, the duration of the DL frame, comprising the beacon and the two data frames is 48 μ s, whereas the duration of each of the UL frames is 12 μ s. Then, $T_s = 100 \mu$ s can be achieved if $\text{IFS}_{\text{UL-DL}} = 13 \mu$ s and $\text{IFS}_{\text{UL-UL}} = 2 \mu$ s. This setup has been successfully validated in the hardware platform. A capture of the oscilloscope during one superframe is depicted in Fig. 9, which comprises three frames: a DL frame transmitted by the AP to the STAs, and two frames transmitted by the STA 1 and STA 2 to the AP respectively. This setup demonstrates the capabilities of w-SHARP to support ultra-low cycle time.

The configuration of the RT periods for the second setup is summarized in Table III. With $\text{IFS}_{\text{UL-UL}} = 2 \mu$ s and $\text{IFS}_{\text{UL-DL}} = 12 \mu$ s, $T_{RT} = 200 \mu$ s. We have set $T_s = 500 \mu$ s to have enough room to create an STD period to

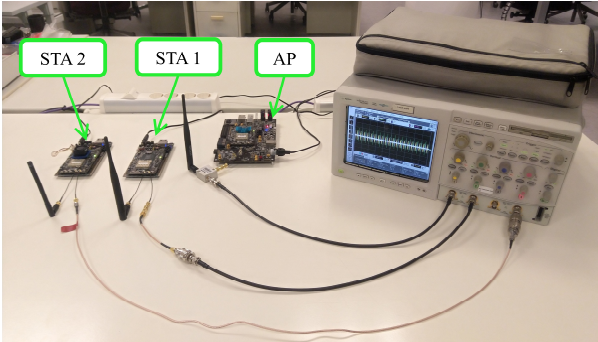


Fig. 8. Validation setup of the ultra-low cycle time of w-SHARP.

TABLE II. SUPERFRAME STRUCTURE WITH $T_s = T_{RT} = 100 \mu$ s.

Subframe type	Payload length [B]	MCS
RT Downlink Subframes		
Beacon	14	BPSK $\frac{1}{2}$
DL Data to STA 1	11	16-QAM $\frac{1}{2}$
DL Data to STA 2	11	QPSK $\frac{1}{2}$
RT Uplink Frames		
UL Data from STA1	50	64-QAM $\frac{3}{4}$
UL Data from STA2	9	QPSK $\frac{1}{2}$

TABLE III. SUPERFRAME STRUCTURE WITH $T_{RT} = 200 \mu$ s, $T_s = 500 \mu$ s.

Subframe type	Payload length [B]	MCS
RT Downlink Subframes		
Beacon	14	BPSK $\frac{1}{2}$
DL Data to STA 1	10	QPSK $\frac{1}{2}$
DL Data to STA 2	16	QPSK $\frac{1}{2}$
DL Data to STA 3	9	BPSK $\frac{1}{2}$
DL Data to STA 4	35	16-QAM $\frac{1}{2}$
DL Data to STA 5	50	64-QAM $\frac{3}{4}$
RT Uplink Frames		
UL Data from STA 1	17	QPSK $\frac{1}{2}$
UL Data from STA 2	23	QPSK $\frac{1}{2}$
UL Data from STA 3	11	BPSK $\frac{1}{2}$
UL Data from STA 4	10	16-QAM $\frac{1}{2}$
UL Data from STA 5	24	64-QAM $\frac{3}{4}$

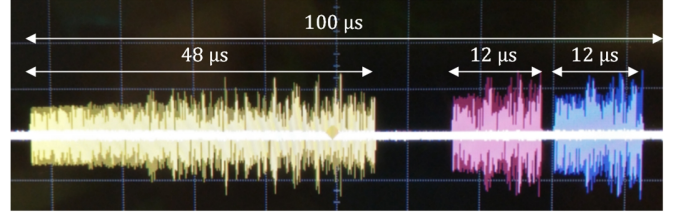


Fig. 9. w-SHARP superframe with $T_s = 100 \mu$ s.

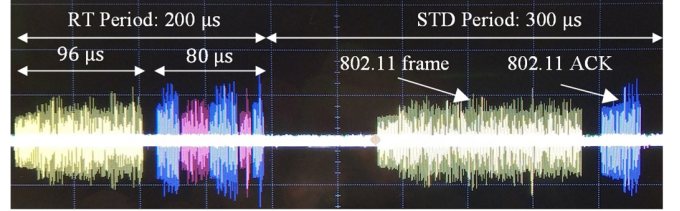


Fig. 10. w-SHARP superframe with RT and STD periods, $T_s = 500 \mu$ s.

transmit 802.11 frames. This setup has been also successfully validated. The data captured by the oscilloscope during 500 μ s is depicted in Fig. 10. In this setup, we were able to accommodate more nodes, thanks to the larger T_s , and non-RT transmissions during the STD period. In Fig. 10, the AP sends a unicast 802.11 frame to the STA 1 during the STD period and the STA 1 answers the 802.11 frame with an ACK.

From these two experiments, it can be concluded that w-SHARP is able to offer ultra-low cycle time operation, yet being flexible in its configuration, supporting different superframe structures for different application requirements.

C. Performance Results

In this subsection, we detail the results obtained in terms of PER, Jitter and latency of our w-SHARP implementation. T_{Rx} , T_{Tx} , and T_{E-E} has been measured in the laboratory since they do not depend on the scenario. T_{Rx} and T_{Tx} as function of the MCS are reported in Table IV. For the scenario reported in Table II, and considering that the data is available at the start of the superframe, T_{E-E} is: 50.2 μ s and 57.8 μ s for the DL data to STA 1 and DL data to STA 2 frames respectively, and 83.7 μ s and 96.8 μ s 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, but appropriate scheduling is required to minimize T_x .

We have used a setup with one w-SHARP AP and one w-SHARP STA to run the jitter and PER experiments. The AP transmits every 500 μ s a DL frame which comprises 4 subframes with 20 bytes each for BPSK $\frac{1}{2}$, QPSK $\frac{1}{2}$, 16-QAM $\frac{1}{2}$, and 64-QAM $\frac{3}{4}$. The STA transmits four UL frames in a row with 20 bytes each and the same modulation schemes. The AP Tx power was set to 10 dBm and f_c was set to 2.6 GHz. The STA Tx power was also set 10 dBm, but it was dynamically adjusted to compensate the channel attenuation. A total of $5 \cdot 10^7$ packets for each modulation and each direction were transmitted in each experiment.

We have run the experiments in a mechanical workshop (see Fig. 11). We have considered four possible scenarios with Line-of-Sight (LoS) or Non-Line-of-Sight (NLoS) that represent different wireless channel conditions in a workshop. The STA

TABLE IV. W-SHARP PHY LATENCY.

MCS	BPSK $\frac{1}{2}$	QPSK $\frac{1}{2}$	16-QAM $\frac{1}{2}$	64-QAM $\frac{3}{4}$
T_{Tx} [μ s]	3.5	3.7	4.9	6.8
T_{Rx} [μ s]	9.1	9.8	10.2	10.7

position was fixed for all the experiments. In scenario 1, shown in Fig. 12 (a), we put the AP on top of one machine at 5.5 meters from the STA with LoS. In scenario 2, we introduced a metal plate 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 Fig. 12 (b), we moved the AP to a distance of 12 meters with large machines blocking the LoS. Finally, in scenario 4, we moved the AP to a corner of the workshop at approximately 23 m and with a wall between them.

Table V summarizes the received power, the jitter, and the PER for the 4 scenarios described above. 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. In scenario 2, the LoS was blocked reducing the Rx power in 6 dB, to -53 dBm. Note that the workshop ceiling and walls are covered with metal that produce strong multipath components which probably contribute to keep a high Rx power. We found 0 errors for BPSK $\frac{1}{2}$, QPSK $\frac{1}{2}$, 16-QAM $\frac{1}{2}$. In 64-QAM $\frac{1}{2}$ 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

TABLE V. 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 $\frac{1}{2}$	$< 10^{-7}$	$< 10^{-7}$	$4 \cdot 10^{-7}$	$2.1 \cdot 10^{-6}$
PER QPSK $\frac{1}{2}$	$< 10^{-7}$	$< 10^{-7}$	$8 \cdot 10^{-7}$	$4.4 \cdot 10^{-6}$
PER 16-QAM $\frac{1}{2}$	$< 10^{-7}$	$< 10^{-7}$	$1.6 \cdot 10^{-5}$	$2.6 \cdot 10^{-4}$
PER 64-QAM $\frac{1}{4}$	$< 10^{-7}$	$6 \cdot 10^{-7}$	$3.3 \cdot 10^{-4}$	0.67

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-dispersive effects. 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 Signal-To-Noise Ratio (SNR) was around 18 dB, which is acceptable for low-order modulations but not for high-order modulations, according to previous results obtained by simulation means [21]. For instance, the PER with 64-QAM was 0.67, which is impractical even for non-RT communications. 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 was in the order of 10^{-6} , which may be acceptable for some industrial applications.

The jitter resulted from the three experiments was very similar and smaller than the required by the targeted applications (1 μ s): 30 ns for scenario 1, 35 ns for scenario 2, 40 ns for scenario 3 and 60 ns for scenario 4. The jitter result validates the precision of the SHARP timer and the synchronization algorithm.

VII. CONCLUSIONS

In this paper, we have presented a major update in the SHARP project, the implementation of a w-SHARP node over an FPGA-based SDR platform. We have described the implementation methodology, the hardware of a w-SHARP node, and several experiments to validate the hardware. The experiments demonstrate that w-SHARP can provide very small control cycles up to 100 μ s. Besides, the optimized PHY implementation achieves a maximum modulation and demodulation latencies of 6.8 μ s and 10.7 μ s respectively. Our w-SHARP implementation outperforms 802.11 and 5G for industrial applications and reduces the gap of the achievable cycle time between industrial wired and wireless networks.

In addition, we have evaluated the jitter and PER in real factory conditions. We have demonstrated through these experiments that w-SHARP was able to provide in both LoS and NLoS conditions a PER around 10^{-7} for short and medium distances (6-12 m), and a PER below $5 \cdot 10^{-6}$ for longer distances (23 m). BPSK and QPSK offer the best performance, far better than the performance obtained with higher-order modulations. Still, the PER for 16-QAM and 64-QAM is below $6 \cdot 10^{-7}$ in short ranges. On the other hand, the jitter is below 59 ns for every experiment, far better than the required jitter for industrial applications. These results demonstrate the potential of w-SHARP to provide wireless communications.

Nonetheless, the prototype has some limitations. In the first place, the w-SHARP prototype is limited to 20 MHz and to only OFDM, i.e. 1 OFDMA channel. As a first step, we may migrate

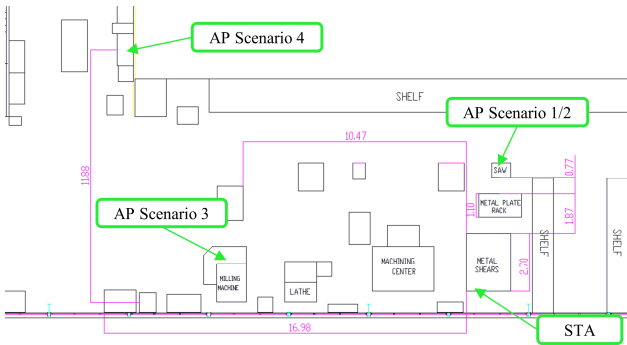


Fig. 11. Map of the mechanical workshop.



Fig. 12. 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.

the w-SHARP design to a new platform with a high-end FPGA and a radio chip such as the ADRV9009 to increase the overall performance of the prototype. However, the implementation of OFDMA itself presents its own challenges. For instance, the Tx and Rx chains have to be replicated in parallel at the w-SHARP AP side to serve different users. The replication dramatically increases the hardware resources consumption that will in turn require further optimization of the PHY hardware by using more efficient structures and higher clock frequencies to enable resource sharing among the different elements of the PHY. Additionally, the MAC and scheduler have not been initially implemented to support multiple data streams and, thus, their architecture must be modified to allow higher data rates.

In the second place, the w-SHARP prototype has not been tested over mobile scenarios with high mobility and fast fading variations. In these scenarios, the Tx equalization plays a vital role to maintain the UL reception quality, though its robustness has not been already assessed through simulations nor by experimental means. It is very likely that specific algorithms should be designed in future works to compensate the channel variation under high mobility conditions.

Another interesting future line of work would consist in building a hybrid wired/wireless TSN by integrating w-SHARP with Ethernet TSN. The first step to do so could be to adjust the w-SHARP design and implementation to comply with some TSN standards, such as 802.Qbv. Then, the hardware architecture design of the hybrid system with minimum wired/wireless scheduling latency and appropriate time synchronization could be designed.

Finally, w-SHARP is focused on supporting industrial time-critical applications and thus it has very stable performance. Consequently, it has limited flexibility when it comes to perform modifications of the superframe structure at running time or allow high data rate best effort applications. Thus, the integration of the strong features of w-SHARP into the latest wireless standards, e.g. 5G-NR or IEEE 802.11be, to build a general flexible design for the needs of nowadays and future wireless TSN is an interesting future line of work.

In summary, we believe that the w-SHARP platform is a milestone in the research of industrial high-performance wireless systems and that it opens up new research possibilities in that field. The implementation of w-SHARP presented in this work demonstrates that custom high-performance wireless systems can be built on an FPGA, though there are still major challenges to solve in terms of performance (provided BW, OFDMA, and mobile scenarios), and interoperability with TSN.

REFERENCES

- [1] K. Montgomery, R. Candell, Y. Liu, and M. Hany, "Wireless User Requirements for the Factory Workcell," *NIST Report, Adv. Manuf. Ser. (NIST AMS) - 300-8*, 2019.
- [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.
- [3] "Time-Sensitive Networking Task Group." [Online]. Available: <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.
- [5] D. Cavalcanti, "Wireless TSN – Definitions , Use Cases & Standards Roadmap," *Avnu Alliance*, pp. 1–16, 2020.
- [6] 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.
- [7] 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-Sept, pp. 1135–1142, 2019.
- [8] Ericsson, "5G-TSN Integration For Industrial Automation," 2019.
- [9] 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.
- [10] 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.
- [11] 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.
- [12] 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.
- [13] R. Costa, J. Lau, P. Portugal, F. Vasques, and R. Moraes, "Handling real-time communication in infrastructure IEEE 802.11 wireless networks: The RT-WiFi approach," *J. Commun. Networks*, vol. 21, no. 3, pp. 319–334, 2019.
- [14] 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.
- [15] 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.
- [16] 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.
- [17] 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.
- [18] 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.
- [19] 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.
- [20] 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.
- [21] 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.
- [22] *802.1Qbv - Enhancements for Scheduled Traffic*. IEEE standard 802.11Qbv, 2016.
- [23] *IEEE Standard for a precision clock synchronization protocol for networked measurement and control systems*. IEEE Standard 1588, 2009.
- [24] 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.
- [25] "AD9361 Reference Manual." [Online]. Available: <http://www.farnell.com/datasheets/2007082.pdf>.
- [26] T. Vilches and D. Dujovne, "GNUradio and 802.11: Performance evaluation and limitations," *IEEE Netw.*, vol. 28, no. 5, pp. 27–31, 2014.
- [27] H. Wu *et al.*, "The tick programmable low-latency SDR system," *Proc. Annu. Int. Conf. Mob. Comput. Networking, MOBICOM*, pp. 101–113, 2017.
- [28] "WARP." [Online]. Available: <https://mangocomm.com/products/kits/warp-v3-kit/>.
- [29] 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.