Wireless Time Sensitive Networking Impact on an Industrial Collaborative Robotic Workcell

Susruth Sudhakaran, Karl Montgomery, Mohamed Kashef, Dave Cavalcanti, and Richard Candell

***Abstract*—In this paper, we describe a methodology and associated models to evaluate a time-sensitive collaborative robotics application enabled by Wireless Time Sensitive Networking (WTSN) capabilities. We also present a method to configure WTSN scheduling to meet the application time budget and validate it in a realistic industrial use case. We detail the methodologies for implementing and characterizing the performance of key WTSN capabilities, namely time synchronization and time-aware scheduling, over an IEEE 802.11-based network. We deploy the WTSN capabilities with a collaborative robotic workcell consisting of two robotic arms, which emulate a material handling application, known as machine tending. We further explore configurations and measurement methodologies to characterize application performance of this use case and correlate it to the performance of the wireless network.**

***Index Terms*—Wireless TSN, IEEE 802.11, Collaborative Robotics**

I. Introduction

Industry 4.0 and smart manufacturing are seen as the new industrial revolution and are characterized by a fusion of physical Operational Technology (OT) and digital Information Technology (IT) worlds through breakthroughs in robotics, Artificial Intelligence (AI) and Internet of Things (IoT) [1]. The sharing of network and computing resources between digital IT and physical OT domains in smart manufacturing applications needs strictly time-synchronized and deterministic low latency communications [2]. Time Sensitive Networking (TSN) [3] and wireless TSN (WTSN) efforts [4] have emerged as an enabling networking technology to achieve precise time synchronization and timeliness in a network. Compared to wired connections, wireless connectivity brings many benefits, such as flexibility, easy reconfigurability, mobility, and lower maintenance and life-cycle costs. However, the ability of wireless technologies to guarantee time-sensitive communications with low latency and extremely high reliability, with ensured protection against cybersecurity attacks, has not been proven [5].

In order to assess the impact of wireless communications on factory automation processes, a testbed was constructed to replicate various data flows in a managed environment. In this paper, a robotic workcell is used to perform a machine tending application with emulated service requirements that represent a typical collaborative industrial use case. Time synchronization was used to keep the collected data aligned in time and for enabling WTSN. As many industrial applications are very sensitive to the performance of the network, it is important to have standard methods to assess this performance. An experimental study is performed using the testbed with an Ethernet-based wired configuration as well as a wireless IEEE 802.11ac based configuration using 2x2 Multiple-Input Multiple-Output (MIMO). The testbed's baseline design with a wired network was introduced in an earlier publication [6].

TSN Standards [7], being developed by the IEEE 802.1 TSN working group, enable time synchronization and time bounded data delivery for time-sensitive traffic over IEEE 802 Local Area Networks (LAN) shared with other (e.g. Best-Effort (BE)) traffic. Extensions of TSN capabilities and standards into wireless domains have been subject of research and standardization activities, as described in [8]. Since Wi-Fi is also an IEEE 802 LAN technology, TSN link layer capabilities can be mapped seamlessly from Ethernet to Wi-Fi, without architecture changes nor protocol translation gateways. However, achieving comparable bounded latency, wired time synchronization accuracy, and reliability performance over Wi-Fi involves numerous research questions that are open.

In this paper, we describe two critical TSN features that extended into IEEE 802.11, specifically, time synchronization (IEEE 802.1AS) and time-aware shaping (IEEE 802.1Qbv). We focus on how these features can be deployed leveraging standard Wi-Fi devices in an industrial testbed and demonstrate how these features help improve the time sensitive performance over wireless. Our 802.1AS implementation is based on the 802.1AS over 802.11 Timing Measurements protocol as defined in the 802.1AS-2020 standard [9]. Our 802.1Qbv implementation is applied at the network stack leveraging the open source 802.1Qbv implementation described in [3].

Our paper is structured as follows: in Section II, we discuss related works. In Section III, we present an overview of WTSN architecture and various system components. In Section IV, the use case and the testbed setup are briefly presented. In Section V, we present the measurement and evaluation methodology with the key performance indicators (KPIs) used. We then present the results in Section VI, followed by conclusions and future directions in Section VII.

II. Related Work

This section covers topics regarding industrial wireless communications for robotics and TSN with wireless communications.

A. Industrial Wireless Communications for Robotics

Some of the first industrial communications for robotics were serial-based field-busses, i.e., Modbus, Profibus, and DeviceNet [10]. These were used due to their reliability;

S. Sudhakaran and D. Cavalcanti are with the Intel Corperation, Hillsboro, Oregon, United States, email: {susruth.sudhakaran, dave.cavalcanti}@intel.com

K. Montgomery, M. Kashef, and R. Candell are with Communications Technology Laboratory, NIST, Gaithersburg, Maryland, United States, email: {karl.montgomery, mohamed.kashef, richard.candell}@nist.gov.

however, they lack the bandwidth and maximum node numbers required for more advanced applications. Ethernet-based field-busses were used to overcome the challenges of serial-based communications [11]. Then, to bring real-time performance, the TCP/IP layers that lacked the required determinism were substituted with a custom stack in Powerlink, Profinet RT, and EtherCAT [10].

There are existing industrial wireless standards that are based on IEEE 802.15.4, found in [11]. A downside of these various protocols is the low supported data rates, making them undesirable for many industrial use cases. In industrial wireless, determinism and real-time communication capabilities are still lacking with higher bandwidth targets [12].

B. TSN with Wireless Communications

With the need to bring real-time performance in Ethernet, a TSN task group was created [3]. One of the group's outputs was deploying time aware traffic scheduling, defined in 802.1Qbv [13]. The schedule is based on time division multiple access (TDMA). Another feature of TSN is time synchronization, accomplished by IEEE 802.1AS, which is a profile of the IEEE 1588 precision time protocol (PTP) standard. The TSN requirement is to deliver time synchronization information over IEEE 802.3 (Ethernet) or IEEE 802.11 (Wi-Fi) links to achieve sub-microsecond clock error between devices [10]. Also, Gutiérrez et al. validates improved wired TSN performance using experimental results and concludes that TSN will "become the de facto standard for communications on layers 1 and 2, in robotics" [10]. An issue with wired TSN is that wired networks do not scale with the large amount of nodes required for some automation applications [14], and hence, a hybrid wired-wireless structure is desired for future TSN industrial robotic networks.

IEEE 801.AS, specified in [9], is a time synchronization component of TSN, which we use and describe later in this work. A work that integrates IEEE 802.1AS synchronization in IEEE 802.11 can be found in [15]. There is also work to bring TSN to 5G in release 16 of 3GPP [16]. Another promising work that proposes a novel method of precise synchronization using IEEE 802.11 networks with FPGAs can be found in [14]. Overall, The TSN features that are incorporated into the wireless domain are promising to bring the advantages of wireless communications into demanding industrial automation and robotic applications.

An existing challenge is how to deploy WTSN features and measure performance in an end-to-end system. In this paper, we address this challenge by implementing and deploying two main TSN features: IEEE 802.1AS and IEEE 802.1Qbv in an IEEE 802.11 wireless network, which is utilized as the communication system for a realistic industrial robotic use case. To the best of our knowledge, we are the first to perform an experimental study assessing the physical and network performance impact of WTSN with varying levels of interfering traffic.

III. Wireless TSN Overview

With the need to bring deterministic real-time performance into standards-based Local Area Networks (LANs), the IEEE 802.1 TSN task group has been defining a set of standards to time synchronization, guaranteed end-to-end latency and jitter, as well as extremely low packet loss. Time Synchronization across the network, as defined by IEEE 802.1AS, Time Aware Scheduling, as defined by IEEE 802.1Qbv, are two of the most fundamental TSN standards. IEEE 802.1AS achieves time synchronization across a set of distributed nodes, which is a fundamental requirement for synchronous applications and other TSN features. For instance, IEEE 802.1Qbv utilizes time synchronization, along with principles of Time Division Multiplexing (TDM), to define protected time windows. The IEEE 802.1Qbv Time Aware Scheduling introduced the concept of Gate Control Lists (GCLs), which are a set of gates, each controlling a queue that is associated with a configurable traffic class. By controlling the mapping of traffic streams to classes/queues and when a given queue is open/closed, it is possible to create a TDM-based schedule that meets strict latency deadlines for high priority traffic. The GCLs for all nodes are centrally defined based on the application requirements and distributed to all nodes. In this paper, we focus on enabling and evaluating TSN features (802.1AS and 802.1Qbv) applied to an IEEE 802.11 network. It is well known that the randomness in the 802.11 MAC (Medium Access Control) is a problem for applications that expect deterministic low latency and low jitter. There has been significant research on replacing the traditional contention-based 802.11 MAC with TDMA-based MAC protocols. For instance, in [17], [18], the authors propose time synchronized slotted MACs for 802.11 and evaluate the capabilities to meet very low latency with high reliability. Although results are promising, replacing the 802.11 MAC with a TDMA-based scheme is not feasible in off the shelf Wi-Fi cards and it would require proprietary chipsets. In this work we focus on applying the principles of 802.1Qbv on top of a traditional random access-based 802.11 MAC/PHY layers to achieve similar behavior expected in a TDMA-based system.

We implement the 802.1Qbv GLC on the network stack above an off-the-shelf 802.11 chipset using TSN functionalities and tools available in the Linux kernel running on the nodes. The Linux kernel supports TSN control plane and configuration through the Linux Traffic Control (TC) system. The specific functionality that we use to implement the Qbv schedule as a GCL is the TAPRIO qdisc (queueing discipline), which implements the Enhancements for Scheduled Traffic introduced by IEEE 802.1Qbv standard on Linux kernel [19]. These are the same technologies that would be used to test and certify devices for Wireless/TSN compliance [20]. A high level overview of the implementation architecture is shown in Fig. 1. We further evaluate the performance of the same in an industrial use case. While previous works [17], [18], [21] focus on optimizing the 802.11 MAC/PHY, this work focus on the reuse of widely available standard-based

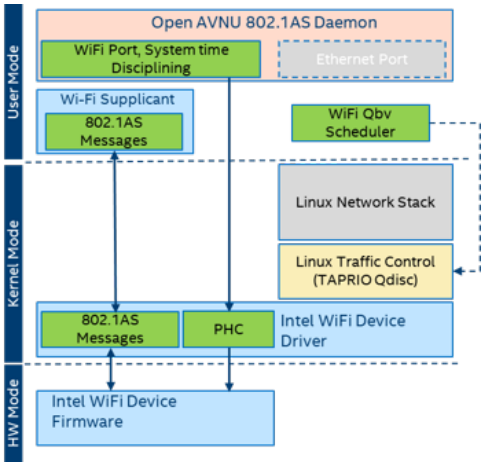


Fig. 1. Implementation overview of IEEE 802.1AS and IEEE802.1Qbv in Linux.

802.11 implementations and enhance their performance by introducing TSN features on the link layer. Fig. 2 illustrates a typical hybrid TSN network architecture where the TSN capabilities in the wired segment are extended into the Wi-Fi segment of the network. It is assumed that the network is centrally managed, which is the case in most industrial IoT deployments that are relevant for the application considered in this paper.

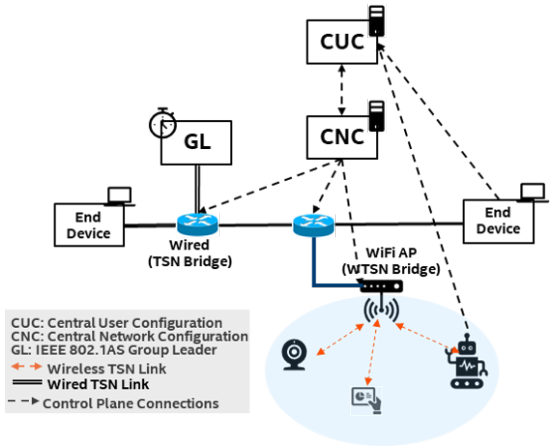


Fig. 2. Wired-Wireless hybrid TSN network architecture.

In a TSN network, every traffic stream is centrally managed and configured. This function is performed by two functional entities namely, the Central User Configuration (CUC) and the Central Network Configuration (TSN-CNC[1]) as specified by IEEE 802.1Qcc specification. The CUC collects traffic stream information from all the end devices and provides the information to the TSN-CNC. The TSN-CNC, using its discovered knowledge of the network topology, negotiates with each network element on the path to configure resources in order to meet the timing requirements of the traffic streams. This may include configuring IEEE 802.1Qbv schedules at the bridges in the infrastructure.

[1]According to the TSN standard this entity is abbreviated as CNC, but since we have other entities having the same abbreviation in this paper, we henceforth refer to this entity as TSN-CNC.

A. Time Synchronization over IEEE 802.11

Achieving precise time synchronization across all the devices in the network is foundational to any TSN capable network. The IEEE 802.1AS standard, a profile of the IEEE 1588 standard, is the protocol defined for time distribution in a TSN network and it can operate over Ethernet and Wi-Fi/IEEE 802.11 [15]. IEEE 802.1AS works above the IEEE 802.11 medium to enable IEEE 802.11 Stations (STAs) synchronize to a Grand Leader (GL) clock anywhere in the network. The IEEE 802.11 standard specifies two protocols to propagate time, as specified in IEEE 802.1AS standard: Timing Measurement (TM) and Fine Timing Measurement (FTM) [15]. The TM and FTM protocols both exchange action frames between nodes to accomplish this. The WTSN implementation in [8], employed in this paper, uses the TM feature defined in the IEEE 802.11-2016 specification. According to this specification, an IEEE 802.1AS layer uses the IEEE 802.11 TM action frames to propagate a reference time, thereby enabling a receiving STA to estimate the path delay for calculating its time offset relative to the reference time for the network, which is provided by a clock source, i.e., the GL. Fig. 3 demonstrates how the exchange of action frames takes place between two peers.

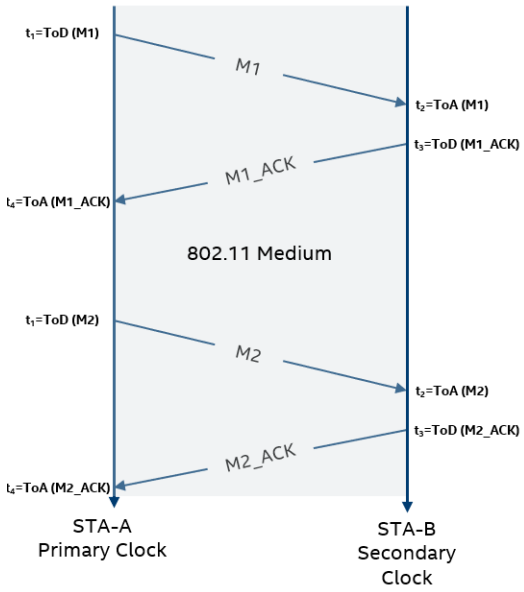


Fig. 3. IEEE 802.1AS Time Propagation over IEEE 802.11 where ToA refers to Time of Arrival and ToD refers to Time of Departure.

B. Time Aware Scheduling based on IEEE 802.1Qbv

Another fundamental TSN feature is enabling delivery of time sensitive data with deterministic time guarantees, in the presence of other background/interfering traffic sharing the network. The IEEE 802.1Qbv standard defines a set of time controlled gates controlling the queues associated with multiple traffic classes on a TSN-enabled node. This allows setting a time schedule for every traffic flow for the entire

network, which is accomplished by configuring time-aware shapers at these nodes. A time-aware shaper schedules packets following a TDMA scheme by mapping traffic classes to time slots during which transmission is allowed. The schedule itself is defined by a Gate Control List (GCL) [3]. The GCL is a list of timed gate control commands controlling the open or closed state of a traffic queue at any given point in time. This way, end-to-end protected periods or windows are created on a per-flow basis. As noted earlier, one key requirement for this to work is that time in every participating device must be synchronized to a common reference time or clock, GL. This synchronization is enabled by an implementation of the IEEE 802.1AS protocol over both wired and wireless. We have extended and implemented the concept of IEEE 802.1Qbv on IEEE 802.11 devices, thereby allowing creation of protected windows for time sensitive traffic by blocking other traffic classes (queues) from accessing the wireless channel during such windows. The concept is illustrated in Fig. 4.

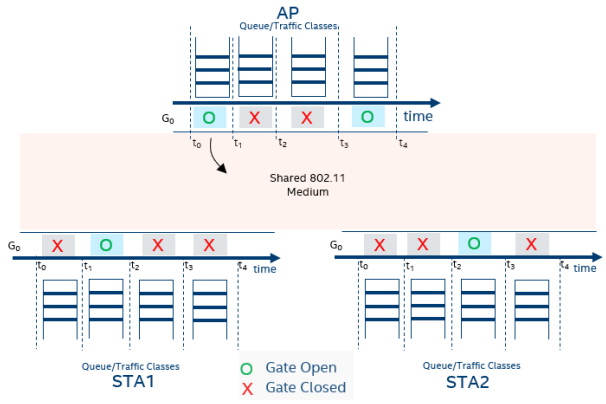


Fig. 4. TSN Scheduling in IEEE 802.11.

In this paper we evaluate an implementation of IEEE 802.1Qbv specification over Wi-Fi using a testbed emulating a industrial workcell. We will evaluate the benefits of enabling TSN over wireless to improve performance of time sensitive flows in the presence of competing background traffic.

IV. The Collaborative Robotic Workcell Testbed

The machine tending use case emulates a generic work cell consisting of multiple components such a supervisory controller, emulated CNC machines, interstage buffers, robots, and human workers. This use case is designed to replicate typical and common industrial use cases, such as pick-and-place applications mentioned in [22]. Supervisory control applications allow for some packet loss and latency issues in the communication links without directly affecting the motion path of the robots, which may happen when the wireless channel is subject to stress.

In this section, we will describe the baseline design of the testbed including the various components and network interfaces defined. Then, we present the timing budget of the operator's control loop. Lastly, we describe the extensions to the testbed to enable the WTSN features described above.

A. The Baseline Testbed

In this work, we extend the testbed with WTSN capabilities and a measurement method to allow for characterizing and measuring of the use case performance in conjunction with the network performance. In this section we describe the machine tending use case and discuss the architecture and components used to introduce WTSN capabilities to the testbed.

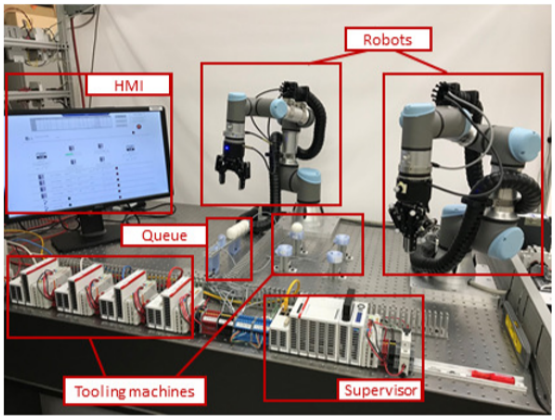


Fig. 5. Collaborative robotic workcell testbed.

Fig. 5 shows the collaborative industrial workcell and its components. The workcell consists of two Universal Robots UR3 robotic arms, a work zone, a Beckhoff CX2020 supervisor Programmable Logic Controller (PLC) equipped with a Beckhoff IEEE 1588 Precision Time Protocol (PTP) module, and 4 Beckhoff CX9020 PLCs functioning as computer numerically controller (CNC) emulators. The supervisor coordinates an emulated workflow between the robotic arms and the CNCs to implement a machine tending use case. The two robots in the workcell play two different roles – the Operator (OPT), or tender, and the other is the Inspector (INS). The OPT robot transports parts to the work zone, where the emulated CNCs, implemented using PLCs, simulate a tooling operation by waiting for a predetermined time. When a CNC emulation process is finished, the INS robot emulates a quality inspection check. The cooperation between the emulated CNC machines and the robots is managed by the supervisor by coordinating communication between the CNC machines and the robots

The communications in the workcell are centered around the supervisor, which coordinates various operational data flows. The network backbone is formed by two Cisco IE-4000 industrial Ethernet switches. The data exchanged between the supervisor and the CNC machines are formatted as Automation Device Specification (ADS) commands and carried in Transmission Control Protocol/Internet Protocol (TCP/IP) packets. Similarly, the supervisor exchanges information with the robots through Modbus communications, for which the supervisor takes the role of the Modbus server. The cycle rate of both the supervisor PLC and robot controllers is 125 Hz for operation and measurement purposes. A Meinberg M900 PTP time server plays the role of a GL clock. The clocks in all the individual measurement points are being synchronized

routinely to the GL through their connection to the Cisco IE-4000 switches as well.

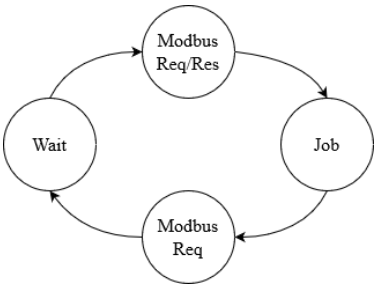


Fig. 6. Operator robot's control loop, after initialization

B. Time Budget Model of Operator Robot's Control Loop

In order to realize the improvements that WTSN can provide in a use case, a time budget model is used to determine how a WTSN schedule can be applied, and how latencies in communications can effect the application's efficiency. In Fig. 6, the state diagram of the operator robot's control loop is shown from the perspective of the supervisor. Here, the robot is either in an idle state or job state. To get to the job state from idle, the robot must receive a new job using an exchange of Modbus request and response messages. Then, after the robot is finished with a job, it sends a Modbus request message to the supervisor, indicating it is finished. This message does not require a response message, as it is writing directly to a register on the supervisor PLC. Derived from the OPT's control loop algorithm, we present Eq. (1), Eq. (2), and Eq. (3). The definitions of all the variables are presented in Table I. The time of the operator's control loop from the perspective of the supervisor, T_{optCtrl}, is defined in Eq. (1), which includes times of the Modbus read, T_{modR}, and Modbus write, T_{modW}, messages. The equations of the Modbus read and write messages are defined in Eq. (2) and Eq. (3) respectively. Since the Modbus messages are sent over wireless communications, they are non-deterministic in nature, such that the time of the application may increase in multiples of 8 ms if latencies are too large.

$$T_{\text{optCtrl}} = T_{\text{optProc}} + T_{\text{modR}} + T_{\text{wait}} + T_{\text{job}} + T_{\text{modW}} + T_{\text{sync}} \quad (1)$$

$$T_{\text{modR}} = T_{\text{cycle}}(1+\lfloor T_{\text{modReq}}+T_{\text{supProc}}+T_{\text{modRes}}+T_{\text{optProc}})/T_{\text{cycle}}\rfloor) \quad (2)$$

$$T_{\text{modW}} = T_{\text{optProc}} + T_{\text{modReq}} + T_{\text{supProc}} \quad (3)$$

C. Wireless Time Sensitive Network Extensions

In order to enable WTSN in the testbed, the two robots are bridged to the controlling PLC through a wireless network. The two robots acts as Wi-Fi stations connecting to a Wi-Fi access point (AP), which bridges them to the rest of the network. All three wireless nodes have the WTSN software stack installed, extending TSN features over the Wi-Fi domain. Hardware wise, these nodes are Intel-based Next Unit of Computing (NUC) systems (Onlogic ML100G-51) equipped with Intel 9260 IEEE 802.11ac devices [23]. The AP node is connected to the main Cisco switch through Ethernet, as seen in Fig. 7 and synchronizes its time with the GL in the network over IEEE 1588 protocol. The AP then propagates this time over wireless to the two stations.

TABLE I
TIME BUDGET DEFINITIONS

Label	Definition
T_{optCtrl}	The total time of the operator's control loop from the supervisor's perspective, starting when the supervisor writes a new job in its Modbus register to receiving a complete job in its Modbus register from the operator.
T_{optProc}	The total time for the operator to process messages or commands in the operator's loop.
T_{modR}	Total time it takes to read in a variable's value using polled Modbus messages at 125 Hz.
T_{wait}	The time spent waiting due to the supervisor not assigning new job to the operator when it is ready to receive one.
T_{job}	The time spent moving to complete a job.
T_{modW}	The total time spent from when the operator sends a Modbus write request message that signals the supervisor it is done with a job, to the time when the supervisor has read that message.
T_{sync}	A variable wait time at the end of the operator's control loop, such that the loop time is a multiple of 8 ms.
T_{cycle}	8 ms, which is determined by the cycle time of the application.
T_{modReq}	The transmission delay it takes for a Modbus request message.
T_{modRes}	The transmission delay it takes for a Modbus response message.
T_{supProc}	The total time spent at the supervisor processing messages.

V. WIRELESS TSN EVALUATION

A. Measurement setup and methodology

In this section, we describe the extensions that were added to enable measurement and comparative analysis of WTSN. Fig. 7 shows the wired and wireless network configurations of the testbed, with the red lines representing the data collection flows. The WTSN nodes described in the previous section also function as measurement probes capturing and measuring wireless time synchronization data. Wired Ethernet Test Access Points (TAPs) are installed at strategic points in the network, shown in Fig. 7, to capture all relevant network packets sent between the nodes. The traffic flows from the TAPs are simultaneously captured by multiple 4-port Ethernet PCIe cards installed at the collection machine. The 4-port Ethernet cards are Intel i210 gigabit based Ethernet cards that support TSN features and hardware timestamping features for each port.

The various network configurations include wired and wireless (with and without WTSN). The time-synchronized collection machine is responsible for collecting each of the TAPs network data, along with the robot state and position data by utilizing the real-time-data-exchange (RTDE) interface provided by the robot controllers. In this paper, the wireless configurations with WTSN and with no WTSN are subject to a UDP competing traffic stream using iperf with packet length of 1000 bytes and data rates of 1, 2, 10, and 20

Mbps, respectively. The interfering load in this scenario is synthetically generated to mimic Best Effort (BE) traffic from non-critical sensors, such as a camera stream. This interference is generated by a separate NUC, not shown in the network setup, and acts as the client, whereas the OPT Ethernet-wireless NUC acts as the server. In this configuration, the competing traffic is transmitted to the OPT NUC using the same access point that is used by the operational traffic of the testbed. The operational traffic in the testbed consists of traffic between the supervisor PLC and the two robots as well as with the CNCs. The flow between the supervisor PLC and OPT, and the flow between the supervisor PLC and INS are critical flows that must be protected from interfering traffic, which can be accomplished by using TSN schedules.

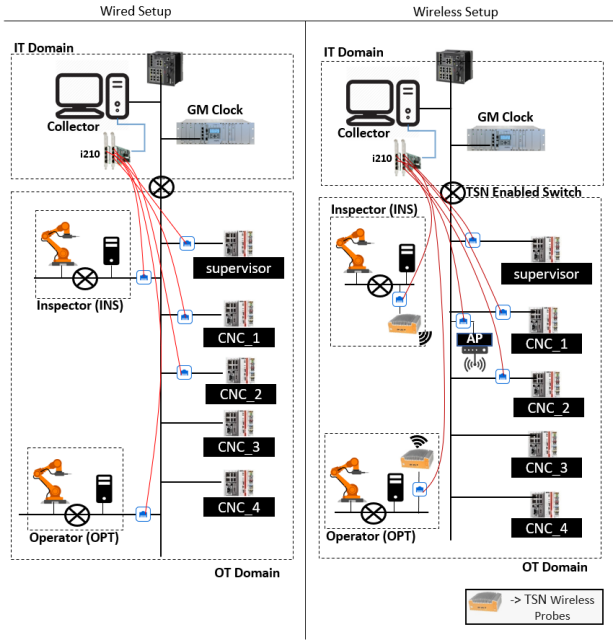


Fig. 7. Wireless TSN measurement setup of workcell testbed.

TSN schedules are used over the wireless network segment to prioritize time critical data over the BE traffic. As previously mentioned, all the measurement nodes are synchronized to the GL clock, these devices include the collection machine, the supervisor PLC, and the wireless NUCs. The accuracy of the wired time synchronization error between the collector and the GL was observed to be less than 1 microsecond. The error of the time synchronization over the wireless links were observed to be less than 100 microseconds with 99% confidence. This accuracy can be further improved to 1 microsecond by switching to an integrated Intel 9560 IEEE 802.11ac WiFi card [24], [25]. The accuracy that can be achieved is very dependent on the Wi-Fi device that is being used. This is because one important component that contributes to this accuracy is the correlation between the timestamps generated at the host platform and the timestamp generated in the device as the time synchronization packets are exchanged. In [24], the authors have shown that using a Wi-Fi device that is more tightly integrated into the host platform, like the Intel 9560 802.11ac, it is possible to achieve a time synchronization accuracy of 1 microsecond. These Wi-Fi devices, however, are not industrial grade and our use case requires industrial grade devices [24], [25]. We expect these Wi-Fi devices to be made available for industrial use in near future. The integrated version has its baseband firmware running much closer to the host processor in the platform to achieve a tighter synchronization.

The physical performance data is derived from the robot state change information logged at the collection machine. The state changes of the use case are sensitive to the network performance, as the instructions from the supervisor PLC are sent using Modbus messages transmitted over the wireless channel. Thus, increases in network delay is correlated with an increase in the idle time of the robots, which is the time spent by the robots in the idle state waiting for jobs from the supervisor. We measured this impact by using the state transition logs collected from the robot controller to determine how long the robot was in the idle state. Every 8 ms, the robot's state is recorded as an integer value at the time-synchronized data collection machine. The integer value indicates what job is assigned to the robot, or for it to be idle. The actual impact of competing wireless traffic can be measured as any delay in the communication link that causes a delay in the supervisor's state messages to the robots, thereby increasing the idle time of the robots.

To measure the network performance metrics, the traffic streams captured at TAP points at the supervisor, OPT, and INS are analyzed to calculate the delay and jitter of packets between the supervisor and each robot. We can derive the path delay of these packets using the synchronized packet data collections.

B. Key Performance Indicators (KPIs)

Two categories of KPIs have been identified to evaluate the performance of WTSN and the testbed - network performance KPIs and application KPIs. Each type of these KPIs has associated metrics - Cumulative Distribution Functions (CDFs) of Packet Delivery Ratio (PDR) of observed latency measure performance of time sensitive streams, and Idle Time of the application measures the performance of the application. Table II summarizes these metrics.

TABLE II
KPI METRICS

KPI	Description
Packet Delivery Ratio (PDR)	PDR is the ratio of network packets delivered within a defined latency bound belonging to time sensitive streams.
Latency Cumulative Distribution Function (CDF)	The Latency CDF is the distribution of latency values for all network packets observed for a traffic stream.
Idle Time	Measures the amount of time the application driving the robots have spent in the idle state.

Since the application is supervisory control, the idle time will be directly impacted by any latency in the network so any KPI's representing these two metrics would help evaluate

the use case performance. Some other KPI's that could be measured include time synchronization accuracy, packet error rate, production rate, robot response times etc., however we selected the most direct KPI's to assess the network and use case performance.

C. Deriving Time-aware schedules for the wireless nodes

To enable WTSN, a schedule was derived to match the requirements of critical application flows over wireless. We let the schedule start before the initialization phase of the application such that the application traffic will align with the schedule. In this case, the worst-case initial delay is the cycle period, which in this use case is 8 ms. Afterwards the application will continue to be aligned within the worst-case time synchronization accuracy of the testbed, which is 100 microseconds. To configure a WTSN schedule, we use the following steps.

1) Compute the cycle time for a WTSN schedule using Base Period (BP) [26] of the time critical flows.

$$T_{\text{cycle}} = GCD(T_i : F_i \in \mathcal{F}), \tag{4}$$

where $\mathcal{F}$ is the set of all flows, T_i is the cycle time of the ith flow in the set $\mathcal{F}$, and $GCD()$ computes the Greatest Common Divisor.

2) Derive the per-flow relative start time and relative offset in a cycle such that

$$T_{\text{start},i} = T_{\text{start},i-1} + T_{\text{offset},i-1}, F_i \in \mathcal{F} \text{ and } i \neq 0, \tag{5}$$

where $T_{\text{start},i}$ denotes the relative start time of the ith flow to the schedule start where $T_{\text{start},0} = 0$ and $T_{\text{offset},i}$ denotes the offset of the next flow relative to the ith, and hence, it represents the scheduled time for the ith flow's transmissions including a guard time window.

3) Compute the slot duration of each slot, which is equal to the transmission time of all packet exchange during that slot.

$$P_{\text{tdelay}} = N_{\text{P}} * \frac{P_{\text{avg-size}}}{transmission\ rate} + T_{\text{processing}}, \tag{6}$$

where N_{P} is the number of packets and $T_{\text{processing}}$ denotes any processing delay associated with the packet exchange.

There are two critical flows that are relevant in this experiment: the flow between the supervisor PLC and OPT, and the flow between the supervisor PLC and INS. We will denote these flows as F_{OPT} and F_{INS} respectively. We use the formulation above to compute the T_{cycle} as follows

$$T_{\text{cycle}} = GCD(T_{\text{OPT}}, T_{\text{INS}}) = 8 \text{ ms}. \tag{7}$$

We use the start time, average transaction duration, and the average packet size of the Modbus packets exchanged in these flows to derive the per-flow relative offsets in each cycle. The relative start time for the two flows F_{OPT} and F_{INS} has been configured as follows

$$T_{\text{start,OPT}} = 0;\ T_{\text{offset,OPT}} = 2 \text{ ms};$$

$$T_{\text{start,INS}} = T_{\text{start,OPT}} + 2 \text{ ms}. \tag{8}$$

This configuration gives a budget of 2 ms for the total transmission delay (P_{tdelay}) for the Modbus transaction for F_{OPT} in each cycle. The average transaction duration for each of the transaction happening in a cycle is given by adding T_{modReq}, T_{supProc} and T_{modRes} as defined in section IV. C. So, the total transmission delay is given by

$$P_{\text{tdelay}} = (T_{\text{modReq}} + T_{\text{modRes}} + T_{\text{supProc}}) = 4 * (P_{\text{avg-size}}/transmission\ rate) + T_{\text{supProc}}, \tag{9}$$

where $P_{\text{avg-size}}$ is the average packet size associated with the Modbus packets exchanged during this transaction and transmission rate refers to the link speed of the wireless link between the OPT and the supervisor, or INS and the supervisor. Also note that each Modbus packet is a TCP packet and is associated with a corresponding acknowledgement message. We used a client device as software-based AP (SoftAP) in our experiments operating at the 2.4 GHz band with a channel bandwidth of 20 MHz. From this test configuration, we assume a conservative transmission rate of 54 Mbps to create a transmission schedule according to IEEE 802.1Qbv. This assumption is made from the observation of sniffed Wi-Fi packets, and the testbed is isolated without mobility or other significant interferers during the experiments. Also, our assumptions enable a reliable schedule to be made. We further observe, by analyzing the packet trace of the traffic exchange between these entities, that the average packet size of packets exchanged in such transactions is around 80 bytes. We also observe that the average processing time of the supervisor (T_{supProc}) is around 1 ms. So, the total transmission delay in this case can be computed as 1.05 ms, which is well below the budget available. Ideally, we need to take the worst case of the requirement while formulating the schedule, however in this case, the packet sizes and rate of generation are very consistent. Consequently, the deviation in measured values from the average for transmission time and processing times is in the nanoseconds, which is small compared to the worst-case time synchronization error of 100 microsecond. Also, each time critical flow has been allocated a time budget that is sufficiently larger than what is required.

Using the formulation described, a schedule is derived as illustrated in Fig. 8. Each cycle is divided into two time slots – one for TS traffic, which is 5 ms long, and the other for BE traffic, which is 3 ms long. This 5 ms window was further split into slots for F_{OPT} and F_{INS}, according to the formulations above into 2 ms slots, with some room acting as guard bands. The traffic streams have been classified as Time Sensitive (TS) and BE, and are directed into correspondingly labelled queues on the interfaces at each of the wireless nodes. The queues are then gated by a time-based schedule, which repeats every 8 ms. When gate G_1 is open, BE traffic is let through, and when gate G_0 is open, the TS traffic is let through.

Each wireless transmitter/receiver node implements a Priority Mapping Function and a Time Aware Scheduling function. These working of these functions are described in Fig. 9.

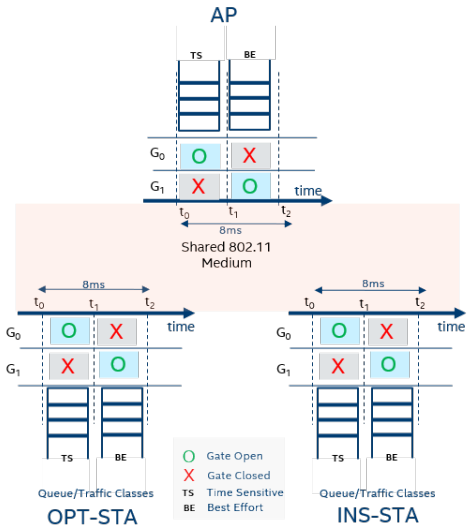


Fig. 8. WTSN Schedule in the wireless medium.

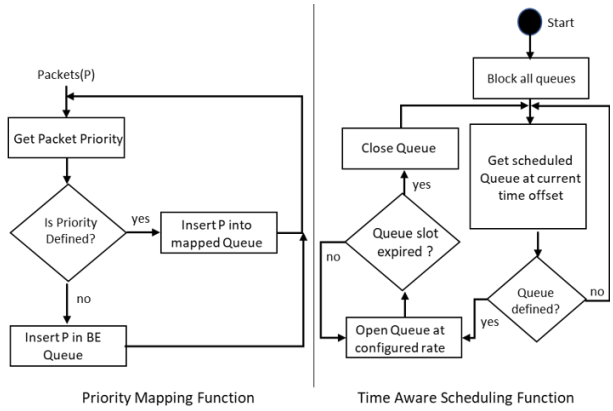


Fig. 9. Main Functions in WTSN Node.

The application start time and the schedule start time is synchronized as close as possible using automated scripts. In a TSN network, this schedule and time synchronization are coordinated by the central entities CUC and TSN-CNC. Although these functional entities have been defined in the standards, implementation of these functional entities and derivation of schedules is still an active area of research, especially in a hybrid wired-wireless network. The available bandwidth for BE traffic is reduced as this traffic only gets, approximately, one-third of the cycle period for transmission after adequately allocating bandwidth for shaping and protective guard bands.

VI. Results

The machine tending use case experiments were run with the various network configurations and interfering traffic described in section IV-A. In summary, the experiments consist of wired and wireless baselines, then wireless without TSN, and wireless with TSN scenarios, for which there was 1, 2, 10, and 20 Mbps of BE interfering traffic. The resulting traces collected with the TAPs, RTDE, and the supervisor PLC, are analyzed to comparatively evaluate any benefits achieved using TSN in the presence of competing BE traffic. In this section, we discuss the results that are observed as a result of the analysis.

Using the measurement methodology highlighted in section IV-A, the traffic streams from all the scenarios previously discussed are analyzed for latency. Fig. 10 shows the PDR for packets having latency less than 5 ms, beyond which the efficiency and stability of the use case is degraded. We can clearly see in Fig. 10 that, in the presence of BE traffic, when there are no TSN schedule applied over wireless, there is a drastic drop in the PDR. When compared to the similar metric in the case of wireless baseline (99.81%) and wired baseline (100%), we can see that enabling the TSN schedule is able to bring the overall latency profile of the time sensitive streams closer to the benchmark.

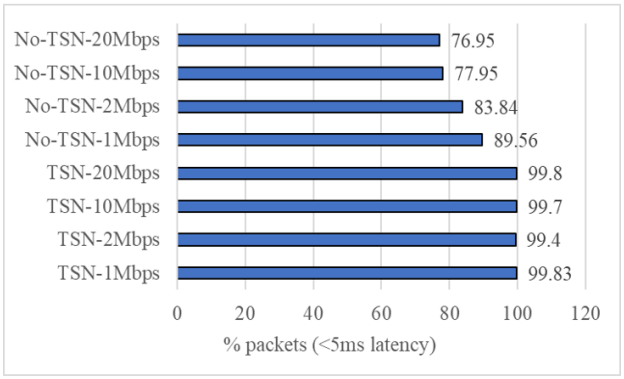


Fig. 10. Packet Delivery Ratio for latency $<$ 5 ms with TSN and no-TSN cases with varying levels of interference traffic.

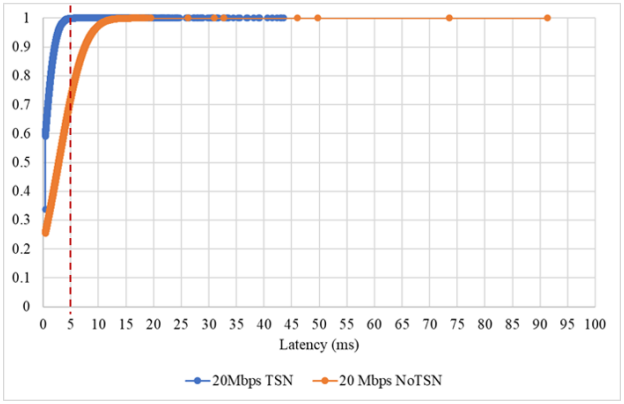


Fig. 11. Latency CDF of packets with and without TSN enabled.

Fig. 11 shows a more detailed picture of the latency distribution and how TSN helps in improving the overall latency experienced by the use case. We show the CDF comparison of the latency distribution experienced by time sensitive traffic in the presence of the worst-case interference (20Mbps) when TSN schedule is active and when it is not. Here, we can see that, when TSN schedule is active, more than 99% of time sensitive packets experience a bounded latency of less than 5 ms. This contrasts with the scenario when TSN schedule is not active, where the percentage of packets within the latency

bound of 5 ms drops down to around 77%. Bounded latency is very important for industrial robotic use cases and has a direct impact on the performance and efficiency of the use case. The latency CDF figure also shows that the number and maximum range of outliers are reduced when wireless TSN is enabled.

This increase in the percentage of packets experiencing latency outside of tolerable bounds is also reflected in the application performance, as shown in Fig. 12.

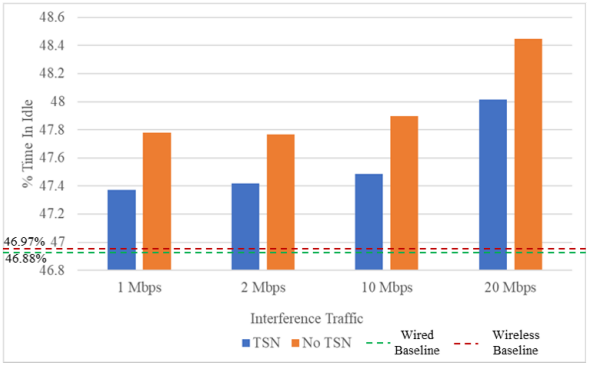


Fig. 12. Operator (OPT) Idle time across all scenarios.

We can clearly see here that the idle time experienced by the operator robot is higher compared to the case when TSN schedules are employed or compared to the baseline cases. The reason for this can be attributed to the fact that when there is competing BE traffic in the network there is no protection for the time sensitive traffic, so the operator spends more time waiting to receive state change commands from the supervisor PLC as the packets are getting delayed. This will have a cumulative effect as the delay from the last state change from the robot will result in a delay in the subsequent state changes. The wireless link delay is a direct contributor to an efficiency drop in an industrial setting where collaborative robots are jointly accomplishing supervisory tasks over wireless.

VII. Conclusions

Time sensitive and deterministic connectivity will play a key role in enabling next generation industrial systems, which require deterministic data delivery with high reliability. In this paper, WTSN features were utilized to enable low latency wireless communications in an industrial collaborative robotics use case. The analysis of network latency and its correlation to the use case efficiency was presented. Although we have shown an implementation of TSN on top of an existing traditional WiFi MAC and PHY without any support from the underlying wireless device, it should be noted that an ideal implementation would involve changes to the MAC packet scheduling mechanism to incorporate the Time Aware Scheduling concepts outlined in 802.1Qbv, as well as expose interfaces to support and configure this. In the results, we have shown how we have been able to limit the latency for over 99 percent of the time sensitive packets to less than 5 ms, but this is still not ideal, and an ideal implementation would help us improve this further. To ensure interoperability, these modifications will have to be accomplished in a way that is consistent with the evolving 802.11 standards and not in a proprietary way [17], [18], [21]. As far as we are aware, this paper is the first kind of paper exploring TSN implementation over standards-based Wi-Fi with a robotics-based industrial testbed. In future work we intend to explore further improvements with TSN over standards-based Wi-Fi with some of the newer features, such as trigger-based OFDMA scheduling in Wi-Fi 6 and multi-link enhancements being developed in 802.11be the next generation of Wi-Fi standard. Beside these, new scheduling capabilities enabled by Wi-Fi 6 can help reduce latency and provide more deterministic access. We plan to extend the WTSN implementation to Wi-Fi 6 to enable further experiments and hope to revisit these scenarios to examine further improvements in the KPIs presented.

Disclaimer

Certain commercial equipment, instruments, or materials are identified in this paper in order to specify the experimental procedure adequately. Such identification is not intended to imply recommendation or endorsement by the National Institute of Standards and Technology, nor is it intended to imply that the materials or equipment identified are necessarily the best available for the purpose.

References

[1] Martinez, B., Cano, C., Vilajosana, X. (2019). A Square Peg in a Round Hole: The Complex Path for Wireless in the Manufacturing Industry. IEEE Communications Magazine, 57(4), 109–115. https://doi.org/10.1109/MCOM.2019.1800570

[2] Y. Liu, M. Kashef, K. B. Lee, L. Benmohamed and R. Candell, "Wireless Network Design for Emerging IIoT Applications: Reference Framework and Use Cases," in Proceedings of the IEEE, vol. 107, no. 6, pp. 1166-1192, June 2019.

[3] IEEE 802.1 Time Sensitive Networking (TSN) Task Group: https://1.ieee802.org/tsn/.

[4] M. Eisen, M. M. Rashid, A. Ribeiro, and D. Cavalcanti. "Scheduling Low Latency Traffic for Wireless Control Systems in 5G Networks." arXiv preprint arXiv:1910.13587 (2019).

[5] Seferagić, A., Famaey, J., De Poorter, E., and Hoebeke, J. (2020). Survey on Wireless Technology Trade-Offs for the Industrial Internet of Things. Sensors, 20(2), 488. https://doi.org/10.3390/s20020488

[6] Y. Liu, R. Candell, M. Kashef, and K. Montgomery, "A collaborative workcell testbed for industrial wireless communications—the baseline design," 2019 IEEE 28th International Symposium on Industrial Electronics (ISIE), pp 1315–1321, June 2019.

[7] J. Farkas, L. L. Bello and C. Gunther, "Time-Sensitive Networking Standards," in IEEE Communications Standards Magazine, vol. 2, no. 2, pp. 20-21, JUNE 2018.

[8] 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," in Proceedings of the IEEE, vol. 107, no. 6, pp. 1132-1152, June 2019.

[9] IEEE, "802.1AS-2020 - IEEE Standard for Local and Metropolitan Area Networks–Timing and Synchronization for Time-Sensitive Applications," 2020.

[10] C. S. V. Gutiérrez, L. U. S. Juan, I. Z. Ugarte, and V. M. Vilches, "Time-Sensitive Networking for robotics," Apr. 2018, doi: arXiv:1804.07643v2.

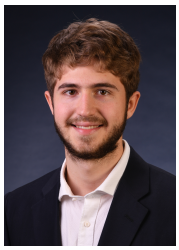
[11] P. Zand, S. Chatterjea, K. Das, and P. Havinga, "Wireless Industrial Monitoring and Control Networks: The Journey So Far and the Road Ahead," J. Sens. Actuator Networks, vol. 1, no. 2, pp. 123–152, Aug. 2012, doi: 10.3390/jsan1020123.

[12] T. Adame, M. Carrascosa, and B. Bellalta, "Time-Sensitive Networking in IEEE 802.11be: On the Way to Low-latency WiFi 7," Dec. 2019.

[13] T. Jeffree, "802.1Qbv - Enhancements for Scheduled Traffic," 2016. [Online]. Available: https://www.ieee802.org/1/pages/802.1bv.html.

[14] A. M. Romanov, F. Gringoli, and A. Sikora, “A Precise Synchronization Method for Future Wireless TSN Networks,” IEEE Trans. Ind. Informatics, vol. 17, no. 5, pp. 3682–3692, May 2021, doi: 10.1109/TII.2020.3017016.
[15] M. Gundall, C. Huber, and S. Melnyk, “Integration of IEEE 802.1AS-based Time Synchronization in IEEE 802.11 as an Enabler for Novel Industrial Use Cases,” Jan. 2021, doi: arXiv:2101.02434v1.
[16] 3GPP TR 22.804 V16.2.0, “3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Study on Communication for Automation in Vertical Domains (Release 16).”
[17] Seijo, J. A. López-Fernández and I. Val, “w-SHARP: Implementation of a High-Performance Wireless Time-Sensitive Network for Low Latency and Ultra-low Cycle Time Industrial Applications,” in IEEE Transactions on Industrial Informatics, vol. 17, no. 5, pp. 3651-3662, May 2021, doi: 10.1109/TII.2020.3007323.
[18] O. Seijo, I. Val, M. Luvisotto and Z. Pang, “Clock Synchronization for Wireless Time-Sensitive Networking: A March From Microsecond to Nanosecond,” in IEEE Industrial Electronics Magazine, doi: 10.1109/MIE.2021.3078071.
[19] TSN Documentation Project for Linux, “Configuring TSN Qdiscs”, Available at https://tsn.readthedocs.io/qdiscs.html (Accessed 2022).
[20] Dave Cavalcanti, et al, “Wireless TSN – Definitions, Use Cases Standards Roadmap”, Available at https://avnu.org/wirelessTSN/ (Accessed 2022).
[21] M. Luvisotto, Z. Pang and D. Dzung, “High-Performance Wireless Networks for Industrial Control Applications: New Targets and Feasibility,” in Proceedings of the IEEE, vol. 107, no. 6, pp. 1074-1093, June 2019, doi: 10.1109/JPROC.2019.2898993.
[22] E. Najafi and M. Ansari, ”Model-Based Design Approach for an Industry 4.0 Case Study: A Pick and Place Robot,” 2019 23rd International Conference on Mechatronics Technology (ICMT), 2019, pp. 1-6, doi: 10.1109/ICMECT.2019.8932132.
[23] Intel Wi-Fi 5 Discrete Client Reference, https://ark.intel.com/content/www/us/en/ark/products/99445/intel-wireless-ac-9260.html, Accessed: 2/11/2021.
[24] Intel WIFi 5 Integrated Client Reference, https://ark.intel.com/content/www/us/en/ark/products/99446/intel-wireless-ac-9560.html, Accessed: 2/11/2021.
[25] Ganesh Venkatesan, et al, “Performance Evaluation of IEEE 802.1AS-based Time Distribution over 802.11,” IEEE ISPCS 2019, September 2019.
[26] Nayak, N.G.; Dürr, F.; Rothermel, K. Time-sensitive Software-defined Network (TSSDN) for Real-time Applications. In Proceedings of the 24th International Conference on Real-Time Networks and Systems, New York, NY, USA, 4–10 August 2021; pp. 193–202.

Susruth Sudhakaran is a Senior Researcher at Intel Corporation, with over 15 years of experience in the field of Wireless Systems development and performance engineering. His current research focus is in Wireless Time Sensitive Networks and enabling deterministic performance over standards-based Wi-Fi networks. At Intel Labs, Susruth is leading research in methods of measurement, monitoring and configuration of Wireless Time Sensitive Networks. His other research interests include designing and building physical testbeds as well as co-simulation systems to model and study performance of complex end-to-end multi-agent system and its application as digital twin in actual physical deployment. He received his M.S in electrical engineering in 2003 from Southern Illinois University, USA.



Karl Montgomery is an Electronics Engineer in the Networked Control Systems Group at the National Institute of Standards and Technology (NIST), Gaithersburg, MD. He received a B.S. in Electrical Engineering from the University of Maryland at College Park in 2018. He is pursuing an M.S. in Electrical and Computer Engineering at the Johns Hopkins Whiting School of Engineering, with a focus in Communications and Networking. He has been working on the industrial wireless project at NIST for over 3 years, for which he and the wireless team perform research to assess performance of wireless communications systems in typical industrial automation applications by utilizing the NIST industrial wireless testbed.

Mohamed Kashef received the B.Sc. and M.S. degrees (with Hons.) in electronics and electrical communications engineering from the Cairo University, Cairo, Egypt, in June 2006 and August 2009, respectively, and the Ph.D. degree in electrical engineering from the University of Maryland at College Park, College Park, MD, USA, in 2013. He is currently a Research Scientist at the National Institute of Standards and Technology (NIST). His current focus is on industrial wireless systems including wireless networks deployment, channel modelling, applying artificial intelligence for data analysis and test methods for wireless networks in industrial scenarios. His research interests also include wireless communication systems and networks, visible light communication networks, and optimization of stochastic systems.

Dave Cavalcanti is Principal Engineer at Intel Corporation where he develops next generation wireless connectivity and distributed computing technologies to enable autonomous, time-sensitive systems and applications. He received his PhD in computer science and engineering in 2006 from the University of Cincinnati. He leads Intel Lab’s research on Wireless Time-Sensitive Networking (TSN) and industry activities to enable time-critical systems and applications of wireless technologies, including Wi-Fi and beyond 5G systems. He is Senior Member of the IEEE and serves as the chair of the Wireless TSN working group in the Avnu Alliance, an industry group facilitating an ecosystem of interoperable TSN devices and deterministic networking across Ethernet, Wi-Fi and 5G technologies.

Richard Candell has over two decades of experience in wireless systems design engineering, test, and evaluation. Dr. Candell served as the lead systems engineer in developing spread spectrum interference cancellation and performance evaluation strategies for satellite ground stations and mobile phased array beam steering transceivers. He holds patents in successive interference cancellation and transmission burst detection applied to spread-spectrum satellite. He holds a Ph.D. in Computer Science from the University of Burgundy and BS and MS degrees in Electrical Engineering from The University of Memphis. He joined the National Institute of Standards and Technology (NIST) in the US in 2014 where he now leads the Industrial Wireless Systems research lab. His current research interests include the performance of wireless mechatronics automation and safety applications. Dr. Candell was a primary author of the NIST Guide to Industrial Wireless Systems Deployments and he serves as the chair of IEEE P1451.5p.