

Synchronizing AES67 over Non-Cellular 5G

Alexander Poets^{*†}, Maxim Penner[‡], Stephan Preihs^{*}, M. Dolores Pérez Guirao[†], and Jürgen Peissig^{*}

^{*}Leibniz University Hannover, Hannover, Germany, Email: {poets, preihs, peissig}@ikt.uni-hannover.de

[†]Ostfalia University of Applied Sciences, Wolfenbüttel, Germany, Email: {a.poets, m.perez}@ostfalia.de

[‡]RFmondial GmbH, Hannover, Germany, Email: penner@rfmondial.de

Abstract—Audio-over-IP systems such as AES67 rely on the IEEE 1588 Precision Time Protocol (PTP) to align their media clocks for sample-accurate playback across networked devices. Maintaining this synchronization over a wireless link is difficult, since a relay forwarding PTP messages introduces a variable delay that, left uncorrected, degrades the receiver’s time estimate. In previous work, we investigated DECT NR+, the world’s first non-cellular 5G standard, as a cable replacement for professional networked audio with Dante, and found that low-latency audio transport was feasible but that the PTP synchronization did not match that of a wired connection. In this paper, we address this limitation for AES67 with a PTP transparent clock that bridges the DECT NR+ link and corrects for the forwarding delay each message accumulates in transit. Using an open-source software-defined radio implementation of DECT NR+, we set up a link between two PTP endpoints and measure the error of the receiver’s recovered clock against an independent 1PPS reference, demonstrating synchronization as accurate as over the wired connection the link replaces. We further show that synchronization is maintained while the link carries two eight-channel AES67 streams. The implementation is publicly available as open source.

Index Terms—DECT-2020 NR, networked audio, IEEE 1588 Precision Time Protocol (PTP), transparent clock.

I. INTRODUCTION AND RELATED WORK

Networked audio systems reproduce sound across many separately clocked devices, and coherent reproduction requires the devices to convert each sample to sound at precisely aligned instants. Audio-over-IP (AoIP) standards such as AES67 meet this requirement by deriving each device’s media clock from a shared network time base, so that every endpoint advances its sample counter against a common reference [1]. Under AES11, the recommended practice for synchronizing digital audio equipment, a device’s output must align with the reference to within 5% of the sample period (approximately 1 μ s at 48 kHz), and a device outside this window is no longer considered synchronized [2]. Such timing errors are not merely a question of specification compliance, as they have audible consequences for multichannel reproduction. Human spatial hearing is sensitive on a similar scale: an interaural time difference of roughly 25 μ s already produces the smallest shift in perceived direction a listener can detect, about 3° [3]. In

a line array, where the outputs of several amplifiers combine by interference to form a directional radiation pattern, microsecond-scale timing mismatches between those outputs can degrade that pattern, reducing its directivity control and spectral consistency [4].

In AES67, Dante, and Ravenna alike, the common clock is distributed by the IEEE 1588 Precision Time Protocol (PTP). Each device recovers its offset from a time transmitter through an exchange of timestamped messages and, across a well-behaved wired network, tracks the transmitter to within a fraction of a microsecond [5]. Any node adding a variable, direction-dependent delay when forwarding PTP messages introduces synchronization error, be it a cascade of PTP-unaware switches or a wireless relay. How far this degrades the recovered clock depends on the link. Over an unmodified Wi-Fi link, the PTP offset reaches around 100 ms, against roughly 120 ns over Ethernet, and even with the packets transmitted redundantly over two separate Wi-Fi networks under SMPTE ST 2022-7 seamless protection switching, the offset is reduced only to about 20 μ s [6]. Closing this gap calls for an on-path mechanism that accounts for the delay the forwarding node introduces. This is the role of a transparent clock: it measures the residence time each PTP event message spends in the forwarding node and records it as a correction in the PTP common header for the receiver to subtract when recovering its offset from the time transmitter [5].

In evaluating DECT-2020 NR as a cable replacement for professional networked audio with Dante, we proved low-latency audio transport feasible but found that carrying PTP natively over the link held the devices in sync only with a forgiving clock servo, and even then short of a wired connection [7]. Still, DECT-2020 NR, marketed by the DECT Forum as DECT NR+ and the world’s first non-cellular 5G technology, has several properties that make it a good fit for Programme Making and Special Events (PMSE) applications: it supports scheduled channel access for predictable transmission timing; it can operate in license-exempt, technology-exclusive spectrum; and it is designed for nomadic, operator-free deployments [8]. In recent follow-up work, we compensated for the synchronization error the radio introduces when forwarding PTP uncorrected by implementing a transparent clock spanning the wireless link, cutting the root-mean-square of the receiver’s reported offset from 141 μ s to 21 ns [9].

This paper answers the question that measurement left open, i.e., how accurately the link synchronizes in *absolute* terms, beyond the receiver’s own estimate. Because PTP

The authors would like to thank the German Federal Ministry for Economic Affairs and Climate Action (BMWK) for the funds received as part of the MERCI project (grant numbers 01MJ22016B, 01MJ22016D, and 01MJ22016E). The authors from Ostfalia further thank the German Federal Ministry of Research, Technology and Space (BMFTR) for the funds received as part of the OpenDECT-X project (grant number 01MK26002A).

cannot detect a constant asymmetry between the forward and reverse paths, a converged clock servo reports near-zero offset regardless of the bias the asymmetry introduces. We therefore measure the error against an independent 1PPS reference, the established way of measuring synchronization error against an external reference [10], and of verifying a transparent clock's accuracy in particular [11]. Since AES67 uses the 2008 revision of IEEE 1588 that introduced the transparent clock, the same testbed can carry synchronized AES67 audio, and we examine whether synchronization is maintained once the link also carries media. The implementation is released as open source.¹ The remainder of the paper is organized as follows. Section II reviews synchronization in networked audio, including the mechanics of PTP; Section III details the architecture of our transparent-clock implementation; Section IV describes the experimental setup, including the 1PPS reference measurement and the AES67 audio transport; Section V presents and discusses the results; Section VI concludes; and Section VII outlines future work.

II. SYNCHRONIZATION IN NETWORKED AUDIO

In AES67, every endpoint disciplines a local media clock to the PTP-distributed network time and expresses the Real-time Transport Protocol (RTP) timestamp of each audio stream on that same timescale, so a receiver maps every packet to an absolute instant and advances its sample counter in lockstep with the time transmitter [1]. What the listener hears is thus only as accurate as each endpoint's agreement on PTP time: an offset in the recovered clock shifts the media clock by the same amount. Since AES67 defines no normative tolerance for this offset, AES11 lends a sense of scale: if the media clock were used to drive an AES3 output, that output would have to track the reference to within a microsecond, the phase offset AES11 tolerates at 48 kHz [2]. In its media profile, AES67 also specifies bounds on the PTP message rates and clock attributes [1]. These ranges overlap those of SMPTE ST 2059-2, the PTP profile that SMPTE ST 2110 mandates for professional media over IP [12], closely enough that, on an agreed set of parameters, AES67 and ST 2110 equipment can share one PTP domain [13].

To track the common network time, the receiver must determine how its local clock is offset from the time transmitter's. A single timestamped message cannot reveal that offset on its own, since the difference between its departure and arrival timestamps includes both the offset and the unknown transit delay. PTP therefore uses a two-way exchange of timestamped messages, shown in Fig. 1, which lets the receiver isolate the offset from the transit delay. The time transmitter sends a Sync message, timestamping its departure as t_1 , which the receiver timestamps on arrival as t_2 ; the receiver then returns a Delay_Req at t_3 , which the transmitter timestamps on arrival as t_4 . These two are *event* messages, timestamped at the instant they cross the wire. To compute its offset, the receiver needs all four timestamps, but it measures only t_2 and t_3 itself; t_1

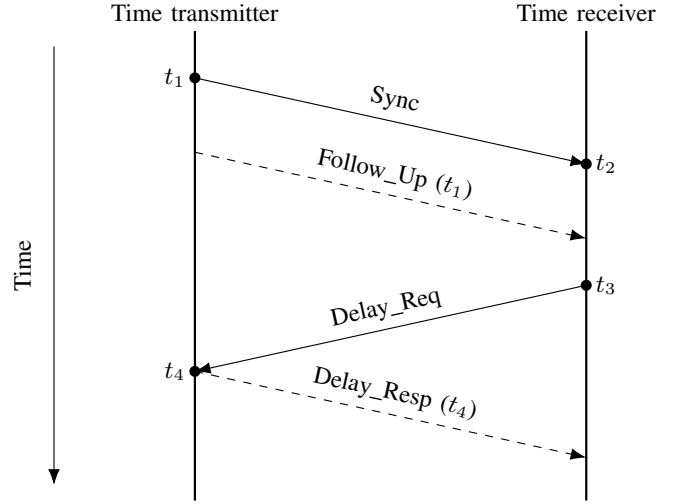


Fig. 1. Two-step IEEE 1588 delay request–response exchange. Solid arrows are the timestamped *event* messages; dashed arrows are the *general* messages that report the transmitter-side timestamps. The receiver forms meanPathDelay and offsetFromMaster from t_1 – t_4 through (1) and (2); the offset is recovered exactly only if the forward and reverse propagation delays are equal.

and t_4 are recorded at the transmitter and must be reported to the receiver. The transmitter reports t_4 in a Delay_Resp, and reports t_1 in one of two ways: a one-step transmitter embeds t_1 directly in the Sync as the message departs, whereas a two-step transmitter, for which t_1 becomes available only once the Sync is already on the wire, sends it in a separate Follow_Up (shown here). The Delay_Resp and Follow_Up are *general* messages: unlike the event messages, they are not timestamped as they cross the wire but merely carry timestamps taken earlier. Every PTP message includes a correctionField in its header, where a forwarding node along the path can record an adjustment to the message's timing. The receiver computes the mean path delay and its offset from the transmitter, (1) and (2), using the four timestamps and two such corrections: c_f accumulated on the forward path, traversed by the Sync, and c_r on the reverse path, traversed by the Delay_Req.

$$\text{meanPathDelay} = \frac{1}{2} [(t_2 - t_1 - c_f) + (t_4 - t_3 - c_r)] \quad (1)$$

$$\text{offsetFromMaster} = \frac{1}{2} [(t_2 - t_1 - c_f) - (t_4 - t_3 - c_r)] \quad (2)$$

To see how the equations separate the two unknowns, suppose the receiver's clock runs ahead of the transmitter's by some offset. The forward trip then appears longer than it truly took, by exactly that offset, because the Sync's departure is stamped by the transmitter but its arrival by the receiver; the return trip, stamped by the same clocks in swapped roles, appears shorter by the same amount. Adding the two measurements cancels the offset, so (1) yields the mean of the forward and reverse transit times whether or not the two are equal. Subtracting the measurements cancels the transit time instead, but only the part common to both directions: (2) therefore recovers the offset only if a message takes equally long in each direction, and any uncorrected asymmetry between the forward and reverse paths biases the recovered offset by half its magnitude.

¹ Available: https://github.com/po3ts/DECT-NR-Plus-SDR/tree/tfw_ptp_tc

Without on-path correction, i.e., $c_f = c_r = 0$, the time a message spends inside a forwarding node is indistinguishable from path delay. That time is rarely equal in the two directions, and the resulting asymmetry is invisible to the protocol. In measurements against an external reference, deliberately asymmetric delay caused offset errors of up to several hundred microseconds while PTP continued to operate normally [10]. IEEE 1588 defines two on-path mechanisms for neutralizing the delay PTP event messages accrue in intermediate nodes. A Boundary Clock (BC) terminates PTP at its time-receiver port, disciplines a common local clock to the recovered time, and regenerates PTP from that clock on its time-transmitter ports, so that the forwarding delay is not passed on. A Transparent Clock (TC), introduced in the 2008 revision of IEEE 1588, instead relays each event message without terminating it, measuring the residence time the message accumulates inside the node and adding it to the correctionField, c_f or c_r . The receiver then subtracts this correction in (1) and (2), so that the residence time no longer biases the path delay and offset estimates [5].

Both mechanisms presuppose an intermediate node that is a single device. A switch satisfies this trivially: the same clock supplies both the ingress and the egress timestamp, so the residence time is a duration measured on a single clock, into which only that clock's frequency error enters [14]. Even with a frequency error of ten parts per million, a residence of hundreds of microseconds is misjudged by only nanoseconds. The TC presented in the next section keeps every residence measurement within a single node, so that no clock is ever compared across the radio link. Within a node, however, a message still enters and leaves through interfaces governed by different clocks, which must be reconciled before a residence time can be computed.

III. PTP TRANSPARENT CLOCK ARCHITECTURE

Fig. 2 shows the two-step end-to-end TC that bridges the wireless link. It consists of two radio nodes, each attaching to one PTP endpoint via Ethernet and carrying that endpoint's frames to the other node over the DECT NR+ air interface. Each node acts as a local TC, measuring how long each PTP event message resides in the node from ingress to egress. Consecutive TCs cascade by adding their residence times into the same correctionField, which the receiver then subtracts [5]. For the Sync message traced in the figure, the corrections sum to $c_f = r_0 + r_1$; a Delay_Req message crosses the link in the opposite direction and accumulates c_r in the same manner, so the estimates the receiver forms through (1) and (2) are corrected for the forwarding delay of both nodes. In effect, the nodes do for the wireless link what a PTP-aware switch does in a wired network.

Within each node, the message crosses two interfaces, each keeping time separately. The Network Interface Card (NIC) timestamps PTP messages in hardware against its PTP Hardware Clock (PHC); the radio times the transmission and reception in its sample-clock domain. A residence time is therefore a difference between readings on two different

clocks, which requires the relation between the clocks to be known. A frequency standard at each node establishes this relation by supplying a 1PPS that aligns the second boundaries of both clocks, so that they differ only by an integer-second offset. This offset, δ_n at node n , is constant while the clocks stay locked and is computed at startup as the difference between the values of the PHC and the sample clock at the same 1PPS edge, so a PHC timestamp converts to the sample-clock domain by subtracting δ_n .

As the Sync message traced in Fig. 2 traverses the link, four instants of its transit are captured, only two of which are hardware timestamps. At the first node, the NIC timestamps the message's arrival against its PHC as A , and the radio schedules its transmission for B , the sample-clock instant of the node's next available transmission opportunity. The wait for that opportunity, spent in the transmit buffer, thus lies between A and B and is included in the correction rather than left to appear as path delay. Since the ingress timestamp A is read from the PHC, the time elapsed on the sample clock is $B - A + \delta_0$. At the second node, the radio recovers the arrival instant C by locating the preamble of the enclosing DECT NR+ packet in the received sample stream, and the NIC timestamps the departure against its PHC as D . With the PHC now supplying the egress timestamp, the offset enters with the opposite sign, giving $D - C - \delta_1$.

Since B and C mark instants in the radio's sample stream rather than at the antenna, the latency of the signal chain in between (the digital filters, the data converters, and the analog front end) is not yet accounted for. It falls between B and C , alongside the over-the-air propagation, the only one of the two that is path delay in the sense of (1). We measure this latency as the delay τ of a transmit-to-receive loopback. The loopback yields only the combined latency of the transmit and the receive chain, but since only the sum $r_0 + r_1$ reaches the receiver, how τ divides between the two nodes does not matter, and we assign it in full to the transmitting node. The complete residence times are then $r_0 = B - A + \delta_0 + \tau$ and $r_1 = D - C - \delta_1$.

The egress timestamp D is captured by the NIC only once the frame has departed, so the correction cannot travel in the event message itself. Each node therefore operates as a two-step TC, forwarding the event message unmodified and adding the correction to the associated general message, just as a two-step time transmitter reports t_1 in the Follow_Up after the Sync has left (Section II). For a Delay_Req, each node retains the residence time and includes it in the Delay_Resp sent in reply, which travels in the opposite direction. Since the correction alters the UDP payload, the UDP checksum no longer matches; IEEE 1588 permits an on-path node to update it incrementally or, over IPv4, to set it to zero [5], of which our implementation does the latter.

The over-the-air propagation and the wired segments between the PTP endpoints and the NICs remain uncorrected. Both are path delay, which the delay request-response exchange recovers under its assumption of symmetric forward and reverse paths. The receiver itself cannot confirm that the

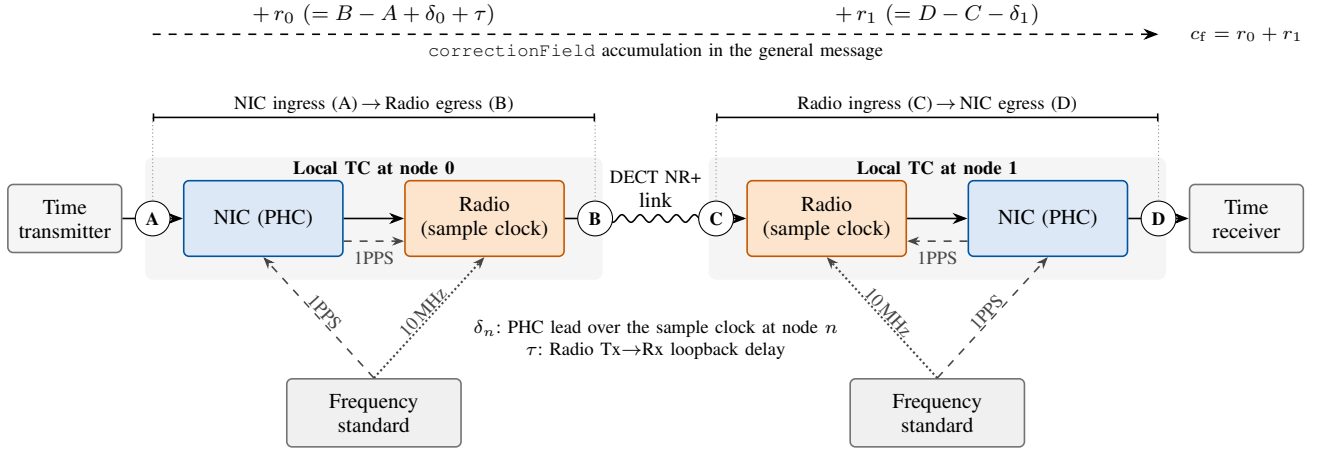


Fig. 2. Two cascaded local TCs, one per node, bridging the DECT NR+ link (adapted from [9]). The figure traces a Sync message from left to right; a Delay_Req message travels in the opposite direction and is handled symmetrically. An event message’s residence time is measured within each node, where it crosses two clock domains, and accumulated in the corresponding general message’s correctionField. This way, no clock is compared across the link.

corrected link satisfies this assumption: once its servo has converged, the offset it reports settles near zero whether or not a constant error remains, be it a residual path asymmetry or a miscalibration of δ_n or τ (Section II). Section IV therefore measures the synchronization the link achieves against a reference external to the protocol.

IV. EXPERIMENTAL SETUP

The testbed comprises two identical hosts, one for each node in Fig. 2. Each host has an Intel Core Ultra 9 285K with 64 GB of RAM and an Intel E810-XXVDA4T NIC, selected for two features the residence-time measurement requires: an SMA input through which an external 1PPS reference can discipline its PHC, and hardware timestamping of UDP-encapsulated PTP packets under the Data Plane Development Kit (DPDK). The radio is an Ettus Universal Software Radio Peripheral (USRP) N310, interfaced via the USRP Hardware Driver (UHD) and feeding a single antenna, so the link is single-input single-output (SISO). Two SRS FS725 rubidium frequency standards supply the 10 MHz and 1PPS references, one at each node. The NIC’s ports are split across two drivers: Port 0 remains on the kernel driver, owning the PHC and carrying the radio’s management traffic, while IQ streaming to the radio and the connection to the PTP endpoint run on DPDK-bound ports. As all ports share the single PHC, disciplining it through Port 0 also governs the hardware timestamps captured at the port attached to the PTP endpoint.

DECT NR+ operates in Time Division Duplex (TDD) on a 10 ms radio frame divided into 24 slots of 416.67 μ s each [8]. A radio device can take on one or both of two roles: a Fixed Termination point (FT) coordinates the link, scheduling channel access and emitting periodic beacons, whereas a Portable Termination point (PT) associates to an FT. The roles are not mutually exclusive: a device can act as a PT toward a parent while serving as an FT for its own children, which is how DECT NR+ forms the clustered-tree topologies the standard supports. Besides random access with Listen Before

Talk (LBT), the standard offers scheduled, contention-free access, in which the FT grants recurring transmission opportunities suited to periodic traffic with latency and reliability demands [8]. Both nodes run an open-source Software-Defined Radio (SDR) implementation of DECT NR+ [15], one host acting as the FT and the other as the PT.

The link reuses the configuration with the highest rate of transmission opportunities from our previous work: scheduled access with the resources split evenly between uplink and downlink [7]. Each transmission opportunity spans one half-slot (208.33 μ s), and as the two directions alternate, each radio can transmit up to 2400 packets per second, one every 416.67 μ s. The TC corrects the transmit-buffer wait, so ideally the frequency of these opportunities would not affect synchronization. In practice, however, any residual rate difference between the PHC and the sample clock distorts the measured residence time by an amount that scales with the wait in the transmit buffer. Frequent transmission opportunities bound this error while also lowering the latency, which matters once the link also carries media. The configuration requires more bandwidth than the DECT band permits, so the link operates at a center frequency of 3767.828 MHz, within the band allocated for private 5G campus networks in Germany. The full transmission numerology is reported in [7]. For this radio and configuration, we measured the loopback delay of Section III as $\tau = 2.77 \mu$ s.

At each node, the rubidium’s 10 MHz output feeds the radio’s external frequency reference input and syntonizes the sample clock. Its 1PPS feeds the NIC on SMA1, where an onboard Digital Phase-Locked Loop (DPLL) locks to it and re-emits a 1PPS on SMA2 into the radio’s external time reference input. Separately, `ts2phc` from the `linuxptp` suite phase-aligns the PHC to that DPLL. At startup, the radio latches its sample-clock time at a 1PPS edge, the NIC timestamps that same edge against the PHC as an external timestamp event, and the difference of the two readings gives δ_n .

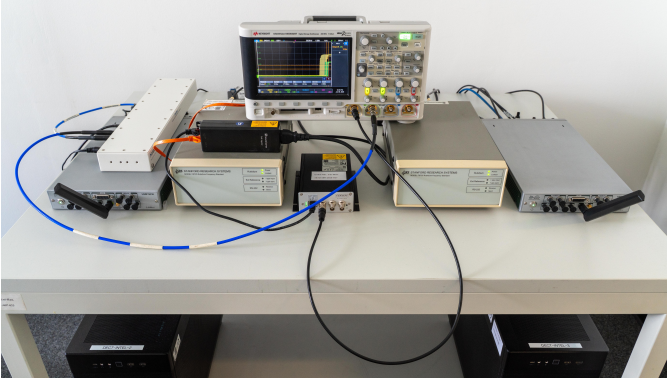


Fig. 3. The assembled testbed during the 1PPS reference measurement. Each of the two nodes pairs a host PC (beneath the table) with an Ettus USRP N310 radio and an SRS FS725 rubidium frequency standard, the latter supplying the 10 MHz and 1PPS references. On the FT node (left), the OctoClock distributes the reference 1PPS from the E810 NIC to the radio and to the oscilloscope (center), which compares it against the 1PPS the Meinberg SyncBox N2X time receiver (center front) recovers from the PTP stream it receives over the DECT NR+ link at the PT node.

AES67 defaults to the end-to-end delay mechanism and carries PTP over UDP/IPv4. The implementation corrects two-step flows of this kind; all other traffic, including peer-delay messages, is bridged across the link unmodified.

A. 1PPS Reference Measurement

Comparing the receiver’s recovered time against the transmitter’s requires both timescales as electrical signals, i.e., a transmitter and a receiver that each emit a 1PPS whose rising edge coincides with their second boundaries. On the FT node, IQ streaming therefore moves to a separate Intel X710 NIC, freeing two of the four E810 ports, on one of which `ptp4l` runs as the time transmitter, physically looped back into the DPDK-bound port through which an external PTP endpoint normally interfaces with the SDR. `ptp4l` serves the PHC time shared by all E810 ports. The 1PPS re-emitted on SMA2 marks the second boundaries the PHC is aligned to, making the transmitter’s timescale available as an electrical signal. `ptp4l` operates within the implementation’s correction scope (i.e., in two-step mode, over UDP/IPv4, with the end-to-end delay mechanism) and sends Sync messages every 125 ms, the default rate of the AES67 media profile [1], and Announce messages once per second.

The time receiver is a Meinberg SyncBox N2X, which synchronizes via PTP and emits a 1PPS aligned to its recovered time; the manufacturer specifies the pulse output as accurate to within ± 100 ns of the time transmitter after the initial synchronization phase [16]. Its 10/100BASE-T port and the E810’s SFP28 interface, which runs at 1/10/25 Gbit/s, support no common link speed, so the two cannot negotiate a link directly; a TP-Link TL-SG105 switch bridges the incompatible PHYs, connecting the SyncBox to the PT node’s E810 port where it receives the time transmitter’s PTP stream from the SDR. Since the switch is not PTP-aware, its forwarding delay remains uncorrected and is present in the measurement

conditions described below. On the FT node, shown in Fig. 3, an Ettus OctoClock fans out the 1PPS from the E810’s SMA2 output, distributing it to the radio’s external time reference input and to a Keysight InfiniiVision DSOX3024T oscilloscope, which measures the synchronization error as the interval between each reference edge and the corresponding edge of the SyncBox’s output.

We measure three conditions. The first is a wired reference, in which the port on which `ptp4l` runs is connected to the SyncBox with only the switch in between, bypassing the radio link. This condition serves as a benchmark, establishing the time receiver’s synchronization error over the connection the wireless link is meant to replace. In the second, the PTP exchange crosses the DECT NR+ wireless link with the residence-time correction enabled. A run with the correction disabled is omitted, as the SyncBox emits a 1PPS only once synchronized, and the queuing delays PTP messages experience in the radio nodes vary far beyond what its servo is designed to handle. The third condition repeats the measurement over the wireless link, now with the SyncBox’s fixed offset compensation enabled to correct for the mean synchronization error observed in the second condition, in line with what IEEE 1588 specifies for handling known path asymmetries [5].

B. AES67 Audio Transport

In the second experiment, we let the link also carry audio and investigate whether synchronization is maintained under load. Two AES67 endpoints replace the loopback time transmitter and the SyncBox, each connected directly to a node’s SDR-governed port with no switch in the path. The audio sink and time transmitter on the FT node is an RME Digiface Dante audio interface operating in AES67 compatibility mode; it exchanges two-step PTP over UDP/IPv4 with the end-to-end delay mechanism and therefore lies within our transparent clock’s correction scope. The audio source and time receiver on the PT node is a Raspberry Pi 5 running the open-source AES67 Linux daemon [17].

The source transmits two multicast streams of eight channels each: L24-coded 48 kHz audio with a packet time of 1 ms, sent as 1000 packets per second per stream [1]. A transport block of 1552 bytes (i.e., the payload of a DECT NR+ physical layer packet) carries one such Ethernet frame per transmission opportunity, so the two streams claim 2000 of the 2400 opportunities available per second in the uplink; a third stream, needing a further 1000, would not fit. Packet counts at the FT node’s port confirm that this load crossed the link throughout each run.

While the Delay_Req messages share the uplink with the two audio streams, the Sync messages travel the downlink, which carries no audio. The queuing delay the audio packets introduce thus affects one direction only and, left uncorrected, would enter the receiver’s estimates as path asymmetry. As in [9], we assess the synchronization at the receiver: `ptp4l` runs on the receiver’s port alongside the daemon’s own PTP client and reports the offset and path delay its servo computes,

once with the residence-time correction enabled and once with it disabled. Measuring with the correction disabled, which we omitted in Section IV-A, is possible here because the metric differs: the 1PPS comparison presupposes a receiver that synchronizes, whereas `ptp41`'s reported diagnostics are observations of the message exchange itself, available whether or not its servo can track them. Transport latency over the link, studied in [7], is outside the scope of this experiment.

V. RESULTS AND DISCUSSION

Table I reports the synchronization error for the three conditions described in Section IV-A. Each row presents summary statistics of the delay measured between the reference edge and the SyncBox's recovered edge, accumulated over 3750 acquisitions per condition. The error is positive throughout, i.e., the recovered edge leads the reference, so the receiver runs ahead of the time transmitter. Over the wired connection, the receiver's pulse rises 3026 ns early on average, with a standard deviation of 12.1 ns.

This offset is the result of the rate conversion at the PTP-unaware switch. With the overhead of UDP, IPv4, and Ethernet, a 44-byte PTP event message crosses the network as a 720-bit frame, be it a Sync or a Delay_Req message. The store-and-forward switch retransmits a frame only once fully received. From the transmitter at 1 Gbit/s, reception completes after 0.72 μ s; from the SyncBox at 100 Mbit/s, after 7.2 μ s. This leaves a path asymmetry of 6.48 μ s, which biases the clock offset by half its magnitude (Section II). Therefore, the time receiver overestimates the forward transit by 3.24 μ s and settles ahead by that amount. As this calculation covers the switch alone and does not account for asymmetries elsewhere in the path, we measure the fixed offset and compensate for it empirically. That measurement must come from outside the protocol: PTP cannot detect a constant path asymmetry (Section II), so the receiver's own estimate reads near zero, and the lead appears only against the independent 1PPS reference.

Running PTP over the DECT NR+ wireless link leaves this picture essentially unchanged: the mean error moves by 115 ns to 2910 ns with a standard deviation of 3.4 ns. This invariance is due to the transparent clock correcting the residence time every PTP event message spends in each radio node (Section III). This way, the link's variable forwarding delay is compensated on-path, and PTP crosses the wireless link as though over wire, rather than being carried natively as in our earlier work [7]. As can be seen, the static offset is a property of the shared wired segment and not of the radio link,

TABLE I
1PPS SYNCHRONIZATION ERROR IN NANOSECONDS

Condition	Mean	Std. dev.	Min.	Max.
Wired	3025.8	12.1	2995.6	3061.6
Wireless	2910.4	3.4	2884.4	2917.6
Wireless (offset-compensated)	25.5	5.4	10.4	54.8

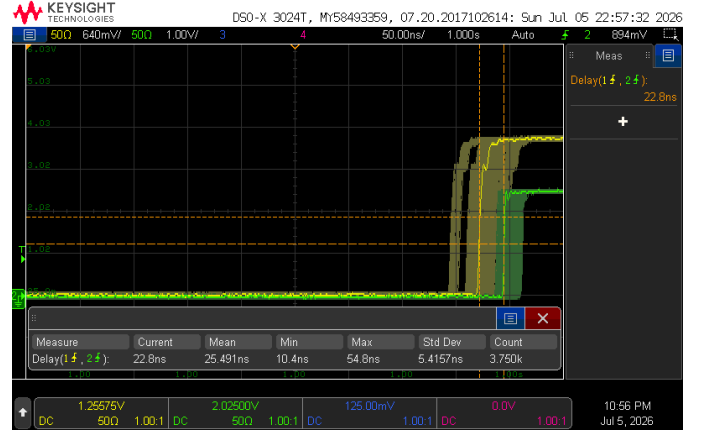


Fig. 4. PTP run across the wireless link, with a per-node transparent clock correcting for the variable forwarding delays and the static offset compensated at the Meinberg SyncBox time receiver. The oscilloscope overlays the recovered 1PPS from the SyncBox (channel 1, yellow) with the reference 1PPS (channel 2, green) in a persistence display, at 50 ns per horizontal division. The shaded bands highlight the positions the edges visited over the measurement.

whose own contribution is of the same order as the SyncBox's specified accuracy of ± 100 ns.

For the third condition, we configured the SyncBox to compensate for a fixed offset of 2910 ns, the mean measured during the preceding run over the wireless link. The residual error then averages 25.5 ns with a standard deviation of 5.4 ns and a peak-to-peak of 44.4 ns, comfortably within the receiver's specified ± 100 ns. With the transparent clock correcting the radio's forwarding delay on-path and the static offset removed at the SyncBox, the receiver tracks the transmitter's second boundaries as closely as its manufacturer specifies. Figure 4 shows a screen capture of the oscilloscope's persistence display, where the shaded band traces the range of edge positions the recovered pulse visited over the run.

Table II reports summary statistics of the offset and path delay `ptp41` logs at the receiver during the audio transport described in Section IV-B, once with the residence-time correction enabled and once with it disabled. The statistics cover steady state, excluding the first 30 s after the servo locks. With the correction enabled, the receiver tracks under load: the reported offset averages -1.7 ns with a root-mean-square of 21.2 ns, and the path delay reads 2.26 μ s with a standard deviation of 8.9 ns. The wait behind queued audio packets accrues inside the PT node, between a Delay_Req's arrival at the NIC and its transmission at the radio, so it is part of the measured residence time and enters the correctionField, which the receiver accounts for in its estimates.

With the correction disabled, the path delay reads 792 μ s, as the receiver folds the uncorrected residence of both nodes, now including the audio queueing on the uplink, into its estimate. The reported offset spreads over hundreds of microseconds, with a root-mean-square of 155 μ s. Its mean of 106 ns is small only because large errors of either sign cancel, not because the receiver is synchronized. The root-mean-square exceeds by two orders of magnitude the tolerance of approximately

TABLE II
SUMMARY STATISTICS OF PTP4L OUTPUT IN NANoseconds

Quantity		TC disabled	TC enabled
offsetFromMaster	mean	106.1	-1.7
	rms	155 173.1	21.2
meanPathDelay	mean	791 808.8	2257.5
	std	70 573.7	8.9

1 μ s within which AES11 considers a device synchronized (Section I). Synchronized AES67 over a wireless link therefore depends on an on-path mechanism to correct for the residence time accumulated in the forwarding nodes, here a PTP transparent clock.

VI. CONCLUSION

In this paper, we measured how accurately a PTP time receiver synchronizes across a DECT NR+ wireless link bridged by a transparent clock, evaluating the recovered time against a reference independent of the protocol. Compared on their 1PPS outputs, the receiver tracked the time transmitter over the corrected link as accurately as over the wired connection it replaces, the mean synchronization errors of the two conditions differing by 115 ns. The constant offset near 3 μ s common to both conditions was due to a PTP-unaware switch in the shared wired segment. The path asymmetry the switch introduced remained invisible to the protocol and surfaced only against the external reference; once compensated, the residual error averaged 25.5 ns. The bridged radio link thus contributed less to the synchronization error than the wired path around it.

Synchronization also held up when the link was put under load carrying audio data: with two eight-channel AES67 streams, the offset reported at the receiver kept a root-mean-square of 21.2 ns, whereas disabling the residence-time correction let it spread two orders of magnitude beyond the window within which AES11 considers a device synchronized. In wired AES67 networks, synchronization withstands competing traffic only where the switches support PTP or prioritize it [18]; on the wireless link, the transparent clock provides that support without prioritization, since the correction accounts for the queueing delay rather than preventing it.

Overall, with the transparent clock correcting the forwarding delay on-path, synchronization no longer stands in the way of DECT NR+ as a cable replacement for professional networked audio.

VII. FUTURE WORK

Future work should extend the implementation toward deployment. The precision reported here rests on the rubidium standards supplying each node's references, which in the field could be replaced by disciplining the NIC directly to GNSS. Whether comparable results can still be achieved with its DPLL left free-running is worth investigating. Our implementation does not solve synchronization for Dante, which natively relies on PTPv1, a version of IEEE 1588 that predates

the correctionField a transparent clock writes to; bridging a Dante network would therefore call for a boundary clock. Beyond the point-to-point link studied here, the system should be operated point-to-multipoint, validated with professional-grade equipment such as Merging Anubis audio interfaces in a representative live environment, and extended with transmit and receive diversity or full multiple-input multiple-output (MIMO) operation to improve reliability.

REFERENCES

- [1] "AES standard for audio applications of networks - High-performance streaming audio-over-IP interoperability," *AES67-2023 (Revision of AES67-2018)*, 2023.
- [2] "AES recommended practice for digital audio engineering - Synchronization of digital audio equipment in studio operations," *AES11-2020 (Revision of AES11-2009)*, 2020.
- [3] S. Forrest, "Wireless Speaker Synchronization: Solved," in *Audio Engineering Society Convention 139*, Oct. 2015, Engineering Brief 202.
- [4] B. Duval, X. Doukopoulos, D. Hannotin, N. Epain, A. Iortoman, and G. Kronauer, "Qualifying Timing Errors in Audio-over-Ethernet Networks for Live Sound," in *Audio Engineering Society Convention 160*, May 2026, Express Paper 427.
- [5] "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems," *IEEE Std 1588-2008 (Revision of IEEE Std 1588-2002)*, 2008.
- [6] M. Henry and W. Aubry, "Setup and first experimentation over an AES67 over 802.11 network," in *Audio Engineering Society Convention 146*, Mar. 2019, Engineering Brief 510.
- [7] A. Poets, I. Mergler, M. Penner, M. Waßmann, S. Preihs, and J. Peissig, "Professional Networked Audio over DECT NR+: Feasibility and Performance," in *2025 IEEE 6th International Symposium on the Internet of Sounds (IS2)*, 2025, pp. 1–8.
- [8] *DECT:2020 New Radio (NR); Part 1: Overview; Release 2*, ETSI TS 103 636-1, Rev. 2.2.1, May 2026.
- [9] A. Poets, M. Penner, S. Preihs, M. D. Pérez Guirao, and J. Peissig, "Cross-Domain Residence-Time Measurement for a PTP Transparent Clock over Non-Cellular 5G," in *2026 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control, and Communication (ISPCS)*, 2026, to be published.
- [10] R. Zarick, M. Hagen, and R. Bartoš, "The Impact of Network Latency on the Synchronization of Real-World IEEE 1588-2008 Devices," in *2010 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication*, 2010, pp. 135–140.
- [11] J. Han and D.-K. Jeong, "A Practical Implementation of IEEE 1588-2008 Transparent Clock for Distributed Measurement and Control Systems," *IEEE Transactions on Instrumentation and Measurement*, vol. 59, no. 2, pp. 433–439, 2010.
- [12] "Professional Media over Managed IP Networks: System Timing and Definitions," *SMPTE ST 2110-10:2022 (Revision of SMPTE ST 2110-10:2017)*, 2022.
- [13] "AES Standards Report - PTP parameters for AES67 and SMPTE ST 2059-2 interoperability," *AES-R16-2021 (Revision of AES-R16-2016)*, 2021.
- [14] T. Striffler and H. D. Schotten, "The 5G Transparent Clock: Synchronization Errors in Integrated 5G-TSN Industrial Networks," in *2021 IEEE 19th International Conference on Industrial Informatics (INDIN)*, 2021, pp. 1–6.
- [15] M. Penner. (2025, March) DECT-NR-Plus-SDR. License: AGPL-3.0. [Online]. Available: <https://github.com/maxpenner/DECT-NR-Plus-SDR>
- [16] Meinberg Funkuhren GmbH & Co. KG, *SyncBox N2X Manual*, Bad Pyrmont, Germany, September 2019.
- [17] A. Bondavalli. (2025) AES67 Linux Daemon. License: GPL-3.0. [Online]. Available: <https://github.com/bondagit/aes67-linux-daemon>
- [18] M. Henry, L. Rémond, and N. Sturmel, "Setting Up and Making an AES67 Network Coexist with Standard Network Traffic," in *Audio Engineering Society Convention 140*, Jun. 2016, Engineering Brief 241.