IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 14, 2020, accepted May 4, 2020, date of publication May 7, 2020, date of current version May 21, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2993204

Testbed Evaluation of Distributed Radio Timing Alignment Over Ethernet Fronthaul Networks

IGOR FREIRE[1], (Member, IEEE), IGOR ALMEIDA[2], (Member, IEEE), EDUARDO MEDEIROS[3], MIGUEL BERG[3], (Senior Member, IEEE), CHENGUANG LU[3], ELMAR TROJER[3], AND ALDEBARO KLAUTAU[1], (Senior Member, IEEE)

[1]LASSE–5G & IoT Research Group, Federal University of Pará, Belém 66075-750, Brazil
[2]Ericsson Research, Indaiatuba 13330-300, Brazil
[3]Ericsson Research, 164 80 Stockholm, Sweden

Corresponding author: Igor Freire (igorfreire@ufpa.br)

This work was supported in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), Brazil, under Finance Code 001, and in part by the Innovation Center, Ericsson Telecomunicações S. A., Brazil.

ABSTRACT This work investigates a mechanism for alignment of the timing on which spatially distributed and cooperative radio units transmit in radio-frequency (RF) when served over a packet-based fronthaul. It analyzes the problem by considering the imperfect clock synchronization of the radio units and the packet delay variation that fronthaul packets are subject to. Following the analysis, this paper proposes an implementation architecture for distributed RF transmission timing alignment based on synchronized triggering among radio units and centralized processing units. Throughout this discussion, special attention is given to the scheme's impact on the overall achievable fronthaul latency. Subsequently, this work discusses both hardware and software aspects of a prototype that was developed based on field-programmable gate arrays (FPGAs). In the end, it presents results obtained on an Ethernet fronthaul testbed where the referred FPGA-based prototypes implement radio units that are synchronized using the IEEE 1588 precision time protocol or by pulse-per-second references. Results validate the functionality of the proposed architecture and illustrate various relevant choices concerning system parameters.

INDEX TERMS 5G, clock synchronization, Ethernet, fronthaul, precision time protocol.

I. INTRODUCTION

Centralized and cloud radio access networks (C-RAN) have been widely investigated as key enablers for fifth-generation (5G) mobile communications [1]. An essential component of C-RAN is the fronthaul (FH) network, which has evolved substantially to meet 5G demands. While the FH of Long-term Evolution (LTE) deployments typically consists of dedicated, synchronous and point-to-point links between baseband unit (BBU) and remote radio units (RRU), in 5G the strongest trend is for packet-based and statistically multiplexed FH [2], [3]. Furthermore, there is wide acceptance of Ethernet as the main transport [4], using time-sensitive networking (TSN) features such as the ones standardized in IEEE 802.1CM [5] and encapsulation protocols such as eCPRI [6] and IEEE 1914.3 Radio over Ethernet (RoE) [7].

A common aspect of eCPRI and RoE is that they do not provide time and frequency synchronization, unlike previous generation FH protocols such as the Common Public Radio Interface (CPRI) [8]. Instead, timing references are expected to be distributed by the typical combination of global navigation satellite system (GNSS) disciplined clocks, the IEEE 1588 precision time protocol (PTP) [9] and Synchronous Ethernet [10]. As of today, with the recently approved IEEE 1588-2019 draft standard featuring the High Accuracy profile for sub-nanosecond accuracy [11], and with ongoing efforts from the Question 13 group of ITU-T study group 15, such as the enhanced synchronous Ethernet Equipment Clock (eEEC) [12], this combination remains the state-of-the-art.

Despite the extensive literature and availability of technologies to provide synchronization over a FH network, the actual usage of timing signals in the context of C-RAN equipment is not widely treated. Our previous work in [13], for instance, discusses how an RRU's timebase synchronized via PTP can be used to implement flow control and to synthesize coherent sampling and carrier frequencies. The present work, in turn, focuses on another essential form of

The associate editor coordinating the review of this manuscript and approving it for publication was Abbas Jamalipour.

synchronization, which is the alignment of the timing of radio transmissions.

The referred alignment can be achieved in terms of the timing of *radio frame* boundaries. For example, by aligning the start of the 10 ms frames of LTE or 5G new radio (5G-NR) [14] among the radio-frequency (RF) transmissions carried out by spatially distributed RRUs. Hence, the process is termed radio frame synchronization (RFS) in this text.

RFS is essential for distributed RRUs that cooperate on transmission or reception. For example, for inter-site carrier aggregation (CA), coordinated multi-point (CoMP) [15], [16], ranging-based localization [17], time-domain inter-cell interference coordination (ICIC) [18] and distributed multiple-input multiple-output (MIMO) [19]. Also, RFS is required between any pair of time division duplex (TDD) cells transmitting on the same frequency band with overlapping coverage areas in order to avoid interference between their downlink (DL) and uplink (UL) time slots [15], [16]. Each RFS use case has its accuracy requirement, although not necessarily standardized. For instance, inter-site CA requires a relative time alignment error (TAE) below 260 ns [20]. In contrast, TDD over LTE small cells requires the TAE to be less than 3 μs [21], a requirement that dates from Universal Mobile Telecommunications System (UMTS) [16] and continues for 5G-NR [22].

The difficulty of satisfying such stringent radio requirements with RFS is magnified by the asynchronous FH transport and asymmetrical FH delays. The FH packets experience packet delay variation (PDV) and traverse different paths from BBU to RRUs. Hence, the coordinated radio data sent for independent RRUs arrive in different instants. Ultimately, in the absence of an RFS mechanism, the corresponding on-air transmissions lack time alignment. Besides, while the so-called *playout buffers* can clean up the PDV of in-phase/quadrature (IQ) sample streams at the RRU [23], [24], they do not suffice for RFS. An additional mechanism is necessary when the goal is to time-align the IQ streams to be transmitted cooperatively by distributed RRUs.

To solve the problem, the BBUs and RRUs need to rely on a shared understanding of what time it is or on synchronized events. For example, the RRUs can align blocks of 100 LTE frames with their individual pulse-per-second (PPS) events [25], provided that the latter, in turn, are synchronized to a common reference. Alternatively, a BBU can continuously schedule when the RRUs should transmit IQ samples on air via the so-called *presentation time*, which is sent on FH frames of the RoE standard [7]. Similarly, BBUs and RRUs can rely on FH transmission and reception time windows in order to achieve synchronized IQ playout, as in eCPRI [6].

There are few experimental evaluations of timing alignment among cooperative RRUs and especially less concerning FH-driven schemes to align their RF transmissions. A range of studies for distributed beamforming and distributed MIMO investigate the provisioning not only of timing alignment but also of phase coherency among distributed transmitters, using testbeds based on field-programmable gate arrays (FPGAs) and software-defined radio (SDR) [26], [27]. Their solutions typically involve signal processing and over-the-air signaling between the coordinated transmitters [26] or between each transmitter and receivers [27]. However, they do not rely on the backhaul or FH network in order to achieve the timing alignment. Meanwhile, most testbed-based FH studies such as [28]–[30] focus on metrics like FH latency and PDV, rather than the RF timing alignment among cooperative RRUs. To the best of our knowledge, there are no publications focusing on practical evaluations of eCPRI's or RoE's mechanisms for timing alignment of distributed transmissions. While [1] briefly explains RoE's presentation time, it is from a theoretical standpoint only.

Nonetheless, the essence of the RFS problem appears in numerous contexts. For example, SDR applications commonly require that IQ samples generated by a central host are transmitted in RF simultaneously by network-distributed SDR boards. Thus, radio transport protocols such as the VITA 49 convey timestamps associated to the timing of IQ samples [31]. For audio and video streams, the Audio/Video Transport Protocol (AVTP) defined by IEEE 1722 [32] offers a similar mechanism for normalizing network latency and maintaining sample coherence among distributed "talkers" (e.g. coordinated speakers) [33]. More generally, RFS relates to distributed, time coordinated applications, which are well discussed in [34] in the context of TSN. A particular example of interest concerns applications where a controller commands time-coordinated actions to controlled devices in advance of their scheduled time, such as discussed in [35]. This is the principle that we exploit in this work, with the controlled action being the actual RF transmission.

In this work, we aim at contributing with an in-depth analysis of radio transmission timing alignment in the FH. We describe an RFS architecture based on synchronized triggering among BBUs and RRUs and discuss various challenging practical aspects, such as the impact of clock disciplining algorithms and the RFS scheme's contribution to the end-to-end FH latency. The latter is of particular interest, given the stringent FH latency budgets [36]. Moreover, we discuss a practical hardware implementation and present experimental results acquired with our FPGA-based prototypes under both PTP-based and PPS synchronization.

The work is organized as follows: Section II formulates the problem of distributed radio timing alignment; Section III describes the adopted RFS architecture; Section IV discusses the hardware and software of a prototype implementation; Section V gives an overview of our testbed; Section VI presents experimental results. Finally, Section VII summarizes the conclusions. For convenience, Table 1 lists the acronyms and abbreviations that are used throughout the text.

II. PROBLEM FORMULATION

The scenario of interest comprises a cluster of K cooperative RRUs served by a central BBU, where the BBU generates coordinated (e.g. jointly baseband-processed) radio data

TABLE 1. List of acronyms and abbreviations.

5G	Fifth-Generation
5G-NR	Fifth-Generation New Radio
AVTP	Audio/Video Transport Protocol
AxC	Antenna Carrier
BBU	Baseband Unit
BC	Boundary Clock
BG	Background
CA	Carrier Aggregation
CBRFL	Constant Bitrate Fixed-Length
CDC	Clock-Domain Crossing
CoMP	Coordinated Multi-Point
CPRI	Common Public Radio Interface
C-RAN	Centralized or Cloud Radio Access Network
DAC	Digital-to-Analog Converter
DL	Downlink
DMA	Direct Memory Access
E2E	End-to-End
eEEC	enhanced synchronous Ethernet Equipment Clock
EMAC	Ethernet Medium Access Control
FFT	Fast Fourier transform
FH	Fronthaul
FIFO	First-In First-Out
FPGA	Field-Programmable Gate Array
GbE	Gigabit Ethernet
GNSS	Global Navigation Satellite System
ICIC	Inter-Cell Interference Coordination
IFFT	Inverse Fast Fourier Transform
IQ	In-phase/Quadrature
LSB	Least Significant Bit
LTE	Long-Term Evolution
MAC	Medium Access Control (layer)
MIMO	Multiple-Input Multiple-Output
OCXO	Oven-Controlled Crystal Oscillator
PDV	Packet Delay Variation
PHY	Physical (layer)
PLL	Phase-Locked Loop
PPS	Pulse-per-Second
PTP	Precision Time Protocol
RF	Radio-Frequency
RFS	Radio Frame Synchronization
RoE	Radio over Ethernet
RRU	Remote Radio Unit
RTC	Real-Time Clock
SDR	Software-Defined Radio
TAE	Time Alignment Error
TC	Transparent Clock
TDD	Time Division Duplex
TIC	Time Interval Counter
TSN	Time-Sensitive Networking
UL	Uplink
UMTS	Universal Mobile Telecommunications System
USRP	Universal Software Radio Peripheral
UTC	Coordinated Universal Time
VLAN	Virtual Local Area Network
XO	Crystal Oscillator

streams for the RRUs. As illustrated in Fig. 1, the BBU delivers the coordinated streams to the RRUs over Ethernet frames and through transmission iterations. In the i-th iteration, it sends one Ethernet frame to each RRU k in order, starting from RRU 1 and ending at RRU K. Due to PDV and distinct paths to each RRU, these frames arrive asynchronously on the FH interface of the RRUs. Meanwhile, their content, once unpacked and processed by the RRUs, forms radio frames (such as 10 ms 5G-NR radio frames [14]) that must be aligned in time on the air interface. Hence, the content cannot be transmitted on air as soon (or as late) as received from the

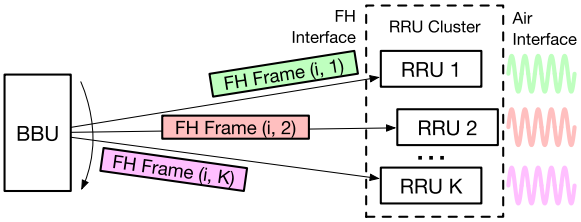

FIGURE 1. BBU transmission iteration: the central BBU delivers a round of FH (Ethernet) frames to each RRU in a cluster of RRUs.

FH. Instead, the RRUs must simultaneously trigger the on-air transmissions of coordinated radio frames.

Due to BBU's sequential delivery of Ethernet frames, in this work, we index Ethernet frames and associated metrics using tuple (i, k). Furthermore, we interchangeably refer to Ethernet frames that carry radio data as *FH frames*. These are neither to be confused with radio frames nor PTP frames. A single radio frame typically extends across hundreds of FH frames. For example, in LTE 20 MHz with 30.72 MHz sampling frequency, a 10 ms radio frame has 307, 200 samples. Meanwhile, each FH frame typically carries content corresponding to dozens or a few hundred samples. Besides, note that the format of the data carried by FH frames depends on the adopted functional split [37], but in general, we can assume that the RRUs derive IQ samples from them.

For RFS, the goal is to align the timing on which the radio frames go on air based on the timing of IQ samples. To this end, we must consider both FH delays and hardware latencies. Thus, we define end-to-end (E2E) delay $d_{i,k}$ as the delay measured from an internal point in the BBU hardware that handles RFS to a point in the hardware of the k-th RRU that is as close as possible to the antenna interface (c.f. reference point *Ra* in [6]). In contrast, delay $\tau_{i,k}$ denotes the FH transport delay experienced by frame (i, k), excluding BBU and RRU hardware latencies.

Fig. 2 illustrates the two delay definitions for a BBU that delivers radio data towards two RRUs. The BBU timestamps the departure of the first FH frame on instant **A**, but the effective departure occurs later on instant **B**, after the Tx hardware latency from **A** to **B**. Subsequently, the frame traverses the variable-delay FH and arrives at the RRU on instant **C**. The content of the frame is unpacked, buffered internally and ultimately reaches the RRU's RF interface on instant **D**. Thus, the FH delay $\tau_{i,1}$ is the interval from **B** to **C**, whereas the E2E delay of interest for RFS is the delay $d_{i,1}$ from **A** to **D**. Similarly, the E2E delay $d_{i,2}$ of the second FH frame (to RRU 2) is the delay measured from $\bar{\mathbf{A}}$ to $\bar{\mathbf{D}}$.

In Fig. 2, the RRUs can achieve RFS if the RRU 1 defers its RF transmission to instant $\bar{\mathbf{D}}$.[1] However, to do so, the first RRU would need to know the E2E delay $d_{i,2}$ from the BBU to RRU 2 during the i-th iteration. Since this would involve further communication between the RRUs, it is instead more practical for the BBU to collect FH delay information and

[1]In addition to coincident **D** and $\bar{\mathbf{D}}$, the content of the two frames must itself be aligned. We assume this is always the case, guaranteed by the BBU.

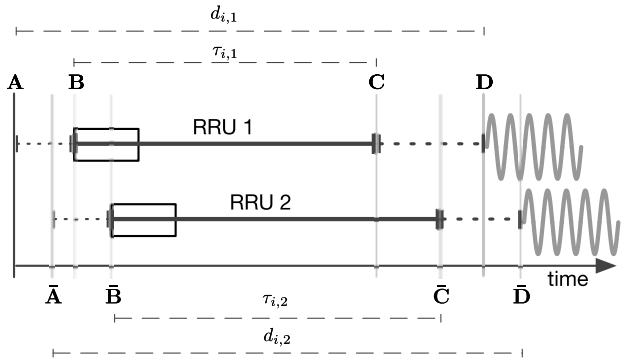

FIGURE 2. FH delay measurement instants for a BBU serving two RRUs.

schedule time-aligned RF transmissions to be executed by the RRUs. The BBU already communicates with all RRUs of the cluster. Hence, it can receive DL delay measurements taken and fed back by the RRUs or, when DL and UL delays are similar, it can measure the UL E2E delays directly.

In the approach of RoE [7], the BBU uses its knowledge of the individual E2E delays to define specific presentation times for radio data such that the FH delay differences from BBU to RRUs are normalized (c.f. *latency normalization* in [33]). The BBU sends this presentation time information within FH frames such that the RRUs become aware of when they should play IQ samples out. Although the overhead for this information is negligible, the approach that we consider in this work accomplishes latency normalization with no need for RRUs to become aware of the playout time. As will become apparent, synchronized triggers generated at the RRUs based on local clock time alone complete the work.

The drawback of the process of latency normalization is that it enlarges the delay of all FH frames to the maximum delay observed among them. Hence, this process penalizes the frames that otherwise would traverse the FH with less delay. Besides, for extra margin, the normalization may aim at a delay that is even higher than the maximum delay among FH frames. The motivation is to increase the probability of IQ samples being available on the RRUs when the time comes to play them out. Nevertheless, given the stringent FH latency requirements, such as the 100 μs budget mentioned in 802.1CM [5], it may be practically infeasible to assign too much margin for delays. Therefore, on practical FH deployments, it is critical for an RFS mechanism to incur the minimum (or a controllable) amount of latency.

III. PROPOSED ARCHITECTURE AND MODEL

The proposed RFS mechanism relies on triggers that are asserted periodically by the BBU and the RRUs of a cluster. Furthermore, it involves tight control of the timing on which the BBU delivers data to the RRUs, with corresponding buffering of data on the RRU side. In this section, we thoroughly explain this RFS architecture and the model of its associated parameters, such as the periodicity of the triggers and the required buffer depths. Furthermore, we analyze the

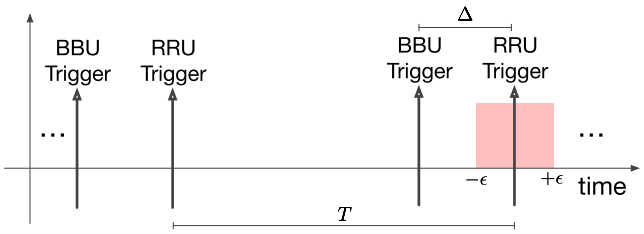

FIGURE 3. Periodic trigger events generated at BBU and RRUs.

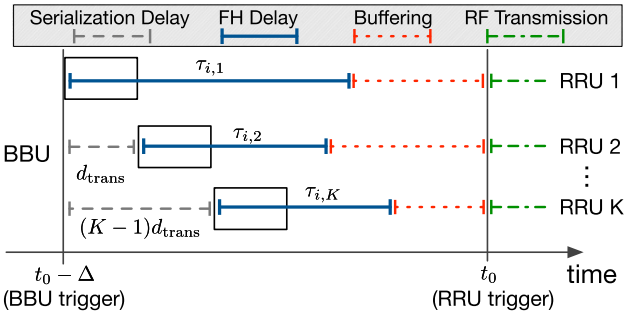

FIGURE 4. BBU transmission iteration through shared Ethernet interface with indication of arrival at the RRUs after variable FH delays, RRU buffering intervals and the serialization delays preceding each departure from the BBU.

scheme's impact on the FH latency and the main factors to consider when choosing or estimating parameters in practice.

A. TIME-COORDINATED TRIGGERING APPROACH

In the proposed architecture, the BBU and the RRUs of a cluster generate triggers once after every interval of duration T, as illustrated in Fig. 3. The trigger generated by an RRU marks the instant when it should start the transmission on air (i.e. in RF) of a new block of radio data. Thus, for RFS across distributed RRUs, all RRUs should strive to assert their triggers simultaneously. Meanwhile, on a BBU, the trigger indicates when it should start sending frames over the FH to the RRUs, such that all RRUs receive the coordinated radio data timely, i.e. before their triggers. Hence, the BBU triggers FH transmissions with time advance Δ relative to the RRU trigger time, as also illustrated in Fig. 3.

When the BBU asserts its trigger, it starts delivering a new *block* of radio data to the RRUs. In this work, we define a *block* as a collection of radio data with enough information for all served RRUs to produce IQ samples spanning the duration T of the trigger period. Once the BBU starts, the block progressively traverses the FH over FH frames. Each RRU receives a subset of the block and stores the IQ samples that derive from it in their buffers. By the time an RRU asserts its trigger, the first IQ sample derived from its subset of the block must be available in the buffer for RF transmission. On success, provided that the RRUs assert their triggers at nearly the same time, the RF transmissions are ultimately time-aligned. Fig. 4 illustrates this sequence for the first K FH frames carrying the start of a *block* delivered to K RRUs.

For example, under functional split E [6], a block consists of baseband IQ samples for J antenna carriers (AxCs) served

by the BBU. An AxC, in turn, consists of an IQ flow to be transmitted on or received from one carrier at one independent antenna element [8]. Assuming for simplicity that all AxCs served by the BBU run at the same sample rate, the block must contain the following number of IQ samples:

$$n_b = (J)\,(f_s)\,(T), \tag{1}$$

where f_s is the sample rate of the radio interface.

To generate triggers, we assume BBU and RRUs rely on the time count of their local real-time clocks (RTCs), which keep track of seconds and nanoseconds in hardware. Furthermore, we assume that the RTCs are continuously synchronized to a common timebase (e.g. using PTP). Hence, the corresponding triggers are also time-synchronized.

Note that although we are considering a single cluster of RRUs such as illustrated in Fig. 1, it is possible to extend the RF timing coordination to multiple clusters (served by distinct BBUs). This is feasible if all clusters synchronize to the same global time base, such as Coordinated Universal Time (UTC). Also, it requires that all RRUs target at the same trigger instant, such as the beginning of UTC seconds.

With that, there are two parameters to be defined next: the trigger interval T and the time advance Δ. The following subsections thoroughly discuss them.

B. TRIGGER INTERVAL

Two factors influence the choice of the trigger interval T. The first is that every trigger on the RRU side will lead to a realignment of the radio frame timing on air, so that interval T defines the periodicity of realignments. The second relates to when the triggering effectively happens. Since transmission timing updates can disturb the radio signal quality, an appropriate instant to handle realignments is at the boundary between radio frames, which is when mobile terminals restart their timing synchronization. Thus, it is reasonable to choose interval T as an integer multiple of the radio frame interval, e.g. a multiple of 10 ms for LTE and 5G-NR [14].

Additionally, in order for the RF timing realignment instant to coincide with the beginning of DL radio frames on air, the RFS system must be able to place the content corresponding to the start of radio frames at the start of RFS *blocks*. This alignment can be guaranteed by the BBU alone, with no intervention from RRUs. For example, under functional split E, the BBU can ensure that a block starts with the first J IQ samples corresponding to the start of a radio frame sent through J AxCs. The RRUs, in turn, only need to follow their triggers, i.e. to a start a new block on air when a trigger comes. Besides, the RRUs must know where in the incoming FH stream lies the beginning of a new block. We assume the BBU can assist with this task by flagging the start of a block within the FH frame's metadata or through auxiliary information such as sequence numbers.

In the end, a convenient choice for interval T is 1 second. One advantage is that a PPS signal is commonly available on timing equipment, such as GNSS receivers and IEEE 1588 clocks. Secondly, this choice simplifies the trigger generation logic. This is because the trigger can come directly from the least significant bit (LSB) of the seconds register of the RTC, which is simpler than detecting the passage of an arbitrary T in nanoseconds and suitable to avoid slow combinational logic when using high clock frequency RTCs. Third, this option is a multiple of the 10 ms radio frame interval.

C. TIME ADVANCE

As mentioned earlier, the BBU asserts its trigger with time advance Δ relative to the nominal triggering instant of the RRUs. This time advance is supposed to allow sufficient time for the FH packets sent by the BBU to arrive on all RRUs before the RRU trigger instant. Hence, it must consider the largest expected E2E delay from BBU to RRUs. Additionally, the time advance must tolerate the worst-case clock time offset, which is the offset that leads to an RRU triggering the earliest, namely when the RRU's local time is mostly ahead of the reference time (with a positive time offset).

Moreover, when the BBU has a single outbound Ethernet interface, the advance Δ must also account for FH frame transmission (serialization) delays. Fig. 4 includes such serialization delays on a scenario where a BBU serves $K = 3$ RRUs. Note the FH frame sent to RRU 1 is the one that departs when the time-advanced trigger is asserted at the BBU (at time $t_0 - \Delta$). Since this frame has serialization delay d_{trans}, the next frame (to RRU 2) departs only after delay d_{trans} and so on until the frame to RRU K departs $(K-1)d_{\text{trans}}$ time units after the time-advanced trigger. For simplicity, in this figure and the remainder of this work, we assume that all FH frames sent by the BBU have the same length and, consequently, the same transmission delay. Hence, in the formulations that follow, we consider that the DL FH frame sent to the k-th RRU departs after delay $(k-1)d_{\text{trans}}$.

In Fig. 4, we also assume that the BBU has the full radio content of all K FH frames of the transmission round by the time it triggers at time $t_0 - \Delta$. Hence, the BBU can serve the round of frames (one to each RRU) in a row, such that d_{trans} is the only interval separating frame transmissions of the same delivery round. This is not a general requirement, but it is how the hardware that we evaluate in this work operates, as our BBU prototype processes all J served AxCs in parallel.

Once transmission delays are accounted for, the time advance that is ultimately required at the BBU becomes:

$$\Delta = (K-1)d_{\text{trans}} + \epsilon + \max_{k\in\mathcal{K}}\{d_k\}, \tag{2}$$

where we define $d_k = \max_i\{d_{i,k}\}$, i.e. as the maximum E2E delay (in DL) of frames destined to the k-th RRU over the realizations i. Consequently, term $\max_{k\in\mathcal{K}}\{d_k\}$ is the maximum observed E2E delay among all RRUs, while $\mathcal{K}$ denotes the set of all K RRUs. Meanwhile, term ϵ represents the worst-case time offset at an RRU relative to the reference time. As illustrated in Fig. 3, we assume for simplicity that the RRU trigger always remains within $\pm\epsilon$ relative to the nominal trigger instant.

The computation of (2) considers the worst case in terms of delay, which is when the maximum E2E delay lies in the path to the last RRU of the serving round, namely in the path that has the latest departure when considering a shared outbound interface at the BBU. Correspondingly, this computation represents the worst-case time advance, which can be inefficient. One strategy to minimize the required time advance is to sort the serving of RRUs in descending order of E2E delays, i.e. RRU with the highest delay served first. In this case, assuming the ordered set of RRUs is $\mathcal{C}$, the time advance required for the k-th RRU becomes:

$$\Delta_k = (k-1)d_{\text{trans}} + \epsilon + d_k. \quad \forall\, k \in \mathcal{C} \tag{3}$$

In the model of (3), term $(k-1)d_{\text{trans}}$ grows with k while d_k decreases (given $k \in \mathcal{C}$), such that one compensates the other. Furthermore, in contrast to (2), the model in (3) provides the individual time advance required for each RRU. One possible implementation is to authorize the departure of each frame (to each RRU) of the serving round individually while respecting such distinct values. Alternatively, a sub-optimal global time advance can be used, as given by:

$$\Delta = \max_{k \in \mathcal{C}} \{\Delta_k\}, \tag{4}$$

which can be shown to be less than or equal to Δ from (2).

Lastly, note that the RRUs could also employ a similar time advance strategy to normalize their latencies from digital IQ output to RF transmission. For instance, to normalize the latencies of the paths that cross the digital-to-analog converter (DAC), analog RF filtering, and cabling, such as coaxial feeders. However, this process can be demanding, e.g. based on manual calibration. In this work, since our RRUs are composed of nominally identical hardware, we assume that the ''last inch'' latency (c.f. [38]) does not compromise the relative time alignment among RRUs and, therefore, we neglect the need for time advance on RRU side.

D. LATENCY DUE TO TIME ADVANCE

Due to stringent FH latency budgets, it is critical to analyze the extra latency that the time advance procedure imposes. It must be noted that the time advance mechanism does not correspond entirely to additional delay. A significant portion of its value corresponds to delay that would exist regardless. In (3), for example, term d_k typically represents the major part of the time advance and is meant to guard against the intrinsic FH PDV from BBU to this k-th RRU alone. The latency incurred by this part of the time advance is the difference between the actual delay $d_{i,k}$ of frame (i, k) and the maximum observed delay d_k. This difference is equivalent to the latency that a de-jitter buffer would introduce in order to obtain a PDV-free stream towards the DAC (c.f. [24]).

With the latency $(d_k - d_{i,k})$ of a de-jitter buffer as a baseline, note term ϵ in (3) is what determines the latency that is incurred specifically by the RFS mechanism. When using the time advance of (4), in contrast, the delay $d_{i,k}$ of frame (i, k) is extended (through buffering) based on the maximum observed delay among all RRUs, which can result in more latency than that of a de-jitter buffer. Also, term $(k-1)d_{\text{trans}}$ in (3) can introduce additional latency when (4) is adopted.

One solution to alleviate the latency imposed by the RFS mechanism is to shorten d_{trans} and ϵ. The transmission delay d_{trans} reduces by making FH frames shorter and more frequent, at the expense of increased overhead in the Ethernet links, or by deploying higher Ethernet line rates. Meanwhile, the time synchronization error ϵ decreases by provisioning better clock synchronization accuracy to the RRUs, e.g. with better algorithms or better timing support from the network.

E. BUFFER DESIGN

A typical Ethernet-based RRU continuously stores the data that it receives from the BBU into buffers. These buffers are the aforementioned de-jitter (or playout) buffers, which accommodate FH delay variations. However, in the case of the RFS architecture, the same buffers are also involved in the time-triggered processing approach. While waiting for a trigger, an RRU disables the reading of FH data from the buffers. Then, once a trigger finally comes, it resumes the processing of buffered data in order to handle RF transmissions. Hence, the buffer sizes must be appropriate to accommodate all samples that arrive while an RRU is waiting for a trigger.

In contrast to the time advance Δ, which considers the worst-case delay, the RRU buffer size should support the quickest path, i.e. the best case in terms of E2E delay. This is because an RRU buffers the IQ samples that derive from an incoming FH frame until the target playout time, which is when the RRU asserts its trigger. Hence, the RRU on the path with the earliest arrival experiences the longest buffering.

Various FH properties are relevant in order to model the required buffer depths. For example, signal compression, flow control and functional split. For tractability and fidelity to our hardware, we assume again that the BBU operates with functional split E and that it feeds uncompressed samples into the FH with a controlled average rate of $R = (J \cdot f_s)$ samples per second. This rate corresponds to the number of samples that the K RRUs collectively put on air per second, specifically by reading IQ samples from J buffers (one for each AxC) on every sample period.

Hence, the following buffer depth (in units of samples) suffices on all RRUs to store the IQ samples of a single AxC:

$$L = \left(\Delta - \min_{k \in \mathcal{K}} \{\bar{d}_k\}\right)(f_s)\,, \tag{5}$$

where $\bar{d}_k = \min_i \{d_{i,k}\}$, i.e. the minimum E2E delay for the k-th RRU over the realizations i, while Δ is a global time advance either from (2) or (4). The rationale is that Δ defines the target E2E latency from BBU departure to RF transmission. The RRU on the shortest path, however, originally had a minimum E2E delay of $\min_{k \in \mathcal{K}} \{\bar{d}_k\}$, such that now, with RFS, it may need to buffer the data for up to $\left(\Delta - \min_{k \in \mathcal{K}} \{\bar{d}_k\}\right)$ time units until the trigger. By multiplying this interval with f_s, then, one obtains the corresponding number of IQ samples to be buffered for one AxC. We assume that an RRU running

multiple AxCs would have one buffer capable of holding L samples for each AxC.

In practice, (5) is slightly excessive because it neglects transmission delays d_{trans} and determines a global requirement for all RRUs. Nevertheless, there is much less penalty in adopting oversized buffers compared to e.g. the latency penalty of an exaggerated time advance. Hence, the expression provides a reasonable guideline for hardware design.

F. TIME ADVANCE ESTIMATION

The theoretical time advances from (2) or (3) are unknown in practice. Nevertheless, they can be estimated in real-time based on delay measurements. In the sequel, we discuss estimation approaches based on both UL and DL measurements.

1) ESTIMATION BASED ON UPLINK DELAYS

We assume that FH frames carry their departure timestamp, analogously to PTP one-step mode [9]. Thus, the BBU measures the E2E delay of UL frame (i, k) as follows:

$$\tilde{u}_{i,k} = t_{i,k}^{\text{rx,bbu}} - t_{i,k}^{\text{tx,rru}}, \tag{6}$$

where $t_{i,k}^{\text{tx,rru}}$ is the time of departure from the k-th RRU and $t_{i,k}^{\text{rx,bbu}}$ is the corresponding arrival time at the BBU.

Since these timestamps originate from the independent RRU and BBU clocks, the delay measurement is impaired by their time-varying time offset $x_{i,k}$, which is given by:

$$x_{i,k} = t_{i,k}^{\text{tx,rru}} - t_{i,k}^{\text{tx,bbu}}, \tag{7}$$

where $t_{i,k}^{\text{tx,bbu}}$ represents the BBU's (reference) time exactly when the RRU takes the departure timestamp $t_{i,k}^{\text{tx,rru}}$. Hence, it follows that (6) in reality measures:

$$\begin{aligned}\tilde{u}_{i,k} &= t_{i,k}^{\text{rx,bbu}} - \left(t_{i,k}^{\text{tx,bbu}} + x_{i,k}\right) \\ &= u_{i,k} - x_{i,k},\end{aligned} \tag{8}$$

where $u_{i,k}$ is the true E2E delay of UL frame (i, k). Correspondingly, the maximum UL delay measurement taken up to the i-th iteration is modeled by:

$$\tilde{u}_k = \max_i\{\tilde{u}_{i,k}\} = \max_i\{u_{i,k} - x_{i,k}\}. \tag{9}$$

For simplicity, it can be assumed that $\tilde{u}_k$ converges to $u_k - \min\{x_{i,k}\}$, where $u_k = \max_i\{u_{i,k}\}$. That is, it converges to the true maximum UL E2E delay, biased negatively by the minimum time offset experienced over the observation window. Thus, by further assuming that DL and UL E2E delays are symmetric (i.e. $u_k \approx d_k$), and by substituting $u_k \approx \tilde{u}_k + \min\{x_{i,k}\}$ in place of the maximum theoretical DL E2E delay d_k in (3), one can infer that a time advance estimator based on UL measurements should satisfy:

$$\tilde{\Delta}_k^{\text{ul}} \geq (k-1)d_{\text{trans}} + \epsilon + \tilde{u}_k + \min_i\{x_{i,k}\}. \tag{10}$$

In (10), term ϵ represents the maximum time offset affecting the clock used for RFS triggering on the k-th RRU. In contrast, term $\min_i\{x_{i,k}\}$ represents the minimum time offset experienced by the clock that provides UL departure timestamps for delay measurement. In typical RRUs, the two clocks are the same. Nevertheless, the RRU prototype that we explore in this work can use distinct clocks. Hence, for generality, we assume $x_{i,k}$ ranges within $\pm\gamma$, such that the following estimator satisfies the inequality:

$$\tilde{\Delta}_k^{\text{ul}} = (k-1)d_{\text{trans}} + \epsilon + \gamma + \tilde{u}_k. \tag{11}$$

This estimator assumes the worst-case realization of $\min_i\{x_{i,k}\}$, which is when $x_{i,k}$ remains constant at its maximum specified value $+\gamma$ throughout the observation.

On RRUs that use the same RTC for timestamping and RFS triggering, note $\gamma = \epsilon$, so that (11) reduces to:

$$\tilde{\Delta}_k^{\text{ul}} = (k-1)d_{\text{trans}} + 2\epsilon + \tilde{u}_k. \tag{12}$$

Finally, similar to (4), one can adopt the sub-optimal maximum time advance estimate within a set $\mathcal{C}$ of RRUs in descending order of UL E2E delays:

$$\tilde{\Delta}^{\text{ul}} = \max_{k\in\mathcal{C}}\{\tilde{\Delta}_k^{\text{ul}}\}. \tag{13}$$

2) ESTIMATION BASED ON DOWNLINK DELAYS

Following the same analysis of (8), note that the time offset $x_{i,k}$ disturbs DL E2E delay measurements as follows:

$$\tilde{d}_{i,k} = d_{i,k} + x_{i,k}. \tag{14}$$

Hence, the time advance can be estimated by:

$$\tilde{\Delta}_k^{\text{dl}} = (k-1)d_{\text{trans}} + \tilde{d}_k, \quad \forall\, k \in \mathcal{C} \tag{15}$$

where $\tilde{d}_k = \max_i\{\tilde{d}_{i,k}\}$, i.e. the maximum DL E2E delay measurement over realizations i. This estimator approaches:

$$\tilde{\Delta}_k^{\text{dl}} \approx (k-1)d_{\text{trans}} + d_k + \max_i\{x_{i,k}\}. \tag{16}$$

For typical RRUs that use the same RTC for RFS triggering and timestamping of FH frames (such that $\gamma = \epsilon$), this tends to be a sufficient time advance, as it compensates the maximum time offset ever experienced by the k-th RRU. Besides, in the long term, $\max_i\{x_{i,k}\}$ may converge to ϵ, in which case the estimator converges to (3).

This analysis reveals that, in order to estimate the time advance based on DL delay measurements, it may not be necessary to know the maximum expected time offset of the RRU in advance. Nevertheless, this relies on certain simplifications that may not hold well in practice, such as that $\tilde{d}_k$ approaches $d_k + \max_i\{x_{i,k}\}$ promptly. Thus, an extra margin can be added to the time advance estimate from (15).

IV. PROTOTYPE IMPLEMENTATION

A. FPGA BLOCKS AND OPERATION

Fig. 5 presents a simplified view of the hardware developed for the FPGA-based BBU and RRU prototypes that we evaluate in this work. The so-called *RFS controller* block is present on both devices. On the BBU, this block regulates the timing of Ethernet frame transmissions towards the RRUs. On the

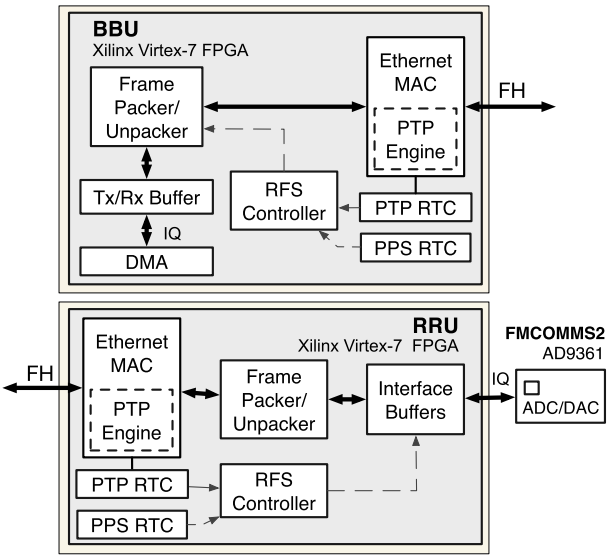

FIGURE 5. Block diagrams illustrating the BBU and RRU FPGA design.

RRU, in turn, it controls when IQ samples are delivered to the DAC of the RF frontend for on-air transmissions.

The operation is such that the *Frame Packer* block of the BBU continuously forms Ethernet frames with IQ data acquired via a direct memory access (DMA) engine. On every interval T, this block has to wait until the RFS controller asserts its time-advanced trigger before releasing frames downstream. Once asserted, the trigger grants permission for transmitting a block of n_b IQ samples over the FH, over as many Ethernet frames as necessary to convey them all.

Meanwhile, at the RRU, the *Frame Unpacker* block continuously extracts IQ samples from incoming FH frames. However, these samples are not immediately made available to the DAC for on-air transmission. Instead, they are only provided when the RFS controller determines so. On every interval T, the RRU's RFS controller will feed n_b/J IQ samples to the DAC for each of its AxCs.

As mentioned in Section III-E, we assume the BBU feeds $R = (J \cdot f_s)$ samples per second into the FH. In our implementation, it does so by executing periodic serving rounds, as illustrated in Fig. 6. After its time-advanced trigger, the BBU delivers fixed-length frames to each RRU until it completes the delivery of all the data spanning the trigger period T. More specifically, the BBU sends a single frame to each RRU on every interval i_{tx} and each such frame carries $(i_{\text{tx}} \cdot f_s)$ samples for each AxC. As a result, the BBU achieves rate R on average. Also, if the frame transmissions finish in less than i_{tx}, the BBU leaves the link idle until the next interval i_{tx}, as illustrated in Fig. 6. In [39], we name this traffic pattern as constant bitrate fixed-length (CBRFL) FH traffic.

Lastly, in addition to IQ samples, each FH frame includes metadata containing the frame's departure timestamp and a sequence number. The timestamp allows the receive-end to measure the frame's E2E delay. The sequence number,

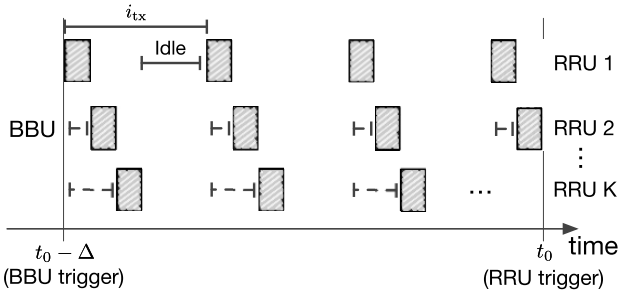

FIGURE 6. Model of BBU frame departures starting from the BBU trigger.

in turn, enables verification of frame sequence integrity on tests.

B. TIME SOURCES

The FPGA design includes two sources of time. The first is a PTP-synchronized RTC, labeled as PTP RTC in Fig. 5, which is disciplined based on PTP timestamps that are taken in hardware[2] by the PTP engine of the Ethernet MAC (EMAC). The second is a self-developed RTC that synchronizes to a PPS reference signal, labeled for brevity as the PPS RTC.

In our design, the software can assign different time sources to independent blocks, such as the *RFS controller* and the *frame packer/unpacker*. Thus, we can control which RTC provides departure and arrival timestamps for FH frames. Also, we can evaluate the RFS scheme under both PTP and PPS synchronization. An evaluation based on the PPS RTC can indicate the RFS performance when the synchronization accuracy and stability are better than achieved by PTP over typical timing-aware networks, such as the case of RRUs featuring GNSS-disciplined clocks.

C. INTERFACE BUFFERS

A relevant part of the implementation concerns the RRU *interface buffers*, which are shown in Fig. 5 and detailed in Fig. 7. They consist of two layers of first-in first-out (FIFO) buffering between the *Frame Unpacker* (in DL direction) and the DAC. The first buffering stage stores samples that arrive from the FH and maintains these samples until an RFS trigger. After a trigger, the RFS controller authorizes the transport of samples from the first stage to the second. The second stage, in turn, is used for clock-domain crossing (CDC) from the internal clock domain (on its write-side) to the external DAC's clock domain (on its read-side). This dual-buffer circuit repeats for each I and Q component of each AxC since the DAC reads theses components through parallel lanes. Thus, e.g. for 2 AxCs, there are four circuits.

One motivation for this approach is simplicity in terms of CDC. Since the RFS controller regulates the path between the two FIFOs, the scheme allows the RFS controller to operate entirely within the internal clock domain. More importantly, however, it offers flexibility for dealing with problems that

[2]Timestamps are taken when the start of frame delimiter is observed on a reference plane of the EMAC that lies after fixed PHY and MAC latencies.

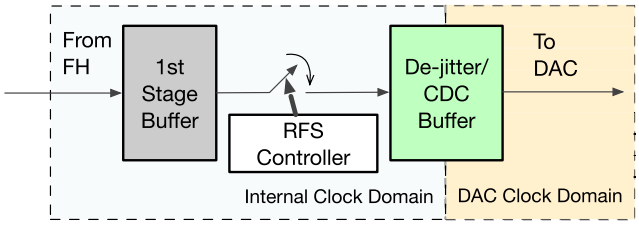

FIGURE 7. Dual-buffer scheme used for RFS control in the RRU.

originate from fluctuations (or jitter) in the RFS trigger instant. We elaborate on this in the following subsection.

D. SOLUTIONS FOR TRIGGER FLUCTUATIONS

In the proposed mechanism, the BBU and the RRUs of a cluster periodically assert a trigger signal based on their local RTCs. However, since the RTCs are continuously time-disciplined, one can expect that the interval between consecutive triggers oscillates and deviates from the nominal interval T. Additionally, on the BBU end, when the time advance is continuously adapted, the time-advanced trigger fluctuates accordingly. Such fluctuations can create two disturbing scenarios: when the interval between consecutive triggers is significantly lower than the nominal interval T and the opposite scenario, when significantly higher than T.

The remainder of this section explains that the interval $\Lambda[n]$ between the n-th trigger and the preceding trigger becomes problematic whenever:

$$|T - \Lambda[n]| > T_s, \tag{17}$$

where T_s is the sample period (equal to $1/f_s$). That is, $\Lambda[n]$ is problematic when it deviates from the nominal interval T by more than the sample period T_s.

One way to avoid this condition is to adopt smooth RTC time corrections, i.e. corrections that do not exceed T_s in the course of any interval of duration T. Otherwise, other solutions may be applied, as discussed next.

1) DROPPING OF DELAYED IQ SAMPLES

When an RRU experiences a shorter trigger interval such that $\Lambda[n] < T - T_s$, one can expect that it does not complete the transmission in RF of the samples (n_b/J per AxC) corresponding to this n-th interval. On the other hand, since a trigger should mark the start of a new block of samples on air, when a new trigger comes, any sample of the previous block that has not been transmitted in RF yet can be deemed as delayed. To ensure that the RF transmission timing can be realigned among RRUs on every trigger, an RRU must be capable of discarding such delayed samples.

To detect delayed samples, the RFS controller monitors the path between the two buffers of Fig. 7 and counts the samples that traverse it. If a new trigger comes and the count is still less than n_b/J for each AxC, the controller discards the remaining samples by popping them quickly from first-stage buffers without writing them to second-stage (CDC) buffers. This mechanism is particularly feasible because in our hardware the path between the two buffer stages operates with a clock of 100 MHz, whereas the DAC reads samples from the CDC buffers with rate f_s, which is much lower than 100 MHz in our tests. Thus, delayed samples can be dropped quickly without compromising the delivery of new samples (of the new block) to the CDC buffers.

This sample dropping scheme is also only achievable because our hardware controls the maximum occupancy that is allowed for the CDC buffers. For example, by specifying a maximum occupancy of 2 samples, at any point in time all the remaining samples are held back at the first-stage FIFOs. Consequently, if a new trigger comes and they are still there waiting to go on air, the RFS controller can still drop them. In our hardware, we set the maximum occupancy to 3 samples on all CDC FIFOs. This is the minimum that we can achieve with the dual-clock FIFOs that we adopt for CDC buffers.

An alternative to achieve the same re-synchronization behavior while not constraining the CDC buffer occupancy would be to reset the CDC buffers entirely on every trigger. By doing so, the samples of the new block would find these FIFOs empty and go immediately on air. However, this approach faces problems with the resetting of FIFOs. On dual-clock FIFOs, there is usually some time to propagate the reset into the two clock domains. As a result, there may be one (or a few) clock cycles in which the buffers remain empty and starve the DAC, which is undesirable.

Besides, a similar problem of delayed samples can arise on the BBU end due to short trigger intervals. When the BBU increases its time advance, the new (more time-advanced) trigger can come before the BBU finishes transmission of the past data block. In this scenario, the BBU may either drop the delayed data of the preceding block while starting the new block immediately or burst the delayed data such that it catches up without data loss. Our current version of the BBU prototype does not feature such a catch-up mechanism and, hence, we do not evaluate this issue in this work.

Our BBU prototype does support the adaptation of the time advance in real-time based on UL E2E delay measurements. This process is seamless in most experiments because the BBU normally finishes block data transmissions slightly earlier than T, i.e. earlier by $(i_{tx} - Kd_{\text{trans}})$, which is the idle interval shown in Fig. 6, such that there is room for time advance adaptation. In contrast, in the specific experiments where the magnitude of the adaptation is larger than the referred idle interval we choose a fixed time advance instead.

2) CDC BUFFER UNDERFLOW

When an RRU experiences a longer interval between consecutive triggers such that $\Lambda[n] > T + T_s$, the CDC buffers can underflow while waiting for the new trigger. One solution is to allow a small number of extra samples (in addition to n_b/J samples per AxC) to traverse from first-stage buffers to CDC buffers during initialization. This way, if a trigger eventually comes too late, there is some backup of samples to avoid underflow. This strategy can be acceptable as long as

all RRUs back up the same number of samples. Nevertheless, since the backup samples remain in the CDC buffers, they are not amenable to dropping. Hence, they can disturb the re-synchronization performance.

In this work, we choose instead to live with CDC underflows. The rationale is that we expect CDC FIFOs to remain empty only briefly when trigger fluctuations cause the underflow. Consequently, we expect the resulting gap on RF transmissions to be negligible for timing alignment.

E. EXPECTED BUFFER OCCUPANCY LEVELS

With the given mechanisms and their dependence on the buffering scheme, it becomes natural to evaluate the system based on buffer occupancy counts. Thus, we clarify the expected occupancy levels in the sequel.

In normal conditions, one can expect the occupancy of the first-stage buffers to be continuously non-zero and the occupancy of CDC buffers to be less than or equal to their maximum allowed occupancy (of 3 samples in our hardware). If the time advance is sufficient, the first-stage buffers should never underflow, meaning they should always hold the next samples to go on air with some margin for clock and FH frame delay variations. Meanwhile, the CDC buffers should only approach unitary occupancy slightly before the occurrence of a new trigger, which is when the RRU finishes transmission of the n_b/J samples of each AxC. In this clock cycle, at the same time that the DAC processes the last sample of the past block, the first IQ sample of the subsequent block of samples should be written in the CDC buffer.

When the first-stage FIFOs experience underflow, the RRU can infer that the time advance has become insufficient. Furthermore, on this occasion, one can expect that the CDC FIFOs will also experience underflow. This is especially true because, as explained in Section IV-D.1, the CDC FIFOs are maintained with very low occupancy in order to ensure that all samples are held back at the pre-RFS FIFOs. In contrast, in case the CDC FIFOs experience underflow while the first-stage buffers do not, one can infer the occurrence of undesirable fluctuations in the RRU trigger instant.

F. SOFTWARE STACK

In our prototype, the software stack runs on a Xilinx MicroBlaze microprocessor core within the FPGA programmable logic. The stack drives all functionalities of the BBU and RRU prototypes, such as PTP, PPS synchronization, FH framing and RFS. The RFS drivers, in particular, are responsible for the real-time tuning of the time advance Δ and configuration of the number of samples n_b (or n_b/J) that the RFS controller should count per interval T.

The real-time tuning of the time advance relies on a hardware register that records the maximum E2E delay ever observed (based on all incoming frames). The software of the BBU periodically fetches the maximum registered UL E2E delay measurement and resets the register, such that it regularly reads the maximum over an observation window. It then uses this value as $\tilde{u}_k$ in (11) in order to estimate the individual time advance $\tilde{\Delta}_k^{\text{ul}}$ for the k-th RRU. Lastly, it adopts the global time advance $\tilde{\Delta}^{\text{ul}}$ from (13). By doing so, the time advance adapts to instantaneous FH conditions.

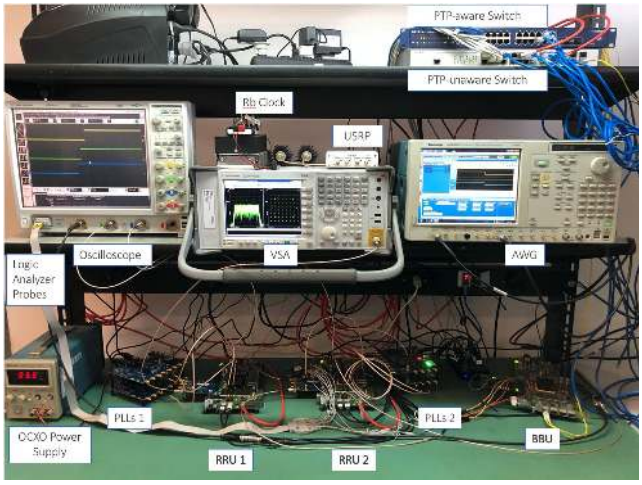

FIGURE 8. Picture of the testbed showing FPGAs and measurement equipment.

V. TESTBED

A. HARDWARE COMPONENTS

Our testbed is composed of one BBU and two RRUs, all of them implemented on Xilinx Virtex-7 FPGAs. Each RRU attaches to an Analog Devices FMCOMMS2 RF frontend board. Moreover, each FPGA is equipped with a Xilinx FMC XM105 Debug Card, through which we expose some hardware signals for logic analysis on an Agilent Infiniium MSO 9104A oscilloscope. Fig 8 shows a real picture of the testbed. Some devices on it are out of scope, but the reader can find further information in [13].

Note in Fig. 8 that each RRU is associated with Analog Devices AD9548 phase-locked loop (PLL) boards. Among other roles, these PLLs allow us to choose the oscillator that drives both PTP and PPS RTCs. More specifically, our RRUs can either run their RTCs based on a crystal oscillator (XO) or a more stable and accurate oven-controlled XO (OCXO).

B. FRONTHAUL, NETWORKING AND TIMING SETUP

Regarding FH transport, the testbed supports functional split E and transports any arbitrary sequence of baseband samples that is pre-filled on the BBU's memory. The testbed's Ethernet network, in turn, comprises gigabit Ethernet (GbE) links. Thus, despite the flexibility on waveforms and sample rates, to comply with the capacity offered by GbE under split E, we choose to evaluate RFS in this work by transporting LTE 5 MHz AxCs over the FH. More specifically, a total of 4 AxCs, 2 for each RRU, since each RRU features 2 RF outputs.

Fig. 9 shows the network topology used for the evaluation. Our BBU always delivers frames in sequence, starting from RRU 1 and then proceeding to RRU 2, as illustrated in Fig. 1. Hence, with RRU 1 on the longest path (after four hops), we guarantee the descending E2E delay order required for (3).

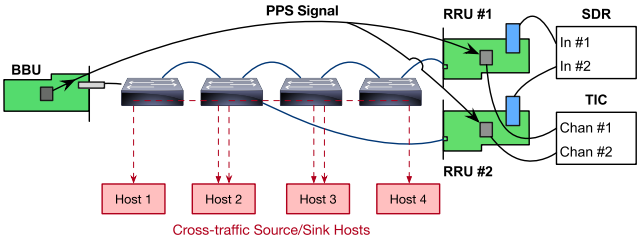

FIGURE 9. Network topology used for the evaluation.

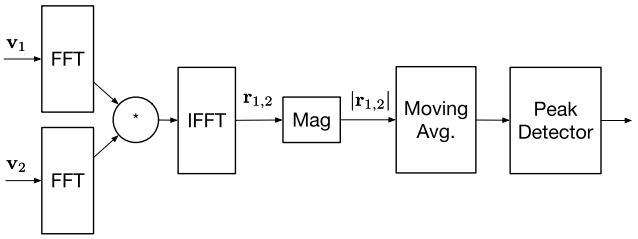

FIGURE 10. Processing chain used to compute the cross-correlation between the RF signals transmitted by each RRU and received by the USRP.

In terms of PTP support, our testbed network can be configured both as PTP-unaware and PTP-aware. When configured as PTP-unaware, all hops of the network consist of independent port-based virtual local area networks (VLANs) with legacy switching, that is, without any boundary clock (BC) or transparent clock (TC) functionality [9]. In this case, PTP messages are handled solely by the endpoints (BBU and RRU). In contrast, when configured as PTP-aware, all hops consist of VLANs with peer-to-peer TC functionality. In both cases, FH and PTP traffic share the same links of the network.

Lastly, the testbed supports the generation of background (BG) traffic, i.e. traffic other than PTP and FH. To do so, as shown in Fig. 9, it uses four hosts. The first host injects BG traffic on the first switch. Host 2 then extracts this stream and injects another stream towards Host 3. The same scheme repeats until Host 4 consumes the BG stream generated by Host 3. In the end, the BG traffic follows the same path as PTP Sync messages (and DL FH frames), which is referred to as *cross-traffic* in the synchronization literature [40].

Furthermore, the hosts can similarly generate BG traffic in the UL direction. In both directions, we adopt the Network Traffic Model 1 of ITU-T Recommendation G.8261 [41]. Nevertheless, by default, and unless we explicitly mention otherwise, cross-traffic is disabled on experiments.

C. MEASUREMENT AND DATA ACQUISITION TOOLS

As illustrated in Fig. 9, we connect one RF output (one AxC) of each RRU to the two independent inputs of a Universal Software Radio Peripheral (USRP) SDR interface model B210 (shown in Fig. 8). The RRUs transmit LTE 5 MHz signals over 2.45 GHz carriers. The USRP, in turn, downconverts these signals and feeds the resulting complex baseband sample streams with rate $f_s = 7.68$ Msps to a PC.

The PC that is attached to the SDR runs a self-developed application based on GNU Radio with the processing blocks that are illustrated in Fig. 10. This application computes the average cross-correlation between the two inputs (each sampled at 7.68 Msps) in real-time and finds the lag of the correlation's peak. The rationale is that, in order to test RFS, we configure the BBU to send identical waveforms through all AxCs. Hence, the cross-correlation successfully captures the timing difference between the RF transmissions.

The scheme of Fig. 10 computes the cross-correlation in the frequency domain. Assuming that $\mathbf{v}_1$ and $\mathbf{v}_2$ are vectors with N baseband samples from SDR input channels 1 and 2, respectively, it computes the cross-correlation vector $\mathbf{r}_{1,2}$ as:

$$\mathbf{r}_{1,2} = \mathcal{F}^{-1}\left\{\mathcal{F}\{\mathbf{v}_1\}\,\mathcal{F}\{\mathbf{v}_2\}^*\right\}, \tag{18}$$

where $\mathcal{F}$ denotes a fast Fourier transform (FFT), $\mathcal{F}^{-1}$ denotes the inverse FFT (IFFT), operator * denotes complex conjugation and the multiplication of FFTs is element-wise.

Next, the application computes the moving average of $|\mathbf{r}_{1,2}|$ (of the magnitude) and detects the peak of the resulting average. The index of this peak is the result of interest, as it represents the timing difference between the two inputs in units of sample periods. In the end, we collect the average peak index among all peaks computed in the course of a second. With 7.68 Msps and FFT length of 2048, this average extends across 3750 peak indexes computed over a second.

In addition to the SDR, Fig. 9 shows that both RRUs are connected to a time interval counter (TIC), more specifically a Keysight 53220A. The signal that the FPGA feeds to the TIC is a synchronized 8 kHz rectangular wave, which is generated in hardware based on the PTP RTC. If the PTP synchronization was perfect, the edges of the waves produced by both RRUs would be perfectly aligned. Correspondingly, by observing the time interval between the edges of the two waves, one obtains the relative time offset between the RRUs. This is the interval that is continuously measured by the TIC.

Lastly, a Python-based application was developed to aggregate results from all devices and applications. It collects metrics from the three FPGAs; the average cross-correlation peak index from the GNU Radio SDR application; and time offset measurements taken by the TIC. The next section discusses results collected by this application.

VI. RESULTS

This section presents results obtained with the FPGA-based testbed. To start, Section VI-A explains the configurations of the experiments and Section VI-B shows results that validate the implementation. Subsequently, we analyze four performance aspects. First, Section VI-C discusses the timing alignment figures that are achievable with the prototype. Then, Section VI-D demonstrates latency and buffering implications of the adopted RFS approach. Next, Section VI-E analyzes the impact of triggering fluctuations that arise in practice due to clock corrections. Lastly, Section VI-F evaluates the impact of BG traffic on the RFS mechanism.

TABLE 2. Summary of parameters adopted along experiments.

Parameter	Value		
d_{trans}	1.856 µs		
i_{tx}	4.16 µs		
J	4		
f_s and T_s	7.68 Msps and 130.2 ns, respectively		
R	30.72 Msps		
T	1 sec		
n_b	30720000 samples		
sync margin ϵ	1500 ns	PTP-unaware FH	PTP RTC
	500 ns	PTP-aware FH	PTP RTC
	20 ns	-	PPS RTC

A. ADOPTED PARAMETERS AND CONFIGURATIONS

Table 2 summarizes the RFS and FH parameters of the experiments. In the adopted FH setup, each FH frame carries 32 IQ samples to each of the two AxCs of the destination RRU, namely a total of 64 IQ samples. The IQ samples, in turn, are each represented with 24 bits, such that the IQ payload of FH frames consists of 192 bytes. Furthermore, FH frames carry 28 bytes of overhead, including the Ethernet header and metadata, such as the departure timestamp. Hence, the total FH frame size is 220 bytes. Consequently, with GbE's line rate, the transmission interval d_{trans} of each FH frame, including a 96 ns inter-packet gap, is of roughly 1.856 μs.

As mentioned in Section V-C, we adopt a baseband sample rate of $f_s = 7.68$ Msps, such that the IQ sample period is $T_s \approx 130.2$ ns. Since each RRU runs 2 AxCs, the BBU serves a total of $J = 4$ AxCs. Correspondingly, the BBU feeds CBRFL traffic into the FH with an aggregate sample rate of $R = (J \cdot f_s) = 30.72$ Msps. Moreover, since IQ samples are generated in real-time by the BBU (all AxCs in parallel), and each FH frame carries 32 IQ samples of each AxC, interval i_{tx} (see Fig. 6) is of 4.16 μs (i.e. $32/f_s$). In terms of bit rates, since the BBU sends 2 frames of 220 bytes on every interval i_{tx}, its output has a bit rate of $(2 \cdot 8 \cdot 220)/i_{tx} = 844.8$ Mbps, with 422.4 Mbps to each RRU.

In terms of time sources, we exploit both PTP and PPS RTCs in the experiments. Furthermore, we explore the two available oscillator options. By default, an XO drives both the PTP and PPS RTCs. The only exception is in Section VI-C, where we present an evaluation using the OCXO.

For RFS, we adopt a one-second trigger interval, i.e. $T = 1$ s. Hence, based on (1), a block is composed of $n_b = 30720000$ samples. Each RRU, in turn, receives a sub-block of 15360000 samples per interval T. Also, in the absence of BG traffic, we consider that the maximum E2E delay of the RRU on the longest path in Fig. 9 (RRU 1) is near 30 μs. Then, assuming that the sync margin needs to be $\epsilon = 1.5$ μs when running PTP over our timing-unaware FH network, (2) leads to a time advance of 33.356 μs. Nevertheless, since the RRUs are arranged in descending order of E2E delay, (3) is applicable and requires a time advance of 31.5 μs for RRU 1. Using (4), this becomes the theoretical global time advance to start with when using the PTP RTC over the PTP-unaware FH. When using the PTP RTC over the PTP-aware FH or when running the RFS controller based on the PPS RTC, we reduce the sync margin ϵ to 500 ns and 20 ns, respectively, so that the starting time advance reduces accordingly.

Moreover, unless mentioned otherwise, the time advance is tuned in real-time based on UL delay measurements using the estimator in (13). Also, in all experiments, we choose to measure FH frame delays using timestamps taken from the PPS RTC, regardless of the time source chosen for the RFS controller (c.f. Section IV-B). Hence, the BBU considers a delay measurement uncertainty of $\gamma = 20$ ns (PPS RTC's maximum time offset) when computing (11).

Lastly, to define buffer depths, we assume each hop adds roughly 4 μs of processing delay. Hence, any FH frame that does not experience queuing delay would traverse the shortest path of two hops (from BBU to RRU 2) in 8 μs. Thus, from (5), we estimate that a buffer depth of $L = 180$ samples would be sufficient for each AxC of the two RRUs.

B. SYSTEM VALIDATION

Next, the goal is to validate the functionality of the implementation. To do so, we first verify that the BBU and RRU prototypes are capable of triggering transmissions timely in hardware in order to accomplish RFS. Then, we analyze whether the distributed RF transmissions carried out by the RRUs are effectively time-aligned and whether the alignment follows the relative offset between their local RTCs.

Fig. 11 shows an acquisition from the logic analyzer with signals from the BBU and RRU 1, particularly taken during BBU's initialization of FH traffic. The signal at the top is the RFS trigger of the BBU, which is time advanced by slightly more than 30 μs (each horizontal division has 10 μs) relative to the RRU's trigger.[3] After asserting the trigger, the BBU starts DL frame transmissions. Label `RRU FH Rx` then shows that the first frame arrives timely on the RRU side, i.e. before the RRU's trigger, which validates the desired time advance functionality. Moreover, from signal `BBU FH Tx LAST`, which marks the end of every frame departing from the FH transmit logic, one can observe the serving rounds with two frames roughly every $i_{tx} = 4.16$ μs.

Next, Fig. 12 contrasts the cross-correlation peak index measured by the SDR application throughout 30 min experiments when running the RFS controller based on both the PTP RTC and the PPS RTC. In this experiment, we use the PTP-unaware FH network. Note that, with the PPS RTC, the peak index stays reliably within ± 2, as also shown in the histogram of Fig. 13. In contrast, with the PTP-synchronized RTC, the timing alignment fluctuates more significantly and randomly. In this specific realization, it varies roughly from -11 to $+1$, which is equivalent to a range of 1.56 μs.

[3] At this point, the time advance had already been tuned based on the UL traffic. Signals labeled as "RRU FH Tx" indicate the activity of UL traffic.

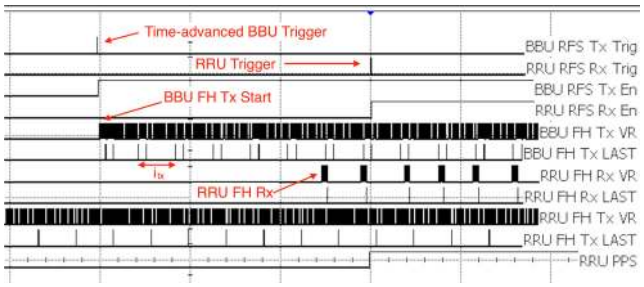

FIGURE 11. Logic analysis of RFS hardware signals, on 10 μs/div scale.

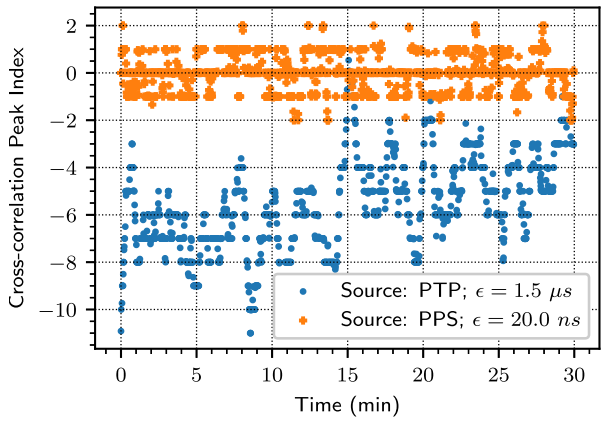

FIGURE 12. Cross-correlation peak indexes observed over 30 min with the RFS controller reading time from the PPS and PTP RTCs.

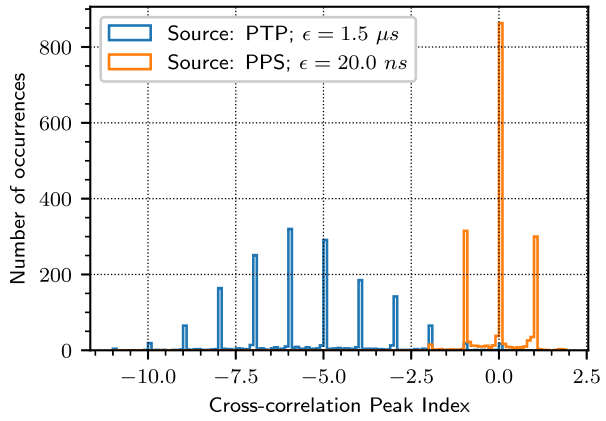

FIGURE 13. Histogram of the cross-correlation peak indexes observed over 30 min.

Finally, Fig. 14 presents the relative clock offset between the PTP RTCs of the RRUs, as measured by the TIC (based directly on RTC signals), and compares it to the TAE measured by the SDR application between the RF transmissions of the RRUs when using the PTP RTC as time source for RFS. The two measurements match to a great extent, which confirms that the RTC is governing the RF transmission timing. This is a desirable property, as it indicates that the RRUs are successfully re-synchronizing on every trigger and that there is no accumulation of error, such as delayed IQ samples held in the two-stage buffering scheme of Fig. 7.

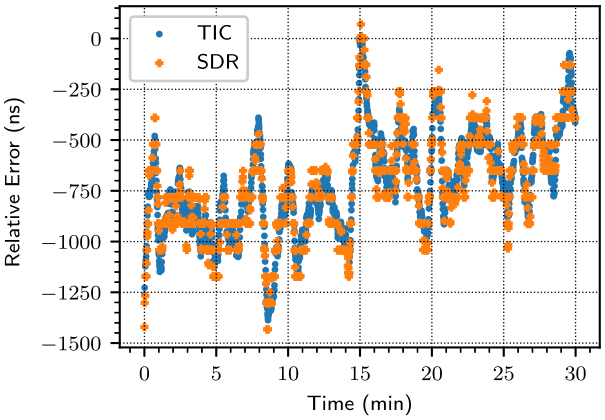

FIGURE 14. Relative time alignment error between RRUs measured by the TIC and the SDR application during the observation of the cross-correlation when using the PTP RTC as time source for RFS.

C. TIMING ALIGNMENT PERFORMANCE

Next, we shall evaluate the best RF timing alignment performance that is achievable in the testbed. At first, the PPS RTC results from Fig 12 do not meet expectations. Since the baseband sample period is 130.2 ns and we expect the RTC synchronization to be better than ± 20 ns (less than a sample period), one would expect the cross-correlation peak index to remain consistently near 0. Nevertheless, there are cross-correlation peak occurrences within ± 2 in Fig. 12.

This unsatisfactory performance does not come from PPS synchronization. Fig. 15 shows the histogram of the error between the RRU's PPS RTC time and the PPS reference (generated by the BBU), measured in hardware on every PPS rising edge. Note the error is indeed within the expected range of ± 20 ns. Instead, the determinant factor is that the current version of the FPGA design always drives the FMCOMMS2 RF frontend with a reference clock signal derived from the PTP RTC. The RF frontend then generates the sampling and carrier frequencies based on this reference signal, as thoroughly explained in [13]. Consequently, even when running the RFS controller based on the PPS RTC, the DAC still consumes samples from the CDC FIFO with a rate governed by the PTP RTC. Ultimately, since the PTP RTC is continuously rate-adjusted, eventual discrepancies can occur.

Due to this hardware limitation, we achieve the best timing alignment performance in our setup with the RFS controller still based on the PTP RTC but over the PTP-aware FH and with the RTC driven by the OCXO, rather than the XO adopted in all other experiments. This is because smoother frequency corrections are applied to the PTP RTC when relying on the PTP-aware FH and, more importantly, because the OCXO has significantly better frequency stability. Fig. 16 shows an experiment carried under these conditions. Note the alignment is reliably within -1 to 0.

D. LATENCY AND BUFFERING DUE TO TIME ADVANCE

Next, we discuss the buffering and the associated latency that the time advance mechanism leads to on RRUs. Fig. 17

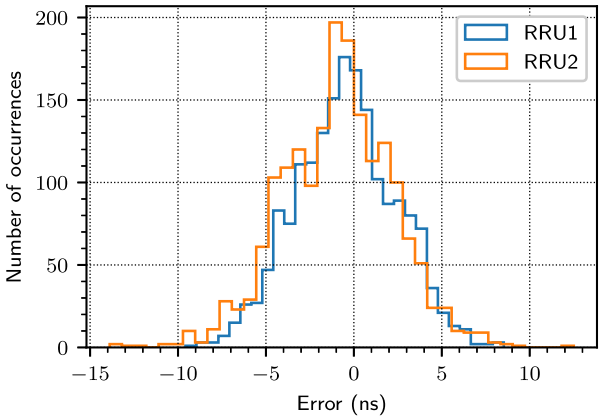

FIGURE 15. Histogram of the error measured between the PPS RTC time and the rising edge instant of the PPS reference, on both RRUs.

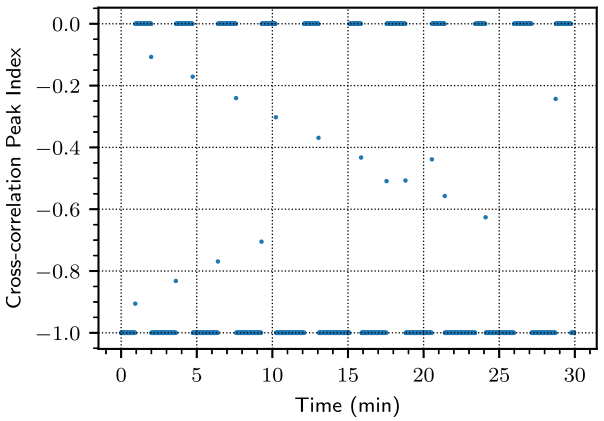

FIGURE 16. Cross-correlation peak indexes observed over 30 min with the RFS controller reading time from the PTP RTC, with the PTP-aware FH network and with OCXOs driving the reference clock of the PTP RTC.

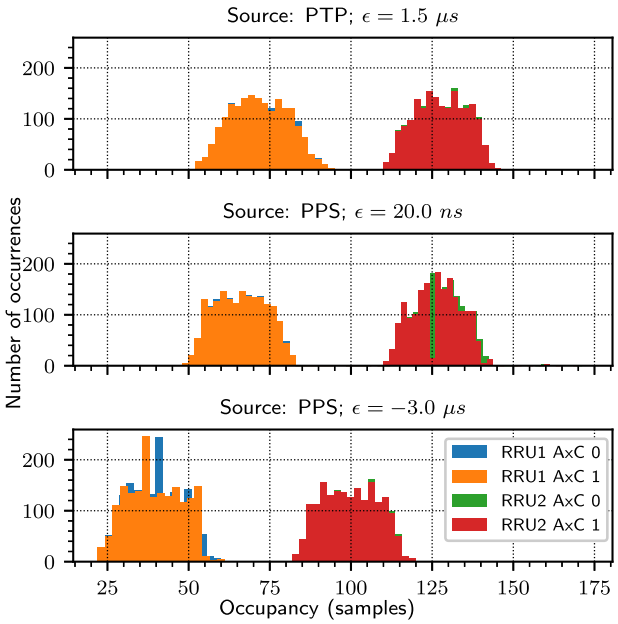

FIGURE 17. Histogram of first-stage buffer occupancy snapshots taken under varying time sources and sync margins on both RRUs.

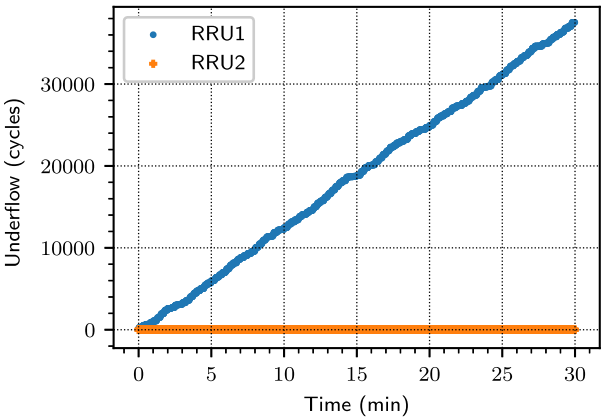

FIGURE 18. Cumulative sum of underflow occurrences observed on first-stage FIFOs when using excessively optimistic time advance.

shows histograms concerning the occupancy of first-stage FIFOs observed throughout 30 min experiments carried in the PTP-unaware FH under three different scenarios of time source and sync margin ϵ. In all scenarios, the frames sent to RRU 1 experience less buffering than frames sent to RRU 2. This is expected in the adopted topology because RRU 1 (located after four hops) receives FH frames later than RRU 2 and closer to when the encapsulated IQ samples should go on air. In the first (top) row of the plot, i.e. with the PTP RTC, note the average buffer occupancy on RRU 1 is of 72 samples, whereas in the second row, with the PPS RTC as time source, it is of 65 samples. This reduced occupancy comes from using a shorter sync margin ϵ with the PPS RTC and, correspondingly, a tighter time advance. More importantly, it confirms that the latency imposed by the RFS mechanism can be alleviated by providing better synchronization accuracy.

Moreover, in all cases of Fig. 17, the occupancy is below the computed buffer depth of $L = 180$, as desired. Also, in the first two plots (from top to bottom), both RRUs show occupancy levels above 32 samples, which indicates the FIFOs continuously hold the content of more than one FH frame (recall each frame carries 32 samples per AxC). The third (bottom) plot, in contrast, concerns a scenario where the BBU adopts a more optimistic time advance. We choose to express this optimistic choice in terms of the sync margin ϵ, which is an additive term in (11) that is configurable offline based on the maximum expected trigger oscillation, unlike $\tilde{u}_k$, which is defined in runtime based on delay measurements. In particular, a negative margin of $-3.0\ \mu s$ was adopted for this experiment to make the time advance abnormally optimistic.

Note that the occupancy levels in the bottom plot of Fig. 17 are substantially lower and that RRU 1 even experiences levels below 32. However, this was not a safe choice for performance. Fig. 18 shows the cumulative sum of the clock cycles over which the first-stage FIFOs were empty during this experiment, measured in hardware. Note that RRU 1 is continuously suffering from unacceptable underflows due to being on the critical path, i.e. the path with the highest E2E

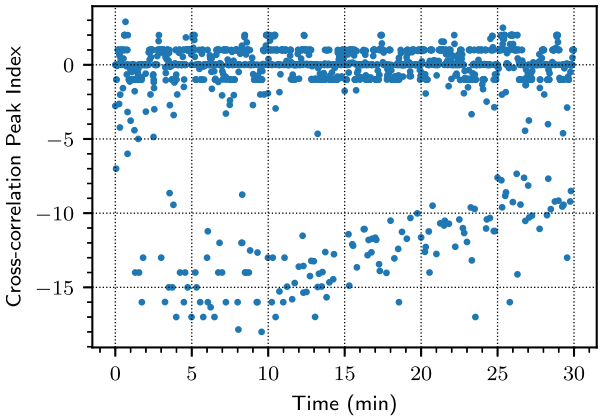

FIGURE 19. Cross-correlation peak index observed over 30 min when configuring the sync margin to $\epsilon = -3.0\ \mu s$ such that the time advance becomes excessively optimistic.

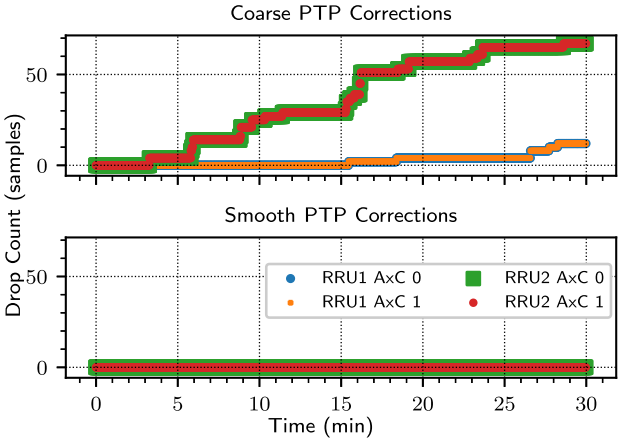

FIGURE 20. Cumulative IQ sample drop count when using two distinct PTP synchronization algorithms, with coarse and smooth corrections.

delay. As a result, despite relying on the PPS RTC, the RF transmissions frequently suffer failures of timing alignment, as shown in the cross-correlation peak indexes of Fig. 19.

E. IMPACT FROM TRIGGER FLUCTUATIONS

Next, we evaluate the effects of fluctuations in the interval between consecutive RFS triggers on the RRUs, which arise due to clock corrections. For this evaluation, we take the PTP RTC as the time source for RFS and rely on the PTP-unaware network, namely a configuration that typically leads to noisy clock corrections. Then, we contrast two clock disciplining algorithms. One executes abrupt time adjustments, whereas the other is a servo loop that applies smoother corrections.

Fig. 20 shows the cumulative sum of IQ samples dropped by the RFS controller during 30 min. Fig. 21, in turn, shows the time offset corrections that were applied to the PTP RTC throughout the experiment. In Fig. 21, note that the coarser algorithm applies corrections that exceed the safe limits of $\pm T_s$ (in this case ± 130.2 ns) given by (17), whereas the smoother algorithm continuously applies corrections within ± 100 ns. Correspondingly, Fig. 20 shows that the RFS controller only drops samples under coarse corrections.

Fig. 22 shows the relative timing alignment between the RF transmissions of the RRUs during the given experiment, as measured by the SDR application. Also, it shows the relative time offset between the PTP RTCs of the RRUs, as measured by the TIC. Note that the RF alignment follows the relative RTC offset even under the coarse clock disciplining algorithm. This reveals that the sample dropping mechanism ensures successful RF timing re-alignment on every trigger.

Lastly, Fig. 23 shows the number of clock cycles over which the CDC buffers remained empty per observed second of the experiment. Note that, similarly to IQ sample dropping, underflows continuously occur solely under the coarse clock disciplining algorithm. Nevertheless, Fig. 22 shows that such CDC underflows do not compromise the match between the RF alignment and the relative time offset between the RTCs. For example, around minute 27 of the experiment,

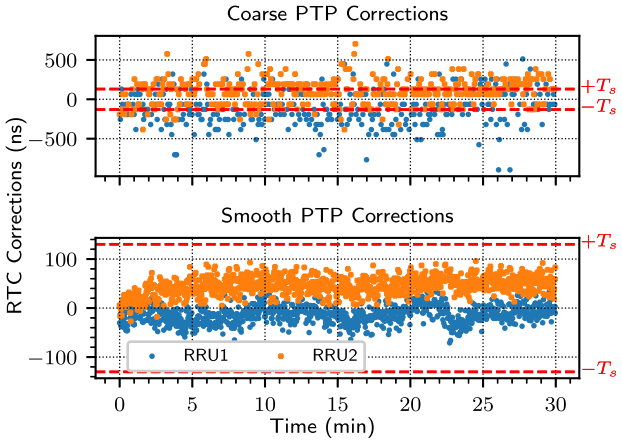

FIGURE 21. Time offset corrections applied to the PTP RTC of the RRUs with the coarse and smooth clock disciplining algorithms.

Fig. 21 shows that the coarse algorithm applied a correction of roughly -800 ns to the PTP RTC of the RRU 1. As a result, Fig. 23 shows that the RRU 1 experienced an underflow of nearly 80 cycles at this point, i.e. of 800 ns given the 100 MHz clock. Finally, Fig. 22 shows that this underflow preserved the match between the RF timing and relative RTC offset around this moment. In contrast, underflows caused by insufficient time advance tend to last over longer intervals and more strongly degrade the timing alignment performance, such as in the case of Fig. 19.

F. PERFORMANCE UNDER BACKGROUND TRAFFIC

Lastly, we evaluate the impact of BG traffic on the dynamics of the time advance. We introduce bidirectional BG traffic as described in Section V-B, with each traffic source host of Fig. 9 generating an aggregate rate ranging from 15 to 25 Mbps in each direction. As a result, the maximum E2E delays of FH frames vary widely and sporadically peak at high values. Besides, the resulting delay variation is particularly strong given that our testbed assigns equal forwarding priority to all traffic streams (FH, PTP and BG).

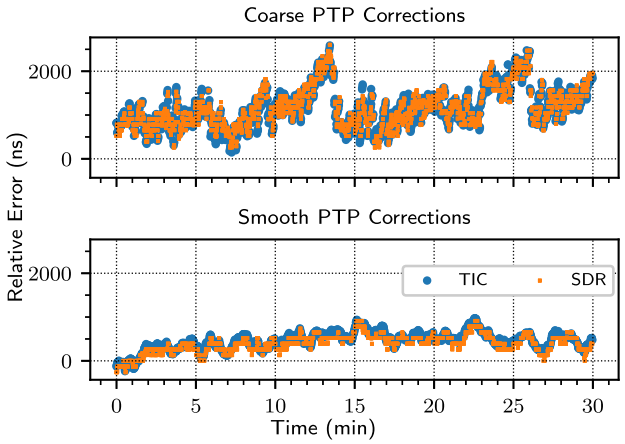

FIGURE 22. Relative time alignment error between RRUs measured by the TIC and the SDR application under the two PTP synchronization algorithms.

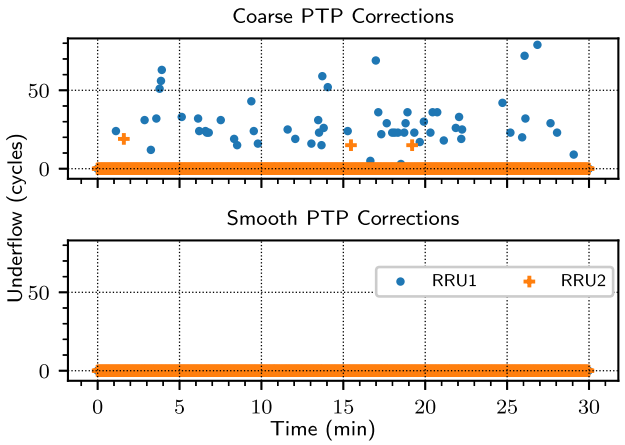

FIGURE 23. CDC FIFO underflows observed per second when using two distinct PTP synchronization algorithms, with coarse and smooth corrections.

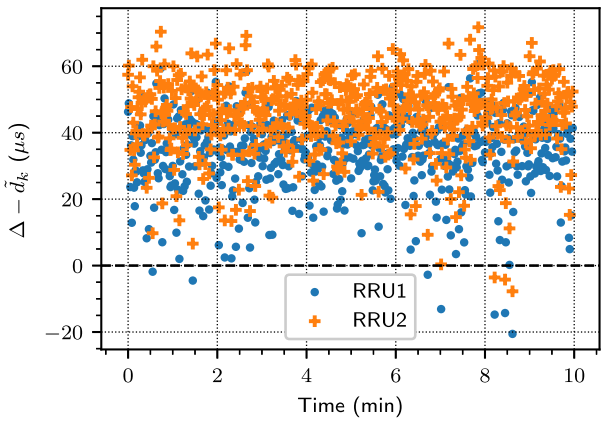

FIGURE 24. Difference between the fixed time advance adopted by the BBU and the DL E2E delays measured by the two RRUs under background traffic.

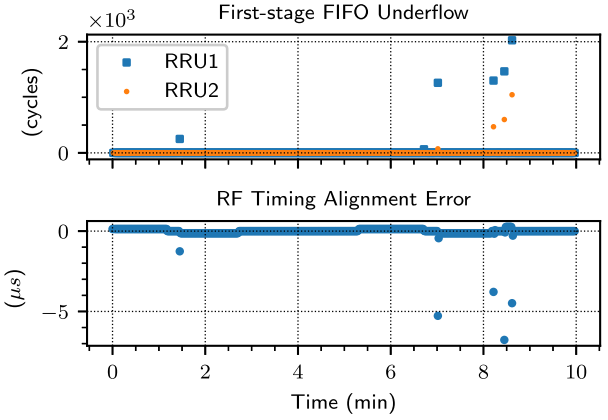

FIGURE 25. First-stage FIFO underflows due to sporadic large DL E2E delays and the corresponding RF timing alignment measured by the SDR application.

In this experiment, we choose to rely on the PTP RTC as time source for RFS. Since the BG traffic also introduces significant PDV on PTP messages, and the experiment focuses instead on how the PDV affects the RFS's time advance performance, we then choose to transport PTP over the PTP-aware network. As a result, the PDV does not degrade the clock synchronization performance significantly. Furthermore, in this experiment, the BBU does not tune the time advance in real-time, unlike in previous experiments. Instead, it adopts a large and fixed time advance of $\Delta = 100.5\ \mu$s. This choice enables the demonstration that follows.

Fig. 24 shows the difference between the constant time advance and the maximum DL E2E delay $\tilde{d}_k$ measured (using the PPS RTC) in real-time by the two RRUs based on observation windows of one second. Note that the DL E2E delays of both RRUs eventually surpass the time advance due to excessive queuing delays over the FH network. Fig 25 shows that precisely around the instants of negative values in Fig. 24, the first-stage FIFOs of the RRUs experience underflows, with RRU 1 (on the critical path) subject to more prolonged starvation. This starvation, in turn, significantly degrades the RF timing alignment, as shown in the bottom part of Fig 25. For example, around minute 8.5 of the experiment, we can see in Fig. 24 that RRU 1 observed a maximum E2E delay exceeding the time advance by approximately 20 μs. Around the same instant, Fig 25 shows that the first-stage buffers of RRU 1 were empty for 2000 clock cycles (i.e. for 20 μs, given the 100 MHz clock) and there was significant loss of timing alignment between the two RRUs.

VII. CONCLUSIONS

This work investigated the problem of timing alignment of RF transmissions carried out by spatially distributed radio units that are served over an Ethernet (or more generally packet-switched) FH. To this end, this work formulated a mechanism based on coordinated triggers among RRUs and BBUs, where BBUs rely on time-advanced triggers. This work then analyzed how parameters of the proposed architecture can be chosen in practice and thoroughly described the hardware and software of an FPGA-based prototype implementation. In the end, this paper presented several experiments that highlighted essential RFS performance aspects.

We discussed in particular that, to achieve RFS, the coordinated RRUs need to buffer the incoming data for some time and that this buffering leads to additional FH transport delay. We showed that this extra delay for RFS is acceptable and that there are ways to alleviate it. For example, the sorted delivery of frames to RRUs in descending order of E2E delays can be helpful. Also, we demonstrated that the better the clock synchronization accuracy of RRUs is, the lower the additional latency that is introduced by the RFS scheme.

We also investigated some desirable properties for an RFS system. For instance, we discussed that coordinated RRUs must be able to discard delayed samples when necessary in order to ensure re-synchronization on every RFS trigger. Secondly, we demonstrated the importance of using clock disciplining algorithms that apply smooth correction steps to avoid jitter on the RFS trigger and, correspondingly, eventual disturbances on the RFS performance.

Our evaluations also included other practical considerations. We presented results under both PTP-aware and PTP-unaware networks. Also, we contrasted the timing alignment performance achieved when using an OCXO and an ordinary XO. Moreover, we experimented with background traffic and demonstrated that it can degrade the RFS significantly in case the BBU does not adapt the time advance proactively.

Future work shall exploit the developed testbed for investigation of the performance that is ultimately achieved in the air interface when using RFS. For example, the impact of RFS timing realignments on the corresponding quality of the signal received by the user equipment.

REFERENCES

[1] C.-L. I, H. Li, J. Korhonen, J. Huang, and L. Han, "RAN revolution with NGFI (xhaul) for 5G," *J. Lightw. Technol.*, vol. 36, no. 2, pp. 541–550, Jan. 15, 2018.

[2] N. J. Gomes, P. Chanclou, P. Turnbull, A. Magee, and V. Jungnickel, "Fronthaul evolution: From CPRI to Ethernet," *Opt. Fiber Technol.*, vol. 26, pp. 50–58, Dec. 2015.

[3] R. M. Rao, M. Fontaine, and R. Veisllari, "A reconfigurable architecture for packet based 5G transport networks," in *Proc. IEEE 5G World Forum (5GWF)*, Jul. 2018, pp. 474–477.

[4] N. J. Gomes, P. Sehier, H. Thomas, P. Chanclou, B. Li, D. Munch, P. Assimakopoulos, S. Dixit, and V. Jungnickel, "Boosting 5G through Ethernet: How evolved fronthaul can take next-generation mobile to the next level," *IEEE Veh. Technol. Mag.*, vol. 13, no. 1, pp. 74–84, Mar. 2018.

[5] *IEEE Standard for Local and Metropolitan Area Networks: Time-Sensitive Networking for Fronthaul*, Standard 802.1CM, May 2018.

[6] *Common Public Radio Interface: eCPRI Interface Specification v2.0*, May 2019. [Online]. Available: http://www.cpri.info/downloads/eCPRI_v_2.0_2019_05_10c.pdf

[7] *IEEE Standard for Radio Over Ethernet Encapsulations and Mappings*, Standard 1914.3-2008, Sep. 2018.

[8] *Common Public Radio Interface (CPRI) specification v7.0*, Oct. 2015. [Online]. Available: http://www.cpri.info/downloads/CPRI_v_7_0_2015-10-09.pdf

[9] *IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, Standard 1588-2008, Jul. 2008, pp. 1–300.

[10] *Timing Characteristics Of Synchronous Equipment Slave Clock*, document Rec. G.8262, ITU-T, Geneva, Switzerland, Nov. 2018.

[11] F. Girela-Lopez, J. Lopez-Jimenez, M. Jimenez-Lopez, R. Rodriguez, E. Ros, and J. Diaz, "IEEE 1588 high accuracy default profile: Applications and challenges," *IEEE Access*, vol. 8, pp. 45211–45220, 2020.

[12] *Rec. G.8262.1: Timing Characteristics of Enhanced Synchronous Equipment Slave Clock*, ITU-T, Geneva, Switzerland, Jan. 2019.

[13] I. Freire, C. Lu, M. Berg, and A. Klautau, "An FPGA-based design of a packetized fronthaul testbed with IEEE 1588 clock synchronization," in *Proc. Eur. Wireless*, May 2017, pp. 1–6.

[14] *5G; New Radio (NR); Physical Channels and Modulation*, document TS 38.211, 3GPP, version 15.7.0 Release 15, Oct. 2019.

[15] H. Li, L. Han, R. Duan, and G. M. Garner, "Analysis of the synchronization requirements of 5G and corresponding solutions," *IEEE Commun. Standards Mag.*, vol. 1, no. 1, pp. 52–58, Mar. 2017.

[16] J.-C. Lin, "Synchronization requirements for 5G: An overview of standards and specifications for cellular networks," *IEEE Veh. Technol. Mag.*, vol. 13, no. 3, pp. 91–99, Sep. 2018.

[17] J. A. del Peral-Rosado, R. Raulefs, J. A. Lopez-Salcedo, and G. Seco-Granados, "Survey of cellular mobile radio localization methods: From 1G to 5G," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 2, pp. 1124–1148, 2nd Quart., 2018.

[18] M. Cierny, R. Wichman, and Z. Ding, "Impact of base station time synchronization mismatch on almost blank subframes," *IEEE Commun. Lett.*, vol. 17, no. 11, pp. 2092–2095, Nov. 2013.

[19] R. Rogalin, O. Y. Bursalioglu, H. Papadopoulos, G. Caire, A. F. Molisch, A. Michaloliakos, V. Balan, and K. Psounis, "Scalable synchronization and reciprocity calibration for distributed multiuser MIMO," *IEEE Trans. Wireless Commun.*, vol. 13, no. 4, pp. 1815–1831, Apr. 2014.

[20] *Evolved Universal Terrestrial Radio Access (E-UTRA); Carrier Aggregation; Base Station (BS) Radio Transmission and Reception*, document TR 36.808, 3GPP, version 10.1.0 Release 10, Jul. 2013.

[21] *Evolved Universal Terrestrial Radio Access (E-UTRA); Requirements for Support of Radio Resource Management*, document TS 36.133, 3GPP, version 15.4.0 Release 15, Jan. 2019.

[22] *5G; New Radio (NR); Requirements for support of radio resource management*, document TS 38.133, 3GPP, version 15.7.0 Release 15, Oct. 2019.

[23] C. Simon, M. Maliosz, and M. Mate, "Design aspects of low-latency services with time-sensitive networking," *IEEE Commun. Standards Mag.*, vol. 2, no. 2, pp. 48–54, Jun. 2018.

[24] S. Bjørnstad, D. Chen, and R. Veisllari, "Handling delay in 5G Ethernet mobile fronthaul networks," in *Proc. Eur. Conf. Netw. Commun. (EuCNC)*, Jun. 2018, pp. 1–9.

[25] V. Jungnickel, T. Wirth, M. Schellmann, T. Haustein, and W. Zirwas, "Synchronization of cooperative base stations," in *Proc. IEEE Int. Symp. Wireless Commun. Syst.*, Oct. 2008, pp. 329–334.

[26] H. V. Balan, R. Rogalin, A. Michaloliakos, K. Psounis, and G. Caire, "AirSync: Enabling distributed multiuser MIMO with full spatial multiplexing," *IEEE/ACM Trans. Netw.*, vol. 21, no. 6, pp. 1681–1695, Dec. 2013.

[27] F. Quitin, M. M. U. Rahman, R. Mudumbai, and U. Madhow, "A scalable architecture for distributed transmit beamforming with commodity radios: Design and proof of concept," *IEEE Trans. Wireless Commun.*, vol. 12, no. 3, pp. 1418–1428, Mar. 2013.

[28] D. Chitimalla, K. Kondepu, L. Valcarenghi, M. Tornatore, and B. Mukherjee, "5G Fronthaul–Latency and jitter studies of CPRI over Ethernet," *J. Opt. Commun. Netw.*, vol. 9, no. 2, pp. 172–182, Feb. 2017.

[29] M. Waqar, A. Kim, and P. K. Cho, "A transport scheme for reducing delays and jitter in Ethernet-based 5G fronthaul networks," *IEEE Access*, vol. 6, pp. 46110–46121, 2018.

[30] N. J. Gomes, P. Assimakopoulos, P. Chanclou, K. Habel, V. Jungnickel, P. Ritosa, J. Zou, J.-P. Elbers, H. Thomas, G. Linne, and C. Juchems, "Testbed verification of new fronthaul technology for 5G systems," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, May 2019, pp. 1–6.

[31] T. Cooklev, R. Normoyle, and D. Clendenen, "The VITA 49 analog RF-digital interface," *IEEE Circuits Syst. Mag.*, vol. 12, no. 4, pp. 21–32, 2012.

[32] *IEEE Standard for a Transport Protocol for Time-Sensitive Applications in Bridged Local Area Networks*, Standard 1722-2016 (Revision of IEEE Std 1722-2011), Dec. 2016, pp. 1–233.

[33] M. D. Johas Teener, A. N. Fredette, C. Boiger, P. Klein, C. Gunther, D. Olsen, and K. Stanton, "Heterogeneous networks for audio and video: Using IEEE 802.1 audio video bridging," *Proc. IEEE*, vol. 101, no. 11, pp. 2339–2354, Nov. 2013.

[34] K. B. Stanton, "Distributing deterministic, accurate time for tightly coordinated network and software applications: IEEE 802.1AS, the TSN profile of PTP," *IEEE Commun. Standards Mag.*, vol. 2, no. 2, pp. 34–40, Jun. 2018.

[35] M. Lipinski, E. van der Bij, J. Serrano, T. Wlostowski, G. Daniluk, A. Wujek, M. Rizzi, and D. Lampridis, "White rabbit applications and enhancements," in *Proc. IEEE Int. Symp. Precis. Clock Synchronization for Meas., Control, Commun. (ISPCS)*, Sep. 2018, pp. 1–7.
[36] G. Otero Perez, D. Larrabeiti Lopez, and J. A. Hernandez, "5G new radio fronthaul network design for eCPRI-IEEE 802.1CM and extreme latency percentiles," *IEEE Access*, vol. 7, pp. 82218–82230, 2019.
[37] L. M. P. Larsen, A. Checko, and H. L. Christiansen, "A survey of the functional splits proposed for 5G mobile crosshaul networks," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 1, pp. 146–172, 1st Quart., 2019.
[38] J. C. Eidson and K. B. Stanton, "Timing in cyber-physical systems: The last inch problem," in *Proc. IEEE Int. Symp. Precis. Clock Synchronization for Meas., Control, Commun. (ISPCS)*, Oct. 2015, pp. 19–24.
[39] I. Freire, I. Sousa, P. Bemerguy, A. Klautau, I. Almeida, C. Lu, and M. Berg, "Analysis of controlled packet departure to support Ethernet fronthaul synchronization via PTP," in *Proc. IEEE Int. Symp. Precis. Clock Synchronization Meas., Control, Commun. (ISPCS)*, Sep. 2018, pp. 1–6.
[40] I. Hadzic and D. R. Morgan, "Adaptive packet selection for clock recovery," in *Proc. IEEE Int. Symp. Precis. Clock Synchronization Meas., Control Commun.*, Sep. 2010, pp. 42–47.
[41] *Rec. G.8261: Timing and Synchronization Aspects in Packet Networks*, ITU-T, Geneva, Switzerland, Aug. 2019.

IGOR FREIRE (Member, IEEE) received the B.Sc. and M.Sc. degrees in electrical engineering from the Federal University of Pará (UFPA), Belém, Brazil, in 2013 and 2015, respectively, where he is currently pursuing the Ph.D. degree in electrical engineering. Since 2012, he has been with LASSE - 5G & IoT Research Group, Belém. His current research interests include clock synchronization, fronthaul technologies, wireless communications, and software-defined radio.

IGOR ALMEIDA (Member, IEEE) received the B.Sc. degree in computer engineering and the M.Sc. degree in electrical engineering from the Federal University of Pará (UFPA), Belém, Brazil, in 2010 and 2013, respectively, where he is currently pursuing the Ph.D. degree in electrical engineering. He has been with Ericsson Research, since 2016, involved with topics such as 5G, fronthaul networking, synchronization, and wireless communications.

EDUARDO MEDEIROS received the M.Sc. degree in electrical engineering from the Federal University of Pará, Belém, Brazil, in 2010, and the Licentiate and Ph.D. degrees from Lund University, Lund, Sweden, in 2015 and 2018, respectively. Since 2011, he holds a Researcher position with Ericsson Research, Stockholm, Sweden. His current research interests include signal processing for broadband communications and fronthaul for 5G systems. He was a co-recipient of two Best Paper Awards from the IEEE International Conference on Communications and three Best Paper Awards from the IEEE Communications Society's Transmission, Access, and Optical Systems Technical Committee.

MIGUEL BERG (Senior Member, IEEE) received the B.S. degree in electrical engineering and computer science from Mid Sweden University, Sweden, in 1995, and the Licentiate and Ph.D. degrees in wireless communication systems from the Royal Institute of Technology (KTH), Sweden, in 1999 and 2002, respectively. As a Principal Researcher at Ericsson Research, Stockholm, Sweden, he is currently leading research on efficient fronthaul and lower layer splits for 4G/5G RAN. Since 2012, he has made significant contributions to research and development for the Ericsson Radio Dot System, targeting indoor small cells. After joining Ericsson in 2007, he initially worked with xDSL line testing and G. fast, as a continuation of his work as a Researcher at Lund University, from 2006 to 2007. From 2004 to 2006, he held a Research position at KTH in the area of wireless access, and from 2002 to 2003, he worked with the development of 2G/3G base station antennas and tower-mounted amplifiers at Radio Components Sweden AB. He has coauthored about 35 scientific publications (journal, letters, magazine, and conference), and 80 filed inventions.

CHENGUANG LU was born in Henan, China, in 1978. He received the M.Sc. degree in digital communication from the Chalmers University of Technology, Sweden, in 2005, and the Ph.D. degree in wireless communication from Aalborg University, Denmark, in 2008. In 2007, he was a Visiting Scholar with the University of Illinois at Urbana–Champaign, USA. Since 2008, he has been with Ericsson Research, Sweden. He is currently a Master Researcher, involved in efficient Fronthaul interface design, as well as edge computing for IoT. He was a part of the Research Team developing the innovative indoor Fronthaul interface for Ericsson Radio Dot system. He has coauthored over 30 publications in peer-reviewed conferences and journals and has filed over 60 patent applications (families).

ELMAR TROJER received the M.B.A. degree from the University of Vienna, Austria, and the Ph.D. degree in electrical engineering from the Vienna University of Technology. He joined Ericsson, in 2005, where he is currently working on fronthaul networking technologies for 5G. He has been active on both fixed and MBB technologies such as VDSL2, G. fast, GPON, and mobile backhaul and fronthaul technologies for 4G and 5G small cells.

ALDEBARO KLAUTAU (Senior Member, IEEE) received the bachelor's degree from the Universidade Federal do Pará (UFPA), in 1990, the M.Sc. degree from the Universidade Federal de Santa Catarina (UFSC), in 1993, and the Ph.D. degree in electrical engineering from the University of California at San Diego (UCSD), in 2003. Since 1996, he has been with UFPA, where he is currently a Full Professor, the ITU-T TIES Focal Point, and directs LASSE. He was a Visiting Scholar with Stockholm University, UCSD, and The University of Texas at Austin. He is also a Researcher of the Brazilian National Council of Scientific and Technological Development (CNPq). He has supervised more than 50 graduate students, published more than 150 articles in peer-reviewed conferences and journals, and has several international patents. His research interests include machine learning and signal processing for telecommunications and embedded systems.