Pulsar: A Wireless Propagation-Aware Clock Synchronization Platform

Adwait Dongare, Patrick Lazik, Niranjini Rajagopal, Anthony Rowe
Electrical and Computer Engineering Department
Carnegie Mellon University
Pittsburgh PA, USA.
{adongare, plazik, niranjir, agr}@ece.cmu.edu

***Abstract*—In this paper, we introduce *Pulsar*, a wireless time transfer platform that can achieve clock synchronization to better than five nanosecond between indoor or GPS-denied devices. Nanosecond-level clock synchronization is a missing capability for many real-time applications like next-generation wireless systems that leverage spatial multiplexing to improve channel capacity and provide services like time-of-flight localization. With fine-grained synchronization, both clock stability and propagation delays introduce significant sources of error. Pulsar leverages a stable clock source derived from a Chip-Scale Atomic Clock (CSAC) along with an Ultra-WideBand (UWB) radio able to perform sub-nanosecond packet timestamping to estimate and correct for clock offsets. We design and evaluate a proof-of-concept network-wide synchronization protocol for Pulsar that selects low-jitter links to both estimate the location of nodes and reduce cumulative synchronization error across multiple hops.**

The Pulsar platform and protocol together provide a phase synchronized one pulse per second (1PPS) signal and 10 MHz reference clock that can be easily integrated with typical end-user applications like software-defined radios and communication systems. We experimentally evaluate the Pulsar platform in terms of clock synchronization accuracy, Allan deviation between pair-wise clocks and ranging accuracy to show a clock synchronization of better than five nanoseconds per hop with an average of 2.12 ns and a standard deviation of 0.84 ns. The platform is able to identify and avoid clock error in cases where there is heavy multi-path or non-Line-of-Sight signals.

I. INTRODUCTION

In this paper, we present *Pulsar* a wireless time transfer platform able to synchronize devices across a multi-hop network to within 5 ns per hop. This level of clock synchronization is often required for applications in scientific experimentation and next generation wireless systems that use collaborative multi-antenna techniques. For example, current wireless systems can use Multiple-Input Multiple-Output (MIMO) antenna arrays to transmit independent and separately encoded data signals from more than one antenna for beamforming or to help collaboratively receive weak uplink signals. Traditionally, MIMO antennas are located on a single device with carefully constructed delay paths. Through tight clock synchronization, it is possible to apply MIMO techniques across multiple spatially separated base stations. This has the potential to dramatically improve both wireless coverage and spectral efficiency especially in cases where coverage is limited by inter-cell interference. Collaborative MIMO (C-MIMO) approaches are already being adapted in cellular service outdoors, but with accessible propagation-aware time synchronization, this could be applied to femtocell and other high-speed wireless found in indoor environments.

The most common form of nanosecond accurate time synchronization is derived from Global Positioning System (GPS) signals. Unfortunately GPS signals cannot easily penetrate buildings and are often distorted by multipath in urban environments. The best wired time synchronization solutions like the Precision Time Protocol (PTP) [1] can achieve accuracies as low as 25 ns, but these access points require both wires and expensive switches throughout the network. Wireless PTP systems are being designed to operate on existing WiFi channel bandwidths which significantly limits their timing resolution (802.11ac has a maximum bandwidth of 160 MHz) and are still predominantly research prototypes that are not commercially available. The Pulsar platform provides an easy-to-use synchronization system for indoor wireless and scientific applications that wish to decouple time synchronization from system details like the underlying communication protocols.

Time synchronization protocols suffer from five main sources of error that are associated with: (1) transmit time, (2) propagation time, (3) receive time, (4) residency delay and (5) clock instability. Transmit and receive timing errors result from the jitter and offset when timestamping packets. Residency delay results from messages being in buffers after a packet has been constructed. In message passing protocols like the Network Time Protocol (NTP) [2], the majority of error is associated with asymmetry in round-trip message passing times. Propagation time is the delay resulting from the time it takes a signal to travel over the air or through a medium like wire or fiber. One nanosecond corresponds to the time it takes light to travel approximately 30 cm. An offset of 100 ns could simply be a 30 meter difference in distance. PTP uses hardware-level timestamps to estimate propagation time of network signals. This is extremely difficult in wireless systems because of the error in timing associated with locking onto preambles in a noisy channel. Finally, clock instability is the result of error in the frequency of local oscillators that can change depending on physical properties like temperature or crystal aging. A quartz crystal found in a typical electronic device could drift by as much as 1 μs per second. Even the best oven controlled oscillators begin to drift on the order of nanoseconds per second over periods longer than a few seconds.

Given these error sources, an ideal wireless time transfer system would benefit from two main technologies: (1) a stable clock source to minimize drift and message passing overhead and (2) a radio that operates across a wide bandwidth to improve the theoretical range resolution, and one that can perform accurate timestamping of packets. Pulsar leverages innovations on both fronts by using a Chip-Scale Atomic Clock (CSAC) as its primary clock source and a commercially available UWB radio capable of sub-nanosecond packet time stamping. For the radio, we use the DW1000 UWB chipset from Decawave [3] that provides a nominal 15.6 ps timestamp precision of packet transmit and reception through use of equivalent time sampling on a repetitive pulse train. The combination of stroboscopic sampling and the fact that UWB uses short pulse durations, make these radios ideal for precise timestamping and ranging applications. The DW1000 is designed primarily for time-of-flight (TOF) ranging applications and can provide centimeter level distance corrections when given line-of-sight. We use these distance estimates to account for speed-of-light propagation delays. Second, the Pulsar includes a Quantum SA.45s CSAC that provides a short-term stability (Allan Deviation) of $2.5 * 10^{-10}$ with an averaging period (τ) of 1 s. The CSAC is connected directly to the UWB radio and an ARM processor using a programmable low-jitter phase-locked loop (PLL). The high stability and low drift of the CSAC not only improves the DW1000 in terms of frequency locking performance, but it enables synchronization and ranging over longer intervals which improves multi-hop performance.

One of the main challenges in our system is utilizing DW1000 timestamps in a manner that allows for precise clock conditioning. The digital subsystem of the DW1000 runs at 38.4 MHz which means that all I/O is discretized to 26 ns. A significant contribution of this work is that we provide a simple hardware mechanism for pushing synchronization accuracy below this I/O discretization level. We utilize a PLL to synchronously clock the radio and processor subsystems while using the CSAC PPS signal as a common event for time stamping. Since the PLL provides frequency locking but cannot phase aligned to the input clock from the CSAC, the radio and clock have an unknown phase offset up to the 26 ns time discretization. We are able to improve error by using a phase measurement sub-system to measure the error between the PPS signal and the outputs from the PLL. We then compensate for this phase error in software to achieve below 5 ns of accuracy. The final output of our system is a synchronous PPS signal along with a phase locked 10 MHz output that is a standard for synchronizing communication equipment like Software-Defined Radios (SDR).

In propagation-aware time transfer systems the device location and timing accuracy are tightly coupled. In protocols like NTP and PTP, time is distributed along the edges of a tree. Prior work has shown that not all links and clocks should be treated equally [4]. One of the benefits of the broadcast nature of wireless communication is that multiple nodes within a network can perform pair-wise ranging with each other to capture information about the topology with a greater number of links as compared to wired systems. As part of Pulsars synchronization protocol, we have a graph realization and a low-jitter link selection step where the system collects range measurements between nodes to capture the topology of the network. Graph realization and link profiling helps find routes that minimize jitter caused by non-line-of-sight (NLOS) communication. This graph also provides the physical location of nodes that is a critical component to many wireless applications.

In summary, the contributions of this paper are: (1) a novel hardware platform that is able to perform wireless time-of-flight propagation-aware clock synchronization at better than 5 ns resolution per communication hop that can be easily integrated with existing SDR systems, (2) an end-to-end analysis and evaluation of timing uncertainty provided by the platform and (3) the design of a propagation-aware clock synchronization algorithm.

II. Related Work

In this section, we discuss related work in clock synchronization and look at mechanisms for accurate ranging, which can be used to remove propagation delay errors. We also discuss related work from the wireless MIMO community.

A. Clock Synchronization Approaches

Significant effort has addressed establishing a common notion of wall-clock time. The Network Time Protocol [2] (NTP) uses round-trip message delay averaging to set times. We adopt many similar concepts to NTP like clock discipline and network-delay estimation. Various message passing approaches have looked at minimizing access, transmission and reception time in wireless systems. The reference broadcast synchronization [5] (RBS) scheme uses timestamps exchanged between multiple receivers to eliminate all transmission delays with the exception of propagation delays. This approach targets the sources of timing jitter associated with wireless devices and averages over multiple transmissions to achieve tight pairwise clock synchronization. The Pulsar platform adopts a similar approach using beacon messages, except that it adjusts for propagation delays. The flooding time synchronization protocol [6] and the time-sync protocol for sensor networks [7] use hardware timestamping to eliminate these similar sources of timing jitter. Messages are flooded across the network forming a spanning tree that periodically compensates for drift. Local clock rates are adjusted to help reduce drift, which could also be achieved using our module. In their original form, propagation delay was not significant compared to achieved accuracy. Both approaches could be applied to the Pulsar platform and would improve their performance due to its fine-grained timestamps.

Multiple synchronization approaches leverage external hardware to receive global time broadcasts. The WWVB atomic clock broadcast from NIST uses a 50 kW radio tower located in Boulder, Colorado to transmit a 60 Khz time beacon. This is ideal for outdoor applications within the tower's broadcast range, but the radio transmission does not penetrate far into buildings. The signal also suffers from high levels of jitter with offsets due to the long transmit distances. The Global Positioning System (GPS) [8] uses precise clock synchronization derived from satellite transmissions for localization and timing. This is achieved using the Time-Difference-of-Arrival (TDOA) of radio messages to estimate location as well as synchronize receiver clocks with an atomic clock driven infrastructure. Unfortunately, GPS does not penetrate buildings and requires at least three satellites in order to compute precise time (a single satellite gives crude time on the order of micro-seconds since it cannot determine distances). GPS time receivers have commonly been used as sources to discipline NTP servers and often use temperature controlled oscillators to improve timing stability. These have even been implemented in software for wireless sensor networks [9].

Previous efforts have shown a number of protocols and time synchronization specific to IEEE 802.15.4 networks. In high multi-path environments, the channel coherence and time resolution is theoretically limited to 200 ns by the 5 MHz bandwidth of the channels [10]. UWB radios like the DW1000 work around this problem by using channels that are 500 MHz or 1 GHz. Clock synchronization schemes like Glossy [11] rely on symbol level constructive interference to perform better than micro-second synchronization. Reverse flooding [12] is similar to Glossy, and additionally compensates for propagation delays. One benefit of this scheme is that it does not require the maintenance of spanning trees in the network. While impressive, the achieved performance again highlights the limitations resulting from I/O discretization (42 ns or 1 clock cycle on their system). The results presented seem optimistic (likely experiments in low multipath areas) in a multipath environment due to the theoretical time resolution limit. Due to a differences in the physical layer, these schemes cannot be easily applied to UWB radios. By utilizing a larger signal bandwidth along with the tightly coupled clock distribution system allows Pulsar to improve upon these state-of-the-art systems by a factor of more than 20 (5 ns vs $\sim$ 100 ns). Due to a differences in the physical layer, this scheme cannot be easily applied to UWB radios.

In [13] the authors present an approach that uses known locations of beacons (using GPS) to back compute propagation times. This approach could utilize the Pulsar platform and given their required time scales, would likely see a significant improvement in terms of the number of supported nodes. Time-of-flight aware time synchronization [14] takes a similar approach for propagation-time estimation as to what we propose; using a sub-GHz CC1101 radio for internal synchronization but with a more sophisticated algorithm. Like most existing platforms in the WSN community, they operate in the 200-1500 μs range. However, it uses similar primitives and the protocol can be applied to the Pulsar platform in the future.

In [15], the authors study using nearly simultaneous receptions of various sources, both natural and man-made, for synchronization. For example, optical pulsars (from which we adopt our name) can broadcast flashes of light simultaneously visible to large regions of the earth. These reception times are known to be nearly simultaneous to all viewers and hence can be used as synchronization points. The best performing wireless time synchronization to date has been achieved at NIST [16] and is able to synchronize clocks optically down to one femtosecond over 100 ms across a 4 km free space link. Designed primarily for ultra-precise scientific experiments, this impressive system is a research prototype that is large, fixed, expensive and fragile. It also requires direct optical LOS which is difficult to achieve in indoor spaces.

B. *Software-Defined Radios and Collaborative MIMO*

There have been multiple proposed approaches for tight clock synchronization from the wireless community. [17] proposed using power-line communication (PLC) as a back channel for wireless synchronization. The system is able to achieve an average synchronization accuracy of 225 ns with as high as 400 ns. While ideal for small area clock distribution, PLC requires repeaters to go across circuits and can be susceptible to noise that is difficult to eliminate. SourceSync [18] presents a system that is able to harness sender diversity through tight time synchronization using an SDR. The system is able to achieve better than 20 ns time synchronization, but is limited to a single collision domain and like many SDR-based approaches requires modifying the underlying MAC protocol to include synchronization capabilities. AirSync [19] and JMB [20] use similar approaches with SDRs that again modify the underlying MAC which makes it difficult to adapt to standard protocols. In contrast, our approach provides an external input at 5 ns per hop across a network with the sole purpose of providing synchronization. This decouples the time synchronization from the underlying wireless MAC. The results from JMB also emphasize the importance of time synchronization in C-MIMO: every 0.1 radian (64 ns) of phase error between the transmitters decreases the SNR at the receiver by 2 dB, which correspondingly affects the system throughput.

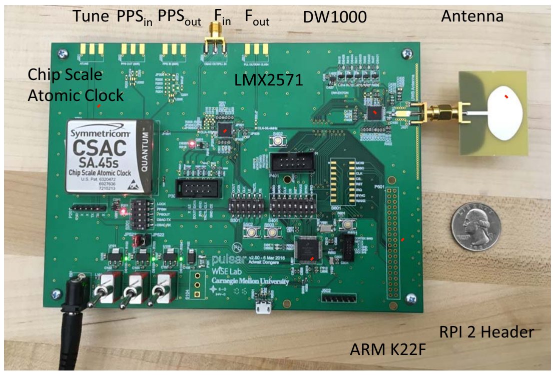


Fig. 1. Pulsar hardware photograph

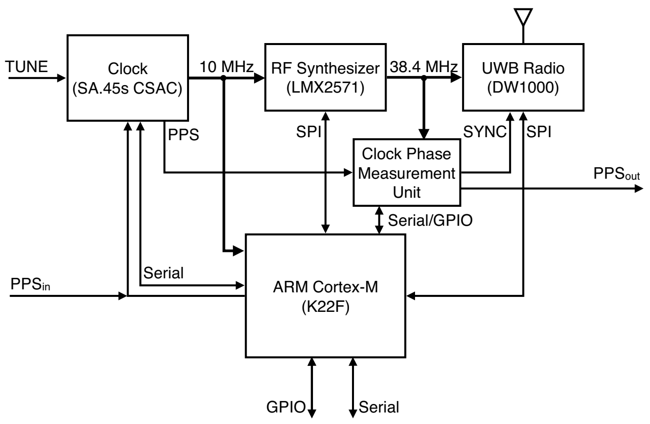


Fig. 2. Pulsar Block diagram with interconnects.

III. Platform Design

In this section, we discuss our hardware design and then address our specific sources of synchronization error. We look at how each source of error can be reduced given our proposed architecture. Our platform is open-source with the all of the hardware and code available on Upverter[1] and GitHub[2].

A. Pulsar Hardware

The Pulsar platform, shown in Figure 1, is 18 cm by 12.5 cm. The block diagram of the Pulsar board in Figure 2 shows five main components: (1) the CSAC, (2) a frequency synthesizer, (3) a UWB radio, (4) an ARM processor and (5) a clock phase measurement unit. The entire Pulsar board consumes a peak of 200 mA at 3.3V, most of which is consumed by the radio and the CSAC heating element. The output of our system is a one pulse per second (1PPS) signal along with a phase locked 10 MHz clock which will be synchronized across the entire network of nodes.

The CSAC is a Microsemi SA.45s module that outputs a 10 Mhz signal with a short-term stability (Allan Deviation) of $2.5 * 10^{-10}$ over a one second averaging period with a long-term aging of less than $9 * 10^{-10}$ per month, and a maximum frequency change of $5 * 10^{-10}$ across a temperature range of -10 to 35 degrees Celsius. The CSAC has the ability to be disciplined from an external high-precision PPS source (PPS_{in} synchronization) improving its phase and frequency performance to within 1 ns and 1 *part per (pp)* 10^{12} respectively. In our experiments, we pre-calibrate the clocks from a single GPS source. The CSAC has a variety of I/O including PPS_{in}, PPS_{out}, an analog tuning input for phase adjustment and a digital interface over serial. It can digitally servo at up to a maximum frequency steer of four pp 10^8 through a serial interface or the analog tuning input. Since it would take an extremely long time to servo PPS outputs into alignment, we feed a GPIO pin from our main processor into the PPS input of the CSAC as part of an initialization process (manual PPS_{in} synchronization). Internally, the CSAC uses an oven controlled crystal oscillator (OCXO) that is disciplined at 1 Hz by a resonance cell containing rubidium 87 that is heated into a vapor. The vapor is illuminated with light from a semiconductor laser diode which is naturally modulated at 6.834 GHz. Once the laser drives the atoms into an oscillating state, they absorb less light, which allows the system to determine if the light is modulated at the same frequency as the atomic source. Based on the light intensity, an inner control loop servos the OCXO frequency. Using the excitation of rubidium atoms as a reference is what provides the long-term frequency stability.

The 10 MHz output from the CSAC is connected to a LMX2561 low-jitter frequency synthesizer from Texas Instruments and to a hardware counter on the main ARM processor. The frequency synthesizer is used to convert the 10 MHz signal into a 38.4 MHz signal that can drive the UWB radio and other related subsystems. The LMX2561 contains a fractional PLL that can be programmed to generate any frequency from 10 MHz to 1344 MHz with very low phase noise. In systems that lack a tunable clock source (like a CSAC), the PLL can also be used to tune an incoming clock signal. Introduction of a PLL into a clock system results in the loss of absolute phase information regarding the output signal with respect to other signals (e.g. 1PPS and 38.4 MHz). Fortunately, stable PLLs will introduce a phase offset which is held constant during lock and we can use this insight to measure and compensate for the error.

At startup, the PLL and CSAC are configured by an NXP Kinetis ARM K22F Cortex-M processor running at 120 MHz. The ARM processor has a variety of connections to control all of the Pulsar's subcomponents as well as interconnect with external devices using a RPI2 compatible header. The main processor has 512 KB of Flash, 128 KB of SRAM, an FPU and on-board DSP.

As previously mentioned the DW1000 UWB radio has the ability to timestamp packet arrival with a resolution of 15.6 ps through equivalent time sampling of a pulse stream that is part of the message preamble. UWB is an excellent communication source for ranging applications since the pulses can be made to be extremely narrow in time and hence wide in frequency. From radar literature, we know that range resolution is derived from a time bandwidth product. The DW1000 has a synchronization line (SYNC) that can be used to reset an internal 40-bit counter that increments at 64 GHz or deterministically trigger a radio transmission. This SYNC input can be used to reset the system time-base for the radio messages.As is the case with most digital radio platforms, the SYNC pin will only be read on the next rising edge of the 38.4 MHz clock driving the I/O subsystem of the radio. This introduces up to 26 ns of error unless the source driving the SYNC line is phase aligned with the 38.4 MHz. To achieve synchronization accuracies below the I/O discretization level of the radio's digital system, we feed the raw CSAC PPS signal into the SYNC line (but only reseting the time-base when required) and then use a clock phase measurement unit (CPMU) to determine

[1]https://upverter.com/WiselabCMU/eab20f02c4d4f096/Pulsar-V2/
[2]https://github.com/WiseLabCMU/pulsar-code

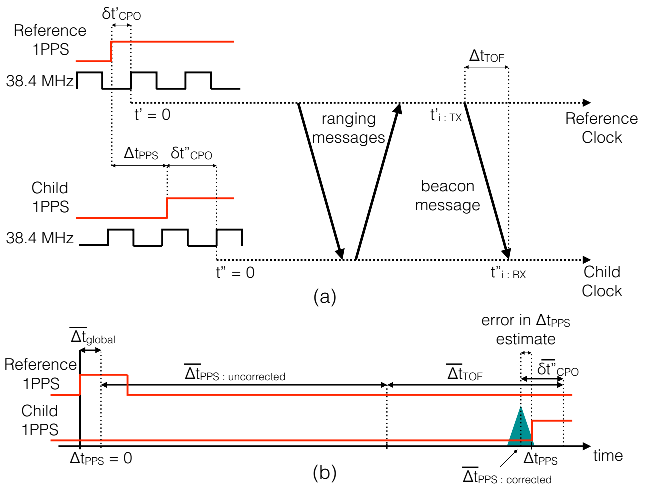


Fig. 3. Timing in the Pulsar platform for phase offset estimation

the phase error between the PPS input and the next 38.4 MHz clock edge. Knowing this phase error allows us to correct the DW1000 time stamps to within a few nanoseconds. This phase error only needs to be computed once at startup and can then be removed as a static offset from the received time stamps in software. In our current hardware implementation, we perform the CPMU measurement externally and feed the phase error back into the main ARM processor over serial. The CPMU may be implemented on-board using an FPGA or time-to-digital converter ICs like the Texas Instruments TDC7200.

B. *Sources of Error*

A variety of errors accumulate in the timing system, which make nanosecond clock synchronization difficult. Some of these errors can be identified, some statistically filtered out while others are completely unobservable and dependent on the system architecture. A critical part of this work is to identify and mitigate various sources of timing errors encountered during synchronization.

Frequency offset and stability errors are the easiest to identify and correct. Receiver-only systems (like the one we describe in Section IV-D) can be used to calibrate for frequency offsets by locking on to signals with known time differences. Frequency stability is a fundamental error source modeled using Allan deviation described later in Section III-B1.

Phase errors are more complex to estimate and correct, stemming from the difficulty in establishing a common reference point across the multiple clock domains found in typical electronic systems. Phase is also highly susceptible to various types of propagation delays in the signal path that do not affect frequency estimates. Figure 3 describes the various phase error sources in our system.

The main quantity of interest is the time offset (Δt_{PPS}) between two 1PPS signal lines on different nodes. If the radio clocks on the nodes are to be started perfectly in synchronization with the 1PPS line ($\delta t_{\text{CPO}} = 0$ for both nodes) then timestamps for a message i provide an estimate of the 1PPS offset:

$$\Delta t_{\text{PPS:uncorrected}} = t''_{\text{i:TX}} - t'_{\text{i:RX}}$$

This estimate does not consider propagation delays due to time-of-flight (Δt_{TOF}) that can be computed and compensated for through message passing as described later in Section IV-A.

As shown in Figure 3, as the clock radio does not start at the same instance as the start of the 1PPS line for two reasons: (a) electronic signals take finite time to rise before the radio's CMOS logic can detect them and (b) the digital I/O on the UWB radio only samples on the positive edges of it's 38.4 MHz I/O clock. We call the combined error due to these effects the *clock phase offset (CPO)* which is represented by δt_{CPO} for each node. The PLL used to bridge our 10 MHz CSAC clock domain and 38.4 MHz UWB radio clock domain, locks the relative phase between them in a 25-to-96 ratio, but we lose information about the absolute phase difference between them which was previously provided by the 1PPS line in the 10 MHz domain.

t'_{i} and $t"_{\text{i}}$ are timestamps for message i on the reference and child nodes respectively. Assuming ideal timestamping on both nodes, Figure 3 (a) gives us the following

$$\Delta t_{\text{PPS}} = t'_{\text{i:TX}} - t''_{\text{i:RX}} + \Delta t_{\text{TOF}} + \delta t'_{\text{CPO}} - \delta t''_{\text{CPO}}$$

We design our synchronization protocol to operate on a spanning tree across the network to simplify distribution to intermediate nodes. We combine the 1PPS offsets computed from upper layers of the tree with $\delta t'_{\text{CPO}}$ into a single variable Δt_{global} that can be passed to the lower layers. This results in the final offset estimation expression as:

$$\begin{aligned} \Delta t_{\text{PPS}} &= \Delta \bar{t}_{\text{PPS:uncorrected}} + \Delta \bar{t}_{\text{TOF}} + \Delta \bar{t}_{\text{global}} - \delta \bar{t}_{\text{CPO}} + \epsilon_t \\ &= \Delta \bar{t}_{\text{PPS:corrected}} + \epsilon_t \end{aligned} \quad (1)$$

where ϵ_t is the error in estimation.

1) Allan Deviation: The traditional characterization of oscillator stability is a plot of Allan deviation, defined using a series of relative frequency estimates between a clock and a reference [21] [22]. Each point on the Allan deviation (σ_y) graph denotes the expected standard deviation in the relative clock frequency (y) for a given sampling interval (τ).

$$\sigma_y^2(\tau) = \frac{1}{2}\left\langle (\bar{y}_i - \bar{y}_{i-1})^2 \right\rangle_i \quad (2)$$

An Allan deviation plot helps understand the limits of an oscillator, with respect to frequency and phase stability. This can then be used to select an optimal message passing rate (τ_{update}).

We measure Allan variance using two nodes placed in close proximity (approximately 1.5 m) with line-of-sight of each other. The radio is configured with the default bandwidth that improves timestamping performance as described in Section V. A GPS calibrated Pulsar board is used as a transmit-only reference node with various receiver nodes. In Figure 4, we compare receiver nodes clocked by a regular Quartz crystal, a temperature-compensated crystal oscillator (TCXO) and another CSAC. Message transmit and receive timestamps are collected over a period of 10 hours and used to estimate fractional frequencies as described in Section IV-A. Allan deviation for *good* clocks (exponential phase noise and Gaussian frequency noise [22]) does not vary much over short intervals. This can be leveraged for estimation if messages are not equally spaced in time. Since the measurements are performed with the complete Pulsar system, they also include any additional errors added by the RF synthesizer, UWB radio and processor.

Allan deviation plots for *good oscillators* are smooth with two distinct parts: the negative slope phase line at lower intervals and the positive slope frequency line at higher intervals. Phase noise is added by PLLs, time-discretization, interrupts (in case of software timestamps), timestamping algorithms etc. and shifts the phase line upwards. This can be observed since we see that the CSAC-clocked Pulsar's phase line is higher than the datasheet value due to phase noise added by other components and the reference CSAC. Frequency noise may be added by factors like temperature variation, motion, errors in frequency locking, etc. The intersection point of these two lines is an oscillator characteristic called the *Allan Intercept*. The effective update time for a clock synchronization protocol must be less than or close to the interval period of the Allan intercept for the best performance synchronization. For our Pulsar experiments, this is set to one second.

C. Pulsar Firmware

The firmware on the Pulsar platform is responsible for configuring the hardware peripherals, tracking peripheral failures as well as arbitrating the message passing and synchronization protocol. We provide a set of FreeRTOS v8.2.3 task routines and driver functions.

A set of watcher tasks are responsible for initializing each of the CSAC, RF Synthesizer and UWB radio. They keep track of events like peripheral lock, reset and halting, and informing dependent tasks of these events. The radio monitor task is also responsible for synchronizing the radio clock with the CSAC PPS so that they share a synchronized time-base. The CSAC watcher task is additionally responsible for bootstrapping PPS alignment corrections in the current implementation of our protocol. A messaging task waits for all required peripherals to lock before starting message passing between nodes as required by the protocol. A disciplining task and some synchronization tasks are responsible for computing and applying all necessary phase and frequency corrections (except the PPS bootstrap, which is delegated to the CSAC watcher task). Phase and frequency offset estimates are provided to higher level applications through a Serial interface to internally correct for them. Finally, a command task accepts inputs from the user over Serial to change mode of operations.

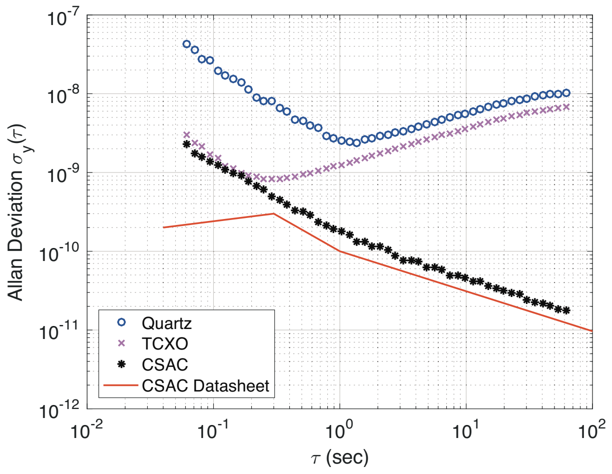


Fig. 4. Allan deviation between nodes given different clocks

IV. Propagation-Aware Clock Synchronization

One of the most challenging aspects of nanosecond synchronization is the need to estimate propagation delay. The protocol described below both estimates range to subtract TOF delay as well as disciplines local clocks.

A. Messaging with Timestamps

The DW1000 UWB radio on the Pulsar platform provides three time-sensitive messaging functions: (a) transmit as soon as possible and record timestamp, (b) transmit at a deterministic future time and (c) receive and timestamp message. Figure 5 describes combinations of these messaging primitives for estimating (and thus allowing for the corrections of) various metrics used in clock synchronization. In our notation, a message i is a beacon message sent by a reference node while $R(i)$ is the response sent by its child.

1) One-Way Messaging: Described in Figure 5 (a), a *reference node* sends messages to a *child node*. These messages could be sent with predetermined transmit times or as soon

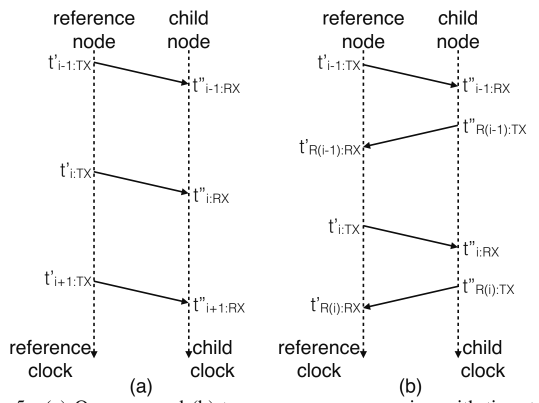
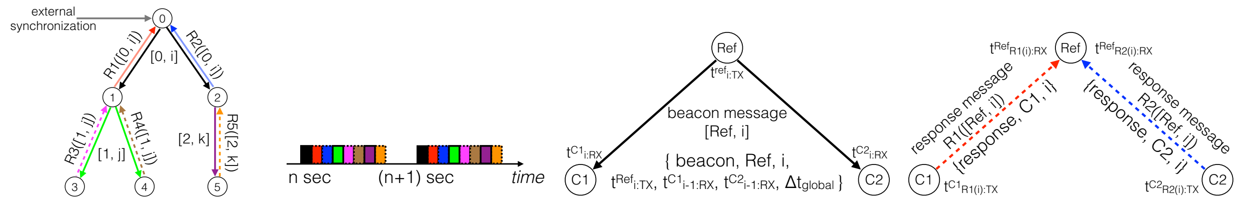


Fig. 5. (a) One-way and (b) two-way message passing with timestamps.



(a) (b) (c) Beacon message (d) Response messages

Fig. 6. A proof-of-concept nanosecond clock synchronization protocol: (a) Nodes are arranged in a tree topology, (b) TDMA schedule is generated for communication and (c)(d) the per-hop messaging scheme

as possible. One-way messaging is sufficient for propagation-agnostic time synchronization protocols like RBS [5].

Multiple one-way message timestamps can compute fractional frequencies with respect to a reference node (y_i) which can then be used for frequency locking stationary devices.

$$y_i = \frac{f_i^{\text{child}}}{f_i^{\text{reference}}} = \frac{t''_{\text{i:RX}} - t''_{\text{i-1:RX}}}{t'_{\text{i:TX}} - t'_{\text{i-1:TX}}} \tag{3}$$

If the radio clocks were started on a known 1PPS signal edge (using the time-base reset functionality of the radio described in Section III-A), one-way messaging can also be used for propagation-agnostic estimation of the 1PPS line offset between a pair of communicating nodes.

$$\Delta t^i_{\text{PPS:uncorrected}} = (t'_{\text{i:TX}} - t''_{\text{i:RX}}) \bmod N \tag{4}$$

where N is the number of clock ticks between two 1PPS edges (nominally the clock frequency).

2) Two-Way Messaging: Two-way messaging as shown in Figure 5 (b) requires both *reference nodes* and *child nodes* to transmit and receive messages. The transmit and receive timestamps from frequency-locked nodes are sufficient for stationary inter-node time-of-flight (and hence, range) estimation.

$$\Delta t^i_{\text{TOF}} = \frac{\left(t'_{\text{R(i):RX}} - t'_{\text{i:TX}}\right) - \left(t''_{\text{R(i):TX}} - t''_{\text{i:RX}}\right)}{2} \tag{5}$$

The error analysis in two-way messaging is well studied in literature [23]. A larger number of messages can be exchanged between the nodes in a generalized N-way messaging scheme, which then attempts to estimate and compensate for higher moments of clock error.

B. Timing Tree Construction

Previous work has shown that particular links or certain clocks exhibit abnormally high levels of variance [4]. As shown later in Section V, UWB radios exhibit increased and often non-Gaussian error in NLOS configurations. For this reason, it is critical to select links within the network that have low-levels of jitter. As part of our network setup, we have a mode that exchanges pair-wise messages between each node in order to capture the link graph and an initial estimate on link variance. We process the graph data using Sparse Full Semi-Definite Programming (SFSDP) relaxation for Sensor Network Localization Problems package in Matlab which generates a graph structure of the network. This provides both position and node geometry. Each link on the graph is weighted based on its variance over 100 messages. There many ways to select a spanning tree across the graph that minimizes cumulative variance. Though not the focus of this paper, we show in Section V that in practice there are cases where minimal hop count, which is often used in PTP, leads to comparatively poor synchronization. It is worth noting that unlike most wireless link assessment problems, timing variance is relatively easy to compute over a series of message exchanges.

C. Protocol

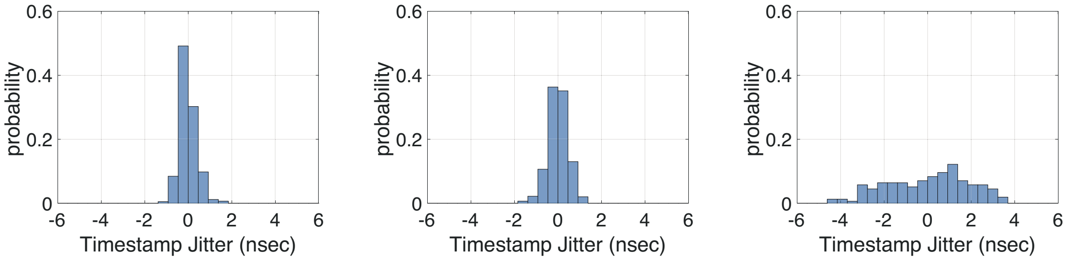
For clock synchronization, we propose a *proof-of-concept protocol* based on PTP that utilizes the messaging schemes described in Section IV-A. Our algorithm has the following prerequisites:

1) A tree-like time distribution network has been created with multiple reference-child relationships as shown in Figure 6(a). The importance of generating a good tree is described in Section IV-B.
2) Based on the network generated previously, a feasible TDMA schedule has been generated for inter-node communication as shown in Figure 6(b) using approaches similar to [24]. A set of beacon message and its responses are combined together to reduce potential errors due to motion, temperature variance etc.
3) The update rate of the algorithm (τ_{update}) has been determined using the clock parameters and the Allan deviation curve for the nodes.

For simplicity, we assume that each node has a list of relevant communication links, update rates and TDMA slots, and these do not change during the process of synchronization. The content of each message is shown in Figure 6(c) and Figure 6(d).

The *reference node algorithm* is described below. $\Delta t_{\text{external}}$ and $\Delta t^{\text{ref}}_{\text{global}}$ are updates from external synchronization and upper-levels of the tree respectively, if applicable.

```
1: Perform a radio time-base reset on the 1PPS edge
2: Update $\Delta t_{\text{global}} = \Delta t_{\text{external}} + \Delta t^{\text{ref}}_{\text{global}} + \delta t_{\text{CPO}}$
3: Listen for valid child node responses and record $t^{\text{Ref}}_{\text{i:TX}}$ to
   the beacon message buffer
4: Send a beacon message at time $t^{\text{Ref}}_{\text{i:TX}}$ in the allotted slot
5: **go to** step 2
```



(a) Range = 7.56 m (b) Range = 17.69 m (c) Non-Line-of-Sight

Fig. 7. Timestamping jitter over various distances and with obstacles.

The *child node algorithm* operates in two stages (and an optional third bootstrap stage described in Section IV-D). The first stage is listen-only and handles frequency estimation and locking. $\bar{\sigma}_y$ is standard deviation in fractional frequency, $\sigma_y(\tau_{\text{update}})$ is from the Allan deviation curve and α is an empirically determined constant. The second stage sends response messages for phase estimation and locking to the reference.

```
Frequency estimation and lock
1: Perform a radio time-base reset on the 1PPS edge
2: Listen for beacon messages from reference node, record
   the receive timestamp ($t^{\text{Cj}}_{\text{i:RX}}$), extract relevant information
   from beacon message ($t^{\text{Ref}}_{\text{i:TX}}$, $t^{\text{Ref}}_{\text{R(i):RX}}$), compute $y_i$ and pass
   information to discipline task.
3: if $\bar{\sigma}_y \leq \alpha\sigma_y(\tau_{\text{update}})$ then go to step 4 else go to step 2
Phase estimation and lock
4: Send a response message in the allotted slot and record
   the transmit timestamp ($t^{\text{Cj}}_{\text{R(i):TX}}$)
5: Estimate $\Delta\bar{t}_{\text{TOF}}$, $\delta\bar{t}_{\text{CPO}}$, $\Delta\bar{t}_{\text{PPS}}$, $y_i$ as described in Sec-
   tion III-B & Section IV-A and pass these to the clock
   discipline task
Phase Correction Bootstrap
6: if Bootstrap is enabled and $\Delta t_{\text{PPS}} > \Delta t_{\text{PPS:threshold}}$ then
7:     Set the clock to manual PPS synchronization mode
8:     Trigger PPS$_{in}$ at the closest 10 $MHz$ clock edge
9:     go to step 1
10: else go to step 2
```

This approach only disciplines the clock on the child node, as compared to more sophisticated protocols which may discipline both the reference and child clocks. If a node is to function as a relay (both reference and child simultaneously), it would start in child mode and wait for frequency lock ($\bar{\sigma}_y \leq \alpha\sigma_y(\tau_{\text{update}})$) to release its reference task. If phase alignment is also enabled, then we wait until both frequency and phase are within predetermined bounds ($\Delta t_{\text{PPS}} \leq \Delta t_{\text{PPS:threshold}}$, a predetermined value, in addition to the frequency lock condition) before releasing the reference task.

D. Clock Disciplining

Controlling clock frequency is an essential requirement for phase estimation (and correction) in our protocol. Our current implementation uses a PID feedback loop designed around fractional frequency estimates and corrections. This is sufficient for frequency correction, but not necessarily for phase correction. The discipline task on a child node waits for new y_i estimates and applies the following correction.

$$y^{\text{Cj}}_{\text{steer}} = F_{\text{PID}}\left(y_{err} = y_i, y_{set} = 1, [K_p, K_i, K_d]\right) \quad (6)$$

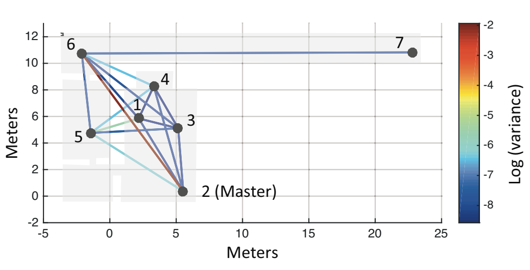
Based on implementations in NTPv4 [25], this could be modified into a hybrid PLL + FLL controller that can correct for both phase and frequency offsets. Since the maximum frequency steer in the CSAC is limited to 2 pp 10^8, a worst case PPS offset of 1 s would take more than 3 years to correct. Thus, we add a bootstrap section to the child node algorithm (steps 6 to 10) for phase correction that forces large corrections instantly using manual PPS$_{\text{in}}$ synchronization. All phase-measurements are reset after this operation and must be measured again.

V. Evaluation

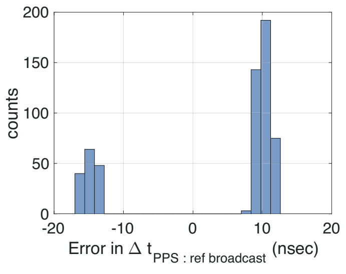
In this section we benchmark the timestamping accuracy of our UWB radio, evaluate link quality for spanning tree generation in our testbed and perform a single-hop evaluation of our clock synchronization and phase estimation protocol with comparisons to a modified reference broadcast implementation. All evaluations for Pulsar are carried out on channel 2 (3774 to 4243.2 MHz) of the DW1000 UWB radio with the slowest data rate of 110 $Kbps$, a preamble length of 1024 symbols and a pulse repetition frequency of 64 MHz. These parameters are chosen to focus on good timestamping performance at the expense of low data rate.

A. Timestamping Jitter

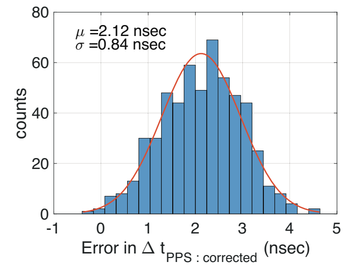
In order to model our system performance, we evaluate the quality and consistency of DW1000 timestamps to determine their error contribution to our clock synchronization. A reference Pulsar node is used in transmit-only mode with a previously frequency calibrated child node in always-listen mode. Beacon messages are periodically sent by the reference node containing embedded transmit timestamps ($t'_{\text{i:TX}}$). Receive timestamps ($t''_{\text{i:RX}}$) are computed on the child node upon successful reception. We evaluate the spread of $t''_{\text{i:RX}} - t'_{\text{i:TX}}$ to estimate timestamp jitter in the combined system. Figure 7 shows the probability-normalized distribution of timestamps for a pair of static nodes separated by various distances and in different configurations. Differing distances do not



(a)



(b)



(c)

Fig. 8. (a) Link variance across testbed network. (b) PPS offset estimation error with modified reference broadcast implementation (that may be used by RBS, FTSP) shows the combined effect of I/O discretization, clock phase offset and other estimation errors. (c) PPS offset estimation error with propagation-aware synchronization.

drastically affect timestamping accuracy ($\sigma_{7.52m} = 0.40\ ns$ & $\sigma_{17.69m} = 0.45\ ns$) which stays accurate to under $0.5\ ns$. However, timestamping consistency drops significantly once received signal quality goes below a minimum signal energy threshold due to non-line-of-sight or the nodes being too far away. Poor links may be unavoidable in a large-scale network, but these are heavily detrimental to synchronization performance. We thus determine that it is essential to identify and prune poor links in the clock synchronization network. Decawave application notes suggest using channel metrics such as first-path power and received signal power to help identify NLOS links.

B. Timing Tree

In order to evaluate network synchronization, we tested our ranging protocol on a seven-hop network deployed across a 300 m^2 area of a campus building. Figure 8(a) shows the node locations with communication links that are colored based on the variance of ranging jitter. The topology was generated using graph realization with an error in 3D of less than 0.1 m per node as compared to laser ground truth. We manually aligned the node cluster to fit the map. The connection between node 2 (master) and 6 has abnormally high variance. Many synchronization protocols would use minimal hop-count as the primary metric for picking a path, but in this case would lead to higher synchronization error for nodes 6 ($\sigma = 0.34\ m$) and 7 ($\sigma = 0.34\ m$). Instead, a variance-based routing scheme would pick node 1 as a relay for 6 ($\sigma = 0.03 + \epsilon_{\text{hop}}\ m$) and 7 ($\sigma = 0.04 + \epsilon_{\text{hop}}\ m$). Thus, we note that it is important to look at the timestamp variance while generating the timing tree.

C. Clock Synchronization

Finally, we evaluate the pairwise clock synchronization performance of two Pulsar boards. Our objective is to estimate the offset between the 1PPS lines between two synchronized nodes. We run our experiment on two nodes that are separated by approximately 3.6 m with Line-of-Sight between each other. Ground truth is collected by connecting the 1PPS lines from the Pulsar nodes to a Saelae logic pro analyzer, that digitally samples the signal lines at 500 $MSps$ over a period of ten minutes. For our baseline comparison, we compare against a reference broadcast implementation [5] with propagation-delay compensation on our platform. A third node acts as the reference broadcaster. Timestamps from response messages are used for delay and phase estimation. Figure 8(b) shows the optimistic pairwise synchronization accuracy of protocols like RBS and FTSP, which use similar primitives. The histogram demonstrates the effects of I/O discretization (two distributions separated by $1/38.4\ MHz \approx 26\ ns$) and an unknown clock phase offset (distributions are not centered at zero and 26 ns).

Next, we evaluate our proposed protocol with both I/O discretization and propagation compensation. For this experiment, two nodes operate in a reference-child tree topology. The child node runs a calibrated PID loop for frequency discipline as described in Section IV-C with $\alpha = 2$ determined empirically. The protocol synchronization update rate $\tau_{\text{update}} = 1\ s$. The raw PPS offset is computed using transmit and receive timestamps recorded from beacon messages. Message passing as described in Section IV-C helps us determine the time-of-flight between the two nodes. The missing clock phase offset parameter is currently measured on an off-board oscilloscope or CPMU module once per run (clock phase offsets are stable as long as all PLLs remain locked). Our final system achieves the pair-wise error distribution in Figure 8(c) which shows our final synchronization accuracy to be better than 5 ns. The mean ($\mu = 2.12\ ns$) and variance ($\sigma = 0.84\ ns$) achieved are within the error bounds expected from various components such as the ground truth measurement on the Saelae logic analyzer, the timestamping inaccuracy introduced by the DW1000 chip, the jitter introduced by the PLL and the frequency errors in our clock source.

VI. Limitations and Future Work

The Pulsar system has two main limitations. First, CSACs are currently still relatively expensive, power hungry and do not operate across a wide temperature range. The current system costs approximately \$1600 in single quantities (\$1500 of which is the clock), consumes a peak of 1 W of power and must be kept between -10 and 35 degrees Celsius. The CSAC cost is predominantly a result of economies of scale and will likely decrease significantly over the next few years if adopted in mass-market products. We also believe that it is possible to achieve nearly the same level of accuracy by increasing the message passing rate with a high precision Oven Controlled Oscillator [26] (that costs less than $100 per clock). Second, multi-path signals caused by RF blocking obstacles will cause increased timing jitter as well as incorrect length estimates. With UWB it is possible to detect packets with high timestamp variance, but currently the only solution is to filter out those links or alert the user to reposition the nodes. We believe that this could be improved through more intelligent selection of timing routes or other forms of distributed synchronization. The drawbacks of our current proof-of-concept synchronization protocol are the need to maintain spanning trees for time distribution, the clock synchronization errors accumulate per hop and the timing quality degrades with increasing depth in the network. One could apply more sophisticated synchronization protocols to this platform in the future.

VII. Conclusions

This paper presented Pulsar, a clock synchronization platform for wireless clock synchronization of indoor devices. The platform combines UWB ranging radios with a stable CSAC timing source that improves upon the state-of-the-art in terms of accuracy. UWB radios are used to estimate TOF ranges between nodes such that the speed of light delays can be estimated and accounted for as part of the synchronization protocol. The CSAC provides long-term stability on the order of 1 μs of pairwise drift per 1.2 days and directly clocks the radio and a PPS output system to provide a phase aligned 1PPS and 10 MHz output signals. We show that Pulsar provides better than 5 ns pairwise synchronization. We also evaluate a synchronization protocol that highlights how the physical link topology can play an important role in synchronization. As technology evolves, we believe that it will become increasingly viable in terms of cost and energy for systems to possess atomic clock stability and fine-grained timestamping capabilities. The Pulsar platform provides an initial means to explore the wide range of next generation wireless applications that will be possible with these future timing systems.

Acknowledgment

This research is supported in part by the Bosch Research and Technology Center (RTC), the National Science Foundation under award CNS-1329644 and TerraSwarm, one of six centers of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA.

References

[1] Ieee 1588 precision time protocol (ptp) version 2. *IEEE*, 2008.

[2] D. Mills. Internet time synchronization: The network time protocol. *IEEE Transactions on Communications*, 1991.

[3] Decawave: http://www.decawave.com/ (viewed 10/12/2015).

[4] Thomas Schmid, Zainul Charbiwala, Zafeiria Anagnostopoulou, Mani B. Srivastava, and Prabal Dutta. A case against routing-integrated time synchronization. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, SenSys '10, pages 267–280, New York, NY, USA, 2010. ACM.

[5] J. Elson and L. Girod and D. Estrin. Fine-grained network time synchronization using reference broadcast. *USENIX OSDI*, 2002.

[6] M. Maroti and B. Kusy and G. Simon and A. Ledeczi. The flooding time synchronization protocol. *Proc. ACM Sensys*, 2004.

[7] S. Ganeriwal and R. Kumar and M. B. Srivastava. Timing-sync protocol for sensor networks. *Proc. ACM Sensys*, 2003.

[8] B.W. Parkinson and S.W. Gilbert. Navstar: Global positioning system - ten years later. *Proceedings of the IEEE*, oct. 1983.

[9] Thomas Schmid, Zainul Charbiwala, Roy Shea, and Mani B Srivastava. Temperature compensated time synchronization. *IEEE Embedded Systems Letters*, 1(2):37–41, 2009.

[10] Kevin J Krizman, Thomas E Biedka, and ST Rappaport. Wireless position location: fundamentals, implementation strategies, and sources of error. In *IEEE Vehicular Technology Conference*, volume 47, pages 919–923. IEEE, 1997.

[11] Federico Ferrari, Marco Zimmerling, Lothar Thiele, and Olga Saukh. Efficient network flooding and time synchronization with glossy. In *Information Processing in Sensor Networks (IPSN), 2011 10th International Conference on*, pages 73–84. IEEE, 2011.

[12] Federico Terraneo, Alberto Leva, Silvano Seva, Martina Maggio, and Alessandro Vittorio Papadopoulos. Reverse flooding: exploiting radio interference for efficient propagation delay compensation in wsn clock synchronization. In *Real-Time Systems Symposium, 2015 IEEE*, pages 175–184. IEEE, 2015.

[13] A. W. Weiser, Y. Orchan, R. Nathan, M. Charter, A. J. Weiss, and S. Toledo. Characterizing the accuracy of a self-synchronized reverse-gps wildlife localization system. In *2016 15th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, pages 1–12, April 2016.

[14] Roman Lim, Balz Maag, and Lothar Thiele. Time-of-flight aware time synchronization for wireless embedded systems. In *Proceedings of the 2016 International Conference on Embedded Wireless Systems and Networks*, EWSN '16, pages 149–158, USA, 2016. Junction Publishing.

[15] David W Allan and HE Machlan. Time transfer using nearly simultaneous reception times of a common transmission. *26th Annual Symposium on Frequency Control*, pages 309–316, 1972.

[16] Jean-Daniel Deschênes, Laura C. Sinclair, Fabrizio R. Giorgetta, William Swann, Esther Baumann, Ian Coddington, and Nathan Newbury. Optical two-way time synchronization at the femtosecond level over a 4-km free space link. In *Imaging and Applied Optics 2015*, page LTh1C.3. Optical Society of America, 2015.

[17] Vivek Yenamandra and Kannan Srinivasan. Vidyut: Exploiting power line infrastructure for enterprise wireless networks. In *Proceedings of the 2014 ACM Conference on SIGCOMM*, SIGCOMM '14, pages 595–606, New York, NY, USA, 2014. ACM.

[18] Hariharan Rahul, Haitham Hassanieh, and Dina Katabi. Sourcesync: A distributed wireless architecture for exploiting sender diversity. In *Proceedings of the ACM SIGCOMM 2010 Conference*, SIGCOMM '10, pages 171–182, New York, NY, USA, 2010. ACM.

[19] H. V. Balan, R. Rogalin, A. Michaloliakos, K. Psounis, and G. Caire. Airsync: Enabling distributed multiuser mimo with full spatial multiplexing. *IEEE/ACM Transactions on Networking*, Dec 2013.

[20] Hariharan Shankar Rahul, Swarun Kumar, and Dina Katabi. Jmb: scaling wireless capacity with user demands. In *Proceedings of the ACM SIGCOMM 2012 Conference*, pages 235–246. ACM, 2012.

[21] David W Allan. Time and frequency(time-domain) characterization, estimation, and prediction of precision clocks and oscillators. *IEEE transactions on ultrasonics, ferroelectrics, and frequency control*, 1987.

[22] David L Mills. Computer network time synchronization. In *Report Dagstuhl Seminar on Time Services Schloß Dagstuhl, March 11.–March 15. 1996*, volume 12, page 332. Springer, 1997.

[23] DecaWave Limited. *APS011 Sources of Error in DW1000 based Two-Way Ranging (TWR) Schemes*, 2014. v1.0.

[24] K Pister and Lance Doherty. Tsmp: Time synchronized mesh protocol. *IASTED Distributed Sensor Networks*, pages 391–398, 2008.

[25] David L Mills. Ntp architecture, protocol and algorithms. Technical report, technical report, Electrical Engineering Department University of Delaware, 2002.

[26] Connor Winfield. *Stratum 3E High Stability Oven Stabilized Oscillator OH200-Series*, 10 2016. Rev. 13.