# Distributed Coherent Transmission Made Seamless

Omid Abari Hariharan Rahul Dina Katabi
Massachusetts Institute of Technology

**Abstract–** Distributed coherent transmission is necessary for a variety of high-gain communication protocols such as distributed MIMO and creating codes over the air. However, distributed coherent transmission is intrinsically difficult because different nodes are driven by independent clocks, which do not have the exact same frequency. This causes the nodes to have frequency offsets relative to each other, and hence their transmissions fail to combine coherently over the air.

This paper presents AirClock, a primitive that makes distributed coherent transmission seamless. AirClock transmits a shared clock over the air and feeds it to the wireless nodes as a reference clock, hence eliminating the root cause for incoherent transmissions. This paper addresses the challenges in delivering such a shared clock. We built AirClock into a small PCB and integrated it in a network of USRP software radios. We show that it achieves tight phase coherence. Further, to illustrate AirClock's versatility, the paper uses it to deliver a coherence abstraction on top of which it demonstrates two cooperative protocols: distributed MIMO and distributed rate adaptation.

## 1 Introduction

Distributed cooperative PHY protocols are theoretically well understood to provide large gains in throughput and reliability in a large variety of scenarios. These include distributed MIMO [1], distributed modulation [2], distributed compressive sensing over the air [3], distributed lattice coding [4], and transmitter cooperation for cognitive networks [5]. These schemes assume that independent wireless nodes can perform distributed coherent transmission–that is, they can transmit their signals without phase drifts with respect to each other.

However, practical radios do not provide distributed coherent transmission. Independent wireless nodes have different crystal oscillators generating clocks with different frequencies. As a result, different nodes always have an offset in their carrier frequencies (CFO); the CFO causes signals transmitted by every pair of nodes to rotate with respect to each other, and their phases to drift over time. Thus, even if two signals start with their phases aligned in a desired manner, the CFO very quickly causes the phases to rotate with respect to each other and the signals to combine in an undesired manner. Typical CFOs between two wireless radios even those that belong to the same technology (e.g., two Wi-Fi radios) vary between 100s of hertz to tens of kilohertz [6], [7]. Such CFOs are large enough to lose coherence even within a single packet.

In this paper, we investigate how practical radios may deliver an abstraction of distributed coherent transmission. Designers of cooperative PHY protocols (distributed MIMO, distributed modulation, etc.) would then leverage this abstraction and free themselves from having to work out the details of coherent transmission. The most straightforward approach for delivering such an abstraction would connect the nodes to a shared clock using wires [8]. Such a system eliminates CFO and ensures coherent transmission. However, it defeats the notion of a wireless network and is not practical for mobile nodes. Alternatively, one may connect each node to a GPS clock. Such clocks use the GPS signal and temperature-controlled crystals to maintain a very low CFO with respect to each other. Unfortunately, however, GPS clocks are power-hungry, cost hundreds to thousands of dollars, and do not work in indoor settings [9]. As a result, they are neither suitable for sensor nodes nor indoor Wi-Fi deployments. In the absence of a suitable generic abstraction for distributed coherence, most wireless cooperation protocols have remained theoretical [4], [5], [10]. The few protocols which were demonstrated empirically address the coherence issue within a particular context. For example, systems like [11], [6] implement distributed MIMO, but focus specifically on OFDM systems in their phase tracking and compensation algorithms. In contrast, solutions like [12] focus on the RFID technology, where nodes are passive reflectors that do not have CFO. Neither of these solutions however provide a generic coherence abstraction that can be leveraged by various cooperative PHY protocols, and applied broadly across technologies (Wi-Fi, ZigBee, Bluetooth).

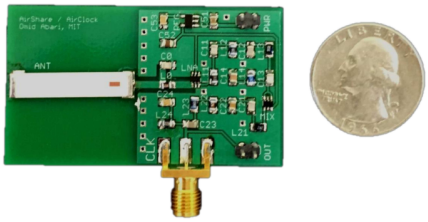



**Fig. 1—Our AirClock Prototype.** The board includes an on-chip antenna.

Ideally, one would like a solution that: (a) avoids wires and supports mobility. (b) Further, it should be independent of the protocol and the radio technology so that it might be used by a variety of technologies (e.g., Wi-Fi, ZigBee, Bluetooth) to build existing or future distributed communication protocols. (c) Finally, it should be cheap and low-power so that it may be incorporated with cheap wireless nodes such as sensors.

This paper presents AirClock, a primitive that enables distributed coherent transmission while satisfying the above three requirements. At a high-level, AirClock transmits a shared clock on the air which wireless nodes use as a reference clock, hence eliminating the root cause for incoherent transmissions.

To demonstrate the practicality of AirClock, we built it into a custom designed printed circuit board (PCB). Our prototype, shown in Fig. 1, is both small and low-cost. We evaluated AirClock and its applications in a wireless testbed with line-of-sight and non line-of-sight scenarios.

## 2 AirClock

In this section, we explain how AirClock works. We start with a description of how radios use a reference clock for transmission and reception, and why the existing system leads

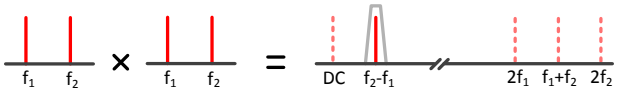



**Fig. 2—Illustration of AirClock's Design.** Wireless nodes multiply the received signal by itself and extract the desired clock signal by applying a band pass filter centered at $f_{ref}$.

to incoherent transmissions. We then describe the structure of AirClock's shared reference signal and how AirClock can be incorporated in a wireless node as an add-on module.

### 2.1 Why Do Wireless Nodes Have CFO?

Wireless signals are transmitted at a particular carrier frequency. The signal is up-converted from baseband to the carrier frequency at the transmitter and down-converted back to the baseband at the receiver. Both conversions are performed by multiplying the signal with the carrier signal.

Each node generates the carrier signal as follows: The node has a local crystal that produces a low frequency sine wave, which is used as a reference clock. A special circuit called Phase Locked Loop (PLL) uses this reference to generate the desired carrier signal.

The key problem is that reference clocks on different nodes have slight differences in their frequencies, because different crystals naturally have different properties. Since the PLLs on different nodes lock to reference clocks with different frequencies, their output signals have different frequencies, and this leads to carrier frequency offsets (CFO) between nodes.

It is important to note that the CFO is not constant. Even minute variations of 0.1° in the temperature can cause CFO variations of a few hundred hertz [13]. Also, noise in the supply voltage cause fast variations in the crystals frequency [13].

### 2.2 How Does AirClock Work?

AirClock eliminates the root cause for incoherent transmission by ensuring that all nodes use the same reference clock that they receive over the wireless medium. However, wireless nodes typically use a 10 MHz to 40 MHz clock for their reference and FCC regulations forbid transmitting such a low-frequency signal for unlicensed use [14]. Also, receiving a signal efficiently at this low frequency requires long antennas [15], which is impractical for typical wireless nodes. [1]

To address this problem, AirClock transmits two single frequency tones (i.e., sine or cosine) separated by the desired clock frequency. These tones might be transmitted in the newly opened white spaces, e.g., for a clock of 10 MHz, AirClock can send tones at 175 MHz and 185 MHz. Let us denote the transmitted tones by $f_1$ and $f_2$, and the desired clock frequency by $f_{ref} = f_2 - f_1$, then the transmitted signal can be written as:

$$S_{tx}(t) = A_1 cos(2\pi \cdot f_1 \cdot t) + A_2 cos(2\pi \cdot f_2 \cdot t). \quad (1)$$

This signal passes over the wireless channel before reception, and the received wireless signal can be written as:

$$S_{rx}(t) = B_1 \cdot cos(2\pi \cdot f_1 \cdot t + \phi_1) + B_2 \cdot cos(2\pi \cdot f_2 \cdot t + \phi_2), \quad (2)$$

1. Note that one cannot simply up-convert the clock at the transmitter and down-convert it at the receiver. Upconversion and downconversion to a band will require independent carrier generation at the transmitter and receiver. Since the reference signals for these independent carriers are generated by different crystals, they will have frequency offset relative to each other, leading to a frequency offset in the retrieved clock signal.

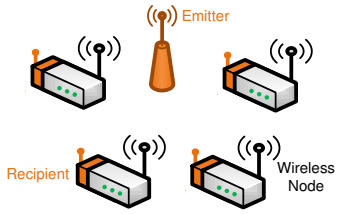



**Fig. 3—AirClock's Network Topology.** An AirClock emitter transmits the reference signal. Wireless nodes that are equipped with AirClock recipient components receive the AirClock signal and extract the reference clock.

where $B_1$, $B_2$, $\phi_1$ and $\phi_2$ capture the channel impact.

To obtain the shared clock, each wireless node multiplies the received signal by itself and applies a band pass filter to extract the desired clock frequency. To see why this works, recall that the multiplication of two tones at different frequencies produces tones whose frequencies are the sum and difference of the original frequencies. Hence, after multiplying the received signal with itself the node obtains:

$$S_m(t) = [B_1 \cdot cos(2\pi \cdot f_1 \cdot t + \phi_1) + B_2 \cdot cos(2\pi \cdot f_2 \cdot t + \phi_2)]^2$$

Simplifying this equation results in:

$$\begin{aligned} S_m(t) = {} & \boldsymbol{B_1 B_2 cos(2\pi \cdot (f_2 - f_1) \cdot t + (\phi_2 - \phi_1))} \\ & + B_1 B_2 cos(2\pi \cdot (f_2 + f_1) \cdot t + (\phi_2 + \phi_1)) \\ & + \frac{B_1^2}{2} + \frac{B_1^2}{2} \cdot cos(2\pi \cdot 2f_1 \cdot t + 2\phi_1) \\ & + \frac{B_2^2}{2} + \frac{B_2^2}{2} \cdot cos(2\pi \cdot 2f_2 \cdot t + 2\phi_2) \end{aligned} \quad (3)$$

This signal includes a DC component, some high frequency components at $2f_1$, $2f_2$ and $f_1 + f_2$, and a component at $f_2 - f_1$ (highlighted in bold in the formula above) which is equal to the desired reference frequency $f_{ref}$. Hence, a simple band-pass filter centered at the reference clock frequency $f_{ref}$ (*e.g.*, 10 MHz) is used to extract the single-tone reference signal, as illustrated in Fig. 2. The signal after the filter will be:

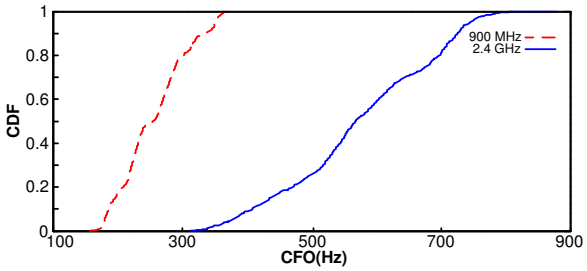
$$\begin{aligned} S_{ref}(t) &= B_1 B_2 cos(2\pi \cdot (f_2 - f_1) \cdot t + (\phi_2 - \phi_1)) \\ &= B_1 B_2 cos(2\pi \cdot f_{ref} \cdot t + \delta\phi). \end{aligned} \quad (4)$$

This signal is then used as an input to the node's PLL. Since all nodes use a reference clock of the exact same frequency, they will have no CFO with respect to each other.

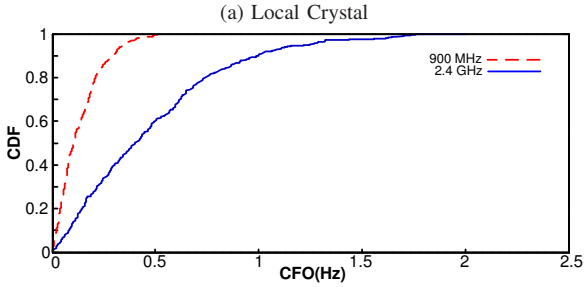
### 2.3 AirClock's System Architecture

Architecturally, AirClock has two components: an emitter and a recipient. To enable a set of nodes to transmit coherently, one deploys a AirClock emitter in the network and equip each node with a AirClock recipient as illustrated in Fig. 3.

**Emitter:** To transmit the AirClock signal, we use a local oscillator (*i.e.*, a crystal) that generates a reference signal, and feed its output to two PLLs to generate two tones, $f_1$ and $f_2$, that are separated by the desired clock frequency $f_{ref}$. The two tones are then amplified using a power amplifier and transmitted on the wireless medium. Note that our signal does not occupy the entire band between $f_1$ and $f_2$, it simply consists of two single-frequency tones which are separated by the desired clock frequency and hence others can transmit in the spectrum between these tones.




(a) Local Crystal




(b) AirClock (Note the difference in the scale of the x-axis)

**Fig. 4—CFO between pairs of nodes at carrier frequencies of 2.4 GHz and 900 MHz: (a) Independent clocks and (b) AirClock.** Comparing (a) and (b), AirClock reduces the CFO by multiple orders of magnitude.

**Recipient:** The AirClock recipient receives the emitted signal. The received signal is passed to a low-noise amplifier (LNA) and the band of interest (from $f_1$ to $f_2$) is filtered out using a band pass filter. After filtering, the signal is mixed with itself and the desired reference clock is extracted using a band pass filter centered at the reference frequency (*e.g.*, 10 MHz), and input to the wireless node's PLL.

Finally, we note two points:

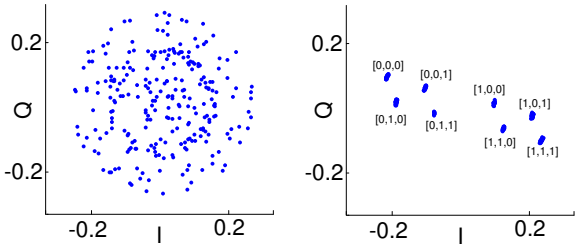
- First, AirClock is protocol and technology independent, and the AirClock recipient circuit can be incorporated in various radios (Wi-Fi, ZigBee, etc.).
- Second, the AirClock recipient circuit is simple, cheap and low-power and hence can be incorporated in low-end wireless nodes. In particular, it is composed of off-the-shelf components and its power consumption is less than 0.1% of the power consumption of a Wi-Fi AP [16], and about 10% of the power consumption of a Zigbee node [17].

## 3 Empirical Evaluation of AirClock

We built a prototype of the AirClock emitters and recipients using off-the-shelf components. We integrate the recipient subsystem with USRP. We evaluate AirClock in an indoor testbed with line-of-sight and non-line-of-sight scenarios. The testbed spans 10m×10m. All experiments in this section are run with USRP nodes that use OFDM, a 1500 byte packet length, and 10MHz bandwidth. The experiments use a total of 6 USRPs. For each evaluation, we run 500 experiments for a variety of the nodes locations.

### 3.1 Eliminating CFO between nodes

A key promise of AirClock is that it can address the CFO problem. We verify if AirClock delivers on this promise. We place an AirClock emitter in one location in the testbed. We place two USRP nodes equipped with AirClock recipients at two random locations in the testbed, with one acting as a transmitter and the other as a receiver. The transmitter




(a) (b)

**Fig. 5—Received constellation for three nodes transmitting BPSK using: (a) Independent clocks, and (b) AirClock. Each point in (b) is labeled with the associated combination of transmitted bits.** Without AirClock, the signals from multiple transmitters do not have a constant phase relationship with each other. Therefore, the received constellation points for a given combination of transmitted signals vary over time. In contrast, with AirClock, the signals from the different nodes are coherent. Hence, the received constellation points for a given combination of transmitted signals stay constant over time.

transmits packets consisting of OFDM symbols. The receiver receives these packets, and computes its CFO with respect to the transmitter. We repeat the experiment for two carrier frequencies: 2.4 GHz and 900 MHz. We repeat each run both with the USRPs operating using their internal crystals and with the USRPs using the AirClock signal as a reference.

Fig. 4 plots the CDF of the observed CFO for both 2.4 GHz and 900 MHz carriers. The graph in Fig. 4(a) correspond to using the internal crystals, whereas the graph in Fig. 4(b) corresponds to using AirClock.

The figure shows that AirClock reduces the CFO by two to three orders of magnitude. Specifically, with AirClock, the median and $95^{th}$ percentile CFO at 2.4 GHz are 0.4 Hz (0.16 parts per billion) and 1.24 Hz (0.5 parts per billion) respectively, and the median and $95^{th}$ percentile CFO at 900 MHz are 0.11 Hz and 0.34 Hz, respectively.

To put these numbers in context, consider the accumulated phase error with and without AirClock for a single 1500B packet at the lowest OFDM rate used by Wi-Fi. This packet takes $\approx 2ms$. Thus, with AirClock the $95^{th}$ percentile phase error across this packet is 0.016 radians, which is negligible and have no effect on coherence. In contrast, in the absence of AirClock, the phase errors across the packet would be between 3.9 to 11.1 radians (*i.e.,* over a 180° change in phase across a packet), and hence the signals are very far from being combined coherently within the packet [11].

### 3.2 Enabling Coherent Transmission

We examine whether AirClock can enable independent nodes to transmit coherently. We place an AirClock emitter and four USRP nodes at random locations in our testbed. One of the USRPs acts as a receiver and the other three as transmitters. The transmitters concurrently transmit random data to the receiver using BPSK. We repeat the experiment with two different schemes: (a) transmitters and receiver using their local crystals, (b) transmitters and receiver using AirClock. Fig. 5 plots the received constellation diagram for these two scenarios.

If the transmitters were coherent with each other, then their signals would combine in a predictable manner across time. In contrast, if the transmitters are not coherent with each other, the same transmitted symbols would rotate relative to each other, and combine in different ways across time producing different received constellation points.

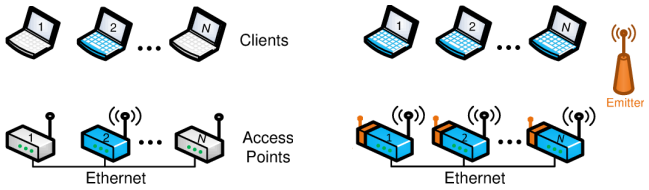



**Fig. 6—Traditional AP deployments (left) vs. Distributed MIMO (right).** A blue node indicates an active transmitter or receiver. With traditional Wi-Fi, only one AP transmits at any time in a given channel. In contrast, with AirClock, multiple APs transmit to multiple clients at the same time in the same channel, thereby scaling network throughput with the number of APs.

We see this latter effect in Fig. 5(a). The transmitters and receiver have significant CFO relative to each other when using their local crystals. As a result, the constellation points produced by joint transmission from the different nodes are smeared uniformly across space. In contrast, when AirClock is used, the received constellation has 8 distinct points (Fig. 5(b)), corresponding to each of the three transmitters transmitting a "+1" or "-1". This is because each combination of transmitted symbols from the three transmitters combines in a predictable manner at the receiver. This experiment demonstrates visual evidence that AirClock provides coherent transmission across wireless nodes.

## 4 Applications of AirClock

Multiple high-gain cooperative PHY protocols assume coherent transmission and hence can benefit from AirClock. We demonstrate AirClock's versatility by explaining how it can be used to build two cooperation protocols: distributed MIMO and distributed rate adaptation.

### 4.1 Distributed MIMO with AirClock

Distributed MIMO beamforming enables independent transmitters to act as if they were antennas on a single virtual MIMO node. Hence, $n$ single antenna transmitters can use distributed MIMO to deliver $n$ packets to $n$ independent clients. By transmitting $n$ independent data units in a unit of time using a unit of spectrum, the system achieves a multiplexing gain of $n$, which translates to a throughput gain that increases linearly with the number of antennas. However for $n$ independent transmitters to act as if they were antennas on a single node, they need to transmit coherently without CFO between them.

While the theory of distributed MIMO has been around for many years, practical implementations have emerged recently [11], [6]. These systems transmit training signals to estimate the rotation due to CFO. They then correct for the impact of the CFO on the channel estimates from different transmitters by applying a time-dependent inverse rotation to the transmitted symbols. These algorithms are OFDM specific, and deeply intertwined with the details of the baseband system.

In contrast, with AirClock, the nodes have a shared reference, which eliminates the need for phase tracking and compensation altogether. It frees the designer from having to think through the interaction of OFDM and coherent transmission, and provides a technology-independent design.

**Evaluation of Distributed MIMO with AirClock** We place an AirClock emitter in our testbed. We also place USRPs with AirClock recipients to act as APs and clients in our testbed.

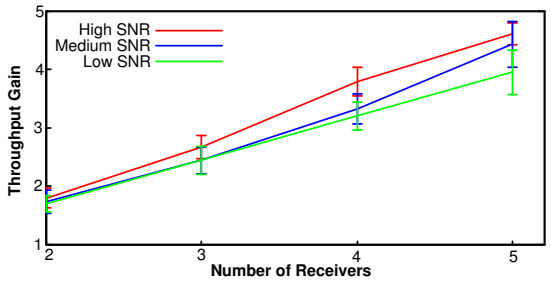



**Fig. 7—Distributed MIMO using AirClock.** AirClock's throughput gain increases linearly with the number of transmitter-receiver pairs in the network.

We evaluate distributed MIMO with AirClock in three different SNR regimes: low (5-10 dB), medium (10-16 dB), and high ($>$ 16 dB). We repeat the experiment for different node placements and different number of transmitter-receiver pairs.

Fig. 7 plots the throughput gain obtained by distributed MIMO using AirClock as a function of the number of transmitting APs, for different SNR ranges. We see that AirClock enables the wireless network throughput to scale with the number of transmitter-receiver pairs, for a gain of $3.95-4.61\times$ across the range of SNRs. This is because, with traditional 802.11, only one transmitter-receiver pair is active at any time irrespective of the number of transmitters. In contrast, distributed MIMO enables all transmitters to transmit jointly to their desired receivers without interfering with each other, and achieving throughput proportional to the number of active transmitters. This shows that AirClock enables distributed MIMO without the need for phase tracking or compensation.

### 4.2 Distributed Rate Adaptation for Wireless Sensors

Sensors typically support only a single modulation scheme, such as on-off keying, BPSK, or QPSK [18]. The modulation supported is low rate so as to ensure that the sensors can communicate even when channel conditions are adverse. Further, sensors avoid supporting high rate (dense) modulations such as 16-QAM, 64-QAM *etc.*, because these modulations require linear transmitter power amplifiers that consume significant power. As a result, wireless sensors do not utilize the wireless channel efficiently. In particular, wireless sensors cannot take advantage of a good channel to send at dense modulation that packs multiple bits into each transmitted symbol.

One can imagine exploiting channel conditions through distributed rate adaptation across the network to overcome the absence of the ability of any single node to adapt its rate. Prior work [12] has proposed such distributed rate adaptation in the context of RFID networks. Specifically, multiple RFID nodes can transmit simultaneously and the receiver receives a collided transmission.The receiver can then decode the individual transmissions from all transmitters using a single collided transmission, if the channel conditions are sufficiently good. If not, it can simply continue to receive additional transmissions and combine these multiple receptions till it can decode the transmitted signals. Such a system will effectively achieve a distributed rateless code across the nodes in the RFID network.

The protocol from [12] described above is designed specifically for RFID networks. RFIDs, however, do not have independent oscillators; they transmit by reflecting a carrier signal from a single device, and hence do not suffer from CFOs relative to each other. In contrast, general wireless devices,

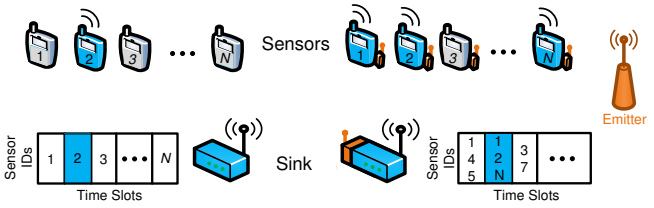



**Fig. 8—Traditional Sensor Networks (left) vs. Distributed Rate Adaptation (right).** A blue node indicates an active sensor. With traditional sensor networks, only one sensor transmits at a time. In contrast, AirClock-equipped sensors perform distributed rate adaptation by transmitting simultaneously. This enables decoding data from multiple sensors in a single joint transmission.

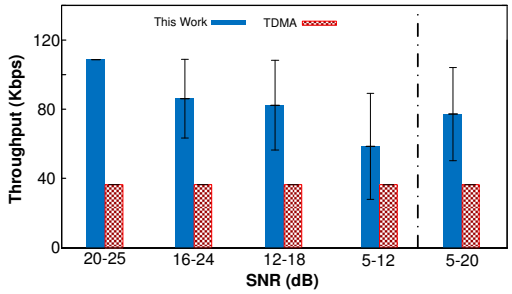



**Fig. 9—Channel Quality versus Throughput for Distributed Rate Adaptation.** AirClock enables distributed rate adaptation for wireless sensors.

*e.g.* sensors, have independent oscillators, which they use for transmission. As a result, transmissions from different nodes rotate relative to each other, and collide in different ways across time preventing the receiver from decoding the transmitted bits from multiple collision receptions. In contrast, AirClock eliminates the problem of differing frequency offsets across sensors. The sensors can therefore perform joint transmission and enable the sink to decode.

**Evaluation of Distributed Rate Adaptation with AirClock:** We deploy an AirClock emitter in the testbed. We use 6 USRPs equipped with AirClock implementing ZigBee and acting as sensors, and one USRP with AirClock acting as a sink. We run 100 experiments for a variety of node locations. We compare distributed rate adaptation with AirClock with TDMA where only one sensor transmits at a time, and the different sensors transmit one after the other.

Fig. 9 plots the throughput of distributed rate adaptation using AirClock, and of traditional TDMA, for different SNR ranges. Distributed rate adaptation achieves $1.64 - 3\times$ the throughput of TDMA. Since TDMA cannot exploit good channel conditions to increase its transmission rate, its throughput is constant independent of SNR. In contrast, distributed rate adaptation can exploit good channel conditions by allowing the receiver to decode multiple simultaneous transmitters from a single collision. Therefore, the throughput gain of distributed rate adaptation increases with the SNR.

## 5 Related Work

One approach for sharing the clock is to equip each node with a GPS disciplined oscillator (GPSDO) or radio-controlled clock. Radio-controlled clocks [19] have 2-6ppm drift (i.e., a CFO of 5-14KHz at 2.4GHz carrier), which is inadequate for coherent transmission [19]. GPSDOs are accurate but cost 100s of dollars and consume 1-10W [9], making them unsuitable for sensors or even APs. Also, GPSDOs do not work indoors. In contrast, AirClock presents a wireless clock that is simple, low-power and low-cost, and can be used in sensors and APs, both for indoor and outdoor scenarios. Concurrently to our work, the authors of [20] proposed using powerlines to distribute a shared clock to the nodes. However, such a design does not address mobile nodes or battery operated sensors.

Finally, some prior work [11], [6] enables coherent transmission for distributed MIMO by designing algorithms to estimate and correct for phase offset between nodes. However, they are designed specifically for OFDM systems in the context of distributed MIMO. In contrast, AirClock's use of a shared clock provides an abstraction for distributed coherent transmission that is independent of technology (WiFi, Zigbee, *etc.*), or application (distributed MIMO, distributed modulation, *etc.*)

## 6 Conclusion

This paper presents AirClock, a system that enables distributed coherent transmission from independent wireless nodes. By sharing a single reference clock across nodes, AirClock provides a coherent radio abstraction that enables implementation of distributed PHY algorithms such as distributed MIMO, and distributed sensor rate adaptation. We believe that AirClock can serve as a building block that brings a large body of distributed information theoretic schemes closer to practice.

## References

[1] A. Ozgur, R. Johari, D. Tse, and O. Leveque, "Information-theoretic operating regimes of large wireless networks," *Info. Theory Trans.*, 2010.

[2] A.-s. Hu and S. D. Servetto, "dFSK: Distributed frequency shift keying modulation in dense sensor networks," in *IEEE ICC*, 2004.

[3] Y. Ji, C. Stefanovic, C. Bockelmann, A. Dekorsy, and P. Popovski, "Characterization of coded random access with compressive sensing based multi-user detection," *CoRR*, vol. abs/1404.2119, 2014.

[4] B. Nazer and M. Gastpar, "The case for structured random codes in network capacity theorems," *European Trans. on Telecomm.*, 2008.

[5] I. Maric, N. Liu, and A. Goldsmith, "Encoding against an interferer's codebook," in *Allerton Conference*, 2008.

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

[7] K. Chintalapudi, B. Radunovic, H. V. Balan, M. Buettener, S. Yerramalli, V. Navda, and R. Ramjee, "wifi-nc:wifi over narrow channels." *NSDI'12*.

[8] Ettus, Universal Software Radio Peripheral, http://www.ettus.com.

[9] Jackson Labs, Fury GPSDO, http://jackson-labs.com/.

[10] W. Bajwa, J. Haupt, A. Sayeed, and R. Nowak, "Compressive wireless sensing," in *IPSN'06*.

[11] H. Rahul, S. S. Kumar, and D. Katabi, "Megamimo: Scaling wireless capacity with user demand," in *SIGCOMM*, 2012.

[12] J. Wang, H. Hassanieh, D. Katabi, and P. Indyk, "Efficient and reliable low-power backscatter networks," *SIGCOMM*, 2012.

[13] HP, "Fundamentals of quartz oscillators," Appl. note 200, Tech. Rep.

[14] FCC, "FCC online table of frequency allocation, April 16, 2013".

[15] C. A. Balanis, *Antenna theory: analysis and design*. Wiley, 2012.

[16] Cisco, "Cisco Aironet 2600 Access Point", http://www.cisco.com/en/US/prod/collateral/wireless/ps5678/ps12534/data_sheet_c78-709514.pdf.

[17] Texas Instruments, "SoC Solution for 2.4 GHz IEEE 802.15.4 and ZigBee Applications.", http://www.ti.com/lit/ds/symlink/cc2530.pdf.

[18] S. C. Ergen, "ZigBee/IEEE 802.15.4 summary," http://pages.cs.wisc.edu/~suman/courses/838/papers/zigbee.pdf, 2004.

[19] NIST., "Radio Station WWVB," http://tf.nist.gov/general/pdf/2429.pdf.

[20] V. Yenamandra and K. Srinivasan, "Vidyut: exploiting power line infrastructure for enterprise wireless networks," in *SIGCOMM*, 2014.