Trinity: Enabling Self-Sustaining WSNs Indoors with Energy-Free Sensing and Networking

FENG LI, Shandong University, China
YANBING YANG, Nanyang Technological University, Singapore
ZICHENG CHI, University of Maryland Baltimore County, USA
LIYA ZHAO, University of Technology Sydney, Australia
YAOWEN YANG, Nanyang Technological University, Singapore
JUN LUO, Nanyang Technological University, Singapore

Whereas a lot of efforts have been put on energy conservation in *wireless sensor networks* (WSNs), the limited lifetime of these systems still hampers their practical deployments. This situation is further exacerbated indoors, as conventional energy harvesting (e.g., solar) may not always work. To enable long-lived indoor sensing, we report in this paper a self-sustaining sensing system that draws energy from indoor environments, adapts its duty-cycle to the harvested energy, and pays back the environment by enhancing the awareness of the indoor microclimate through an "energy-free" sensing. First of all, given the pervasive operation of *heating, ventilation and air conditioning* (HVAC) systems indoors, our system harvests energy from airflow introduced by the HVAC systems to power each sensor node. Secondly, as the harvested power is tiny, an extremely low but synchronous duty-cycle has to be applied whereas the system gets no energy surplus to support existing synchronization schemes. So we design two complementary synchronization schemes that cost virtually no energy. Finally, we exploit the feature of our harvester to sense the airflow speed in an energy-free manner. To our knowledge, this is the first indoor wireless sensing system that encapsulates energy harvesting, network operating, and sensing all together.

CCS Concepts: • **Computer systems organization → Sensor networks**; • **Networks → Network protocols**;

Additional Key Words and Phrases: Sustainable sensor networks, indoor energy harvesting, duty-cycle, synchronization.

This work is partially supported by NSFC (61702304), Shandong Provincial Natural Science Foundation (ZR2017QF005), Key Research & Development Plan of Shandong Province (2017GGX10110), The Fundamental Research Funds of Shandong University (2016HW003) and AcRF Tier 2 Grant MOE2016-T2-2-022. The preliminary results have been presented in Proceedings of the 11th ACM SenSys 2013 [49].

Authors' addresses: Feng Li, Shandong University, School of Computer Science and Technology, China, fli@sdu.edu.cn; Yanbing Yang, Nanyang Technological University, School of Computer Science and Engineering, Singapore, yyang017@sdu.edu.cn; Zicheng Chi, University of Maryland Baltimore County, Department of Computer Science and Electrical Engineering, USA, zicheng1@umbc.edu; Liya Zhao, University of Technology Sydney, School of Mechanical and Mechatronic Engineering, Australia, liya.zhao@uts.edu.au; Yaowen Yang, Nanyang Technological University, School of Civil and Environmental Engineering, Singapore, yyw@ntu.edu.sg; Jun Luo, Nanyang Technological University, School of Computer Science and Engineering, Singapore, junluo@ntu.edu.sg.

1539-9087/0000/0-ART00 $15.00
https://doi.org/0000001.0000001

ACM Reference Format:
Feng Li, Yanbing Yang, Zicheng Chi, Liya Zhao, Yaowen Yang, and Jun Luo. 0000. Trinity: Enabling Self-Sustaining WSNs Indoors with Energy-Free Sensing and Networking. *ACM Trans. Embedd. Comput. Syst.* 00, 00, Article 00 (0000), 25 pages. https://doi.org/0000001.0000001

1 INTRODUCTION

Wireless sensor networks (WSNs) have been involved in many environmental monitoring applications [20], including in particular indoor environments [21, 23, 48]. For such applications, one of the most challenging problems is the conflict between limited lifetime of the WSNs and their purpose for a long-term monitoring [16, 20]. Although harvesting energy from surrounding environment offers a potential solution [27, 36, 52, 54], conventional power sources (e.g., solar) may be unavailable in an indoor environment.

Being able to harvest energy from vibrations, piezoelectric materials have long been claimed as suitable for indoor energy harvesting [11, 51]. In fact, dedicated sensor nodes designed to accept piezoelectricity power can be found in the market [3]. However, the following questions have never been fully answered: i) what kind of vibrations can be harvested indoors to produce sufficient energy? ii) how much power can be produced to support indoor sensing? iii) how to build a power management module for commonly used sensor nodes? and finally, iv) how should a WSN operate under such a harvester? These are indeed questions we intend to tackle in our paper.

One typical indoor application of WSNs is *microclimate control* [45, 46]. As *heating, ventilation and air conditioning* (HVAC) systems are extensively installed indoors and cost a huge amount of energy, indoor microclimate control aims to adapt the output of HVAC systems to population density, which may save energy on one hand while improving the comfort level of occupants on the other hand. To this end, a sensing system is required to "sit" beside the outlets and measure the speed of airflow, whose output is in turn fed to the control system for precise actuation. Obviously, the sensor nodes of this system (close to the outlets) are often far from power grid, and it is not affordable to frequently maintain them (e.g., changing batteries) given the large number of outlets. In fact, such a sensing system (if ready) can serve other indoor applications, such as *illumination control* or *air quality monitoring*. Therefore, we dedicate a prototype of energy harvesting WSN in this paper to such applications.

In this paper, we present Trinity as a self-sustaining sensing system. Trinity harvests energy from the airflow produced by HVAC outlets to power the sensor nodes, and it adapts nodes' duty-cycle to the harvested energy. Moreover, we take advantage of the physical properties of our harvester to i) synchronize sensor nodes such that a sender does not miss its receiver under very low duty-cycle, and ii) create an "energy-free sensor" for detecting the speed of airflow. Note that, because the energy harvested from HVAC system is tiny (only of hundreds of μW), our WSN has to operate under an extremely low and synchronous duty-cycle. Moreover, the off-the-shelf airflow sensors consume higher energy than what our harvester can supply, so an energy-free sensing is necessary for Trinity to work. In summary, we make the following main contributions in this paper:

- We design a special type of energy harvester for drawing energy from indoor airflows with a speed of 2 to 6m/s. Our harvester adopts a bimorph (cantilever with two layers of piezoelectric materials) to convert airflow-induced vibration into electric power.
- We produce a general-purpose power management module to accept piezoelectricity from the harvester. It then powers a connected sensor node and charges the surplus energy into an ultra-capacitor. It also provides necessary functionalities to facilitate our following synchronization and sensing strategies.
- We present two complementary synchronization strategies with very low energy consumptions. The first one calibrates the native clock of a sensor node using the periodic output of

our harvester (i.e., the attached bimorph) when there is no data traffic, and the other one relies on the data traffic to perform constant synchronization.
- We innovate in proposing an energy-free strategy for sensing the airflow speed, where the energy harvester also plays a role of sensor based on its physical features.
- We build Trinity as a prototype of a self-sustaining indoor airflow sensing system. We evaluate its performance by using our research center as a testing site.

In the following, we first briefly introduce the targeted application/problem as well as the Trinity system in Sec. 2. Then we present the three key modules of Trinity, namely airflow-based energy harvesting, synchronous duty-cycling, and power-free sensing, in Sec. 3, 4, and 5 respectively. We also introduce the implementation of Trinity for the application of tree-based data collection in Sec. 6. We report our field tests in Sec. 7, and we survey literature while discussing related issues in Sec. 8 before finally concluding our paper in Sec. 9.

2 PROBLEM AND SYSTEM

We first briefly explain the problem context, then we give a general overview of our Trinity system, as well as the rationales behind its design.

2.1 Indoor Microclimate Control and Sensing

Existing HVAC systems in large buildings are mostly controlled in a centralized manner through an *air handling unit* (AHU). Although tuning the output of certain outlets is possible, it is often not preferred as it normally requires mechanics to physically approach those outlets. Also, as tuning HVAC systems in such a way is not real-time, the outcome may lag far behind the need. For example, when a big but short-term gathering takes place in a certain section of a building (which can be detected by, e.g., [50]), air conditioning may need to run more intensively to cool down the surrounding area. Whereas an adjustment at the AHU causes excessive power consumption for the whole building, tuning the outlets close to the concerning area could be too tardy to be useful for serving the gathering. Consequently, there is an increasing desire on *indoor microclimate control* in recently years, i.e., controlling the individual outlets of a HVAC system such that they adapt to the changes of occupancy in real-time.

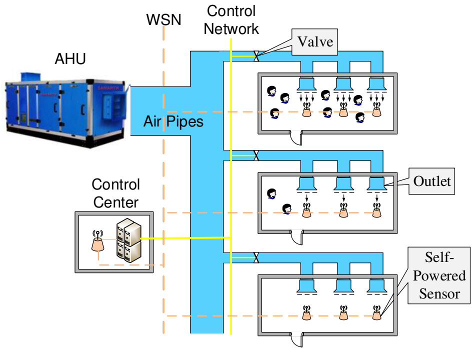


Fig. 1. An indoor microclimate control system.

A key component of indoor microclimate control for precise actuation, airflow speed sensing at individual HVAC outlets delivers an essential feedback to the control unit(s), along with other components such as temperature sensing and occupancy detection [10]. Fig. 1 illustrates an indoor microclimate control system with its airflow speed sensing module highlighted. The data (i.e., the airflow speeds) are sensed at each outlet and gathered to a remote control center, where control decisions are then made accordingly, and finally performed by actuators (e.g., valves in our case). Apparently, a WSN where a number of airflow speed sensor nodes are networked, could be a perfect candidate for data sensing and collection. Nevertheless, these tasks are special in that they have to be performed right at the outlets, such that tackling the power supply issue becomes very challenging. As most of the HVAC outlets are not close to the power grid, wiring the sensor nodes may cause lots of troubles to, for example, indoor wire planning (which is what we try to avoid by wireless sensing), and is hence not a preferred choice. Moreover, a normal battery-powered WSN is not competent either, as changing batteries from time to time for a large building with tens of thousands HVAC outlets is definitely not feasible. Therefore, one would need a self-powered WSN that performs sensing and networking operations (for gathering sensed data) in an autonomous manner. Although BACnet [12] might be used for the same purpose, the penetration of such systems cannot be guaranteed even in developed countries such as Singapore. Moreover, a power-free WSN can complement BACnet systems to further suppress the energy consumption induced by sensing and collecting data.

To tackle the aforementioned issue, we aim to develop a prototype of a *self-sustaining sensing system for airflow monitoring*. In particular, we revive the long-envisioned idea of piezoelectric energy harvesting and make a concrete case of applying it to power *indoor* WSNs. In addition, although the sensing module is designed for airflow monitoring, the energy harvesting and networking modules may serve other indoor sensing requirements (e.g., for temperature, light, and/or air quality). Finally, we intend to answer the four questions raised in Sec. 1 based on our field tests; these answers can yield intriguing insights and hence provide guidance for full-fledged developments and deployments of self-powered indoor sensing systems.

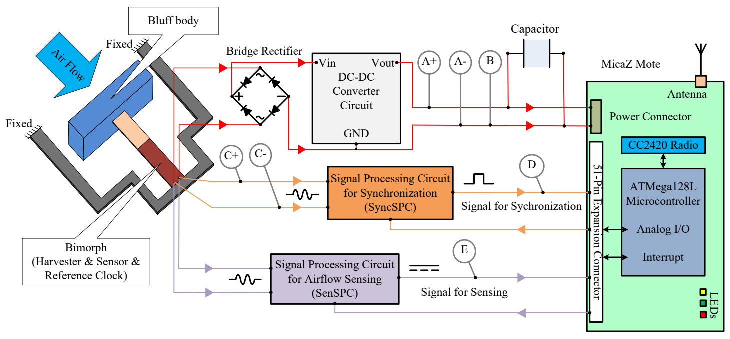

Bluff body
Fixed
Air Flow
Fixed
Bridge Rectifier
Vin
Vout
DC-DC Converter Circuit
GND
A+
A-
B
Capacitor
MicaZ Mote
Antenna
Power Connector
CC2420 Radio
ATMega128L Microcontroller
Analog I/O
Interrupt
51-Pin Expansion Connector
LEDs
C+
C-
Signal Processing Circuit for Synchronization (SyncSPC)
D
Signal for Sychronization
E
Signal Processing Circuit for Airflow Sensing (SenSPC)
Signal for Sensing
Bimorph (Harvester & Sensor & Reference Clock)

Fig. 2. Trinity Indoor Sensing System.

2.2 System Overview of Trinity

We sketch the schematic of Trinity in Fig. 2 by illustrating the key components. The harvester is a frame with a piezoelectric sheets (a piece of bimorph in our prototype) attached in the middle. We use a MicaZ Mote running TinyOS 2.1 as the sensor node platform, but Trinity can be readily adapted to all other commonly used hardware and software platforms. Three “paths” exist between the harvester and the node. The upper one is for power supply; it goes through our power management module. The middle one is for the purpose of synchronization, where a module *SyncSPC* integrating signal processing circuits is designed to transform the output sine wave into square wave in order to calibrate the native clock of the connected sensor node. The lower one is to realize airflow sensing; it involves another module of signal processing circuits (i.e., *SenSPC*) transforming the output AC voltage of the harvester into DC voltage, whose magnitude indicates the airflow speed.

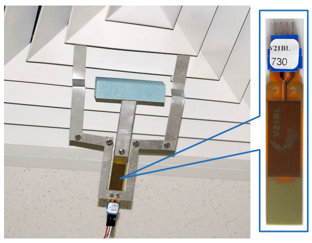


Fig. 3. Energy harvester and its key component bimorph.

2.2.1 Energy Harvesting and Power Management. As our sensor nodes are deployed right beside individual outlets, drawing power from the airflow issued by the outlets comes in handy. We choose piezoelectricity for energy harvesting because it is shown to be more efficient than other schemes (e.g., a micro turbine) given the low airflow speed [51]. In Fig. 3, we show our harvester prototype along with its key component, a bimorph. The harvester consists of a fixed frame that can be stuck to an HVAC outlet and a bimorph (as shown in Fig. 2) with bluff body attached to better absorb the power carried by the airflow. The power generated by the bimorph cannot be directly used by a sensor node due to the extremely high internal resistance (in the scale of mega-ohm, or MΩ) of the bimorph. In addition, we would like to have the operations of a sensor node (power consumer) as independent of the harvesting procedure (power supplier) as possible. Generally speaking, the power produced by a harvester (hundreds of micro-Watts, or μWs) is far lower than the power consumption of a sensor node in its active mode (tens of milli-Watts, or mWs). However, a naive operation mode that a bunch of packets can be sent or received only upon sufficient energy (hence voltage) is accumulated appears to be highly undesirable. Therefore, while low duty-cycle is necessary to keep a node "alive" under the tiny power supply, the actual duty-cycle ratio should be constrained only by the average power output from the harvester, instead of by the time period to accumulate a sufficiently high voltage.

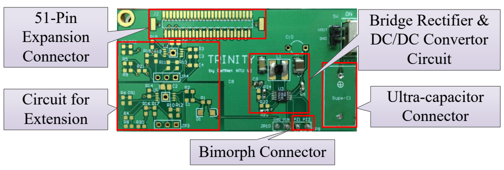

51-Pin Expansion Connector
Circuit for Extension
Bridge Rectifier & DC/DC Convertor Circuit
Ultra-capacitor Connector
Bimorph Connector

Fig. 4. Power management module.

The power management module that we have produced is shown in Fig. 4. It consists of several integrated circuits to regulate the power (both voltage and current), while having an ultra-capacitor to buffer the harvested energy. As we will discuss later, the design of our power management module can decouple power supplier from power consumer: it not only adapts the low efficient energy harvester to the energy-consuming sensor node, but also prevents the output voltage of the harvester (i.e., the bimorph) from being interfered by the load variations induced by the connected MicaZ Mote. Therefore, we can perform energy harvesting, synchronization and airflow sensing simultaneously, based on the output of the bimorph. In another word, the bimorph plays three roles in our system, i.e., energy generator, reference clock and airflow speed sensor, as shown in Fig. 2. Details of energy harvesting and power management module will be elaborated in Sec. 3.

2.2.2 Synchronous Duty-Cycling. Given the low power output of our harvester, our WSN must perform in a very low duty-cycle manner. Furthermore, the duty-cycling of the sensor nodes needs to be synchronized so that the receivers wake up in time to receive data packets sent by the respective senders, as otherwise asynchronous duty-cycling requires extra overhead to notify either receivers or senders and it may wake up nodes not involved in the transmissions [14, 24, 42]. Unfortunately, synchronization needs to be repeated in a regular basis in order to compensate for the clock drifts of individual sensor nodes. Existing synchronization protocols (e.g., FTSP [37]) rely on periodically flooding, resulting in an unaffordable load on our system. In Fig. 5, we illustrate the relative drift between two MicaZ Motes, assuming each has a sleeping-active period of 30s including an active slot of 45ms. Without a regular (re)synchronization, two initially synchronized nodes may communicate with each other only for about 50 minutes (each period takes 30s).

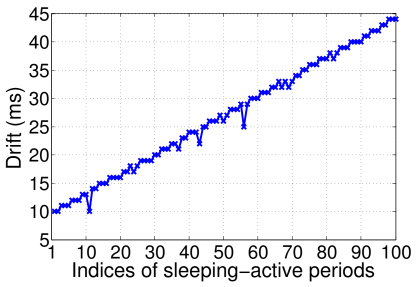


Fig. 5. Relative clock drift.

Our Trinity system first employs a low-power self-calibration strategy. One of the physical properties of our harvester is that its vibration frequency depends only on its structure and material. Moreover, the decoupling effect of our power management module also guarantees the harvester's output alternating power has the same frequency as its vibration. Therefore, the output of the harvester can be used as an external oscillator to calibrate the native clock of the connected sensor node through our SyncSPC module (as shown in Fig. 2). Although we borrow this idea from the FM synchronization reported in [32], our strategy consumes far less energy as the external oscillation comes directly from the harvester and hence needs no additional auxiliary receiver devices. Nevertheless, this self-calibration process still incurs an mW scale energy consumption, so we want to avoid invoking it as much as possible. Consequently, we also propose a complementary strategy for synchronization. This approach piggybacks control information with data traffic: for each transmission link, the sender synchronizes its sleeping time with that of the receiver based on the information piggyback with the acknowledgements sent by the receiver. According to our field tests, our per-link synchronization is sufficient in most cases to maintain an adequate level of successful transmissions. We will provide more details on these two strategies in Sec. 4

2.2.3 Airflow Speed Sensing. Being an important aspect for indoor microclimate monitoring, airflow sensing can be preformed by many off-the-shelf products. However, almost all the sensor products have a high demand on power. For example, the MEMS airflow sensor D6F-10A produced by OMRON Electronic Components LLC has a current consumption of 15mA and requires an input voltage of 10.8 to 26.4V, which entails a power consumption of around a few hundreds of mW. Such a huge power consumption is beyond the capacity of our energy harvester. To this end, Trinity takes a novel approach towards sensing the airflow speed. As it will be shown (in Sec. 3.1) that, as the output voltage of our harvester is a function of the airflow speed, we may sample the voltage of the harvester to infer the airflow speed. To facilitate the process of sampling, we design a module, SenSPC (as shown in Fig. 2), to transform the output AC voltage of the bimorph into DC voltage adapted to the ADC module of MicaZ Mote. More details will be given in Sec. 5.

3 HARVESTING AND MANAGING TINY ENERGY

In this section, we first discuss the physical principles of our energy harvester, then we explain the design of the power management module.

3.1 Harvesting Energy from Airflow

The basic idea behind our energy harvesting scheme is the *piezoelectric effect.* More specifically, certain materials produce electric charges (or *piezoelectricity*) on their surfaces as a consequence of applying mechanical stress. In our case, the mechanical stress is expected to be applied by the airflows. Therefore, our design (shown in Fig. 2 and 3)is to attach a bimorph (containing piezoelectric materials) to a frame fixed on an HVAC outlet such that the bimorph (a type of cantilever with a tip mass, the bluff body) can exhibit forced oscillation under the blow of airflow. As a result, the oscillation exerts mechanical stress on the bimorph that in turn accumulates electric charges.

Based on aforementioned principles, we may establish the following coupled lumped parameter model [51] to emulate the electromechanical coupling behavior of our harvester. The model is characterized by two governing equations:

$$M_{eff}\ddot{w} + C\dot{w} + Kw + \Theta V = F, \quad \frac{V}{R_L} + C_p\dot{V} - \Theta\dot{w} = 0 \tag{1}$$

where w is the tip displacement in the direction normal to the airflow, C and K are the damping coefficient and stiffness of the harvester, respectively, V is the generated voltage, R_L is the applied load resistance, and C_p is the total capacitance of the piezoelectric sheets. Other parameters need a bit more elaborations. $M_{eff} = 33M_b/140 + M_{blu}$ is the effective mass, where M_b is the distributed mass of the cantilever, and M_{blu} is the mass of the bluff body. $\Theta = \sqrt{(\omega_{noc}^2 - \omega_{nsc}^2)M_{eff}C_p}$ is the electromechanical coupling term, where ω_{noc} and ω_{nsc} denote the open circuit and short circuit *natural* frequencies of the harvester, respectively. F represents the aerodynamic force acting on the tip body in the direction normal to the airflow, it can be computed as

$$F = \frac{1}{2}\rho_a h L_{blu} U^2 C_F, \tag{2}$$

where ρ_a is the air density, $h \cdot L_{blu}$ denotes the windward area of the bluff body, U is the airflow speed, and $C_F = \sum_i A_r \alpha^r$ $(r = 1, 2, 3, \ldots)$ is a function of the angle of attack α, and can be determined through experiments (e.g., [13]). Here A_r denotes an empirical coefficients for the polynomial fitting. The attack α is defined by $\alpha = \dot{w}/U + w_a$ where w_a denotes the rotation angle of the cantilever at the free end.

The mechanical part of the equation system (1–2) describes an aeroelastic phenomenon termed *across-wind galloping* [18]. As the system is coupled with the piezoelectric effect, there exists an *implicit function relation* between the airflow speed U and the voltage V. Moreover, the vibrations incurred by galloping are converted into an alternating output voltage with a fixed frequency, while the vibration frequency due to galloping is always consistent with the natural frequency of the harvester according to [41]. We shall use these features later for our synchronization and sensing modules (see Sec. 4.2 and 5), but we focus only on power generation for now.

Table 1. Two different harvester designs.

Parameters	Harvester 1	Harvester 2
Open circuit frequency ω_{noc} (Hz)	13.23	10.99
Short circuit frequency ω_{nsc} (Hz)	13.14	10.87
Capacitance C_p (nF)	52	18
Bimorph Type[a]	V21BL	V22BL
Mass of bluff body M_{blu} (kg)	0.0017	0.0017
Windward area of bluff body(cm^2)	20	20
Airflow speed (m/s)	5.5	5.5
Equiv R_L^a for active mode (kΩ)	1	1
Power P^a under R_L^a (μW)	4.51	2.11
Equiv R_L^s for sleeping mode (kΩ)	425	425
Power P^s under R_L^s (μW)	622	258
Power P^* under optimal load (μW)	667	268

[a]The specification for the two types of bimorphs can be found in http://www.mide.com/pdfs/Volture_Datasheet_001.pdf

According to our estimates and measurements, different bimorphs may produce a peak-peak open-circuit voltage ranging from 20V to 50V and a short-circuit current around 100μA. This suggests a large internal resistance R_I in the scale of MΩ. Given a certain load resistance R_L, the power output is $P = V^2/R_L$, but this V is lower than the open-circuit voltage and is affected by the ratio between R_L and R_I. In Table 1, the parameters of two harvesters (among several others that we have tried) are shown; the main difference comes from the distinct bimorphs used. Apparently, to efficiently utilize the harvested power, a power management circuit should be added between the harvester and sensor node in order to match R_L to R_I, so that sufficient power can be generated regardless of the node's working modes.

3.2 Regulating and Buffering Energy

In general, a power management module that draws power from our harvester and supplies power to a MicaZ Mote has to achieve the following four objectives:

- Regulate the alternating voltage (also its amplitude) to suit the 3.3V direct voltage input of a MicaZ Mote.
- Deliver an equivalent load resistance R_L that matches the internal resistance R_I of the harvester, largely independent of the working modes of a sensor node.
- Buffer the generated power to sustain the connected sensor node.
- Prevent harvester's output (voltage) signal from being interfered by load variations induced by the connected sensor node, for the implementations of our synchronization and airflow sensing strategies.

Our power management module (as shown in Fig. 2 and 4) mainly consists of three components: a bridge rectifier, a DC/DC convert and an ultra-capacitor recharging system. The first two components are built in chip LTC3588-1 [5], on which the efficiency of our energy harvesting circuit mainly depends. Specifically, the bridge rectifier transforms the alternating power generated by the bimorph into DC power with a very low loss of around 400 mV (with the piezoelectric current of around tens of microamps), and the resulting DC power is then converted to the working voltage of a MicaZ Mote through the DC/DC converter with a high efficiency of up to 85%. By integrating such low-loss full wave bridge rectifier and highly efficient converter, LTC3588-1 has an optimized performance in handling piezoelectric energy with high impedance. The output of the chip LTC3588-1 is divided into two parts: one is for direct power supply to the MicaZ Mote, while the other one is used to charge an ultra-capacitor through a charging circuit. As we will show later, our energy harvesting circuit is of sufficient efficiency to harvest energy from even light airflows, for sustainably powering full-functional MicaZ Mote with the help of an elaborately designed duty-cycling strategy.

In our system, we adopt PHB-5R0V155-R (with a capacity of 1.5F) [8], not only to supply power to MicaZ Mote, but also to backup power in case that the harvester temporarily stops working. It is well known that, ultra-capacitors have been increasingly applied as energy storage in many recent embedded devices. Compare with rechargeable batteries (e.g., film batteries adopted in the preliminary version of our system [49]), ultra-capacitors have many advantages, for example, high recharging/discharging efficiency (higher than 90%) and power density (even 10 times higher than batteries), long life time (more than 1 million recharge cycles), simple recharging/discharging circuit, as well as robustness to temperature, shock and vibration.

Except for separating power consumption from generation, the decoupling effect of our power management module also means that the output of bimorph is free from the load variations induced by connected sensor node (e.g., duty-cycling). The main reason is, the ultra-capacitor between LTC3588-1 and MicaZ Mote acts as a buffer against the instantaneous variations of the MicaZ Mote's energy consumption. For example, when the MicaZ Mote switches from sleep mode to active mode, our ultra-capacitor of 1.5F is sufficient to make up for the increase of power consumption of the MicaZ Mote, resulting in a very tiny drop in voltage (at a scale of 10^{-4}V according to our measurements) for the ultra-capacitor. Hence, there is no need for the sensor node to draw more power from LTC3588-1 (thus the bimorph ahead of the LTC3588-1).

4 NETWORK OPERATIONS UNDER EXTREMELY LOW POWER SUPPLIES

According to Table 1, the power generated by our harvester is several hundreds of μWs, but a MicaZ Mote consumes several tens of mWs in various working modes (see Table 2). Due to this deficit in power supply, a Trinity WSN has to operate under a very low duty-cycle. Fortunately, as we shall show in our field tests, such a low duty-cycle does not compromise the functionality of Trinity as an indoor sensing system. In this section, we first introduce the duty-cycling configuration for Trinity in Sec. 4.1, and then we propose two complementary strategies in Sec. 4.2 and Sec. 4.3, respectively, that provide a crucial synchronization service for Trinity to properly operate a WSN with highly limited power supply.

4.1 Duty-Cycling Configuration

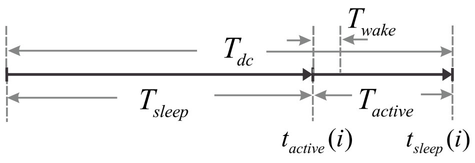


Fig. 6. The sleeping-active period of a sensor node.

A Trinity sensor node works in a periodic on-off manner. As shown in Fig. 6, it has a sleeping-active period T_{dc} consisting of a sleep duration T_{sleep} and an active duration T_{active}. The sensor node usually needs to spend T_{wake} (3 to 4ms) to recover from sleeping mode to active mode. We denote by $\mathcal{R}_{dc} = T_{active}/T_{dc}$ the duty-cycle.

The large deficit in energy supply requires Trinity to not only employ an extremely low duty-cycle but also reduce the power consumption during T_{sleep} as far as possible. According to Table 2,[1] the most energy-consuming components of our MicaZ Motes are the radio module (CC2420 [2])

Table 2. Current draws under different modes.

Sleeping mode	20~60μA
Active mode (transmitting, 0dBm)	23.4mA
Active mode (radio off but MCU on)	8mA
Active mode (transmitting, -10dBm)	21.8mA
Active mode (receiving)	22.4mA
Active mode (transmitting, -25dBm)	21.4mA

and MCU (ATmega128L [1]). While the radio and MCU together may draw up to 23.4mA, the MCU alone already needs 8mA. Therefore, only shutting down the radio during T_{sleep} is not enough, we have to shutdown everything except the oscillator, which leads to the sleeping mode shown in Table 2. The actual power consumption of a sleeping node varies from 66 to 198μW (given the 3.3V working voltage of a node); this allows for several hundreds of μW power surplus to be charged into the ultra-capacitor. Driven by these measurements, we can figure out an upper bound on the duty-cycle $\mathcal{R}_{dc}$. Let I_h, I_{active}, I_{sleep}, and I_c denote the output current of the harvester, the current draws within T_{active} and T_{sleep}, and the current consumed by the power management module (approximate 8 μA), respectively, and $\mathcal{R}_{dc}$ should satisfy

$$\frac{I_{active}V_{node}}{V_{harv}\delta} \cdot \mathcal{R}_{dc} + \frac{I_{sleep}V_{node}}{V_{harv}\delta} \cdot (1 - \mathcal{R}_{dc}) + I_c \leq I_h, \tag{3}$$

where V_{harv} and V_{node} are the the output voltage of the harvester and the working voltage of a node, respectively, and $\delta = 0.85$ is the efficiency of the DC-DC convertor. Any $\mathcal{R}_{dc}$ satisfying this constraint guarantees that the excessive energy cost during T_{active} can be compensated during T_{sleep}.

Detailed calculation of $\mathcal{R}_{dc}$ will be postponed to Sec. 7.2, we hereby briefly introduce our strategy of adaptively adjusting $\mathcal{R}_{dc}$ to satisfy (3). We fix the active duration T_{active} and empirically specify

[1]The data reported in Table 2 are obtained by our own experiments, which may differ significantly from the data sheet of CC2420 [2].

the shortest sleep-active period T_{dc} according to $\mathcal{R}_{dc}$ measured under the maximum airflow speed offered by an HVAC setting. Upon waking up, a sensor node first reads its voltage to check if it is maintained to the same level as before. If not, it doubles T_{dc} until a sustainable $\mathcal{R}_{dc}$ is reached.

Given an mA scale I_{active} and a μA scale I_h, it is obvious that the duty-cycle has to be below 1%. Now how a Trinity WSN operates under such a low duty-cycle is a key question we need to address. Obviously, asynchronous low power MACs such as [14, 24, 42] would not work (which will also be demonstrated later in Sec. 7.3.2), because the *effective* $\mathcal{R}_{dc}$ cannot be made very low given that a sender has to either send a long preamble or wait for a receiver to wake up. Consequently, a Trinity WSN has to operate in a synchronous manner, such that a pair of sender and receiver wake up at roughly (with accuracy down to millisecond scale) the same time. This cannot be done without a constant synchronization service running at the background, as two native clocks of an arbitrary pair of nodes may always exhibit relative drift [32] (also see Fig. 5). Nevertheless, the synchronization service has to be almost energy-free, since Trinity does not have much energy budget for it. We hereby propose two complementary synchronization strategies in the following.

4.2 Self-Calibration with Harvester Vibration

One important property of the harvester vibration (briefly discussed in Sec. 3.1) is that, its frequency (thus that of the output voltage) is consistent with the natural frequency of the harvester, which relies only on the structure and material of the harvester [44]. For our Trinity system, each node is connected to a harvester whose natural frequency is accurately measured. As the structure of a harvester (thus its natural frequency) is robust to environment changes, its output voltage can serve as an external reference oscillator (with a known frequency) to calibrate the native clocks of the connected node in order to eliminate its clock drift. As explained in Sec. 3.2 (also as shown in Fig. 2), we can directly make use of the output voltage signal of the bimorph. In the following, we first explain the rationale of our self-calibration strategy, before diving into the technical details.

4.2.1 Principle of Self-Calibration. Our strategy is based on a clock calibration method, with the goal of estimating the drift rate of a certain clock with respect to a presumably more accurate reference clock. Being aware of the instant clock drift rate, a node in a WSN would easily maintain a native clock that is synchronized with the reference clock. This idea was recently proposed for sensor node clock calibration using FM radio signal [32]. However, while the method proposed in [32] requires an extra signal receiving/processing module with considerable energy cost (it entails a power consumption comparable to that of the data transmission), our method directly gets the reference clock signal from the harvester to trigger interrupts of a node's MCU, resulting in a much lower power consumption.

Given a certain time period $\mathcal{T}$, we measure it using the reference clock by counting the number of periods of reference clock and dividing it by the known frequency f_{ref}. We denote this measurement by $\mathcal{T}_{ref}$. During the same period, a node is also able to measure it using its native clock (similarly by counting the number of ticks and dividing it by the nominal frequency f_{node} of the internal oscillator), which is denoted by $\mathcal{T}_{node}$. Comparing the two measurements on the same time period $\mathcal{T}$, we can compute the clock drift ξ [2] of the node's native clock with respect to the reference clock:

$$\vartheta(\mathcal{T}_{ref}) + \mathcal{T}_{ref} = \mathcal{T}_{node}(1 + \xi) \tag{4}$$

where $\vartheta(\mathcal{T}_{ref}) = \mathcal{T}_{ref} - \mathcal{T}$ is a zero-mean Gaussian variable representing the error of the reference clock when measuring $\mathcal{T}$. As the clock drift rate may vary due to the environmental factors, e.g., energy storage, temperature, and so on, the calibration procedure may need to be performed periodically, such that the estimation of the clock drift rate can be updated in response to the

[2] If the node's native clock runs slower than the reference clock, $\xi > 0$; otherwise, $-1 < \xi < 0$.

drift jitter. Therefore, we take the linear regression approach used in [32]: each node periodically measures both $\mathcal{T}_{ref}$ and $\mathcal{T}_{node}$, and maintains a table of k tuples sampled in the last k periods $\mathfrak{T} = \{\langle \mathcal{T}_{ref}^{k}, \mathcal{T}_{node}^{k} \rangle, \langle \mathcal{T}_{ref}^{k-1}, \mathcal{T}_{node}^{k-1} \rangle, \cdots\}$. Then a linear regression is applied to fit these points and hence to estimate ξ (as $\vartheta(\mathcal{T}_{ref})$ is zero-mean, the intercept of the line should goes to zero with a large k). In the following, we will discuss how to apply this principle to the context of Trinity.

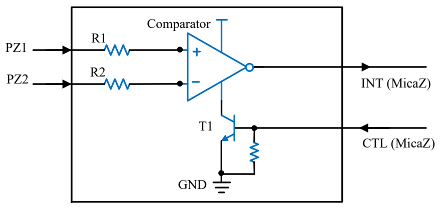


Fig. 7. Signal processing circuit for synchronization (SyncSPC).

4.2.2 *Trinity Self-Calibration.*

We design a module, SyncSPC, integrating signal processing circuits for our self-calibration purpose, as shown in Fig. 7. The alternating power of the harvester (in a form of sine wave) is imported into SyncSPC through PZ1 and PZ2. The resistors R1 and R2 are deployed to adapt the voltage to the MCU module (specifically the interrupt processing module) of the connected MicaZ Mote. The sine wave is then converted into square wave by a comparator, where the zero-passing points can be captured to serve as the ticks for the reference clock. Specifically, using every two falling edges to trigger an interrupt to the sensor node's MCU (via INT), we get a reference clock for self-calibration, whose frequency is indeed the natural frequency of the harvester (measured when it is manufactured). Given that the power consumption of the comparator is at a scale of mW (around 1.3mW according to our measurement), we deploys a NPN transistor T1 as a switch which is controlled by the MicaZ Mote through CTL, to shut down the comparator when the node is in sleeping state or when no calibration task is demanded. Besides, since the MCU is considered to have a "flat rate" current draw of 8mA, we can see that our self-calibration is extremely energy efficient. The input sine wave and output square wave of our SyncSPC (i.e., the voltage signals at points C and D in Fig. 2) are shown in Fig. 8 (a) and (b), respectively. It is obvious that, although the connected MicaZ Mote works in a duty-cycle manner and thus results in load variations, the sine wave has only very slight distortion, thanks to decoupling effect of our power management module we have explained in Sec. 3.2. Through the processing of our SyncSPC module, the resulting square wave becomes regular and its magnitude (≤3.3V) is adapted to the MCU of MicaZ Mote.

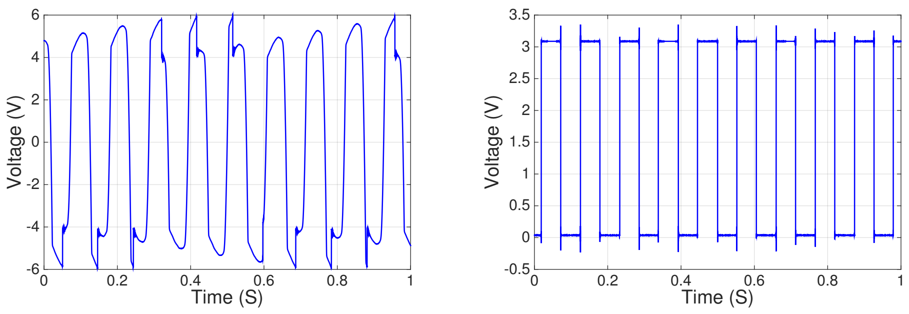
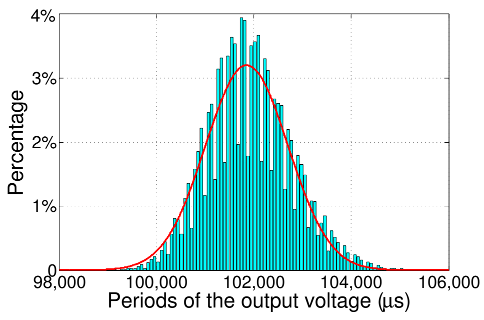


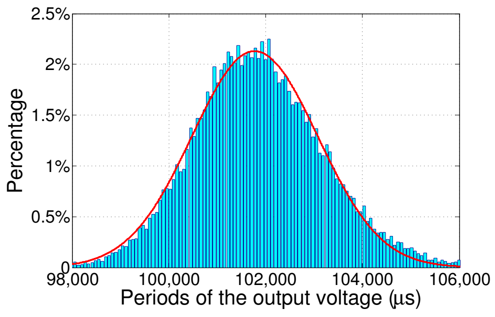
Fig. 8. The signals of the reference clock.

To further show the reference clock is sufficiently stable to measure the drift rate, we first use an accurate clock to measure the interrupt period in Fig. 9(a); the Gaussian fitting gives a mean of 101800μs. Then we use the interrupt period to calibrate the native clock of a node in Fig. 9(b), resulting a mean of 101900μs. Obviously, the 100μs difference results in the drift of this native clock.

As both histograms can be perfectly fitted by a Gaussian distribution, the noise around the mean will be eliminated by the linear regression.



(a) Accurate clock



(b) Native clock

Fig. 9. Frequency stability of the reference clock under an accurate clock (a) and a native clock (b).

Our self-calibration algorithm is summarized in **Algorithm 1**. The basic idea is to perform the calibration during T_{active} in a periodic manner. As the drift is minor within each duty-cycle period, so the period of calibration can be longer than that of duty-cycling. A slight difficulty here is that, as the MCU and the comparator are both shut down during T_{sleep}, the clock cycle of the reference clock is not counted between two calibrations. Therefore, the measurement of the reference clock at the j-th calibrations, $\mathcal{T}_{ref}^{j}$ cannot be directly measured. Within the active duration for the j-th

ALGORITHM 1: Self-Calibration (the j-th calibration period)

Input: The reference clock frequency f_{ref}; the estimated drift rate of the last calibration ξ^{j-1}
Output: The estimation for the clock drift rate ξ^{j}

```
1 upon Interrupt
2     Record $\mathcal{T}_{node}^{j}$
3     $\mathcal{T}_{ref}^{j} \leftarrow \lfloor (1+\xi^{j-1})\mathcal{T}_{node}^{j} \cdot f_{node} + 0.5 \rfloor \cdot f_{ref}^{-1}$
4     insert $\langle \mathcal{T}_{ref}^{j}, \mathcal{T}_{node}^{j} \rangle$ into $\mathfrak{T}$
5     if $j > k$ then delete $\langle \mathcal{T}_{ref}^{j-k}, \mathcal{T}_{node}^{j-k} \rangle$ from $\mathfrak{T}$ ;
6
7     Fit a line $\left(1+\xi^{j}\right) x + b$ to the points in $\mathfrak{T}$ using linear regression and return $\xi^{j}$
8 end
```

calibration, the node simply waits for the next interrupt (driven by the rising edges shown in Fig. 8 (b). When an interrupt is fired, the native clock time $\mathcal{T}_{node}^{j}$ is recorded (line 2). As no interrupt has been counted since the last calibration, we have to figure out the index of this interrupt in order to compute $\mathcal{T}_{ref}^{j}$. Using $\mathcal{T}_{node}^{j}$, f_{ref} and the last estimated drift rate ξ^{j-1}, this index can be computed by rounding $(1+\xi^{j-1})\mathcal{T}_{node}^{j} \cdot f_{node}$ to the nearest integer, which in turn suggests $\mathcal{T}_{ref}^{j}$ (line 3). The remaining part simply maintains the table $\mathfrak{T}$ and performs a linear regression to estimate the drift rate ξ^{j}. The correctness of this computation relies on an assumption that the drift jitter of the native clock of the node can only cause a misalignment less than half of the reference clock period, which can be easily guaranteed by choosing a proper calibration period. As our reference clock has a frequency of around 10Hz, so given a drift jitter of 1ms/minute, this condition can still be guaranteed even if the calibration period is about an hour.

Assuming all nodes are initially synchronized, the calibrations at individual nodes allow every node to slightly adjust its T_{sleep} and T_{active}, such that the duty-cycling remains synchronized. This synchronization does not have to be perfect, a several-millisecond misalignment between a pair of sender and receiver (as far as it is stable) is tolerable. We shall discuss in the following a complementary strategy that can be used to perform the initial synchronization and also to further reduce the power consumption of synchronization.

4.3 Per-Link Synchronization through ACKs

The self-calibration presented in Sec. 4.2 only compensates for the clock drift for individual sensor nodes without actually synchronizing them. Moreover, its mW scale power consumption may still be a burden for Trinity. So we hereby propose a per-link synchronization strategy. It virtually consumes no power as the information is piggyback with ACKs, and it complements the self-calibration, allowing nodes to be synchronized with each other. We first introduce the basic protocol, then we discuss its limitations and extensions.

4.3.1 The Synchronization Protocol. The objective of our per-link synchronization is to let the sender and receiver wake up and go dormant at the same time. In practice, we actually require the receiver to wake up slightly before the sender in order to accommodate certain system errors. In the following, we use lower-case t to represent a point in time and upper-case T for a time period. Also let t (or T) denote the wall-clock time, $\hat{t}$ (or $\hat{T}$) denote the reading from the native clock of a node, and a superscript s (resp. r) denote the sender (resp. the receiver).

Given $t^s_{sleep}(i)$ and $t^r_{sleep}(i)$ as the times for the sender and receiver to go dormant in the i-th sleeping-active period, the times for them to wake up in the $i+1$-th period are

$$t^s_{active}(i+1) = t^s_{sleep}(i) + \hat{T}^s_{sleep}(\xi^s + 1), \qquad t^r_{active}(i+1) = t^r_{sleep}(i) + \hat{T}^r_{sleep}(\xi^r + 1) \tag{5}$$

where ξ^s and ξ^r are the clock drift rates of the sender and receiver, respectively. Assuming $t^s_{sleep}(i) = t^r_{sleep}(i)$ and the drift rates are known, we can (re)set the duty-cycling parameters (i.e., $\hat{T}^s_{sleep}$ and $\hat{T}^r_{sleep}$) for the sender and receiver to guarantee that $t^r_{active}(i+1) < t^s_{active}(i+1)$. As the drift rates are already estimated by our self-calibration strategy, the per-link synchronization problem is now reduced to synchronizing t^s_{sleep} and t^r_{sleep} for each sleeping-active period.

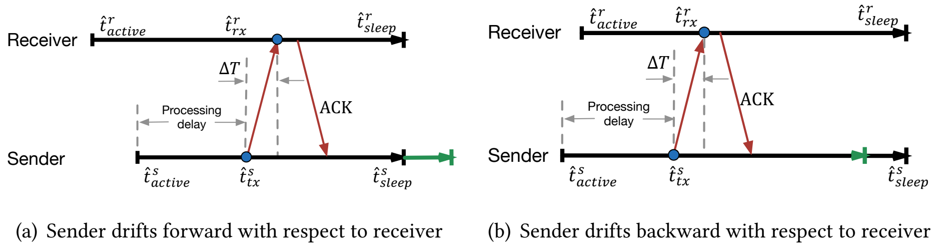
For the actual protocol, the time can only be measured by the native clock for a node. Let us consider one specific active period. The sender records a timestamp $\hat{t}^s_{tx}$ when sending a data packet. When the receiver gets the incoming packet, it also records a timestamp $\hat{t}^r_{rx}$, and it computes its time-to-dormant as $\Lambda = \hat{t}^r_{sleep} - \hat{t}^r_{rx}$. If being aware of Λ and ξ^r, the sender sets its sleep time locally

$$\hat{t}^s_{sleep} = \hat{t}^s_{tx} + \left[\Delta T + \Lambda(1+\xi^r)^{-1}\right](1+\xi^s) \tag{6}$$

where ΔT is the time cost in packet delivery (including MCU interrupt handling, decoding/encoding and propagation delay) and can be estimated offline. In practice, ΔT is only hundreds of microseconds, so omitting ΔT does not cause any problem in our synchronization protocol. As the clock drift during Λ (caused by ξ^s and ξ^r) is also negligible, we may simplify (6) to

$$\hat{t}^s_{sleep} = \hat{t}^s_{tx} + \Lambda \tag{7}$$

Therefore, being aware of Λ is sufficient for the sender to get synchronized to the receiver. In our implementation, the receiver can notify the sender by piggybacking Λ in its ACKs. We give two examples in Fig. 10 to visualize our per-link synchronization strategy. Basically, it shows that, regardless of the drifting direction of the sender with respect to the receiver, the two nodes can be re-synchronized as far as the two-way communications can be preserved.



(a) Sender drifts forward with respect to receiver (b) Sender drifts backward with respect to receiver

Fig. 10. Per-link synchronization between two nodes. The green arrows denote the original dormant times for the sender before synchronizing with the receiver.

While combining (5) and (6) allows for an accurate per-link synchronization by making use of the outcome of our self-calibration, the per-link synchronization can also be a stand-alone protocol if we apply (7) and the following:

$$t^s_{active}(i+1) = t^s_{sleep}(i) + \hat{T}^s_{sleep}, \quad t^r_{active}(i+1) = t^r_{sleep}(i) + \hat{T}^r_{sleep} \tag{8}$$

Under this simplified protocol, special care has to be taken to guarantee the receiver receives the data packet sent by the sender in the $(i+1)$-th duty-cycle period. Typically, we would require $0 < t^s_{active}(i+1) - t^r_{active}(i+1) < T^r_{active}$. According to our experience, setting $T^r_{sleep} - T^s_{sleep} = 5$ms is sufficient to guarantee the difference in active time is larger than 0, and the clock drift during the sleep period cannot result in a gap larger than T^r_{active}.

4.3.2 Discussions. In order for the per-link synchronization to be stable in a WSN, the network topology has to satisfy certain conditions. As we will show in Sec. 6, our Trinity WSN is organized as a tree rooted at the sink, for the application of data collection. Within the tree, non-root nodes report data to their respective parent nodes, and thus get synchronized with these parent nodes. Obviously, as our per-link synchronization forces a sender to keep up with its receiver, running our per-link synchronization protocol within a tree network simply lets every node to stay roughly synchronized with the sink node, such that the network get synchronized globally.

Since our per-link synchronization relies on data transmissions, packet or ACK loss may reduce the opportunity for nodes to get synchronized. Fortunately, Trinity WSN has a rather high packet delivery ratio even with a fixed tree topology (more than 90% per link as will be shown in Sec. 7.4), so losing less than 10% of the synchronization opportunities should not cause much trouble. Moreover, with the drift rate estimated by the self-calibration, nodes may stay synchronized for quite a long time even without data traffic.

5 ENERGY-FREE SENSING

As mentioned in Sec. 2.1, airflow speed sensing is a crucial part of an indoor microclimate control system. However, the off-the-shelf airflow sensors often consume more power than a sensor node. Obviously, Trinity has to look for an alternative solution. According to Sec. 3.1, there exits an implicit function relation between airflow speed U and the harvester output voltage V. Therefore, if we could work out this function, our harvester may also serve as an airflow sensor. Moreover, similar with our self-calibration strategy, our airflow sensing can also be directly based on the output of the bimorph, due to the decoupling effect of our power management module.

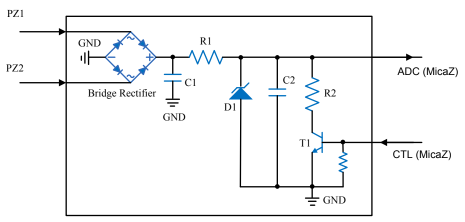


Fig. 11. Signal processing circuit for airflow sensing (SenSPC).

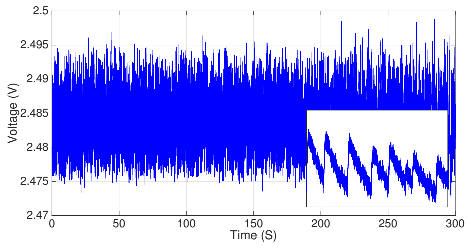


Fig. 12. The output of SenSPC for airflow sensing and its zoom-in view.

We design a specialized module, i.e., SenSPC (as shown in Fig. 11), to handle the output signal of the bimorph. Continuously sampling the AC voltage outputted by the bimorph to acquire its peak value is one of the choices; but it entails heavy energy and time cost, especially considering the period of the output AC power is around 100ms. Therefore, we first adopt a bridge rectifier to transform the AC voltage into DC voltage that is proportional to airflow speed, as measuring DC voltage requires much fewer number of samples. The resistors R1 and R2 are used to linearly reduce the DC voltage to adapt to the ADC module of the MicaZ Mote, while the capacitors C1 and C2 are employed for filtering out signal noise. Additionally, we use a diode, D1, to protect the I/O interference of MicaZ Mote in case that the input voltage is higher than 3.3V. Again, in order to enhance energy efficiency, we deploy a transistor T1 to control the on-off of SenSPC. The output of our SenSPC is then introduced into the ADC module of the MicaZ Mote's MCU via the 51-pin expansion connector.

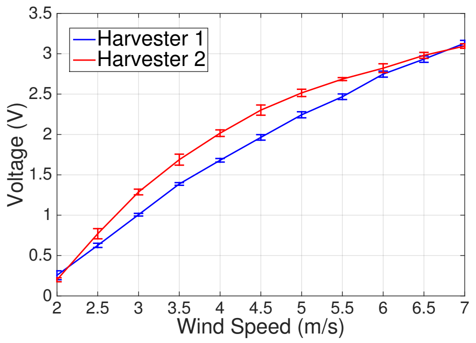


Fig. 13. Output voltage of SenSPC under different wind speeds.

Taking the same input as shown in Fig. 8(a), we hereby illustrate in Fig. 12 the output of SenSPC as well as its zoom-in view (at point E in Fig. 2). It can be seen that, our SenSPC transforms the output of the bimorph into direct power, and restrict the voltage below 3.3V. Furthermore, we notice that the SenSPC's output voltage signal is very stable (under certain airflow speed) with a variation range of only 0.02V. This suggests again that, the activities of the MicaZ mote does not induce interference to our sensing strategy.

To exhibit the relationship between airflow speed and SenSPC's output voltage, we perform extensive experiments in a wind tunnel (where wind speed can be controlled) with our two harvesters on which different types of bimorph are attached (with details given in Table 1). We record the voltage samples taken by the ADC module of a MicaZ Mote and the corresponding wind speeds; the results are shown in Fig. 13. We illustrate both mean values and standard deviations in the figure: both harvesters output distinguishable voltage values with respect to each wind speed interval. We choose the first harvester design for Trinity, as it has smaller variations under each airflow speed, and thus higher accuracy. Specifically, the voltage value under a given wind speed has a variation only up to ±0.05V, which suggests that using voltage to infer wind speed has an adequate accuracy with an error of roughly ±0.1m/s. Based on our measurement, the power consumption of our SencSPC is only at a scale of μW. Moreover, we enable it only when there is sensing task going on and tens of samples are sufficient for measuring DC voltage according to experience. Therefore, we may conclude that our sensing strategy is virtually energy free.

6 TREE-BASED DATA COLLECTION

In our Trinity system, the sensed data (i.e., the airflow speed at each outlet) are collected based on a sink-rooted tree topology. We hereby take a fixed tree topology, instead of allowing for a flexible route selection (e.g., [17, 26, 28]), since the latter strategy would entail unaffordable energy consumption for Trinity to perform competent link quality assessment. Fortunately, as a Trinity WSN is deployed on the ceiling, the link quality is rather stable due to the absence of common disturbance such as human movements. As will shown by our field tests in Sec. 7.4, our Trinity WSN achieves an adequate packet delivery performance, even with the network topology fixed.

We initially synchronize the Trinity WSN by adapting per-link synchronization protocol. Exploiting the energy pre-stored in the ultra-capacitors, the sensor nodes remain active (i.e., with no duty-cycling) long enough before being synchronized. We push a SYNC message downstream from the sink node along the tree. Compared with per-link synchronization, the protocol remains the same except that this SYNC replaces the ACKs: each parent node notifies its child nodes of its time-to-dormant, and the synchronized dormant time sets a common entry point to duty-cycling. Such a procedure could be completed in seconds for a WSN consisting of several tens of nodes; hence, it can definitely be supported Trinity's ultra-capacitor.

In the synchronized data collection tree with fixed topology, the sensed data are delivered to the sink node through multi-hop transmissions. Considering the active period is very short (around 60ms in our case) due to the extremely low duty-cycle of Trinity, a node either sends or receives data in one active period. Therefore, each sensor node broadcasts its transmitting/receiving schedule to its child nodes by the SYNC message [3] in the initialization phase, such that the child nodes can set up their own ones and then broadcast downstream. Specifically, a sensor node transmits data packets only when its parent node is ready for receiving in some active period; otherwise, it just turns on the radio and listens for the data packets from its children. In our implementation, we let the sink node "virtually" alternate between receiving and transmitting such that we finally get a layer-based transmitting/receiving schedule on the given tree-structured sensor network: the sensor nodes at the same layer have identical transmitting/receiving operations in an active period, while the parent node and its child nodes (belonging to adjacent layers) have alternant ones. The result of applying this scheduling method to our prototype system will be reported in Sec. 7.1.

When a sensor node switches into transmitting period, it first reads a airflow speed value from the "airflow speed sensor" (i.e., the energy harvester), and then transmits a data packet encapsulating both the latest sensed data and the ones received in the last receiving period to its parent node. Upon receiving the data packet, the parent node feeds back ACK messages to its child nodes. The sensor node drops the packet only when it receives the corresponding ACK message from its parent node or the number of transmission attempts exceeds a pre-defined threshold. Since there may be concurrent transmissions in our layer-based transmitting/receiving schedule, we rely on the default MAC of MicaZ Mote (equipped with TinyOS) to coordinate the resulting interference.

Another concern of our prototype implementation is how the Trinity WSN is re-synchronized in face of power failure. Again, we extend our per-link synchronization strategy as a solution. Specifically, when a sensor node is recovered from power failure, it stays in active status to snoop the

[3]To this end, a SYNC message carries both the synchronization information and the transmitting/receiving schedule.

ACK messages from its parent node or sibling nodes [4], since they have roughly the same dormant time. Moreover, the parent and sibling nodes have alternant transmitting/receiving schedules, which results in a big chance for the sensor node to snoop ACKs invoked by data delivery and thus to get synchronized as soon as possible. Once getting synchronized, the sensor node pushes a SYNC message downstream to re-initialize the sub-tree where it serves as a root.

So far, Trinity has been deployed only in Singapore (a tropical country) where the HVAC systems are always on. The airflow speeds may be decreased for energy-saving purpose, but a minimum airflow speed required by our harvesters can be guaranteed. Hence, widespread power failures rarely occur, especially considering the longevity and robustness of ultra-capacitors. For special cases where most of outlets even the whole HVAC system may be turned off such that the above snooping-based re-synchronization does not work, the initialization operation presented above may be leveraged to re-start the whole network.

7 SYSTEM EVALUATIONS

We have built a prototype of Trinity and deployed a small scale WSN in our research center for more than one month . In the following, we report various evaluations on this prototype. We start with a brief description of the deployment and evaluation settings, then we present the results concerning individual components, namely energy harvesting, networking, and sensing.

7.1 System Settings

Our Trinity prototype is built using the schematic shown in Fig. 2. Its hardware contains i) an energy harvester with a bimorph (the first type shown in Table 1), to harvest energy and to produce voltage signal for both self-calibration and airflow speed sensing, ii) a power management module (see Fig. 4), as well as SyncSPC and SenSPC (see Fig. 7 and Fig. 11, respectively) for processing the voltage signal from the bimorph, and iii) a MicaZ Mote to serve as the processing and networking unit. We implement the software part (including the two synchronization protocols and the airflow speed sensing strategies) using NesC under TinyOS-2.1. To accurately measure the power supplying process under different networking scenarios, we make use of an NI9229 (a 4-Channel, 24-Bit analog input module) [7] and NI LabVIEW [4] to monitor the current and voltage in Trinity. Finally, we use a hand-held anemometer to verify the airflow sensing data collected by our Trinity nodes.

We deploy a Trinity WSN consisting of 17 nodes on the ceiling of our research center (see Fig. 14). Each node is fixed on an HVAC outlet. The WSN constantly monitors the airflow speeds of individual outlets, and reports them to the sink (node n_0), possibly through multi-hop transmissions. Given the need for a synchronous duty-cycling, we deliberately make the network topology a tree rooted at the sink (see explanations in Sec. 4.3.2), and the WSN uses this topology to perform data collection. The result of applying the transmitting/receiving schedule to our three-layer data collection tree is also illustrated in Fig. 14. In particular, each period of the layer-based transmitting/receiving schedule consists of two time slots (i.e., sleeping-active periods), and $l_{i,j}^{(t)}$ represents the link along which n_i transmits data packets to n_j in the t-th time slot where $t \in \{1, 2\}$.

7.2 Energy Harvesting and Power Supplying

To study the interactions between energy supplier (involving the energy harvester and the power management module) and energy consumer (i.e., the connected sensor node), we use NI9229 and LabVIEW to monitor the current at the point A and voltage at point B (shown in Fig. 2). We first let the connected MicaZ Mote work with a duty-cycle ratio of 0.2% (T_{sleep} = 30s and T_{active} = 60ms).

[4]Without of loss of generality, we assume that a sensor node has sufficient energy storage in its ultra-capacitor when recovering from power failure.

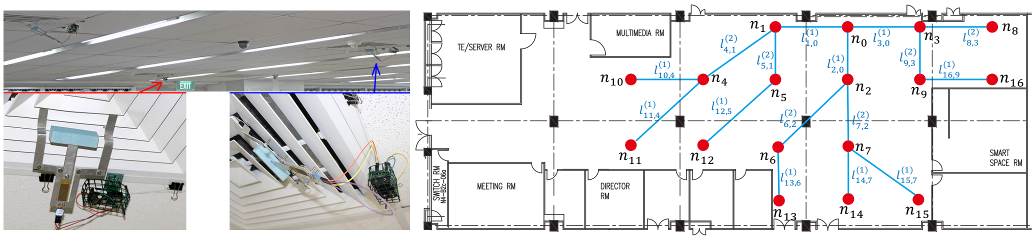
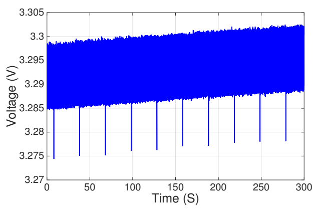
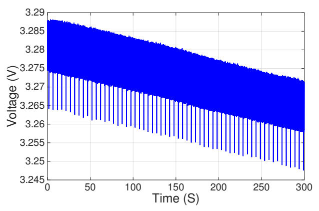


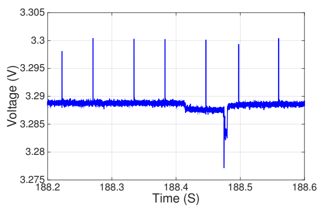
Fig. 14. The Trinity WSN deployed in our 800m^2 research center. Node n_0 serves as the sink, while other nodes sense the airflow speeds of individual outlets and report the data to the sink. The layer-based transmitting/receiving schedule is also indicated by the side of each link.



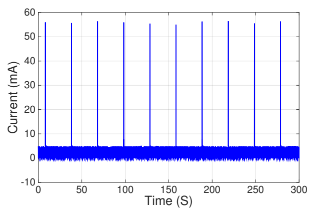
(a) Voltage variations under a duty-cycle ratio of 0.2%



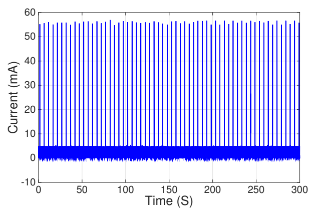
(b) Voltage variations under a duty-cycle ratio of 1.2%



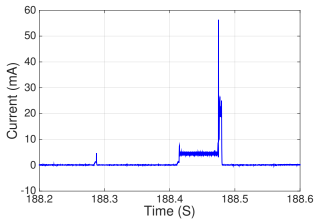
(c) Zoom-in view of voltage variations



(d) Current variations under a duty-cycle ratio of 0.2%



(e) Current variations under a duty-cycle ratio of 1.2%



(f) Zoom-in view of current variations

Fig. 15. Voltage variations at testing point A and current variations at testing point B during several sleeping-active periods under different duty-cycle ratios, as well as their zoom-in views.

During each active interval, the MicaZ Mote first senses airflow speed and then transmits a data packet containing the sensed values. The observed current and voltage variations are illustrated in Fig. 15 (a) and (b), respectively. This duty-cycle ratio respects the constraint (3) given $I_h = 70\mu$A for the harvester output. Therefore, the supply of the energy generated by the harvester is more than the need, and the surplus energy is then charged into the ultra-capacitor, as suggested by gradual increase of the voltage in Fig. 15 (a). Since the MicaZ Mote is sustained with adequate power supply, the current exhibits a periodic pattern in Fig. 15 (b) according to the duty-cycle setting. We then try another duty-cycle ratio of 1.2% ($T_{sleep} = 5$s and $T_{active} = 60$ms), and plot the measurements in Fig. 15 (c) and (d). Obviously, this duty-cycle ratio is unaffordable, as demonstrated by the gradual decrease of the voltage values. Fortunately, since the energy storage in the ultra-capacitor can temporarily allow for such a high duty-cycle ratio [5], the regularity of the current variations is still exhibited in Fig. 15 (d). By carefully observing the zoom-in view of the voltage variations in Fig. 15

[5]In our experiment, the MicaZ Mote is sustained for roughly four hours; it is shuttled down when the voltage is decreased to about 2.5V.

(e), the voltage suffers only a very small drop at the beginning of the active duration, i.e., during the stage of self-calibration or airflow sensing (see our analysis in Sec. 3.2). A drastic drop appears in the end of the active duration when the MicaZ Mote transmits a packet through ZigBee radio. After the node goes back into sleep mode, the voltage recovers immediately, due to the extremely high in-charging/dis-charging speed of the ultra-capacitor. Note that, the periodic upward peaks are caused by the buck circuit of the LT3588 chip instead of the MicaZ Mote. We also demonstrate the details of the current changing in Fig. 15 (f). It is shown that, the current variations correspond to voltage changing: when the MicaZ mote is activated with MCU and/or radio enabled, the current draw is increased to a scale of mA, which leads to the voltage decrease shown in Fig. 15 (e).

In fact, given different airflow speeds, we can estimate the equivalent output currents of the harvester. Using the constraint (3), we may figure out the upper bounds on the corresponding sustainable duty-cycles. Fig. 16 shows such a correspondence within the range of airflow speeds that are meaningful for indoor environments (our outlets offer airflow speeds varying from 3 to 6m/s), assuming V_{harv} = 9.2V, V_{node} = 3.3V, I_{active} = 25mA, I_{sleep} = 60μA, and I_c = 8mA. We also verify the feasibilities of these duty-cycle ratios through experiments. But, due to space limit, we do not plot the current and voltage variations hereby, especially considering they all show similar trend to Fig. 15 (a) and (b).

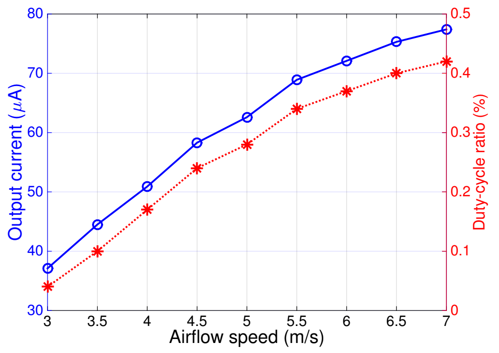


Fig. 16. Duty-cycle bounds for different airflow speeds.

7.3 Synchronous Duty-Cycling

We evaluate the effectiveness of our synchronization strategies by checking an arbitrary link within our Trinity testbed. To demonstrate the necessity of a synchronous duty-cycling, we employ A-MAC [24] on that link and compare it with our duty-cycling in terms of sustainability.

7.3.1 The Effectiveness of Trinity Synchronization. We apply our two synchronization strategies separately on an arbitrarily link in our Trinity WSN, and we run the tests for more than 8 hours with CSMA enabled. As the objective is to avoid the receiver missing the packet from the sender rather than perfectly synchronize the sensor nodes in μs scale, we report the results as, instead of synchronization accuracy, the *time-to-receive* (TTR) interval between the active time and the receiving time at the side of receiver, i.e., $\hat{t}^r_{rx} - \hat{t}^r_{active}$ in Fig. 17. Besides the original data shown in the left side, we also plot the interquartile range and median value to facilitate understanding the experiment results. According to our measurements, the average time in sending one packet is about 10ms for MicaZ Motes. Considering the receiver wakes up 5ms earlier than the transmitter, the receiver is supposed to get the data packet in 15ms after waking up, which accords with the median values shown in the box plots. The fluctuation in TTR intervals stems from the variations in channel accessing and sending. In summary, both our synchronization strategies could effectively correct the clock drift between a pair of sender and receiver.

7.3.2 Synchronous vs. Asynchronous. In order to show the necessity of using synchronous duty-cycling for Trinity, we apply A-MAC [24] to one of the links in our Trinity WSN using the

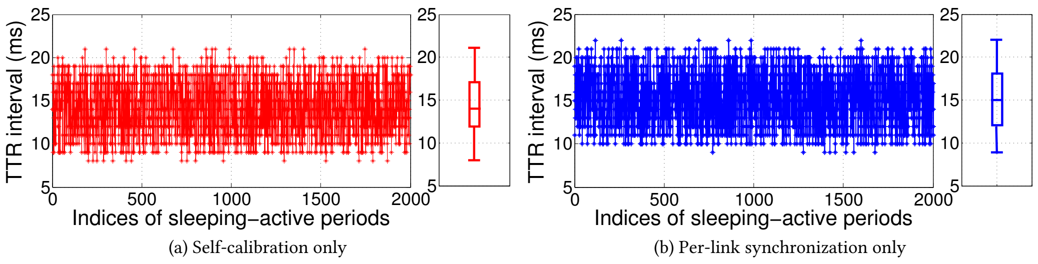


Fig. 17. Performance of Trinity synchronization.

same duty-cycle (i.e., 0.2% with T_{active} = 60ms and T_{sleep} = 30s) as Trinity. [6] In the network, every node generates a packet during T_{active}. The basic idea of A-MAC, a receiver initiated link layer protocol, is that a sender waits for the receiver to wake up with both MCU and radio enabled, so the effective duty-cycle for the sender depends on the waiting time. In fact, although the sender does not need to wait for a long time at the beginning since the whole WSN is initially synchronized, the effective duty-cycle will become larger due to the relative clock drift between the sender and receiver. To favor A-MAC, we deploy the sensor nodes in a "clean" environment without external interference such that the transmissions of the receiver's notification messages are ensured.

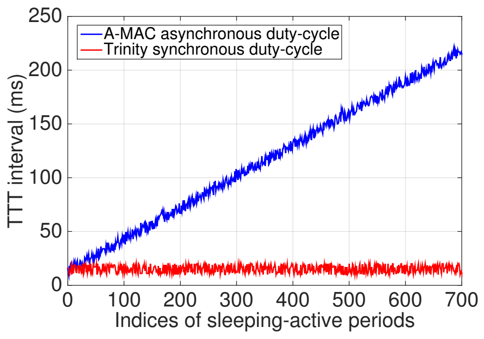


Fig. 18. Our synchronous duty-cycling vs. A-MAC.

We hereby use *time-to-transmit* (TTT) as the metric to compare our Trinity synchronous duty-cycling with A-MAC asynchronous duty-cycling, i.e., the time interval between the active time and the transmitting time a the side of sender. To facilitate our observation, we give the TTT measurements taken from the first hundreds of sleep-active periods. According to the results shown in Fig. 18, the TTT interval of the sender running A-MAC protocol is gradually increased due to the drift between its native clock and the receiver's native clock, which implies that the sender has to spend more time on maintaining active state at a price of considerable energy consumption, to wait for the receiver. In contrast, the TTT of the sender is very small if using Trinity's synchronous duty-cycle strategy, as the sender can immediately transmit its data packets after sensing the airflow.

Based on our experiments, due to the growth of the time drift along the tendency that has been illustrated in Fig. 18, the waiting time time of the sender will be gradually increased up to $T_{active} = 30s$, when their sleeping-active periods are completely "staggered". The resulting duty-cycle is obviously unaffordable for our Trinity system that has only limited energy supply. Our experiments exhibit that the link running A-MAC can remain "alive" for less than 10 hours due to the increasing effective duty-cycle caused by the clock drift. However, if we deploy the WSN in a "noisy" environment (e.g., in a WiFi-interfering scenario), the notifications from the receiver may get lost, the sender is forced to wait for several seconds until the next notification arrives.

[6]The sender-initiated low-power listening (LPL) [42] does not support very low duty-cycle due to the limit on the preamble length.

Since such a situation drastically increases the effective duty-cycle, it kills a node much sooner than expected (sometimes in an hour or so).

7.4 Network Performance

Our Trinity WSN shown in Fig. 14 has been running for months, hence demonstrating the validity of its self-powered design. However, as nodes in the WSN are all running under very low duty-cycles (determined by the airflow speeds of the individual outlets to which they are attached), there might be a suspicion on the *packet delivery ratio* (PDR) for individual nodes. Therefore, we compare our Trinity WSN with the same network running CTP [26] but powered by alkaline batteries. Note that the routing topology of CTP is not fixed, but rather determined by the link qualities. For fairness purpose, the sending rates of individual nodes are the same for both scenarios (i.e., one packet per 15.04 seconds), but the CTP WSN does not involve duty-cycling (to serve as a convincing reference). As a result, whereas we collect data through the whole lifetime (months for now) of the Trinity WSN, the data from CTP is obtained for a couple of days since batteries run out.

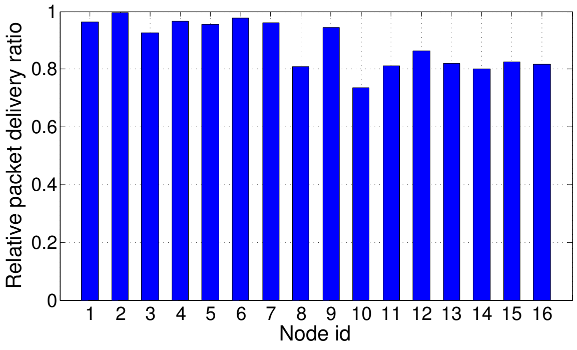


Fig. 19. Relative PDRs between Trinity and CTP.

Due to the low packet generation rate, CTP can achieve an almost 100% PDR for every node: a PDR of 96.8% can be reached even in the worst case (e.g., for node 16). This makes CTP a good reference for us to evaluate the performance of Trinity. We calculate the ratio of PDR (relative PDR) between our Trinity WSN and the CTP WSN for every node. As shown by Fig. 19, Trinity works sufficiently well in the sense that most of the relative PDRs are beyond 80% and 8 out of 16 are over 90%. Moreover, the relative PDR is lower for a node further (in hop) from the sink. This is expectable as Trinity uses a fixed tree topology for data collection, where CTP adapts its routing paths to the link qualities. However, this is a (small) price Trinity has to pay for being able to operate under a very low duty-cycle.

7.5 Airflow Monitoring in Action

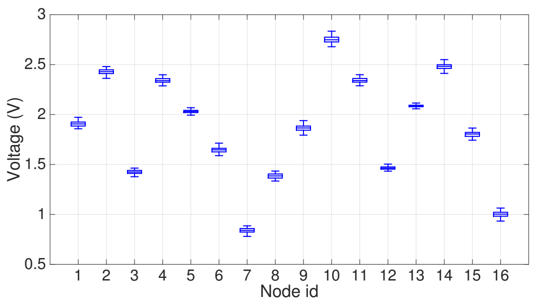


Fig. 20. Voltage readings.

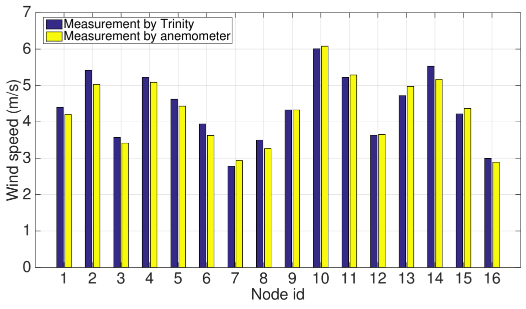


Fig. 21. Average airflow speeds.

According to Sec. 5, our Trinity WSN can sense the airflow speeds of the outlets that they are attached to, with an almost zero energy consumption. Therefore, we collect the readings from our energy-free “airflow sensors” through the WSN during the Sep 2015. At the same time, we also use the hand-held anemometer to check the same spots intermittently. Since our research institute does not allow the HVAC system to be freely tuned, we may simply observe a constant airflow speed at each given spot. We first show the statistics of readings of the MicaZ Motes in Fig. 20. It can be observed that the variances of the voltage values are negligible and the mean values can definitely be used to infer the airflow speeds (by consulting Fig. 13).

In Fig. 21, we show the monthly averages of the airflow speeds measured at the 16 outlets by both Trinity and the hand-held anemometer. The differences between the two sets of measurements are minor, and they can result from the errors of both approaches. Most importantly, we are now convinced that we can totally rely on our self-sustaining Trinity to report the measurements.

8 RELATED WORK AND DISCUSSIONS

The recent decade has witnessed an increasing interest in harvesting energy for sustainable WSNs [15]. As WSNs are usually applied to monitor surrounding environment, one choice is to extract ambient energy, e.g., solar [19, 36, 39, 52, 54], vibration [47], heat [38, 53], and radio [9, 40, 43].

Solar power is popularly explored by many proposals. As the first solar-based system integrating energy harvesting and power management, Heliomote [36] equips Mica2 Motes with solar panels and uses an adaptive schedule mechanism to bridge the applications at upper layer to the power management at lower layer. Corke *et al.* [19] studies the effects of combining a DC-DC converter with NiMH batteries to build a solar-drive wireless sensor networks. Zhu *et al.* [54] build their solar-based platform using ultra-capacitors instead of rechargeable batteries. Keeping an eye on the leakage of the capacitor, the network behaviors can be adaptively adjusted thereby prolonging the lifetime of the network.

Thermal energy harvesting is a recently explored field. Temperature difference of the steam pipeline is used to produce power up to 0.8W [53], and the thermoelectric harvester is again used in [38] to harness the small temperature difference in water. Due to the heavily limited energy supply, the latter system adopts an extremely low power wake up strategy: a node wakes up only upon a water flow event that supplies it with sufficient energy. Our work goes close to [38, 53] in that we all deliver an integrated sensing system that draws energy from the sensing objects, but Trinity differs from these existing proposals by positioning itself as an indoor sensing system, and it indeed faces great challenges very different from those in [38, 53]. While the energy that can be harvested by Trinity is far lower than that from steam pipelines [53], Trinity has to constantly monitor the environment instead of being triggered only by events [38].

Harvesting energy from ambient sources is also possible indoors. [43] presents WISP as a RFID-based platform for battery-free sensing and computation. Since harvesting energy from RF results in an extremely low voltage, a charging pump is used in WISP to raise the voltage. In contrast, the piezoelectric effect in Trinity features a high voltage (up to tens of volts, as shown in Sec. 3), due to the extremely high internal resistance (in a scale of MΩ) of the piezoelectric sheet. Therefore, our power management module employs a low-loss full wave bridge rectifier and a highly efficient DC/DC converter (both of which are build in chip LTC3588-1) to regulate the high alternating voltage to suit the 3.3V direct voltage input of a full-functional MicaZ Mote. In [52], sensor nodes equipped with solar cells can draw energy from ambient light. However, this strategy is of quite limited efficiency such that the energy-harvesting sensor nodes should be integrated with another well-powered backbone network. Moreover, light source may not always be available indoors. In some specific indoor application scenarios, e.g., power monitoring, a sensor network may be deployed close to power grid. Therefore, [21] proposes a self-sustaining power metering system. Connecting an AC load to a wired supplying power results in a magnetic field, from which the system can harvest energy. Similar to our Trinity system, the harvester also acts as an energy-free sensor (or meter) to measure the connected AC load, based on the fact that the rate of harvesting energy indicates the power draw of the load. Therefore, we believe that the principle of our Trinity system, i.e., harvesting energy from the event being sensed, has the potential of being extended to a broader indoor application scenario.

Clock synchronization and calibration are fundamental services for sensor networks. To estimate and remove clock drifts, one choice is to rely on frequent flooding, e.g., [25, 37], which lead to unfordable energy consumption for Trinity . [30] proposes a calibration strategy for low duty-cycle sensor networks, where clock drift is modeled as a stochastic process. Although it works with sparse radio communications, it is not suitable for Trinity due to the resulting significant calculations (and thus energy consumption again). The other category of solutions resort to external reference signals, e.g., FM radios [32], light [29, 34], and electromagnetic radiation [33], and usually rely on dedicated devices to receive external reference signals (which may be energy-consuming). In fact, our Trinity combines the above two kinds of methods. On one hand, our per-link synchronization relies on radio communications, but the synchronization information is piggybacked with data traffic and it thus is virtually energy-free. On the other hand, our energy harvester outputs reference signals for calibration while supplying energy to sensor node; therefore, dedicated receiving devices are not required, and only a small amount of energy is spent on processing the reference signals.

Improving the performance of data delivery in duty-cycle sensor networks is also widely investigated in many existing proposals. For example, [28] proposes a dynamic switching-based data forwarding strategy, to optimize data delivery ratio, communication delay and energy consumption through adaptively selecting forwarding nodes at each hop. The idea of dynamic switching is then applied to data flooding [17]. Specifically, the tree structure can be dynamically adjusted based on packet reception results for the purpose of energy efficiency and delay reduction. Unfortunately, the flexible route selection is usually based on competent link quality assessment and thus may entail (relatively) extensive message exchange, while our Trinity cannot afford the induced energy consumption. Moreover, the duty-cycle ratio of Trinity is restricted to around 0.2%; hence, we may not be able to assess link quality accurately, based on such limited communications. Fortunately, although our current prototype implementation adopts a fixed network topology, the experiment results in Sec. 7.4 show that we do not sacrifice the data delivery performance too much.

9 CONCLUSIONS

We have presented Trinity as a self-sustaining and integrated indoor sensing system. Trinity encapsulates three main components: energy harvesting, synchronous duty-cycling, and sensing, and it fully sustains itself by harvesting the energy from the airflow issued by the HVAC outlets. Being the very first one of its kind (to the best of our knowledge), Trinity endows us with the privilege to answer several questions that were never fully addressed. First of all, we are now convinced that the vibration caused by indoor airflow is a promising resource for piezoelectric energy harvesting. Secondly, we have obtained the first-hand experiment results on the amount of power that can be generated (under various conditions) by this type of piezoelectric energy harvesting (which is far lower than what can be obtained outdoors). Thirdly, we have gained sufficient experience on designing a proper power management module to marry a piezoelectric energy harvester and a commonly used sensor node. Finally, although indoor energy harvesters can only generate limited energy, Trinity serves as a perfect demonstration that a meaningful indoor sensing system can sustain itself by relying on these energy harvesters, with a carefully designed network operation (in particular duty-cycling) mode.

At the meantime, we are testing Trinity in a data center, where a sensing system monitoring rack temperatures is crucial [35] but the HVAC system works in a different manner. We are also integrating Trinity with a novel multi-channel MAC [31] to improve its reliability. Moreover, a new integrated design that packs everything into one circuit board is being studied. Finally, we intend to add more nodes into our current deployment and open it for serving public testing. Different from existing testbeds (e.g., [6, 22]), our Trinity-based testbed will be dedicated to testing indoor applications that accommodate a low duty-cycle.

The system building cost, which is about 80USD except the radio board, may be another concern. The current cost is reasonable for a prototype, but we are working towards a better system integration that may further lower the cost.

REFERENCES

[1] *8-bit Atmel Microcontroller with 128KBytes In-System Programmable Flash.* www.atmel.com/Images/doc2467.pdf.
[2] *Chipcon's CC2420 2.4G IEEE 802.15.4/ZigBee-ready RF Transceiver.* www.ti.com/lit/ds/symlink/cc2420.pdf.
[3] *EH-Link: Energy Harvesting Wireless Node.* http://www.microstrain.com/wireless/eh-link.
[4] *LabVIEW System Design Software.* www.ni.com/labview/.
[5] *LTC3588-1 Piezoelectric Energy Harvesting Power Supply.* cds.linear.com/docs/en/datasheet/35881fa.pdf.
[6] *MoteLab: Harvard Sensor Network Testbed.* motelab.eecs.harvard.edu/.
[7] *NI 9229 4-Channel, 24-Bit Analog Input Module.* sine.ni.com/nips/cds/view/p/lang/en/nid/208796.
[8] *Supercapacitors PHB Series.* http://www.cooperindustries.com/content/dam/public/bussmann/Electronics/Resources/product-datasheets/bus-elx-ds-4402-phb-series.pdf.
[9] A. Abdulhadi and R. Abhari. 2016. Multiport UHF RFID-Tag Antenna for Enhanced Energy Harvesting of Self-Powered Wireless Sensors. *IEEE Trans. on Industrial Informatics* 12, 2 (2016), 801–808.
[10] Y. Agarwal, B. Balaji, R. Gupta, J. Lyles, M. Wei, and T. Weng. 2010. Occupancy-driven Energy Management for Smart Building Automation. In *Proc. of the 2nd ACM BuildSys.* 1–6.
[11] S. Anton and H. Sodano. 2007. A Review of Power Harvesting Using Piezoelectric Materials (2003-2006). *Smart Materials and Structures* 16, 3 (2007), R1–R21.
[12] ASHRAE. 1995. ANSI/ASHRAE Standard 135-1995: BACnet. (1995).
[13] A. Barrero-Gil, A. Sanz-Andrés, and M. Roura. 2009. Transverse Galloping at Low Reynolds Numbers. *Journal of Fluids and Structures* 25, 7 (2009), 1236–1242.
[14] M. Buettner, G. Yee, E. Anderson, and R. Han. 2006. X-MAC: A Short Preamble MAC Protocol for Duty-cycled Wireless Sensor Networks. In *Proc. of the 4th ACM SenSys.* 307–320.
[15] A. Chandrakasan, R. Amirtharajah, S. Cho, J. Goodman, G. Konduri, J. Kulik, W. Rabiner, and A. Wang. 1999. Design Considerations for Distributed Microsensor Systems. In *Proc. of IEEE CICC.* 279–286.
[16] J. Chen, J. Li, and T. Lai. 2013. Trapping Mobile Targets in Wireless Sensor Networks: An Energy-Efficient Perspective. *IEEE Trans. on Vehicular Technology* 62, 7 (2013), 3287 – 3300.
[17] L. Cheng, J. Niu, Y. Gu, and T. He. 2016. Achieving Efficient Reliable Flooding in Low-Duty-Cycle Wireless Sensor Networks. *IEEE/ACM Trans. on Networking* 24, 6 (2016), 3676–3689.
[18] R. Clark, D. Cox, H. Curtiss, J. Edwards, K. Hall, D. Peters, R. Scanlan, E. Simiu, F. Sisto, Th. Strganac, and E. Dowell. 1995. *A Modern Course in Aeroelasticity.* Springer.
[19] P. Corke, P. Valencia, P. Sikka, T. Wark, and L. Overs. 2007. Long-duration Solar-powered Wireless Sensor Networks. In *Proc. of the 4th ACM EmNets.* 33–37.
[20] P. Corke, T. Wark, R. Jurdak, W. Hu, P. Valencia, and D. Moore. 2010. Environmental Wireless Sensor Networks. *Proceedings of IEEE* 98, 11 (2010), 1903–1917.
[21] S. DeBruin, B. Campbell, and P. Dutta. 2013. Monjolo: An Energy-Harvesting Energy Meter Architecture. In *Proc. of the 11th ACM SenSys.* 18:1–18:14.
[22] M. Doddavenkatappa, M. Chan, and A. Ananda. 2012. Indriya: A Low-cost, 3D Wireless Sensor Network Testbed. *Testbeds and Research Infrastructure. Development of Networks and Communities* 90 (2012), 302–316.
[23] Q. Dong, L. Yu, Z. Hong, and Y. Chen. 2010. Design of Building Monitoring Systems Based on Wireless Sensor Networks. *Wireless Sensor Netowrks* 2, 9 (2010), 703–709.
[24] P. Dutta, S. Dawson-Haggerty, Y. Chen, C. Liang, and A. Terzis. 2010. Design and Evaluation of a Versatile and Efficient Receiver-initiated Link Layer for Low-power Wireless. In *Proc. of the 8th ACM SenSys.* 1–14.
[25] S. Ganeriwal, R. Kumar, and M. Srivastava. 2003. Timing-Sync Protocol for Sensor Networks. In *Proc. of the 1st ACM SenSys.* 138–149.
[26] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis. 2009. Collection Tree Protocol. In *Proc. of the 7th ACM SenSys.* 1–14.
[27] Y. Gu, L. He, T. Zhu, and T. He. 2014. Achieving Energy-Synchronized Communication in Energy-Harvesting Wireless Sensor Networks. *ACM Trans. on Embedded Computing Systems* 13, 2 (2014), 68:1–68:26.

[28] Y. Gu and T. He. 2007. Data Forwarding in Extremely Low Duty-Cycle Sensor Networks with Unreliable Communication Links. In *Proc. of the 5th ACM SenSys.* 321–334.

[29] X. Guo, M. Mohammad, S. Saha, M. Chan, S. Gilbert, and D. Leong. 2016. PSync: Visible light-based time synchronization for Internet of Things (IoT). In *Proc. of the 35th IEEE INFOCOM.* 1–9.

[30] H. Huang, J. Yun, Z. Zhong, S. Kim, and T. He. 2013. PSR: Practical Synchronous Rendezvous in Low-Duty-Cycle Wireless Networks. In *Proc. of the 32nd IEEE INFOCOM.* 2661–2669.

[31] F. Li, J. Luo, G. Shi, and Y. He. 2017. ART: Adaptive fRequency-Temporal co-existing of ZigBee and WiFi. *IEEE Trans. on Mobile Computing* 16, 3 (2017), 662–674.

[32] L. Li, G. Xing, L. Sun, H. Wei, R. Zhou, and H. Zhu. 2011. Exploiting FM Radio Data System for Adaptive Clock Calibration in Sensor Networks. In *Proc. of the 9th ACM MobiSys.* 169–182.

[33] Y. Li, R. Tan, and D. Yau. 2017. Natural Timestamping Using Powerline Electromagnetic Radiation. In *Proc. of the 16th ACM/IEEE IPSN.* 55–66.

[34] Z. Li, W. Chen, C. Li, M. Li, X. Li, and Y. Liu. 2014. Flight: Clock Calibration and Context Recognition using Fluorescent Lighting. *IEEE Trans. on Mobile Computing* 13, 7 (2014), 1495–1508.

[35] C. Liang, J. Liu, L. Luo, A. Terzis, and F. Zhao. 2009. RACNet: A High-Fidelity Data Center Sensing Network. In *Proc. of the 7th ACM SenSys.* 15–28.

[36] K. Lin, J. Yu, J. Hsu, S. Zahedi, D. Lee, J. Friedman, A. Kansal, V. Raghunathan, and M. Srivastava. 2005. Heliomote: Enabling Long-lived Sensor Networks Through Solar Energy Harvesting. In *Proc. of the 3rd ACM SenSys.* 309–309.

[37] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi. 2004. The Flooding Time Synchronization Protocol. In *Proc. of the 2nd ACM SenSys.* 39–49.

[38] P. Martin, Z. Charbiwala, and M. Srivastava. 2012. DoubleDip: Leveraging Thermoelectric Harvesting for Low Power Monitoring of Sporadic Water . In *Proc. of the 10th ACM SenSys.* 225–238.

[39] A. Matthews, S. Bobovych, N. Banerjee, J. Parkerson, R. Robucci, and C. Patel. 2015. Perpetuu: A Tiered Solar-Powered GIS Microserver. *ACM Trans. on Embedded Computing Systems* 14, 4 (2015), 78:1–78:21.

[40] H. Nishimoto, Y. Kawahara, and T. Asami. 2010. Prototype Implementation of Ambient RF Energy Harvesting Wireless Sensor Networks. In *Proc. of IEEE Sensors.* 1282–1287.

[41] M. Paidoussis, S. Price, and E. de Langre. 2011. *Fluid-Structure Interactions: Cross-Flow-Induced Instabilities.* Cambridge University Press.

[42] J. Polastre, J. Hill, and D. Culler. 2004. Versatile Low Power Media Access for Wireless Sensor Networks. In *Proc. of the 2nd ACM SenSys.* 95–107.

[43] A. Sample, D. Yeager, P. Powledge, A. Mamishev, and J. Smith. 2008. Design of an RFID-Based Battery-Free Programmable Sensing Platform. *IEEE Trans. on Instrumentation and Measurement* 57, 11 (2008), 2608–2615.

[44] J. Sirohi and R. Mahadik. 2012. Harvesting Wind Energy Using a Galloping Piezoelectric Beam. *Journal of Vibration and Acoustics* 134, 1 (2012), 011009.1–011009.6.

[45] J. Taneja, A. Krioukov, S. Dawson-Haggerty, and D. Culler. 2013. Enabling Advanced Environmental Conditioning with a Building Application Stack. In *Proc. of the 4th IGCC.*

[46] N. Watthanawisuth, A. Tuantranont, and T. Kerdcharoen. 2009. Microclimate Real-time Monitoring based on ZigBee Sensor Network. In *Proc. of IEEE Sensors.* 1814–1818.

[47] C. Williams and R. Yates. 1995. Analysis Of a Micro-electric Generator For Microsystems. In *Proceedings of the 8th International Conference on Solid-State Sensors and Actuators Eurosensors.* 369–372.

[48] Z. Wu, Z. Liu, X. Huang, and J. Liu. 2009. Real-time Indoor Monitoring System Based on Wireless Sensor Networks. In *Proc. of SPIE.* 22–25.

[49] T. Xiang, Z. Chi, F. Li, J. Luo, L. Tang, L. Zhao, and Y. Yang. 2013. Powering Indoor Sensing with Airflows: A Trinity of Energy Harvesting, Synchronous Duty-cycling, and Sensing. In *Proc. of the 11th ACM SenSys.* 16:1–16:14.

[50] Y. Yang, J. Hao, J. Luo, and S. Pan. 2017. CeilingSee: Device-Free Occupancy Inference through Lighting Infrastructure Based LED Sensing. In *Proc. of the 15th IEEE PerCom.* 247–256.

[51] Y. Yang, L. Zhao, and L. Tang. 2013. Comparative Study of Tip Cross-Sections for Efficient Galloping Energy Harvesting. *Appl. Phys. Lett.* 102, 6 (2013), 064105:1 – 064105:4.

[52] L. Yerva, B. Campbell, A. Bansal, T. Schmid, and P. Dutta. 2012. Grafting Energy-Harvesting Leaves onto The Sensornet Tree. In *Proc. of the 11th ACM/IEEE IPSN.* 197–208.

[53] C. Zhang, A. Syed, Y. Cho, and J. Heidemann. 2011. Steam-powered Sensing. In *Proc. of the 9th ACM SenSys.* 204–217.

[54] T. Zhu, Z. Zhong, G. Yu, T. He, and Z. Zhang. 2009. Leakage-aware Energy Synchronization for Wireless Sensor Networks. In *Proc. of the 7th ACM MobiSys.* 319–332.

Received 0000; revised 0000; accepted 0000