U-Wear: Software-Defined Ultrasonic Networking for Wearable Devices

G. Enrico Santagati and Tommaso Melodia
Wireless Networks and Embedded Systems Laboratory
Northeastern University, Boston, MA, U.S.A.
{santagati, melodia}@ece.neu.edu

ABSTRACT

Wearable medical sensing devices with wireless capabilities have become the cornerstone of many revolutionary digital health applications that promise to predict and treat major diseases by acquiring and processing health information. Existing wireless wearable devices are connected through radio frequency (RF) electromagnetic wave carriers based on standards such as Bluetooth or WiFi. However, these solutions tend to almost-blindly scale down traditional wireless technologies to the body environment, with little or no attention to the peculiar characteristics of the human body and the severe privacy and security requirements of patients. We contend that this is not the only possible approach, and we present U-Wear, the first networking framework for wearable medical devices based on ultrasonic communications.

U-Wear encloses a set of physical, data link and network layer functionalities that can flexibly adapt to application and system requirements to efficiently distribute information between ultrasonic wearable devices. U-Wear also offers reconfiguration functionalities at the application layer to provide a flexible platform to develop medical applications. We design two prototypes that implement U-Wear and operate in the near-ultrasonic frequency range using commercial-off-the-shelf (COTS) speakers and microphones. Despite the limited bandwidth, i.e., about 2 kHz, and COTS hardware components not optimized for operating at high frequency, our prototypes (i) achieve data rates up to 2.76 kbit/s with bit-error-rate lower than 10^{-5} using a transmission power of 20 mW; (ii) enable multiple nodes to share the medium; and (iii) implement reconfigurable processing to extract medical parameters from sensors with high accuracy.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design

Keywords

Wearable Devices, Acoustic Communications, Digital Health, Body Area Networks

MobiSys'15, May 18–22, 2015, Florence, Italy.

http://dx.doi.org/10.1145/2742647.2742655 .

1. INTRODUCTION

Wearable medical sensing and actuating devices with wireless capabilities have become the cornerstone of many revolutionary digital health applications [1]. Wearable electrocardiography (ECG) devices and blood pressure sensors can enable remote cardiovascular monitoring for early diagnosis of cardiac arrhythmias and hypertension [2], and therefore prevent heart failures and strokes. Skin patches with wireless connectivity can be arranged in a closed-feedback-loop drug delivery system [3]. For instance, a sensor patch can measure the level of subcutaneous blood glucose, while a drug pump patch can adaptively deliver the required amount of insulin [4]. Motion sensors, e.g., accelerometers, can collect large amounts of data for fitness and medical applications. For example, wireless motion trackers can record athletes' step rate, speed, and acceleration for performance monitoring. Similar devices can also collect information for post-surgery telerehabilitation in case of lower-limb injuries or strokes [5], or measure the motor degradation of patients affected by *Parkinson* disease [6]. Likewise, by correlating motion with sleep activity, the same sensors can monitor the REM sleep duration of a patient and provide information on the development of *post-traumatic stress disorders* [7].

Existing wireless wearable medical devices are connected through radio frequency (RF) electromagnetic waves. Standards in use are often scaled down versions of wireless technologies (e.g., Bluetooth and WiFi), with little or no attention to the peculiar characteristics of the human body and to the severe privacy and security requirements of patients. For example, many commercial activity trackers [8], as well as smart glasses [9], smart watches [10] and smart clothing [11] connect to smartphones using Bluetooth or WiFi. Alternatively, other medical monitoring solutions [12, 13] use proprietary RF-based technologies to collect medical data in a centralized manner. We contend that this is not the only possible approach, and that RF-based technologies have several limitations that can negatively affect the patients' medical experience with wearable devices.

Limitations of RF Technology. First, the RF frequency spectrum is scarce and already crowded with many devices interfering with one another. At the same time, the number of wireless devices that compete to access the RF spectrum is growing exponentially. This includes wireless sensors, but also more largely and pervasively deployed RF devices such as WiFi and Bluetooth, and even microwave ovens. Quoting the FDA's recent guideline on wireless med-

This material is based upon work supported by the National Science Foundation under grant CAREER CNS-1253309.

ical devices, "an increasingly crowded RF environment could impact the performance of RF wireless medical devices" [14]. Therefore, RF-based technologies raise serious concerns about potential interference from existing RF communication systems that can unintentionally undermine the reliability and security of the wearable network, and ultimately the safety of the patient. Second, RF communications can be easily jammed, i.e., intentionally disrupted by artificially generated interference, or eavesdropped by malicious agents using cheap and off-the-shelf equipment, i.e., a simple radio device. Jamming may not even be illegal on ISM spectrum frequencies where devices are allowed to transmit with no need for special permissions. This raises major privacy and security red flags for wearable networks, and a risk for the patient. Third, the RF spectrum is strictly regulated. This clearly constrains the system in terms of flexibility in allocating spectrum resources. Fourth, the medical community is still divided on the risks caused by continuous exposure of human tissues to RF radiation - the World Health Organization classifies RF waves as "possibly carcinogenic to humans" [15]. Therefore, a massive deployment of RF wearable devices on the body may represent a potential risk for the patient. Finally, the dielectric nature of the human body also affects the coupling between on-body RF antennas and the body itself. In particular, the gain and the radiation pattern of the antenna deteriorate because of the contact or proximity with the human body [16], while the resonant frequency and the input impedance of the antenna may shift from their nominal values.

Based on these observations, in this paper, we propose to use ultrasonic waves to interconnect wearable devices, and present U-Wear, the first software-defined networking framework for wearable medical devices based on ultrasonic communications. Ultrasonic waves are acoustic waves with frequency higher than the upper threshold for human hearing (nominally 20 kHz). Acoustic waves have found application in underwater communications [17], in airborne applications [18] such as indoor localization [19, 20] and gesture recognition systems [21], and, massively, in ultrasonic medical imaging [22]. Finally, ultrasonic waves have been proven to be a viable alternative to RF waves for creating wireless networks of implantable medical devices [23, 24, 25, 26].

Advantages of U-Wear. U-Wear has several advantages over traditional networking frameworks based on RF communications.

i. U-Wear eliminates any potential conflict with existing RF systems and overcrowded RF environments.

ii. The ultrasonic frequency spectrum is (at least currently) unregulated and enables nodes to flexibly adapt the occupied frequency to specific requirements such as maximum level of tolerable co-channel interference, maximum tolerable channel multipath and Doppler spreading in the channel and minimum data rate needed at the application layer, among others.

iii. As compared to RF waves, ultrasonic waves do not easily penetrate through solid materials and do not propagate far in air; therefore, ultrasonic communication systems are inherently more secure with respect to eavesdropping and jamming attacks that would require close proximity between the attacker and the victim.

iv. The medical experience of the last decades has demonstrated that ultrasounds are fundamentally safe, as long as acoustic power dissipation in tissues is limited to predefined safety levels [27, 23].

v. By equipping wearable devices with ultrasonic transducers, U-Wear can implement ultrasonic power transmission schemes [28] that enable wireless battery charging functionalities.

vi. On-board ultrasonic transducers can be used to enable acoustic localization and tracking, which are known to have better accuracy than RF-based systems because of the low propagation speed of sound in air [29].

vii. U-Wear can easily be interfaced with ultrasonic intra-body networks [24], and can work as a bridge between implantable intra-body sensors and the external world.

viii. The U-Wear software-defined framework runs on general-purpose hardware; thus, it allows commercial devices, such as smartphones, laptops and smart-TVs, to communicate with wearable devices in the near-ultrasonic frequency range, i.e., $17 - 22$ kHz, using commercial-off-the-shelf (COTS) speakers and microphones [30].

ix. Finally, the U-Wear software-defined ultrasonic networking functionalities can be reconfigured to adapt to application requirements, offering more flexibility with respect to RF-based networking systems entirely implemented in hardware, e.g., Bluetooth or WiFi.

U-Wear consists of a set of software-defined cross-layer functionalities designed to network ultrasonic wearable devices that offer real-time reconfigurability at different layers of the protocol stack, i.e., the physical (PHY), data link, network and application layer. Specifically, U-Wear encloses a set of PHY, data link and network functionalities that can flexibly adapt to the application and system requirements to efficiently distribute information among wearable devices. U-Wear also offers real-time reconfigurability at the application layer to provide a flexible platform to develop medical applications. In particular, sensor data processing applications running in the nodes are decomposed into *primitive building blocks* that can be arbitrarily arranged to create new sensing applications that fit user requirements.

As a proof of concept, we design two prototypes that implement the U-Wear framework and operate in the near-ultrasonic frequency range, i.e., $17 - 22$ kHz, using COTS speakers and microphones. Specifically, we operate at 18kHz with a bandwidth of about 2 kHz. Despite the use of COTS audio components not optimized for operations at high frequency, our prototypes (i) can achieve data rates up to 2.76 kbit/s with bit-error-rate (BER) lower than 10^{-5} at a transmission power of 20 mW; (ii) enable multiple nodes to share the same medium, with tradeoffs between packet delivery delay and packet drop rate; and (iii) implement reconfigurable data processing to extract medical parameters from sensors with high accuracy. Moreover, U-Wear proposes for the first time the use of a Gaussian minimum-shift keying (GMSK) signaling scheme for the near-ultrasonic frequency range that allows our prototypes to achieve relatively high data rates when compared to previously proposed near-ultrasonic systems; more importantly, it ensures virtually inaudible[1] click-free transmission because of the GMSK phase-continuity as we will discuss in Section 3.1.1.

Note that we do not claim improved performance as compared to RF in terms of traditional performance metrics

[1]The upper threshold for human hearing is nominally 20kHz, however hearing starts degrading significantly above 15 kHz.

(e.g., data rate, bit error rate, energy consumption). Any such comparison would clearly be premature - current performance of RF systems is the result of many years of research and development in multiple fields and of a multi-billion dollar industry. Yet, we believe that, for the reasons discussed above, ultrasonic networking is a viable alternative, and in this paper we present a proof of concept and explore several system tradeoffs.

The paper also reports on the design principles and on the core challenges faced during the design and implementation of U-Wear. In Section 2, we discuss the fundamentals of ultrasonic communications along the body. This study highlights the core differences between ultrasonic propagation and RF propagation in air, including the relatively low speed of sound compared to RF, Doppler spreading, and the multipath effect caused by reflections and scattering. In Section 3, we introduce U-Wear, the first networking framework for wearable medical devices, and discuss its architecture. We highlight design choices at different layers of the protocol stack made to overcome limitations posed by the propagation characteristics of ultrasonic waves in air. For example, two signaling schemes (GMSK and orthogonal frequency-division multiplexing) are selected because of their high spectral efficiency and resilience to multipath. Two different synchronization modes can be alternatively used for channels strongly affected by multipath or by Doppler effect. Upper layer protocols and functionalities are designed to address challenges posed by the long propagation delays of ultrasounds in air that might prevent accurate timing. In Section 4, we present the design and implementation of two prototypes. Hardware components are selected with the objective of striking a reasonable balance between keeping the hardware design simple and inexpensive, and meeting the communication performance requirements of the family of applications we consider. The software architecture has also been carefully designed to enable reuse of existing code and to meet the constraints posed by the resource constrained devices in use. In Section 5, we thoroughly discuss the performance of U-Wear through experimental results. In Section 6, we discuss related work, and finally, in Section 7, we conclude the paper.

2. ULTRASONIC COMMUNICATIONS IN AIR

Ultrasounds are mechanical pressure waves that propagate through elastic media at frequencies above the upper limit for human hearing, i.e., 20 kHz.

Attenuation. Two mechanisms mainly contribute to acoustic attenuation in air, i.e., spreading loss and absorption loss [31]. The former includes spherical spreading, i.e., the acoustic pressure falls off proportionally to the surface area of a sphere. The latter is mainly related to atmospheric absorption caused by the interaction of the acoustic wave with the gas molecules of the atmosphere, and is frequency-, temperature-, and humidity-dependent.

For a signal at frequency f [Hz] over a transmission distance d [m], the attenuation can be expressed in [dB] as

$$A_{dB} = 20\log_{10}(d) + d\,\alpha(f), \quad (1)$$

where $\alpha(f)$ [dB/m] is the absorption coefficient, which increases quadratically with the frequency, but also depends on the ambient atmospheric pressure, temperature, and the molar concentration of water vapor, i.e., humidity [32].

Propagation Speed. The propagation speed of acoustic waves in air is approximately 343 m/s at $20^{\circ}C$ and at atmospheric pressure of 101.325 kPa, as compared to 3×10^8 m/s for RF waves. The speed of sound in air increases with temperature and humidity, going from 331 m/s at a temperature of $0^{\circ}C$ and 10% relative humidity to 351 m/s at a temperature of $30^{\circ}C$ and 90% relative humidity.

Doppler Spreading. Doppler spreading occurs as a result of Doppler shifts caused by relative motion between source and receiver, and is proportional to their relative velocity. Doppler spreading generates two different effects on signals: a frequency translation, and a continuous spreading of frequencies that generates intersymbol interference (ISI), thus causing degradation in the communication performance. Since the speed of sound is several orders of magnitude lower than the speed of electromagnetic waves, the resulting doppler effect is severe, even at relatively low speeds.

Reflections and Scattering. The on-body ultrasonic channel is composed of several interfaces between air and human body, and between air and on-body and near-body objects. Because of this inhomogeneous pattern, the on-body channel can be modeled as an environment with pervasive presence of *reflectors* and *scatterers*. The direction and magnitude of the reflected wave depend on the orientation of the boundary surface and on the acoustic impedance[2] of the different media involved. Scattered reflections occur when an acoustic wave encounters an object that is relatively small with respect to its wavelength. Consequently, the received signal is obtained as the sum of numerous attenuated, possibly distorted, and delayed versions of the transmitted signal.

Air-coupled Ultrasonic Transducers. An ultrasonic transducer is a device that converts electrical signals into ultrasonic signals and vice versa. Ultrasonic transducers can be categorized into two main classes based on the physical mechanism that enables the conversion, i.e., *piezoelectric* and *electrostatic* transducers. A piezoelectric transducer produces a mechanical vibration through a thin piezoelectric element under an external voltage variation, and produces a voltage variation under an external mechanical vibration. In *electrostatic* transducers the fundamental mechanism is the vibration of a thin plate under electrostatic forces.

When sound passes across an interface between two materials, it is in part transmitted and in part reflected. To maximize the acoustic energy radiated by the transducer, the acoustic impedance of the radiating surface should match the acoustic impedance of the propagation medium. Today, microelectro-mechanical (MEMS) technology has enabled the fabrication of microscopic piezoelectric and electrostatic transducers, so-called Micromachined Ultrasonic Transducers (MUTs). With MUTs, the acoustic impedance can be controlled to match the external medium by manipulating the device geometry, making them ideally suited for air-coupled applications [33].

When the operating frequency of the ultrasonic communications falls in the near-ultrasonic frequency range, i.e., $17 - 22$ kHz, acoustic waves can be recorded and generated using COTS components, such as microphones and speakers. Even though COTS components are often designed to op-

[2]The acoustic impedance is defined as the product between the density of a medium ρ and the speed of sound in the medium c.

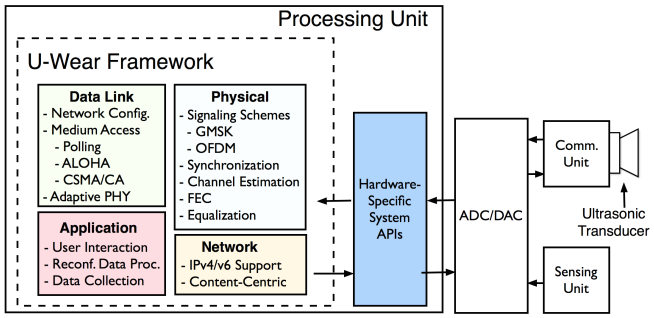

Figure 1: Overview of the U-Wear framework.

erate at lower frequencies, i.e., $0 - 17$ kHz, they can still sense and generate, albeit less efficiently, near-ultrasonic frequency waves. Since many commercial devices such as smartphones, tablets and laptops among others, are equipped with audio interfaces, they can support near-ultrasonic communications with no additional hardware [30].

3. U-Wear ARCHITECTURE

U-Wear consists of a set of software-defined multi-layer functionalities that can be implemented on general-purpose processing units, e.g., microprocessors, microcontrollers or FPGAs, among others, to enable networked operations between wearable devices equipped with ultrasonic connectivity, i.e., air-coupled ultrasonic transducers, and sensing capabilities, i.e., sensors.

Figure 1 shows an overview of the U-Wear framework. U-Wear runs on a processing unit. It accesses an hardware analog-to-digital converter (ADC) and digital-to-analog converter (DAC) through hardware-specific system APIs. In the transmit (Tx) chain, the DAC collects and digital-to-analog converts U-Wear's outputs, i.e., the waveforms to be transmitted, before passing these to the communication unit. In the receive (Rx) chain, the ADC analog-to-digital converts and passes to U-Wear the received waveforms coming from the communication unit. The communication unit consists of an ultrasonic transducer and an amplification stage, i.e., preamplifier in Rx chain and power amplifier in Tx chain. U-Wear also collects the analog-to-digital converted data coming from the sensing unit. U-Wear consists of (i) PHY layer functionalities, e.g., modulation and synchronization, (ii) data link layer functionalities including forward error control and medium access control (MAC) protocols, (iii) network layer functionalities, e.g., IPv4 and IPv6 support and content-centric networking, and (iv) application layer functionalities, i.e., reconfigurable sensing data processing and user interface.

3.1 Physical Layer

U-Wear PHY layer libraries define the signaling scheme, channel estimation, equalization, synchronization and forward error correction (FEC) functionalities.

3.1.1 Signaling Schemes

U-Wear offers two fully-functional signaling schemes, a narrowband scheme based on GMSK modulation, and a wideband scheme based on orthogonal frequency-division multiplexing (OFDM). Moreover, U-Wear includes a set of software-defined primitive blocks, e.g., programmable filters, and Fast Fourier Transform (FFT) modules, among others, that can be used to implement additional signaling schemes.

Narrowband GMSK. GMSK is a continuous-phase modulation (CPM) used worldwide in GSM cellular systems [34]. In frequency-shift keying (FSK) and phase-shift keying (PSK), information is encoded in the variations of the carrier frequency, or carrier phase, respectively. Since frequency and phase switches occur instantaneously, FSK and PSK signals do not have continuous phase. Phase discontinuity generates out-of-band power, leading to poor spectral efficiency. Moreover, in near-ultrasonic transmissions based on COTS speakers and microphones the out-of-band power introduces audible noise (clicks), which makes the communication perceptible to humans.

Differently, GMSK signals have phase continuity, and each symbol is represented by a phase variation, from a start value to a final value, over the symbol duration. Thus, the initial phase of each symbol is determined by the cumulative total phase variation of all previous symbols. A Gaussian filter is used to smooth the phase variation and improve the spectral efficiency. The product between the signal bandwidth B and the symbol time T is a measure of the scheme spectral efficiency. Lower BT product leads to higher spectral efficiency, but increases the intersymbol interference (ISI). Based on these characteristics, GMSK is the signaling scheme of choice for U-Wear narrowband communications in the near-ultrasonic frequency range based on COTS speakers and microphones. Thanks to its phase-continuity, GMSK enables click-free transmissions, and therefore is to be preferred over non-continuous-phase modulations such as FSK and PSK.

Wideband OFDM. OFDM has been extensively used in underwater acoustic communications [35] because of its robustness against frequency-selective channels with long delay spreads. The idea behind OFDM is to use a large number of closely spaced orthogonal sub-carriers, such that for each sub-carrier the channel is subject to flat fading. In each sub-carrier a conventional modulation is used, e.g., M-PSK or M-Quadrature-Amplitude-Modulation (QAM). OFDM offers high spectral efficiency and robustness against narrowband co-channel interference, intersymbol interference (ISI) and multipath fading. Finally, OFDM can be efficiently implemented using FFT and inverse FFT (IFFT) algorithms. These characteristics make OFDM ideal for ultrasonic communications based on wideband transducers.

3.1.2 Synchronization

Synchronization in U-Wear is achieved in two steps. First, an energy collection identifies any incoming packet, i.e., coarse synchronization. Once a packet is detected, the receiver performs a fine synchronization operation that identifies the exact starting point of the packet. Fine synchronization is achieved by correlating the received signal with a local copy of the preamble, i.e., a sequence that precedes each packet, which outputs a peak corresponding to the first sample of the packet. U-Wear offers two synchronization modes:

PN-sequence mode. The pseudo noise (PN)-sequence mode uses PN-sequences as a preamble, i.e., binary sequences with sharp autocorrelation peak and low cross-correlation peaks, that can be deterministically generated. Specifically, we consider maximum length sequences (MLSs), a family of PN-sequences that can be generated in software and hardware through linear feedback shift registers (LFSRs). Be-

cause of their desirable correlation characteristics, PN-sequences have been widely used to enable strong resilience to multipath [36]. Therefore, they are well suited for ultrasonic in-air communications, as discussed in Section 2.

Chirp-based mode. The chirp-based mode uses a chirp signal as preamble, i.e., a sinusoidal waveform whose frequency varies from an initial frequency f_0 to a final frequency f_1 within a certain time T. Chirp signals have been widely used in radars [37] due to their good autocorrelation and robustness against Doppler effect. In fact, a frequency-shifted chirp still correlates well with the original chirp, although with lower amplitude and time-shifted peak. This characteristic makes chirp synchronization desirable in ultrasonic in-air communications under severe Doppler effect conditions as experienced, for example, under fast movement conditions as in sensors worn by athletes for performance monitoring. The price we pay for the Doppler robustness is higher cross-correlation peaks compared to PN-sequences that result in lower resilience to multipath.

3.1.3 Channel Estimation and Equalization

As discussed in Section 2, ultrasonic communications in air are strongly affected by multipath and Doppler spread, leading to frequency selectivity and ISI that compromise the bit recovery operations at the receiver. U-Wear implements channel estimation and equalization functionalities to estimate the channel impulse response (CIR) and mitigate the distortion produced by the channel.

Channel Estimation. U-Wear offers a training-based channel estimation approach that requires the presence of a training sequence known *a-priori* in the transmitted packet. In particular, U-Wear leverages the good autocorrelation property of the synchronization preamble sequence, discussed in Section 3.1.2, to estimate the CIR. By correlating the output of the channel, i.e., the received signal, with the input, i.e., the known preamble sequence, we obtain an estimate of the time-domain CIR [38].

Zero-forcing Equalization. U-Wear implements a linear equalization technique, zero-forcing (ZF) [39], that aims to minimize the ISI signal distortion produced by the channel. As the name suggests, a ZF equalizer is a finite-impulse-response (FIR) filter of order N that, for each input symbol, "forces to zero" the ISI components introduced by the $2N$ adjacent symbols. The filter taps are numerically calculated starting from an estimate of the CIR, which also accounts for the ISI effect.

3.1.4 Forward Error Correction

U-Wear offers a forward error correction (FEC) functionality based on Reed-Solomon (RS) codes. RS codes are linear block error-correcting codes widely used in data storage and data transmission systems. An RS encoder takes k information symbols and adds t *parity* symbols to make an n symbol block. Therefore, there are $t = n - k$ overhead symbols. On the other hand, an RS decoder is able to decode the received n-symbol block, and can correct up to $\frac{t}{2}$ data symbols that may contain potential errors due to the channel fluctuation or collisions with interfering packets. The RS coding rate can be defined as the ratio between the message length and the block length, i.e., k/n.

3.2 Data Link Layer

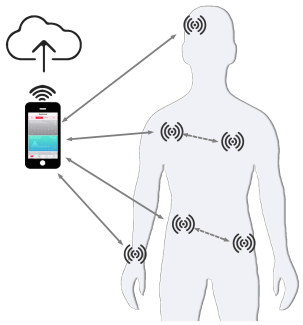

Figure 2: U-Wear scenario with both M/S (continuous line) and P2P (dashed line) network configurations.

The U-Wear data link layer provides a set of functionalities that allow multiple nodes to efficiently access the medium under the challenges posed by the ultrasonic in-air channel, e.g., long propagation delays, among others, as discussed in Section 2.

3.2.1 Network Configuration

U-Wear is designed to internetwork wearable devices in master/slave (M/S) or peer-to-peer (P2P) configurations. Both configurations can coexist in the same network in what we refer to as hybrid configurations. Figure 2 shows a hybrid configuration system design.

Master-Slave Configuration. In the M/S configuration, one node takes the role of master, i.e., network coordinator, while the remaining nodes operate as slaves. In this scenario, the network control is concentrated on a master node, typically with higher resources available, e.g., processing, memory, power and connectivity. For example, M/S configurations may be used in continuous monitoring systems where a master node, e.g., a smartphone or a laptop, is used to fetch, analyze and display data collected by wearable sensors. Wireless or wired Internet connectivity may allow the master node to connect the wearable network with a medical center where the patient's data can be stored, and analyzed remotely.

Peer-to-Peer Configuration. In the P2P configuration, all the network wearable nodes are treated as peers. This scenario suits, among others, applications that require distributed coordination among nodes for closed-feedback-loop monitoring and actuating tasks. For example, this may include a skin patch drug-delivery system where a drug pump can trigger a sensor for measurement, or where a sensor may trigger the drug pump for drug injection after a measurement.

3.2.2 Medium Access Control Protocols

U-Wear offers three fully-functional multiple access protocols, i.e., polling, ALOHA and carrier sense multiple access (CSMA) with collision avoidance (CA), as well as primitive functions to implement custom protocols, e.g., idle listening, random backoff, or checksum-based error control mechanisms.

Polling Protocol. Polling is a deterministic access protocol for the M/S network configuration. In a polling scheme, the master node has complete control over channel access, while each slave node is granted access to the medium in a round-robin fashion.

ALOHA. ALOHA is a random access protocol where nodes do not check whether the channel is busy or idle before transmitting [40]. Nodes that want to transmit data simply access the channel and transmit the data. When collisions occur, nodes attempt retransmissions after a random time interval, i.e., backoff time.

Carrier Sense Multiple Access. CSMA/CA is a multiple access technique based on carrier detection, which allows multiple nodes to share the channel by avoiding simultaneous transmissions, therefore avoiding collisions among transmitted packets [41]. When a node wants to transmit a data packet, it first listens to the channel. If the channel is sensed as idle during a fixed time interval, the node transmits, otherwise it waits for a backoff time before attempting a new transmission.

3.2.3 PHY Layer Adaptation

U-Wear defines a set of cross-layer functionalities that enable real-time reconfiguration of PHY layer parameters from upper layers of the protocol stack, e.g., data link or network layer. By leveraging the flexibility of the software-defined architecture, upper layer protocols can reconfigure on-the-fly PHY layer parameters such as modulation, signal bandwidth and FEC coding rate, among others. Reconfiguration functionalities allow to develop reactive or proactive control algorithms to adapt the underlying communication link to the channel variations or to upper layer protocol requirements [25, 35].

3.3 Network Layer

3.3.1 IPv4 and IPv6 Support

U-Wear provides interoperability with the Internet by defining an adaptation layer that integrates IPv4 and IPv6 protocol support [42]. The adaptation layer consists of a set of functionalities that interface the traditional IP network layer with the U-Wear data link layer, by offering IP header compression and IP packet fragmentation functions optimized for ultrasonic wearable networks with long propagation delays discussed in Section 2 that potentially prevent accurate timing of network protocols. For example, by leveraging cross-layer header information, the long IPv4 and IPv6 headers can be shortened to reduce network delay and energy consumption when exchanging small information packets.

3.3.2 Content-centric Networking

U-wear offers content-centric networking (CCN) functionalities that make the network content directly addressable and routable. Each sensor data or actuation command, i.e., each *content object*, is labeled with a name, and can be accessed through this name. Nodes can request content objects by broadcasting a *request message*. When a match is found, i.e., the content is found on a network node, a *response message* containing the requested content is sent back.

3.4 Application Layer

3.4.1 Reconfigurable and Modular Data Processing

U-Wear adopts the idea of decomposing the data processing applications running in the sensor nodes into *primitive blocks*, and offering real-time reconfigurability at the application layer. The sensing application consists of a sequence of basic operations that are executed on the sensed data

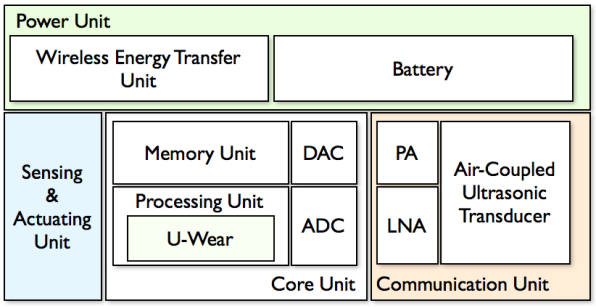

Figure 3: Proposed hardware design of a wuMote.

to extract desired medical parameters. Real-time modular reconfiguration offers three main advantages. First, the network coordinator can wirelessly transmit and install new applications on sensor nodes at runtime, as needed. Based on this, resources are allocated only when the application is requested, thus reducing the processing and memory overhead due to static applications continuously running in background. Second, modular reconfiguration enables programmers to easily create new applications by arranging the primitive building blocks in the desired execution sequence. As a consequence, new medical parameters can be extracted from the raw data coming from a sensor, while maximizing code reusability. Finally, in case of template matching applications, e.g., ECG anomaly detection by matching traces with known templates [43], adding or updating templates becomes very easy with a reconfigurable application layer.

Defining new applications consists of specifying inputs, a chain of primitive blocks, and outputs. An input is the physical sensor that generates the data, e.g., accelerometer or electrocardiogram (ECG). An output can be either the local memory for storing a measured parameter, or a transmission for sending a measured parameter to another node. We divide the set of primitive blocks into three main classes, *filters*, *data operations*, and *detectors*. Filters enable filtering the raw data to remove offsets, drift of the sensors and any other noise components coming from external sources. Data operations include common signal processing operations performed on sensor data, e.g., correlation with templates, and FFT, among others. Finally, detectors allow measuring the desired parameters by detecting specific elements in the processed signal, e.g., peaks, patterns and time distances, among others.

3.4.2 Data Collection

The application layer can operate in two different modalities to exchange and collect data, *fetch* and *push* mode. Fetch mode is used when the application layer requires content from the network. Push mode is used when sensed data needs to be pushed to another node, e.g., high glucose level in the blood, or when a node requires another node to accomplish some actuating operation, e.g., inject insulin or trigger a neurostimulation. In case of actuating commands, the push packet may contain further information about the required action, e.g., the quantity of insulin to inject or the pattern of the neurostimulation.

4. U-Wear PROTOTYPES

We now present the design of two prototypes that implement the U-Wear framework discussed in Section 3. The first U-Wear prototype is a wearable ultrasonic sensor node

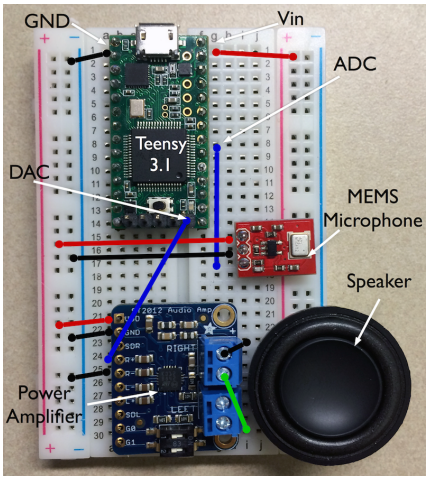

Figure 4: Circuit diagram of the wuMote prototype.

based on a custom hardware platform, which we refer to as *wuMote*. The second prototype is a wearable ultrasonic coordinator based on an iOS commercial smartphone device, which we refer to as *wuMaster*.

4.1 wuMote Prototype

4.1.1 Hardware Design

In Fig. 3, we show the hardware architecture of the wuMote. The *core unit* includes a processing unit, e.g., microprocessor or microcontroller, a memory unit, e.g., RAM or flash memory, and digital-to-analog and analog-to-digital converters. The *processing unit* executes the U-Wear functionalities discussed in Section 3. The *communication unit* enables ultrasonic wireless connectivity by embedding power and low noise amplifiers, and air-coupled ultrasonic transducers. The *power unit* mainly consists of a battery to power the wuMote. An optional *wireless energy transfer unit* can be installed to leverage ultrasonic power transmission to wirelessly charge the node battery. Finally, the *sensing and actuating unit* can incorporate several sensors and actuators according to the specific application design.

We implemented a prototype of the architecture in Fig. 3 based on Teensy 3.1 [44]. The wuMote implementation offers near-ultrasonic capability, by using COTS audio speakers and microphones as air-coupled ultrasonic transducers. Figure 4 shows the basic circuit design of the wuMote prototype on a solderless breadboard. The prototype includes a Teensy 3.1, i.e., the core unit, a power amplifier, a microphone with low noise amplifier, and a small audio speaker, i.e., the communication unit. A lithium ion polymer battery, not shown in the figure, is connected to the bus strip of the breadboard to power the electronic components. A smaller, more compact, and more visually appealing prototype board can be designed by developing a customized PCB board that embeds the wuMote hardware components.

Teensy 3.1. Teensy 3.1 is a small-footprint, i.e., about 3.5×1.8 cm, inexpensive development board, based on a 32 bit ARM Cortex-M4. It comes with 64K of RAM, 256K of Flash, 12 bit DAC, dual ADC, and USB connectivity. Teensy 3.1 can be programmed in C and C++ using Teensyduino, a customized version of the Arduino integrated development environment (IDE), and supports many of the code libraries designed for Arduino and others specifically designed for Teensy, e.g., the audio library [45], among others. Teensy 3.1 can be powered via USB, or through external coin batteries connected to the Vin and GND pins.

We selected Teensy 3.1 over other available COTS platforms such as USRP N210, Raspberry Pi, or Arduino Uno, for the following reasons. Compared to USRP N210 and Raspberry Pi, where software operations are executed on top of an operating system running on external or internal microprocessors, Teensy 3.1 and Arduino Uno are designed around a low-power microcontroller that provides low-level control of the hardware peripherals. Microcontroller-based platforms offer higher hardware flexibility and computational efficiency that suit the design requirements of wireless wearable devices. Finally, we selected Teensy 3.1 over Arduino Uno because of the more powerful microcontroller and larger available memory that can support high audio sampling rates compatible with the near-ultrasonic communication range, e.g., 44.1 kHz for acoustic frequencies up to 22 kHz. Teensy 3.1 still supports the Arduino libraries that can significantly ease the prototyping process of the wuMote. Ideally, lower-power and smaller-packaged solutions would be desirable for a more stable product. In a final product, the Cortex M4 currently used in our prototype would be likely replaced by a Cortex M0 microcontroller such as the mm-size low-power Freescale KL03 microcontroller [46]. Even though development boards for Cortex M0 microcontrollers exist, none of these are comparable to the Teensy platform in terms of hardware and software capabilities, and they do not offer Arduino library support. Thus, at this stage we select the less energy efficient hardware platform Teensy 3.1 in exchange for shorter prototyping time.

Power Amplifier. The wuMote includes a small and efficient class D audio amplifier [47] able to deliver a maximum of 1 W into 4 ohm impedance speakers, with a voltage supply of 3.3 V DC, and efficiency up to 80%. The amplifier consumes less than 2 mA of current when quiescent and less than 2 μA in standby mode. In Fig. 4, the right channel of the power amplifier is connected to Teensy via the DAC pin, and to the speakers via the 3.5 mm screw-terminal blocks. The Vcc and GND pins are connected to the bus strip to power the device.

Microphone. The wuMote includes a tiny breakout board that embeds an ADMP401 MEMS microphone and a low-noise amplifier (LNA) [48]. The ADMP401 offers a mostly flat bandwidth, i.e., -3 dB roll off, between 100 Hz and 15 kHz, omnidirectional sensitivity pattern, and requires a supply voltage between 1.5 V and 3.3 V DC. Although a microphone with larger bandwidth would perform better [49], we selected ADMP401 because of the COTS breakout board package that eases prototyping. Moreover, even though with lower sensitivity, the ADMP401 can still detect higher frequency acoustic waves up to 22 kHz. The microphone is connected to one of the analog pins (ADC) available in Teensy 3.1, and is powered by connecting the Vcc and GND pins to the bus strip.

Audio Speaker. The output of the wuMote is a small and compact COTS speaker, Dayton Audio CE28A-4R [50], with 4 ohm impedance, 4 W maximum output power supported, and flat frequency response between 100 Hz and 15 kHz. The speaker is connected to the power amplifier using 3.5 mm screw-terminal blocks.

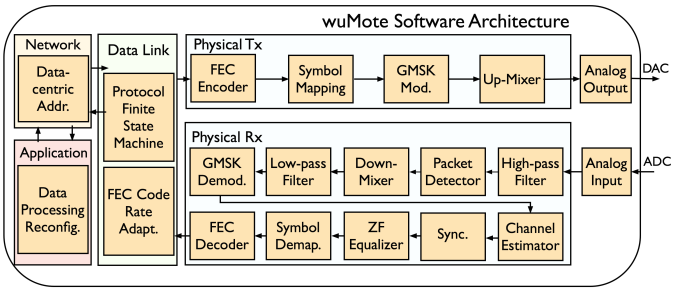

Figure 5: Software architecture of the wuMote prototype.

4.1.2 Software Architecture

We implemented the U-Wear framework in Teensy 3.1 to enable ultrasonic wireless connectivity and networking on the wuMote hardware prototype. In Fig. 5, we show the block diagram of the wuMote software architecture that includes (i) narrowband GMSK transceiver with synchronization, channel estimation, equalization, and FEC functionalities at the PHY layer, (ii) polling and ALOHA multiple access protocol with FEC coding rate reactive adaptation at the data link layer, (iii) content-centric addressing at the network layer, and (iv) data processing reconfiguration with fetch and push support at the application layer.

We implemented the U-Wear functionalities using Teensyduino, an add-on for the Arduino IDE, leveraging many of the code libraries available for the Arduino platform. Since ultrasonic waves are nothing but sound at higher frequencies, we based the PHY layer signal processing on the audio library specifically designed for Teensy 3.1 [45]. The Teensy audio library consists of a set of objects that enable recording, processing, and playback of audio sampled at 44.1 kHz. Objects instantiate specific audio functionalities, e.g., a waveform synthesizer and finite-impulse-response (FIR) filters, while new functionalities can be enabled by creating new objects. A cascade of objects forms a processing chain that performs a set of operations on inputs to produce a desired output. Each object in the chain operates in pipeline on chunks of 128 audio samples, which correspond to 2.9 ms of audio. To guarantee audio continuity, each block must execute its processing operation within 2.9 ms.

In the wuMote implementation we built custom-made objects that implement specific signal processing operations. Finally, since some computationally expensive operations exceed the audio library time constraints of 2.9 ms, we implemented these outside the audio library. We refer to these as *off-the-chain* objects.

Physical Tx. The first object of the PHY layer Tx chain is the `FEC Encoder`. Here, each data packet coming from the data link layer is coded, as discussed in Section 3.1.4, and overhead symbols are appended to the original packet. We select $n = 255$ symbols and parity symbols t to achieve different coding rates. Because of the computation complexity of RS coding, the `FEC Encoder` is implemented as an *off-the-chain* object. The coded packet is then passed to the `Symbol Mapping` object that inputs the audio stream in the processing chain. Here, the coded packet is serialized, i.e., converted into a stream of bits, differentially encoded, and transformed into a non-return-to-zero (NRZ) signal. The NRZ signal is then GMSK modulated by the `GMSK Modulator` object and up-converted to the carrier frequency by the `Up-Mixer` object. The modulated and up-converted waveforms are passed to the `Audio Output` object, i.e., a system API that interfaces U-Wear with the DAC, digital-to-analog converted and transmitted through the audio speaker.

Physical Rx. The received acoustic signal is converted into an electrical signal by the MEMS microphone. The signal is amplified by the LNA, and analog-to-digital converted by the Teensy 3.1 ADC at 44.1 kHz. The `Audio Input` object is a system API that interfaces U-Wear with the embedded Teensy 3.1 ADC, and inputs the audio stream into the PHY layer Rx chain. The received digital signal is first high-pass filtered by the `High-pass Filter` object to eliminate low-frequency noise and interference, i.e., external human voice and ambient noise. The `Packet Detector` processes the input signal to detect incoming packets using an energy-based approach to check wether there is energy at the expected carrier frequency. The incoming packet is then down-converted by the `Down-Mixer`, i.e., converted into a complex in-phase/quadrature baseband signal, and low-pass filtered to eliminate undesired higher-frequency harmonics introduced by the nonlinearity of the down-conversion operation. Channel estimation, synchronization and equalization operations normally follow down-conversion and are applied to the complex baseband signal. However, these operations are computationally expensive, and their execution exceeds the audio library time constraints of 2.9 ms. To overcome this limitation, we first demodulate the complex baseband signal in the `GMSK Demodulator` object to extract the phase variation that carries the coded information bits. Then, we execute off-the-chain the computationally expensive operations. The `Channel Estimator` object estimates the CIR using the packet preamble as training sequence, as discussed in Section 3.1.3, while the `Synchronizer` object attempts to achieve fine synchronization through the PN-based mode discussed in Section 3.1.2. The `ZF Equalizer` object filters the input for ISI recovery, as discussed in Section 3.1.3. The equalized symbols are demapped into a bitstream, collected into a packet structure, and passed to the `FEC Decoder` object. Here, FEC decoding operations attempt to correct potential bit errors, as discussed in Section 3.1.4. Finally, the error-corrected packet is passed to the data link layer.

Data Link Layer. The wuMote data link layer is implemented in a Finite State Machine (FSM) block that currently includes two of the MAC protocols discussed in Section 3.2, i.e., polling and ALOHA. The wuMote data link layer also implements a PHY layer adaptation to optimally select the FEC coding rate that minimizes the number of retransmissions. During packet transmission, the MAC FSM collects data from upper layer protocols and creates the data-link-layer packet. The packet is then forwarded to the PHY layer Tx chain, where it is encoded in a digital waveform before being transmitted. At the receiver side, the MAC FSM detects the received packet based on information coming from the `Packet Detector` block and triggers the PHY layer to start processing the received waveform.

Network Layer. The wuMote network layer implements the content-centric addressing scheme discussed in Section 3.3. Although IPv4 and IPv6 support is an important aspect of the U-Wear framework, at this prototyping stage we left most IP-related aspects out of the scope of this work. Each *content object* is mapped into a binary mask, where it is represented by the bit position in the mask. In an M/S configuration, a node joining the network is first paired with the master. The master maps the content available in the

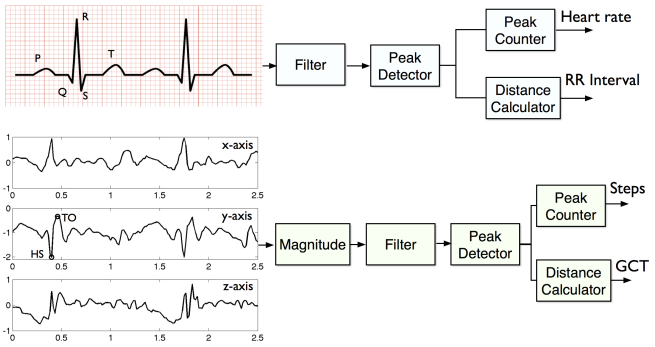

Figure 6: Primitive blocks of heart rate and RR interval monitor (top), and footstep and GCT monitor (bottom).

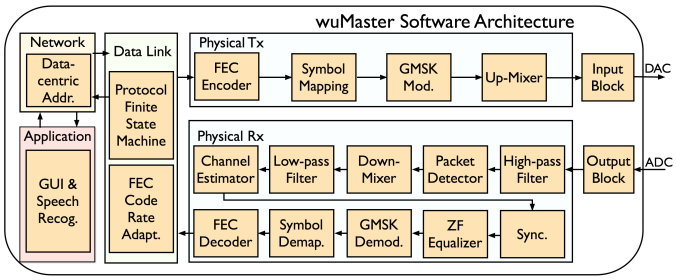

Figure 7: Software architecture of the wuMaster prototype.

new node into the binary mask, and broadcasts the updated mapping to all the nodes in the network. Each node builds a local mask based on the entities that it possesses. To request an entity, the master broadcasts a request message containing a *request mask* with '1' set in the position mapped to the desired content.

Application Layer. The wuMote application layer implements the real-time modular reconfiguration functionalities discussed in Section 3.4.

Based on this modular approach, applications can be represented by chains of binary sequences, i.e., keys. Each primitive function is mapped to a binary key. A concatenation of keys represents a concatenation of operations, and therefore represents an application. The application is encapsulated into reconfiguration packets and transmitted over-the-air. The receiving node extracts the binary keys, and feeds these into an FSM where each state represents a primitive block function. By parsing the consecutive function keys, the FSM transitions from state to state, processing inputs and producing outputs. Inputs and outputs of each function are mapped to binary keys as well, and are implemented in a C-struct `output_struct` that contains a pointer to an array, and a binary key variable. The input and output keys allow to parametrically pass data between functions.

As a proof of concept, we developed some applications based on the primitives discussed above.

Electrocardiogram Processing. We consider a single-electrode ECG signal. Fig. 6 shows a template signal with five labelled characteristic waveforms, P, Q, R, S and T, that correspond to three electrical events during one heart beat, i.e., atrial contraction (P), ventricular contraction (QRS) and ventricular recovery (T). The first application measures the heart rate in beat-per-minute [bpm]. This is done following two different approaches. The first approach, here *R method*, counts the number of peaks, *R waves*, in a 6-second trace and multiplies the result by 10. The second approach, here *RR method*, finds the number of heartbeats per second by inverting the average of the *RR interval* duration, i.e., distance between two consecutive R waveforms in the trace, and multiplies this by 60. This approach has higher complexity with respect to the R-method, but results in higher resolution, i.e., 1 bpm against 10 bpm of the R method. The second application measures average and standard deviations of temporal separation between electrical events in the ECG. For example, the distance between peaks, i.e., R waves, gives the RR interval duration. The mean and standard deviation of the RR interval provide information about potential heart arrhythmias [51]. Figure 6 (top) shows the simplified primitive block sequences of the R method heart rate detector and the RR interval monitor applications.

Accelerometer Processing. The accelerometer trace in Fig. 6 shows the frontal acceleration on the x-axis, the vertical acceleration on the y-axis, and the lateral acceleration on the z-axis. We label in the y-axis two main events that occur during a single step, i.e., *heel strikes* (HSs) and *toe-off* (TO), that correspond to the instants at which the foot touches the ground, and the instants at which the foot leaves the ground, respectively. Based on this, the first application calculates the magnitude of the acceleration from the three-axis components, low-pass filters it to remove high frequency noise, and counts the number of peaks in the resulting signal, i.e., the number of HSs. The peaks within a time interval represent the number of footsteps performed by the patient. The second application measures average distances between events in the accelerometer trace. For example, the distance between non-consecutive peaks in the acceleration magnitude gives the *gait cycle time* (GCT), i.e., time between consecutive HSs on the same foot. GCT offers a measure of the motor degradation of patients affected by *Parkinson* disease [6]. Figure 6 (bottom) shows the simplified primitive block sequences of the footstep counter and the GCT monitor applications.

4.2 wuMaster Prototype

We implemented the wuMaster prototype on the iOS 8 platform for Apple iPhone smartphones. The prototype consists of an app running on an iPhone that implements the U-Wear multi-layer functionalities. In Fig. 7, we show the software architecture of the wuMaster prototype that includes (i) narrowband GMSK transceiver with synchronization, channel estimation, equalization, and FEC functionalities at the PHY layer, (ii) polling and ALOHA multiple access protocol with FEC coding rate reactive adaptation at the data link layer, (iii) content-centric networking at the network layer, and (iv) a graphic user interface (GUI) and speech recognition functionalities at the application layer that allow users to interact with the wearable network.

The iOS prototype wirelessly communicates with the wuMotes through an ultrasonic narrowband GMSK link, using the phone's embedded microphone and speaker. We developed the software prototype in Objective-C programming language using Xcode 6 integrated development environment (IDE) [52]. We use (i) the *vDSP library* [53] of the iOS Accelerate framework that implements digital signal processing operations (DSP), (ii) *Novocaine* [54], a high performance audio library that enables record/play audio func-

tionalities, and (iii) *wit.ai* framework [55] that offers speech recognition services.

vDSP Library. The vDSP library is part of the Accelerate framework available in iOS, and provides several DSP functionalities including vector and matrix arithmetic, Fourier transforms, convolution and correlation operations between real or complex data types. We leverage the vDSP functionalities to perform arithmetic operations and correlations on real and complex vectors in the PHY layer.

Novocaine. Novocaine is a high performance audio processing library for iOS. Novocaine hides all the low-level audio implementation details, giving simple block-based callbacks that are called when audio comes in, and when audio needs to go out. Specifically, a Novocaine object, i.e., the Audio Manager, offers `InputBlock` and `OutputBlock` callbacks, inside which we can simply place the DSP code for processing input and output data.

Wit.ai Framework. Wit.ai provides natural language processing in the form of multi-platform APIs, which we use to integrate voice command in U-Wear. Wit.ai allows developers to create commands, and to match these commands with intents. A command is what the user would say to trigger an operation while the intent represents the operation itself. Voice commands are sent to the wit.ai server, and the intent with maximum confidence is returned as a response.

PHY Layer Tx/Rx. The PHY layer Tx and Rx are implemented in two classes named `PHYLayerTx` and `PHYLayerRx`, respectively. Here, the Novocaine Audio Manager triggers the `InputBlock` and `OutputBlock` callbacks to record and play audio, and the vDSP functions process the input and output data. At the transmitter, the `PHYLayerTx` class gets the data from the data link layer, generates the GMSK waveform, and then passes it to the Audio Manager. The latter transmits the GMSK waveform through the speaker. At the receiver, the operations in `PHYLayerRx` match those implemented in the wuMote PHY layer, discussed in Section 4.1.2. Because of the less stringent memory and processing constraints of the iOS platform, here channel estimation, synchronization and equalization follow the down-conversion, and are applied to the complex baseband signal.

Data Link and Network Layer. The wuMaster data link layer implements polling and ALOHA MAC protocols, as well as FEC coding rate adaptation. The MAC functionalities are implemented in a class named `MACLayer`, where a FSM implements the MAC protocol operations. The wuMaster network layer implements the same content-centric addressing scheme seen in the wuMote prototype, with the exception that here the centralized mapping functionalities are also implemented.

Application Layer. The wuMaster application layer consists of a GUI and a set of wit.ai commands that allow users to interact with the U-Wear multi-layer functionalities. The GUI's principal element is a three-tab `TabViewController` class. The first tab, shown in Fig. 8 (left), contains a `PHYViewController` object. It inherits from `UIViewController`, i.e., a basic view controller in iOS, and is implemented to test the PHY layer performance. The second tab contains a `MACViewController` object that inherits from `UIViewController` and allows the user to test the MAC Layer functionalities by requesting, receiving and visualizing sensed data coming from the deployed wuMotes. The `MACViewController` embeds a `UICollectionView`, a collection of objects that represent the sensed data. In Fig. 8 (center) we show six objects in the collection, which are associated with the number of footsteps, sleep hours, the heart rate, the breathing rate and the respiratory minute volume, the glucose level in the blood and the diastolic/systolic blood pressure. Finally, a `WITMicButton`, defined in the wit.ai framework, enables voice command processing. The third tab contains an `APPViewController` object that gives access to application layer reconfiguration functionalities discussed in Section 3.4. In the `APPViewController` we group the applications based on the sensing unit that provides the required input data, e.g., accelerometer and ECG. Each `UIButton` represents a group of applications, and it gives access to a `PopViewController` object that shows the available applications in that group. Users can select which application to run on the wuMote. For example, in Fig. 8 (right) we show how users can select to install heart rate or RR interval monitor on wuMotes equipped with ECG.

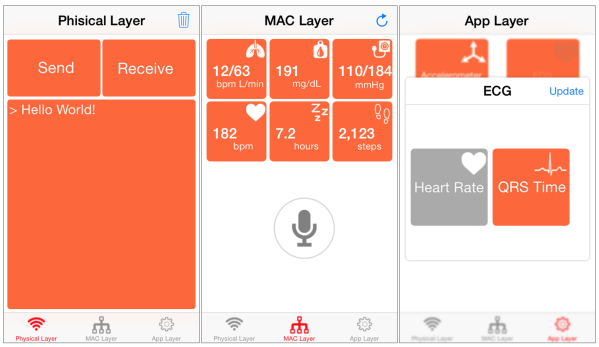

Figure 8: wuMaster GUI for PHY layer (left) and MAC layer (center) and application layer (right).

5. PERFORMANCE EVALUATION

In this Section, we demonstrate the feasibility of ultrasonic communications for wearable devices through testbed experiments, and we evaluate the performance of the U-Wear prototypes discussed in Section 4. We start by evaluating the physical layer performance of the prototypes in terms of BER as a function of (i) the signal-to-noise ratio (SNR) measured at the receiver, and of (ii) FEC coding rate. Then, we show how the U-Wear MAC protocols allow network nodes to coexist while enabling users to efficiently access the sensed medical parameters. Finally, we leverage the U-Wear reconfigurable data processing to install and run three applications built using the set of primitive blocks. We evaluate the three applications in terms of processing accuracy, i.e., the gap between the obtained outputs and a ground truth.

5.1 PHY Layer Performance

Experiment Setup. The experiment setup consists of a wuMote communicating bidirectionally with a wuMaster in two different scenarios, line-of-sight (LOS) and near-line-of-sight (nLOS). In the LOS scenario the two devices are aligned, 50 cm apart, without obstacles in between, so as to minimize reflections and scattering. In the nLOS scenario, we locate the wuMotes along the body of a user, on the chest and on the right leg, as shown in Fig. 9. The wuMaster, i.e., the smartphone, is held in the user's right hand. Under this setup, objects within the propagation area cause reflections and scattering that introduce ISI and degrade the communication performance. In Fig. 10, we show the uplink CIRs of

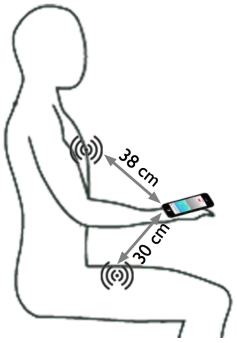

Figure 9: Near-line-of-sight (nLOS) experimental setup.

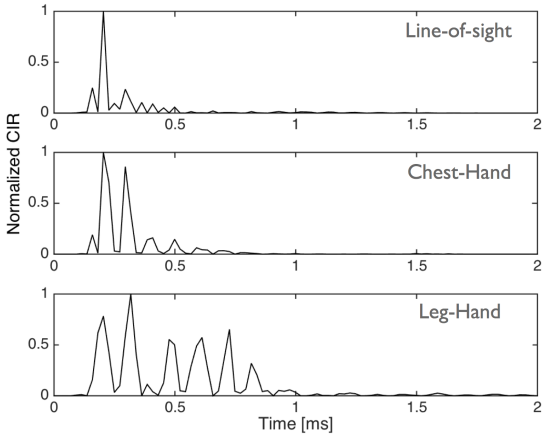

Figure 10: Ultrasonic in-air CIR for LOS (top), chest-hand nLOS (center) and leg-hand nLOS (bottom).

the three scenarios discussed above. We observe that, in the LOS scenario, the CIR contains a single dominant component. In the nLOS scenario, because of the multipath there are multiple components that contribute to ISI distortion at the receiver. In particular, in the chest-hand setup, the CIR clearly presents a second path, most likely because of a reflection from the user's hand, while in the leg-hand setup we can count up to 6 paths, most likely caused by multiple reflections from the user's trunk and hand. The coherence bandwidth in these three scenarios is approximately 21 kHz, 14 kHz and 6 kHz, respectively.

For each BER measurement we transmit up to 600 packets of 32 bytes, i.e., approximately 256 kilobits, containing pseudorandom-generated raw data. The experiment was performed indoor with a temperature of about 21° C and relative humidity around 30%. We configure the physical layer such that each GMSK symbol is represented by 16 samples. The sampling rate is set to 44.1 kHz as required by the audio hardware in use. Based on this, the raw physical layer data rate, obtained as the ratio between sample rate and sample per symbol, is approximately 2.76 kbit/s. We fix the GMSK BT product to 0.7, which represents a good tradeoff between ISI distortion and spectral efficiency. The resulting signal bandwidth is about 2 kHz, which is lower than the coherence bandwidth of the three experimental setups, thus complying with the definition of narrowband transmission scheme. The central frequency is set to 18 kHz, which,

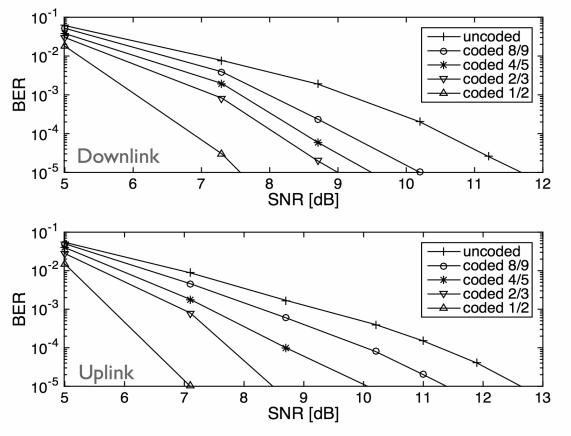

Figure 11: BER of the downlink (top) and uplink (bottom) in LOS as a function of the SNR for different coding rates.

while still in the audible frequency range, represents a good tradeoff between low audibility, fair propagation efficiency, and fair acoustic generation and detection with the COTS microphones and speakers in use. Specifically, we found that 18kHz is the highest frequency, given the spectral response of microphones and speakers in use, for which we could obtain highly reliable communications, i.e., relatively low BER, in the range of distances of interest, i.e., up to $1m$. At the same time, the signal transmission is almost inaudible by the user wearing the device. Finally, a 64-bit PN-sequence is used as preamble for synchronization and channel estimation.

BER Performance in LOS. In Fig. 11 (top), we show BER results for the downlink, i.e., from the wuMaster to the wuMote, and we compare the performance of an uncoded transmission scheme to four coded transmission schemes with coding rates in {8/9, 4/5, 2/3, 1/2}. The information rate for the five transmission schemes ranges from 2.76 kbit/s for the uncoded transmissions to 2.45 kbit/s for coding rate 8/9, 2.20 kbit/s for coding rate 4/5, 1.84 kbit/s for coding rate 2/4, and 1.38 kbit/s for coding rate 1/2. Figure 11 (bottom) shows the same comparison for the uplink, i.e., from the wuMote to the wuMaster. The SNR is calculated at the receiver as the ratio between the received average signal power and the average noise power measured after amplification and high-pass filtering. We vary the measured SNR by reducing the signal power driving the transmitter speaker. In the downlink, we do so by reducing the volume of the smartphone speaker, while in the uplink, we reduce the signal full-scale at the input of the amplifier. The maximum power is selected such that the transmitted sound results inaudible to people in proximity of the transmitter.

From Fig. 11, we observe that the BER is a decreasing function of the SNR, and that the FEC scheme mitigates the channel distortion by recovering part of the channel errors. At 5 dB SNR the BER is too high for the FEC to have an impact on the communication performance. Over 5 dB SNR, higher coding rate transmissions have clearly better mitigation performances, thus lower BER.

By measuring the power at the output of the wuMote amplifier, we see how our prototypes achieve 2.76 kbit/s on an uncoded uplink transmission, with a 10^{-5} BER, using a transmission power of 20mW, i.e., 13dB SNR at the receiver. We can lower the transmission power by compensating with

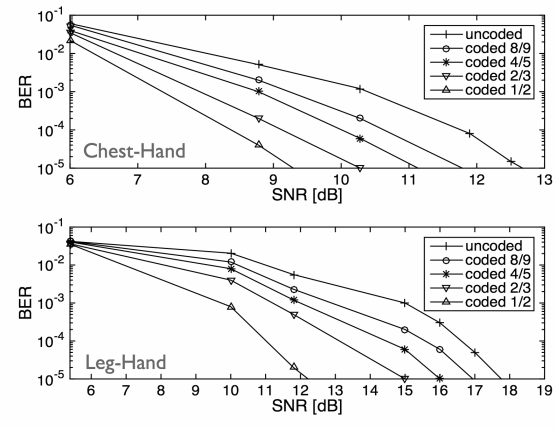

Figure 12: BER of the chest-hand (top) and of the leg-hand uplinks as a function of the SNR for different coding rates.

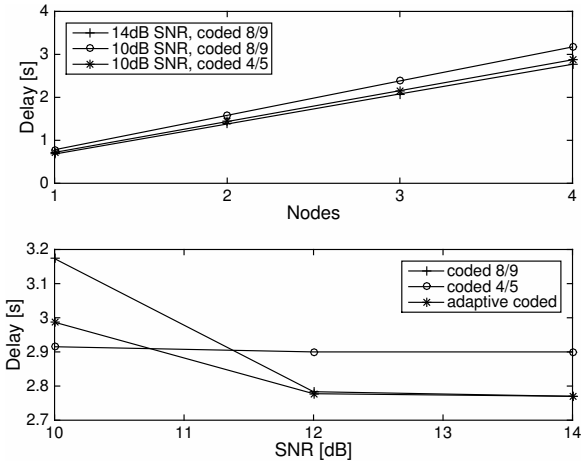

Figure 13: Polling data delivery delay as a function of number of nodes for different level of SNR and coding rates (top), and as a function of the SNR for non-adaptive and adaptive scenarios.

lower FEC coding rate, thus reducing the information rate. For example, in the current implementation, for a transmission power of 10 mW, i.e., 7 dB SNR, our prototypes achieve 1.38 kbit/s with a 10^{-5} BER using a coding rate of 1/2. Finally, by proposing for the first time a GMSK scheme for the near-ultrasonic frequencies our prototypes achieve relatively high data rates when compared to previously proposed near-ultrasonic systems; more importantly, it ensures virtually inaudible click-free transmission because of the GMSK phase-continuity as discussed in Section 3.1.1.

BER Performance in nLOS. Figure 11 shows the BER performance of uplink transmissions in nLOS scenario chest-hand setup (top) and leg-hand setup (bottom). We observe that, while the curves follow the same pattern as in the LOS scenario, the corresponding BER levels are higher because of the worse channel conditions. The BER in the chest-hand scenario is slightly higher than the LOS one, i.e., about 1 dB more of SNR is required for the same BER. Differently, in the leg-hand scenario we need an increase of 4dB SNR to achieve the same BER performance of the LOS scenario. In the chest-hand uplink, our prototypes achieve 2.76 kbit/s with a 10^{-5} BER using a transmission power of 45 mW, i.e., about 13 dB SNR at the receiver, while the same BER is obtained with 45 mW transmission power, i.e., approximately 9 dB SNR at the receiver, halving the data rate through FEC coding. In the leg-hand uplink, we obtain 10^{-5} BER with a transmission power of 250 mW, i.e., about 18 dB SNR at the receiver, for uncoded transmission at 2.76 kbit/s and, and 130 mW of transmission power, i.e., approximately 12 dB SNR at the receiver, for coded transmission at 1.78 kbit/s.

These results show how multipath effect and higher attenuation caused by the user's clothing require higher power transmission as compared to the LOS scenario. Even though ultrasonic signals are further attenuated by solid materials, they can still be used to communicate over short distances through clothing. In general, the transmission power can be reduced by using speakers and microphones with wider flat bandwidth or custom-made optimized ultrasonic transducers. In fact, a significant portion of the transmission power is lost during the electro-acoustic conversion in the COTS speaker and microphone in use, which are not designed to operate efficiently at near-ultrasonic frequencies.

5.2 MAC Layer Performance

In this Section, we evaluate the performance of the MAC layer protocols implemented on the prototypes, i.e., polling and ALOHA, in terms of data delivery delay as a function of the number of nodes in the network.

Experiment Setup. We set up a M/S configuration where devices lay in nLOS on a 2-dimensional surface, and each wuMote is positioned 40 cm apart from the wuMaster. The experiment consists of collecting data at the wuMaster from up to four wuMotes using polling or ALHOA MAC protocols. We consider six different parameters than can be fetched, and we distribute these among four wuMotes.

Adaptive Polling. Using the polling protocol, the wuMaster fetches data from one node a time. The wuMotes are addressed through physical addresses, e.g., node ID. The PHY layer adaptation allows to reactively adapt the FEC coding rate based on the packet drop rate experienced at the wuMaster to minimize the number of consecutive retransmissions. Specifically, every time the wuMaster retransmits a fetching packet, a lower coding rate is used from the set {8/9, 4/5, 2/3, 1/2}. We fix the maximum number of retransmissions for each fetch command to four. We evaluate the protocol in terms of data delivery delay, which we define as the time between the instant when the first fetching packet is transmitted by the wuMaster and the instant when the last data packet is correctly received at the wuMaster. In Fig. 13 (top), we show the polling data delivery delay as a function of the number of nodes in the network for two levels of SNR measured at the wuMaster, i.e., 10 dB and 14 dB, and two coding rates, i.e., 8/9 and 4/5. As expected, since each node in average is granted the same time to transmit, we observe that the delivery delay increases linearly with the number of nodes in the network. Moreover, since retransmissions are only caused by the channel conditions, i.e., there are no collision among different users, the delivery delay decreases by increasing the SNR or the coding rate. Figure 13 (bottom) shows the delivery delay as a function of the SNR for two fixed coding rates, i.e., 8/9 and 4/5, and for the adaptive scenario. We observe that at lower SNR, a coding rate of 8/9 gives delivery delays higher than a coding rate of 4/5 because of the frequent retransmissions

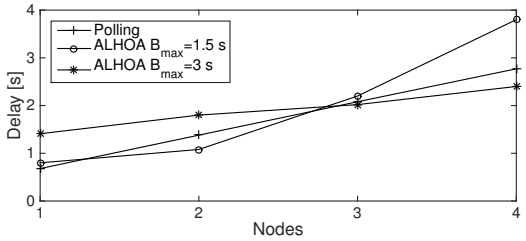

Figure 14: ALOHA data delivery delay as a function of number of nodes for different B_{max}, compared to polling.

due to higher BER at the PHY layer. On the contrary, at higher SNR a coding rate of 4/5 introduces more overhead than needed, giving higher delivery delays than coding rate 8/9. As expected, the adaptive scenario results in delivery delays in between the two fixed coding rate scenarios.

ALOHA. With ALOHA, we use the content-centric addressing scheme discussed in Section 4.1.2. Hence, the wuMaster broadcasts a request message to fetch data from multiple wuMotes. The wuMotes transmit the requested data, if available, by accessing the channel randomly. Finally, we select the backoff time between transmissions from 0 to a maximum backoff B_{max}, and we vary it during our experiments, while fixing the SNR to 14 dB and FEC coding rate to 8/9. Figure 14 shows the data delivery delay as a function of the number of nodes in the network for two different values of B_{max}, i.e., 1.5 s and 3 s. We compare the results with the data delivery delay experienced by the polling protocol for 14 dB SNR and 8/9 coding rate. When the number of nodes in the network is lower than three, we observe that $B_{max} = 1.5$ s gives lower delay than $B_{max} = 3$ s. Here, a higher B_{max} increases the probability of selecting a higher backoff time, leading to channel underutilization. On the other hand, for number of nodes higher and equal to three, $B_{max} = 1.5$ s gives high probability of collisions, thus higher delay due to retransmissions.

5.3 Data Processing Performance

To test the effectiveness of the application layer reconfiguration functionalities, we evaluate the data processing accuracy in terms of displacement between the obtained outputs, i.e., what the application reads on a given sensor trace, and the expected ones, i.e., what the application should read on that given sensor trace. We consider three applications, two running on ECG-equipped sensor nodes, i.e., heart rate monitor, ECG *RR interval* monitor, and one running on accelerometer-equipped sensor nodes, i.e., footstep counter.

ECG Processing. To show a fair comparison, we feed the application-layer processing with reference raw sensor data that are externally recorded and then loaded in the wuMote memory. For the ECG-based applications, we use traces from the MIT-BIH Normal Sinus Rhythm Database [56], which collects ECG recording from patients that were found to have no significant arrhythmias. The MIT-BIH database also specifies heart rate and RR interval of the traces. The traces are sampled at 128 Hz. We extract, and load to the wuMote, 10 one-minute long traces from the MIT-BIH recording. In Table 1, we show the heart rate estimation of the wuMote using the *R method*, second column, and *RR method*, third column, discussed in Section 4.1.2, and we compare these with the heart rate reference provided by the MIT-BIH database, fourth column. The first column shows the database trace ID. We observe that both *R method* and *RR method* give a good estimate of the reference heart rate, offering an average accuracy of 96.1% and 98.7%, respectively.

In Table 2, we show the *RR interval* mean μ and standard deviation σ estimated by the wuMote, second and fourth columns, and we compare these with the *RR interval* reference statistics provided by the MIT-BIH database, third and fifth columns. We observe that the wuMote accurately estimates the *RR interval* mean, i.e., around 99.6% of accuracy. For the standard deviation σ we obtain lower accuracy, i.e., 83.6%, for two reasons, (i) the relatively low sampling rate gives a sensibility of 8 ms, which can affect the measurement of small quantities such as the standard deviation, (ii) failures in the peak finding algorithm also affect the measurement. Higher sampling rate and outlier detection techniques could be used to further enhance the standard deviation measurement.

Table 1:
Results for heart-rate (HR) with R and RR method.

Trace	6s R [bpm]	6s RR [bpm]	Ref HR [bpm]
16265	100	97	96
16272	60	62	62
16273	100	97	95
16420	90	94	95
16483	100	97	97
16539	80	80	79
16773	70	74	75
16786	70	72	71
16795	70	67	65
17052	70	68	69

Table 2:
Results for RR interval mean μ and std. deviation σ.

Trace	μ [s]	σ [s]	Ref. μ [s]	Ref. σ [s]
16265	0.62	0.62	0.016	0.019
16272	0.96	0.96	0.109	0.115
16273	0.64	0.64	0.023	0.049
16420	0.63	0.63	0.015	0.018
16483	0.62	0.62	0.015	0.012
16539	0.74	0.75	0.062	0.054
16773	0.79	0.79	0.056	0.058
16786	0.85	0.84	0.046	0.036
16795	0.91	0.92	0.070	0.069
17052	0.85	0.85	0.047	0.047

Accelerometer Processing. We record ten 3-dimensional accelerometer traces with a sample rate of 60 Hz using Sensor Log [57], an iOS app that allows to read sensor data from the device, and export them in character-separated values (CSV) format. Sensor Log also provides information about the number of footsteps counted by the iOS device. We use this as a reference to evaluate the accuracy of the footstep counter application in the wuMote. In Table 3, we show the footstep count estimated by the wuMote, second column, and we compare this with the footstep estimate of the iOS device, third column, and real footstep number counted by the user while performing the steps, fourth column. The first column shows the trace name, where we list 3 walking traces, 3 running traces and 3 stair climbing traces, i.e.,

downward, upwards and down/upwards. We observe that, in average, the wuMote counts footsteps with the same accuracy of the iOS device, i.e., approximately 94% with respect to the number of steps counted by the user.

Table 3:
Evaluation results for footstep counter.

Trace	wuMote	iOS	Real
walk_0	44	49	46
walk_1	39	39	40
walk_2	48	48	50
run_0	32	33	34
run_1	37	42	40
run_2	32	33	32
climb_up	19	19	18
climb_down	17	18	18
climb_do_up	34	34	39

6. RELATED WORK

The idea of using ultrasonic waves for in-air communications has been proposed in the past, and several studies have successfully proven the viability of using ultrasonic waves as an alternative to RF waves for short-range and medium-range in-air communications. In [58], the authors discuss the possibility of using sounds for in-air communication purposes, and explore several different physical-layer approaches that can be used in a ubiquitous computing environment. In [59, 60], the authors studied the performance of ultrasonic in-air transmissions over directional short-range links of a few meters using custom-made transducers. The proposed solutions achieve data rates in the order of 100 kbit/s, on relatively large bandwidths, i.e., 80 kHz and 56 kHz, using QPSK and on-off-keying (OOK). In [61], medium-range communications, i.e., up to 20m, are achieved with data rates up to 100 bit/s, on a 4 kHz FSK-modulated bandwidth. While U-Wear is based on the same idea of using ultrasonic waves for in-air communications, the framework goes well beyond a simple point-to-point transmission scheme, such as those presented in the above works. In this paper, to the best of our knowledge, we propose for the first time the use of ultrasonic waves for interconnecting wearable devices in a network, and we present the first networking framework based on ultrasonic communications that offers multi-layer functionalities spanning the PHY, data link, network and application layer.

In the last few years, researchers have proposed to use acoustic waves in the audible and near-ultrasonic frequency ranges to enable in-air communications between devices equipped with COTS microphones and speakers. In [62, 63], the authors use audible sound around 8 kHz for near-field-communications between smartphones, using OFDM and FSK schemes, and achieving data rates in the order of a few kbit/s. In [64, 30], the authors propose in-air communication systems that operate in the near-ultrasonic frequency range, and achieve low data rates, i.e., up to 20 bit/s, over medium-range directional links, i.e., up to 20 m. While U-Wear can certainly operate in the near-ultrasonic frequency range to enable communications with commercial devices such as smartphones and laptops, among others, it has been primarily designed to provide connectivity between wearable devices at higher ultrasonic frequency ranges. In this paper, we implement near-ultrasonic communications to demonstrate interoperability of the wuMote with commercial smartphones, and also because of the availability of inexpensive COTS audio speakers and microphones that can be used as a proxy for air-coupled ultrasonic transducers. With this idea in mind, we proposed for the first time the use of a narrowband GMSK modulation for the near-ultrasonic frequency range, which enables relatively high data rates when compared to other existing works, i.e., up to 2.76 kbit/s on a 2 kHz bandwidth around 18 kHz, and ensures virtually inaudible click-free transmission because of the phase-continuity of the GMSK waveform. Higher data rates can be achieved by using microphones and speakers with wider bandwidth, or by developing custom-made air-coupled transducers. For example, by using OFDM signaling schemes with high-order modulations, e.g., 16-PSK or 32-QAM, on a 24 kHz bandwidth centered around 100 kHz, we could possibly achieve data rates in the order of hundreds of kbit/s.

Finally, other multilayer solutions have been specifically designed to network wearable devices. ANT+ [65], for example, is a wireless sensor network protocol operating in the 2.4 GHz ISM band that offers low-power and easy-to-use functionalities to network sensing devices. Although ANT+ and U-Wear share the same goals, i.e., networking wearable devices, the two solutions have significant differences. First, U-Wear is not just a networking solution as ANT+. U-Wear is a networking framework that can be used to build several networking solutions through software-defined functionalities. Secondly, U-Wear operates using ultrasonic waves, while ANT+ uses RF waves and offers a predefined PHY layer upon RF carriers in the 2.4 GHz ISM. Finally, ANT+ requires hardware add-ons to connect with commercial devices such as tablets or smartphones.

7. CONCLUSIONS

In this paper, we presented U-Wear, the first networking framework for wearable medical devices based on ultrasonic communications. U-Wear consists of a set of cross-layer functionalities to network ultrasonic wearable devices that offer real-time reconfigurability at the physical, data link, network and application layer. We designed two prototypes that implement the U-Wear framework and operate in the near-ultrasonic frequency range, using commercial-off-the-shelf (COTS) speakers and microphones. Despite the limited bandwidth, i.e., about 2 kHz, and the COTS audio hardware components not optimized for operating at high frequency, we showed how our prototypes (i) can operate at data rate up to 2.76 kbit/s with bit-error-rate (BER) lower than 10^{-5}, using a transmission power of 20 mW; (ii) enable multiple nodes to share the same medium offering tradeoffs between packet delivery delay and packet drop rate; (iii) can implement reconfigurable data processing applications that can extract medical parameter from sensor traces with high accuracy. U-Wear can offer higher performance through specialized hardware components. Future smart devices, e.g., smartphones and laptops, with wider-bandwidth speakers and microphones (e.g., flat over a bandwidth of 100kHz) will enable higher data rates and lower energy consumption. Through custom-made ultrasonic air-coupled transducers that operate at higher frequency ranges and larger bandwidth, wuMotes may be able to achieve data rates in the order of hundreds of kbit/s with lower energy consumptions.

8. REFERENCES

[1] G-Z Yang. *Body Sensor Networks*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[2] J.J. Oresko, Zhanpeng Jin, Jun Cheng, Shimeng Huang, Yuwen Sun, H. Duschl, and A.C. Cheng. A wearable smartphone-based platform for real-time cardiovascular disease detection via electrocardiogram processing. *IEEE Transactions on Information Technology in Biomedicine*, 14(3):734–740, May 2010.

[3] N. Roxhed, B. Samel, L. Nordquist, P. Griss, and G. Stemme. Painless drug delivery through microneedle-based transdermal patches featuring active infusion. *IEEE Transactions on Biomedical Engineering*, 55(3):1063–1071, March 2008.

[4] Heller A. Drug-delivering integrated therapeutic systems. In *Proc. of the Intl. Workshop on Wearable and Implantable Body Sensor Networks (BSN)*, London, UK, 2005.

[5] S. Patel, R. Hughes, T. Hester, J. Stein, M. Akay, J. Dy, and P. Bonato. Tracking motor recovery in stroke survivors undergoing rehabilitation using wearable technology. In *Prof of the IEEE Intl. Conf. of Engineering in Medicine and Biology Society (EMBC)*, Buenos Aires, Argentina, Aug 2010.

[6] A. Salarian, H. Russmann, F.J.G. Vingerhoets, C. Dehollaini, Y. Blanc, P.R. Burkhard, and K. Aminian. Gait assessment in parkinson's disease: toward an ambulatory system for long-term monitoring. *IEEE Transactions on Biomedical Engineering*, 51(8):1434–1443, Aug 2004.

[7] Maurice M. Ohayon and Colin M. Shapiro. Sleep disturbances and psychiatric disorders associated with posttraumatic stress disorder in the general population. *Comprehensive Psychiatry*, 41(6):469 – 478, 2000.

[8] Fitbit fitness trackers, `https://www.fitbit.com/`.

[9] Google Glass, `https://www.google.com/glass/`.

[10] Apple iWatch, `https://www.apple.com/watch/`.

[11] Hexoskin, wearable body metrics, `http://www.hexoskin.com/`.

[12] APDM, movement monitoring solutions, `http://www.apdm.com/`.

[13] Zephyr, measure life...anywhere! , `http://zephyranywhere.com/`.

[14] U.S. Food Drug Administration (FDA), Wireless Medical Devices, `http://goo.gl/8k3qZJ`.

[15] WHO/International Agency for Research on Cancer. IARC classifies radio frequency electromagnetic fields as possibly carcinogenic to humans. May 2011.

[16] K.Y. Yazdandoost and R. Kohno. UWB antenna for wireless body area network. In *Asia-Pacific Microwave Conference (APMC)*, Yokohama, Japan, 2006.

[17] T. Melodia, H. Kulhandjian, L. Kuo, and E. Demirors. Advances in Underwater Acoustic Networking. In S. Basagni, M. Conti, S. Giordano, and I. Stojmenovic, editors, *Mobile Ad Hoc Networking: Cutting Edge Directions*, pages 804–852. John Wiley and Sons, Inc., Hoboken, NJ, second edition edition, 2013.

[18] T. Dahl, J. L. Ealo, Hans J. Bang, Sverre Holm, and Pierre Khuri-Yakub. Applications of airborne ultrasound in human-computer interaction. *Ultrasonics*, 54(7):1912 – 1921, 2014.

[19] P. Lazik and A. Rowe. Indoor pseudo-ranging of mobile devices using ultrasonic chirps. In *Proc. of the ACM Conf. on Embedded Network Sensor Systems (SENSYS)*, 2012.

[20] G. Oberholzer, P. Sommer, and R. Wattenhofer. Spiderbat: Augmenting wireless sensor networks with distance and angle information. In *Proc. of Intl. Conf. on Information Processing in Sensor Networks (IPSN)*, Apr. 2011.

[21] R.J. Przybyla, Hao-Yen Tang, S.E. Shelton, D.A Horsley, and B.E. Boser. 3D ultrasonic gesture recognition. In *Proc of IEEE Intl. Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, Feb. 2014.

[22] F.L. Thurstone, H.E. Melton. Biomedical Ultrasonics. *IEEE Transaction on Industrial Electronics and Control Instrumentation*, 17(2), April 1970.

[23] G.E. Santagati, T. Melodia, L. Galluccio, and S. Palazzo. Ultrasonic networking for e-health applications. *IEEE Wireless Communications*, 20(4):74–81, 2013.

[24] G.E. Santagati, T. Melodia, L. Galluccio, and S. Palazzo. Medium access control and rate adaptation for ultrasonic intra-body sensor networks. *IEEE/ACM Transactions on Networking*, 2015.

[25] G. E. Santagati and T. Melodia. Sonar Inside Your Body: Prototyping Ultrasonic Intra-body Sensor Networks. In *Proc. of IEEE Conf. on Computer Communications (INFOCOM)*, Toronto, Canada, April 2014.

[26] Z. Guan, G. E. Santagati, and T. Melodia. Ultrasonic Intra-body Networking: Interference Modeling, Stochastic Channel Access and Rate Control. In *Proc. of IEEE Conference on Computer Communications (INFOCOM)*, Hong Kong S.A.R., PRC, April 2015.

[27] K. Kirk Shung. *Diagnostic ultrasound: imaging and blood flow measurements*. CRC Press, 2006.

[28] Yong Zhu, S. O R Moheimani, and M.R. Yuce. A 2-DOF MEMS ultrasonic energy harvester. *IEEE Sensors Journal*, 11(1):155–161, Jan 2011.

[29] M. Scherhaufl, R. Pfeil, M. Pichler, and A. Berger. A novel ultrasonic indoor localization system with simultaneous estimation of position and velocity. In *IEEE Topical Conference on Wireless Sensors and Sensor Networks (WiSNet)*, Santa Clara, CA, USA, Jan 2012.

[30] H. Lee, T. H. Kim, J. W. Choi, and S. Choi. Chirp Signal-Based Aerial Acoustic Communication for Smart Devices. In *Proc. of IEEE Conf. on Computer Communications (INFOCOM)*, Hong Kong S.A.R., PRC, April 2015.

[31] L. B. Evans, H. E. Bass, and L. C. Sutherland. Atmospheric absorption of sound: Theoretical predictions. *The Journal of the Acoustical Society of America*, 51(5B):1565–1575, 1972.

[32] Iso. ISO 9613-1, Acoustics - Attenuation of sound during propagation outdoors - Part 1: Calculation of the absorption of sound by the atmosphere, Dec 2010.

[33] M. Pappalardo, G. Caliano, A. S. Savoia, and A. Caronti. Micromachined ultrasonic transducers. In *Piezoelectric and Acoustic Materials for Transducer Applications*, pages 453–478. Springer US, 2008.

[34] K. Murota and K. Hirade. GMSK modulation for digital mobile radio telephony. *IEEE Transactions on Communications*, 29(7):1044–1050, Jul 1981.

[35] E. Demirors, G. Sklivanitis, G. E. Santagati, T. Melodia, and S. N. Batalama. Design of a software-defined underwater acoustic modem with real-time physical layer adaptation capabilities. In *Proc. of ACM Intl. Conf. on Underwater Networks & Systems (WUWNet)*, Rome, Italy, November 2014.

[36] F. Tufvesson, O. Edfors, and M. Faulkner. Time and frequency synchronization for OFDM using PN-sequence preambles. In *Proc. of the IEEE Vehicular Technology Conference*, Houston, TX, USA, 1999.

[37] M. Skolnik. *Radar Handbook, Third Edition.* Electronics electrical engineering. McGraw-Hill Education, 2008.

[38] Robert J. Polgember and E.M. Mitchell. Impulse response determination by cross correlation. *IEEE Transactions on Aerospace and Electronic Systems*, AES-6(1):91–97, Jan 1970.

[39] J. Proakis and M. Salehi. *Digital Communications.* McGraw-Hill higher education. McGraw-Hill Education, 2007.

[40] Norman Abramson. The ALOHA system: Another alternative for computer communications. In *Proc. of the Fall Joint Computer Conference (FJCC)*, AFIPS, Houston, Texas, Nov. 1970.

[41] L. Kleinrock and F.A. Tobagi. Packet switching in radio channels: Part i–carrier sense multiple-access modes and their throughput-delay characteristics. *IEEE Transactions on Communications*, 23(12):1400–1416, Dec. 1975.

[42] Y. Sun and T. Melodia. The Internet Underwater: An IP-compatible Protocol Stack for Commercial Undersea Modems. In *Proc. of ACM Intl. Conf. on UnderWater Networks and Systems (WUWNet)*, Kaohsiung, Taiwan, November 2013.

[43] Vessela Krasteva and Irena Jekova. QRS template matching for recognition of ventricular ectopic beats. *Annals of Biomedical Engineering*, 35(12):2065–2076, 2007.

[44] Teensy USB Development Board, `https://www.pjrc.com/teensy/`.

[45] Teensy Audio Library, `https://www.pjrc.com/teensy/td_libs_Audio.html`.

[46] Freescale, Kinetis KL03, `http://goo.gl/ckPkl0`.

[47] Stereo 2.8W Class D Audio Amplifier - TS2012, `http://www.adafruit.com/product/1712`.

[48] MEMS Microphone Breakout - INMP401 (ADMP401), `https://www.sparkfun.com/products/9868`.

[49] ST MEMS microphones, `http://goo.gl/9Eojhq`.

[50] Dayton Audio CE28A-4R , `http://goo.gl/u5VMRR`.

[51] C.L. Chang, K.P. Lin, T.H. Tao, T. Kao, and W.H. Chang. Validation of automated arrhythmia detection for holter ECG. In *Proc. of the IEEE Intl. Conf. of the Engineering in Medicine and Biology Society (EMBC)*, Oct 1998.

[52] Xcode 6, the complete toolset for building great apps, `https://developer.apple.com/xcode/`.

[53] vDSP Library, iOS Accelerate Framework, `http://goo.gl/6gvpFT`.

[54] Novocaine, An analgesic for high-performance audio on iOS and OSX, `https://github.com/alexbw/novocaine`.

[55] wit.ai, Natural Language for the Internet of Things, `https://wit.ai`.

[56] MIT-BIH Normal Sinus Rhythm Database, `http://physionet.org/physiobank/database/nsrdb/`.

[57] SensorLog by Bernd Thomas, `https://itunes.apple.com/us/app/sensorlog/id388014573`.

[58] C.V. Lopes and P.M.Q. Aguiar. Acoustic modems for ubiquitous computing. *IEEE Pervasive Computing, Mobile and Ubiquitous Systems*, 2(3):62–71, July 2003.

[59] Chuan Li, D.A. Hutchins, and R.J. Green. Short-range ultrasonic communications in air using quadrature modulation. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, 56(10):2060–2072, October 2009.

[60] W. Jiang and W. M.D. Wright. Ultrasonic wireless communication in air using OFDM-OOK modulation. In *Proc. of IEEE Intl. Ultrasonics Symposium (IUS)*, Chicago, IL, USA, Sept 2014.

[61] S. Holm, O.B. Hovind, S. Rostad, and R. Holm. Indoors data communications using airborne ultrasound. In *Proc of the IEEE Intl. Conf. of Acoustics, Speech, and Signal Processing (ICASSP)*, Philadelphia, PA, USA, March 2005.

[62] R. Nandakumar, K. K. Chintalapudi, V. Padmanabhan, and R. Venkatesan. Dhwani: Secure peer-to-peer acoustic NFC. In *Proc. of the ACM Conf. on SIGCOMM*, Hong Kong, China, 2013.

[63] B. Zhang, Q. Zhan, S. Chen, M. Li, K. Ren, C. Wang, and D. Ma. Priwhisper: Enabling keyless secure acoustic communication for smartphones. *IEEE Internet of Things Journal*, 1(1):33–45, Feb 2014.

[64] M. Hanspach and M. Goetz. On covert acoustical mesh networks in air. *Journal of Communications*, 8(11):758–767, 2013.

[65] ANT+ , `http://www.thisisant.com`.