The Design of a Responsive and Energy-efficient Event-triggered Wireless Sensing System

Felix Sutton, Reto Da Forno, David Gschwend, Tonio Gsell, Roman Lim, Jan Beutel, and Lothar Thiele
Computer Engineering and Networks Laboratory
ETH Zurich, Switzerland
{firstname.lastname}@tik.ee.ethz.ch

Abstract

We present the design, implementation and experimental evaluation of an efficient event-triggered wireless sensing system. We propose an event-triggered logical component model, which encapsulates adaptability, responsiveness and energy efficiency design constraints. We then design a wireless sensing system for monitoring acoustic emissions through the composition of event-triggered logical components. We present the integration of these components onto a novel dual-processor wireless sensing platform, and extensively evaluate the developed prototype with respect to responsiveness and energy efficiency using real-world acoustic emission traces.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*wireless communication*; C.3 [**Computer Systems Organization**]: Special-Purpose and Application-Based System—*real-time and embedded systems*

General Terms

Design, Experimentation, Performance

Keywords

Event-triggered wireless sensing; acoustic emission sensing; wireless sensor networks; cyber-physical systems

1 Introduction

Motivation. The detection and characterization of acoustic emissions provide important insights into the behavior of underlying physical processes that drive a wide range of application domains. Wireless sensor networks have been widely adopted for monitoring acoustic emissions with high spatial coverage and temporal resolution, and relatively low infrastructure costs. Example systems from the literature include structural monitoring [22], seismic monitoring [38, 10], and sniper localization [34], whereby each of these wireless sensing systems demand the following properties:

- ***Adaptability:*** The ability to adapt the system configuration at run-time in response to changes in the environment. For example, the adaption of sensor sensitivity, characterization parameters or communication bandwidth in response to a detected acoustic emission.
- ***Responsiveness:*** The ability to rapidly detect acoustic emissions, quickly characterize their specific features, and rapidly communicate the associated data through a multi-hop network for post-processing.
- ***Energy Efficiency:*** In order to achieve long-term operation, the system must minimize the energy consumed by detection, characterization and communication of acoustic emissions.

Challenges. The realization of efficient wireless acoustic sensing systems is difficult in practice, due to the conflicting nature of the aforementioned design requirements. For example, in order to rapidly detect acoustic emissions, one must continuously monitor the acoustic sensor, thereby consuming precious energy reserves during periods where there are no acoustic emissions of interest. Adapting the system to changes in the environment at run-time requires efficient network-wide coordination, which again leads to increased energy consumption through additional wireless communication activity. Finally, energy efficiency may be achieved through duty-cycling or deactivation of individual system components, however the time to wake-up or turn-on components may adversely impact the responsiveness and adaptability of the system.

Proposed Approach. In order to effectively manage these conflicting design goals, we partition the wireless acoustic sensing system into a pipeline of logical event-triggered components. Each event-triggered component is characterized by design-time and run-time parameters, and adheres to an event-triggered interface specification. The design and implementation of each logical component follows the guideline: *sleep whenever possible, wake-up fast, and operate efficiently*. Components are then integrated onto a physical platform architecture, whilst preserving the underlying design constraints imposed on each individual component.

Contributions. We make the following contributions:

- We present a logical event-triggered component model for the construction of adaptable wireless sensing sys-

International Conference on Embedded Wireless Systems and Networks (EWSN) 2017
20–22 February, Uppsala, Sweden
© 2017 Copyright is held by the authors.
Permission is granted for indexing in the ACM Digital Library
ISBN: 978-0-9949886-1-4

tems, subject to responsiveness and energy efficiency design constraints.

- We exemplify our approach by designing and implementing an always-on ultra-low power acoustic sensor interface dissipating only 6.2 μW for the detection of acoustic emissions, an event-triggered characterization pipeline, and an event-based wireless protocol for the multi-hop dissemination of acoustic emissions under contention with an average latency as low as 113.2 ms.
- We demonstrate the integration of logical event-triggered components onto a novel dual-processor platform architecture and experimentally evaluate its performance.

The paper is structured as follows: Sec. 2 proposes an event-triggered component model, which is then leveraged in Sec. 3 to design and implement an efficient wireless acoustic sensing system. The developed prototype is extensively evaluated in Sec. 4 in the context of a realistic application scenario. We discuss related work in Sec. 5, and present a summary of conclusions in Sec. 6.

2 Event-triggered System Design

In this section, we present a design process for adaptable, responsive and energy efficient sensing systems that are suitable for certain classes of event-triggered wireless sensing applications. We next detail each step in the design process, before exemplifying its use through the realization of a wireless acoustic emission sensing system in Sec. 3.

(1) Partitioning. We begin by partitioning an event-triggered system into a pipeline of logical components. Each component represents a functional building block of the system and adheres to an event-triggered interface, whereby a component is triggered by an input event and produces an output event in order to trigger the next component in the pipeline. To give a concrete example, a wireless acoustic emission sensing system, which will be extensively detailed in Sec. 3, must first *(i)* detect acoustic emission, then *(ii)* characterize the acoustic event, before *(iii)* disseminating the event data through a multi-hop wireless network. It follows that such a system may be partitioned into a directed graph of three logical event-triggered components representing the aforementioned functionality.

(2) Modeling. We then represent each logical component with an event-triggered model, as illustrated in Fig. 1. The component model is specified by *(i)* its input and output interface consisting of an *event* and a *data* stream, *(ii)* its run-time adaptability represented by an *event filtering* random process, and *(iii)* its functional behavior represented by an *event processing* finite state machine. We next use an analytical framework to derive a components responsiveness and average power dissipation given the characteristics of its input event stream.

The *event filtering* element observes a stream of events with an average arrival rate α. We denote the arrival time of an event i by t_i, where $t \in [0,\infty)$, and $i \in \mathbb{Z}$ represents the order of event arrival. We define the continuous random variable $\tau_d = t_{i+d} - t_i$, representing the time between events separated by arrival index $d \geq 1$. We note that when $d = 1$, the variable τ_1 represents the inter-arrival time of sequential events. Assuming event arrivals are independent and identically distributed, and using known properties of sums of continuous random variables [15], the probability density function of τ_d is given by the d-fold continuous convolution of $f_{\tau_1}(\tau)$, as given in (1). It then follows that the average arrival rate of events is given by (2).

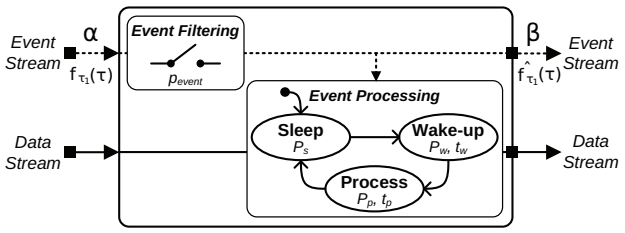

Figure 1. Logical event-triggered component model.

$$f_{\tau_d}(\tau) = (f_{\tau_k} * f_{\tau_{d-k}})(\tau) \quad \text{for } 1 \leq k \leq d-1 \tag{1}$$

$$\alpha = \frac{1}{\int_0^\infty \tau \cdot f_{\tau_1}(\tau) d\tau} \tag{2}$$

The run-time adaptability of a component is represented by the random sampling of the input event stream. In the context of acoustic emission sensing, an example of such event filtering is the configuration of the detection threshold. If the detection threshold is configured very low, acoustic emissions will be detected with a probability close to unity. However, if the detection threshold is configured very high, acoustic emissions will only be detected with a probability significantly less than one. We model this random sampling as independent Bernoulli trials, where an event is processed with probability p_{event}, and is neglected with probability $1 - p_{event}$. The probability p_d of an event having an arrival index difference d is then given by the product of the probability of a processed event and the probability of $d-1$ neglected events, as expressed in (3).

$$p_d = p_{event} \cdot (1 - p_{event})^{d-1} \tag{3}$$

It then follows that the probability density function of the inter-arrival time of events after filtering $\hat{f}_{\tau_1}(\tau)$ is given by (4). We note that due to the exponential decay of p_d, the summation may be approximated using a finite number of index differences d. Due to the random sampling, the average rate of events produced at the output is equal to the average arrival rate scaled by the run-time parameter p_{event}, as expressed in (5).

$$\hat{f}_{\tau_1}(\tau) = \sum_{d=1}^{\infty} p_d \cdot f_{\tau_d}(\tau) \tag{4}$$

$$\beta = p_{event} \cdot \alpha \tag{5}$$

The *event processing* element specifies the functional behavior of a logical component with a finite state machine annotated with design-time parameters. A component resides in a default *sleep* state dissipating power P_s until it is triggered to execute. The arrival of an event after filtering will transition the component into a *wake-up* state, where the component awakes from sleep mode and performs necessary pre-processing tasks, taking time t_w and dissipating power P_w. The component then enters the *process* state, taking the input event and data stream and producing an appropriate

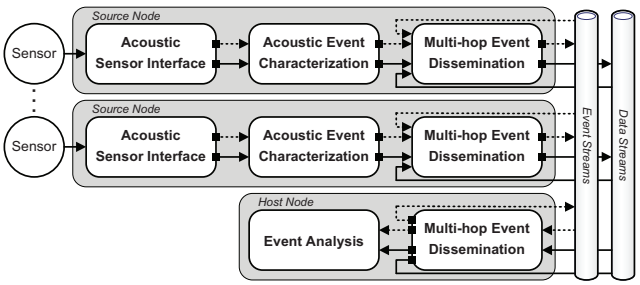

Figure 2. A wireless acoustic emission sensing system composed of logical event-triggered components.

output event and data stream, taking time t_p and dissipating power P_p. Once the component has completed processing, it immediately returns to the *sleep* state.

The responsiveness of a component is given by the time spent in wake-up and processing states, *i.e.*, $t_w + t_p$, while the energy consumption per event is determined by the sum of the energy consumed in all three states. Assuming constant power dissipation within each state, the energy consumption u per event is a linear function of inter-arrival time τ and the parameterization of the finite state machine, as given by (6).

$$u(\tau) = P_w t_w + P_p t_p + P_s (\tau - t_w - t_p) \tag{6}$$

Using known results of functions of continuous random variables [30], the probability density function of the energy consumption per event $f_u(u)$ is given by (7), which can then be used to evaluate the average power dissipation P_{avg} using (8).

$$f_u(u) = \begin{cases} \delta(u - P_w t_w - P_p t_p), & \text{if } P_s = 0. \\ \frac{1}{P_s} \hat{f}_{\tau_1}\left(\frac{u - t_w(P_w - P_s) - t_p(P_p - P_s)}{P_s}\right), & \text{if } P_s > 0. \end{cases} \tag{7}$$

$$P_{avg} = p_{event} \cdot \alpha \cdot \int_0^\infty u \cdot f_u(u) du \tag{8}$$

Therefore, given the statistical properties of the input event stream, α and $f_{\tau_1}(\tau)$, the run-time adaptability parameter p_{event}, and the design-time parameters of the finite state machine, P_s, P_w, t_w P_p, and t_p, we can determine *(i)* the properties of the output event stream, β and $\hat{f}_{\tau_1}(\tau)$, *(ii)* the responsiveness of the component by $t_w + t_p$, and *(iii)* the component's average power dissipation P_{avg}.

(3) Concretization. We now systematically design and implement each logical component in the pipeline according to the following guideline: *sleep whenever possible, wake-up fast, and operate efficiently*. This guideline not only resembles the overall responsiveness and energy efficiency design constraints, but translates directly onto the design-time parameterization of the logical event-triggered component model depicted in Fig. 1. Specifically, a component must minimize the processing time t_p and enforce a low-power sleep state during periods of inactivity, minimize the wake-up time t_w in order to transition rapidly from sleep to the process state, and finally, minimize the power dissipation P_s, P_w and P_p in order to operate efficiently. Through the application of appropriate design tools, such as design space exploration, simulation, rapid prototyping, etc., a concrete realization may be found which incorporates the components run-time adaptability, design-time parameterization, and any additional domain-specific requirements.

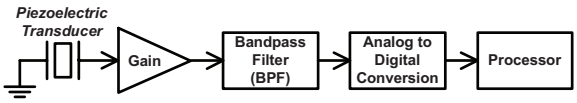

Figure 3. Continuous acoustic emission sensing.

(4) Integration. Now with a concrete realization of each logical component, we next integrate them onto a physical platform architecture whilst preserving adaptability, responsiveness and energy efficiency design constraints. In Sec. 3.4, we demonstrate this integration using a novel dual-processor platform architecture featuring *(i)* limited resource interference between components and *(ii)* support for composable construction using a formally verified interface between components.

3 Design and Implementation of a Wireless Acoustic Emission Sensing System

In this section, we demonstrate the design process introduced in Sec. 2 to construct a wireless acoustic emission sensing system. We consider a system where *source* nodes equipped with an acoustic sensor are deployed to detect, characterize and disseminate acoustic emissions to a *host* node for analysis. As illustrated in Fig. 2, we partition the system into a pipeline of logical event-triggered components, namely, *acoustic sensor interface*, *acoustic event characterization*, and *multi-hop event dissemination* components. We next detail the design and implementation of each logical component, and then demonstrate their integration onto a physical platform architecture.

3.1 Acoustic Sensor Interface

Related Work. Acoustic emissions are typically detected and characterized in commercial monitoring solutions such as [16], using a combination of continuously powered analog circuits interfaced to a processor, as illustrated in Fig. 3. The architecture consists of an acoustic sensor, *i.e.*, a piezoelectric transducer, that converts dynamic motions into an electrical signal, followed by a fixed gain amplifier, bandpass filter and an analog-to-digital converter (ADC). This continuous acoustic sensing architecture has been widely adopted in the literature using a variety of different processors, including 16-bit microcontrollers [38, 22, 1, 33, 11], 32-bit microcontrollers [2, 9], DSPs [23, 36], ASPs [32] and FPGAs [34, 37, 25, 4].

However, since this architecture does not leverage a low-power *sleep* state, significant energy is consumed during periods of inactivity, *i.e.*, where no acoustic emission of interest is observed. In order to reduce energy consumption, a *sensor-initiated wake-up* concept was proposed in [8], and further demonstrated in [20, 29, 28]. The key idea is to employ an ultra-low power analog circuit to wake-up the high power components only when an acoustic event is detected.

We extend this body of work by incorporating event-triggered wake-up *to all system components*, in conjunction with component-level run-time adaptability. As summarized in Table 1, the proposed approach achieves less than half the power dissipation P_s during sleep state compared to state-of-the-art acoustic sensor interfaces.

Application Requirements. The acoustic sensor interface component must satisfy the following requirements:

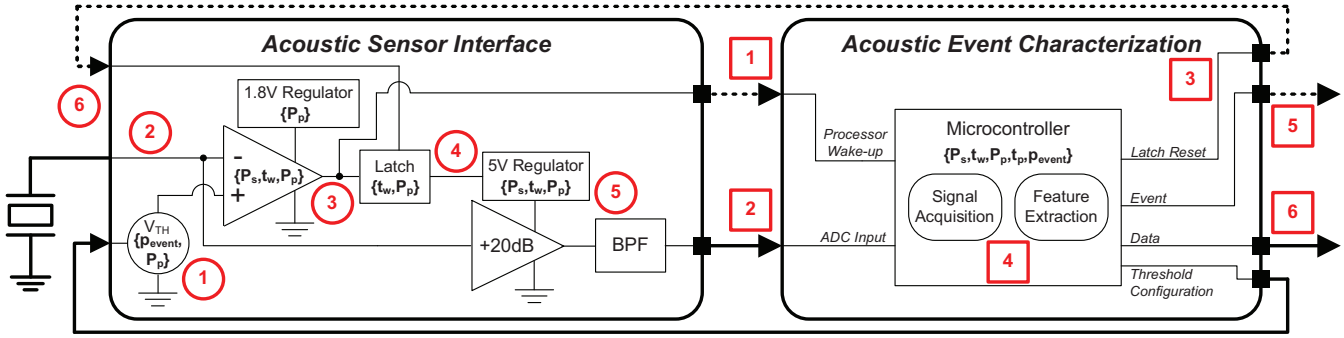

Figure 4. Block diagram of the acoustic sensor interface and acoustic event characterization components.

Table 1. Power dissipation of state-of-the-art acoustic sensor interfaces compared to the developed prototype.

System	Component	P_s
Lucid Dreaming [20]	Acoustic Sensor Interface (without LDO)	16.5 μW
CargoNet [29]	Acoustic Sensor Interface (without LDO)	3 μW
This Work	Acoustic Sensor Interface (without LDO)	1.2 μW
	Acoustic Event Characterization	2.5 μW
	Multi-hop Event Dissemination (T = 15 s)	52.8 μW

- ***Adaptable Detection.*** In order to support diverse operating conditions, the detection sensitivity of the interface must be configurable at run-time.
- ***Sensitivity.*** It is important to not only detect very large signals *i.e.*, having amplitudes of volts, but also detect very small signals, *i.e.*, having amplitudes of millivolts.
- ***Noise Resilience.*** The sensor interface must be resilient against internal and external electrical noise sources, *e.g.*, such as the noise associated with high-gain amplification, or an external power supply with a large ripple voltage.

Component Model. Fig. 4 illustrates the block diagram of the acoustic sensor interface using the proposed event-triggered logical component model, and highlights the run-time and design-time parameters that drive the realization of the component. We next describe the behavior of the component, followed by its concretization through design space exploration, circuit simulation and rapid prototyping.

The operation of the acoustic sensor interface begins with *(1)* the configuration of the detection threshold V_{TH}. According to the application requirements this can be adjusted at run-time, and is therefore modeled by the random event filtering process with probability p_{event}. The intuition is that the random filtering of events has the effect of decreasing the average output event arrival rate β, which is analogous to increasing the detection threshold. The generated threshold voltage provides a stable reference to an analog comparator *(2)*, which monitors the output of the acoustic sensor. When the sensor produces a voltage greater than the detection threshold, *(3)* the comparator generates an output event in order to trigger the next component in the pipeline, *i.e.*, acoustic event characterization. The output of the comparator *(4)* also activates a latch, which turns on the voltage regulator supplying the amplification and filtering circuitry. The acoustic signal is then amplified and filtered according to application requirements, and *(5)* the output data stream is made available to the next component. Once the acoustic event characterization is complete, *(6)* an input event resets the latch, thereby turning off the high power amplifier circuit.

3.1.1 Design Space Exploration and Simulation

Threshold Generation. Since the threshold generation is always-on, we aim to minimize the active power dissipation P_p associated with the circuitry providing the programmable variable voltage source. Rather than employing a digitally controlled variable voltage regulator, we seek a solution that drains significantly less than 1 μA quiescent current. To this end, we employ a resistive voltage divider circuit [17] as defined by two series resistors, together with a bank of four parallel resistors each gated by n-channel transistors. Using SPICE circuit simulation, we select resistor values in the kilo- and megaohm range in order to support a threshold voltage range between tens of millivolts and several hundreds of millivolts, while as listed in Table 2, only dissipating a fraction of a microwatt.

Comparator and Voltage Regulator. We perform an extensive design space exploration to find the most suitable comparator and voltage regulator. The design space exploration considers the following five metrics, which directly relate to the parameterization of the logical event-triggered component model and the stipulated application requirements:

- *Quiescent Current:* We minimize the quiescent current of comparator and voltage regulator in order to decrease the power dissipation P_s during periods of inactivity.
- *Propagation Delay:* We minimize the comparator propagation delay in order to reduce the wake-up time t_w, and thus improve responsiveness.
- *Active Current:* We minimize the active current of comparator and voltage regulator in order to decrease the power dissipation P_p.
- *Minimum Input Voltage:* In order to maximize the sensitivity of the acoustic sensor interface, we seek a comparator with a very low minimum input voltage.
- *Power Supply Rejection Ratio (PSRR):* We maximize the PSRR of both comparator and voltage regulator in order to be resilient against external noise sources.

The design space exploration is performed in two distinct phases. The first phase reduces the search space by evaluating metrics of candidate devices using the respective datasheets, before selecting the three best performing devices. The second phase determines the most suitable device through the experimental evaluation of a rapid prototype.

Assessing the datasheets of 10 commercial comparators indicated a clear trade-off between the quiescent current and propagation delay metrics, where a lower quiescent current

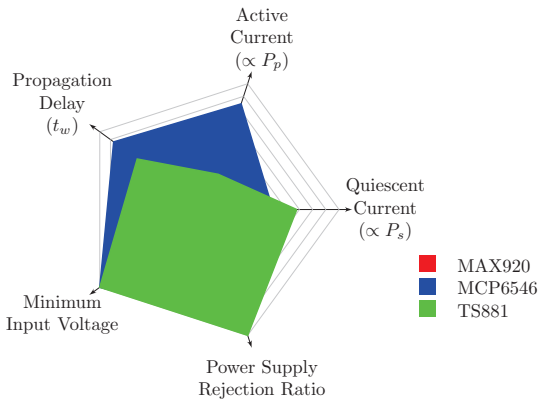

Figure 5. Visualization of the design space exploration for the MAX920, MCP6546, and TS881 comparators.

is achieved at the cost of a longer propagation delay. Due to the importance of these two metrics on the overall responsiveness and energy efficiency, we select the best three comparators, *i.e.*, MAX920, MCP6546, and TS881, based only on a balance of these two metrics.

A custom printed circuit board is produced in order to experimentally evaluate the metrics of the three comparators under realistic conditions. Fig. 5 depicts the experimental results by representing each metric on a unique axis, where the metric value has been transformed so that a higher value on the axis is more favorable. All metrics are evaluated using either a mixed signal oscilloscope or a precision multimeter, except for the PSRR, where the datasheet value is included for completeness. As represented in Fig. 5, the MCP6546 comparator performs best compared to the other two devices, and is therefore chosen to implement the acoustic event detection component. The aforementioned design space exploration is repeated to find the most suitable voltage regulator, based on 13 commercial devices. In summary, the MCP1711 1.8 V regulator is selected due to its favorable active current, quiescent current, and PSRR metrics.

Latch. A digital latch provides a stable power gating of the amplification and filtering circuitry for the duration of an acoustic event. Instead of employing a commercial single-packaged latch designed for high-frequency operation and thus exhibits a quiescent current on the order of microamps, we design a custom set-reset (SR) latch [17] using n-channel and p-channel transistors. We verified the functionality of the latch using SPICE simulation, and measured its negligible power dissipation P_p of 45 nW and fast wake-up time t_w of 5 μs using a rapid prototype.

Amplification and Filtering. The voltage regulator for the high-gain amplifier circuit is chosen using a design space exploration, similar to that performed for the comparator. In addition to quiescent current and PSRR metrics, the *enable delay* metric was also considered in order to minimize the amplifier wake-up time t_w. The design space exploration considers 10 commercial devices, with the NCP603 5.0 V regulator chosen due to its low enable delay, and favorable quiescent current and PSRR.

The LM6482 operation amplifier is selected to implement the high-gain amplification, based on the frequency response of the acoustic sensor, and the application requirements placed on the amplification gain and output noise performance. A SPICE simulation is used to select the passive components determining the gain of the amplifier and the frequency response of the passive first-order bandpass filter.

Table 2. Parameterization of the acoustic sensor interface and acoustic event characterization components.

Device	P_s	t_w	P_p	t_p
Threshold Generator	-	-	354 nW	Always-on
Comparator & Regulator	5.8 μW	11 μs	5.8 μW	2.5 ms
Latch	45 nW	5 μs	45 nW	2.5 ms
Amplifier & Regulator	22 nW	23 μs	5.6 mW	2.5 ms
Microcontroller	2.5 μW	29 μs	15 mW	4.3 ms

3.1.2 Evaluation of the Acoustic Sensor Interface

Experimental Setup. We evaluate the responsiveness and power dissipation of the acoustic sensor interface through a controlled laboratory experiment. We emulate the acoustic sensor using an arbitrary waveform generator programmed with a real-world acoustic signal, measure the analog and digital outputs using a mixed-signal oscilloscope, and measure the current drain using a precision multimeter.

Results. As illustrated in Fig. 6, once the input acoustic signal surpasses the detection threshold, the comparator output triggers the wake-up of the acoustic event characterization component. After the 5 μs wake-up delay of the latch, the voltage regulator of the amplifier is turned on. This is visible in Fig. 6 by the sudden spike on the ADC input line. Once the ADC awakes from its sleep state, as indicated by the rising edge of the signal acquisition line in Fig. 6, the acoustic signal is sampled according to application requirements. Once sufficient digital samples are collected, the latch is reset, and the acoustic signal is characterized.

The design-time parameterization of the acoustic sensor interface component is summarized in Table 2. We can conclude from these measurements that the acoustic sensor interface exhibits a power dissipation of only 6.2 μW during periods of inactivity, supports a wake-up delay of 16 μs, and dissipates 5.6 mW during processing.

3.2 Acoustic Event Characterization

The purpose of the acoustic event characterization component is to extract important features from a detected acoustic emission in order to facilitate application-specific analysis at the remote host node. There are two common types of features for acoustic emission sensing, namely, *(i)* temporal features, *e.g.*, zero-crossing rate, rise-time, energy, and *(ii)* spectral features, *e.g.*, bandwidth, spectrum centroid, pitch.

While custom hardware blocks may be used to extract low-complexity features as exemplified in [32, 21], we instead leverage a state-of-the-art microcontroller to support complex and adaptive event characterization. We propose that the acoustic signal is first converted into the digital domain by an analog-to-digital converter (ADC), before computing a set of features, possibly in conjunction with pattern classification techniques [6], according to the run-time configuration and application requirements.

Component Model. The block diagram of the acoustic event characterization component is illustrated in Fig. 4, and is annotated with the critical design-time parameters used to facilitate its design and implementation. We next describe

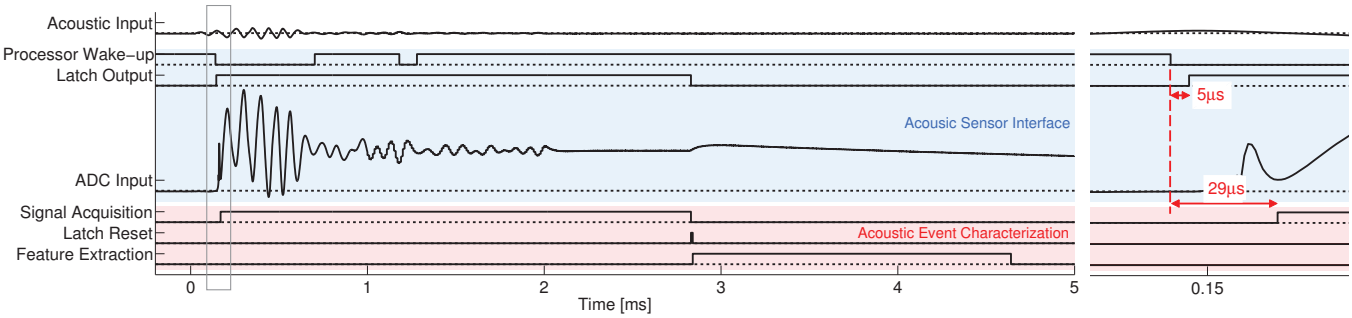

Figure 6. Timing behavior of the acoustic sensor interface and acoustic event characterization components. The figure to the right provides a magnified view of the component interaction centered around 0.15 ms into the trace.

the behavior of the component followed by its concretization using a state-of-the-art low-power microcontroller.

The acoustic event characterization component is triggered by *(1)* an input event originating from the acoustic sensor interface. This event awakes the microcontroller from a deep sleep state, and *(2)* triggers it to begin sampling the acoustic signal using its built-in ADC. We leverage a built-in ADC in order to minimize the wake-up time t_w associated with peripheral initialization. Once sufficient samples have been collected, *(3)* an output event triggers the acoustic sensor interface to turn off its high-power amplifier. The microcontroller *(4)* computes the application-specific feature set, and produces an output event *(5)* indicating to the next component, *i.e.*, multi-hop event dissemination, that characterization is complete and *(6)* a data stream is available. A second output stream is provided to configure the detection threshold of the acoustic sensor interface at run-time.

Since the event-triggered component model presented in Sec. 2 makes no assumptions on stream buffering between components, if the time to perform the event characterization is longer than the time to digitize the input acoustic signal, then depending on the inter-arrival time between events, some events may never be characterized. We model this behavior using a random event filtering process with probability p_{event}, which provides an important design-time tool for appropriately constraining the wake-up t_w and process t_p times given the input event characteristics $f_{\tau_1}(\tau)$.

Concretization. While the choice of potential commercial microcontrollers is plentiful, we narrow the search space according to the parameterization of the logical event-triggered component model. Specifically, we seek *(i)* a microcontroller with a built-in ADC module supporting the application-specific sample frequency and resolution, *(ii)* a core that exhibits a low wake-up delay t_w from deep sleep, *(iii)* a low sleep power dissipation P_s, and *(iv)* a favorable active power dissipation P_p relative to the microcontroller clock frequency. After a survey of state-of-the-art microcontrollers, we select the MSP432 from Texas Instruments, as it combines the computational resources of a 32-bit ARM Cortex-M4 processor with the energy efficiency of the widely adopted MSP430 family of microcontrollers.

3.2.1 Evaluation of Acoustic Event Characterization

Results. We use the identical experimental setup described in Sec. 3.1.2 to measure the wake-up time t_w and power dissipation P_s and P_p of the acoustic event characterization component. Fig. 6 depicts the timing behavior of signal acquisition consisting of 1000 14-bit samples at a sampling rate of approximately 400 kHz, followed by feature extraction consisting of the rise time, min/max amplitude, and signal energy. We note that using this specific implementation, the time taken for feature extraction is less than that for signal acquisition, and therefore we may assume $p_{event} = 1$ since sequential events will be successfully characterized.

As illustrated in Fig. 6, the time taken to wake-up the acoustic event characterization component is 29 μs. As summarized in Table 2, the measured power dissipation P_s during sleep state is 2.5 μW, and the power dissipation P_p during process state is 15 mW.

3.3 Multi-hop Event Dissemination

The purpose of the multi-hop event dissemination component is to rapidly and reliably deliver an event and its associated data to a host over a wireless multi-hop network. We next list the application requirements of this component, followed by a detailed analysis of how the proposed event-triggered logical component model is used to tailor a state-of-the-art wireless protocol for efficient multi-hop event dissemination.

Application Requirements. In order to analyze acoustic emissions at the host, we require the following properties:

- ***Time Synchronization.*** All source nodes must be tightly synchronized to a common time base, so to differentiate between acoustic emissions with respect to time.
- ***Reliability.*** The probability that transmitted acoustic event characteristics is successfully received at the host must be sufficiently high.
- ***On-demand Dissemination.*** A source node must be able to request network bandwidth on-demand for the dissemination of an event and its associated data.
- ***Simultaneous Events.*** The dissemination of simultaneous acoustic events must be supported, as the deployment of source nodes does not prohibit more than one source node from detecting the same acoustic emission.

Related Work. The past decade of wireless sensor network research has produced a plethora of low-power wireless communication protocols, as surveyed in [18]. In this work, we concentrate on the *synchronous* class of protocols, as they support tight time synchronization by design, and have been shown experimentally to exhibit high end-to-end reliability. For example, Orchestra [7], a synchronous slot-based channel-hopping protocol for RPL and IPv6 networks, and

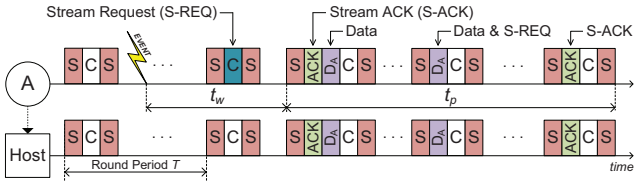

Figure 7. Example operation of the LWB, where source node *A* disseminates a packet triggered by an event.

Glossy [13], a synchronous flooding-based protocol, both support tight global *time synchronization* and demonstrate end-to-end *reliability* above 99.9% on testbed networks with up to 92 nodes.

While both protocols represent the state-of-the-art, we focus on Glossy, as its underlying time synchronization mechanism is achieved through radio-driven packet flooding using constructive interference, and therefore is insensitive to higher-layer protocol interactions such as routing. To this end, there have been several protocols proposed in the literature that build upon the Glossy flooding primitive, such as the Low-power Wireless Bus [12], Splash [3], Chaos [24], and Pando [5], where each protocol has been tailored to a specific data dissemination scenario.

In this work, we choose to tailor the Low-power Wireless Bus (LWB), although we acknowledge that alternative protocols may indeed be feasible. We next present an overview of the LWB, followed by a detailed description of how we tailor the protocol to support the remaining application requirements of *on-demand dissemination* and *simultaneous events*.

3.3.1 Overview of the Low-power Wireless Bus

The open-source Low-power Wireless Bus (LWB) [12] transforms a physical multi-hop wireless topology into a logical shared bus using time-slotted Glossy floods. The LWB is structured using periodic communication rounds, having period T, where each round consists of a sequence of slots. Each slot is represented by a Glossy flood in which all nodes within the network participate. The radio of each node is turned off between rounds in order to save energy. The synchronization maintained by the Glossy flooding primitive ensures that each node awakes in time to participate in the next communication round.

We briefly describe the operation of the LWB using a single-hop network, as depicted in Fig. 7, in the context of the multi-hop dissemination component model. We note that while we only consider a single-hop topology, the operation of the LWB is equally applicable to multi-hop topologies. Every LWB round is initiated by the host, and begins with the *schedule* slot *(S)*. The schedule contains the structure and allocation of slots within the round. The next slot, called the *contention* slot *(C)*, gives the opportunity for a node to request a periodic data stream so to disseminate data to the host. The round finishes with a schedule slot, which informs all nodes of the next round period as computed by the host.

In the example illustrated, node *A* detects an acoustic event that must be disseminated to the host for further analysis. During the next round, the node indicates its communication demands to the host, *e.g.*, a periodic data stream specified by an inter-packet-interval and a start time, by transmitting a stream request during the contention slot. Once the host receives the stream request, it will schedule appropriate bandwidth by allocating periodic *data* slots *(D)* to node *A*, and update the round period accordingly. During the next round, the schedule slot defines an *acknowledgement* slot *(ACK)* in response to the stream request, a data slot allocated to node *A*, together with the contention and schedule slots. Node *A* then disseminates its event data to the host, possibly over several rounds depending on the bandwidth allocated to it by the host. Once node *A* has no further data to transmit, it piggybacks a stream request into its last data slot so to remove the data stream. The removal of the data stream is confirmed by the host in the next round using an *ACK* slot.

Component Model. The time-triggered operation of the LWB, *i.e.*, sleep, wake-up, communicate, and return to sleep, is analogous to the event processing state machine embedded within the logical event-triggered component model introduced in Sec. 2. Specifically, we denote the time from an event arrival until the allocation of data slots as the wake-up time t_w, the time from the allocation of data slots until the removal of the data stream as the processing time t_p, and the average sleep state power dissipation P_s as the power dissipated for schedule and contention slots during round period T. We next describe the limitations of the LWB in supporting on-demand dissemination under the presence of contention, before detailing how we tailor the protocol in the context of the logical component model.

3.3.2 Event-based Low-power Wireless Bus (eLWB)

The LWB is designed to support data streams, making it particularly well suited to periodic data delivery with slowly changing traffic demands. However, when we consider event-triggered wireless sensing applications, the means by which periodic data streams are requested from the host give rise to the following challenges:

- Multiple source nodes may simultaneously detect an event, resulting in these nodes transmitting their stream request during the same contention slot. Due to the poor scaling of the capture effect, as experimentally evaluated in [24], the probability of a stream request being successfully decoded at the host reduces significantly as the number of contending nodes increases, thus increasing the wake-up time t_w due to random back-off mechanisms.
- The LWB supports the *sequential* allocation of *periodic and fixed* bandwidth to source nodes, which is in contrast to the requirements of event-triggered wireless sensing, where the *simultaneous* allocation of *aperiodic and variable* bandwidth to source nodes is needed. Therefore, even if the challenge of contending stream requests is overcome, it will still take several rounds for all contending nodes to have their bandwidth demands administered by the host, thus adversely increasing processing time t_p.

We address these challenges by modifying the behavior of the LWB, which we term the *Event-based Low-power Wireless Bus* (eLWB), without increasing the average sleep state power dissipation P_s. Specifically, we *(i)* reduce the wake-up time t_w by notifying the host when *at least one* event has been detected using an event contention slot, and *(ii)* reduce the processing time t_p by providing fixed bandwidth for the event streams and on-demand bandwidth for data streams.

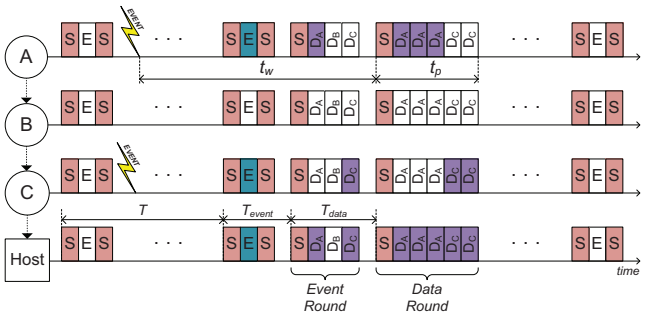

Figure 8. Example operation of the Event-based Low-power Wireless Bus (eLWB). Source nodes *A* and *C* detect an event within the same round period, disseminate their events and acquire bandwidth for the data streams.

Event Contention Slot. According to the simulation results presented in [39], the packet reception rate of concurrent transmissions with identical packet payloads is significantly higher than the concurrent transmission of packets with independent payloads. We leverage this result to improve the likelihood that the host is notified of at least one event in the case when multiple source nodes detect an event within the same round period. We achieve this by replacing the original contention slot with an *event contention* slot *(E)*, as illustrated in Fig. 8, whereby each node wishing to disseminate an event transmits a packet with an identical payload, *e.g.*, a packet containing the single byte `0x00`. Additionally, the host disregards the validation of the cyclic redundancy check when processing the contention slot. All source nodes are informed of a successful event contention by inspecting the change of round period to T_{event}, as specified in the schedule slot immediately following the event contention slot.

Event and Data Rounds. Once the host is notified of at least one event within the network, an *event round* provides all source nodes an opportunity to *(i)* disseminate an event and *(ii)* request bandwidth for data stream dissemination. The event round consists of a schedule slot and a unique data slot for each node within the network. The data slot has a fixed length of 1 byte, which is sufficient to indicate an event type and the required bandwidth for the data stream. In the multi-hop example depicted in Fig. 8, all source nodes are provisioned with a data slot during the event round, but only nodes *A* and *C* disseminate an event and request bandwidth for their data stream. The schedule slot associated with the event round defines the new round period, T_{data}, which informs all nodes of the beginning of the *data round*.

Once the host collects all the data stream requests, it partitions the available network bandwidth accordingly. In the example depicted in Fig. 8, the host allocates data slots to nodes *A* and *C* according to their demands. The schedule slot of the data round defines the allocation of data slots to the respective source nodes, and specifies the new round period $T - T_{data} - T_{event}$. In order to facilitate the rapid dissemination of event and data streams, the time offsets of the event round T_{event}, and data round T_{data}, are chosen to be significantly less than the round period T.

While we have only considered until now the dissemination of events from source to host, it is important to highlight that the eLWB supports bi-directional dissemination of events and periodic data streams. A source node may utilize the event round to indicate a periodic stream, *e.g.*, for node health information, and the host will schedule the corresponding data slot during each round. Additionally, the host may allocate data slots to facilitate unicast or broadcast communication between host and source nodes.

Protocol Limitations. Since the event round provides each source node with a dedicated data slot, the duration of the event round increases linearly with the number of nodes in the network. However, as each source node is only allocated one byte in the event round, the overhead remains realistic for typical deployments where the available bandwidth, $T - T_{data} - T_{event}$, constrains the number of simultaneous event disseminations to a small number of nodes, *e.g.*, around 20 nodes. If larger networks are required, hierarchical structures may be employed, possibly in combination with non-overlapping communication channels.

As with all synchronous protocols, the eLWB exhibits a fundamental trade-off between end-to-end latency and energy efficiency. In order to achieve a lower event latency, the round period T must be reduced, resulting in an increase of energy consumption during periods of inactivity. However, as we experimentally evaluate in Sec. 4 using an indoor testbed, the proposed eLWB protocol achieves a best-case event latency of 113.2 ms, while dissipating on average as low as 52.8 μW.

3.4 Physical Platform Architecture

Using the system design process introduced in Sec. 2, we designed and implemented each component such that the *logical* system architecture illustrated in Fig. 1 adheres to adaptability, responsiveness and energy efficiency design constraints. We now propose that these logical components be integrated onto a *physical* system architecture while preserving these same properties, if the following two conditions are supported:

- ***Limited Interference.*** Components that are active concurrently may exhibit resource interference with respect to processor clock cycles, memory or peripherals. Such resource interference may adversely impact the performance of components, for example, where a component is prevented from entering a low-power sleep mode due to the concurrent processing of another component. In order to preserve the responsiveness and energy-efficiency of components, we must limit resource interference wherever possible.
- ***Composable Construction.*** In order to retain the properties of each event-triggered logical component, the physical platform architecture must support composability [19]. This well-established system design principle makes it possible to interconnect components together without changing the properties, *i.e.*, responsiveness and energy efficiency, of the integrated parts. This powerful property is facilitated by the interconnection of components using interfaces with formally defined semantics.

Proposed Architecture. We achieve these two requirements by *(i)* mapping components that encounter resource interference onto dedicated processors, and *(ii)* interconnect the processors using an interface with predictable run-time behav-

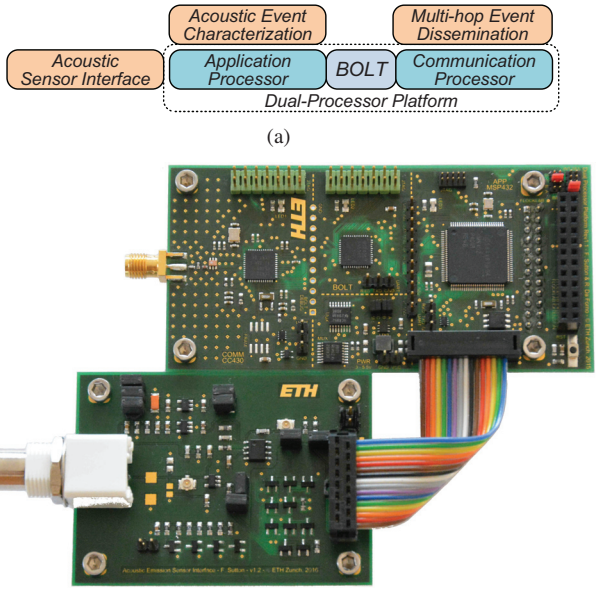

Figure 9. (a) Architecture of the physical platform, and (b) prototype implementation of the Dual-Processor Platform (top) and the acoustic sensor interface (bottom).

ior. As shown in Fig. 9(a), we propose to map the acoustic event characterization components to a dedicated application processor, and the multi-hop event dissemination components to a dedicated communication processor. We then interconnect these two processors using BOLT [35], a publicly-available ultra-low power processor interconnect that supports bi-directional asynchronous message passing with predictable run-time behavior. BOLT decouples the two processors with respect to time, power and clock domains, thus enabling the processors to execute concurrently without risk of resource interference, while also facilitating composable construction of event-triggered components.

Prototype Implementation. A prototype of the proposed platform architecture, termed the *Dual-Processor Platform*, and the acoustic sensor interface are depicted in Fig. 9(b). The platform consists of a 32-bit MSP432P401R ARM Cortex-M4 application processor running at 48 MHz, which is interconnected by BOLT to a 16-bit CC430F5147 communication processor running at 13 MHz.

Summary. We began this section by introducing a logical event-triggered component model that captures adaptability, responsiveness and energy efficiency design constraints. We then constructed an wireless acoustic sensing system from a pipeline of logical components, presented a concrete realization of each component, and demonstrated their integration onto a novel physical platform architecture. We next experimentally evaluate the developed prototype in the context of a real-world wireless acoustic sensing scenario.

4 Case Study: Codetection of Acoustic Events

In this section, we consider a specific real-world application, the monitoring of acoustic emissions in steep fractured rock walls [14]. The goal is to identify rock damage and fracture propagation by detecting and characterizing acoustic emissions caused by cryogenic processes, *e.g.*, volumetric expansion of freezing water within the rock wall. The deployment of a wireless acoustic sensing system makes it possible to capture these acoustic events with unprecedented spatial coverage and temporal resolution, which may in the future be leveraged to develop early warning systems.

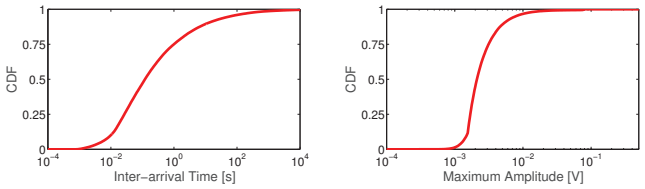

Figure 10. Characteristics of real-world acoustic events.

A field experiment spanning several months using a piezoelectric transducer installed 10 cm below the surface of a rock wall at 3500 m a.s.l [14] has identified these acoustic events to be sporadic in nature, and often occur in bursts. As depicted in Fig. 10, the observed acoustic events have inter-arrival times between a few milliseconds and several hours, and exhibit a maximum amplitude from a few millivolts up to one hundred millivolts.

Given the severe implications of rockfall and the harsh deployment conditions, a wireless sensor deployment must *(i)* rapidly detect, characterize and communicate acoustic events for analysis, and *(ii)* maximize operational lifetime. It follows that the system design and implementation of the wireless acoustic sensing system presented in Sec. 3 not only satisfies these requirements by design, but also supports the application-specific requirements.

In order to demonstrate the suitability of the developed prototype to this real-world application, we experimentally evaluate the prototype in a *codetection* use case. This is a particularly challenging scenario where several wireless acoustic sensors detect and characterize the same acoustic event, before each node simultaneously disseminates the event through the network as rapidly and energy efficiently as possible. To this end, we first assess the responsiveness of the eLWB protocol using an indoor testbed, and then evaluate the power dissipation of the developed prototype while being triggered by a real-world acoustic signal.

4.1 Responsiveness of Event Dissemination

Experimental Setup. We emulate the codetection of acoustic events at the network-level by utilizing the FlockLab [26] indoor testbed configured with fine-grained tracing capabilities [27]. Using 20 Olimex MSP430-CCRF nodes programmed with the eLWB, and deployed according to Fig. 11, we instruct source nodes 6, 22 and 28 to emulate the simultaneous detection of an event by transmitting during every eLWB event contention slot.

We evaluate the performance of the eLWB using three metrics: *(i) event detection* is the number of event contention slot transmissions that are successfully received by the host, *(ii) event and data dissemination* is the number of event and data streams transmitted by a source that are received at the host without error, and *(iii) event and data latency* is the time between a source transmitting in the event contention slot and the successful reception of the data stream at the host. We compute all metrics based on 100 eLWB rounds using a static protocol configuration, as listed in Table 3.

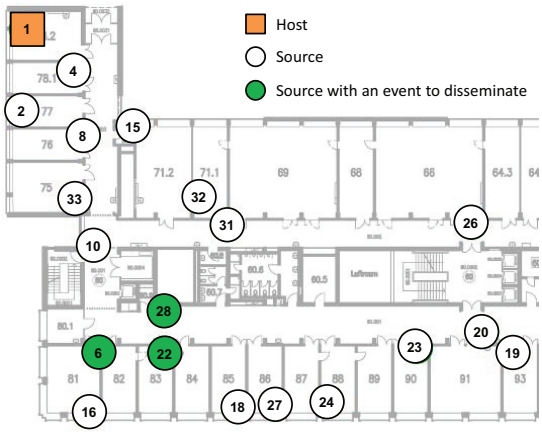

Figure 11. Indoor testbed deployment.

Table 3. eLWB parameters used in testbed experiments.

Parameter	Description
$T = 5, 10, 15$ s	Round period T
$T_{event} = 40$ ms	Event round time offset T_{event}
$T_{data} = 60$ ms	Data round time offset T_{data}
$M = 16$ bytes	Number of bytes per data slot
$N_S = 3$	Max. number of transmissions for schedule slots
$N_E = N_D = 2$	Max. number of transmissions for all other slots

Results. As illustrated in Fig. 12 (top), the success rate of event detection for all three contending source nodes is 100%. This means that despite the simultaneous event contention slot transmissions, the host successfully identifies that at least one event must be disseminated each round. In order to evaluate a lower-bound performance, the host only provides three data slots in the event round, during which each node requests two data slots. The experimental results show that all three source nodes disseminate their respective event and data streams with a success rate above 98%.

The event and data latency represents the best-case delay between an event detection and the successful dissemination of both event and data streams. We evaluate this metric for both eLWB and LWB protocols for each round, with the average presented in Fig. 12 (bottom). Firstly, we highlight that the latency per source node using the eLWB is approximately constant for all three round periods. This is the expected and desired behavior, since the operation of the event and data rounds are independent on the round period. Secondly, the latency of each source node, *i.e.*, 113.2 ms for node 6, 142.8 ms for node 22, and 169.8 ms for node 28, differs only by the duration of approximately two data slots. This is to be expected in our implementation, as the host schedules two data slots within the data round and assigns them in the order the of source node identities, *i.e.*, node 6 is assigned the first two, while node 28 is assigned the last two data slots.

In order to highlight the superior responsiveness of the eLWB protocol under contention, we compare against the original LWB. We adjust all LWB configuration parameters in order to improve its performance with respect to latency. Specifically, we set the LWB maximum and minimum round periods to $T_{max} = T$, and $T_{min} = 1$ s, respectively, and request a stream with a negative start time and an inter-packet-interval of T_{min} a total of 40 times. Furthermore, and most importantly, we analyze the LWB event dissemination for only one source, *i.e.*, node 6. This scenario represents

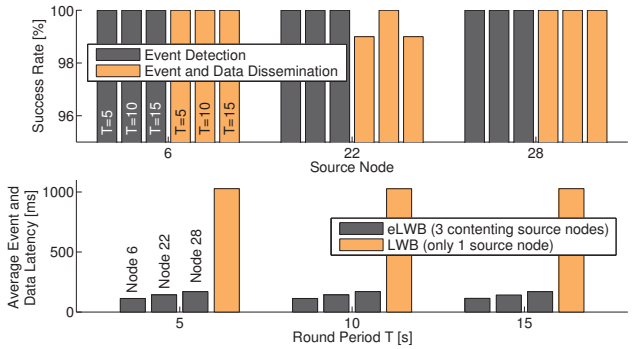

Figure 12. (top) Event contention and dissemination metrics of the eLWB, and (bottom) the average event and data latency of the eLWB compared to the LWB.

the absolute best-case scenario for the LWB, as simultaneous stream request transmissions would undoubtedly invoke random back-off, thereby delaying event dissemination by multiples of T_{min}. In summary, the experimental results show that the mean event latency of the eLWB is, at the very least, five times better than the LWB, whilst providing reliable dissemination for the codetection of acoustic events.

4.2 Power Dissipation of Developed Prototype

We first experimentally evaluate the power dissipation of the developed prototype under a static configuration, before leveraging the event-triggered component model presented in Sec. 2 to estimate the average power dissipation of the system with alternative run-time configurations.

4.2.1 Static Configuration Measurement

Experimental Setup. We measure the power dissipation of the developed prototype using a DC power analyzer at a supply voltage of 2.5 V. The source node is allowed to reach a steady operational state by synchronizing itself to the periodic eLWB rounds initiated by the host. We emulate a real-world acoustic signal by connecting an arbitrary waveform generator to the input of the acoustic sensor interface and playback an acoustic signal extracted from the field.

Results. Fig. 13 illustrates the power dissipation of the prototype source node during periods of inactivity and during the detection, characterization and dissemination of an acoustic event. Approximately half a second into the experiment, the multi-hop dissemination component awakes from sleep mode and checks if there are any pending messages in BOLT. As there are no messages, the component participates in the eLWB round without initiating transmission in the event contention slot, and returns back to sleep. At approximately five seconds into the trace, a real-world acoustic signal is injected into the acoustic sensor interface, awaking it from sleep, and subsequently triggering the acoustic event characterization component. Once all features have been extracted from the digitized acoustic signal, a message is written into BOLT containing the event feature set. The next eLWB round begins approximately half a second later, and the pending message is read out from BOLT, thus triggering multi-hop event dissemination. The source node indicates an event by initiating a transmission during the event contention slot and proceeds to disseminate the event using the event and data rounds according to the eLWB protocol

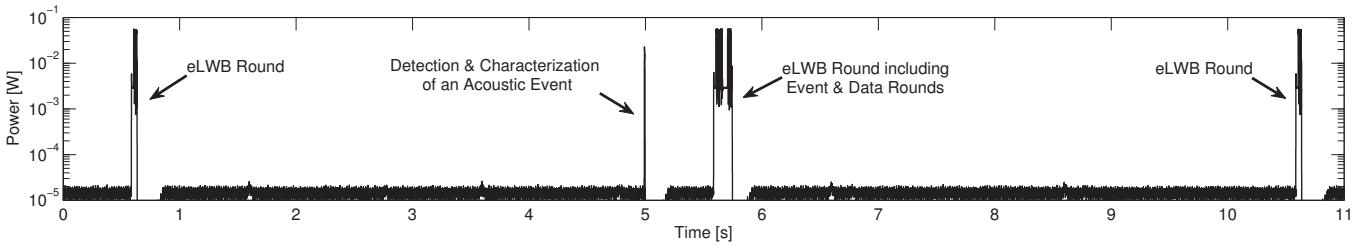

Figure 13. Power dissipation of the developed prototype detecting and characterizing an acoustic event extracted from a real-world application scenario, before dissemination using the eLWB protocol configured with round period T = 5 s.

Table 4. Component power dissipation of the developed prototype during periods of inactivity.

	Component	P_s
Dual Processor Platform	Acoustic Sensor Interface	6.2 μW
	Acoustic Event Characterization	2.5 μW
	BOLT Processor Interconnect	1.3 μW
	Multi-hop Event Dissemination (T = 15 s)	52.8 μW
		62.8 μW

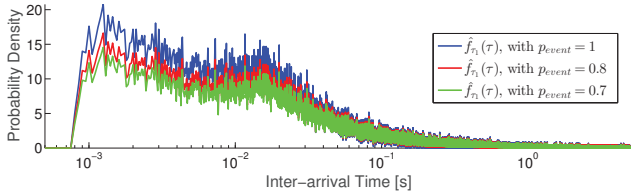

Figure 14. Probability density of input acoustic events for alternative run-time configurations.

detailed in Sec. 3.3.2, before returning to sleep. As there are no further acoustic events emulated, the source node awakes five seconds later for the next eLWB round.

Table 4 lists the component-level power dissipation of the developed prototype during periods of inactivity. In summary, the prototype dissipates 62.8 μW during sleep state, thus making it possible to support the codetection of acoustic events for several years using medium-capacity batteries.

4.2.2 *Adaptive Configuration Estimation*

In practice, a system designer is not only interested in the power dissipation of the system during periods of inactivity, but also during realistic operating conditions with alternative run-time configurations. In order to evaluate the impact of component-level adaptivity, we leverage the logical event-triggered component model presented in Sec. 2 to estimate the average power dissipation of the developed prototype.

Simulation Setup. We consider a scenario where the detection threshold for acoustic emissions is adjusted at run-time. We randomly filter an acoustic event stream extracted from the field to produce the probability density $\hat{f}_{\tau_1}(\tau)$ for a set of p_{event} values, as illustrated in Fig. 14. The choice of p_{event} increases the average event inter-arrival time, thereby decreasing the event arrival rate β, which is synonymous to increasing the detection threshold. We then estimate the average power dissipation P_{avg} of each component according to the analytical framework presented in Sec. 2.

Results. Table 5 lists the average power dissipation P_{avg} of each component for three run-time configurations. As expected, the power dissipation of each component decreases as the p_{event} decreases since the random filtering of events reduces the average rate at which acoustic events are detected by the system. It is evident from the analysis that the difference in power dissipation of the multi-hop event dissemination component between run-time configurations is up to two orders of magnitude larger than the power dissipation of the other two components.

Table 5. Average power dissipation of system components with alternative run-time configurations.

	Average Power Dissipation P_{avg}		
Component	$p_{event} = 1$	$p_{event} = 0.8$	$p_{event} = 0.7$
Acoustic Sensor Interface	38.5 μW	31.9 μW	28.7 μW
Acoustic Event Characterization	150.8 μW	125.1 μW	112.4 μW
Multi-hop Event Dissemination	3.6 mW	3.0 mW	2.7 mW

5 Related Work

The PinPtr [34] sniper detection system, the volcanic monitoring system presented in [38] and the environmental monitoring system in [14] all perform continuous sampling of an acoustic sensor, which leads to limited operational lifetime or necessitates appropriately dimensioned energy harvesting capabilities. We present a solution based on sensor-initiated wake-up, ensuring event characterization is performed only when an event actually occurs, therefore reducing energy consumption during periods of inactivity.

CargoNet [29] and the structural monitoring system presented in [25] are the closest to our work with respect to sensor-based wake-up and multi-hop data dissemination. The node used in CargoNet also employs an ultra-low power comparator to detect events but relies on an RFID transponder to initiate the communication of stored measurements. The sensing platform presented in [25] continually samples a strain gauge at 100 Hz in order to start the acquisition and signal processing of its attached acoustic sensors, before the data is transmitted using Low-Power Listening [31]. Our solution not only integrates an ultra-low power acoustic interface, but also incorporates a responsive and energy efficient wireless protocol that supports dissemination when an event is simultaneously detected by multiple nodes.

Zahedi et al. [40] present a passive wireless sensing system where an acoustic emission signal is directly modulated onto a high-frequency carrier. A high-powered reader generates the high-frequency carrier, and demodulates the received signal in order to recover the acoustic emission signal. While this solution is a very elegant single-hop solution between sensor and reader, it does not easily scale to a multi-hop network scenario due to limitations in communication range and challenges associated with concurrent transmissions.

6 Conclusions

This paper presents the design, implementation and evaluation of an efficient event-triggered wireless sensing sys-

tem. We introduce a logical component model that encapsulates adaptability, responsiveness and energy efficiency design constraints, and we demonstrate how this model can be used to facilitate the design and implementation of a wireless acoustic emission sensing system. An experimental evaluation of the developed prototype demonstrates significant advances in the state-of-the-art with respect to the responsiveness of multi-hop event dissemination under contention, and the ultra-low power dissipation of an event-triggered wireless sensing system during periods of inactivity.

Acknowledgements. We thank Marco Zimmerling, Romain Jacob, the anonymous reviewers, and our shepherd Neal Patwari for valuable feedback, and we thank Samuel Weber and Matthias Meyer for assistance with the acoustic field data. This work was scientifically evaluated by the SNSF and financed by the Swiss Confederation and Nano-Tera.ch.

7 References

[1] K. Chebrolu, B. Raman, N. Mishra, P. K. Valiveti, and R. Kumar. Brimon: A sensor network system for railway bridge monitoring. In *MobiSys*. ACM, 2008.

[2] M. Delgado Prieto, D. Zurita Millan, W. Wang, A. Machado Ortiz, J. A. Ortega Redondo, and L. Romeral Martinez. Self-powered wireless sensor applied to gear diagnosis based on acoustic emission. *IEEE Transactions on Instrumentation and Measurement*, 65(1):15–24, 2016.

[3] M. Doddavenkatappa, M. C. Chan, and B. Leong. Splash: Fast data dissemination with constructive interference in wireless sensor networks. In *NSDI*, 2013.

[4] J. Dong, E. Lowenhar, V. Godinez, and M. Carlos. State-of-the-art wireless acoustic emission system for structural health monitoring. In *Advances in Acoustic Emission Technology*, pages 15–22. Springer, 2015.

[5] W. Du, J. C. Liando, H. Zhang, and M. Li. When pipelines meet fountain: Fast data dissemination in wireless sensor networks. In *SenSys*. ACM, 2015.

[6] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley & Sons, 2012.

[7] S. Duquennoy, B. Al Nahas, O. Landsiedel, and T. Watteyne. Orchestra: Robust mesh networks through autonomously scheduled TSCH. In *SenSys*. ACM, 2015.

[8] P. Dutta, M. Grimmer, A. Arora, S. Bibyk, and D. Culler. Design of a wireless sensor network platform for detecting rare, random, and ephemeral events. In *IPSN*. IEEE Press, 2005.

[9] N. Dziengel, M. Seiffert, M. Ziegert, S. Adler, S. Pfeiffer, and J. Schiller. Deployment and evaluation of a fully applicable distributed event detection system in wireless sensor networks. *Ad Hoc Networks*, 37:160–182, 2016.

[10] M. Ervasti, S. Dashti, J. Reilly, J. D. Bray, A. Bayen, and S. Glaser. iShake: mobile phones as seismic sensors–user study findings. In *Proceedings of the 10th International Conference on Mobile and Ubiquitous Multimedia*. ACM, 2011.

[11] G. Feltrin, N. Popovic, K. Flouri, and P. Pietrzak. A wireless sensor network with enhanced power efficiency and embedded strain cycle identification for fatigue monitoring of railway bridges. *Journal of Sensors*, 501:4359415, 2016.

[12] F. Ferrari, M. Zimmerling, L. Mottola, and L. Thiele. Low-power wireless bus. In *SenSys*. ACM, 2012.

[13] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh. Efficient network flooding and time synchronization with Glossy. In *IPSN*. IEEE, 2011.

[14] L. Girard, J. Beutel, S. Gruber, J. Hunziker, R. Lim, and S. Weber. A custom acoustic emission monitoring system for harsh environments: application to freezing-induced damage in alpine rock walls. *Geoscientific Instrumentation, Methods and Data Systems*, 1(2):155–167, 2012.

[15] C. M. Grinstead and J. L. Snell. *Introduction to Probability*. American Mathematical Society, 2012.

[16] C. U. Grosse and M. Ohtsu. *Acoustic Emission Testing*. Springer Science & Business Media, 2008.

[17] P. Horowitz and W. Hill. *The Art of Electronics*. Cambridge University Press, 1989.

[18] P. Huang, L. Xiao, S. Soltani, M. W. Mutka, and N. Xi. The evolution of MAC protocols in wireless sensor networks: A survey. *IEEE Communications Surveys & Tutorials*, 15(1):101–120, 2013.

[19] A. Jantsch. Models of computation for networks on chip. In *ACSD*. IEEE, 2006.

[20] S. Jevtic, M. Kotowsky, R. P. Dick, P. A. Dinda, and C. Dowding. Lucid Dreaming: Reliable analog event detection for energy-constrained applications. In *IPSN*. ACM, 2007.

[21] B. M. Kelly, B. Rumberg, and D. W. Graham. An ultra-low-power analog memory system with an adaptive sampling rate. In *MWSCAS*, pages 302–305. IEEE, 2012.

[22] S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, and M. Turon. Health monitoring of civil infrastructures using wireless sensor networks. In *IPSN*. IEEE, 2007.

[23] H. Kwon, V. Berisha, and A. Spanias. Real-time sensing and acoustic scene characterization for security applications. In *ISWPC*, pages 755–758. IEEE, 2008.

[24] O. Landsiedel, F. Ferrari, and M. Zimmerling. Chaos: Versatile and efficient all-to-all data sharing and in-network processing at scale. In *SenSys*. ACM, 2013.

[25] Á. Lédeczi, T. Hay, P. Völgyesi, D. R. Hay, A. Nádas, and S. Jayaraman. Wireless acoustic emission sensor network for structural monitoring. *Sensors Journal, IEEE*, 9(11):1370–1377, 2009.

[26] R. Lim, F. Ferrari, M. Zimmerling, C. Walser, P. Sommer, and J. Beutel. FlockLab: A testbed for distributed, synchronized tracing and profiling of wireless embedded systems. In *IPSN*. ACM, 2013.

[27] R. Lim, B. Maag, B. Dissler, J. Beutel, and L. Thiele. A testbed for fine-grained tracing of time sensitive behavior in wireless sensor networks. In *Local Computer Networks Workshops*, 2015.

[28] G. Lu, D. De, M. Xu, W.-Z. Song, and J. Cao. TelosW: Enabling ultra-low power wake-on sensor network. In *INSS*. IEEE, 2010.

[29] M. Malinowski, M. Moskwa, M. Feldmeier, M. Laibowitz, and J. A. Paradiso. CargoNet: A low-cost micropower sensor node exploiting quasi-passive wakeup for adaptive asychronous monitoring of exceptional events. In *SenSys*. ACM, 2007.

[30] A. Papoulis. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill, 1965.

[31] J. Polastre, J. Hill, and D. Culler. Versatile low power media access for wireless sensor networks. In *SenSys*. ACM, 2004.

[32] B. Rumberg, D. W. Graham, and V. Kulathumani. A low-power, programmable analog event detector for resource-constrained sensing systems. In *MWSCAS*, pages 338–341. IEEE, 2012.

[33] J. Segura-Garcia, S. Felici-Castell, J. J. Perez-Solano, M. Cobos, and J. M. Navarro. Low-cost alternatives for urban noise nuisance monitoring using wireless sensor networks. *Sensors Journal, IEEE*, 15(2):836–844, 2015.

[34] G. Simon, M. Maróti, Á. Lédeczi, G. Balogh, B. Kusy, A. Nádas, G. Pap, J. Sallai, and K. Frampton. Sensor network-based countersniper system. In *SenSys*. ACM, 2004.

[35] F. Sutton, M. Zimmerling, R. Da Forno, R. Lim, T. Gsell, G. Giannopoulou, F. Ferrari, J. Beutel, and L. Thiele. Bolt: A stateful processor interconnect. In *SenSys*. ACM, 2015.

[36] N. Tatlas, S. Potirakis, S. Mitilineos, S. Despotopoulos, D. Nicolaidis, and M. Rangoussi. A wireless acoustic sensor network for environmental monitoring based on flexible hardware nodes. In *Proceedings of the Audio Mostly 2015 on Interaction With Sound*. ACM, 2015.

[37] P. Volgyesi, G. Balogh, A. Nadas, C. B. Nash, and A. Ledeczi. Shooter localization and weapon classification with soldier-wearable networked sensors. In *MobiSys*. ACM, 2007.

[38] G. Werner-Allen, J. Johnson, M. Ruiz, J. Lees, and M. Welsh. Monitoring volcanic eruptions with a wireless sensor network. In *EWSN*. IEEE, 2005.

[39] M. Wilhelm, V. Lenders, and J. B. Schmitt. On the reception of concurrent transmissions in wireless sensor networks. *IEEE Transactions on Wireless Communications*, 13(12):6756–6767, 2014.

[40] F. Zahedi, J. Yao, and H. Huang. A passive wireless ultrasound pitch–catch system. *Smart Materials and Structures*, 24(8):085030, 2015.