Mobile Underwater Backscatter Networking

Purui Wang, Sayed Saad Afzal, and Fadel Adib
Massachusetts Institute of Technology

Abstract

Underwater backscatter is a promising technology for ultra-low-power underwater networking, but existing systems break down in mobile scenarios. This paper presents EchoRider, the first system to enable reliable underwater backscatter networking under mobility.

EchoRider introduces three key components. First, it incorporates a robust and energy-efficient downlink architecture that uses chirp-modulated transmissions at the reader and a *sub-Nyquist chirp decoder* on backscatter nodes—bringing the resilience of LoRa-style signaling to underwater backscatter while remaining ultra-low-power. Second, it introduces a *NACK-based full-duplex retransmission protocol*, enabling efficient, reliable packet delivery. Third, it implements a *Doppler-resilient uplink decoding pipeline* that includes adaptive equalization, polar coding, and dynamic retraining to combat channel variation.

We built a full EchoRider prototype and evaluated it across over 1,200 real-world mobile experiments. EchoRider improves bit error rate by over 125× compared to a state-of-the-art baseline and maintains underwater goodput of 0.8 kbps at speeds up to 2.91 knots. In contrast, the baseline fails at speeds as low as 0.17 knots. Finally, we demonstrate EchoRider in end-to-end deployments involving mobile drones and sensor nodes, showing its effectiveness in practical underwater networked applications.

CCS Concepts

• **Hardware → Wireless devices**; • **Networks → Link-layer protocols**; • **Computer systems organization → Sensor networks**.

Keywords

Underwater Acoustic Backscatter, Mobility, Chirp spread-spectrum, Full-duplex wireless, Doppler effect

ACM Reference Format:
Purui Wang, Sayed Saad Afzal, and Fadel Adib, Massachusetts Institute of Technology. 2025. Mobile Underwater Backscatter Networking. In *ACM SIGCOMM 2025 Conference (SIGCOMM '25), September 8–11, 2025, Coimbra, Portugal.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3718958.3750507

1 Introduction

Low-power underwater sensor networks have important applications in long-term ocean climate monitoring, offshore aquaculture, and underwater infrastructure (such as submarine cables, offshore wind farms, oil and gas sites, etc.) [1–4]. These systems typically deploy a network of sensor nodes on a riverbed or ocean floor, and having these sensors upload their data to a mobile access point (such as an underwater drone, surface vessel, or ship) when in proximity [5, 6].

Recent advances in underwater backscatter could enable long-term deployments of such networks by reducing power consumption by 5 – 6 orders of magnitude compared to traditional underwater acoustic networks. Underwater backscatter nodes differ from traditional underwater acoustic networks[1] in that they communicate by reflecting rather than generating their own acoustic signals. This architecture achieves net-zero power consumption at nodes by offloading signal generation to an access point on an energy-equipped drone or ship. By eliminating energy-intensive acoustic transmission at sensor nodes, underwater backscatter conserves battery life for sensing tasks, enabling longer-term deployments of large-scale low-power sensor networks.

However, existing underwater backscatter systems still cannot work with mobility—a critical requirement for practical data collection scenarios like those described earlier, where backscatter sensor nodes need to upload their data to a mobile node. In particular, state-of-the-art systems have only been demonstrated in static configurations [7–11], where both the access point and backscatter nodes were fixed via underwater mounts or surface suspensions. In fact, even in static environments, natural currents may cause nodes to sway, leading to a significant increase in BER (and reduction in goodput) as we demonstrate empirically in §2.

Enabling underwater backscatter to operate reliably under mobility faces two key challenges:

- **Doppler:** Unlike traditional RF backscatter communication systems (e.g., RFID or ambient backscatter), underwater backscatter networks suffer from a significant Doppler shift when the nodes and/or the access point is mobile. The Doppler effect refers to the shift in frequency that occurs due to mobility. A well-known example is how we hear an ambulance siren go faster (higher frequency) as it approaches and slower (lower frequency) as it goes away. The Doppler shift in frequency is inversely proportional to the speed of propagation of the carrier wave. As a result, the frequency shift experienced by underwater acoustic backscatter is $200,000\times$ more significant than that experienced by RF backscatter, and thus cannot be neglected. [2]

  Leaving this Doppler shift unaddressed causes significant errors in synchronization, packet detection, and decoding. While this problem has been addressed in previous underwater communication systems, state-of-the-art solutions like OFDM are

[1]Underwater communication relies on sound/acoustics rather than radio frequency (RF) signals since RF decays exponentially in water.
[2]The speed of RF propagation is 3×10^8m/s, while underwater acoustics is around $1,500$m/s. Thus, the object needs to move at much higher speeds at RF in order to suffer from a problematic Doppler shift. Indeed, past work has introduced techniques to address Doppler for high-speed rails [12, 13], but such techniques are neither applicable nor needed for RF backscatter.

This work is licensed under a Creative Commons Attribution 4.0 International License.
SIGCOMM '25, Coimbra, Portugal
© 2025 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-1524-2/25/09
https://doi.org/10.1145/3718958.3750507

complex and power-hungry [14], making them infeasible to deploy on low-power underwater backscatter nodes.

- **Lack of robust link layer:** Underwater acoustic communication channels exhibit high bit error rates from multipath delay spread and channel variability [15, 16], exacerbated by mobility. In principle, one could improve reliability via ACK-based retransmission in the link layer. However, per-packet acknowledgments would introduce significant overhead, particularly in underwater environments due to high RTTs. Thus, such an approach would reduce network goodput while increasing energy consumption at the backscatter nodes.

To overcome these challenges, we present EchoRider, the first system enabling reliable mobile underwater networking. EchoRider introduces three key innovations:

- **Sub-Nyquist chirp decoder:** EchoRider's first component is a robust downlink modulation scheme, inspired by LoRa's frequency diversity in mobile multipath environments [17]. However, in contrast to typical LoRa chips, underwater backscatter nodes face prohibitive energy and complexity constraints for classical de-chirping methods, *e.g.,* frequency mixers and/or high sampling rate ADCs (with digital de-chirping). Thus, EchoRider introduces a new sub-Nyquist decoding scheme that preserves chirp robustness on low-power underwater backscatter nodes. The key novelty of this scheme is that it performs *direct sub-Nyquist bandpass sampling*, made possible by two factors: (1) the center frequency and bandwidth are within the same order of magnitude, and (2) EchoRider leverages the transducer and its matching network as a natural anti-aliasing filter. In §3.1, we describe this method in detail, and how EchoRider deals with the aliasing from sub-Nyquist and extends this method with an algorithm for frame synchronization and robust decoding.
- **NACK-based full-duplex backscatter retransmission protocol:** EchoRider's second component is a retransmission protocol that enables efficient recovery from packet errors. Unlike typical wireless networks, where sending acknowledgment packets (ACKs) has negligible overhead, these ACKs cause excessive overhead in underwater backscatter due to the long propagation delay of the acoustic channel coupled with the full-duplex (and asymmetric) nature of backscatter communication. To overcome this challenge, EchoRider introduces a NACK-based protocol, whereby the backscatter node opportunistically transmits on the uplink until it receives a downlink NACK for retransmission. A key challenge is enabling the backscatter node to decode the downlink NACKs in full duplex while backscattering its data. In §3.2, we describe this full-duplex protocol in detail, and how it performs active recovery upon retransmission.
- **Doppler-resilient uplink decoding:** Finally, to handle mobility-induced uplink channel variation in underwater backscatter, EchoRider's reader-side pipeline combines an adaptive equalization algorithm, differential decoder for Doppler phase cancellation, polar coding, and an equalizer retraining mechanism, as detailed in §3.3.

We implemented and field-tested an EchoRider prototype through extensive river experiments using a custom speed-control platform and an off-the-shelf underwater drone. The prototype comprises a reader (acoustic projector + receiver array) and up to 3 backscatter nodes. Our in-house backscatter nodes implement open-source designs for the acoustic transducers, matching networks, and control hardware [18]. The downlink decoding pipeline and uplink modulation were implemented on the micro-controller of the backscatter node, while the downlink retransmission protocol and uplink equalization were implemented on the reader.

We evaluated EchoRider in over 1,240 real-world experimental trials spanning different ranges, node speeds, and coding rates, benchmarking against the state-of-the-art underwater backscatter system as a baseline. Key findings include:

- While both EchoRider and the baseline can achieve a goodput around 1.4 kb/s when static (which is typical for underwater acoustic communication), the baseline's goodput drops to zero beyond 0.1 m/s, whereas EchoRider's goodput remains around 800 bps at speeds up to 1.5 m/s (which is often the maximum speed for commercial underwater drones [19, 20]).
- EchoRider achieves an uncoded median BER of 4×10^{-3} at a throughput of 2.5 kb/s when the node is moving with a speed of 0.9 m/s. In contrast, at the same throughput and power level, the baseline has a median BER of 0.25 even at a relatively low speed of 0.05 m/s. This shows that EchoRider outperforms the state-of-the-art baseline by over 125× in terms of BER under mobile settings.

Contributions: EchoRider is the first underwater backscatter system that can operate reliably under mobility. The design of EchoRider introduces multiple innovations including robust energy-efficient downlink decoding, a NACK-based retransmission protocol, and a Doppler-resilient backscatter decoding pipeline. The paper also contributes a prototype implementation and real-world evaluation of EchoRider, including the first demonstration of underwater backscatter communication involving a drone.

2 The Problem

Underwater backscatter is an asymmetric networking technology where a reader enables communication and arbitrates medium access for ultra-low-power nodes. In a typical setup (Fig. 1), the reader transmits an acoustic signal, and the node communicates by modulating the reflections (*e.g.,* switching between "reflective" and "non-reflective" states) of this signal. The reader senses the changes in reflection to decode the backscatter packets. Past research has studied underwater backscatter in various environments and developed applications, but has been limited to static setups [7, 21].

To understand the challenges of mobility, we conducted an experiment with underwater backscatter in a river environment (see §4.3), reproducing a state-of-the-art single-transducer node system [21].[3] The node backscattered packets on the uplink at 2.5 kb/s. We compared static and mobile scenarios, with the node moving at 0.05 m/s and 0.4 m/s.

[3]For simplicity, we did not consider multi-transducer arrayed nodes [21], whose main benefit is range extension.

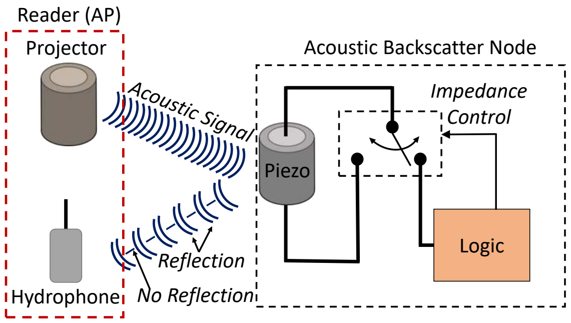


Figure 1: Piezo-Acoustic Backscatter. Underwater backscatter systems communicate bits of zero and one by switching piezoelectric load impedance. The switch is controlled by some low-power logic (*e.g.,* a low-power microcontroller).

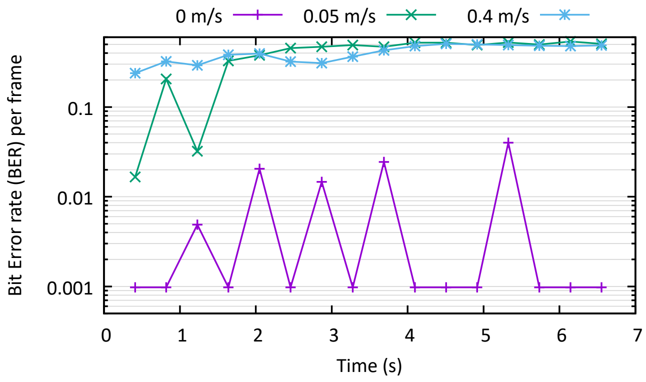


Figure 2: BER versus Mobility. The figure plots the BER per frame of typical underwater backscatter uplink as a function of time and different mobility speeds. The BER quickly rises to 0.5 with speeds as low as 0.05 m/s.

Fig. 2 shows the per-frame bit error rate (BER) over time with and without mobility. [4] In the static case, the BER fluctuates between 0.02 and 10^{-3}, which falls within the expected BER range for uncoded bits in underwater acoustic communication.[5] However, even at low speeds like 0.05 m/s, the BER quickly rises to 0.5 (becoming undecodable) under mobility. This is because current underwater backscatter decoders cannot handle rapid channel fluctuations. Moreover, even if the initial BER is acceptable (e.g., 10^{-2}) in the first frame after the header, the estimated channel from the header quickly diverges due to mobility, making subsequent frames undecodable. In principle, one could introduce a new header before each frame, but such an approach would introduce significant overhead (thus draining the sensor's battery). Moreover, at the higher speed of 0.4 m/s, even the first frame is undecodable. This motivates the need for a new design that is robust to mobility and dynamic channels.

[4]Note that these results were generated after applying an adaptive equalizer to mitigate the impact of multipath [21].
[5]The pattern of fluctuation is due to natural channel variability, *e.g.,* caused by river currents.

3 Design

EchoRider is an underwater backscatter system with a robust link-layer protocol that operates reliably in both static and mobile environments. Fig. 3 shows the end-to-end system architecture of EchoRider. The system architecture comprises three key components: a robust downlink modulation scheme, a retransmission protocol, and a Doppler-resilient uplink decoder. In this section, we provide a detailed explanation of each of these subcomponents.

3.1 Energy-Efficient Robust Downlink

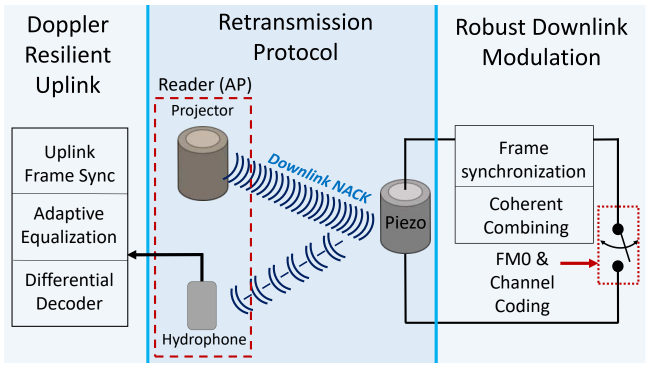


Figure 3: An overview of EchoRider's architecture. The figure shows the three main sub-components that make up the EchoRider system. First, the backscatter node processes the uplink data to remove Doppler effects. In the event of an error, it requests a retransmission via a downlink NACK. The backscatter node then uses its ultra-low power downlink decoder to respond to the NACK interrupt.

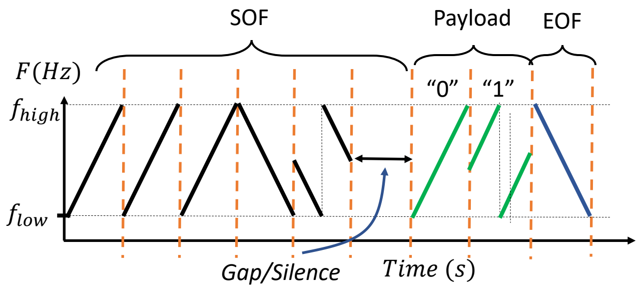


Figure 4: EchoRider's Downlink modulation scheme. The figure depicts a downlink frame, which comprises a Start of Frame (SOF) of three up chirps, two downchirps (one zero and one half-bandwidth offset), and a silence period; a payload (*e.g.,* 2 bits); and a downchirp for End of Frame (EOF).

The first component of EchoRider is a downlink modulation scheme, supporting link layer functions such as querying the backscatter node or retransmission. Previous underwater backscatter-based systems relied on single-carrier downlink schemes, *e.g.,* pulse width modulation (PWM). However, the single carrier becomes unreliable for underwater environments, especially under mobility, where the signal is distorted by multipath frequency selectivity and rapid fading of underwater channels. Unfortunately, complex equalization schemes—often used to handle such channel impairments—are too power hungry [14] for the energy-and-compute-constrained backscatter nodes. This is why existing downlink decoders suffer under mobility (as we demonstrate empirically in §5.3).

3.1.1 Chirp-based downlink: To overcome this problem, EchoRider employs a chirp-based modulation scheme for downlink communication. Chirps provide inherent frequency diversity, ensuring resilience to multipath fading. Furthermore, Doppler shifts minimally affect chirps since frequency shifts manifest as time delays that can be easily compensated.

Fig. 4 shows EchoRider's downlink command, comprising three parts: a start of frame (SOF), a payload with query data, and an end of frame (EOF) downchirp. The SOF contains 3 upchirps, 2 downchirps, and a silent interval. The payload encodes bits as sequential upchirps. During decoding, EchoRider first detects and synchronizes to the SOF; then, it decodes the data symbols sequentially until detecting an invalid symbol. It validates the EOF downchirp to finalize the downlink packet. Next, we describe EchoRider's decoding and frame synchronization:

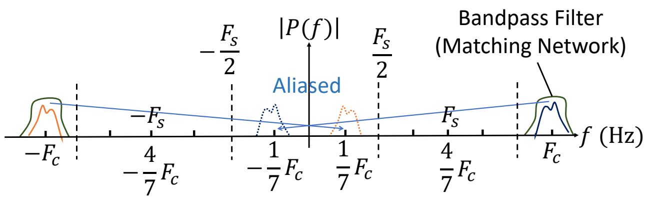
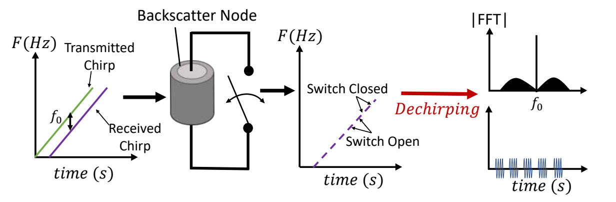


Figure 5: Sampling Downlink Chirp. Although aliasing occurs when the chirp is sampled below the Nyquist rate (at $\frac{4}{7}F_c$), with the bandpass filter (green), it moves the chirp bands (depicted in yellow and blue) as a whole without impacting decoding.

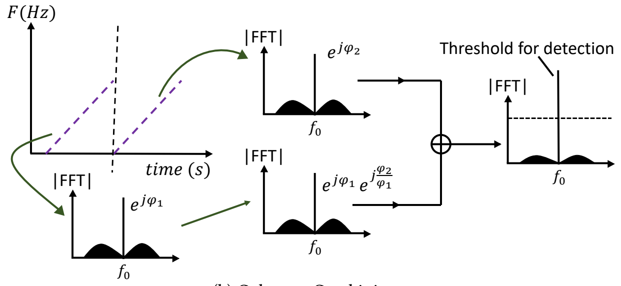
3.1.2 Sub-Nyquist chirp decoding: While chirp-based downlink communication resists mobility-induced distortions, conventional decoding demands complex hardware with high computational and power overhead. In particular, existing chirp-decoding approaches would require the node to either sample the full chirp (with higher-speed ADCs) [22] or to locally generate the chirp signal prior to mixing [23]; both approaches would consume significant energy, significantly exceeding the power consumption of existing underwater backscatter systems.[6]

To enable low-power decoding, EchoRider introduces a sub-Nyquist sampling mechanism. This method leverages aliasing to both downconvert and sample the chirp signal. Note, however, that our choice of sub-Nyquist sampling frequency needs to be higher than the bandwidth of the chirp itself – in order to capture the full chirp content – albeit still lower than the carrier (or twice the carrier, which would otherwise be required for Nyquist sampling).

[6]In Sec. 4.1.4, we show that implementing such classical approaches would consume more than 2 mW, which is 4× higher than past implementations of underwater backscatter [7, 24].



(a) Single Node Full-Duplex Backscatter



(b) Coherent Combining

Figure 6: Downlink Synchronization and Detection. (a) shows how EchoRider enables full-duplex backscatter by dechirping a downlink signal (already modulated with uplink) and extracting the peak in FFT domain. (b) shows how EchoRider reduces chances of false alarms by dechirping two consecutive upchirps in the SOF followed by coherent combining to improve SNR.

To perform sub-Nyquist sampling, our architecture uses a bandpass filter[7] – which ensures that other signals and noise aren't aliased – followed by an ADC sampling at $\frac{4}{7}F_c$.[8] , enabling signal acquisition below the Nyquist limit to minimize power consumption. While sub-Nyquist typically induces aliasing, EchoRider leverages aliasing to shift the entire spectrum into the target band. Fig. 5(b) illustrates this concept in the frequency domain: by sampling the chirp at $\frac{4}{7}F_c$, the entire signal undergoes aliasing, and the aliased copies move closer to the DC frequency, now centered at $\frac{1}{7}F_c$.

Interestingly, the reason this method is possible in the context of underwater backscatter/communications – but not RF – is that while the bandwidth is significantly smaller than the carrier frequency, they are still within an order of magnitude, which allows us to do direct sampling at sub-Nyquist.

3.1.3 Downlink frame synchronization and detection: To facilitate downlink communication between the transmitter and the backscatter node, it is necessary to detect and synchronize to the beginning of the transmitted frame. This is challenging due to several reasons: First, the synchronization algorithm needs low latency so that downlink decoding can be performed in real-time. Second, the algorithm needs to perform downlink detection during simultaneous uplink backscatter transmission (*i.e.,* full-duplex mode). This is because downlink decoding is necessary not only for initiating the transmission, but also for receiving NACKs from the reader to restart the transmission (the NACK protocol will be described in §3.2). Third, robust detection of downlink signals is needed to

[7]This is integrated with the transducer matching per §4.1.
[8]The exact choice of $\frac{4}{7}F_c$ and the mathematical relation between sampling rate and central frequency are detailed in Appendix §A.

prevent missed alarms (causing repeated downlink) or false alarms (causing spurious retransmissions) that harm the overall goodput of the system.

A straightforward solution for synchronization involves correlating with the transmitted chirps and identifying the time index that maximizes the correlation. However, applying standard correlation can be computationally inefficient or may require complex computations that introduce significant delays [25].

Instead, EchoRider introduces a new synchronization algorithm that leverages the underlying physical properties of the SOF sequence. It operates in two key steps: first, it performs *buffer-based synchronization* for real-time calculation of the SOF offset within a buffer segment; second, it performs *coherent combining* to identify which segment contains the SOF. This detection is robust to missed or false alarms under full-duplex mode. We detail these steps below:

EchoRider's buffer-based synchronization step is inspired by past literature on LoRa synchronization [17, 26]. The method involves using the chirp frequency offset to resolve the SOF time offset within each segment. Let us define a segment as having the same length as the chirp. The key idea is that a downchirp *anywhere* within the SOF upchirps will result in a peak when using the equally sized sampling window for dechirping. However, without prior synchronization, a chirp might have span two segments; yet at least two consecutive segments covered by the SOF chirp will have the same unique peak FFT frequency after dechirping, because the three repeated upchirps in the SOF (Fig. 4) have *the same* time offset from the buffer start.

This peak frequency f_o in the FFT is then used to determine the time offset t_s using $t_s = \frac{f_o}{s_c}$ (where s_c is the slope of the chirp) for frame synchronization. EchoRider then realigns the segments according to t_s in real time.

Detection Under Self-Interference: To complete the loop for SOF synchronization, it remains necessary to determine which segment contains the SOF, or equivalently in our real-time system, to *detect whether the current segment contains the SOF.* A possible criterion first involves the peak-to-sum ratio in the FFT domain as an SNR metric: if a chirp overlaps with the detection window, dechirping it results in a single tone (assuming aliasing) and a unique peak in the FFT domain; but if it is noise, the energy will be spread across all SNR bins. Then, thresholding this SNR metric can distinguish a SOF-containing segment from noise.

Specifically, Fig. 6a illustrates the backscatter node receiving the downlink chirp while transmitting uplink data, resulting in chirp voltage being modulated with the uplink data and an on-off keying (OOK)-modulated single-tone signal upon dechirping. In the FFT domain, the peak amplitude is reduced due to some power being distributed across the uplink bandwidth. Consequently, this can lead to detection failures or false alarms in noisy channels, which causes protocol malfunction and severely affects system goodput.

To address this challenge, EchoRider leverages *coherent combining* in the second step to make frame detection robust against false alarms by employing the peak amplitude of the frequency f_o after dechirping. The peak amplitude serves an additional purpose—by applying a threshold over the peak value, EchoRider can determine whether a downlink signal was transmitted (*i.e.,* minimize false alarms). Specifically, after frame synchronization, EchoRider employs the three up-chirps in the SOF and extracts two copies from the received chirps. Subsequently, it individually de-chirps each copy, identifies the peak amplitude for each, and estimates the phases φ_1 and φ_2 from the FFT's maximum amplitude peak for both copies. The first copy of the chirp is then multiplied by $e^{j\frac{\varphi_2}{\varphi_1}}$. After aligning phases, adding them coherently enhances the peak component by $\sim$ 6dB while the uplink data is only enhanced by $\sim$ 3dB, as illustrated in Fig. 6b. EchoRider then repeats this combining process for the two down-chirps (with a circular shift in FFT bins) in the SOF and ensures that the peak amplitude is above a threshold after combining for both up-chirps and down-chirps. We provide the pesudocode for the SOF synchronization algorithm in Appendix §B.

This results in improved SINR for downlink SOF detection based on the peak-to-sum ratio metric, reducing the chance of false alarms or missed commands. Once EchoRider successfully detects the two up-chirps and down-chirps, the logic controller directs the backscatter node to cease the ongoing uplink during the "silence" period of the downlink SOF (Fig. 4), so that the remaining chirp symbols in the payload are no longer modulated with the uplink data. Subsequently, EchoRider can proceed with decoding the remaining downlink payload using the standard LoRa protocol [17].

3.2 Full-Duplex NACK Protocol

EchoRider employs a retransmission protocol to ensure link-layer reliability. Conventional wireless networks use acknowledgments (ACKs), where receivers confirm successful packet decoding. For underwater backscatter, this would require interrupting sensor data streaming (e.g., of camera images [8]) to await reader ACKs.

However, ACK-based protocols introduce substantial overhead. Each uplink transmission requires a long training sequence (see §3.3) to enable reader decoding under multipath. If transmissions pause for ACKs, the node must retransmit this sequence, incurring up to 50% throughput loss.[9] Instead, EchoRider employs a NACK-based protocol: the reader signals decoding failures, prompting retransmission only when necessary, avoiding redundant training sequences.

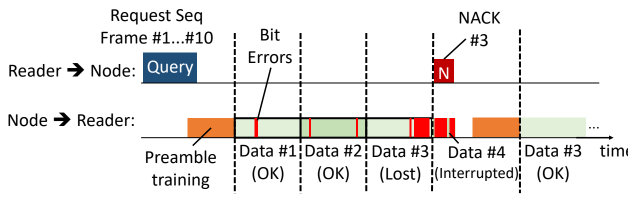


Figure 7: EchoRider's Retransmission Protocol. The figure shows how EchoRider sends a NACK to request a retransmission. After receiving a downlink query, the backscatter node responds with a preamble and data packets. Since there are many bit errors in data packet #3, the reader sends a downlink NACK and the backscatter node resends the preamble followed by packet #3.

As shown in Fig. 7, EchoRider's protocol begins when the reader sends a downlink query. The backscatter node responds with a

[9]Training sequences span hundreds of milliseconds [21]. For a 1024-bit frame with a 512-bit training sequence at 2.5 kbps, training occupies 205 ms, critical for channel tracking with adaptive equalizers.

preamble (training sequence) and data frames. The reader uses the preamble to train the decoder, and then decodes the data. While forward error correction (FEC, per §3.3.1) corrects most errors, if the reader detects uncorrectable errors (*e.g.,* in Data #3), it sends a NACK. This triggers the backscatter node to retransmit the preamble and the corrupted frame. Retransmission temporarily disrupts decoding of the subsequent frames (*e.g.,* Data #4), but the backscatter node resumes normal uplink transmissions afterward.

3.3 Differential Backscatter Decoder

Having detailed EchoRider's robust chirp-based downlink active recovery mechanism, we now present its Doppler-resilient uplink decoding methodology for mobile underwater backscatter.

During uplink communication, the reader transmits a single frequency on the downlink and captures the backscattered reflection modulated by the node. While prior implementations used phase-tracking equalizers effective in static/semi-static environments [21], these fail under mobility (Fig. 2).

To address this issue, EchoRider leverages the inherent differential encoding of backscatter FM0 modulation to remove phase shifts due to Doppler. FM0 encoding features phase transitions at the beginning ($0 \longrightarrow \pi$) and in the middle of the symbol, where the data is encoded in the latter (0: $\pi \longrightarrow 2\pi$, 1: no shift), creating differential phase relationships between consecutive symbols. Since Doppler phase shifts remain coherent between adjacent symbols, calculating the phase difference between successive symbols cancels the excess Doppler component [27]. This phase difference calculation uses the previous symbol phase conjugate $\varphi^*_{k-1} = \hat{d}^*_{k-1}/|\hat{d}_{k-1}|$ and multiplies it by the current symbol $\hat{d}_k$, yielding the Doppler-free phase difference $\hat{b}_k = \hat{d}_k \varphi^*_{k-1}$.

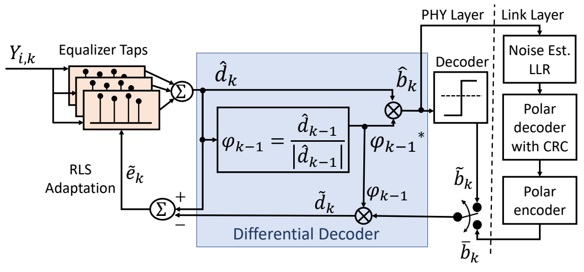


Figure 8: EchoRider's Uplink Decoding Pipeline. Received symbols first pass through an equalizer to combat ISI. The equalizer output $\hat{d}_k$ enters a differential decoder that removes Doppler phase shifts by multiplying $\hat{d}_k$ with φ^*_{k-1} (conjugated previous symbol phase). The reference symbol $\tilde{d}_k = \tilde{b}_k \varphi_{k-1}$ enables error computation $\tilde{e}_k = \hat{d}_k - \tilde{d}_k$ for equalizer taps updates.

Fig. 8 illustrates this decoding pipeline. Baseband symbols $Y_{i,k}$ [10] from the ith hydrophone channel at the kth time index undergo per-symbol equalization with dynamically updated filter taps. The combined equalized outputs pass through differential decoding (removing residual Doppler phase) followed by threshold-based bit detection to obtain $\tilde{b}_k$, which drives timely equalizer adaptation during decoding.

By leveraging the differential nature of FM0 backscatter to remove the Doppler induced phase, we ensure proper equalizer filter updates despite substantial phase fluctuations in underwater acoustic channels. However, practical decoding faces implementation challenges requiring further consideration:

3.3.1 Forward error correction (FEC) and CRC check: While our enhanced equalizer mitigates bit errors from dynamic multipath, residual ISI and low SNR at range necessitate additional error mitigation. EchoRider adopts Polar code with CRC16 (3GPP TS 38.212 [28]) to ensure error-free uplink communication. Each frame combines payload bits of a specific size, CRC16, and zero-padding bits (*i.e.,* "frozen bits") to form a 1024-bit codeword for data transfer [11]. Standard reliability-based bit placement and polar transforms generate the final codeword for backscatter modulation. Smaller payloads increase redundancy for low-SNR resilience.

At the receiver, equalized symbols $\hat{b}_k$ convert to Log Likelihood Ratios (LLRs) using sliding-window SNR estimation. Finally, a CRC-aided list decoder [29] either recovers error-free payloads (validated via CRC16) or triggers retransmission (§3.2).

3.3.2 Equalizer refresh: Despite dynamic equalizer tap adaptation and Doppler-resistant differential decoding, equalizer performance degrades over time—primarily from sampling desynchronization (Doppler-induced symbol stretching/compression), and error propagation by hard-decision feedback ($\tilde{b}_k$). While initial synchronization and tap training use a known preamble sequence (§3.2), prolonged frames exacerbate these issues.

EchoRider refreshes the equalizer using the FEC-decoded ground truth (§3.3.1). After CRC validation, the reader regenerates the exact modulated bits $\bar{b}_k$ to:

(1) Re-synchronize sampling via correlating the received samples against $\bar{b}_k$ as the template;
(2) Reinitialize equalizer taps, using $\bar{b}_k$ as training sequence.

These refresh steps mitigate error accumulation, sustaining equalizer robustness across frames, as depicted on the right side of Fig. 8.

4 Implementation

4.1 Backscatter Node

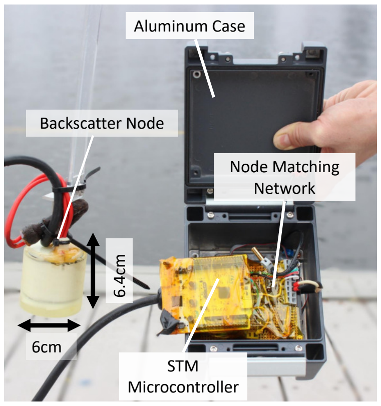
Fig. 9a shows the underwater backscatter node used by EchoRider. The backscatter node architecture includes an underwater transducer, transformer-based matching network, and an on-board tag firmware.

4.1.1 Transducer: The transducer's core element is a customized piezoelectric cylinder (PZT-42-⌀50.8 mm *⌀44.704 mm * 38.1 mm) from Zibo Yuhai Electronic Ceramics Co., Ltd., with a nominal resonance frequency of 22 kHz in radial mode. The transducer was fabricated using a process similar to that detailed in prior works [7, 30].

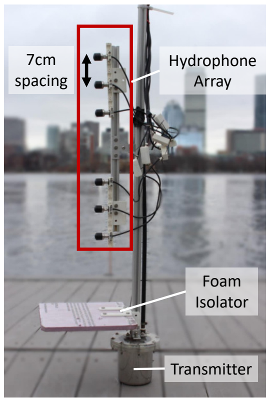
4.1.2 Matching Network: We implemented wideband matching for our backscatter node for maximum power transfer to the ADC, which improves both downlink SNR and uplink communication

[10]We implemented the preprocessing of down-conversion at upper and lower subcarrier sidebands, which eliminates the carrier leakage and remaps the $\{\pi, 2\pi\}$ FM0 phase shifts into a $\{0, \pi\}$ differential BPSK waveform [21].

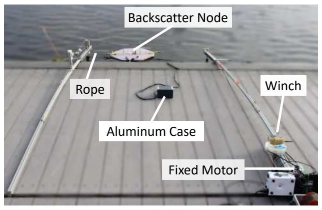
[11]256-bit codeword for 64-bit MAC UID, per §6.2.



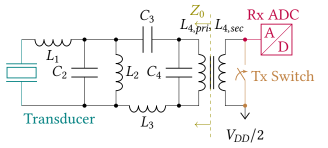
(a) Underwater Backscatter Node.



(b) Reader.



(c) Sample Experimental Setup.



(d) Matching Network.

Figure 9: System Implementation. (a) shows the implementation of the backscatter node (b) shows the reader with the hydrophone array and the acoustic transmitter (c) shows the platform used to move the node underwater using ropes and pulleys (d) shows the matching network implemented to improve the downlink and uplink SNR.

range. As shown in Fig. 9d, the matching consists of a 4th order LC ladder network to create a wideband filter. We used 1206 C0G capacitors and custom-wound inductors on TDK RM-4 ferrite cores. The values of the capacitors $C_2 = 292\,\text{nF}$,$C_3 = 16\,\text{nF}$,$C_4 = 141\,\text{nF}$, and inductors $L_1 = 2.52\,\text{mH}$, $L_2 = 164\,\mu\text{H}$, $L_3 = 2.99\,\text{mH}$, $L_{4,pri} = 337\,\mu\text{H}$ and load impedance $Z_0 = 131\,\Omega$ were selected using numerical optimization,[12] where we minimized the reflection coefficient S_{11} parameter of the network within a certain bandwidth. This can be expressed mathematically as:

$$\min_{L_{1,2,3,4},C_{2,3,4},Z_0} \max_{f\in[f_l,f_h]} |S_{11,\text{ref}Z_0}|$$

where Z_0 is the load impedance expected at the transformer primary coil $L_{4,pri}$. f_l, f_h represent the limits of the band of interest. We used an optimized $L_{4,sec} = 5.1\,\text{mH}$ to scale up $Z_{0,sec} \approx 1.9\,\text{k}\Omega$ so the voltage is maximized at the ADC input (without causing saturation). Since the ADC cannot sample negative voltages, the input is biased with a DC offset of $V_{DD}/2$. Our matching network achieved $|S_{11}| < -10\text{dB}$ over the frequency range of $22\text{kHz} \pm 4$ kHz.

4.1.3 The MCU and firmware: The firmware of our backscatter runs on an STM32 microcontroller unit (MCU). To generate the uplink payload, this firmware runs a Maximum-Length Sequence (MLS) pseudorandom generator, and on top of it, it encodes with the Polar encoder (§3.3.1) and an FM0 modulator, which is subsequently clocked out via low-power SPI3 that drives the backscatter Tri-state Tx switch 74AXP1G125 [31] (between Low and Hi-Z). For downlink reception, the ADC4 monitors the voltage of the switching port using the aliased $f_s = 12.8\,\text{kHz}$, equivalent to 3.2 kHz I/Q (see §3.1, and §A), and the subsequent dechirping and 64-point complex FFT are performed in real-time in the ADC's DMA interrupt handler.

[12]These values will vary with different transducers or operating frequencies.

4.1.4 Power consumption: To evaluate the power consumption of the backscatter node, we measured the current consumption of different components with the digital multimeter DMM6500[32] (at a voltage of 2.8V and using a 1-second averaging window). Table 1 compares the power consumption of EchoRider with the partial implementation that performs full-Nyquist sampling. The rows represent the two phases of operation: (a) the downlink phase, when the backscatter node is only listening, and (b) the full-duplex phase, where the node simultaneously transmits uplink data while listening for downlink commands (for NACKs). We make the following remarks:

- EchoRider achieves around 3.6× reduction in power consumption for both operation modes (downlink phase, and full-duplex) when compared to the full-Nyquist ADC implementation. This is because EchoRider's pipeline is capable of correctly decoding the downlink despite sampling the received signal at a much lower rate (*i.e.,* sub-Nyquist sampling) as discussed in §3.1.2.
- The power consumption for full-duplex communication (uplink and downlink) is slightly higher than that of the downlink phase alone. This is because, in full-duplex mode, EchoRider requires additional computational power for uplink transmission. Specifically, before sending uplink data, EchoRider must perform polar coding on the data bits, which increases computational load and power consumption.
- Crucially, this shows the downlink dominates system power (as uplink only contributes < 3%), highlighting the necessity of sub-Nyquist sampling (§3.1.2).

Finally, it is worth noting that the power consumption of EchoRider is comparable to past state-of-the-art underwater backscatter systems, which range from $120\mu W$ to $500\mu W$ [7, 24, 33]. The power consumption is slightly higher, which is justifiable with the significantly higher reliability.

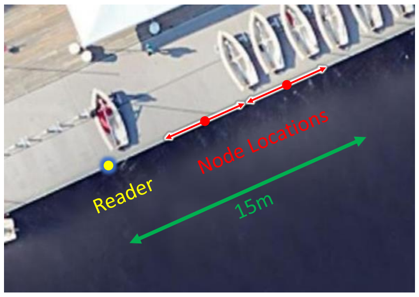
Operation Mode	**EchoRider**	**Partial Implementation (Full Nyquist)**
Downlink Only (μW)	586.60	2168.32
Full-Duplex (μW)	602.56	2196.32

Table 1: Power Consumption for Different Scenarios. We compare the power consumption between EchoRider with sub-Nyquist sampling and a partial implementation that uses full-Nyquist ADC sampling rate ($4 \times f_c$), in downlink phase (only the downlink reception is enabled) and full-duplex phase (both downlink reception and uplink encoding are enabled).

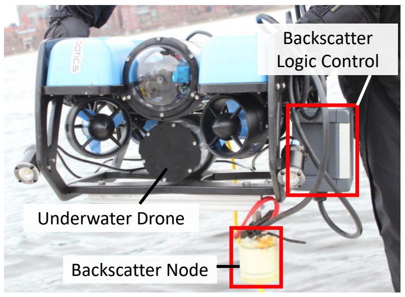
4.2 Reader

Fig. 9b shows our reader consisting of 4 main components:

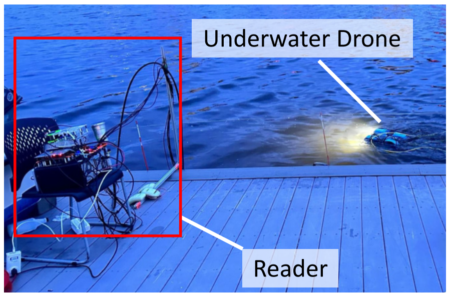
4.2.1 Transmitter: We use a piezo cylinder transducer resonant at 23 kHz enclosed within a customized thin-wall stainless steel housing. The interior of the housing is filled with liquid paraffin, providing insulation and relaying the acoustic signal. In order to



(a) Controlled experiment layout.



(b) Underwater drone.



(c) Drone Evaluation Setup.

Figure 10: EchoRider Evaluation. (a) shows an overhead view of the dock used for evaluation, marking sample locations where the experiments were conducted and the positions of the reader and backscatter node. (b) shows a BlueROV underwater drone equipped with a backscatter node and logic control circuit. (c) shows the setup used to test the performance of EchoRider in a mobile setting with an underwater drone.

transmit the wideband chirp signal, the transmitting chain employs a matching network made of film capacitors [34] and PM50-39 inductor core [35]. It employs a similar second-order design (*i.e.,* C_2, L_2, C_3, L_3 not used) as in §4.1.2, which uses $L_1 = 2.53$ mH, $C_4 = 278$ nF, $L_{4,pri} = 179\,\mu$H for the 23.04 ± 3 kHz band. The transformer secondary $L_{4,sec} = 8\,\mu$H matches the 4Ω speaker load, driven by a TI TPA3245 [36] power amplifier and an ESS ES9018 [37] DAC.

4.2.2 Receiver Array: The receiving system is built around six low-cost Aquarian Audio H1C hydrophones, arranged in a vertical array above the projector. An AD8656 preamp is used specifically for the H1C hydrophones [38], in conjunction with a Cirrus Logic CS5381 ADC [39]. A foam board that reflects sound waves is placed between the hydrophone and projector to prevent hydrophone saturation due to self-interference of the excitation carrier.

4.2.3 Digital Backend and Software: We designed and fabricated a custom PCB for the digital back-end of EchoRider's reader. The PCB incorporates eight channels of ADCs and DACs connected to an XC7A200T FPGA. This PCB is controlled via PCIe by a computer, which generates the carrier and downlink commands and processes the hydrophone data all in real-time. We carefully designed the software to enable the real-time retransmission protocol: we implemented sample streaming from/to the FPGA as a hardware abstraction layer in a C++ gRPC service. The rest of the reader was implemented using Python client scripts that generate and process chunks of samples for real-time downlink generation and uplink decoding.

4.3 Evaluation

We evaluate our system in a realistic setting, detailed below:

4.3.1 Motion Mechanism: To evaluate underwater backscatter under mobility, we created a towed buoy mechanism to move the node in a controlled trajectory. As shown in Fig. 9c, the backscatter transducer was mounted on a buoy made of XPS foam, using a rigid acrylic bar. A closed-loop rope-pulley system was designed to drive the buoy. A 400 W AC servo motor [40] was utilized, driving two coaxially mounted winches that contained a continuous loop of ⌀1.6 mm Kevlar rope. This mechanism can achieve a maximum speed of 1.5 m/s with a positioning accuracy of around 5 cm.

4.3.2 Mobility Environment, Dataset, and Metrics: For the evaluation of EchoRider, we created a dynamic test environment in a 4 m deep river, using a dock-mounted pulley system to achieve a travel range of 15 m for the backscatter node. We tested with speeds ranging from 0 m/s to 1.5m/s and divided our data into two range groups: 5–10 m and 10–15 m from the reader. Fig. 10a shows the different locations where we tested in the river.

4.3.3 Baseline: We compared EchoRiderto a state-of-the-art open-source baseline for underwater backscatter [21]. Across the experimental trials, the baseline used the same number of receiving hydrophones and transmitting power as EchoRider's implementation. For fairness, we focused on single-piezo backscatter nodes (not Van Atta nodes).

This work does not raise any ethical issues.

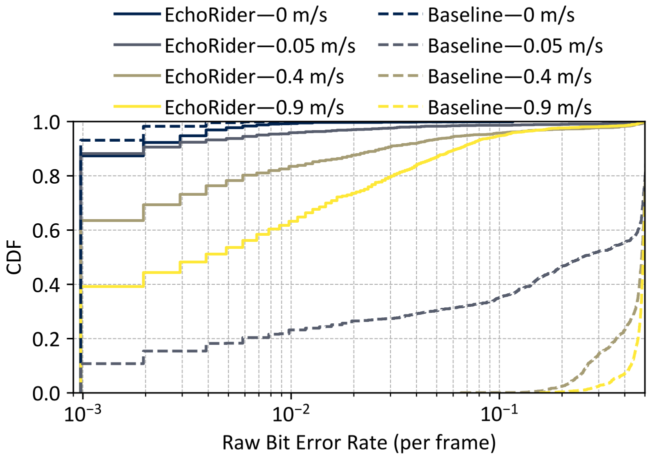


Figure 11: BER vs. Mobility. This figure plots the CDF of the raw BER of uncoded data, comparing EchoRider to a state-of-the-art baseline at different speeds.

5 Results

5.1 BER

5.1.1 Comparison with the baseline: First, to assess the uplink physical layer performance under mobile conditions, we evaluated the per-frame bit-error rate (BER) of EchoRider by testing more than 700 frames at mobility levels, as described in §4.3.[13] We compared EchoRider's decoder to the state-of-the-art baseline described in §4.3.

Fig. 11 plots the CDF distribution of BER, comparing EchoRider (solid lines) and the baseline (dotted lines) at 4 mobility levels (0 m/s, 0.05 m/s, 0.4 m/s, and 0.9 m/s). We make the following observations:

- In static settings (i.e., at zero speed), both the baseline and EchoRider maintained a low median BER (10^{-3}), which is expected as the baseline is designed for static environments.
- The BER increases for both EchoRider and the baseline at higher speeds as expected (due to the Doppler effect). However, the baseline suffers at speeds as low as 0.05 m/s (median BER of 0.17) and becomes undecodable at speeds of 0.4 m/s (median BER of 0.5). In contrast, EchoRider maintains a median BER of 4×10^{-3} at speeds up to 0.9 m/s.

These results demonstrate the benefit of EchoRider's differential decoder under mobility conditions (§3.3).

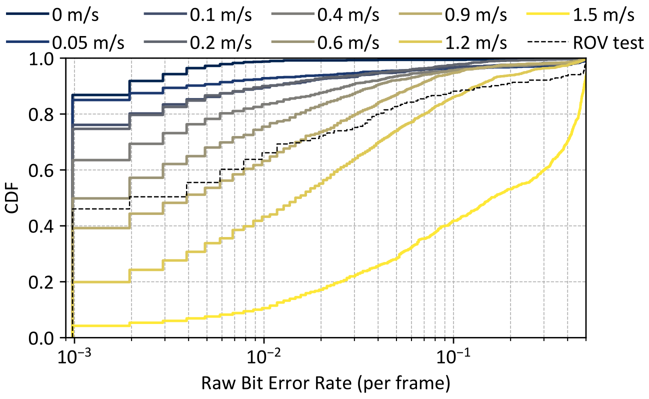


Figure 12: EchoRider BER vs speed. This figure plots the CDF of EchoRider's uncoded BER as a function of different speeds. The BER is computed per frame where each frame contains 1024 bits. The dashed line indicates the BER results from the Drone test in §6.1.

5.1.2 Impact of Mobility Level: We further evaluated the BER (per frame) performance of EchoRider as a function of different speeds, using more than 700 uplink frames at mobility levels from 0 m/s to 1.5m/s, as described in §4.3.

Fig. 12 plots the CDF of our results. We note the following:

- As the speed increases, the median BER increases, which is expected due to increased Doppler.
- At speeds up to 0.4 m/s, 80% of frames maintain a raw (i.e. without channel coding) BER below 0.01. Such BER can be decoded with a code rate of 3/4.
- At higher speeds (0.6 m/s to 1.5 m/s), the BER is higher, but the majority of the frames remain decodable. These higher error rates demonstrate the importance of our retransmission and recovery mechanisms to achieve reliable communication in mobile environments.

[13]We use the standard uncoded raw BER per frame to isolate the physical layer. This BER is defined as the number of errors in the pre-FEC decision $\tilde{d}_k$ divided by the frame length of 1024 bits.

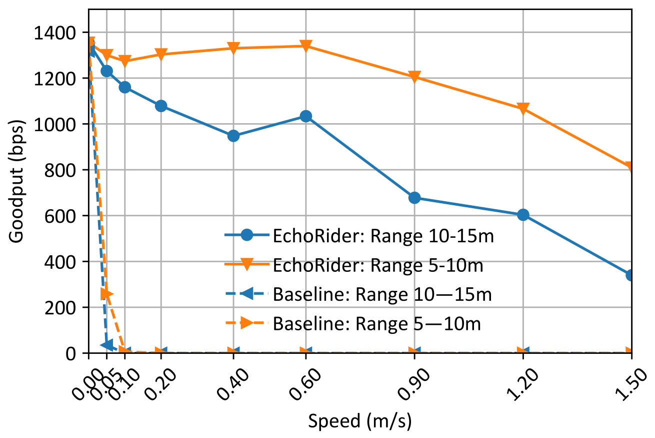


Figure 13: Goodput vs. Mobility. This figure plots the goodput as a function of mobility at different ranges. It compares the performance of EchoRider with traditional underwater backscatter.

5.2 Goodput

We further evaluated the end-to-end real-time goodput of EchoRider as a function of mobility level and communication range. The goodput is defined as the ratio between the number of correctly received uplink bits and the duration the reader queries the node for these bits. In this experiment, EchoRider employs the end-to-end retransmission protocol, attempting to reliably deliver 22 frames for each of 30 runs, which were tested at different combinations of distances and speeds. The goodput is computed from the total time the system takes to transmit the frames (including any delays caused by each NACK in real-time).

We compared the performance of EchoRider to the baseline mentioned earlier [21]. For context, the baseline transmits 22 frames per run over a fixed duration of 13.2 seconds (one downlink and one uplink). The baseline's goodput is calculated using the number of correct frames. Both EchoRider and the baseline use a 3/4-rate polar code.

Fig. 13 plots the goodput of EchoRider (solid) and the baseline (dotted) as a function of mobility level, across both close ranges (5–10 m) and far ranges (10–15 m). We make the following remarks:

- In the static case, both systems achieve full goodput at both close and far ranges.
- The goodput generally decreases at higher mobility levels for both systems, as expected under more challenging Doppler conditions.
- Despite an increase in mobility levels up to 1.5 m/s, EchoRider maintained a 60% goodput at closer ranges and 30% at farther ranges. In contrast, the baseline's goodput drops to $<$ 400 bps at speeds as low as 0.05 m/s and to near-zero at higher mobility.

This result shows the importance of EchoRider's Doppler mitigation and retransmission techniques for robust end-to-end mobile communication.

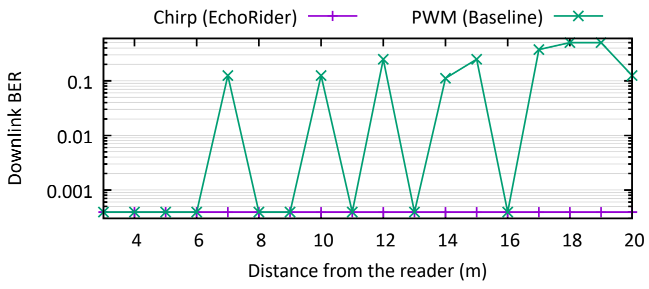


Figure 14: Downlink BER vs. Distance. This figure plots the downlink BER as a function of distance and compares the performance of EchoRider 's chirp-based downlink with the PWM downlink baseline.

5.3 Microbenchmarks

5.3.1 Downlink reliability: To validate the effectiveness of our chirp-based downlink design in EchoRider, we evaluated the downlink BER across various distances from the reader, ranging from 3 m to 20 m. For this microbenchmark, we used a transmit power of 3.2 W and transmitted a total of 2540 downlink bits at a rate of 60 bps for each range location.

We wanted to compare the performance to the standard downlink (PWM) decoder used by past underwater backscatter systems [7]. To ensure that our comparison is performed under fair channel conditions, we collected the channel impulse response at the same locations. Then, we obtained the downlink signal using a trace-driven simulation that applies the channel to a PWM signal whose carrier frequency is 23 kHz, assuming the forward-biased voltage (Vf) of 0.1 V for the decoder.

Fig. 14 plots the downlink BER for EchoRider and the baseline. EchoRider's chirp-based downlink maintains a BER of zero across all distances, demonstrating no detectable errors. In contrast, the PWM baseline fluctuated significantly in BER with increasing distance. This is due to sensitivity to multipath when transmitting a single carrier on the downlink, which suffers from nulls due to destructive interference at certain ranges. In contrast, EchoRider's chirp downlink harnesses frequency diversity, achieving a low BER across all distances.

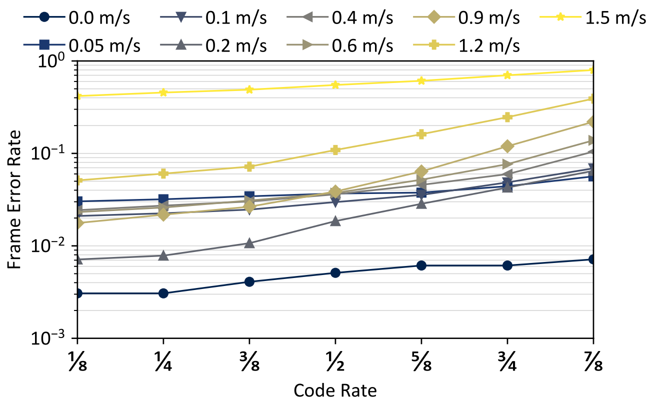


Figure 15: FER vs. Code Rate This figure plots the CDF of frame error rate (FER) at different mobility speeds and error correction code rates. The lower the code rate, the higher the redundancy.

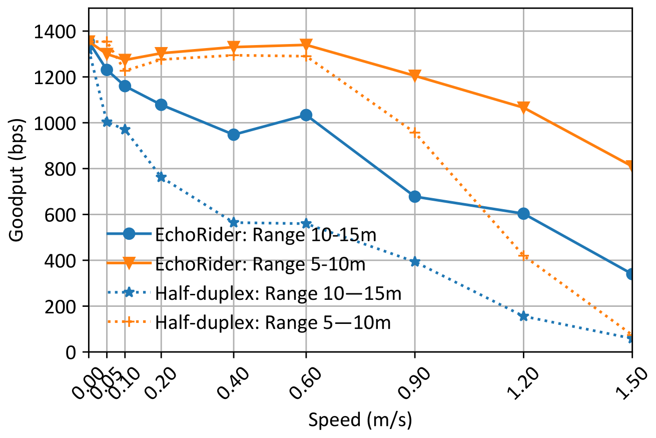


Figure 16: Impact of Retransmission. This figure plots the goodput as a function of mobility speeds for both EchoRider and an ablated implementation of EchoRider that does not use the full-duplex NACK retransmission strategy (*i.e.,* "half-duplex").

5.3.2 Impact of uplink FEC coding rate: To understand the tradeoff between robustness and efficiency, we evaluated the frame error rate (FER) performance of EchoRider using different polar coding rates. The experiments were performed by varying coding rates from 1/8 to 7/8 using the 700 frames[14] (§4.3) tested at speeds from 0.0 m/s to 1.5 m/s.

As shown in Fig. 15, we can observe that the frame error rate decreases with increased redundancy (lower coding rate) due to more robust coding. It is worth noting that at higher mobility speeds, we observe a non-zero frame error rate even at the lowest coding rates. This could be due to several reasons: loss of channel tracking, issues in frame synchronization, or errors in the equalization process. This result again demonstrates the need for link-layer recovery mechanisms in conjunction with FEC.

5.3.3 Impact of retransmission strategy: To evaluate the impact of our full-duplex NACK retransmission, we evaluate a partial implementation that incorporates only our differential decoder without the full-duplex NACK retransmission protocol ("half-duplex"). In this half-duplex scenario, since the uplink cannot be interrupted, the reader waits for the full uplink duration despite possible erroneous frames in it. We calculate the goodput from the number of frames until we encounter the first decode error. We compared the goodput of this partial implementation to the full implementation across two operational ranges: 5–10 m and 10–15 m. Fig. 16 plots the goodput as a function of speed (0.05 m/s–1.5 m/s. We make the following remarks:

- At close range and with no or lower mobility (0.05 m/s to 0.6 m/s), the partial implementation exhibited similar goodput to EchoRider, which is expected because both systems were able to cope with modest mobility using the differential-decoder-based Doppler phase cancellation strategy. However, at higher mobility levels (notably from 0.9 m/s–1.5 m/s), there was a significant

[14]For repeatability of the mobile environments, we use the 3/4-rate dataset, while utilizing the fact that Polar transform is bijective, so *any* known 1024-bit sequence can be reinterpreted as a codeword of *any* rate, with a different frozen bit assignment [41].

decline 20%–87% for the partial implementation compared to EchoRider.

- At farther ranges, EchoRider consistently outperforms the partial implementation in terms of goodput across all mobility speeds. This is because at farther ranges, the number of erroneous frames is higher, which requires frequent retransmission; however, in the half-duplex mode, more time would be wasted waiting for the uplink to complete, while EchoRider could immediately interrupt the erroneous uplink to request a retransmission without wasting any time.

6 End-to-End Demonstrations

We demonstrated EchoRider in two end-to-end applications.

6.1 Underwater Drones

We mounted the backscatter node, along with its waterproof circuit assembly, onto an underwater drone (as illustrated in Fig. 10b and 10c). Across 10 end-to-end throughput test runs and 240 frames transmitted, the node moved at a speed of ~ 0.5 m/s and maintained a distance of ~ 10 m from the reader. The results demonstrated that the node could successfully communicate with the reader, achieving an average throughput of 869 bps, and a median BER of 0.003, as shown in the dashed line in Fig. 12. These findings align with our controlled experiments.

6.2 Multi-Node Networking

We tested EchoRider in a multi-node deployment scenario, involving three backscatter nodes placed within 5–10 m around the reader. We implemented an Aloha-based collision avoidance protocol in EchoRider similar to the ISO-15693 NFC standard [42]. Fig. 17 illustrates how EchoRider employs its MAC protocol to receive data from multiple nodes. The reader initiates communication by transmitting a downlink query, which the nodes receive and respond to within predefined time slots based on their IDs. Each response comprises a guard band (silence followed by pilot symbols), a preamble, and a payload. Once the reader identifies the nodes within its vicinity, it can issue a targeted downlink request to any specific node, following the physical and link layer protocols as described in §3.

7 Related Work

(a) Underwater Backscatter: Past work on underwater backscatter has demonstrated the feasibility of the technology [7], higher throughput [33, 43], longer range [21], link budget [9], and introduced new applications in imaging [8] and localization [44]. However, all past work focused on static environments. EchoRider builds on this past literature and is the first to enable reliable underwater backscatter under mobility.

(b) Reliable Underwater Networking: Various techniques have been developed to address the challenging conditions of the underwater communication channel, particularly the high delay spread from multipath reflections and Doppler effects stemming from mobility relative to the slow propagation speed of acoustic waves. However, these techniques were developed for classical underwater acoustic communications [27, 45–48] and are too power-hungry for energy-constrained underwater backscatter nodes. EchoRider builds on these past approaches and adapts them to the energy-constrained underwater backscatter problem. Notably, although high mobility (5–15 m/s) can be addressed using advanced TX waveforms like OFDM [16] and chirp [49], these waveforms are infeasible to implement due to the inherent binary switching of backscatter. Therefore, we focus on single-carrier FM0 uplink and improve its mobility resilience with a mixture of link and physical-layer techniques.

(c) Error Recovery Schemes: Past research has focused on different error recovery methods to counter the unique challenges (*i.e.,* poor quality, asymmetric connectivity, long propagation delays, *etc.*) in underwater acoustic channels. For example, researchers have worked on the application of Forward Error Correction (FEC) to deal with low received SNRs [50–55], the use of rateless codes for ACK-free correction [56], and ARQ schemes (ACK/NACK) to handle packet loss [57–59]. EchoRider builds on these methods, making the realization that NACK-based protocols are ideal for underwater backscatter nodes to minimize overhead and optimize efficiency. Moreover, its architecture enables underwater backscatter nodes to decode downlink NACKs despite their full-duplex (and energy-constrained) nature.

(d) Chirp Spread-Spectrum and LoRa: EchoRider is related to past work on chirps – including in-air and underwater – and introduces a novel approach for low-power decoding by applying direct sub-Nyquist bandpass sampling to chirp signals. While previous studies have recognized the benefits of chirp modulation for robust communication in high-mobility underwater channels [49, 60], their decoders were more power-hungry than EchoRider's. This was acceptable in their contexts, as their systems were not as power-constrained as our underwater backscatter nodes. Other efforts have explored low-power downlink chirp reception techniques for RF systems [61–63], but these are inherently vulnerable to Doppler effects due to their reliance on mixing tones, making them unsuitable for underwater mobile environments. In contrast, EchoRider's approach of directly (sub-)sampling chirps would be impractical for RF-based systems (including LoRa), because the much higher center frequency in RF would necessitate a prohibitively high Q-factor for anti-aliasing (e.g., 2.4 GHz / 250 kHz $\approx$ 9600).

8 Conclusion

EchoRider represents a significant advance in low-power underwater networking by enabling reliable piezo-acoustic backscatter under mobility. Our results show that EchoRider achieves typical underwater acoustic communication throughput in scenarios where either the backscatter node or the access point (i.e., reader) is mobile. This capability paves the way for underwater drones or surface vehicles to query and collect data from long-term deployed backscatter sensors as they approach them on the ocean floor or riverbed—while preserving the sensors' battery life. Furthermore, our findings demonstrate that EchoRider's techniques—chirp-based downlink and a NACK-based full-duplex retransmission protocol—also improve reliability, efficiency, and goodput in static deployments. As this research evolves, it would be valuable to further increase throughput by exploring more complex modulation

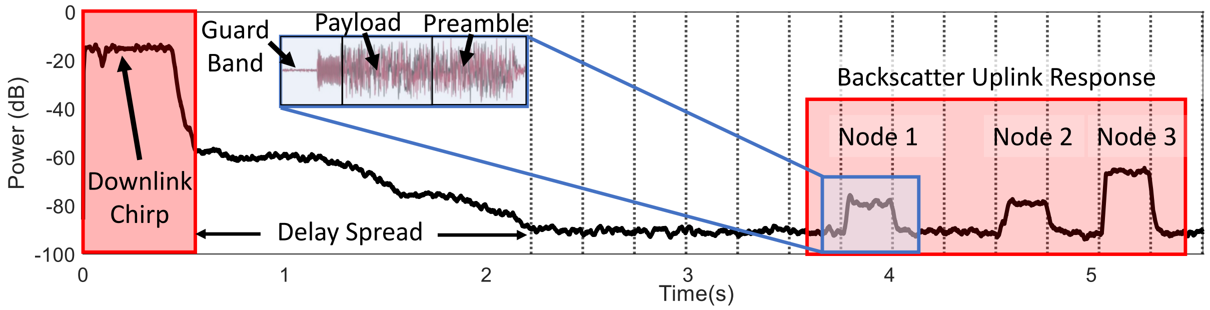


Figure 17: EchoRider MAC Protocol. The figure shows how EchoRider utilizes its MAC protocol to resolve the Unique ID from multiple nodes. The reader transmits a downlink query, which the nodes receive and respond to in 16 predefined time slots (shown as dotted lines) determined by the first 4 bits of their assigned IDs [42]. Each response consists of a guard band (silence followed by pilot symbols), a preamble of 256 bits, and a payload encoding the 64-bit ID in a 256-bit Polar codeword.

schemes, and to boost the communication range by using more complex underwater backscatter nodes [21].

Acknowledgments

We thank the anonymous reviewers, our shepherd Haitham Hassanieh, and the Signal Kinetics group for their feedback. This work was supported by the Office of Naval Research and the MIT Media Lab. It was also partially supported by the National Science Foundation under grant 2308901, and by the National Research Foundation of Korea (NRF), funded by the Ministry of Science and ICT (MSIT), under grant RS-2024-00464269.

References

[1] Sustainable fisheries and aquaculture for food security and nutrition. https://www.fao.org/3/i3844e/i3844e.pdf.

[2] Dana R Yoerger, Annette F Govindarajan, Jonathan C Howland, Joel K Llopiz, Peter H Wiebe, Molly Curran, Justin Fujii, Daniel Gomez-Ibanez, Kakani Katija, Bruce H Robison, et al. A hybrid underwater robot for multidisciplinary investigation of the ocean twilight zone. *Science Robotics*, 6(55):eabe1901, 2021.

[3] Alexander G Rumson. The application of fully unmanned robotic systems for inspection of subsea pipelines. *Ocean Engineering*, 235:109214, 2021.

[4] Jörg Kalwa, A Pascoal, Pere Ridao, A Birk, M Eichhorn, L Brignone, M Caccia, J Alves, and RS Santos. The european r&d-project morph: Marine robotic systems of self-organizing, logically linked physical nodes. *IFAC Proceedings Volumes*, 45(27):226–231, 2012.

[5] John G Proakis, Ethem M Sozer, Joseph A Rice, and Milica Stojanovic. Shallow water acoustic networks. *IEEE communications magazine*, 39(11):114–119, 2001.

[6] John Heidemann, Milica Stojanovic, and Michele Zorzi. Underwater sensor networks: applications, advances and challenges. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 370(1958):158–175, 2012.

[7] Junsu Jang and Fadel Adib. Underwater backscatter networking. In *Proceedings of the ACM Special Interest Group on Data Communication*, pages 187–199, 2019.

[8] Sayed Saad Afzal, Waleed Akbar, Osvy Rodriguez, Mario Doumet, Unsoo Ha, Reza Ghaffarivardavagh, and Fadel Adib. Battery-free wireless imaging of underwater environments. *Nature communications*, 13(1):1–9, 2022.

[9] Waleed Akbar, Ahmed Allam, and Fadel Adib. The underwater backscatter channel: Theory, link budget, and experimental validation. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*, pages 1–15, 2023.

[10] Ananya Bhardwaj, Ahmed Allam, Alper Erturk, and Karim G Sabra. Ultrasound-powered wireless underwater acoustic identification tags for backscatter communication. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, 2023.

[11] Alper Bereketli. Interference-free source deployment for coverage in underwater acoustic backscatter networks. *Peer-to-Peer Networking and Applications*, 15(3):1577–1594, 2022.

[12] Jiakang Zheng, Jiayi Zhang, Emil Björnson, Zhetao Li, and Bo Ai. Cell-free massive mimo-ofdm for high-speed train communications. *IEEE Journal on Selected Areas in Communications*, 40(10):2823–2839, 2022.

[13] Yuanjie Li, Qianru Li, Zhehui Zhang, Ghufran Baig, Lili Qiu, and Songwu Lu. Beyond 5g: Reliable extreme mobility management. In *Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication*, pages 344–358, 2020.

[14] Ryota Yoshizawa and Hideki Ochiai. Energy efficiency improvement of coded ofdm systems based on papr reduction. *IEEE Systems Journal*, 11(2):717–728, 2015.

[15] Milica Stojanovic and James Preisig. Underwater acoustic communication channels: Propagation models and statistical characterization. *IEEE communications magazine*, 47(1):84–89, 2009.

[16] Baosheng Li, Shengli Zhou, Milica Stojanovic, Lee Freitag, and Peter Willett. Multicarrier communication over underwater acoustic channels with nonuniform doppler shifts. *IEEE Journal of Oceanic Engineering*, 33(2):198–209, 2008.

[17] Reza Ghanaatian, Orion Afisiadis, Matthieu Cotting, and Andreas Burg. Lora digital receiver analysis and implementation. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1498–1502. IEEE, 2019.

[18] Van-Atta Backscatter Github Repository. https://github.com/signalkinetics/vab.

[19] BlueROV2. Affordable and capable underwater rov. https://bluerobotics.com/store/rov/bluerov2/, 2024.

[20] Schmidt Ocean Institute. Rov faqs, 2024. Accessed: 2024-06-03.

[21] Aline Eid, Jack Rademacher, Waleed Akbar, Purui Wang, Ahmed Allam, and Fadel Adib. Enabling long-range underwater backscatter via van atta acoustic networks. In *Proceedings of the ACM SIGCOMM 2023 Conference*, pages 1–19, 2023.

[22] Nicholas R Rypkema, Erin M Fischell, and Henrik Schmidt. One-way travel-time inverted ultra-short baseline localization for low-cost autonomous underwater vehicles. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4920–4926. IEEE, 2017.

[23] Fadel Adib, Zachary Kabelac, Dina Katabi, and Robert C. Miller. 3d tracking via body radio reflections. In *Usenix NSDI*, 2014.

[24] Sayed Saad Afzal, Waleed Akbar, Osvy Rodriguez, Mario Doumet, Unsoo Ha, Reza Ghaffarivardavagh, and Fadel Adib. Battery-free wireless imaging of underwater environments. *Nature Communications*, 13(1):5546, Sep 2022.

[25] Haitham Hassanieh, Fadel Adib, Dina Katabi, and Piotr Indyk. Faster gps via the sparse fourier transform. In *Proceedings of the 18th annual international conference on Mobile computing and networking*, pages 353–364, 2012.

[26] Mathieu Xhonneux, Orion Afisiadis, David Bol, and Jérôme Louveaux. A low-complexity lora synchronization algorithm robust to sampling time offsets. *IEEE Internet of Things Journal*, 9(5):3756–3769, 2022.

[27] Milica Stojanovic. An adaptive algorithm for differentially coherent detection in the presence of intersymbol interference. *IEEE Journal on selected areas in communications*, 23(9):1884–1890, 2005.

[28] European Telecommunications Standards Institute. 5G; NR; Multiplexing and Channel Coding. ETSI TS 138 212 V16.2.0, ETSI, July 2020.

[29] Ido Tal and Alexander Vardy. List decoding of polar codes. *IEEE transactions on information theory*, 61(5):2213–2226, 2015.

[30] ALine Eid, Jack Rademacher, Waleed Akbar, Purui Wang, Ahmed Allam, and Fadel Adib. Enabling long-range underwater backscatter via van atta acoustic networks. In *Proceedings of the ACM SIGCOMM 2023 Conference*, 2023.

[31] Tri state gate 74AXP1G125. https://assets.nexperia.com/documents/data-sheet/74AXP1G125.pdf.

[32] Tektronix. Dmm6500 6½-digit graphical touchscreen multimeter. https://www.tek.com/en/products/keithley/digital-multimeter/dmm6500, n.d.

[33] Reza Ghaffarivardavagh, Sayed Afzal, Osvy Rodriguez, and Fadel Adib. Ultra-wideband underwater backscatter via piezoelectric metamaterials. In *Proceedings of the ACM Special Interest Group on Data Communication*, 2020.

[34] KEMET Electronics Corporation. Metallized polypropylene film EMI suppression capacitors R46, class X2, 310 VAC, 125°C. https://content.kemet.com/datasheets/KEM_F3122_R46_X2_310_125C.pdf.

[35] TDK PM-50-39 ferrite core. https://www.tdk-electronics.tdk.com/inf/80/db/fer/pm_50_39.pdf.

[36] TI tpa3245 power amplifier. https://www.ti.com/lit/gpn/tpa3245.

[37] ESS ES9018 DAC chip. https://www.esstech.com/wp-content/uploads/2021/02/ES9018S_Datasheet_v2.2.pdf.

[38] H1C Hydrophone. https://www.aquarianaudio.com/h1c-hydrophone.html.

[39] CS5381 ADC chip. https://www.cirrus.com/products/cs5381/.

[40] T6 Series 400W AC Servo Motor Kit 3000RPM 1.27Nm 17-bit Encoder IP65 – STEPPERONLINE. https://www.omc-stepperonline.com/t6-series-400w-ac-servo-motor-kit-3000rpm-1-27nm-17-bit-encoder-ip65-t6-rs400h2a3-m17s, 2023.

[41] Erdal Arikan. Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Transactions on information Theory*, 55(7):3051–3073, 2009.

[42] ISO. Identification cards — contactless integrated circuit(s) cards - vicinity cards — part 3: Anticollision and transmission protocol. http://olmicrowaves.com/menucontents/designsupport/rfid/ISO15693-3.pdf, 1999.

[43] Sayed Saad Afzal, Reza Ghaffarivardavagh, Waleed Akbar, Osvy Rodriguez, and Fadel Adib. Enabling higher-order modulation for underwater backscatter communication. In *Global Oceans 2020: Singapore–US Gulf Coast*, pages 1–6. IEEE, 2020.

[44] Reza Ghaffarivardavagh, Sayed Saad Afzal, Osvy Rodriguez, and Fadel Adib. Underwater backscatter localization: Toward a battery-free underwater gps. In *Proceedings of the 19th ACM Workshop on Hot Topics in Networks*, HotNets '20, page 125–131, New York, NY, USA, 2020. Association for Computing Machinery.

[45] Milica Stojanovic, Josko A Catipovic, and John G Proakis. Phase-coherent digital communications for underwater acoustic channels. *IEEE journal of oceanic engineering*, 19(1):100–111, 1994.

[46] Mark Johnson, Lee Freitag, and Milica Stojanovic. Improved doppler tracking and correction for underwater acoustic communications. In *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 575–578. IEEE, 1997.

[47] Rolf Weber, Andreas Waldhorst, Florian Schulz, and JF Bohme. Blind receivers for msk signals transmitted through shallow water. In *MTS/IEEE Oceans 2001. An Ocean Odyssey. Conference Proceedings (IEEE Cat. No. 01CH37295)*, volume 4, pages 2183–2190. IEEE, 2001.

[48] Trym H Eggen, Arthur B Baggeroer, and James C Preisig. Communication over doppler spread channels. part i: Channel and receiver presentation. *IEEE journal of oceanic engineering*, 25(1):62–71, 2000.

[49] Chengbing He, Jianguo Huang, Qunfei Zhang, and Kaizhuo Lei. Reliable mobile underwater wireless communication using wideband chirp signal. In *2009 WRI International Conference on Communications and Mobile Computing*, volume 1, pages 146–150. IEEE, 2009.

[50] EM Sozer, John G Proakis, and F Blackmon. Iterative equalization and decoding techniques for shallow water acoustic channels. In *MTS/IEEE Oceans 2001. An Ocean Odyssey. Conference Proceedings (IEEE Cat. No. 01CH37295)*, volume 4, pages 2201–2208. IEEE, 2001.

[51] Tommy Oberg, Bernt Nilsson, Niten Olofsson, Magnus Lundberg Nordenvaad, and Erland Sangfelt. Underwater communication link with iterative equalization. In *OCEANS 2006*, pages 1–6. IEEE, 2006.

[52] Jun Won Choi, Thomas J Riedl, Kyeongyeon Kim, Andrew C Singer, and James C Preisig. Adaptive linear turbo equalization over doubly selective channels. *IEEE journal of oceanic engineering*, 36(4):473–489, 2011.

[53] Atulya Yellepeddi and James C Preisig. Adaptive equalization in a turbo loop. *IEEE Transactions on Wireless Communications*, 14(9):5111–5122, 2015.

[54] Shengming Jiang. On reliable data transfer in underwater acoustic networks: A survey from networking perspective. *IEEE Communications Surveys & Tutorials*, 20(2):1036–1055, 2018.

[55] Yuriy Zakharov, Benjamin Henson, Roee Diamant, Yuan Fei, Paul D Mitchell, Nils Morozs, Lu Shen, and Tim C Tozer. Data packet structure and modem design for dynamic underwater acoustic channels. *IEEE Journal of Oceanic Engineering*, 44(4):837–849, 2019.
[56] Mandar Chitre and Mehul Motani. On the use of rate-less codes in underwater acoustic file transfers. In *OCEANS 2007-Europe*, pages 1–6. IEEE, 2007.
[57] Mingsheng Gao, W-S Soh, and Meixia Tao. A transmission scheme for continuous arq protocols over underwater acoustic channels. In *2009 IEEE International Conference on Communications*, pages 1–5. IEEE, 2009.
[58] Mandar Chitre and Wee-Seng Soh. Reliable point-to-point underwater acoustic data transfer: To juggle or not to juggle? *IEEE Journal of Oceanic Engineering*, 40(1):93–103, 2014.
[59] Salvador Climent, Nirvana Meratnia, and Juan Vicente Capella. Impact analysis of different scheduling and retransmission techniques on an underwater routing protocol. In *Proceedings of the 6th International Workshop on Underwater Networks*, pages 1–8, 2011.
[60] Zhenyu Jia, Wenjun Zheng, and Fei Yuan. A two-dimensional chirp-mfcsk modulation method for underwater lora system. *IEEE Internet of Things Journal*, 9(23):24388–24397, 2022.
[61] Xiuzhen Guo, Longfei Shangguan, Yuan He, Nan Jing, Jiacheng Zhang, Haotian Jiang, and Yunhao Liu. Saiyan: Design and implementation of a low-power demodulator for {LoRa} backscatter systems. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*, pages 437–451, 2022.
[62] Yihang Song, Li Lu, Jiliang Wang, Chong Zhang, Hui Zheng, Shen Yang, Jinsong Han, and Jian Li. {μMote}: enabling passive chirp de-spreading and {μW-level}{Long-Range} downlink for backscatter devices. In *20th USENIX symposium on networked systems design and implementation (NSDI 23)*, pages 1751–1766, 2023.
[63] Pol Maistriaux, Marco Gonzalez, Jérôme Louveaux, and David Bol. Leveraging a digital chirp spread spectrum detector for lpwan wake-up receivers. In *2024 14th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP)*, pages 638–643. IEEE, 2024.
[64] Zhiyu Ru, Eric A. M. Klumperink, and Bram Nauta. Discrete-time mixing receiver architecture for rf-sampling software-defined radio. *IEEE Journal of Solid-State Circuits*, 45(9):1732–1745, 2010.

APPENDIX

A Discrete-time mixing

In this section we derive the constraints and detail the frequency parameters used in sub-Nyquist mixing presented in 3.1.2. This technique is known as discrete-time mixing in RFIC literature [64].

We obtain the I/Q samples $r_{\text{IQ}}(t)$ with the carrier signal at f_c multiplied by the input signal $y(t)$.

$$r_{\text{IQ}}(t) = y(t)c(t) \tag{1}$$

In discrete time mixing, the signal and carrier is sampled at F_s, i.e. for k-th ADC sample, $t = t[k] = k/f_s$. Now let

$$m = \frac{2f_c}{f_s} - \frac{1}{2} \tag{2}$$

We substitute f_c with m in the sampled carrier expression used in Eqn. 1:

$$\begin{aligned} c(t[k]) &= c(k/f_s) \\ &= \exp(j2\pi(\frac{f_c}{f_s})k) \\ &= \exp(j(m\pi + \frac{\pi}{2})k) \\ &= (-1)^{mk}\exp(\frac{jk\pi}{2}) \end{aligned} \tag{3}$$

Therefore, when m is an integer, the sampled carrier sequence has the simplest form of $[1, \pm 1j, -1, \mp 1j, \cdots]$. This form enables us to obtain each IQ sample (at $f_{\text{IQ}} = f_s/4$) by combining every 4 ADC samples:

$$\begin{aligned} \Re\{r_{IQ}[k]\} &= y(t[4k]) - y(t[4k+2]) \\ \Im\{r_{IQ}[k]\} &= (-1)^m(y(t[4k+1]) - y(t[4k+3])) \end{aligned}$$

We chose a sampling frequency $f_s = 12.8$ kHz and $m = 3$, which is available from the low power oscillator to achieve a high bandwidth ($f_s/4 = 3.2$ kHz) available while minimizing the power consumption. Then, by setting $m = 3$ in Eqn. 2, the center frequency of chirp $F_c = \frac{m+1/2}{2} \cdot F_s = \boxed{7/4F_s} = 22.4$ kHz, which falls in the resonance range of the transducer. We detail the interrelation of these frequencies in Tab. 2.

Name	Value	Relation
f_{osc}	3072 kHz	Hardware frequency
DIV	240	Sampling clock divider
f_s	12.8 kHz	$=f_{osc}/\text{DIV}$
f_{IQ}	3.2 kHz	$= Bandwidth = f_s/4$
f_c	22.4 kHz	$=7f_s/4$
$[f_{lo}, f_{hi}]$	[20.8, 24] kHz	$=[f_c - f_{IQ}/2, f_c + f_{IQ}/2]$
N	64	FFT size
f_{sym}	50 bps	$=f_{IQ}/N$

Table 2: Downlink parameters.

B SOF detection algorithm

In this section, we detail EchoRider's downlink SOF synchronization algorithm's pesudocode. Starting at the lowest layer, we first define the low-level dechirping routine:

```
DECHIRP(samples, startIdx, isUp)
1   Allocate $\varphi[0..N-1]$ // Per-sample reference phase
2   Allocate $H[0..N-1]$ // Dechirped signal
3   if isUp
4       $m = -2\pi$ // mix with conjugate
5   else
6       $m = 2\pi$
7   for $i = 0$ to $N-1$
8       $\varphi[i] = m \times (i/N - 0.5)$
9       if $i > 0$
10          $\varphi[i] = \varphi[i] + \varphi[i-1]$
11      $H[i] = samples[startIdx + i] \times \exp(j\varphi[i])$
12  $H = \text{FFT}(H)$
13  return $H$
```

It's worth noting the following:

• The constant N as FFT size. We use $N = 64$, which corresponds to a bit rate of 3200/64 = 50bps (Tab.2).
• We pass a boolean *isUp* to indicate the polarity of the chirp, to detect both the downchirp part and upchirp part of the SOF. The multiplier m is the *conjugate* of the chirp being sent, which is why the upchirps use $m = -2\pi$ (decreasing frequency), and downchirps use $m = 2\pi$ (increasing frequency).
• For brevity, we use linear indexing of an infinitely-long buffer *samples*; in firmware implementation however, we use a circular buffer, keeping only recent samples.
• For further speedup in implementation, the chirp waveform $\exp(j\phi[i])$ is precomputed. We also perform memoization for dechirp results, because adjacent COHERENTCOMBINEENERGY invocations would dechirp the same block both as H_{curr} and as H_{prev}.

To handle self-interference, we define the coherent combining routine. The combine process looks for the phase at the peak bin, defined by *refIdx* (line 14), and corrects the phase difference φ between adjacent symbols (Fig. 6). It's worth noting that we will use this routine for both upchirps and downchirps, and specifically for the second downchirp in the SOF waveform (*isConvertingBit1* == True), it has a frequency shift of N/2 bins, requiring a circular shift before detection. This frequency shift helps resolve one-symbol alignment (discussed at the end of this section).

```
CoherentCombineEnergy(samples, startIdx, refIdx, isUp, isConvertingBit1)
1  // Global constant $N$ (FFT size) is used
2  $H_{curr}$ = Dechirp(samples, startIdx, isUp)
3  $H_{prev}$ = Dechirp(samples, startIdx $- N$, isUp)
4  if isConvertingBit1
5      // Circularly shift $H_{curr}$ by $N/2$ bins,
6      // to convert 2nd downchirp '1' into '0'
7      Allocate $temp[0..N-1]$
8      for $i = 0$ to $N/2 - 1$
9          $temp[i] = H_{curr}[i + N/2]$
10     for $i = N/2$ to $N - 1$
11         $temp[i] = H_{curr}[i - N/2]$
12     $H_{curr} = temp$
13
14 $\varphi = H_{curr}[refIdx] \times H^{*}_{prev}[refIdx]$
   // Phase correction coefficient from refIdx
15 $\varphi = \varphi / |\varphi|$
16 for $i = 0$ to $N - 1$
17     $H_{curr}[i] = H_{curr}[i] + \varphi \times H_{prev}[i]$
18 return $|H_{curr}|^2$
```

Next we define routine for the generating SNR metric from the per-bin energy. It tolerates a Doppler frequency drift of L bins (a small number, *e.g.*, 4, in contrast to $N/2 = 32$ bins that differentiate bit '0' and '1'): For each position *i*, *eSeg* represents the segment sum of $\pm L$ neighbor bins' energy, so it reflects the peak value even if the peak is off the desired location. We give the direct definition below, but actual implementation optimizes it with the prefix sum technique. The output *dop* is *eSeg* normalized by the total energy *eTotal*, representing an SNR value independent of absolute signal strength differences.

```
NormalizeDoppler(e, L)
1  Allocate $dop[0..N-1]$
2  $eTotal = \sum_{j=0}^{N-1} e[j]$
3  for $i = 0$ to $N - 1$
4      $eSeg = \sum_{j=-L}^{L} e[(i + j) \bmod N]$
5      $dop[i] = \frac{eSum}{eTotal}$
6  return dop
```

Finally, we can define the SOF detection algorithm. To detect the SOF under self interference, it is necessary to perform CoherentCombine Energy on both upchirps and downchirps. The detection starts with no prior assumption for upchirps timing, therefore, for the initial upchirps, the maximum bin *eUpArgMax* is used, and the threshold condition is there is *any* position *j* such that $dopUp[j] > \theta$ (line 11). This criterion is loose so it may generate false or misaligned detections, which would be removed by the downchirp filtering.

In the downchirp part, it is required that the peak position is the *same* as previous upchirps (which is very unlikely for random noise signal). Therefore, once the starting sample moves to *posDown* (determined by the frequency difference *eUpArgMax*), no further offset is allowed, and both the *refIdx* and index for *dopDown* are 0.

It's important to note that the frequency difference *eUpArgMax* is subject to wrapping when the signal is noisy and starting index *i* is close to actual symbol boundary, causing the *posDown* to be one-symbol misaligned. When detecting the downchirp, we ensure such result is rejected by offsetting the final downchirp by $N/2$ bins (Fig. 4) in the SOF waveform.

To match this offset during reception, the '1' bins with FFT peak at $N/2$ are circularly shifted by $N/2$ before combining (CoherentCombineEnergy lines 4-12). Under normal condition, this ensures the final downchirp's offset are shifted back and the energy peaks at bin 0. But in case of misalignment, this technique results in the peak energy being found at $N/2$ instead of 0, thus $dopDown[0] > \theta$ will not hold (line 16).

```
DetectSof(samples, L, θ)
1  $i = 0$
2  Allocate $dopUp[0..N-1]$
3  Allocate $dopDown[0..N-1]$
4  repeat
5      // Upchirps
6      hUp = Dechirp(samples, i, True)
7      $eUpArgMax = \arg\max_j(|hUp[j]|^2)$
8      eUpCombine = CoherentCombineEnergy(samples, i, eUpArgMax, False, False)
9      dopUp = NormalizeDoppler(eUpCombine)
10     $dopUpArgMax = \arg\max_j(|dopUp[j]|^2)$
11     if $dopUp[dopUpArgMax] > \theta$
12         // Downchirps
13         $posDown = i + 2N - eUpArgMax$
14         eDownCombine = CoherentCombineEnergy(samples, posDown, 0, True, True)
15         dopDown = NormalizeDoppler(eDownCombine)
16         if $dopDown[0] > \theta$
17             return posDown
18     $i = i + N$
```