Demo: sd-ahoi — An Open Source Software-Defined Modem for Acoustic Underwater Communication

Julian Schuler
julian.schuler@tuhh.de
Hamburg University of Technology
Hamburg, Germany

Konstantin Gosch
konstantin.gosch@tuhh.de
Hamburg University of Technology
Hamburg, Germany

Bernd-Christian Renner
christian.renner@tuhh.de
Hamburg University of Technology
Hamburg, Germany

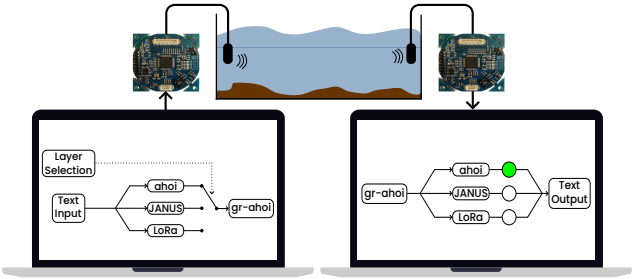

Figure 1: Simplified demonstration setup, showing the data flow between the two hosts.

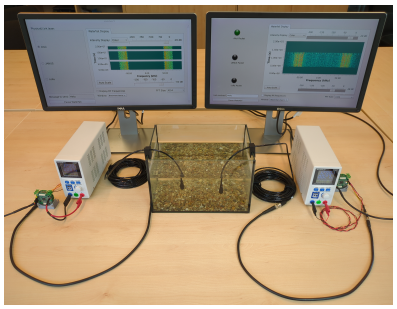

Figure 2: Picture of the demonstration setup.

Abstract

Software-Defined Acoustic Modems (SDAMs) are powerful tools for developing, prototyping, evaluating, and comparing novel physical, link, layer and network layer protocols tailored to the specific challenges of acoustic underwater communication. We introduce sd-ahoi, an accessible open source SDAM compatible with GNU Radio and based on the hardware of the open source ahoi modem. We present our design goals and provide a detailed description of the architecture of sd-ahoi. In our demonstration, we transmit text messages between two modems using sd-ahoi while interactively switching between different physical and link layer implementations at runtime.

CCS Concepts

• **Hardware** → **Communication hardware, interfaces and storage**; • **Networks** → *Network components.*

Keywords

Software-Defined Acoustic Modem; Acoustic Underwater Communication; ahoi Modem; GNU Radio

1 Introduction

Acoustic underwater communication is fundamental for underwater wireless sensor networks [6], remote control [1] and communication with Micro Autonomous Underwater Vehicle (μAUV) swarms [2]. However, the challenging underwater communication channel—characterized by limited bandwidth, strong multipath, and long propagation delays—makes efficient physical, link, and network layer protocols an active research topic [8].

Current research uses a variety of different acoustic modems, impeding reproducibility and comparability with existing protocols. Software-Defined Acoustic Modems (SDAMs) abstract from the specific hardware constraints of a modem, simplifying protocol development, comparison, and evaluation. Furthermore, SDAMs enable prototyping of sophisticated but resource-consuming channel-adaptation mechanisms for battling the challenging effects of the acoustic underwater communication channel.

While commercially available SDAMs exist, they are prohibitively expensive (often several thousands of Euros). We improve accessibility by converting the ahoi modem [3, 5] into sd-ahoi, an open source SDAM, which uses GNU Radio for implementing physical and link layers on a host computer. This enables rapid prototyping and easy runtime adjustments without being constrained by limited processing power and memory of embedded hardware, while not requiring low-level programming knowledge. Our sd-ahoi implementation is fully compatible to the ahoi modem by using a custom high-speed modem-host protocol. Our demonstration combines sd-ahoi with three runtime-selectable physical and link layer implementations for transmitting text messages between two modems interactively.

2 Architecture

We set out to address the following goals for our sd-ahoi implementation:

- Hardware compatibility to the ahoi modem,
- inclusion in the stock ahoi modem firmware, and
- ability to use the same physical and link layer implementations in simulation and in the real world.

The number of existing ahoi modems in both our institute and other research facilities as well as their commercial availability motivates compatibility. The second and third goal strive to improve the developer experience and make sd-ahoi as accessible as possible, without requiring flashing different firmwares for the SD mode. Another major motivation for the third goal is the desire to simulate protocols and validate these simulation results via real-world experiments without being affected by the side effects of a protocol reimplementation in firmware.

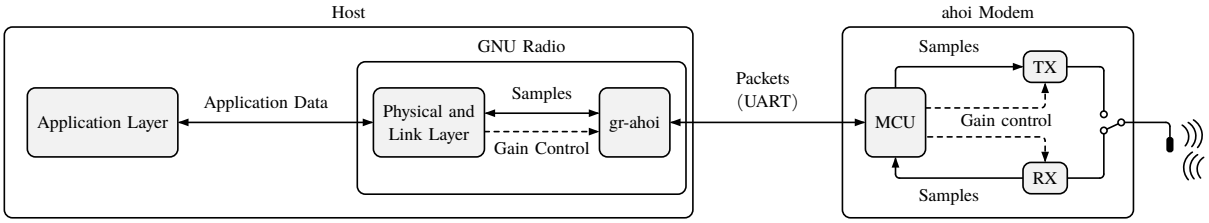

Figure 3: Architecture of sd-ahoi, showing the data flow between the different hardware and software components.

Figure 3 shows sd-ahoi's architecture and the interaction of hard- and software components, with our GNU Radio module *gr-ahoi* at its heart. An application layer on the host exchanges data with a GNU Radio instance, which implements the physical and link layers as flow graphs, interacting with the application layer and modem hardware via gr-ahoi. This architecture addresses our third goal, since the physical and link layers may be developed and simulated within GNU Radio fully independent of the rest of sd-ahoi.

The gr-ahoi module handles the communication with the ahoi modem, providing two intuitive sink and source blocks for streaming samples to and from the modem, and exposing gain control for the modem's transmitter and receiver circuitry. We implemented a custom high-speed protocol for communication between gr-ahoi and the modem over the existing Universal Asynchronous Receiver Transmitter (UART) interface. On the modem, we extended the existing firmware by a software-defined mode, which reconfigures the interface and implements the host-to-modem communication. This mode is inactive at boot, until explicitly activated by gr-ahoi. Hence, including it in the stock firmware is backwards compatible with conventional usage, which addresses our second goal. In the software-defined mode, the modem streams sample packets and gain control data between modem and host.

The main implementation challenge for streaming the sample data is the limited bandwidth of the USB-to-UART converter. Our modem-host protocol, however, is fast enough to stream passband samples at the full sampling rate of 192 kHz from and to the ahoi modem. Our sd-ahoi thus makes the full bandwidth of the ahoi modem accessible to the host.

3 Demonstration Setup and Evaluation

Figure 1 and Figure 2 show our demonstration setup, using sd-ahoi to interactively send text messages over a 10 cm underwater channel in a small aquarium. The transmitter and receiver each consist of a computer, an ahoi modem and an Aquarian AS-1 hydrophone. At the transmitter, the user can select between three physical and link layer protocols: The default ahoi modem, the JANUS standard [4], and a LoRa [7] implementation. The transmitter sends a message every second, allowing the user to change the transmitted text at will.

At the receiver, all three physical and link layer implementations process the received samples. As soon as one detects a valid packet, a GUI element indicates which protocol succeeded. The receiver further displays the text of the received message. Both transmitter and receiver visualize the spectrum of the outgoing and incoming sample streams over time using waterfall plots, providing comprehensible insight into protocol's spectral characteristics.

During testing, all three physical and link layer implementations were able to transmit and receive text messages at high packet reception rates. While the ahoi modem transmitter and receiver hardware is capable of communicating over several hundred meters, our small scale demonstration portrays the benefits and properties of sd-ahoi. The demonstration further illustrates the processing chain and interfacing with sd-ahoi, while allowing users to interactively investigate the properties of the underwater channel, e.g., by introducing waves and obstacles.

4 Conclusion

We introduced sd-ahoi, an open source SDAM based on the ahoi modem, enabling fast prototyping and comparison of physical and link layer protocols in underwater networks. By using the existing modem-host interface, sd-ahoi remains fully compatible with existing ahoi modems. On the host side, we developed a GNU Radio module providing intuitive blocks for configuring the modem and transmitting and receiving sample data. With our implementation, we were able to achieve all three of our design goals. Our demonstration provides an interactive experiment highlighting the flexibility and real-world applicability of sd-ahoi for implementing and comparing different physical and link layer protocols.

References

[1] Filippo Campagnaro, Alberto Signori, and Michele Zorzi. 2020. Wireless Remote Control for Underwater Vehicles. *Journal of Marine Science and Engineering* 8, 10 (Sept. 2020). doi:10.3390/jmse8100736

[2] Saverio Iacoponi, Godfried Jansen Van Vuuren, Gaspare Santaera, Nikita Mankovskii, Igor Zhilin, Federico Renda, Cesare Stefanini, and Giulia De Masi. 2022. H-SURF: Heterogeneous Swarm of Underwater Robotic Fish. In *OCEANS 2022, Hampton Roads*. 1–5. doi:10.1109/OCEANS47191.2022.9977253

[3] Institute for Autonomous Cyber-Physical Systems (Hamburg University of Technology). 2026. Ahoi · GitLab. https://collaborating.tuhh.de/e-24/public/ahoi.

[4] John Potter, João Alves, Dale Green, Giovanni Zappa, Ivor Nissen, and Kim McCoy. 2014. The JANUS Underwater Communications Standard. In *2014 Underwater Communications and Networking (UComms)*. 1–4. doi:10.1109/UComms.2014.7017134

[5] Bernd-Christian Renner, Jan Heitmann, and Fabian Steinmetz. 2020. Ahoi: Inexpensive, Low-power Communication and Localization for Underwater Sensor Networks and μAUVs. *ACM Transactions on Sensor Networks (TOSN)* 16, 2 (Jan. 2020), 18:1–18:46. doi:10.1145/3376921

[6] Alberto Signori, Filippo Campagnaro, Fabian Steinmetz, Bernd-Christian Renner, and Michele Zorzi. 2019. Data Gathering from a Multimodal Dense Underwater Acoustic Sensor Network Deployed in Shallow Fresh Water Scenarios. *Journal of Sensor and Actuator Networks* 8, 4 (Nov. 2019). doi:10.3390/jsan8040055

[7] Joachim Tapparel, Orion Afisiadis, Paul Mayoraz, Alexios Balatsoukas-Stimming, and Andreas Burg. 2020. An Open-Source LoRa Physical Layer Prototype on GNU Radio. In *2020 IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*. 1–5. doi:10.1109/SPAWC48557.2020.9154273

[8] Deniz Unal, Sara Falleni, Kerem Enhos, Emrecan Demirors, Stefano Basagni, and Tommaso Melodia. 2024. Design and Performance Evaluation of SEANet, a Software-Defined Networking Platform for the Internet of Underwater Things. *Computer Networks* 250 (Aug. 2024), 110579. doi:10.1016/j.comnet.2024.110579