Underwater Acoustic Modem for a MOrphing Distributed Autonomous Underwater Vehicle (MODA)

Emanuele Coccolo[‡], Filippo Campagnaro[‡⋆], Davide Tronchin[‡], Antonio Montanari[‡],
Roberto Francescon[⋆], Lorenzo Vangelista[‡⋆], Michele Zorzi[‡⋆]

[‡] Department of Information Engineering, University of Padova, via Gradenigo 6/B, 35131 Padova, Italy,
[⋆] Wireless and More srl, Via della Croce Rossa 112, 35129 Padova, Italy
[‡]{coccoloe,campagn1,tronchin,montanar,vangelista,zorzi}@dei.unipd.it
[⋆]{roberto.francescon,filippo.campagnaro,lorenzo.vangelista,michele.zorzi}@wirelessandmore.it

***Abstract*—In the last ten years several simulation studies on Autonomous Underwater Vehicle (AUV) swarm fleet formation have been performed, and some preliminary sea demonstrations of proof-of-concept prototypes were carried out. However, their actual realization is hindered on one side by the high cost of acoustic modems, whose price can easily exceed that of small AUVs, and, on the other side, by the difficulties of keeping track of the vehicles' positions due to the long latency required by traditional round trip time ranging measures. One-way travel-time (OWTT) halves the latency, at the cost of a high precision oscillator, such as an atomic clock or an oven controlled crystal oscillator, installed in the modem processing unit. In this work we present the design of a low-cost software-defined acoustic modem developed with off-the-shelf components and a low-complexity integration. We also present an analysis on how to perform OWTT depending on the mission duration and the requirements in ranging precision. The results obtained from a tank test of the modem proof-of-concept prototype proved the effectiveness of the design choices, encouraging us to move towards further implementation and realize a prototype with a higher technology readiness.**

I. Introduction

Underwater Acoustic Networks (UANs) are widely used in the marine domain to support several military and industrial applications. Although researchers and acoustic modem manufacturers have primarily focused their work on developing high power long range acoustic modems [1], [2] for military deployment in order to establish a communication link between submarines and large ships, in the last ten years the new developments of small Underwater Unmanned Vehicles (UUVs) [3], [4], like Remotely Operated Vehicles (ROVs) and Autonomous Underwater Vehicles (AUVs), called for small acoustic modems able to communicate up to a few hundreds of meters and with a power consumption of less than 10 Watts. For this reason, both manufacturers and research institutes are working actively on a wide variety of acoustic modems able to support not only large scale military and industrial applications [1], [2], [5], but also low-cost research-oriented deployments for small scale applications [6], [7], such as AUV swarm coordination.

This work has been partially supported by the Italian General Directorate of Naval Armaments (NAVARM) and the Italian National Plan for Military Research (PNRM) (contract 20593).

The main issues when designing an AUV swarm are often related to the deployment costs and the difficulty of maintaining the formation due to the long communication latency caused by the acoustic channel and used not only for exchanging information, but also for measuring the range between the vehicles [8]. While the cost issue has been mitigated with the development of both low-cost AUVs and low-cost modems, the obstacle of network latency is still unresolved, as the round-trip-time ranging operations performed by acoustic modems take a significant amount of time. A possible solution to limit the network latency is to use high precision clock oscillators to perform one-way travel-time (OWTT) ranging. Indeed, given that neither Global Positioning System (GPS) signals nor the Internet are available underwater, traditional timing mechanism cannot be used to correct the timing error introduced by common quartz oscillators, hence more precise oscillators need to be used. Although the feasibility of this approach has already been proven in [9], OWTT has not been used in low-cost AUV swarm deployments yet, due to the high cost of atomic clocks and their integration complexity. In summer 2021, Facebook released a first open-source [10] PCI Express card that embeds an atomic clock and can turn almost any computer into a very precise Stratum-1 NTP and PTP server. In addition, the availability of low-cost precise Oven Controlled Crystal Oscillator- (OCXO) based PTP and NTP servers [11] can provide a valuable alternative to the more expensive atomic clocks, and deserves further investigation. In this paper we compare the two solutions, providing a guideline for the development of an underwater network with OWTT ranging abilities.

The goal of this paper is to present the design and an experimental evaluation of an acoustic modem that will be used in a multi-vehicle system, where each vehicle acts as an extension of the remote vessel from which it was launched. This modem is developed in the context of the Italian PNRM co-funded MOrphing Distributed Autonomous underwater vehicle (MODA) project, that aims to develop a distributed underwater vehicle able to transform its shape in a fluid, gradual and seamless way, while maintaining wireless connections between the composing robotic parts. The modems should operate at high frequencies and transmit with low power in order to minimize the medium-range acoustic footprint, and

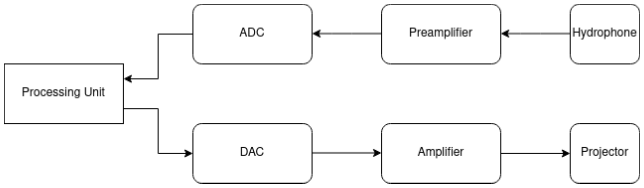

Fig. 1. Hardware block diagram.

are equipped with a very precise clock oscillator to supply high precision in the distance measurements. The distance measurements obtained from the communication links allow to improve the navigation performance and to estimate the spatial configuration of the robotic team through distributed localization techniques. In addition, the main part of the modem is built with low-cost off-the-shelf components and an ARM-based Single Board Computer (SBC) used as signal processing unit.

The rest of the paper is organized as follows. Section II presents the hardware components used to create the modem prototype; Section III presents the software architecture, that is the main contribution of our work, where the complete transmission flow is analyzed, and signal processing techniques and the OWTT ranging protocol are discussed. Section IV presents a comparison of different precise oscillators, providing a guideline of which oscillator should be used depending on the mission duration and the ranging precision required. Some in-laboratory results are presented in Section V while Section VI draws our concluding remarks.

II. Hardware description

Figure 1 shows a block diagram presenting the components of the first prototype of the modem. A processing unit performs the signal processing and transmits the modulated signal through a digital to analog converter (DAC) to a signal amplifier, that is connected with an acoustic projector. The same processing unit samples the pre-amplified signal received by a hydrophone using an analog to digital converter (ADC). All signal processing is performed in software using a combination of C and C++ libraries, hence developing a modular software defined modem. With this approach, the signal processing unit and DAC/ADC can be changed without any need to adapt the code: for instance, in Figure 2 a first transmission was performed using a Linux PC as the processing unit and its 192 kHz sound card DAC and ADC, while in Figure 3 the same transmission was performed using a Raspberry PI 4 with a high quality sound card. In both tests the processing units were transmitting data using Frequency Shift Keying (FSK) modulation.

One Aquarian Scientific-1 (AS-1) transducer [12] is used as acoustic projector for signal transmission and another AS-1 is used as hydrophone for signal reception: in order to avoid saturation of the receiver, its pre-amplifier is deactivated when the modem prototype is transmitting. In a more advanced version of the modem, we envision the use of a transmitter/receiver switch that will allow us to use the same

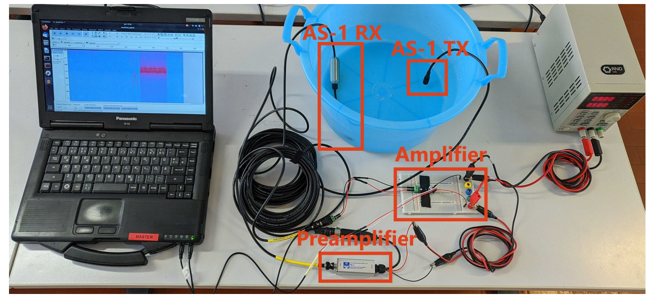

Fig. 2. Test components with a PC running a software defined modem using Hi-Fi integrated sound cards as ADC and DAC.

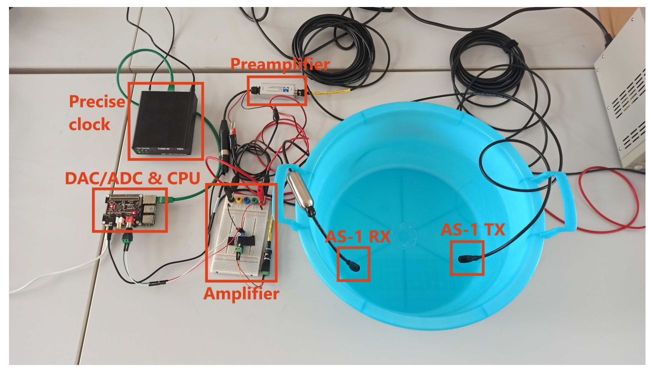

Fig. 3. Test components of the modems with a Raspberry PI 4b running a software defined modem using HiFiBerry DAC+ ADC Pro and a TM2000 OCXO-based time server.

transducer to both transmit and receive: this switch will be designed in house because commercial devices for piloting reception and transmission of acoustic transducers are more expensive than the AS-1 transducer itself (that costs about 400 US Dollars). Actually, AS-1 is not a transducer designed for acoustic communication, but it is a simple omnidirectional off-the-shelf hydrophone mainly used for underwater sound measurements in marine and industrial environments, and has low power transmission capabilities, in the frequency range between 35 and 90 kHz, used by the factory only for simple calibration purposes. The signal received by the hydrophone placed in the reception unit is amplified with a PA-4 Hydrophone Preamplifier [13] before reaching the ADC, so as to have a stronger signal to decode. The transmitting unit uses a low cost class AB audio amplifier to amplify the signal provided by the DAC before the actual transmission. Although the more expensive class D audio amplifiers are more efficient in terms of power consumption, they employ a low pass filter that cuts all frequencies above 20 kHz, thus becoming unusable in the ultrasonic frequency band. In the next version of the modem, the class AB audio amplifier will be substituted with a custom amplifier, as off-the-shelf audio amplifiers are designed for loads of 4-16 Ω, significantly different from the impedance of the transducer that is approximately equal to 200 Ω in the frequencies used for transmission. The central frequency of the modem is 50 kHz and its bandwidth limited to 30 kHz, as a result of the limitations imposed by transducer,

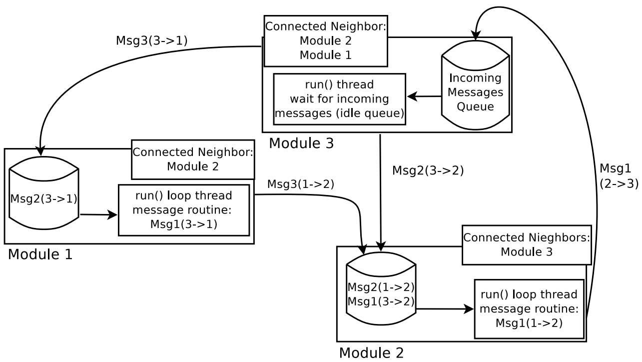

Fig. 4. Communication between software modules.

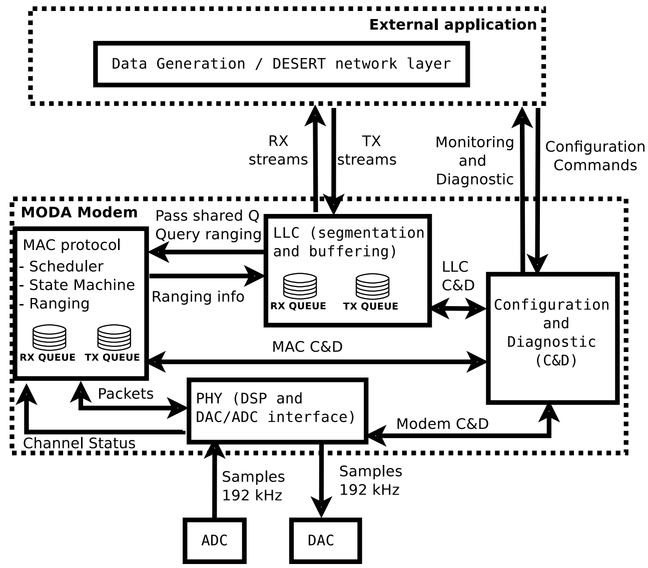

Fig. 5. Software design with explanation of the transmission flow between the user application and the DAC/ADC.

DAC and ADC.

To build a distributed localization system in which each node estimates its local position, a high precision clock oscillator is used. This clock allows to measure the distance of each node from the emitter with the transmission of one hop signal, while the traditional techniques compute the distance by measuring the round trip time. In Section IV, we analyze different solutions based on different types of oscillators: the best solution depends on the mission duration and the required precision. In Figure 3 we present the components used in our first proof-of-concept modem prototype, where a Raspberry PI 4 is used as processing unit, a HiFiBerry DAC+ ADC Pro [14] performs the sampling operations, two AS-1 are used as projector and hydrophone, a PA-4 Hydrophone Preamplifier [13] amplifies the signal received by the hydrophone, and a low cost audio amplifier increases the transmitted signal source level. A TM2000B OCXO-based NTP server is used to maintain the Raspberry PI 4 clock synchronized with a stable reference.

III. Software Design

The software defined modem presented in this paper has a modular architecture, where each component derives from a common base class, called `Module`, that provides common Application User Interfaces (APIs) and a mechanism to enable the communication between different software modules. During the software initialization, all connections between modules are performed using the `connectModule()` member function. In order to communicate with a connected module, it is sufficient to create a message that contains the request, and push it to the module message queue with the `enqueue()` member function. Finally, each module has a running thread checking in its concurrent queue of messages the requests incoming from other connected modules: once a message is received the message routine operation is executed, and its result is reported back to the module that issued the message via callback.

We remark that the connection between modules may be unidirectional: for instance, `Module A` can send request messages to `Module B`, but not vice-versa. An example of communication between three software modules is presented in Figure 4, where `Module 1` is connected to `Module 2`, `Module 2` is connected to `Module 3`, and `Module 3` is connected to `Module 1` and `Module 2`.

A. Transmission Flow

The software architecture, used in the modem, is depicted in Figure 5 and described as follows.

A logic link control (`LLC`) module slices the data arriving from the user (via TCP socket) in segments, embeds each data segment in a packet composed of header containing information about source, destination address, modulation and coding scheme; then, it stores the packets in a concurrent queue for packet transmission (`TX_QUEUE`), shared with the Media Access Control (`MAC`) protocol module, that decides when to transmit the packet according to its `scheduler` and to the state of the physical module (`PHY`). The latter receives the outgoing packets from `MAC`, applies forward error correction (FEC), modulates the data in symbols, and sends the sampled symbols to the DAC. In reception, `PHY` receives the samples from the ADC and performs all the signal processing required to demodulate the symbols and decode the packet, that is then sent to the `MAC`. The latter checks if the packet is a data packet or a signaling packet generated by the `MAC` protocol: in the former case, it stores the packet in the concurrent reception queue (`RX_QUEUE`) shared with `LLC`, otherwise it processes the signaling packet. `LLC` checks if the packets are received in order, extracts the data and sends it to the user application. All signal processing is performed in software, using a wrapper of the open-source *liquid-dsp* library [15]. DAC and ADC are instead accessed using a wrapper of the Advanced Linux Sound Architecture (ALSA) library [16]. A Configuration and Diagnostic (`C&D`) module allows a user to receive status information about the modem, and to issue commands to configure the modem's parameters, such as transmission power and modulation and

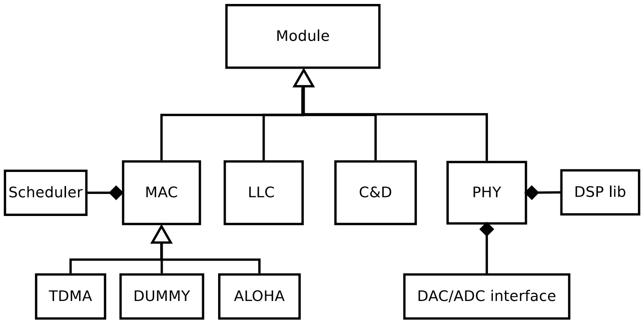

Fig. 6. Unified Modeling Language representation of the software components.

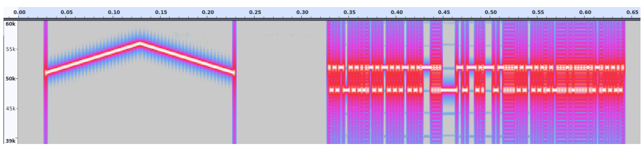

Fig. 7. Spectrogram of a packet transmitted with an FSK modulation with an upchirp and downchirp preamble.

coding scheme. In this first version, we used strings sent via TCP socket for configurations and diagnostics, however we envision to use in the near future a web application to configure the system via REST APIs [17].

This modem performs only data-link and physical layer operations, as it relies on other software tools, such as DESERT Underwater [18], to perform routing and transport layer tasks. `LLC`, `MAC`, `PHY` and `C&D` classes extend from the *base class* module, while `scheduler`, `DSP` and `DAC/ADC interface` are standalone classes. In the next version of the modem, we envision to support different `MAC` protocols, simply extending the `MAC` module: this architecture allows us to use a different `MAC` protocol without changing the other modules. The software Unified Modeling Language (UML) representation is depicted in Figure 6.

B. Signal Processing

The *liquid-dsp* [15] signal processing library is used to perform modulation, demodulation, signal synchronization, equalization and all other mathematical operations used in signal processing. In the tests presented in Figures 2 and 3 we transmitted data using an FSK modulation. Using the *liquid-dsp* APIs, we added a simple repetition FEC and an 8 bit Cyclic Redundancy Check (CRC8) at the end of the payload, in order to validate the data. To identify when the reception of a packet starts, an upchirp and a downchirp are used as the packet preamble. When a packet is received, the signal first passes through a matched filter detecting the beginning of the chirps in order to find the exact point in which the data starts. The chirps can also be used for Doppler estimation [19] and to estimate the impulse response of the acoustic channel [20]. The spectrogram of the packet can be observed in Figure 7.

This preliminary structure implementation has a drawback, as it was implemented to quickly perform the transmission

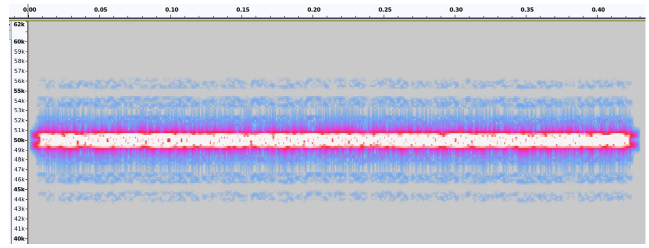

Fig. 8. Spectrogram of a packet transmitted with a liquid-dsp flexframe, interpolated and moved to pass-band. The modulation used is QPSK.

and reception chains to evaluate the hardware components, rather than focusing on real-time signal processing applied sample-by-sample. In fact, the receiver first records a large number of samples and then passes them to the decoder, relaxing the real-time reception requirements needed to both perform OWTT and maximize the throughput. For this reason, after this preliminary evaluation, for the first version of the modem we used the *liquid-dsp* Flexible Framing Structure (`flexframe`) [21], a transmission frame composed of an initial ramp-up and a final ramp-down to reduce spectral side lobes, a BPSK phasing preamble, a pseudo noise (PN) sequence used for synchronization at the receiver, a 20-byte header, and variable size payload, that can be modulated using phase shift- and amplitude-based modulations, as well as different types of FEC and CRC. The entire process is performed in complex base-band, using 2 samples per symbol. In order to increment the number of samples per symbol, zero-interpolation followed by a band-pass filter was applied; then the signal is moved to real pass-band using a numerically-controlled oscillator[1]. The transmission is finally performed with the audio DAC using the ALSA interface.

At the receiver, the signal coming from the ADC is then moved to complex base-band and decimated. During the reception, the *liquid-dsp* flexframe corrects to its best the channel impairments such as noise and multipath using various signal processing elements (including, but not limited to, least mean square (LMS) equalizer, phase-locked loop (PLL) for phase recovery, automatic gain control, etc.) and attempts to decode the frame. When seeking a frame, the receiver initially sets its internal loop bandwidths high for acquisition, including those for symbol timing recovery, and carrier frequency/phase recovery (acquisition mode). Once the PN sequence has been found, the receiver assumes it has a sufficiently good estimate of the channel impairments and reduces its control loop bandwidths significantly, moving to tracking mode, where the signal is demodulated and the FEC applied. All this procedure is performed in real time. The spectrogram of a packet modulated with Quadrature Phase Shift Keying (QPSK) is presented in Figure 8.

Although *liquid-dsp* supports various modulations including PSK, AM, FSK, OFDM and DSSS, the flexframe can use only PSK and AM based modulations. Still, FSK-frame,

[1]The process to move the signal from complex base-band to real pass-band is presented in [22].

OFDM-frame and DSSS-frame are available and can be used instead of the flexframe; hence, extending our physical layer to support also those modulations is straightforward and left to future modem improvements. In the future, we also envision the addition of the upchirp and downchirp preamble that is proven to outperform the PN sequence preamble in underwater acoustic networks [20], strongly affected by multipath and Doppler.

C. Ranging

The modem can perform OWTT ranging operations either upon request or along data transmission. In the former case, the ranging operation is required by the user through the command channel: the command, received at the `C&D` module, is sent to the `MAC` that exchanges two OWTT packets with the required destination. In the latter case, the `MAC` includes in the header the transmission time t_{tx}, hence the receiving node can measure the time elapsed from t_{tx} to the reception time t_{rx} and infer the propagation delay d_p, given that

$$t_{rx} - t_{tx} = t_{proc,tx} + t_{pkt} + d_p + t_{proc,rx} + \Delta_{clk}, \quad (1)$$

where t_{pkt} is the known packet duration, $t_{proc,tx}$ is the processing time of the transmitter, $t_{proc,rx}$ the processing time of the receiver and Δ_{clk} the difference between the transmitter and receiver clocks. In order to minimize $t_{proc,tx}$, the packet is modulated a fraction of second before its actual transmission, hence ensuring the packet is actually transmitted at the scheduled moment. A solution to neglect also $t_{proc,rx}$ is to keep track of the time when each sample is created by the ADC: this is possible given that ALSA continuously records data from its initialization onward. Δ_{clk}, instead, depends on the precision of the clocks used the the two nodes, and is discussed in Section IV.

IV. High Precision Oscillators

Different high precision clocks are available on the market, that provide a clock drift performance which is usually proportional to their price. We investigated different products, discussing their power consumption, their output and their performance in terms of clock drift and, therefore, of distance measurement error, assuming a speed of sound equal to 1500 m/s. We remark that the performance provided by the oscillators' manufacturers is obtained after several days (between 2 to 30, depending on the clock) of 1 pulse per second (1 PPS) disciplining calibration with high precision oscillators, such as the atomic clock reference given by a Global Navigation Satellite System (GNSS) receiver. Temperature Compensated Crystal Oscillators (TCXO) are 10 to 100 times more precise then the not-compensated quartz oscillators commonly used in desktop personal computers and laptops. For instance, the model TC-500 [23] in one day of operations has a drift of about 16 ms, while a normal quartz oscillator has a drift of up to 1 s per day. Despite this improvement in drift precision, a TCXO is not suitable for our application, as in one day of mission it will introduce an OWTT distance error of about 24 m, that in a deployment where the nodes are spaced less then 100 m from each other will result in an error of more than 24%. A Microprocessor Controlled Oscillator (MCXO) like MX-503 [24], instead, can ensure a daily drift of 0.2 ms and, therefore, an OWTT distance error of 0.315 m, at a price very similar to that of TC-500 (about 200 EUR). Both MCXO an TCXO have a power consumption of about 100 mW. OCXO oscillators, such as OX-208 [25], are even more precise, providing an error of less than half meter for a mission that lasts one week, but cost more than twice the price of an MCXO and have a high power consumption (2 W). Chip Scale Atomic Clocks (CSAC) [26], instead, can provide an OWTT error of less then 10 cm in one week of mission and of about 1.75 m in one month, with a power consumption of less than 150 mW, but the cost of these low power rubidium-based oscillators is more than 5000 EUR per unit (similar performance can also be obtained with the CSAC developed by Teledyne [28]). Finally, Miniature Atomic clocks (MACs) can provide a higher precision at half the price of a CSAC, providing an OWTT ranging error of about 10 cm in one month, but with a higher power consumption (6.3 W). Also, Orolia has a set of OCXO and atomic clocks with similar characteristics and price [29]. The characteristics of different types of high precision oscillators are summarized in Table I, while Figure 9 depicts the time offset (corresponding to Δ_{clk} in Eq. (1)) of the most relevant clocks versus the mission duration, providing an idea of which clock should be employed according to the mission and the ranging precision requirements.

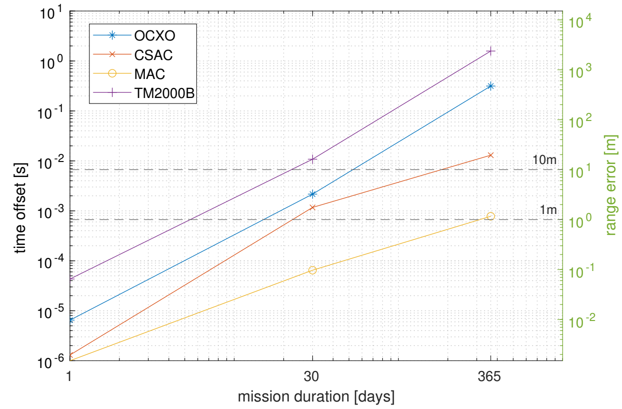

Fig. 9. Comparison of the time drift of different clocks according to the manufactures' datasheets.

TABLE I
CLOCK COMPARISON

Model	Power	Out. Interface	Price	Err. @24h
TC-500 TCXO [23]	0.115 W	50:160 MHz	185 EUR	16 ms
MX-503 MCXO [24]	0.040 W	8:50 MHz	200 EUR	0.21 ms
OX-208 OCXO [25]	2 W	5:20 MHz	550 EUR	6.5 μs
SA.45s CSAC [26]	0.12 W	10 MHz, 1 PPS	5600 EUR	1.3 μs
SA.55s MAC [27]	6.3 W	10 MHz, 1 PPS	2100 EUR	1 μs
TM2500B [11]	2.4 W	NTP, PTP, 10MHz, 1 PPS	640 EUR	50:100 μs

The main issues we faced when selecting the oscillator were related to the delivery time due to the global shortage of semiconductors, and the complex integration of the oscillator chip in our prototype, as our idea is to keep the system flexible to future modifications and changes, hence avoiding the soldering of expensive components in prototypes that will eventually become deprecated. This issue can easily be solved purchasing the clocks development kit, at the price of a few hundred EUR, depending on the model, that allows the use of the clocks without any additional integration effort. Another issue is that all nodes of the network require a precise initial synchronization and to be disciplined with a 1 PPS signal during the warm-up: this means that all nodes need to take the time from a common source, such as a web time server or the GPS. For this reason all manufactures provide also a board where the clock is integrated with a GNSS receiver, used to both take the absolute time and to calibrate the clock. This solution is even more expensive, because an integrated board with an atomic clock costs about 6000 EUR.

In order to simplify the acquisition of precise clock references for our first prototype, where we will test the OWTT in maximum one day of mission, we opted for the TM2000B [11], a small form factor OCXO- and GNSS-based time server, that allows a node connected via Ethernet to get the time via either NTP or PTP. When GNSS is not available, the OCXO ensures a holdover time with a daily drift of 50 μs when using PTP, and 100 μs when using NTP (both mostly caused by the time protocol itself rather than the clock drift), resulting in an error of less than 15 cm in one day, which is in line with our requirements. These values are provided by the factory [30] and confirmed by the tests we performed in the laboratory.

The model TM2500B has additional 1 PPS and 10 MHz output, that can be used for disciplining the Raspberry Pi clock using its GPIOs, like we envision to do when we will receive the SA.55s MAC atomic clocks. This operation is possible thanks to the availability of open source libraries: among others, we managed to test the PPS-Client v2.0.3 [31] on the Raspberry Pi.

To further simplify the integration, Facebook launched the open-source Facebook Time Card [10], that allows to integrate either an OCXO or an atomic clock in a PCI Express PTP card. Indeed, although PTP is currently not supported by Raspberry Pi, this new development is giving a big push to the Raspberry community to find some workaround and to provide support to PTP in the next Operating System release.

V. In-Tank Testing and Results

In this section, we present some preliminary tests performed using the SIGNET laboratory (http://signet.dei.unipd.it/) 200-liter water tank coated with phono-absorbing material [32] to mitigate the multipath effect. The test setup is depicted in Figure 10 and described as follows. Two AS-1 transducers are deployed 10 centimeters apart and kept in the same positions for the entire test. All experiments are performed immediately one after the other, trying to maintain the same

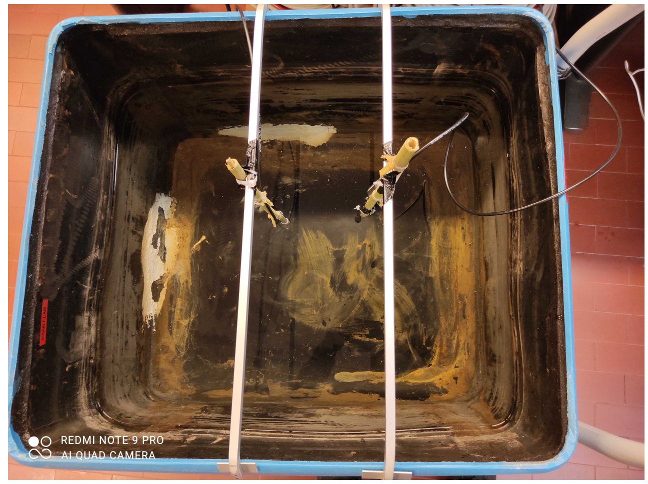

Fig. 10. In-laboratory tank test setup, where two AS-1 transducers are deployed few centimeters apart, kept in the desired position with bamboo sticks.

conditions, such as water temperature and composition (due to the very slow but constant dissolution of neoprene rubber and bicomponet glue used to stick the neoprene to the tank edges) from one test to another. The receiving transducer is connected with the preamplifier and then to the microphone input of the HiFiBerry DAC+ ADC Pro, connected with the Raspberry Pi used as receiving unit. Conversely, the transmitting transducer is connected directly to the output of a second HiFiBerry DAC+ ADC Pro that sends the signal generated by a second Raspberry Pi used as transmitting unit. During the test, we transmitted repetitively the 13-byte long string "Hello, world!", using the *liquid-dsp* flexframe structure, that introduces a QPSK-modulated 20-byte header, containing information on the modulation and coding scheme of the payload as well as the payload size and header CRC. Different modulation and coding schemes are tested in the payload. The flexframe creates a signal with 2 samples per symbol, and we interpolated and filtered each sample with 99 zeros, with a resulting 200 samples per symbol. In the case of a QPSK modulated signal, given that the sampling rate is 192 kHz and that each symbol carries 2 bits, the resulting raw bitrate can be computed as 2*192000/202 = 1900 bps, that is suddenly reduced due to the preamble, CRC, FEC, and the ramp-up and ramp-down signals. Specifically, 64 BPSK-modulated bits are used for the preamble, 8 bits are used for CRC, and ramp-up and ramp-down signals are composed of 7 symbols each. Furthermore, both the header and the header's CRC are coded with a Single-error correction and double-error detection (SECDED) Hamming code 72/64 concatenated with a Hamming(8,4), thus more than doubling the number of bits transmitted. Therefore, without FEC at the payload, the signal duration is 0.38 s, and the resulting bitrate is 673 bps. Given that the packet header and preamble structure and modulation are fixed, we tested different configurations of payload FEC and modulation, starting from the case where the FEC is not used, arriving to the case where two

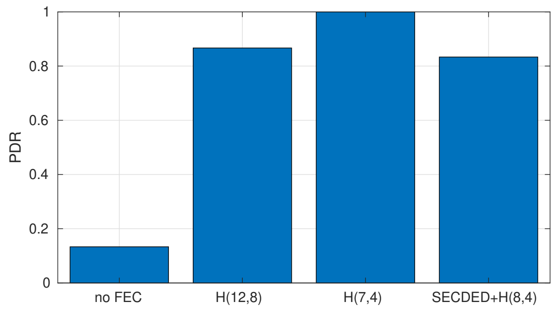

Fig. 11. Packer delivery ratio of QPSK modulated packet payload when using different FEC.

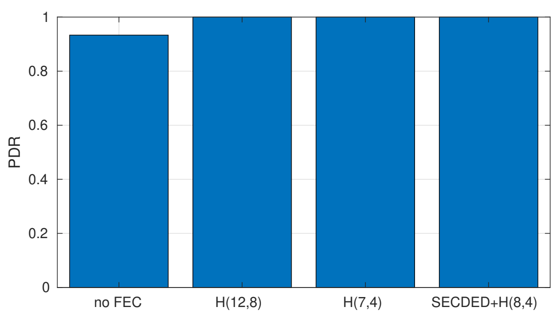

Fig. 12. Packer delivery ratio of OOK modulated packet payload when using different FEC. Similar results were observed for BPSK, ASK2 and DPSK2 modulations, where the PDR = 1 even without FEC.

levels of FEC are applied, called inner FEC and outer FEC. Specifically, we tested the performance of Hamming(7,4) and Hamming(12,8) as inner FEC without any outer FEC, and the case where a SECDED(72/64) inner FEC is concatenated with a Hamming(8,4) outer FEC, hence applying the same FEC structure used in the packet header. In addition, five different modulations are tested, namely: BPSK, QPSK, On-Off Keying (OOK), Amplitude and Phase Shift Keying 4 (APSK4) and Differential PSK 4 (DPSK4). The size of the payload CRC is set to 8 bits. The test consisted in the transmission of 30 packets per configuration.

In order to provide a comparison between the different payload configurations, we observed the packet delivery ratio (PDR) of all packets correctly detected, computed as the number of times the payload string was correctly received at the destination divided by the total number of packets whose header was correctly demodulated. The reason we did not compute the PDR observing the total number of packets correctly received divided by the total number of packets transmitted is because we want to focus on the comparison of the payload modulation and coding schemes, and given that some payload modulations we selected, such as BPSK, are more robust than the header modulation, the PDR computed in such a way would be biased by the lower probability of reception of the header.

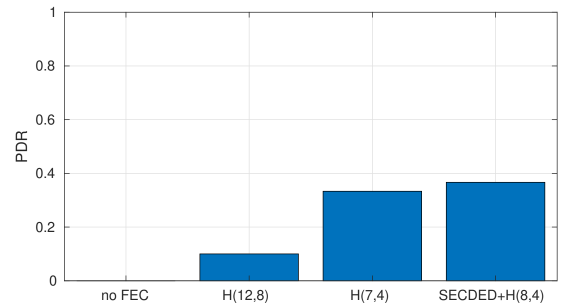

Fig. 13. Packer delivery ratio of APSK4 modulated packet payload when using different FEC.

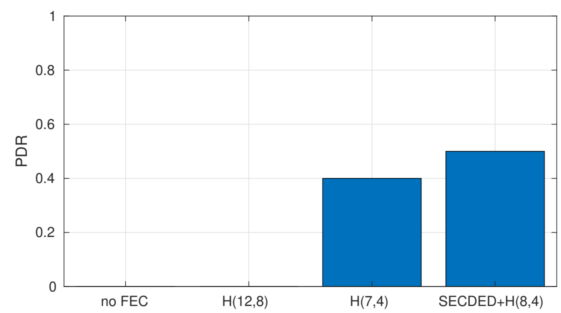

Fig. 14. Packer delivery ratio of DPSK4 modulated packet payload when using different FEC.

A. Results

The PDR observed when transmitting a QPSK-modulated payload is depicted in Figure 11. Clearly, the lowest PDR was observed when using no FEC, where a PDR = 0.233 was experienced. Hamming(7,4) corrects more errors than Hamming(12,8) at the price of a higher redundancy, while the concatenated FEC composed of SECDED(72/64) and Hamming(8,4) surprisingly performed slightly worse than Hamming(12,8). Similar results were observed with Quadrature Amplitude Modulation 4 (QAM4), since it has the same constellation as QPSK. In other tests performed in other days, SECDED(72/64) and Hamming(8,4) provided a PDR equal to the one obtained with Hamming(7,4), indicating that the tank setup should be used to evaluate the system components in order to have the feeling that the system is actually working, rather than to have a fair comparison between very similar schemes, because the channel may slightly change even in a short period of time.

Both OOK and BPSK (Figure 12) modulations were observed to be more robust than QPSK, in fact with FEC we experienced a PDR = 1 with both OOK and BPSK, and, while OOK without FEC had an error rate of less than 5%, with BPSK we did not observe any packet loss even with no coding, and the same happened with Amplitude Shift Keying 2 (ASK2) and DPSK2 modulations.

While with more efficient but less robust modulations such as QAM8, PSK8 and PSK16 we observed a PDR = 0 both with and without FEC, we managed to receive some packets with APSK4 (Figure 13) and DPSK4 (Figure 14) modulations, that presented similar results. Specifically, with APSK4 no packets were received without FEC, 10% of the packets were received with Hamming(12,8), and 30% of the packets were received when Hamming(7,4) was applied. Conversely, the concatenated FEC slightly outperformed the others, with a resulting PDR = 0.37.

Finally, with DPSK4 no packets were received without FEC and with Hamming(12,8), while 40% of the packets were received with Hamming(7,4) and the concatenated FEC definitely outperformed the others, with a resulting PDR = 0.50.

VI. Conclusions an Future Works

In this paper, we presented the design of the modem used in the MOrphing Distributed Autonomous underwater vehicle (MODA) project, presenting all software and hardware components, and both hardware and software architectures. The modem will be able to perform one way travel time ranging leveraging on the precise time reference provided by an accurate clock. Specifically, depending on the envisioned mission duration, the timing system can be based either on OCXO or on atomic clocks, the latter being more accurate and able to keep a precise timing reference for up to several months, but at a price that is one order of magnitude higher than OCXO-based timing systems, still able to keep a precise timing reference for up to one week. An in-tank evaluation of the proof-of-concept prototype proved the effectiveness of the proposed solution, encouraging further development in this direction. Future work will focus on the implementation of a new prototype with a higher technology readiness level, and on the integration of atomic clocks and OCXOs in the modem.

References

[1] "Develogic Subsea Systems," Last time accessed: Dec. 2021. [Online]. Available: http://www.develogic.de/

[2] "Kongsberg acoustic modems," Last time accessed: Dec. 2021. [Online]. Available: https://www.kongsberg.com/maritime/products/Acoustics-Positioning-and-Communication/modems/

[3] "BlueROV2," Last time accessed: Dec. 2021. [Online]. Available: https://bluerobotics.com/store/rov/bluerov2/

[4] "Hydromea Exray," Last time accessed: Dec. 2021. [Online]. Available: https://www.hydromea.com/exray-wireless-underwater-drone/

[5] "Evologics Acoustic Modems," Last time accessed: Dec. 2021. [Online]. Available: https://evologics.de/acoustic-modems/

[6] "Waterlinked modem m64," Last time accessed: Dec. 2021. [Online]. Available: https://store.waterlinked.com/product/modem-m64/

[7] C. Renner and A. J. Golkowski, "Acoustic Modem for Micro AUVs: Design and Practical Evaluation," in *Proc. International Conference on Underwater Networks & Systems (WUWNet)*, Shanghai, China, Oct. 2016.

[8] F. Mason, F. Chiariotti, F. Campagnaro, A. Zanella, and M. Zorzi, "Low-cost AUV swarm localization through multimodal underwater acoustic networks," in *MTS/IEEE Global Oceans 2020: Singapore – U.S. Gulf Coast*, 2020, pp. 1–7.

[9] K.G. Kebkal, O.G. Kebkal, E. Glushko, V.K. Kebkal, L. Sebastião, A. Pascoal, J. Gomes, J. Ribeiro, H. Silva, M. Ribeiro, and G. Indivery, "Underwater acoustic modems with integrated atomic clocks for one-way travel-time underwater vehicle positioning," in *Proc. UACE*, Skiathos, Greece, September 2017.

[10] "Open-sourcing a more precise time appliance," Last time accessed: Dec. 2021. [Online]. Available: https://engineering.fb.com/2021/08/11/open-source/time-appliance/

[11] "GPS NTP+PTP Network Time Server," Last time accessed: Dec. 2021. [Online]. Available: https://timemachinescorp.com/product/gps-ntpptp-network-time-server-10mz-output-tm2500/

[12] "AS-1 Hydrophone," Last time accessed: Dec. 2021. [Online]. Available: https://www.aquarianaudio.com/as-1-hydrophone.html

[13] "PA-4 Hydrophone Preamplifier," Last time accessed: Dec. 2021. [Online]. Available: https://www.aquarianaudio.com/pa4.html

[14] "Hifiberry," Last time accessed: Dec. 2021. [Online]. Available: https://www.hifiberry.com/

[15] "liquid-dsp," Last time accessed: Dec. 2021. [Online]. Available: https://liquidsdr.org/

[16] "ALSA project - the C library reference," Last time accessed: Dec. 2021. [Online]. Available: https://www.alsa-project.org/alsa-doc/alsa-lib/index.html

[17] L. Richardson and S. Ruby, *RESTful Web Services*. Sebastopol, CA: O'Reilly Media, Inc, 2007.

[18] F. Campagnaro, R. Francescon, F. Guerra, F. Favaro, P. Casari, R. Diamant, and M. Zorzi, "The DESERT underwater framework v2: Improved capabilities and extension tools," in *Proc. Ucomms*, Lerici, Italy, Sep. 2016.

[19] R. Diamant, A. Feuer, and L. Lampe, "Choosing the right signal: Doppler shift estimation for underwater acoustic signals," in *ACM WUWNet*, November 2012.

[20] R. Diamant, F. Campagnaro, M. de Filippo de Grazia, P. Casari, A. Testolin, V. Sanjuan Calzado, and M. Zorzi, "On the relationship between the underwater acoustic and optical channels," *IEEE Transactions on Wireless Communications*, vol. 16, no. 12, pp. 8037–8051, December 2017.

[21] "Flexible framing structure (flexframe)," Last time accessed: Dec. 2021. [Online]. Available: https://liquidsdr.org/doc/flexframe/

[22] "liquid-dsp conversion example," Last time accessed: Dec. 2021. [Online]. Available: https://github.com/jgaeddert/liquid-dsp/blob/master/examples/conversion_example.c

[23] "TX-500 Temperature Compensated Crystal Oscillators (TCXO)," Last time accessed: Dec. 2021. [Online]. Available: https://www.vectron.com/products/tcxo/tx-500.htm

[24] "MX-503 Microprocessor Corrected Crystal Oscillator," Last time accessed: Dec. 2021. [Online]. Available: https://www.microsemi.com/product-directory/5247

[25] "OX-208 High Stability Oven Controlled Crystal Oscillator," Last time accessed: Dec. 2021. [Online]. Available: https://www.microsemi.com/product-directory/ocxo/5264-ox-208

[26] "Chip Scale Atomic Clock (CSAC)," Last time accessed: Dec. 2021. [Online]. Available: https://www.microsemi.com/product-directory/clocks-frequency-references/3824-chip-scale-atomic-clock-csac

[27] "Miniature Atomic Clock (MAC - SA5X)," Last time accessed: Dec. 2021. [Online]. Available: https://www.microsemi.com/product-directory/embedded-clocks-frequency-references/5570-miniature-atomic-clock-mac-sa5x

[28] "Teledyne Chip Scale Atomic Clock," Last time accessed: Dec. 2021. [Online]. Available: http://www.teledyne-si.com/products-and-services/scientific-company/csac

[29] "Atomic Clocks and Oscillators," Last time accessed: Dec. 2021. [Online]. Available: https://www.orolia.com/solution/atomic-clocks-and-oscillators/

[30] "TimeMachines Time Server Accuracy TM1000A and TM2000A/B and TM2500B," Last time accessed: Dec. 2021. [Online]. Available: https://www.timemachinescorp.com/wp-content/uploads/TMTimeServerAccuracyRevB.pdf

[31] "PPS-Client v2.0.3," Last time accessed: Dec. 2021. [Online]. Available: https://github.com/rascol/PPS-Client

[32] F. Meneghello, F. Campagnaro, R. Diamant, P. Casari, and M. Zorzi, "Design and evaluation of a low-cost acoustic chamber for underwater networking experiments," in *Proceedings of the 11th ACM International Conference on Underwater Networks & Systems*, ser. WUWNet '16. New York, NY, USA: Association for Computing Machinery, October 2016.