# Full Reconfiguration of Underwater Acoustic Networks through Low-Level Physical Layer Access

**Filippo Campagnaro**
DEI, University of Padova
Padova, Italy
campagn1@dei.unipd.it

**Roberto Francescon**
DEI, University of Padova
Padova, Italy
frances1@dei.unipd.it

**Oleksiy Kebkal**
EvoLogics GmbH
Berlin, Germany
lesha@evologics.de

**Paolo Casari**
IMDEA Networks Institute
Madrid, Spain
paolo.casari@imdea.org

**Konstantin Kebkal**
EvoLogics GmbH
Berlin, Germany
kebkal@evologics.de

**Michele Zorzi**
DEI, University of Padova
Padova, Italy
zorzi@dei.unipd.it

## ABSTRACT

Underwater acoustic communications experiments often involve custom implementations of schemes and protocols for the physical and data link layers. However, most commercial modems focus on providing reliable or optimized communication links, rather than on allowing low-level reconfiguration or reprogramming of modulation and coding schemes. As a result, the physical layer is typically provided as a closed, non-reprogrammable black box, accessible by the user only through a specific interface. While software-defined modems would be the ultimate solution to overcome this issue, having access to the symbols transmitted by the modems using a proprietary modulation format already opens up a number of research opportunities, e.g., aimed at the cross-layer design and optimization of channel coding schemes and communication protocols.

In this paper, we take the latter approach. We consider the commercial EvoLogics modem, driven by a custom firmware version that bypasses the channel coding methods applied by the modem, and allows the user to set the transmit bit rate to any desired value within a given set. This makes it possible to evaluate different coding schemes in the presence of different bit rates. Our results show that the custom firmware offers sufficient flexibility to test different configurations of the coding schemes and bit rates, by providing direct access both to correctly decoded and to corrupted symbols, which can be separated at the receiver for further processing. In addition, we show that the DESERT Underwater framework can also leverage the same flexibility by employing low-level physical layer access in more complex networking experiments.

## KEYWORDS

Underwater acoustic networks; forward error correction; EvoLogics S2CR modems; acoustic chamber; DESERT Underwater; experiments

## 1 INTRODUCTION AND RELATED WORK

To date, acoustic communications are the main carrier of underwater digital data transmissions. Considerable progress has been made towards reliable underwater communications that can adapt to different scenarios and applications, which generally imply different requirements in terms of energy consumption, likelihood of correct data delivery, and spatial coverage. After the first underwater acoustic modems had been made available to cover specific market niches, their number has rapidly increased. This includes both commercial off-the-shelf (COTS) products [6, 8, 9, 11, 17, 20] and research efforts aimed at next generation, high-bit rate [5, 23], short range, as well as fully reprogrammable acoustic modems [7].

Typically, COTS modems are optimized for reliability and long-term deployments. This is specifically the case if the modem must be employed in critical scenarios such as those involving security or tactical applications. Among other consequences, the modems are provided as "blackboxes," which are not directly controllable or reconfigurable by the end user. If, on the one hand, the above criteria provide the user with additional guarantees and a more dependable product, on the other hand the same criteria constitute a hinder to research and experimentation, as the same hardware cannot double as a general-purpose research platform for researchers and developers to experiment with.

Still, in some cases the users are allowed additional degrees of freedom, e.g., through the adjustment of the parameters

of the modem's physical layer [1, 14, 15, 22], For instance, the standard firmware of the EvoLogics S2CR modem [8] provides the user with either a fixed bit rate of about 1 kbit/s to transmit short messages, or an adaptive communication system, where the channel is probed periodically, and the bit rate adapted to the sensed channel conditions. The actual data rate is then equal to one half the bit rate due to a rate-1/2 channel code. Teledyne Benthos [17] modems can support seven different data rates, from 80 to 2400 bit/s; however, no information is provided on the coding scheme. The Woods Hole Oceanographic Institution's Micro-Modem also provides seven different configurations. Each differs in terms of data rate, channel code rate, and packet size. The AQUAtec AQUAmodem [11] provides instead a fixed bit rate, which however can be set during the production stage according to the user specifications.

More recent developments opened to the complete reprogrammability of the physical layer and/or network layers of the modem [3–5, 18, 21]. Remarkably, this evolution involved both academic research efforts and consumer-grade equipment. For example, the Develogic HAM.Node [1] provides an open physical layer, fully configurable and suitable both for commercial and for research applications. The Subnero software-defined acoustic modem [20] runs UnetStack [4], which allows the user to reprogram the modem down to the physical layer. Similarly, EvoLogics releases a software-defined version of their modem as well, where it is possible to transmit and receive arbitrary waveforms and set a reference to trigger signal detection.

It may still happen that the modems are designed around specific architectures, or incorporate specialized hardware components such as digital signal processors (DSPs) and field-programmable gate arrays (FPGAs). Alternatively, some modems may require the knowledge of specific languages or frameworks to (re)program the network protocol stack logic. As a result, the user often needs to master different aspects of hardware and software reprogramming in order to take full advantage of the degrees of freedom offered by a given device. Software-defined, open-architecture modems, designed to run on general-purpose processors using a largely known set of processing tools, represent one further step towards making underwater modems more open to the typical expertise and toolsets owned by an end user [7].

In this paper, we seek an intermediate point between the black box model of several commercial acoustic modems and the full freedom offered by reprogrammable, software-defined devices. Specifically, we would like to maintain the right of a modem's manufacturer to keep some aspects of their device closed, while at the same time permitting additional physical layer as well as network experimentation. We aim at avoiding that special-purpose proof-of-concept

prototypes have to be built to test, e.g., channel coding algorithms [10] as, in most cases, the time spent to implement such prototypes (that are usually less reliable than a well known state-of-art commercial modem), can be longer than the time spent to design and test the actual coding and networking algorithms.

One such solution has been provided by EvoLogics through a new version of the S2CR modem's firmware. This version allows the user to fully configure the modem, except that the modulation scheme and the related waveforms are fixed. Most commonly, protocol design and testing can easily be performed in software from the data-link layer upwards, as the manufacturers provide serial or Ethernet interface to their devices. Conversely, it becomes more difficult to design and test schemes that imply bit stream manipulation, as this task is normally carried out within the electronics of the devices and can rarely be set or disabled. It became of interest, however, to study algorithms and techniques that compress, encode or even modulate data and so work at a very low level in the perspective of the ISO/OSI communication stack. Implementing such algorithms in software is clearly inefficient as the processing time increases and more resources would be needed to let the executable code to work correctly, but would be very helpful to the scientific community who needs to evaluate and test new strategies. In this paper, we present a tool to test the full ISO/OSI stack, with software-based physical layer bit stream manipulation capabilities. To this aim, we incorporated the capability to fully exploit the low level firmware of the EvoLogics modem in both DESERT Underwater [19] and the Tool to Modem (To2Mo) framework [2].

The remainder of this paper is organized as follows. In Section 2, we review the capabilities of the EvoLogics modem and present the low-level firmware; in Section 3, we present different relevant scenarios for physical layer- and network-related aspects of underwater modem design. In Section 4 we present the results obtained in each use case through the integration of our DESERT Underwater [19] and To2Mo [2] frameworks. Finally, in Section 5 we provide our conclusions.

## 2 EVOLOGICS S2CR MODEM: LOW-LEVEL FIRMWARE IMPLEMENTATION

The EvoLogics GmbH S2CR series underwater acoustic modems [8] are integrated underwater communication systems that encompass an acoustic transducer with transmit/receive amplifier electronics, a USBL antenna and a digital communications stack. Different kinds of transducers with different physical properties are employed for different operational frequency bands, and support different kinds of practical applications. For example, the HS model, operating in the 80–120 kHz band, is suitable for short-range,

high bit rate applications, whereas the LF modem, operating in the 7–17 kHz band, is a better choice for long-range data transmission The modem amplifier is matched with the characteristics of each transducer type and is optimized in terms of end-to-end (transmit-receive) voltage response of the wideband hydroacoustic tract, and ensures a linear phase response throughout the available frequency band.

The digital communications equipment consists of analog-to-digital and digital-to-analog converters, DSP and FPGA implementing the S2CR physical layer modulation scheme, (S2CPhy for short from now on) [12], as well as an ARM processor, that implements the D-MAC data-link layer protocol [13] shipped by default with all EvoLogics modems. S2CPhy is based on the assumption that the received acoustic signal can be described by a sum of discrete multipath arrivals with random amplitudes and phases. In these conditions, spread-spectrum digital modulation schemes hold good promise for communications. In fact, spread spectrum signals are characterized by a short auto-correlation response. The broader the spreading, the shorter the response. After passing the underwater acoustic channel, the received signal contains a sum of multipath arrivals which, after matched filtering, can be represented as a series of time-shifted correlation responses. For a signal that is significantly spread, these responses can be isolated, and signal distortions due to multipath propagation can be eliminated or significantly mitigated.

S2CPhy modulates a linear frequency-swept carrier signal via phase keying. Thus, contrary to several digital underwater acoustic communication methods, an S2CR signal is characterized by two levels of modulation: the first one is applied to the carrier in the analog domain, whereas the second one encodes the information within the carrier phase through discrete signal processing. With frequency bands up to tens of kHz wide, S2CPhy signals can be shorter than 1 ms, and the physical bit rate can be on the order of tens of kbit/s. The frequency-swept carrier signals used by S2CPhy have the key advantage to decouple delayed multipath arrivals in the frequency domain after demodulation. This makes it possible to properly filter the strongest arrivals while strongly attenuating other multipath components. As a result, frequency-selective fading is reduced or eliminated.

Specifically, the S2CPhy implemented in the EvoLogics modems carries out the following tasks:

- modulation at the desired bit rate;
- estimation of the multipath intensity profile of the underwater acoustic channel at transmission time;
- selection of a discrete (and sufficiently intense) multipath arrival for receiver synchronization;
- packet and symbol synchronization;

- demodulation: processing of the received signals by focusing on the selected reference multipath arrival;
- ranging through the selection of a multipath arrival corresponding to the shortest propagation path from the source;
- positioning: estimation of time-phase differences in the detection of the acoustic signal on elements of the USBL transducer grid.

For research purposes, EvoLogics disabled the D-MAC data-link layer protocol running on the ARM processor and ran a simple sub-layer, providing access to the S2CPhy interface with the following command set:

- `gpio`: to control hardware pins that power or enable the modem hardware components;
- `config_dsp`: to configure such parameters as the source level and the detection threshold;
- `bitrate`: to set the bit rate for data transmission and reception;
- `send_sync_ctrl`: to transmit a packet with the lowest bit rate of 976 bit/s;
- `send_sync_data`: to transmit a packet with the bit rate set by the command `bitrate`;
- `rcv_sync_data`: to start signal detection; demodulation takes place upon a successful detection;
- `rssi` and `phase`: to return the phase and amplitude of each demodulated symbol of the most recent received packet;
- `stop_listen`: to disable the signal detection mode.

The signal transmission start and the duration of the signal reception (detection mode) can be controlled via command parameters. Command responses from S2CPhy contain timer values corresponding to the following events: signal detection stop, signal detection time, signal transmission start and current timer value. Having this values and tools, it is possible to implement protocols combining communication and localization tasks.

## 3 EXPERIMENT SCENARIOS

### 3.1 Scenario 1: software FEC techniques

The first scenario we propose is aimed at testing the transmission of a user-custom, channel-coded bit stream. This bit stream should be transmitted by the modem's hardware "as is," with no changes due to further coding or interruptions due to proprietary MAC protocols. This scenario is most typically found in forward error correction (FEC) experiments.

We chose two different encoding schemes, in order to analyze the effects of the channel errors on the messages, and to measure the bit error performance that can be achieved with either scheme. Both schemes work on symbols of 1 byte. The first code is a (15,11) Hamming code, capable of
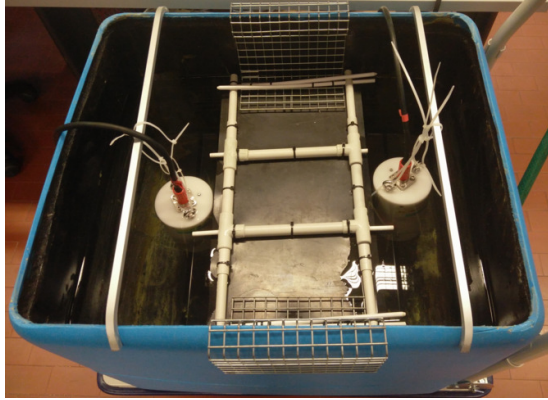
**Figure 1: Setup of the FEC experiment. The water tank used in the experiment is covered with phono-absorbing material.**



**Figure 2: Setup of the FEC experiment in air in Scenario 1. The signals experienced little if any multipath propagation.**



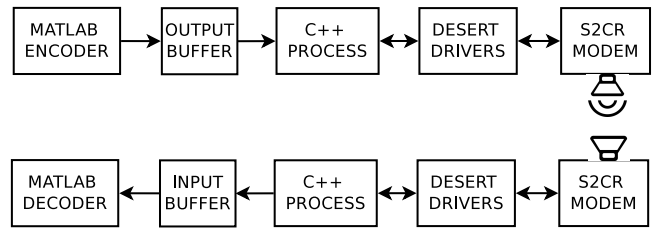**Figure 3: Structure of the Tool-to-Modem (To2Mo) framework in the Matlab-to-Modem (M2Mo) configuration used for the water tank test.**

correcting up to one symbol error, or to detect up to two erroneous symbols. In this case, the original message is composed of 22 8-bit symbols for a total of 176 bits and the encoded message sums up to 240 bits. The second code is a (31,21) Reed-Solomon code that can correct up to five symbols and detect up to 10 erroneous symbols. The message sent was composed of 21 8-bit symbols encoded into a total of 31 8-bit symbols or 248 bits. Among other results, this scenario makes it possible to observe and quantify whether the different correction capabilities of the two codes actually reflect into different transmission performance in the context of underwater acoustic communications. We remark that even though the experiment involves two known and well-tested encoding techniques, the scenario fits the analysis of other works such as, e.g., [10], where low-level access to the physical layer of the modem would have been advantageous, and would have provided greater flexibility for the presented results.

In order to test the feasibility of testing software implementations of channel encoding schemes through low-level firmware access, we set up a basic experiment to realize the previously described tests. We deploy the modems in two different settings: in clear water (using a water tank for laboratory tests) and in air. The water tank has size 60×70×70 cm³, and is shown in Fig. 1. The walls and the bottom of the tank have been covered with phono-absorbing material, so as to reduce the strength of multipath propagation in such a small-sized environment [16]. However, we have removed part of the phono-absorbing surface coverage layer. This way, about 50% of the water surface remains reflective, leading to a limited but detectable set of residual multipath arrivals. Contrary to the water tank setting, the tests in air make it possible to evaluate both coding schemes in a more controlled, multipath-free environment (see Fig. 2). The transmit

power was set to EvoLogics' modems predefined source level 2, which corresponds to 172 dB re 1 $\mu$Pa measured at 1 m from the transducer. The nodes are located 30 cm apart, both in water and in air.

The software that drives the test is built on top of the modem drivers that are part of the DESERT Underwater framework [19]. These drivers allow us to connect the software implementation of the coding schemes to the transceivers, thus providing them with a bit stream to be transmitted. The encoding and decoding algorithms were programmed in Matlab, and run in an instance of this environment as depicted in Fig. 3. These programs exchanged messages through the To2Mo tool [2]. We remark that Matlab executables cannot be used for real-time experiments, because the interpreted nature of the language would most likely imply exceedingly large running times. Still, it was possible to subdivide the test transmissions into multiple messages of a predefined length, known by both the transmitter and the receiver. These messages have been encoded using the symbol-based (15,11) Hamming code and (31,21) Reed-Solomon code mentioned above. The decoding was then attempted at the receiver side: as low-level firmware does not allow access to the acoustic
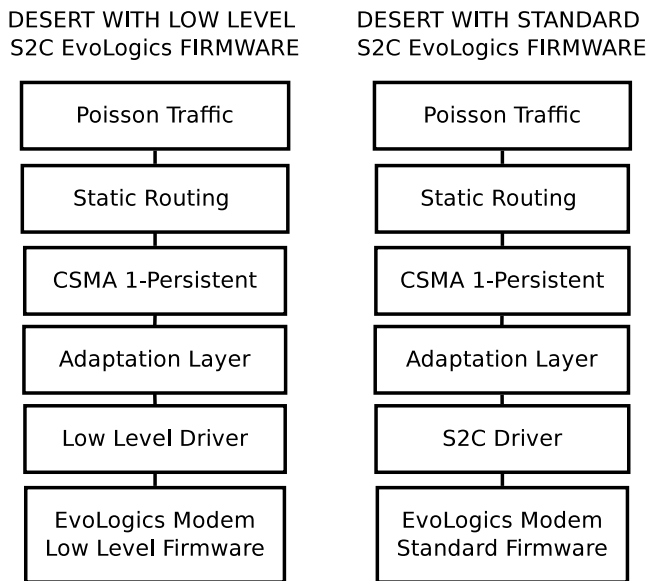
DESERT WITH LOW LEVEL
S2C EvoLogics FIRMWARE

| |
|---|
| Poisson Traffic |
| Static Routing |
| CSMA 1-Persistent |
| Adaptation Layer |
| Low Level Driver |
| EvoLogics Modem Low Level Firmware |

DESERT WITH STANDARD
S2C EvoLogics FIRMWARE

| |
|---|
| Poisson Traffic |
| Static Routing |
| CSMA 1-Persistent |
| Adaptation Layer |
| S2C Driver |
| EvoLogics Modem Standard Firmware |

**Figure 4: DESERT Underwater protocol stack employed in Scenario 2, showing use of either the standard EvoLogics S2CR firmware, or the low-level firmware.**

signal samples, we cannot employ soft decoding techniques and thus resort to hard decoding. Still, the possibility to test custom FEC techniques in a laboratory without the need to build specific hardware or software-defined modems is successfully demonstrated.

## 3.2 Scenario 2: networking experiment with bit rate control

The second experiment scenario is designed to test underwater networks where both the modem's bit rate and the raw bit stream to be transmitted can be fully controlled by the user. We consider the case where the transmit bit rate is equal to the actual data rate, i.e., where no coding is applied. Additionally, we compare the results to those obtained through the native message format of the EvoLogics S2CR modem firmware, which transmits short messages at a bit rate of about 1 kbit/s, with a rate-1/2 code, resulting in a net data rate of about 500 bit/s. For this experiment, we employed the DESERT Underwater framework with the patch of the ns-2 real time scheduler presented in [2].

The experiment setting involves three S2CR EvoLogics 18/34 White line Science Edition (WiSE) modems, having 16 kHz of bandwidth and a central frequency of 26 kHz. The transmit power was set to source level 3, that corresponds to 164 dB re 1 $\mu$Pa at 1 m from the transducer. In this scenario, we test the 1-persistent Carrier Sense Multiple Access (CSMA) MAC protocol in a single-hop network, where each
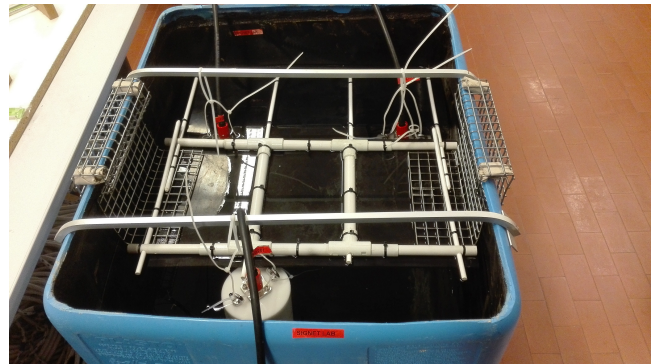


**Figure 5: Setup of the networking experiment in Scenario 2. Both nodes to the top of the figure are covered with phono-absorbing neoprene rubber, in order to mitigate multipath arrivals.**

device can possibly send packets to all other nodes. The packet size was fixed to 160 bits, and each point-to-point packet flow is characterized by Poisson packet arrivals of rate $1/10$ s$^{-1}$, so that every node generates traffic at a total average rate of one packet every 5 s. We employed the DESERT Underwater protocol stack presented in Fig. 4, where we could easily switch between the standard S2CR firmware and the low-level firmware by simply changing the DESERT drivers in the experiment configuration, without the need to rewrite any code. A picture of the tank experiment is shown in Fig. 5, where the nodes are placed in a triangular configuration, and the distance between the nodes is about 35 cm.

## 3.3 Use case Scenario 3: combining FEC and networking tests

The last scenario we consider relates to a fully configurable network, where a user can employ its own encoding and decoding schemes as part of a complete network protocol stack. In order to implement this setting, we extended the adaptation layer of DESERT (which handles the translation between ns2 packets and the actual bit stream), in order to include the desired FEC algorithm. The adaptation layer, as shown in Fig. 4, is placed between the MAC protocol and the modem drivers. With the standard EvoLogics S2CR firmware, adding FEC to this layer does not provide any significant benefits, as it would imply to add a completely independent outer coding layer to the already existing hardware-based 1/-rate FEC scheme. On the contrary, implementing FEC within the adaptation layer in association with the driver for the low-level firmware introduces the possibility to exploit a convenient software-based FEC system. Ultimately, this configuration could potentially enable the design and implementation of self-adaptive algorithms with dynamic FEC
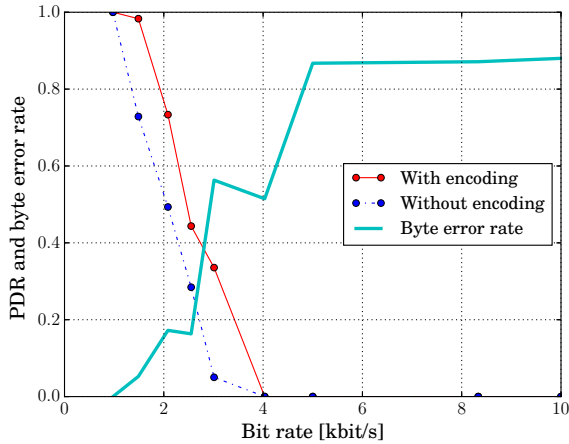
Figure 6: PDR and raw byte error rate for the (15,11) Hamming code, water tank. The packet size is 240 bits. The code allows to increase the bit rate by about 1 kbit/s while maintaining the same PDR.
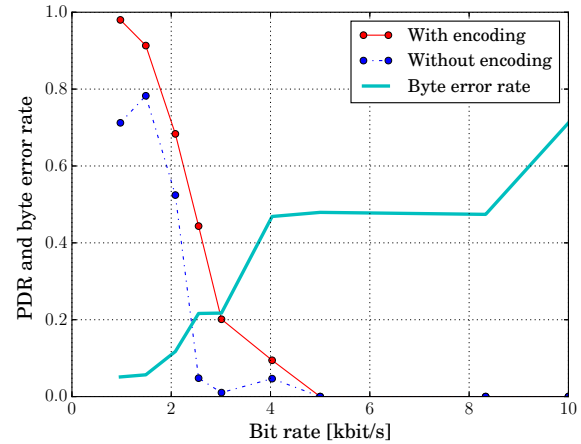


Figure 7: PDR and raw byte error rate for the (31,21) Reed-Solomon code, water tank. The packet size is 248 bits. The code compensates the packet errors up to a bit rate of ≈2.5 kbit/s, where the PDR starts decreasing below 0.5.

and hybrid ARQ. As this development is still in progress, we leave this scenario as a future extension of our work.

## 4 EXPERIMENTAL RESULTS

### 4.1 Scenario 1: FEC performance results

We quantify the correction capability of the considered Hamming and Reed-Solomon codes by measuring the packet delivery ratio (PDR), defined as the ratio of the total number of packets that were correctly received with and without encoding to the total number of transmitted packets. Along with these quantities, we also measured the total number of erroneous bytes (recall that both codes operate on bytes), in order to characterize the raw transmission performance in the different experiment settings. The byte error rate is particularly important because, despite the multipath mitigation obtained through the phono-absorbing tank cover, multipath propagation is still present due to the uncovered section of the water surface, and still impacts the reception capabilities. We will first consider the experiment in the water tank, and then compare its result to the second setting with all modems deployed in the air and transmitting at the lowest possible source level: we recall that multipath is highly attenuated in this second environment.

Figs. 6 and 7 show the PDR and raw byte error rate measured in the water tank as a function of the transmit bit rate, for the Hamming and Reed-Solomon codes, respectively. We observe that the PDR achieved by both codes is practically equal to 1 at the minimum bit rate of 1 kbit/s. As the bit rate increases, the error correction capability of the Hamming code (Fig. 6) decreases sharply, even if the PDR is also equal

to 1 for a bit rate of 1 kbit/s. This is reflected by the raw byte error rate, whose trend is a rapid increase as the bit rate increases from 1 to 10 kbit/s. The PDR falls to 0 for a bit rate of about 4 kbit/s: at this point, the raw byte error rate remains too high to allow the code to correct any transmissions.

A similar behavior is observed in Fig. 7, which shows the PDR achieved through the Reed-Solomon code as a function of bit rate. We observe a slightly better performance for a bit rate of around 4 kbit/s, where the higher correction capabilities of the code maintain the PDR above 0. For both the above experiments, we also tested the PDR performance using the standard EvoLogics S2CR firmware. The result was a PDR equal to 1 in both cases. This is due to the automatic adaptation of the bit rate to the channel conditions.

The in-air experimental results are shown in Fig. 8 for the Hamming code and in Fig. 9 for the Reed-Solomon code. In both cases, we achieve a higher PDR due to the better channel condition: at 4 kbit/s, the PDR of coded transmissions is about 0.9 using the Hamming code, and 0.8 using the Reed-Solomon code. This difference with respect to the performance in the water tank originates from the lower byte error rate that results from the strongly attenuated multipath arrivals.

### 4.2 Scenario 2: networking experiment

In this section we present the performance results for the system presented in Section 3.2. We consider a network of three nodes arranged in a triangular topology inside the water tank. The nodes transmit uncoded packets using a 1-persistent CSMA protocol. In Fig. 10 we evaluate the packet
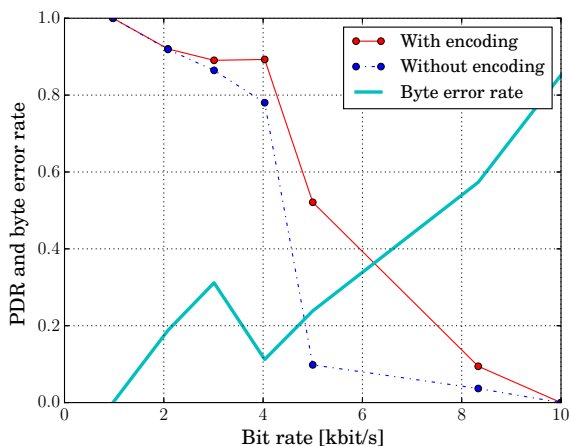
**Figure 8: PDR and raw byte error rate for the (15,11) Hamming code, in-air experiment. The packet size is 240 bits. The absence of significant multipath results in higher PDR with respect to the water tank test.**
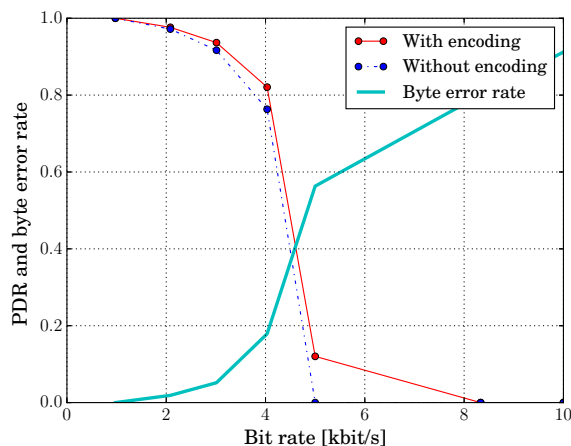


**Figure 9: PDR and raw byte error rate for the (31,21) Reed-Solomon code, in-air experiment. The packet size is 248 bits. The PDR remains significantly high for a bit rate of up to 4.5 kbit/s.**

delivery ratio (PDR) of the system as a function of the link data rate, which equals the PHY bit rate due to the lack of channel coding. The blue line shows the PDR achieved by the low-level firmware, whereas the red triangle represents the performance achieved through the standard instant message (IM) settings of the normal EvoLogics S2CR modem firmware. We remark that no FEC is employed when using the low-level firmware, whereas the EvoLogics S2CR firmware employs a rate-1/2 FEC code, hence the actual data rate is about 500 bit/s. With this configuration, the achieved PDR is about 0.8. Given that single-link connections in our water tank have a PDR close to 100%, packet losses in the network are due to a combination of collisions, node deafness, and additional signal deterioration caused by the additional strong multipath reflections that take place off the steel cages that encircle the transducers of the modems. The PDR results obtained with the low-level firmware range between 0.65 to and 0.82 for data rates between 1 and 2.25 kbit/s. These values are upper bounded by those of the standard S2CR firmware. The irregular behavior of the PDR curve in this range of data rates is due to collisions, multipath and energy per bit. For example, at 2 kbit/s we achieve better PDR than at 1.5 kbit/s due to the shorter packet duration, resulting in a smaller collision probability. After 2.25 kbit/s, the PDR starts to decrease significantly, because the higher data rate reduces the signal-to-noise ratio of the transmissions. We remark that in this experiment, on average, we obtained a higher single-link packet delivery ratio than in the FEC tests, where we left the water surface partly uncovered. Conversely, in this test we deployed a complete surface cover, so as to mitigate multipath as much as possible.
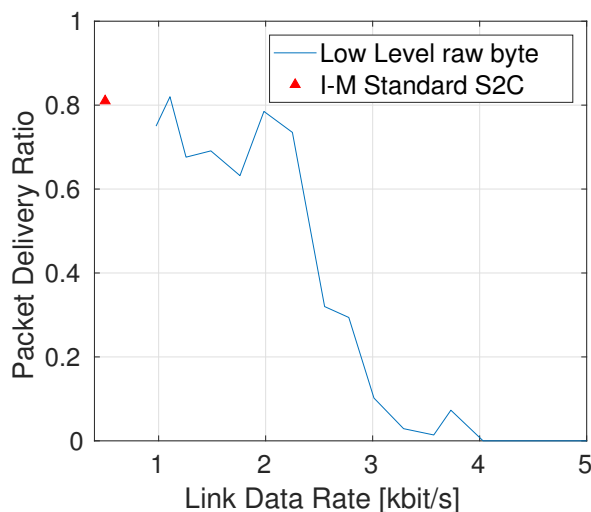


**Figure 10: PDR as a function of the bit rate for a three-node network using a CSMA MAC protocol to transmit uncoded bit streams. The data rate equals the PHY bit rate. The red triangle represents the performance achieved by instant messages transmission settings of the standard EvoLogics S2CR firmware.**

## 5 SUMMARY AND CONCLUSIONS

In this paper, we presented link-level and network performance results using a low-level modem firmware version that allows the user to reconfigure most parameters of the PHY of the EvoLogics S2CR acoustic modem. This includes the bit rate, the transmit power and the applied FEC scheme,

and constitutes an intermediate step between a black-box PHY layer and a fully software-defined modem. The higher flexibility does not require the user to know specialized programming languages (e.g., for FPGA and DSP programming), and still makes it possible to experiment with user-defined physical layer algorithms, such as channel coding schemes and data compression strategies. More generally, the user can force the modem to transmit any arbitrary bit stream, by turning off the PHY coding and bit rate adjustments. We tested this capability using the DESERT Underwater framework and its tool-to-modem extensions, in addition to a software implementation of Hamming and Reed-Solomon codes. This setting was employed to test the performance of the codes over in-water and in-air acoustic links. We also extended the test scenario to simple network and MAC protocol tests. While we conducted the tests in a laboratory tank, the signals propagating in this tank are subject to harsh multipath distorsion, especially if the tank walls and the water surfaces are not insulated with phono-absorbing materials. This is representative of highly reflective harbor environments, and is typically harsher than horizontal transmissions in open waters, except for the fact that it is impossible to reproduce Doppler distortion in a small tank.

Our experiments demonstrate the flexibility of low-level PHY access and the potential of this setting as an intermediate step between black-box and fully software-defined modem setup. For example, unlike many other hardware and software settings, our system makes it possible to directly experiment with adaptive coding techniques, hybrid ARQ, and adaptive medium access control with tunable bit rate.

## ACKNOWLEDGMENT

## REFERENCES

[1] F. Berning, T. Radtke, S. Rautenberg, M. Motz, and Nissen. 2014. A Realization of the Software Defined Radio Concept in an Underwater Communication Modem. In *Proc. UCOMMS*. Sestri Levante, Italy.

[2] F. Campagnaro, R. Francescon, F. Guerra, F. Favaro, P. Casari, R. Diamant, and M.Zorzi. 2016. The DESERT underwater framework v2: Improved capabilities and extension tools. In *Proc. UComms*. Lerici, Italy.

[3] G. Cario, A. Casavola, M. Lupia, and C. Rosace. 2015. SeaModem: a Low-Cost Underwater Acoustic Modem for Shallow Water Communication. In *Proc. MTS/IEEE Oceans*. Genova, Italy.

[4] M. Chitre, R. Bhatnagar, and W.-S. Soh. 2014. UnetStack: An agent-based software stack and simulator for underwater networks. In *Proc. MTS/IEEE OCEANS*. St. John's, Canada.

[5] E. Demirors, G. Sklivanitis, T. Melodia, S.N. Batalama, and D.A. Pados. 2015. Software-defined underwater acoustic networks: toward a high-rate real-time reconfigurable modem. *IEEE Commun. Mag.* 53, 11 (Nov. 2015), 64–71.

[6] Develogic Subsea Systems. 2017. (2017). http://www.develogic.de/

[7] H. S. Dol, P. Casari, T. van der Zwan, and R. Otnes. 2017. Software-Defined Underwater Acoustic Modems: Historical Review and the NILUS Approach. *IEEE J. Ocean. Eng.* 42, 3 (July 2017), 722–737.

[8] EvoLogics GmbH. 2017. (2017). http://www.evologics.de

[9] L. Freitag, M. Grund, S. Singh, J. Partan, P. Koski, and K. Ball. 2005. The WHOI Micro-Modem: An Acoustic Communications and Navigation System for Multiple Platforms. http://www.whoi.edu. (2005).

[10] A. Goalic, J. Trubuil, and N. Beuzelin. 2006. Channel Coding for Underwater Acoustic Communication System. In *Proc. IEEE Oceans*. Boston, MA, USA.

[11] AquaTec group. 2017. Underwater Communications. (2017). http://www.aquatecgroup.com/9-offshore-energy/12-communication

[12] K. G. Kebkal and R. Bannasch. 2002. Sweep-spread carrier for underwater communication over acoustic channels with strong multipath propagation. *The Journal of the Acoustical Society of America* 112, 5 (2002), 2043–2052.

[13] O. Kebkal, M. Komar, K. Kebkal, and R. Bannasch. 2011. D-MAC: Media access control architecture for underwater acoustic sensor networks. In *Proc. IEEE/OES Oceans*. Santander, Spain.

[14] H. Kulhandjian, L. Kuo, and T. Melodia. 2012. Extended Abstract: Development of a Reconfigurable Underwater Networking Testbed. In *Proc. ACM WUWNet*. Los Angeles, CA.

[15] M. Chitre, I. Topor and T.-B. Koay. 2012. The UNET-2 modem: An extensible tool for underwater networking research. In *Proc. MTS/IEEE OCEANS*. Yeosu, South Korea.

[16] F. Meneghello, F. Campagnaro, R. Diamant, P. Casari, and M.Zorzi. 2016. Design and evaluation of a low-cost acoustic chamber for underwater networking experiments. In *Proc. ACM WUWNet*. Shanghai, China.

[17] Teledyne-Benthos Acoustic Modems. 2017. (2017). https://teledynebenthos.com/product_dashboard/acoustic_modems

[18] J. A. Neasham, G. Goodfellow, and R. Sharphouse. 2015. Development of the "Seatrac" miniature acoustic modem and USBL positioning units for subsea robotics and diver applications. In *Proc. MTS/IEEE Oceans*. Genova, Italy.

[19] P. Casari, C. Tapparello, F. Guerra, F. Favaro, I. Calabrese, G. Toso, S. Azad, R. Masiero and M. Zorzi. 2014. Open-source Suites for Underwater Networking: WOSS and DESERT Underwater. *IEEE Network, special issue on "Open Source for Networking: Development and Experimentation"* 28, 5 (Sept. 2014), 38–46. http://desert-underwater.dei.unipd.it/

[20] Subnero. 2017. (2017). https://subnero.com/

[21] W. van Kleunen, N. A. Moseley, P. Havinga, and N. Meratnia. 2015. Proteus II: Design and Evaluation of an Integrated Power-Efficient Underwater Sensor Node. *Hindawi Int. J. of Distributed Sensor Networks* (Aug. 2015). http://dx.doi.org/10.1155/2015/791046

[22] L. Wan, H. Zhou, X. Xu, Y. Huang, S. Zhou, Z. Shi, and J.-H. Cui. 2015. Adaptive Modulation and Coding for Underwater Acoustic OFDM. *IEEE J. Ocean. Eng.* 40, 2 (April 2015), 327–336.

[23] J. Younce, A. Singer, T. Riedl, B. Landry, A. Bean, and T. Arikan. 2015. Experimental results with HF underwater acoustic modem for high bandwidth applications. In *Proc. Asilomar Conf. on SS&C*. Pacific Grove, CA.