A Versatile Lab Testbed for Underwater Sensor Networks

Yishan Su[1,2], Yuzhi Zhang[3,2], Son Le[2], Haining Mo[2], Li Wei[2], Yi Huang[2],
Zheng Peng[2] and Jun-Hong Cui[2]
[1] School of Electronic & Information Engineering, Tianjin University, Tianjin, China 300072
[2] Underwater Sensor Network Lab, University of Connecticut, Storrs, CT, USA 06269
[3] School of Marine Engineering, Northwestern Polytechnical University, Xi'an, Shaanxi, China, 710072
[1] *suyishan_tj@hotmail.com* [2] {*sonle, haining.mo, li.wei, zhengpeng, yih11007, jcui*}*@engr.uconn.edu*
[3] *zhangyuzhinpu@hotmail.com*

***Abstract*—The past few years have witnessed a significant progress in the research of underwater wireless networks. Many protocols and applications have been proposed. However, due to high costs of field tests, not all research work can be evaluated in real environment. To facilitate the researches of underwater networks, we develop a lab testbed based on the Orthogonal Frequency Division Multiplexing (OFDM) lab modems and a software protocol stack: SeaLinx. The testbed has the same design as the real system we deployed in sea. With the help of the lab testbed, we can conduct various tests in the lab including protocols evaluations and underwater application developments.**

I. Introduction

UWSNs (Underwater Sensor Networks) have gained tremendous attention in recent years because of their wide range of applications such as scientific exploitation, commercial exploitation, disaster prediction and coastline protection [1] [2]. This motivates more research for a reliable and efficient network design, from the physical layer to the application layer of the UWSN protocol.

However, unlike RF networks, high frequency electromagnetic waves do not work well in the water due to high absorption. For this reason, acoustic communication becomes the most commonly and practically used method for long range underwater wireless communication. In the past few years, due to the expense of building real underwater network systems and conducting field experiments in the ocean, most research is based on theoretical analysis and simulations. The complicated underwater environment and dynamic of communication systems and environment, however, cannot be well reflected by pure software simulators.

In order to effectively evaluate the performance of underwater networks, experiments are expected to be done with the real acoustic communication system. Constructing a user friendly infrastructure is also critical for those who wish to efficiently develop, deploy and debug applications in realistic underwater environments. To fill the gap between simulators and field trials, we target at developing a lab testbed for UWSNs which can be used to develop, test and evaluate different network algorithms, protocols and applications.

In this paper we will introduce our underwater lab testbed system, which is developed in laboratory based environments. Our testbed consists of both hardware module, Orthogonal Frequency Division Multiplexing (OFDM) lab modem which is an advanced lab modem using OFDM technique, and software module, a protocol stack named SeaLinx [3], which is easy to install and assemble while providing a lot of freedom for developers and users.

The rest of the paper is organized as follows. In section II, we briefly introduce the related work of underwater testbed. The hardware and software modules of the testbed are presented in section III. In section IV, some cases studies are given to show some functionality of the testbed. Finally, conclusions are drawn in section V.

II. Related work

Several testbeds for UWSNs have been proposed for the past few years. [4] is an early testbed developed by US navy. It is a suspended electronic pressure canister with integral battery pack, which includes four transducers. The testbed supports basic research into the physical transmission channel. It can provide some means for raw data acquisition and, furthermore, can help researchers to develop modem and transducer hardware. The main purpose for this testbed is for hardware developer and it is a high-cost testbed since it is deployed in sea.

In [5], the researchers develop a lab testbed in Kookmin University based on ARM9 processor. In that paper they specify to develop baseboard using characteristics of ARM9 processor based on low power consumption to enhance the performance and to improve a variety of restrictions. The demonstration shows that the six-node network of the testbed can fulfill sensing and data transmission task by implementing TDMA/CDMA MAC protocols. However, the lack of software stack makes it difficult for protocol and application development.

B. Chen et al. provide an out-of-water testbed that emulates underwater acoustic communications using various hardware modules including WHOI Micro-Modems, an audio interface to process audio signals, a gumstix motherboard, and two desktop PCs [6]. The main purpose of this testbed is to provide users a configurable environment so to evaluate the

978-0-933957-40-4 ©2013 MTS

performance of underwater vehicle formation and steering algorithms.

Since there are many mature simulators for ground based networks, some institutions promote smooth transition between simulation and experimentation by providing a transition path from simulators to real underwater modems [7][8]. For example, in paper [7], the authors joint Bellhop and NS-MIRACLE to provide a flexible and customizable environment. In these two testbeds they present a new generic architecture to simulate, emulate and test new protocols for underwater sensor networks in real life experiments. The most advantage for these kinds of testbeds is that they allow using the same source codes through simulators then to the real at-sea testing. Using the proposed framework, the code written for simulation test with NS-2 can be reused to perform real tests in the water. This will make it easier for the networking community to design and test protocol stacks. However, by adopting NS-2 design, all events are managed by a scheduler and occur in sequence, which might be inefficient in cases when one event that is waiting on I/O stops other unrelated ones from being executed.

In the Underwater Sensor Network lab at the University of Connecticut, we develop a lab UWSNs testbed , named Aqua-Lab [9] and a lake testbed, named Aqua-Tune [10]. For Aqua-lab, it consists of a water tank, several WHOI Micro modems and software Application Programming Interfaces (API). For Aqua-tune, each node contains a floating platform for an acoustic modem, a RF based monitor and remote control system and the software platform: Aqua-Net that accommodates the protocols building an underwater network. These testbeds offer a variety of services that open many potentials to the researchers and application developers. However, there are two main drawbacks for these testbeds. Firstly, the real modems may be too expensive for some institutions. Secondly, the software structure named Aqua-Net allows only one protocol suite per modem, including a MAC, network, transport, and application protocol. Along with this is the inconvenience of inter-layer communication. To overcome the flaws and test the most recent acoustic modems, we develop a new lab testbed, which equipped with several high speed underwater OFDM modems and can be used for various tests.

III. Testbed Description

This testbed is hosted in the UnderWater Sensor Network (UWSN) Lab at the University of Connecticut. It consists of a water tank, a set of communication hardware and a software suite. The communication hardware consists of four OFDM lab Modems [11] and one server. The software suite is composed of a modem driver and SeaLinx, which is a layered protocol stack for UWSNs. In this work, we will present the structure of the lab testbed and use some cases study to demonstrate its advantages.

A typical hardware setting of our lab is shown in Fig. 1, with the structure of each node is illustrated in Fig. 2. Each UWSN node runs SeaLinx, our protocol stack, which interfaces with the OFDM modem through a modem driver.

A. Hardware

One major target of our lab testbed is to provide researchers with test experiences close to real world experiments at affordable costs. To achieve this goal, we setup the hardware components as follows. A water tank is utilized to provide a real underwater environment. Four OFDM modems are placed into the water tank for acoustic communications. The modems are connected to a server or one laptop for each modem, upon which instances of SeaLinx and the OFDM modem driver are running.

1) Acoustic modem: Since OFDM is an effective method to mitigate the multi-path effect in underwater which seriously degrades the efficiency of acoustic communications, we employ the OFDM modem developed by the UWSN lab at the University of Connecticut as the physical layer communication device. The lab modem is consisted of hydrophones and data processing box. Besides this, the voltage source and cables are needed to set up the physical layer device [11].

The modem sends out data in packets. Each data burst consists of two parts: the preamble and the data payload. The preamble is used for packet detection and synchronization which has a duration of 221.9 ms. The data payload contains one or more ZP-OFDM (Zero-Padded OFDM) blocks. As each packet can have 0 byte or more bytes, the modem will divide a packet into one or multiple data blocks, whose size depends on the transmission mode. The maximum number of data blocks in one packet is 16, and the duration of each OFDM block is $T = 170.7ms$. There is a guard interval between two data blocks. The default ZP guard interval is of $Tg = 50ms$, which can be changed depending on the channel delay spread. The transmission mode and corresponding block and packet size of the OFDM modem are listed in Table 1. And Fig. 3 shows the packet format in the physical layer.

The OFDM lab modem is implemented on OMAP L137, with an integration of a transmitter and a receiver to support two way communications. For the real-time implementation, TMS320C6713 is used in the board. It is among the most powerful floating-point DSPs from the TMS320C6000 family, whose core has eight independent functional units, including two fixed-point ALUs, four floating-/fixed-point ALUs, and two multipliers. It can execute up to eight instructions simultaneously in one cycle. The board has a plenty of peripherals

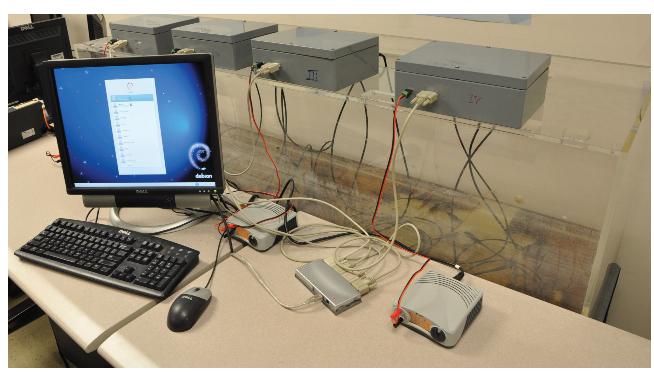

Fig. 1. Testbed Overview

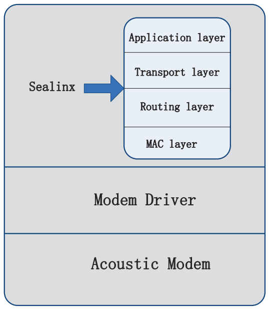

Fig. 2. Architecture of a node

TABLE I
TRANSMISSION MODE TABLE

Mode	Bytes per block	Bytes per packet
1	38	608
2	80	1280
3	122	1952
4	164	2624
5	248	3968

that facilitate the development. Each DSP board is equipped with one 16-bit AIC23 codec, one enhanced direct memory access (EDMA) controller, two multichannel buffered serial ports (McBSPs), 4 Mbytes external dynamic RAM (DRAM), which is sufficient for our implementation.

The majority of the implementation effort is algorithm optimization and programming optimization. For the algorithm optimization, proper algorithms are adopted for those computationally intensive operations. For example, the overlap-add and FFT-based algorithms are used to perform linear convolution; Adaptive searching algorithm is implemented for Carrier Frequency Offset (CFO) estimation, recursively compute the auto-correlation. For the programming optimization, we apply some coding strategies, such as loop unrolling and compiler directives, which aim to speed up the processing. Furthermore, the hardware resources on the DSP board is used efficiently such as Very Long Instruction Word (VLIW) architecture and on-chip memory. The 32-bit timer on the DSP board is used to measure the execution time. As a result, the processing time of one OFDM block is 38.6ms for Single Input Single Output (SISO) OFDM and 76.65ms for Multiple Input Multiple Output (MIMO) OFDM. The per-block processing time is much less the block duration 220.7ms, which achieves real-time processing and allows more functionalities to be implemented in the future.

2) Server: In our lab testbed, the four OFDM modems can be connected to the server through a USB to RS232 converter or each of them can be connected to an individual computer. SeaLinx and the modem driver are running on the server to control the lab modems to conduct acoustic communications.

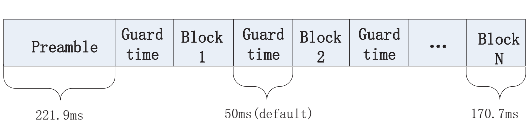

Fig. 3. Physical Packet format

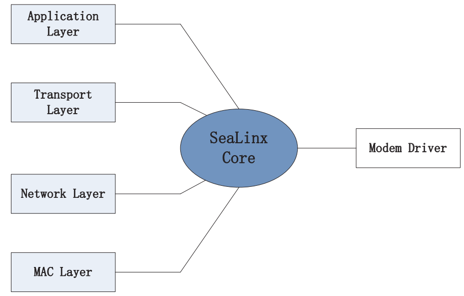

Fig. 4. Structure of SeaLinx

B. Software

Sealinx provides a set of Socket based APIs which allow developers to easily plug in one protocol running as a single layer or multiple protocols running on corresponding layers within the protocol stack. The stack can also support simultaneous protocol bundles to support various applications running on a single node as well as a cross-layer communication method.

Using Sealinx, one can efficiently test the performance of an individual protocol or a combination of multiple protocols. Sealinx also provides configuration files allowing users to dynamically configure network topology, data packet length, data rate, transmission power and so on on the fly. Furthermore, SeaLinx provides a set of cooperative processes instead of a monolithic process which includes a core module, a modem driver and protocol modules as shown in Fig. 4. The purpose of multiple processes is that it is more safety compare to monolithic process.

1) Protocol organization: SeaLinx is a layer based stack including four layers: MAC, network, transport and application. All these four module are connecting to the core module as shown in Fig. 4.

Each layer contains one or multiple protocols which provide different functionality. For example, we have implemented UW_ALOHA and Slotted FAMA in MAC layer and dynamic routing in network layer. When we initialize a protocol module, we should specify the lower layer protocol to which it connects. For most of layer-based protocol stacks, a protocol can only communicate with the protocols in the layer right above or underneath. However, SeaLinx allows one layer to communicate with another directly if the network protocol can handle the message. For example, the application can send a message to the network protocol to request for information without sending a message to the transport layer first.

For the core module, it is responsible for receiving then forwarding a message from one module to another. With the help of core module, cross layer communication becomes possible. There are two message flows in SeaLinx external and internal. For the external flow, the message is passed from one node to another through the core module. While for the internal flow, the message resides within one single node.

2) Modem driver: Relying on the OFDM Modem driver, Sealinx provides well-formatted APIs completing different operations on the OFDM modem, which saves developers the hassle of directly interacting with the modem hardware. The driver separates upper layer protocol from modems which enables the a protocol can be run on different modems with minimal change to its code

IV. Case Study

In this section, we show the experiment results of several experiments using the testbed.. These experiments include physical layer communications, a network protocol and a application test, from point to point communication experiments to small scale network tests.

A. Point to point MAC test

In this part, we evaluate and compare the performance of two underwater MAC protocols which are widely used: Slotted FAMA [12], which is a handshaking based MAC protocol and UW_ALOHA, which is a random access based MAC protocol, in a point to point scenario. The main goal for this test is to find out the upper bound of the goodput for both protocols while considering the features of the hardware. In this test, two nodes are deployed in the water tank. One of the node is the sender, and the other one is the receiver. We use static routing on the network layer, a plain transport layer and Possion traffic generator on the application layer. The packet length is fixed to be 50 and 400 bytes respectively while varying packet generation rate have been tested to study its impact on the goodput, which is measure as the number of successful data transmission per unit time, of these two MAC protocols.

Fig. 5 and Fig. 6 show the goodput of these two MAC protocols with different packet length. For each parameter setting, we run the protocol for 10^4 seconds. For a long packet setting, the plot shows that the goodput of SFAMA is about 40% higher than UW_ALOHA and which increases to 100% for a short packet scenario. Obviously, SFAMA can get a higher goodput performance compare to UW_ALOHA in this scenario. This is because SFAMA increases its efficiency by supporting a packet chain transmission which allows multiple packets transmission for a successful reservation. So the figures show that the gaps of the goodput between these two protocols are increased when the packet generation rate gets higher. However, SFAMA achieves a higher goodput performance at the expense of more transmissions of control packets. The result indicates the SFAMA may be more suitable for a heavy load network.

These two figures also show that the goodput for long packet settings is much higher than short packet scenario. This is because the total transmission time of a packet is summation of a fix preamble time and a variable data payload. A shorter packet will decrease the efficiency of the transmission. This may be a inspiration for upper layer design to reduce the transmission of some short packets.

B. Network MAC test

In this part, we conduct a MAC protocol test in a small scale network. We deploy four nodes in the water tank, one of them is the receiver and the rest of them are there senders. We test the same protocols as we did in the previous test: SFAMA and UW_ALOHA. All the other settings are the same as the point to point MAC test. In this test, the packet length is fixed to be only 400 bytes while varying data rate from $0.05packet/s$ to $0.3packet/s$.

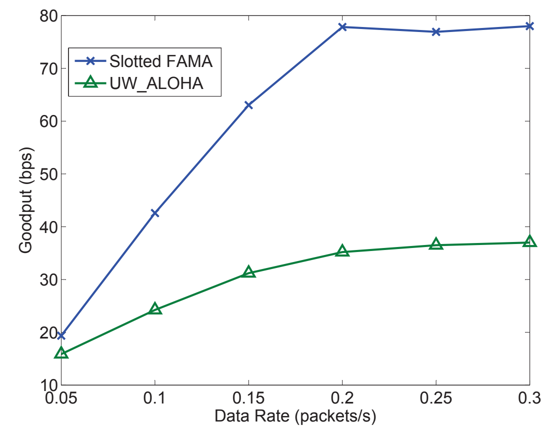

Fig. 5. Goodhput with Varying Data Rate (50B)

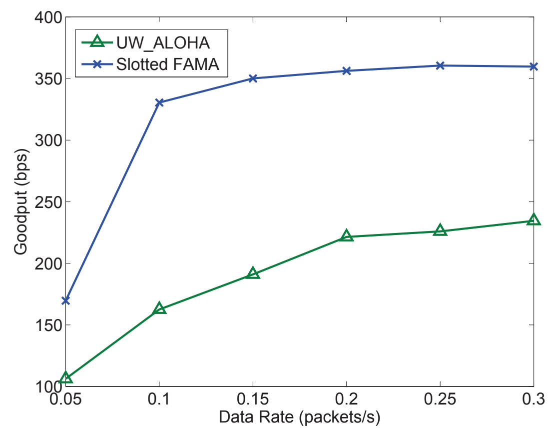

Fig. 6. Goodhput with Varying Data Rate (400B)

Fig. 7 depicts the network goodput with varying the packet generation rate for the two target MAC protocols. This figure shows that SFAMA outperforms UW_ALOHA regarding the goodput since the latter one incurs a high collision probability because of its random access scheme. Additionally, the plot also shows that the highest goodput they can reach is almost the same as that they reached in the point to point scenario.

From the two tests above, we have some knowledge of the upper bound of the goodput for these two different MAC protocols in a real network system.

C. Application for testbed

For this lab testbed, we also develop an application for image and message transfer, named Aqua-chadle. It realizes functionalities including image transfer, compression and display as well as message transfer and display. This application is developed with Ubuntu and Qt. For example, for image transfer, at the sending side, the application layer is responsible for splitting the image into chunks and numbering these chunks. After that it will send the message to the lower layer. At the receiver side, the application layer is responsible for reassembling and displaying images. The transport, network and MAC layer can provide network service. Finally, the

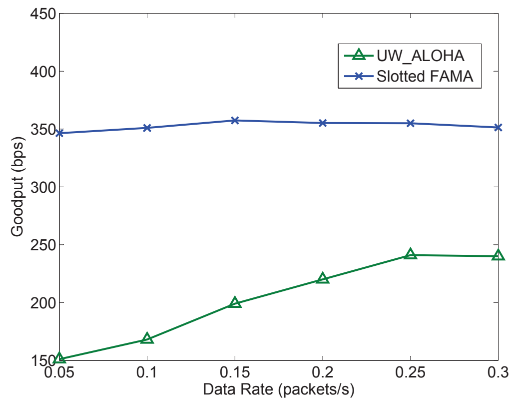

Fig. 7. Goodhput with Varying Data Rate

OFDM modems transmit the data packets. The Graphical User Interface (GUI) of this application is shown in Fig. 8

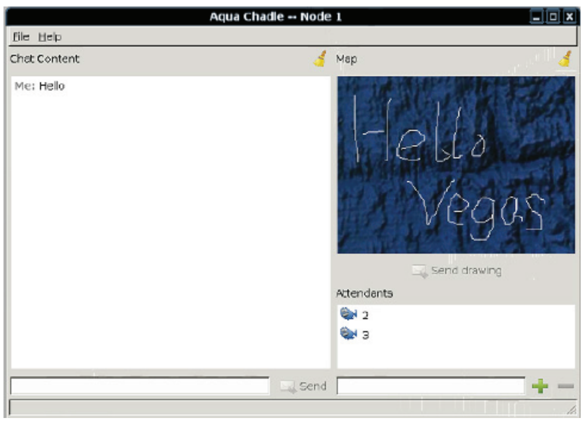

Fig. 8. GUI of Aqua-chadle

V. Conclusion

In this paper, we present a configurable lab testbed for underwater acoustic sensor networks. The water tank, software suite and OFDM modems provide the capability of conducting acoustic communications in underwater environments. Furthermore, the software suite allows testbed users to easily integrate their network protocols and applications into the testbed. In this way, users can test and evaluate network protocols and develop applications for UWSNs in a laboratory environment.

Acknowledgment

We thank Dr. Shengli Zhou and his team for helping set up the lab testbed.

References

[1] J.-H. Cui, J. Kong, M. Gerla, and S. Zhou, "Challenges: Building Scalable Mobile Underwater Wireless Sensor Networks for Aquatic Applications," *IEEE Network*, vol. 20, no. 3, pp. 12–18, 2006.

[2] J. Heidemann, W. Ye, J. Wills, A. Syed, and Y. Li, "Research Challenges and Applications for Underwater Sensor Networking," in *Proceedings of IEEE Wireless Communications and Networking Conference*, Las Vegas, Nevada, USA, 2006, pp. 228–235. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1683469

[3] S. Le, Z. Peng, J.-H. Cui, H. Zhou, and J. Liao, "Sealinx: A multi-instance protocol stack architecture for underwater networking," UCONN CSE, Tech. Rep. UbiNet-TR13-05, 2013.

[4] V. K. McDonald, J. A. Rice, and C. L. Fletcher, "An underwater communication testbed for telesonar rdt&e," in *OCEANS'98 Conference Proceedings*, vol. 2. IEEE, 1998, pp. 639–643.

[5] Y.-P. Kim, J.-I. Namgung, N.-Y. Yun, H.-J. Cho, I. A. Khan, and S.-H. Park, "Design and implementation of the test-bed for underwater acoustic sensor network based on arm9 processor," in *Embedded and Ubiquitous Computing (EUC), 2010 IEEE/IFIP 8th International Conference on*. IEEE, 2010, pp. 302–306.

[6] B. Chen and D. Pompili, "A testbed for performance evaluation of underwater vehicle team formation and steering algorithms," in *Sensor Mesh and Ad Hoc Communications and Networks (SECON), 2010 7th Annual IEEE Communications Society Conference on*. IEEE, 2010, pp. 1–3.

[7] R. Masiero, S. Azad, F. Favaro, M. Petrani, G. Toso, F. Guerra, P. Casari, and M. Zorzi, "Desert underwater: An ns-miracle-based framework to design, simulate, emulate and realize test-beds for underwater network protocols," in *OCEANS, 2012-Yeosu*. IEEE, 2012, pp. 1–10.

[8] C. Petrioli and R. Petroccia, "Sunset: Simulation, emulation and real-life testing of underwater wireless sensor networks," *Proceedings of IEEE UComms 2012*, pp. 12–14, 2012.

[9] Z. Peng, J.-H. Cui, B. Wang, K. Ball, and L. Freitag, "An underwater network testbed: design, implementation and measurement," in *Proceedings of the second workshop on Underwater networks*. ACM, 2007, pp. 65–72.

[10] Z. Peng, S. Le, M. Zuba, H. Mo, Y. Zhu, L. Pu, J. Liu, and J.-H. Cui, "Aqua-tune: A testbed for underwater networks," in *OCEANS, 2011 IEEE-Spain*. IEEE, 2011, pp. 1–9.

[11] H. Yan, L. Wan, S. Zhou, Z. Shi, J.-H. Cui, J. Huang, and H. Zhou, "Dsp based receiver implementation for ofdm acoustic modems," *Physical Communication*, vol. 5, no. 1, pp. 22–32, 2012.

[12] M. Molins and M. Stojanovic, "Slotted FAMA: a MAC Protocol for Underwater Acoustic Networks," in *IEEE OCEANS'06*, Sigapore, 2006, pp. 1–7. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.120.677&rep=rep1&type=pdf