A Prototype Underwater Acoustic Sensor Network Platform with Topology-aware MAC Scheme

Fei Hu*, Paul Tilghman, Yamin Malkawi
Rochester Institute of Technology
Computer Engineering Department,
Rochester Institute of Technology
83 Lomb Memorial Drive,
Rochester, New York 14623-5603
E-mail: fxheec@rit.edu
*Corresponding author

Yang Xiao
Department of Computer Science,
The University of Alabama
101 Houser Hall
Box 870290
Tuscaloosa, AL 35487-0290 USA
E-mail: yangxiao@ieee.org

Abstract: Underwater Wireless Sensor Networks (UWSNs) can be used in ocean exploration and lake pollution monitoring. They use acoustic media to achieve sensor communications. UWSNs are essentially different from terrestrial RF-based sensor networks due to its highly variable, long propagation delay, and mobility nature. In this paper, we first describe our experimental design of low-cost underwater sensor nodes for deployment in a shallow underwater environment. Then, we propose two UWSN Medium Access Control (MAC) enhancements that can adapt to long acoustic delay: 1) topology-ware scheduling scheme to save more energy, and 2) a coordination protocol among neighbours to reduce the frame queuing delay during MAC operations. Our experimental results show the energy efficiency of our UWSN MAC protocol.

Keywords: Underwater sensor networks, MAC, wireless communications, acoustic communications

Reference to this paper should be made as follows: .

Biographical notes: Fei Hu received the BS and MS degrees from Shanghai Tongji University (China) in 1993 and 1996, respectively. He received the first Ph.D. degree in the field of Telecommunication Engineering from Shanghai Tongji University (China) in 1999 and received his second Ph.D. from the Department of Electrical and Computer Engineering at Clarkson University (USA) in 2002. He is currently an assistant professor in the Computer Engineering Department at RIT, New York. His research interests are in cognitive radios, ad hoc sensor networks, 3G wireless and mobile networks, and network security.

Paul Tilghman is an undergraduate student of Computer Engineering Department, Rochester Institute of Technology

Yamin Malkawi is an undergraduate student of Computer Engineering Department, Rochester Institute of Technology

Yang Xiao is currently with Department of Computer Science at The University of Alabama. Dr. Xiao was a voting member of IEEE 802.11 Working Group from 2001 to 2004. He is an IEEE Senior Member. He currently serves as Editor-in-Chief for *International Journal of Security and Networks* (IJSN), *International Journal of Sensor Networks* (IJSNet), and *International Journal of Telemedicine and Applications* (IJTA). His

research areas are wireless networks, mobile computing, network security, and telemedicine. He has published more than 190 papers in major journals (more than 50 in various IEEE Journals/magazines), refereed conference proceedings, book chapters related to these research areas.

1 INTRODUCTION

Underwater wireless sensor networks (UWSNs) have a large potential set of useful applications. A sensor network distributed in a reservoir or source of water can be very important for monitoring the water quality of the level of pH, toxicity, salinity, etc. Additionally there is a growing area of research using distributed sensor networks for anti-submarine warfare, torpedo defense, mine countermeasures, and other underwater defense applications.

Terrestrial and underwater sensor networks have vast disparities not only between their operating environments but also between their technical developments. For example, terrestrial sensor networks use RF (radio frequency) as wireless medium. However, RF propagation has very limited communication range, e.g., Mica-2 transmit range has been measured as less than one meter in fresh water (Heidemann et al. 2006). Even through long-wavelength RF can penetrate water for a longer distance, it requires large transmit power and large antenna, making it inappropriate for small and low-power sensor nodes. Hence, underwater acoustic communication provides an important alternative since acoustic sound travels faster and longer in water than in air (Heidemann et al. 2006).

The speed of sound under water is typically around 1500 m/s. RF propagates over air at approximately 3×10^8 m/s, five orders of magnitude faster than the sound under water. Furthermore, the speed of sound changes dramatically (up to 100 m/s) due to pressure, salinity, and temperature (Heidemann et al. 2006). The slow propagation time also causes interference problems among multiple signals. For example, once a given symbol (or bit) of information reaches a receiver, it continues to propagate through the medium. In shallow water, reflections off the waters surface as well as the bottom are common. This reflection can potentially continue to reverberate inside the acoustic channel, and arrive at the receiver again. This information can combine with another symbol or symbols, causing an effect known as intersymbol interference (ISI) (Sozer et al. 2000). Because of the slow propagation time, one symbol may interfere with many other symbols as it continues to reverberate throughout the channel. Additionally bandwidth of the underwater channel is limited to approximate 100 kHz due to an absorptive factor of water at frequencies above 100 kHz. For longer range applications, communications are limited to an even more restricted bandwidth as higher frequencies attenuate more quickly under water.

Most commercially available underwater communication systems (LinkQuest; DSPComm) are designed for long range communications with link distances of several kilometres. These modems can carry sustained data rates of approximately 100 bps to 40 kbps. Furthermore, these systems function only as modems and they are not capable of adding any of the protocol layers above the data link layer without the addition of another processing unit, i.e., laptop or microcontroller. The authors in (Yang et al. 2002) describe a design of a low cost sensor node for ad-hoc underwater communications with an interface to a surface buoy. This system uses Amplitude Shift Keying (ASK) modulation; however achieved bit rate is not specified. The use of ASK modulation underwater is problematic due ISI caused by reflections in an underwater channel. The system allows for no easy expansion of its current capacity and throughput without a complete redesign of the hardware. Commercially available underwater communication systems have an extremely high cost, i.e., US$10,000 or more (Heidemann et al. 2006; LinkQuest). The necessity for a cost effective system to function as an underwater testbed is paramount.

There are very few works on UWSN MAC design. The authors in (Xie et al. 2000) developed code distribution techniques with CDMA on MAC for underwater acoustic networks. The authors in (Molins 2006) extended radio-based sensor network MAC approaches to UWSNs based on TDMA principle. The paper (Rodoplu et al. 2005) adapted Carrier sense multiple access (CSMA) techniques for underwater networks. An upper bound on the performance of a land sensor network for multi-hop sensor networks was presented in (Gibson et al. 2007a), and the bound is achievable with a perfect scheduling algorithm. The paper (Gibson et al. 2007b) provides a multi-hop analysis under a string topology under Aloha protocol. The paper (Syed et al. 2006) presented a reservation based MAC protocol, called Tone Lohi (T-Lohi), which detects and counts the number of contenders during the reservation and uses this information in building a traffic adaptive back-off algorithm. The slow propagation enables nodes to detect and count contenders, as long as the contention frames occupy the channel with duration much less than the propagation delay. However, we believe that the opportunities in underwater MACs have not been fully explored, particularly in low-cost and short-range networks.

In this paper, we first introduce our UWSN design hardware/software prototype design with acoustic communication capabilities, while transport layer, routing layer, and some other protocols are omitted. Our system architecture utilizes a single fixed-point DSP, to handle all signal processing and communications, coupled with an analog board responsible for amplification and signal conditioning. Then we propose two MAC enhancements for reducing energy consumption and queuing delay. Our developed system to be reported in this article is a prototype that can provide a low-cost test-bed to be used for short-to-

medium range sensor networks. Since an UWSN typically uses a tree-based routing architecture to collect sensing data from the deep water to the surface in a hop-to-hop relay approach, it is not necessary to get all neighbors involved into the MAC scheduling calculations. We will first propose a topology-aware MAC scheduling scheme to save more energy for those neighbors not in the relay path. Then we adopt a coordination protocol among neighbors to reduce the frame queuing delay during MAC operations.

2 UNDERWATER NODE DESIGN

While this design's focus lies in transferring traditionally hardware-centric tasks, such as modulation and demodulation, into more easily configurable software routines to create an end unit with relatively low cost, one area of hardware (transducers) still represents a significant cost for any underwater communication platform. Hydrophones, used to convert the electrical signals into sound waves and vice versa, cost approximately US$1000, even for small transducers designed for shorter range applications (International Transducer Corporation). Therefore a cheaper alternative than commercially available hydrophones was needed. Two concepts were used to construct cheap, but effective transducers.

The first concept we came up with was to utilize the back of a sports-watch, with two leads soldered onto the piezzo-disc used to make the watch beep. While this alternative is proved to be effective, optimal frequency response of all sports-watches tested did not occur until at least 10 kHz or higher. For this application good frequency response would be needed in the 3 to 4 kHz range (to reduce the number of baseband samples taken at the analog-to-digital (ADC) converter without re-sampling, further explained in the later of the section about software). These hydrophone alternatives can be created fairly inexpensively, depending on the cost of the watch.

The second concept we used is a small loud speaker in which the paper cone was waterproofed. These speakers produced an audibly louder tone at the desired frequency range, and were used as both the transmitting and receiving hydrophones, shown in Figure 1. The cost of this solution is around US$1.50, and this allows effective communication for the desired range without the expense of traditional transducers.

Each underwater node needs to interface with sensors which provide information about the environment being monitored. These sensors can cover a large variety of fields and such an underwater node may have to interface with a large number of sensors providing a variety of data. Our sensor node, like many others, deals strictly with voltage supply sensors. These sensors change their voltage in reaction to changes in the phenomenon they monitor. For this application two sensors were used, pH and temperature. Regardless of the type of sensor used, the signal must be conditioned so that it produces a valid signal for the analog-to-digital converter (ADC). The signal conditioning circuits for the two sensors used are provided here.

The pH amplifier has a constant gain of 2.4. The pH amplifier design consists of a non-inverting amplifier, as shown in Figure 2. An operational amplifier which can accept high impedance sources was needed because the Sensorex pH sensor used (S200CD) has an impedance of 50MΩ, and the underwater node's ADC can only accept at most 5kΩ sources. The temperature amplifier has a constant gain of 2, and this helps better cover the analog range covered by the ADC and provide greater precision

Figure 1 *Waterproofed loudspeakers serving as hydrophones*

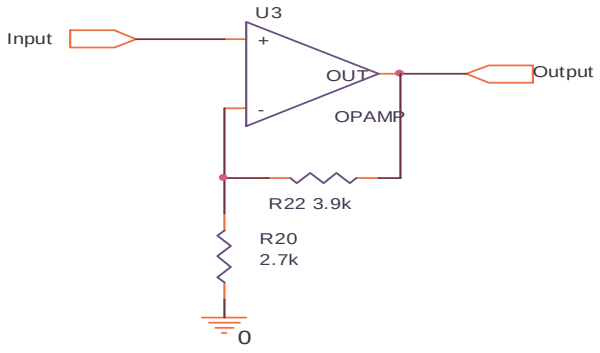

Figure 2 *pH Signal Conditioning Circuit*

To make local intelligent sensor data processing (such as noise removing) and to execute acoustic communications protocols with neighbouring underwater node, a microcontroller is used to control all sensors. It is a Microchip (model: dsPIC33FJ256GP710). It is a fixed-point digital signal processor with 256kB of DMA-capable memory. A fixed-point processor was chosen because a floating-point processor is much more expensive, and requires substantially more power to operate. The fixed-point processor is much more suited towards our low cost/low power goal.

The microcontroller also has a substantial amount of RAM (100 K bytes), which is necessary due to the large amount of memory consumed by sampling and signal processing. The microcontroller's Direct Memory Access (DMA) system is also in use, which allows the sampled symbols to be placed into memory directly, instead of consuming valuable CPU cycles moving the samples from the Analog-to-Digital

Convert (ADC) to memory. It allows a symbol to be demodulated concurrently with sampling, so that no samples are lost due to a long demodulation calculation.

The microcontroller is packaged with an interface board, which contains peripherals that are directly accessible to the processor. These include a variety of digital I/O ports, a digital potentiometer-based Digital-to-Analog Converter (DAC), and a 10-bit/12-bit ADC. Having these peripherals integrated directly with the CPU saves substantial development time, power, and cost.

Thus, each underwater node we designed includes the following components: (1) Microcontroller with intelligent data processing and wireless communication capabilities, (2) sensors to collect analog underwater parameters such as pH values, (3) acoustic transmitter and receiver, and (4) batteries and others. Figure 3 shows the fabricated underwater node we made.

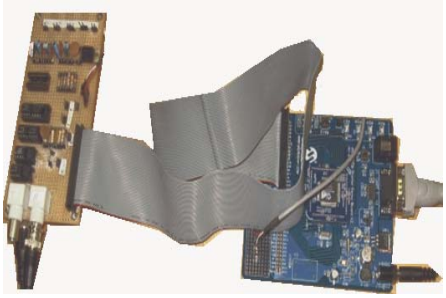

Figure 3 *Microcontroller (right) and Interface board (left)*

The modulation scheme chosen for this project was Frequency Shift Keying (FSK) modulation. FSK works by changing the frequency of the carrier depending upon which binary symbol is supposed to be represented. In this system only a 2-level FSK is used, i.e., there are only two carriers. In this scenario each carrier represents only a single bit. A more complex system might use several different carriers, to represent multiple bits at a single time. For instance four carriers could represent two bits, where the presence of any given carrier would represent a specific combination of two bits (‘00’, ‘01’, ‘10’, ‘11’).

For sensor node described in this paper the two carriers were located at 2.5 kHz representing a ‘0’ bit and 3.5 kHz representing a ‘1’ bit. The spacing between these two carriers was arbitrarily decided upon. Carriers in FSK must be spaced far enough apart so that the receiver can correctly discern between any two given carriers. Since this system was design to only use two carriers the carriers were placed 1 kHz apart on the assumption that this would be enough to distinguish between them at the receiver. This will be discussed at length in the demodulation section.

Modulation was designed to originally be able to generate the carriers at runtime within the system. However, due to truncation problems with fixed-point arithmetic, the two carriers were statically generated and stored with in the system. Each of the carriers is generated using the following formula:

$$c = \left\lfloor \frac{MaxVal}{2} \times \sin(2\pi f t) \right\rfloor + \frac{MaxVal}{2} \quad (1)$$

where t is a vector of times defined from 0 to the end of the period of the wave, $T=1/f$. The resolution of this vector is defined by the sampling frequency of D/A converter. *MaxVal* defines the maximum amplitude for the waveform. The microprocessor’s D/A has a resolution of 8-bits, so its maximum value is 255. For this system the D/A sampling frequency has been set to 8 kHz. The sampling frequency was selected because the maximum frequency to transmit is 3.5 kHz, and Nyquist’s sampling theorem states that to maintain the properties of the original waveform it must be sampled at greater than twice its maximum frequency, and 8 kHz was the closest sampling frequency to Nyquist that could be obtained on the digital signal processor board.

The modulation is accomplished by repeating the appropriate carrier wave stored in memory multiple times to achieve the desired bit rate in the system. The maximum bit-rate achieved was 16 bits/second. The system originally needed to maintain 8 bits/second to keep up with the receivers DMA, however additional gains were necessary to assure no loss of time samples. This will be discussed further in the later of the section about demodulation.

Additionally it was assumed that the receiver would be able to detect energy in the frequencies of interest, and immediately start sampling, however the sampling system takes time to start up. To help the receiver align each symbol to be demodulated, a chirp wave is transmitted prior to the frames preamble. This necessity will again be discussed at length in the later of the section about demodulation. An oscilloscope capture of transmitted symbols from the modulation circuit is shown in Figure 4.

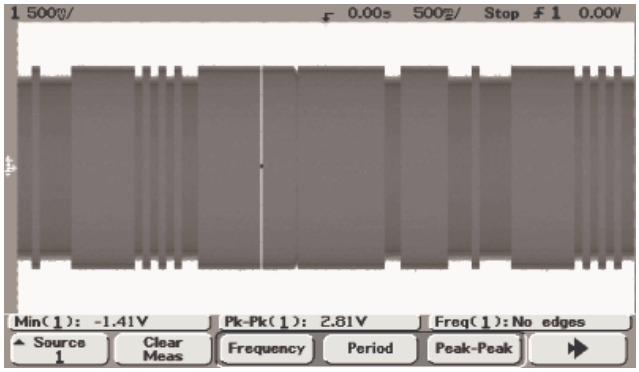

Figure 4 *Modulation of a typical frame*

Once the digital signal processor receives a signal from the level detector circuit the demodulation process begins. This starts by initiating a Direct Memory Access (DMA) routine capturing samples from the A/D to internal buffers, without the need for direct interaction of the digital signal processor. The DMA allows the A/D to take samples and place them in buffers without taking clock cycles away from the DSP. This allows the DSP to accomplish other computationally expensive tasks. The DSP copy the samples after a block is completed by the sampling routine. To

ensure that the DSP does not miss any samples, the DMA runs two buffers in what is known as ping-pong mode. In this mode one buffer is filled with samples, and then while that buffer is being internally copied, it fills a second buffer. The DSP allocates a total of 1024 bytes for the total DMA memory space. This space was simply split in half, and each buffer takes up 512 samples. Each carrier wave is also constructed of 512 samples, to match this buffer length. The A/D has a resolution of 12 bits, is also clocked at 8 kHz.

However, the DMA takes time to initiate, and since the level-detector does not take a deterministic amount of time to determine the presence of a signal, the exact starting location of a symbol must be computed within the buffer. A chirp wave is used to help compute exactly where in any given buffer the symbol begins (an example of a delayed chirp occurring in a buffer is shown in Figure 5). The chirp wave is half of the duration of a symbol and sweeps linearly from 2.5kHz to 3.5kKz. To ensure that the level-detector will activate prior to the arrival of the chirp, it is preceded by a 2.5kHz carrier wave of the same duration.

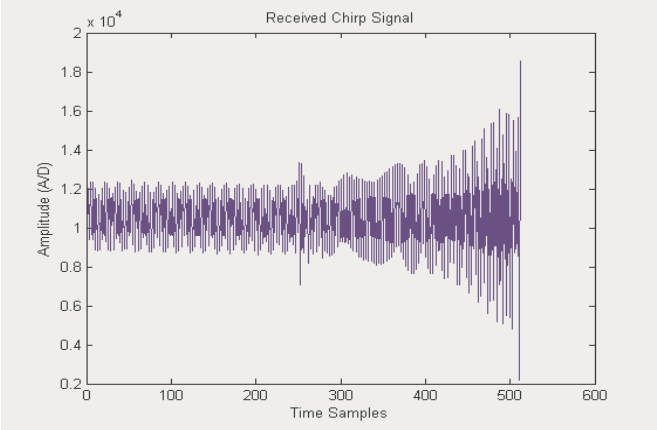

Figure 5 *Received Chirp Signals*

The starting location of a chirp within the first buffer is computed using a matched filter. The reason a chirp must be transmitted and single carrier cannot be used is because a carrier has virtually no bandwidth, and matched filters produce resolution in time (i.e., a precise location indicating the start of the chirp wave) only when given sufficient frequency bandwidth. The index into the buffer where the chirp wave begins is computed using a matched filter as follows:

$$t = Max\left(abs(IFFT(x_{ref})^* \times FFT(x))\right) \quad (2)$$

where x_{ref} is the reference signal, i.e., the expected signal, in this case the chirp signal, and x is the actual received signal. The output of the matched filter function is shown in Figure 6. As it shows the function has a maximal output where the reference signal begins within the given buffer of samples. The delay incurred in computing this index however, need only be computed once per transmission, and introduces more robust receiving capabilities into the system for further expandability.

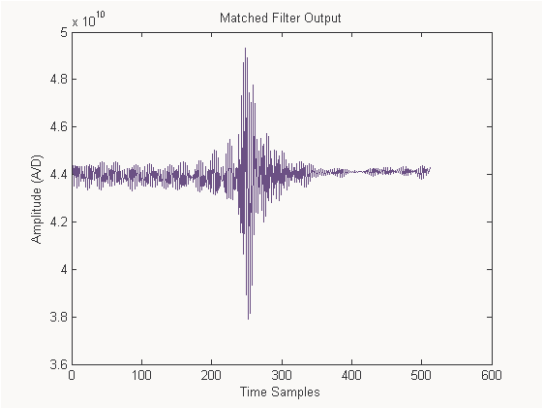

Figure 6 *Matched Filter Output*

In terms of system software architecture, during normal operation, the sensor node is in reception mode, listening for commands from the main node. The sensor node only transmits when it is told to by the master node. The following state machine (Figure 7) shows the operational behaviour of the sensor node:

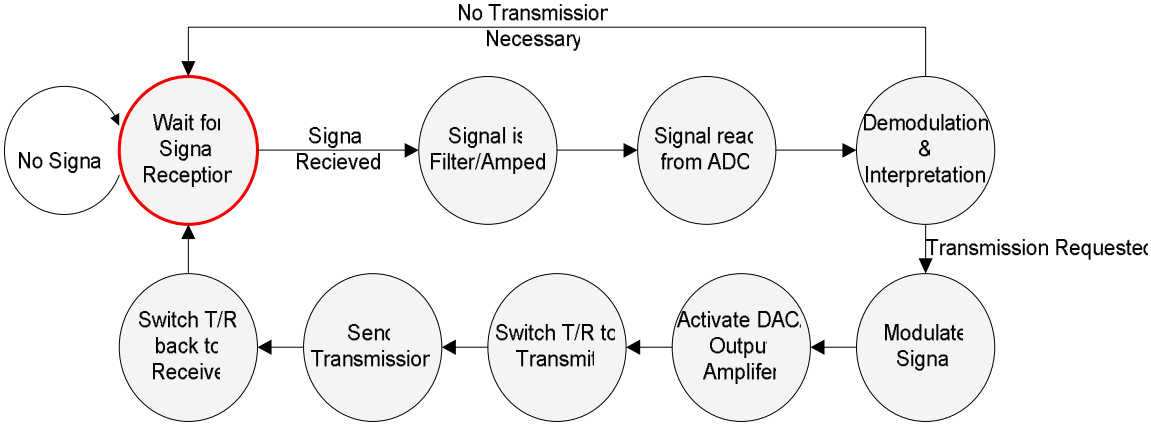

Figure 7 *Software/Hardware data flow diagram*

The majority of the system is in sleep mode until the level detection circuitry detects an input signal strong enough to be a possible command. The level detector then wakes up the DSP, ADC, and Input Amplifier to read and demodulate the signal. If this signal is not an actual frame (i.e., the preamble is not detected) the system goes back into sleep mode and awaits another signal. If the signal is a valid frame, the system switches into transmission mode to transmit an acknowledgement and any requested. The DAC and Output Amplifier are activated, and the T/R switch is set to transmit mode. After the new message is modulated, it is sent through the circuitry. The system will wait for an acknowledgement, and retransmit the data if one is not received. Once completed, the DSP sets the T/R circuitry back to receive mode, powers down non-essential peripherals, and waits for the next reception. In the case of the master node, the data is also forwarded to the computer GUI via the serial port.

A test was conducted using these procedures. The system master node is connected to a PC via a RS-232 connection at 2400 baud. The master node has no sensors of its own in this case. The slave nodes maintain the sensors for the system (in this case the pH and temperature sensors). The slave nodes are set poll its sensors for information every 30 seconds. Once the sensor data is retrieved a frame is constructed and the CRC of the frame computed. The system achieved its theoretical bit-rate of 15.625 bits/sec. The maximum number of retransmissions observed (within the Rubber-made test tank used) was 10. An average bit error rate (BER) of 0.091 was observed. As predicted the CRC did not avoid all errors, some frames were deemed error free when there was at least one bit error contained within the frame. These can be seen in Figure 8 where spikes are present on the GUI graph of pH temperature results.

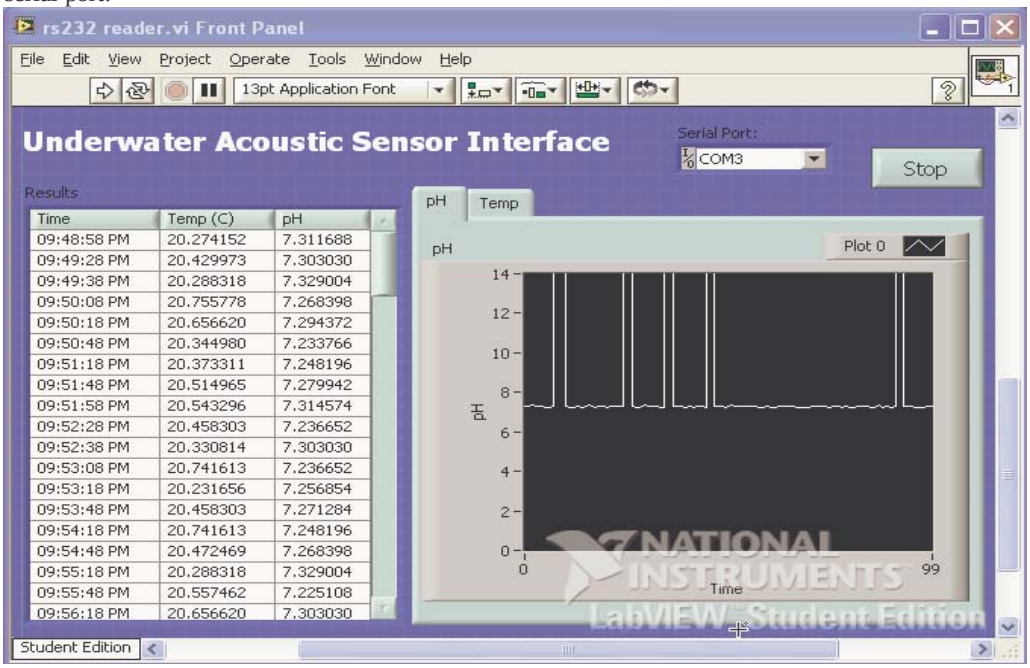

Figure 8 *Underwater Sensor Data Collection GUI*

3 UWSN MEDIUM ACCESS CONTROL DESIGN

3.1 Background and Protocol

There are only a few works on underwater sensor network acoustic-oriented MAC design (Xie et al. 2000; Molins et al. 2006; Rodoplu et al. 2005; Syed et al. 2006). Among them the Rodoplu's protocol (Rodoplu et al. 2005) is one of the good designs. The main idea behind Rodoplu's protocol is to minimize the energy wastage in acoustic channel access. The main two sources of this energy wastage are channel access collisions (CAC) and idle listening (IL).

In the CAC case, at least one of the frames is destroyed by the other frame's signal. As the basics of our future discussion, here we simply review three typical types of CAC cases as follows:

(1) *CAC-RT*: called as Receiving-Transmission (RT) collision. It happens when a node A begins transmitting a frame while it is still in the process of receiving frame from node B (see Figure 9). Implementing a simple carrier sense mechanism can avoid the collision completely.

Figure 9 *CAC-RT case*

(2) *CAC-TR*: i.e., Transmission-Receiving (TR) collision. It happens when node A is transmitting a frame while another frame form node B begins arriving at the A's receiver (see Figure 10). This sort of collisions cannot be avoided through channel sensing because the reception is happening after the transmission has begun. The main reason behind such type of collisions is the high acoustic propagation delay in the underwater environment (it could be 8 ms for a 10m of distance).

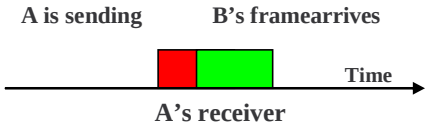

Figure 10 *CAC-TR case*

(3) *CAC-RR*: called receiving-receiving (RR) collision. As shown in Figure 11, it takes place where the frames from nodes B and C arrives at the A's receiver with overlapping reception time, thus resulting in losing the frames from both B and C. It may or may not be due to the hidden terminal problem, i.e., neither B nor C are aware of other's transmissions. A simple channel sensing can not solve the problem since they cannot hear from each other. A solution is to use request-to-send and clear-to-send (RTS/CTS) control frames before official data transmission.

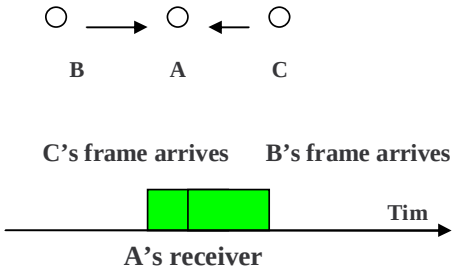

Figure 11 *CAC-RR case*

The second reason for energy wastage is, as mentioned earlier, the idle listening, which is defined by the time that the node's transceiver is actually fully functioning while there is no data to receive. Turning off the transceiver when there is no communication around can save the node's energy. This is also the motivation of using scheduling schemes among neighbouring nodes.

The concept of scheduling is to accurately predict the times to turn on the transceiver and receive a frame from a neighbour. Determining the schedules can be done in two ways: centralized and distributed. The centralized scheduling depends on a single point of command broadcast from a controller (it could be any node). On the other hand, the distributed idea could be faster and more robust by letting a node discover its own schedule based on its neighbours' situations.

Rodoplu's protocol allows the nodes to construct their schedules individually. Basically the node listens to the channel in the first frame in order to collect the receiving schedules from all the first-hop neighbours. Then it can select its own transmitting slot. After that, the node can go to sleep in the non-working slots of future frames to save its energy. The improvement in energy consumption is significant because the node can limit its idle listening times and turn on the receiver only when there is a scheduled time slot and when it has data to send.

Figures 12 and 13 give a scheduling example of a group of nodes with the frame length of 1 second. The un-numbered slots represent the frame transmission of a node, and the numbered slots reflect the scheduled receiving slots tagged with the sending source of the frame. The grey periods represent the sleeping times where there are no data to send / receive.

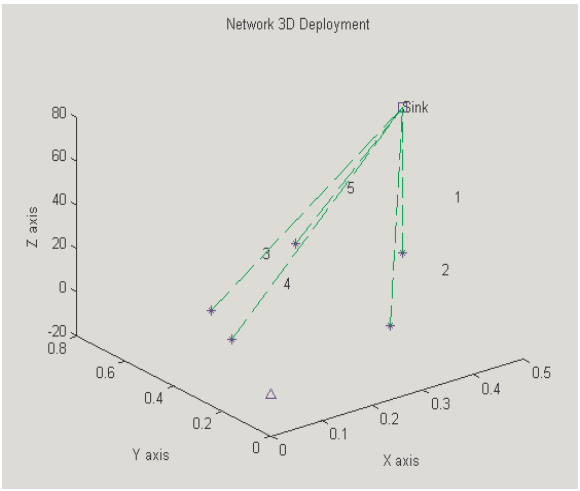

Figure 12 *Rodoplu's scheduling example– Topology*

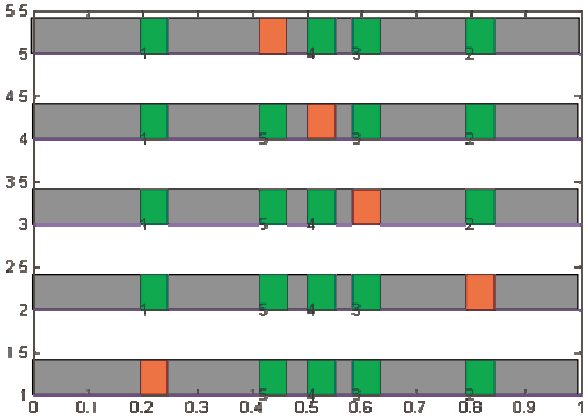

Figure 13 *Rodoplu's scheduling example-Scheduling*

The Rodoplu's protocol has also introduced an algorithm to enable new comers to join the already formed schedules even after the nodes sleep out of the scheduled periods. The

key point is to allow the nodes to listen for extra time after the receiving is done to see if there are changes in the neighbourhood's schedule. Dealing with mobility and propagation fluctuations due to the underwater environment was also addressed by introducing the early/late wake up times and missing neighbours' list respectively.

3.2 Enhancement MAC – Topology-aware approach

In Rodoplu's protocol, a node has to listen and schedule the receiving for all its first-hop neighbours even if the node does not consider a neighbour as a relaying node. In many routing protocols, each node usually selects a set of nodes from its neighbours as relaying nodes in order to deliver the sent frames to the destination. In other words, the nodes are concern with communicating with only the nodes that give the connectivity to the network or to the tree root in the following case. In a typical UWSN, a node collects underwater sensing data and sends it to its parent node (see Figure 14 shaded dots in water Depth *i+1* and Depth *i*). This depth-to-depth relay approach can finally deliver the data to the tree root – the surface station. If a node does not have parent node (see Figure l4 empty dots), it can search a multi-hop path in the same water depth and send the data to the node with parent node, which can help to deliver the data to a upstream node. We call the nodes without parent nodes as "orphans".

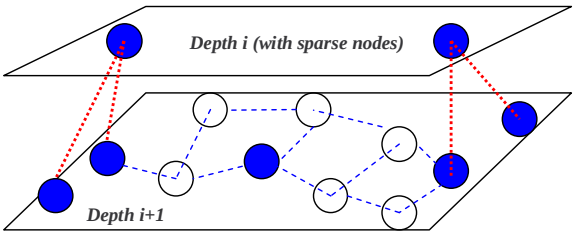

Figure 14 *Tree-based UWSN communication architecture*

In topology discovery protocol we developed in this research, any node can select only one node from its neighbours to be the relaying node. This relay node can be called a parent node (in the vertical direction) or a brother (in the horizontal direction) depending on the position of the selected node.

We define a node's "vital neighbour" as a set including its parent node, brother nodes (with parent nodes), and its children nodes in the tree (see below notation). Those vital neighbours form the providers of the vital links in the network since they can help relay its sensing data.

$$\left[\{Parent \parallel Brother\} \cup \{Children\}\right] \subseteq \{Vital\ Neighbors\}$$

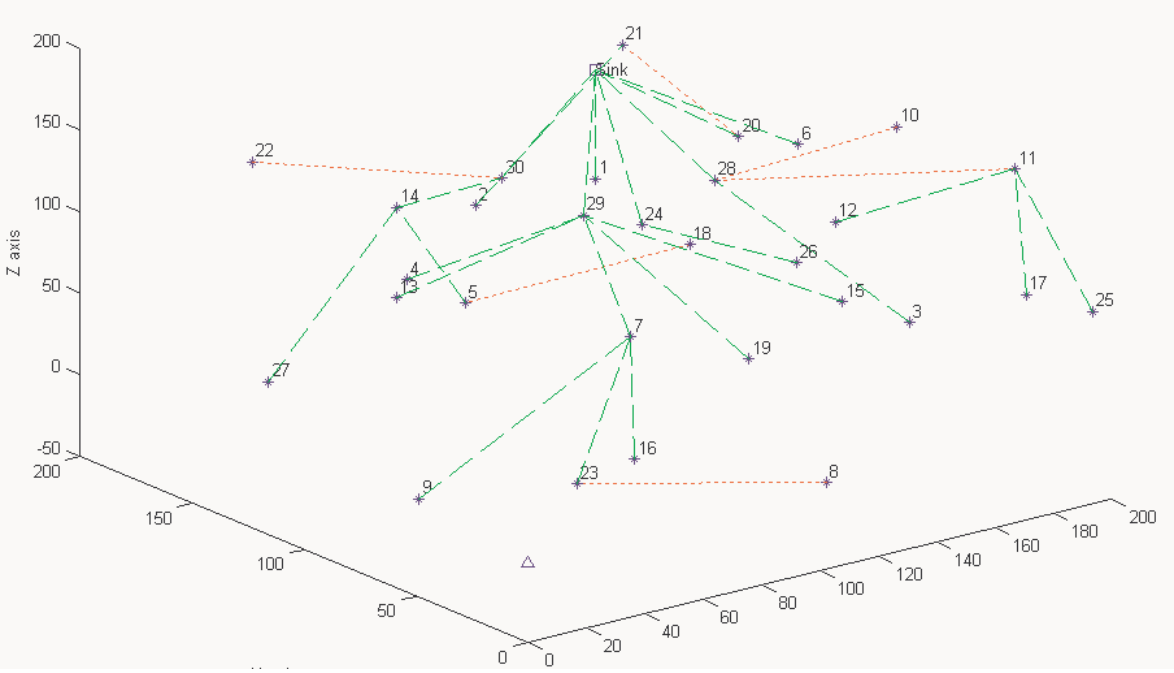

Figure 15 *Parent nodes and Brother nodes*

Figure 15 has some examples of vital neighbour nodes. In the first example, node 7 can use its parent (node 29) to relay data; in the second example, node 11 uses its brother (node 28) to relay data since node 11 is an orphan (i.e., without a parent node).

Figure 16 shows the receiving schedules of nodes 7 and 11's neighbouring nodes, respectively. The marked time slots (using rectangle around the node ID) represent the receiving times from vital neighbours. Suppressing non-vital neighbours' time slots by keeping the receivers of those nodes in the sleep mode, in other words, deleting the corresponding scheduled slots of non-vital neighbours, does not affect the connectivity of the network at all (from the viewpoint of data forwarding to the root node in the water

surface). A direct benefit of this suppressing is to decrease the energy consumption of the network by increasing the nodes' sleeping times.

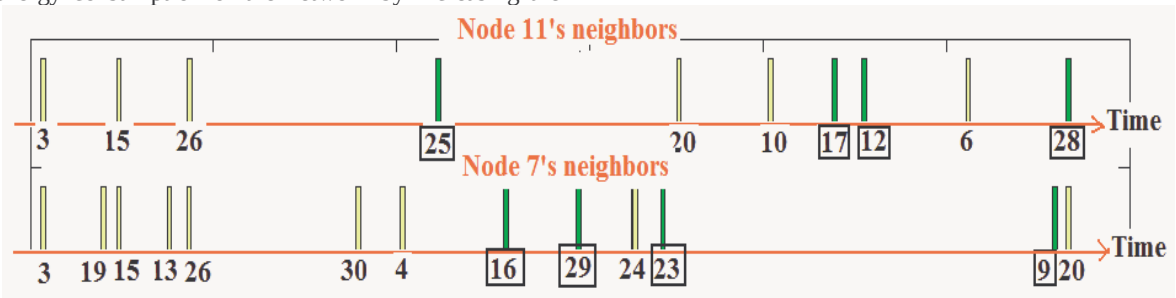

Figure 16 *Nodes 11 and 7's neighbours' receiving schedules*

We have used Matlab to conduct UWSN MAC simulations for the above ideas. Figures 17-18 shows the effect of suppressing the non-vital neighbours on the total energy consumption of the network. As expected, the life span of the network has been increased because of the decreasing of the energy wastage in idle and non-vital link listening. This can be seen from the slope changes of different curves.

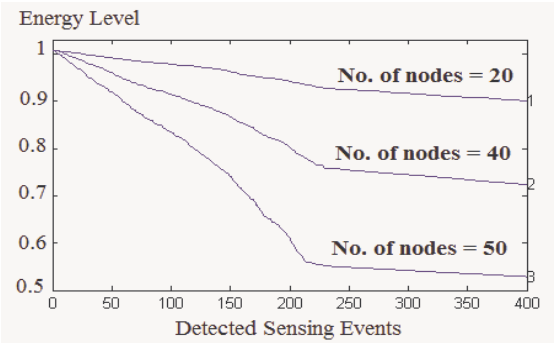

Figure 17 *Residual Energy Level (Rodoplu's MAC)*

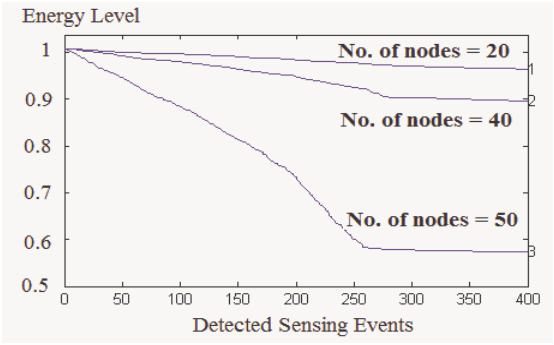

Figure 18 *Residual Energy Level (Topology-aware MAC)*

As we can see from Figures 19-20, the network connectivity (how much percentage of nodes are reachable) gets better due to slower speed of node energy exhaustion. In Figures 21-22, we can see that the communication overhead is much less due to the suppression of non-vital neighbours' transceivers.

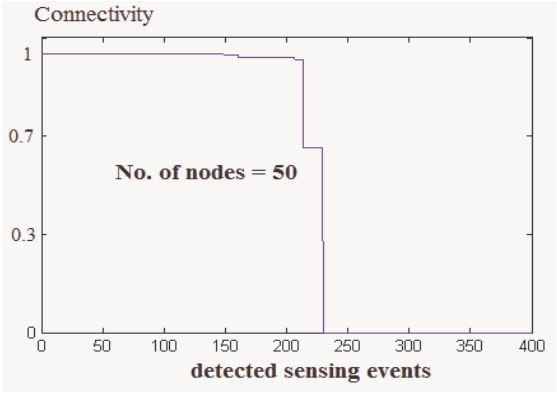

Figure 19 *Network Connectivity (Rodoplu's MAC)*

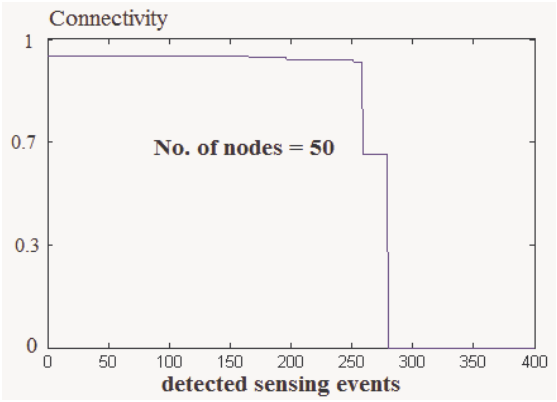

Figure 20 *Network Connectivity (Topology-aware MAC)*

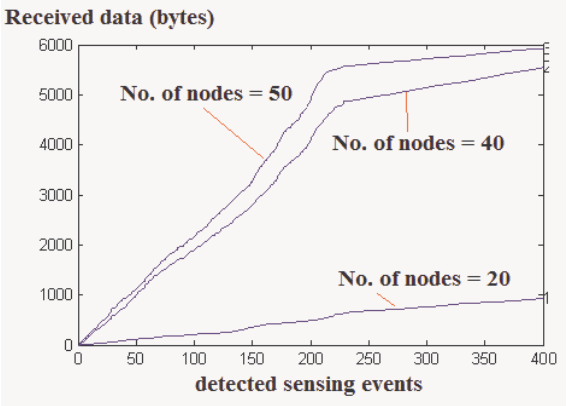

Figure 21 *Communication Overhead (Rodoplu's MAC)*

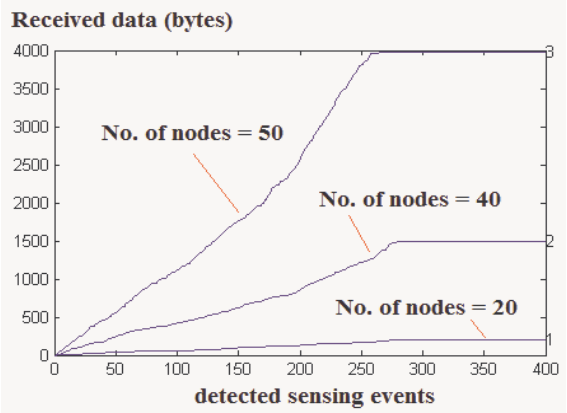

Figure 22 *Communication Overhead (Topology-aware MAC)*

3.3 MAC Enhancement through Queuing Delay Reduction

In Rodoplu's protocol the nodes select their transmissions slots randomly. Basically, each node has to listen for the channel at least for one frame in which it collects the receiving schedules for all first-hop neighbours. So the only coordination that a node can do is to make sure that its transmission is not colliding with any of its neighbour's transmissions. This means that each node has to make sure that its transmission is not causing Type 1 collisions (CAC-RT). This can be achieved by the carrier sense mechanism.

On the other hand, Type 2 collisions (CAC-TR) can still happen. However a complex processing and interacting between the two nodes can detect the occurrence of such a collision. This can be achieved by forcing the nodes to perform an extra check on the selected transmission time slot. This extra work can cause unnecessary amount of processing overheads.

Type 3 collisions (CAC-RR) have the highest probability to happen because of the hidden terminal problem. While in MAC layer each node only considers the first-hop neighbours, the knowledge about the two-hop neighbours is needed to avoid CAC-RR. Again the processing overhead could be high.

In addition to the abovementioned collision problem, the Rodoplu's protocol does not account for the queuing delay introduced by the randomness in selecting the transmission time slot at its next-hop neighbours. In more details, a node (say A) selects its transmission time slot randomly without considering the transmission schedule for a next-hop neighbour (say B), as shown in Figure 23. In this particular case, any transmission from A has to wait for at least a queuing delay occurring at node B, in addition to the acoustic propagation delays (it is significant compared to RF case).

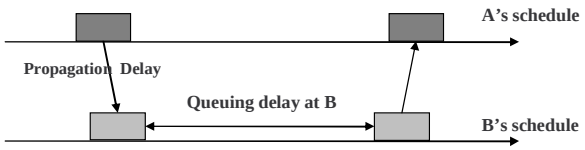

Figure 23 *Queuing delay introduced by the random time slot selection in next-hop neighbours*

The *minimum queuing delay* that could happen in one-hop distance is zero. This is the case that the receiving from Node A happens exactly at the scheduled transmission time slot at Node B. On the other hand, the *maximum queuing delay* can happen when node B is receiving A's frame at the very beginning of the frame. The extreme starting time for a reception from any node is bounded by the time of completely receiving the frame and the propagation delay. This is actually the case where A is scheduling its transmission at the beginning of the frame that is time zero. So the receiving delay of the frame at B will be the propagation delay plus the transmission delay (or 1 time slot time), i.e., *Propagation Delay + Slot Duration.*

To solve the queuing delay problems, we propose an enhanced scheme to Rodoplu's MAC protocol. To minimize the queuing delay at the next hop node, each node has to schedule its transmission in such a way to make the receiving schedule at the next-hop node happen just before the sending time slot of the next-hop node. Following the same example given previously, (i.e., node A is a child of node B; and node B is a child of node C), the basic concept is to design the schedules of those nodes as shown in Figure 24.

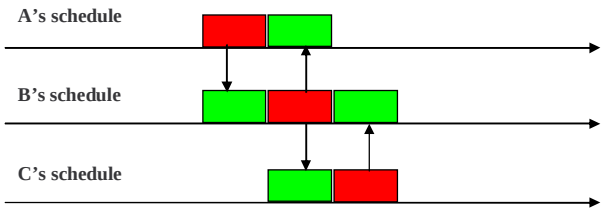

Figure 24 *Ideal Scheduling Case to Avoid the Queuing Delay*

To make our above idea more complete, we need to consider more complex cases such as the hidden terminal problem. As we know, the RTS/CTS handshaking can avoid most collisions caused by the hidden terminal problems. The solution lying behind is to ask the nodes to listen to the RTS or CTS frames, and to defer their transmission for the data transmission duration announced in the NAV field of CTS frames. However, in the underwater environment, implementing the above RTS/CTS handshaking is not encouraged because of the high acoustic propagation delay. This will increase the probability of causing a collision of the RTS frames and degrade the performance accordingly. So there is a need to exploit the two-hop information through scheduling procedure rather than depending on RTS/CTS exchanging.

Our ultimate goal is to design a protocol that would consider the network topology information to minimize the queuing delay at the next-hop node and to decrease the probability of collisions, specially the collisions caused by the hidden terminal problem. As mentioned earlier, solving the hidden terminal collision can not be achieved without the knowledge of the two-hop nodes' scheduling information. We can then ask the nodes to broadcast their schedule information at the beginning of their transmission slot.

Assuming that all the nodes have the same frame and time slot lengths, our defined MAC layer frame format is shown in Figure 25. This frame format has been implemented in our UWSN prototype (described in Section 2).

Time for 1st receiving: the scheduled transmission	
Time for 2nd receiving: the scheduled transmission	
Time for 3rd receiving: the scheduled transmission	
Time for 4th receiving: the scheduled transmission	
Scheduled transmission	**Next time to transmit**
Routing information	**Payload**

Figure 25 *Scheduling Frame Format*

The *receiving fields* can be more than 4 depending on the complexity of the sender's operations. *Scheduled transmission* tells the receiver the time that the sender started to transmit. The field is mainly for the local synchronization purpose. *Next time to transmit*, the same concept as in Rodoplu's protocol, is used at the receiver to expect the next receiving schedule in the next frame of the sender. The *routing information* contains the fields of general ad hoc network routing protocols (such as the AODV broadcast ID), and the *payload* contains the sensed data. Figure 26 is an example of constructing a schedule for a node that has the shown frame fields with a frame length of 2 seconds.

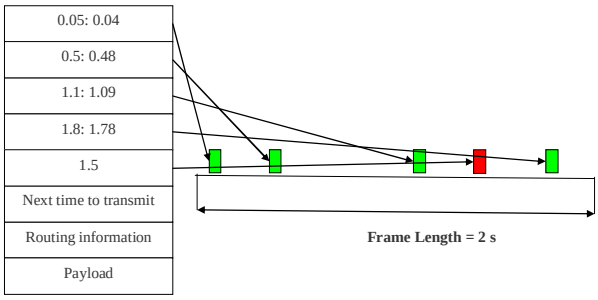

Figure 26 *The principle of constructing a schedule*

Any node that wants to join the network has to first listen to the medium for at least one complete frame in order to get the whole picture about the neighbours' schedules, and to sense and construct its local receiving schedules. The scheme is similar to (Xiao et al. 2006). Then it compares all the routing information collected from neighbours to select a single node to be its next hop node or the parent. Note that there are some good UWSN routing protocols available. Please see (Heidemann et al. 2006) for a good survey. We have also designed a 3-D tree-based routing scheme, which will be published in the separate paper. After that, it combines all the schedules received from neighbours to calculate its own transmission time slot so that the receiving at the selected parent node will be as close as the parent's transmission time slot.

As an example, consider our experimental UWSN topology in which node A is trying to connect to node B which is connected to node F (see Figure 27 right diagram).

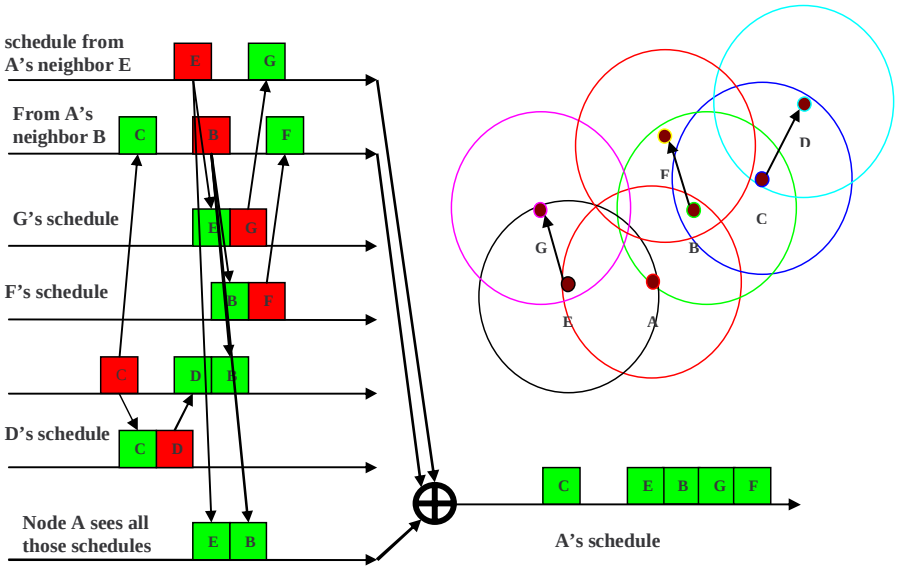

Figure 27 *Node A constructs its schedule from its neighbours B and E's schedules*

The decision to connect to node B should be made after listening for the medium for at least one frame time. In Figure 28, suppose that node C is connected to node D, and node E is connected to node G. At the end of the first frame, the data collected at node A will look like Figure 27 (left diagram). Please notice the existence of propagation delays. Now node A can add the schedules from node B and node C to its own schedule.

Then the schedules of node A and all the affected neighbouring nodes B, E will be like Figure 28.

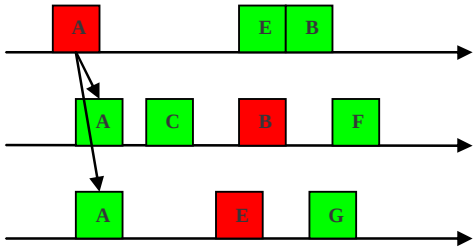

Figure 28 *Nodes A, B, E – Final schedules*

The only scenario that may cause a problem is when there is another node X that is close to node A, and also tries to connect to node B. Both the nodes A and X will select the same time slot to schedule a transmission since both hear the same schedules. To avoid such a situation, the nodes A and X will select a random time slot to send requests to node B that will announce the calculated transmission schedules. For the first one to get through, node B will reply to it in the next frame. Then only one node is allowed to join at a certain moment. This technique can be thought as forcing serialization at node B in order to avoid future collisions from node A and node X.

A node cannot send its schedule until it gets the permission from its parent node after requesting the participation. In this way, node X will be invisible to node A when it selects the transmission slot. When node A gets the permission from node B, it starts sending the schedule as soon as possible, so that node X would rely on node A to calculate the transmission slot later on.

From our above proposed ideas, each node's receiving schedule will be as close as possible to its next-hop nodes' transmission schedules. In many cases it would not be just before the transmission slot, because of the risk of colliding with other transmissions from other nodes. However, compared to Rodoplu's protocol, our scheme decreases the queuing delay dramatically and allows faster source-to-sink event reporting. Another advantage is minimizing the number of collisions in the network and hence increasing the connectivity of the network.

Most of our MAC experimental results showed zero collisions, shown in Table 1, which is due to the sharing of the schedules of two-hop neighbourhood.

Table 1 MAC Experimental Results

No. of Nodes	Rodoplu's MAC scheme: No. of collisions per second	Rodoplu's MAC scheme: queuing delay (ms)	Our MAC scheme: No. of Collisions	Ours MAC scheme: Queuing delay (ms)
3	6	15.0290	0	1.7110
5	5	29.6438	0	2.0320
7	21	16.8711	0	4.1847

4 CONCLUSIONS

This research has laid ample foundation for a cost-effective scalable UWSN system. Our developed system is a prototype that can provide a low-cost test-bed to be used for short to medium range UWSN communications. It has signal conditioning circuit to collect pH, temperature and other underwater parameters. It also has all stack layers to perform tree-based communications. For those UWSN protocols, we have concentrated on the MAC design. Our proposed MAC enhancements schemes show benefits in terms of saving more energy and reducing number of frame transmission collisions.

ACKNOWLEDGEMENT

The authors acknowledge the wireless communications group in Lockheed Martin Inc. (the branch in Rochester, NY) for their support of our underwater sensor network research. We also thank U.S. National Science Foundation (NSF) (Award No. 0511098) for supporting the development of the Pervasive Computing Laboratory at Rochester Institute of Technology. We also thank Cisco Inc. University Research Program (URP) for supporting the development of wireless security projects.

REFERENCES

Heidemann, J., Ye, W., Wills, J., Syed, A., and Li, Y. (2006), "Research Challenges and Applications for Underwater Sensor Networking," IEEE Wireless Communications and Networking Conference, April 2006.

Sozer, E., Stojanovic, M., and Proakis, J. (2000), "Underwater Acoustic Networks," *IEEE Oceanic Engineering*, 2000 Vol 25, 1.

LinkQuest, http://www.link-quest.com

DSPComm, www.dspcomm.com

Yang, X., Ong, K., Dreschel, W., Zeng, K., Mungle, C., and Grimes, C. (2002), "Design of a Wireless Sensor Network for Long-term, In-Situ Monitoring of an Aqueous Environment," *Sensors* 2002, 2, 455-472.

Xie, G. G. and Gibson, J. (2000), "A networking protocol for underwater acoustic networks," Technical Report TR-CS-00-02, Department of Computer Science, Naval Postgraduate School, December 2000.

Molins, M., and Stojanovic, M. (2006), "Slotted FAMA: A MAC protocol for underwater acoustic networks," In *Proceedings of the IEEE OCEANS'06 Asia Conference*, Singapore, May 2006.

Rodoplu, V., and Park, M. (2005), "An energy-efficient MAC protocol for underwater wireless acoustic networks," In Proceedings of the IEEE OCEANS'05 Conference, September 2005.

Syed, A. A., Ye, W., and Heidemann, J. (2006), "Medium Access for Underwater Acoustic Sensor Networks," extended abstract for work-in-progress poster. Technical Report USC/Information Sciences Institute, October, 2006.

International Transducer Corporation, www.itc-transducer.com

Xiao, Y., Li, H., Wu, K., Leung, K. K., and Ni, Q. (2006), "On Optimizing Backoff Counter Reservation and Classifying Stations for the IEEE 802.11 Distributed Wireless LANs," IEEE Transactions on Parallel and Distributed Systems, Vol. 17, No. 7, July 2006, pp. 713-722.

J. Gibson, G.G. Xie, Y. Xiao (2007a), "Performance Limits of Fair-Access in Sensor Networks with Linear and Selected Grid Topologies," Submitted to GLOBECOM 2007.

J. H. Gibson, G. G. Xie, Y. Xiao, and H. Chen (2007b), "Analyzing the Performance of Multi-hop Underwater Acoustic Sensor Networks," in Proc. IEEE/OES Oceans 07 Aberdeen Conference.